



Middleware Configuration for WebSphere Application Server plug-in

Version 5.0 This information is under development and can change at any time; the released version will be available on ibm.com.

Note

Before using this information and the product it supports, read the information in "Notices," on page 17.

This edition applies to version 4.0 of Middleware Configuration for Websphere Application Server plug-in for IBM UrbanCode Deploy and to all subsequent releases and modifications until otherwise indicated in new editions.

© Copyright IBM Corporation 2011, 2014.

US Government Users Restricted Rights – Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

Contents

Middleware Configuration for WebSphere plug-in	1
Deploying WebSphere configurations and applications.	2
1. Set up plug-ins in UrbanCode Deploy	3
2. Create a component for the configuration template.	3
3. Model a WebSphere cell in Resources	4
4. Map the desired resource to the configuration template component.	5
5. Associate the component with an application.	5
6. Associate other needed components with the application	5
7. Associate the resources with an application environment	6
8. Generate a configuration template	6
9. Deploy the configuration template	7
Specifying multiple profiles with the websphere.profilePath property	7

Using custom properties when deploying WebSphere configurations and applications	8
Using XPath to specify property values	9
Comparing WebSphere configurations	14
Creating an application process to apply a configuration to a cell	14
Creating an application process to run the comparison	15
Process steps in the IBM Middleware Configuration for WebSphere plug-in.	15
Compare configuration	15
Compare server configuration	15
Generate Template	15
Generate configuration	15
Manage configuration	16
Manage server configuration	16
Appendix. Notices	17
Trademarks	19

Middleware Configuration for WebSphere plug-in

The Middleware Configuration for WebSphere® plug-in includes steps that deploy configuration templates of IBM® WebSphere Application Server that you generate from an existing cell.

This plug-in allows you to automate the process of deploying WebSphere configurations and applications to different environments (for example, development, test, and production environments). The following steps provide an overview of the process you follow.

1. Set up UrbanCode Deploy to support the process. Load plug-ins and install an agent on the host where the WebSphere installation resides.
2. Create a component using the configuration template.
3. Capture topology information from an existing WebSphere installation and model it in Resources. A resource tree represents scopes in the installation.
4. Map the resources (scopes) to manage to the configuration template component.
5. Associate the component with an application.
6. Associate other components needed for deployment, for example components containing EAR or WAR files (application binaries)
7. Associate the resources with an application environment.
8. Generate the configuration template
9. Deploy the configuration template to create new instances.

The process for using the plug-in is designed to support one application per scope (cell, cluster, server, or node). The process is not designed to support use cases where multiple applications are deployed within the same cell scope or the same cluster scope.

You can use custom properties on the component to differentiate resources across environments. The custom properties are used to generate *tokens* when you generate the templates. Tokens are used like variables to substitute values in configuration information that varies depending on environment. For example, you may use a different database or a different JDBC resource in each of the deployment environments. See “Using custom properties when deploying WebSphere configurations and applications” on page 8

Updates for Version 5 of the plug-in

Various defects were fixed for Version 5 of the plug-in, which is supported in UrbanCode Deploy 6.0.1.

Updates for Version 4 of the plug-in

The following features were added for Version 4 of the plug-in, which is supported in UrbanCode Deploy 6.0.1.

- Support for multiple profiles on the same host. Support was added for multiple profiles for top-level resource group properties and multiple profiles for agent properties.

- Use of XPath to create tokens. The previous method of using name-value pairs in a promote.properties file is still supported.
- Updated method for importing information from a configuration: The method is called **QuickCapture** and is enabled by default. It can be disabled for debugging purposes. An example usage would be that generating the configuration template does not work after upgrading to this version of the Middleware Configuration for WebSphere plug-in
- Base template specification: If the **SkipConfigCapture** flag is enabled when you import the a template, an existing WebSphere Application Server configuration is used as a base template. Normally this flag is enabled in order to tokenize an existing template.

Updates for Version 3 of the plug-in

The updates for Version 3 were internal and required documentation changes.

Updates for Version 2 of the plug-in

The following features were added for Version 2 of the plug-in, which is supported in UrbanCode Deploy 6.0.1.

- Requirements: version 2 is supported with version 6.0.1 of UrbanCode Deploy. It is not supported in prior versions of UrbanCode Deploy.
- Additional scopes: Support for the node and server scopes was added to the existing support for cell and cluster scopes.
- Automated generation of configuration templates: The automated process requires less connection setup and fewer steps than the previous manual, script-based process in version 1.0.
- Compare configurations: You can now compare a component (configuration template) to a live cell. This is useful for detecting when a live cell's configuration "drifts," or departs from the desired configuration. See "Comparing WebSphere configurations" on page 14.

Deploying WebSphere configurations and applications

To deploy WebSphere configurations and applications, you use plug-ins to read information about a live deployment into a model in Resources. You then define components, applications, environments, and processes to automate deployments that create new instantiations of the modeled WebSphere environment.

Before you begin

The following setup and preparation is required.

- Select or set up a WebSphere cell to use. The configuration template you create will use its configuration.
- Install an UrbanCode Deploy agent on the appropriate WebSphere host. Note that you need the agent name during the procedures.
 - For ND deployments, install the agent where DMGR is running.
 - For server deployments, install the agent where Base is running.
- Determine what tokens you need. Tokens are used to represent information that is different among deployment environments. For example, you might specify a different database server or JDBC resource in each of a development, test, and production environment. For each token, you need a name and an initial value.

You are prompted to enter the names and values at the end of the process. See “Using custom properties when deploying WebSphere configurations and applications” on page 8

- If WebSphere Application Server is deployed in a directory other than the default directory, you must set a property on the agent before you can run a process to deploy a WebSphere cell.
 1. Go to **Resources > Agents > <agent-name> > Configuration > Agent Properties**.
 2. Add a property named **wsadmin.path**.
 3. Set **wsadmin.path** to the fully qualified path to the wsadmin script (including the script name).

1. Set up plug-ins in UrbanCode Deploy

About this task

In this task you prepare the required UrbanCode Deploy resources.

Procedure

1. Install the Middleware Configuration for Websphere plug-in.
 - a. In UrbanCode Deploy, go to **Settings > Automation Plugins**.
 - b. Click **Load Plugin**.
 - c. In the dialog, click **Choose File**, then choose the MCWASPlugin-<version>.zip
 - d. Click **Submit**.
2. Install the Application Deployment for WebSphere plug-in.
 - a. In UrbanCode Deploy, go to **Settings > Automation Plugins**.
 - b. Click **Load Plugin**.
 - c. In the dialog, click **Choose File**, then choose the ApplicationDeploymentForWebSphere-<version>.zip file.
 - d. Click **Submit**.

During installation a component template is automatically created for Middleware Configuration for WebSphere. It is placed in the following file: <plugin_home>/imports/componenttemplates/Middleware Configuration for WebSphere.json.

2. Create a component for the configuration template

About this task

The component is at the center of the deployment process. It is associated with a configuration template and with resources so that it holds a configuration definition to be deployed.

Procedure

Create a component if one does not exist.

1. Click the **Components** tab.
2. Click **Create New Component**. Provide the following information:
 - **Name**: Enter a name for the component.
 - **Template**: select **Middleware Configuration for WebSphere**.
3. Click **Save**.

3. Model a WebSphere cell in Resources

About this task

Using autodiscovery, you locate a WebSphere cell through the agent that is installed on the host where it is running. Using auto-configure, you populate the resource tree with data about the WebSphere cell. For non-ND environments, you model a server resource. The resource tree is the centralized source of configuration data. Multiple processes in UrbanCode Deploy can include components that contain resources selected from the resource tree.

Procedure

1. In UrbanCode Deploy, go to **Resources**.
2. Click **Create Top-Level Group** to create a group.
3. Optional: if you need to specify multiple profiles for the group, add `websphere.profilePath` as a property on the group and specify paths to the profiles in its value. See “Specifying multiple profiles with the `websphere.profilePath` property” on page 7 for details.
4. Add one or more agents, as required.
 - a. Hover over the row for the resource group, click **Actions**, and select **Add agent**.
 - b. Select the agent to add. Use the agent that is installed on the host for the WebSphere environment you are going to model.
 - c. Wait 10 to 30 seconds, then click **Refresh**. A twisty is now next to the agent. When you expand it, there is a sub-resource cell, **WebSphereCell**.
 - d. Hover over the row, then click **Edit**.
 - e. Enter values for the following properties.

If you want to use `soap.properties` to provide the values, you must leave the WebSphere User and WebSphere Password properties blank.

 - WebSphere Profile Path
 - WebSphere User
 - WebSphere Password

Leave the **Cell Name** property blank.
 - f. Optional: if you need the agent to manage multiple profiles, add `websphere.profilePath` as a property on the agent and specify paths to the profiles in its value. See “Specifying multiple profiles with the `websphere.profilePath` property” on page 7 for details.
 - g. Click **Save**.
5. Set Auto configure options for **WebSphereCell**.
 - a. Hover over the row for **WebSphereCell**, click **Actions**, then click **Auto Configure**.
 - b. Click **No auto configure for resource**.
 - c. Check **Websphere Topology Discovery** box.
 - d. Click **OK**.
 - e. Click **Save**.
 - f. Wait 30-60 seconds, then click **Refresh**. A twisty is now next to **WebSphereCell**. Expand it and make sure the resource tree matches your WebSphere Application Server topology.
 - g. On the **WebSphereCell** entry, click **Edit**. Check that **Cell Name** was filled in and is correct.

4. Map the desired resource to the configuration template component

About this task

During mapping you choose a scope from the resource tree to be used for deployments. Note that a captured scope configuration can be applied only to a like scope. For example, if you capture configuration information for a cluster scope, it can only be used to deploy a cluster configuration.

Procedure

1. Click **Resources** to view the resource tree for the desired WebSphere cell.
2. Add a component to the scope or scopes you want to use: cell, cluster, node, or server.
 - a. Select the scope you want to manage. Hover over a row to select a single scope to show a menu, then click **Add component** in the menu.
 - b. Select the component to use. You created it above.
 - c. Click **Save**.

5. Associate the component with an application

About this task

Create the application if it does not exist. The application is the container for deployment data.

Procedure

1. In the **Dashboard**, click **Applications**.
2. Create the application if it does not exist.
 - a. Click **Create New Application**.
 - b. Enter an application name, then click **Save**.
3. Add the component to the application.
 - a. In the **Dashboard**, click **Applications**.
 - b. Click the application to use.
 - c. Click the **Components** tab, then add the component that you created for the configuration template.

6. Associate other needed components with the application

About this task

To deploy applications, you need to have components configured for the WAR or EAR files. Add them to the application.

Procedure

1. In the **Dashboard**, click **Applications**.
2. Click the application to use.
3. Click the **Components** tab, then add the desired component.
4. Repeat for all necessary components.

7. Associate the resources with an application environment

About this task

The resources define the configuration data to be deployed with the application.

Procedure

1. In the **Dashboard**, click the **Applications** tab.
2. Click the application you created.
3. Create an application environment.
 - a. Click **Create New Environment**.
 - b. Enter an environment name, then click **Save**.
4. Add the resource group to the environment.
 - a. In the application, click **Environments**.
 - b. Click the environment name.
 - c. Click **Add Base Resource**. Choose the group for your exemplar cell and environment.
5. Optional: view the environment properties for the component.
 - a. In the **Dashboard**, click **Components**.
 - b. Click the component.
 - c. Click **Configuration**.
 - d. Click **Environment Property Definitions**.

8. Generate a configuration template

About this task

Run an application process to generate the configuration template.

Procedure

1. In the application you created, click the **Processes** tab.
2. Define the process as follows.
 - a. In the process design page, locate each component that you have added to the application.
 - b. Click the component to show a **Generate Template** process step.
 - c. Drag the **Generate Template** step to the process editor, then give the process a name.
 - d. Click **Save**.
3. Run the application process to generate the template.
4. Enter information for the following fields.
 - **New Component Version**: Specify a new version to use. The configuration template files are associated with that version.
 - **SkipConfigCapture**: If this flag is enabled, an existing WebSphere Application Server configuration is used as a base template. Normally this flag is enabled in order to tokenize an existing template.
 - **UseQuickCapture**: Enabled by default. Leave it enabled unless there are problems generating the template. You may uncheck it to debug problems with generating the template.
5. Click **Submit**.

Wait for the process to finish running before continuing.

9. Deploy the configuration template

About this task

Create an application process of type **Deployment** for configuration and deployment, then run the process to test it.

Procedure

1. In the **Dashboard**, click the **Applications** tab.
2. Click the application you created.
3. Click the **Processes** tab.
4. Define the process.
 - a. Drag and drop the **Install Component** step onto the process editor. Choose the **Configure WebSphere Application Server** component process.
 - b. Select the component you created
 - c. Choose the **Configure WebSphere Application Server (Template)** component process
5. Click **Save**.
6. Run the process to test the newly created component template.

Specifying multiple profiles with the `websphere.profilePath` property

Specifying multiple profiles enables you collect information for multiple cells, either for the top-level group or for the agent.

You specify multiple profiles during deployment. You specify them in a `websphere.profilePath` property in one of two places:

- *Top-level group*: Set the `websphere.profilePath` here to collect profiles from multiple cells deployed across multiple hosts. All hosts must use the same installation directory for the cell.
- *Agent*: Set the `websphere.profilePath` here to collect profiles for multiple profiles on the same host.

Values for the `websphere.profilePath` property

Specify one or more paths as the property value. Separate multiple paths with a comma. The paths can be one of the following:

- path to a profile. The path does not have to specify `/bin/wsadmin.sh`, but autodiscovery works with paths that do.
- path to a container directory of profiles. In this case the autodiscovery code loops over each first-level directory. It registers base and nd profiles. It skips node and server profiles.

When you specify multiple profiles, the following values are read during autodiscovery:

- SOAP port (read from `portdef.properties`)
- profile path
- install path

Example value with two container directories and one profile path:

```
"/opt/IBM/WebSphere/Profiles/,/opt/WAS/Profiles,/opt/IBM/profiles/dmgr"
```

Restriction: If you specify multiple container directories, there should be no duplicates in the profile names they contain. If autodiscover finds the same profile name, it overwrites the previously found profile name.

Example: /opt/IBM/WebSphere/Profiles and /opt/WAS/Profiles both contain a dmgr profile. A resource is created only for the second dmgr resource, because the first profile is overwritten.

If you encounter this situation, you can work around it by creating a separate top-level group and segregate the profile directories.

Using custom properties when deploying WebSphere configurations and applications

Use custom properties on a component to differentiate configurations between environments.

Before you begin

- Understand the process for deploying WebSphere configurations and applications. See “Deploying WebSphere configurations and applications” on page 2.

Set up and deploy WebSphere configurations and applications without custom properties. Examine the deployment to determine where you need to differentiate deployment environments.

About this task

Typically multiple environments are used in a staged software development process. The environments are segregated, for example into Development, Test, and Production areas. Code is promoted from one area to the next after exit criteria are met.

Configuration data for a deployed WebSphere instance may need to vary according to the environment where it is deployed. For example, a different database may be used for backing a JDBC resource in each environment. Defining custom properties allows you to customize deployment into each environment.

You define custom properties on the component you create for the deployment. When you generate the template, the custom properties are scanned and tokens are created in the configuration template. The tokens are substituted with values from the environment during deployment.

Property values can be either an explicit value or an XPath expression. You typically use XPath if the value could be confused with another value or part of a value if you used a simple value. For example, specifying 80 (HTTP port) could be confused with another value (9080) and result in an incorrect token. Note that only some XPath functionality is available for this purpose.

Procedure

1. Click **Components**.
2. Click on the component, then the **Configuration** tab, then **Environment Property Definitions**.

- are not unique within the data file, for example, the values true and false might be assigned to several variables
- cannot be replaced everywhere in the data file with the same value

For information about XPath expressions, see the XML Path Language (XPath) Version 1.0 produced by the World Wide Web Consortium (WC3).

XPath expression syntax

The syntax of a generic XPath expression is as follows:

```
value_name.value
value_name.xpath=/XmlNode/Child[@attrName="value"]/XmlNode/@matchAttrName
```

The table provides descriptions of the XPath expression elements and are provided as a guide to create a basic XPath expression.

See the WC3 documentation on XPath language: World Wide Web Consortium (WC3).

XPath expression element	Description
<code>value_name.value</code>	Assigns a <code>value_name</code> to the <code>value</code> that you want to locate in a data file.
<code>value_name.xpath</code>	Assigns the same <code>value_name</code> to the XPath expression.
<code>.xpath</code>	Query to locate that attribute whose <code>value</code> is to be replaced by <code>value_name</code>
<code>/XmlNode/</code>	Selects an element node named <code>XmlNode</code> in the XML data file.
<code>/XmlNode/Child</code>	Selects an element node named <code>Child</code> with a parent named <code>XmlNode</code> .
<code>/XmlNode[@attrName="value"]</code>	Selects an element node of type <code>XmlNode</code> with an attribute of <code>attrName</code> that has a specific <code>value</code> .

Rules for XPath expressions used in promote.properties files

- Do not select an element node with your XPath query.
- The XPath expression must point to an attribute that you want to update, not to an element node in the XML file.
- XPath queries should always end with `/@attrName` to target an attribute and not the node element.
- When you configure datasources, do not use absolute paths. Specify paths that are relative to container nodes.

Example of XPath expressions in promote.properties files

The following is an example of an XPath expression that locates and updates a value in the `jdbc.xml` data file.

The example XPath expression looks for a `DataSource` with the name `widgetDB` and updates the `J2EEResourceProperty` attribute to `50000` where the name attribute is `portNumber`.

```
dbport.value=50000
dbport.xpath=//DataSource[@name='widgetDB']/J2EEResourcePropertySet/J2EEResourceProperty
[@name='portNumber']/@value
```

The XML data file in the following example shows an excerpt from the `jdbc.xml` file with the updated port number value.

```

<jdbc>
<RAFW_JDBCDataSources>
<DataSource
  name="widgetDB"
  >
<J2EEResourcePropertySet
  WASKey="propertySet"
  >
  <J2EEResourceProperty
    WASKey="resourceProperties"
    name="portNumber"
    required="false"
    type="java.lang.Integer"
    value="50000"
  >
</J2EEResourceProperty>
</J2EEResourcePropertySet>
</DataSource>
</RAFW_JDBCDataSources>
</jdbc>

```

Example of relative paths to datasources in XPath expressions

When you run the action `was_common_configure_jdbc_datasources` in promote mode, the starting point for the XPath expression must be the relative path to the container node. You must therefore omit the absolute path declarations `/jdbc/RAFW_JDBCDataSources` from the expression and start with `/DataSource`. For example,

```

<DataSource
  name="widgetDB"
  >
<J2EEResourcePropertySet
  WASKey="propertySet"
  >
<J2EEResourceProperty
  WASKey="resourceProperties"
  name="portNumber"
  required="false"
  type="java.lang.Integer"
  value="50000"
  >
</J2EEResourceProperty>
</J2EEResourcePropertySet>
</DataSource>

```

Example of using an XPath expression

The following example is set in an IBM UrbanCode Deploy environment.

The example uses a WebSphere Application Server configuration file, `cache.xml`.

1. Original `cache.xml` file:

```

<?xml version="1.0" encoding="UTF-8" ?>
- <CacheProviders xmlns="http://raf.rational.ibm.com/xml/ns/websphere/cache" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://raf.rational.ibm.com/xml/ns/websphere/cache http://raf.rational.ibm.com/xml/ns/websphere/cache" >
- <RAFW_cacheInstance >
- <ObjectCacheInstance cacheSize="2000" defaultPriority="1" disableDependencyId="false" diskCacheCleanupFrequency="0" diskCacheSizeInEntries="0" diskCacheSizeInMB="0" enableCacheReplication="false" enableDiskOffload="false" flushToDiskOnStop="false" replicationType="NONE" useListenerContext="false" >
  <CacheProvider RAFW_TYPE="reference" WASKey="provider" name="CacheProvider" />
  <DiskCacheCustomPerformanceSettings WASKey="diskCacheCustomPerformanceSettings" maxBufferedCacheIdsPerMetaEntry="1024" jndiName="jndi2" memCacheEvictionPolicy WASKey="diskCacheEvictionPolicy" algorithm="NONE" highThreshold="80" lowThreshold="70" />
</ObjectCacheInstance >
- <ServletCacheInstance cacheSize="4000" defaultPriority="1" diskCacheCleanupFrequency="0" diskCacheEntrySizeInMB="0" diskCacheSizeInEntries="0" enableCacheReplication="false" enableDiskOffload="false" flushToDiskOnStop="false" hashSize="1024" jndiName="jndi2" memCacheEvictionPolicy WASKey="diskCacheEvictionPolicy" algorithm="NONE" highThreshold="80" lowThreshold="70" />
  <CacheProvider RAFW_TYPE="reference" WASKey="provider" name="CacheProvider" />
  <DiskCacheCustomPerformanceSettings WASKey="diskCacheCustomPerformanceSettings" maxBufferedCacheIdsPerMetaEntry="1024" jndiName="jndi2" memCacheEvictionPolicy WASKey="diskCacheEvictionPolicy" algorithm="NONE" highThreshold="80" lowThreshold="70" />
</ServletCacheInstance >
</RAFW_cacheInstance >
</CacheProviders >

```

2. Cache resources as seen in the WebSphere Application Server console:

Object cache instances

An object cache instance is a location, in addition to the default shared dynamic cache, where Java(TM) Platform, Enterprise Edition (Java EE) applications can store, distribute, and share data. This gives applications greater flexibility and better tuning of the cache resources. Use the DistributedObjectCache programming interface to access this cache instance. See the DistributedObjectCache API documentation in the WebSphere(R) Application Server API documentation for more information.

Scope: Cell=cell, Cluster=cluster1

Show scope selection drop-down list with the all scopes option

Scope specifies the level at which the resource definition is visible. For detailed information on what scope is and how it works, [see the scope settings help](#).

Cluster=cluster1

Preferences

New... Delete

Select	Name	JNDI name	Scope	Cache size
<input type="checkbox"/>	objCache	jndi1	Cluster=cluster1	2000

total 1

Servlet cache instances

A servlet cache instance is a location, in addition to the default shared dynamic cache, where dynamic cache can store, distribute, and share data. This gives applications greater flexibility and better tuning of the cache resources. The Java(TM) Naming and Directory Interface (JNDI) name that is specified for the cache instance is mapped to name attribute in the cachespec.xml configuration file.

Scope: Cell=cell, Cluster=cluster1

Show scope selection drop-down list with the all scopes option

Scope specifies the level at which the resource definition is visible. For detailed information on what scope is and how it works, [see the scope settings help](#).

Cell

cell

Node Browse Nodes

Server Browse Servers

Cluster cluster1 Browse Clusters

Apply

Preferences

New... Delete

Select	Name	JNDI name	Scope	Cache size
<input type="checkbox"/>	servletCache	jndi2	Cluster=cluster1	4000

3. In IBM UrbanCode Deploy, the resources are expressed as environment property definitions on the component.

Environment Property Definitions

Define properties here to be given values on each environment the component is used in.

Add Property

Version 5 of 5

< >

Name	Label	Pattern	Required	Default Value
websphere.objCacheSize.xpath	websphere.objCacheSize.xpath		false	//ObjectCacheInstance[@name='objCache']/@cacheSize
websphere.objCacheSize.value	websphere.objCacheSize.value		false	2000
websphere.servletDiskCacheEvictionPolicyLT.xpath	websphere.servletDiskCacheEvictionPolicyLT.xpath		false	//ServletCacheInstance[@name='servletCache']/@DiskCacheEvictionPolicy/@lowThreshold
websphere.servletDiskCacheEvictionPolicyLT.value	websphere.servletDiskCacheEvictionPolicyLT.value		false	70

4 records - Refresh Print

<< 1 / 1 >>

4. When you generate the template, tokens are inserted into the file to represent the locations for the cache resources.


```

?xml:lang="UTF-8" ?>
<template xmlns:ibm="http://www.ibm.com/xml/ns/websphere/cache" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:base="http://www.ibm.com/xml/ns/websphere/cache" http://www.ibm.com/xml/ns/websphere/cache.xsd">
  <property name="objCacheSize.value" type="int" value="5000" />
  <property name="objCacheSize.xpath" type="string" value="//ObjectCacheInstance[@name='objCache']/@cacheSize" />
  <property name="servletDiskCacheEvictionPolicyLT.value" type="int" value="75" />
  <property name="servletDiskCacheEvictionPolicyLT.xpath" type="string" value="//ServletCacheInstance[@name='servletCache']/@diskCacheEvictionPolicyLT" />
</template>

```

- Run the Configure step to apply the template to a live cell. You can specify values for the properties. In the following dialog, 5000 is specified for *objCache* and 75 is specified for *servletCache*.

The changes produced by running the configuration process are shown in the WebSphere Application Server console, as the following example shows for *objCache*.

Cache name	JNDI name	Scope	Cache size
objCache	jndi1	Cluster=cluster1	5000

You can also import the configuration data from the cell to see the changed contents of cache.xml. Note the highlighted changes.

```
<?xml version="1.0" encoding="UTF-8" ?>
- <CacheProviders xmlns="http://raf.rational.ibm.com/xml/ns/websphere/cache" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://raf.rational.ibm.com/xml/ns/websphere/cache http://raf.rational.ibm.com/xml/ns/websphere/cache" >
- <RAFW_cacheInstance>
- <ObjectCacheInstance cacheSize="5000" defaultPriority="1" disableDependencyId="false" diskCacheCleanupFrequency="0" diskCacheSizeInEntries="0" diskCacheSizeInMB="0" enableCacheReplication="false" enableDiskOffload="false" flushToDiskOnStop="false" pushFrequency="1" replicationType="NONE" useListenerContext="false">
- <CacheProvider RAFW_TYPE="reference" WASKey="provider" name="CacheProvider" />
- <DiskCacheCustomPerformanceSettings WASKey="diskCacheCustomPerformanceSettings" maxBufferedCacheIdsPerMetaEntry="10" maxBufferedCacheIdsPerMetaEntry="10" maxBufferedCacheIdsPerMetaEntry="10" />
- <DiskCacheEvictionPolicy WASKey="diskCacheEvictionPolicy" algorithm="NONE" highThreshold="80" lowThreshold="70" />
- </ObjectCacheInstance>
- <ServletCacheInstance cacheSize="4000" defaultPriority="1" diskCacheCleanupFrequency="0" diskCacheEntrySizeInMB="0" diskCacheSizeInEntries="0" diskCacheSizeInMB="0" enableCacheReplication="false" enableDiskOffload="false" flushToDiskOnStop="false" hashSize="1024" jndiName="jndi2" memCacheSize="1024" useListenerContext="false">
- <CacheProvider RAFW_TYPE="reference" WASKey="provider" name="CacheProvider" />
- <DiskCacheCustomPerformanceSettings WASKey="diskCacheCustomPerformanceSettings" maxBufferedCacheIdsPerMetaEntry="10" maxBufferedCacheIdsPerMetaEntry="10" maxBufferedCacheIdsPerMetaEntry="10" />
- <DiskCacheEvictionPolicy WASKey="diskCacheEvictionPolicy" algorithm="NONE" highThreshold="80" lowThreshold="75" />
- </ServletCacheInstance>
- </RAFW_cacheInstance>
</CacheProviders>
```

Comparing WebSphere configurations

Comparing configurations allows you to see how a deployed cell needs to be adjusted to fit a norm defined in a configuration template.

About this task

You can compare a WebSphere component (configuration template) to a live cell using the IBM Middleware Configuration for WebSphere plug-in. There are two steps to follow:

1. Create an application process to apply a configuration to a cell
2. Create an application process to run the comparison

Creating an application process to apply a configuration to a cell

Procedure

1. Create an application process.
2. View the application.
3. Click the **Processes** tab for the application.
4. Click **Create New Process**, then save.
5. Edit the process.
6. Drag and drop the **Install Component...** step onto the process editor.
 - a. Enter a name for the step.
 - b. Select the template component to use (from the template generation process).
 - c. Choose the **Configure WebSphere Application Server** component process.
7. Click **Save**.
8. Save the process.

Note: You can create custom processes to configure subsets of the configuration data.

Creating an application process to run the comparison Procedure

1. Create an application process.
2. View the application.
3. Click the **Processes** tab for the application.
4. Click **Create New Process**, then save.
5. Edit the process.
6. Drag and drop the **Run Process for Each Version...** step onto the process editor.
 - a. Enter a name for the step.
 - b. Select the template component to use (from the template generation process).
 - c. Choose the **Compare WebSphere Configuration (template)** component process.
7. Click **Save**.
8. Save the process.

Note: You can create custom processes to compare subsets of the configuration data.

Process steps in the IBM Middleware Configuration for WebSphere plug-in

Compare configuration

Compares the configuration from the mapped component to the configuration of the corresponding resource

Table 1. Input properties for the Compare configuration step

Name	Type	Description	Required
Options	String	Not supported.	No

Compare server configuration

Compares the server configuration from the mapped component to the configuration of the corresponding resource

Table 2. Input properties for the Compare server configuration step

Name	Type	Description	Required
Options	String	Not supported.	No

Generate Template

Generates the configuration template and creates a component version

Generate configuration

Generates the configuration representation using tokenized values

Manage configuration

Manages the configuration for the corresponding resource

Table 3. Input properties for the Manage configuration step

Name	Type	Description	Required
Options	String	Not supported.	No

Manage server configuration

Manages the server configuration for the corresponding resource

Table 4. Input properties for the Manage server configuration step

Name	Type	Description	Required
Options	String	Not supported.	No

Appendix. Notices

This information was developed for products and services offered in the U.S.A. IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

Intellectual Property Licensing
Legal and Intellectual Property Law
IBM Japan, Ltd.
1623-14, Shimotsuruma, Yamoto-shi
Kanagawa, 242-8502 Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law:

INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

Intellectual Property Dept. for Rational Software
IBM Corporation
5 Technology Park Drive
Westford, MA 01886
U.S.A.

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement or any equivalent agreement between us.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

This information is for planning purposes only. The information herein is subject to change before the products described become available.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs.

Each copy or any portion of these sample programs or any derivative work, must include a copyright notice as follows:

Portions of this code are derived from IBM Corp. Sample Programs. © Copyright IBM Corp. 2003, 2014.

If you are viewing this information softcopy, the photographs and color illustrations may not appear.

Trademarks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the Web at www.ibm.com/legal/copytrade.shtml.

Adobe, the Adobe logo, PostScript, and the PostScript logo are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States, and/or other countries.

Intel is a trademark or registered trademark of Intel Corporation or its subsidiaries in the United States and other countries.

Windows is a registered trademark of Microsoft Corporation in the United States, other countries, or both.

Java and all Java-based trademarks are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Other company, product, or service names may be trademarks or service marks of others.