WebSphere® Process Server

IBM

Version 6.0

Administering

Note

Before using this information, be sure to read the general information in "Notices" on page 219.

September 29 2005

This edition applies to version 6, release 0, of WebSphere Process Server (product number 5724-L01) and to all subsequent releases and modifications until otherwise indicated in new editions.

To send us your comments about this document, email doc-comments@us.ibm.com. We look forward to hearing from you.

When you send information to IBM, you grant IBM a nonexclusive right to use or distribute the information in any way it believes appropriate without incurring any obligation to you.

Contents

Administering WebSphere Process Server . . . 1
Overview of administering WebSphere Process Server . . . 1
  The administrative console for WebSphere Process Server . . . 3
Setting up the WebSphere Process Server administration environment . . . 10
  Setting up the administrative architecture . . . 10
  Configuring the environment . . . 11
  Working with server configuration files . . . 12
  Administering process servers . . . 12
  Configuring WebSphere Process Server resources . . . 15
  Installing and managing applications . . . 16
  Administering applications and application services . . . 16
Configuring WebSphere Process Server settings . . . 16
  Configuring the Application Scheduler for a standalone server . . . 17
  Managing Events Service . . . 17
  Enabling the extended messaging service . . . 18
  Configuring the staff plug-in provider . . . 18
  Working with WebSphere Business Integration Adapters . . . 21
  Administering the relationship service . . . 25
Administering Business Process Choreographer . . . 26
  Administering the compensation service for a server . . . 27
  Refreshing the failed message counts . . . 28
  Querying and replaying failed messages, using the administrative console . . . 28
  Querying and replaying failed messages, using administrative commands . . . 31
  Refreshing staff queries, using the administrative console . . . 33
  Refreshing staff queries, using administrative commands . . . 33
  Enabling Common Base Events and the audit trail . . . 35
  Deleting audit log entries, using administrative commands . . . 37
  Removing unused staff queries, using administrative commands . . . 38
  Deleting process templates and task templates that are no longer valid . . . 39
Administering applications and application services . . . 42
  Administering business processes and human tasks . . . 42
  Business rules . . . 54
  Administering enterprise applications . . . 77
  Application Scheduler . . . 78
Administering WebSphere Process Server resources . . . 83
  Administering adapters . . . 83
  Selector component administration overview . . . 125
  Considerations for modules containing business rules and selectors . . . 128
  Overview of targets . . . 130
  Administering extended messaging resources: Overview . . . 133
Administering relationships . . . 140
  View relationship types . . . 141
  View relationship details . . . 141
  Query relationship instances . . . 142
  View relationship instance details . . . 143
  Edit relationship instance details . . . 143
  Create new relationship instance . . . 144
  Delete relationship instance. . . . 144
  Roll back relationship instance data . . . 145
  View role types associated with a relationship . . . 145
  Query relationship instances based on roles . . . 145
  View role details . . . 146
  View role instance details . . . 147
  Create role instance . . . 147
  Delete role instance . . . 148
  Edit role instance properties . . . 148

Tutorial: Relationship manager administration . . . 149
Managing WebSphere Process Server failed events . . . 150
Role-based access for failed event manager . . . 151
Finding failed events . . . 151
Working with data in failed events . . . 156
Resubmitting failed events . . . 159
Deleting failed events . . . 160
Using the Common Event Infrastructure . . . 160
Administering the Common Event Infrastructure . . . 160
Viewing events with the Common Base Event browser . . . 214
Specifying the events to view . . . 215
Working with events returned from the event browser . . . 216
Troubleshooting the Common Base Event browser . . . 216

Notices . . . 219
Programming interface information . . . 221
Trademarks and service marks . . . 221

Administering WebSphere Process Server

WebSphere Process Server documentation PDFs

Overview of administering WebSphere Process Server

Administering WebSphere® Process Server involves preparing, monitoring, and modifying the environment into which applications and resources are deployed, as well as working with the applications and resources themselves.

The administrative interfaces

WebSphere Process Server offers several interfaces for administering the runtime environment:

The administrative console
The administrative console is a browser-based interface that enables users to monitor, update, stop, and start a wide variety of applications, services, and resources. The administrative console can also be used to work with relationships and to locate and resolve failed WebSphere Process Server events.

The administrative console extends to provide administration capabilities for WebSphere Application Server, as well as other customer-defined products.

For more information, see "The administrative console for WebSphere Process Server" on page 3.

Business Process Choreographer Explorer
Business Process Choreographer Explorer is a stand-alone Web application that provides a basic set of administration functions for managing business process and human tasks. You can view information about process templates, process instances, task instances, and their associated objects. You can also act on these objects; for example, you can start new process instances, repair and restart failed activities, manage work items, and delete completed process instances and task instances.

Scripting (wsadmin)
The WebSphere administrative (wsadmin) scripting program is a non-graphical command interpreter environment that enables you to run administrative options in a scripting language and to submit scripting language programs for execution. It supports the same tasks as the administrative console. The wsadmin tool is intended for production environments and unattended operations.

See **Reference > Scripting interface** in this information center for details about the scripting tools.

Command-line tools
Command-line tools are simple programs that you run from an operating system command-line prompt to perform specific tasks. Using these tools, you can start and stop application servers, check server status, add or remove nodes, and other tasks.

The WebSphere Process Server command-line tools include the serviceDeploy command, which processes .jar, .ear, .war and .rar files exported from a WebSphere Integration Developer environment and prepares them for installation to the production server.

See **Reference > API documentation** in this information center for details about the command-line tools.

Administrative programs

A set of Java™ classes and methods under the Java Management Extensions (JMX) specification provide support for administering Service Component Architecture (SCA) and business objects. Each programming interface includes a description of its purpose, an example that demonstrates how to use the interface or class, and references to the individual method descriptions.

See **Reference > Programming interfaces** in this information center for details about the programming interfaces.

Configuration information

Most configuration data for WebSphere Process Server is stored in XML files, which are kept in directories in the configuration repository tree (the master repository). The directory in which a configuration file exists determines its scope, or how broadly or narrowly that data applies.

- Files in an individual server directory apply to only that server.
- Files in a node-level directory apply to every server on that node.
- Files in a cell directory apply to every server on every node within the entire cell.

Table 1. WebSphere Process Server configuration files

Configuration file	Description
server-wbi.xml	Identifies a process server and its components, including Adaptive Entity Service, Extended Messaging Service, and WebSphere Business Integration Adapter Service configuration.
resources-wbi.xml	Defines operating environment resources for WebSphere Process Server and is present at the cell, node, and server scopes. This includes Extended Messaging Providers and WebSphere Business Integration Adapters.
cell-wbi.xml	Identifies a cell. This file is used to store the Relationship Service configuration, and is only present at the cell scope.
server-bpc.xml	Identifies a Business Process Choreographer container and its components, which include the Business Flow Manager, Human Task Manager, Staff Service, and Service Reference Service.
resources-bpc.xml	Defines operating environment resources for a Business Process Choreographer container, including configuration information for Staff Plugin Providers. This file is present at the cell, node, and server scopes.

Table 1. WebSphere Process Server configuration files (continued)

Configuration file	**Description**
deployment-bpc.xml	Configures application deployment settings for a business process container.
server-core.xml	Identifies configuration information for core WebSphere Process Server configurations, including the Artifact Loader Service, Events Service, and Business Context Data Service.

WebSphere Process Server configuration files can be edited through the administrative console, wsadmin, and scripting. No manual editing is required.

See the WebSphere Application Server Information Center for complete information about server configuration files.

The administrative console for WebSphere Process Server

The administrative console is a browser-based interface used to administer WebSphere Process Server applications, services, and other resources. It can be used to administer at a cell, node, or server scope, and is available from both stand-alone process servers and deployment managers that manage all servers in a cell in a networked environment.

Note: The WebSphere Process Server administrative console is part of the larger WebSphere Application Server administrative console. As a result, many administrative tasks (for example, setting security, viewing logs, and installing applications) are the same for both WebSphere Process Server and WebSphere Application Server. Those tasks are documented in the WebSphere Application Server Information Center.

Understanding the WebSphere Process Server tasks associated with the console

Common WebSphere Process Server tasks that are performed in the console include:
- Setting up the administrative architecture and environment
- Configuring process servers and their settings
- Deploying new applications to a server
- Managing existing applications and configurations
- Managing resource providers for applications
- Managing server resources, such as relationships, business processes, tasks, adapters, and selectors
- Administering the Business Process Choreographer
- Managing failed events on the process server
- Configuring product security
- Collecting data for troubleshooting purposes

Understanding the administrative console interface

The administrative console has three distinct areas:

Task bar

The task bar is located at the very top of the console. It provides options for logging out of the console, accessing product information, and accessing support.

Navigation tree

The navigation tree is on the left side of the console. It provides links to console pages that you use to create and administer servers, applications, and other resources.

Click the plus sign (+) beside an item in the navigation tree to expand it, or click the minus sign (-) to collapse the item. You can also click the item itself to toggle between its expanded and collapsed state.

Workspace

The workspace is located on the right side of the console. It displays pages that you use to create and administer servers, applications, and other resources. You access these pages by clicking the links in the navigation tree, or by clicking links within the workspace pages themselves.

See "Getting started with the administrative console" on page 5 for a discussion of the types of pages that are displayed in the workspace.

On the far right side of the workspace is the help portal. It provides brief information about each field on the current page, as well as a link to more detailed information in the help browser.

Locating WebSphere Process Server-specific areas of the administrative console

WebSphere Process Server resources are grouped into several areas of the administrative console. Use the navigation tree to locate these resources, as follows:

- **Integration Applications**—Provides access to the following:
  - Failed event manager
  - Relationship manager
  - Common Base Event Browser
- **Resources** —Provides access to the following:
  - WebSphere Business Integration Adapters
  - Common Event Infrastructure Provider
  - Staff plug-in provider
  - Extended Messaging Provider
- **Servers > Application servers >** *server_name*— Provides access to the following:
  - Container settings for business processes and human tasks
  - Application Scheduler
  - Business rules
  - Events service
  - Extended Messaging Service
  - Selectors
  - Staff service
  - WebSphere Business Integration Adapter Service
  - Web service reference service

Accessing online help from the administrative console

The administrative console provides online help for each page and field. Access the help in one of the following ways:

- Click **Help** from the console task bar to view online help in a new Web browser.

  From the help browser, you can do the following:

  - Browse for the topic you want to view in the Index tab. Click the link for that topic to open it in the right panel of the browser.
  - Search for a topic by specifying one or more key words in the Search tab. All matching topics are displayed in the navigation tree; click a topic link to view it.
- Place the cursor over a field to view hover help about that field.
- Place the cursor over a field and wait for the question mark (**?**) icon to appear. When the icon appears, click the field name to display brief help about it in the help portal (the right-most panel in the workspace).

  If you want to view extended information about the field, or about the entire page and its associated tasks, click the **More information about this page** link at the bottom of the help portal.

Related concepts

"Getting started with the administrative console"

"Getting started with the administrative console"

Getting started with the administrative console

The following list of tasks can help you get started using the administrative console to manage and administer WebSphere Process Server resources.

- **Start the server for the administrative console.**

  The startServer command reads the configuration file for the specified application server and starts the server. See the WebSphere Application Server Information Center for details on using startServer.
- **Start the administrative console.**

  See "Starting and stopping the administrative console" on page 6 for details.
- **Specify console preferences.**

  Preferences control how data is displayed in the administrative console, as well as how the workspace behaves. See "Starting and stopping the administrative console" on page 6.
- **Set the console scope.**

  The scope specifies the level at which a resource is visible on the administrative console. A resource can be visible in a console collection table at the cell, node, cluster, or server scope. See the WebSphere Application Server Information Center for details on setting the scope.
- **Create filters to view information.**

  Filters specify which data is shown in a column on a collection page. See "Starting and stopping the administrative console" on page 6.
- **Optional: Set the session timeout for the console.**

  By default, a console session times out after 30 minutes of inactivity. You can change this value by editing the deployment.xml configuration file, as described in the WebSphere Application Server Information Center.
- **Save your work to the master repository.**

Until you save your changes to the master repository, the console uses a local workspace to track the changes. To save your changes, click **System Administration > Save Changes to Master Repository** to display the Save page, and then click **Save**.

Starting and stopping the administrative console

To access the administrative console, you must start it and then log in. After you finish working in the console, save your work and log out.

Perform the following steps to start and stop the console:

1. Start the administrative console:
   a. Distributed platforms: Verify that the administrative console runs on the server1 application server for the WebSphere base product.
   b. Enable cookies in the Web browser that you plan to use to access the administrative console.
   c. Distributed platforms: In your cookie-enabled Web browser, type the following: `http://`*your_fully_qualified_server_name*`:9060/ibm/console`

      where *your_fully_qualified_server_name* specifies the fully qualified host name for the machine that contains the administrative server. When the administrative console is on the local machine, *your_fully_qualified_server_name* can be `localhost` unless security is enabled.

      On Windows platforms, use the actual host name if `localhost` is not recognized.

      If security is enabled, your request is redirected to https://*your_fully_qualified_server_name*:9043/ibm/console, where *your_fully_qualified_server_name* is the fully qualified host name for the machine that contains the administrative server.

      Note: The console uses the ports 9060 and 9043 by default. The port numbers for your actual installation can vary.

   The administrative console loads in the browser, displaying a login page.
2. Log into the console:
   a. In the **User ID** field, enter your user name or user ID.

      Note: If you enter an ID that is already in use (and in session) you are prompted to do one of the following:
      - Force the existing user out of the session. The configuration file for the existing user ID is saved in the temporary area.
      - Wait for the existing user ID to log out or time out of the session.
      - Specify a different user ID.

      The user ID lasts only for the duration of the session for which it was used to log in. Any changes made to server configurations are saved to the user ID. Server configurations are also saved to the user ID if a session times out.
   b. If the console uses global security, you must also enter a password in the **Password** field.
   c. Click **OK**.

      The administrative console now displays the Welcome page.
3. Stop the console:
   a. Click **System administration > Save changes to master repository > Save** to save all work you have done during this session.

b. Click **OK**.

Your changes are saved to the master repository and the administrative console closes.

Setting administrative console preferences

The display of data on a collection page (a page that lists collections of data or resources in a table) can be customized through administrative console preferences. Preferences are set on a user level, and typically must be set separately for each area of the administrative console.

You can set the following display preferences for collection pages:

- **Maximum rows—**Specifies the maximum number of rows that are displayed when the collection is large. If there are more rows than the specified maximum, they are displayed on subsequent pages. The default value is 20.
- **Retain filter criteria—**Specifies whether the last search criteria entered in the filter function is retained. If this is enabled, the console collection pages initially use the retained filter criteria to display the data in the table following the preferences. See "Starting and stopping the administrative console" on page 6 for more information.
- **Max result set size—**Specifies the maximum number of resources that a search can return. The default value is 500.
- **Max column width—**Specifies the maximum number of characters viewable in a collection column. The default value is 18.

Perform the following steps to set display preferences for a collection page:

1. From any collection page, click **Preferences**.

   The page expands to display the preference fields.
2. Modify the values for the **Maximum rows**, **Retain filter criteria**, **Max result set size**, and **Maximum column width** fields as desired.
3. Click **Apply**.

   The collection table is refreshed to display according to the values you specified.

You can also set global administrative console preferences, such as whether the workspace is automatically refreshed and which scope to use by default. To access the Preferences page in the administrative console, click **System administration > Console settings > Preferences.** See the WebSphere Application Server information center for documentation on setting these preferences.

Setting administrative console filters

Each table on a collection page in the administrative console displays a list of WebSphere Process Server data or resources. You can use a filter to specify exactly which resources or data to display in a particular column of the table. Filters can be set on a single column only.

1. From the buttons at the top of the table, click **Filter the view**.

   The filter dialog box opens above the top row of the table.
2. Use the **Filter** drop-down menu to select the column you want to include in the filter.
3. In the **Search term(s)** field, specify the filter criteria.

   The criteria is a string that must be found in the name of a table entry in order for it to be displayed. The string can contain the percent sign (%), asterisk (*), or question mark (?) symbols as wildcard characters. For example, on the

Resource Adapters page, you can enter `*JMS*` as the filter criteria for the Name column to find any resource adapter whose name contains the string `JMS`.

Prefix each of the following characters that appear as part of the string with a backslash (\) so that the regular expression engine performing the search correctly matches the search criteria: `( ) ^ * % { } \ + & .`

For example, if you want to search for all Java DataBase (JDBC) providers containing `(XA)` in the provider name, specify the following string in the Search term(s) field:

```
*\(XA\)*
```

4. Click **Go**.

   The table refreshes, and only those items in the selected column that meet the filter criteria are displayed.

Administrative console pages

Administrative console pages are formatted in one of three ways: Collection, detail, and wizard pages. Understanding the layout and behavior of each type of page can help you use them more effectively.

Collection pages

A collection page manages a collection of existing administrative objects (for example, relationships, failed events, or resource adapters). It contains one or more of the following elements:

Scope and Preferences
: The scope and preferences help determine which administrative objects are displayed in the table, and how they should appear.

Table of existing objects
: The table displays existing administrative objects of the type specified by the collection page. The table columns summarize the values of the key settings for these objects. If no objects exist yet, the table is empty. Use the available buttons to create a new object.

Buttons for performing actions
: The typical buttons are described in "Getting started with the administrative console" on page 5. In most cases, you need to select one or more objects in the collection table, then click a button. The action is applied to all selected objects.

Sorting toggle buttons
: After each column heading in the table are icons to sort the entries in ascending (^) or descending (v) order. By default, items such as object names are sorted in descending order (alphabetically).

Detail pages

A detail page is used to view details about an object and to configure specific objects (such as an application server or a listener port extension). It typically contains one or more of the following elements:

Configuration tabbed page
: This tabbed page is used to modify the configuration of an administrative object. Each configuration page has a set of general properties specific to the object. Additional properties can be displayed on the page, depending on the type of administrative object you are configuring.

Runtime tabbed page
: This tabbed page displays the configuration that is currently in use for the administrative object. It is most often read-only. Note that some detail pages do not have runtime tabs.

Local topology tabbed page
: This tabbed page displays the topology that is currently in use for the administrative object. View the topology by expanding and collapsing the different levels of the topology. Note that some detail pages do not have local topology tabs.

Buttons for performing actions
: Buttons to perform specific actions display only on configuration tabbed pages and runtime tabbed pages. The typical buttons are described in "Getting started with the administrative console" on page 5.

Wizard pages

Wizard pages help you complete a configuration process comprised of several steps. Be aware that wizards can show or hide certain steps, depending on the characteristics of the specific object you are configuring.

Administrative console buttons

The administrative console interface contains a number of buttons, depending on which page you are currently viewing. This topic describes the available console buttons.

The following graphical buttons are located at the top of a table that displays WebSphere Process Server resources:

Button	Resulting action
Check all	Selects each resource (for example, a failed event or a relationship instance) that is listed in the table, in preparation for performing an action against those resources.
Uncheck all	Clears all selected resources so that no action is performed against them.
Show the filter view	Opens a dialog box to set a filter. Filters are used to specify a subset of resources to view in the table. See "Starting and stopping the administrative console" on page 6.
Hide the filter view	Hides the dialog box used to set a filter.
Clear filter value	Clears all changes made to the filter and restores the most recently saved values.

The following buttons appear at the bottom of a WebSphere Process Server administrative console page. Not all buttons appear on all pages.

Add Adds the selected or typed item to a list, or produces a dialog box for adding an item to a list.

Apply Saves your changes to a page without exiting the page.

Back Displays the previous page or item in a sequence. The administrative console does not support using the Back and Forward options in the web browser, which can cause intermittent problems. Use the **Back** or **Cancel** buttons in the console instead.

Cancel
Exits the current page or dialog box, discarding all unsaved changes. The administrative console does not support using the Back and Forward options in the web browser, which can cause intermittent problems. Use the **Back** or **Cancel** buttons in the console instead.

Clear Clears your changes and restores the most recently saved values.

Clear selections
Clears any selected cells in the tables on this tabbed page.

Close Exits the dialog.

Delete Removes the selected instance.

OK Saves your changes and exits the page.

Reset Clears your changes on the tab or page and restores the most recently saved values.

Save Saves the changes in your local configuration to the master configuration.

For a complete list of buttons used in the administrative console (for administering both WebSphere Application Server and WebSphere Process Server resources), refer to the WebSphere Application Server Information Center.

Setting up the WebSphere Process Server administration environment

Specific tasks are needed to ensure that the WebSphere Process Server administration environment is set up correctly.

After properly installing the WebSphere Process Server, you must set up your administrative and server environment.

Setting up the WebSphere Process Server administration environment includes the following tasks:
1. Configuring the product after installation
2. Configuring ports
3. Setting up the administrative architecture
4. Configuring the environment
5. Working with server configuration files
6. Administering process servers
7. Configuring WebSphere Process Server resources
8. Installing and managing applications
9. Administering applications and application services

Setting up the administrative architecture

After you install and set up WebSphere Process Server, you mainly need to monitor and control incorporated nodes and the resources on those nodes by using the administrative console.

You must complete the following tasks that are documented in the Setting up administrative architecture topic in the WebSphere Application Server Network Deployment, Version 6.0.x information center including the additional steps to configure WebSphere Process Server that are noted below.

1. Use the settings page for an administrative service to configure administrative services.
2. Configure cells.

   Prerequisites:

   a. An instance of the database product must be installed.
   b. A WebSphere Process Server database must be created in that instance.
   c. Tables for that database must be set up. This can be done during installation with the Profile Wizard. Or, later, changes can be made for database configuration using DBUtility.bat and DBUtility.sh batch files that will automatically set up the tables for the supported database types. DBUtility.bat and DBUtility.sh are available in *install_root*\bin\ .
3. Configure deployment managers.

   In additional to the steps in the Configure deployment managers topic in the WebSphere Network Deployment information center, WebSphere Process Server requires a variable of where the database is installed. On the administrative console, select **Environment > WebSphere Variables > (the scope of node) >** *DatabaseName*_**JDBC_DRIVER_PATH**. The following variables can be used:

   - For DB2: DB2UNIVERSAL_JDBC_DRIVER_PATH
   - For Informix: INFORMIX_JDBC_DRIVER_PATH
   - For Oracle: ORACLE_JDBC_DRIVER_PATH
   - For SQL Server: CONNECTJDBC_JDBC_DRIVER_PATH
   - For a user-defined database: User-defined _JDBC_DRIVER_PATH
   - For Cloudscape: not supported for distributed environments

   For more information, see Configure variables.
4. Manage nodes.

   In addition to the steps in the Manage nodes topic in the WebSphere Application Server Network Deployment information center, WebSphere Process Server requires a variable of where the database is installed. On the administrative console, select **Environment > WebSphere Variables > (the scope of cell) >** *DatabaseName*_**JDBC_DRIVER_PATH**. The following variables can be used:

   - For DB2: DB2UNIVERSAL_JDBC_DRIVER_PATH
   - For Informix: INFORMIX_JDBC_DRIVER_PATH
   - For Oracle: ORACLE_JDBC_DRIVER_PATH
   - For SQL Server: CONNECTJDBC_JDBC_DRIVER_PATH
   - For a user-defined database: User-defined _JDBC_DRIVER_PATH
   - For Cloudscape: not supported for distributed environments

   For more information, see Configure variables.
5. Manage node agents according to the instructions in the WebSphere Application Server Network Deployment information center.
6. Manage node groups according to the instructions in the WebSphere Application Server Network Deployment information center.
7. Configure remote file services according to the instructions in the WebSphere Application Server Network Deployment information center.

Configuring the environment

To assist in handling requests among Web applications, Web containers, and application servers, you can configure cell-wide settings for virtual hosts, and variables.

Complete the steps in Configuring the environment in the WebSphere Application Server Network Deployment, Version 6.0.x information center, including the additional steps to configure WebSphere Process Server that are noted below:

- Configure virtual hosts according to the instructions in the WebSphere Application Server Network Deployment information center.
- Configure variables with the following addition to the WebSphere Application Server Network Deployment information center.

  WebSphere Process Server requires a variable of where the database is installed. Add the *DatabaseName*_JDBC_DRIVER_PATH variable for scope of node and cell.

  The following variables can be used:

  – For DB2: DB2UNIVERSAL_JDBC_DRIVER_PATH
  – For Informix: INFORMIX_JDBC_DRIVER_PATH
  – For Oracle: ORACLE_JDBC_DRIVER_PATH
  – For SQL Server: CONNECTJDBC_JDBC_DRIVER_PATH
  – For Sybase: SYBASE_JDBC_DRIVER_PATH
  – For a user-defined database: User-defined _JDBC_DRIVER_PATH
  – For Cloudscape: not supported for distributed environments

Working with server configuration files

You can change the default locations of configuration files, as needed. Server configuration files define the available application servers, their configurations, and their contents.

To work with server configuration files in WebSphere Process Server, follow the instructions in the Working with server configuration files topic in the WebSphere Application Server Network Deployment, Version 6.0.x information center.

Administering process servers

Server configuration provides settings that control how a process server provides services for running enterprise applications and their components. Administrators can create and configure process servers in an existing application server environment.

WebSphere Process Server administrators can configure one or more servers and perform tasks such as the following, described in the links below and in the Administering application servers topic in the WebSphere Application Server Network Deployment, Version 6.0.x information center.

- Create servers.
- Manage application servers.
- Configure transport chains, as described in the WebSphere Application Server Network Deployment, Version 6.0.x information center.
- Develop custom services, as described in the WebSphere Application Server Network Deployment, Version 6.0.x information center.
- Configure the instance of the WebSphere Process Server by following steps in Define processes for the application server in the WebSphere Application Server Network Deployment, Version 6.0.x information center.
- Use the Java virtual machine, as described in the WebSphere Application Server Network Deployment, Version 6.0.x information center.

Creating servers

For WebSphere Process Server, you can create servers using either the createApplicationServer wsadmin commands or the Create New Application Server wizard in the administrative console.

Perform the following steps to create a server:

1. In the administrative console, **Application Servers** page, click **New** to open the Create New Application Server page. You can also create a server using the wsadmin createApplicationServer command. For information, see Commands for the AdminTask object in the WebSphere Application Server Network Deployment, Version 6.0.x information center.
2. On the Create New Application Server page and define your server by doing the following:
   a. Select a node for the server.
   b. Type a name for the server. The name must be unique within the node.
   c. Select the **defaultProcessServer** template to create the server. You can also use an existing application server as a template. The new process server inherits all properties of the template server.
3. Select whether the new server will have unique ports for each HTTP transport. By default, this option is enabled. If you select this option, you may need to update the alias list for the virtual host that you plan to use with this server to contain these new port values. (This includes the host alias that will be used for the Business Process Choreographer Explorer created in step 4 below. The implementation of the Explorer is itself an application and requires host_alias management.) If you clear this option, ensure that the default port values do not conflict with other servers on the same workstation.

   Note: If you create the server using an existing application server as a model, don't select to map applications from the existing server to the new server. By default, this option is disabled.
4. To enable the server to support business processes, configure the Business process container using the following steps:
   a. From the **Application servers** page of the administrative console, expand **Container settings**.
   b. Click **Business process container settings** to run the Business process container installation wizard.
   c. After successful completion of the Business process container installation wizard, use the administrative console prompt to save your changes to the master repository.
   d. Restart to use the Business process container.

   Note: For additional information on configuring the business process container, see Configuring the business process container.

   Note: You can use the business process container without installing the human task container. However, using the Business Process Choreographer explorer requires configuration of both the business process container and the human task container.
5. To enable the server to work with tasks, configure the Human task container using the following steps:
   a. From the **Application servers** page of the administrative console, expand **Human task container settings**.

b. Click **Human task container installation wizard** to run the Human task container installation wizard.
c. After successful completion of the Human task container installation wizard, use the administrative console prompt to save your changes to the master repository.
d. Restart to use the Human task container.

Note: For additional information on configuring the human task container, see Configuring the human task container.

Note: You must install the business process container in order to use the human task container.

6. Install the SchedulerCalendars application on one or more servers using the following steps:
   a. Select **Applications > Install New Application**.
   b. In the file selector window, browse to the **c:\WebSphere\ProcServer\installableApps** directory, where *c:\WebSphere\ProcServer* is the directory where WebSphere Process Server is installed.
   c. Select **ScheduleCalendars.ear**.
   d. Select **Next**.
   e. Accept the default values and select **Next** again.
   f. Continue to accept the default values until you get to the 'Map modules to servers' step, then select the Servers and/or Clusters on which to load the ScheduleCalendars application and select **Next**.
   g. On the Summary step, select **Finish**.
   h. After the application has finished installing, select **Save to Master Configuration**.
   i. Save and Synchronize changes.

   To install the SchedulerCalendars application on additional servers or clusters after the initial installation, use the following steps:
   a. Select **Applications > Enterprise Applications**.
   b. Select **SchedulerCalendars**.
   c. Under the Additional Properties section, select **Map modules to servers**.
   d. Select the checkbox for the **Module Calendars**.
   e. Select the servers and the clusters on which you want to install SchedulerCalendars, be sure to also select any servers or clusters where you want SchedulerCalendars to remain.
   f. Select **Apply**.
   g. Select **OK**.
   h. Save and Synchronize changes with Nodes.
7. The server requires a database to manage the business processes and human tasks to be deployed on the server. Create a Business Process Choreographer database and a WebSphere MQ queue manager (necessary when using WebSphere MQ as the JMS provider) for each deployment target. For more information, see Configuring Business Process Choreographer.

   Note: In a network deployment environment, the Business Process Choreographer database requires additional configuration on the deployment manager. For additional information, see Creating the database for the business process container and Granting permission to the JDBC driver on the deployment manager.

8. Optional: To use multiple language encoding support in the administrative console, configure an application server with UTF-8 encoding enabled, as described in the WebSphere Application Server Network Deployment, Version 6.0.x information center.

Managing application servers

You can use the Application Servers panel of the administrative console, or command line tools such as startServer and stopServer to manage servers.

To view information about an server, use the **Application Servers** panel on the administrative console.

Note: You can upgrade a portion of the nodes in a cell, while leaving others at the older release level. This means that, for a period of time, you may be managing servers that are at the current release and servers that are running the newer release in the same cell. In this mixed environment, there are some restrictions on what you can do with servers that are running the older release level. There are no restrictions on what you can do with the servers that are running on the newer release level. For details, see Creating servers.

Perform the following steps in the administrative console to view and manage an application server, as described in the WebSphere Application Server Network Deployment, Version 6.0.x information center:

- Access the Application Servers page. Click **Servers > Application Servers** in the console navigation tree.
- View information about servers.

  The Application servers page lists the servers and the cell and the nodes holding the servers. It also shows the status of each server. The status indicates whether a server is started, stopped, or unavailable. To view additional information about a particular server or to further configure a server, click a specific server name under **Name**. This accesses the settings page for that server.
- Create a process server for WebSphere Process Server.
- Start a server.
- Monitor the running of servers.
- Stop a server.
- Delete a server.
  1. Click **Servers > Application Servers** in the console navigation tree to access the **Application Servers** page.
  2. Select the check box beside a server to delete it.
  3. Click **Delete**.
  4. Click **OK** to confirm the deletion.

Configuring WebSphere Process Server resources

After installation, you must configure key resources for your WebSphere Process Server environment, such as application scheduler, business processes, Common Event Infrastructure, business rules, selectors and relationships.

An important step in completing your post-installation tasks for setting up the administrative environment is configuring your key resources. Use the links below for more information on setting up these resources.

- Configuring application scheduler
- Configuring Business Process Choreographer
- Configuring the Common Event Infrastructure
- Configuring business rules
- Administering selector components
- Administering the relationship service

Installing and managing applications

After product installation, you must complete a number of post-installation tasks concerning the administration of applications and their environments.

After installation, you must administer applications and their environment, including customizing tasks, deploying applications and administering server environments.

- For more information, refer to Administering applications and their environment in the WebSphere Application Server Network Deployment, Version 6.0.x information center.
- For more information on installing applications on WebSphere Process Server, refer to Overview of preparing and installing modules.
- For information on stopping, starting and modifying applications, refer to Deploying and administering applications in the WebSphere Application Server Network Deployment, Version 6.0.x information center.

Administering applications and application services

This section describes how to use the administrative interfaces to administer WebSphere Process Server applications and application services, including business processes and tasks, business rules, and schedules.

Configuring WebSphere Process Server settings

This topic provides an overview of configuring settings for Business Integration Server, as well as links to specific tasks.

Before configuring the WebSphere Process Server, you must install the server.

Upon successful installation of the WebSphere Process Server, you must complete configuration by specifying settings for the server. The following sections describe the information and steps necessary to complete configuration of the WebSphere Process Server.

1. Configure the Application Schedule settings.

   "Configuring the Application Scheduler for a standalone server" on page 17
2. Configure the Events Service settings.

   "Managing Events Service" on page 17
3. Configure Extended Messaging service.

   "Configuring WebSphere Process Server settings"
4. Configure Staff service.

   "Configuring the staff plug-in provider" on page 18

5. Configure WBI Adapter service.

   "Working with WebSphere Business Integration Adapters" on page 21
6. Configuring the relationship service.

   "Configure Relationship Service" on page 25

Configuring the Application Scheduler for a standalone server

To use Application Scheduler, you must make sure that it is installed. See Creating and augmenting profiles by using the Profile Wizard elsewhere in the information center.

You need to create a profile first.

This is an optional component and you must configure Application Scheduler to migrate WebSphere InterChange Server schedule entries to WebSphere Process Server Complete these steps to install the application server for a standalone server.

1. Type in the*Servername* (or use the default of *server1*).
2. Create or use an existing database. If you are creating a database name the database **WPRCSDB** (all caps). The server uses the default database tables and structure.

The installation is complete.

Application Scheduler is ready to use.

Managing Events Service

The Events Service configuration panel lists the properties set to ensure that information about the WebSphere server is automatically included in each event passed to the Events Infrastructure.

Use the Events Service configuration panel to set the events service configuration to on or off.

The Events Service configuration panel provides access to the events infrastructure for WebSphere applications. When active, the events service ensures that any event that passes to the events infrastructure automatically includes information about the WebSphere server.

To configure the Events Service Startup property for an application server, use the administrative console to complete the following steps:

1. Start the Administrative Console.
2. Select the application server to configure.

   In the navigation panel, click **Servers > Application servers**. Select the application server from the list.

   The server properties display.
3. Display the Events Service configuration panel.

   From the Business Integration properties, select **Events service**.

   The Events Service configuration panel displays.
4. Select or deselect the Enable service at startup field.

   The default value is to enable the service automatically at startup. If this option is deselected and subsequent applications require this service to run, the system administrator must manually start the service. Optionally, the system administrator can restart this service by selecting the option and restarting the server.

5. Verify that the JNDI name is correct.

   Review the Events infrastructure emitter factory JNDI name field. If you have not generated an alternative profile, it is recommended that you accept the default JNDI name.
6. Select any custom properties, if necessary.

   Click the Custom properties link and select any custom properties from the listing.
7. Save your configuration.

   Click **Apply** to save your configuration and remain on the Events Service configuration panel. Click **OK** to save your configuration and return to the Business Integration panel.
8. Apply your changes to the server.

   Stop and restart the application server to apply your changes.

Enabling the extended messaging service

The Extended Messaging Service provides runtime service to support container-managed messaging (extended messaging). The service can be automatically started when the application server is started, or it can be started manually.

Security role required: You must be logged in as administrator or configurator to complete this task.

1. Ensure that the administrative console is running.
2. Click **Servers > Application servers >** *server_name* **> Extended Messaging Service** to display the Extended Messaging Service page.
3. If you want to enable the Extended Messaging Service to start automatically with server startup, select the **Enable service at server startup** check box. If you want to start the service manually, ensure the check box is cleared.
4. Click **OK**.
5. When prompted, click **Save** on the console task bar to save your changes to the master repository.
6. Stop and restart the application server in order for the changes to take effect.

You can also use the Extended Messaging Service page to configure a listener port extension to handle late responses, as described in “Configuring WebSphere Process Server settings” on page 16.

Configuring the staff plug-in provider

Use this task to configure the staff plug-in provider that Business Process Choreographer uses to determine who can start a process or claim an activity or a task.

Each type of supported user directory service requires a corresponding staff plug-in. The following staff plug-ins are supported:

Table 2. Supported staff plug-in providers

User directory service	Plug-in provider
Lightweight Directory Access Protocol (LDAP)	`LDAP_Staff_Plugin_Provider`
Local operating system user registry	`System_Staff_Plugin_Provider`
WebSphere Application Server user registry	`User_Registry_Staff_Plugin_Provider`

All of these plug-ins are already installed. You can use the user registry and system plug-ins without any configuration.

The LDAP staff plug-in is configured for an LDAP server with anonymous access; the LDAP server is local to the installed application server. You can change the configuration of the LDAP plug-in.

1. In the administrative console, click **Resources** → **Staff Plugin Provider**.
2. To create a new LDAP configuration:
   a. Click the name of the LDAP staff plug-in provider.
   b. Select **Staff Plugin Configuration**.
   c. Click **New** → **Browse**, and select the sample Extensible Stylesheet Language (XSL) transformation file to use. The standard XSL transformation for LDAP is located:
      - On Windows systems, in `install_root\ProcessChoreographer\Staff\LDAPTransformation.xsl`
      - On Linux and UNIX systems in `install_root/ProcessChoreographer/Staff/LDAPTransformation.xsl`

      Do not modify this transformation file. If you need to customize the transformations to match the LDAP schema of your organization, modify a copy that has a different file name.
   d. Click **Next**.
   e. Enter an administrative name for the staff plug-in provider.
   f. Enter a description.
   g. Enter the Java Naming and Directory Interface (JNDI) name for business processes to use in referencing this plug-in, for example, `bpe/staff/ldapserver1`
   h. Click **Apply**.
   i. Click **Custom Properties**.
   j. For each of the required properties and for any optional properties that you want to set, click the name of the property, enter a value, and click **OK**.
   k. To apply the changes, click **Save**. This table describes each property for the LDAP plug-in.

LDAP plug-in property	Required or optional	Comments
AuthenticationAlias	Optional	The authentication alias used to connect to LDAP, for example, `mycomputer/My LDAP Alias`. You must define this alias in the administrative console by clicking **Security** → **JAAS** → **Configuration JAAS Configuration** → **J2C Authentication Data**. If this alias is not set, anonymous logon to the LDAP server is used.

LDAP plug-in property	Required or optional	Comments
AuthenticationType	Optional	If the AuthenticationType property is not set, the default logon is anonymous authentication. In all other cases, the default is simple authentication.
BaseDN	Required	The base distinguished name (DN) for all LDAP search operations, for example, "`o=mycompany, c=us`"
CasesentivenessForObjectclasses	Optional	Determines whether the names of LDAP object classes are case-sensitive.
ContextFactory	Required	Sets the Java Naming and Directory Interface (JNDI) context factory, for example, `com.sun.jndi.ldap.LdapCtxFactory`
ProviderURL	Required	This Web address must point to the LDAP JNDI directory server and port. The format must be in normal JNDI syntax, for example, `ldap://localhost:389`
SearchScope	Required	The default search scope for all search operations. Determines how deep to search beneath the baseDN property. Specify one of the following values: `objectScope`, `onelevelScope`, or `subtreeScope`
additionalParameterName1-5 and additionalParameterValue1-5	Optional	Use these name-value pairs to set up to five arbitrary JNDI properties for the connection to the LDAP server.

3. To activate the plug-in, stop and start the server.
4. If you have problems with any of these steps, refer to troubleshooting the staff service and staff plug-ins.

Processes can now use the staff support services to resolve staff queries, and to determine which activities can be performed by certain people.

Continue configuring at step 2.

Depending on the queries that you want to create and your directory structure, you might need to create your own transformations. For more information about this topic, see About the staff service.

Staff service settings

Use this page to enable or disable the staff service, which manages staff plug-in resources used by the server.

To view this administrative console page, click **Servers** → **Application Servers** → *server_name*. Then click **Business Integration** → **Staff Service**.

Enable service at server startup:

Specifies whether the server attempts to start the staff service.

Default	Selected

Range	**Selected** When the application server starts, it attempts to start the staff service automatically. **Cleared** The server does not try to start the staff service. If staff plug-in resources are used on this server, the system administrator must start the staff service manually or select this startup property and then start the server again.

Working with WebSphere Business Integration Adapters

WebSphere Business Integration Adapters can be installed and administered using the administrative console.

To perform this task you must have access to the administrative console, and have security permission to alter server settings within the console.

These instructions assume that you have a WebSphere Business Integration Adapter that you wish to administer. Perform the steps described in subsequent topics to configure and use the WebSphere Business Integration adapter from the administrative console.

Generating service component definitions and the MQClientLink configuration file

Before using WebSphere Business Integration Adapters, it is necessary to generate service component definitions (SCA artifacts) and the MQClientLink configuration file (.wbia file). This is achieved with the WebSphere Business Integration Adapter Artifact Importer in the WebSphere Integration Developer environment.

The WebSphere Integration Developer uses the WebSphere Business Integration Adapter Artifact Importer to discover and import the WebSphere Business Integration Adapter Connector Configuration File and WebSphere Business Integration Adapter Business Objects directory and generate the desired service component definitions supporting the specified interaction-styles for the WebSphere Business Integration Adapter.

Note: This task is performed in WebSphere Integration Developer, and is described here only for reference. See the WebSphere Integration Developer information center for more details.

1. Obtain the necessary configuration files and Business Objects.
2. Use the WebSphere Business Integration Adapter Artifact Importer to generate the necessary service component definitions (SCA artifacts).

The service component definitions (SCA artifacts) and the WebSphere Process Server MQClientLink configuration file (.wbia file) are generated.

When an application is deployed to WebSphere Process Server, the service component definitions and the MQClientLink configuration (.wbia) file are handled automatically by the deployment tools. It is recommended that the configuration file be left in its default state, but it is possible to manually edit the file if required.

Configuring WebSphere Business Integration Adapters

The process of configuring and using the WebSphere Business Integration Adapters is three-fold:

1. Install the application EAR file.

   Installation of the application EAR file is accompanied by the creation of all necessary artifacts for the WebSphere Business Integration Adapter to work.
2. Set up the administration of the WebSphere Business Integration Adapter.

   In order to administer a WebSphere Business Integration Adapter you must:
   a. Create a connection queue factory
   b. Create a WebSphere Business Integration Adapter resource
   c. Enable the WebSphere Business Integration Adapter service.

      Note: When you enable or disable a WebSphere Business Integration Adapter service, you must restart the server in order for the changes to take effect.
3. Manage the WebSphere Business Integration Adapter

   Use the administrative console to manage the WebSphere Business Integration Adapters.

Create the artifacts necessary for the WebSphere Business Integration Adapter to work:

To create the artifacts necessary for the WebSphere Business Integration Adapter to work, install the application EAR file.

In order to create the artifacts that you need to use a WebSphere Business Integration Adapter in WebSphere Process Server, you should follow the instructions for installing an application.

Setting up administration of a WebSphere Business Integration Adapter:

You must perform several administrative functions before you can manage a WebSphere Business Integration Adapter. The required steps are described here.

You must install the application EAR file to create the artifacts required for the WebSphere Business Integration Adapter before you perform this task.

In order to have administrative control over a WebSphere Business Integration Adapter you must first perform the following administrative functions.

1. Create a Queue Connection Factory.

   From the top level of the administrative console follow these steps:
   a. Expand **Resources**.
   b. Expand **JMS Providers**.
   c. Select **Default Messaging**.
   d. Select **JMS queue connection factory**.

      Under the JMS subheading select **JMS queue connection factory**.
   e. Create a new JMS queue connection factory.

      Click **New**.
   f. Accept all the default values with the following exceptions:
      - Name: QueueCF

- JNDI Name: jms/QueueCF
- Bus. Name: *your business name*

g. Complete the creation of your new JMS queue connection factory.

   Click **OK**.

   A message window appears at the top of the JMS queue connection factory panel.

h. Apply the changes that you have made at the local configuration level to the master configuration.

   Click **Save** in the message window.

2. Create a WebSphere Business Integration Adapter resource

   From the top level of the administrative console follow these steps:

   a. Expand **Resources**.

   b. Open the WebSphere Business Integration Adapters panel.

      Select **WebSphere Business Integration Adapters**.

   c. Create a new WebSphere Business Integration Adapter.

      Click **New**.

   d. Accept all the default values with the following exceptions:
      - Name: EISConnector
      - Queue Connection Factory JNDI Name: jms/QueueCF
      - Administration Input Queue JNDI Name: *connectorName/AdminInQueue*
      - Administration Output Queue JNDI Name: *connectorName/AdminOutQueue*

   e. Complete the creation of the WebSphere Business Integration Adapter.

      Click **OK**.

      A message window appears at the top of the WebSphere Business Integration Adapters panel.

   f. Apply the changes that you have made at the local configuration level to the master configuration.

      Click **save** in the message window.

3. Enable the WebSphere Business Integration Adapter Service.

   From the top level of the administrative console follow these steps:

   a. Expand **Servers**.

   b. Select **Application Servers**.

   c. From the list of servers select a server where the WebSphere Business Integration Adapter Service is to be enabled.

      Click on the name of the server that hosts the resources of interest.

   d. Select **WebSphere Business Integration Adapter Service**.

      Under the Business Integration subheading on the Configuration tab select **WebSphere Business Integration Adapter Service**.

   e. Ensure that the **Enable Service at startup** check box is selected.

   f. Click **OK**.

      A message window appears at the top of the WebSphere Business Integration Adapters panel.

   g. Repeat steps 3c to 3f for each server on which the WebSphere Business Integration Adapter Service is to be enabled.

   h. Apply the changes that you have made at the local configuration level to the master configuration.

Click **save** in the message window.

Note: When you enable or disable a WebSphere Business Integration Adapter service, you must restart the server in order for the changes to take effect.

Managing the WebSphere Business Integration Adapter:

When a WebSphere Business Integration Adapter is running, it can be managed using the Manage the WebSphere Business Integration Adapter resources panel on the administrative console.

The WebSphere Business Integration Adapter must be running in order to be managed.

You can manage a WebSphere Business Integration Adapter from the administrative console. The Manage the WebSphere Business Integration Adapter resources panel allows you to choose one or more resources to manage and perform various administrative actions upon these resources.

1. Select the resource or resources to manage.

   From the top level of the administrative console follow these steps:

   a. Expand Servers.
   b. Select Application Servers.
   c. From the list of servers select the server where the resources you intend to manage reside.

      Click on the name of the server that hosts the resources of interest.
   d. Select WebSphere Business Integration Adapter Service

      Under the Business Integration subheading on the Configuration tab select WebSphere Business Integration Adapter Service.
   e. Select Manage the WebSphere Business Integration Adapter resources.
   f. From the list of resources choose those that you want to manage.

      Select the check boxes associated with the resources you intend to manage.
2. Manage the selected resources.

   Click one of the command buttons to act upon the selected resources.

Command	**Description**
Deactivate	Changes the status of the selected resources from active to paused or inactive.
Activate	Changes the status of the selected resources from inactive to active.
Suspend	Changes the status of the selected resources from active to paused.
Resume	Changes the status of the selected resources from paused to active.
Shutdown	Changes the status of the selected resources from active to unavailable.

WebSphere Business Integration Adapter applications as members of clusters

Cluster deployment is not supported in WebSphere Process Server v6.0. This information is included for completeness. WebSphere Business Integration Adapter module applications can be cloned as members of a cluster under certain conditions.

WebSphere Business Integration Adapter module applications can be one of three types, depending on the flow of information through the adapter:

- A WebSphere Business Integration Adapter application with only EIS exports - only inbound traffic.
- A WebSphere Business Integration Adapter application with only EIS imports - only outbound traffic.
- A WebSphere Business Integration Adapter application with both EIS imports and exports - bidirectional traffic.

Clusters are used to provide scalability and availability to your applications in a network deployment environment.

WebSphere Business Integration Adapter module applications that have either inbound or bidirectional traffic, cannot be cloned as members of a cluster. An application with purely outbound traffic can be cloned as a member of a cluster.

An application which has inbound or bidirectional WebSphere Business Integration Adapter (i.e., including EIS exports) can still be given availability in a network deployment by use of an external Operating System High Availability (HA) management software package, such as HACMP, Veritas or Microsoft Cluster Server.

Administering the relationship service

The Relationship Service allows relationships and roles to be explicitly represented and persisted and enables the management and manipulation of relationship instances at runtime.

The Relationship Service exposes a set of application programming interfaces (APIs) to various tools, to enable user interaction with the relationships and roles.

- Using Relationship Designer, you design and model relationships and roles and generate the descriptor files for them. The relationship and role definitions are deployed as part of a J2ee application. At run time, instances of the relationships are populated with data that associates information from different applications. The data is stored in the relationship tables specified in the relationship definition.
- Using Relationship Manager, you can create and manipulate (retrieve, view, modify, delete) the relationship runtime instance data.

For more information, see the topics on the Relationship Designer in the WebSphere Integration Developer Information Center, Relationship Manager, and the administrative console.

Configure Relationship Service

Configure parameters for the Relationship Service.

You must be authenticated as a Configurator or Administrator in order to modify this configuration. Any WebSphere security role can view this configuration.

1. Open the Relationship Manager panel of the administrative console. Log into the WebSphere Process Server administrative console and navigate to **Integration Applications > Relationship Manager**.
2. Click **Relationship Services configuration**.
3. Specify the default data source for the Relationship Service in the Data source field.
4. Specify the maximum results to be shown per page when relationship instances are queried using Relationship Manager in the Page Size field.
5. You then have the following options:
   - Click **Apply** to update the entry with your information and keep you in the panel.
   - Click **OK** to update the entry with your information and return you to the previous panel.
   - Click **Reset** to return the entry to the most previous set of changes made.
   - Click **Cancel** to discard any changes made and return you to the previous panel.

View relationships managed by the Relationship Service

View a list of the relationships that this Business Integration Relationship Service manages. Each row shows the version and data source for the associated relationship type.

Any WebSphere security role can view this configuration.

1. Open the Relationship Manager panel of the administrative console. Log into the WebSphere Process Server administrative console and navigate to **Integration Applications > Relationship Manager**.
2. Click **Relationship Services configuration > Relationships**.
3. To set the maximum number of rows to be displayed when the collection is large, click **Preferences**. Those rows not displayed will appear on the next page. To save the search criteria for this display, select the check box next to **Retain filter criteria**, and click **Apply** (or **Reset**, as necessary). When you return to this panel, this information will be displayed.

Configure relationship parameters

View the configuration parameters for individual relationships that the Relationship Service manages.

Any WebSphere security role can view this configuration.

1. Open the Relationship Manager panel of the administrative console. Log into the WebSphere Process Server administrative console and navigate to **Integration Applications > Relationship Manager**.
2. Click **Relationship Services configuration > Relationships**.
3. Click on the relationship name from the table.
4. The configuration parameters for the fields are automatically set. To return to the Relationship collection panel, click **Back**.

Administering Business Process Choreographer

You can use the administrative console to manage some aspects of Business Process Choreographer, such as administering the compensation service or querying and replaying failed messages. Scripts are provided for other administrative tasks, such as deleting audit log entries.

You can use the administrative console to do the following administrative tasks:
- “Administering the compensation service for a server”
- “Querying and replaying failed messages, using the administrative console” on page 28
- “Refreshing staff queries, using the administrative console” on page 33
- “Enabling Common Base Events and the audit trail” on page 35

You can use scripts to do the following administrative tasks:
- “Querying and replaying failed messages, using administrative commands” on page 31
- “Refreshing staff queries, using administrative commands” on page 33
- “Deleting audit log entries, using administrative commands” on page 37
- “Removing unused staff queries, using administrative commands” on page 38
- “Deleting process templates and task templates that are no longer valid” on page 39

Administering the compensation service for a server

Use the administrative console to start the recovery service automatically, when the application starts, and to specify the location and maximum size of the recovery log.

The compensation service must be started on an application server, when business processes are run on that server. The compensation service is used to manage updates that might be made in a number of transactions before the process completes.

You can use the administrative console to view and change properties of the compensation service for application servers.

1. Display the administrative console.
2. In the navigation pane, click **Servers** → **Application servers** → *server_name*.
3. On the Configuration tab, under Container Settings, click **Container services** → **Compensation service**. This action displays a panel with the compensation service properties.

   Enable service at server startup
   Specifies that, whenever the application server starts, it automatically tries to start the compensation service.

   Important: Make sure that this check box is selected. The compensation service must be enabled when you use business processes.

   Recovery log directory
   Specifies the name of a directory for this server where the compensation service stores log files for recovery. When compensation is used, the WebSphere product stores information that is required for compensation.

   Recovery log file size
   Specifies the maximum size, in megabytes, for compensation log files on this application server.
4. **Optional:** If necessary, change the compensation service properties.
5. Click **OK**.

6. To save your configuration, click **Save** in the Messages box of the administrative console window. Then click **Save** on the Application Servers Save pane.

Refreshing the failed message counts

Use the administrative console to refresh the count of failed messages.

The displayed number of messages on the hold queue and on the retention queue, and the number of message exceptions, remain static until refreshed. This task describes how to update and display the number of messages on those queues and the number of message exceptions.

1. Select the appropriate application server.

   Click **Servers** → **Application servers** → *server_name*.
2. Refresh the message counts.

   On the **Configuration** tab, in the Container Settings section, click one of the following sequences:
   - For business processes: **Business process container settings** → **Replay messages** → **Refresh Message Counts**
   - For human tasks: **Human task container settings** → **Replay messages** → **Refresh Message Counts**

The following updated values are displayed under **General Properties**:
- For business processes: The number of messages on the hold queue and on the retention queue
- For human tasks: The number of messages on the hold queue
- The number of message exceptions

Querying and replaying failed messages, using the administrative console

When a problem occurs while processing an internal message, this message ends up in the retention queue or hold queue.

This task describes how to determine whether any failed messages exist, and to send those messages to the internal queue again.

1. To check how many messages are on the hold and retention queues:
   a. Click **Servers** → **Application servers** → *server_name*.
   b. On the **Configuration** tab, in the Container Settings section, click one of the following sequences:
      - For business processes: **Business process container settings** → **Replay messages**
      - For human tasks: **Human task container settings** → **Replay messages**

      The number of messages on the hold queue and retention queue are displayed under **General Properties**.
2. If either the hold queue or the retention queue contains messages, you can move the messages to the internal work queue.

   Click one of the following options:
   - For business processes: **Replay Hold Queue** or **Replay Retention Queue**
   - For human tasks: **Replay Hold Queue**

The business process engine of the Business Process Choreographer tries to service all replayed messages again.

Business Process Choreographer: Failed message handling and quiesce mode

Business Process Choreographer provides a facility for handling temporary infrastructure failures. This facility consists of two numerical limits, two queues, quiesce mode, and the message retry behavior.

Long-running processes consist of a sequence of transactions. The transactions are separated by Java Message Service (JMS) messages, which the server sends to a message-driven bean. This bean passes the incoming messages to the process server, for processing. Each transaction consists of the following actions:

- Receive a message.
- Navigate, on behalf of the message.
- Send messages that trigger follow-on transactions.

The server might fail to process a message received by the message-driven bean for either of the following reasons:

- A specified number of consecutive messages cannot be processed. The infrastructure is therefore assumed to be unavailable.
- Only some messages can be processed. Any single message that cannot be processed is assumed to be damaged.

The responses to these causes are as follows:

Cause	**Response**
Unavailable infrastructure	The message-driven bean tries, for a specified time, to recover from that situation. It tries to keep all messages available until the server is again operational. This problem might be caused by a database failure, for example.
Damaged message	After a specified number of retries, the message is put into the hold queue, where it can be manipulated or reviewed. From the hold queue, it can also be moved back to the input queue, to retry the transaction.

The implementation is as follows:

- If a message fails to be processed, the server puts it into a retention queue, where it is kept available, in case this is an infrastructure problem that is fixed within a specified time.
- When a message is in the retention queue, the options are as follows:
  - When a subsequent message can be processed successfully, all messages from the retention queue are moved back to the input queue of the message-driven bean. For each message, a count is maintained of how often the message has been sent to the retention queue. If this count exceeds the retry limit for a given message, the message is put in the hold queue.
  - If the next message fails to be processed, it is also put in the retention queue. This process continues until the threshold of maximum messages in the retention queue is reached. When this threshold is reached, the message-driven bean moves all messages from the retention queue to the input queue and switches into quiesce mode.

When the message-driven bean operates in quiesce mode, it periodically tries to process a message. Messages that fail to be processed are put back in the input queue, without incrementing either the delivery count or the retention queue traversal count. As soon as a message can be processed successfully, the message-driven bean switches back into normal processing mode.

Retry limit

The retry limit defines the maximum number of times that a message can be transferred through the retention queue before being put on the hold queue.

To be put on the retention queue, the processing of a message must fail three times.

For example, if the retry limit is 5, a message must go through the retention queue five times (it must fail for 3 * 5 = 15 times), before the last retry loop is started. If the last retry loop fails two more times, the message is put on the hold queue. This means that a message must fail (3 * *RetryLimit*) + 2 times before it is put on the hold queue.

In a performance-critical application running in a reliable infrastructure, the retry limit should be small: one or two, for example. This parameter can be found in the administrative console, on the Business Process Container configuration page.

Retention queue message limit

The retention queue message limit defines the maximum number of messages that can be on the retention queue. If the retention queue overflows, the system goes into quiesce mode. To make the system enter quiesce mode as soon as one message fails, set the value to zero. To make the business process container more tolerant of infrastructure failures, increase the value. This parameter can be found in the administrative console, on the Business Process Container page. (To locate this parameter, click **Servers** → **Application Servers** → *server_name*. Then, under the heading Business Process Settings, click **Business Process Container**. The **Retry Limit** field is under the heading General Properties.)

Retention queue

The retention queue holds failed messages that are replayed by moving them back to the business process container's internal work queue. A message is put on the retention queue if it fails three times. If the message fails (3 * *RetryLimit*) + 2 times, it is put on the hold queue. (For details of the retry limit, see "Retry limit.") If the retention queue is full to the limit defined by the retention queue message limit and another message fails, the queue overflows, and the system goes into quiesce mode. The administrator can move the messages in this queue back to the internal queue performing the task Querying and replaying failed messages.

Hold queue

The hold queue contains messages that have failed (3 * *RetryLimit*) + 2 times. (For details of the retry limit, see "Retry limit.") The administrator can move the messages in this queue back to the internal queue performing the task Querying and replaying failed messages.

Replay Messages

The administrator can move the messages from the hold or retention queues back to the internal queue. This is done using the Replay Messages panel for the business process container.

Quiesce Mode

Quiesce mode is entered when the retention queue overflows. When this happens, it is assumed that there is a serious, though possibly temporary, infrastructure failure. The purpose of quiesce mode is to prevent the system from using a lot of resources, while an infrastructure failure means that most messages will probably fail anyway. In quiesce mode, the system sleeps for two seconds before attempting to process the next message. As soon as a message is successfully processed, the system resumes normal message processing.

Querying and replaying failed messages, using administrative commands

Use the administrative commands to determine whether there are any failed messages and, if there are, to return them to the internal queue.

Before you begin this procedure, the following conditions must be met:

- The user ID that you are using must have administrative rights.
- The application server on which the messages are to be queried or replayed must be running. That is, the `-conntype none` option of the wsadmin script cannot be used, because a server connection is required.

When a problem occurs while processing an internal message, this message ends up on the retention queue or hold queue. To determine whether any failed messages exist, and to send those messages to the internal queue again:

1. Change to the Business Process Choreographer utilities directory where the scripts are located:

   On Windows® systems, enter:

   ```
   cd install_root\ProcessChoreographer\util
   ```

   On UNIX® and Linux® systems, enter:

   ```
   cd install_root/ProcessChoreographer/util
   ```
2. Query the number of failed messages on both the retention and hold queues. Enter one of the following commands:

   On Windows systems, enter:

   ```
   install_root\bin\wsadmin -f queryNumberOfFailedMessages.jacl
                               -cluster clusterName
                               [-profileName profileName]

   install_root\bin\wsadmin -f queryNumberOfFailedMessages.jacl
                               -node nodeName
                               -server serverName
                               [-profileName profileName]
   ```

   On UNIX and Linux system, enter:

   ```
   install_root/bin/wsadmin.sh -f queryNumberOfFailedMessages.jacl
                               -cluster clusterName
                               [-profileName profileName]

   install_root/bin/wsadmin.sh -f queryNumberOfFailedMessages.jacl
   ```

```
                              -node nodeName
                              -server serverName
                              [-profileName profileName]
```

Where:

cluster *clusterName*
: The name of the cluster. Required if the business process container is configured for a WebSphere cluster.

node *nodeName*
: Optional when specifying the server name. This name identifies the node. The default is the local node.

server *serverName*
: The name of the server. Required if the cluster name is not specified.

profileName *profileName*
: The name of a user-defined profile. Specify this option if you are not working with the default profile.

If you want to check for a server on the local node, enter:

```
wsadmin -f queryNumberOfFailedMessages.jacl -server serverName
```

3. Replay all failed messages on the hold queue, retention queue, or both queues by entering one of the following commands:

   On Windows systems, enter:

```
install_root\bin\wsadmin -f replayFailedMessages.jacl
                         -cluster clusterName
                         -queue replayQueue
                         [-profileName profileName]

install_root\bin\wsadmin -f replayFailedMessages.jacl
                         -node nodeName
                         -server serverName
                         -queue replayQueue
                         [-profileName profileName]

install_root\bin\wsadmin -f replayFailedMessages.jacl
                         -server serverName
                         -queue replayQueue
                         [-profileName profileName]
```

   On UNIX and Linux systems, enter:

```
install_root/bin/wsadmin.sh -f replayFailedMessages.jacl
                         -cluster clusterName
                         -queue replayQueue
                         [-profileName profileName]

install_root/bin/wsadmin.sh -f replayFailedMessages.jacl
                         -node nodeName
                         -server serverName
                         -queue replayQueue
                         [-profileName profileName]

install_root/bin/wsadmin.sh -f replayFailedMessages.jacl
                         -server serverName
                         -queue replayQueue
                         [-profileName profileName]
```

   Where:

   queue *replayQueue*
   : Must have one of the following values:

     `holdQueue`

     `retentionQueue`

```
both
```

cluster *clusterName*
The name of the cluster. Required if the business process container is configured for a WebSphere cluster.

node *nodeName*
Optional when specifying the server name. This name identifies the node. The default is the local node.

server *serverName*
The name of the server. Required if the cluster name is not specified.

profileName *profileName*
The name of a user-defined profile. Specify this option if you are not working with the default profile.

Refreshing staff queries, using the administrative console

Use the administrative console to refresh staff queries. Staff queries remain static until refreshed.

Business Process Choreographer caches the results of staff assignments evaluated against a staff directory, such as an Lightweight Directory Access Protocol (LDAP) server, in the run-time database. If the staff directory changes, you can force the staff assignments to be evaluated again.

To refresh the staff queries:

1. Click **Servers** → **Application servers** → *server_name*.
2. On the **Configuration** tab, in the Container Settings section, click **Human task container settings** → **Replay messages** → **Refresh Staff Query**.

All staff queries are refreshed.

Refreshing staff queries, using administrative commands

Use the administrative commands to refresh staff queries. Staff queries remain static until refreshed.

Before you begin this procedure, the following conditions must be met:

- The user ID that you are using must have administrative rights.
- The application server on which the messages are to be queried or replayed must be running. That is, the `-conntype none` option of wsadmin cannot be used, because a server connection is required.

Business Process Choreographer caches the results of staff assignments evaluated against a staff directory, such as an Lightweight Directory Access Protocol (LDAP) server, in the run-time database. If the staff directory changes, you can force the staff assignments to be evaluated again.

1. Change to the Business Process Choreographer utilities directory where the scripts are located.

   On Windows systems, enter:

   ```
   cd install_root\ProcessChoreographer\util
   ```

   On UNIX and Linux systems, enter:

   ```
   cd install_root/ProcessChoreographer/util
   ```
2. Force the staff assignment to be evaluated again.

On Windows systems, enter one of the following commands:

```
install_root\bin\wsadmin -f refreshStaffQuery.jacl
                         -server serverName
                         [-template templateName |
                          -userList username{,username}...]
                         [-profileName profileName]

install_root\bin\wsadmin -f refreshStaffQuery.jacl
                         -node nodeName
                         -server serverName
                         [-template templateName |
                         -userList username{,username}...]
                         [-profileName profileName]

install_root\bin\wsadmin -f refreshStaffQuery.jacl
                         -cluster clusterName
                         [-template templateName |
                          -userList username{,username}...]
                         [-profileName profileName]
```

On UNIX and Linux systems, enter one of the following commands:

```
install_root/bin/wsadmin.sh -f refreshStaffQuery.jacl
                         -server serverName
                         [-template templateName |
                          -userList username{,username}...]
                         [-profileName profileName]

install_root/bin/wsadmin.sh -f refreshStaffQuery.jacl
                         -node nodeName
                         -server serverName
                         [-template templateName |
                          -userList username{,username}...]
                         [-profileName profileName]

install_root/bin/wsadmin.sh -f refreshStaffQuery.jacl
                         -cluster clusterName
                         [-template templateName |
                          -userList username{,username}...]
                         [-profileName profileName]
```

Where:

cluster *clusterName*
: The name of the cluster. Required if the business process container is configured for a WebSphere cluster.

node *nodeName*
: Optional when specifying the server name. This name identifies the node. The default is the local node.

server *serverName*
: The name of the server. Required if the cluster name is not specified.

template *templateName*
: The name of the process template. Staff assignments that belong to this process template are refreshed.

userList *userName*
: A comma-separated list of user names. Staff assignments that contain the specified names are refreshed.

profileName *profileName*
: The name of a user-defined profile. Specify this option if you are not working with the default profile.

Attention: If you omit both `templateName` and `userList`, all staff queries are refreshed.

Enabling Common Base Events and the audit trail

Use this task to enable Business Process Choreographer events to be emitted to the Common Event Infrastructure as Common Base Events or stored in the audit trail.

You can change the state observers settings permanently on the Configuration tab of the Business process container pane or temporarily on the Runtime tab.

Any choices you make on these Configuration or Runtime tabs affect all applications.

Changing the configured infrastructure

Use this task to change the configured infrastructure.

Choices made on the Configuration tab are activated the next time the server is started. The chosen settings remain in effect whenever the server is started.

Make changes to the configuration, as follows:

1. Display the Business process container or Human task container pane.

   Click **Servers** → **Application servers** → *server_name*. Then, under Container Settings, click one of the following sequences:
   - For business processes: **Business process container settings** → **Business process container**
   - For human tasks: **Human task container settings** → **Human task container**
2. In the General Properties section, select the logging to be implemented. The state observers are independent of each other: you can enable or disable either or both of them.

   Enable Common Event Infrastructure logging
   Select this check box to enable event emission that is based on the Common Event Infrastructure.

   Enable audit log
   Select this check box to store the audit log events in the audit trail tables of the relational database.
3. Accept the change.
   a. Click **Apply**.
   b. In the Messages box, click **Save**.
   c. On the Application servers pane, click **Save**.

The state observers are set, as you required. The changes take place after server restart.

Restart the container, to effect the changes.

Configuring the infrastructure for the session

Use this task to configure the infrastructure for the session.

Choices made on the Runtime tab are effective immediately. The chosen settings remain in effect until the next time the server is started.

Make changes to the session infrastructure, as follows:

1. Display the Replay messages pane.

   Click **Servers** → **Application servers** → *server_name*. Then, under Container Settings, click one of the following sequences:
   - For business processes: **Business process container settings** → **Business process container**
   - For human tasks: **Human task container settings** → **Human task container**
2. In the **State observer logging** field, select the logging to be implemented. The state observers are independent of each other: you can enable or disable either or both of them.

   Enable Common Event Infrastructure logging
   Select this check box to enable event emission that is based on the Common Event Infrastructure.

   Enable audit log
   Select this check box to store the audit log events in the audit trail tables of the relational database.
3. Accept the change.

   Click **Update State Observers**.

The state observers are set, as you required.

Event emission and storage

Events are generated on behalf of the execution of processes. The events contain information about state changes of business process elements and human task elements.

Two infrastructures emit or store the events, such that those events can be retrieved by applications. Applications might use events to monitor business processes and to analyze the history of business processes and human tasks.

Because the generation of events impacts system performance, you can select the infrastructure to be used to store and emit events:

Common Event Infrastructure
Events can be both stored and published to subscribing applications. To use this event infrastructure, make sure that the Common Event Infrastructure is installed and configured.

Use event emission that is based on the Common Event Infrastructure to retrieve events in the format of Common Base Events, through the application programming interface (API) of the Common Event Infrastructure. You can connect consuming applications either by subscription or by using the query-oriented interface of the Common Event Infrastructure.

Event emission that is based on the Common Event Infrastructure has considerably greater impact on system performance than have audit log events. However, it provides greater flexibility for consuming applications.

Audit trail
Events are stored as records of a table in a relational database.

This is a fast event-storing infrastructure that has little impact on performance. Consuming applications need Structured Query Language (SQL) queries to retrieve the events from the database.

You can select either, both, or neither of the infrastructures. Selecting an infrastructure does not imply that events are necessarily stored or emitted. The selection enables the infrastructure, while you can control the actual generation of events later, by additional mechanisms. However, the enablement of an infrastructure results in a basic overhead that affects system performance.

Deleting audit log entries, using administrative commands

Use the administrative commands to delete audit log entries.

Before you begin this procedure, the following conditions must be met:

- The user ID that you are using must have administrative rights.
- The application server on which audit log entries are to be deleted must be running. That is, the `-conntype none` option of wsadmin cannot be used, because a server connection is required.

You can use the `deleteAuditLog.jacl` script to delete audit log entries from the database.

1. Change to the Business Process Choreographer utilities directory where the scripts are located.

   On Windows systems, enter:

   ```
   cd install_root\ProcessChoreographer\util
   ```

   On UNIX and Linux systems, enter:

   ```
   cd install_root/ProcessChoreographer/util
   ```

2. Delete the entries in the audit log table.

   On Windows systems, enter one or more of the following commands:

   ```
   install_root\bin\wsadmin -f deleteAuditLog.jacl
                            -server serverName
                            [-profileName profileName]
                            [options]

   install_root\bin\wsadmin -f deleteAuditLog.jacl
                            -node nodeName
                            -server serverName
                            [-profileName profileName]
                            [options]

   install_root\bin\wsadmin -f deleteAuditLog.jacl
                            -cluster clusterName
                            [-profileName profileName]
                            [options]
   ```

   On UNIX and Linux systems, enter one or more of the following commands:

   ```
   install_root/bin/wsadmin.sh -f deleteAuditLog.jacl
                            -server serverName
                            [-profileName profileName]
                            [options]

   install_root/bin/wsadmin.sh -f deleteAuditLog.jacl
                            -node nodeName
                            -server serverName
                            [-profileName profileName]
                            [options]

   install_root/bin/wsadmin.sh -f deleteAuditLog.jacl
                            -cluster clusterName
                            [-profileName profileName]
                            [options]
   ```

   Where:

cluster *clusterName*
The name of the cluster. Required if the business process container is configured for a WebSphere cluster.

node *nodeName*
Optional when specifying the server name. This name identifies the node. The default is the local node.

server *serverName*
The name of the server. Required if the cluster name is not specified.

profileName *profileName*
The name of a user-defined profile. Specify this option if you are not working with the default profile.

The available options are:

-all
Deletes all the audit log entries in the database. The deletion is done in multiple transactions. Each transaction deletes the number of entries specified in the slice parameter, or the default number.

-*slice*
Used with the *-all* parameter to specify the number of entries included in each transaction. The value depends on the available log size for your database system. Higher values require fewer transactions but you might exceed the database log space. Lower values might cause the script to take longer to complete the deletion. The default size for the *-slice* parameter is 250.

-*time*
Deletes all the audit log entries that are older than the time you specify for this parameter. The time used is coordinated universal time (UTC). Its format must be: `YYYY-MM-DD['T'HH:MM:SS]`. If you specify only the year, month, and day, the hour, minutes, and seconds are set to `00:00:00`.

The *-time* and *-processtime* options are mutually exclusive.

-*processtime*
Deletes all the audit log entries that belong to a process that finished before the time you specify for this parameter. Use the same time format as for the *-time* parameter.

The *-time* and *-processtime* options are mutually exclusive.

Removing unused staff queries, using administrative commands

Use the administrative commands to remove unused staff queries from the database tables.

Before you begin this procedure, the following conditions must be met:

- The user ID that you are using must have administrative rights.
- The application server on which unused staff queries are to be deleted must be running. That is, the `-conntype none` option of `wsadmin` cannot be used, because a server connection is required.

Business Process Choreographer maintains lists of user names in the run-time database for staff expressions that have been evaluated. Although these processes instances are finished, the lists of user names are maintained in the database until the corresponding business process application is uninstalled.

If the size of the database is affecting performance, you can remove the unused staff lists that are cached in the database tables.

1. Change to the Business Process Choreographer utilities directory where the scripts are located.

   On Windows systems, enter:

   ```
   cd install_root\ProcessChoreographer\util
   ```

   On UNIX and Linux systems, enter:

   ```
   cd install_root/ProcessChoreographer/util
   ```

2. Remove the unused staff lists.

   On Windows systems, enter one of the following commands:

   ```
   install_root\bin\wsadmin -f cleanupUnusedStaffQueryInstances.jacl
                            -server serverName
                            [-profileName profileName]


   install_root\bin\wsadmin -f cleanupUnusedStaffQueryInstances.jacl
                            -node nodeName
                            -server serverName
                            [-profileName profileName]

   install_root\bin\wsadmin -f cleanupUnusedStaffQueryInstances.jacl
                            -cluster clusterName
                            [-profileName profileName]
   ```

   On UNIX and Linux systems, enter one of the following commands:

   ```
   install_root/bin/wsadmin.sh -f cleanupUnusedStaffQueryInstances.jacl
                            -server serverName
                            [-profileName profileName]

   install_root/bin/wsadmin.sh -f cleanupUnusedStaffQueryInstances.jacl
                            -node nodeName
                            -server serverName
                            [-profileName profileName]

   install_root/bin/wsadmin.sh -f cleanupUnusedStaffQueryInstances.jacl
                            -cluster clusterName
                            [-profileName profileName]
   ```

   Where:

   cluster *clusterName*
   : The name of the cluster. Required if the business process container is configured for a WebSphere cluster.

   node *nodeName*
   : Optional when specifying the server name. This name identifies the node. The default is the local node.

   server *serverName*
   : The name of the server. Required if the cluster name is not specified.

   profileName *profileName*
   : The name of a user-defined profile. Specify this option if you are not working with the default profile.

The number of entries deleted from the database is displayed.

Deleting process templates and task templates that are no longer valid

Use the following scripts to delete, from the database, process templates and task templates that are no longer valid.

Before you begin this procedure, the following conditions must be met:

- The user ID that you are using must have administrative rights.
- The application server on which templates to be deleted must be running. That is, the `-conntype none` option of `wsadmin` cannot be used, because a server connection is required.

Use the methods described here to remove, from the database, templates and all objects that belong to them if no corresponding valid application in the WebSphere configuration repository contains them. This situation can occur if an application installation was canceled or not stored to the Configuration Repository by the user. These templates usually have no impact. They are not shown in Business Process Choreographer Explorer.

There are rare situations in which these templates cannot be filtered. They must then be removed from the database with the following scripts.

You cannot use the scripts to remove templates of valid applications from the database. This condition is checked and a ConfigurationError exception is thrown if the corresponding application is valid.

- Change to the Business Process Choreographer utilities directory where the scripts are located.

  On Windows systems, enter:

  ```
  cd install_root\ProcessChoreographer\util
  ```

  On UNIX and Linux systems, enter:

  ```
  cd install_root/ProcessChoreographer/util
  ```

- Delete, from the database, business process templates or human task templates that are no longer valid.

  To delete, on Windows systems, a business process template that is no longer valid, enter the following command:

  ```
  install_root\bin\wsadmin -f deleteInvalidProcessTemplate.jacl
                           -server serverName
                           -template templateName
                           -validFrom validFromString
                           [-profileName profileName]

  install_root\bin\wsadmin -f deleteInvalidProcessTemplate.jacl
                           -server serverName
                           -node nodeName
                           -template templateName
                           -validFrom validFromString
                           [-profileName profileName]

  install_root\bin\wsadmin -f deleteInvalidProcessTemplate.jacl
                           -cluster clusterName
                           -template templateName
                           -validFrom validFromString
                           [-profileName profileName]
  ```

  To delete, on UNIX and Linux systems, a business process template that is no longer valid, enter the following command:

  ```
  install_root/bin/wsadmin.sh -f deleteInvalidProcessTemplate.jacl
                           -server serverName
                           -template templateName
                           -validFrom validFromString
                           [-profileName profileName]

  install_root/bin/wsadmin.sh -f deleteInvalidProcessTemplate.jacl
                           -server serverName
  ```

```
                         -node nodeName
                         -template templateName
                         -validFrom validFromString
                         [-profileName profileName]

install_root/bin/wsadmin.sh -f deleteInvalidProcessTemplate.jacl
                         -cluster clusterName
                         -template templateName
                         -validFrom validFromString
                         [-profileName profileName]
```

To delete, on Windows systems, a human task template that is no longer valid, enter the following command:

```
install_root\bin\wsadmin -f deleteInvalidTaskTemplate.jacl
                         -server serverName
                         -template templateName
                         -validFrom validFromString
                         -nameSpace nameSpace
                         [-profileName profileName]

install_root\bin\wsadmin -f deleteInvalidTaskTemplate.jacl
                         -server serverName
                         -node nodeName
                         -template templateName
                         -validFrom validFromString
                         -nameSpace nameSpace
                         [-profileName profileName]

install_root\bin\wsadmin -f deleteInvalidTaskTemplate.jacl
                         -cluster clusterName
                         -template templateName
                         -validFrom validFromString
                         -nameSpace nameSpace
                         [-profileName profileName]
```

To delete, on UNIX and Linux systems, a human task template that is no longer valid, enter the following command:

```
install_root/bin/wsadmin.sh -f deleteInvalidTaskTemplate.jacl
                         -server serverName
                         -template templateName
                         -validFrom validFromString
                         -nameSpace nameSpace
                         [-profileName profileName]

install_root/bin/wsadmin.sh -f deleteInvalidTaskTemplate.jacl
                         -server serverName
                         -node nodeName
                         -template templateName
                         -validFrom validFromString
                         -nameSpace nameSpace
                         [-profileName profileName]

install_root/bin/wsadmin.sh -f deleteInvalidTaskTemplate.jacl
                         -cluster clusterName
                         -template templateName
                         -validFrom validFromString
                         -nameSpace nameSpace
                         [-profileName profileName]
```

Where:

cluster *clusterName*
: The name of the cluster. Required if the business process container is configured for a WebSphere cluster. You can specify the cluster name or the server name and node name.

node *nodeName*
: Optional when specifying the server name. This name identifies the node. The default is the local node. You can specify the server name and node name or the cluster name.

server *serverName*
: The name of the server. Required if the cluster name is not specified. You can specify the server name and node name or the cluster name.

template *templateName*
: The name of the process template or task template to be deleted.

validFrom *validFromString*
: The date from which the template is valid.

nameSpace *nameSpace*
: The target namespace of the task template.

profileName *profileName*
: The name of a user-defined profile. Specify this option if you are not working with the default profile.

The process templates and task templates that are no longer valid are deleted.

Administering applications and application services

This section describes how to use the administrative interfaces to administer WebSphere Process Server applications and application services, including business processes and tasks, business rules, and schedules.

Administering business processes and human tasks

Business processes and human tasks are deployed and installed as part of an enterprise application. You can use the administrative console or the administrative commands to administer process templates and task templates, and Business Process Choreographer Explorer to administer process instances and task instances.

Administering process templates and process instances

Use the administrative console or the administrative commands to administer process templates. Use Business Process Choreographer Explorer to administer process instances.

Process templates define business processes within an enterprise application. When an enterprise application that contains process templates is installed, deployed, and started, the process templates are put into the start state. You can use the administrative console or the administrative commands to stop and start process templates.

Process templates are shown in Business Process Choreographer Explorer. A process instance can be a long-running process or a microflow. Use Business Process Choreographer Explorer to display information about process templates and process instances, or act on process instances. These actions can be, for example, starting process instances; and for long-running processes other process life cycle actions, such as suspending, resuming, or terminating process instances; or repairing activities.

Business process administration—frequently asked questions:

Answers to a set of frequently asked questions about administering business processes.

- "What happens if a process template is in the started state, but the application it belongs to is in the stopped state?"
- "What are the semantics of the valid-from attribute?"
- "How do I stop new process instances being created?"
- "What happens to running instances when a newer process template becomes valid?" on page 44
- "What happens to a running instance if the template it was created from is stopped?" on page 44
- "How can I tell if any process instances are still running?" on page 44
- "Why can't I stop a business process application if it has any process instances?" on page 44

What happens if a process template is in the started state, but the application it belongs to is in the stopped state?

If a currently valid process template is in the started state, but the application is in the stopped state, no new process instances are created from the template. Existing process instances cannot be navigated while the application is in the stopped state.

What are the semantics of the valid-from attribute?

When a new process instance is required, there could be many process templates in the database that have the same process template name. The following rules determine which version of the process is used to create new process instances:

- The process template states 'started' and 'stopped' control whether new instances can be created from the template. Process instances are never created from a process template that is in the stopped state.
- Of the process templates that are in the started state and have a valid-from value in the past, the process template with the most recent valid-from date is used to create the process instance. This is the only template that is considered to be valid now.
- If the template that is currently valid is deleted, a process template that was no longer valid can become valid if it now has the most recent valid-from value.
- If none of the process template have a valid-from value that lies in the past, no instance is created, and an error is generated.

These semantics allow you to define when changes to business processes come into effect, for example, tax changes that come into effect at midnight in a new year. The switchover to the new process templates is automatic, and requires no operator actions.

How do I stop new process instances being created?

Using the administrative console, select a process template, and click **Stop**. This action put the process template into the stopped state, and no more instances are created from the template. After the template stops, any attempts to create a process instance from the template result in an `EngineProcessModelStoppedException` error.

What happens to running instances when a newer process template becomes valid?

If a process template is no longer valid, this fact has no effect on running instances that were instantiated from the template. Existing process instances continue to run to completion. Old and new instances are run in parallel until all of the old instances have finished, or until they have been terminated.

What happens to a running instance if the template it was created from is stopped?

Changing the state of a process template to ’stopped’ only stops new instances being created. Existing process instances continue running until completion in an orderly way.

How can I tell if any process instances are still running?

Log on to the Business Process Choreographer Explorer as a process administrator, and go to the Process Instances Administered By Me page, this displays any running process instances. If necessary, you can terminate and delete these process instances.

Why can’t I stop a business process application if it has any process instances?

For a process instance to run, its corresponding application must also be running. If the application is stopped, the navigation of the process instance cannot continue. For this reason, you can only stop a business process application if it has no process instances.

Stopping and starting process templates with the administrative console:

You can use the administrative console to start and stop each installed process template individually.

Verify that you are logged with a user ID that has either business process administrator or process administrator authorization. The server on which the application is installed must be running.

The following steps describe how to use the administrative console to administer process templates. You must stop a process template, for example, before you can uninstall the business process application to which it belongs.

1. Select the application that you want to manage.

   In the navigation pane of the administrative console, click **Applications → Enterprise Applications**, and then the application that you want to manage.
2. Select an EJB module.

   Under Related Items, click **EJB Modules** and then an EJB module.
3. Select the process template that you want to manage.

   Under Additional Properties, click **Business Processes** and then a process template.
4. Stop the process template.

   Existing instances of the process templates continue to run until they end normally. However, you cannot create process instances from a stopped template.

5. Start the process template that is in the stopped state.

Stopping and starting process templates with administrative commands:

Administrative commands provide an alternative to the administrative console for stopping and starting process templates.

Verify that you have either business process administrator or process administrator authorization.

The following steps describe how to use the administrative commands to administer process templates. You must stop a business template, for example, before you can uninstall the business process application to which it belongs.

1. Change to the Business Process Choreographer samples directory.

   On Windows systems, enter:

   ```
   cd install_root\ProcessChoreographer\sample
   ```

   On UNIX and Linux systems, enter:

   ```
   cd install_root/ProcessChoreographer/sample
   ```

2. Stop the process template.

   On Windows systems, enter:

   ```
   install_root\bin\wsadmin -f bpcTemplates.jacl
                                -stop application_name
   ```

   On UNIX and Linux systems, enter:

   ```
   install_root/bin/wsadmin -f bpcTemplates.jacl
                                -stop application_name
   ```

   Where *application_name* is the name of the application to which the template belongs.

   Existing instances of the process templates continue to run until they end normally. When the application stops, you cannot create process instances from the stopped templates.

3. Start the process template.

   On Windows systems, enter:

   ```
   install_root\bin\wsadmin -f bpcTemplates.jacl
                                -start application_name
   ```

   On UNIX and Linux systems, enter:

   ```
   install_root/bin/wsadmin -f bpcTemplates.jacl
                                -start application_name
   ```

   The process template starts. You can use Business Process Choreographer Explorer to start process instances from the process template.

Starting a new process instance:

You can start a new process instance from any of the process templates that you are authorized to use.

All of the installed process templates are shown in the list of process templates in Business Process Explorer. To start a new process instance, complete the following steps.

1. Display the process templates that you are authorized to use.

   Click **My Process Templates** under Process Templates in the navigation pane.

2. Select a process template from the list and click **Start Instance**.

This action displays the Process Input Message page. Here you can provide the input data that is needed to start an instance of the business process from this page.

If the process has more than one operation, this action displays a page that contains all of the available operations. Select the operation that is to start the process instance.

3. Provide the input data to start the process instance.

   If the process is a long-running process, you can type in a process instance name. If you do not specify a name, a system-generated name is assigned to the new process instance.

   Complete the input for the process input message.
4. To start the process, click **Submit**.

The process instance is started. If the business process contains an activity that requires human interaction, tasks are generated for all of the potential owners. If you are one of these potential owners, this task appears in the list on the My Tasks page.

Depending on the type of process, a process output message is displayed immediately after the process finishes. Not all processes have output messages, for example, if the process implements a one-way operation, an output message is not displayed.

Suspending and resuming process instances:

You can suspend a running process instance and then resume it again later.

To suspend and resume process instances, you must have process administrator authorization.

To suspend a process instance, the process instance must be in either the running or failing state. To resume a process, the process instance must be in the suspended state.

You can suspend a long-running, top-level process instance while it is running. You might want to do this, for example, so that you can configure access to a back-end system that is used later in the process. When the prerequisites for the process are met, you can then resume running the process instance.

To suspend or resume a process instance, complete the following steps in Business Process Choreographer Explorer:

1. Display a list of process instances.

   For example, click **Administered By Me** under Process Instances in the navigation pane.
2. Suspend the process.

   Select a process instance and click **Suspend**. The process instance and its subprocess are put into the suspended state and the process instance stops running. However, you can still complete any active activities and tasks that belong to the process instance.
3. Resume the process instance.

   Select a process instance in the suspended state and click **Resume**. The process instance and its subprocesses resume.

Administering compensation for process instances:

If compensation fails for a process instance, there are a number of administrative actions you can take.

For process instances to be compensated, the compensation service must be started in the administrative console.

When a process instance runs, it can encounter problems. For these situations, compensation might have been defined for the process template. Compensation allows you to undo previous completed steps, for example to reset data and states so that you can recover from these problems.

However, the compensation processing might also fail. When a compensation for a microflow instance fails, the process administrator must intervene to resolve the problems.

In Business Process Choreographer Explorer, complete the following steps to administer failed compensation actions.

1. Display a list of the compensation actions that failed.

   Click **Failed Compensations** under Process Instances in the navigation pane.

   The Failed Compensations page is displayed. This page contains information as to why the named compensation action failed. This information can help you to decide what actions to take to correct the failed compensation.
2. Select an activity and then click one of the available actions.

   The following administrative actions are available:

   Skip Skips the current compensating action and continue with compensating the process instance. This action might result in a non-compensated activity.

   Retry If you have taken action to correct the failed compensation action, click **Retry** to try the compensation action again.

   Stop Stops the compensation processing.

Compensation in business processes:

Compensation is the means by which operations in a process that have successfully completed can be undone.

Compensation processing starts because an error occurs in a running process instance for which compensation is defined in the process model. Compensation reverses the effects of operations that were committed up to when the error occurred to get back to a consistent state. To take advantage of compensation, you must define compensation for a business process and its activities in the process model.

You can define compensation for long-running processes and for microflows in your process model.

Compensation for long-running processes

Compensation for long-running processes is also known as *business-level compensation*. This type of compensation is defined on the scope level. This means that either part of the process or the entire process can be compensated.

Compensation is triggered by fault handlers or the compensation handler of a scope or a process; compensation is another navigation path of the process.

A long-running process automatically compensates child processes that have successfully completed when the enclosing parent scope is compensated. Within a process, only invoke and scope activities that complete successfully are compensated.

Compensation for microflows

Compensation for microflows is also known as *technical compensation*. This type of compensation is triggered when the work unit (the transaction or the activity session) that contains the microflow is rolled back. Therefore, undo actions are typically specified for activities that cannot be reversed by rolling back the unit of work. When a process instance runs, undo actions for compensable activities are registered with the enclosing unit of work. Depending on the outcome of this unit of work (rollback or commit), compensation starts.

If the microflow is a child of a compensable, long-running process, the undo actions of the microflow are made available to the parent process when the microflow completes. It can, therefore, potentially participate in the compensation of the parent process. For these types of microflows, it is a good practice to specify undo actions for all of the activities in the process when you define your process model.

If an error occurs during compensation processing, the compensation action requires manual resolution to overcome the error. You can use Business Process Choreographer Explorer to repair these compensation actions.

Terminating process instances:

To terminate a process instance, you must have process administrator authorization.

You might want to terminate a process instance, for example, if the work or documents it represents are no longer needed, if no one is available to complete the process instance, if you have encountered problems with the process template and it needs to be redesigned, and so on.

If compensation is defined for the business process model, you can choose to terminate the process instance with compensation.

In Business Process Choreographer Explorer, complete the following steps to terminate a process instance.

1. Search for the process instance.

   Click **Search** in the menu bar. Enter your search criteria on the Process Instances page. This action displays a list of process instances that meet your search criteria.
2. Select the process instance that you want to stop.
   - To terminate the process instance with compensation, click **Compensate**.

     This action terminates the process instance and starts compensation processing.
   - To terminate the process instance without compensation, click **Terminate**.

This action stops the process instance immediately without waiting for any outstanding activities or tasks. Process instances that are terminated are not compensated.

Deleting process instances:

Not all process instances are automatically deleted when they complete. You can explicitly delete process instances that have completed.

To delete a process instance, you must have process administrator authorization.

Completed processes instances are automatically deleted from the Business Process Choreographer database if the corresponding property is set for the process template in the process model.

You might want to keep process instances in your database, for example, to query data from process instances that are not written to the audit log, or if you want to defer the deletion of processes to off-peak times. However, old process instance data that is no longer needed can impact disk space and performance. Therefore, you should regularly delete process instance data that you no longer need or want to maintain. Make sure that you run this maintenance task at off-peak times.

In Business Process Choreographer Explorer, complete the following steps to delete a process instance.

1. Display the process instances that you administer.

   Click **Administered By Me** under Process Instances in the navigation pane.
2. Select the process instance that you want to delete and click **Delete**.

This action deletes the selected process instance from the database.

Administering task templates and task instances

Use the administrative console or the administrative commands to administer task templates. Use Business Process Choreographer Explorer to work with task instances.

You can use the administrative console or the administrative commands to start and stop task templates for stand-alone tasks, and Business Process Choreographer Explorer to work with activities in a business process that require human interaction and with task instances. If you are authorized as a potential owner, editor, or reader of a task, the tasks are added automatically to the list of tasks that are assigned to you.

Stopping and starting task templates with the administrative console:

Use the administrative console to start and stop task templates.

Verify that you have task administrator authorization.

Task templates define Service Component Architecture (SCA) services that are represented as stand-alone tasks within an enterprise application. When an enterprise application that contains task templates is installed, deployed, and started, the task templates are put into the start state.

1. Select the application that you want to manage.

   In the navigation pane of the administrative console, click **Applications** → **Enterprise Applications**, and then the application that you want to manage.

2. Select an EJB module.

   Under Related Items, click **EJB Modules** and then an EJB module.
3. Select the task template that you want to manage.

   Under Additional Properties, click **Human Tasks** and then a task template.
4. To stop the task template, click **Stop**.
5. To start the task template, click **Start**.

Stopping and starting task templates with the administrative commands:

Administrative commands provide an alternative to the administrative console for stopping and starting task templates.

Verify that you have task administrator authorization.

Task templates define Service Component Architecture (SCA) services that are represented as stand-alone tasks within an enterprise application. When an enterprise application that contains task templates is installed, deployed, and started, the task templates are put into the start state.

1. Change to the Business Process Choreographer samples directory.

   On Windows systems, enter:
   ```
   cd install_root\ProcessChoreographer\sample
   ```
   On UNIX and Linux systems, enter:
   ```
   cd install_root/ProcessChoreographer/sample
   ```
2. Stop the task template.

   On Windows systems, enter:
   ```
   install_root\bin\wsadmin -f bpcTemplates.jacl
                              -stop application_name
   ```
   On UNIX and Linux systems, enter:
   ```
   install_root/bin/wsadmin -f bpcTemplates.jacl
                              -stop application_name
   ```
   Where *application_name* is the name of the application to which the template belongs. Existing instances of the task template continue to run until they end normally.
3. Start the task template.

   On Windows systems, enter:
   ```
   install_root\bin\wsadmin -f bpcTemplates.jacl
                              -start application_name
   ```
   On UNIX and Linux systems, enter:
   ```
   install_root/bin/wsadmin -f bpcTemplates.jacl
                              -start application_name
   ```

   The task template starts. You can use Business Process Choreographer Explorer to work with task instances associated with the task template.

Working on your tasks:

To work on a task, you must claim the task and then perform the actions needed to complete it.

You can claim a task that is in the ready state if you are a potential owner or the administrator of that task. If you claim a task, you become the owner of that task and are responsible for completing it.

To claim and complete a task with Business Process Choreographer Explorer, complete the following steps.

1. Display the tasks that have been assigned to you.

   Click **Task Instances** → **My Tasks**.

   This action displays the My Tasks page, which lists the tasks that have been assigned to you.
2. Claim the task on which you want to work.

   Select the check box next to the task and click **Work on**.

   This action displays the Task Message page.
3. Provide the information to complete the task.

   If you need to interrupt your work, for example, because you need more information from a co-worker to complete the task, click **Save** to save the changes you made.
4. Click **Complete** to complete the task with the information that you provide.

The task that you completed is in the finished state. If you leave the task without completing it, the task remains in the claimed state.

Managing task assignments:

After a task has started, you might need to manage task assignments for the task.

A *work item* is the assignment of a task, to a person or a group of people for a particular reason. These tasks then appear on the list of tasks for that person in Business Process Choreographer Explorer. The reason for the task assignment is derived from the role that a person has for a task, for example, potential owner, editor, or administrator.

A task instance can have several work items associated with it because different people can have different roles. For example, John, Sarah, and Mike are all potential owners of a task instance and Anne is the administrator; work items are generated for all four people. John, Sarah, and Mike see only their own work items as tasks on their list of tasks. Because Anne is the administrator, she gets her own work item for the task and she can manage the work items generated for John, Sarah, and Mike.

Sometimes, you might need to change a task assignment after a task has been started, for example, to transfer a work item from the original owner to someone else. You might also need to create additional work items or delete work items that are not needed anymore.

You can use Business Process Choreographer Explorer to manage work items in the following ways:

- Transfer a work item
- Create a work item
- Delete work item

Transferring work items:

You can transfer work items for task instances from one owner to another owner.

To transfer a work item for a task instance, you must be the administrator of the task. Also, the task instance or the process instance must be in one of the following states:

Task instance
States: claimed, waiting, ready, stopped

Process instance
States: running, failing, terminating

You might want to transfer a work item, for example, if the work-item owner is on vacation and the work item must be completed before he returns.

In Business Process Choreographer Explorer, complete the following steps to transfer a work item.

1. Display the task instances that you administer.

   Click **Administered By Me** under Task Instances in the navigation pane.
2. Display the work items for a task instance.

   In the Task Instances Administered By Me page, select a task instance and click **Work Items**.
3. Transfer the work item.
   a. In the **New Owner** field, specify the user ID of the new work-item owner.
   b. Select one or more roles from the **Reason** list. These roles determine the actions that the assigned person can perform on the transferred work item.
   c. Click **Transfer**.

The transferred work item appears on the list of tasks belonging to the new work-item owner.

Creating work items:

To create a work item for a task instance, you must be the administrator of the task. Also, the task instance or the process instance must be in one of the following states:

Task instance
States: claimed, waiting, ready, stopped

Process instance
States: running, failing, terminating

You might want to create work items for new potential owners, for example, when none of the current potential owners can accept any additional work. You might also want to create work items if the query against the staff repository does not return any potential owners. This might happen, for example, in a long-running process if the organization has changed since the process started.

In Business Process Choreographer Explorer, complete the following steps to create a work item.

1. Display the task instances that you administer.

   Click **Administered By Me** under Task Instances in the navigation pane.
2. Display the work items for a task instance.

   In the Task Instances Administered By Me page, select a task instance and click **Work Items**.
3. Create the work items.

a. In the **New Owner** field, specify the user ID of the new work-item owner.
b. Select one or more roles from the **Reason** list.

   These roles determine the actions that the assigned person can perform on the new work item.
c. Click **Create**.

A work item is created for each role that you specify for the new work-item owner. The new task appears on the list of tasks assigned to this person.

Deleting work items:

To delete a work item for a task instance, you must be the administrator of the task. Also, the task instance or the process instance must be in one of the following states:

Activity instance
States: claimed, waiting, ready, stopped

Process instance
States: running, failing, terminating

You might want to delete work items, for example, if you created work items in error or if work items are generated for someone who no longer works for the company.

In Business Process Choreographer Explorer, complete the following steps to delete a work item.
1. Display the task instances that you administer.

   Click **Administered By Me** under Task Instances in the navigation pane.
2. Display the work items for a task instance.

   In the Task Instances Administered By Me page, select a task instance and click **Work Items**.
3. Delete the work items.

   Select a work item and click **Delete**.

The work items are deleted.

Viewing task escalations:

An escalation notifies the escalation receiver that a user might have problems completing their assigned task on time.

When a task becomes overdue, it might result in an escalation. An escalation can result in the following actions:
- A new task is created, for example, for a manager to take action to support the resolution of the problem
- An e-mail is sent to a designated person to inform them the escalated task.

To view escalations, click **My Escalations** under Task Instances.
- To view information about an escalation, click the escalation ID.
- To view information about an escalated task, click the task name.

Business rules

Use business rules to control the behavior of a business practice. (For more information about building and deploying business rules, see the WebSphere Integration Developer information center.)

What is a business rule?

A business rule is anything that imposes structure upon, or controls the behavior of a business practice. A rule can enforce business policy, establish common guidelines within an organization, or control access in a business environment. (For more information about building and deploying business rules, see the WebSphere Integration Developer information center.)

When to use a business rule

Use business rules to officiate over frequently changing business practices that can come from within a business or mandated from outside a business, such as regulatory agencies. Some typical uses for business rules are:

- Determining current interest rates
- Calculating discounts for products
- Calculating sales tax
- Determining special groups such as senior citizens or preferred customers

(For more information about building and deploying business rules, see the WebSphere Integration Developer information center.)

How to use business rules

Create and modify business rules using an eclipse-based WebSphere Integration Developer tool. (For more information about building and deploying business rules, see the WebSphere Integration Developer information center.) Manage and modify business rule values using the Web-based business rules manager tool.

Installing the business rules dynamic repository for a standalone server

The server installs the dynamic repository in a standalone server configuration during creation or augmentation of the standalone server profile.

When setting up a profile using the Profile Wizard, the following common database options are supported for business rule group:

- Cloudscape
- DB2 Universal Database
- DB2 Universal OS/390 V7.1
- DB2 Universal OS/390 V8.1
- DB2 CLI

If you select a database other than one of these, the business rule group component will use Cloudscape.

If you manually install the business rules dynamic repository, use the provided configureArtifactRepository command for DB2.

```
wsadmin -f configureArtifactRepository.jaclproperties file -profile default wsadmin
-f configureArtifactRepository.jacl -profile default
```

where:

properties file is the file containing the database properties

The database properties you specify in the properties file are:

cellName
: The name of the cell.

  Example: cellName=T40Cell04

nodeName
: The name of the node

  Example: nodeName=T40CellManager04

profilePath
: The profile root name

  Example: profilePath=*install_root*\profiles*profile_name*

profileName
: the profile name

  Example: profileName=Dmgr04

WBI_HOME
: *install_root* = the installation directory of WebSphere Process Server

  Example: C:\Program Files\IBM\WebSphere\ProcServer)

WAS_HOME
: *install_root* = the installation directory of WebSphere Application Server or WebSphere Application Server Network Deployment

  Example: C:\Program Files\IBM\WebSphere\ProcServer

dbName
: database name

  Example: dbName=WPRCSDB

dbType
: DB2_Universal, DB2UDBOS390_V7_1, DB2UDBOS390_V8_1, DB2_CLI or CLOUDSCAPE

If Cloudscape is specified for the `dbtype`, the following properties must also be provided:

dbServerPort
: The Cloudscape server port

  Example: dbServerPort=50000

dbServerName
: The Cloudscape server name

  Example: dbServerName=dbserver

dbHostName
: The Cloudscape host name

  Example: dbHostName=dbserver

dbPassword
: The Cloudscape server password

Example: db2passwd

dbClassPath
The path to the Cloudscape JDBC driver

Example: dbClassPath=C:\IBM\SQLLIB\java

Note: The Cloudscape JDBC driver shipped with WebSphere Process Server is located in *WPS_INSTALL_ROOT*\Cloudscape\lib. Where *WPS_INSTALL_ROOT* is the directory where you installed WebSphere Process.

The following is an example of the configureArtifactRepository command.

```
wsadmin -f configureArtifactRepository.jacl
C:\Program Files\IBM\WebSphere\ProcServer\properties\artifactRepository.properties
   -profile default
```

Installing the business rules dynamic repository for network deployment

Before installing any applications containing business rules, you must install the dynamic repository for business rules.

The dynamic repository supports a centralized configuration that allows you to configure all servers to use the same repository thus allowing all applications to use the same data. This is important for those users who will be updating this data dynamically using the business rule manager tool. The central repository allows:

- changes to be made once
- changes to take effect across all server installations

In the network deployment environment, the dynamic repository only supports DB2 databases (DB2 Universal Database, DB2 Universal OS/390 V7.1, DB2 Universal OS/390 V8.1, and DB2 CLI). If you configure the WPRCSDB database to be a non-DB2 database during profile creation or augmentation, the server prompts you with a message stating that you must configure the dynamic repository manually using the configureArtifactRepository command.

Here is the syntax of the configureArtifactRepository command.

```
wsadmin -f configureArtifactRepository.jacl -profile {default | dmgr}
```

where:

```
properties file = a file containing the database properties
-profile {default | dmgr} = the profile type
```

The database properties you specify in the properties file are:

cellName
The name of the cell

Example: cellName=T40Cell04

nodeName
The name of the node

Example: nodeName=T40CellManager04

profilePath
The profile root name

Example: profilePath=*install_root*\profiles*profile_name*

profileName
The profile name

Example: profileName=Dmgr04

WBI_HOME
install_root = the installation directory of WebSphere Process Server

Example: C:\Program Files\IBM\WebSphere\ProcServer

WAS_HOME
install_root = the installation directory of WebSphere Application Server Network Deployment

Example: C:\Program Files\IBM\WebsSphere\ProcServer

dbName
database name

Example: dbName=dynamicDB

dbType
DB2_Universal, DB2UDBOS390_V7_1, DB2UDBOS390_V8_1, or DB2_CLI

Here is an example of the configureArtifactRepository command.

```
wsadmin -f configureArtifactRepository.jacl C:\Program Files\IBM\
WebSphere\ProcServer\properties\artifactRepository.properties -profile dmgr
```

Accessing business rule components

Displaying business rule components is the first step in administering a business rule group. From the display you can export any or all of the business rule groups or display the tables that comprise the business rule groups.

You must be at the administrative console for WebSphere Process Server to perform this task.

Perform this task to determine what business rule groups exist in your server.

1. From the administrative console, select **Servers > Application Servers**.
2. Click *servername* to select the server from the server list that displays business rules.
3. Click **Business rules** under Business Integration.

The console displays a list of all the business rule components defined with a description of each group.

Exporting business rules using the administrative console:

Exporting business rule components creates a file that you import into your development environment, thereby keeping the development artifacts synchronized with the actual production system artifacts.

Before starting this task, you should already have displayed your business rule components as described in "Accessing business rule components."

Export business rule components when you have made changes to the business rule tables to synchronize your development environment with your production environment. This task begins with the business rule component display.

1. Choose the business rule groups to export.

Click the check boxes next to the business rule groups and then click **Export**. The browser displays a list of HTML links to the business rule groups you chose. (This is the Business rules to export panel.) Each business rule group has a file extension of `.zip`.

2. Download the files.

   Click on each filename and the system prompts you to save the file. When prompted, click **OK** to place the file in your file system.

   Note: If you choose to, you can rename the file as you download it.

3. Return to the business rules display panel.

   Click **Back** to return to the list of business rule groups.

The system saves file where you specified. You can then copy it to your test system.

You must import this file into your WebSphere Integration Developer environment. See the information center for WebSphere Integration Developer for more information.

Exporting business rules using the command line:

You can also export business rule components using the exportBusinessRuleArtifacts command.

Purpose

Use the exportBusinessRuleArtifacts command to export business rule components from the command line.

Syntax

```
wsadmin -f exportBusinessRuleArtifacts.jacl target namespace business rule
group component name<user> -zipf
```

where the arguments are:

`target namespace` = the name of the target namespace of the business rule group component to export

`business rule group component name` = the name of the business rule group component to export

user = user

-*`zipf`* (optional) = the name of a zip file to export the business rule group component ; if not specified, it defaults to *`business rule group component name`*`.zip`

Example

```
 wsadmin   -f exportBusinessRuleArtifacts.jacl
http://test.oo.brules OnlineOrder admin -zipf c:/artifacts/onlineorder.zip
```

The business rules manager

The business rules manager is a Web-based tool that assists the business analyst in modifying business rule values. This tool is an option of WebSphere Process Server that you install after the initial installation of the server. The business rules manager uses a Web-based interface for you to browse and edit business rules.

How business rules manager works:

The business rules manager component is the main WebSphere Process Server tool that a business analyst uses to manage the rules that run their business.

Use the business rules manager to perform the following tasks
- Open a copy of a business rule from the repository
- Browse and edit a business rule
- Publish a business rule to the repository

Figure 1 shows how the business rule manager calls and publishes rules.

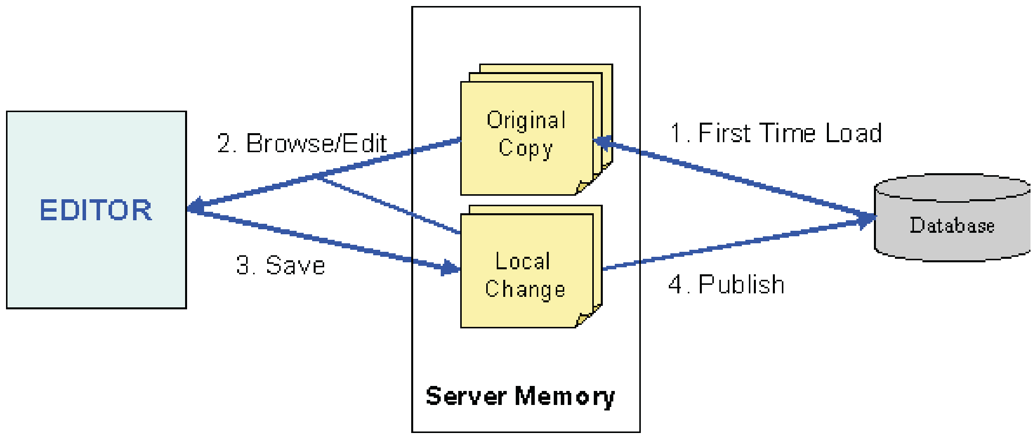


Figure 1. Business rules manager sequence of events

After you log on to the business rules manager, the following events occur when you modify a business rule.
1. When you select a business rule, the business rules manager accesses a rule from the database and stores it in the server memory as an original copy.
2. The rule is available for browsing or editing.
3. You save the business rule as a copy in the local server memory.
4. You publish local copy back to the database.

Business rule templates:

Templates provide the mechanism for Web-based rule authoring in WebSphere Process Server. Business rule templates provide a constrained way of allowing the business analyst to author rules in real-time on an application server. Create templates in WebSphere Integration Developer to provide the means for a business analyst to edit a business rule using the business rule manager. (For more information about building and deploying business rules, see the WebSphere Integration Developer information center.) A template is created for a table cell in a decision table or a rule in a rule set. You must place all rules, condition cases or actions into a template.

Use a template to:
- Modify the values (within the constraints of the rule) of a business rule.

- Create a rule within a decision table or rule set.

Installing the business rules manager:

The business rules manager is installed as a plug-in for WebSphere Process Server. You can either install it using the administrative console or use a `.jacl` script for Windows or Linux.

To install business rules manager manually from the command prompt, you must execute an installation script in *WAS_HOME*/bin after starting process server.

For Windows and Unix/Linux use these command lines.

```
  - run shell/command prompt and change directory to WAS_HOME/bin

  - run the following command with WebSphere Process Server started
     Win32 : wsadmin.bat -f installBRManager.jacl -s servername
                         -n nodename -c cellname -r rootname
                         -a applicationname
     Linux : /wsadmin.sh -f ./installBRManager.jacl -s servername
                         -n <nodename> -c <cellname> -r rootname
                         -a <applicationname
```

where:

servername
: The name of the application server.

  This pair of arguments is required in the Network Deployment configuration.

nodename
: The name of the installation node.

  This pair of arguments is required in the Network Deployment configuration.

cellname
: The name of the installation cell.

  This pair of arguments is optional.

rootname
: The name of the application root.

  This pair of arguments is optional. If missing, the default value of *rootname* is "br/webclient."

applicationname
: The name of the application.

  If missing, the default value of *applicationname* is "BusinessRulesManager".

Note: If WebSphere Process Server is configured in a single-server environment , all of these pairs of arguments are optional. If WebSphere Process Server is configured for a Network Deployment environment, the only two argument pairs required are:

- -s *servername*
- -n *nodename*

the other argument pairs are optional.

Note: Missing arguments will take the following default values:

- "server1" = *servername*
- "br/webclient" = *rootname*
- "BusinessRulesManager" = *applicationname*

Configuring a server:

You must configure the server that is using the business rules manager.

Configure server by:

These are the areas that are addressed when configuring your server.

Note: The next three items need to be addressed only if you are installing business rules manager from the administrative console.

- Preparing userids

  You must enable global security when creating userids. Create userids and map them to the role BusinessRuleUser or business analyst. You must assign a role to each userid.
- Setting the session tracking mechanism

  Use cookies to track sessions.
- Setting other parameters

  Minimally, set an appropriate session timeout value.
- Enabling security on your sever (optional)

  If you have different roles or userids, you must enable global security when configuring your server.

Configuring a client:

The server configures a client automatically while installing business rule manager.

These two areas are configured in client workstations.

If you encounter problems using business rules manager, examine the following:

- Scripting is enabled in the Web Browser

  Business rules manager requires scripting to function.
- Cookies are enabled

  When necessary, cookies are used to track the session when you are using business rules manager. Therefore, enable cookies on your browser when tracking sessions. Contact your system administrator if you enable cookies.

Invoking the business rule manager:

Make sure that the server and client are configured correctly for business rules manager.

Contact your system administrator for the URL to start business rules manager.

An example is:`https://hostname:9443/br/webclient` where `hostname:` is the name of the server for example, `server1`.

Accessing business rules manager:

Business rules manager is accessed using a Web browser.

Make sure that both the server and client are configured correctly. See "Installing the business rules manager" on page 60 for more information.

The default URLs for accessing the business rules manager are:
- https://hostname.9443/br/webclient (if security is enabled)
- https://hostname.9080/br/webclient (if security is not enabled)

where `hostname` is the name of the host.

Note: URLs may vary according to the environment.

Note: The Login page opens only when you have enabled global security on the server. See "Configuring a server" on page 61 for more information. If the global security is not enabled, the Rule books page opens when business rules manager is accessed.

If global security is enabled, follow these steps to login.
1. At the Login page, type your **User ID**.
2. Type your **Password**.
3. Click **Login**. The business rules manager Initial page opens.

The Initial page opens with the existing rule books listed in the Navigation Area. You can now make changes to any business rule listed.

Business rules manager page layout: The page layout on the business rules manager is described below.

Toolbar

The toolbar contents are:

Welcome
Shows the user's name.

User identification
Provides the current user's name preceded with **Welcome User Name**.

Logout
Opens the Login page if the global security is enabled.

Note: If you logout without publishing, all unpublished changes are lost.

Help
Provides access to information center for the business tools editor. (For more information, see the WebSphere Integration Developer information center.)

Left pane

The left pane of the page contains the navigation tree. This tree enables you to drill-down to the rule level you need. You can expand or collapse a rule book by clicking either the plus (+) or minus (-) beside the resource.

Note: This area is not displayed in any page that is in the edit mode.

Publish and Revert
Opens the Publish and Revert page where you can publish changes to the database or revert back to the original copy that was on the database.

Rule books
Opens the Rule books page.

Note: Listed beneath Rule books is a list of business rule groups. It is the top level of browsing.

Right pane

The right pane of the page is divided into top and bottom sections.

Top section

The top section contains the following elements.

Path information
Provides the name of the rule book and rule page shown in this example.

```
BusinessRuleGroup01 > Table1_operation1
```

Rule title
Includes the name of the resource and type of the business rule shown in this example.

```
Ruleset1 12 - Ruleset
```

Buttons
The buttons that appear on a page depend on the function of that particular page. This table shows a list of all the possible push buttons that can appear on a page.

Table 3. Button List

Button Name	Function
Back	Returns to the previous page
Edit	Enables editing of the Rule page, decision table, or rule set
Save	Saves the changes and returns to the previous page
Cancel	Discards and changes to the resource and returns to the previous page
Copy	Copies either a decision table, or rule set in order to create a new decision table, or rule set
Publish	Publishes the Rule Page to the server
Revert	Erases all changes and reverts back to the original copies that reside in the database

Message field
Shows either an action that has been taken to the rule or an error that has occurred. This is an example of a status message.

```
"calculateDiscount" has been temporarily saved.
```

```
You may publish the changes from the "Publish and Revert" page.
```

Bottom section

The bottom section provides the main viewing and editing area of the business rules manager. This section contains the following information.

Note: The information depends on depending on the level of the rule you are viewing.

Rule group
Lists all the rules contained within a rule book.

Rule page
Shows a rule and all the information concerning a particular business rule record in table form. Here is a list of the column headings for a Rule page.
- **Effective date and time**
- **Actions**
- **Description**
- **Rule sets or decision tables**

Publish and Revert
Is a link to the Publish page where you can either publish a rule to the database or revert a rule to the original state. See "How business rules manager works" on page 59 for more information.

Rule books page:

The Rule books page lists the rule groups created for a particular application.

The Navigation tree in the left pane shows the list of rule groups. You can expand the tree by clicking the plus **(+)** next to the rule books to show all rules. Select a rule book in the left pane navigation tree and all the children rules are listed in the right pane.

The content in the right pane is in a table format has the following column headings.

Business Rule Resources
Lists the name of the business rules, decision tables, and rule sets.

Description
Provides either a brief description or name of the rule, decision table, and rule set.

Action
Is initially empty but when the group is expanded an **Edit** button appears beside each rule.

Rule book page:

When a rule book is selected, the Rule book page opens and lists all the rules in a rule group.

The navigation tree in the left pane shows the list of rule rooks. You can expand the tree by clicking the plus **(+)** next to the rule book to show all rules. Select a rule book in the navigation tree and all the children rules are listed in the right pane.

The content in the right pane is in a table format has the following column headings.

Business Rule Resources
Lists the name of the business rules, decision tables, and rule sets.

Description
Provides either a brief description or name of the rule, decision table, and rule set.

Action
Is initially empty but when the group is expanded an **Edit** button appears beside each rule.

Rule page:

The Rule page lists a set of rules and business rule records (decision tables and rule sets) in table form that belong to each rule.

The content in the right pane contains the following elements.

Title Section

Rule Title
Shows the rule name with the following format, Rule Name-Rule Page. Here is an example of a rule title:

```
calculateDiscount-Rule Page
```

Function buttons
Provide specific functionality for that page. Here is a list of the buttons.

- **Back** - returns to the previous page
- **Edit** - opens the edit mode for the selected rule

Messages
Provides a message that signifies a particular error such as incorrect date or time or relating to the status of the rule such as the message shown here.

```
"calculateDiscount" has been temporarily saved.

You may publish the changes from the "Publish and Revert" page.
```

General Information section

This section contains the following fields:

Last Published
Shows the date that the rule was last published, the status.

Status Lists the status of the rule, whether it has been published or not.

Description
Provides a brief description of the rule.

Business Rule Selection Records section

Provides a list of effective business rules that are the building blocks of that rule and have the following information:

Start date
Provides the options of either a specific date or ″no start date.″

Note: The ″no start date″ signifies that the target rule logic is effective for any date before the end date.

End date
Provides the option of either a specific date or ″no end date.″

Note: The ″no end date″ signifies that the rule logic is effective for the start date and any date after it

Effective Business Rule
Provides a decision table or rule set for that rule logic record.

Default Business Rule
Provides the default values for the business rule if no other rule logic is applicable. It is selected when the date does not match any of the other selection records.

Note: The date used for the rule logic selection is either supplied by the application program or the current date of the server. In the latter case, the date is converted to **UTC** before its use and time zone of the server does not matter.

Decision table page:

The Decision table page is opened from the Rule page.

The content in the right pane of the decision table page contains the following elements in table format.

Title section

Table Title
Shows the rule name with the following format, `Decision Table Name-Decision Table (table)`. Here is an example of a table title:

```
Table2212-Decision Table
```

Function buttons
Provide specific functionality for that page. The buttons are:

- **Back** - returns to the previous page.
- **Edit** - opens the edit mode for the selected rule.
- **Copy** - creates a copy of an the rule in order to create a new rule.

  Note: You cannot create a new rule in the business rule manager. You must copy an existing rule and then modify the values in order to make a new rule. SeeCreating a decision table for more information.

General Information section

This section contains the following fields:

Last Published
Shows the date that the table was last published, the status.

Status Lists the status of the table, whether it has been published or not.

Description
Provides a brief description of the table.

Decision Table section

This section provides a list of rules in a table format that vary according to the rule.

Columns can have an **orientation** icon that switches orientation from horizontal to vertical. See "Decision tables" on page 70

Rule set page:

The Rule set page provides a list of rules for a business rule.

The content in the right pane of the Rule set page contains the following elements in table format.

Top section

This section provides the following fields and information:

Rule Title
Shows the rule name with the following format, "Ruleset Name-Ruleset". Here is an example of a rule set title:

```
Ruleset112-Ruleset
```

Function buttons
Provide the following functionality:

- **Back** - returns to the previous page.
- **Edit** - opens the edit mode for the selected rule.
- **Copy** - creates a copy of an the rule in order to create a new rule.

  Note: You cannot create a new rule in the business rules manager. You must copy an existing rule and then modify the values in order to make a new rule. SeeCreating a rule set for more information.

General Information section

This section contains the following fields:

Last Published
Shows the date that the rule set was last published, the status.

Status Lists the status of the rule set, whether it has been published or not.

Description
Provides a brief description of the rule set.

Rules section

Provides a list of rules that are the building blocks of that rule and have the following information.

Name Provides the name of the rule.

Rule Lists the variables, constraints, range, and enumeration that defines the rule.

Working with business rule records

Use business rules manager, to create, modify or delete business rule records.

Creating a business rule record:

You must create business rule records from existing records.

Make sure you are in the **Edit** mode for the rule you want to create.

To create a new business rule record, do the following:

1. Click the **Add Selection Record** button. A new business rule record appears at the bottom of the list with the **Start Date Time** field set a **Jan 1**. A message appears in the **Messages** field stating that the date is invalid.
2. Setting the Start Date/Time:
   a. Select the**Month**
   b. Select or type in the **Day**
   c. Type in the **Year**
   d. Type in a **time** (in 24-hour format)
3. Setting the End Date/Time.
   a. Select or type in the **Day**
   b. Type in the **Year**
   c. Type in a **time** (in 24-hour format)

   Note: Only one rule can be in effect at any one point in time. Rule dates cannot have date/time ranges that overlap.

   Note: Gaps in date/time ranges are allowed. If a default business rule is declared, it is used during the gap.
4. Select the **Effective Rule Logic**
5. Click the **Save** button

A message appears in the message field stating that the record has been temporarily saved and you can publish the changes in the Publish and Revert page. (See "Publishing business rules" on page 74 for more information.)

Related tasks

"Deleting a business rule, decision table or rule set record" on page 74

Modifying a business rule record:

Modify the date and time values of existing business rule selection records by clicking the **Edit** button for that rule.

Make sure you are in the **Edit** mode in the rule you want to modify by clicking the **Edit** button in the action column. To add, modify or delete a business rule record see "Decision tables" on page 70 or "Rule sets" on page 73 for more information.

To modify a business rule, do the following:

1. Edit the **Start Date/Time** of the selection record:
   a. Select the **Month**
   b. Select or type in the **Day**
   c. Type in the **Year**

   d. Type in a **time** (in 24-hour format)
2. Edit the**End Date/Time** of the selection record:
   a. Select or type in the **Day** for the *End Date/Time*
   b. Type in the **Year** for the *End Date/Time*
   c. Type in a **time** (in 24-hour format)

      Note: Only one rule can be in effect at any one point in time. Rule dates cannot have date/time ranges that overlap.

      Note: Gaps in date/time ranges are allowed. If a default business rule is declared, it is used during the gap.
3. Click the **Save** button.

   Note: If the **Time/Date** fields are invalid, the fields will turn **red** and a message appears in the message field that the time/dates are invalid.

The record is saved locally and is ready to be published to the database. (See "Publishing business rules" on page 74 for more information.)

See "Splitting dates in business rules" for more information on setting business rule dates.

Related tasks

"Deleting a business rule, decision table or rule set record" on page 74

Date/Time selections:

There are four Date/Time selections you can use.

Note: The continuous date selection is only available on the **End Date/Time** field. Using this selection automatically sets the end date to the earliest start date that is later than the selection record.

Specify Date/Time
This selection specifies a date manually.

Continuous
This selection uses an automatic date calculation that sets the end date to the earliest start date that is later than the selection record.

Note: The continuous selection is used when date ranges of two business rule selection records are contiguous. A continuous attribute is set to the end date to the first selection record. When this attribute is set, the start date of the second selection record is set to the end date of the first selection record so that you do not have to specify both dates.

No Start Date or No End Date
The selection does not set a starting or ending boundary, depending on which is selected

Close This selection closes the **End Date/Time** menu.

Splitting dates in business rules:

Splitting a date in a business rule provides a shortcut for modifying a business rule for another purpose.

Select the business rule to be modified and make sure you are in the **edit** mode.

To split a business rule record, do the following:

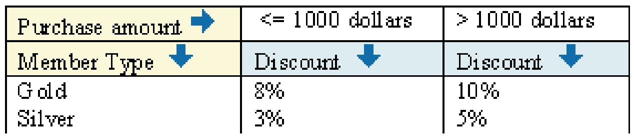
1. Click the **Split** button for the record to be modified. A new record is created with a start date of Jan 1. The fields will be in red and a message appears in the Message field that there is an`invalid date/time field`.
2. Select the **Start date/time** for the new record. The **End date/time** for the original record changes from continuous to the **start date** of the previous record and the **End date/time** of the new record changes to the **start date/time** of the next record.
3. Modify the date/times of the record that followed the original record.
4. Modify the **Effective Rule Logic** to fit the needs of the new rule.

The business rule date has been modified.

Decision tables

Modify the values in a decision table using the business rule manager.

A decision table is a business rule record in table format as shown in this figure.



Purchase amount ➡	<= 1000 dollars	> 1000 dollars
Member Type ⬇	Discount ⬇	Discount ⬇
Gold	8%	10%
Silver	3%	5%

Figure 2. Decision table

The business rules manager provides a Web interface that presents decision tables in tabular form with each variable in**bold** characters. The orientation of conditions and actions are shown by arrows like the ones shown in Figure 1. A decision table functions as a condition with a corresponding action. If a condition is met, then the corresponding action or actions are performed.

Creating a decision table record:

To create a new decision table record, you must first copy an existing decision table.

1. 1. Select a decision table and copy it.
   a. Click **Copy** in the selected business rule record.
   b. Click **OK** to copy the record. The **Edit** screen opens with a title, Edit Mode:Copy_of_TableName-Decision Table
   c. Click **OK** to copy the record.
2. Type in the **name** of the new business rule record in the **Title field**.
3. Type in a short **Description** of the new record.
4. Modify the **values** in each condition. **Note:** To display t the parameter settings for each value, place your cursor over a field. A rollover message appears showing the type of variable and its range.
5. Click on the **Up arrow** to place the rule in the correct sequence.
6. Click **Save**.

A message appears in the message field stating that the record has been temporarily saved and you can publish the changes in the Publish and Revert page. See"Publishing business rules" on page 74 for more information.

Related tasks

"Deleting a business rule, decision table or rule set record" on page 74

Modifying a decision table record:

Edit a decision table by directly typing in the values into the input fields or selecting a value form the field's list box. There is also a special **Edit** menu which appears by clicking on the **Page** icon.

Note: Reordering the columns or rows only effects the visual presentation of the table and has no effect on the order in which the conditions and actions are processed.

Menu		**Condition**	**Action**
Add below	Adds a new condition value below the cell (orientation is vertical)	Yes	
Add to the right	Adds a new condition value on the right of the cell (orientation is horizontal)	Yes	
Change template	Changes the template of a cell	Yes	Yes
Move up	Moves the condition value up one row (orientation is vertical)	Yes	
Move down	Moves the condition value down orientation is horizontal)	Yes	
Move left	Moves the condition value to the left (orientation is horizontal)	Yes	
Move right	Moves the condition value to the right (orientation is vertical)	Yes	
Delete	Deletes the condition value	Yes	
Close menu	Close the menu	Yes	yes

To modify the values of a decision table do the following:

1. From either the **Navigation** or **Content** sections, select the**Table** to be modified.
2. Click the **Edit** button.

3. Modify the appropriate values.

   Either directly type in the values using the field's list box or click the page icon beside the field.
4. Click the **Save** button.

The rule is modified locally and is ready to be published to the server. See "Publishing business rules" on page 74 for more information.

Related tasks

"Deleting a business rule, decision table or rule set record" on page 74

Using special actions menu in a decision table template:

The Decision Table page has a special actions menu to modify a template structure and values.

You need to be in the edit mode for the decision table and need to click the **Special Actions** icon ( ) to show the special actions menu. Any variable with the special actions icon can be changed so you can make multiple changes to a template.

The *Special Actions* menu has the following options.

Selection	Action
Add Below	Creates a new row below the present one
Change Template	Allows modifications to the cell values
Move Down	Moves the variable to the row below
Delete	Deletes the variable
Close Menu	Closes the menu and returns to the edit mode.

Modifying a template value of a decision table:

You must be in the edit mode for the decision table and click the special actions icon ( ) to show the special actions menu. You can change any variable with the special actions icon can be changed and you can make multiple changes to a template.

To change the template values, , do the following:
1. Change the template values.
   a. Select **Change Template** from the menu.
   b. Type in the new **values** for the that template.
2. Click **Close**.

   Note: If you only change one value, you can also click **Change** in the *Action* column.
3. Click **Save**.

The Decision Table template has been modified and is now ready for publishing. See "Publishing business rules" on page 74 for more information.

Rule sets

Create and modify rule sets in the business rules manager.

A rule set is a group of if/then rules that the server executes sequentially. The sequence is from top to bottom of the set. Therefore, when you modify or add a rule, make sure that it is in the correct sequence.

A rule set is created using templates. The template provides the structure that determines how the rule set functions.

Creating a rule set record:

To create a new rule set, you must first copy an existing rule set.

1. Copy the rule set you have selected.
   a. Click **Copy** in the rule logic record.
   b. Click **OK** to copy the record The **Edit** screen opens with a title, Edit Mode:Copy_of_TableName-Ruleset.
   c. Click **OK** to copy the record.
2. Type in the **name** of the new business rule record in the title field.
3. Type in a short **Description** of the new record.
4. Modify the **values** in each condition.

   Note: To display the parameter settings for each value, place your cursor over a field. A rollover message appears showing the type of variable and its range.
5. Click on the **Up** or **Down** arrow to place the rule in the correct sequence.
6. Click **Save**.

A message appears in the message field notifying stating that the record has been temporarily saved and you can publish the changes in the Publish and Revert page.

The rule set is ready for publishing. See "Publishing business rules" on page 74 for more information.

Related tasks

"Deleting a business rule, decision table or rule set record" on page 74

Creating a rule within a rule set from a template:

Create a new rule within a rule set using the rule templates associated with that rule set.

Open the rule set in the edit mode.

To create a new rule from a template, do the following:

1. Click the **New Rule from Template** button. A list of templates appears at the bottom of the page.
2. Create a new rule using an existing template.
   a. Type **Name** of the new rule in the Name field.
   b. Type or select the appropriate **variable(s) or constraint(s)** for the rule.
   c. Click **Add**.
3. Click the **Up** or **Down** arrows to place the rule in the proper order.

Note: For a rule set to function correctly, all rules must be in the right order.

4. Click **Save**.

The rule set is ready for publishing. See “Publishing business rules” for more information.

Related tasks

“Deleting a business rule, decision table or rule set record”

Modifying a rule within a rule set using a template:

Modify a rule in a rule set using templates associated with that rule set.

Open the rule set.

To edit a rule using an existing template, do the following:

1. Edit the **rule set**.
2. Click **Edit** button.
3. Change the **value** in the rule by typing over the existing value or selecting the appropriate value from the pull down list.
4. If necessary, click the **Up** or **Down** arrows to place the rule in the proper order.

   Note: For a rule set to function correctly, all rules must be in the right order.
5. Click **Save**.

The modified rule set is ready for publishing. See “Publishing business rules” for more information.

Related tasks

“Deleting a business rule, decision table or rule set record”

Deleting a business rule, decision table or rule set record

You must be in the edit mode.

Note: Each operation on a business rule group must have at least one business rule associated with it. Attempting to delete all business rules will result in an error.

There are times when you will need to delete a business rule, decision table, or rule set record. To delete a record, do the following:

1. Select the **record** from the business rule list.
2. Click **Delete** button. You are warned that you are about to delete a record.
3. Click **OK**.
4. Click the **Save** button.

The rule is modified locally and is ready to be published to the server. See “Publishing business rules”for more information.

Publishing business rules

Once you have saved any part of a business rule group, you are reminded that the changes have been saved locally and that you need to store the changes to the database. The server publishes changes at the Rule page level. You can publish multiple rule pages at the same time as a single transaction. An error can occur when another user updates the same rule logic or rule book that you are updating. See "Access conflict errors" on page 76 for more information.

To publish a business rule form the Business Rule Editor, do the following:

1. Click the **Publish and Revert** option in any screen or page that has a *Navigation* area . The Publish and Revert page opens.
2. 2. Select the pages to send to the database by clicking the check box on the left-hand column of the *Content* area. The selected Rule Pages do not appear when you refresh the screen, signifying that they have been written to the database.

The edited rules that have been published are stored on the database that resides on the application server. The business rule is ready to be exported to the server. (See "Exporting business rules using the administrative console" on page 57 for more information.)

Troubleshooting business rules manager

Some areas to examine if you experience problems with business rules manager are: login error, login conflict, and access conflict.

Login error:

Upon logging in, you receive a login error message.

The login error message:

```
Unable to process login. Please check User ID and password and try again.
```

This error occurs when global security is enabled and either the userid, the password, or both, are incorrect.

Note: Login errors occur only when global security is enabled.

1. Click **OK** on the error message.

   You return to the login page.
2. Enter valid **User ID** and **Password**.

   Make sure that Caps Lock key is not on, if passwords are case sensitive.

   Make sure the userid and password are spelled correctly.

   Check with the system administrator to see that the userid and password are correct.
3. Click the **Login** button.

If you resolve the login error, you will now be able to login to the business rules manager. If the error is not resolved, contact your system administrator.

Login conflict error:

This event occurs when another user with the same userid is already logged in to the application.

The login conflict message is:

```
Another user is currently logged in with the same User ID. Select from the
following options:
```

Usually this error occurs when a user closed the browser without logging out. When this condition occurs, the next attempted login before the session timeout expires results in a login conflict.

Note: Login conflict occurs only when global security is enabled.

There are three options that you can choose.

- Return to the login page.

  Use this option if you want to open the application with a different userid.
- Logout the other user with the same userid.

  Use this option to logout the other user and start a new session.

  Note: Any unpublished local changes made in the other session are lost.
- Inherit the context of the other user with the same userid and logout that user.

  Use this option to continue work already in progress. All unpublished local changes in the previous session that have been saved are not lost. The business rules manager opens to the last page displayed in the previous session.

Access conflict errors:

Access conflicts occur when a business rule is updated in the database by one user at the same time another user is updating the same rule.

This error is reported when you publish your local changes to the database.

These are the actions to correct access conflict errors.

- Publish the Rule page.
- Find the source of the business rule that is causing the error and check if your changes on the local machine are still valid. Your change may no longer be required after the changes done by another user.
- If you choose to continue working in the business rule manager, you must reload Rule Pages in the error from the database as your local changes of Rule pages in error are no longer usable. You can still use local changes in other Rule pages that are not in error.
- Reload a Rule page, by clicking **Reload** in the Publish and Revert page of the rule for which the error was reported. See "Publishing business rules" on page 74for more information.

Uninstalling and reinstalling applications using business rules manager

Business rules and selectors add flexibility to your modules. The added flexibility affects how you install or delete a module because the server saves business rules and selectors in a central repository.

Access the business rules manager through the administrative console which provides the ability to dynamically modify business rules and their effective dates. You can also use the administrative console to export business rule data from the repository to the file system.

In addition to the dynamic capabilities of the business rules manager, application installation and removal also affects the data in the repository. For more information on installing and removing applications that use business rules and selectors see the related topics.

Related concepts

Considerations for modules containing business rules and selectors
This topic contains information to consider when you install or delete modules that contain business rules and selectors

Related tasks

Removing business rule and selector data from the repository
When you uninstall an application that uses business rules or selectors, the server does not remove these artifacts from the repository. This task removes unneeded business rule and selector artifacts from the repository.

Administering enterprise applications

Use the console's Enterprise Application page (viewed by clicking **Applications > Enterprise Applications**) to view and administer enterprise applications installed on the server.

To view the values specified for an application's configuration, click the application name from the list. The application details page opens and displays the application's configuration properties and, if appropriate, local topology. From this page, you can modify existing values and link to additional console pages for configuring the application.

To administer an enterprise application, select it by clicking the check box next to its name and then use one of the following buttons:

Table 4. Buttons for administering enterprise applications

Button	**Resulting action**
Start	Attempts to run the application. After the application starts successfully, the state of the application changes to one of the following: • `Started`—The application has started on all deployment targets • `Partial Start`—The application is still starting on one or more of the deployment targets
Stop	Attempts to stop the processing of the application. After the application stops successfully, the state of the application changes to one of the following: • `Stopped`—The application has stopped on all deployment targets • `Partial Stop`—The application is still stopping on one or more of the deployment targets
Install	Opens a wizard to help you deploy an enterprise application or module (such as a .jar, .war, or .ear file) onto a server.
Uninstall	Deletes the application from the WebSphere Application Server configuration repository and deletes the application binaries from the file system of all nodes where the application modules are installed after the configuration is saved.

Table 4. Buttons for administering enterprise applications (continued)

Button	Resulting action
Update	Opens a wizard to help you update application files deployed on a server. You can update the full application, a single module, a single file, or part of the application. If a new file or module has the same name as a file or module already on the server, the new file or module replaces the existing one. Otherwise, it is added to the deployed application.
Remove File	Deletes a file from the deployed application or module. This button deletes the file from the configuration repository and from the file system of all nodes where the file is installed.
Export	Opens the Export Application EAR files page so you can export an enterprise application to an EAR file. Use the Export action to back up a deployed application and to preserve its binding information.
Export DDL	Opens the Export Application DDL files page so you can export DDL files in the EJB modules of an enterprise application.

For more information on administering applications, see the WebSphere Application Server Information Center.

Application Scheduler

Application Scheduler allows an administrator to schedule the starting and stopping of applications that are installed on WebSphere Process Server. Use the Application Scheduler panel in the administrative console to control the scheduling of any installed application. Additionally, you can generate scheduler entries during the migration of an WebSphere InterChange Server repository that includes WebSphere InterChange Server scheduler entries. (See the section on Migrating from WebSphere InterChange Server and the ReposMigrate command.) Use the Application Scheduler panel in the administrative console to administer these migrated scheduler entries as well.

In an Network Deployment environment, the Application Scheduler is automatically installed for every managed server and cluster member created - no additional action is needed. See "Planning to install Network Deployment" in WebSphere Application Server Network Deployment, Version 6.0 information center for instructions on creating new managed servers and cluster members.

In a standalone server environment, Application Scheduler is optional. While creating the standalone server profile, you select a check box to configure and install Application Scheduler on that server.

Configuring the Application Scheduler for a standalone server

To use Application Scheduler, you must make sure that it is installed. See Creating and augmenting profiles by using the Profile Wizard elsewhere in the information center.

You need to create a profile first.

This is an optional component and you must configure Application Scheduler to migrate WebSphere InterChange Server schedule entries to WebSphere Process Server Complete these steps to install the application server for a standalone server.

1. Type in the*Servername* (or use the default of *server1*).
2. Create or use an existing database. If you are creating a database name the database **WPRCSDB** (all caps). The server uses the default database tables and structure.

The installation is complete.

Application Scheduler is ready to use.

Accessing the Application Scheduler

Access Application Scheduler either programmatically using the Application Scheduler Mbean interface, or through the Application Scheduler panel of the administrative console.

For more information on accessing Application Scheduler see:

- "Accessing Application Scheduler using Application Scheduler MBean interface"
- "Displaying scheduler entries using the administrative console" on page 80

Accessing Application Scheduler using Application Scheduler MBean interface

Use the command line to invoke the Application Scheduler MBean

Perform the following to invoke Application Scheduler MBean.

1. Set the properties SOAP_HOSTNAME and SOAP_PORT in the class com.ibm.wbiserver.migration.ics.Parameters. This class is in the migration-wbi-ics.jar file in the *WAS_HOME*\lib directory. `SOAP_HOSTNAME` is the name of the host where Application Scheduler is running. `SOAP_PORT` is the port where the Application Scheduler is running.

   ```
   Parameters.instance.setProperty(Parameters.SOAP_HOSTNAME, "localhost");
   Parameters.instance.setProperty(Parameters.SOAP_PORT, "8880");
   ```

   Note: If security is turned on, you must specify a userid and password in the soap properties file found at the location in*WAS_HOME*\profiles*profiles*\properties\soap.client.props.

   This properties file name must be set in the Parameters instance shown here.

   ```
   Parameters.instance.setProperty(Parameters.SOAP_PROPERTIES,
   "WAS_HOME\profiles\profiles\properties\soap.client.props";
   ```

2. Create an instance of the class com.ibm.wbiserver.migration.ics.utils.MBeanUtil that implements calls to the AppScheduler Mbean.

   To instantiate an MBeanUtil, you must pass this query string to its constructor that invokes the correct Mbean based on its `name`, `type`, `server name` and `node name`.

   ```
   protected static final String WEBSPHERE_MB_QUERY_CONSTANT = "WebSphere:*";
   protected static final String NAME_QUERY_CONSTANT = ",name=";
   protected static final String WBI_SCHED_MB_NAME = "WBISchedulerMB1";
   protected static final String TYPE_QUERY_CONSTANT = ",type=";
   protected static final String WBI_SCHED_MB_TYPE = "WBIScheduler";
   protected static final String SERVER_QUERY_CONSTANT = ",process=";
   ```

```
    protected static final String NODE_QUERY_CONSTANT = ",node=";

    serverName = "server1";
    nodeName = "myNode";

    String queryString = new StringBuffer(WEBSPHERE_MB_QUERY_CONSTANT)
       .append(NAME_QUERY_CONSTANT).append(WBI_SCHED_MB_NAME).append(
       TYPE_QUERY_CONSTANT).append(WBI_SCHED_MB_TYPE).append(
       SERVER_QUERY_CONSTANT).append(serverName).append(
       NODE_QUERY_CONSTANT).append(nodeName).toString();

    MBeanUtil mbs = new MBeanUtil(queryString.toString());
```

3. Call Mbean methods using the invoke() method of the MbeanUtil instance and pass it the name of the method.

Here is an example of invoking the createSchedulerEntry method of the Scheduler Mbean. The first step is to create a `SchedulerEntry` and to set various parameters like name, `type`, `version`, `transition`, `entry status`, `recurrence type`, `recurrence week`, `recurrence period`, `initial date`, `repeat interval` and `component id`.

```
try
 {
 //First we set up the Schedule entry

 ScheduleEntry entry1 = new ScheduleEntry();
 entry1.setCName("BPEWebClient_localhost_server1");
 entry1.setCType("Application");
 entry1.setCVersion("ver1");
 entry1.setCTransition("startApplication");
 entry1.setSchedulerNumberOfRepeats(3); // Fire Three times
 entry1.setScheduleEntryStatus(TaskStatus.SCHEDULED);
 entry1.setRType(Recurrence.MINUTES);
 entry1.setRWeekNumber(-1);
 entry1.setRPeriod(2);
 entry1.setInitialDate(new Date(System.currentTimeMillis()+SIXTY_SECOND_OFFSET));
 entry1.setRepeatInterval(entry1.getInitialDate(), entry1.getRType(),
  entry1.getRWeekNumber(),
     entry1.getRPeriod());
 entry1.setComponentID(entry1.getCName(), entry1.getCType(), entry1.getCVersion(),
     entry1.getCTransition());
```

Then invoke the Mbean's createSchedulerEntry method. We pass it the scheduler entry `entry1` as a parameter along with the name of the ScheduleEntry class.

Then invoke the MBean's createScheduleEntry method:

```
mbs.invoke(schedulerExtMBName, "createScheduleEntry", new Object[]{entry1},
  new String[]{"com.ibm.wbiserver.scheduler.common.ScheduleEntry"});
```

Finally, read all the Schedule entries including the one that was just added by calling the readAllScheduleEntries method.

```
result = mbs.invoke("readAllScheduleEntries", null, null);
 }
 catch (MigrationException e)
 { e.printStackTrace();
 }
```

Displaying scheduler entries using the administrative console

Use the Application Scheduler panel of the administrative console to create, modify, or delete scheduler events.

You must be at the administrative console for the server to perform this task.

To display this panel and view existing scheduler events, follow these steps.

1. Select **Servers > Application Servers> ServerName**.
2. Select **Application Scheduler** from **Business Integration**.
3. Select the **scope** (cell, node, server) of the entries to display. The existing scheduled events for that scope are listed.

You can now edit the scheduler event, create a new scheduler event, or delete an existing event.

Creating a scheduled event

Administrative console provides a panel for creating new scheduled events.

To create a new scheduled event, you must be at the Application Scheduler collection panel in the administrative console for the server. See "Displaying scheduler entries using the administrative console" on page 80 for more information.

There are times that you will have to create an event to fit a specific need. To create a new scheduled event, follow these steps.

Note: The fields with an "*" on the panel are required fields.

1. Click **New**. The Add panel opens.
2. Configure the scheduled event.
   a. Select **Group Application**
   b. Select **Status**
   c. Type in the **Initial Date** with the following format *Abbrv month, dd, yyyy* for example, type in **Apr 15, 2005** for April 15, 2005.
   d. Type in the **Initial Time** using a 12-hour format*hh:mm*

      Note: You must also signify either **am** or **pm** and **time zone**.

         Note: After you have moved from this field, the **Next Fire Time** is automatically calculated.
   e. Select the **Action**.

      Optional: You can also fill in the **Recurrence** parameters.
      - Start-by-period
      - Whether the schedule entry should recur at a specified time.
        - One or more times a minute, hour, day, month or year.
        - A certain day (Sunday thru Saturday) of a certain week (first, second, third, fourth or last) of every one or more months.
        - The last day of every one or more months.
3. Click **Apply** or **OK** to set the event.

   Note: To create another event, click **Reset** to clear the panel.

Application Scheduler creates and displays a new scheduled event in the Application Scheduler panel.

Event status and action descriptions:

Each event must have a status and an action.

Status

The status field shows what state the event is in for monitoring purposes. This table lists each status.

Status	Description
Scheduled	A task is to fire at a predetermined date, time and interval. Each subsequent firing time is calculated.
Suspended	A task is suspended and will not fire until its status is changed to scheduled.
Complete	A task is completed.
Cancelled	A task has been cancelled. The task will not fire and it cannot be resumed, but it can be purged.
Invalid	Normally the reason that a task has a status of invalid is because either the task has been purged or the information used to query for that task is invalid.
Running	A task is in the midst of firing. **Note:** This status should be rarely seen since it just monitors the event for the very short duration that the event is firing.

Action

Each event must have an action associated with it. The action signifies what what to do with the event. There are only two actions available for an event:

- **Start Application** - starts all applications that are under the system's deployment manager.
- **Stop Application** - stops all applications that are under the system's deployment manager.

Modifying a scheduled event

Modify migrated or existing scheduled events from the administrative console.

To modify a scheduled event, you must be at the Application Scheduler collection panel in the administrative console for the server. See "Displaying scheduler entries using the administrative console" on page 80 for more information.

There are times that you need to modify an event to fit a need. To modify an event, follow these steps.

1. Click the **Event** you want to modify. The Event panel opens.
2. Modify any of the following fields:

   Note: Since all applications on the server are listed, you must be careful when changing the status of an existing event. You may stop an application that is running on the server.

   - **Group Application**
   - **Status**

- **Initial Date** with the following format *Abbrv month, dd, yyyy*
- **Initial Time** using a 12-hour format*hh:mm*
- **Action**

Optional: You can also fill in the **Recurrence** parameters.

3. Click **Apply** or **OK** to set the modifications for the event.

   Note: If you modify a scheduled event, the server assigns a new Schedule Entry ID. The server deletes the currently scheduled event and schedules a new event with the new ID.

The panel displays the modified event with the new ID in the Application Scheduler collection panel.

Deleting a scheduled event

Application Scheduler provides a panel for deleting scheduled events.

To delete a scheduled event, you must be at the Application Scheduler collection panel in the administrative console for the server. See "Displaying scheduler entries using the administrative console" on page 80 for more information.

As events become obsolete, you can delete them from the list of events in the collection panel. Follow these steps to delete a scheduled event.

1. Select the event to be deleted from the **Select** column.
2. Click **Delete**.
3. Click **OK** at the prompt.

The event is deleted.

Administering WebSphere Process Server resources

The administrative interfaces enable you to administer the resources associated with WebSphere Process Server, including selectors, target components, adapters, and the Extended Messaging Service.

Administering adapters

WebSphere Adapters, version 6.0, and WebSphere Business Integration Adapters (based on WebSphere Business Integration Framework, version 2.6) provide an approach to Enterprise Information Systems (EIS) integration which is application module oriented.

WebSphere Adapters (sometimes referred to as JCA Adapters or J2C Adapters) are compliant with J2EE Connector Architecture (JCA 1.5). JCA is the J2EE standard for EIS connectivity. EIS Import and EIS Export provide SCA application modules with a consistent view of those services outside the module. This allows components to communicate with the variety of EIS systems using the consistent SCA programming model.

WebSphere Adapters are assembled in WebSphere Integration Developer from imported RAR files and then exported as an Enterprise Application Archive (EAR) file and deployed on WebSphere Process Server.

Importing and exporting capabilities within the Service Component Architecture define the external interfaces or access points that a service module has in

WebSphere Process Server. Imports and exports can be either to other modules within the same application, or to other applications on EISs. Imports identify services outside a module, making them callable from within the module. Exports allow components in a module to provide their services to external clients. A module level import or export permits modules to access other modules. A system level import or export permits your applications to access applications on EISs as if they were local components, which allows your applications to work with WebSphere Adapters and WebSphere Business Integration Adapters.

WebSphere Business Integration Adapters consist of a collection of software, Application Programming Interfaces (APIs), and tools to enable applications to exchange business data through an integration broker. Each business application requires its own application-specific adapter to participate in the business integration process. You can install, configure, and test the adapter using current WebSphere Business Integration Adapter Framework and Development Kit System Manager tools. You can use WebSphere Integration Developer to import existing business objects and connector configuration files, to generate artifacts, and to assemble the solution for WebSphere Process Server. Operational commands for the WebSphere Business Integration Adapters are part of the WebSphere Process Server administrative console. For more information about working with these adapters and WebSphere Process Server, see the WebSphere Business Integration Adapter information center.

Working with WebSphere Adapters

WebSphere Adapters can be installed and administered using the administrative console.

To perform this task you must have access to the administrative console, and have security permission to alter server settings within the console.

This task should be performed if you have a WebSphere Adapter, that may or may not be embedded within an application. Use the administrative console to work with this adapter.

1. Install the WebSphere Adapter

   The process of installing WebSphere Adapters depends on whether they are embedded within an application:

Option	Description
Standalone adapters	Use the administrative console to install the adapter. **Note:** Standalone WebSphere Adapters are not supported in WebSphere Process Server v6.0.
Embedded adapters	The adapter will be installed as part of the application installation.

2. Administer the adapter
3. Configure J2EE connection factories.

Overview of adapters

There are two types of adapter that can be used in WebSphere Process Server: WebSphere Business Integration Adapters and WebSphere Adapters. An overview of the functionality of these adapters is provided.

Adapters provide connectivity to access data, technologies and protocols that enhance integration infrastructure. They extract data and transaction information from cross-industry and industry-specific packaged applications and connect them to a central server.

Two types of adapter are supported in WebSphere Process Server:
- **WebSphere Business Integration Adapters** consist of a collection of software, application program interfaces (APIs) and tools that enable applications to exchange business data through an integration broker.
- **WebSphere Adapters** (sometimes referred to as Resource Adapters) are the preferred technology. They enable managed, bidirectional connectivity between Enterprise Information Systems (EISs) and J2EE components supported by WebSphere Process Server.

Each business application requires its own application-specific adapter to participate in the business integration process. Adapters allow communication from various Enterprise Applications, such as Seibel or PeopleSoft to WebSphere Process Server.

WebSphere Business Integration Adapters allow multiple applications to communicate to fully integrate your system. Users are encouraged to use WebSphere Adapters which are fully compliant with the Java 2 Enterprise Edition (J2EE) Connector Architecture (JCA) version 1.5. The WebSphere Business Integration Adapters are not JCA-compliant, and run outside the application server.

The WebSphere Business Integration Adapters are the appropriate choice if a WebSphere Adapter does not yet exist for your application. If you have purchased WebSphere Business Integration Adapter licenses you may continue to use them, but you are encouraged to migrate to WebSphere Adapters as these will provide an updated way to connect to popular packaged software.

For information on developing applications using adapters see the WebSphere Integration Developer Information Center at WebSphere Integration Developer Information Center .

For information on installing, deploying and configuring WebSphere Adapters see the WebSphere Adapters Information Center at WebSphere Adapters Information Center.

For information on developing WebSphere Adapters for WebSphere Process Server, see (DeveloperWorks with the WebSphere Adapter Toolkit Guide).

Introduction to WebSphere Adapters: WebSphere Adapters implement the Java 2 Enterprise Edition (J2EE) Connector Architecture (JCA) version 1.5. They are referred to as WebSphere Adapters or Resource Adapters. They manage bidirectional connectivity between Enterprise Information Systems (EISs) and J2EE components supported by WebSphere Process Server.

JCA is designed to facilitate data sharing and to integrate new J2EE applications with legacy and other EISs. JCA stipulates how to develop a WebSphere Adapter that can:
- Plug into any J2EE-compliant application server.
- Connect an application running on that server with an EIS.
- Enable data exchange between the J2EE application and the EIS.

The JCA standard accomplishes this by defining a series of contracts that govern interactions between an EIS and J2EE components within an application server. Fully compliant with the JCA standard, WebSphere Adapters have been developed to run on WebSphere Process Server. Compliance with JCA has several advantages:

- JCA is an open standard,
- JCA is the J2EE standard for EIS connectivity,
- JCA provides a managed framework.

Each WebSphere Adapter is made up of the following:

- **Foundation classes** These implement a generic set of contracts that WebSphere Process Server uses to manage interactions between J2EE applications and all WebSphere Adapters. These quality of service and life cycle management contracts, also known as system contracts, define the service provider interface (SPI). For example, system contracts specify security credential management, connection pooling and transaction management parameters.
- **EIS subclasses** These generic and EIS-specific subclasses define the Common Client Interface (CCI) and EIS API contracts. For example, Activation and Connection Specs allow WebSphere Process Server to manage incoming and outgoing events for the WebSphere Adapter.
- **Enterprise Metadata Discovery** This utility introspects the EIS to generate service data objects (SDOs) and other artifacts that are compiled in a standard Enterprise Application Archive (EAR) file.

A simplified version of the operation of a WebSphere Adapter is shown schematically in figure 1:

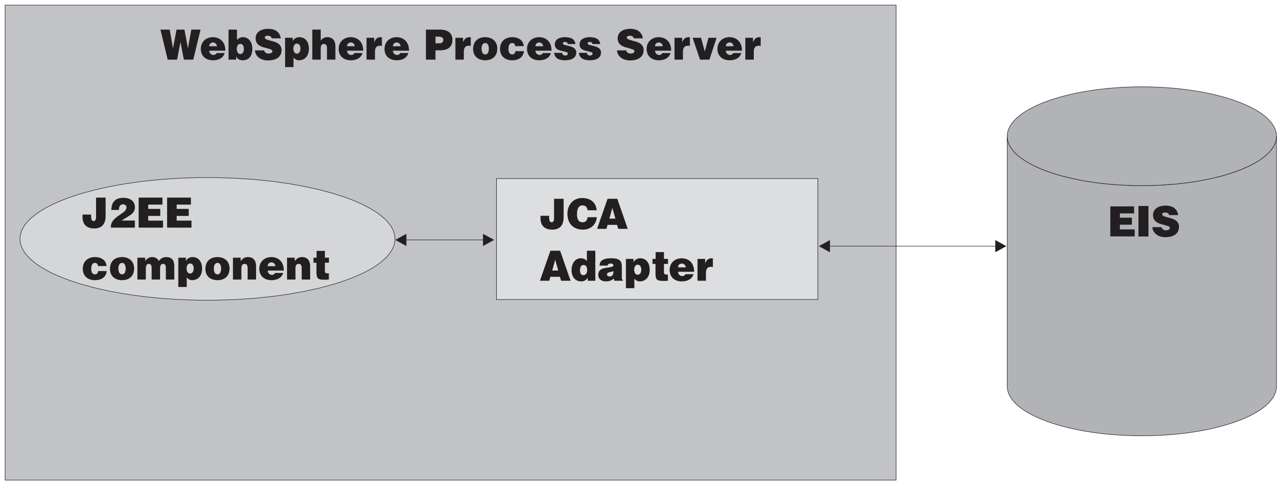


Figure 3. Simplified schematic of a WebSphere Adapter.

Differences between WebSphere Adapters and WebSphere Business Integration Adapters: There are several differences between WebSphere Adapters and WebSphere Business Integration Adapters. These distinctions are most important during development of applications. When deploying applications to a running server, the nature of the adapters used affects some of the steps which need to be followed.

Adapters provide communication mechanisms between Enterprise Information Systems (EISs) and WebSphere applications. To illustrate the operation of the adapters figures, 1 and 2 provide details of the communication between WebSphere Process Server and the EIS for the two types of adapter.

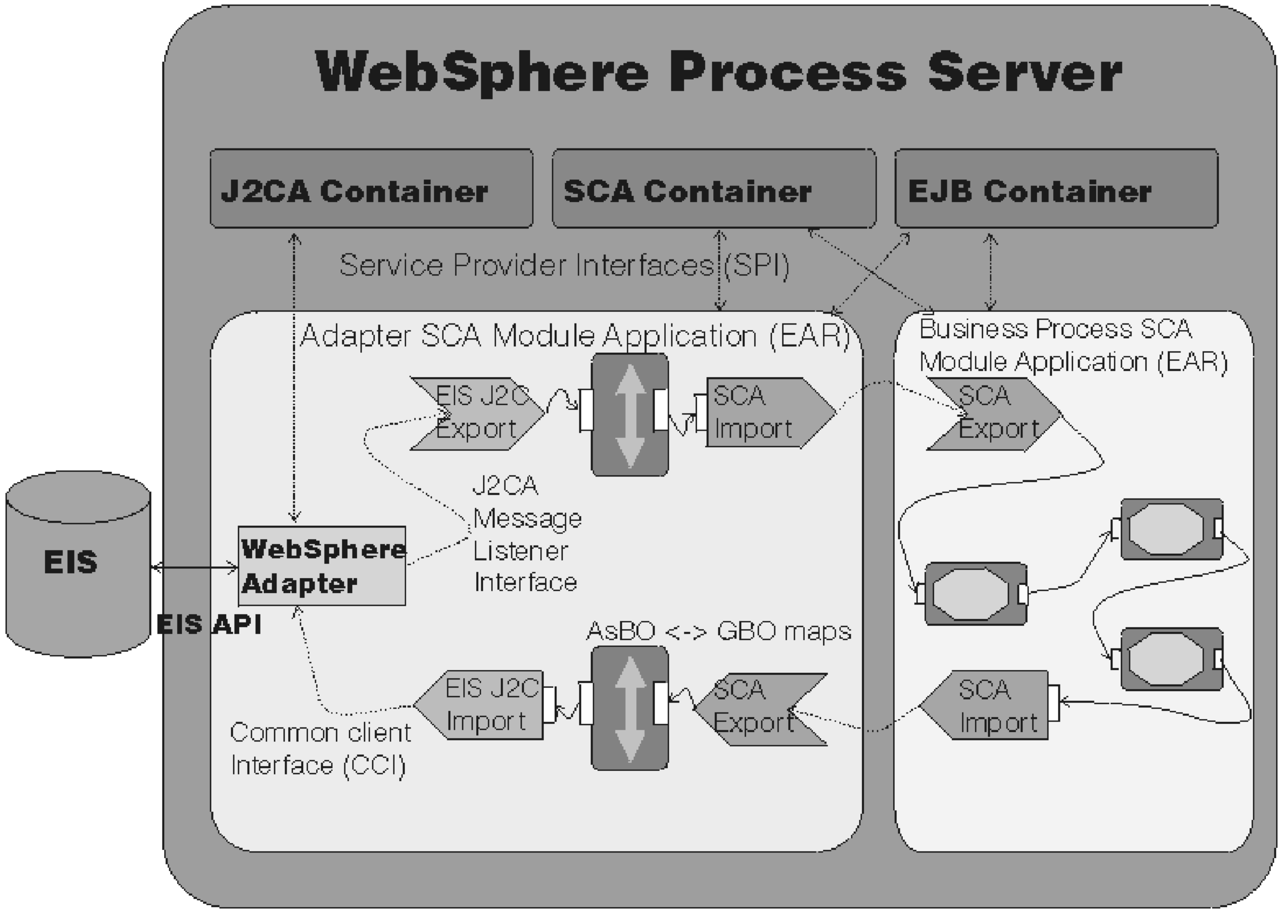


Figure 4. Detailed schematic of a WebSphere Adapter.

Figure 1 depicts a WebSphere Adapter managing the connectivity between a J2EE component supported by WebSphere Process Server and the EIS. The WebSphere Adapter resides inside WebSphere Process Server.

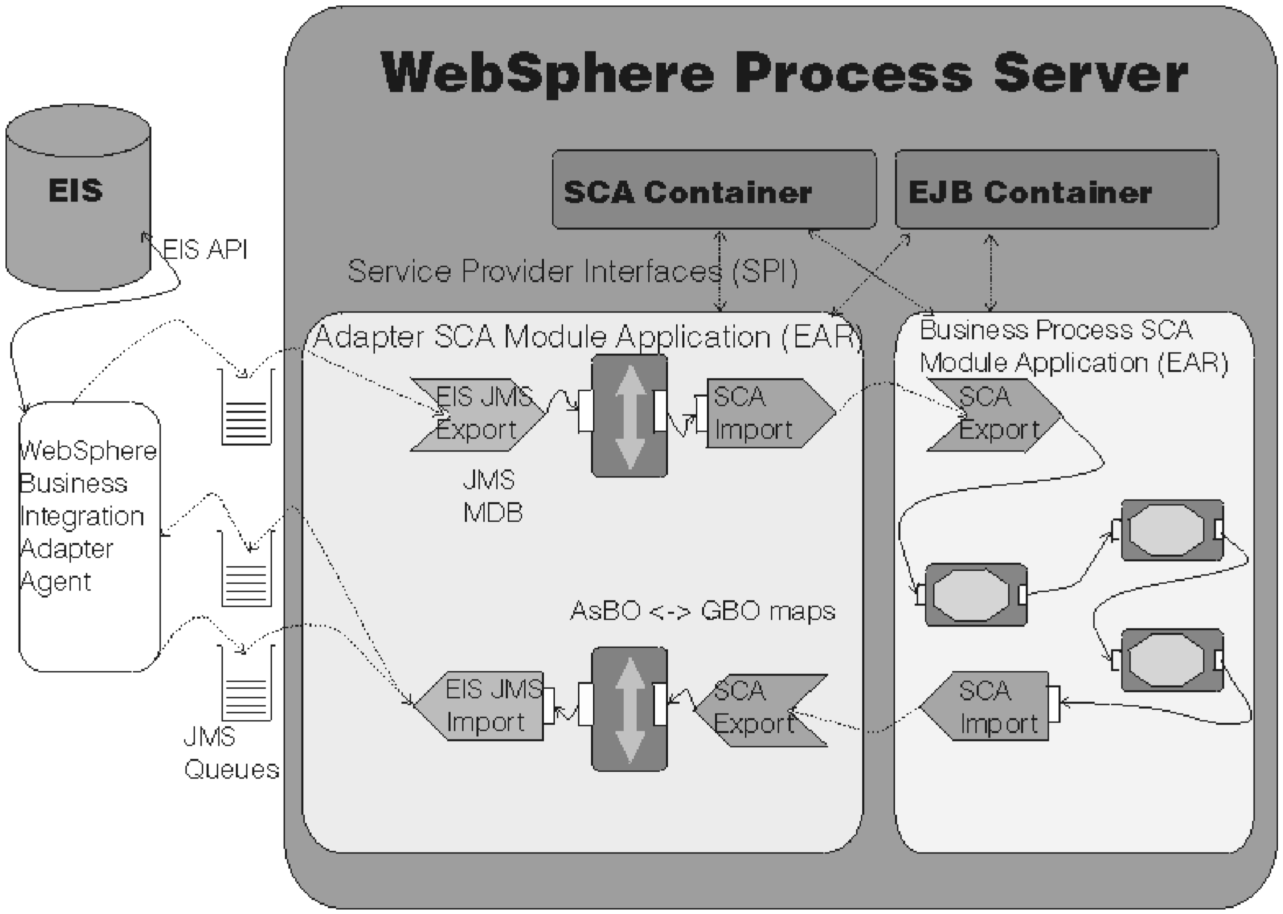


Figure 5. Detailed schematic of a WebSphere Business Integration Adapter.

Figure 2 shows a WebSphere Business Integration Adapter mediating communication between the WebSphere Integration Broker and the EIS. The integration broker communicates with the WebSphere Business Integration Adapter via a Java Messaging Service (JMS) transport layer.

Differences between the two types of adapter include:

Table 5. Differences between WebSphere Adapters and WebSphere Business Integration Adapters.

Feature	WebSphere Adapters	WebSphere Business Integration Adapters
JCA Compliance	Fully JCA compliant (version 1.5).	Not JCA-compliant.
Connectivity Manager	Rely on standard JCA contracts to manage life cycle tasks such as starting and stopping.	Rely on WebSphere Adapter Framework to manage connectivity.
Event Notification	Use an EventStore subclass to retrieve events from an EIS.	Manage event notification using a pollFor Events method.
Request Processing	Clients directly invoke one of several interaction contracts to query or modify data in the EIS.	Rely on an integration server and the WebSphere Adapter Framework to initiate and help process requests.

Table 5. Differences between WebSphere Adapters and WebSphere Business Integration Adapters. (continued)

Feature	WebSphere Adapters	WebSphere Business Integration Adapters
Data Models	Use an Enterprise Metadata Discovery (EMD) utility to parse an EIS and develop Service Data Objects (SDOs) and other useful artifacts. The EMD is part of the WebSphere Adapter implementation.	Use a separate Object Discovery Agent (ODA) to introspect an EIS and generate business object definition schemas.
Integration	Run on WebSphere Process Server.	Reside outside the application server. The server or integration broker communicates with the adapter via a Java Messaging Service (JMS) transport layer.

WebSphere Adapters are the recommended tool.

Advantages of using WebSphere Adapters: WebSphere Adapters provide several advantages over WebSphere Business Integration Adapters. These advantages are summarized here:

- Integration - WebSphere Adapters are integrated into WebSphere Process Server, WebSphere Business Integration Adapters are outside the application server.
- JCA compliance - only WebSphere Adapters are fully compliant with JCA version 1.5.
- Request processing - WebSphere Adapters do not rely on the WebSphere Adapter Framework nor an integration server for request initiation with Enterprise Information Systems.
- Connectivity - WebSphere Adapters do not rely on the WebSphere Adapter Framework for connectivity, but use JCA contracts to manage life cycle tasks.
- Data models - WebSphere Adapters use an enterprise service discovery wizard to parse an EIS and develop Service Data Objects (SDOs). The enterprise service discovery wizard is part of the WebSphere Adapter implementation. WebSphere Business Integration Adapters use a separate object.
- Event notification - WebSphere Adapters use a subclass of EventStore to retrieve events from an EIS, whereas WebSphere Business Integration Adapters use a pollFor Events method.

There are a limited set of WebSphere Adapters available but it is recommended to use them wherever possible.

Installing applications with embedded WebSphere Adapters

If an application is developed with a WebSphere Adapter embedded, the adapter is deployed with the application. You do not need to install the adapter separately. The steps to install an application with an embedded adapter are described.

This task should only be performed if the application is developed with an embedded WebSphere Adapter.

1. Assemble an application with resource adapter archive (RAR) modules in it. See Assembling applications.
2. Install the application following the steps in Installing a new application. In the Map modules to servers step, specify target servers or clusters for each RAR

file. Be sure to map all other modules that use the resource adapters defined in the RAR modules to the same targets. Also, specify the Web servers as targets that serve as routers for requests to this application. The plug-in configuration file (plugin-cfg.xml) for each Web server is generated based on the applications that are routed through it.

Note: When installing a RAR file onto a server, WebSphere Application Server looks for the manifest (MANIFEST.MF) for the connector module. It looks first in the connectorModule.jar file for the RAR file and loads the manifest from the _connectorModule.jar file. If the class path entry is in the manifest from the connectorModule.jar file, then the RAR uses that class path. To ensure that the installed connector module finds the classes and resources that it needs, check the Class path setting for the RAR using the console. For more information, see Resource Adapter settings and WebSphere relational resource adapter settings.

3. Save the changes. Click **Finish > Save**.
4. Create connection factories for the newly installed application
   a. Open the administrative console.
   b. Select the newly installed application Click **Applications > Enterprise Applications >** *application name*.
   c. Click **Connector Modules** in the Related Items section of the page.
   d. Select the RAR file. Click on *filename.rar*
   e. Click **Resource adapter** in the Additional Properties section of the page.
   f. Click **J2C Connection Factories** in the Additional Properties section of the page.
   g. Click on an **existing connection factory** to update it, or **New** to create a new one.

      Note: If the WebSphere Adapter was configured using an EIS Import or EIS Export a ConnectionFactory or ActivationSpec will exist and can be updated.

If you install an adapter that includes native path elements, consider the following: If you have more than one native path element, and one of the native libraries (native library A) is dependent on another library (native library B), then you must copy native library B to a system directory. Because of limitations on most UNIX systems, an attempt to load a native library does not look in the current directory.

After you create and save the connection factories, you can modify the resource references defined in various modules of the application and specify the Java Naming and Directory Interface (JNDI) names of the connection factories wherever appropriate.

Note:

A given native library can only be loaded one time for each instance of the Java virtual machine (JVM). Because each application has its own classloader, separate applications with embedded RAR files cannot both use the same native library. The second application receives an exception when it tries to load the library.

If any application deployed on the application server uses an embedded RAR file that includes native path elements, then you must always ensure that you shut down the application server cleanly, with no outstanding transactions. If the application server does not shut down cleanly it performs recovery upon server restart and loads any required RAR files and native libraries. On completion of recovery, do not attempt any application-related work. Shut down the server and restart it. No further recovery is attempted by the application server on this restart, and normal application processing can proceed.

WebSphere Adapter:

A WebSphere Adapter (or JCA Adapter, or J2C Adapter) is a system-level software driver that a Java application uses to connect to an enterprise information system (EIS). WebSphere Adapters conform to version 1.5 of the JCA specification.

A WebSphere Adapter plugs into an application server and provides connectivity between the EIS, the application server, and the enterprise application.

An application server vendor extends its system once to support the J2EE Connector Architecture (JCA) and is then assured of seamless connectivity to multiple EISs. Likewise, an EIS vendor provides one standard WebSphere Adapter with the capability to plug into any application server that supports the connector architecture.

WebSphere Process Server provides the WebSphere Relational Resource Adapter (RRA) implementation. This WebSphere Adapter provides data access through JDBC calls to access the database dynamically. The connection management is based on the JCA connection management architecture. It provides connection pooling, transaction, and security support. WebSphere Process Server version 6.0 supports JCA version 1.5.

Data access for container-managed persistence (CMP) beans is managed by the WebSphere Persistence Manager indirectly. The JCA specification supports persistence manager delegation of the data access to the WebSphere Adapter without specific knowledge of the back-end store. For the relational database access, the persistence manager uses the relational resource adapter to access the data from the database. You can find the supported database platforms for the JDBC API at the WebSphere Process Server prerequisite Web site.

IBM supplies resource adapters for many enterprise systems separately from the WebSphere Process Server package, including (but not limited to): the Customer Information Control System (CICS), Host On-Demand (HOD), Information Management System (IMS), and Systems, Applications, and Products (SAP) R/3.

In WebSphere Process Server, EIS Imports and EIS Exports are used to interface with WebSphere Adapters. As an alternative, applications with WebSphere Adapters can be written by developing EJB session beans or services with tools such as Rational Application Developer. The session bean uses the javax.resource.cci interfaces to communicate with an enterprise information system through the WebSphere Adapter.

WebSphere Adapter deployment considerations:

The deployment of WebSphere Adapters requires specific options regarding scope.

You can deploy a WebSphere Adapter in two ways, using the administrative console:

- Standalone - the adapter is installed at the node level and is not associated with a specific application.

  Note: Deployment of standalone WebSphere Adapters is not supported in WebSphere Process Server v6.0.
- Embedded - the adapter is part of an application, deploying the application also deploys the adapter.

For embedded WebSphere Adapters:
- the RAR file can be application-scoped within an SCA module (with EIS imports or exports).
- the RAR file can be application-scoped within a non-SCA module. The application itself, containing the EIS imports and exports, is a separate SCA module.

You should not install standalone WebSphere Adapters.

Note: The administrative console does not preclude the installation of standalone WebSphere Adapters, but this should not be done. WebSphere Adapters should be embedded in applications.

Only embedded WebSphere Adapters are appropriate for deployment in WebSphere Process Server. Furthermore, deployment of an embedded WebSphere Adapter is only supported for RAR files that are application-scoped within an SCA module; deployment in a non-SCA module is not supported.

Installing Standalone WebSphere Adapters:

WebSphere Adapters should be embedded in applications. Standalone WebSphere Adapters are not support in WebSphere Process Server v6.0. These instructions are for reference only. If you intend to use a standalone WebSphere Adapter you should install it, as described here. You can alternatively use an embedded adapter, which is installed automatically as part of the installation of the associated application.

You should configure the database before installing the adapter.

You must have access to, and be part of the necessary security role for, the administrative console to perform this task.
1. Open the Install RAR file dialog window.

   On the administrative console:
   a. Expand **Resources**
   b. Click **Resource Adapters**
   c. Select the scope at which you want to define this resource adapter. (This scope becomes the scope of your connection factory). You can choose cell, node, cluster, or server.
   d. Click **Install RAR**

   A window opens in which you can install a JCA connector and create, for it, a WebSphere Adapter. You can also use the New button, but the New button creates only a new resource adapter (the JCA connector must already be installed on the system).

Note: When installing a RAR file using this dialog, the scope you define on the Resource Adapters page has no effect on where the RAR file is installed. You can install RAR files only at the node level. The node on which the file is installed is determined by the scope on the Install RAR page. (The scope you set on the Resource Adapters page determines the scope of the new resource adapters, which you can install at the server, node, or cell level.)

2. Install the RAR file

   From the dialog, install the WebSphere Adapter in the following manner:

   a. Browse to the location of the JCA connector. If the RAR file is on the local workstation select Local Path and browse to find the file. If the RAR file is on a network server, select **Server path** and specify the fully qualified path to the file.
   b. Click **Next**
   c. Enter the resource adapter name and any other properties needed under General Properties. If you install a J2C Resource Adapter that includes native path elements, consider the following: If you have more than one native path element, and one of the native libraries (native library A) is dependent on another library (native library B), then you must copy native library B to a system directory. Because of limitations on most UNIX systems, an attempt to load a native library does not look in the current directory.
   d. Click **OK**.

WebSphere Adapter applications as members of clusters:

Cluster deployment is not supported in WebSphere Process Server v6.0. This information is included for completeness. WebSphere Adapter module applications can be cloned as members of a cluster under certain conditions.

WebSphere Adapter module applications can be one of three types, depending on the flow of information through the adapter:
- A WebSphere Adapter application with only EIS exports - only inbound traffic.
- A WebSphere Adapter application with only EIS imports - only outbound traffic.
- A WebSphere Adapter application with both EIS imports and exports - bidirectional traffic.

Clusters are used to provide scalability and availability to your applications in a network deployment environment.

WebSphere Adapter module applications that have either inbound or bidirectional traffic, cannot be cloned as members of a cluster. An application with purely outbound traffic can be cloned as a member of a cluster.

An application which has an inbound or bidirectional WebSphere Adapter (i.e., including EIS exports) can still be given availability in a network deployment by use of an external Operating System High Availability (HA) management software package, such as HACMP, Veritas or Microsoft Cluster Server.

WebSphere Business Integration Adapter applications as members of clusters:

Cluster deployment is not supported in WebSphere Process Server v6.0. This information is included for completeness. WebSphere Business Integration Adapter module applications can be cloned as members of a cluster under certain conditions.

WebSphere Business Integration Adapter module applications can be one of three types, depending on the flow of information through the adapter:

- A WebSphere Business Integration Adapter application with only EIS exports - only inbound traffic.
- A WebSphere Business Integration Adapter application with only EIS imports - only outbound traffic.
- A WebSphere Business Integration Adapter application with both EIS imports and exports - bidirectional traffic.

Clusters are used to provide scalability and availability to your applications in a network deployment environment.

WebSphere Business Integration Adapter module applications that have either inbound or bidirectional traffic, cannot be cloned as members of a cluster. An application with purely outbound traffic can be cloned as a member of a cluster.

An application which has inbound or bidirectional WebSphere Business Integration Adapter (i.e., including EIS exports) can still be given availability in a network deployment by use of an external Operating System High Availability (HA) management software package, such as HACMP, Veritas or Microsoft Cluster Server.

Administering a WebSphere Adapter using the administrative console

The administrative console allows the user to administer and configure WebSphere Adapters.

In order to perform this task, you must have security permission to change settings in the administrative console.

When you have installed a WebSphere Adapter, you can administer it using the administrative console.

1. Open the resource adapter console for the adapter you wish to administer.

   On the administrative console:

   a. Expand **Resources**

   b. Click **Resource Adapters**

   c. Choose the WebSphere Adapter to administer.

2. Alter desired properties under the General Properties or Additional Properties headings.

   When you have finished making changes click the **Apply** button.

Changes are only applied to the local configuration until you save them to the master configuration.

Note: The server may need to be restarted in order for changes to become effective.

Related reference

"WebSphere Adapter administrative console settings"
The administrative console settings for WebSphere Adapters and their default values are described here.

WebSphere Adapter administrative console settings:

The administrative console settings for WebSphere Adapters and their default values are described here.

Purpose

Use the administrative console to edit the settings for your WebSphere Adapters.

To view the administrative console:
- Expand **Resources**
- Click **Resource Adapters**
- Choose the resource adapter from the list.

The various fields are described here:

Scope Specifies the level to which this resource definition is visible. For general information, see Administrative console scope settings in the Related Reference section. The Scope field is a read only string field that shows where the particular definition for a resource adapter is located. This is set either when the resource adapter is installed (which can only be at the node level) or when a new resource adapter definition is added.

Name Specifies the name of the resource adapter definition. This property is required. A string with no spaces meant to be a meaningful text identifier for the resource adapter.

Data type	String

Description Specifies a text description of the resource adapter. A free-form text string to describe the resource adapter and its purpose.

Data type	String

Archive path Specifies the path to the RAR file containing the module for this resource adapter. This property is required.

Data type	String

Class path Specifies a list of paths or JAR file names which together form the location for the resource adapter classes. This includes any additional libraries needed by the resource adapter. The resource adapter code base itself is automatically added to the class path, but if anything outside the RAR is needed it can be specified here.

Data type	String

Native path Specifies a list of paths which forms the location for the resource adapter native libraries. The resource adapter code base itself is automatically added to the class path, but if anything outside the RAR is needed it can be specified here.

Data type	String

ThreadPool Alias Specifies the name of a thread pool that is configured in the server that is used by the resource adapter's Work Manager. If there is no thread pool configured in the server with this name, the default configured thread pool instance, named Default, is used. This property is only necessary if this resource adapter uses Work Manager.

Data type	String

Administering Connection Factories

A connection factory is used in communication between an application and enterprise information system (EIS).

An application component uses a connection factory to access a connection instance, which the component then uses to connect to the underlying EIS.

Examples of connections include database connections, Java Message Service connections, and SAP R/3 connections.

Configuring J2EE Connector connection factories in the administrative console:

Connection factories are used in mediating communication between an application and an enterprise information system (EIS). The administrative console can be used to administer connection factories.

This task should be performed if you have a standalone resource adapter. Embedded WebSphere Adapters require slightly different treatment. You must also have security permission to edit the administrative console.

If your application requires access to a non-relational database, you need to configure a resource adapter and a connection factory rather than a JDBC provider and a data source.

1. Select the resource adapter to administer.

   From the top level of the administrative console, perform the following steps:

   a. Expand **Resources**
   b. Select **Resource Adapters**
   c. Select the WebSphere Adapter that you want to administer.

2. Create a new connection factory for this adapter.

   Create the new connection factory by:

   a. Select JCA Connection Factories under the Additional Properties
   b. Click **New**
   c. Specify general properties
   d. Specify authentication preference
   e. Select aliases for component-managed authentication, container-managed authentication, or both.

      If no aliases are available, or you want to define a different one,

1) Click **Apply**
2) Click **J2C Authentication Data Entries** under Related Items
3) Click **New**
4) Specify general properties
5) Click **OK**

f. Click **OK**

3. Modify the connection pool properties of the newly created connection factory to optimize the behavior of the connection pool manager.

   Change connection pool values by:

   a. Select the new Connection Factory
   b. Under the Additional Properties heading select **Connection pool properties**.
   c. Change any desired values by clicking the property name.
   d. Click **OK**
   e. Under Additional Properties click **Custom Properties**.
   f. Click any property name to change its value.
   g. Click **OK**.

Setting general properties for connection pools:

You can assign general settings to a connection pool using the administrative console.

To change general settings for a connection pool you must have security permission to alter values on the administrative console.

In order to assign general properties of a connection pool, you must first create a connection factory for your WebSphere Adapter.

1. Open the Connection Pool Properties panel in the administrative console.

   From the top level of the administrative console:

   a. Expand **Resources**
   b. Click **Resource Adapter**
   c. Click on the WebSphere Adapter which has the connection factory you wish to administer.
   d. Click **J2C Connection Factories** under the Additional Properties heading.
   e. Select the connection factory to administer.
   f. Click **Connection Pool Properties** under the Additional Properties heading.
2. General connection pool properties such as timeouts, maximum and minimum connections and purge policy can be altered on this panel. Default values are supplied.
3. After changing the desired properties, click **OK**.

   Related reference

   "Connection pool settings"
   You can change the values of various properties of a connection pool on the Connection Pool Properties panel of the administrative console.

Connection pool settings:

You can change the values of various properties of a connection pool on the Connection Pool Properties panel of the administrative console.

Purpose

The Connection Pool Settings panel is used to assign general property values for connection pools. You can edit properties such as time-outs, purge policies and connection limits.

Configuration Tab

Connection timeout

Specifies the interval, in seconds, after which a connection request times out and a ConnectionWaitTimeoutException is thrown.

This value indicates the number of seconds a request for a connection waits when there are no connections available in the free pool and no new connections can be created, usually because the maximum value of connections in the particular connection pool has been reached. For example, if Connection Timeout is set to 300, and the maximum number of connections are all in use, the pool manager waits for 300 seconds for a physical connection to become available. If a physical connection is not available within this time, the pool manager initiates a ConnectionWaitTimeout exception. It usually does not make sense to retry the getConnection() method; if a longer wait time is required you should increase the Connection Timeout setting value. If a ConnectionWaitTimeout exception is caught by the application, the administrator should review the expected connection pool usage of the application and tune the connection pool and database accordingly.

If the Connection Timeout is set to 0, the pool manager waits as long as necessary until a connection becomes available. This happens when the application completes a transaction and returns a connection to the pool, or when the number of connections falls below the value of Maximum Connections, allowing a new physical connection to be created.

If Maximum Connections is set to 0, which enables an infinite number of physical connections, then the Connection Timeout value is ignored.

Data type	Integer
Units	Seconds
Default	180
Range	0 to max int

Maximum connections

Specifies the maximum number of physical connections that you can create in this pool.

These are the physical connections to the back-end resource. Once this number is reached, no new physical connections are created and the requester waits until a physical connection that is currently in use returns to the pool, or a ConnectionWaitTimeout exception is issued.

For example, if the Maximum Connections value is set to 5, and there are five physical connections in use, the pool manager waits for the amount of time specified in Connection Timeout for a physical connection to become free.

If Maximum Connections is set to 0, the connection pool is allowed to grow infinitely. This also has the side effect of causing the Connection Timeout value to be ignored.

If multiple standalone application servers use the same data source, there is one pool for each application server. If clones are used, one data pool exists for each clone. Knowing the number of data pools is important when configuring the database maximum connections.

You can use the Tivoli Performance Viewer to find the optimal number of connections in a pool. If the number of concurrent waiters is greater than 0, but the CPU load is not close to 100%, consider increasing the connection pool size. If the Percent Used value is consistently low under normal workload, consider decreasing the number of connections in the pool.

Data type	Integer
Default	10
Range	0 to max int

Minimum connections

Specifies the minimum number of physical connections to maintain.

If the size of the connection pool is at or below the minimum connection pool size, the Unused Timeout thread does not discard physical connections. However, the pool does not create connections solely to ensure that the minimum connection pool size is maintained. Also, if you set a value for Aged Timeout, connections with an expired age are discarded, regardless of the minimum pool size setting.

For example if the Minimum Connections value is set to 3, and one physical connection is created, the Unused Timeout thread does not discard that connection. By the same token, the thread does not automatically create two additional physical connections to reach the Minimum Connections setting.

Data type	Integer
Default	1
Range	0 to max int

Reap time

Specifies the interval, in seconds, between runs of the pool maintenance thread.

For example, if Reap Time is set to 60, the pool maintenance thread runs every 60 seconds. The Reap Time interval affects the accuracy of the Unused Timeout and Aged Timeout settings. The smaller the interval, the greater the accuracy. If the pool maintenance thread is enabled, set the Reap Time value less than the values of Unused Timeout and Aged Timeout. When the pool maintenance thread runs, it discards any connections remaining unused for longer than the time value specified in Unused Timeout, until it reaches the number of connections specified in Minimum Connections. The pool maintenance thread also discards any connections that remain active longer than the time value specified in Aged Timeout.

The Reap Time interval also affects performance. Smaller intervals mean that the pool maintenance thread runs more often and degrades performance.

To disable the pool maintenance thread set Reap Time to 0, or set both Unused Timeout and Aged Timeout to 0. The recommended way to disable the pool maintenance thread is to set Reap Time to 0, in which case Unused Timeout and Aged Timeout are ignored. However, if Unused Timeout and Aged Timeout are set to 0, the pool maintenance thread runs, but only physical connections which timeout due to non-zero timeout values are discarded.

Data type	Integer
Units	Seconds
Default	180
Range	0 to max int

Unused timeout

Specifies the interval in seconds after which an unused or idle connection is discarded.

Set the Unused Timeout value higher than the Reap Timeout value for optimal performance. Unused physical connections are only discarded if the current number of connections exceeds the Minimum Connections setting. For example, if the unused timeout value is set to 120, and the pool maintenance thread is enabled (Reap Time is not 0), any physical connection that remains unused for two minutes is discarded. Note that accuracy of this timeout, as well as performance, is affected by the Reap Time value. See Reap Time for more information.

Data type	Integer
Units	Seconds
Default	1800
Range	0 to max int

Aged timeout

Specifies the interval in seconds before a physical connection is discarded.

Setting Aged Timeout to 0 supports active physical connections remaining in the pool indefinitely. Set the Aged Timeout value higher than the Reap Timeout value for optimal performance. For example, if the Aged Timeout value is set to 1200, and the Reap Time value is not 0, any physical connection that remains in existence for 1200 seconds (20 minutes) is discarded from the pool. Note that accuracy of this timeout, as well as performance, are affected by the Reap Time value. See Reap Time for more information.

Data type	Integer
Units	Seconds
Default	0
Range	0 to max int

Purge policy

Specifies how to purge connections when a stale connection or fatal connection error is detected.

Valid values are:

- EntirePool: All connections in the pool are marked stale. Any connection not in use is immediately closed. A connection in use is closed and issues a stale connection exception during the next operation on that connection. Subsequent getConnection() requests from the application result in new connections to the database opening. When using this purge policy, there is a possibility that some connections in the pool are closed unnecessarily when they are not stale. However, this is a rare occurrence. In most cases, a purge policy of EntirePool is the best choice.
- FailingConnectionOnly: Only the connection that caused the stale connection exception is closed. Although this setting eliminates the possibility that valid connections are closed unnecessarily, it makes recovery from an application

perspective more complicated. Only the currently failing connection is closed, so there is a good possibility that the next getConnection() request from the application will return a connection from the pool that is also stale, resulting in more stale connection exceptions. The connection pretest function attempts to insulate an application from pooled connections that are not valid. When a back-end resource, such as a database, goes down, pooled connections that are not valid might exist in the free pool. This is especially true when the purge policy is failingConnectionOnly; in this case, the failing connection is removed from the pool. Depending on the failure, the remaining connections in the pool might not be valid.

Data type	String
Default	EntirePool

Setting advanced properties for connection pools:

You can assign advanced settings to a connection pool using the administrative console.

To change advanced settings for a connection pool you must have security permission to alter values on the administrative console.

In order to assign advanced properties of a connection pool, you must first create a connection factory for your WebSphere Adapter.

1. Open the Advanced Connection Pool Properties panel in the administrative console.

   From the top level of the administrative console:

   a. Expand **Resources**
   b. Click **Resource Adapter**
   c. Click on the WebSphere Adapter which has the connection factory you wish to administer.
   d. Click J2C Connection Factories under the Additional Properties heading.
   e. Select the connection factory to administer.
   f. Click Connection Pool Properties under the Additional Properties heading.
   g. Click Advanced Connection Pool Properties under the Additional Properties heading.
2. Advanced connection pool settings such as partitioning, protection from connection overloading and dealing with connections that are not-responding. Default values are supplied, but tuning these properties is likely to provide improved performance and is recommended.
3. After changing any desired properties click **OK**.

   Related reference

   "Connection pool advanced settings"
   You can change advanced settings for the connection pool on the Advanced Connection Pool Settings panel of the administrative console.

Connection pool advanced settings:

You can change advanced settings for the connection pool on the Advanced Connection Pool Settings panel of the administrative console.

Purpose

The Advanced Connection Pool Settings panel is used to assign values for connection pools. On this panel you can alter properties such as partitioning, avoiding connection overloading and dealing with connections that are not responding.

Configuration Tab

Number of shared partitions

Specifies the number of partitions that are created in each of the shared pools.

Data type	Integer
Default value	0
Range	0 to max int

Number of free pool partitions

Specifies the number of partitions that are created in each of the free pools.

Data type	Integer
Default value	0
Range	0 to max int

Free pool distribution table size

The free pool distribution table size is used for better distribution of the Subject and CRI hash values within a hash table to minimize collisions for faster retrieval of a matching free connection.

If there are many incoming requests with varying credentials, this value can help with the distribution of finding a free pool for a connection for that user. Larger values are more common for installations that have many different credentials accessing the resource. Smaller values (1) should be used if the same credentials apply to all incoming requests for the resource.

Data type	Integer
Default value	0
Range	0 to max int

Surge Threshold

Specifies the number of connections created before surge protection is activated.

Surge protection is designed to prevent overloading of a data source when too many connections are created at the same time. Surge protection is controlled by two properties, surge threshold and surge creation interval.

The surge threshold property specifies the number of connections created before surge protection is activated. After you reach the specified number of connections, you enter surge mode.

The surge creation interval property specifies the amount of time, in seconds, between the creation of connections when in surge mode.

For example, assume the follow settings:

- maxConnections = 50
- surgeThreshold = 10

- surgeCreationInterval = 30 seconds

If the connection pool receives 15 connection requests, 10 connections are created at about the same time. The 11th connection is created 30 seconds after the first 10 connections. The 12th connection is created 30 seconds after the 11th connection. Connections continue to be created every 30 seconds until there are no more new connections needed or you reach the maxConnections value.

Surge connection support starts if the surge threshold is > -1 and the surge creation interval is > 0. The surge threshold property has a default value of -1, which indicates that it is turned off.

wsadmin example

```
$AdminControl getAttribute $objectname surgeCreationInterval
$AdminControl setAttribute $objectname surgeCreationInterval 30
$AdminControl getAttribute $objectname surgeThreshold
$AdminControl setAttribute $objectname surgeThreshold 15
```

Data type	Integer
Default value	-1
Range	-1 to max int

Surge creation interval

Specifies the amount of time between connection creates when you are in surge protection mode.

If the number of connections specified in the surge threshold property have been made, each request for a new connection must wait to be created on the surge creation interval. This property has a default value of 20, which indicates that at least 20 seconds should pass between connections being created. Valid values for this property are any positive integer.

Data type	Integer
Default value	20
Range	0 to max int

Stuck timer time

A stuck connection is an active connection that is not responding or returning to the connection pool. If the pool appears to be stuck (you have reached the stuck threshold), a resource exception is given to all new connection requests until the pool is unstuck. The stuck timer time property is the interval for the timer. This is how often the connection pool checks for stuck connections. The default value is 5 seconds.

If an attempt to change the stuck time, stuck timer time, or stuck threshold properties using the **wsadmin** scripting tool fails, an IllegalState exception occurs. The pool cannot have any active requests or active connections during this request. For the stuck connection support to start, all three stuck property values must be greater than 0 and maximum connections must be greater than 0.

Also, the stuck timer time, if it is set, must be less than the stuck time value. In fact, it is suggested that the stuck timer time should be one-quarter to one-sixth the value of stuck time so that the connection pool checks for stuck connections 4 to 6 times before a connection is declared stuck. This reduces the likelihood of false positives.

wsadmin example

```
$AdminControl getAttribute $objectname stuckTime
$AdminControl setAttribute $objectname stuckTime 30
$AdminControl getAttribute $objectname stuckTimerTime
$AdminControl setAttribute $objectname stuckTimerTime 15
$AdminControl getAttribute $objectname stuckThreshold
$AdminControl setAttribute $objectname stuckThreshold 10
```

Data type	Integer
Default value	5
Range	0 to max int

Stuck time

A stuck connection is an active connection that is not responding or returning to the connection pool. If the pool appears to be stuck (you have reached the stuck threshold), a resource exception is given to all new connection requests until the pool is unstuck. The stuck time property is the interval, in seconds, allowed for a single active connection to be in use to the back-end resource before it is considered to be stuck.

Data type	Integer
Default value	0
Range	0 to max int

Stuck Threshold

A stuck connection is an active connection that is not responding or returning to the connection pool. If the pool appears to be stuck (you have reached the stuck threshold), a resource exception is given to all new connection requests until the pool is unstuck. An application can explicitly catch this exception and continue processing. The pool will continue to periodically check for stuck connections when the number of stuck connections is past the threshold. If the number of stuck connections drops below the stuck threshold, the pool will detect this during its periodic checks and enable the pool to begin servicing requests again. The stuck threshold is the number of connections that need to be considered stuck for the pool to be in stuck mode.

Data type	Integer
Default value	0
Range	0 to max int

Configuring connection factories for WebSphere Adapters within applications:

WebSphere Adapters can be standalone or embedded within applications. The process of configuring the connection factories depends whether the adapter is within the application. When a WebSphere Adapter is embedded in the application use the following instructions to configure your connection factory.

You must have the necessary permissions to edit the settings on the administrative console in order to perform this task.

1. Indicate the EAR file which contains the RAR file that you wish to upload and install.

   From the top level of the administrative console follow these steps:

   a. Expand **Applications**.

   b. Select **Install New Application**.

   c. Browse to the EAR file on the local or remote system.

      Select the radio button associated with the remote or local file system, and then specify a path to the EAR file.

   d. Click Next.

2. Install the application indicating that you wish to map to a J2C connection factory.

   Select resource ref mapping to a J2C connection factory and click Next.

3. After the application installs, create and configure a connection factory for the WebSphere Adapter embedded in the newly installed application.

   From the top level of the administrative console follow these steps:

   a. Expand **Applications**.
   b. Select **Enterprise Applications**.
   c. Select the newly installed application.
   d. Click **Connector Modules** under Related Items.
   e. Select the RAR file.
   f. Under Additional Properties, select **Resource Adapters**.
   g. Click **J2C Connection Factories** under Additional Properties.
   h. Click **New**.
   i. Specify any necessary general properties.
   j. Optional: Specify an Authentication alias for XA recovery, or use a component-managed authentication alias. This field is only displayed for resources that support XA transactions.
   k. Optional: Select a Component-managed authentication alias.

      If a desired alias is not available, or you want to define a different one,

      1) Click **Apply**.
      2) Click **J2C Authentication Data Entries** under Related Items.
      3) Click **New**.
      4) Specify general properties.
      5) Click **OK** to return to the J2C Connection Factories Settings panel.

      Select or define an alias, if any components of your application obtain connections from this connection factory with the empty-argument getConnetcion() method and with Application or Per connection factory authentication specified in the resource reference.

   l. Complete the creation of your new connection factory. Click **OK**.

4. **Optional:** Change any connection pool properties of your new connection factory.

   From the J2C Connection Factory Collection panel:

   a. Select the connection factory that you just created
   b. Open the Connection Pool settings panel Click **Connection pool properties** under the Additional Properties heading.
   c. Change the values of any properties by clicking on the property name.
   d. Confirm these changes to the connection pool settings. Click **OK**.

5. **Optional:** Change any custom properties

   From the Connection Factory Settings panel of the new connection factory:

   a. Click **Custom Properties** under the Additional Properties heading.
   b. Click any property name to change its value.

Note: UserName and Password, if present, are overridden by the component-managed authentication alias you specified in a previous step.

c. Click **Save**.

J2C connection factory collection:

The J2C Connection factory collection panel provides a selectable list of connection factories.

Purpose

Use this panel to select a connection factory, which represents one set of connection configuration values.

Application components such as enterprise beans have resource reference descriptors that refer to the connection factory, not the WebSphere Adapter. The connection factory is really a configuration properties list holder. In addition to the arbitrary set of configuration properties defined by the vendor of the WebSphere Adapter, there are several standard configuration properties that apply to the connection factory. These standard properties are used by the Java 2 Connectors connection pool manager in the application server run time and are not known by the vendor-supplied WebSphere Adapter code.

Connection Factory collection panel

Name

Specifies a list of the connection factory display names.

Data type	String

JNDI Name

Specifies the Java Naming and Directory Interface (JNDI) name of this connection factory.

Data type	String

Description

Specifies a text description of this connection factory.

Data type	String

Category

Specifies a string that you can use to classify or group this connection factory.

Data type	String

J2C connection factory settings:

You specify settings for various properties of a connection factory on the J2C Connection Factory Settings panel of the administrative console.

Purpose

The J2C Connection Factory Settings panel is used to assign general property values for the selected connection factory. You can edit properties such as time-outs, purge policies and connection limits.

Configuration tab

Connection Factory Interface

Specifies the fully qualified name of the Connection Factory Interfaces supported by the resource adapter.

This is a required property. For new objects, the list of available classes is provided by the resource adapter in a drop-down list. After you create the connection factory, the field is a read only text field.

Data type	Drop-down list or text

Authentication Alias for XA recovery

This optional field is used to specify the authentication alias that should be used during XA recovery processing.

If the WebSphere Adapter does not support XA transactions, then this field will not be displayed. The default value will come from the selected alias for application authentication (if specified).

Use Component-managed Authentication Alias

Selecting this radio button specifies that the alias set for component-managed authentication is used at XA recovery time.

Data type	Radio button

Specify:

Selecting this radio button enables you to choose an authentication alias from a drop-down list of configured aliases.

Data type	Radio button

Name

Specifies a list of connection factory display names.

This is a required property.

Data type	String

JNDI Name

Specifies the JNDI name of this connection factory.

For example, the name could be *eis/myECIConnection*.

After you set this value, save it and restart the server. You can see this string when you run the dumpNameSpace tool. This is a required property. If you do not specify a JNDI name, it is filled in by default using the Name field.

Data type	String
Default	eis/*display name*

Description

Specifies a text description of this connection factory.

Data type	String

Category

Specifies a string that you can use to classify or group this connection factory.

Data type	String

Component-managed Authentication Alias

Specifies authentication data for component-managed signon to the resource.

Choose from aliases defined under Security>JAAS Configuration> J2C Authentication Data.

To define a new alias not already appearing in the pick list:
- Click **Apply** to expose Related Items.
- Click **J2C Authentication Data Entries**.
- Define an alias.
- Click the connection factory name at the top of the J2C Authentication Data Entries page to return to the connection factory page.
- Select the alias.

Data type	Pick-list

Container-managed Authentication Alias

Specifies authentication data (a string path converted to userid and password) for container-managed signon to the resource.

Note: The container-managed authentication alias is superseded by the specification of a login configuration on the resource-reference mapping at deployment time, for components with res-auth=Container.

Choose from aliases defined under Security>JAAS Configuration> J2C Authentication Data.

To define a new alias not yet included in the list:
- Click **Apply** to expose Related Items.
- Click **J2C Authentication Data Entries**.
- Define an alias.
- Click the connection factory name at the top of the J2C Authentication Data Entries page to return to the connection factory page.
- Select the alias.

Data type	Pick-list

Authentication Preference

Specifies the authentication mechanisms defined for this connection factory.

Note: The authentication preference is superseded by the combination of the <res-auth> application component deployment descriptor setting and the specification of a login configuration on the resource-reference mapping at deployment time.

This setting specifies which of the authentication mechanisms defined for the corresponding resource adapter applies to this connection factory. Common values, depending on the capabilities of the resource adapter, are: KERBEROS, BASIC_PASSWORD, and None. If None is chosen, the application component is expected to manage authentication (<res-auth>Application</res-auth>). In this case, the user ID and password are taken from one of the following:

- The component-managed authentication alias
- UserName, Password Custom Properties
- Strings passed on the getConnection method

For example, if two authentication mechanism entries are defined for a resource adapter in the *ra.xml* document:

- <authentication-mechanism-type>BasicPassword</authentication-mechanism-type>
- <authentication-mechanism-type>Kerbv5</authentication-mechanism-type>

the authentication preference specifies the mechanism to use for container-managed authentication. An exception is issued during server startup if a mechanism that is not supported by the WebSphere Adapter is selected.

Data type	Pick-list
Default value	BASIC_PASSWORD

Mapping-Configuration Alias

Allows users to select from the Security > JAAS Configuration > Application Logins Configuration list.

Note: The Mapping-Configuration Alias is superseded by the specification of a login configuration on the resource-reference mapping at deployment time, for components with res-auth=Container.

The DefaultPrincipalMapping JAAS configuration maps the authentication alias to the userid and password. You may define and use other mapping configurations.

Data type	Pick-list

J2C connection factory advanced settings:

You can change advanced settings for the J2C connection factory on the J2C Connection Factory Advanced Settings panel of the administrative console.

Purpose

The J2C Connection Factory Advanced Settings panel is used to assign values for advanced properties of the connection factory. On this panel you can alter properties pertaining to the management of cached handles and logging of missing transactions.

Configuration tab

Manage cached handles

If checked, cached handles (handles held in inst vars in a bean) are tracked by the container.

Data type	Check box

Log missing transaction contexts

If checked, the container logs that there is a missing transaction context when a connection is obtained.

Data type	Check box

Configuring WebSphere Business Integration Adapters

The process of configuring and using the WebSphere Business Integration Adapters is three-fold:

1. Install the application EAR file.

   Installation of the application EAR file is accompanied by the creation of all necessary artifacts for the WebSphere Business Integration Adapter to work.
2. Set up the administration of the WebSphere Business Integration Adapter.

   In order to administer a WebSphere Business Integration Adapter you must:
   a. Create a connection queue factory
   b. Create a WebSphere Business Integration Adapter resource
   c. Enable the WebSphere Business Integration Adapter service.

      Note: When you enable or disable a WebSphere Business Integration Adapter service, you must restart the server in order for the changes to take effect.
3. Manage the WebSphere Business Integration Adapter

   Use the administrative console to manage the WebSphere Business Integration Adapters.

Create the artifacts necessary for the WebSphere Business Integration Adapter to work:

To create the artifacts necessary for the WebSphere Business Integration Adapter to work, install the application EAR file.

In order to create the artifacts that you need to use a WebSphere Business Integration Adapter in WebSphere Process Server, you should follow the instructions for installing an application.

Setting up administration of a WebSphere Business Integration Adapter:

You must perform several administrative functions before you can manage a WebSphere Business Integration Adapter. The required steps are described here.

You must install the application EAR file to create the artifacts required for the WebSphere Business Integration Adapter before you perform this task.

In order to have administrative control over a WebSphere Business Integration Adapter you must first perform the following administrative functions.

1. Create a Queue Connection Factory.

   From the top level of the administrative console follow these steps:

   a. Expand **Resources**.

   b. Expand **JMS Providers**.

   c. Select **Default Messaging**.

   d. Select **JMS queue connection factory**.

      Under the JMS subheading select **JMS queue connection factory**.

   e. Create a new JMS queue connection factory.

      Click **New**.

   f. Accept all the default values with the following exceptions:

      - Name: QueueCF
      - JNDI Name: jms/QueueCF
      - Bus. Name: *your business name*

   g. Complete the creation of your new JMS queue connection factory.

      Click **OK**.

      A message window appears at the top of the JMS queue connection factory panel.

   h. Apply the changes that you have made at the local configuration level to the master configuration.

      Click **Save** in the message window.

2. Create a WebSphere Business Integration Adapter resource

   From the top level of the administrative console follow these steps:

   a. Expand **Resources**.

   b. Open the WebSphere Business Integration Adapters panel.

      Select **WebSphere Business Integration Adapters**.

   c. Create a new WebSphere Business Integration Adapter.

      Click **New**.

   d. Accept all the default values with the following exceptions:

      - Name: EISConnector
      - Queue Connection Factory JNDI Name: jms/QueueCF
      - Administration Input Queue JNDI Name: *connectorName/AdminInQueue*
      - Administration Output Queue JNDI Name: *connectorName/AdminOutQueue*

   e. Complete the creation of the WebSphere Business Integration Adapter.

      Click **OK**.

      A message window appears at the top of the WebSphere Business Integration Adapters panel.

   f. Apply the changes that you have made at the local configuration level to the master configuration.

      Click **save** in the message window.

3. Enable the WebSphere Business Integration Adapter Service.

   From the top level of the administrative console follow these steps:

   a. Expand **Servers**.

   b. Select **Application Servers**.

c. From the list of servers select a server where the WebSphere Business Integration Adapter Service is to be enabled.

   Click on the name of the server that hosts the resources of interest.

d. Select **WebSphere Business Integration Adapter Service**.

   Under the Business Integration subheading on the Configuration tab select **WebSphere Business Integration Adapter Service**.

e. Ensure that the **Enable Service at startup** check box is selected.

f. Click **OK**.

   A message window appears at the top of the WebSphere Business Integration Adapters panel.

g. Repeat steps 3c on page 23 to 3f on page 23 for each server on which the WebSphere Business Integration Adapter Service is to be enabled.

h. Apply the changes that you have made at the local configuration level to the master configuration.

   Click **save** in the message window.

Note: When you enable or disable a WebSphere Business Integration Adapter service, you must restart the server in order for the changes to take effect.

Managing the WebSphere Business Integration Adapter:

When a WebSphere Business Integration Adapter is running, it can be managed using the Manage the WebSphere Business Integration Adapter resources panel on the administrative console.

The WebSphere Business Integration Adapter must be running in order to be managed.

You can manage a WebSphere Business Integration Adapter from the administrative console. The Manage the WebSphere Business Integration Adapter resources panel allows you to choose one or more resources to manage and perform various administrative actions upon these resources.

1. Select the resource or resources to manage.

   From the top level of the administrative console follow these steps:

   a. Expand Servers.

   b. Select Application Servers.

   c. From the list of servers select the server where the resources you intend to manage reside.

      Click on the name of the server that hosts the resources of interest.

   d. Select WebSphere Business Integration Adapter Service

      Under the Business Integration subheading on the Configuration tab select WebSphere Business Integration Adapter Service.

   e. Select Manage the WebSphere Business Integration Adapter resources.

   f. From the list of resources choose those that you want to manage.

      Select the check boxes associated with the resources you intend to manage.

2. Manage the selected resources.

Click one of the command buttons to act upon the selected resources.

Command	Description
Deactivate	Changes the status of the selected resources from active to paused or inactive.
Activate	Changes the status of the selected resources from inactive to active.
Suspend	Changes the status of the selected resources from active to paused.
Resume	Changes the status of the selected resources from paused to active.
Shutdown	Changes the status of the selected resources from active to unavailable.

Generating service component definitions and the MQClientLink configuration file

Before using WebSphere Business Integration Adapters, it is necessary to generate service component definitions (SCA artifacts) and the MQClientLink configuration file (.wbia file). This is achieved with the WebSphere Business Integration Adapter Artifact Importer in the WebSphere Integration Developer environment.

The WebSphere Integration Developer uses the WebSphere Business Integration Adapter Artifact Importer to discover and import the WebSphere Business Integration Adapter Connector Configuration File and WebSphere Business Integration Adapter Business Objects directory and generate the desired service component definitions supporting the specified interaction-styles for the WebSphere Business Integration Adapter.

Note: This task is performed in WebSphere Integration Developer, and is described here only for reference. See the WebSphere Integration Developer information center for more details.

1. Obtain the necessary configuration files and Business Objects.
2. Use the WebSphere Business Integration Adapter Artifact Importer to generate the necessary service component definitions (SCA artifacts).

The service component definitions (SCA artifacts) and the WebSphere Process Server MQClientLink configuration file (.wbia file) are generated.

When an application is deployed to WebSphere Process Server, the service component definitions and the MQClientLink configuration (.wbia) file are handled automatically by the deployment tools. It is recommended that the configuration file be left in its default state, but it is possible to manually edit the file if required.

Using JMS resources of a generic provider

This topic is the entry-point into a set of topics about enabling WebSphere applications to use JMS resources provided by a generic messaging provider. The term “generic messaging provider” is used to mean messaging providers other than the WebSphere default messaging provider or WebSphere MQ. Note that WebSphere MQ is a separate product, and can only be used as a messaging provider if installed.

You can install a messaging provider other than the default messaging provider. WebSphere MQ is one choice of messaging provider, but is not installed as part of WebSphere Process Server. WebSphere applications can use the JMS 1.1 interfaces or JMS 1.0.2 interfaces to access JMS resources provided by the generic messaging provider, in addition to JMS resources provided by the default messaging provider or by WebSphere MQ (if installed).

You can use the WebSphere Process Server administrative console to administer the JMS connection factories and destinations provided by generic messaging providers.

Defining a generic messaging provider:

Use this task to define a new messaging provider to WebSphere Process Server, for use instead of the default messaging provider or instead of WebSphere MQ, if you have this installed.

Before starting this task, you should have installed and configured the messaging provider and its resources by using the tools and information provided with the messaging provider.

To define a new generic messaging provider to WebSphere Process Server, use the administrative console to complete the following steps:

1. In the navigation pane, click **JMS Providers > Generic**. This displays the existing generic messaging providers in the content pane.
2. To define a new generic messaging provider, click **New** in the content pane. Otherwise, to change the definition of an existing messaging provider, click the name of the provider. This displays the properties used to define the messaging provider in the content pane.
3. Specify the following required properties. You can specify other properties, as described in a later step.
   - Name. The name by which this messaging provider is known for administrative purposes within IBM WebSphere Application Server.
   - External initial context factory. The Java classname of the initial context factory for the JMS provider.
   - External provider URL. The JMS provider URL for external JNDI lookups.
4. **Optional:** Click **Apply**. This enables you to specify additional properties.
5. **Optional:** Specify other properties for the messaging provider. Under Additional Properties, you can use the **Custom Properties** link to specify custom properties for your initial context factory, in the form of standard javax.naming properties.
6. Click **OK**.
7. Save the changes to the master configuration.
8. To have the changed configuration take effect, stop then restart the application server.

You are now in position to configure JMS resources for the generic messaging provider.

Displaying administrative lists of generic messaging resources:

Use the WebSphere Process Server administrative console to display administrative lists of JMS resources provided by a messaging provider other than the default messaging provider or by WebSphere MQ, if you have this installed.

You can use the WebSphere Process Server administrative console to display lists of the following types of JMS resources provided by a generic messaging provider. You can use the panels displayed to select JMS resources to administer, or to create or delete JMS resources (where appropriate).

To display administrative lists of JMS resources for a generic messaging provider, complete the following general steps:
1. Start the WebSphere administrative console.
2. In the navigation pane, click **Resources > JMS Providers > Generic**.
3. If appropriate, in the content pane, change the scope of the generic messaging provider.
4. In the content pane, under Additional Resources, click the link for the type of JMS resource. This displays a list of any existing resources of the selected type. For more information about the settings panels displayed for resources, see the related reference topics.

JMS provider collection:

Use this panel to list JMS providers, or to select a JMS provider to view or change its configuration properties.

To view this administrative console page, click **Resources > JMS providers > Generic**

To view or change the properties of a JMS provider or its resources, select its name in the list displayed.

To define a new generic JMS provider, click **New**.

To act on one or more of the JMS providers listed, click the check boxes next to the names of the objects that you want to act on, then use the buttons provided.

Name The name by which this JMS provider is known for administrative purposes.

Description A description of this JMS provider for administrative purposes.

Generic JMS connection factory collection:

The JMS connection factories configured in the associated generic messaging provider for both point-to-point and publish/subscribe messaging. Use this panel to create or delete JMS connection factories, or to select a connection factory to browse or change its configuration properties.

Collection panel

This panel shows a list of the generic JMS connection factories with a summary of their configuration properties.

To view this administrative console page, use the administrative console to complete the following steps:
1. In the navigation pane, expand **Resources > JMS Providers > Generic**
2. In the content pane, click the name of the generic messaging provider that you want to support the JMS connection factory

3. Under Additional Properties, click **JMS connection factories**
4.

To define a new JMS connection factory, click **New**.

To view or change the properties of a JMS connection factory, select its name in the list displayed.

To act on one or more of the JMS connection factories listed, click the check boxes next to the names of the objects that you want to act on, then use the buttons provided.

Generic JMS connection factory settings:

Use this panel to browse or change the configuration properties of the selected JMS connection factory for use with the associated generic JMS provider. These configuration properties control how connections are created to the JMS destinations on the provider.

Purpose

A JMS connection factory is used to create connections to JMS destinations. The JMS connection factory is created by the associated JMS provider.

To view this page, use the administrative console to complete the following steps:
1. In the navigation pane, expand **Resources > JMS Providers > Generic**.
2. In the content pane, click the name of the messaging provider that you want to support the JMS connection factory.
3. If appropriate, in the content pane, change the scope of the generic messaging provider.
4. Under Additional Properties, click **JMS connection factories**.
5. Click the name of the JMS connection factory that you want to work with.

A JMS connection factory for a generic JMS provider (other than the default messaging provider or, if you have it installed: WebSphere MQ, as a JMS provider) has the following properties:

Properties

Name
The name by which this JMS connection factory is known for administrative purposes within IBM WebSphere Process Server. The name must be unique within the associated messaging provider.

Type

Whether this connection factory is for creating JMS queue destinations or JMS topic destinations.

Select one of the following options:
- **QUEUE** A JMS queue connection factory for point-to-point messaging.
- **TOPIC** A JMS topic connection factory for publish/subscribe messaging.

JNDI name

The JNDI name that is used to bind the connection factory into the WebSphere Process Server name space.

As a convention, use the fully qualified JNDI name; for example, in the form jms/Name, where Name is the logical name of the resource.

This name is used to link the platform binding information. The binding associates the resources defined by the deployment descriptor of the module to the actual (physical) resources bound into JNDI by the platform.

Description

A description of this connection factory for administrative purposes within WebSphere Process Server.

Category

A category used to classify or group this connection factory, for your WebSphere Process Server administrative records.

External JNDI name

The JNDI name that is used to bind the connection factory into the name space of the generic messaging provider.

As a convention, use the fully qualified JNDI name; for example, in the form jms/Name, where **Name** is the logical name of the resource.

This name is used to link the platform binding information. The binding associates the resources defined by the deployment descriptor of the module to the actual (physical) resources bound into JNDI by the platform.

Component-managed Authentication Alias

This alias specifies a user ID and password to be used to authenticate connection to a JMS provider for application-managed authentication.

This property provides a list of the J2C authentication data entry aliases that have been defined to WebSphere Process Server. You can select a data entry alias to be used to authenticate the creation of a new connection to the JMS provider.

If you have enabled global security for WebSphere Process Server, select the alias that specifies the user ID and password used to authenticate the creation of a new connection to the JMS provider. The use of this alias depends on the resource authentication (res-auth) setting declared in the connection factory resource reference of an application component's deployment descriptors.

Container-managed Authentication Alias

This alias specifies a user ID and password to be used to authenticate connection to a JMS provider for container-managed authentication.

This property provides a list of the J2C authentication data entry aliases that have been defined to WebSphere Process Server. You can select a data entry alias to be used to authenticate the creation of a new connection to the JMS provider.

If you have enabled global security for WebSphere Process Server, select the alias that specifies the user ID and password used to authenticate the creation of a new connection to the JMS provider. The use of this alias depends on the resource authentication (res-auth) setting declared in the connection factory resource reference of an application component's deployment descriptors.

Connection pool

Specifies an optional set of connection pool settings.

Connection pool properties are common to all J2C connectors.

The application server pools connections and sessions with the messaging provider to improve performance. You need to configure the connection and session pool properties appropriately for your applications, otherwise you may not get the connection and session behavior that you want.

Change the size of the connection pool if concurrent server-side access to the JMS resource exceeds the default value. The size of the connection pool is set on a per queue or topic basis.

Session pool

An optional set of session pool settings.

This link provides a panel of optional connection pool properties, common to all J2C connectors.

The application server pools connections and sessions with the messaging provider to improve performance. You need to configure the connection and session pool properties appropriately for your applications, otherwise you may not get the connection and session behavior that you want.

Custom properties

An optional set of name and value pairs for custom properties passed to the messaging provider.

Container-managed Authentication Alias

This alias specifies a user ID and password to be used to authenticate connection to a JMS provider for container-managed authentication.

This property provides a list of the J2C authentication data entry aliases that have been defined to WebSphere Process Server. You can select a data entry alias to be used to authenticate the creation of a new connection to the JMS provider.

If you have enabled global security for WebSphere Process Server, select the alias that specifies the user ID and password used to authenticate the creation of a new connection to the JMS provider. The use of this alias depends on the resource authentication (res-auth) setting declared in the connection factory resource reference of an application component's deployment descriptors.

Container-managed Authentication Alias

This alias specifies a user ID and password to be used to authenticate connection to a JMS provider for container-managed authentication.

This property provides a list of the J2C authentication data entry aliases that have been defined to WebSphere Process Server. You can select a data entry alias to be used to authenticate the creation of a new connection to the JMS provider.

If you have enabled global security for WebSphere Process Server, select the alias that specifies the user ID and password used to authenticate the creation of a new connection to the JMS provider. The use of this alias depends on the resource authentication (res-auth) setting declared in the connection factory resource reference of an application component's deployment descriptors.

Configuration tab

Scope

Specifies the level to which this resource definition is visible to applications.

Resources such as messaging providers, namespace bindings, or shared libraries can be defined at multiple scopes, with resources defined at more specific scopes overriding duplicates which are defined at more general scopes.

The scope displayed is for information only, and cannot be changed on this panel. If you want to browse or change this resource (or other resources) at a different scope, change the scope on the messaging provider settings panel, then click Apply, before clicking the link for the type of resource.

Mapping-Configuration Alias

The module used to map authentication aliases.

his field provides a list of the modules that have been configured on the **Global Security > JAAS Configuration > Application Logins Configuration** property.

Generic JMS destination collection:

The JMS destinations configured in the associated messaging provider for point-to-point and publish/subscribe messaging. Use this panel to create or delete JMS destinations, or to select a JMS destination to browse or change its configuration properties.

Collection panel

To view this administrative console page, use the administrative console to complete the following steps:

1. In the navigation pane, expand **Resources > JMS Providers > Generic**
2. In the content pane, click the name of the generic messaging provider that you want to support the JMS destination
3. Under Additional Properties, click **JMS destinations**
4.

To define a new JMS destination, click **New**.

To view or change the properties of a JMS destination, select its name in the list displayed.

To act on one or more of the JMS destinations listed, click the check boxes next to the names of the objects that you want to act on, then use the buttons provided.

Generic JMS destination settings:

Use this panel to browse or change the configuration properties of the selected JMS destination for use with the associated JMS provider.

Purpose

A JMS destination is used to configure the properties of a JMS destination for the associated generic messaging provider (not the default messaging provider or WebSphere MQ). Connections to the JMS destination are created by the associated JMS connection factory.

To view this page, use the administrative console to complete the following steps:

1. In the navigation pane, expand **Resources > JMS Providers > Generic**.

2. In the content pane, click the name of the messaging provider that you want to support the JMS destination.
3. If appropriate, in the content pane, change the scope of the generic messaging provider.
4. Under Additional Properties, click **JMS destinations**.
5. Click the name of the JMS destination that you want to work with.

A JMS destination for use with a generic messaging provider has the following properties.

Properties

Name

The name by which this qqueue is known for administrative purposes within IBM WebSphere Process Server.

Type

Whether this JMS destination is a queue (for point-to-point) or topic (for publish/subscribe).

Select one of the following options:

- **QUEUE** A JMS queue destination for point-to-point messaging.
- **TOPIC** A JMS topic destination for publish/subscribe messaging.

JNDI name

The JNDI name that is used to bind the connection factory into the WebSphere Process Server name space.

As a convention, use the fully qualified JNDI name; for example, in the form **jms/Name**, where **Name** is the logical name of the resource.

This name is used to link the platform binding information. The binding associates the resources defined by the deployment descriptor of the module to the actual (physical) resources bound into JNDI by the platform.

Description

A description of the queue, for administrative purposes.

Category

A category used to classify or group this queue, for your IBM WebSphere Process Server administrative records.

External JNDI name

The JNDI name that is used to bind the queue into the WebSphere Process Server name space.

As a convention, use the fully qualified JNDI name; for example, in the form jms/Name, where **Name** is the logical name of the resource.

This name is used to link the platform binding information. The binding associates the resources defined by the deployment descriptor of the module to the actual (physical) resources bound into JNDI by the platform.

Configuration tab

Scope

Specifies the level to which this resource definition is visible to applications.

Resources such as messaging providers, namespace bindings, or shared libraries can be defined at multiple scopes, with resources defined at more specific scopes overriding duplicates which are defined at more general scopes.

The scope displayed is for information only, and cannot be changed on this panel. If you want to browse or change this resource (or other resources) at a different scope, change the scope on the messaging provider settings panel, then click **Apply**, before clicking the link for the type of resource.

Configuring JMS resources for a generic messaging provider:

Use the following tasks to configure the JMS connection factories and destinations for a generic messaging provider (not the default messaging provider or WebSphere MQ).

You only need to complete these tasks if your WebSphere Process Server environment uses a messaging provider other than the default messaging provider or WebSphere MQ to support enterprise applications that use JMS. To enable use of such a generic messaging provider, you must have installed and configured the messaging provider, as described in Defining a new JMS provider to WebSphere Application Server.

Note: WebSphere MQ is only available as a messaging provider is you have it installed. It is not part of WebSphere Process Server.

To configure JMS resources for a generic messaging provider, complete the subsequent tasks.

Configuring a JMS connection factory, generic JMS provider:

Use this task to browse or change the properties of a JMS connection factory for use with a generic JMS provider, other than the default messaging provider or, if you have it installed, WebSphere MQ.

To configure a JMS connection factory for use with a generic JMS provider, use the administrative console to complete the following steps:

1. Display the generic messaging provider. In the navigation pane, expand **Resources > JMS Providers > Generic**.
2. **Optional:** Change the **Scope** setting to the level at which the connection factory is visible to applications.
3. In the content pane, under Additional Properties, click **JMS connection factories**. This displays a table listing any existing JMS connection factories, with a summary of their properties. .
4. To browse or change an existing JMS connection factory, click its name in the list. Otherwise, to create a new connection factory, complete the following steps:
   a. Click New in the content pane.
   b. Specify the following required properties. You can specify other properties, as described in a later step.
      - **Name** The name by which this JMS connection factory is known for administrative purposes within IBM WebSphere Process Server.
      - **Type** Select whether the connection factory is for JMS queues (QUEUE) or JMS topics (TOPIC).

- **JNDI Name** The JNDI name that is used to bind the JMS connection factory into the WebSphere Application Server name space.
- **External JNDI Name** The JNDI name that is used to bind the JMS connection factory into the name space of the messaging provider.

c. Click **Apply**. This defines the JMS connection factory to WebSphere Process Server, and enables you to browse or change additional properties.

5. **Optional:** Change properties for the JMS connection factory, according to your needs.
6. Click **OK**.
7. Save any changes to the master configuration.
8. To effect the changed configuration, stop then restart the application server.

Configuring a JMS destination, generic JMS provider:

Use this task to browse or change the properties of a JMS destination for use with a generic JMS provider, other than the default messaging provider or, if you have it installed, WebSphere MQ.

To configure a JMS destination for use with a generic JMS provider, use the administrative console to complete the following steps:

1. Display the generic messaging provider. In the navigation pane, expand **Resources > JMS Providers > Generic**.
2. **Optional:** Change the **Scope** setting to the level at which the connection factory is visible to applications.
3. In the content pane, under Additional Properties, click **JMS destinations**. This displays a table listing any existing JMS destinations, with a summary of their properties. .
4. To browse or change an existing JMS destination, click its name in the list. Otherwise, to create a new destination, complete the following steps:
   a. Click **New** in the content pane.
   b. Specify the following required properties. You can specify other properties, as described in a later step.
      - **Name** The name by which this JMS destination is known for administrative purposes within IBM WebSphere Process Server.
      - **Type** Select whether the destination is for JMS queues (QUEUE) or JMS topics (TOPIC).
      - **JNDI Name** The JNDI name that is used to bind the JMS destination into the WebSphere Process Server name space.
      - **External JNDI Name** The JNDI name that is used to bind the JMS destination into the name space of the messaging provider.
   c. Click **Apply**. This defines the JMS destination to WebSphere Process Server, and enables you to browse or change additional properties.
5. **Optional:** Change properties for the JMS destination, according to your needs.
6. Click **OK**.
7. Save any changes to the master configuration.
8. To effect the changed configuration, stop then restart the application server.

Installing EIS applications

An EIS application module, a service component architecture (SCA) module that follows EIS application module pattern can be deployed to either a J2SE platform or a J2EE platform.

The steps required to deploy an EIS module depend on the platform.

See the subsequent tasks for detailed information.

Deploying an EIS application module to the J2SE platform:

The EIS Module can be deployed to J2SE platform however only EIS Import will be supported.

You need to create an EIS application module with a JMS Import binding in the WebSphere Integration Development environment before commencing this task.

An EIS application module would be furnished with a JMS Import binding when you want to access EIS systems asynchronously through the use of message queues.

Deploying to the J2SE platform is the only instance where the binding implementation can be executed in the non-managed mode. The JMS Binding requires asynchronous and JNDI support, neither of which is provided by the base service component architecture or the J2SE. The J2EE Connector Architecture does not support non-managed inbound communication thus eliminating EIS Export.

When the EIS application module with the EIS Import is deployed to J2SE, in addition to the module dependencies, the WebSphere Adapter used by the import has to be specified as the dependency, in the manifest or any other form supported by SCA.

Deploying an EIS application module to the J2EE platform:

The deployment of EIS module to the J2EE platform results in an application, packaged as an EAR file deployed to the server. All the J2EE artifacts and resources are created, the application is configured and ready to be run.

You need to create an EIS module with a JMS Import binding in the WebSphere Integration Development environment before commencing this task.

The deployment to the J2EE platform creates the following J2EE artifacts and resources:

Table 6. Mapping from bindings to J2EE artifacts

Binding in the SCA module	Generated J2EE artifacts	Created J2EE resources
EIS Import	Resource References generated on the module Session EJB.	ConnectionFactory
EIS Export	Message Driven Bean, generated or deployed, depending on the listener interface supported by the Resource Adapter.	ActivationSpec

Table 6. Mapping from bindings to J2EE artifacts (continued)

Binding in the SCA module	Generated J2EE artifacts	Created J2EE resources
JMS Import	Message Driven Bean (MDB) provided by the runtime is deployed, resource references generated on the module Session EJB. Note that the MDB is only created if the import has a receive destination.	• ConnectionFactory • ActivationSpec • Destinations
JMS Export	Message Driven Bean provided by the runtime is deployed, resource references generated on the module Session EJB	• ActivationSpec • ConnectionFactory • Destinations

When the import or export defines a resource like a ConnectionFactory, the resource reference is generated into the deployment descriptor of the module Stateless Session EJB. Also, the appropriate binding is generated into the EJB binding file. The name, to which resource reference is bound, is either the value of the target attribute, if one is present, or default JNDI lookup name given to the resource, based on the module name and import name.

Upon deployment, the implementation locates the module session bean and uses it to lookup the resources.

During deployment of the application to the server, the EIS installation task will check for the existence of the element resource to which it is bound. If it does not exist, and the SCDL file specifies at least one property, the resource will be created and configured by the EIS installation task. If the resource does not exist, no action is taken, it is assumed that resource will be created before execution of the application.

When the JMS Import is deployed with a receive destination, a Message Driver Bean (MDB) is deployed. It listens for replies to requests that have been sent out. The MDB is associated (listens on) the Destination sent with the request in the JMSreplyTo header field of the JMS message. When the reply message arrives, the MDB uses its correlation ID to retrieve the callback information stored in the callback Destination and then invokes the callback object.

The installation task creates the ConnectionFactory and three destinations from the information in the import file. In addition, it creates the ActivationSpec to enable the runtime MDB to listen for replies on the receive Destination. The properties of the ActivationSpec are derived from the Destination/ConnectionFactory properties. If the JMS provider is a SIBus Resource Adapter, the SIBus Destinations corresponding to the JMS Destination are created.

When the JMS Export is deployed, a Message Driven Bean (MDB) (not the same MDB as the one deployed for JMS Import) is deployed. It listens for the incoming requests on the receive Destination and then dispatches the requests to be processed by the SCA. The installation task creates the set of resources similar to the one for JMS Import, an ActivationSpec, ConnectionFactory used for sending a reply and two Destinations. All the properties of these resources are specified in the export file. If the JMS provider is an SIBus Resource Adapter, the SIBus Destinations corresponding to JMS Destination are created.

Selector component administration overview

As businesses change, the business processes that drive them must change, too. Some of those changes may require that certain processes return different results than as originally designed without changing the design of the process. The selector component provides the framework for that flexibility.

Selector components provide a single interface to a service that may change results based on certain criteria. The selector component comprises an interface and a selector table. The selector table determines, based on a criteria, which component (named the target component) processes the request. The server returns the processing result provided by a target component to the client.

When building a business process, the solution architect identifies the need for a selector component and defines the interface and selector table the selector component uses to complete processing. The tasks involved in developing a selector component are described in the WebSphere Integration Developer information center.

Administering a selector component consists of tasks related to the selector component or tasks related to the selector table.

Displaying selector components

Displaying selector components is the first step in administering selector components. From the display you can export any or all of the selector components or display the selector tables which comprise the selector components.

You must be at the administrative console for the WebSphere Process Server to perform this task.

Perform this task to determine what selector components exist in your server.

1. Click on **Servers** to display the different server types.
2. Click on **Application servers** to expand the Application server list.
3. Click *servername* to select the server from the list server list which has the selectors to display.
4. Click **Selectors** under **Business Integration**.

   The console displays a list of all the selector components defined with each components' descriptions.

Displaying selector tables

Displaying selector tables is the first step in administering the tables. The resulting display is a list of target components from which you can alter the processing criteria, change the target component that runs for a specific criterion, add a new target component or delete a target component from the table, thereby removing a criterion.

You must be at the administrative console for the WebSphere Process Server to perform this task.

To perform other selector table-related tasks, display a selector table when you are determining the entries that comprise the table. This task begins after you display selector components.

1. Click on the selector component from the Selector Components display. The browser displays the selector tables in the selected component.

2. Click on one of the selector tables in the display. The browser displays the target components that comprise the selector table.

Changing target components

Changing target components allows you to alter selector component processing by either changing the selection criteria for a specific target component, changing the target component for a selection criteria, or changing both the selection criteria and the target component.

To perform this task, a selector table must exist.

Change a target component to alter the selection criteria or use a different target component for an entry in the selector table.

1. Display the selector table as described in "Displaying selector tables" on page 125.
2. Click the **Target ID** next to the target component to change. The system displays the Target component details panel.
3. Change the entry.

Portion of entry to change	Steps to change
Target destination	1. Click arrow next to the target component list. The system displays the eligible target components. 2. Select the target component from the list.
Selection criteria	1. Type over either the **Start Date**, **End Date** or both. The date you enter depends on the locale of your system and will be displayed according to the locale format. For the US English locale the format displayed is: • Month • Day of month • Year in YYYY format. • Time in HH:MM:SS format • Time zone
Target destination and selection criteria	1. Click arrow next to the target component list. The system displays the eligible target components. 2. Select the target component from the list. 3. Type over either the **Start Date**, **End Date** or both. The date you enter depends on the locale of your system and will be displayed according to the locale format. For the US English locale the format displayed is: • Month • Day of month • Year in YYYY format. • Time in HH:MM:SS format • Time zone

4. **Optional:** (Optional) Click the **Default** check box to make this the default target component.

   If the selection criteria does not fall within the range of any other target components, the selector component uses this target component.
5. Click **Apply** or **OK**.

To continue working in this display, click **Apply**. To return to the target component display, click **OK**.

6. Click **Save** on the target component display to save the changes to the selector table.

The selector table file now contains the updated selection criteria and target components. The selector component uses the updated selector table to process the next request received.

Adding target components

Add a target component when you need additional processing for a different selection criterion than currently exists in the selector table.

To perform this task, a selector table must exist.

Add a target component when you need additional flexibility in your business process. The new components can be added while the selector component is active.

1. Display the selector components as described in “Displaying selector components” on page 125.
2. Display the selector table as described in “Displaying selector tables” on page 125.
3. Click **New** on the selector table display. The browser displays a pre-filled target component detail panel.
4. Edit the target destination information to fit the application requirements as described in “Changing target components” on page 126.
5. Click **OK** to save the target component and return to the target component display.

The selector table now contains the new target components. The selector component uses the updated selector table to process the next request received.

Deleting target components

Deleting target components alters selector component processing by removing the entry in the selector table for a specific selection criterion.

To perform this task, a selector table must exist.

Delete a target component when the processing is no longer required for the business process. After deleting a target component, if a request comes in and it does not match any other specific selection criteria, the default criteria processes the request.

1. Display the target components as described in “Displaying selector tables” on page 125. The panel displays the selector table display.
2. Click the check box next to the target components to delete, then click **Delete**. The system updates the panel by displaying the remaining target components.
3. Click **Save**. The system saves the updated selector table with the entries representing the remaining target components.

The selector table file now contains only the remaining target components. The selector component uses the updated selector table to process the next request received.

Exporting selector components

Exporting selector components creates a file that you can then import into your development environment, thereby keeping the development artifacts synchronized with the actual production system artifacts.

Before starting this task, you should already have displayed your selector components as described in “Displaying selector components” on page 125.

Export selector components when you have made changes to the selector tables and you need to synchronize your development environment with your production environment. This task begins with the selector component display.

1. Choose the selectors to export.

   Click the check boxes next to the selectors and then click **Export**. The browser displays a list of HTML links to the selector components you chose. (This is the **Selectors to Export** panel.) Each selector has a file extension of zip.
2. Download the files.

   Click on each filename and the system prompts you to save the file. When prompted, click **OK** to place the file in your file system.

   Note: If you choose to, you can rename the file as you download it.
3. Return to the selectors display.

   Click **Back** to return to the list of selectors.

The system saves file where you specified. You can then copy it to your test system.

You will need to import this file into your WebSphere Integration Developer environment. See the information center for WebSphere Integration Developer.

Considerations for modules containing business rules and selectors

This topic contains information to consider when you install or delete modules that contain business rules and selectors

Business rules and selectors add flexibility to your modules. The added flexibility affects how you install or delete a module because the server saves business rules and selectors in a central repository.

Considerations for changing business rules or selectors

You can change business rules and selectors in your production environment without reassembling and reinstalling the affected modules. These changes are made directly to the repository and are not copied into any of the files that contain the business rules or the selectors. After making a change to business rules or selectors, export the business rules or selectors and reimport them into your development environment. If you are unfamiliar with exporting business rules and selectors, see the topics that describe those tasks in this information center.

Considerations for replacing a module containing business rules or selectors

When you replace a module that contains business rules or selectors, the server overwrites the copies of the business rules and selectors in the repository. When you replace a module, any changes that you made dynamically are lost. To prevent that loss, export the business rules and selectors used by the module, reimport them into your development environment, and rebuild the module before replacing the module on your production system.

If you have made changes to the business rules or selectors implemented by one module, other modules running in the server need the current copies of the business rules or selectors. If this is the case, you will have to configure different repositories so that the updated module has no effect on the other modules when you install that module in the server. The topic "Configuring the environment" describes configuring the databases.

Considerations for deleting a module containing business rules or selectors

When you delete a module that contains business rules or selectors from the server, the server does not remove the business rules and selectors from the repository. It keeps these artifacts because it cannot determine if another application or module requires the rules.

If you determine that there is no requirement for a business rule or selector, remove it from the repository. "Removing business rule and selector data from the repository" describes how to clear out unneeded business rules or selectors.

"Removing business rule and selector data from the repository"
When you uninstall an application that uses business rules or selectors, the server does not remove these artifacts from the repository. This task removes unneeded business rule and selector artifacts from the repository.

Removing business rule and selector data from the repository

When you uninstall an application that uses business rules or selectors, the server does not remove these artifacts from the repository. This task removes unneeded business rule and selector artifacts from the repository.

Make sure that you uninstall all copies of applications that use the business rules or selectors to be removed from all servers.

When you install an application containing business rule or selector artifacts, the server stores these artifacts in database tables so that you can dynamically update them without changing the application. This also allows other servers to share these artifacts. When you uninstall an application, the server does not automatically remove these artifacts from the database tables because the application may still be installed and running on another server. Deleting the artifacts from the database causes the other running copies of the application to fail when they try to use business rules or selectors.

To delete the unused artifacts from the database, you must do so manually after you uninstall all applications that use them. Remove artifacts using the tools supplied by the database platform of your repository. The repository is either a Cloudscape™ or a DB2® database.

1. Locate the database.

   Locating the database depends on the database platform.

Database platform	Location
Cloudscape	*WASHOME*\cloudscape\ / databases\RepositoryDB

Database platform	Location
DB2	Depends on the location configured during installation and profile creation of the server. For example, if you configured the server automatically and selected the default database name, the name of the database is WPSDB.

2. Locate the following database tables from which you will delete rows:

   BYTESTORE
   The main table that contains the business rule and selector artifacts

   BYTESTOREOVERFLOW
   The overflow table for the main table

   APPTIMESTAMP
   The installed applications that contain business rule and selector artifacts

3. Delete the artifacts for an application.

   Using the tools for your database platform, follow these steps to delete all business rule and selector artifacts for a given application:

   a. Find all of the rows in the BYTESTORE table where the **APPNAME** column is the same as the name of the application.
   b. Record the values of the primary key columns for all the rows found. The primary key columns for the BYTESTORE table are **ARTIFACTTNS**, **ARTIFACTNAME**, and **ARTIFACTTYPE**.
   c. Delete the rows found in step 3a from the BYTESTORE table.
   d. For each set of primary key values recorded in step 3b, find rows in the BYTESTOREOVERFLOW table that have the same values in the corresponding columns.

      Note: For a given set of primary key values, there may be zero, one, or more than one row in the BYTESTOREOVERFLOW table.
   e. Delete rows found in step 3d from the BYTESTOREOVERFLOW table.
   f. Delete the row in the APPTIMESTAMP table where the **APPNAME** column equals the name of the application.

You have removed the unneeded business rules and selector artifacts from the database tables.

Overview of targets

Targets provide additional flexibility by providing the capability of modifying processing by changing the target configured for a reference.

A component can call a component in another module thereby reusing existing logic to minimize the time and cost in building an application. WebSphere Process Server provides additional flexibility through targets. Targets allow an installed application to benefit from advances in processing by allowing the application to change the endpoint of a cross-module invocation, using the administrative console, without rewriting or redeploying the application.

To take advantage of the flexibility provided, you must understand how the system names the targets. The link from the calling module must connect to the correct target.

Target names

Target names are derived from how the calling component invokes the target. The names have the following format:

Invocation type
: **Name format**

Synchronous
: A name that follows the Java Naming and Directory Interface (JNDI) format, for example:

  `folder/export/fullpath_to_target/target_component_name`

Asynchronous
: A name with the format

  ```
  folder/calling_component_name/
  full_path_to_target_component/target_component_name
  ```

Multiple destinations
: This name is the same as an asynchronous invocation but the target actually sends a message to multiple destination components.

Related tasks

"Changing targets"
Changing the target of a reference provides applications with the flexibility of taking advantage of advances in components as they happen without recompiling and reinstalling the application.

Changing targets

Changing the target of a reference provides applications with the flexibility of taking advantage of advances in components as they happen without recompiling and reinstalling the application.

Before changing the target for a reference you must:

- Make sure that the name of the new target matches the name of the reference
- Make sure the new target uses the same data object type
- Know whether the module is synchronously or asynchronously invoking the target
- Know whether the reference targets a single or multiple destinations

Change the target of a reference from a module when another component with the same name as the original reference provides new or improved functionality that your module can use.

1. Stop the module that contains the reference that you are changing.
   a. Select your module in the administrative console and press **Stop**.

      This panel is located at **Applications > Enterprise Applications**. The display updates to show the application as stopped.
2. Change the target destination of the reference.

How you make the change depends on how the module invokes the target.

Type of invocation	How to change
Asynchronous	1. Display the buses on the system on which the module resides. Navigate to the panel using **Service Integration > Buses**. 2. Select the service bus for this server. The name of the bus has the format SCA.SYSTEM.*cellname*..Bus. 3. Display the targets for the bus by clicking **Destinations**. 4. Select the target by clicking on the binding identifier that represents the import that connects the calling module to the target. This identifier will contain the word import. 5. Change the **Default forward routing path** to the new target. The target must be an export link. 6. **Optional:** Change any other properties, if needed. 7. Save the change by clicking **OK**.
Synchronous	1. Display the EJB bindings on the system on which the module resides. Navigate to the panel using **Environment > Naming > Name Space Bindings**. 2. Select the EJB binding that represents the import for the target you are changing. The **Name in Name Space** field will contain the word import. 3. Change the value in the **JNDI Name** field to the name of the export that represents the new target. That name will contain the word export. 4. Save the change by clicking **OK**.

Type of invocation	How to change
Multiple destinations	1. Display the buses on the system on which the module resides. Navigate to the panel using **Service Integration > Buses**. 2. Display the targets for the bus by clicking **Destinations**. 3. Select the target by clicking on the binding identifier that represents the import that connects the calling module to the target. This identifier will contain the word import. 4. Display the list of targets by clicking **Context properties**. 5. Select the target to change by clicking on the target from the list. 6. Change the **Context value** field to the new targets. 7. Return to the **Context properties** panel by clicking **OK**. 8. Associate the import with a mediation by checking the select box next to the identifier, clicking **Mediate**, and following the prompts. 9. Save the change by clicking **OK**.

3. Save your changes. Click **Save** when prompted.

Restart the server, if necessary, and start the module and make sure the module receives the expected results.

Administering extended messaging resources: Overview

Extended messaging enables container-managed messaging. It extends the base Java Message Service (JMS) support, the Enterprise Java Bean (EJB) component model, and support for EJB 2.0 message-driven beans to allow use of the existing container-managed persistence and transactional behavior.

Extended messaging uses the bean-managed messaging implementation to provide the JMS interfaces, which ensures that both bean-managed and extended messaging use consistent JMS support. JMS usage is simplified since its support is managed by the extended messaging service.

The administrative console interface enables you to configure the resources needed by the extended messaging service and by the applications that use the service.

For a complete description of extended messaging, see the following articles in the WebSphere Business Integration Server Foundation information center:
- Extended messaging: Overview
- Using extended messaging in applications

Note: The Extended Messaging Service feature is being deprecated in WebSphere Process Server 6.0. Although you can continue to use extended messaging with new and existing applications in this release, you will need to replace these applications with ones that use standard JMS APIs, or replace them with equivalent messaging technologies.

Related tasks

"Configuring WebSphere Process Server settings" on page 16

"Configuring WebSphere Process Server settings" on page 16

"Configuring WebSphere Process Server settings" on page 16

"Configuring WebSphere Process Server settings" on page 16

"Configuring WebSphere Process Server settings" on page 16

Enabling the extended messaging service

The Extended Messaging Service provides runtime service to support container-managed messaging (extended messaging). The service can be automatically started when the application server is started, or it can be started manually.

Security role required: You must be logged in as administrator or configurator to complete this task.

1. Ensure that the administrative console is running.
2. Click **Servers > Application servers >** *server_name* **> Extended Messaging Service** to display the Extended Messaging Service page.
3. If you want to enable the Extended Messaging Service to start automatically with server startup, select the **Enable service at server startup** check box. If you want to start the service manually, ensure the check box is cleared.
4. Click **OK**.
5. When prompted, click **Save** on the console task bar to save your changes to the master repository.
6. Stop and restart the application server in order for the changes to take effect.

You can also use the Extended Messaging Service page to configure a listener port extension to handle late responses, as described in "Configuring WebSphere Process Server settings" on page 16.

Configuring listener port extensions to handle late responses

An application's listener port can be configured with an extension to handle late responses in an extended messaging environment.

Security role required: You must be logged in as administrator or configurator to complete this task.

1. Ensure you have a listener port defined and configured.
2. From the administrative console, click **Servers > Application servers >** *server_name* **> Extended Messaging Service > Listener Port Extensions**. The Listener Port Extensions page opens.
3. Click **New** to create a new listener port extension. The New Listener Port Extension page opens.
4. Select the **Enabled** check box to enable late response handling.
5. In the **Request Interval** field, either accept the default value or specify a new value. The request interval specifies how often the listener port checks for late responses.

6. In the **Request Timeout** field, either accept the default value or specify a new value. The request timeout value specifies how long the listener port waits for late responses. Responses received after that time are discarded.
7. Use the **Listener Ports** drop-down menu to specify the listener port you want to use for the extension.
8. Click **OK**.
9. When prompted, click **Save** on the console task bar to save your changes to the master repository.
10. Stop and restart the application server in order for the changes to take effect.

Managing extended messaging providers

An extended messaging provider manages the resources defined for use with container-managed messaging (extended messaging). Use the Extended Messaging Provider page in the administrative console to view the general properties for each resource and to select a resource for editing.

Extended messaging resources can be defined at a cell, node, or server scope:
- Cell scope—The most general scope. Extended messaging resources defined at the cell scope are visible from all nodes and servers, unless they have been overridden.
- Node scope—Extended messaging resources defined at the node scope override any duplicates defined at the cell scope. They are visible to all servers on the same node, unless they have been overridden at a server scope on that node.
- Server scope—Extended messaging resources defined at the server scope override any duplicate definitions defined at the cell or parent node scope. They are visible only to a specific server.

For detailed information about scopes, see the WebSphere Application Server Information Center.

Security role required: You must be logged in as administrator, operator, configurator, or monitor to complete this task.
1. From the administrative console, click **Resources > Extended Messaging Provider**. The Extended Messaging Provider page opens.
2. Click the radio button next to the appropriate scope.
3. Click **Apply**. The **Scope**, **Name**, and **Description** fields on the bottom of the page are updated to reflect the values for the selected resource provider.

You can now create, modify or delete input ports, output ports, or other custom properties for the selected extended messaging provider.

Related tasks

"Configuring WebSphere Process Server settings" on page 16

"Configuring WebSphere Process Server settings" on page 16

Adding a new input port:

A receiver bean constructed from a session bean requires an input port to define the properties for the receiving Java Message Service (JMS) destination. The input port can also provide details for selecting and handling messages and for the reply destination (when required).

Note: Receiver beans that are constructed from message-driven beans do not require an input port; the details they contain are associated with the deployed message-driven bean and the Message Listener Service.

During this task, you configure the initial properties of the input port. You can later change the properties of the port as needed.

Security role required: You must be logged in as administrator or configurator to complete this task.

1. From the administrative console, click **Resources > Extended Messaging Provider**. The Extended Messaging Provider page opens.
2. Select the scope for the resource provider to which you want to add the new port.
3. Click **Apply**.
4. Click **Input Ports** from the Additional Properties table. The Input Port page opens.
5. Click **New**. The Input Port configuration page opens.
6. Specify the appropriate properties for the new input port. See "Overview of administering WebSphere Process Server" on page 1 for detailed information about property settings.
7. Click **OK**.
8. When prompted, click **Save** on the console task bar to save your changes to the master repository.
9. Stop and restart the application server in order for the changes to take effect.

   Related tasks

   "Configuring WebSphere Process Server settings" on page 16

Modifying the configuration of an input port:

You can modify the configuration of an existing input port as needed. Input ports define the properties for the receiving Java Message Service (JMS) destination. They also provide details for selecting and handling messages and for the reply destination (when required).

Security role required: You must be logged in as administrator or configurator to complete this task.

1. From the administrative console, click **Resources > Extended Messaging Provider**. The Extended Messaging Provider page opens.
2. Select the scope for the resource provider to which you want to add the new port.
3. Click **Apply**.
4. Click **Input Ports** from the Additional Properties table. The Input Port page opens.
5. Select the input port that you want to modify. The Input Port configuration page opens. It displays the current configuration properties for that port.
6. Modify the properties for the input port. See "Overview of administering WebSphere Process Server" on page 1 for detailed information about property settings.
7. Click **OK**.
8. When prompted, click **Save** on the console task bar to save your changes to the master repository.

9. Stop and restart the application server in order for the changes to take effect.

Input port settings: An input port has the following configuration properties:

Scope The scope at which the extended messaging provider is defined. The value represents the location of the configuration file.

Name The name of the input port, used for administrative purposes. The value for this field must be a string.

JNDI Name
The Java Naming and Directory Interface (JNDI) name for the resource. The value for this field must be a string.

Description
A description of the input port, used for administrative purposes. The value for this field must be a string.

This field is optional.

Category
A category string to use when classifying or grouping the resource. The value for this field must a a string between 1 and 30 ASCII characters in length.

This field is optional.

JMS Connection Factory JNDI Name
The JNDI name for the Java Message Service (JMS) connection factory used by the input port (for example, jms/connFactory1). The value for this field must be a string.

JMS Destination JNDI Name
The JNDI name for the JMS destination used by the input port (for example, jms/destn1). The value for this field must be a string.

JMS Acknowledgement Mode
The JMS mode that is used to acknowledge messages. This property is used only for message-driven beans that use bean-managed transaction demarcation (in other words, the transaction type is set to `Bean`).

The valid values for this field are as follows:

- Auto Acknowledge—The session automatically acknowledges a message in either of the following cases:
  - When the session has successfully returned from a call to receive a message
  - When the session has called a message listener to process the message and received a successful response from that listener
- Dups OK Acknowledge—The session acknowledges only the delivery of messages. This can result in the delivery of duplicate messages if JMS fails.

The default mode is Auto Acknowledge.

Destination Type
The JMS resource type. The valid values for this field are as follows:

- Queue—The receiver bean receives messages from a queue destination
- Topic—The receiver bean receives messages from a topic destination

The default value is Queue.

Subscription Durability

Specifies whether a JMS topic subscription is durable. The valid values for this field are as follows:

- Durable—A subscriber registers a durable subscription with a unique identity that is retained by JMS. Subsequent subscriber objects with the same identity resume the subscription in the state in which it was left by the previous subscriber. If there is no active subscriber for a durable subscription, JMS retains the subscription's messages until they are received or they expire.
- Nondurable—Nondurable subscriptions last for the lifetime of their subscriber. A client sees the messages published on a topic only while its subscriber is active. If the subscriber is inactive, the client misses the messages published on its topic.

The default value is Durable.

This field is required only if the JMS destination type is a topic.

Reply JMS Connection Factory JNDI Name

The JNDI name of the JMS connection factory that is used for replies. The value for this field must be a string.

Reply JMS Destination JNDI Name

The JNDI name of the JMS destination that is used for replies. The value for this field must be a string.

Adding a new output port:

An output port specifies the properties needed by sender beans to define the destination for the message being sent. It also specifies optional properties if a response is expected. The output port is associated with the sender bean at deployment time.

During this task, you are configuring the initial properties of the output port. You can modify these properties later, as needed.

Security role required: You must be logged in as administrator or configurator to complete this task.

1. From the administrative console, click **Resources > Extended Messaging Provider**. The Extended Messaging Provider page opens.
2. Select the scope for the resource provider to which you want to add the new port.
3. Click **Apply**.
4. Click **Output Ports** from the Additional Properties table. The Output Port page opens.
5. Click **New**. The Output Port configuration page opens.
6. Specify the appropriate properties for the new output port. See "Overview of administering WebSphere Process Server" on page 1 for detailed information about property settings.
7. Click **OK**.
8. When prompted, click **Save** on the console task bar to save your changes to the master repository.
9. Stop and restart the application server in order for the changes to take effect.

Related tasks

"Configuring WebSphere Process Server settings" on page 16

Modifying the configuration of an output port:

You can modify the configuration of an existing output port as needed. Output ports define the destination for the message being sent and provide the destination if a response is expected.

Security role required: You must be logged in as administrator or configurator to complete this task.

1. From the administrative console, click **Resources > Extended Messaging Provider**. The Extended Messaging Provider page opens.
2. Select the scope for the resource provider to which you want to add the new port.
3. Click **Apply**.
4. Click **Output Ports** from the Additional Properties table. The Output Port page opens.
5. Select the output port that you want to modify. T he Output Port configuration page opens. It displays the current configuration properties for that port.
6. Modify the properties for the output port. See "Overview of administering WebSphere Process Server" on page 1 for detailed information about property settings.
7. Click **OK**.
8. When prompted, click **Save** on the console task bar to save your changes to the master repository.
9. Stop and restart the application server in order for the changes to take effect.

Output port settings: An input port has the following configuration properties:

Scope This field specifies the extended messaging provider scope. The value represents the location of the configuration file.

This field cannot be edited.

Name The name of the output port, used for administrative purposes. The value for this field must be a string.

JNDI Name
The Java Naming and Directory Interface (JNDI) name for the output port. The value for this field must be a string.

Description
A description of the output port, used for administrative purposes. The value for this field must be a string.

This field is optional.

Category
A category string to use when classifying or grouping the resource. The value for this field must a a string with a maximum of 30 ASCII characters.

This field is optional.

JMS Connection Factory JNDI Name
The JNDI name for the Java Message Service (JMS) connection factory used by the output port (for example, jms/connFactory1). The value for this field must be a string.

JMS Destination JNDI Name
The JNDI name for the JMS destination used by the output port. The value for this field must be a string (for example, jms/destn1).

JMS Delivery Mode
The JMS mode that is used to deliver messages. The value must be one of the following:
- Persistent—Messages put onto the destination are persistent.
- Nonpersistent—Messages put onto the destination are not persistent.

The default value is Persistent.

JMS Priority
The message priority for the queue destination. The value must be an integer from 0 to 9. The default value is 4.

JMS Time To Live
The time, in milliseconds, a message remains in the queue. After the specified time elapses, the message expires.

The value must be an integer from 0 to *n*
- 0—Messages never time out.
- *n*—Messages time out after *n* milliseconds

The default value is 0.

JMS Disable Message I.D.
Specifies whether the system generates a JMS message ID. The value must be one of the following:
- Selected—The system does not generate JMS message IDs
- Cleared—The system generates JMS message IDs automatically

By default, JMS message IDs are generated.

JMS Disable Message Time Stamp
Specifies whether the system generates a JMS message timestamp. The value must be one of the following:
- Selected—Message timestamps are not added to sent messages
- Cleared—Message timestamps are automatically added to sent messages

By default, message timestamps are added.

Response JMS Connection Factory JNDI Name
The JNDI name of the JMS connection factory that is used for responses handled by the output port (for example, jms/connFactory1). The value for this field must be a string.

Response JMS Destination JNDI Name
The JNDI name of the JMS destination that is used for responses handled by the output port (for example, jms/destn1). The value for this field must be a string.

Administering relationships

Relationship Manager is a tool for manually manipulating relationship data to correct errors found in automated relationship management or provide more complete relationship information. In particular, it provides a facility for retrieving as well as modifying relationship instance data.

Relationship Manager allows you to configure, query, view, and perform operations on relationship runtime data, including participants and their data. You create relationship definitions with Relationship Designer. At run time, instances of the relationships are populated with the data that associates information from different applications. This relationship instance data is created when the maps or other WebSphere Process Server components run and need a relationship instance. The Relationship Service exposes a set of application programming interfaces (APIs) to retrieve relationship metadata and to create, retrieve, and manipulate the instance data. The data is stored in the relationship tables specified in the relationship definition. Relationship Manager provides a graphical user interface to interact with the relationships and relationship instances, independent of the database vendor.

For each relationship instance, Relationship Manager can display a hierarchical listing of its participants. Each participant fills a role in the relationship and has instance data, properties, and key attributes. The relationship tree also provides detailed information about each of the participants in the relationship instance, such as the type of entity, its value, and the date it was last modified. A relationship instance ID is automatically generated when the relationship instance is saved in the relationship table. Relationship Manager will display this instance ID at the top level of the relationship tree.

You can use Relationship Manager to manage entities at all levels: the relationship instance, participant instance, and attribute data and property data levels. For example, you can use Relationship Manager to:

- Create and delete relationship instances
- Modify the contents of a relationship instance, such as adding and deleting participants
- Add a participant's data (and copy and paste the data of a participant from another relationship into the relationship instance, thus creating a new participant (as long as the participant types are identical)
- Activate and deactivate participants
- Retrieve participants based on instance IDs or data
- Salvage a situation when problems arise. For example, when corrupt or inconsistent data from a source application has been sent to the generic and destination application relationship table, you can use Relationship Manager to rollback the data to a point in time when you know the data is reliable.

For more information, see the topics on the Relationship Designer in the WebSphere Integration Developer Information Center, Relationship Services, and the administrative console.

View relationship types

View information related to the relationship type, including the relationship name, display name, static or identity attributes, and roles.

You must be authenticated as a Monitor, an Operator, or an Administrator.

1. Open the Relationship Manager panel of the administrative console. Log into the WebSphere Process Server administrative console and navigate to **Integration Applications > Relationship Manager**.
2. Open the Relationships panel for the server you want to manage. Click **Relationships** next to that Relationship Services MBean.

View relationship details

View detailed information for the relationship type, including the relationship name, display name, property values, role type attributes, and static and identity attributes.

You must be authenticated as a Monitor, an Operator, or an Administrator.

1. Open the Relationship Manager panel of the administrative console. Log into the WebSphere Process Server administrative console and navigate to **Integration Applications > Relationship Manager**.
2. Open the Relationships panel for the server you want to manage. Click **Relationships** next to that Relationship Services MBean.
3. There are two methods for viewing the details of the relationships shown in the table.
   a. Click the radio button in the Select column, and click **Details**.
   b. Click the name of the relationship type.
4. To go back to the list of relationship details, click **Relationships** from the path at the top of the screen.

Query relationship instances

Perform relationship-based instance queries.

To perform this task, you must be logged in as a Monitor, an Operator, or an Administrator.

Select a query parameter (All, By ID, or By property) to retrieve all or a subset of the instance data for a relationship. The return is the result set of that query, and is displayed based on how you set the Preferences fields on the Relationship instances panel.

1. Open the Relationship Manager panel of the administrative console. Log into the WebSphere Process Server administrative console and navigate to **Integration Applications > Relationship Manager**.
2. Open the Relationships panel for the server you want to manage. Click **Relationships** next to that Relationship Services MBean.
3. Select the relationship by clicking the radio button in the Select column and click **Query**.
4. Choose one of the query options by selecting the appropriate tab.

Option	Description
All tab	Retrieve a list of all instances in the relationship. You can select to display all activated, all inactivated, or all activated and inactivated relationship instance data.
By ID tab	Retrieve relationship instances in the range of the starting and ending instance identifiers. If one field is left blank, it will return only the single instance. The query will return all of the roles for the instances it finds.
By property tab	Retrieve relationship instances by specific property values.

5. Once you have selected the query parameter, you then have the following options.
   - Click **Apply** to display the result data from the query.
   - Click **OK** to display the result data from the query.
   - Click **Reset** to clear your selections or return the entry to the most recent set of changes made.

- Click **Cancel** to discard any changes made and return you to the list of relationship types.

View relationship instance details

View detailed information for the selected relationship instance, including the relationship name, relationship instance ID, role base information, and property values.

You must be authenticated as a Monitor (read-only), an Operator, or an Administrator.

1. Open the Relationship Manager panel of the administrative console. Log into the WebSphere Process Server administrative console and navigate to **Integration Applications > Relationship Manager**.
2. Open the Relationships panel for the server you want to manage. Click **Relationships** next to that Relationship Services MBean.
3. Select the relationship by clicking the radio button in the Select column, and click **Query**.
4. Choose one of the three query options.
   - Select the All tab to retrieve a list of all instances in the relationship. You can select to display all activated, all inactivated, or all activated and inactivated relationship instance data.
   - Select the By ID tab to retrieve relationship instances in the range of the starting and ending instance identifiers. If one field is left blank, it will return only the single instance. The query will return all of the roles for the instances it finds.
   - Select the By property tab to retrieve relationship instances by specific property values.
5. Once you have selected the query option and filled in the necessary information, select either **Apply** or **OK**.
6. You can view the relationship instance details in one of two ways:
   - Click the relationship instance ID
   - Click the radio button next to the relationship instance ID, and click **Details**.

Edit relationship instance details

Edit the information for the selected relationship instance.

You must be authenticated as an Operator or an Administrator.

1. Open the Relationship Manager panel of the administrative console. Log into the WebSphere Process Server administrative console and navigate to **Integration Applications > Relationship Manager**.
2. Open the Relationships panel for the server you want to manage. Click **Relationships** next to that Relationship Services MBean.
3. Select the relationship by clicking the radio button in the Select column, and click **Query**.
4. Choose one of the three query options, fill in the query information, and select either **Apply** or **OK**.
5. You can view the relationship instance details in one of two ways:
   - Click the relationship instance ID.
   - Click the radio button next to the relationship instance ID, and click **Details**.
6. Changes the values under Property values as necessary.

7. When you are finished making changes, you have the following options.
   - Click **Apply** to save the changes locally.
   - Click **OK** to save the changes locally.
   - Click **Reset** to clear your changes or return the entry to the most recent set of changes made.
   - Click **Cancel** to discard any changes made and return you to the list of relationship instances.
8. Click **Apply Changes** to save any changes made to this point in the database.

Create new relationship instance

Create a relationship instance.

You must be authenticated as an Operator or an Administrator.

1. Open the Relationship Manager panel of the administrative console. Log into the WebSphere Process Server administrative console and navigate to **Integration Applications > Relationship Manager**.
2. Open the Relationships panel for the server you want to manage. Click **Relationships** next to that Relationship Services MBean.
3. Select the relationship by clicking the radio button in the Select column, and click **Query**.
4. Choose one of the three query options.
5. Once you have selected a query option and filled in the necessary information, select either **Apply** or **OK**.
6. Click **Create**.
7. Add the value information for the Property values, if you want something other than the default values, and click either **Apply** or **OK** to save the new relationship instance locally.
8. You must also create a role instance for the relationship instance, as you cannot have a relationship instance without a role instance.
9. Once you have created both a role instance and a relationship instance, click **Apply Changes** to save any changes made to this point in the database.

Delete relationship instance

Delete a selected relationship instance.

You must be authenticated as an Operator or an Administrator.

1. Open the Relationship Manager panel of the administrative console. Log into the WebSphere Process Server administrative console and navigate to **Integration Applications > Relationship Manager**.
2. Open the Relationships panel for the server you want to manage. Click **Relationships** next to that Relationship Services MBean.
3. Select the relationship by clicking the radio button in the Select column, and click **Query**.
4. Choose one of the three query options, fill in the necessary information, and select either **Apply** or **OK**.
5. Click the radio button next to the Relationship instance ID you want to delete.
6. Click **Delete** to delete the relationship instance locally.
7. Click **Apply Changes** to save any changes made to this point in the database.

Roll back relationship instance data

Perform a rollback of the instance data for a relationship to a specified date and time.

You must be authenticated as an Operator or an Administrator.

1. Open the Relationship Manager panel of the administrative console. Log into the WebSphere Process Server administrative console and navigate to **Integration Applications > Relationship Manager**.
2. Open the Relationships panel for the server you want to manage. Click **Relationships** next to that Relationship Services MBean.
3. Click the radio button next to the specific relationship.
4. Click **Rollback**.
5. Select the time period for the rollback by filling in the From date and To date fields.
6. Click **Apply** or **OK**.

View role types associated with a relationship

View descriptions of the roles for a relationship, including the role name, display name, key (for the business object), role object type (business object name), and managed attribute setting.

You must be authenticated as a Monitor, an Operator, or an Administrator.

1. Open the Relationship Manager panel of the administrative console. Log into the WebSphere Process Server administrative console and navigate to **Integration Applications > Relationship Manager**.
2. Open the Relationships panel for the server you want to manage. Click **Relationships** next to that Relationship Services MBean.
3. Click on the role names in the Roles column of the relationship.

   Note: You can set viewing options for the table by clicking **Preferences** and entering the maximum number of rows to be displayed when the collection is large and entering the maximum number of characters viewable in a truncated column. To save the search criteria for this display, click **Retain** filter criteria and click **Apply** (or **Reset**, as necessary). When you return to this panel, this information will be displayed.

Query relationship instances based on roles

Perform different role-based instance queries.

You must be authenticated as a Monitor, an Operator, or an Administrator.

Select a query parameter (By key, By date, By property, or Advanced) to retrieve all or a subset of the instance data for a relationship based on the chosen role. The return is the result set of that query, and is displayed based on how you set the Preferences fields on the Relationship instances panel.

1. Open the Relationship Manager panel of the administrative console. Log into the WebSphere Process Server administrative console and navigate to **Integration Applications > Relationship Manager**.
2. Open the Relationships panel for the server you want to manage. Click **Relationships** next to that Relationship Services MBean.

3. Click on the role names in the Roles column of the relationship.
4. Click the radio button next to the Role name, and click **Query**.
5. Choose one of the query options by selecting the appropriate tab.

Option	Description
By key tab	Retrieve relationship instances based on values for key attributes of selected participants. To retrieve this information, fill in the value of the ID and click **Apply** or **OK**.
By date tab	Retrieve relationship instances with roles that were created or modified between the specified dates. To retrieve this information, fill in the From date and To date fields with the desired dates, and click **OK** or **Apply**.
By property tab	Retrieve relationship instances by specific role property values. To retrieve this information, perform these steps: 1. Select the role property name and type parameters from the drop-down list. 2. Enter the value parameter for the specified role property in the Property value field, and click **OK** or **Apply**.
Advanced tab	Request a more refined search by combining fields from the other query panels. Fill in the information as described in the other options, and click OK or Apply.

Note: You also have the following choices on each tab:
- Click Reset to clear your selections or return the entry to the most recent set of changes made.
- Click Cancel to discard any changes made and return you to the list of roles.

View role details

View the detailed information for the role, including the relationship name, role name, display name, property values, keys, role object type, and managed attribute setting.

You must be authenticated as a Monitor, an Operator, or an Administrator.

1. Open the Relationship Manager panel of the administrative console. Log into the WebSphere Process Server administrative console and navigate to **Integration Applications > Relationship Manager**.
2. Open the Relationships panel for the server you want to manage. Click **Relationships** next to that Relationship Services MBean.
3. Click on the role names in the Roles column of the relationship.
4. There are two options for viewing the role details.

- Click the radio button next to the role in the Select column, and click **Details**.
- Click on the specific role name.

View role instance details

View detailed information for the selected role instance, including the role name, role element, key attributes, property values, status, and logical state.

You must be authenticated as a Monitor, an Operator, or an Administrator.

1. Open the Relationship Manager panel of the administrative console. Log into the WebSphere Process Server administrative console and navigate to **Integration Applications > Relationship Manager**.
2. Open the Relationships panel for the server you want to manage. Click **Relationships** next to that Relationship Services MBean.
3. Select the relationship by clicking the radio button in the Select column, and click **Query**.
4. Choose one of the query options, and select Apply or OK.
5. Click on the role names in the Roles instances column of the relationship.
6. There are two options for viewing the role instances.
   - Click the radio button next to the role in the Select column, and click **Instances**.
   - Click on the role name.
7. There are two options for viewing the instance details.
   - Click the radio button next to the role instance in the Select column, and click **Details**.
   - Click on the role instance name.

Create role instance

Create a new role instance for a relationship.

You must be authenticated as an Operator or an Administrator.

Note: Additional role instances cannot be added to identity relationship instances.

1. Open the Relationship Manager panel of the administrative console. Log into the WebSphere Process Server administrative console and navigate to **Integration Applications > Relationship Manager**.
2. Open the Relationships panel for the server you want to manage. Click **Relationships** next to that Relationship Services MBean.
3. Select the relationship by clicking the radio button in the Select column, and click **Query**.
4. Choose one of the query options, fill in the information, and select **Apply** or **OK**.
5. Click on the role names in the Roles instances column of the relationship.
6. Click the radio button next to the role, and click **Create**.
7. Add the value for the ID and the value information for the property values, and click either **Apply** or **OK** to save the new role instance locally. The key value information can only be set when creating the role instance. It cannot be changed after you have applied the changes back to the database. However, the property values can be edited later.

   Note: You also have the following choices:

- Click **Reset** to clear your selections or return the entry to the most recent set of changes made.
- Click **Cancel** to discard any changes made and return you to the list of role instances.

8. Click **Apply Changes** to save any changes made to this point in the database.

Delete role instance

Delete a selected role instance of a relationship.

You must be authenticated as an Operator or an Administrator.

1. Open the Relationship Manager panel of the administrative console. Log into the WebSphere Process Server administrative console and navigate to **Integration Applications > Relationship Manager**.
2. Open the Relationships panel for the server you want to manage. Click **Relationships** next to that Relationship Services MBean.
3. Select the relationship by clicking the radio button in the Select column, and click **Query**.
4. Choose one of the query options, fill in the information, and select **Apply** or **OK**.
5. Click on the role names in the Roles instances column of the relationship.
6. Select the radio button in the Select column, and click **Instances**.
7. Click the radio button in the Select column, and click **Delete** to delete the role instance locally.
8. Click **Apply Changes** to save any changes made to this point in the database.

Edit role instance properties

Edit the information for the selected role instance.

You must be authenticated as an Operator or an Administrator.

1. Open the Relationship Manager panel of the administrative console. Log into the WebSphere Process Server administrative console and navigate to **Integration Applications > Relationship Manager**.
2. Open the Relationships panel for the server you want to manage. Click **Relationships** next to that Relationship Services MBean.
3. Select the relationship by clicking the radio button in the Select column, and click **Query**.
4. Choose one of the query options, fill in the information, and select **Apply** or **OK**.
5. Click on the role names in the Roles instances column of the relationship.
6. Click the radio button next to the role in the Select column, and click **Instances**.
7. Click the radio button next to the role instance in the Select column, and click **Details**.
8. Edit the information as necessary, and click **Apply** or **OK** to save these changes locally.

   Note: You also have the following choices:
   - Click **Reset** to clear your changes or return the entry to the most recent set of changes made.
   - Click **Cancel** to discard any changes made and return you to the list of role instances.

9. Click **Apply Changes** to save any changes made to this point in the database.

Tutorial: Relationship manager administration

This tutorial demonstrates the basic functions of the WebSphere Process Server Relationship Manager. Relationships are used to correlate identifiers from different environments for the same item of data. For example, in one environment, states are identified by two-letter abbreviations (AZ, TX). In another environment, different abbreviations are used (Ariz., Tex.). A relationship would be created to correlate ″AZ″ in the first environment to ″Ariz″ in the second environment.

The sample relationship referenced here correlates customer IDs. Many business applications maintain databases of customers, and most of these applications assign their own ID to each customer. In an enterprise environment, the same customer likely has a different ID in each business application. In this tutorial, a relationship is defined to correlate customer IDs. The relationship name is ″SampleCustID″. There are two roles defined for this relationship. One role is for the Customer Information System (CIS) and the other role is for the General Ledger (GL) application. This relationship was created by the Relationship Services sample along with the roles and a small amount of sample data.

The Relationship Manager is designed to add, modify, and remove role instances of a relationship instance as well as add, modify, and remove relationship instances. WebSphere Integration Developer should be used to create and deploy new relationship definitions. The definitions are stored in schema files or J2EE applications which are deployed to a particular WebSphere Integration Developer server.

Objectives of this tutorial

After completing this tutorial, you will be able to change the values of relationship instances.

Time required to complete this tutorial

This tutorial requires approximately 10 minutes to complete.

Prerequisites

This tutorial uses a relationship that is created by the Relationship Services Technical sample. Before following the steps of this tutorial, go to the samples gallery and perform the steps described in the Relationship Services sample to create the required relationship and roles.

Changing the values of a relationship instance

One of your customers has a customer ID of A004 in your CIS application. This same customer has a customer ID of 801 in your GL application. However, due to a data entry error, the relationship instance that correlates the customer IDs of this customer currently has a value of 901 instead of 801 for the GL customer ID. This tutorial takes you through the steps to correct this entry in the relationship.

1. Open the Websphere Process Server administrative console.
2. If security is enabled, log on as a user with Administrator privileges.
3. In the left menu bar, navigate to **Integration Applications > Relationship Manager**.

4. Under Relationships, click **Relationships** next to the name of the server for which you are administering.

   A relationship named SampleCustID should be visible.
5. Locate the Roles column, and click the link in this column for the SampleCustID relationship.

   Three roles should be available: MyCISCustomer_0, MyGLCustomer_0, and GenericCustomer_0.
6. Search for a particular value on the MyGLCustomer_0 role to locate the relationship that must be modified. Select the radio button next to MyGLCustomer_0, and click **Query**.
7. Under the By Key tab, enter the value **901** in the Value field under Key attributes and click **Apply**. This locates the relationship instance for the requested customer.
8. Click the relationship instance ID. This screen shows the actual relationship data for customer ID 901 in the GL application. Since a query was done against a particular role, only data for that role was returned.
9. Click the MyGLCustomer_0 link in the Name column. This displays all of the roles that match the query for this relationship instance.

   Note: The role should have customerNumber=901 and no property values. If anything else is shown, you need to go look at the role instance and record any data you want to keep.
10. Click the radio button next to the role instance, and click **Delete**.
11. Click **Create** to create a new role instance for this relationship instance.
12. Enter **1001** as the Value for customerNumber, and click **OK**. You should see a new unsaved instance in the table.
13. Click **Apply Changes** to save all the updates into the relationship database tables and return the changed instance.

You now have the correct customer ID value in the relationship instance for the GL application.

Managing WebSphere Process Server failed events

What is a failed event?

In the context of WebSphere Process Server, an event is a request that is received by a WebSphere Process Server application. It can come from an external source (such as an inbound application adapter) or an external invocation to a web service. The event is comprised of a reference to the business logic it wants to operate and its data, stored in a Service Data Object (a business object). When an event is received, it is processed by the appropriate WebSphere Process Server application business logic.

A single thread of execution can branch off into multiple branches (or threads); the individual branches are linked to the main invoking event by the same session context.

If this business logic in one of these branches cannot execute completely due to system failure, component failure, or component unavailability, the event moves into the failed state. If multiple branches fail, a failed event is created for each. The WebSphere Process Server Recovery subsystem handles the following types of failed events:

- Event failures that occur during an asynchronous invocation of a Service Component Architecture (SCA) operation
- Event failures that are caused by a runtime exception (in other words, any exception that is not declared in the methods used by the business logic)

The Recovery subsystem collects these types of failed events and makes them available for administrative purposes through the failed event manager interface.

How are failed events managed?

An administrator uses the failed event manager available in the administrative console to browse and manage all WebSphere Process Server failed events. Failed events can be resubmitted or deleted from the system.

Common tasks for managing failed events include:
- Browsing all failed events
- Searching for failed events by specific criteria
- Editing data for a failed event
- Resubmitting failed events
- Deleting failed events

To access the failed event manager, click **Integration Applications > Failed Event Manager**.

Related concepts

"Getting started with the administrative console" on page 5

"Overview of administering WebSphere Process Server" on page 1

Related tasks

Finding failed events

Deleting failed events

Role-based access for failed event manager

The failed event manager uses role-based access control to the failed event data and tasks. Only the administrator and operator roles are authorized to perform tasks within the failed event manager. Users logged in as either administrator or operator can view all data associated with failed events and can perform all tasks.

Note: This security infrastructure is inherited from the base WebSphere Application Server product. For more information about security, see the information centers for WebSphere Application Server and WebSphere Process Server.

Finding failed events

Before you can edit, resubmit, or delete failed events, you must identify them. Use the search functionality in the failed event manager to find all failed events on the server, or to find a specific subset of failed events.

This topic provides instructions for finding all failed events on the server, with references to topics for conducting other searches based on source, destination, date, business object type, exception text, or a combination of those criteria.

Security role required: You must be logged in as administrator or operator to perform this task.

1. Ensure the administrative console is running.
2. Click **Integration Applications > Failed Event Manager** to enter the failed event manager.
3. From the **Failed events on this server** box, click **Get all failed events**.

   The Search Results page opens, displaying a list of all the WebSphere Process Server failed events on the server.

   Related concepts

   "Getting started with the administrative console" on page 5

   "Overview of administering WebSphere Process Server" on page 1

   Related tasks

   "Searching for failed events by source" on page 153

   Searching for failed events by destination

   Searching for failed events by date

   Searching for failed events by business object type

   Searching for failed events by exception

   Performing an advanced search for failed events

   Deleting failed events

Searching for failed events by destination

Use the Search page's **By Destination** tab to find only those failed events that are associated with a specific destination module, component, or method.

When performing a search, note the following:

- The values for the fields are case sensitive.
- The fields accept the asterisk (*) wildcard character.
- If you leave any field on this tab blank, the blank field is treated as a wild card. The failed event manager will search in all components, modules, or methods.
- You can search on a single destination criteria or on multiple criteria. Searching on two or more of the destination criteria provides a more refined list of failed events.

Security Role Required: You must be logged in as administrator or operator to perform this task.

1. Ensure the administrative console is running, and then click **Integration Applications > Failed Event Manager** to enter the failed event manager.
2. From the main failed event manager page, click **Search by destination**.

   The Search page opens with the **By Destination** tab selected.
3. Specify the search criteria you want to use. You can use any combination of the following fields to customize your search:
   - The **Destination module** field—Use this field to specify the failed event's destination module.
   - The **Destination component** field—Use this field to specify the failed event's destination component.
   - The **Destination method** field—Use this field to specify the failed event's destination method.
4. Click **OK** to begin the search.

The Search Results page opens and displays a list of all failed events that were destined for the specified module, component, or method.

Searching for failed events by source

Use the Search page's **By Source** tab to find only those failed events that originated from a specific source module, component, or both.

When performing a search, note the following:
- The values for the fields are case sensitive.
- The fields accept the asterisk (*) wildcard character.
- If you leave either field on this tab blank, the blank field is treated as a wild card. The failed event manager will search in all components or modules.
- To get the most refined list of failed events, use both the **Source module** and **Source component** fields.

Security Role Required: You must be logged in as administrator or operator to perform this task.

1. Ensure the administrative console is running, and then click **Integration Applications > Failed Event Manager** to enter the failed event manager.
2. From the main failed event manager page, click **Search by source**.

   The Search page opens with the **By Source** tab selected.
3. Specify the search criteria. You can use one or both of the following fields:
   - The **Source module** field—Use this field to specify the module that the failed event originated from.
   - The **Source component** field—Use this field to specify the component that the failed event originated from.
4. Click **OK** to begin the search.

   The Search Results page opens and displays a list of all failed events that originated from the specified module, component, or both.

Searching for failed events by date

Use the Search page's **By Date** tab to find only those events that failed during a specific time period.

When performing a search, note the following:
- The format for the date and time are locale-specific. An example of the appropriate format is provided with each field.
- The time is always local to the server. It is not updated to reflect the local time of individual machines running the administrative console.
- If you leave either field on this tab blank, the blank field is treated as a wild card. For example, if you specify a starting date but not an ending date, the failed event manager finds failed events that occurred between the specified start date and the current date.

Security Role Required: You must be logged in as administrator or operator to perform this task.

1. Ensure the administrative console is running, and then click **Integration Applications > Failed Event Manager** to enter the failed event manager.
2. From the main failed event manager page, click **Search by date**.

   The Search page opens with the **By Date** tab selected.

3. Specify the search criteria you want to use. You can use one or both of the fields on this tab:
   - The **From Date** field—Use this field to specify the starting date and time. The **From Date** field automatically defaults to 4:00 PM on December 31, 1969, formatted appropriately for the computer's specified locale (for example, `12/31/69 4:00 PM` on a computer with a locale set for United States English).
   - The **To Date** field—Use this field to specify the ending date and time.
4. Click **OK** to begin the search.

   The Search Results page opens and displays a list of all failed events that originated during the specified time period.

Searching for failed events by business object type

Use the Search page's **By Type** tab to find only those failed events that are associated with a specific business object.

Security Role Required: You must be logged in as administrator or operator to perform this task.

1. Ensure the administrative console is running, and then click **Integration Applications > Failed Event Manager** to enter the failed event manager.
2. From the main failed event manager page, click **Search by business object type**.

   The Search page opens with the **By Type** tab selected.
3. Specify the business object type you want to search against by using one of the following:
   - The **Select the business object type** menu—Use this drop-down menu to select the type of business object associated with the failed events. This menu contains a list of all business object types found in the failed events on the server.
   - The **Other business object type** field—Use this field to specify the type of business object associated with the failed events. The field accepts the asterisk (*) wildcard character. All values are case sensitive.
4. Click **OK** to begin the search.

   The Search Results page opens and displays a list of all failed events that are associated with the specified business object type.

Searching for failed events by exception

Use the Search page's **By Exception** tab to find only those failed events that are associated with a specific exception. You can specify part or all of the exception text.

Security Role Required: You must be logged in as administrator or operator to perform this task.

1. Ensure the administrative console is running, and then click **Integration Applications > Failed Event Manager** to enter the failed event manager.
2. From the main failed event manager page, click **Search by exception text**.

   The Search page opens with the **By exception** tab selected.
3. In the **Exception text** field, type the text associated with the exception you want to search against.

You can specify all or part of the exception text, as well as the asterisk (*) wildcard character to make the search easier. The values in this field are case sensitive.

Note: If you leave the **Exception text** field blank, it is treated as a wild card; all failed events are returned.

4. Click **OK** to begin the search.

   The Search Results page opens and displays a list of all failed events that are associated with the specified exception text.

Performing an advanced search for failed events

Use the Search page's **Advanced** tab to perform a more refined search for failed events by using a combination of the criteria found on the other search tabs: source, destination, date, business object type, and exception text.

Note the following:

- Unless otherwise noted below, all fields accept the asterisk (*) wildcard character.
- Leaving a field blank causes it to be treated as a wild card.

The advanced search is not optimized; executing an advanced search on a large set of failed events can reduce performance.

Security Role Required: You must be logged in as administrator or operator to perform this task.

1. Ensure the administrative console is running, and then click **Integration Applications > Failed Event Manager** to enter the failed event manager.
2. From the main failed event manager page, click **Advanced search**.

   The Search page opens with the **Advanced** tab selected.
3. Specify the search criteria you want to use. You can use any combination of the following fields to customize your search:
   - The **Destination module** field—Use this field to specify the failed event's destination module.
   - The **Destination component** field—Use this field to specify the failed event's destination component.
   - The **Destination method** field—Use this field to specify the failed event's destination method.
   - The **Source module** field—Use this field to specify the module that the failed event originated from.
   - The **Source component** field—Use this field to specify the component that the failed event originated from.
   - The **From Date** field—Use this field to specify the starting date and time if you want to search within a specific time period. This field does not accept the asterisk (*) wildcard character.
   - The **To Date** field—Use this field to specify the ending date and time if you want to search within a specific time period. This field does not accept the asterisk (*) wildcard character.
   - The **Business object type** field—Use this field to specify the type of business object associated with the failed events.
   - The **Exception text** field—Use this field to specify the text associated with the exception you want to search against.

4. Click **OK** to begin the search.

   The Search Results page opens and displays a list of all failed events that meet the specified criteria.

Working with data in failed events

Each failed event has data associated with it; often, that data can be edited before an event is resubmitted. There are two basic types of data for a failed event: data about the event, and business data.

Data about the failed event

Each failed event has the following data associated with it:

- The unique message ID and session ID for the event
- The service invocation type between SCA components
- The names of the module and component from which the event originated (the source)
- The names of the destination module, component and method for the event
- The time the event failed
- The exception thrown when the event failed

This data cannot be edited. In addition, failed events can have associated trace and expiration data, both of which can be edited.

Business data

Events typically include business data. Business data can be encapsulated in a business object, or it can be simple data that is not part of a business object. Business data is edited with the business data editor available in the failed event manager.

Related tasks

Browsing data in failed events

Editing trace or expiration data in a failed event

“Configuring WebSphere Process Server settings” on page 16

Browsing data in failed events

Each failed event has two types of data associated with it:

- Failed event data—Information about the failed event itself, including the source and destination for the event, the time it failed, the exception it failed with, its message and session IDs, and its trace and expiration settings.
- Business data—Information contained in the event. The business data can be encapsulated in a business object, or it can be simple data that is not part of a business object.

Security role required: You must be logged as administrator or operator to perform this task.

1. Ensure that the failed event manager is open and that you have retrieved a list of the failed events on your system.
2. From the failed event manager’s Search Results page, click the ID (found in the Message ID column) of the failed event whose data you want to browse.

   The Failed Event Details page opens and displays all of the information about the event.

3. If you want to browse the business data associated with the failed event, click **Edit business data**.

   The Business Data Editor collection page opens, displaying the business data associated with the failed event. Each parameter name in the hierarchy is a link. If the parameter is a simple data type, clicking its name will open up a form so you can edit the parameter's value. If the paramter is a complex data type, clicking its name will expand the hierachy further.

   Related tasks

   Finding failed events

   Editing trace or expiration data in a failed event

   "Configuring WebSphere Process Server settings" on page 16

Editing trace or expiration data in a failed event

The Failed Event Details page enables you to set or modify values for the trace control and expiration date associated with a failed event.

Important: Any edits you make to the trace or expiration data are only saved locally until you resubmit the event. If you perform any other action before resubmitting the event, all edits are lost.

Failed events can be resubmitted with trace to help you monitor the event processing. When you view the failed event data on the Failed Event Details page, the default trace value `SCA.LOG.INFO;COMP.LOG.INFO` is shown for the event. If you resubmit the event with this default setting, no trace occurs when the session calls an SCA service or executes a component.

Some failed events also have an expiration. If a user has specified an expiration with the asynchronous call that sends the event, that data persists even if the event fails, and the expiration time appears in the **Resubmit Expiration Time** field on the Failed Event Details page. Expired failed events cannot be resubmitted successfully. To prevent a second failure, you can edit the expiration date for the event to ensure that it is not expired when it is resubmitted.

Security role required: You must be logged in as administrator or operator to perform this task.

1. Ensure that the failed event manager is open and that you have retrieved a list of the failed events on your system.
2. From the failed event manager's Search Results page, click the ID (found in the Message ID column) of the failed event whose data you want to edit.

   The Failed Event Details page opens.
3. If the event has an expiration date that causes it to expire before it is resubmitted, edit the expiration in the **Resubmit expiration time** field.

   The expiration time shown is local to the server. The value for this field must be formatted according to your specified locale. An example of the correct format for your locale is provided above the field.
4. If you want to enable tracing for the failed event, specify a new value in the **Trace Control** field.
5. Do one of the following:
   - If the edited data is correct and you want to resubmit the event, click **Resubmit** to make the changes at a server level.
   - If you want to remove the changes you made, click **Undo local changes**.

The edited failed event is resubmitted for processing and is removed from the failed event manager.

Related tasks

Finding failed events

Editing business data in a failed event

The failed event manager provides a business data editor so you can edit the business data associated with a failed event before you resubmit it. For each failed event, the editor displays the associated business data in a hierarchical format; the navigation tree at the top of the table is updated as you navigate through the parameters to give you a clear picture of where you are in the hierarchy.

Business data can be encapsulated into a business object, or it can be simple data that is not part of a business object. A failed event can have both simple data and a business object associated with it.

You can edit only simple data types (for example, String, Long, Integer, Date, Boolean). If a data type is complex (for example, an array or a business object), you must navigate through the business data hierarchy until you reach the simple data types that make up the array or business object. Complex data is denoted by an ellipsis (`...`) in the Parameter Value column.

Important: Any edits you make to business data are saved locally. Changes are not made to the corresponding business data in the server until you resubmit the failed event.

Security role required: You must be logged in as administrator or operator to perform this task.

1. Ensure that the failed event manager is open and that you have retrieved a list of the failed events on your system.
2. From the failed event manager's Search Results page, click the ID (found in the Message ID column) of the failed event whose data you want to edit.

   The Failed Event Details page opens.
3. From the Failed Event Details page, click **Edit business data** to access the Business Data Editor collection page.

   This page displays a hierarchical view of all of the data associated with the failed event.
4. Navigate through the business data hierarchy by clicking on the name of each parameter (these appear as links in the Parameter Name column). When you have located the parameter whose value you want to edit, click its name.

   If the parameter has an editable value, the Business Data Editor page opens.
5. In the **Parameter value** field, specify the new value for the parameter.
6. Click **OK**.

   The change is saved locally and you are returned to the Business Data Editor collection page.
7. If you want to remove the changes you made, click **Undo local business data changes**.

   All of the edits are removed and the business data is returned to its original state.
8. If the edited business data is correct, click **Resubmit** to make the changes at a server level.

The edited failed event is resubmitted for processing and is removed from the failed event manager.

Resubmitting failed events

If you want to try to execute the event again, you must resubmit it from the failed event manager. You can resubmit an event without changes, or you can edit the business data parameters before resubmitting it.

When a failed event is resubmitted, the processing resumes only for the failed branch, not for the entire event.

Tracing is available for resubmitted events to help monitor the event's processing. You can also use the event's unique message ID to track its success or failure. If a resubmitted event fails again, it is returned to the failed event manager with its original message ID and an updated failure time.

See the following topics for specific instructions on resubmitting failed events:
- Resubmitting an unchanged failed event
- Resubmitting a failed event with trace

Resubmitting an unchanged failed event

You can resubmit one or more unchanged failed events to be processed again. Processing resumes only for the failed branch, not for the entire event.

Security role required: You must be logged in as administrator or operator to perform this task.
1. Ensure that the failed event manager is open and that you have retrieved a list of the failed events on your system.
2. From the Search Results page, select the check box next to each failed event you want to resubmit.
3. Click **Resubmit**.

Each selected event is resubmitted for processing and is removed from the failed event manager.

Resubmitting a failed event with trace

You can monitor the resubmission of a failed event to determine whether it executes successfully. The failed event manager provides optional tracing for all failed events.

Security role required: You must be logged in as administrator or operator to perform this task.
1. Ensure that the failed event manager is open and that you have retrieved a list of the failed events on your system.
2. From the Search Results page, select the check box next to each failed event you want to resubmit.
3. Click **Resubmit with trace**.

   The Resubmit With Trace page opens.
4. Specify the level of trace you want to use in the **Trace control** field.

   By default, the value is SCA.LOG.INFO;COMP.LOG.INFO. With this setting, no trace occurs when the session calls an SCA service or executes a component.
5. Click **OK** to resubmit the failed event and return to the Search Results page.

To view the trace log for a resubmitted event, open the corresponding component logger or use the CEI log viewer.

Deleting failed events

If you do not want to resubmit a failed event, or if you have failed events that have expired, use the failed event manager to delete them from the server. The failed event manager provides three options for deleting failed events.

Security role required: You must be logged in as administrator or operator to perform this task.

1. Ensure that the failed event manager is open and that you have retrieved a list of the failed events on your system.
2. From the failed event manager's Search Results page, do one of the following:
   - If you want to delete one or more specific failed events, select the check box next to each event and then click **Delete**.
   - If you want to delete only those failed events that have expired, click **Delete expired events**.
   - If you want to delete all failed events on the server, click **Clear all on server**.

Using the Common Event Infrastructure

The Common Event Infrastructure is a core component of WebSphere Process Server.

The Common Event Infrastructure provides facilities for the run-time environment to persistently store and retrieve events from many different programming environments. Events are represented using the Common Base Event model a standard, XML-based format that defines the structure of an event.

The Common Event Infrastructure provides WebSphere Process Server with standard formats and mechanisms for managing event data. The following facilities are provided:

- Standard interfaces and services for WebSphere applications to create event objects, store them, send them, and retrieve them later.
- Facilities that pass event objects to registered applications either directly, in the context of the producing (source) application, or indirectly through Java Message Service (JMS). There are event emitters for Business Process Execution Language (BPEL) -based processes and for Enterprise JavaBeans (EJB) invocations based on deployment descriptor extensions.
- The Common Base Event browser for browsing stored events.

Administering the Common Event Infrastructure

You can perform the following administrative tasks to control the operation of the Common Event Infrastructure components at run-time.

Logging and tracing in the Common Event Infrastructure

You can enable logging and tracing in order to debug problems with an application using the Common Event Infrastructure.

The Common Event Infrastructure components use the JSR47 Java logging framework, which is available in WebSphere Process Server and client environments. For more information about using the logging framework, refer to the WebSphere Application Server troubleshooting documentation.

The following table indicates the logger names used by the Common Event Infrastructure components.

Table 7. Logger names

Component	**Logger name**
Root logger name	com.ibm.events
Event catalog	com.ibm.events.catalog
Event server subcomponents	com.ibm.events.access com.ibm.events.bus com.ibm.events.distribution com.ibm.events.server
Default data store plug-in	com.ibm.events.datastore
Event emitter	com.ibm.events.emitter
Notification helper	com.ibm.events.notification
Configuration	com.ibm.events.configuration
Installation	com.ibm.events.install
Miscellaneous utilities	com.ibm.events.util

DB2 database maintenance

If you are using a DB2 event database on Linux, UNIX, or Windows, you should periodically perform database maintenance by running the provided scripts.

The DB2 versions currently supported for running these maintenance scripts are
- DB2 Universal Version 8.1
- DB2 Universal Version 8.2.1

Updating database statistics:

To enable the DB2 database to optimize queries and find free space, update the database statistics using the `runstats` script.

It is recommended that you update the database statistics regularly, and especially under any of these circumstances:
- Events have been purged from the database
- A large number of events have been inserted into the database
- Tables have been reorganized using the `reorg` script
- Indexes have been added or removed from a table

The `runstats` script is located in the *`install_root`*`/event/dbscripts/db2 directory`.

To update the database statistics, run one of the following commands:
- On Windows systems:
  `runstats.bat` *`db_user`* `[`*`db_password`*`]`
- On Linux and UNIX systems:
  `runstats.sh` *`db_user`* `[`*`db_password`*`]`

The parameters are as follows:

db_user
: The database user ID to use. This parameter is required.

db_password
: The database password. This parameter is optional. If you do not specify the password on the command line, the DB2 database will prompt you for it.

For example, the following command updates the DB2 database statistics on a Windows system, using the database user ID `dbadmin` and the password *`mypassword`*:

```
runstats.bat dbadmin mypassword
```

Reorganizing database tables:

After events are purged from a DB2 event database, reorganize the database tables using the `reorg` script.

The `reorg` script is located in the *`install_root`*`/event/dbscripts/db2` directory.

To reorganize the event database tables, run one of the following commands:
- On Windows systems:
  ```
  reorg.bat db2_version db_user [db_password]
  ```
- On Linux and UNIX systems:
  ```
  reorg.sh db2_version db_user [db_password]
  ```

The parameters are as follows:

db2_version
: The DB2 Universal Database major release number; this must be either `8.1` or `8.2.1`. This parameter is required.

db_user
: The database user ID to use. This parameter is required.

db_password
: The database password. This parameter is optional. If you do not specify the password on the command line, the DB2 database prompts you.

For example, the following command reorganizes the event database tables on a Windows system running DB2 Universal Database Version 8.1 or Version 8.2.1, using the database user ID `dbadmin` and the password `mypassword`:

```
reorg.bat 8 dbadmin mypassword
```

After you run the `reorg` script, you should update the database statistics using the `runstats` script. For more information, see *Updating database statistics*.

Purging events from the event database:

You can use the provided scripts to rapidly purge large numbers of events from the event database.

The default data store plug-in provides a set of utilities you can use to periodically perform a rapid purge of large numbers of old events from the event database. These utilities are distinct from the **eventpurge.jacl** event server command, which deletes events matching specified criteria.

The rapid purge capability uses the concept of **buckets**. A bucket is a set of tables used to store events in the event database. The default data store plug-in uses two buckets:

- The **active bucket** is the bucket containing the most recent events; new events are stored in the active bucket. The active bucket cannot be purged using the rapid purge utility.
- The **inactive bucket** contains older events. Events stored in the inactive bucket can be queried, deleted, or modified, but typically no new events are stored in the inactive bucket. The inactive bucket can also be purged by the rapid purge utility.

Each event is stored in only one bucket. From the perspective of an event consumer, the distinction between the active and inactive buckets is invisible; a consumer can query, modify, or delete a specific event without knowing which bucket the event is stored in. The advantage of this approach is that the inactive bucket can be rapidly purged using database-specific interfaces without affecting the active bucket; normal event traffic can continue even while the purge operation is taking place.

After the inactive bucket is purged, you can then swap the buckets so that the active bucket becomes inactive and the inactive bucket becomes active. Swapping buckets is possible only when the inactive bucket is empty.

Note: Although new events are generally stored only in the active bucket, under some circumstances events might be stored in the inactive bucket immediately after the buckets are swapped. The data store plug-in checks periodically to determine which bucket is currently marked as active, but until the next check takes place, some events might continue to be stored in the inactive bucket. Similarly, events sent as part of a batch are all stored in the same bucket, even if that bucket becomes inactive while the batch is still being processed.

If you want to use the fast purge capability, it is your responsibility to determine how frequently to swap buckets or purge the inactive bucket, depending upon event traffic, storage space, archival requirements, or other considerations.

Viewing or changing the active bucket status:

The active bucket status indicates which bucket is currently active and which is currently inactive.

To view or change the active bucket status, use the **eventbucket.jacl** script:

```
wsadmin -f eventbucket.jacl [-status] [-change]
```

This command has the following options:

-status
Use this option to see information about the current bucket configuration, including the active bucket setting and the bucket check interval (the frequency with which the data store plug-in checks to determine which bucket is active).

-change
: Use this option to swap the active and inactive buckets. The inactive bucket must be empty before you can use this option.

Purging the inactive bucket:

The method used to purge the inactive bucket varies depending on the database software.

Note: The rapid purge utility is not supported for Cloudscape databases.

Purging the inactive bucket for a DB2 database (Linux, UNIX, or Windows systems):

On Linux, UNIX, and Windows systems, the rapid purge utility for a DB2 database is implemented as a shell script or batch file.

To purge the inactive bucket, use one of the following commands:

- To purge the inactive bucket on a Windows system, run the following command:
  ```
  fastpurge dbalias dbuser [password=dbpassword] [copydir=copydir]
  ```
- To purge the inactive bucket on a Linux or UNIX system, run the following command:
  ```
  fastpurge.sh dbalias dbuser [password=dbpassword] [copydir=copydir]
  ```

The parameters of this command are as follows:

dbalias
: The database alias. The event database must be cataloged on the DB2 client; if you are running the script on the DB2 server, the database is already cataloged.

dbuser
: The database user ID to use when connecting to the event database.

dbpassword
: The password to use for the specified user ID. This parameter is optional; if you do not specify the password, the DB2 database will prompt for it.

copydir
: The path to a directory to be used for the files generated by the load utility. This parameter is required only if you have enabled forward recovery for the event database (with the LOGRETAIN and USEREXIT database configuration settings enabled). By default, the event database does not use forward recovery.

Purging the inactive bucket for a DB2 database (z/OS systems):

On z/OS systems, the rapid purge utility for a DB2 event database is implemented using the DB2 load utility.

To purge the inactive bucket:

1. Use the **eventbucket.jacl** command to identify the inactive bucket (bucket 0 or bucket 1).
2. Upload the appropriate utility control file. These files are generated during database configuration and are located in the *profile_path*/event/dbscripts/db2zos directory (*profile_path* is the path to the directory containing the profile for the WebSphere Process Server runtime environment. Upload one of the following files:
   - If bucket 0 is inactive, upload fastpurge00.ctl.

- If bucket 1 is inactive, upload fastpurge01.ctl.

  Note: The control file must be uploaded with a fixed record format and a logical record length of 80.

3. On the z/OS host, go to the ISPF DB2I Primary Option Menu and select the **Utilities** option.
4. Specify the following information:

Field	Value
Function	`EDITJCL`
Utility	`LOAD`
Statement Data Set	The name of the data set containing the uploaded control file
LISTDEF	`NO`
Template	`NO`

5. Press the Enter key to continue to the next panel.
6. In the recdsn entry field, specify the name of the data set containing the uploaded control file.
7. Press the Enter key. The JCL script to purge the inactive bucket is generated.
8. Press the Enter key to clear the output messages.
9. Edit the generated JCL script as needed.
10. Submit the JCL script.

Purging the inactive bucket for an Oracle database:

The rapid purge utility for an Oracle event database is implemented as a stored procedure.

To use the utility for an Oracle database, you must have SQL*Plus installed on the Oracle client, and the client must be configured to communicate with the Oracle database (the tnsnames.ora file must be configured properly). Refer to the Oracle documentation for more information.

To purge the inactive bucket, run the stored procedure using SQL*Plus:

```
sqlplus connect_string @fastpurge.sql
```

The *connect_string* parameter is the Oracle connection string. Use the same database user ID used to create the event database.

Changing the bucket check interval:

The bucket check interval is specified in the DataStoreEjb.jar file.

This value specifies how frequently the data store plug-in checks to determine which bucket is active. The default value is 5 minutes (300 seconds). A shorter interval reduces the likelihood that events will be stored in the inactive bucket after swapping, but might decrease performance.

To change the bucket check interval:

1. In the WebSphere Process Server administrative console, navigate to **Enterprise Applications > CommonEventInfrastructureServer > EJB Modules > DataStoreEjb.jar > View Deployment Descriptor**.
   a. Look for the DefaultDataStoreEJB parameters.
   b. Expand the`<session id="DataStoreHelperEJB">` menu, and look for the`<env-entry-value>`.

      This is the bucket check interval value, which is set in number of seconds.
2. Modify the value of the BucketCheckInterval environment variable to specify the bucket check interval in seconds.

Working with events

Java objects are used as representations of the Common Base Event specification.

The Common Event Infrastructure represents events as Java objects. Specifically, each event is an instance of a class implementing the org.eclipse.hyades.logging.events.cbe.CommonBaseEvent interface, which is a Java representation of the Common Base Event specification. The org.eclipse.hyades.logging.events.cbe package is part of the Eclipse-based Hyades environment, which is a set of standards and open-source tools for testing, tracing, and monitoring. For more information, see *http://www.eclipse.org/hyades/*.

The typical life cycle of an event is as follows:

1. To send an event, an event source creates a new instance of CommonBaseEvent, populates it with property data, and then submits it to an emitter.
2. The emitter optionally uses the content completion mechanism (if implemented) to populate the event with required property data. The emitter then validates the event and checks it against the currently configured filter criteria. If the event is valid and passes the filter criteria, the emitter sends the event to the event server. For more information about event processing by the emitter, see *"Sending events" on page 174*.
3. If persistence is enabled, the event server stores the event in a persistent data store.
4. If publishing is enabled, the event server publishes the event to one or more Java Messaging Service (JMS) destinations. Event consumers subscribing to these destinations then receive notifications of the new event. The event consumers then use the Notification Helper to convert the received JMS messages back into a CommonBaseEvent instance.

   An event consumer might also submit a query to retrieve the event from the data store. Typically, a consumer uses the query interface to retrieve historical events, especially during startup processing.

   After receiving the event, an event consumer reads the event property data and processes the event.
5. When it is no longer needed, the event can be purged from the data store.

The Common Base Event specification, which is based on the XML Schema definition language, defines two kinds of event property data:

- Properties represented by simple data types, encoded in XML as attributes of the CommonBaseEvent element. These include properties such as *globalInstanceId*, *severity*, and *msg*. The CommonBaseEvent Java class represents these values as strings or integers, as appropriate.
- Properties represented by complex data types and encoded in XML as subelements of the CommonBaseEvent element. These include properties such as *situation*, *sourceComponentId*, and *extendedDataElements*, each of which has nested

properties of its own. These complex types are represented by specialized Java classes defined in the org.eclipse.hyades.logging.events.cbe package. For example, the *sourceComponentId* property is represented by an instance of ComponentIdentifier.

The CommonBaseEvent interface defines getter and setter methods for each property, as well as helper methods to simplify creation of complex properties. An event source uses the setter methods (or the helper methods) to populate an event with property data before submitting it to an emitter; an event consumer uses the getter methods to retrieve the property data from a received event.

For more information about the XML Schema specification, see *http://www.w3.org/XML/Schema*.

Creating an event object:

New events are created by using an *event factory*.

To create new events in your event source, you use an *event factory*, which is an object that returns new instances of CommonBaseEvent or of the specialized classes representing complex property data types.

There are two ways you can access an event factory:
- You can create a new event factory using the *event factory factory*. Use this approach if no appropriate event factory is already available. When you create a new event factory, you can optionally specify a content handler to provide automatic content completion.
- You can use an existing event factory that has been bound into a Java Naming and Directory Interface (JNDI) namespace. Use this approach if an administrator has provided an event factory for you to use; this ensures that any events you create conform to the appropriate business rules, because the event factory might be configured with a content handler.

Creating a new event factory:

An *event factory* is used to create new events.

To create a new event factory, use the event factory factory, implemented as the class EventFactoryFactory. This class has no instances; instead, it provides two static methods used to create event factories. The choice of which method to use depends upon whether you want to use a content handler to implement automatic content completion. For more information, see *"Completing event content automatically" on page 169*.

To create a generic event factory with no content handler, use the createEventFactory() static method of EventFactoryFactory:

```
EventFactory eventFactory =
     (EventFactory) EventFactoryFactory.createEventFactory();
```

To create an event factory with a content handler, use the createEventFactory(ContentHandler) method, specifying the content handler you want to use:

```
EventFactory eventFactory =
     (EventFactory) EventFactoryFactory.createEventFactory(contentHandler);
```

In either case, the returned object is an event factory you can use to create new events.

Getting an event factory by JNDI lookup:

Use a JNDI lookup to retrieve event factories, which use JNDI for event sources.

If an administrator has bound an existing event factory into JNDI for event sources to use, perform a standard JNDI lookup to retrieve the event factory:

```
import javax.naming.*
import org.eclipse.hyades.logging.events.cbe.*

Context context = new InitialContext();
EventFactory eventFactory =
     (EventFactory) context.lookup("com/ibm/events/EventFactory");
```

The returned object is the provided event factory; if the event factory is configured with a content handler, an instance of the content handler is also created locally. For more information about content handlers and JNDI, see *"Completing event content automatically" on page 169*.

Creating and populating an event:

After obtaining an event factory, you can create event objects and populate them with property data.

Most event properties are defined as optional by the Common Base Event specification, but the following properties are required:

- *version* (a string attribute)
- *creationTime* (an XML Schema dateTime attribute; see *http://www.w3.org/TR/2001/REC-xmlschema-2-20010502/#dateTime*)
- *sourceComponentId* (a complex ComponentIdentification element)
- *situation* (a complex Situation element)

Note: The *version* attribute is defined as optional by the Common Base Event specification, but if it is not specified, the default value `1.0` is assumed. Because the Common Event Infrastructure supports only version 1.0.1 of the specification, this value must be specified.

If you try to send an event that is missing any of these properties, the emitter rejects the event and throws an EventsException exception.

The following code fragment creates an event and populates it with the minimal required property data:

```
CommonBaseEvent event = eventFactory.createCommonBaseEvent();

event.setVersion("1.0.1");                     // set version

long currentTime = System.currentTimeMillis(); // get current time
event.setCreationTimeAsLong(currentTime);      // and set creationTime

// set sourceComponentId (a complex type)
event.setSourceComponentId("Windows",          // application
                           "svchost.exe",      // component
                           "tlntsvr.exe",      // subcomponent
                           "http://www.ibm.com/namespaces/autonomic/Windows",
                                               // componentType
                           "win386_svc",       // componentIdType
```

```
                                "9.45.72.138",      // location
                                "IPV4"              // locationType
                                );

// create situation object
Situation situation = eventFactory.createSituation();

// set situationType to AvailableSituation (a complex type)
situation.setAvailableSituation("EXTERNAL",          // reasoningScope
                                "NOT AVAILABLE",     // availabilityDisposition
                                "STARTABLE",         // operationDisposition
                                "FUNCTION_PROCESS"); // processingDisposition

// set situation
event.setSituation(situation);
```

This example first uses an event factory to create a new event instance, *event*. First it sets the *version* property; then it retrieves the current system time and uses the setCreationTimeAsLong(long) method to set the value of the *creationTime* property. An alternative is to use the setCreationTime(String) method, which sets the creation time using the XML dateTime format (for example, `"2004-07-29T13:12:00-05:00"`).

The next required property, *sourceComponentId*, is a complex property represented by an instance of ComponentIdentification, which has properties of its own. However, it is not necessary to directly instantiate or interact with this object (although it is possible to do so). Instead, the next statement in the example uses a helper method, setSourceComponentId(), to specify the nested properties; the helper method uses these values to create an instance of ComponentIdentification, which it then uses to set the value of the *sourceComponentId* property of the event.

Similar helper methods exist for setting other complex properties (for example, setMsgDataElement(), addAssociatedEvent, and addExtendedDataElement()). Many of these methods exist in multiple versions with different signatures, making it possible to specify property values in different ways. Refer to the Javadoc API documentation for complete information on these methods.

The last required property in the example, *Situation*, is another complex property. In this case the situation object must be instantiated directly using an event factory; the example then uses a helper method to set the *situationType* property, which is itself a complex subelement.

In an actual application, a useful event needs to include more information than is shown here, but this is the minimum required by the Common Base Event specification and the Common Event Infrastructure. The event is now valid and can be submitted to an emitter.

Completing event content automatically:

By setting properties and policies, event content can be automatically completed.

In some situations, you might want some event property data to be automatically set for every event you create. This is a way to fill in certain standard values that do not change (such as the application name), or to set some properties based on information available from the runtime environment (such as creation time or thread information). You can also set policies that govern event content according to business rules; for example, you might require that any event with a particular extension name have its severity set to a certain value.

You can do this by creating a *content handler*. A content handler is an object that automatically sets the property values of each event based on any arbitrary policies you want to use. The Common Event Infrastructure does not restrict how a content handler can modify event data, so long as the event still conforms to the Common Base Event specification.

To ensure that all event sources comply with the same policies, you can create an event factory associated with a content handler (using EventFactoryFactory) and then bind the created event factory into a JNDI namespace. Instead of creating their own event factories, event sources can then perform JNDI lookups to access the event factory that already exists, without any knowledge of the content handler. If your business rules later change, you can modify the content handler in one place.

An event source does not need to do anything to enable content completion. If an event factory is associated with a content handler, each event it creates carries with it a reference to that content handler. When the event is submitted to an emitter, the event calls the completeEvent() method of the content handler, passing a reference to itself. This ensures that the correct policies are applied to the event after the event source is finished setting event-specific properties, but before the event is validated and processed by the emitter.

Note: When an event is transmitted from one process to another, the reference to the content handler is not transmitted with it. This is because content completion relies upon the environment where the event originates, and the necessary information might not be available elsewhere. This restriction does not affect calls between applications that are local to one another (for example, a call to an enterprise bean using its local interface).

To create a content handler, follow these steps:

1. Create a new Java class implementing the org.eclipse.hyades.logging.events.cbe.ContentHandler interface. This interface defines a single method called completeEvent(CommonBaseEvent); the parameter is the event whose content is to be completed. In your implementation of this method, you can use the getter and setter methods of CommonBaseEvent to process the event property data in accordance with any policies that apply.

   Note: When an event source uses JNDI to retrieve an event factory, the content handler is returned along with the event factory. For this reason, the content handler must be serializable.

   The following example is a simple content handler that automatically sets the extension name of each event:

   ```
   import java.io.Serializable;
   import org.eclipse.hyades.logging.events.cbe.*;

   public class BusinessContentHandler
        implements ContentHandler, Serializable {

          public void completeEvent(CommonBaseEvent event)
             throws CompletionException {
                 event.setExtensionName("business");
        }
   }
   ```

2. Associate the content handler with an event factory. To do this, specify the content handler when creating the event factory:

```
EventFactory eventFactory =
     (EventFactory) EventFactoryFactory.createEventFactory(contentHandler);
```

The returned event factory is permanently associated with the specified content handler.

Retrieving data from a received event:

An event source uses methods of the CommonBaseEvent to retrieve event property data.

When an event source receives an event, it can then use the getter methods of CommonBaseEvent to retrieve the event property data. For example, the following code fragment retrieves a single event and then reads the content of the *msg* property.

```
CommonBaseEvent event = eventAccess.queryEventByGlobalInstanceId(eventId);
String eventMessage = event.getMsg();
```

If the property you want to retrieve is a complex property (a subelement of CommonBaseEvent in the Common Base Event specification), the returned value is an instance of the specialized class representing the complex data type. You can then use the getter methods of the returned object to retrieve the property data from that object. For example, the following code fragment retrieves the value of *componentId*, which is a complex property; it then retrieves the content of the nested *component* property, which is a string, to read the name of the source component.

```
CommonBaseEvent event = eventAccess.queryEventByGlobalInstanceId(eventId);
ComponentIdentification componentId = event.getSourceComponentId();
String componentName = componentId.getComponent();
```

Converting XML events:

An event source can convert XML-formatted events from other applications.

In addition to creating new events, an event source might convert events received from other applications in XML format. Similarly, an event consumer might convert events to XML format for forwarding to another application. The org.eclipse.hyades.logging.events.cbe.EventFormatter class provides methods you can use to convert between CommonBaseEvent instances and XML.

Using EventFormatter, you can convert an instance of CommonBaseEvent into a string containing either an XML document or an XML fragment. Similarly, you can convert from an XML document or fragment into an instance of CommonBaseEvent.

For more information about EventFormatter, refer to the Javadoc documentation in the org.eclipse.hyades.logging.events.cbe package.

Accessing event instance metadata:

Accessing event instance metadata is done through the Java-based Eclipse Modeling Framework.

The org.eclipse.hyades.logging.events.cbe package, which provides the classes and interfaces required for working with event objects, is based on the Eclipse Modeling Framework (EMF). EMF is a Java framework used to generate application code based on a structured data model; it also provides interfaces in the generated code that can be used to access metadata describing the data model. (Refer to the Eclipse Modeling Framework documentation at http://www.eclipse.org/emf for more information about EMF.)

By using these interfaces, EMF-compatible tools can interact with CommonBaseEvent event data without any prior knowledge of the data model or access to the implementation. This makes it possible for development tools to generate code that transfers data from other data models into the CommonBaseEvent model. Application developers can then focus on writing code that uses the data rather than code that builds the data.

For example, consider an event source that monitors network events and describes its own data model in terms of EMF. With access to both data models, a development tool could display the fields of the event source data model alongside the fields of the CommonBaseEvent data model. A developer could then use a graphical interface to indicate how the fields in the event source model are mapped to fields in the CommonBaseEvent model; for example, a field called Workstation.name in the event source data model might correspond to the CommonBaseEvent.sourceComponentId.location field in the CommonBaseEvent data model. Because both data models are described using standard EMF interfaces, the tool could generate code that handles the transfer of data between the two models.

The following code fragment is a simple example of how a development tool might use EMF interfaces to query information about the CommonBaseEvent data model and then use that information to interact with an event instance. This example could be part of a simple event consumer; it iterates through all of the fields of an event instance and, for each one, prints the name and value of the field.

```
// event is a valid CommonBaseEvent instance

// Get list of event instance structural features (fields)
List features = event.eClass().getEAllStructuralFeatures();

// iterate through list; print names and values
for (int i = 0 ; i < features.size() ; i++)
  {
    EStructuralFeature feature = (EStructuralFeature)features.get(i);
    Object value = eObj.eGet(feature);
    System.out.println(feature.getName() + ":" + value);
  }
```

The CommonBaseEvent data model is described in the EMF files cbe.ecore and cbe.genmodel. These files are included with the Common Event Infrastructure SDK; you can import them into an Eclipse-based development environment and then use EMF to generate code that interacts with CommonBaseEvent objects.

Creating an event source

An event source interacts with the event server *through* an emitter object.

An *event source* is any application that uses an emitter to send events to the event server. The following applications are examples of event sources:

- An adapter or monitor that generates events related to monitored resources
- An application that generates notification events
- An application that forwards events from other sources

An event source is implemented in the Java programming language, using either the Java 2 Platform, Standard Edition (J2SE) or the Java 2 Platform, Enterprise Edition (J2EE). An event source must submit valid events conforming to the Common Base Event model. Each event is represented as a Java object.

Emitters and emitter factories

An event source does not interact directly with the event server; instead, it interacts with an object called an emitter (an implementation of the com.ibm.events.emitter.Emitter interface). An emitter is a local object that provides methods for sending events.

In general, the emitter handles the details of event transmission; the developer of an event source does not need to be concerned about the event server location, the filter settings, or the underlying transmission mechanism. Details such as these are governed by the *emitter factory*, an object configured by an administrator and bound in a Java Naming and Directory Interface (JNDI) namespace. An emitter factory is an an instance of com.ibm.events.emitter.EmitterFactory and is used to create emitter objects. It also defines the behavior of the emitters it creates; it includes settings for the following:

- The preferred *transaction mode*. This setting specifies whether the emitter attempts to send each event in a new transaction or within the current transaction. An event source can change this setting for a particular emitter or event submission, but the profile specifies the default value. (This setting is valid only in a J2EE container; the J2SE platform does not provide transaction controls.)
- The preferred *synchronization mode*. This setting specifies whether events are sent using synchronous or asynchronous transmission. *Synchronous transmission* means that the sendEvent() method does not return control to the caller until the event has been processed; *asynchronous transmission* means that the method returns immediately after the event is submitted, and the caller has no further information about event processing. An event source can change this setting for an emitter or for an event submission, but the default value is specified by the profile.
- The *transmission profiles* to use. A transmission profile is a configuration object that defines a specific transmission mechanism for sending events to the event server. An emitter factory profile can specify two transmission profiles, one for synchronous transmission and one for asynchronous transmission. An event source cannot change the transmission profiles used by an emitter.
- The filter configuration to use for the emitter. The filter configuration defines what filtering plug-in is used to filter events submitted to the emitter. The Common Event Infrastructure includes a default filter plug-in, but you can also implement your own filter plug-in if you want to use a different filtering engine.

An administrator can create multiple emitter factory profiles, each one defining a different emitter configuration. An event source obtains an emitter using the emitter factory associated with an existing emitter factory profile; therefore, all emitters created by a particular emitter factory will have the same default behavior. For more information, see *Obtaining an emitter*.

Note: If your event source is running with Java 2 security enabled, and you want to generate your own globally unique identifiers (GUIDs), you must modify your policy file to enable correct processing. Add the following entries:

```
      permission java.io.FilePermission "${java.io.tmpdir}${/}guid.lock",
        "read, write, delete";
      permission java.net.SocketPermission "*", "resolve";
```

Obtaining an emitter:

Before you can obtain an emitter, there must be at least one emitter factory profile configured.

For each emitter factory profile, an emitter factory is automatically created and is accessible using the JNDI name of the emitter factory profile.

To obtain an emitter, follow these steps:
1. Perform a JNDI lookup specifying the name of the emitter factory you want to use for your emitter. This is the JNDI name specified by an administrator when defining an emitter factory profile.
2. Call the *getEmitter()* method of the emitter factory. The returned object is an emitter configured according to the options defined in the emitter factory profile you specified. If the emitter factory is unable to obtain an emitter, it throws an EmitterException exception.

   Note: If your event source is a J2EE client application running in a secure environment, and the emitter profile you are using specifies asynchronous transmission profiles, you must specify a JMS user name and password in order to get an emitter. To do this, use the *getEmitter(String, String)* method, passing the JMS user name and password you want to use. For more information, refer to the Javadoc documentation for the com.ibm.events.emitter class.

The following code fragment obtains an emitter configured with the profile *Default*:

```
import javax.naming.*
import com.ibm.events.emitter.*

Context context = new InitialContext();
EmitterFactory emitterFactory =
  (EmitterFactory) context.lookup("com/ibm/events/configuration/emitter/Default");
Emitter emitter = emitterFactory.getEmitter();
```

Sending events:

An event source sends events in the form of Java objects.

Specifically, each event is an instance of a class implementing the org.eclipse.hyades.logging.events.cbe.CommonBaseEvent interface, which is a Java representation of the Common Base Event specification. For more information, see *The Common Base Event model*.

To send an event, use the sendEvent() methods of the Emitter interface. When you submit an event to an emitter, the following occurs:
1. The emitter calls the complete() method of the event, triggering optional content completion. See *"Completing event content automatically" on page 169* for more information.
2. The emitter assigns a sequence number and global instance identifier to any event that does not already have them.
3. The emitter validates the event to ensure that it conforms to the Common Base Event specification.

Note: The current Common Base Event specification allows only one extended data element with a given name at each level of the event containment hierarchy, but this restriction will not be included in future versions of the specification and is not enforced by the Common Event Infrastructure.

4. If filtering is active, the emitter checks the event against the current filter criteria to determine whether the event should be sent or discarded.
5. Finally, if the event is valid and passes the filter criteria, the emitter sends the event to the event server for persistence and distribution to event consumers.

If the event is not valid, or if the emitter encounters a problem when trying to send the event to the event server, an exception is thrown.

Note: If WebSphere security is enabled, the application user ID must be mapped to the eventCreator or eventAdministrator role to send events using synchronous EJB transmission.

Sending an event with the current emitter settings:

Events can be sent with current emitter settings, if transmission modes do not need to be specified.

If you do not need to specify a particular transmission mode or transaction mode, you can send an event using the current emitter settings. These settings are initially defined by an administrator in the emitter factory profile, but they can later be changed by event consumers.

To send an event using the current emitter settings, use the `sendEvent(CommonBaseEvent)` method.

```
String eventId = emitter.sendEvent(event);
```

In this example, `emitter` is an Emitter instance, and `event` is a CommonBaseEvent instance.

The returned value, `eventId`, is the globally unique identifier of the event (the value of the *globalInstanceId* field of CommonBaseEvent). If the event does not have a global instance identifier when you submit it, the emitter assigns one automatically.

Note: Submitting an event to an emitter does not guarantee that the event is sent to the event server, because the filter settings might cause the event to be discarded. A successful call to sendEvent() means only that the event was successfully processed by the emitter.

Overriding the current emitter settings:

Emitter factory profile settings can be changed by event consumers.

When sending an event, you can specify options that override the current transaction mode, synchronization mode, or both, currently configured for the emitter. These settings are initially defined by an administrator in the emitter factory profile, but they can later be changed by event consumers.

An emitter might not support all synchronization and transaction modes. The available modes are subject to the following limitations:

- The synchronization modes supported by an emitter are defined by the emitter factory profile. You can find out which modes are supported by a particular emitter by calling the isSynchronizationModeSupported() method; see the Javadoc API documentation for com.ibm.events.emitter.Emitter for more information.
- Transactions are supported only in a J2EE container.

If you attempt to use a mode that is not supported, the emitter throws a TransactionModeNotSupportedException or SynchronizationModeNotSupportedException exception.

To override the emitter settings, use the sendEvent(CommonBaseEvent, int, int) method.

```
String eventId = emitter.sendEvent(event,
                                   synchronizationMode,
                                   transactionMode);
```

The parameters are as follows:

event
: The event object (an instance of CommonBaseEvent) you want to send.

synchronizationMode
: An integer constant defined by the interface SynchronizationMode. This should be one of the following constants:
  - SynchronizationMode.ASYNCHRONOUS (send the event asynchronously)
  - SynchronizationMode.SYNCHRONOUS (send the event synchronously)
  - SynchronizationMode.DEFAULT (use the current emitter setting)

transactionMode
: An integer constant defined by the interface TransactionMode:
  - TransactionMode.NEW (send the event in a new transaction)
  - TransactionMode.SAME (send the event in the current transaction)
  - TransactionMode.DEFAULT (use the current emitter setting)

The event is sent with the options you specify. These options apply only to the single event being sent; no changes are made to the emitter settings, and subsequent event submissions are not affected.

The returned value, `eventId`, is the globally unique identifier of the event (the value of the `globalInstanceId` field of CommonBaseEvent). If the event does not have a globalInstanceId when you submit it, the emitter assigns one automatically.

Note: Submitting an event to an emitter does not guarantee that the event is sent to the event server, because the filter settings might cause the event to be discarded. A successful call to sendEvent() means only that the event was successfully processed by the emitter.

The following example overrides the emitter setting to send an event in a new transaction, but does not override the synchronization mode:

```
String eventId = sendEvent(event,
                           SynchronizationMode.DEFAULT,
                           TransactionMode.NEW);
```

Sending multiple events:

If your event source needs to send multiple events in a batch, you can improve performance by sending them with a single call to the sendEvents() method.

Batching events in this way can also be useful for logical groups of events that should be sent only if an underlying transaction has completed successfully. All of the submitted events are sent as part of a single transaction.

- To send multiple events with the current emitter settings, use the `sendEvents(CommonBaseEvent[])` method:

  ```
  String[] eventIds = emitter.sendEvents(events);
  ```

  In this example, `emitter` is an Emitter instance, and `events` is an array of CommonBaseEvent instances.
- To send multiple events and override the current emitter settings, use the sendEvents(CommonBaseEvent, int, int) method, specifying the synchronization mode and transaction mode you want to use:

  ```
  String eventId = emitter.sendEvent(event,
                                     synchronizationMode,
                                     transactionMode);
  ```

The returned value, `eventIds`, is an array containing the globally unique identifiers of the sent events.

Each event is validated and checked against the current filter criteria. All of the valid events that pass the filter criteria are then sent using the appropriate mechanism:

- If you are using synchronous event transport, the events are sent using a single EJB call. If an error occurs during the EJB call, an exception is thrown and none of the events are sent.
- If you are using asynchronous event transport, all of the events are sent using a single JMS message. If an error occurs during JMS processing, an exception is thrown and none of the events are sent.

Changing the emitter settings:

The settings in an emitter factory profile can be changed by an event source.

An event source can make changes to the transaction mode and synchronization mode configured for the emitter. These settings are initially defined by the emitter factory profile. In addition, an event source can query the current transaction mode to determine what setting is currently in effect for the emitter.

Changing the synchronization mode:

An event source can change the synchronization mode being used by an emitter.

This change remains in effect for subsequent event submissions, but it does not change the preferred synchronization mode defined in the emitter factory profile.

The synchronization modes supported by an emitter are defined by the emitter factory profile. You can find out which modes are supported by a particular emitter by calling the isSynchronizationModeSupported() method; see the Javadoc API documentation for com.ibm.events.emitter.Emitter for more information. If you attempt to use a mode that is not supported, the emitter throws a SynchronizationModeNotSupportedException exception.

To change the synchronization mode, use the setSynchronizationMode(int) method.

```
emitter.setSynchronizationMode(synchronizationMode);
```

The *synchronizationMode* is an integer constant defined by the interface SynchronizationMode:
- SynchronizationMode.ASYNCHRONOUS (send the event asynchronously)
- SynchronizationMode.SYNCHRONOUS (send the event synchronously)
- SynchronizationMode.DEFAULT (send the event using the current emitter settings)

Changing the transaction mode:

An event source can change the transaction mode being used by an emitter.

This transaction mode change remains in effect for subsequent event submissions, but it does not change the transaction mode defined in the emitter factory profile.

Note: Transactions are supported only in a J2EE container.

To change the transaction mode, use the setTransactionMode(int) method.
```
emitter.setTransactionMode(transactionMode);
```

The *transactionMode* is an integer constant defined by the interface TransactionMode:
- TransactionMode.NEW (send the event in a new transaction)
- TransactionMode.SAME (send the event in the current transaction)
- TransactionMode.DEFAULT (send the event using the current emitter settings)

Querying the transaction mode:

An event source can query the transaction mode being used by an emitter.

Note: Transactions are supported only in a J2EE container.

To query the current transaction mode, use the getTransactionMode() method.
```
int transactionMode = emitter.getTransactionMode();
```

The returned value is an integer corresponding to one of the transaction mode constants:
- TransactionMode.NEW
- TransactionMode.SAME

Freeing emitter resources:

If your event source has finished sending events with a particular emitter, you should free the resources the emitter is using.

To free the emitter resources, use the close() method:
```
emitter.close();
```

This method releases all resources being used by the emitter.

Filtering events:

An emitter can optionally be configured to filter events at the source.

Event filtering provides a mechanism for reducing event traffic by screening out events that are not important. Each time an event source submits an event to an emitter, the emitter checks the event against the current filter criteria. If the event passes the filter criteria, the emitter sends the event to the event server; otherwise, the emitter discards the event. In any case, an event source cannot change the filter settings, which are configured by an administrator.

The emitter filter is implemented as a separate component called a *filter plug-in*. The Common Event Infrastructure includes a default filter plug-in, which provides filtering of submitted events based on XPath event selectors. If you want to use a different filter mechanism, you can implement your own filter plug-in.

In the Common Event Infrastructure configuration, each emitter factory is associated with a *filter factory*. A filter factory is an object used to create instances of a filter plug-in. When you create an emitter using an emitter factory, the emitter is automatically associated with an instance of the specified filter plug-in, which provides filtering of events submitted to that emitter.

Filtering events with the default filter plug-in:

A default emitter filter plug-in is included with the Common Event Infrastructure.

The Common Event Infrastructure includes a default emitter filter plug-in, which can be configured with an XPath event selector defining which events are sent to the event server and which are discarded. For example, the filter settings might specify that only events with severity greater than 20 (harmless) should be sent.

To filter events using the default filter plug-in, follow these steps:
1. In the WebSphere administrative console, navigate to the **Common Event Infrastructure Provider > Filter Factory Profile** page.
2. Create a new filter factory profile. For more information, see the online help for the administrative console.
3. In the **Filter Configuration String** field, type an XPath event selector describing events you want to use for filtering events. Events matching this event selector are sent to the event server; events that do not match are discarded by the emitter.
4. Navigate to the **Common Event Infrastructure Provider > Emitter Factory Profile** page.
5. Create a new emitter factory profile, or go to an existing emitter factory profile. For more information, see the online help for the administrative console.
6. In the **Filter Factory JNDI Name** field, type the JNDI name of the new filter factory profile you created.

Event sources can now use the new emitter factory to create instances of an emitter using the new filter configuration. If you later want to adjust the filter settings for event sources using that emitter factory, you can modify the event selector specified in the filter factory.

You can use tracing to find out what events are being discarded by the default filter plug-in; for more information, see *“Logging and tracing in the Common Event Infrastructure” on page 160*.

Note: The default filter plug-in uses the Apache JXPath component to process XPath event selectors. If Java 2 security is enabled, you must modify your policy file to include entries allowing the required access:

```
permission java.util.PropertyPermission "*", "read";
permission java.io.FilePermission
  "${was.install.root}${/}java${/}jre${/}lib${/}jxpath.properties",
  "read";
```

Implementing a filter plug-in:

You can implement a custom filter plug-in.

If you want to use your own filtering engine as an emitter filter, you can implement a custom filter plug-in by following these steps:

1. Develop your filter plug-in as a Java class implementing the interface com.ibm.events.filter.Filter. This interface defines the following methods:

   isEventEnabled(CommonBaseEvent)
   Returns a boolean value indicating whether the specified event passes the filter criteria. Each time an event is submitted to an emitter, the emitter calls this method, passing the submitted event. If the return value is `true`, the emitter sends the event to the event server for persistence and distribution. If the return value is `false`, the emitter discards the event.

   getMetaData()
   Returns information about the filter plug-in, such as the provider name and version number.

   close() Frees all resources used by the filter plug-in. This method is called when the close() method of an emitter is called.
2. Develop a filter factory class that implements the interface com.ibm.events.filter.FilterFactory. This interface defines a single method, getFilter(), which returns an instance of your filter class (an implementation of the Filter interface).
3. Bind an instance of your filter factory into a JNDI namespace. During initialization, an emitter performs a JNDI lookup to access the filter factory.
4. In the WebSphere Process Server administrative console, modify your emitter factory profile or create a new profile. In the **Filter Factory JNDI Name** field, specify the JNDI name of your FilterFactory implementation. For more information about emitter factory profiles, see the online help for the administrative console.

When you create an emitter using the emitter factory profile that specifies your filter factory, the new emitter uses an instance of your filter implementation. You can now send events using the standard emitter interfaces, and your filter plug-in is used.

Creating and populating an event using the ECSEmitter class:

Through the Java Naming and Directory Interface, you can indirectly access the event factory.

If an event factory is bound into the Java Naming and Directory Interface namespace, you can access the event factory indirectly. You can use the com.ibm.websphere.cem.ECSEmitter class to create and populate a Common Base Event. This class provides the following methods:

- createCommonBaseEvent method. If you use this method, you need provide only the extension name and the situation properties for the Common Base Event. All other properties are set automatically.

- addUserDataEvent method. If you use this method, all of the mandatory properties are set automatically. The extension name is set to `ECS:UserDataEvent` and the situation is set to `ReportSituation`. You can set extended data elements for the Common Base Event by passing a set of properties.

You can create and populate a Common Base Event in one of the following ways.

- Use the createCommonBaseEvent method to create and populate an event.

  The following code fragment starts a new event correlation sphere, newECSID, and then uses the createCommonBaseEvent method to create an event object. For more information on event correlation spheres, see

  ```
  ECSEmitter myEmitter = new ECSEmitter("JNDI Emitter Factory Name", "newECSID");
  CommonBaseEvent myEvent = myEmitter.createCommonBaseEvent("myEventType");
  // get situation object
  Situation mySituation = myEvent.getSituation();
  // set situation properties
  mySituation.setCategoryName("ReportSituation");
  mySituation.setReportSituation("EXTERNAL", "STATUS");
  // add other information to the the event
  // send the event
  myEmitter.sendEvent(myEvent);
  ```

  This example uses the constructor method of the ECSEmitter class to create an emitter, passing the JNDI name of an existing Common Event Infrastructure emitter and the identification of a new event correlation sphere.

  The new emitter is then used to create a Common Base Event that contains a situation object that is accessed using the getSituation call. The setCategoryName and setReportSituation methods are used to set the mandatory data in the situation object to emit an event with a `ReportSituation`. To create other situation types, use different category names in the setCategoryName call and different setter method calls for the situation.

  All other mandatory information is provided automatically by the runtime environment. If mandatory information is set explicitly in the Common Base Event, this information is not overwritten with the default information. The event is now valid and can be submitted to an emitter using the sendEvent method.

  In an actual application, a useful event needs to include more information than is shown in this example, but these properties are the minimum required by the Common Base Event specification and the Common Event Infrastructure.

- Use the addUserDataEvent method to create and populate an event.

  The following code fragment uses the addUserDataEvent method to create an event object in the current event correlation sphere.

  ```
  ECSEmitter myEmitter = new ECSEmitter("JNDI Emitter Factory Name", null);
  // prepare a set of user data properties
  Properties myUserData = new Properties();
  myUserData.setProperty("UserData1","UserDataValue1");
  myUserData.setProperty("UserData2","UserDataValue2");
  // create and send the event
  myEmitter.addUserDataEvent(myUserData);
  ```

  This example uses the constructor method of the ECSEmitter class to create an emitter, passing the JNDI name of an existing Common Event Infrastructure emitter. An event correlation sphere identifier is not passed (`null`) and therefore a new event correlation sphere is not started. If an event correlation sphere exists, the user data event is added to this correlation sphere.

  A set of user data properties is then prepared. Name and value pairs are added to a property list.

The last step in the example creates and sends a Common Base Event using the addUserDataEvent method of the new emitter. The extensionName property of the new Common Base Event is set to `ECS:UserDataEvent`, the situation is set to `ReportSituation`, and all other mandatory information is provided automatically by the runtime environment.

Creating an event consumer

An *event consumer* is any application that receives events from the event server.

An *event consumer* might be an application that receives asynchronous event notifications, or it might be an application that queries and processes historical event data from the persistent data store. The event consumer receives events in the form of Java objects; it can then use the CommonBaseEvent interface to retrieve event property data, or convert the event to another supported format (such as XML) for forwarding to another application.

An event consumer can receive events in either of two ways:

- It can use the Java Messaging Service (JMS) interface to subscribe to a queue or topic, receiving event notifications asynchronously as JMS messages. This is the most efficient approach for an event consumer that needs to process new and changed events as they arrive at the event server.
- It can use the Event Access interface to query historical events from the persistent data store, retrieving the requested events synchronously. This is useful for startup processing; by querying the data store for historical events, an event consumer can determine current state information before beginning to receive new events through JMS.

In addition to receiving events, an event consumer can also modify events, delete events, and purge old events from the data store.

Using the Java Messaging Service interface:

Using the Java Messaging Service (JMS) interface, you can develop event consumers that receive event notifications asynchronously from JMS queues or topics.

An event consumer can be implemented as a standard Java class or as a Message-Driven Bean (MDB).

By using the JMS interface, you can implement your event consumer using standard Java tools and programming models, and you can avoid the performance disadvantages of directly querying the event data store. Instead of interacting with the Common Event Infrastructure directly, your event consumer subscribes to JMS destinations (queues and topics) and receives event notifications in the form of JMS messages.

The Common Event Infrastructure organizes events in event groups, which are logical collections of events defined in the Common Event Infrastructure configuration. A particular event consumer typically needs to receive only events from specific event groups.

The configuration profile for each event group associates that event group with one or more JMS destinations through which notifications related to that event group are distributed. The relationships between event groups and JMS destinations are as follows:

- An event group can be associated with multiple queues.
- An event group can be associated with only one topic. (Multiple event consumers can subscribe to the same topic, so publishing the same event group to more than one topic is redundant.)
- A JMS destination (queue or topic) should typically be associated with only one event group.

To receive messages from an event group, a JMS consumer subscribes to the appropriate destination. Each time an event matching the associated event group is created, modified, or deleted, a notification is delivered in the form of a JMS message containing an event notification. The content of the notification depends upon its type:

- For a new or modified event, the notification includes the complete event data, which can be converted into a CommonBaseEvent instance.
- For a deleted event, the notification includes the global instance identifier of the event that has been deleted.

In addition to the standard JMS interfaces, a JMS event consumer interacts with a facility called the Notification Helper. The Notification Helper translates between Common Event Infrastructure entities (events and event groups) and equivalent JMS entities (messages and destinations). The Notification Helper provides the following functions:

- It identifies the JMS topic or queues associated with a specified event group. Your event consumer can then use the appropriate destination to create subscriptions.
- It converts event notifications for new and changed events into instances of CommonBaseEvent.
- It can provide filtering of events at the consumer. Each Notification Helper can be associated with an event selector specifying which events should be returned to consumers. When a consumer uses the Notification Helper to convert an event notification into an event instance, the event instance is returned only if it matches the specified event selector.

  Note: The notification helper uses the Apache JXPath component to process XPath event selectors. If Java 2 security is enabled, you must modify your policy file to include the following entries:

  ```
  permission java.util.PropertyPermission "*", "read";
  permission java.io.FilePermission
    "${was.install.root}${/}java${/}jre${/}lib${/}jxpath.properties",
    "read";
  ```

Developing an event consumer as a message-driven bean (MDB):

A J2EE event consumer is implemented as a message-driven bean, which is associated with a JMS destination and connection factory at deployment time. To receive events, follow these steps:

1. Obtain a notification helper. A JMS event consumer uses a notification helper to identify JMS destinations associated with an event group, to convert received JMS messages into event notifications, and to perform filtering of received events. To obtain a notification helper, use a notification helper factory, which is an instance of NotificationHelperFactory that has been bound into a JNDI namespace. The following code fragment uses a notification helper factory to obtain a notification helper.

```
// Get notification helper factory from JNDI
InitialContext context = new InitialContext();
Object notificationHelperFactoryObject =
  context.lookup("com/ibm/events/NotificationHelperFactory");
NotificationHelperFactory nhFactory = (NotificationHelperFactory)
  PortableRemoteObject.narrow(notificationHelperFactoryObject,
                              NotificationHelperFactory.class);

// Create notification helper
NotificationHelper notificationHelper =
  nhFactory.getNotificationHelper();
```

2. **Optional:** Specify the event selector. If you want to filter received events, you can use the setEventSelector() method to set an event selector on the notification helper. Your event consumer can then use the notification helper to check received events against the event selector. The following code fragment sets an event selector specifying events with severity greater than 30 (warning).

```
notificationHelper.setEventSelector("CommonBaseEvent[@severity > 30]");
```

3. Convert received messages into event notifications.

   In the onMessage() method of your listener, use the notification helper to convert each received JMS message into an array containing an event notification. (If the event does not match the event selector specified on the Notification Helper, the array is empty.) An event notification is an instance of a class implementing the EventNotification interface.

```
public void onMessage(Message msg) {
  EventNotification[] notifications =
                      notificationHelper.getEventNotifications(msg);
  // ...
```

4. Check the notification type and retrieve the event data as appropriate. Each event notification has a field representing the notification type (an integer whose value is one of the notification type constants defined by the NotificationHelper interface). Three notification types are currently supported:

Notification type	Description
`CREATE_EVENT` `_NOTIFICATION_TYPE`	A new event has been created in the event group associated with the destination. This means either that a new event has been sent, or that an existing event has changed so that it now matches the event group definition. The notification also contains the complete event data.
`REMOVE_EVENT` `_NOTIFICATION_TYPE`	An event stored in the event database has been removed from the event group associated with the destination. This means either that an event has been deleted from the event database, or that an existing event has changed so that it no longer matches the event group definition. The notification also contains the global instance identifier of the deleted event.
`UPDATE_EVENT` `_NOTIFICATION_TYPE`	An event stored in the event database has been updated in a way that does not change its membership in the event group associated with the destination. The notification also contains the complete event data.

   Use the getNotificationType() method of EventNotification to check the notification type of each received notification. Based on the notification type, you can determine whether your event consumer should process the notification further, and what kind of event data the notification contains:

- If the notification type is CREATE_EVENT_NOTIFICATION_TYPE or UPDATE_EVENT_NOTIFICATION_TYPE, your consumer can use EventNotification.getEvent() to attempt to retrieve the new or updated event. This method is valid only for notifications of new or updated events.
- If the notification is REMOVE_EVENT_NOTIFICATION_TYPE, your consumer can use EventNotification.getGlobalInstanceId() to retrieve the global instance identifier of the deleted event. This method is valid only for notifications of deleted events.

```
for (int i = 0; i < notifications.length; i++)
{
  int notifType = notifications[i].getNotificationType();

  if(notifType == NotificationHelper.CREATE_EVENT_NOTIFICATION_TYPE)
  {
    CommonBaseEvent event = notifications[i].getEvent();
    if (event != null) {
      // process the new event
      // ...
    }
  }

  else if(notifType == NotificationHelper.UPDATE_EVENT_NOTIFICATION_TYPE)
  {
    CommonBaseEvent event = notifications[i].getEvent();
    if (event != null) {
      // process the updated event
      // ...
    }
  }

  else if(notifType == NotificationHelper.REMOVE_EVENT_NOTIFICATION_TYPE)
  {
    String eventId = notifications.[i].getGlobalInstanceId();
    // process the event deletion
    // ...
  }
}
```

In its deployment descriptor, a message-driven bean must be associated with a listener port, which specifies a JMS destination and connection factory. You must create a listener port for your event consumer before deploying the MDB, specifying the destination and connection factory associated with the event group from which you want to receive events (these are defined in the event group profile).

Note: Do not use the CommonEventInfrastructure_ListenerPort listener port when deploying your MDB. This listener port is used by the event server and is not intended for use by event consumers.

Developing a non-MDB event consumer:

An event consumer can also be created using a non-message driven bean.

To write an event consumer that is not a message-driven bean, follow these steps:

1. Obtain a notification helper. A JMS event consumer uses a notification helper to identify JMS destinations associated with an event group, to convert received JMS messages into event notifications, and to perform filtering of received events. To obtain a notification helper, use a notification helper factory, which is

an instance of NotificationHelperFactory that has been bound into a JNDI namespace. The following code fragment uses a notification helper factory to obtain a notification helper.

```
// Get notification helper factory from JNDI
InitialContext context = new InitialContext();
Object notificationHelperFactoryObject =
  context.lookup("com/ibm/events/NotificationHelperFactory");
NotificationHelperFactory nhFactory = (NotificationHelperFactory)
  PortableRemoteObject.narrow(notificationHelperFactoryObject,
                              NotificationHelperFactory.class);

// Create notification helper
NotificationHelper notificationHelper =
  nhFactory.getNotificationHelper();
```

2. **Optional:** Specify the event selector. If you want to filter received events, you can use the setEventSelector() method to set an event selector on the notification helper. Your event consumer can then use the notification helper to check received events against the event selector. The following code fragment sets an event selector specifying events with severity greater than 30 (warning).

   ```
   notificationHelper.setEventSelector("CommonBaseEvent[@severity > 30]");
   ```

3. Use the notification helper to find the JMS destination to subscribe to.

   Each event group can be associated with a single JMS topic and any number of JMS queues. You can query the notification helper to find out what destinations are associated with a particular event group. To find the topic associated with an event group, use the getJmsTopic(String) method of NotificationHelper, specifying the name of the event group:

   ```
   MessagePort msgPort = notificationHelper.getJmsTopic("critical_events");
   ```

   To find the queues associated with an event group, use the getJmsQueues(String) method:

   ```
   MessagePort[] msgPorts = notificationHelper.getJmsQueues("critical_events");
   ```

   The returned object is either a single MessagePort object representing a JMS topic or an array of MessagePort objects representing JMS queues. A MessagePort instance is a wrapper object containing the JNDI names of the destination and its connection factory.

4. Connect to the destination. Use the getter methods of MessagePort to retrieve the JNDI names of the destination and connection factory. You can then use the standard JMS interfaces to connect to the destination. The following code fragment subscribes to a JMS topic:

   ```
   String connectionFactoryName = msgPort.getConnectionFactoryJndiName();
   String destinationName = msgPort.getDestinationJndiName();

   // create connection and session
   ConnectionFactory connectionFactory =
        (ConnectionFactory) context.lookup(connectionFactoryName);
   Connection connection = connectionFactory.createConnection();
   Session session = connection.createSesion(false,
                                             Session.CLIENT_ACKNOWLEDGE);

   // Create consumer and register listener
   Topic topic = (Topic) context.lookup(destinationName);
   MessageConsumer consumer = session.createConsumer(topic);
   consumer.setMessageListener(this);
   connection.start();
   ```

5. Convert received messages into event notifications.

   In the onMessage() method of your listener, use the notification helper to convert each received JMS message into an array containing an event notification. (If the event does not match the event selector specified on the Notification Helper, the array is empty.) An event notification is an instance of a class implementing the EventNotification interface.

```
public void onMessage(Message msg) {
  EventNotification[] notifications =
                    notificationHelper.getEventNotifications(msg);
  // ...
```

6. Check the notification type and retrieve the event data as appropriate. Each event notification has a field representing the notification type (an integer whose value is one of the notification type constants defined by the NotificationHelper interface). Three notification types are currently supported:

Notification type	**Description**
`CREATE_EVENT` `_NOTIFICATION_TYPE`	A new event has been created in the event group associated with the destination. This means either that a new event has been sent, or that an existing event has changed so that it now matches the event group definition. The notification also contains the complete event data.
`REMOVE_EVENT` `_NOTIFICATION_TYPE`	An event stored in the event database has been removed from the event group associated with the destination. This means either that an event has been deleted from the event database, or that an existing event has changed so that it no longer matches the event group definition. The notification also contains the global instance identifier of the deleted event.
`UPDATE_EVENT` `_NOTIFICATION_TYPE`	An event stored in the event database has been updated in a way that does not change its membership in the event group associated with the destination. The notification also contains the complete event data.

Use the getNotificationType() method of EventNotification to check the notification type of each received notification. Based on the notification type, you can determine whether your event consumer should process the notification further, and what kind of event data the notification contains:

- If the notification type is CREATE_EVENT_NOTIFICATION_TYPE or UPDATE_EVENT_NOTIFICATION_TYPE, your consumer can use EventNotification.getEvent() to attempt to retrieve the new or updated event. This method is valid only for notifications of new or updated events.
- If the notification is REMOVE_EVENT_NOTIFICATION_TYPE, your consumer can use EventNotification.getGlobalInstanceId() to retrieve the global instance identifier of the deleted event. This method is valid only for notifications of deleted events.

```
for (int i = 0; i < notifications.length; i++)
{
  int notifType = notifications[i].getNotificationType();

  if(notifType == NotificationHelper.CREATE_EVENT_NOTIFICATION_TYPE)
  {
    CommonBaseEvent event = notifications[i].getEvent();
    if (event != null) {
      // process the new event
      // ...
    }
  }
```

```
      else if(notifType == NotificationHelper.UPDATE_EVENT_NOTIFICATION_TYPE)
      {
        CommonBaseEvent event = notifications[i].getEvent();
        if (event != null) {
          // process the updated event
          // ...
        }
      }

      else if(notifType == NotificationHelper.REMOVE_EVENT_NOTIFICATION_TYPE)
      {
        String eventId = notifications.[i].getGlobalInstanceId();
        // process the event deletion
        // ...
      }
    }
```

Querying events from the event server:

An event consumer can synchronously retrieve historical events from the persistent data store by querying the event server.

The persistent data store is implemented as a separate component called a *data store plug-in*. The Common Event Infrastructure includes a default data store plug-in, which supports event queries based on a subset of XPath syntax. If you want to use a different data store, you can implement your own data store plug-in.

To query the event server, use the event access interface.

1. Create an event access bean.

   This bean is a Java 2 Platform, Enterprise Edition (J2EE) stateless session bean. The bean interface provides methods for querying the event server. An event consumer uses an instance of the event access bean for all synchronous event queries.
2. Query events.

   You can query events in the following ways:
   - Specify a global instance identifier to retrieve a specific single event.
   - Specify an event group to retrieve events associated with that event group. You can optionally refine the query by specifying an additional event selector. This action retrieves only those events that match both the event group and the event selector.
   - Specify a known event and an association type to retrieve events that are associated with the known event.
   - Query and purge events.

Creating an event access bean:

Getting event access starts with creating an instance of the event access bean.

The event access interface is implemented as a stateless session bean using the Enterprise JavaBeans architecture. To query the event server using the event access interface, an event source must first create an instance of the event access session bean. The event access bean can be either local or remote.

To create an instance of the event access session bean, use the appropriate home interface: com.ibm.events.access.EventAccessHome or com.ibm.events.access.EventAccessLocalHome.

```
// use home interface to create remote event access bean
InitialContext context = new InitialContext();
Object eventAccessHomeObj = context.lookup("ejb/com/ibm/events/access/EventAccess");
EventAccessHome eventAccessHome = (EventAccessHome)
  PortableRemoteObject.narrow(eventAccessHomeObj,
                              EventAccessHome.class);
eventAccess = (EventAccess) eventAccessHome.create();
```

Querying events by global instance identifier:

Events can be queried by a primary key.

The Common Base Event specification defines a globalInstanceId event property that can be used as a primary key for event identification. The content of this property is a globally unique identifier that is generated either by the application or by the emitter. Although the Common Base Event specification defines the globalInstanceId property as optional, the event emitter automatically assigns an identifier to any event that does not already have an identifier.

You can retrieve a specific single event from the event server by querying with the globalInstanceId property of the event that you want to retrieve. This query can be useful for testing purposes (to confirm that events are stored in the event database), or to retrieve an event associated with one that was received previously.

To query an event by the global instance identifier, use the queryEventByGlobalInstanceId method of the event access bean.

1. **Optional:** Create an event access bean.
2. Call the queryEventByGlobalInstanceId(String) method of the EventAccess bean, specifying the global instance identifier of the event that you want to retrieve.
   ```
   CommonBaseEvent event = eventAccess.queryEventByGlobalInstanceId(eventId);
   ```
   The returned object is the event with the specified global instance identifier. If there is no matching event in the persistent data store, the returned object is null.

Querying events by event group:

To make specified events easier to access, they can be associated with event groups.

You can associate an event with one or more *event groups*. An event group is a logical grouping of events that match a particular event selector. Event groups are defined in the event infrastructure configuration. For more information about event groups, see *Default configuration*.

You can use the event access interface to retrieve events that belong to a specified event group. You can restrict the query results by specifying an additional event selector. You can also query events without retrieving them.

You can query event groups in the following ways:

Querying a limited number of events from an event group:

You can query a limited number of events from an event group.

To query a limited number of events from an event group, use the queryEventsByEventGroup(String, String, Boolean, int) method of the EventAccess bean.

1. If you need to create an event access bean, see *"Creating an event access bean" on page 188*.
2. Call the EventAccess.queryEventsByEventGroup(String, String, boolean, int) method.

```
CommonBaseEvent[] events = eventAccess.queryEventsByEventGroup(eventGroup,
                                                               eventSelector,
                                                               ascendingOrder,
                                                               maxEvents);
```

   The parameters of this method are as follows:

   eventGroup
      A string containing the name of the event group you want to query events from. This must be the name of an existing event group defined in the event infrastructure configuration.

   eventSelector
      A string containing an optional event selector that further refines the query. The query returns only events that match both the specified event group and the additional event selector. An event selector is specified in the form of an XPath expression (for more information, see *"Writing event selectors" on page 193*). If you do not want to specify an additional event selector, this parameter can be null.

   ascendingOrder
      A boolean value specifying whether the returned events are to be sorted in ascending or descending order according to the value of the creationTime property. If this parameter is *true*, the events are sorted in ascending (chronological) order; if it is *false*, they are sorted in descending (reverse chronological) order.

   maxEvents
      An integer specifying the maximum number of events you want returned.

   The returned object is an array containing the events from the specified event group.

   Note: If the number of matching events exceeds the query threshold defined in the data store profile, a QueryThresholdExceededException exception is thrown. The default query threshold is 100 000.

The following code fragment returns all events that belong to an event group called *critical_hosts* and whose severity is greater than 30 (warning), but specifies that no more than 5000 matching events should be returned:

```
CommonBaseEvent[] events =
        eventAccess.queryByEventGroup("critical_hosts",
                                      "CommonBaseEvent[@severity > 30]",
                                      true,
                                      5000);
```

Querying all events from an event group:

To query all events from an event group, use the queryEventsByEventGroup(String, String, boolean) method of the EventAccess bean.

1. If you need to create an event access bean, see *"Creating an event access bean" on page 188*.

2. Call the EventAccess.queryEventsByEventGroup(String, String, boolean) method.

   ```
   CommonBaseEvent[] events = eventAccess.queryEventsByEventGroup(eventGroup,
                                                                  eventSelector,
                                                                  ascendingOrder);
   ```

   The parameters of this method are as follows:

   eventGroup
   : A string containing the name of the event group you want to query events from. This must be the name of an existing event group defined in the event infrastructure configuration.

   eventSelector
   : A string containing an optional event selector that further refines the query. The query returns only events that match both the specified event group and the additional event selector. An event selector is specified in the form of an XPath expression (for more information, see *"Writing event selectors" on page 193*). If you do not want to specify an additional event selector, this parameter can be null.

   ascendingOrder
   : A boolean value specifying whether the returned events are to be sorted in ascending or descending order according to the value of the creationTime property. If this parameter is *true*, the events are sorted in ascending (chronological) order; if it is *false*, they are sorted in descending (reverse chronological) order.

   The returned object is an array containing the events from the specified event group.

   Note: If the number of matching events exceeds the query threshold defined in the data store profile, a QueryThresholdExceededException exception is thrown. The default query threshold is 100 000.

The following code fragment returns all events that belong to an event group called *critical_hosts* and whose severity is greater than 30 (warning):

```
CommonBaseEvent[] events =
       eventAccess.queryByEventGroup("critical_hosts",
                                     "CommonBaseEvent[@severity > 30]",
                                     true);
```

Querying the existence of events in an event group:

You can determine the existence of events without retrieving them.

In some situations, you might want to find out whether any events exist in a particular event group without actually retrieving the events. To do this, use the eventExists() method of the event access bean.

1. If you need to create an event access bean, see *"Creating an event access bean" on page 188*
2. Call the eventExists(String, String) method of the EventAccess bean.

   ```
   boolean hasEvents = eventAccess.eventExists(eventGroup,
                                               eventSelector);
   ```

   The parameters of this method are as follows:

eventGroup
: A string containing the name of the event group you want to check for events. This must be the name of an existing event group defined in the event infrastructure configuration.

eventSelector
: A string containing an optional event selector that further refines the query. The query only checks for events that match both the specified event group and the additional event selector. An event selector is specified in the form of an XPath expression (for more information, see *"Writing event selectors" on page 193*). If you do not want to specify an additional event selector, this parameter can be null.

The returned boolean object equals `true` if any events exist that match the specified event group and event selector, `false` if none exist.

The following code fragment checks for the existence of any events in an event group called *critical_hosts* and retrieves any that exist.

```
if (eventAccess.eventExists("critical_hosts",null)) {
   CommonBaseEvent[] events =
     eventAccess.queryByEventGroup("critical_hosts",
                                   null,
                                   true);
   }
```

Querying events by association type:

Events can be queried by association types.

The Common Base Event specification defines properties that establish relationships between events. The associatedEvents property is a complex element that contains one or more subelements of the AssociatedEvent type, each representing an associated event. Each AssociatedEvent element, contains subelements that identify the type of association and the application that established the association. Examples of association types might include CausedBy or Correlated.

By specifying the global instance identifier of a known event and a type of association, you can retrieve events that satisfy the specified association. To query events by association type, use the EventAccess.queryEventsByAssocation(String, String) method.

1. **Optional:** Create an event access bean.
2. Call the EventAccess.queryEventsByAssociation(String, String) method.

   ```
   CommonBaseEvent[] events = eventAccess.queryEventsByAssocation(associationType,
                                                                  eventId);
   ```

   The parameters of this method are as follows:

   associationType
   : The type of association. This is the name of an association type specified by the associationEngineInfo property.

   eventId
   : The global instance identifier of a known event.

   The returned object is an array that contains the events that satisfy the specified type of association with the known event. Only events that are still in the event database at the time of the query are returned (an associated event might be purged from the database).

The following code fragment returns all of the events from the event database that have a `CausedBy` association with a known event:

```
String eventId = causeEvent.getGlobalInstanceId();
CommonBaseEvent[] resultEvents = eventAccess.queryEventsByAssociation("CausedBy",
                                                                     eventId);
```

Deleting events from the data store:

An event consumer or administrative tool can delete events from the data store using the event access interface.

You can delete all events from the data store, or you can limit the deleted events by specifying event groups, event selectors, or both.

Note: If WebSphere security is enabled, the application user ID must be mapped to the eventAdministrator role to delete events.

To delete events from the data store, use the purgeEvents() method of the event access bean.

```
int purged = eventAccess.purgeEvents(eventGroup,
                                     eventSelector,
                                     transactionSize);
```

The parameters are as follows:

eventGroup
: A string containing the name of the event group that includes the events you want to delete. This must be the name of an existing event group defined in the event infrastructure configuration. If you do not want to specify an event group, this parameter can be null.

eventSelector
: A string containing an optional event selector that identifies the events to delete. An event selector is specified in the form of an XPath expression (for more information, see *"Writing event selectors"*). If you do not want to specify an event selector, this parameter can be null.

transactionSize
: A nonzero integer specifying the number of events you want deleted in a single database transaction. In most cases, you can use the constant DEFAULT_PURGE_TRANSACTION_SIZE, which is defined by the EventAccess interface.

The purgeEvents() method deletes all of the events that match all of the criteria you specify. If the *eventGroup* and *eventSelector* parameters are both null, all events in the data store are deleted. Events that arrive after the delete operation starts are not purged. The returned value is an integer specifying how many events were deleted.

Note: If the value of the *transactionSize* parameter exceeds the maximum purge transaction size defined in the data store profile, a PurgeThresholdExceededException exception is thrown and no events are deleted. The default maximum purge transaction size is 100 000.

Writing event selectors:

An event selector defines a set of events.

An event selector is a regular expression that defines a set of events based on their property data (attributes or sub-elements). For example, an event selector might specify all events from a particular host whose severity is greater than 30 (warning). Event selectors are used to define event groups, specify filter criteria, and query the event server.

Because the Common Base Event specification is based on XML, event selectors are written using a subset of XPath syntax. The specific syntax you can use for an event selector depends on how the event selector is to be used, as summarized by the following table.

Event selector purpose	Syntax
Event group definition	Limited to XPath subset supported by default data store plug-in
Event query and purge through event access interface	Limited to XPath subset supported by default data store plug-in
Emitter filter configuration	Any valid XPath
Subscription through Notification Helper interface	Any valid XPath

Note: The default data store plug-in uses a subset of XPath syntax. However, if you are using a different data store plug-in, it might support a different subset of XPath. The event selectors you write for event group definitions and for the event access interface must use the syntax that is supported by your data store plug-in.

Writing XPath event selectors

XPath is a standard language used to identify parts of an XML document; for more information, see the XPath specification at http://www.w3.org/TR/xpath.

A simple XPath event selector that specifies an attribute value takes the following form:

```
CommonBaseEvent[@attribute = value]
```

The *value* can be either a numeric value or a string enclosed in single or double quotation marks.

You can also specify an attribute of a subelement:

```
CommonBaseEvent[/subelement/@attribute = value]
```

When using XPath operators, keep the following general rules in mind:

- When used to compare XML dateTime values, the comparison operators perform logical comparisons that recognize time zone differences.
- Logical operators and function names must be specified using all lowercase letters (for example, `and` rather than `AND`).
- Operators must be separated with white space from the surrounding attribute names and values (`@severity > 30` rather than `@severity>30`).
- Parentheses can be used to change operator precedence.

The following examples are valid XPath event selectors.

XPath event selectors	Selector definition

`CommonBaseEvent[@extensionName = 'ApplicationStarted']`	All events with the *extensionName* attribute `ApplicationStarted`
`CommonBaseEvent[sourceComponentId/ @location = "server1"]`	All events containing a sourceComponentId element with the *location* attribute `server1`
`CommonBaseEvent[@severity]`	All events with a *severity* attribute, regardless of its value
`CommonBaseEvent[@creationTime < '2003-12-10T12:00:00-05:00' and @severity > 30]`	All events created before noon EST on 10 December 2003 and with severity greater than 30 (warning):
`CommonBaseEvent[contains(@msg, 'disk full')]`	All events with the phrase `disk full` occurring within the msg attribute
`CommonBaseEvent[(@severity = 30 or @severity = 50) and @priority = 100]`	All events whose *severity* atrribute is equal to 30 or 50, and whose *priority* is equal to 100.

Writing event selectors for the default data store plug-in

If your event selector might be used to define an event group or to query the persistent data store, it is subject to the restrictions of the default data store plug-in. These restrictions are as follows:

- An event property can be specified only on the left side of an operator or XPath function. The value on the right side of an operator must be a literal value. The following example is not a valid event selector:
  ```
  CommonBaseEvent[30 < @priority and
                  contains('this message', @msg)]
  ```
  Instead, this could be rewritten as follows:
  ```
  CommonBaseEvent[@priority > 30 and
                  contains(@msg, 'this message')]
  ```
- Only the following XPath functions are supported:
  - contains
  - starts-with
  - false
  - true
  - not
- The union operator (|) is not supported.
- An event selector must take the following form:
  ```
  CommonBaseEvent[predicate_expression]
  ```
  Only a single predicate expression can be associated with the CommonBaseEvent element. Stacked predicates are not supported (for example, `CommonBaseEvent[@extensionName = "server_down"][@severity = 10]`).
- A predicate can be only be associated with the last step of a location path. The following example is not a valid event selector:
  ```
  CommonBaseEvent[contextDataElement[@contextValue = "myContextValue"]
                  /@contextId = "myContextId"]
  ```
  Instead, this could be rewritten as follows:
  ```
  CommonBaseEvent[contextDataElement[@contextValue = "myContextValue"
                  and @contextId = "myContextId"]]
  ```
- If an event selector refers to properties of extended data elements that are at different levels of the XML containment hierarchy, these elements must be

grouped together by level. The following example is not a valid event selector, because the references to the *type* and *value* attributes (both top-level) of *extendedDataElements* are separated:

```
CommonBaseEvent[extendedDataElements[@type = 'int' and
                                     children/@type = 'intArray' and
                                     children/@name = 'myName' and
                                     @value = 10]]
```

Instead, this could be rewritten as follows, grouping the top-level and second-level attributes together:

```
CommonBaseEvent[extendedDataElements[@type = 'int' and
                                     @value = 10 and
                                     children/@type = 'intArray' and
                                     children/@type = 'myName']]
```

- Node indexes are not supported (for example, `CommonBaseEvent[extendedDataElements[1]]`).
- Wildcard characters are not supported (for example, `CommonBaseEvent[extendedDataElements/*/children/values = "text"]`).
- When referring to the *values* property of an extended data element, you must specify not only the value but also the type of the property:

```
CommonBaseEvent[extendedDataElements[values = "myVal"
  and @type = "string"]]
```

You can specify the type for multiple comparisons within a compound expression by grouping them with parentheses:

```
CommonBaseEvent[extendedDataElements[(values = "myVal" or
                                      values = "yourVal") and
                                      @type = "string"]]
```

In this example, the *type* expression applies to both parts of the compound expression in parentheses. You cannot override this by specifying a different *type* expression inside the parentheses.

You can also group together multiple related types by using the `starts-with` or `contains` functions. For example, the following expression would match a property with either the string or stringArray type:

```
CommonBaseEvent[extendedDataElements[values = "myVal" and
                                     starts-with(@type, 'string')]]
```

Creating an event catalog application

The *event catalog* is a repository of event metadata.

The *event catalog* metadata consists of event definitions, which describe classes of events and their allowed content. (This is distinct from the event instance metadata you can access using the Eclipse Modeling Framework interfaces described in *"Accessing event instance metadata" on page 171*.) Applications can use the event catalog to manage their enterprise-specific event definitions, but must implement their own validation logic to ensure that events conform to these definitions.

Events defined according to the Common Base Event specification can be categorized into event classes based upon extension name (the value of the *extensionName* attribute). Using the event catalog, you can define the permitted content of a particular class of event by specifying what extended data elements events of that class can contain, as well as the permitted values for other Common Base Event properties. An event definition defines constraints on event content above and beyond those of the Common Base Event specification.

Event definitions are defined hierarchically and inherit the definitions of their parents. A single root event definition, `event`, defines the basic requirements of any event that conforms to the Common Base Event specification. All other event definitions inherit from this root definition. By default, this root event definition is automatically installed in the Event Catalog, along with event definitions for Event Catalog notification events (for more information, see *"Change notification" on page 201*).

Note: Currently, event definitions do not support all of the forms of constraints required to fully describe the Common Base Event specification (for example, the requirement that the *globalInstanceId* property must begin with an alphabetic character). Therefore, it is possible that an event might conform to the `event` event definition and still not pass validation by the event emitter.

By using the event catalog interfaces, you can create, delete, and query event definitions. (Once created, an event definition cannot be modified.) You can also list existing event definitions in a readable format, as well as importing and exporting event definitions in XML format.

Event definitions:

Event definitions are comprised of several types of information.

An *event definition* contains several kinds of information:

Name The name of the event definition, which is the same as the extension name of the events described by the definition. All events with a particular extension name share the same event definition.

Parent The name of the parent event definition. Any event definition (with the exception of the root definition `event`) has a parent event definition from which it inherits property descriptions and extended data element descriptions (although some aspects of the inherited data can be overridden). The parent can be any valid event definition that exists in the event catalog.

Property descriptions
Descriptions of the permitted Common Base Event properties for the event definition. A property description can describe any property defined in the Common Base Event specification as a simple type, including properties of complex subelements.

Extended data element descriptions
Descriptions of the permitted extended data elements for the event definition. An extended data element description defines the name and type of the extended data element; it can also define default values, how many of the extended data element are allowed, and descriptions of child extended data elements.

Represented as an XML document, an event definition takes the following general form:

```
<eventDefinition name="eventDefinitionName"
                parent="parentEventDefinitionName">
     <property name="propertyName" ... />
     <extendedDataElement name="extendedDataElementName"
                          type="type" ... />
</eventDefinition>
```

Property descriptions:

Property descriptions are defined by the Common Base Event specification.

A property description describes a property that an event can contain. This can be any property defined by the Common Base Event specification as a simple type. A property description cannot describe a complex property such as *msgDataElement*, but it can describe a simple property that is a child of a complex property. An event definition can contain any number of property descriptions (including none).

A property description includes the following fields:

name The name of the property. This must be the name of an attribute of the CommonBaseEvent element, or an attribute of a complex subelement of CommonBaseEvent. Some examples are `severity`, `priority`, and `globalInstanceId`.

path An XPath location path specifying the path to the property, if the property is not an attribute of CommonBaseEvent. The path identifies the parent property of the property being described. These are examples:

- To describe a property of CommonBaseEvent such as *severity*, do not specify a path. A null path specifies a top-level property.
- To describe a property of *msgDataElement*, which is a complex property of CommonBaseEvent, you specify the path `msgDataElement`.
- To describe a property of *msgHelp*, which is itself a complex property of *msgDataElement*, specify the path `msgDataElement/msgHelp`.

The path can also describe a specific instance of a repeated property. For example, if an event definition describes several *contextDataElements* properties, you might specify one called *businessContext*, you would use the path `contextDataElements[@name='businessContext']`.

defaultValue
The default value of the property. The default value represents the value that should be used during content completion for an event that is missing a required property. (Therefore, it is meaningful for a property description to be required and to define a default value.) This field is optional.

required
A boolean value specifying whether the property is required or optional. If this field is equal to `true`, the property is required. This field is optional; if it is not specified, the property is assumed to be optional.

permittedValue
A permitted value for the property. If an event definition allows only certain values for a property, each one is represented by a *permittedValue* field in the property description. A property description can include any number of permitted values. This field is optional and must not be specified if the *minValue* or *maxValue* fields are specified.

minValue
maxValue
The minimum and maximum permitted values for the property. If an event definition allows a range of values for a property, these fields defines the lower and upper bounds of that range. If you specify only *minValue*, the permitted range has no upper bound; similarly, if you specify only *maxValue*, the permitted range has no lower bound. These fields are optional and must not be specified if *permittedValue* fields are specified.

Extended data element descriptions:

Extended data elements are one type of information an event definition may contain.

An extended data element description describes an extended data element that an event of a particular event class can contain. An event definition can contain any number of extended data element descriptions (including none).

An extended data element description includes the following fields:

name The name of the extended data element. This defines the value of the *name* attribute of the element.

type The data type of the extended data element. This defines the value of the *type* attribute of the element. This must be one of the following supported data types:
- noValue
- byte
- short
- int
- long
- float
- double
- string
- dateTime
- boolean
- byteArray
- shortArray
- intArray
- longArray
- floatArray
- doubleArray
- stringArray
- dateTimeArray
- booleanArray
- hexBinary

defaultValue
The default value of the extended data element, or multiple default values if the type is an array. The default value represents the value that should be used during content completion for an event that is missing a required extended data element. This field is optional.

minOccurs
The minimum number of instances of the extended data element that must appear. This field is optional; the default value is 1.

maxOccurs
The maximum number of instances of the extended data element that can appear. This field is optional; the default value is 1.

Note: The current Common Base Event specification allows only one extended data element with a given name at each level of the event containment hierarchy, but this restriction will not be included in future versions of the specification and is not enforced by the Common Event Infrastructure.

Event catalog inheritance:

Event definitions inherit the properties of their parents.

By default, an event definition inherits the property descriptions and extended data element descriptions of its parent. However, a child event definition can override these inherited descriptions, subject to certain restrictions. When you add an event definition to the event catalog, the catalog verifies that the new event definition does not violate the rules governing inheritance; if it does, an InheritanceNotValidException exception is thrown. Similarly, if you replace an existing event definition that has descendants, the event catalog verifies the validity of the existing inheritance relationships and throws an InheritanceNotValidException exception if any of them are no longer valid. In either case, the new event definition is not added to the catalog unless all inheritance relationships are valid.

An event definition can exist in either of two forms, *unresolved* and *resolved*:

- An unresolved event definition includes only those property definitions and extended data element descriptions that are defined within the event definition itself.
- A resolved event definition includes the data in the unresolved event definition in addition to the property definitions and extended data element descriptions it inherits.

Overriding inherited property descriptions

A child event definition inherits each property description from its parent without change unless it already has a locally defined property description of the same name and path (note that case is significant). If the child has a property description of the same name and path, the fields of the child description can override the fields of the parent description as follows:

Default value
The child can override the default value specified by the parent property description. If the child does not specify a default value, it inherits the value from the parent.

Required or optional
The child always overrides the parent. However, if the parent defines a property as required, the child must also specify that it is required. An inherited required property cannot be redefined as optional.

Permitted values or minimum and maximum values
If the parent defines permitted values or minimum and maximum values, the child can override these by specifying either permitted values or minimum and maximum values. Note that an event definition can contain only permitted values or minimum and maximum values, not both:

- If the parent defines minimum and maximum values, but the child defines permitted values, the minimum and maximum values defined by the parent are ignored.

- If the parent defines permitted values, but the child defines minimum and maximum values, the permitted values defined by the parent are ignored.
- If the parent defines only a maximum value, but the child defines only a minimum value, the child inherits the maximum value defined by the parent.
- If the child does not specify permitted values or minimum and maximum values, the values specified by the parent are inherited.

Overriding inherited extended data element descriptions

A child event definition inherits each extended data element description from its parent without change unless it already has a locally defined extended data element description of the same name. If the child does have an extended data element description of the same name, the fields of the child description can override the fields of the parent description as follows:

Type The child must specify the same type as the parent.

Minimum occurrence
The child always overrides the parent.

Maximum occurrence
The child always overrides the parent.

Default values
The child can override the default values specified by the parent extended data element description. If the child does not specify default values, it inherits the values from the parent.

Default hexadecimal value
The child can override the default hexadecimal value specified by the parent extended data element description. If the child does not specify a default hexadecimal value, it inherits the value from the parent.

Nested extended data element description
The child can override a nested extended data element description by defining a nested description of the same name. If the child overrides an inherited nested description, the same rules apply to overriding the individual fields. If the child does not specify a nested extended data element description of the same name, it inherits the nested description from the parent.

Change notification:

Each time an event definition is added, removed, or replaced, the event catalog sends an event to the event server indicating that this has happened.

An event consumer can subscribe to these events to receive notification of changes in the event catalog. By default, the event catalog uses the default emitter factory to obtain an emitter for sending these events; however, this can be changed in the Event Catalog configuration.

The event catalog can send three classes of notification events, using the following extension names:
- cei_event_definition_added
- cei_event_definition_replaced
- cei_event_definition_removed

These three event classes inherit property descriptions from a common parent class, cei_event_definition. Event definitions for all four event classes are automatically loaded into the event catalog during installation, along with the default root event definition.

Note: When an event definition is removed from the event catalog, any children or other descendants of that event definition are also removed. The event catalog sends a separate change notification event for each event definition that is removed.

Each change notification event contains the following properties:

Property	Value
version	`1.0.1`
globalInstanceId	A globally unique identifier for the event
creationTime	Current® date and time when the event is generated
severity	10 (information)
priority	10 (low)
sourceComponentId	Identification of the Event Catalog component and event server host machine
situation	Situation data, including one of the following values for situation category: • CreateSituation (event definition added) • ConfigureSituation (event definition replaced) • DestroySituation (event definition removed)
extensionName	One of the following values: • cei_event_definition_added • cei_event_definition_replaced • cei_event_definition_removed
extendedDataElements	A single extended data element with one attribute, *eventDefinitionName*. This attribute is a string specifying the name of the event definition that has been added, replaced, or removed.

Creating an event definition:

Event definitions are instances of the EventDefinition class.

An event definition is an instance of the class EventDefinition. To create an event definition, first create a new instance of this class and then populate it with property descriptions and extended data element descriptions. After you have created an event definition, you can add it to the event catalog; for more information, see *"Adding an event definition to the event catalog" on page 205.*

To create a new, empty event definition, create an instance of EventDefinition:

```
EventDefinition definition = new EventDefinition(name, parent);
```

The parameters of this constructor are as follows:

name
: The name of the event definition. This is the value of the *extensionName* attribute for the events you are describing.

parent
: The name of the parent event definition. If you do not want your event definition to inherit any property descriptions or extended data element descriptions other than those required by the Common Base Event specification, this parameter should be `event`. If this parameter is null, the new event definition is defined as a root event definition; a root event definition can only be added to the catalog if it is empty, or if you intend to replace the current root event definition.

The returned object is a new unresolved event definition containing no property descriptions or extended data element descriptions.

The following code fragment creates a new event definition called `insurance_claim_start_auto`, which is a child of the event definition `insurance_claim_start`:

```
EventDefinition definition = new EventDefinition("insurance_claim_start_auto",
                                                 "insurance_claim_start");
```

You can now populate the event definition with property descriptions and extended data element descriptions.

Adding property descriptions to an event definition:

A property description is an instance of the class PropertyDescription.

To add a property description to an event definition, you must first create a new property description and then set the values of its fields. You can then add the property description to the event definition.

1. To create a new property description, create an instance of PropertyDescription, specifying the name and path of the property.

   ```
   PropertyDescription propDesc = new PropertyDescription(name, path);
   ```

   The parameters of this constructor are as follows:

   name
   : The name of the property. This must be the name of a simple property either of the CommonBaseEvent element or one of its children.

   path
   : An XPath location path specifying the path to the property. For a top-level property of CommonBaseEvent (such as *severity* or *priority*), *path* should be null.

   The returned object is a new PropertyDescription object.
2. Populate the fields of the property description. The PropertyDescription class provides a setter method for each of the fields in a property description. Refer to the Javadoc API documentation for complete information about these methods. For example, to specify that a property is required, you would set the *required* property to `true` using the setRequired(boolean) method:

   ```
   propDesc.setRequired(true);
   ```
3. Add the property description to the event definition using the EventDefinition.addPropertyDescription() method.

   ```
   definition.addPropertyDescription(propDesc);
   ```

   If the event definition already includes another property description with the same name and path, a DescriptionExistsException exception is thrown.

The following code fragment creates a new property description, populates it with data, and adds it to an event definition.

```
PropertyDescription propDesc = new PropertyDescription("severity",null);
propDesc.setRequired(true);
propDesc.setMinValue('30');

// definition is a valid event definition
definition.addPropertyDescription(propDesc);
```

Adding extended data element descriptions to an event definition:

An extended data element description is an instance of the ExtendedDataElementDescription class.

To add an extended data element description to an event definition, you must first create a new extended data element description and then set the values of its fields. You can also add nested (child) extended data element descriptions, which describe nested extended data elements. You can then add the extended data element description to the event definition.

1. To create a new extended data element description, create an instance of ExtendedDataElementDescription, specifying the name and type of the extended data element.

   ```
   ExtendedDataElementDescription edeDesc =
        new ExtendedDataElementDescription(name, type);
   ```

   The parameters of this constructor are as follows:

   name
      The name of the extended data element. This must be the value of the *name* property of the extended data element you want to describe.

   type
      The data type of the extended data element. This must be one of the following integer constants defined by the org.eclipse.hyades.logging.events.cbe.ExtendedDataElement class:
      - TYPE_BOOLEAN_ARRAY_VALUE
      - TYPE_BOOLEAN_VALUE
      - TYPE_BYTE_ARRAY_VALUE
      - TYPE_BYTE_ARRAY
      - TYPE_DATE_TIME_ARRAY_VALUE
      - TYPE_DATE_TIME_VALUE
      - TYPE_DOUBLE_ARRAY_VALUE
      - TYPE_DOUBLE_VALUE
      - TYPE_FLOAT_ARRAY_VALUE
      - TYPE_FLOAT_VALUE
      - TYPE_HEX_BINARY_VALUE
      - TYPE_INT_ARRAY_VALUE
      - TYPE_INT_VALUE
      - TYPE_LONG_ARRAY_VALUE
      - TYPE_LONG_VALUE
      - TYPE_NO_VALUE_VALUE
      - TYPE_SHORT_ARRAY_VALUE
      - TYPE_SHORT_VALUE

- TYPE_STRING_ARRAY_VALUE
- TYPE_STRING_VALUE

The returned object is a new ExtendedDataElementDescription object.

2. Populate the fields of the extended data element description. The ExtendedDataElementDescription class provides a setter method for each of the fields in an extended data element description. Refer to the Javadoc API documentation for complete information about these methods. For example, to specify that an extended data element must occur at least once, you would set the *maxOccurs* property to 4 using the setMaxOccurs(int) method:
```
edeDesc.setMaxOccurs(4);
```
3. **Optional:** To add a child extended data element description, use the ExtendedDataElementDescription.addChild() method.
```
edeDesc.addChild(childEdeDesc);
```
The *childEdeDesc* parameter must be a valid extended data element description.
4. Add the extended data element description to the event definition using the EventDefinition.addExtendedDataElementDescription() method.
```
definition.addExtendedDataElementDescription(edeDesc);
```
If the event definition already includes another extended data element description with the same name and path, a DescriptionExistsException exception is thrown.

The following code fragment creates a new extended data element description, populates it with data, and adds it to an event definition.
```
ExtendedDataElementDescription edeDesc =
     new ExtendedDataElementDescription("age", TYPE_SHORT_VALUE);
edeDesc.setMinOccurs(1);
edeDesc.setMaxOccurs(1);

// definition is a valid event definition
definition.addExtendedDataElementDescription(edeDesc);
```

Creating an event catalog bean:

An event catalog bean is used to access the event catalog.

The event catalog is implemented as a stateless session bean using the Enterprise JavaBean architecture. To access the event catalog, an event catalog application must first create an instance of the event catalog session bean.

Use the home interface to create an instance of the event catalog session bean.
```
//use home interface to create event catalog bean
InitialContext context = new InitialContext();
Object eventCatalogHomeObj =
     context.lookup("ejb/com/ibm/events/catalog/EventCatalog");
EventCatalogHome eventCatalogHome = (EventCatalogHome)
  PortableRemoteObject.narrow(eventCatalogHomeObj,
                              EventCatalogHome.class);
eventCatalog = (EventCatalog) eventCatalogHome.create();
```

Adding an event definition to the event catalog:

Newly created events can be added to the event catalog.

After you have created a new event definition and populated it with property descriptions and extended data element descriptions, you can add it to the event catalog. Once added to the event catalog, an event definition cannot be modified, but it can be replaced.

Note: If WebSphere security is enabled, the application user ID must be mapped to the catalogAdministrator role to add event definitions to the event catalog.

To add an event definition to the event catalog, use the addEventDefinition method.

```
boolean result = eventCatalog.addEventDefinition(definition, replace)
```

The parameters of this method are as follows:

definition
: The event definition you want to add. This must be a valid instance of EventDefinition.

replace
: A Boolean value indicating whether the specified event definition replaces an existing definition that has the same name.

If the *replace* parameter is `false`, the name of the specified event definition must not match that of any existing event definition in the catalog. If it does, an EventDefinitionExistsException exception is thrown.

If the *replace* parameter is `true`, the new event definition replaces any existing event definition with the same name that is already in the catalog. However, to preserve the inheritance hierarchy, the new event definition must name the same parent as the old event definition; otherwise, a ParentNotValidException exception is thrown.

The returned Boolean indicates whether an existing event definition was replaced. This is equal to true only if *replace* is equal to `true` and an event definition with the same name was replaced by the new definition.

When an event definition is added to the event catalog, the event catalog sends an event to the event server notifying event consumers that this change occurred. See *"Change notification" on page 201*.

Note: If you attempt to add an event definition that violates inheritance rules, an InheritanceNotValidException exception is thrown and the event definition is not added to the catalog. This can happen if a new event definition overrides inherited property or extended data element descriptions in ways that are not valid, or if replacing an existing event definition would cause descendants to override inherited descriptions in ways that are not valid. For more information, see *"Event catalog inheritance" on page 200*.

Removing an event definition from the catalog:

Event definitions no longer needed can be removed from the event catalog.

If an event definition is no longer needed, you can remove it from the event catalog.

Note: If WebSphere security is enabled, the application user ID must be mapped to the catalogAdministrator role to remove event definitions from the event catalog.

To remove an event definition from the event catalog, use the removeEventDefinition method.

```
eventCatalog.removeEventDefinition(name)
```

The *name* parameter is the name of the event definition you want to remove from the event catalog. If no matching event definition exists in the event catalog, an EventDefinitionNotFoundException exception is thrown.

When an event definition is removed from the event catalog, its children and all other descendants are also removed. For each event definition that is removed, the event catalog sends an event to the event server notifying event consumers that this change has taken place. See *"Change notification" on page 201* for more information.

Note: When an event definition is removed, the event catalog does not check the event server to determine whether any existing events in the event data store are described by that event definition. Therefore, you should make certain that an event definition is no longer needed before removing it from the event catalog.

Command reference

Command-line scripts are available to access some functions of the Common Event Infrastructure.

These scripts are implemented as Jacl scripts, which must be run using the WebSphere wsadmin tool (located in the `install_root/bin` directory). For more information about the wsadmin tool, see the WebSphere Application Server documentation.

Use the following syntax to run the scripts:

```
wsadmin -f scriptname.jacl
```

You can shorten parameter names, as long as you provide enough of the name to distinguish it from other parameters. For example, when you use the `eventquery.jacl` script, you can type `-ex` instead of `-extensionname`. However, `-e` is not valid, because it can represent either `-extensionname` or `-end`.

To get help with the syntax and usage for a command, type the command followed by the word `help`:

```
wsadmin -f scriptname.jacl help
```

If you are using the wsadmin tool with the Simple Objects Access Protocol (SOAP) protocol, a command might time out before the operation can complete. For example, this might happen if you query or purge a large number of events from the event server. If this happens, the wsadmin tool displays an error message indicating a failed SOAP RPC call:

```
Failed to make the SOAP RPC call: invoke
```

If you get this error message, try the command again, specifying `RMI` as the connection type and `2809` as the destination port. For example, the following command purges events from the event server using an RMI connection:

```
wsadmin -conntype rmi -port 2809 eventpurge.jacl -seconds 0
```

For more information about the `-conntype` parameter of the wsadmin tool, refer to the WebSphere Application Server documentation.

The eventcatalog.jacl script:

You can use a command line interface to access the event catalog.

Purpose

Lists event definitions or source categories in the event catalog and imports and exports event definitions.

wsadmin -f eventcatalog.jacl [**-listdefinitions** | **-listcategories** | **-exportdefinitions** | **-importdefinitions**] [**-file** *filename*] [**-name** *event_def_name*] [**-pattern**] [**-resolve**] [**-replace**]

Description

The `eventcatalog.jacl` script provides command-line access to the contents of the event catalog. It also provides support for importing and exporting event definitions.

Parameters

-listdefinitions
Lists the specified event definitions in a readable format, sorted by name in ascending order. The listing is written to the file specified by the **-file** parameter. If this parameter is not specified, the listing is written to the standard output.

-listcategories
Lists all of the defined event source categories and the event classes they contain, sorted by source category in ascending order. The listing is written to the file specified by the **-file** parameter. If this parameter is not specified, the listing is written to the standard output.

-exportdefinitions
Lists the specified event definitions in a format that is suitable for importing. The listing is written as an XML document conforming to the `eventdefinition5_0_1.xsd` XSD schema, which is packaged in the `events-client.jar` file. The listing is written to the file specified by the **-file** parameter. If this parameter is not specified, the listing is written to the standard output.

-importdefinitions
Reads a listing of event definitions from a file and adds the event definitions to the event catalog. The listing of event definitions to import must be written as an XML document that conforms to the `eventdefinition.xsd` XSD schema.

-file *filename*
For a list or export operation, the name of the file to which the output is written. For an import operation, the file that contains the event definitions to be imported. This parameter is required for import operations and optional for list and export operations. If this parameter is not specified for a list or export operation, the output is written to the standard output.

-name *event_def_name*
A name that identifies the event definitions to be listed or exported. If the **-pattern** parameter is not specified, the **-name** parameter identifies a single specific event definition. If **-pattern** is specified, **-name** specifies a pattern against which event definition names are compared. In this pattern, a percent character (%) matches any sequence of zero or more characters, and an underscore (_) matches any single character. All other characters are treated literally.

This parameter is valid only with the **-listdefinitions** and **-exportdefinitions** options. It is not valid with the **-listcategories** or **-importdefinitions** options.

-pattern
Specifies that the value specified with the **-name** parameter is to be treated as a pattern. This parameter is valid only with the **-listdefinitions** and **-exportdefinitions** options. It is not valid with the **-listcategories** or **-importdefinitions** options.

-resolve
Specifies that the event definitions to be listed or exported are resolved. A resolved event definition includes the property and extended data element descriptions that are inherited from its ancestors in the inheritance hierarchy. If this parameter is not specified, the event definition listing contains only the raw event definitions.

This parameter is valid only with the **-listdefinitions** and **-exportdefinitions** options. It is not valid with the **-listcategories** or **-importdefinitions** options.

-replace
Specifies that the event definitions to be imported replace existing event definitions with the same names. If this parameter is not specified, a name collision between an existing event definition and an imported event definition results in an error, and no event definitions are imported.

This parameter is valid only with the **-importdefinitions** option. It is not valid with the **-listdefinitions**, **-listcategories**, or **-exportdefinitions** options.

Examples

This example displays the contents of a single, resolved event definition named `insurance_claim_start` and writes the result to standard output:

```
wsadmin -f eventcatalog.jacl -listdefinitions -name insurance_claim_start -resolve
```

This example exports a set of event definitions, the names of which begin with the string `insurance_claim_start` and writes the result to an XML file:

```
wsadmin -f eventcatalog.jacl -exportdefinitions -file d:\myexport.xml
  -name insurance_claim_start% -pattern
```

This example imports a set of event definitions from the file `myimport.xml`and replaces existing definitions with the same names:

```
wsadmin -f eventcatalog.jacl -importdefinitions -file d:\myimport.xml -replace
```

This example displays a listing of all defined event source categories and the events they contain. The result is written to standard output:

```
wsadmin -f eventcatalog.jacl -listcategories
```

The emitevent.jacl script:

You can use a command line interface to submit events to the event server.

Purpose

Sends an event to the event server.

wsadmin -f emitevent.jacl [**-xml** *url*] [**-msg** *message*] [**-severity** *severity*] [**-extensionname** *extension_name*] [**-emitter** *profile_name*] [**-synchronous** | **-asynchronous**]

Description

The `emitevent.jacl` script provides a command-line interface for submitting events to the event server. You can provide the event content by providing a source XML file or by specifying property values on the command line.

Events generated by this script have the following default content:

```
<CommonBaseEvent creationTime=current_system_time
version="1.0.1"> <sourceComponentId component="emitevent.jacl"
componentIdType="Application" location=local_hostname
locationType="Hostname"
subComponent="com.ibm.events.cli.util.EmitEventCliHelper"
componentType="http://www.ibm.com/namespaces/autonomic/Tivoli
/EventInfrastructure"/> <situation categoryName="ReportSituation">
  <situationType xsi:type="ReportSituation" reasoningScope="EXTERNAL"
   reportCategory="CLI"/>
 </situation>
</CommonBaseEvent>
```

The current_system_time parameter is the system time at which the event is generated, specified as an XML dateTime string.

Parameters

-xml *url*
: A uniform resource locator (URL) that specifies the location of an XML document that contains the event to be submitted. This XML document must conform to the Common Base Event version 1.0.1 XSD schema. If no URL scheme (such as `http://`) is specified, a local file is assumed. This parameter is optional.

  Two sample XML files, `eventsample1.xml` and `eventsample2.xml`, are available in the *`install_root`*`/events/samples` directory.

-msg *message*
: The value to use for the message property of the event. If the message contains spaces, enclose this value in quotation marks. This parameter is optional. If you specify this parameter in addition to an XML file, the value of the **-msg** parameter overrides any value specified in the XML file for the msg property.

-severity *severity*
: The value to use for the severity property of the event. This parameter is optional. If you specify this parameter in addition to an XML file, the value of the **-severity** parameter overrides any value specified in the XML file for the severity property.

-extensionname *extension_name*
: The value to use for the extensionName property of the event. If the extension name contains spaces, enclose this value in quotation marks. This parameter is optional. If you specify this parameter in addition to an XML file, the value of the **-extensionname** parameter overrides any value specified in the XML file for the extensionName property.

-emitter *profile_name*
The Java Naming and Directory Interface (JNDI) name of the emitter factory profile to use when obtaining an emitter. This parameter is optional. If this parameter is not specified, the default emitter factory profile (`/com/ibm/events/configuration/emitter/Default`) is used.

-synchronous | -asynchronous
The synchronization mode to use for event transmission. This parameter is optional. If it is not specified, the preferred synchronization mode configured for the emitter is used.

Examples

The following example sends an event to the event server with a severity of 30 and the extension name `test_event` (all other properties have the default values):

```
wsadmin -f emitevent.jacl -severity 30 -extensionname test_event
```

The following example sends an event using the properties specified in `eventsample1.xml`:

```
wsadmin -f emitevent.jacl -xml ../samples/eventsample1.xml
```

The eventquery.jacl script:

You can use a command line interface to query the event database.

Purpose

Generates a report listing events in the event database.

wsadmin -f eventquery.jacl [**-globalinstanceid** *global_instance_id* | **-group** *event_group*] [**-severity** *severity*] [**-extensionname** *extension_name*] [**-start** *start_time*] [**-end** *end_time*] [**-number** *number*] [**-ascending**|**-descending**]

Description

The `eventquery.jacl` script queries the event database and generates a report that lists the result. You can query events based on the event group, the severity, or the extension name. You can also query events that were created during a specified period of time.

Parameters

-globalinstanceid *global_instance_id*
The global instance identifier of the event to query. Either this parameter or **-group** (but not both) is required.

-group *event_group*
The event group from which to query events. The *event_group* value must be the name of an event group defined in the Common Event Infrastructure configuration. Either this parameter or **-globalinstanceid** (but not both) is required.

-severity *severity*
The severity of the events that you want to include in the report. The *severity* value must be an integer. Only events with a severity equal to the value that you specify are included in the report. This parameter is optional.

-extensionname *extension_name*
The extension name of events that you want include in the report. Use this parameter to restrict the query to events of a specific type. Only events with the extensionName property equal to *extensionName* are included in the report. This parameter is optional.

-start *start_time*
The earliest time of the events that you want to include in the report. Use this parameter to restrict the query to events that were generated after a specified date and time. This parameter must be a date and time that is specified according to the XML dateTime data type. The basic format is `CCYY-MM-DDThh:mm:ss`, optionally followed by a time zone indicator. For example, noon on 1 January 2004 in Eastern Standard Time is `2004-01-01T12:00:00-05:00`. For more information about the dateTime data type, refer to the *XML schema at www.w3.org*.

-end *end_time*
The latest time of the events that you want to include in the report. Use this parameter to restrict the query to events that were generated before a specified date and time. This parameter must be a date and time that is specified according to the XML dateTime data type. For more information, see the description of the **-start** parameter.

-number *number*
The maximum number of events that you want to include in the report. This parameter must be an integer. If the number of matching events in the database exceeds the specified value, the report is truncated. If the report is sorted in ascending order, this means that the most recent matching events are omitted. If the report is sorted in descending order, the oldest matching events are omitted.

-ascending|-descending
The chronological order in which the events in the report are sorted. This must be one of the following values:

ascending
Ascending (chronological) order, with the oldest events first. This is the default value.

descending
Descending (reverse chronological) order, with the most recent events first.

Example

The following example lists all of the events from the database that belong to the **All events** event group and were generated on 17 February 2004. The report is sorted in reverse chronological order:

```
eventquery.jacl -group "All events" -start "2004-02-17T00:00:00-05:00"
  -end "2004-02-17T23:59:59-05:00" -order DESC
```

The eventbucket.jacl script:

You can use a command line interface to change the event database bucket configuration.

Purpose

Displays or changes the event database bucket configuration.

wsadmin -f eventbucket.jacl [**-status**] [**-change**]

Description

The **eventbucket.jacl** script displays or changes the event database bucket configuration. Buckets are used by the rapid purge utility to purge old event data from the event database. By running this command, you can determine the current bucket configuration, or you can swap the active and inactive buckets.

Note: If WebSphere security is enabled, your user ID must be mapped to the eventAdministrator role to view or change the event database bucket configuration.

Parameters

-status
Displays information about the current bucket configuration, including the active bucket setting and the bucket check interval (the frequency with which the data store plug-in checks to determine which bucket is active).

-change
Swaps the buckets so the active bucket becomes inactive and the inactive bucket becomes active. The inactive bucket must be empty before you can use this option.

Examples

This example displays the current bucket configuration:

```
wsadmin -f eventbucket.jacl -status
```

This example swaps the active and inactive buckets:

```
wsadmin -f eventbucket.jacl -change
```

The eventpurge.jacl script:

You can use a command line interface to purge events from the event database.

Purpose

Purges events from the event database.

wsadmin -f eventpurge.jacl [**-seconds** *seconds* | **-end** *end_time*] [**-group** *event_group*] [**-severity** *severity*] [**-extensionname** *extension_name*] [**-start** *start_time*] [**-size** *size*]

Description

The **eventpurge.jacl** script purges events from the event database. You can purge all events from the event database, or you can limit the purge to events meeting certain criteria.

Note: If WebSphere security is enabled, your user ID must be mapped to the eventAdministrator role to delete events.

Parameters

-seconds *seconds*
: The minimum age of events you want purged. The *seconds* value must be an integer. Only events older than the specified number of seconds are purged. This parameter is required if you do not specify the **-end** parameter.

-end *end_time*
: The end time of the group of events you want to delete. Only events generated before the specified time are deleted. The *end_time* value must be specified in the XML dateTime format (`CCYY-MM-DDThh:mm:ss`). For example, noon on 1 January 2006 in Eastern Standard Time would be `2006-01-01T12:00:00-05:00`. For more information about the dateTime data type, refer to the XML schema at www.w3.org.

  This parameter is required if you do not specify the **-seconds** parameter.

-group *eventGroup*
: The event group from which to purge events. The *event_group* value must be the name of an event group defined in the Common Event Infrastructure configuration. This parameter is optional.

-severity *severity*
: The severity of events you want purged. The *severity* value must be an integer; only events whose severity is equal to the value you specify are purged. This parameter is optional.

-extensionname *extension_name*
: The extension name of events you want included in the purge. Use this parameter to restrict the purge to events of a specific type. Only events whose extensionName property is equal to *extensionName* are purged. This parameter is optional.

-start *start_time*
: The beginning time of the group of events you want to delete. Only events generated after the specified time are deleted. The *start_time* value must be specified in the XML dateTime format (`CCYY-MM-DDThh:mm:ss`). This parameter is optional.

-size *size*
: The number of events to purge in a single transaction. The *size* value must be an integer. After this number of events have been purged, the command commits the transaction before continuing in a new transaction. This parameter is optional.

Example

The following example purges all events from the database whose severity is 20 (harmless) and were generated earlier than 10 minutes ago.

```
eventpurge.jacl -group "All events" -severity 20 -seconds 600
```

Viewing events with the Common Base Event browser

Use the Common Base Event browser to select, sort, and view events.

This task assumes you are logged into the WebSphere Process Server administrative console.

The event browser uses the event access interface to query event data. The results of the query are shown in the browser.

1. Begin by opening the event browser. Select **Integration Applications** and then **Common Base Event Browser** in the navigation pane of the administrative console.
2. Specify the events you want to view.
3. Select the view of the returned events.
4. In any of the browser panels, when you have finished selecting search or sort criteria, click the **Get Events** button at the bottom of the browser panel to display the desired events.

Specifying the events to view

How to use the Common Base Event browser to specify search criteria for querying events in the event database.

This task assumes that you have already opened the event browser and are viewing the Get Events panel.

The Event Data Store Properties fields require completion. The Event Filter Properties fields are optional, and allow you to narrow your events search based on time, date, server name, sub-component name, and event severity parameters.

1. **Required:** Specify the Event Data Store to search.

   The field is a Java Naming and Directory Interface (JNDI) name, an Enterprise JavaBeans (EJB) reference that can be configured in the administrative console. The WebSphere Process Server default is `java:comp/env/eventsaccess`.
2. **Required:** Specify the Event Group to search.

   This is the event group from which events are retrieved. The default group is `All events`.
3. **Required:** Specify the number of events to retrieve.

   The maximum number of events to search is `500`.
4. **Optional:** Specify the Creation Date (calendar period) for the report.

   Enter the start and end dates.
5. **Optional:** Specify the Creation Time (time period) for the report.

   Enter the start and end times.
6. **Optional:** Specify the server name.
7. **Optional:** Specify the sub-component name, if applicable.
8. **Optional:** Specify an event's priority. The range of events priorities to retrieve is from 0 (lowest priority) to 100 (highest priority).
9. **Optional:** Specify an event's severity.

   The range of events severities to retrieve is from 0 (least severe) to 70 (most severe).
10. Click **Get Events**.

The number of Common Base Events matching the search criteria is displayed. If the results you queried do not appear, see *"Troubleshooting the Common Base Event browser" on page 216*.

To view the returned events, select a view from the navigation bar. You can choose **All Events**, **BPEL Process Events**, **User Data Events**, or **Server Events**. When you view event data, you can change your search criteria at any time by clicking **Get Events**.

Once events are returned, you can work with them to get various levels of event detail.

Working with events returned from the event browser

You use the event browser to view the events returned from a query.

This task acts on data that is returned by a submitted query, as described in *Specifying the events to view*.

The query returns all the events that meet your criteria.

1. Select a view from the navigation bar.

   The navigation bar offers the following views of the returned query:

   All Events
   All the events returned.

   BPEL Process Events
   Business Process Choreographer events for a specific process instance.

   User Data Events
   Events with the extension name `ECS:UserDataEvent`. This event type is created by the `addUserDataEvent` method of the ECSEmitter class.

   Server Events
   Events for a specific server.
2. Perform one of the following actions.
   - If you select **BPEL Process Events** in step 1, you must select a process template, and then a process instance.
   - If you select **Server Events** in step 1, you must select a server.
3. Click an event, to display the event data in the pane at the bottom of the browser window.

Troubleshooting the Common Base Event browser

There are four primary conditions under which you will be unable to access the Common Base Event browser.

Conditions

″Cannot find server″
WebSphere Process Server (or network server) is unavailable. When you attempt to launch the event browser URI, a ″Cannot find server″ browser page will be returned, which indicates that the server is unavailable. In this case, you need to contact the IBM Help Desk to determine the cause of the problem.

″File not found″
WebSphere Process Server is available; however, the event browser application may not be installed or started on the server. When you attempt to launch the event browser URI, a ″File not found″ browser page will be returned, which indicates that the server is available, but the URI is

not available on that server. In this case, you need to contact the IBM Help Desk to determine the cause of the problem.

Logon panel appears
The WebSphere Process Server and the event browser are available; however, you have not been mapped to the proper role to allow access to the event browser. You will be prompted with a logon panel. When you enter your userID and password, attempting to log in, the login will fail. In this case, you need to contact the IBM Help Desk to get the proper authorization to launch the event browser.

Error message on ″Get event data″ panel
The WebSphere Process Server and the event browser are available, and you have the proper authority to gain access; however, the Common Event Infrastructure server is unavailable. An error message will be displayed on the event browser **Get Events** panel, when you click the **Get Events** button. The error information will be logged to the message log.

Notices

Programming interface information

Programming interface information, if provided, is intended to help you create application software using this program.

General-use programming interfaces allow you to write application software that obtain the services of this program's tools.

However, this information may also contain diagnosis, modification, and tuning information. Diagnosis, modification and tuning information is provided to help you debug your application software.

Warning: Do not use this diagnosis, modification, and tuning information as a programming interface because it is subject to change.

Trademarks and service marks

Printed in USA