



BEA WebLogic Integration™

Upgrade Guide

Copyright

Copyright © 1995-2006 BEA Systems, Inc. All Rights Reserved.

Restricted Rights Legend

This software is protected by copyright, and may be protected by patent laws. No copying or other use of this software is permitted unless you have entered into a license agreement with BEA authorizing such use. This document is protected by copyright and may not be copied photocopied, reproduced, translated, or reduced to any electronic medium or machine readable form, in whole or in part, without prior consent, in writing, from BEA Systems, Inc.

Information in this document is subject to change without notice and does not represent a commitment on the part of BEA Systems. THE DOCUMENTATION IS PROVIDED “AS IS” WITHOUT WARRANTY OF ANY KIND INCLUDING WITHOUT LIMITATION, ANY WARRANTY OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. FURTHER, BEA SYSTEMS DOES NOT WARRANT, GUARANTEE, OR MAKE ANY REPRESENTATIONS REGARDING THE USE, OR THE RESULTS OF THE USE, OF THE DOCUMENT IN TERMS OF CORRECTNESS, ACCURACY, RELIABILITY, OR OTHERWISE.

Trademarks and Service Marks

Copyright © 1995-2005 BEA Systems, Inc. All Rights Reserved. BEA, BEA JRockit, BEA WebLogic Portal, BEA WebLogic Server, BEA WebLogic Workshop, Built on BEA, Jolt, JoltBeans, SteelThread, Top End, Tuxedo, and WebLogic are registered trademarks of BEA Systems, Inc. BEA AquaLogic, BEA AquaLogic Data Services Platform, BEA AquaLogic Enterprise Security, BEA AquaLogic Service Bus, BEA AquaLogic Service Registry, BEA Builder, BEA Campaign Manager for WebLogic, BEA eLink, BEA Liquid Data for WebLogic, BEA Manager, BEA MessageQ, BEA WebLogic Commerce Server, BEA WebLogic Communications Platform, BEA WebLogic Enterprise, BEA WebLogic Enterprise Platform, BEA WebLogic Enterprise Security, BEA WebLogic Express, BEA WebLogic Integration, BEA WebLogic Java Adapter for Mainframe, BEA WebLogic JDriver, BEA WebLogic Log Central, BEA WebLogic Network Gatekeeper, BEA WebLogic Personalization Server, BEA WebLogic Personal Messaging API, BEA WebLogic Platform, BEA WebLogic Portlets for Groupware Integration, BEA WebLogic Server Process Edition, BEA WebLogic SIP Server, BEA WebLogic WorkGroup Edition, Dev2Dev, Liquid Computing, and Think Liquid are trademarks of BEA Systems, Inc. BEA Mission Critical Support, BEA Mission Critical Support Continuum, and BEA SOA Self Assessment are service marks of BEA Systems, Inc.

All other names and marks are property of their respective owners.

Contents

1. Overview

Scope of this Document	1-1
What's New that Impacts the Upgrade Process	1-1
The Upgrade Process	1-4
Terminology Used in This Document	1-6

2. The Upgrade Process

Prerequisites	2-1
Upgrading Your WebLogic Integration Domain to 9.2	2-1
Upgrading Applications to WebLogic Integration 9.2	2-2
Before You Begin	2-3
The Upgrade Process	2-3
Using the Import Wizard to Upgrade Your Application	2-4
Using the Command Line to Upgrade Applications	2-5
Using an Ant task to Upgrade Your Applications	2-7
Understanding the Upgrade Log	2-8
Outages During or After Deployment	2-9
Manual Changes You Need to Do After Upgrade	2-9
Testing the Upgrade	2-9

3. Upgrading Business Processes and Control Files for Use with WebLogic Integration 9.2

Upgrading Business Processes (JPDs)	3-1
---	-----

Upgrading JCX or WebLogic Integration Control Files	3-6
Upgrading JCS Control Files	3-7

4. Control Annotations

Application View Controls	4-2
Data Transformation Controls	4-3
Email Controls	4-5
File Controls	4-7
HTTP Controls	4-8
Message Broker Controls	4-9
MQSeries Controls	4-10
Process Controls	4-13
Service Broker Controls	4-15
Task Control Control-level Annotations	4-16
Task Control Method-level Annotations	4-20
Task Worker Control Control-level Annotation	4-31
Task Worker Control Method-level Annotations	4-31
Dynamic Transformation Controls	4-39
WebLogic Integration JMS Controls	4-40
TIBCO RV Controls	4-42

5. Other Component Changes

Control Factories	5-1
XQuery Files	5-2
JPD and Control Callbacks	5-2
JPD Process Language	5-3
DTF Transformation	5-3
Channel Files	5-4

Overview

This section includes the following topics:

- [“Scope of this Document” on page 1-1](#)
- [“What’s New that Impacts the Upgrade Process” on page 1-1](#)
- [“The Upgrade Process” on page 1-4](#)
- [“Terminology Used in This Document” on page 1-6](#)

Scope of this Document

This document describes the procedures required to upgrade your application environment from BEA WebLogic Integration™ 8.1 SP4, 8.1 SP5 and 8.1 SP6 or 8.5, 8.5 SP5 and 8.5 SP6 to BEA WebLogic Integration 9.2. An application environment includes applications, the WebLogic domains in which they are deployed, and any application data associated with the domain, and may include external resources, such as database servers, firewalls, load balancers, and LDAP servers.

What’s New that Impacts the Upgrade Process

[Table 1-1](#) introduces a subset of enhancements that are being introduced in WebLogic Integration 9.2 that also impact the upgrade process. For a comprehensive list of new features in this release, see [WebLogic Integration 9.2 Release Notes](#).

Note: Table 1-1 does not provide a complete list of new features. As a result of these enhancements, WebLogic Platform 8.1 applications will not be binary-compatible and will require automated or manual changes during an upgrade to WebLogic Integration 9.2.

Table 1-1 What's New that Impacts the Upgrade Process

Enhancement	Description
Eclipse-based IDE	<p>The BEA Workshop for WebLogic Platform 9.2 IDE is now based on Eclipse 3.1.2, delivering a software development platform that blends open source and commercial software, and is standards-based. The IDE provides access to core Eclipse features, such as source editing, JUnit test integration, and refactoring. It also includes a robust tool set available from the Eclipse Web Tools Platform (WTP) 1.0 project, including server plug-ins for multiple runtimes. For more information about Eclipse 3.1.2 and Eclipse WTP 1.0, see http://www.eclipse.org.</p> <p>In WebLogic Integration 9.2, the IDE delivers design views for developing JPDs. Additional design views to support Web Service and Java control development will be provided in the future.</p> <p>Note: In February 2005, BEA joined the Eclipse Foundation as a Strategic Developer and Board Member to further its commitment to open source and standards organizations.</p>
Apache Beehive 2.0	<p>BEA Workshop for WebLogic Platform 9.2 provides tools to make building applications with Apache Beehive 2.0 easier, including support for:</p> <ul style="list-style-type: none"> • Java controls—based on Plain Old Java Objects (POJO) architecture. • NetUI—based on Struts, and including Page Flows and JSP tags. <p>Apache Beehive is an open-source programming model designed to simplify J2EE programming tasks and is built on J2EE and Struts.</p> <p>BEA developed Beehive, which evolved from its BEA Workshop for WebLogic Platform product, to provide a simplified development model for all WebLogic applications. For more information about Apache Beehive, see http://beehive.apache.org.</p>

Table 1-1 What's New that Impacts the Upgrade Process

Enhancement	Description
Metadata Annotations	<p>The programming model for Web Services, EJBs, Java controls, and Java Page Flows uses the new J2SE 5.0 metadata annotation language (specified by JSR-175). In this programming model, you create a Java file that uses annotations to specify the shape and characteristics of the component. From these annotations, the compiler takes care of generating the required supporting artifacts, including Java source code, deployment descriptors, and so on.</p> <p>The annotations that you can specify include:</p> <ul style="list-style-type: none"> • Web Service annotations defined by <i>Web Services Metadata for the Java Platform specification</i> (JSR-181). For more information, see http://www.jcp.org/en/jsr/detail?id=181. • EJB annotations as defined in the EJBGen Reference in <i>Programming WebLogic Enterprise JavaBeans</i>. • Java control and NetUI (Page Flow) annotations as defined by Apache Beehive 2.0. For more information, see http://beehive.apache.org. • WebLogic-specific annotations to support security policy configuration, asynchronous failure and response, conversational Web Service support, and more. For more information, see Programming the JWS File in <i>Programming Web Services for WebLogic Server</i>.
Web Service Policy Framework	<p>Security and authentication configuration has been enhanced to use the standards-based Web Services Policy Framework (WS-Policy), as described in Configuring Message-Level Security for Web Services.</p>
XMLBean and XQuery API Standards	<p>WebLogic 9.2 supports new standards for XMLBeans and XQuery APIs, as described in XMLBeans and XQuery Implementations.</p>

Table 1-1 What's New that Impacts the Upgrade Process

Enhancement	Description
Changes in Directory Structure	<p>WebLogic Server 9.2 offers the following enhancements to the structure of the WebLogic domain directory:</p> <ul style="list-style-type: none"> To improve configuration management and promote XML file validation, WebLogic Server supports the specification of domain configuration data in multiple files, including <code>config.xml</code> in the new <code>domain_name/config</code> directory. (Here, <code>domain_name</code> specifies the domain directory.) In previous releases, the <code>config.xml</code> file was the repository for all configuration information. Now, new subdirectories of the <code>config</code> directory maintain configuration modules for diagnostic, JDBC, JMS, Node Manager, and security subsystems. Each configuration file adheres to an XML Schema definition. Startup and shutdown scripts are maintained in the <code>domain_name/bin</code> directory. In previous releases, they were stored in the root directory of the domain. <p>In addition to the structural enhancements to the domain directory, WebLogic Server supports new utilities for managing changes to server configuration. These new tools enable you to implement a secure, predictable means for distributing configuration changes in a domain. For more information, see Understanding Domain Configuration.</p>

The Upgrade Process

WebLogic Integration allows you to upgrade using any one of the following methods:

- **Single-Step**—provides you the option to import the 8.x applications into the Eclipse workspace and then begin the upgrade process.
- **Multi-Step**—provides you the option to first import the files, upgrade them (individually or on a folder basis) and then cancel the upgrade process, if you would like to continue at a later point of time.
- **Upgrade from the command line**—provides you the option to upgrade 8.x applications using an Ant task. In this method you use an 8.x work file as the source parameter and the Eclipse workspace as the destination parameter.

At a high-level, the steps involved in upgrading from WebLogic Integration 8.x to WebLogic Integration 9.2 are:

- Use the Upgrade Wizard, or the `upgradeStarter` command or the upgrade Ant task to upgrade WebLogic Integration 8.x.
- Use the WebLogic Upgrade Wizard to upgrade the domain. The Wizard updates the directory structure, and the following to WebLogic Integration 9.2:
 - a. WebLogic Domain
 - b. Domain Database Tables
 - c. Custom Security Providers
 - d. Node Managers
- Upgrade External resources such as Firewalls, Load Balancers, Databases, and LDAP servers. For example, Apache 1.3 should be upgraded to 2.0 and Oracle 8.1.7 should be upgraded to Oracle 9i to function with WebLogic Integration 9.2.
- Check and compare the supported configurations for [WebLogic Integration 9.x](#) and [8.x](#) and ensure that the configurations are upgraded to match version 9.x specifications.
- Use the Application Upgrade tool to upgrade the Application Source. You can run the tool from the Eclipse IDE or the command line. The Eclipse plug-in is an extension to the BEA Workshop for WebLogic Platform framework. It updates the:
 - a. Project model
 - b. Application source code
 - c. WebLogic Integration 8.x artifacts such as JPD, DTF/XQuery, JCX Controls and JCS files to WebLogic Integration 9.2 standards. It changes all file extensions such as `.jpd`, `.jpf`, `.app`, `.jcs`, `.jcx`, and `.jws` to `.java`. It also updates all JPD, DTF, JCX, and JCS Annotations to the JSR 175 based Annotation model.
 - d. Optionally upgrade XQuery 2002 files to XQuery 2004. You may require to update these files manually.
- If required, you need to manually upgrade application components.
 - Note:** Ensure that WebLogic Integration 8.x application process instances are run to completion in the appropriate environment before they are used in WebLogic Integration 9.2 environment.
- You need to recompile and redeploy the applications once the upgrade is complete.

Terminology Used in This Document

We recommend that, before proceeding, you familiarize yourself with the following terminology:

- **Compatibility**—The capability of an application built using one release or service pack to run in another release or service pack, with or without rebuilding the application.
- **DTF**—Data Transformation File. DTF files have an extension of `.dtf` and contain definitions of a data transformation that can be invoked from a JPD. For more information, see, <http://e-docs.bea.com/workshop/docs81/doc/en/integration/dttutorial/tutWLIDataTransIntro.html>
- **IDE**—Integrated Development Environment. This refers to the BEA Workshop for WebLogic Platform development environment based on Eclipse, which is a development platform that blends open source and commercial software, and is standards-based.
- **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 BEA WebLogic Platform™ components to communicate with third-party software via standard protocols.
- **JCS**—Java Control Source file. JCS files have an extension of `.jcs`. For more information, see, <http://edocs.bea.com/workshop/docs81/doc/en/workshop/guide/controls/conGettingStartedWithJavaControls.html>.
- **JCX**—Java Control Extension file. JCX files have an extension of `.jcx`. For more information, see <http://edocs.bea.com/workshop/docs81/doc/en/workshop/guide/devenv/conJwiFiles.html>.
- **JPD**—A Java Process defined in a Process Definition for a Java file.
- **JSR**—A Java Specification Request. For more information, see <http://jcp.org/en/jsr/overview>.
- **Migrate**—To move an application or domain configuration from a third-party product to a BEA product.
- **Upgrade**—To upgrade your JPD 8.1 source and related files to JPD 9.2 artifacts.
- **XQ**—A short form for XQuery in some cases. XQuery files on Weblogic Platform have an extension of `.xq`. They contain only the XQuery. So, the term XQ could refer to the XQ file or the XQuery itself.

The Upgrade Process

This document provides information on upgrading from WebLogic Integration™ 8.1 to WebLogic Integration 9.2. Topics discussed include:

- “Prerequisites” on page 2-1
- “Upgrading Your WebLogic Integration Domain to 9.2” on page 2-1
- “Upgrading Applications to WebLogic Integration 9.2” on page 2-2

Prerequisites

Before beginning the upgrade process, go through the *Upgrading WebLogic Application Environments* Guide. This guide describes the procedures to upgrade your application environment to WebLogic 9.2. An application environment includes applications, the WebLogic domains in which they are deployed, any application data associated with the domain, and may include external resources, such as database servers, firewalls, load balancers, and LDAP servers.

Upgrading Your WebLogic Integration Domain to 9.2

The WebLogic 9.2 Upgrade Wizard allows you to upgrade domains created only in WebLogic Integration 8.1 SP4, 8.1 SP5, 8.1 SP6, 8.5, 8.5 SP5 and 8.5 SP6 (also referred to as 8.1.x and 8.5.x in this document).

At a high-level, the steps performed by WebLogic Integration during a domain upgrade are as follows:

- Adds resources to support advanced Web services including the `file store`, `WseeFileStore`, and the JMS server, `WseeJmsServer`, and its associated JMS module.
- Updates and adds JMS and JDBC resources to support WebLogic Platform applications.
- Removes user-defined applications that have been deployed in the domain.
- Removes deprecated applications that have been deployed in the domain.
- Removes the `JWSQueueTransport EJB`, if it is present in the domain.
- Adds shared library modules to support Personalization (`P13n`) applications.
- Adds External Event Generators.
- Adds the `SQLAuthenticator` security provider to the domain.

Note: The users `portaladmin` and `weblogic` are added to the `SQLAuthenticator` security provider. You can remove these users from the `DefaultAuthenticator` security provider after the domain is upgraded.

- Updates the following, if any data source is configured to use the PointBase database:
 - The database is automatically loaded in embedded mode and upgraded to PointBase v5.1.
 - The `pointbase.ini` file is updated to set `database.home`, `documentation.home` and `pbembedded.lic` for PointBase v5.1.
 - The database files are renamed from `workshop` to `weblogic_eval` and the associated `datasource JDBC` driver URLs accordingly fixed.

The PointBase related environment settings are carried over to the upgraded domain scripts, `setDomainEnv.cmd` and `setDomainEnv.sh`.

For more information on the domain upgrade process and things you need to keep in mind during upgrade, see *Upgrading a WebLogic Domain* available at the following URL:
http://edocs.bea.com/common/docs92/upgrade/upgrade_dom.html

Upgrading Applications to WebLogic Integration 9.2

WebLogic Integration 9.2 provides a set of utilities that allow you to upgrade your 8.1.x or 8.5.x includes applications to 9.2. This section describes how to upgrade applications built using WebLogic Integration.

Note that during upgrade, the logic and intent of the application is not altered. WebLogic Integration simply migrates the code to make it compatible with 9.2. This would involve changes like making the applications compatible with the Eclipse Framework and converting Javadoc annotations to JSR-175 compliant annotations, among others.

Before You Begin

Verify that you have completed the following tasks:

- Migrated all your applications to 8.1 (SP4, SP5, or SP6) or to 8.5 (or higher). For information on upgrading your older applications to these versions, see [WebLogic Integration 8.1 Upgrade Guide](#).
- Undeployed all version 8.1 applications before you upgrade the server.
- Verified that the WebLogic domain is not running.
- Check out your version 8.1.x or 8.5.x application sources that need to be upgraded.
- Upgraded the WebLogic Integration domain using the WebLogic Platform Domain Upgrade Wizard. For more information on upgrading your domain, see [Upgrading a WebLogic Domain](#) available at the following URL:
http://edocs.bea.com/common/docs92/upgrade/upgrade_dom.html

The Upgrade Process

Application upgrade is a three-step process: going through a list of items that will be upgraded, performing the application upgrade and fixing errors reported in the log to ensure your applications run in WebLogic Integration 9.2 without any problems.

You can choose to upgrade your user applications using the Import Wizard or the Command Line utility - both provided by BEA Workshop for WebLogic Platform. Alternatively, you could use an Ant task. The subsequent sections describe these methods.

Notes:

- The upgrader does not support upgrade of user-developed helper source files such as Helper classes and 7.x controls.
- If you have specified any custom classloader hierarchies in addition to the standard classloader inversion hierarchy enforced by the 8.x process application, the application upgrader will not recognize these hierarchies. Instead, it generates a standard classloader inversion hierarchy that a WebLogic Integration 9.2 process application

requires. You will then need to re-create your custom class loader hierarchy after the application upgrade is complete and then specify the Classloader hierarchy in the `weblogic-application.xml` file.

Using the Import Wizard to Upgrade Your Application

You can use the Import Wizard provided by BEA Workshop for WebLogic Platform to upgrade your applications to 9.2. The wizard does not alter the logic and intent of the existing 8.1 application, nor extract the application from any source repository. It migrates the 8.1 source artifacts into the 9.2 source and project model. However, it retains the 8.1 Javadoc annotations as they do not require any special processing in 9.2. These annotations are also retained to facilitate any manual processing that may be required after upgrading the application.

Following are some of the tasks executed by the import wizard.

- Imports upgraded source code to the WebLogic Integration 9.2 workspace that you have defined.
- Upgrades 8.1.x or 8.5.x annotations to WebLogic Integration 9.2.
- Migrates your WebLogic Integration 8.1.x or 8.5.x source artifacts to WebLogic Integration 9.2. This involves the following steps:
 - Converts WebLogic Integration 8.1.x or 8.5.x project types to WebLogic Integration 9.2.
 - Optionally moves libraries from the 8.1.x or 8.5.x application Libraries folder to a new EAR project in the upgraded application.
 - Moves JSP files into a WebContent directory.
 - Upgrades Beehive NetUI JSP tags to WebLogic Integration 9.2.
 - Optionally migrates Beehive NetUI JSP tags to Apache Beehive JSP tags.
 - Moves XSD files that are in a Schema project into a Schemas folder of the Utility project.
 - Moves Java packages and source into a `src` directory.

Note: When you upgrade an 8.x application with an EJB or non-web or non-utility project that uses JPD or Process Proxy to make an RMI call to the JPD, do not add a process facet to all the non-web or non-utility projects. Instead, add the Library (Process Libraries) to the project's `java` build path as follows:

- Select **Project**→**Properties**→**Java build**.

- Select the Libraries tab, click **Add Library**, and select **Process Libraries**.

Using the Command Line to Upgrade Applications

BEA Workshop for WebLogic Platform also provides a command line utility that converts the entire application to work with WebLogic Integration 9.2.

The utility does not check out or delete files. It also does not check in the newly upgraded files automatically. It just copies the essential files over to the WebLogic Workshop 9.2 workspace for migration.

Note: When you run the command line utility, use a 1.5 implementation of the JRE. Ensure that the classpath includes `<%ECLIPSE_HOME%/startup.jar`.

The command to upgrade your application is as follows:

```
java -cp %ECLIPSE_HOME%/startup.jar
-Dwlv.application=%WORK_FILE%
-Dweblogic.home=%WL_HOME%
org.eclipse.core.launcher.Main
-application com.bea.wlv.upgrade.upgradeStarter
-data %WORKSPACE%
-pluginCustomization %PREFS_FILE%
```

where,

<code>ECLIPSE_HOME</code>	Refers to the path to the directory containing the <code>startup.jar</code> . The default for BEA Workshop for WebLogic Platform is: <code>BEA_HOME/workshop92/eclipse</code>
<code>-Dweblogic.home=WL_HOME</code>	Refers to the location of WebLogic Server root folder. By default, this is: <code>BEA_HOME/weblogic92</code>
<code>-Dwlv.application=WORK_FILE</code>	Refers to the application that requires the upgrade. Replace <code>WORK_FILE</code> with the work file name corresponding to the WebLogic Workshop 8.1 that you want to upgrade.
<code>-application com.bea.wlv.upgrade.upgradeStarter</code>	Refers to the Eclipse plug-in extension point used to execute this command.
<code>-data WORKSPACE</code>	Refers to the name of the target workspace where you want the upgraded application to go. This can be any directory in which you want the version 9.2 application files generated.

<code>[-pluginCustomization Prefs_FILE]</code>	<p>Specifies a properties file used to set options for the upgrade. Replace the <code>Prefs_FILE</code> with the name of a properties file containing a number of key-value pairs. The possible properties are:</p> <ul style="list-style-type: none"> • <code>application</code> refers to the plug-in extension point to execute at run time. • <code>weblogic.home</code> refers to the location of the WebLogic Server root folder. • <code>data</code> refers to the name of the target workspace where the upgraded application resides. The name of the parameter is provided by Eclipse and it cannot be overwritten. • <code>wlw.application</code> refers to the name of the application work file. • <code>pluginCustomization</code> refers to the name of a properties file containing a number of key-value pairs.
Optional Parameters	
<code>com.bea.wlw.upgrade/upgradeHarnessAbortOnError=true/false</code>	<p>If you do not specify this attribute, the default is <code>false</code>. In this case, the upgrader tries to continue after an error. When it is set to <code>true</code>, the upgrade process fails when it encounters any error. These errors are listed in the log file.</p>
<code>com.bea.wlw.upgrade/upgradeHarnessMessageLevel</code>	<p>This attribute indicates a message level. If you do not specify this attribute, the upgrader logs all messages. You can specify the following values for this attribute:</p> <ul style="list-style-type: none"> • <code>INFO</code>: Displays all messages. • <code>WARNING</code>: Displays warning, error, and fatal messages, and suppresses informational messages. • <code>ERROR</code>: Displays only error and fatal messages.
<code>com.bea.wlw.upgrade/migrateJSPPreferences=true/false</code>	<p>If you do not specify this attribute, the default is <code>false</code>. When it is set to <code>true</code>, the upgrade process migrates the JSP files to their new Beehive annotation.</p>
<code>com.bea.wlw.upgrade/useJ2EESharedLibraries=true/false</code>	<p>When you set this attribute to <code>false</code>, the upgrade copies the web application libraries to <code>WEB-INF/lib</code>. The upgrade uses J2EE shared libraries by default.</p>
<code>com.bea.wlw.upgrade/upgradeHarnessReportOnly=true/false</code>	<p>When you set this attribute to <code>true</code>, the upgrade report is generated. The default setting is <code>false</code>, and with this setting both the report and upgrade are performed.</p>

<code>com.bea.wlw.upgrade /upgradeHarnessLogFile= log file location></code>	Use this attribute to specify the location of the upgrade log file. The default value is <code><workspace location>/metadata/upgrade.log</code>
<code>com.bea.wlw.upgrade /upgradeProjectImportOverwrite=true/false</code>	Use this attribute to specify whether an existing project is overwritten in the event of a conflict in project name. The default value is <code>false</code> .
<code>com.bea.wlw.upgrade /upgradeProjectImportPrefix</code>	Use this attribute to specify an optional prefix to append to all imported projects.
<code>com.bea.wlw.upgrade /upgraderPrefMoveResourceBundle = true/false</code>	Use this attribute to specify whether files with the <code>.properties</code> extension are copied or moved from the web content folder to the source file folder. The default value is <code>false</code> .

Using an Ant task to Upgrade Your Applications

You can use the Ant task to upgrade to WebLogic Integration 9.2.

The command line upgrade contains an Ant task. You can locate the class of the Ant task in the `wlw-upgrade.jar`, deployed in the

`./<WORKSHOP_HOME>/eclipse/plugins/com.bea.wlw.upgrade_9.2.0` folder.

Note: When you run the Ant task, ensure that the `<%ECLIPSE_HOME%/startup.jar` is on the classpath of the task, as specified by the `classpathref` attribute in the following sample Ant task.

A following sample shows how you can invoke an Ant task:

```
<target name="workshopUpgrade">

  <echo message="${workshop.home}/eclipse"/>
  <path id="eclipse.classpath">

    <fileset dir="${workshop.home}/eclipse/plugins"
      includes="com.bea.wlw.**/wlw-upgrade.jar"/>

  </path>

  <taskdef name="upgradeTask"
    classname="com.bea.wlw.upgrade.cmdline.UpgradeTask"
    classpathref="eclipse.classpath"/>
```

```
<upgradeTask data=%WORKSPACE%
eclipseHome=%ECLIPSE_HOME%
weblogicHome=%WL_HOME%
pluginCustomization=%PREFS_FILE%
wlwApplication=%WORK_FILE%/>
</target>
```

where,

WORKSPACE	Refers to the Eclipse workspace that the WebLogic Integration 8.1.x or 8.5.x application is imported and upgraded to.
ECLIPSE_HOME	Refers to the Eclipse directory containing the <code>startup.jar</code> .
WL_HOME	Refers to the location of the root folder of WebLogic Server.
PREFS_FILE	Refers to the location of an optional preference file used during import or upgrade.
WORK_FILE	Refers to the location of the work file WebLogic Workshop 8.1 application to be imported or upgraded.

Understanding the Upgrade Log

WebLogic Integration 9.2 generates a log of the upgrade changes, errors, and warnings, irrespective of the upgrade process you choose. If you use the wizard, this log is displayed in a dialog that you can review before the process is complete.

The log file is generated after the upgrade is completed and it is saved as:

```
UPGRADE_WORKSPACE_HOME\.metadata\upgrade.log
```

A log message in the file appears as follows:

```
!SUBENTRY 1 com.bea.wlw.upgrade severity_level date time
!MESSAGE Upgrade-related message.
```

The severity level contains two numbers with the same meaning. The date and time entries refer to when the upgrade was attempted. The upgrade-related message describes what was done, warned about, or the error that occurred. The following is a snippet containing two log entry examples:

```
!SUBENTRY 1 com.bea.wlw.upgrade 2 2 2006-02-27 17:17:53.687
```

```
!MESSAGE The 9.2 control context only supports a subset of the 8.1 control
context APIs. Please see the Workshop for WebLogic upgrade documentation for
more information.
```

```
!SUBENTRY 1 com.bea.wlw.upgrade 1 1 2006-02-27 17:17:53.687
```

```
!MESSAGE The import "com.bea.control.JwsContext" needs to be updated.
```

Outages During or After Deployment

You might encounter certain outages while trying to deploy your upgraded application. For information on outages, see the [“Known Limitations”](#) section, in WebLogic Integration Release Notes.

Manual Changes You Need to Do After Upgrade

- After upgrading your 8.1 domain, ensure that you have set the security policies on the Compatibility 8.1.x Task Plan and enabled the 'Anonymous' role in the Create Policy. Use the Worklist Administration Console (the default authorization provider) to set the Create Policy for the Compatibility 8.1.x task plan. If you are using a third-party authorizer, use the related third-party client tools to set the policy.
- If you are directly using MFL-derived XMLBeans types for internal use or during conversion of data from non-XML to XML as an intermediate form, you need to manually specify namespaces in the element constructors of these XQuery transformations upgraded from 8.x.

Testing the Upgrade

After the upgrade is complete, you can optionally build and deploy the upgraded application to verify if the upgrade is successful. You can ensure that the required files have been moved or are available in the correct locations as follows:

- JPD Annotation Processor:
 - The project and component beans that JPD requires must be available in the `build/EJB` directory.
 - The `wli-process.xml`, `wli-subscriptions.xml`, and `wlw-manifest.xml` should be available in the `build/processoutput/WEB-INF/` directory.
- Channel Builder contains the `wli-channels.xml` file in the `earProject/ear/META-INF/` directory.

The Upgrade Process

- JDT Builder
- Beehive Control Builder
- XML Beans Builder

Upgrading Business Processes and Control Files for Use with WebLogic Integration 9.2

The following sections describe updates required to Business Processes and Control files before they can be used with WebLogic Integration 9.2.

- [Upgrading Business Processes \(JPDs\)](#)
- [Upgrading JCX or WebLogic Integration Control Files](#)
- [Upgrading JCS Control Files](#)

Upgrading Business Processes (JPDs)

In the WebLogic Integration 9.2 environment, all JPD files are given a `.java` extension rather than their proprietary extension of `.jpd`. All WebLogic Integration JPD 8.1 annotations are upgraded to JSR 175-based annotations. All the JPD 8.1 or 8.5 annotations are categorized into: common, control and JPD annotations.

In WebLogic Integration 8.1.x and 8.5.x, `jpdContext` within a JPD was annotated with `@common:context`. However, in WebLogic Integration 9.2, `jpdContext` is upgraded to `@com.bea.wli.jpd.Context()`.

For example, a WebLogic Integration 8.x JPD Business Process Annotation is as follows:

```
/**
 * @jpd:process process::
 * <process name="EchoAsync">
 *   <clientRequest name="Client Request" method="clientRequest"/>
 *   <perform name="Perform" method="perform"/>
```

```
*   <controlSend name="start" method="myTimerStart"/>
*   <clientCallback name="Client Response"
method="clientResponseCallbackHandler"/>
*   <transaction name="Commit"/>
* </process>::
*/
```

After the JPD is upgraded to WebLogic Integration 9.2, the annotation is as follows:

```
@Process(
    process="<process name=\"EchoAsync\">\n" +
        "   <clientRequest name=\"Client Request\"
method=\"clientRequest\"/>\n" +
        "   <perform name=\"Perform\" method=\"perform\"/>\n" +
        "   <controlSend name=\"start\" method=\"myTimerStart\"/>\n" +
        "   <clientCallback name=\"Client Response\"
method=\"clientResponseCallbackHandler\"/>\n" +
        "   <transaction name=\"Commit\"/>\n" +
    "</process>"
)
```

Note: `wliTimerControl` is the default WebLogic Integration timer control for JPDs.

Note: JMS transport was supported in WebLogic Integration 8.1 using `jws.queue` for use by BEA Workshop for WebLogic Platform based artifacts (such as business processes and JWS). In WebLogic Integration 9.2, JWS uses `weblogic.wsee.DefaultQueue` as the default queue for JMS transport whereas business processes still require `jws.queue`. Also note that even though WebLogic Integration 9.2 allows you to specify any JMS queue for JMS transport, you must not use `jws.queue` for new JWS applications as that causes conflict in the WebLogic Integration enabled domain. Do not use `jws.queue` or customize the queue name (using `jws.properties`) with JWS applications in WebLogic Integration 9.2.

[Table 3-1](#) provides WebLogic Integration JPD 8.1.x or 8.5.x to 9.2 JPD annotation upgrade information.

Table 3-1 JPD Annotations

8.1 Annotation	Attribute	9.2 Annotation	Attribute	Comments
jpd:ebxml		Ebxml		Specifies the ebXML parameters for a process.
	ebxml-action-mode		ebxmlActionMode	
	ebxml-service-name		ebxmlServiceName	
	protocol-name		protocolName	
jpd:ebxml-method		EbXMLMethod		Specifies the ebXML parameters for a method.
	envelope		envelope	
jpd:mb-static-subscription		MessageBroker.StaticSubscription		Specifies the subscription parameters for a business process.
	channel-name		channelName	
	xquery		xquery	
	filter-value-match		filterValueMatch	
	message-metadata		messageMetadata	
	message-body		messageBody	
	suppressible		suppressible	

Table 3-1 JPD Annotations

8.1 Annotation	Attribute	9.2 Annotation	Attribute	Comments
jpd:process		Process		Specifies settings for a business process.
	binding		binding	
	name		name	
	freezeOnFailure		freezeOnFailure	
	onSyncFailure		onSyncFailure	
	retryCount		retryCount	
	retryDelay		retryDelay	
	stateless		isStateless	
	process		process	
jpd:rosettnet		RosettaNet		Specifies the Rosettnet properties for a process.
	protocol-name		protocolName	
	protocol-version		protocolVersion	
	pip-name			
	pip-version		pipVersion	
	pip-role		pipRole	
jpd:selector		Selector		Precedes an XQuery definition in a business process (JPD) file.
	xquery		xquery	

Table 3-1 JPD Annotations

8.1 Annotation	Attribute	9.2 Annotation	Attribute	Comments
<code>jpd:transform</code>		Transform		Annotates a transformation control instance, which is instantiated automatically at run time.
<code>jpd:unexpected-message</code>		UnexpectedMessage		Specifies settings that allow a business process to ignore a message received before the process flow encounters the node at which the message is expected.
	action		action	
<code>jpd:version</code>		Version		Specifies how to invoke subprocesses when different versions of the parent process exist.
	strategy		strategy	
<code>jpd:xml-list</code>		XmlList		Annotates business process variable of Untyped XML - XmlObjectList.
<code>jpd:xquery</code>		Xquery		Precedes the global XQuery definitions in a JPD file.
	version		version	Represents the version of XQuery language specification.
	prologue		prologue	

Table 3-1 JPD Annotations

8.1 Annotation	Attribute	9.2 Annotation	Attribute	Comments
jpd:input-message		InputMessage		Validates the typed XBean parameter at run time.
	validate		validate	

Upgrading JCX or WebLogic Integration Control Files

After the upgrade to WebLogic Integration 9.2:

- All the WebLogic Integration Control files are renamed with a .java extension
- All the WebLogic Integration Control files with 8.x annotations are upgraded to JSR 175 based annotations.
- The Control Interfaces are annotated according to the Beehive standard with `@ControlExtension`.
- New attributes required for any control are added during the upgrade.
- New import statements are added to the existing import statements during the upgrade if required.

For example, if a WebLogic Integration 8.1 JCX contains the following annotation:

```
/**
 * @jc:task-create
 *   name="{name}"
 */
```

In WebLogic Integration 9.2 it is upgraded to:

```
@TaskCreate(name = "{name}",
taskTypeId.path = "/Worklist/Compatibility 8.1.x",
taskTypeId.version = 9.0f,
taskTypeId.worklistHostApplicationId = "worklist-ejbs-81x"
```

)

Other useful references are:

- [Upgrading Controls](#)
- [WebLogic Integration Annotations Reference](#)

Upgrading JCS Control Files

After the upgrade to WebLogic Integration 9.2, JCS control files are renamed with a `.java` extension. The JCS control files that contain WebLogic Integration control annotations are upgraded.

[Table 3-2](#) provides information on upgrades to WebLogic Integration 8.1 to 9.2 JSC annotations.

Table 3-2 JCS Annotations

8.1 Annotation	Attribute	9.2 Annotation	Attribute	Comments
common:xmlns		XmlNamespaces		All annotations will be child node of XMLNamespace.
	prefix	.Entry	prefix	
	namespace		namespace	
common:target-namespace		TargetNamespace		
	namespace	ce	value	

Table 3-2 JCS Annotations

8.1 Annotation	Attribute	9.2 Annotation	Attribute	Comments
common:security		Security		All annotations will be child nodes of schemas.
	roles-allowed		rolesAllowed	
	roles-referenced		rolesReference	
	run-as		runAs	
	run-as-principal		runAsPrincipal	
	single-principal		singlePrincipal	
	callback-roles-allowed		callbackRolesAllowed	
jcs:jc-jar		<none>		Primarily used in WebLogic Workshop 8.1. It is no longer used in WebLogic Workshop for Platform 9.2.
common:schema		Schemas.Entry		All annotations will be child nodes of schemas.
	file		file	
	inline		inline	
common:message-buffer		MessageBuffer		
	enable		enable	
	retry-count		retryCount	
	retry-delay		retryDelay	

Table 3-2 JCS Annotations

8.1 Annotation	Attribute	9.2 Annotation	Attribute	Comments
editor-info:c ode-gen				Primarily used in WebLogic Workshop 8.1. It is no longer used in WebLogic Workshop for Platform 9.2.
jcs:control-t ags		<none>		Primarily used in WebLogic Workshop 8.1. It is no longer used in WebLogic Workshop for Platform 9.2.
common:contro l		Control		The standard Beehive annotation.
common:operat ion		<none>		No longer needed, because Apache Beehive control framework handles it.
jcs:ide		<none>		Not handled as this annotation belong to WebLogic Workshop 8.1.x
jc:conversati on		Conversation		This annotation is identical to jws:conversation annotation
	phase		value	

Table 3-2 JCS Annotations

8.1 Annotation	Attribute	9.2 Annotation	Attribute	Comments
jcs:suppress-common-tags		<none>		Primarily used in WebLogic Workshop 8.1. It is no longer used in WebLogic Workshop for Platform 9.2.

Control Annotations

The following sections describe upgrades to WebLogic Integration Control annotations.

- [“Application View Controls” on page 4-2](#)
- [“Data Transformation Controls” on page 4-3](#)
- [“Email Controls” on page 4-5](#)
- [“File Controls” on page 4-7](#)
- [“HTTP Controls” on page 4-8](#)
- [“Message Broker Controls” on page 4-9](#)
- [“MQSeries Controls” on page 4-10](#)
- [“Process Controls” on page 4-13](#)
- [“Service Broker Controls” on page 4-15](#)
- [“Task Control Control-level Annotations” on page 4-16](#)
- [“Task Control Method-level Annotations” on page 4-20](#)
- [“Task Worker Control Control-level Annotation” on page 4-31](#)
- [“Task Worker Control Method-level Annotations” on page 4-31](#)
- [“Dynamic Transformation Controls” on page 4-39](#)
- [“WebLogic Integration JMS Controls” on page 4-40](#)

- [“TIBCO RV Controls” on page 4-42](#)

Application View Controls

[Table 4-1](#) provides information on upgrades to Application View Control annotations.

Table 4-1 Application View Controls

8.1 Annotation	Attribute	9.2 Annotation	Attribute	Comments
jc:av-identity		AppViewIdentity		Specifies the target Application View for an Application View control.
	name		name	
	App		appName	
	namespaceEnforcementEnabled		namespaceEnforcementEnabled	
jc:av-service		AppViewService		Specifies the Application View service associated with a method of an Application View control.
	name		name	
	async		async	

Data Transformation Controls

Table 4-2 provides information on upgrades to Data Transformation Control annotations.

Table 4-2 Data Transformation Controls

8.1 Annotation	Attribute	9.2 Annotation	Attribute	Comments
dtf:xquery		XQuery		Specifies the global XQuery functions and XQuery namespaces that can be used within the scope of the prologue of the DTF file.
	prologue		prolog	
	<none>		xqueryVersion	
dtf:transform		XQueryTransform XsltTransform		Specifies the XQuery and XSLT abstract methods in a DTF file.
	xquery-ref		transformType	XQueryTransform.TransformMethodType. xquery_ref
	xquery		transformType	XQueryTransform.TransformMethodType. xquery
	xslt-ref		transformType	XsltTransform.TransformMethodType. xslt_ref
	xslt		transformType	XsltTransform.TransformMethodType. xslt
	<none>		value	Value can be: xquery-ref or xquery or xslt-ref or xslt

Table 4-2 Data Transformation Controls

8.1 Annotation	Attribute	9.2 Annotation	Attribute	Comments
	<none>		xqueryVersion	Only when attributes xquery-ref or xquery are available.
dtf:schema-validate		SchemaValidate		Specifies if the source parameters or the return value, or both, should be schema validated.
	return-value		returnValue	
	parameters		parameters	
dtf:xquery-function		XQueryFunction		Specifies that a user-defined Java method (non-abstract) in a DTF file can be invoked from queries.
	<none>		xqueryVersion	

Email Controls

[Table 4-3](#) contains information on upgrades to Email Control annotations.

Table 4-3 Email Controls

8.1 Annotation	Attribute	9.2 Annotation	Attribute	Comments
jc:email		Email		Specifies configuration attributes for the Email control.
	from-address		fromAddress	
	from-name		fromName	
	smtp-address		smtpAddress	
	reply-to-address		replyToAddress	
	reply-to-name		replyToName	
	smtp-username		smtpUsername	
	smtp-password		smtpPassword	
	smtp-password-alias		smtpPasswordAlias	
	header-encoding		headerEncoding	

Table 4-3 Email Controls

8.1 Annotation	Attribute	9.2 Annotation	Attribute	Comments
jc:send-email		EmailControl. Send		Specifies configuration attributes for the Email control.
	to		to	
	cc		cc	
	bcc		bcc	
	subject		subject	
	body		body	
	content-type		contentType	
	attachments		attachments	

File Controls

Table 4-4 contains information on upgrades to File Control annotations.

Table 4-4 File Controls

8.1 Annotation	Attribute	9.2 Annotation	Attribute	Comments
jc:file		FileControl.F ileInfo		Specifies the annotations for the File control.
	directory-name		directoryName	
	file-mask		fileMask	
	suffix-name		suffixName	
	suffix-type		suffixType	
	create-mode		createMode	
	ftp-username-name		ftpUserName	
	ftp-password		ftpPassword	
	ftp-password-alias		ftpPasswordAlias	
	ftp-host-name		hostName	
	ftp-local-directory		ftpLocalDirectory	

Table 4-4 File Controls

8.1 Annotation	Attribute	9.2 Annotation	Attribute	Comments
jc:file-operation		FileControl.Operation	FileControl.IOOperation	Specifies configuration attributes for a File control.
	io-type		ioType	
	file-content		fileContent	
	record-size		recordSize	
	delimiter-string		delimiterString	
	delimiter-checkbox		delimiterCheckbox	
	encoding		encoding	

HTTP Controls

[Table 4-5](#) contains information on upgrades to HTTP Control annotations.

Table 4-5 HTTP Controls

8.1 Annotation	Attribute	9.2 Annotation	Attribute	Comments
jc:httpsend-data		HTTPSendData		Specifies the URL to which an HTTP message is to be sent, and from which a response is to be received.
	url-name		url	

Message Broker Controls

Table 4-6 contains upgrade information for Message Broker Control annotations.

Table 4-6 Message Broker Controls

8.1 Annotation	Attribute	9.2 Annotation	Attribute	Comments
jc:mb-publish-control		MessageBroker .ClassPublish	PublishControl	Defines class level attributes for the Publish control.
	channel-name		channelName	
	message-metadata		metadata	
jc:mb-publish-method		MessageBroker .MethodPublish		Defines method level attributes for the Publish control.
	message-metadata		metadata	
	message-body		body	
jc:mb-subscription-control		MessageBroker .ClassSubscription	SubscriptionControl	Defines class level attributes for the Subscription Control.
	channel-name		channelName	
	xquery		xquery	
	filter-value-match		classFilterValueMatch	
	<none>		xqueryVersion	Indicates the XQuery Version 2002 or 2004. Upgrade sets the version to 2002 by default.

Table 4-6 Message Broker Controls

8.1 Annotation	Attribute	9.2 Annotation	Attribute	Comments
jc:mb-subscription-method		MessageBroker.MethodSubscription		Defines method level attributes for the Subscription Control.
	filter-value-match		filterValueMatch	
jc:mb-subscription-callback		MessageBroker.SubscriptionCallback		Defines callback attributes for the Subscription Control.
	message-metadata		metadata	
	message-body		body	

MQSeries Controls

[Table 4-7](#) contains information on upgrades to MQSeries Control annotations.

Table 4-7 MQSeries Controls

8.1 Annotation	Attribute	9.2 Annotation	Attribute	Comments
jc:MQConnectionType		MQControl.Connection		Specifies the connection type for an MQ Series control.
	connectionType		type	
jc:MQConnectionPoolProps		MQControl.ConnectionPool		Specifies the MQ Series connection pool properties for the MQ Series control.
	mqPoolSize		poolSize	

Table 4-7 MQSeries Controls

8.1 Annotation	Attribute	9.2 Annotation	Attribute	Comments
jc:Connection PoolTimeout		MQControl.ConnectionPool		Specifies the MQ Series connection pool time-out in seconds.
	conTimeout		timeout	
jc:Connection RetrySettings		MQControl.ConnectionPool		Specifies the retry settings for the connection to the MQ Series queue manager.
	retryCount		retryCount	
	retryWaitTime InMilliseconds		retryWaitTimeIn Milliseconds	
jc:MQQueueManager		MQControl.Connection		Specifies the name of the queue manager for connection.
	queueManager		QueueManager	
jc:MQAuthorization		MQControl.Connection		Specifies the MQ Series authorization property for the MQ Series control.
	requireAuthorization		authorization	

Table 4-7 MQSeries Controls

8.1 Annotation	Attribute	9.2 Annotation	Attribute	Comments
jc:TCPSettings		MQControl.TCP Settings		Specifies the TCP connection settings for the MQ Series control.
	host		host	
	port		port	
	channel		channel	
	ccsid		ccsid	
	user		user	
	password		password	
	sendExit		sendExit	
	receiveExit		receiveExit	
	securityExit		securityExit	
jc:SSLSettings		MQControl.SSL Settings		Specifies the SSL settings for the MQ Series control.
	sslRequired		sslRequired	
	twoWaySSLRequired		twoWaySSLRequired	
jc:DefaultQueue		MQControl.Connection		Specifies the default queue name to be used for sending and retrieving messages.
	defaultQueueName		defaultQueueName	

Table 4-7 MQSeries Controls

8.1 Annotation	Attribute	9.2 Annotation	Attribute	Comments
jc:ImplicitTransaction		MQControl.Connection		Specifies the transaction mode of the MQ Series control.
	implicitTransactionRequired		implicitTransaction	

Process Controls

[Table 4-8](#) contains information on upgrades to Process Control annotations.

Table 4-8 Process Controls

8.1 Annotation	Attribute	9.2 Annotation	Attribute	Comments
common:message-buffer		MessageBuffer		Specifies that there should be a queue between the component's implementation code and the message transport wire for the specified method or callback.
	enable		enable	
	retry-count		retryCount	
	retry-delay		retryDelay	

Table 4-8 Process Controls

8.1 Annotation	Attribute	9.2 Annotation	Attribute	Comments
jc:conversation		Conversation		Specifies the role that a control's method or callback plays in a conversation. This annotation is identical to the jws:conversation annotation.
	phase		value	
jc:location		Location		Specifies the URL at which a Web service control accepts requests for each supported protocol. This annotation is identical to the corresponding web service annotation, @jws:location.
	uri		uri	
	http-url		httpUrl	
	jms-url		jmsUrl	

Service Broker Controls

Table 4-9 contains information on upgrades to Service Broker Control annotations.

Table 4-9 Service Broker Controls

8.1 Annotation	Attribute	9.2 Annotation	Attribute	Comments
common:define		Defines.Entry		Defines in-line data with the component class that might otherwise be referenced as an external file.
	name		name	
	value		value	
jc:conversati on		Conversation		Specifies the role that a control's method or callback plays in a conversation. This annotation is identical to the jws:conversati on annotation
	phase		phase	
jc:location		Location		Specifies the URL at which a web service control accepts requests for each supported protocol. This annotation is identical to the corresponding web service annotation, @jws:location.
	uri		uri	
	http-url		httpUrl	Converts to type String[]

Table 4-9 Service Broker Controls

8.1 Annotation	Attribute	9.2 Annotation	Attribute	Comments
	jms-url		<none>	This attribute is ignored.
jc:wsdl		Wsd1		Specifies a WSDL file that is implemented by a Web service.
	file		value	

Task Control Control-level Annotations

[Table 4-10](#) contains information on upgrades to Task Control Control-level annotations.

Table 4-10 Task Control Control-level Annotations

8.1 Annotation	Attribute	9.2 Annotation	Attribute	Comments
N/A		TaskControl.TaskPlanID		Should have @TaskType or @TaskCreate. If no jc:task annotation exists, create a @TaskPlanID annotation.
			path	Hardcoded to /Worklist/Compatibility WebLogic Integration 8.1.x
			version	Hardcoded to WebLogic Integration 9.0
			worklistHostApplicationId	Hardcoded to worklist-ejbs-81x

Table 4-10 Task Control Control-level Annotations

8.1 Annotation	Attribute	9.2 Annotation	Attribute	Comments
jc:task (control)		TaskAnnotations.TaskCreate		
	<NA>		taskPlanId	Hardcoded value
	Name		name	
	description		description	
	comment		comment	
	Priority		priority	
	Owner		owner	

Table 4-10 Task Control Control-level Annotations

8.1 Annotation	Attribute	9.2 Annotation	Attribute	Comments
jc:advanced (control)		TaskAnnotation.TaskCreate		
	can-be-reassigned		canBeReassigned	Compatible with WebLogic Integration 8.1.x only
	can-be-retained		canBeReturned	Compatible with WebLogic Integration 8.1.x only
	can-be-aborted		canBeAborted	Compatible with WebLogic Integration 8.1.x only
	claim-due-business-date		claimDueDate.businessTime.duration	Compatible with WebLogic Integration 8.1.x only
	claim-user-calendar		claimDueDate.businessTime.isUserCalendar = true and claimDueDate.businessTime.calendarName	Compatible with WebLogic Integration 8.1.x only
	Claim-calendar		claimDueDate.businessTime.isUserCalendar = false and claimDueDate.businessTime.calendarName	Compatible with WebLogic Integration 8.1.x only
	completion-due-business-date		completionDueDate.businessTime.duration	

Table 4-10 Task Control Control-level Annotations

8.1 Annotation	Attribute	9.2 Annotation	Attribute	Comments
	completion-user-calendar		completionDueDate. businessTime. isUserCalendar = true and completionDueDate. businessTime. calendarName	
	completion-calendar		completionDueDate. businessTime. isUserCalendar = false and completionDueDate. businessTime. calendarName	
jc:assignee (control)		TaskAnnotations.TaskCreate		
	User		assignmentInstructions81x. users	Comma-separated list converted to String[]
	Group		assignmentInstructions81x. groups	Comma-separated list converted to String[]
	algorithm		assignmentInstructions81x. algorithm	String converted to enum

Task Control Method-level Annotations

[Table 4-11](#) contains information on upgrades to Task Control Method-level annotations.

Table 4-11 Task Control Method-level Annotations

8.1 Annotation	Attribute	9.2 Annotation	Attribute	Comments
jc:task-create		TaskAnnotation.TaskCreate		
	Name		name	
	description		description	
	comment		comment	
	Priority		priority	
	owner		owner	
	can-be-reassigned		canBeReassigned	Compatible with WebLogic Integration 8.1.x only
	can-be-retained		canBeRetained	Compatible with WebLogic Integration 8.1.x only
	can-be-aborted		canBeAborted	Compatible with WebLogic Integration 8.1.x only
	claim-due-business-date		claimDueDate.businessTime.duration	Compatible with WebLogic Integration 8.1.x only
	claim-user-calendar		claimDueDate.businessTime.isUserCalendar = true and claimDueDate.businessTime.calendarName	Compatible with WebLogic Integration 8.1.x only

Table 4-11 Task Control Method-level Annotations

8.1 Annotation	Attribute	9.2 Annotation	Attribute	Comments
	claim-calendar		claimDueDate. businessTime. isUserCalendar = false and claimDueDate. businessTime. calendarName	Compatible with WebLogic Integration 8.1.x only
	completion-due-business-date		completionDueDate. businessTime.d uration	
	completion-user-calendar		completionDueDate. businessTime. isUserCalendar = true and completionDueDate. businessTime. calendarName	
	completion-calendar		completionDueDate. businessTime. isUserCalendar = false and completionDueDate. businessTime. calendarName	
	request	@TaskSetRequestResponse81x	value	

Table 4-11 Task Control Method-level Annotations

8.1 Annotation	Attribute	9.2 Annotation	Attribute	Comments
	request-mime-type	@TaskSetRequestResponse81x	mimeType	
jc:task-assignment		TaskAnnotations.TaskAssign81x		
	User		instructions81x. users	Comma-separated list converted to String[]
	group		instructions81x. groups	Comma-separated list converted to String[]
	algorithm		instructions81x. algorithm	String converted to enum
jc:task-abort		TaskAnnotations.TaskAbort		
	enabled		<none>	This attribute was ignored in the WebLogic Integration 8.1.x code.
jc:task-resume		TaskAnnotations.TaskResume		
	enabled		<none>	This attribute was ignored in the WebLogic Integration 8.1.x code.

Table 4-11 Task Control Method-level Annotations

8.1 Annotation	Attribute	9.2 Annotation	Attribute	Comments
jc:task-suspended		TaskAnnotations.TaskSuspended		
	enabled		<none>	This attribute was ignored in the WebLogic Integration 8.1.x code.
jc:task-get-response		TaskAnnotations.TaskGetResponse81x		
	enabled		<none>	This attribute was ignored in the WebLogic Integration 8.1.x code.
	<none>		property=Property.Response	
jc:task-get-request		TaskAnnotations.TaskGetRequest81x		
	enabled		<none>	This attribute was ignored in the WebLogic Integration 8.1.x code.
	<none>		property=Property.Request	
jc:task-get-property		TaskAnnotations.TaskGetProperties		
	name		propertyNames	

Table 4-11 Task Control Method-level Annotations

8.1 Annotation	Attribute	9.2 Annotation	Attribute	Comments
jc:task-set-property		TaskAnnotations.TaskSetProperty81x		
	name		name	
	value		value	
jc:task-remove-property		TaskAnnotations.TaskRemoveProperties81x		
	name		propertyName	

Table 4-11 Task Control Method-level Annotations

8.1 Annotation	Attribute	9.2 Annotation	Attribute	Comments
jc:task-update		TaskAnnotations.TaskUpdate81x		
	name		name	
	comment		comment	
	priority		priority	
	owner		owner	
	can-be-reassigned		canBeReassigned	Compatible with WebLogic Integration 8.1.x only
	can-be-retained		canBeReturned	Compatible with WebLogic Integration 8.1.x only
	can-be-aborted		canBeAborted	Compatible with WebLogic Integration 8.1.x only
	claim-due-business-date		claimDueDate.businessTime.duration	Compatible with WebLogic Integration 8.1.x only
	claim-user-calendar		claimDueDate.businessTime.isUserCalendar = true and claimDueDate.businessTime.calendarName	Compatible with WebLogic Integration 8.1.x only

Table 4-11 Task Control Method-level Annotations

8.1 Annotation	Attribute	9.2 Annotation	Attribute	Comments
	claim-calendar		claimDueDate. businessTime. isUserCalendar = false and claimDueDate. businessTime. calendarName	Compatible with WebLogic Integration 8.1.x only
	completion-due-business-date		completionDueDate. businessTime.d uration	
	completion-user-calendar		completionDueDate. businessTime. isUserCalendar = true and completionDueDate. businessTime. calendarName	
	completion-calendar		completionDueDate. businessTime. isUserCalendar = false and completionDueDate. businessTime. calendarName	
	request	@TaskSetRequestResponse81x (one per method for Request)		

Table 4-11 Task Control Method-level Annotations

8.1 Annotation	Attribute	9.2 Annotation	Attribute	Comments
			property = Property.Request	
			value = <request>	
request-mime-type	@TaskSetRequestResponse81x (one per method for Request)			
			property = Property.Request	
			contentType = <mime type>	
response	@TaskSetRequestResponse81x (one per method for Response)			
			property = Property.Response	
			value = <response>	
response-mime-type	@TaskSetRequestResponse81x (one per method for Response)			
			property = Property.Response	

Table 4-11 Task Control Method-level Annotations

8.1 Annotation	Attribute	9.2 Annotation	Attribute	Comments
			<code>mimeType =</code> <code><mime type></code>	

Table 4-11 Task Control Method-level Annotations

8.1 Annotation	Attribute	9.2 Annotation	Attribute	Comments
jc:task-event		TaskAnnotation.TaskEventAnnotation		
	name		name	
	comment		comment	
	priority		priority	
	owner		owner	
	can-be-reassigned		canBeReassigned	
	can-be-returned		canBeReturned	
	can-be-aborted		canBeAborted	
	claim-due-business-date		claimDueDate	
	completion-due-business-date		completionDueDate	
	claim-user-calendar		claimDueDate	
	claim-calendar		claimDueDate	
	completion-user-calendar		completionDueDate	
	completion-calendar		completionDueDate	
	request		request	
	request-mime-type		requestMimeType	

Table 4-11 Task Control Method-level Annotations

8.1 Annotation	Attribute	9.2 Annotation	Attribute	Comments
	response		response	
	response-mime-type		responseMimeType	
	completion-due-date		completionDueDate	
	claim-due-date		claimDueDate	

Task Worker Control Control-level Annotation

[Table 4-12](#) contains information on upgrades to the Task Worker Control Control-level annotation.

Table 4-12 Task Worker Control Control-level Annotation

8.1 Annotation	Attribute	9.2 Annotation	Attribute	Comments
jc:task-worker		<none>		You can ignore this annotation.

Task Worker Control Method-level Annotations

[Table 4-13](#) contains information on upgrades to Task Worker Control Method-level annotations.

Table 4-13 Task Worker Control Method-level Annotations

8.1 Annotation	Attribute	9.2 Annotation	Attribute	Comments
jc:select		TaskBatchAnnotations.TaskSelect		
	assigned-user		assignedUsers	
	assigned-group		assignedGroups	
	claimant		claimants	

Table 4-13 Task Worker Control Method-level Annotations

8.1 Annotation	Attribute	9.2 Annotation	Attribute	Comments
	task-id		taskIds	
	task-name		taskName	
	comment		comment	
	owner		owners	
	min-priority		minPriority	
	max-priority		maxPriority	
	states		states	Compatible with WebLogic Integration 8.1.x only
	completion-due-date-before		completionDueDateBefore	
	completion-due-date-after		completionDueDateAfter	
	claim-due-date-before		claimDueDateBefore	
	claim-due-date-after		claimDueDateAfter	
	creation-date-before		creationDateBefore	
	creation-date-after		creationDateAfter	
	property-name		propertyValue.name	
	property-value		propertyValue.value	

Table 4-13 Task Worker Control Method-level Annotations

8.1 Annotation	Attribute	9.2 Annotation	Attribute	Comments
	selector		selectorParamName	Removes enclosing brackets. For example, "{myPara}" becomes "myParam"
jc:task-create		TaskAnnotations.TaskCreate		
	name		name	
	description		description	
	comment		comment	
	priority		priority	
	owner		owner	
	can-be-reassigned		canBeReassigned	
	can-be-retained		canBeReturned	
	can-be-aborted		canBeAborted	
	claim-due-business-date		claimDueDate	
	completion-due-business-date		completionDueDate	
	request		request	
	request-mime-type		requestMimeType	
	claim-user-calendar		claimDueDate	
	claim-calendar		claimDueDate	

Table 4-13 Task Worker Control Method-level Annotations

8.1 Annotation	Attribute	9.2 Annotation	Attribute	Comments
	completion-use r-calendar		completionDueDa te	
	completion-cal endar		completionDueDa te	
	completion-due -date		completionDueDa te	
	claim-due-date		claimDueDate	
jc:task-assig n		TaskAnnotatio ns.TaskAssign 81x		
	user		user	
	group		group	
	algorithm		algorithm	
jc:task-claim		TaskAnnotatio ns.TaskClaim8 1x		
	enabled		<none>	This attribute was ignored inWebLogic Integration 8.x
	claimant		claimant	
jc:task-retur n		TaskAnnotatio ns.TaskReturn 81x		
	enabled		<none>	This attribute was ignored in WebLogic Integration 8.x

Table 4-13 Task Worker Control Method-level Annotations

8.1 Annotation	Attribute	9.2 Annotation	Attribute	Comments
jc:task-start		TaskAnnotations.TaskStart81x		
	enabled		<none>	This attribute was ignored in WebLogic Integration 8.x
jc:task-stop		TaskAnnotations.TaskStop81x		
	enabled		<none>	This attribute was ignored in WebLogic Integration 8.x
jc:task-complete		TaskAnnotations.TaskComplete		
	enabled		<none>	This attribute was ignored in WebLogic Integration 8.x
jc:task-abort		TaskAnnotations.TaskAbort		
jc:task-delete		TaskAnnotations.TaskDelete		
	enabled		<none>	This attribute was ignored in WebLogic Integration 8.x
jc:task-resume		TaskAnnotations.TaskResume		

Table 4-13 Task Worker Control Method-level Annotations

8.1 Annotation	Attribute	9.2 Annotation	Attribute	Comments
jc:task-suspend		TaskAnnotations.TaskSuspended		
jc:task-get-info		TaskAnnotations.TaskGetInfo		
jc:task-get-response		TaskAnnotations.TaskGetResponse81x		
jc:task-get-request		TaskAnnotations.TaskGetRequest81x		
jc:task-get-property-name		TaskAnnotations.TaskGetPropertyNames81x		
	enabled		<none>	This attribute was ignored in WebLogic Integration 8.x
jc:task-get-property		TaskAnnotations.TaskGetProperties		
jc:task-set-property		TaskAnnotations.TaskSetProperty81x		
	name		name	
	value		value	
jc:task-remove-property		TaskAnnotations.TaskRemoveProperties81x		
	name		propertyName	

Table 4-13 Task Worker Control Method-level Annotations

8.1 Annotation	Attribute	9.2 Annotation	Attribute	Comments
jc:task-update		TaskAnnotations.TaskUpdate81x		
	name		name	
	comment		comment	
	priority		priority	
	owner		owner	
	can-be-reassigned		canBeReassigned	
	can-be-retained		canBeReturned	
	can-be-aborted		canBeAborted	

Table 4-13 Task Worker Control Method-level Annotations

8.1 Annotation	Attribute	9.2 Annotation	Attribute	Comments
	claim-due-business-date		claimDueDate	
	completion-due-business-date		completionDueDate	
	claim-user-calendar		claimDueDate	
	claim-calendar		claimDueDate	
	completion-user-calendar		completionDueDate	
	completion-calendar		completionDueDate	
	request		request	
	request-mime-type		requestMimeType	
	response		response	
	response-mime-type		responseMimeType	
	completion-due-date		completionDueDate	
	claim-due-date		claimDueDate	

Dynamic Transformation Controls

The following table contains information on upgrades to Dynamic Transformation Control annotations.

Table 4-14 Dynamic Transformation Control Annotations

8.1 Annotation	Attribute	9.2 Annotation	Attribute	Comments
jc:ddtf		Ddtf		Specifies the XQuery functions that can be used by the queries and the type of encoding used at design time.
	xquery-prologue		xqueryPrologue	
	control-design-time-encoding		controlDesignTimeEncoding	
jc:xquery		XQuery		Specifies the XQuery files and their attributes for XQuery transformations at run time.
	xquery-arg-names		xqueryArgNames	
	validate-params		validateParams	
	validate-return		validateReturn	
	design-time-encoding		designTimeEncoding	

Table 4-14 Dynamic Transformation Control Annotations

8.1 Annotation	Attribute	9.2 Annotation	Attribute	Comments
jc:xslt		Xslt		Specifies the XSL file to be used for the transformation.
	xslt-arg-names		xsltArgNames	

WebLogic Integration JMS Controls

The WebLogic Integration JMS control is an extension of the base JMS control, and its control annotations also apply to the WebLogic Integration JMS control.

Note: The base JMS control no longer supports JMS receive functions. Therefore, WebLogic Integration JMS controls do not have receive functions.

[Table 4-15](#) contains information on upgrades to WebLogic Integration JMS Control annotations.

Table 4-15 WebLogic Integration JMS Control Annotations

8.1 Annotation	Attribute	9.2 Annotation	Attribute	Comments
jc:jms		JMSControl.JMS		Sets the JMS properties for the control
	receive-correlation-property		receivecorrelationproperty	
	send-correlation-property		sendcorrelationproperty	
	auto-topic-subscribe		autotopicsubscribe	
	receive-selector		receiveselector	
	topic-table-data-source		topictabledatasource	

Table 4-15 WebLogic Integration JMS Control Annotations

8.1 Annotation	Attribute	9.2 Annotation	Attribute	Comments
	send-jndi-name		sendjndiname	
	receive-jndi-name		receivejndiname	
	connection-factory-jndi-name		connectionfactoryjndiname	
	receive-type		receivetype	
	send-type		sendtype	
jc:jms-headers		JMSHeader		Set and retrieves values for the JMS message headers.
	JMSCorrelationID		JMSCorrelationID	
	JMSDeliveryMode		JMSDeliveryMode	
	JMSExpiration		JMSExpiration	
	JMSMessageID		JMSMessageID	
	JMSPriority		JMSPriority	
	JMSRedelivered		JMSRedelivered	
	JMSTimestamp		JMSTimestamp	
	JMSType		JMSType	
jc:jms-property		JMSControl.PropertyValue		Sets and retrieves properties of the message.
	key		name	
	value		value	

TIBCO RV Controls

[Table 4-16](#) contains information on upgrades to TIBCO RV Control annotations.

Table 4-16 TIBCO RV Control Annotations

8.1 Annotation	Attribute	9.2 Annotation	Attribute	Comments
jc:Transport		TibcoRV.Transport		
	service		service	
	network		network	
	daemon		daemon	
jc:UseCM		TibcoRV.UseCM		
	usecm		usecm	
jc:CMTransport		TibcoRV.CMTransport		
	cmname		cmname	
	ledgername		ledgername	
	requestold		requestold	
	syncledger		syncledger	

Other Component Changes

This section provides WebLogic Integration 8.1 to 9.2 upgrade information for the following components:

- [Control Factories](#)
- [XQuery Files](#)
- [JPD and Control Callbacks](#)
- [JPD Process Language](#)
- [DTF Transformation](#)
- [Channel Files](#)

Control Factories

The WebLogic Integration upgrades only the WebLogic Integration controls used as a control factory. WebLogic Integration upgrader needs to override the BEA Workshop for WebLogic Platform upgrader behavior if a JPD uses a control factory instead of a BEA Workshop WebLogic Platform control factory.

In the case of timer controls, there is no JCX, but the upgrader creates a `TimerControlFactory` class in the same package as the JPD.

The Eclipse IDE supports the creation of control factories to be used from a JPD. To use the control factory with BEA Workshop for WebLogic Platform:

1. Add the `@ControlFactory` annotation in the JPD.

2. Add `create()` method in the JCX.

Note: To use a timer control factory in the JPD, add the `@ControlFactory` annotation, and create the `TimerControlFactory` class in the same package as the JPD.

XQuery Files

WebLogic Integration upgrades XQuery files through the upgrade of DTF files. The DTF file contains references to XQuery files that are upgraded along with the DTF file. When the XQuery file is upgraded, WebLogic Integration includes a comment, at the top of the file, that indicates that the file belongs to version 2002.

For example, an XQuery file before the upgrade contains the following:

```
{-- Project3/SwitchAssignTransformation.dtf#forAssign2Copy01 --}
xs:boolean( 'false' )
```

The XQuery file after the upgrade contains the following:

```
{-- Project3/SwitchAssignTransformation.dtf#forAssign2Copy01 --}
{-- version=2002 --}
xs:boolean( 'false' )
```

Note: WebLogic Integration displays a warning message in case you select an upgrade action on an XQuery file. This warning message states that the file cannot be upgraded.

Caution: The Xquery within the XQuery files are not upgraded to version 2004: they remain in the version of the original file before the upgrade.

JPD and Control Callbacks

WebLogic Integration upgrades control declarations using `@Control` according to the Apache Beehive standard. The JPD callback field is annotated with `@Callback`. The callback interface is annotated with `@CallbackInterface`. The `Callback` interface declaration remains a part of the JPD definition and extends the `ServiceBrokerControl`.

According to the Apache Beehive standards, WebLogic Integration also annotates control callback handler methods using `@EventHandler()`.

All the methods in the process definition that are referenced from the `<controlReceive\> XML` snippet are annotated during the upgrade with the `@EventHandler` annotation.

Note: Control callbacks can be sent to a JPD only by using `ControlHandle.sendEvent`.

For example, add the following code to the `MyCustomControlImpl.java` file after upgrade:

```
System.out.println("Before sending event to jpd in
MyCustomControlImpl event
handler");

ControlHandle controlHandle = context.getControlHandle();

try {
    Method m =
MyCustomControl.Callback.class.getMethod("response",
XmlObject.class);

    EventRef event = new EventRef(m);
    controlHandle.sendEvent( event, new Object[]{payload});
}

catch(Exception e) {
    e.printStackTrace();
}
```

JPD Process Language

In WebLogic Integration 8.x applications, the entire process language was specified using `@jpd:process`. However, for WebLogic Integration 9.2 the process language is upgraded to `@com.bea.wli.jpd`. The Process annotation has a `process` attribute that contains the entire process language string.

DTF Transformation

When the DTF files are upgraded, they are re-named with a `.java` extension. All the DTF files in WebLogic Integration 8.1 annotations are upgraded to JSR-175 based annotations. All the controls are converted to Apache Beehive controls.

The DTF files in WebLogic Integration 8.1 have similar functions as other WebLogic Integration controls, but they are abstract classes unlike other controls, which are interfaces. The DTF class contain metadata-specified methods, and fully-coded methods that are specified by actual Java method bodies that are called by the XQuery engine.

DTF annotations that contained `xquery` and `xquery-ref` attributes indicating XQuery version 2002 have a new `xqueryVersion` attribute in WebLogic Integration 9.2.

WebLogic Integration 9.2 upgrades all import statements and adds new import statements where required. For example, a WebLogic Integration 8.1 DTF file that contains an annotation is as follows:

```
/**
 * @dtf:transform xquery-ref="switchXqAssign2defaultAssign_1Copy01.xq"
 */
```

When this DTF file is upgraded to WebLogic Integration 9.2, it is as follows:

```
@XQueryTransform(value = "switchXqAssign2defaultAssign_1Copy01.xq",
transformType = XQueryTransform.TransformMethodType.xquery_ref,
@com.bea.wli.common.XQuery(version =
com.bea.wli.common.XQuery.Version.v2002)
```

Channel Files

Channel files do not get upgraded during the upgrade process. They are moved into the Utility projects in Eclipse.