BEA WebLogic Integration™

Tutorial: Upgrading 8.1 Application Source to 9.2

Version 9.2
Document Date: November 2006

Copyright

Restricted Rights Legend

Trademarks and Service Marks

Contents

Tutorial: Overview

Tutorial Objective and Scope 1-1

Prerequisites 1-2

About the Upgrade Import Wizard 1-2

Creating the Source 8.1 Application 1-3

Tutorial: Upgrading the 8.1 Application Source

Application Overview 2-1

Importing the Application Source 2-2

Validating the Upgraded Application 2-11

Deploying and Publishing the Application 2-11

Creating a WebLogic Integration Domain Using the Configuration Wizard . . 2-14

Running the Application 2-19

Summarize the WebLogic Integration Upgrade Process 2-24

Tutorial: Overview

This tutorial provides specific information on how to upgrade a BEA WebLogic Integration™ 8.1 (Service Pack 4 or 5) application, to the BEA WebLogic Integration 9.2 environment. This tutorial describes the upgrade scenario using a Process Application sample created in BEA WebLogic Platform™ 8.1.

WebLogic Integration 9.2 is a part of BEA Workshop for WebLogic Platform™ 9.2. This chapter includes the following sections:

- Tutorial Objective and Scope
- Prerequisites
- About the Upgrade Import Wizard
- Creating the Source 8.1 Application

Tutorial Objective and Scope

You are required to have in depth knowledge of WebLogic Platform, in particular, WebLogic Integration 8.1. The objective of this tutorial is to help you understand and apply the upgrade process to existing applications, when upgrading to WebLogic Integration 9.2. The upgrade process does not change or alter the source application logic and intent in any manner. The process simply upgrades the application for use in the Workshop for WebLogic Platform workspace.

To meet this objective, the steps involved can be grouped as follows:

- Creating the Application in 8.1

  Create the Application in 8.1 and replace the deprecated APIs for smooth upgrade to 9.2.

- Importing the Application into the 9.2 environment using the Import Wizard

  Import the Application including all its projects using the wizard.

- Deploying the Application in 9.2

  Involves creating a server with WebLogic Integration domain, deploying and publishing the Application.

- Running the Application to validate the upgrade process

  Execute the Application using test values, and confirm validity of the upgraded Application by successfully executing a process.

Prerequisites

This tutorial has several dependencies for its smooth execution. For example, you require an application from the 8.1 release as the source for the upgrade exercise. Following are some of the requirements for successful execution of the tutorial:

- Ensure that the application to be upgraded has been built on (or upgraded to) 8.1 Service Pack 4, 5.
- Ensure that you have access to the 8.1 Service Pack 4 or 5 installation as you will be using the Process Application tutorial sample created in that environment.

About the Upgrade Import Wizard

The upgrade process in 9.2 is handled by an import wizard. 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.

- Converting 8.1 project types to 9.2 project types.

- Converting 8.1 Javadoc annotations to JSR 175 compliant annotations.
- Moving the shared libraries from the 8.1 application directory, to the J2EE EAR project that is part of the upgraded application.
- Moving JSP files into a WebContent directory in the 9.2 environment.
- Upgrading NetUI JSP tags to Apache Beehive JSP tags, or just updating existing NetUI JSP tags to the 9.2 compatible NetUI version JSP tags.
- Moving XSD files that are in a 8.1 Schema project into a 9.2 Utility project.
- Moving Java packages and sources into src folders.

Creating the Source 8.1 Application

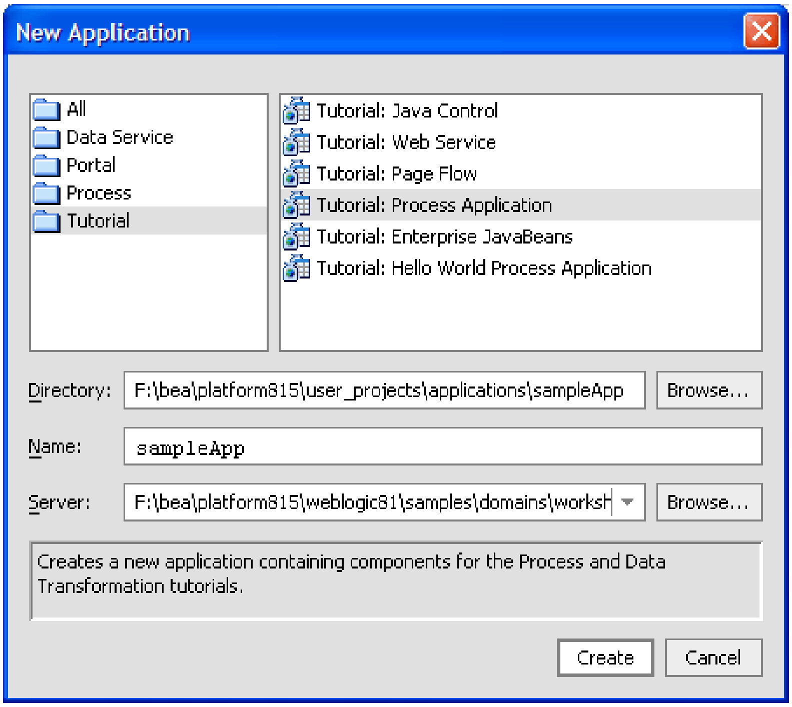
For the purpose of this tutorial, you are going to create a Request Quote application in the WebLogic Platform 8.1 Service Pack 5 release. The primary goal of this section is to create an Application in 8.1 and to replace some deprecated APIs for error-free import into the 9.2 release.

Note: Applications created using Service Pack 4 or 6 can also be upgraded using this tutorial.

Perform the following steps to create the new application in 8.1.

1. In WebLogic Workshop 8.1 select the **File** → **New** → **Application...** menu option. This will display the New Application dialog box.
2. In the left pane select **Tutorial** and in the right pane, select **Tutorial: Process Application**.
3. In the **Name:** field, provide the name **`sampleApp`**. The New Application dialog box should appear as displayed in Figure 1-1.

Figure 1-1 Creating a New Application in WebLogic Platform 8.1



Directory: F:\bea\platform815\user_projects\applications\sampleApp

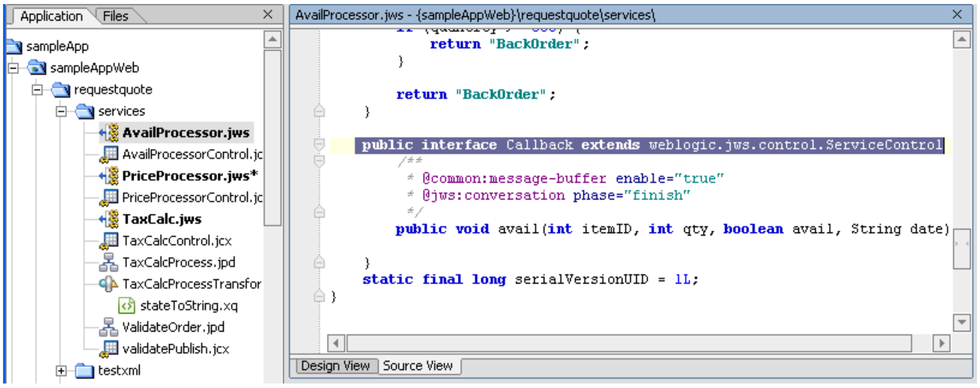
Name: sampleApp

Server: F:\bea\platform815\weblogic81\samples\domains\worksh

Creates a new application containing components for the Process and Data Transformation tutorials.

4. Click **Create** to proceed.
5. In the Application pane of WebLogic Workshop, navigate to view the **sampleApp** → **sampleAppWeb** → **requestquote** → **services** → **AvailProcessor.jws** web service file.
6. Double-click the file to view its **Source View** in the adjacent pane, as shown in Figure 1-2.

Figure 1-2 Viewing the Web Service Source



```
            return "BackOrder";
        }

        return "BackOrder";
    }

    public interface Callback extends weblogic.jws.control.ServiceControl
    {
        /**
         * @common:message-buffer enable="true"
         * @jws:conversation phase="finish"
         */
        public void avail(int itemID, int qty, boolean avail, String date)
    }
    static final long serialVersionUID = 1L;
}
```

7. In the highlighted text (`public interface Callback extends weblogic.jws.control.ServiceControl`) of the source view as shown in Figure 1-2, replace **`weblogic.jws.control.ServiceControl`** with **`com.bea.control.Control`**.

8. At the beginning of the source file, replace the `import` **`weblogic.jws.control.JwsContext`** with `import` **`com.bea.control.JwsContext`**.

   Note: The `weblogic.jws.control.ServiceControl` and the `weblogic.jws.control.JwsContext` are deprecated APIs that need to be replaced in all the web services for an error free upgrade to 9.2.

9. Perform step 5 to step 9 on the remaining two web services, **`PriceProcessor.jws`** and **`TaxCalc.jws`**.

10. Save and Close the application.

Your application, `sampleApp` is now ready for import using the Workshop for WebLogic Platform 9.2 IDE.

CHAPTER 2

Tutorial: Upgrading the 8.1 Application Source

This chapter contains information on how to upgrade the application source using the Import Wizard. It also gives an insight into the upgrade process. It contains the following sections:

- Application Overview
- Importing the Application Source
- Validating the Upgraded Application
- Summarize the WebLogic Integration Upgrade Process

This tutorial does not include validation of all the aspects of the sample application. It helps you upgrade the 8.1 application source and view it in the WebLogic Integration 9.2 environment.

Application Overview

This application used in this tutorial creates a business process that meets the following requirements:

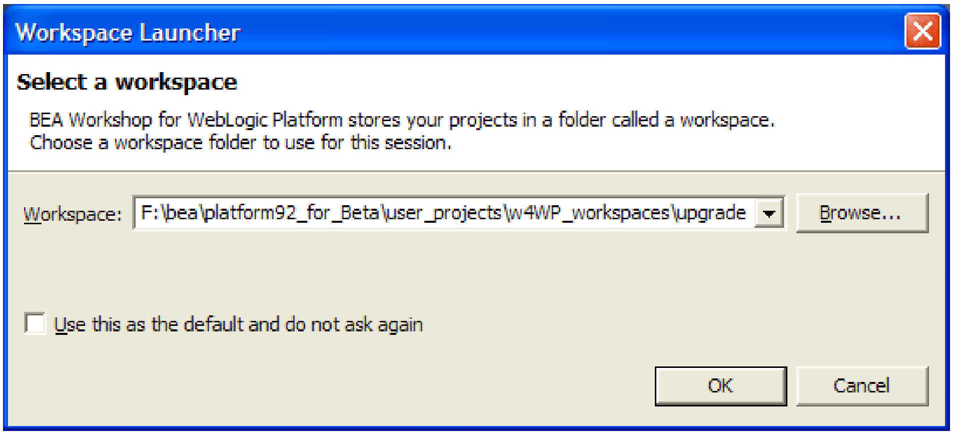
- Receives Request for Quote messages from clients
- Starts the business process on receipt of the Request for Quote
- Validates and processes the request
- Sends the status of the Request for Quote to the client

Importing the Application Source

This section provides detailed step-by-step instructions on how to upgrade your WebLogic Integration 8.1 Service Pack 4 or 5 application source for use in the Workshop for WebLogic Platform 9.2 environment.

1. Start Workshop for WebLogic Platform by selecting **Start** → **All Programs** → **BEA Products** → **Workshop for WebLogic Platform 9.2** from the Start menu. This will display the Workspace Launcher dialog box as shown in Figure 2-1.

Figure 2-1 Setting the Workspace for the Upgraded Application



Workspace Launcher

Select a workspace

BEA Workshop for WebLogic Platform stores your projects in a folder called a workspace.
Choose a workspace folder to use for this session.

Workspace: F:\bea\platform92_for_Beta\user_projects\w4WP_workspaces\upgrade

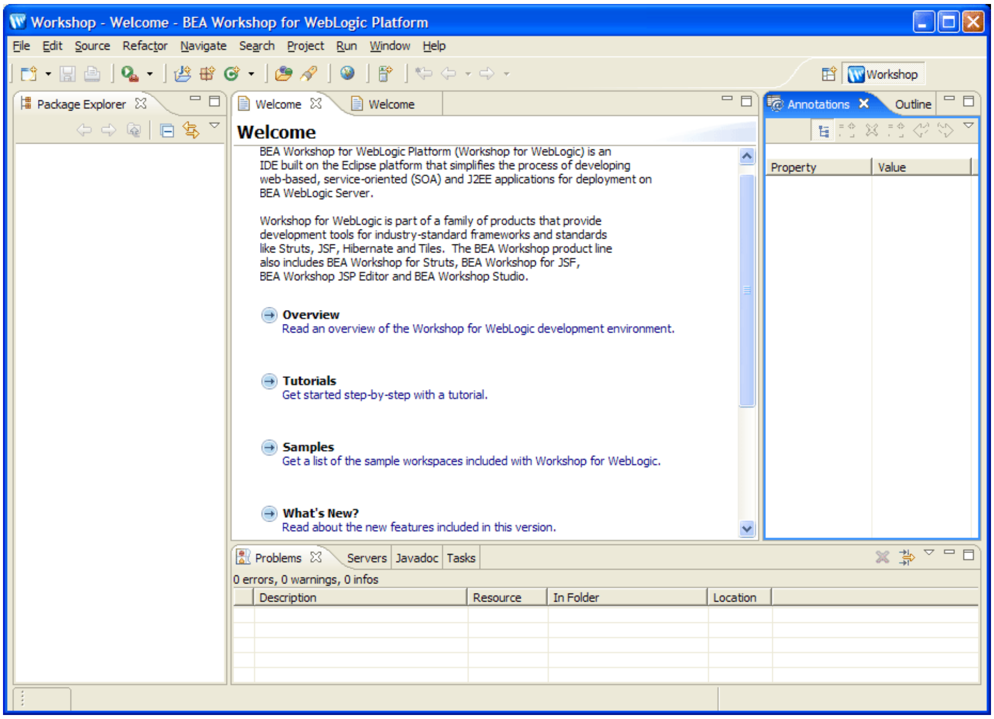
2. Specify the desired location for the upgraded application in the **Workspace** field as shown in Figure 2-1. For this tutorial use the `upgrade` workspace, under the WebLogic Platform installation directory, as shown below:

   $BEA_HOME`\user_projects\w4WP_workspaces\upgrade`

Note: The *$BEA_HOME* used throughout this tutorial is `F:\bea\platform92_for_Beta\` where WebLogic Platform 9.2 is installed.

3. Click **OK** to proceed, and the BEA Workshop for WebLogic Platform IDE is launched, as shown in Figure 2-2.

Figure 2-2 Workshop for WebLogic Platform (IDE)


Workshop - Welcome - BEA Workshop for WebLogic Platform
File Edit Source Refactor Navigate Search Project Run Window Help
Workshop
Package Explorer
Welcome
Welcome
Annotations
Outline
Property
Value
Problems
Servers
Javadoc
Tasks

Welcome

BEA Workshop for WebLogic Platform (Workshop for WebLogic) is an IDE built on the Eclipse platform that simplifies the process of developing web-based, service-oriented (SOA) and J2EE applications for deployment on BEA WebLogic Server.

Workshop for WebLogic is part of a family of products that provide development tools for industry-standard frameworks and standards like Struts, JSF, Hibernate and Tiles. The BEA Workshop product line also includes BEA Workshop for Struts, BEA Workshop for JSF, BEA Workshop JSP Editor and BEA Workshop Studio.

Overview
Read an overview of the Workshop for WebLogic development environment.

Tutorials
Get started step-by-step with a tutorial.

Samples
Get a list of the sample workspaces included with Workshop for WebLogic.

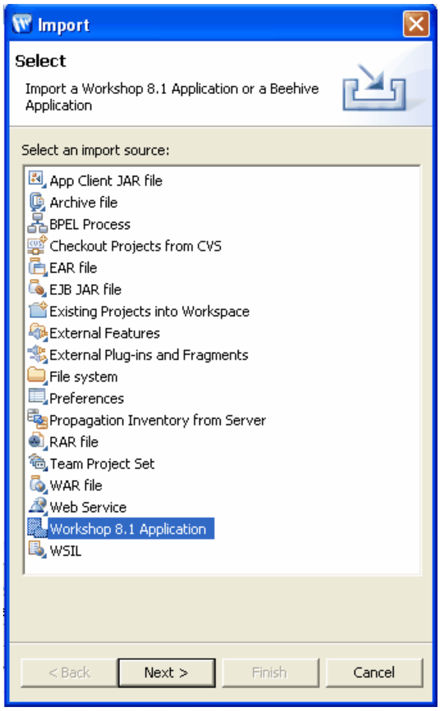
What's New?
Read about the new features included in this version.

0 errors, 0 warnings, 0 infos

Description	Resource	In Folder	Location

4. Start the import procedure by selecting the **File** → **Import...** menu option. This will display the Import Source dialog box as shown in Figure 2-3.

Figure 2-3 Selecting the Application Source to be Imported



Import

Select

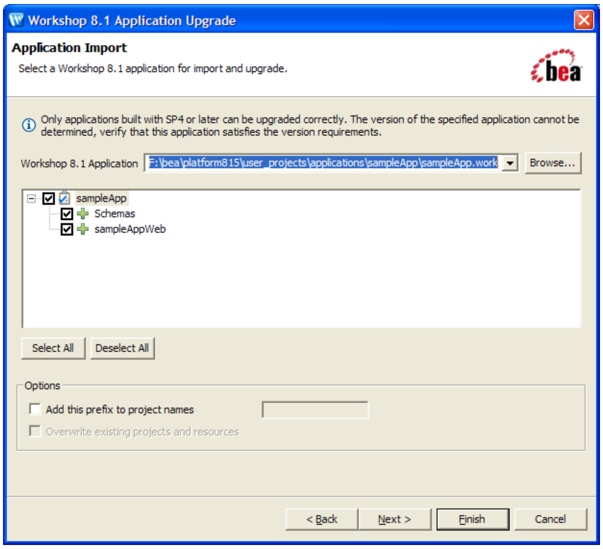
Import a Workshop 8.1 Application or a Beehive Application

Select an import source:

- App Client JAR file
- Archive file
- BPEL Process
- Checkout Projects from CVS
- EAR file
- EJB JAR file
- Existing Projects into Workspace
- External Features
- External Plug-ins and Fragments
- File system
- Preferences
- Propagation Inventory from Server
- RAR file
- Team Project Set
- WAR file
- Web Service
- Workshop 8.1 Application
- WSIL

5. Select the **Workshop 8.1 Application** option as the application source type, as shown in Figure 2-3, and click **Next** to proceed. The Application Import dialog box appears as shown in Figure 2-4.

Figure 2-4 Selecting the Application to be Imported


Workshop 8.1 Application Upgrade
Application Import
Select a Workshop 8.1 application for import and upgrade.
bea
Only applications built with SP4 or later can be upgraded correctly. The version of the specified application cannot be determined, verify that this application satisfies the version requirements.
Workshop 8.1 Application
F:\bea\platform815\user_projects\applications\sampleApp\sampleApp.work
Browse...
sampleApp
Schemas
sampleAppWeb
Select All
Deselect All
Options
Add this prefix to project names
Overwrite existing projects and resources
< Back
Next >
Finish
Cancel

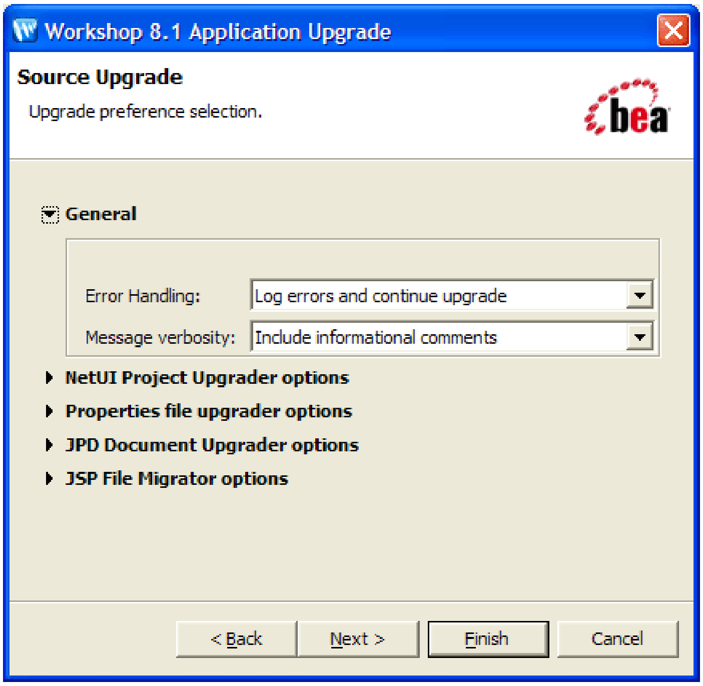
6. Click **Browse** to navigate through the directory structure and select the **`sampleApp.work`** file that you created in 8.1 SP5. For more information, see Creating the Source 8.1 Application. As shown in Figure 2-4, the **`sampleApp.work`** file for this tutorial is located in:

   ```
   F:\bea\platform815\user_projects\applications\sampleApp
   ```

   After selecting the sample application from the 8.1 install, a list of projects in that application are displayed in the dialog box. You have the option of choosing the projects to import and upgrade by clicking on the check boxes adjacent to them. However, it is advisable to include all the projects, as most of them have inter-dependencies.

7. Click **Next** to proceed and the Upgrade preference selection dialog box is displayed, as shown in Figure 2-5.

Figure 2-5 Setting the Upgrade Preferences


Workshop 8.1 Application Upgrade
Source Upgrade
Upgrade preference selection.
General
NetUI Project Upgrader options
Properties file upgrader options
JPD Document Upgrader options
JSP File Migrator options
< Back
Next >
Finish
Cancel

Error Handling: Log errors and continue upgrade

Message verbosity: Include informational comments

You can configure your preference by selecting the various options displayed in the dialog box. These options are available under specific categories, as listed below, depending on their function.

– **General** – You can set the error handling options and the kind of content you would like to record. Use the default value for both the properties as described in Table 2-1.

Table 2-1 Setting General Preferences

Property	Description
Error Handling	Determines how the errors are handled and the subsequent action you would like to implement. Error messages are always recorded in a log file, but your selection here determines how they are delivered to you. There are three options for this field, namely: • Log errors and continue upgrade (default option) • Log errors but abort upgrade • Display dialog on errors
Message Verbosity	Determines the type of messages that will be captured during the upgrade process. There are three types of messages that are captured: information, warning, and error. These three message types are reflected in the three options for this field, namely: • Include informational comments (default option) – All three message types • Include warning comments – Only warning and error type messages • Include error comments – Only error messages

– **NetUI Project Upgrader options** – Selecting the **Use WebLogic J2EE Shared Libraries** option enables you to use the WebLogic J2EE shared libraries, without having to duplicate runtime `.jar` files across projects. You can ignore this option as it usually does not impact the WebLogic Integration applications. By default, this option is selected for upgrade.

– **Properties file upgrader options** – The **Delete copied resource bundle files from the web content folder** option removes any unnecessary `.properties` files. If none of the JSP pages in the web content folder require `.properties` files, they can be deleted. You can ignore this option as it usually does not impact the WebLogic Integration applications. By default, this option is not selected for upgrade.

– **JPD Document Upgrader options** – Select the **Upgrade XQ2002 to XQ2004** option as it converts XQuery statements from XQuery 2002 draft version to XQuery 2004 draft version. This enables the JPD node editor to parse through the inline XQuery statements and convert them to 2004 version. By default, this option is not selected for upgrade.

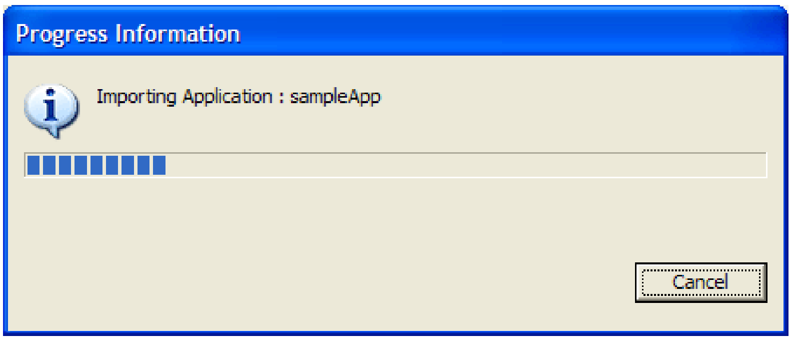
- Ensure you select the **Upgrade XQ2002 to XQ2004** option.

- **JSP File Migrator options** – When selected, the **Replace BEA NetUI tags with Apache Beehive tags** option will upgrade the WebLogic Integration 8.1 supported NetUI tags with Apache Beehive compatible NetUI tags. You can ignore this option as it usually does not impact the WebLogic Integration applications. By default, this option is not selected for upgrade.

Note: You can convert projects or selective JSPs to use Apache Beehive compatible NetUI tags even after the upgrade exercise. However, all new projects created in 9.2 use the Apache Beehive compatible NetUI tags.

8. Click **Next** to proceed with the upgrade, and the Progress Information dialog box is displayed, as shown in Figure 2-6. At this step, the application being upgraded is assessed and a detailed list of upgrade tasks that need to be performed (and not performed) is generated.

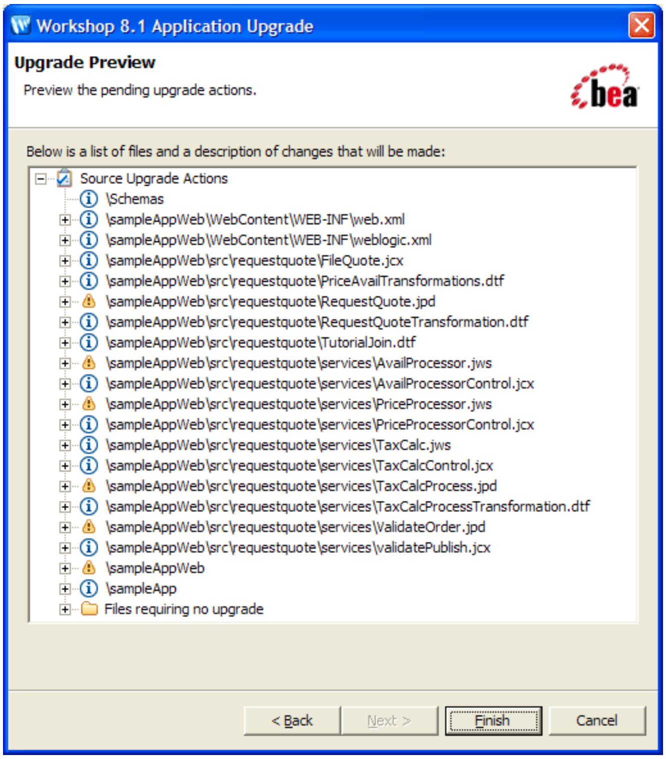
Figure 2-6 Tracking Upgrade Progress



Importing Application : sampleApp

After completing the upgrade process, the **Upgrade Preview** dialog box is displayed, as shown in Figure 2-7. The preview dialog box provides a summary of changes that will be implemented during the upgrade task.

Figure 2-7 Preview of the Upgrade Actions

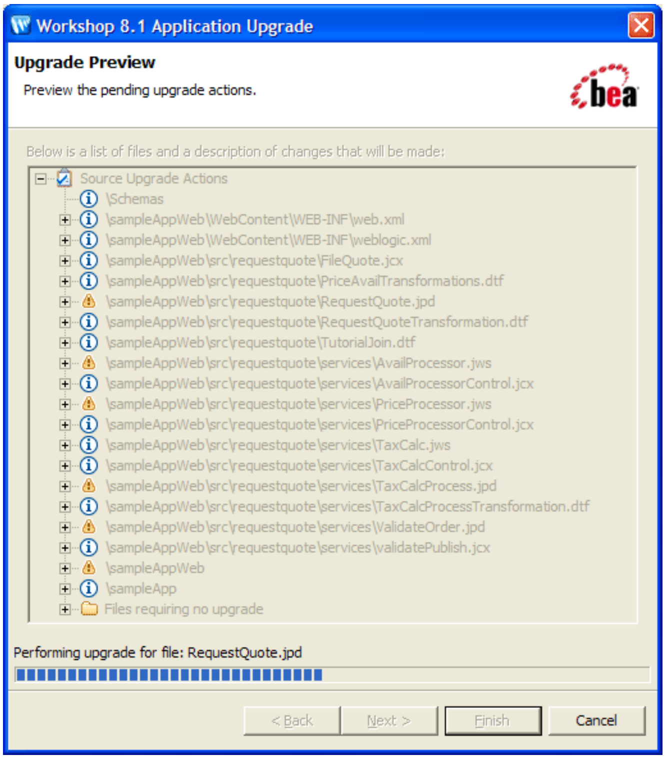

Workshop 8.1 Application Upgrade
Upgrade Preview
Preview the pending upgrade actions.
bea

Below is a list of files and a description of changes that will be made:

```
Source Upgrade Actions
  \Schemas
  \sampleAppWeb\WebContent\WEB-INF\web.xml
  \sampleAppWeb\WebContent\WEB-INF\weblogic.xml
  \sampleAppWeb\src\requestquote\FileQuote.jcx
  \sampleAppWeb\src\requestquote\PriceAvailTransformations.dtf
  \sampleAppWeb\src\requestquote\RequestQuote.jpd
  \sampleAppWeb\src\requestquote\RequestQuoteTransformation.dtf
  \sampleAppWeb\src\requestquote\TutorialJoin.dtf
  \sampleAppWeb\src\requestquote\services\AvailProcessor.jws
  \sampleAppWeb\src\requestquote\services\AvailProcessorControl.jcx
  \sampleAppWeb\src\requestquote\services\PriceProcessor.jws
  \sampleAppWeb\src\requestquote\services\PriceProcessorControl.jcx
  \sampleAppWeb\src\requestquote\services\TaxCalc.jws
  \sampleAppWeb\src\requestquote\services\TaxCalcControl.jcx
  \sampleAppWeb\src\requestquote\services\TaxCalcProcess.jpd
  \sampleAppWeb\src\requestquote\services\TaxCalcProcessTransformation.dtf
  \sampleAppWeb\src\requestquote\services\ValidateOrder.jpd
  \sampleAppWeb\src\requestquote\services\validatePublish.jcx
  \sampleAppWeb
  \sampleApp
  Files requiring no upgrade
```

9. Click **Finish** when you are satisfied with the items listed for upgrade, and those that do not require to be upgraded. The **Upgrade Preview** dialog box is refreshed and displays the upgrade progress status at the bottom of the dialog box, as shown in Figure 2-8.

Figure 2-8 Upgrade Progress Indicator



Workshop 8.1 Application Upgrade

Upgrade Preview

Preview the pending upgrade actions.

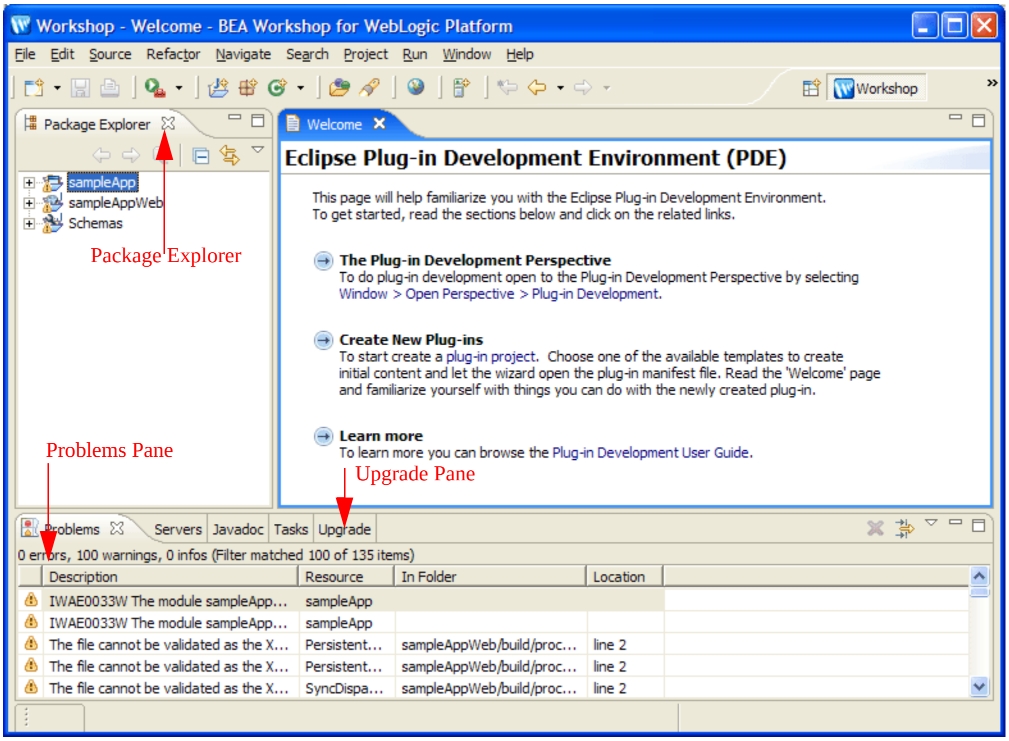
Below is a list of files and a description of changes that will be made:

```
Source Upgrade Actions
  \Schemas
  \sampleAppWeb\WebContent\WEB-INF\web.xml
  \sampleAppWeb\WebContent\WEB-INF\weblogic.xml
  \sampleAppWeb\src\requestquote\FileQuote.jcx
  \sampleAppWeb\src\requestquote\PriceAvailTransformations.dtf
  \sampleAppWeb\src\requestquote\RequestQuote.jpd
  \sampleAppWeb\src\requestquote\RequestQuoteTransformation.dtf
  \sampleAppWeb\src\requestquote\TutorialJoin.dtf
  \sampleAppWeb\src\requestquote\services\AvailProcessor.jws
  \sampleAppWeb\src\requestquote\services\AvailProcessorControl.jcx
  \sampleAppWeb\src\requestquote\services\PriceProcessor.jws
  \sampleAppWeb\src\requestquote\services\PriceProcessorControl.jcx
  \sampleAppWeb\src\requestquote\services\TaxCalc.jws
  \sampleAppWeb\src\requestquote\services\TaxCalcControl.jcx
  \sampleAppWeb\src\requestquote\services\TaxCalcProcess.jpd
  \sampleAppWeb\src\requestquote\services\TaxCalcProcessTransformation.dtf
  \sampleAppWeb\src\requestquote\services\ValidateOrder.jpd
  \sampleAppWeb\src\requestquote\services\validatePublish.jcx
  \sampleAppWeb
  \sampleApp
  Files requiring no upgrade
```

Performing upgrade for file: RequestQuote.jpd

After the upgrade is complete, Workshop builds the workspace for the application in the IDE. A log of the import process is displayed in the Problems pane at the bottom of the dialog box; while a log of the upgrade process is displayed in the Upgrade pane, as shown in Figure 2-9. You can customize the type of information that is displayed by making the appropriate changes in the Upgrade Preferences dialog box. For more information on the type of error messages to be displayed, see Figure 2-5.

Figure 2-9 Import and Upgrade Progress Log in the BEA Workshop for WebLogic Platform IDE



Eclipse Plug-in Development Environment (PDE)

This page will help familiarize you with the Eclipse Plug-in Development Environment.
To get started, read the sections below and click on the related links.

The Plug-in Development Perspective
To do plug-in development open to the Plug-in Development Perspective by selecting Window > Open Perspective > Plug-in Development.

Create New Plug-ins
To start create a plug-in project. Choose one of the available templates to create initial content and let the wizard open the plug-in manifest file. Read the 'Welcome' page and familiarize yourself with things you can do with the newly created plug-in.

Learn more
To learn more you can browse the Plug-in Development User Guide.

0 errors, 100 warnings, 0 infos (Filter matched 100 of 135 items)

Description	Resource	In Folder	Location
IWAE0033W The module sampleApp...	sampleApp		
IWAE0033W The module sampleApp...	sampleApp		
The file cannot be validated as the X...	Persistent...	sampleAppWeb/build/proc...	line 2
The file cannot be validated as the X...	Persistent...	sampleAppWeb/build/proc...	line 2
The file cannot be validated as the X...	SyncDispa...	sampleAppWeb/build/proc...	line 2

Note: For the purpose of this tutorial, you can ignore the warning and information type error messages displayed in the Problems pane.

With this, you have successfully completed the process of importing the application into the 9.2 workspace. The following sections will help you validate the application in the 9.2 environment. The tasks include deploying and running the application in a WebLogic Integration domain.

Validating the Upgraded Application

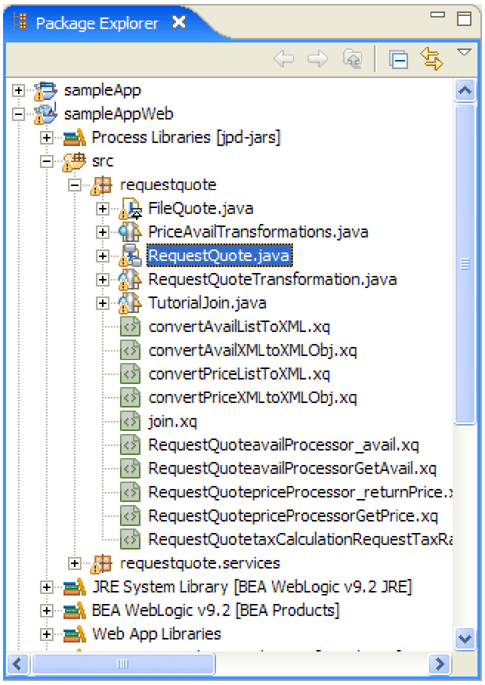
This section describes the steps involved in validating the upgraded application in the 9.2 environment. The task involves deploying the application on the server, publishing it, and subsequently running the application with some test values.

Deploying and Publishing the Application

Perform the following tasks to deploy the application after a successful upgrade.

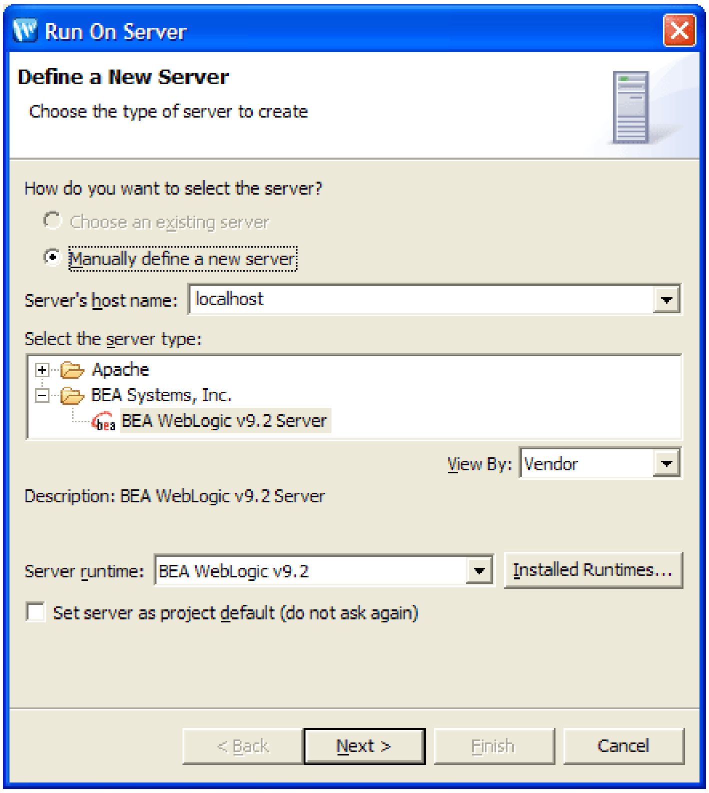
1. In the Package Explorer pane, navigate through the directory structure to view the **sampleAppWeb** → **src** → **requestquote** → **RequestQuote.java** file, as shown in Figure 2-10.

Figure 2-10 Selecting the RequestQuote.java File


Package Explorer
sampleApp
sampleAppWeb
Process Libraries [jpd-jars]
src
requestquote
FileQuote.java
PriceAvailTransformations.java
RequestQuote.java
RequestQuoteTransformation.java
TutorialJoin.java
convertAvailListToXML.xq
convertAvailXMLtoXMLObj.xq
convertPriceListToXML.xq
convertPriceXMLtoXMLObj.xq
join.xq
RequestQuoteavailProcessor_avail.xq
RequestQuoteavailProcessorGetAvail.xq
RequestQuotepriceProcessor_returnPrice.x
RequestQuotepriceProcessorGetPrice.xq
RequestQuotetaxCalculationRequestTaxRa
requestquote.services
JRE System Library [BEA WebLogic v9.2 JRE]
BEA WebLogic v9.2 [BEA Products]
Web App Libraries

2. Right-click the **RequestQuote.java** file and select **Run As** → **1 Run On Server** menu option. The Run on Server dialog box appears, as shown in Figure 2-11.

Figure 2-11 Defining a New Server to Deploy the Application



Run On Server

Define a New Server

Choose the type of server to create

How do you want to select the server?

( ) Choose an existing server

(●) Manually define a new server

Server's host name: localhost

Select the server type:

- Apache
- BEA Systems, Inc.
  - BEA WebLogic v9.2 Server

View By: Vendor

Description: BEA WebLogic v9.2 Server

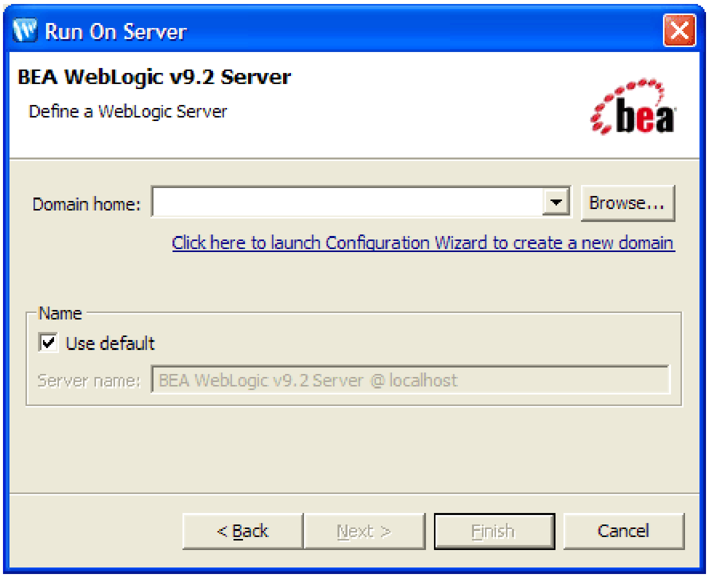
Server runtime: BEA WebLogic v9.2

[ ] Set server as project default (do not ask again)

3. Select the **Manually define a new server** check box and fill up the form as displayed in Figure 2-11.

4. Click **Next** to proceed. The Run On Server dialog box is refreshed with the Define a WebLogic Server page, as shown in Figure 2-12.

Figure 2-12 Defining a WebLogic Server



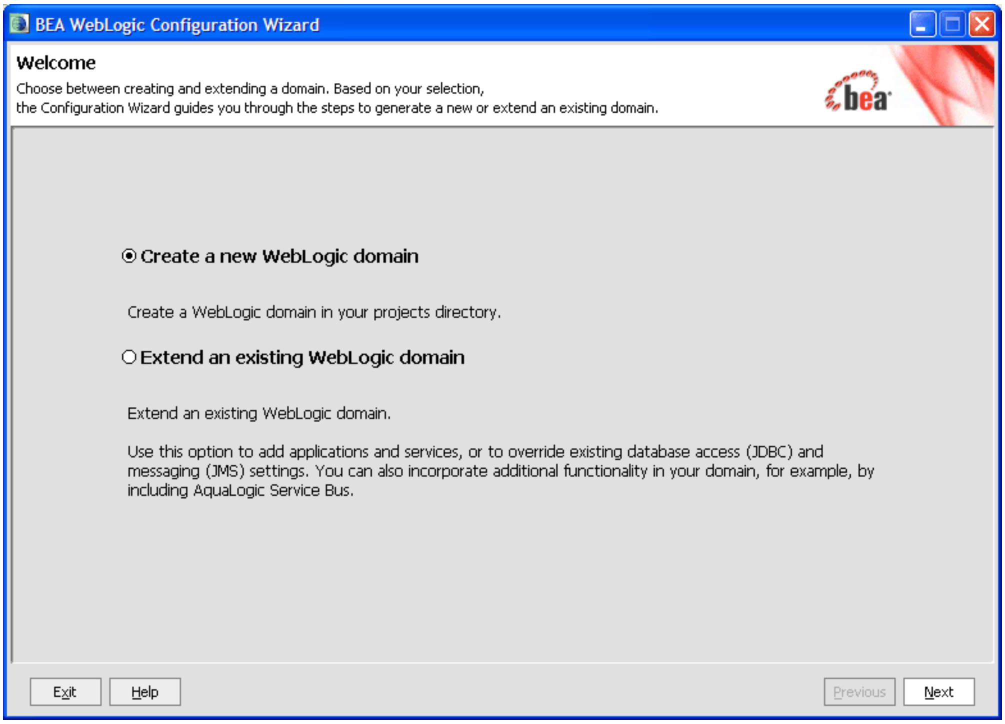
In following section, you will create a new WebLogic Integration domain in the WebLogic Server.

Creating a WebLogic Integration Domain Using the Configuration Wizard

Perform the following steps to setup the WebLogic Integration domain in order to deploy the `sampleApp` application.

1. Select the **Click here to launch Configuration Wizard to create a new domain** option in Run On Server dialog box, to start the BEA WebLogic Configuration Wizard. As shown in Figure 2-13.

Figure 2-13 Creating a New WebLogic Domain



Welcome

Choose between creating and extending a domain. Based on your selection,
the Configuration Wizard guides you through the steps to generate a new or extend an existing domain.

[✓ Create a new WebLogic domain]

Create a WebLogic domain in your projects directory.

Extend an existing WebLogic domain

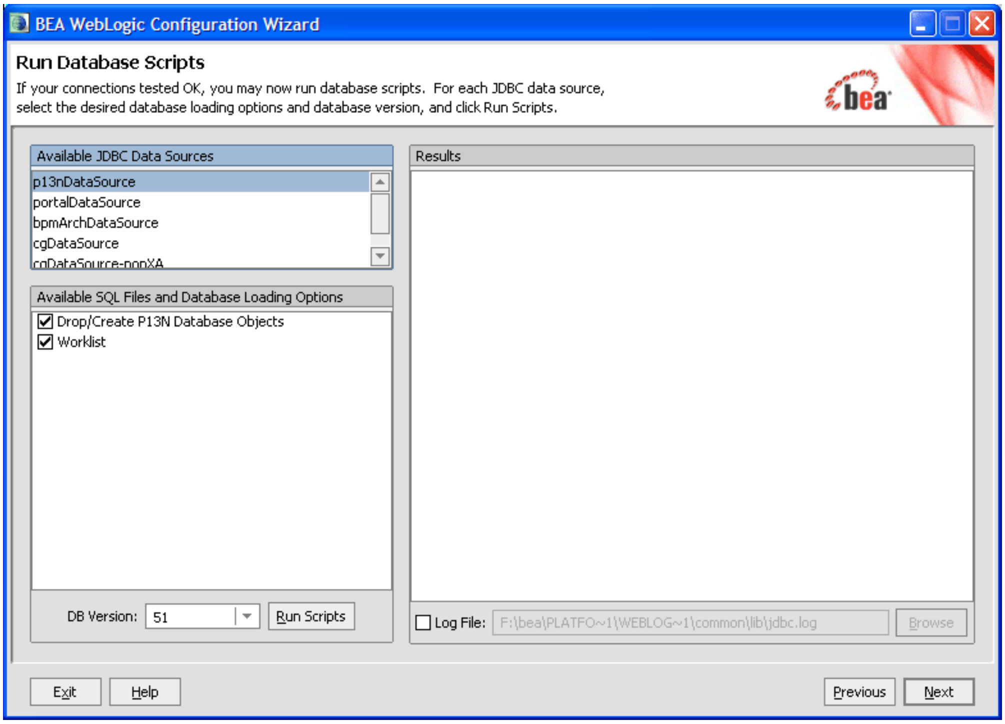
Extend an existing WebLogic domain.

Use this option to add applications and services, or to override existing database access (JDBC) and messaging (JMS) settings. You can also incorporate additional functionality in your domain, for example, by including AquaLogic Service Bus.

2. Select the **Create a new WebLogic domain** option and click **Next**. This will display the Select Domain Source page.
3. Select the **Workshop for WebLogic Platform** and the **WebLogic Integration** options and click **Next**. This will display the Configure Administrator Username and Password page.
4. Enter `weblogic` in the **User name**, **User password**, and the **Confirm user password** fields, and click **Next**. This will display the Configure Server Start Mode and JDK page.
5. In the WebLogic Domain Startup Mode pane, select **Development Mode**. In the JDK Selection pane, select **Sun SDK 1.5.0_04 @ F:\bea\platform92_for_Beta\jdk150_04** and click **Next** to proceed. This will display the Customize Environment and Service Settings page.
6. Select **Yes** and click **Next** to proceed. This will display the Configure the Administrator Server page. Click **Next** to proceed without making any changes.

7. Similarly, do not make any changes in subsequent Configure Managed Servers, Configure Machines, and Configure the JDBC Data Source pages. Just click **Next** and skip to the Run Database Scripts page, as shown in Figure 2-14.

Figure 2-14 Running Database Scripts


BEA WebLogic Configuration Wizard
Run Database Scripts
Available JDBC Data Sources
Results
Available SQL Files and Database Loading Options
DB Version:
Run Scripts
Log File:
Browse
Exit
Help
Previous
Next

If your connections tested OK, you may now run database scripts. For each JDBC data source, select the desired database loading options and database version, and click Run Scripts.

Available JDBC Data Sources: p13nDataSource, portalDataSource, bpmArchDataSource, cgDataSource, cgDataSource-nonXA

Available SQL Files and Database Loading Options: ☑ Drop/Create P13N Database Objects, ☑ Worklist

DB Version: 51

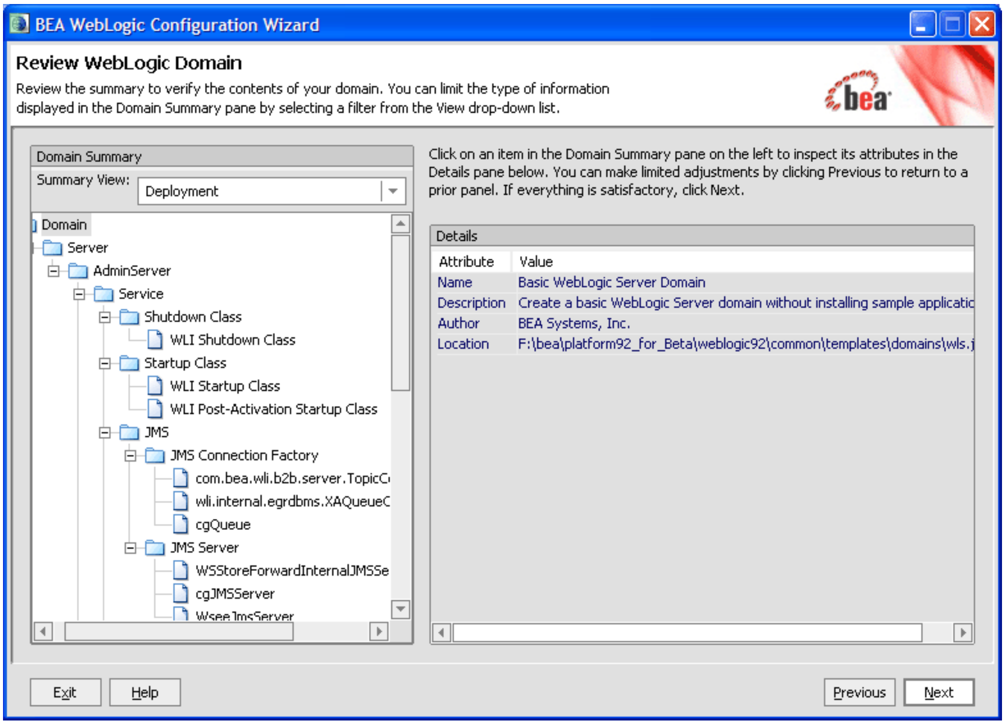
☐ Log File: F:\bea\PLATFO~1\WEBLOG~1\common\lib\jdbc.log

8. From the list of Available JDBC Data Sources, select the **p13nDataSource** option, as shown in Figure 2-14 and click **Run Scripts**. On successful execution, the Results pane will display **`Database Load Successful!`**.

9. Again, from the list of Available JDBC Data Sources, scroll down and select **cgDataSource-nonXA** and click **Run Scripts** again. On successful execution, the Results pane will display **`Database Load Successful!`**.

10. Click **Next** to proceed. This will display the Configure JMS File Stores page, do not make any changes and click **Next** to proceed. This will display the Review WebLogic Domain page, Figure 2-15, which summarizes the contents of the domain, and reflects the options you have selected.

Figure 2-15 Review the Domain Configuration Settings



Review WebLogic Domain

Review the summary to verify the contents of your domain. You can limit the type of information displayed in the Domain Summary pane by selecting a filter from the View drop-down list.

Domain Summary

Summary View: Deployment

```
Domain
  Server
    AdminServer
      Service
        Shutdown Class
          WLI Shutdown Class
        Startup Class
          WLI Startup Class
          WLI Post-Activation Startup Class
        JMS
          JMS Connection Factory
            com.bea.wli.b2b.server.TopicC
            wli.internal.egrdbms.XAQueueC
            cgQueue
          JMS Server
            WSStoreForwardInternalJMSSe
            cgJMSServer
            WseeJmsServer
```

Click on an item in the Domain Summary pane on the left to inspect its attributes in the Details pane below. You can make limited adjustments by clicking Previous to return to a prior panel. If everything is satisfactory, click Next.

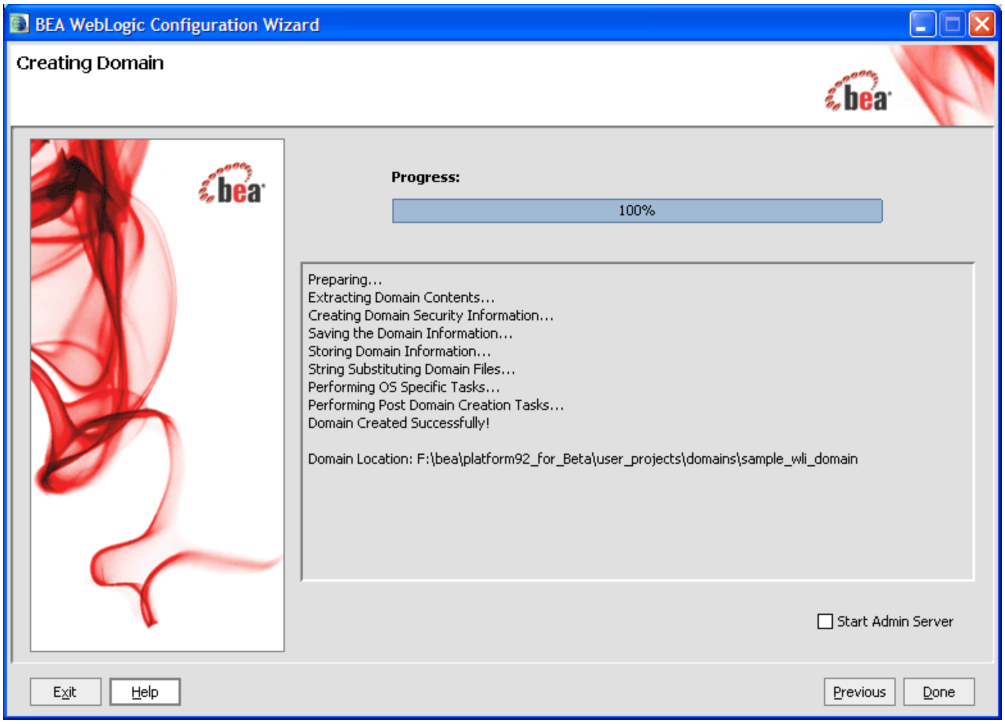
Details

Attribute	Value
Name	Basic WebLogic Server Domain
Description	Create a basic WebLogic Server domain without installing sample applicatic
Author	BEA Systems, Inc.
Location	F:\bea\platform92_for_Beta\weblogic92\common\templates\domains\wls.j

11. Click **Next** to proceed without making any changes. This will display the Create WebLogic Domain page.

12. Provide a name in the **Domain Name** field, for example **`sample_wli_domain`**, and click **Create**. This will start the domain creation process and on successful creation, the Create Domain dialog box will appears as shown in Figure 2-16.

Figure 2-16 Successful Creation of a Domain



```
Preparing...
Extracting Domain Contents...
Creating Domain Security Information...
Saving the Domain Information...
Storing Domain Information...
String Substituting Domain Files...
Performing OS Specific Tasks...
Performing Post Domain Creation Tasks...
Domain Created Successfully!

Domain Location: F:\bea\platform92_for_Beta\user_projects\domains\sample_wli_domain
```

13. Click **Done** to conclude the Domain creation process. You now need to publish the application on the server with the domain you just created.

14. After creating the Domain using the Configuration Wizard, return to the Run On Server dialog box as shown in Figure 2-12.

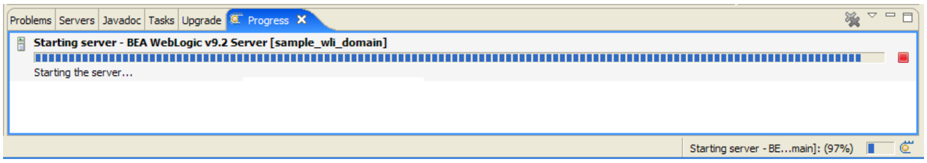
15. Click **Browse** and navigate to the folder where the new WebLogic Integration Domain `sample_wli_domain` was created and select it. As shown in Figure 2-16, the new Domain location for this tutorial is:

    ```
    F:\bea\platform92_for_Beta\user_projects\domains\sample_wli_domain
    ```

16. Click **Next** on the Run On Server dialog box to display the Add and Remove Projects page. Ensure the **sampleApp** project is listed in the **Configured projects** column.

17. Click **Finish** to publish the Application. You can view the progress of the build and publish tasks in the **Progress** tab, as shown in Figure 2-17. This process may take some time if you are starting the server for the first time.

Figure 2-17 Deploy and Publish Status



Starting server - BEA WebLogic v9.2 Server [sample_wli_domain]

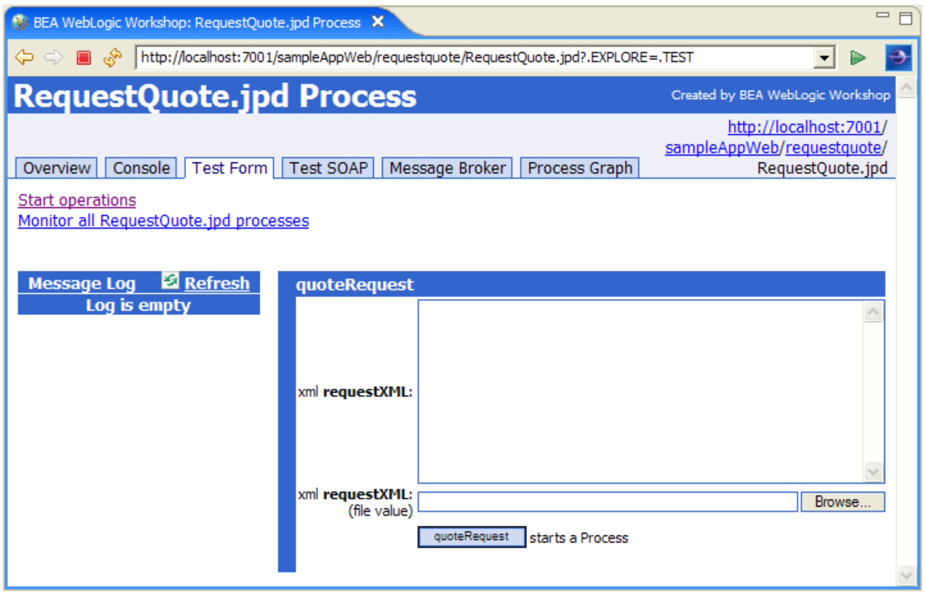
Starting the server...

Starting server - BE...main]: (97%)

After successfully deploying and publishing the Application, a browser pane is opened in the IDE, displaying the **Overview** tab of **RequestQuote.jpd Process**.

18. Click the **Test Form** tab in the browser pane, as shown in Figure 2-18.

Figure 2-18 Published Application Ready for Validation



http://localhost:7001/sampleAppWeb/requestquote/RequestQuote.jpd?.EXPLORE=.TEST

RequestQuote.jpd Process

Created by BEA WebLogic Workshop

http://localhost:7001/sampleAppWeb/requestquote/RequestQuote.jpd

Log is empty

quoteRequest

xml **requestXML**:

xml **requestXML**: (file value)

quoteRequest starts a Process

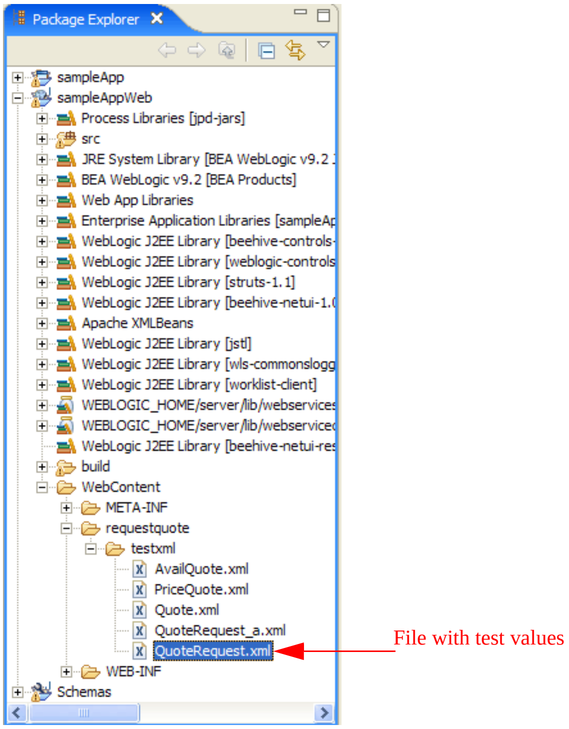
The Application is now ready for the final validation process, running it. Details are provided in the following section.

Running the Application

This section describes the tasks that need to be performed to run the Application. This is the final step in validating the upgraded Application, and the 8.x to 9.2 upgrade procedure. The primary task of this section is to provide a set of test values to execute the Application, and to verify the results.

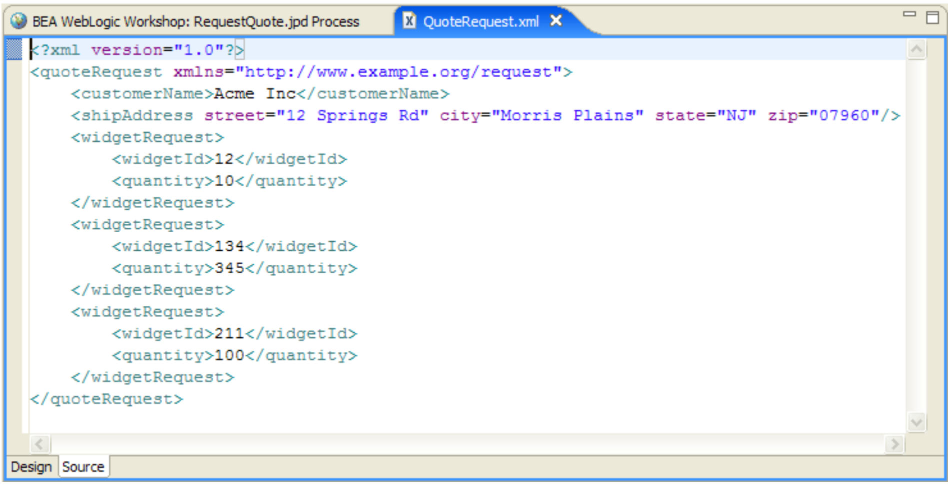
1. Locate the **`QuoteRequest.xml`** file that contains the test values required to validate the upgraded Application. The file is under the `WebContent` folder in the **Package Explorer** pane, as shown in Figure 2-19.

Figure 2-19 Locating the XML Document for Validation


Package Explorer
sampleApp
sampleAppWeb
Process Libraries [jpd-jars]
src
JRE System Library [BEA WebLogic v9.2
BEA WebLogic v9.2 [BEA Products]
Web App Libraries
Enterprise Application Libraries [sampleAp
WebLogic J2EE Library [beehive-controls-
WebLogic J2EE Library [weblogic-controls
WebLogic J2EE Library [struts-1.1]
WebLogic J2EE Library [beehive-netui-1.0
Apache XMLBeans
WebLogic J2EE Library [jstl]
WebLogic J2EE Library [wls-commonslogg
WebLogic J2EE Library [worklist-client]
WEBLOGIC_HOME/server/lib/webservices
WEBLOGIC_HOME/server/lib/webservices
WebLogic J2EE Library [beehive-netui-res
build
WebContent
META-INF
requestquote
testxml
AvailQuote.xml
PriceQuote.xml
Quote.xml
QuoteRequest_a.xml
QuoteRequest.xml
WEB-INF
Schemas
File with test values

2. Right click the **`QuoteRequest.xml`** file and select **Open With** → **Text Editor** to display its contents in the IDE browser, as shown in Figure 2-20.

Figure 2-20 Test Values for Validation

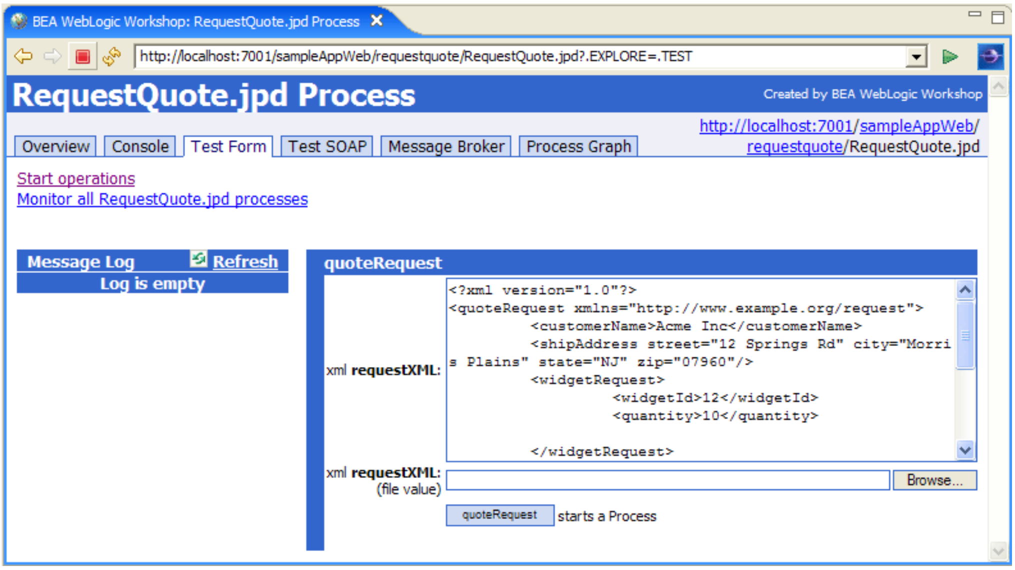


```
<?xml version="1.0"?>
<quoteRequest xmlns="http://www.example.org/request">
    <customerName>Acme Inc</customerName>
    <shipAddress street="12 Springs Rd" city="Morris Plains" state="NJ" zip="07960"/>
    <widgetRequest>
        <widgetId>12</widgetId>
        <quantity>10</quantity>
    </widgetRequest>
    <widgetRequest>
        <widgetId>134</widgetId>
        <quantity>345</quantity>
    </widgetRequest>
    <widgetRequest>
        <widgetId>211</widgetId>
        <quantity>100</quantity>
    </widgetRequest>
</quoteRequest>
```

3. Copy all the contents of the source file and close it.
4. Paste the contents in the **xml requestXML** field of the Test Form tab, as shown in Figure 2-21.

   Another way to provide the test values is to click **Browse** in the **Test Form** tab of the browser pane, as shown in Figure 2-18. Subsequently, select the `F:\bea\platform92_for_Beta\user_projects\w4WP_workspaces\upgrade\sampleAppWeb\WebContent\requestquote\testxml\`**`QuoteRequest`**`.xml` file and click **Open**.

Figure 2-21 Providing the Test Values for Validation

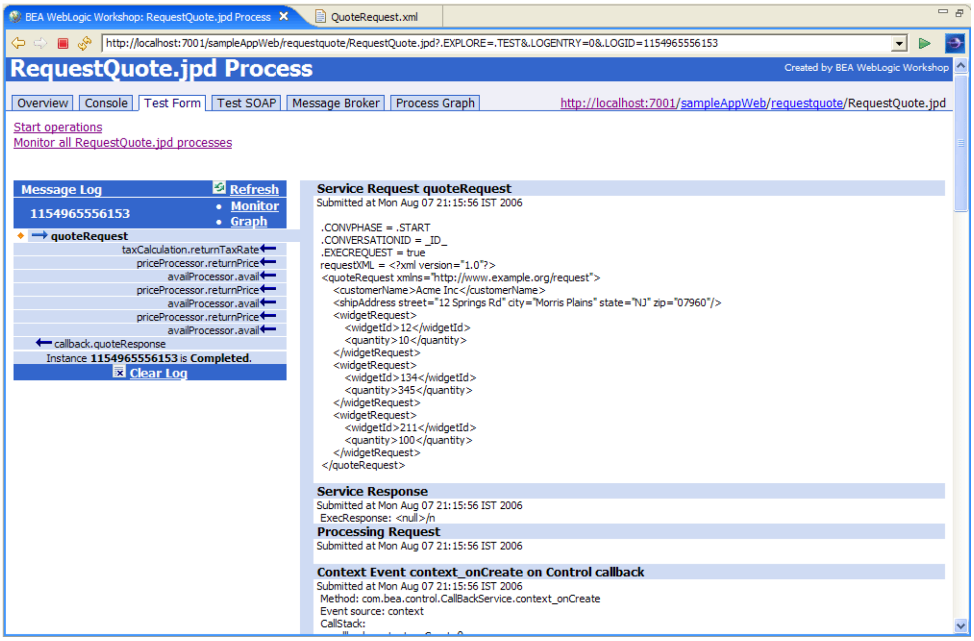

BEA WebLogic Workshop: RequestQuote.jpd Process
http://localhost:7001/sampleAppWeb/requestquote/RequestQuote.jpd?.EXPLORE=.TEST
RequestQuote.jpd Process
Created by BEA WebLogic Workshop
http://localhost:7001/sampleAppWeb/requestquote/RequestQuote.jpd
Overview
Console
Test Form
Test SOAP
Message Broker
Process Graph
Start operations
Monitor all RequestQuote.jpd processes
Message Log
Refresh
Log is empty
quoteRequest
xml requestXML:
xml requestXML: (file value)
Browse...
quoteRequest
starts a Process

```
<?xml version="1.0"?>
<quoteRequest xmlns="http://www.example.org/request">
        <customerName>Acme Inc</customerName>
        <shipAddress street="12 Springs Rd" city="Morri
s Plains" state="NJ" zip="07960"/>
        <widgetRequest>
                <widgetId>12</widgetId>
                <quantity>10</quantity>

        </widgetRequest>
```

5. Click **quoteRequest** to start the process. On successful completion of the process, the **Test Form** tab is refreshed as shown in Figure 2-22.

Figure 2-22 Successful Execution


BEA WebLogic Workshop: RequestQuote.jpd Process
QuoteRequest.xml
http://localhost:7001/sampleAppWeb/requestquote/RequestQuote.jpd?.EXPLORE=.TEST&.LOGENTRY=0&.LOGID=1154965556153
RequestQuote.jpd Process
Created by BEA WebLogic Workshop
Overview
Console
Test Form
Test SOAP
Message Broker
Process Graph
http://localhost:7001/sampleAppWeb/requestquote/RequestQuote.jpd
Start operations
Monitor all RequestQuote.jpd processes
Message Log
Refresh
Monitor
Graph
Clear Log

```
1154965556153
quoteRequest
        taxCalculation.returnTaxRate
        priceProcessor.returnPrice
        availProcessor.avail
        priceProcessor.returnPrice
        availProcessor.avail
        priceProcessor.returnPrice
        availProcessor.avail
callback.quoteResponse
Instance 1154965556153 is Completed.
```

Service Request quoteRequest
Submitted at Mon Aug 07 21:15:56 IST 2006

```
.CONVPHASE = .START
.CONVERSATIONID = _ID_
.EXECREQUEST = true
requestXML = <?xml version="1.0"?>
<quoteRequest xmlns="http://www.example.org/request">
   <customerName>Acme Inc</customerName>
   <shipAddress street="12 Springs Rd" city="Morris Plains" state="NJ" zip="07960"/>
   <widgetRequest>
      <widgetId>12</widgetId>
      <quantity>10</quantity>
   </widgetRequest>
   <widgetRequest>
      <widgetId>134</widgetId>
      <quantity>345</quantity>
   </widgetRequest>
   <widgetRequest>
      <widgetId>211</widgetId>
      <quantity>100</quantity>
   </widgetRequest>
</quoteRequest>
```

Service Response
Submitted at Mon Aug 07 21:15:56 IST 2006
ExecResponse: <null>/n

Processing Request
Submitted at Mon Aug 07 21:15:56 IST 2006

Context Event context_onCreate on Control callback
Submitted at Mon Aug 07 21:15:56 IST 2006
Method: com.bea.control.CallBackService.context_onCreate
Event source: context
CallStack:

You have now successfully imported, published, and validated an 8.1 Application in the 9.2 workspace.

Summarize the WebLogic Integration Upgrade Process

This section briefly summarizes the post-upgrade WebLogic Integration source artifacts.

- The following source file types are renamed to have a `.java` file extension:
  - DTF: Data Transformation File
  - JPD: Java Process Definition file
  - JCS: Java control source file
  - JCX: Java control extensions file
- The 8.1 XQuery files are updated with a comment indicating they belong to the XQuery version 2002. This helps differentiate the XQuery files as the 9.2 release uses the XQuery version 2004 as the default.
- All controls, WebLogic Integration included, are converted to use Apache Beehive with a `.java` file extension.
- The 8.1 channel files are moved to the 9.2 Utility projects, though they are not modified in any manner.
- The upgraded application is adapted to the 9.2 workspace, with an additional project that is created during upgrade. This additional EAR project combines the other projects into a J2EE application.