BEA WebLogic Portal®

Upgrade Guide

Version 10.0
Revised: October 2007

Contents

B. Performing Database Upgrade Tasks Manually

Overview of the Upgrade Process to WebLogic Portal 10.0

This section provides an overview of the strategies and procedures for upgrading BEA WebLogic Portal to 10.0. You can upgrade from WebLogic Portal 9.2 and 9.2 Maintenance Packs (MP1 and MP2) applications to WebLogic Portal 10.0. You can also upgrade from WebLogic Portal 8.1 SP4, SP5, and SP6 applications directly to 10.0.

The following topics are covered in this chapter:

- Definitions
- Portal 10.0 MP1 Upgrade Overview
- Portal 9.2 and 9.2.x to 10.0 Upgrade Overview
- Portal 8.1 SP4, SP5, and SP6 to 10.0 Upgrade Overview
- Library Module Changes
- Support for Binary Compatibility
- Supported Features Comparison

Much of the WebLogic Portal upgrade is performed by running the WebLogic Upgrade Wizard. The WebLogic Upgrade Wizard is described in *Upgrading WebLogic Application Environments*.

Definitions

To clarify the different activities described by this document, a brief list of terms is included:

Migration

Moving an application and domain from a third-party technology to a BEA product. (For example, migrating a customer from IBM, webMethods or "home grown" to BEA.)

Upgrade

Updating BEA platform (and components) from older release or Service Pack to newer release or Maintenance Pack. This includes updating existing application and domain to run in a newer version, for example, 9.2 MP1 to 10.0.

The process required to upgrade an application environment depends on the scope of the application. An application environment includes a WebLogic domain and any applications and application resources associated with the domain. It may also include external resources, such as firewalls, load balancers, databases, and LDAP servers.

Interoperability

(1) The capability of an application deployed in one release or service pack to communicate with another application that is deployed in a different release or service pack. (2) The capability of WebLogic Platform components to communicate with third-party software using standard protocols.

Compatibility

Application built using one release/Service Pack running in another release or Service Pack. This might involve rebuilding the application.

Portal 10.0 MP1 Upgrade Overview

You can directly upgrade to WebLogic Portal 10.0 MP1 from the following versions:

- 10.0 to 10.0 MP1
- 9.2 or 9.2.x to 10.0 MP1
- 8.1.x to 10.0 MP1

The upgrade process involves upgrading portal applications and resources to WebLogic Portal 10.0 MP1.

Note: If you upgrade WebLogic Portal 10.0 installation to 10.0 MP1 using the upgrade installer, ensure that you run workshop4WP.exe with `-clean` and `-initialize` options after the upgrade. If you do not start Workshop with `-clean` and `-initialize` options after the upgrade, you will not be able to deploy web service applications.

Database upgrade from WebLogic Portal 10.0 to 10.0 MP1 is not required because the database structure is the same for 10.0 and 10.0 MP1.

The high-level steps in the upgrade process include the following:

1. Upgrade your domain using the WebLogic Upgrade Wizard. For more information, see "Upgrade Steps" on page 2-4.
2. Upgrade existing WebLogic Portal applications to run in WebLogic Portal 10.0 MP1. You can do this automatically using the Import utility that is provided in WebLogic Workshop. For additional instructions on using this utility, refer to Appendix B of the Portal Development Guide.

Tip: Ensure that you back up the customized domain start scripts before upgrading the domain to WebLogic Portal 10.0 MP1, because the changes will be overwritten when you run the WebLogic Portal 10.0 MP1 start scripts. You should manually copy any start script (and `setDomainEnv.cmd/sh`) modifications you made and want to preserve in the upgraded domain.

Portal 9.2 and 9.2.x to 10.0 Upgrade Overview

You can easily upgrade your WebLogic Portal 9.2 and 9.2.x applications to 10.0. The WebLogic Portal APIs have been maintained in WebLogic Portal 10.0 (except for the Commerce API, which was deprecated in 10.0), and most core formats for the database and file based assets have not changed. Where changes have been made, tools are provided to upgrade you to the new format, or provide manual changes where needed.

The upgrade process involves upgrading WebLogic Portal 9.2 or 9.2.x portal applications and resources to WebLogic Portal 10.0.

The high-level steps in the upgrade process include the following:

1. Upgrade your domain using the WebLogic Upgrade Wizard. For more information, see "Upgrade Steps" on page 2-4.
2. Upgrade existing WebLogic Portal 9.2 or 9.2.x applications to run in WebLogic Portal 10.0. You can do this automatically using the Import utility that is provided in WebLogic Workshop. For additional instructions on using this utility, refer to Appendix B of the Portal Development Guide.

Tip: If you customized how you set the domain in your start scripts, your changes will be overwritten when you run the WebLogic Portal 10.0 start scripts. You should manually copy any start script (and `setDomainEnv.cmd/sh`) modifications you made and want to preserve in the upgraded domain.

Portal 8.1 SP4, SP5, and SP6 to 10.0 Upgrade Overview

WebLogic Portal enables you to upgrade your 8.1 SP4, SP5, and SP6 applications directly to 10.0. Most WebLogic Portal APIs have been maintained in WebLogic Portal 10.0 (except for the Commerce API, which was deprecated in 10.0), and most core formats for the database and file based assets have not changed. Where changes have been made, tools are provided to upgrade you to the new format, or provide manual changes where needed.

The upgrade process involves upgrading WebLogic Portal 8.1 SP4, SP5, and SP6 portal applications and resources to WebLogic Portal 10.0. You are not required to upgrade from WebLogic Portal 8.1 to 9.2, and then to 10.0.

The high-level steps in the upgrade process include the following:

1. Upgrade your domain using the WebLogic Upgrade Wizard. For more information, see "Upgrade Steps" on page 2-4.
2. Upgrade existing WebLogic Portal 8.1 SP4, SP5, and SP6 applications to run in WebLogic Portal 10.0. You can do this automatically using the Import utility that is provided in WebLogic Workshop. For additional instructions on using this utility, refer to Appendix B of the Portal Development Guide.

Tip: If you customized how you set the domain in your start scripts, your changes will be overwritten when you run the WebLogic Portal 10.0 start scripts.

Library Module Changes

If you are upgrading from WebLogic Portal 8.1 to 10.0, the new libraries listed in Table 1 -1 are added to your `config.xml` file. Removed libraries are deleted from the `config.xml` file. If you are upgrading from WebLogic Portal 9.2 or 9.2 MP1 to 10.0, the new libraries are added and the module version number changes to 10.0 in your `config.xml` file.

Table 1 -1 lists the library modules that are new or have changed in WebLogic 10.0.

Table 1 -1 Changes to WebLogic Portal Library Modules

Library Module	New or Removed
content-management-web-lib	New in 10.0
content-management-app-lib	New in 10.0
wlp-analytics-app-lib	New in 10.0
wlp-analytics-web-lib	New in 10.0
wlp-tools-common-app-lib	New in 10.0
wlp-tools-common-web	New in 10.0
wlp-tools-framework-app-lib	New in 10.0
wlp-tools-framework-web-lib	New in 10.0
wlp-tools-custom-app-lib	New in 10.0
wlp-tools-custom-web-lib	New in 10.0
wlp-tools-content-app-lib	New in 10.0
wlp-tools-content-web-lib	New in 10.0
wlp-tools-analytics-app-lib	New in 10.0
wlp-tools-analytics-web-lib	New in 10.0
wlp-tools-im-app-lib	New in 10.0
wlp-tools-im-web-lib	New in 10.0
wlp-tools-portal-app-lib	New in 10.0
wlp-tools-portal-web-lib	New in 10.0
wlp-tools-serviceadmin-app-lib	New in 10.0
wlp-tools-serviceadmin-web-lib	New in 10.0
wlp-tools-ugm-app-lib	New in 10.0

Table 1 -1 Changes to WebLogic Portal Library Modules

Library Module	New or Removed
wlp-tools-ugm-web-lib	New in 10.0
wlp-tools-admin-web-lib	Removed from 9.2 and 9.2 MP1

Support for Binary Compatibility

WebLogic Portal 10.0 supports binary compatibility of Portal 9.2 applications deployed into a WebLogic Portal 10.0 domain (upgraded or new domain). Any references to pre-10.0 Workshop library modules will automatically be replaced by references to new Workshop libraries through the `/<BEAHOME>/wlserver_10.0/common/deployable-libraries/wlp-compat/` directory.

Supported Features Comparison

This section outlines significant feature changes between the WebLogic Portal 8.1 or 9.2 and the WebLogic Portal 10.0 release.

Security

Note: The following section applies only to an upgrade from WebLogic Portal 8.1 to 10.0.

WebLogic Portal 8.1 included a WebLogic Portal-specific RDBMSAuthenticator. This has been deprecated. WebLogic Server 9.2 contains a new default SQLAuthenticator authentication provider, which contains an RDBMS user store for users and groups. BEA recommends upgrading to the new WebLogic Server SQLAuthenticator.

When you run the WebLogic Upgrade Wizard to upgrade your domain, it determines whether or not you are using the RDBMSAuthenticator in your 8.1 installation. If the RDBMSAuthenticator is detected, the WebLogic Upgrade Wizard prompts you to choose whether to upgrade to the WebLogic SQLAuthenticator as your default authentication provider or continue to use your existing RDBMS user store:

- Upgrade users and groups – Choose to automatically upgrade your users and groups from WebLogic Portal 8.1 to the new RDBMS user store, which is part of the new default WebLogic SQLAuthenticator authentication provider. When you run the WebLogic Upgrade Wizard and it detects the Portal 8.1 RDBMSAuthenticator, you can select the

Upgrade RDBMSAuthenticator option. Selecting this option replaces the existing authentication provider with the new SQLAuthenticator authentication provider and upgrades your user store, including users and groups. Your `config.xml` file is also updated.

- Do not upgrade users and groups – Choose to continue to use the RDBMS user store from the default RDBMSAuthenticator in your WebLogic Portal 8.1 installation. When you run the WebLogic Upgrade Wizard and it detects your Portal 8.1 RDBMSAuthenticator, you can select the Do not upgrade RDBMSAuthenticator option. You can choose to manually upgrade your users and groups to the Portal 9.2 RDBMS user store later.

If you do not upgrade your user store during the domain upgrade process, you can perform a manual upgrade later. The script to upgrade from the WebLogic Portal-specific RDBMS Authenticator to the WebLogic SQL Authenticator is
`WEBLOGIC_HOME``/common/p13n/db/`*`dbms`*`/upgrade_fromdbmsauth_towlssqlauth.sql`

For additional information, see “Upgrading 8.1 and 9.2 PointBase Databases” on page B-3.

Note: If you upgrade a WebLogic Portal 8.1 application to 10.0 and you use the `UserProviderControl.createUser()` class in the upgraded domain, you might see a `javax.security.auth.login.LoginException` error when a new user attempts to log into WebLogic Portal. This occurs because by default new users in a WebLogic Portal 10.0 domain are created in the SQLAuthenticator and not in a migrated authentication provider (which normally is configured with a JAAS flag set to `REQUIRED`). Since the WebLogic Portal domain upgrade wizard does not adjust your JAAS settings or remove your existing authentication provider, you must adjust the JAAS setting or delete the authentication provider (as appropriate) to avoid this exception.

Interaction Management (Personalization)

Interaction Management enables you to develop, manage, and measure personalized portal applications. Personalization and Campaign management combine to form the foundation of Interaction Management. These functions help you target content to a desired audience.

When you run the BEA WebLogic Upgrade Wizard, the wizard upgrades your WebLogic Portal 8.1 or 9.2 interaction features, including Content Selectors, Placeholders, Segments, and Campaigns.

When you run the BEA WebLogic Upgrade Wizard and it detects your Portal 8.1 or 9.2 installation, you can select the Upgrade RDBMSAuthenticator option, as described in “Security” on page 1-6. Selecting this option replaces the existing authentication provider with the new WebLogic Server SQLAuthenticator and upgrades all content, including personalization features. You can also choose to manually upgrade your personalization features to the Portal 10.0 RDBMS user store later.

Note: Events will fire for a content repository that was upgraded to 10.0 (unless you turned event tracking turned off at the repository level). Events can include repository configuration changes, as well as content additions, updates, and deletions to the repository. Events will not fire for content in an 8.1 or 9.2 repository that was not upgraded. Events will be fired for content that is added, updated, or removed from that repository.

If you created a separate behavior tracking database in version 8.1 or 9.2, upgrade it as described in "Upgrading Separate 8.1 Behavior Tracking Databases" on page B-3.

Autonomy

A new Autonomy engine is installed with WebLogic Portal 9.2 and higher. It is disabled by default after upgrade so it can be properly configured before launching. Autonomy services that have been enabled in WebLogic Portal 8.1 are also disabled after upgrade. For information about installing and configuring search services, see the WebLogic Portal Integrating Search Guide.

In addition, the Autonomy APIs and the all content management APIs, including search, were deprecated in WebLogic Portal 9.2. If you want to continue to use the deprecated APIs, you need to manually add the older Autonomy APIs to your application, as described in "Using 8.1 Search within a 10.0 Portal Application" on page A-11.

For information about upgrading WebLogic Portal 8.1 to the new version of Autonomy that is included with WebLogic Portal 10.0, see "Upgrading Autonomy Search Services" on page A-10.

Upgrading to WebLogic Portal 10.0

This chapter describes upgrade tasks related to upgrading your WebLogic Portal *product* to 10.0. Tasks related to upgrading individual *portal projects* that you created with previous releases of WebLogic Portal are described in Appendix B of the *Portal Development Guide*.

This chapter contains the following sections:

- Before You Begin
- Upgrade Steps
- Functional Changes for WebLogic Portal 10.0

Before You Begin

You can upgrade WebLogic Portal 8.1 and 9.2 portal applications directly to 10.0. For information on upgrading earlier versions to WebLogic Portal 8.1 SP4, SP5, or SP6, see the Upgrade Guide for Version 8.1.

Database Changes During Upgrade

The WebLogic Upgrade Wizard executes database scripts to add and modify database tables for WebLogic Portal 10.0. Before you run the Upgrade Wizard to perform database upgrade tasks, you should perform full backups of all WebLogic Portal databases.

Note: Oracle 8.1.7 (DBMS version and drivers) is no longer supported. Upgrade to either Oracle 9i or 10G by following your vendor's instructions before you upgrade to WebLogic 10.0.

WebLogic Portal 8.1 supports only the Oracle Thin driver. A WebLogic Portal 8.1 domain that was configured with either the `weblogic.jdbcx.oracle.OracleDataSource` or the `weblogic.jdbc.oracle.OracleDriver` driver is not supported. The domain might upgrade successfully, but the following error will occur when you start the server:

```
<Warning> <JDBC> <BEA-001129> <Received exception while creating
connection for pool "portalDataSourceAlwaysXA": Invalid Oracle URL
specified>
<Error> <Deployer> <BEA-149205> <Failed to initialize the application
'portalDataSourceAlwaysXA' due to error
weblogic.application.ModuleException: - with nested exception:
[weblogic.common.ResourceException: Invalid Oracle URL specified]
```

When you run the 10.0 Upgrade Wizard, the default is to automatically upgrade the following two databases:

- The main WebLogic Portal database
- The GroupSpace database (for 9.2 to 10.0 upgrades)

Note: If the database objects associated with the database being upgraded do not exist (for example, the appsGroupSpaceDataSource exists in the domain, but the GroupSpace database objects do not exist in the database), then the database upgrade from WebLogic Portal 9.2.x to 10.0.x will return an SQLException. To successfully complete the database upgrade, you will have to run database scripts from the WebLogic Upgrade Wizard by selecting appsGroupSpaceDataSource and clicking on the **Run Scripts** button. Alternatively, you can use the following command to run the scripts:

```
create_db.cmd -database.properties=groupspace_database.properties
```

You can upgrade a database to a particular version only once.

You can choose to defer database upgrades and do them manually. Certain database upgrade tasks, such as dropping deprecated database objects, upgrading separate behavior tracking databases, and upgrading separate content management databases require a manual database upgrade.

For additional information on how to perform database upgrade tasks manually, see Appendix B, “Performing Database Upgrade Tasks Manually.”

DDL Files

Table 2 -2 lists the .SQL files containing the Data Definition Language (DDL) for the database upgrade from WebLogic Portal 8.1 to 10.0.

Table 2 -2 Files Containing the DDL for Upgrading from 8.1 to 10.0

Directory Name	File Name
<WL-HOME>/common/p13n/db/<DBMS>	p13n9_create_tables.sql
	p13n9_create_indexes.sql
	upgrade_fromdbmsauth_towlssqlauth.sql (if the domain is configured for the DBMS authenticator)
	dep9_drop_tables.sql (manual upgrade only)
<WL-HOME>/cm/db/<DBMS>	cm9_create_tables.sql
	cm9_create_indexes.sql
	cm9_create_fkeys.sql
	cm9_create_triggers.sql
	cmv9_create_tables.sql
	cmv9_create_indexes.sql
<WL-HOME>/portal/db/<DBMS>	pf9_create_tables.sql
	pf9_create_fkeys.sql
	pf9_create_views.sql
	comm_create_tables.sql
	comm_create_fkeys.sql
	comm_create_indexes.sql
	comm_create_views.sql
	comm_create_triggers.sql
	dep9_drop_tables.sql (manual upgrade only)

Table 2-3 lists the .SQL files containing the DDL for the database upgrade from WebLogic Portal 9.2 to10.0

Table 2-3 Files Containing the DDL for Upgrading from 9.2 to 10.0

Directory Name	File Name
<WL-HOME>/portal/db/<DBMS>	pf10_create_tables.sql
<WL-HOME>/cm/db/<DBMS>	cm10_create_tables.sql
	cmv10_create_tables.sql

Backing Up Your Applications and Data

Before you upgrade your application environment, you should manually back up the domain and any external application and application database resources in a separate process. You should back up the relevant information on all machines in the domain. The wizard backs up only the domain directory and does not preserve file permissions.

Upgrade Steps

This section provides general upgrade information for WebLogic Portal database and metadata files. The process consists of the following steps:

1. Verify that the WebLogic domain is not running.
2. Upgrade the portal domain using the WebLogic Upgrade Wizard before upgrading any applications.

   To start the WebLogic Upgrade Wizard in graphical mode and upgrade a WebLogic domain on a Windows platform, either:

   – Run `upgrade.cmd` from the `WL_HOME\common\bin` directory

     or

   – Go to **Start > All Programs > BEA Products > Tools > Domain Upgrade Wizard**

   Note: The WebLogic Upgrade Wizard is described in detail in *Upgrading WebLogic Application Environments*.

WARNING: If you have a platform domain containing WebLogic Integration 8.1, you cannot perform a domain upgrade using this wizard. Contact Technical Support.

The wizard upgrades your portal, content management, and personalization database data. It optionally upgrades your 8.1 RDBMSAuthenticator to the WebLogic 10.0 SQLAuthenticator. For more information on upgrading your user store, see "Security" on page 1-6.

3. Read Appendix A, "Functional Changes Affecting Your WebLogic Portal Environment".
4. As needed, upgrade individual applications, as described in Appendix B of the Portal Development Guide.

Note: The *BEA Workshop for WebLogic Platform Programmer's Guide*, available as part of the Workshop for WebLogic help, contains several useful topics that you should review as you prepare to upgrade your portal application. The Workshop for WebLogic documentation includes step-by-step instructions for using the Import Wizard, and detailed information about what happens during the upgrade process and any required manual pre- or post-upgrade tasks.

Caution: You should be familiar with the Workshop for WebLogic upgrade steps, and any related limitations, before you attempt to upgrade a WebLogic Portal application from 8.1 to 10.0. Before proceeding, refer to the *Workshop for WebLogic* documentation on *e-docs* or by choosing **Help** > **Help Contents** > **BEA Workshop for WebLogic Platform Programmer's Guide** in the main menu of the product.

Functional Changes for WebLogic Portal 10.0

Review the functional changes that are described in Appendix A, "Functional Changes Affecting Your WebLogic Portal Environment." If any manual upgrade tasks are required for your particular environment, perform those tasks as instructed.

Functional Changes Affecting Your WebLogic Portal Environment

This appendix describes functional changes in WebLogic Portal 9.2 and 10.0 that affect your upgraded environment and might require that you perform manual tasks.

This chapter includes the following sections:

Functional Changes from Portal 8.1 to 10.0

The following sections describe changes that occur when you upgrade directly from WebLogic Portal 8.1 to 10.0. You might be required to perform manual tasks.

This section includes these topics:

- Upgrading Look & Feels
- Import Wizard Does Not Handle Some Legacy Jar Files
- Changes in Behavior Between Struts 1.1 and 1.2
- Portlet State Persistence
- WSRP Security Compatibility
- Working with Encoding in HTTP Responses
- Disconnected Desktop Requires desktopStateShared Property
- Correcting Duplicate Portlet Category Names in an Upgraded Application
- Propagation Utility Web Application Obsolete
- Definition Labels Not Editable in 10.0
- Propagation Inventory Compatibility

Upgrading Federated Portals from 8.1 to 10.0

New features added in WebLogic Portal 10.0 to support federated portal propagation require you to perform the upgrade procedures described in this section. This section includes the following topics:

- Overview
- Upgrading Both Your Producer and Consumer Applications
- Upgrading Only the Producer Applications
- Upgrading Only the Consumer Applications
- Listing Producer Handles
- Updating Producer Registration Handles

Overview

WebLogic Portal 10.0 supports features of WSRP 2.0 that permit a more flexible and practical approach to propagation of federated portals. With WebLogic Portal 10.0, the consumer applications in staging and production environments can point to separate producers. The primary advantage of this new capability is that you can create and modify remote (proxy) portlets in a staging environment in isolation from the production environment.

Before WebLogic Portal 10.0, if you wanted to propagate WSRP consumer applications, the consumers on the source and destination systems had to point to the same producer. This configuration, which is described in detail in "WSRP Propagation" in the *WebLogic Portal 9.2 Production Operations Guide*, included several limitations.

This section explains the upgrade procedure for federated portals. The procedures described here apply to the following scenarios:

- Upgrading Both Your Producer and Consumer Applications
- Upgrading Only the Producer Applications
- Upgrading Only the Consumer Applications

Upgrading Both Your Producer and Consumer Applications

It is recommended that you upgrade both your consumer and producer applications to WebLogic Portal 10.0 in your source (staging) and destination (production) environments. Doing so allows you to take advantage of new propagation features and simplifies the propagation process.

1. Upgrade your consumer applications to WebLogic Portal 10.0. Perform the upgrade in both the source and destination environments.

   a. If you ever performed a bidirectional propagation (propagated from staging to production and then back to staging again) or if propagation fails, you need to follow this sub-step. If neither of these conditions applies to you, skip this sub-step and proceed to Step 2.

      Obtain the producer registration handles from each consumer application from both the source and destination systems. To do this, run the List Producers JSP utility as described in "Listing Producer Handles" on page A-5.

2. Upgrade your producer application to WebLogic Portal 10.0. Perform the upgrade in both the staging and production environments.

   a. If you ever performed a bidirectional propagation (propagated from staging to production and then back to staging again) or if propagation fails, you need to follow this sub-step. If neither of these conditions applies to you, skip this sub-step and proceed to Step 3.

      Update the producer registration handles. To do this, run the Update Registration Handles JSP utility as described in "Updating Producer Registration Handles" on page A-6.

3. Propagate the consumer application from the staging to the production environment using the propagation tools. See the *Production Operations Guide* for information on propagation.

Tip: When you propagate, you can adjust the scope to include only the Portal Framework resources.

This completes the upgrade process. It is now possible for consumer applications in staging and production environments to point to separate producers.

Note: If you want to retain a configuration where consumers in staging and production point to the same producer, you can do so; however, limitations described in WSRP Propagation in the *WebLogic Portal 9.2 Production Operations Guide* will no longer apply.

See the *Production Operations Guide* for detailed information on propagating portals.

Upgrading Only the Producer Applications

If you upgrade your domain and producer application but not the consumers, you need to follow the upgrade procedure described in this section. The procedure described in this section applies equally whether the consumer application(s) are currently at WebLogic Portal 8.1.x or 9.2.

Note: If you upgrade only the producer, you are required to use the propagation model described in WSRP Propagation in the *WebLogic Portal 9.2 Production Operations Guide*. In this model, consumer applications in both staging and production environments must point to the same producer.

Note: If you upgrade only the producer, you must perform steps 2 and 3 below each time you propagate remote portlets in your consumer applications.

1. Upgrade your producer application to WebLogic Portal 10.0.
2. Obtain the producer registration handles from each consumer application on both the source and destination system. To do this, run the List Producers JSP utility as described in "Listing Producer Handles" on page A-5.
3. Update the producer registration handles for each upgraded producer application on the destination system. To do this, run the Update Registrations JSP utility as described in "Updating Producer Registration Handles" on page A-6.

See the *Production Operations Guide* for detailed information on propagating portals.

Upgrading Only the Consumer Applications

If you upgrade your consumer application(s) but not the producer(s), you need to follow the upgrade procedure described in this section. The procedure described in this section applies equally whether the producer application is currently at WebLogic Portal 8.1.x or 9.2.

Note: If you want to retain a configuration where consumers in staging and production point to the same producer, you can do so; however, limitations described in WSRP Propagation in the *WebLogic Portal 9.2 Production Operations Guide* will no longer apply.

Note: If you upgrade only the consumer(s), you are required to use the propagation model described in WSRP Propagation in the *WebLogic Portal 9.2 Production Operations Guide*. In this model, consumer applications in both staging and production environments must point to the same producer.

Note: If you upgrade only the consumers, steps 2 and 3 are recommended each time you propagate your consumer applications.

1. Upgrade your consumer applications to WebLogic Portal 10.0. Perform the upgrade in both the staging and production environments.

   Note: If you are currently running a configuration where consumer applications in staging and production environments point to the same producer, the following steps are optional but recommended.

2. (Optional/Recommended) Obtain the producer registration handles from each consumer application on both the source and destination system. To do this, run the List Producers JSP utility as described in "Listing Producer Handles" on page A-5.

3. (Optional/Recommended) Update the producer registration handles for each upgraded producer application on the destination system. To do this, run the Update Registrations JSP utility as described in "Updating Producer Registration Handles" on page A-6.

Note: If you upgrade only the consumers, steps 2 and 3 are recommended each time you propagate your consumer applications.

Listing Producer Handles

Note: You only need to run the utility described in this section on the systems where your consumer applications are deployed.

This section explains how to run the List Producer JSP utility (`listProducers.jsp`) on both the staging and production systems on which your consumer applications are deployed. This utility obtains the registration handles for producers that have been previously added to your consumers.

Note: You must have administration privileges to run the JSPs.

Note: If your consumer application is deployed in a WebLogic Portal 9.2 environment, you must perform steps 1 and 2 below. If your consumer environment was already upgraded to WebLogic Portal 10.0, steps 1 and 2 are not required.

1. (This step is only required if your consumer application is deployed in a WebLogic 9.2 environment.) Extract the `listProducers.jsp` file from the following J2EE Shared Library. You can use an archive tool, such as WinZip, to extract the `listProducers.jsp` file from the archive:

   ```
   WEBLOGIC_HOME/portal/lib/modules/wlp-propagation-web-lib.war
   ```

2. (This step is only required if your consumer application is deployed in a WebLogic 9.2 environment.) Copy the `listProducers.jsp` file you extracted into the web applications in which your consumers are deployed. Do this on both the staging and production systems.
3. Run the List Producers JSP utility on the staging or source system. To do this, open a browser and enter the following URL:

   http://*host*:*port*/*EarProjectName*Propagation/wsrp/listProducers.jsp

   Where *EarProjectName* is the name of the enterprise application deployed to the server. For example:

   ```
   http://localhost:7001/myEarPropagation/wsrp/listProducers.jsp
   ```

4. When prompted, enter the correct user name and password for the server.
5. In the List Producers JSP, select the name of a consumer web application in the source system.
6. Click **Submit Query**. The JSP returns a table that lists the producer handle, WSDL, and registration handle for each producer that was added to the consumer.
7. Print or copy the information in this table.
8. Repeat these steps on the production/destination system.

Updating Producer Registration Handles

Note: You only need to run the utility described in this section on the systems where your producer application is deployed.

This section explains how to run the Update Registrations JSP utility (`updateRegistrations.jsp`) on both the staging and production systems. This utility updates the registration handles for each consumer application the currently references a given producer.

Note: If your producer application is deployed in a WebLogic Portal 9.2 environment, you must perform steps 1 and 2 below. If your producer environment was already upgraded to WebLogic Portal 10.0, steps 1 and 2 are not required.

1. (This step is only required if your producer application is deployed in a WebLogic 9.2 environment.) Extract the `updateRegistrations.jsp` file from the following J2EE Shared Library. You can use an archive tool, such as WinZip, to extract the `updateRegistrations.jsp` file from the archive:

   ```
   WEBLOGIC_HOME/portal/lib/modules/wlp-propagation-web-lib.war
   ```

2. (This step is only required if your producer application is deployed in a WebLogic 9.2 environment.) Copy the `updateRegistrations.jsp` file you extracted into the web application in which your producer is deployed. Do this on both the staging and production systems.
3. On the destination system, run the Update Registrations JSP. To do this, To do this, open a browser and enter the following URL:

   `http://`*`host`*`:`*`port`*`/`*`EarProjectName`*`Propagation/wsrp/updateRegistrations.jsp`

   Where *`EarProjectName`* is the name of the enterprise application deployed to the server. For example:

   ```
   http://localhost:7001/myEarPropagation/wsrp/updateRegistrations.jsp
   ```

4. When prompted, enter the correct user name and password for the server.
5. In the JSP, enter the source and corresponding destination registration handles that you obtained from the consumer applications by running the List Producers utility described previously in "Listing Producer Handles" on page A-5.
6. Click **Submit**.
7. Repeat this procedure for each consumer that is registered with the producer.

Note: The Update Registrations JSP copies all portlets related to the destination registration handle and gives the copies the source registration handle.

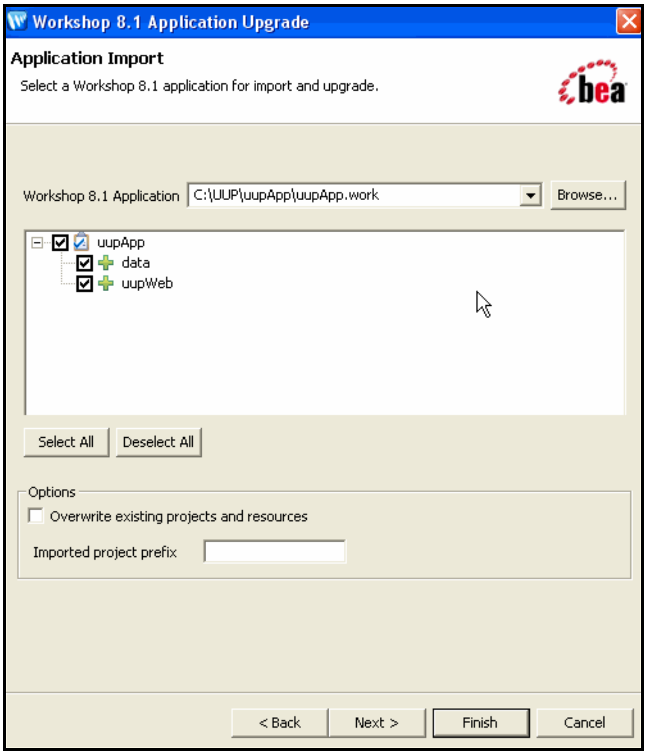
Upgrading UUPs

When you upgrade a Unified User Profile (UUP) from WebLogic Portal 8.1 to 10.0, the `p13n_ejb.jar` file is deleted and replaced with a new version of the WebLogic Portal 9.2 file. The new `p13n_ejb.jar` file is packaged in the library modules that ship with WebLogic Portal 9.2.

Perform the following steps to upgrade a UUP configured in WebLogic Portal 8.1 to WebLogic Portal 9.2:

1. Start WebLogic Workshop and create a new Workspace.
2. Create a new portal domain. Do not create a Portal EAR Project. For instructions on creating a new domain, see the *Portal Development Guide*.
3. Import your Portal 8.1 UUP application into your new environment by choosing **File > Import**.
4. In the Import dialog, open the **Other** folder, select **Workshop 8.1 Application**, and click **Next**.
5. In the Application Import dialog, click **Browse** and locate your 8.1 UUP application. Select the `.work` file and click **Open**. Verify that the check boxes for the UUP application are selected and click **Next**, as shown in Figure A-1.

Figure A-1 Locate the 8.1 UUP Application


Workshop 8.1 Application Upgrade
Application Import
Select a Workshop 8.1 application for import and upgrade.
bea
Workshop 8.1 Application
C:\UUP\uupApp\uupApp.work
Browse...
uupApp
data
uupWeb
Select All
Deselect All
Options
Overwrite existing projects and resources
Imported project prefix
< Back
Next >
Finish
Cancel

6. In the Source Upgrade dialog, click **NetUI Project Upgrader options** and select the **Use WebLogic J2EE Shared Libraries** check box. You can also click **JSP File Migrator options** and select the **Replace BEA NetUI tags with Apache Beehive tags** check box (if desired) and click **Finish**.

7. After the upgrade finishes, verify that the following actions occurred:

   – The `p13n-ejb.jar` file was removed from the EARContent directory of the UUP application.

   – The UUP EJB JAR file (for example, `UUPExample.jar`) exists in the EARContent directory of the UUP application.

   – The UUP EJB JAR file is referenced in a module entry in the `application.xml` file in the `<`*`UUPApplication`*`>/EARContent/META-INF/` directory.

   – As an example, the cache entry below was added to the `p13n-cache-config.xml` file in the `<`*`UUPApplication`*`>/EARContent/META-INF/` directory:

```
<p13n:cache>
     <p13n:name>UUPExampleCache</p13n:name>
     <p13n:description>Cache for UUP Example</p13n:description>
     <p13n:time-to-live>60000</p13n:time-to-live>
     <p13n:max-entries>100</p13n:max-entries>
</p13n:cache>
```

– Verify that the User Profile file (for example, `UUPExample.usr`) file exists in the `data/src/userprofiles/` directory (or where your Datasync folder exists).

8. Associate your portal application with your WebLogic Server by selecting the server in the **Servers** tab, right-clicking the server, and choosing **Add and Remove Projects**. Select the portal application from the **Available Projects** section, click **Add**, and then click **Finish**.

9. Build and publish your application. Verify the application by starting the WebLogic Server Administration Console and clicking **Deployments**. Verify that the UUP application is active. Then open the UUP application by expanding the tree and verifying that the UUP JAR file appears as an EJB.

Understanding JSP Tag Changes

Two JSP tags, `<ugm:login>` and `<ugm:logout>`, are deprecated in WebLogic Portal 9.2 and 10.0. The `<ugm:login>` and `<ugm:logout>` JSP tags were moved from the `ugm_taglib.jar` file to the `auth_taglib.jar` file.

After you upgrade to 10.0, you should use the `<auth:login>` and `<auth:logout>` JSP tags. The attributes and parameters are the same.

Upgrading Autonomy Search Services

If you are upgrading from WebLogic Portal 8.1 and used Autonomy search, you need to do the following to upgrade to the new version of Autonomy that is included with WebLogic Portal 10.0. The upgrade process does not automatically migrate your existing search settings and databases to the new search capabilities. You need to manually configure Autonomy search to work with your upgraded applications.

The new version of Autonomy is installed into the `portal/thirdparty/autonomy-wlp100` directory. For more information about configuring Autonomy search, see the *Search Guide*.

After installing and configuring Autonomy search, do the following to upgrade your application to use the new features:

1. Modify your WebLogic Portal 10.0 configuration files to match any Autonomy customizations you used.Table A-1 lists the files should be modified. If other configuration files were used that reference Autonomy search tools, you need to modify those as well.
2. Edit any start scripts that may start the 8.1 version of Autonomy services to ensure that they are not restarted. These are stopped automatically when you upgrade.

Table A-1 Mapping of WebLogic Portal 8.1 Configuration Files to WebLogic Portal 10.0 Configuration Files

File used with WebLogic Portal 8.1	File used with WebLogic Portal 10.0
`PortalSearchDRE.cfg`	`AutonomyIDOLServer.cfg`
`PortalSearchDiSH.cfg`	`AutonomyDiSH.cfg`
`PortalSearchHTTPFetch.cfg`	`HTTPFetch.cfg`
`PortalSearchAutoIndexer.cfg`	`FileSystemFetch.cfg`

Using 8.1 Search within a 10.0 Portal Application

If you wrote applications using the WebLogic Portal `com.bea.query` classes or using the 8.1 Autonomy API and want to continue using these applications without using the new 9.2 version of the Autonomy API, you need to manually add the older, deprecated APIs to your application.

However, if you continue to use the 8.1 API (either WebLogic Portal or Autonomy's), those APIs will take priority over the 10.0 API and are NOT compatible with the 10.0 Autonomy engine. This means that some 10.0 content management features will not work, such as full-text search.

If you want to continue to use the 8.1 version of Autonomy with your portal application, you must manually add the older APIs to your installation.

To add the older Autonomy version to your 10.0 application:

1. Upgrade your domain and application to 10.0.
2. Copy `autonomyCompat.jar` and `autonomyClient.1.5.0.jar` from `weblogic_home/portal/lib/thirdparty/search/81-compat-only` into the `application_home/APP-INF/lib` directory.
3. Using the Portal Administration Console, turn off full-text search for your repository. For more information, see the *Content Management Guide*.

4. Modify your domain start script to ensure that the 8.1 versions of the Autonomy are started rather than the new versions. For more information about start scripts, see the WebLogic Server documentation.
5. Start your domain.
6. Verify that search functionality is working.
   a. Index some content into Autonomy. For example, use FileSystemFetch.
   b. Execute a search in the Portal Search portlet.
   c. Verify that results are returned and no exceptions are encountered.
   d. Add content to the BEA Repository instance using the content management tools.
   e. Ensure that no exceptions are displayed; this would only occur if attempting to index the content, which will not occur if the preceding steps have been successfully executed.

The 8.1 Autonomy APIs and engines are now running.

Upgrading to Use 10.0 Search after Using 8.1 Search with a 10.0 Portal Application

If you want to upgrade to use the 10.0 version of Autonomy, after you have completed the steps in "Using 8.1 Search within a 10.0 Portal Application" on page A-11, you can reverse the implementation.

To upgrade to use the 10.0 version of Autonomy (after you have implemented 8.1 Autonomy with 9.2):

1. Locate and update any usages of the `com.bea.query.*` classes or any calls to the Autonomy client APIs in your applications and replace them with the appropriate calls to the 10.0 version of the Autonomy API.
2. Remove the `autonomyCompat.jar` and `autonomyClient1.5.0.jar` file from the `app-inf/lib` directory.
3. Modify your domain start script to execute the 10.0 version of the Autonomy engines.
4. Configure the 10.0 Autonomy engines to index your content. For more information, see the *Search Guide*.
5. Using the Portal Administration Console, turn on full-text search for your repository. For more information, see the *Content Management Guide*.

6. Run the startup script.

Manually Upgrading Passwords in Content Management Repositories

After the upgrade is complete, you must manually re-enter the passwords for your third-party repositories using the Administration Console; see the *Content Management Guide* for more information about editing repository settings.

Until you manually re-enter the passwords for your third-party repositories, you cannot access those repositories.

Maintaining Content Queries

In WebLogic 8.1 through WebLogic Portal 8.1 SP5, content query expressions were generated differently due to an order of precedence problem. The order of precedence was not maintained when executing a content query expression. For example, the following expression: (a && (b || c), gets evaluated/executed as (a && b || c).

This problem was fixed in Weblogic Portal 10.0 such that the order of precedence is now maintained when executing a content query expression. However, if you want to continue using the WebLogic Portal 8.1 through WebLogic Portal SP5 query behavior, you need to modify your domain scripts to define the following system property: `-Dwlp.disable.content.rule.fix=true.`

Upgrading Look & Feels

Portal Look & Feels in WebLogic Portal 8.1 used two configuration files for skins and skeletons (in the `/skins/<skin_name>` and `/skeletons/<skeleton_name>` directories): `skin.properties` and `skeleton.properties`. Both were text files, and `skeleton.properties` was optional.

In WebLogic Portal 10.0, both files are now XML, and both are required.

To upgrade a WebLogic Portal 8.1 Look and Feel to the WebLogic Portal 10.0 format:

1. Make sure the portal application containing the Look and Feel has been converted to WebLogic Portal 10.0, as described in the Portal Development Guide.

2. Open the Look & Feel in Workshop for WebLogic and re-save it. The configuration files are automatically converted to the new XML format.

Import Wizard Does Not Handle Some Legacy Jar Files

The `cm_taglib.jar` and the `pz_compat_taglib.jar` are deleted when you upgrade and Import Wizard flags all JSPs that refer to these taglibs, which have an unsupported taglib URIs. The JSPs will fail.

The `cm_taglib.jar` file was not installed by default in a new 8.1 web application, but if you added it to your application for backward compatibility, you must handle this file manually in your upgraded application.

Change all references to the `cm_taglib.jar` and the `pz_compat_taglib.jar` so that they use supported tags and APIs, and delete the obsolete jar files.

Changes in Behavior Between Struts 1.1 and 1.2

WebLogic Portal support for Struts is slightly different in 10.0 if you upgrade to Struts 1.2.

Struts 1.1 support in WebLogic Portal is the same as in previous releases, with the struts-adapter taglibs mapped to URIs using `web.xml`. You can use the `struts-1.1.war` library module instead of the new `struts-1.2.war` library module.

If you are upgrading to Struts 1.2, instead of mapping the struts and struts-adapter taglibs using `web.xml`, WebLogic Portal now relies on the JSP 1.2 implicit taglib mapping, wherein any `.tld` files in the `META-INF` directory in a JAR are implicitly mapped by the web container to the URI specified in the tld. For WebLogic Portal. these are in `struts-adapter.jar`, in the path `META-INF/tlds`.

Choose to use one of these two methods to upgrade to Struts 1.2:

- Modify all JSPs that use the struts taglibs to reference `http://bea.com/struts/adapter/tags-html` and `http://bea.com/struts/adapter/tags-nested` for the HTML and nested taglibs, and `http://struts.apache.org/tags-*` for the remainder of the taglibs that our adapter does not override.
- Extract the `.tlds` from both `struts.jar` (in `struts-1.1.war`) and from `struts-adapter.jar` (in our portal web library module) and copy them to `WEB-INF/tlds`. This allows for the case where you want to continue using the explicit tld mapping via `web.xml`.

Portlet State Persistence

In WebLogic Portal 8.1, minimized portlet states were persisted only for the session. You can use a workaround, described in the *Upgrade Guide for Version 8.1*, to set up a backing file that controls the states of portlets under the desktop.

The solution used in 8.1 will continue to work if you depend on this behavior, but BEA recommends that you use a new method of persisting the portlet state.

In 9.2, you can persist the portlet state in the database by using the `persistence-enabled` attribute in the `control-state-location` element of the `netuix-config.xml` file.

Here is an example of the `persistence-enabled` attribute:

```
<control-state-location>
    <session persistence-enabled="true"/>
</control-state-location>
```

If you set the `persistence-enabled` attribute to `true`, then the control tree state will be loaded from the database when a user logs in, and the new state will be stored when a user logs out. The control tree state is cleared when a user interacts with a portal.

The default and the only persistence type implemented is JDBC. The default implementation uses the BEA_PORTAL_FRAMEWORK_CONTROL_TREE_STATE property set of the user's ProfileWrapper; the ProfileWrapper must be created and stored in the user's http session. The ProfileWrapper is created by PortalServletFilter, which should be configured in the `web.xml` file. If the ProfileWrapper is not found in the user's session, the control tree state is not persisted.

Note: Page flow- and struts-related states are not persisted, as they are not part of the control tree state; page flow and struts portlets might not be in the exact same state when user logs out.

The child elements reader-class-name/writer-class-name provide reader and writer class names for this state-location. If you want to customize reader/writer behavior, you can implement the ControlStateReader/ControlStateWriter interfaces and configure them in the `netuix-config.xml` file.

WSRP Security Compatibility

Producer and consumer applications developed with WebLogic Portal 10.0 are compatible with producers and consumers developed with WebLogic Portal 8.1. That is, a portal developed with WebLogic Portal 10.0 can consume portlets deployed in a WebLogic Portal 8.1 domain.

Similarly, portlets exposed in a WebLogic Portal 10.0 producer can be consumed by an 8.1 consumer.

However, if you want to use your own key for the 8.1 or 9.2 consumer, you need to follow the procedures outlined the chapter “Establishing WSRP Security with SAML” in the *Federated Portals Guide*.

Working with Encoding in HTTP Responses

This section describes changes in how the encoding is set on the HTTP response.

8.1 Methodology in Setting Encoding

In WebLogic Portal 8.1, the following method of setting encoding was used:

1. Examine the `.portal` file for a `directive.page` element. If that element is present, obtain the encoding from an attribute there.
2. If the element is not present, use the JSP encoding configuration, which looks at the `<encoding>` element in the `<jsp-param>` section of the `web.xml` file; the default is ISO-8859-1 if these elements are missing.

Both of these mechanisms are now deprecated.

10.0 Methodology in Setting Encoding

In WebLogic Portal 10.0, the following method of setting encoding is used:

1. Examine the `netuix:desktop` element for an `encoding` attribute, and use that value if present.
2. If the first check is not applicable, examine the `.portal` file for the deprecated `directive.page` element. If that element is present, pick up the encoding from an attribute there.
3. Examine `netuix-config.xml` for a `<defaultEncoding>` element, and use the `encoding` attribute there.
4. If the previous check is not applicable, fall back to the deprecated `<encoding>` element in the `<jsp-param>` section of the `web.xml` file.

The old methods continue to work, but to eliminate any deprecation warnings, BEA recommends that you use the new mechanisms; for example:

```
<netuix:desktop ... encoding="UTF-8" /> in your .portal file
```

or

`<defaultEncoding encoding="UTF-8" />` in your netuix-config.xml file

Editing Encoding in Workshop for WebLogic

When you select the desktop element from the portal view or outline view while editing a `.portal` file, the workbench now includes a new `encoding` property in the property view. The value for a new portal defaults to `UTF-8`. You can edit the value using an editable drop-down menu.

The drop-down menu initially displays a list of all the basic IANA encoding values as well as the extended encodings for Chinese, Korean, and Japanese. The values presented in the list are descriptive display names that are converted to actual IANA names when saved to the `.portal` file.

If a desired value does not display in the drop-down menu, you can type it in. When you press Enter, the validator checks the encoding to verify that it is a valid and supported encoding. The value you enter can be from the extended encoding set and can be an IANA name, alias, or canonical name for the encoding. If the encoding fails validation, a warning message displays; you can choose to override the validation and accept the value anyway. The value will be stored as is, in the `.portal` file for the desktop `encoding` attribute.

Disconnected Desktop Requires desktopStateShared Property

For compatibility with 8.1 and prior releases, and to support the few desirable uses of a disconnected desktop, WebLogic Portal 10.0 provides a new, but deprecated, boolean property called `desktopStateShared` for the `StandalonePortletURL` and the associated JSP tag. You can use this property to retain the previous "disconnected desktop" behavior. The default value of this property and attribute will be `true`, causing the portlet to be connected to a desktop.

Explicitly setting either the `path` or `contextualPath` properties on the URL or tag also disassociates the resulting standalone portlet from its originating desktop. This also holds true for any URLs generated within the context of the standlone portlet—setting `path` or `contextualPath` causes the resulting URLs to become disassociated with the originating desktop.

Correcting Duplicate Portlet Category Names in an Upgraded Application

In 8.1 and previous releases of WebLogic Portal it was possible, though not recommended, to create more than one portlet category with the same name, at the same level in the hierarchy. In 10.0, this operation is not permitted. (You can use the same name for more than one category, but they must not be “peers” in the hierarchy.)

When you upgrade a portal application to 10.0, any duplicate portlet category names that were used previously are preserved. It is extremely important that you edit these category names to be unique; otherwise the WebLogic Portal propagation tools might cause unexpected results, or errors might occur during the propagation process.

Propagation Utility Web Application Obsolete

The Propagation Utility web application, `propagation.war`, became obsolete as of WebLogic Portal 9.2. This application was introduced in a patch for WebLogic Portal 8.1 SP4 and later incorporated into WebLogic Portal 8.1 SP5. If you are upgrading a WebLogic Portal web application in which you previously installed the file `propagation.war` into the root directory of any WebLogic Portal enterprise applications, it is recommended that you remove the file before or after upgrading to WebLogic Portal 10.0.

Definition Labels Not Editable in 10.0

In WebLogic Portal 8.1, the capability to edit definition labels existed, but was not recommended. Modifying the definition label could have unintended implications; for example, exposing a protected resource or breaking WSRP (which uses the definition label as the portlet handle).

As of WebLogic Portal 9.2, this functionality has been replaced by a much richer ability to move portal resources (books, pages, desktops) between production and development environments, without losing user customizations or changing labels. These new features include XIP and the propagation utility. XIP allows you to target individual portal resources to import and export between development and production systems, and you can specify the scope (library, admin, or visitor). For more information about WebLogic Portal’s propagation tools, see the Production Operations Guide.

Propagation Inventory Compatibility

WebLogic Portal inventories saved with WebLogic Portal 8.1 or 9.2 cannot be used with WebLogic Portal 10.0 propagation tools.

Functional Changes from Portal 9.2 to 10.0

The following sections describe changes that occur when you upgrade from WebLogic Portal 9.2 or 9.2 MP1 to 10.0. You might be required to perform manual tasks.

This section includes these topics:

- Upgrading UUPs
- Disconnected Desktop Requires desktopStateShared Property
- Upgrading Search Services
- WSRP Security Compatibility
- Upgrading Propagation Scripts
- Propagation Inventory Compatibility

Upgrading Federated Portals from 9.2 to 10.0

The procedure for upgrading federated portals from 9.2 to 10.0 is identical to the procedure for updating from 8.1 to 10.0. See "Upgrading Federated Portals from 8.1 to 10.0" on page A-2 for these detailed instructions.

Upgrading UUPs

Your WebLogic Portal 9.2 or 9.2 MP1 UUP automatically works in WebLogic Portal 10.0. You do not need to upgrade your 9.2 UUP.

Disconnected Desktop Requires desktopStateShared Property

For backward compatibility to WebLogic Portal 8.1 and previous releases, and to support the few desirable uses of a disconnected desktop, WebLogic Portal 9.2 provided a new, but deprecated, boolean property called `desktopStateShared` for the `StandalonePortletURL` and the associated JSP tag.

Note: This property is retained in 10.0 and is still deprecated.

You can use this property to retain the previous "disconnected desktop" behavior. The default value of this property and attribute will be `true`, causing the portlet to be connected to a desktop. Explicitly setting either the `path` or `contextualPath` properties on the URL or tag also disassociates the resulting standalone portlet from its originating desktop. This also holds true for any URLs generated within the context of the standlone portlet—setting `path` or `contextualPath` causes the resulting URLs to become disassociated with the originating desktop.

Upgrading Search Services

Upgrading search services involves updating your Autonomy configuration files and re-indexing your content.

Optionally, you can choose to migrate your existing Autonomy installation to use the 10.0 Autonomy installation directory. WebLogic Portal 10.0 ships with the same version of Autonomy as was shipped with WebLogic Portal 9.2, but the installation directory has changed. The new installation directory is:

```
\\WebLogic_HOME\cm\thirdparty\autonomy-wlp10
```

For production environments, BEA recommends continuing to use the 9.2 installation directory (`\\WebLogic_HOME\portal\thirdparty\autonomy-wlp92`).

For development environments, BEA recommends discontinuing the use the of the previous installation directory and using the new installation directory.

Note: You **MUST** re-index your BEA content within your production environment, see the *Integrating Search Guide* for detailed instructions.

Update Autonomy Configuration Files

Update all of necessary Autonomy configuration files to use the new location, see Configuring Autonomy on Your Target Server" in the *Integrating Search Guide* for more details on Autonomy configuration files.

Supporting Double Quotes in BEA Content Search Queries

To continue to support double-quotes in your search queries, you must make the following changes to your BEACMRepoFetch.cfg file.

To support double-quotes (for example, “quotes”):

1. On the Autonomy host, edit the file at

   ```
   //autonomyhome/operatingsystem/internal/BEACMRepoFetch/BEACMRepoFetch.c
   fg
   ```

   For example, on a Linux host, this might be at

```
//WebLogic_Home/portal/thirdparty/autonomy/linux-intel/internal/BEACMRe
poFetch/BEACMRepoFetch.cfg
```

2. In the `[BEACMRepoIDXImport]` section, add the line:

```
ImportBifIncludeQuotes=true
```

Supporting Japanese Characters in your BEA Content Queries

To continue support of querying Japanese characters, you must make the following modifications to your `AutonomyIDOLServer.cfg` file:

1. In the `[FieldProcessing]` section, do the following:
   a. Increment the `[FieldProcessing]` Number field value by one.
   b. Add a line of the form `19=SetLanguage` using one number greater than the largest number listed in the `[FieldProcessing]` section.

   For example, change from this:

   ```
   [FieldProcessing]
   Number=19
   0=SetIndexFields
   ...
   18=SetMoreReferenceFields
   ```

   To this:

   ```
   [FieldProcessing]
   Number=20
   0=SetIndexFields
   ...
   18=SetMoreReferenceFields
   19=SetLanguage
   ```

2. Before the `[Properties]` section, add the following section:

   ```
   [SetLanguage]
   Property=LanguageFields
   PropertyFieldCSVs=*/DRELANGUAGETYPE
   ```

3. In the`[Properties]` section, add a line of the form `19=LanguageFields` using one number greater than the largest number listed in the `[Properties]` section

For example, change from this:

```
[Properties]
0=IndexFields
...
18=MoreReferenceFields
```

To this:

```
[Properties]
0=IndexFields
...
18=MoreReferenceFields
19=LanguageFields
```

4. Before the `[Security]` section, add the following section:

   ```
   [LanguageFields]
   LanguageType=TRUE
   ```

5. Save your changes.

Migrating your Autonomy Installation to the 10.0 Directory

Optionally, you can migrate your 9.2 Autonomy configuration to the 10.0 installation.

Note: For upgraded production environments, BEA recommends continuing to use the existing 9.2 Autonomy location.

To migrate your 9.2 Autonomy configuration and data to the new 10.0 location:

1. Copy Required Files to the New Location.
2. Export Indexed Data from the 9.2 IDOL Location.
3. Import 9.2 Indexed Data to the New 10.0 IDOL Location.
4. Re-index all BEA content, see the *Integrating Search Guide* for detailed instructions.

Copy Required Files to the New Location

On the IDOL server, copy the following folders from the 9.2 location to the 10.0 location.

1. From the `\\wlp-92\IDOLserver\IDOL\content\` folder, copy the following subfolders to the new content folder in the `\\wlp-10\IDOLserver\IDOL\content\` location:

   - `\dynterm folder\`
   - `\main`
   - `\nodetable`
   - `\numeric`
   - `\refindex`
   - `\sortfield`
   - `\status`
   - `\storedstate`
   - `\tagindex`

2. From the `\\wlp-92\IDOLserver\IDOL\indextasks\` folder, copy the following subfolders to the new `\\wlp-10\IDOLserver\IDOL\indextasks\` folder:
   - `\failed`
   - `\incoming`
   - `\main`
   - `\outgoing`
   - `\queue`
   - `\temp`
3. From the `\\wlp-92\IDOLserver\IDOL\agentstore\` folder, copy the following subfolders to the new `\\wlp-10\IDOLserver\IDOL\agentstore\` folder:
   - `\dynterm folder`
   - `\main`
   - `\nodetable`
   - `\numeric`
   - `\refindex`
   - `\sortfield`
   - `\status`
   - `\storedstate`
   - `\tagindex`
4. From the `\\wlp-92\IDOLserver\IDOL\category\` folder, copy the following subfolders to the new `\\wlp-10\IDOLserver\IDOL\category\` folder:
   - `\cluster`

- `\imex`
- `\taxonomy`
- `\category`
- `\failed`
- `\outgoing`

5. From the \\wlp-92\IDOLserver\IDOL\community\ folder, copy the following subfolders to the new \\wlp-10\IDOLserver\IDOL\agentstore\ folder in the 10.0 location:
   - `\users`
   - `\temp`
   - `\queue`

Export Indexed Data from the 9.2 IDOL Location

You need to export all indexed data from your 9.2 IDOL Server and re-import it into the 10.0 location.

Use the following command from your browser to export indexed content. Use the correct host name and port for your Autonomy configuration. You also need to indicate a file name, including a directory, to which to index content.

```
http://host:port/DREEXPORTIDX?FileName<filename>&Compress<true\false>
```

Import 9.2 Indexed Data to the New 10.0 IDOL Location

After exporting your indexed data from the 9.2 location, use the following command to import the content to the new WebLogic Portal 10.0 location. Use the correct host name and port for your Autonomy configuration and the file you created when you exported your data.

```
http://host:port/DREADD?<filename>
```

Re-Index BEA Content

After completing the above sections, you must re-index your BEA content. For detailed instructions on re-indexing BEA content, see the *Integrating Search Guide*.

WSRP Security Compatibility

Producer and consumer applications developed with WebLogic Portal 10.0 are compatible with producers and consumers developed with WebLogic Portal 9.2. That is, a portal developed with WebLogic Portal 10.0 can consume portlets deployed in a WebLogic Portal 9.2 domain.

Similarly, portlets exposed in a WebLogic Portal 10.0 producer can be consumed by a 9.2 consumer.

However, if you want to use your own key for the 9.2 or 10.0 consumer, you need to follow the procedures outlined the chapter “Establishing WSRP Security with SAML” in the *Federated Portals Guide*.

Upgrading Propagation Scripts

If you created any propagation Ant scripts in 9.2, you must either manually update them or regenerate the scripts in 10.0. This change is required because several JAR file names have changed and the package name for all propagation Ant tasks has changed.

If you regenerate your scripts using Workshop for WebLogic, the correct filenames and package name will be used automatically. If you must update your scripts manually (for instance, if you created them manually in the first place), you need to make the following changes:

The propagation ant tasks are now located in the package `com.bea.propagation.ant.taskdefs.`

The following table lists the 9.2 JAR file names and the updated names for 10.0:

9.2 JAR Name	10.0 JAR Name
`p13n_prop.jar`	`propagation.jar`
`p13n_prop_online.jar`	`propagation_online.jar`
`p13n_prop_ant.jar`	`propagation_ant.jar`

Propagation Inventory Compatibility

WebLogic Portal inventories saved with WebLogic Portal 8.1 or 9.2 cannot be used with WebLogic Portal 10.0 propagation tools.

Performing Database Upgrade Tasks Manually

This appendix describes how to perform database upgrade tasks manually if you do not use the WebLogic Upgrade Wizard to upgrade from 8.1 SP4, SP5, or SP6, or from 9.2 or 9.2 MP1 to WebLogic Portal 10.0.

See the `README.txt` file and the `upgrade_db.properties` file in the *`WL_HOME`*`\portal\upgrade\db` directory for detailed information about the files used in the upgrade and specific upgrade steps.

This appendix contains the following sections:

Upgrading your Main WebLogic Portal 8.1 or 9.2 Database

If you did not upgrade your main WebLogic Portal database from 8.1 or 9.2 to 10.0 using the WebLogic Upgrade Wizard, you can perform the upgrade manually. If you have a 9.2 GroupSpace database, you must also upgrade that database. See "Upgrading a 9.2 GroupSpace Database" on page B-4 for more information.

Perform the following steps to upgrade your WebLogic Portal database:

1. Shut down WebLogic Server.
2. Back up your database data as described by your database vendor.
3. Edit the `WL_HOME\portal\upgrade\db\upgrade_db.properties` file for your database environment. Replace the @ symbols and the text between the symbols with the correct values for `@DB_USER@`, `@DB_PASSWORD@`, `@DB_HOST@`, `@DB_PORT@`, and `@DB_NAME@`.

   Note: For PointBase, follow the instructions in "Upgrading 8.1 and 9.2 PointBase Databases" on page B-3.

4. In the `upgrade_db.properties` file, modify and uncomment the `files=` setting for your portal database. Follow the instructions in the `upgrade_db.properties` file for your specific type of upgrade (9.2 to 10.0 or 8.1 to 10.0.)
5. Run the following script: `upgrade_db.cmd/.sh`.
6. If you are upgrading from 8.1 to 10.0, run the `upgrade_db_data.cmd/.sh` script to upgrade database data.

   Tip: To upgrade your user store, follow the instructions in "Upgrading to the 9.2 and 10.0 WebLogic Server SQL Authenticator" on page B-3.

7. If you are upgrading a main WebLogic Portal database from 8.1 SP4, SP5, or SP6, determine if patch # CR244936 has been applied. If this patch was applied, manual upgrade of the main WebLogic Portal database is complete. This patch is described in BEA WebLogic Portal 8.1 SP 5 Release Notes (`http://edocs.beasys.com/wlp/docs81/relnotes/relnotes.html#1117746`) as follows:

   CR237251 – If markup contains more than 4000 bytes, an attempt to store it in the database causes an error. If your database does not contain a PF_MARKUP_XML table that has been populated with data (for example, Select count(*) from PF_MARKUP_XML returns 0 rows) and the PF_MARKUP_DEFINITION table does not contain the BEGIN_XML and END_XML columns, then this patch was not applied.

8. If you are upgrading a main WebLogic Portal database from 8.1 SP4, SP5, or SP6 and patch # CR244936 has not been applied, run the following script against your database:

   `<WL_HOME>\portal\db\dbms_name\pf9_drop_columns.sql`

Upgrading 8.1 and 9.2 PointBase Databases

To manually upgrade your PointBase database from 8.1 or 9.2, you must first run the WebLogic Upgrade Wizard to perform a domain upgrade, and select **No** to skip the database upgrade. You must also copy the database files themselves (`weblogic_eval.dbn` and `weblogic_eval$#.wal`) into the `\WL_HOME\portal\upgrade\db` directory and then copy them back to the domain directory.

See the `README.txt` file for the specific steps to manually upgrade your PointBase database.

Upgrading to the 9.2 and 10.0 WebLogic Server SQL Authenticator

If you did not upgrade your user store using the WebLogic Upgrade Wizard during the domain upgrade process, you can perform a manual upgrade later. Use the following script to upgrade from the WebLogic Portal-specific RDBMS Authenticator to the WebLogic SQL Authenticator:

`WL_HOME\common\p13n\db\dbms_name\upgrade_fromdbmsauth_tosqlauth.sql`

Upgrading Separate 8.1 Behavior Tracking Databases

If you created a separate behavior tracking database in 8.1, you can upgrade it manually.

Perform the following steps to upgrade a separate behavior tracking database:

1. Shut down WebLogic Server.
2. Back up your database data as described by your database vendor.
3. Edit the settings in the `upgrade_db.properties` file for your behavior tracking database. Replace the @ symbols and the text between the symbols with the correct values for `@DB_USER@`, `@DB_PASSWORD@`, `@DB_HOST@`, `@DB_PORT@`, and `@DB_NAME@`.

4. In the `upgrade_db.properties` file, modify and uncomment the `files=` setting for your behavior tracking database. Follow the instructions in the `upgrade_db.properties` file.
5. Run the following script: `upgrade_db.cmd/.sh`.

Upgrading Additional 8.1 and 9.2 Content Management Databases

The default content management database is upgraded automatically. If you created an additional content management database in 8.1 or 9.2,you can upgrade it manually.

Perform the following steps to upgrade an additional Content Management database:

1. Shut down WebLogic Server.
2. Back up your database data as described by your database vendor.
3. Update the settings in the `upgrade_db.properties` file for your content management database. Replace the @ symbols and the text between the symbols with the correct values for `@DB_USER@`, `@DB_PASSWORD@`, `@DB_HOST@`, `@DB_PORT@`, and `@DB_NAME@`.
4. In the `upgrade_db.properties` file, modify and uncomment the `files=` setting for your content management database. Follow the instructions in the `upgrade_db.properties` file.
5. Run the following script: `upgrade_db.cmd/.sh`.

Upgrading a 9.2 GroupSpace Database

If you did not use the WebLogic Upgrade Wizard to upgrade your WebLogic Portal 9.2 GroupSpace repository database, you can manually upgrade it.

Perform the following steps to upgrade a GroupSpace database:

1. Shut down WebLogic Server.
2. Back up your database data as described by your database vendor.
3. In the `upgrade_db.properties` file, modify and uncomment the `files=` setting for your GroupSpace database. Follow the instructions in the `upgrade_db.properties` file.
4. Run the following script: `upgrade_db.cmd/.sh`.

Dropping Deprecated RDBMS Authenticator Tables After Upgrade

After you upgrade to the WebLogic Server SQL Authenticator, you can drop the tables associated with the WebLogic Portal RDBMS Authenticator using the following script:

WL_HOME`\common\p13n\db\`*dbms_name*`\dep9_drop_tables.sql`

Dropping Deprecated Compoze Database Tables After Upgrade

After you upgrade to WebLogic Portal 10.0, you can drop the tables associated with Compoze/Collaboration using the following script:

WL_HOME`\portal\db\`*dbms_name*`\dep9_drop_tables.sql`