Oracle® Fusion Middleware

Upgrading Oracle WebLogic Server

12*c* (12.1.2)

E28255-03

February 2014

This document describes the procedures to upgrade an application environment to Oracle WebLogic Server 12*c* (12.1.2).

Oracle Fusion Middleware Upgrading Oracle WebLogic Server, 12*c* (12.1.2)

E28255-03

Contents

Preface vii

Documentation Accessibility vii
Conventions vii

1 Introduction

1.1 Version Compatibility 1-2
1.2 Important Terminology 1-2
1.3 Upgrading From a WebLogic Version Prior to WebLogic Server 10.3.1 1-2
1.4 Overview of the Upgrade Process 1-3
1.5 Before You Begin 1-4
1.6 Interoperability and Compatibility with Previous Releases 1-6

2 Roadmap for Upgrading Your Application Environment

2.1 Plan the Upgrade 2-1
2.1.1 Step 1: Inventory the Application Environment 2-1
2.1.2 Step 2: Verify Supported Configuration Information 2-2
2.1.3 Step 3: Review the Compatibility Information 2-3
2.1.4 Step 4: Create an Upgrade Plan 2-3
2.2 Prepare to Upgrade 2-3
2.2.1 Step 1: Check Your Applications (Undeploy If Necessary) 2-3
2.2.2 Step 2: Shut Down Servers in the Application Environment 2-4
2.2.3 Step 3: Back Up the Application Environment 2-4
2.2.4 Step 4: Install Required Oracle Products 2-4
2.2.5 Step 5: Set Up the Environment 2-4
2.3 Upgrade Your Application Environment 2-4
2.4 What to Do If the Upgrade Process Fails 2-6

3 Reconfiguring WebLogic Domains

3.1 Before You Begin 3-1
3.1.1 Upgrading Domains Created Prior to WebLogic Server 10.3.0 3-1
3.1.2 Setting CONFIG_JVM_ARGS on UNIX and Linux Systems 3-2
3.1.3 Backing Up the Domain 3-2
3.2 Reconfiguring a WebLogic Domain 3-2
3.2.1 Reconfiguring a WebLogic Domain in Graphical Mode 3-3
3.2.2 Reconfiguring a WebLogic Domain Using WebLogic Scripting Tool 3-5

3.2.3 Completing the Node Manager Configuration 3-5
3.3 Updating a Managed Server Domain on a Remote Machine 3-7
3.4 Important Notes About the Domain Upgrade Process 3-8
3.5 Completing Post-Upgrade Tasks 3-9
3.5.1 Re-apply Customizations to Startup Scripts 3-9
3.5.1.1 Default Startup Scripts 3-9
3.5.1.2 Custom Startup Scripts 3-10
3.5.2 Verify File Permissions 3-10
3.5.3 Verify Remote Server Startup Options 3-10
3.5.4 Promote the Application Environment to Production 3-11
3.6 Upgrading a Domain that Uses an Evaluation Database 3-11

4 Upgrading WebLogic Web Services

4.1 Upgrading a 10.3.*x* RESTful Web Service (JAX-RS) to 12.1.*x* 4-1
4.1.1 Upgrading a 10.3.*x* RESTful Web Service That Uses the Jersey 1.9 Shared Libraries 4-1
4.1.2 Upgrading a 10.3.*x* RESTful Web Service That Uses the Jersey 1.1.5.1 Shared Libraries 4-2
4.2 Upgrading a 10.x WebLogic Web Service (JAX-WS or JAX-RPC) to 12.1.*x* 4-4
4.3 Upgrading an 8.1 WebLogic Web Service to 12.1.*x* 4-4
4.3.1 Upgrading an 8.1 WebLogic Web Service to the WebLogic JAX-RPC Stack 4-4
4.3.1.1 Upgrading an 8.1 Java Class-Implemented WebLogic Web Service to 12.1.*x*: Main Steps 4-5
4.3.1.1.1 Example of an 8.1 Java File and the Corresponding 12.1.*x* JWS File 4-7
4.3.1.1.2 Example of an 8.1 and Updated 12.1.*x* Ant Build File for Java Class-Implemented Web Services 4-8
4.3.1.2 Upgrading an 8.1 EJB-Implemented WebLogic Web Service to 12.1.*x*: Main Steps 4-9
4.3.1.2.1 Example of 8.1 EJB Files and the Corresponding 12.1.*x* JWS File 4-12
4.3.1.2.2 8.1 SessionBean Class 4-12
4.3.1.2.3 8.1 Remote Interface 4-13
4.3.1.2.4 8.1 EJB Home Interface 4-13
4.3.1.2.5 Equivalent 12.1.*x* JWS File 4-14
4.3.1.2.6 Example of an 8.1 and Updated 12.1.*x* Ant Build File for an 8.1 EJB-Implemented Web Service 4-14
4.3.1.3 Mapping of servicegen Attributes to JWS Annotations or jwsc Attributes 4-16
4.3.2 Upgrading an 8.1 WebLogic Web Service to the WebLogic JAX-WS Stack 4-20

5 Reconfiguration Wizard Screens

5.1 Select Domain 5-2
5.2 Reconfiguration Setup Progress 5-3
5.3 Domain Mode and JDK 5-4
5.4 Database Configuration Type 5-5
5.5 JDBC Component Schema 5-6
5.6 GridLink Oracle RAC Component Schema 5-8
5.7 Oracle RAC Multi Data Source Component Schema 5-10
5.8 JDBC Component Schema Test 5-12

5.9 JDBC Data Sources 5-13
5.10 GridLink Oracle RAC Data Sources 5-15
5.11 Oracle RAC Multi Data Sources 5-17
5.12 JDBC Data Sources Test 5-18
5.13 Database Scripts 5-19
5.14 Credentials 5-20
5.15 Keystore 5-21
5.16 Advanced Configuration 5-22
5.17 Managed Servers 5-23
5.18 Clusters 5-25
5.19 Assign Servers to Clusters 5-26
5.20 HTTP Proxy Applications 5-27
5.21 Coherence Clusters 5-28
5.22 Machines 5-30
5.23 Assign Servers to Machines 5-32
5.24 Deployments Targeting 5-33
5.25 Services Targeting 5-34
5.26 JMS File Stores 5-36
5.27 Configuration Summary 5-37
5.28 Reconfiguration Progress 5-38
5.29 Reconfiguration Success 5-39

A WebLogic Server 12.1.2 Compatibility with Previous Releases

A.1 Maximum POST Size A-2
A.2 WLDF Schema Upgrade A-2
A.3 jdbc-connection-timeout-secs Element A-3
A.4 Commitment of Local Transactions A-3
A.5 JVM Settings A-3
A.5.1 Setting the Location of the Java Endorsed Directory A-3
A.5.2 Setting permgen space A-4
A.6 Node Manager startScriptEnabled Default A-5
A.7 Enterprise Java Beans (EJBs) A-5
A.8 WebLogic Server 8.1 Web Services Stack Has Been Removed A-5
A.9 Universal Description and Discover (UDDI) Registry Has Been Removed A-6
A.10 Certicom SSL Implementation A-6
A.11 Coherence Version A-6
A.12 Deprecated and Obsolete Web Application Features A-6
A.13 Evaluation Database Changed From PointBase to Derby A-6
A.14 Data Source Profile Logging A-6
A.15 ONS Debugging A-7
A.16 Oracle Type 4 JDBC drivers from DataDirect A-7
A.17 Default Message Mode Has Changed A-7
A.18 Modifications to SSLMBean A-7
A.19 New Web Services Features A-7
A.20 Introduction of JSSE A-8
A.21 Performance Enhancements for Security Policy Deployment A-8
A.22 ActiveCache A-9

A.23 Class Caching A-9
A.24 Deprecated JDBC Drivers A-9
A.25 Changes to weblogic.jms.extension API A-9
A.26 Persistent Store Updates A-10
A.27 Oracle Internet Directory and Oracle Virtual Directory Authentication Providers A-10
A.28 capacityIncrement Attribute A-10
A.29 Middleware Home Directory A-10
A.30 Resource Registration Name A-10
A.31 Servlet Path Mapping A-11

Preface

This preface describes the document accessibility features and conventions used in this guide—*Upgrading Oracle WebLogic Server.*

Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at `http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc`.

Access to Oracle Support

Oracle customers have access to electronic support through My Oracle Support. For information, visit `http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info` or visit `http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs` if you are hearing impaired.

Conventions

The following text conventions are used in this document:

Convention	Meaning
boldface	Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.
italic	Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.
`monospace`	Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.

1 Introduction

This document describes the process of upgrading to WebLogic Server 12*c* Release 1 (12.1.2) and moving an existing application to WebLogic Server 12.1.2. Although you may decide to change your application when upgrading to WebLogic Server 12.1.2, and although in some cases, you must change your application, this document focuses on issues that you should consider when moving an application to WebLogic Server 12.1.2 without making application changes.

The instructions in this document are for the following upgrade scenarios:

- Upgrading from any WebLogic Server 10.3.x release to WebLogic Server 12.1.2
- Upgrading from WebLogic Server 12.1.1 to WebLogic Server 12.1.2

Note: If you are upgrading from a release prior to WebLogic Server 10.3.1, see Section 1.3, "Upgrading From a WebLogic Version Prior to WebLogic Server 10.3.1."

This document also describes how to update (reconfigure) an existing WebLogic Server 10.3.x or 12.1.1 domain to be compatible with WebLogic Server 12.1.2, as well as how to upgrade Web Services.

WebLogic Server generally supports very high levels of upgrade capability across WebLogic Server versions. This document is intended to provide WebLogic Server upgrade support and identify issues that may surface during an upgrade so that they can be easily resolved.

Note: For information about upgrading your Java EE environment and your deployed applications from Oracle Application Server 10*g* and Oracle Containers for Java EE (OC4J) to WebLogic Server 12*c* Release 1 (12.1.2), see *Oracle Fusion Middleware Upgrade Guide for Java EE*.

WebLogic Server 12.1.2 includes the Fusion Middleware Reconfiguration Wizard to assist you with upgrading WebLogic Server and your application environments.

Most WebLogic Server applications can be run without modifications in the new WebLogic Server 12.1.2 application environment.

The following topics are discussed in this chapter:

- Section 1.1, "Version Compatibility"
- Section 1.2, "Important Terminology"

- Section 1.3, "Upgrading From a WebLogic Version Prior to WebLogic Server 10.3.1"
- Section 1.4, "Overview of the Upgrade Process"
- Section 1.5, "Before You Begin"
- Section 1.6, "Interoperability and Compatibility with Previous Releases"

1.1 Version Compatibility

Prior to upgrade, you should review the WebLogic Server and domain compatibility requirements for WebLogic Server 12.1.2. For more information, see "Compatibility Within a Domain" in *Understanding Oracle WebLogic Server*.

1.2 Important Terminology

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

- **Upgrade**—In this document, the term upgrade refers to the process of upgrading WebLogic Server and moving an existing application, unchanged, to a new (upgraded) WebLogic Server version.
- **Reconfiguration**—The process of upgrading a domain that was created with a previous WebLogic Server version so that it is compatible with the WebLogic Server version to which you have upgraded. This can be done using either the Reconfiguration Wizard or WLST.
- **Application Environment**—An application environment includes applications and the WebLogic domains in which they are deployed. It also includes any application data associated with the domain, and may include resources such as database servers, firewalls, load balancers, and LDAP servers.
- **Migrate**—To move an application or domain configuration from a third-party product to an Oracle product.
- **Interoperability**—(1) The ability of an application deployed in one WebLogic Server version to communicate with another application that is deployed in a different WebLogic Server version. (2) The ability of Oracle product components to communicate with third-party software using standard protocols.
- **Compatibility**—The capability of an application built using one WebLogic Server release to run in another WebLogic Server release, regardless of whether the application was rebuilt.

1.3 Upgrading From a WebLogic Version Prior to WebLogic Server 10.3.1

If you are currently using a WebLogic version prior to WebLogic Server 10.3.1, upgrading to version 12.1.2 is a two-stage process:

- you must first upgrade your installation to WebLogic Server 10.3.6. To do so, follow the instructions in *Upgrade Guide for WebLogic Server 10.3.6*. See `http://docs.oracle.com/cd/E23943_01/web.1111/e13754/toc.htm`. Be sure to run the WebLogic Server 10.3.6 Domain Upgrade Wizard to upgrade your domains.

Note: To download a WebLogic Server 10.3.6 upgrade installer, enter the appropriate patch number on My Oracle Support:

- Patch 13529623—10.3.6 Generic Upgrade Installer (does not include a bundled JDK)
- Patch 13529653—10.3.6 Linux 32-bit Upgrade Installer
- Patch 13529639—10.3.6 Windows 32-bit Upgrade Installer
- Patch 13529649—10.3.6 Solaris 32-bit Upgrade Installer

- Upgrade WebLogic Server 10.3.6 to WebLogic Server 12.1.2 per the instructions in this guide.

Note: As of WebLogic Server 12.1.2, Oracle no longer provides upgrade installers. You must install WebLogic Server 12.1.2 to a new directory location. You cannot install it over an existing installation.

1.4 Overview of the Upgrade Process

The process required to upgrade an application environment depends on the scope of the application. An *application environment* includes a WebLogic domain and any applications and application data associated with the domain. It may also include external resources, such as firewalls, load balancers, and LDAP servers. Figure 1–1 shows an example of a WebLogic application environment.

Figure 1–1 Example WebLogic Application Environment

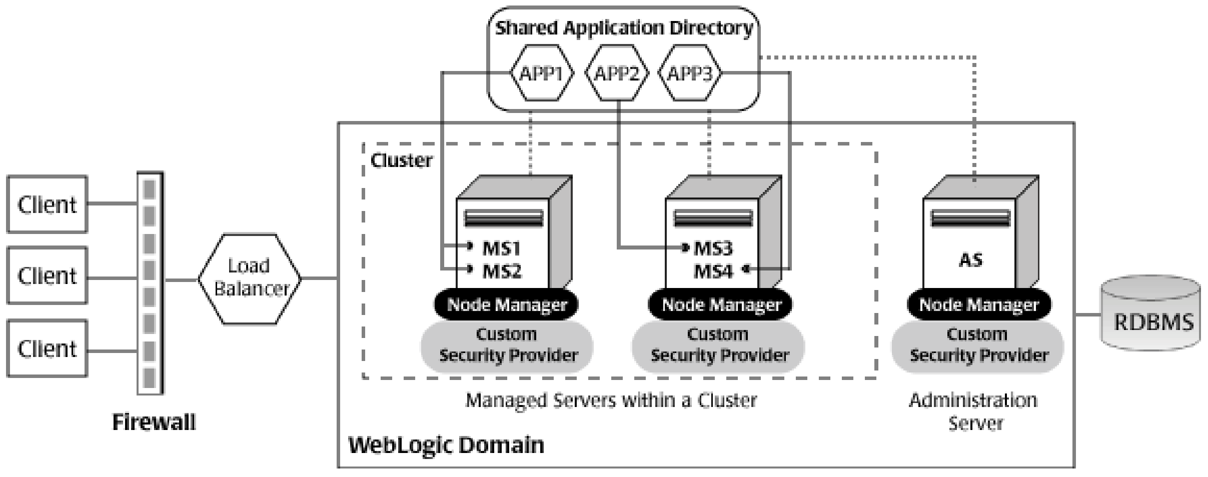


Table 1–1 lists the components of the WebLogic application environment shown in Figure 1–1 and the upgrade requirements for each.

Table 1–1 Upgrade Requirements for Components in Example WebLogic Application Environment

Component	Description	Upgrade Requirements
WebLogic domain	Includes the Administration Server (AS) and optionally one or more Managed Servers (for example, MS1, MS2, MS3, and MS4). The servers in a domain may span multiple machines. Furthermore, you can group Managed Servers into clusters to support load balancing and failover protection for critical applications. For more information about WebLogic domains, see "Understanding Oracle WebLogic domains" in *Understanding Domain Configuration for Oracle WebLogic Server*.	Upgrade the domain directory on each computer in the domain.
Applications	Any Java EE applications, including Web applications, EJBs, and so on. Typically, applications are deployed to one or more Managed Servers in a domain. Depending on the deployment strategy, applications may reside locally on a computer or be accessible using a shared directory. In addition, external client applications may access the application environment from outside a firewall.	Most WebLogic Server applications can be run without modifications in the new WebLogic Server 12.1.2 application environment. For more information, see Section 1.6, "Interoperability and Compatibility with Previous Releases."
External resources	Software components, such as databases for storing domain and application data, load balancers, and firewalls.	Verify that all external resources are compatible with WebLogic Server 12.1.2. For more information, see the Oracle Fusion Middleware Supported System Configurations page at `http://www.oracle.com/technology/software/products/ias/files/fusion_certification.html`.

Upgrading business applications that are deployed to WebLogic Server may involve upgrading multiple WebLogic Server applications, and in some cases domains, in a coordinated fashion in order to:

- maintain consistency in the WebLogic Server versions being used
- use the same supported configurations environment across the entire installation
- meet specific interoperability requirements.

For example, you may want to upgrade all applications and domains simultaneously, upgrade them in a well-defined sequence, or upgrade some applications and domains while leaving other applications and domains on older WebLogic Server versions.

1.5 Before You Begin

Before you begin the upgrade process, you should consider the scope of the environments that you are upgrading and which applications will be upgraded in which sequence. Covering all of the permutations of an upgrade is beyond the scope of this document. Therefore, you should consider the following items prior to planning your upgrade. These items focus on upgrades that involve a single application running in a single domain.

- Oracle generally recommends that you upgrade an application in development environments and use a standard QA, testing, and staging process to move upgraded applications to a production environment.

- You will typically upgrade an application either by upgrading an existing domain or by creating a new domain, from which you can run the application on the new WebLogic Server version. In some cases, you may prefer to create new domains using the Fusion Middleware Configuration Wizard or other configuration tools (such as WLST) in order to test the applications that you are upgrading.

  Note: If the domain was created using a WebLogic Server version prior to version 10.3.1, you must first install WebLogic Server 10.3.6. After installing WebLogic Server 10.3.6, run the WebLogic Server 10.3.6 Domain Upgrade Wizard to upgrade the domain. You can then use the Reconfiguration Wizard to upgrade the domain to WebLogic Server 12.1.2.

- When planning a WebLogic Server version upgrade, you should review the Fusion Middleware Supported Systems Configurations page on Oracle Technology Network (OTN) to ensure that your upgraded environment is supported by Oracle, in particular:
  - current and planned JVM and JDK versions
  - operating system versions
  - database versions
  - Web services versions
  - versions of other products that interoperate with or run on WebLogic Server, to ensure that the upgraded environment is supported by Oracle or other vendors' products that you are using with WebLogic Server.
- On an ongoing basis, Oracle documents APIs and features that have been deprecated (that is, planned for removal in a future release). This is intended to inform you that you should avoid using these APIs and features to ensure upgradability. Oracle also documents the APIs and features that have actually been removed in the current release so that if you are upgrading from prior versions, you can determine if your applications will be affected by an upgrade.

  APIs and feature removals are cumulative. For example, if you are upgrading from WebLogic Server 10.0 to WebLogic Server 12.1.2, your applications may be affected by APIs or features that were removed in WebLogic Server 10.3, as well as by APIs or features that were removed in WebLogic Server 12.1.2. When upgrading, you should review all documentation of deprecated and removed features for all applicable WebLogic Server versions.
- You should consider the impact (if any) that the upgrade process may have on any automation (such as WLST scripts) that you are using to configure, deploy, start/stop, or monitor your WebLogic Server applications. You may need to upgrade such automation along with the applications and domains you are upgrading.
- You should consider the potential impact that may result from the use of third-party libraries in your applications, as they may conflict with different versions of those same libraries that are embedded in WebLogic Server. In particular, new versions of WebLogic Server may change the version of open source libraries that are embedded in WebLogic Server. Applications that may run successfully on earlier WebLogic Server versions may encounter new class conflicts after upgrade.

If you are upgrading an application that contains embedded third-party libraries, you should consider using the Classloader Analysis Tool, and filtering classloaders when upgrading WebLogic Server applications to WebLogic Server 12.1.2. This tool enables you to identify, diagnose and resolve such conflicts, and may simplify the upgrade process.

- If you are running applications on prior versions of WebLogic Server, and are using WebLogic Server patches or bug fixes, you should investigate whether or not those patches or bug fixes have been incorporated into the version of WebLogic Server to which you are upgrading.

1.6 Interoperability and Compatibility with Previous Releases

Most existing WebLogic Server 10.3.1 or greater applications can be run without modification in the new WebLogic Server 12.1.2 application environment. You should review the compatibility information described in Appendix A, "WebLogic Server 12.1.2 Compatibility with Previous Releases," to determine whether any feature changes affect the applications in your environment. If your application uses APIs that have been deprecated or removed, then you may encounter warnings or exceptions at run time.

2 Roadmap for Upgrading Your Application Environment

This chapter describes how to prepare for and perform an upgrade of your WebLogic application environments.

Topics include:

- Section 2.1, "Plan the Upgrade"
- Section 2.2, "Prepare to Upgrade"
- Section 2.3, "Upgrade Your Application Environment"
- Section 2.4, "What to Do If the Upgrade Process Fails"

Note: If your WebLogic Server installation includes other Oracle Fusion Middleware products, refer also to *Planning an Upgrade*.

2.1 Plan the Upgrade

Planning how you will upgrade an application environment is an important step in the process. To ensure that your plan addresses all of the aspects of upgrading that are necessary for your environment, complete the following steps:

- Step 1: Inventory the Application Environment
- Step 2: Verify Supported Configuration Information
- Step 3: Review the Compatibility Information
- Step 4: Create an Upgrade Plan

2.1.1 Step 1: Inventory the Application Environment

Generate an inventory of the application environment by identifying the following components:

- Administration Server and the computer on which it resides
- Managed Servers and the computer(s) on which they reside
- Location of the applications (including all external client applications)
- External resources, for example:
  - Databases used to store persisted and application data
  - Firewalls

  - Load balancers

- Tools, scripts, templates, and source code used for automating the tasks required to create the application environment

You can view a sample application environment in Section 1.4, "Overview of the Upgrade Process."

2.1.2 Step 2: Verify Supported Configuration Information

Supported configurations (for example, JDK versions, Operating System versions, Web server versions, and database versions) have changed for WebLogic Server 12.1.2. You may be required to upgrade your environments to the supported versions of these and other products.

For information about supported configurations, see *Oracle Fusion Middleware Supported System Configurations* on Oracle Technology Network (OTN).

For databases, please note that:

- WebLogic Server 12.1.2 supports PointBase 5.7; however, PointBase is no longer included in the WebLogic Server installation program. Derby replaces PointBase for running WebLogic Server samples.

  To upgrade to WebLogic Server 12.1.2, you must create a new WebLogic Server installation. Therefore, the PointBase installation directory is not included. To continue using PointBase, see Section 3.6, "Upgrading a Domain that Uses an Evaluation Database."

  Note: The pre-5.7 version of PointBase that was distributed with earlier versions of WebLogic Server can be used only for WebLogic domains.

- As of WebLogic Server 10.3.3, the evaluation database available from the installation program that is provided for use by the sample applications and code examples, and as a demonstration database, is changed from PointBase to Derby. Derby is an open source relational database management system based on Java, JDBC, and SQL standards. For more information about Derby, see `http://db.apache.org/derby/`.

  If you have a domain based on PointBase from an earlier version of WebLogic Server, and you plan to upgrade that domain to WebLogic Server 12.1.2, you can continue to use PointBase. But you must obtain a license from `http://www.pointbase.com` to use it. For more information, see Section 3.6, "Upgrading a Domain that Uses an Evaluation Database."

- As of WebLogic Server 10.3, the Oracle Thin Drivers are included as part of the WebLogic Server installation.

- If you are using an Oracle OCI database driver and want to change to use a Thin database driver, you must remove the `server` property (as illustrated below) from the generated JDBC module. For example:

  ```
  <property>
    <name>server</name>
    <value>servername</value>
  </property>
  ```

- The Oracle Thin Drivers are installed with WebLogic Server and are ready for use. For more information about using these drivers, see "JDBC Drivers Installed with WebLogic Server" in *Administering JDBC Data Sources for Oracle WebLogic Server*.

2.1.3 Step 3: Review the Compatibility Information

Most existing WebLogic Server applications can be run without modification in the new WebLogic Server 12.1.2 application environment. However, you should review Appendix A, "WebLogic Server 12.1.2 Compatibility with Previous Releases," to determine whether any feature changes affect the applications in your environment.

2.1.4 Step 4: Create an Upgrade Plan

Using the information gathered in the preceding steps, create a plan for upgrading your application environment. Identify the scope and timing of the upgrade process, based on your business needs. Please note the following:

- Oracle does not recommend upgrading an application environment that is currently deployed in production. Instead, you should upgrade your application environment while it is under development or test and execute standard procedures for quality assurance and performance tuning before promoting the upgraded environment to production.
- If your application is complex, for example, if it includes multiple clustered domains and a large number of deployed applications, you may choose to upgrade the components of the application environment in stages.
- You may consider limiting the number of WebLogic Server versions used in any single application environment to minimize the diversity and cost of systems being administered.
- If you plan to use the RDBMS security store in a WebLogic domain, Oracle recommends that you create a new domain in which the RDBMS security store is configured. If you have an existing domain in which you want to use the RDBMS security store, you should create the new domain, then migrate your security realm to it. Oracle does not recommend "retrofitting" the RDBMS security store to an existing domain. For more information, see "Managing the RDBMS Security Store" in *Administering Security for Oracle WebLogic Server*.

2.2 Prepare to Upgrade

Before you upgrade the application environment, you must perform the following steps:

- Step 1: Check Your Applications (Undeploy If Necessary)
- Step 2: Shut Down Servers in the Application Environment
- Step 3: Back Up the Application Environment
- Step 4: Install Required Oracle Products
- Step 5: Set Up the Environment

2.2.1 Step 1: Check Your Applications (Undeploy If Necessary)

It is not necessary for WebLogic Server applications to be undeployed before upgrading the domain. In most cases, WebLogic Server applications can be run without modifications in the new WebLogic Server 12.1.2 application environment. Review the compatibility information in Appendix A, "WebLogic Server 12.1.2 Compatibility with Previous Releases," to determine whether any features changes affect the applications in your environment. Note that if you use deprecated or removed APIs in the application, you might encounter warnings or exceptions at run time.

2.2.2 Step 2: Shut Down Servers in the Application Environment

Before you upgrade, you must shut down all servers in the application environment.

2.2.3 Step 3: Back Up the Application Environment

Oracle recommends that before upgrading your application environment, you manually back up the components defined in Table 2–1. You should back up the relevant information on all machines in the domain.

Table 2–1 Recommendations for Backing Up the Application Environment

Component	Recommendations
Domain directory	Back up the Administration Server and any remote Managed Server domain directories that are defined in the application environment. **Note:** The Domain Upgrade Wizard, which automatically backed up the domain being upgraded, is no longer provided with WebLogic Server. You must manually back up your domain directory prior to upgrading the domain.
Applications and application-persisted data	Back up any applications and data that reside outside of the domain directory.
Log files	If it is important for you to maintain a record of all messages that are logged, back up the log files. As log files can be large, you may want to delete them to conserve disk space if it is not important to retain them.

2.2.4 Step 4: Install Required Oracle Products

Before upgrading your application environment, you must install the Oracle WebLogic Server 12.1.2 products that you require on each computer in the domain. For more information about installing Oracle WebLogic products, see *Installing and Configuring Oracle WebLogic Server and Coherence*.

2.2.5 Step 5: Set Up the Environment

To set up the environment for an upgrade:

1. Open an MS-DOS command prompt window (on Windows) or a command shell (on UNIX).
2. Add the WebLogic Server classes to the CLASSPATH environment variable and `*WL_HOME*\server\bin` to the PATH environment variable, where *`WL_HOME`* refers to the top-level installation directory for WebLogic Server 12.1.2.

   You can perform this step by running the `*WL_HOME*\server\bin\setWLSEnv` script.

> **Note:** On UNIX operating systems, the `setWLSEnv.sh` command does not set the environment variables in all command shells. Oracle recommends that you execute this command using the Korn shell or bash shell.

2.3 Upgrade Your Application Environment

Figure 2–1 identifies the steps required to upgrade your application environment.

Figure 2–1 Roadmap for Upgrading Your Application Environment

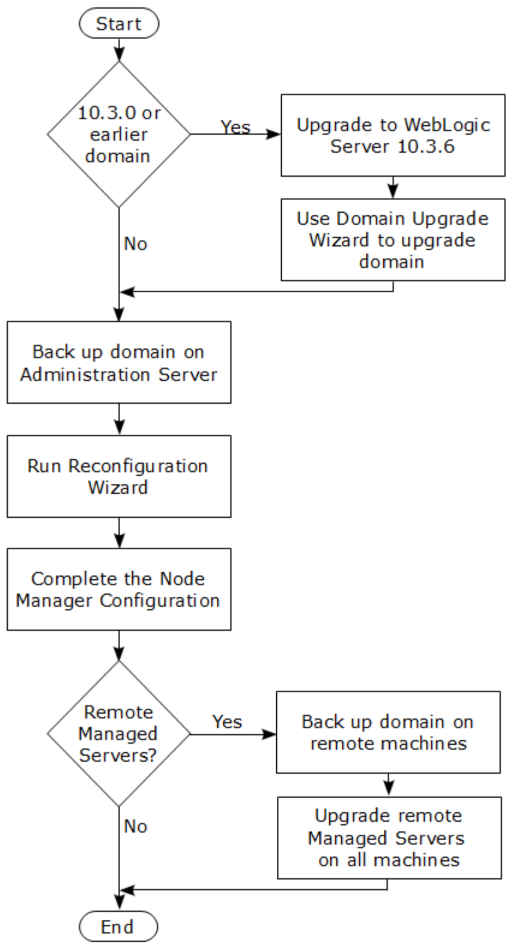

Start
10.3.0 or earlier domain
Yes
No
Upgrade to WebLogic Server 10.3.6
Use Domain Upgrade Wizard to upgrade domain
Back up domain on Administration Server
Run Reconfiguration Wizard
Complete the Node Manager Configuration
Remote Managed Servers?
Yes
No
Back up domain on remote machines
Upgrade remote Managed Servers on all machines
End

Table 2–2 summarizes the steps for updating an application environment. Some steps are mandatory, others are optional. Each step that is performed must be done on every computer in the domain and in the sequence shown in this table.

Table 2–2 Procedure for Upgrading an Application Environment

Task	Description
Upgrade to WebLogic Server 10.3.6	If the domain was created prior to WebLogic Server 10.3.0, you must first upgrade to WebLogic Server 10.3.6. You can do this using the WebLogic Server 10.3.6 upgrade installer. For more information, see the *Installation Guide for Oracle WebLogic Server 10.3.6* at `http://docs.oracle.com/cd/E23943_01/doc.1111/e14142/toc.htm`.
Run the Domain Upgrade Wizard	If the domain was created prior to WebLogic Server 10.3.0, run the WebLogic Server 10.3.6 Domain Upgrade Wizard to upgrade the domain. See Section 3.1.1, "Upgrading Domains Created Prior to WebLogic Server 10.3.0."
Back up the domain	Prior to upgrading the domain on the Administration Server, make a backup copy of the domain. See Section 3.1.3, "Backing Up the Domain."
Upgrade WebLogic domain (Administration Server)	Run the Reconfiguration Wizard to upgrade the WebLogic domain on the computer that hosts the Administration Server. See Section 3.2, "Reconfiguring a WebLogic Domain." **Note:** Oracle recommends that you completely upgrade the domain on the Administration Server before upgrading the domain on the Managed Servers.

Table 2–2 (Cont.) Procedure for Upgrading an Application Environment

Task	Description
Complete Node Manager configuration	Configure Node Manager to run as it did in the pre-upgraded domain. See Section 3.2.3, "Completing the Node Manager Configuration."
Back up the domain on each Managed Server.	Prior to upgrading the domain on a Managed Server, make a backup copy of the domain. See Section 3.1.3, "Backing Up the Domain."
Upgrade WebLogic domain (remote Managed Servers)	Use the `pack` and `unpack` commands or the WLST `writeTemplate()` command in online mode to upgrade the WebLogic domain on every computer that hosts any Managed Servers. See Section 3.3, "Updating a Managed Server Domain on a Remote Machine." For more information on `pack` and `unpack`, see *Creating Templates and Domains Using the Pack and Unpack Commands*. **Note:** Managed Servers that reside on the same computer as the Administration Server do not require additional upgrade steps.

2.4 What to Do If the Upgrade Process Fails

If any step in the upgrade process fails, the Reconfiguration Wizard displays a message indicating the reason for the failure and terminates. To proceed, perform the following steps:

1. Restore the application environment to its original state using the backup files you created in Section 2.2.3, "Step 3: Back Up the Application Environment."
2. Correct the failure reported by the Reconfiguration Wizard.
3. Run the Reconfiguration Wizard again to upgrade the domain.

3

Reconfiguring WebLogic Domains

This chapter describes how to use the Reconfiguration Wizard to reconfigure WebLogic Server domains that were created using any WebLogic Server 10.3.1 or greater release or WebLogic Server 12.1.1.

When you use the Reconfiguration Wizard to reconfigure a WebLogic Server domain, the following items are automatically updated, depending on the applications in the domain:

- WLS core infrastructure
- Domain version

Note: The Reconfiguration Wizard does not update any of your own applications that are included in the domain. For information about how to upgrade your own applications, see Appendix A, "WebLogic Server 12.1.2 Compatibility with Previous Releases."

This chapter contains the following sections:

- Section 3.1, "Before You Begin"
- Section 3.2, "Reconfiguring a WebLogic Domain"
- Section 3.3, "Updating a Managed Server Domain on a Remote Machine"
- Section 3.4, "Important Notes About the Domain Upgrade Process"
- Section 3.6, "Upgrading a Domain that Uses an Evaluation Database"

3.1 Before You Begin

Refer to the following two sections prior to beginning the upgrade process.

3.1.1 Upgrading Domains Created Prior to WebLogic Server 10.3.0

If a domain was created prior to WebLogic Server 10.3.0, you must first upgrade to WebLogic Server 10.3.6. After upgrading to WebLogic Server 10.3.6, run the Domain Upgrade Wizard to upgrade the domain.

For information about upgrading to WebLogic Server 10.3.6 and running the Domain Upgrade Wizard, see *Upgrade Guide for Oracle WebLogic Server 10.3.6* at the following URL:

`http://docs.oracle.com/cd/E23943_01/web.1111/e13754/toc.htm`

3.1.2 Setting CONFIG_JVM_ARGS on UNIX and Linux Systems

Prior to running the Reconfiguration Wizard to reconfigure a domain on a UNIX or Linux operating system, if you have not already done so, set the `CONFIG_JVM_ARGS` environment variable to the following value to use the operating system's random number generator:

```
-Djava.security.egd=file:/dev/./urandom
```

This decreases the amount of time it takes for the Reconfiguration Wizard to reconfigure a domain.

3.1.3 Backing Up the Domain

Prior to running the Reconfiguration Wizard, make a backup copy of the domain directory. For example, copy C:\domains\mydomain to C:\domains\mydomain_backup.

Prior to updating the domain on each remote Managed Server, make a backup copy of the domain directory on each remote machine.

If domain reconfiguration fails for any reason, you must copy all files and directories from the backup directory into the original domain directory to ensure that the domain is returned entirely to its original state prior to reconfiguration.

3.2 Reconfiguring a WebLogic Domain

You can reconfigure a WebLogic domain:

- In graphical mode, by running the Fusion Middleware Reconfiguration Wizard, or
- From the command line using WebLogic Scripting Tool.

When you reconfigure a domain:

- The domain version number in the config.xml file for the domain is updated to the Administration Server's installed WebLogic Server version major and minor version number (for example, 12.1.2.0).
- Reconfiguration templates for all installed Oracle products are automatically selected and applied to the domain. These templates define any reconfiguration tasks that are required to make the WebLogic domain compatible with the current WebLogic Server version.
- Start scripts are updated.
- You must manually complete the Node Manager configuration to use the existing Node Manager configuration for the domain.

Note: If you want the Node Manager for the domain to be a per-domain type rather than host-based, Oracle recommends that you use the Configuration Wizard or WLST to create a new domain instead of reconfiguring the existing domain. For information about Node Manager types, see "Configuring Java Node Manager" in *Administering Node Manager for Oracle WebLogic Server*.

Note the following items when reconfiguring a domain:

- After reconfiguring the domain on the Administration Server, you must port the reconfigured domain to all remote Managed Servers in the domain. For more information, see Section 3.3, "Updating a Managed Server Domain on a Remote Machine."

- After reconfiguring a domain using either WLST or the Reconfiguration Wizard, you must take additional steps to complete the Node Manager configuration. See Section 3.2.3, "Completing the Node Manager Configuration."

3.2.1 Reconfiguring a WebLogic Domain in Graphical Mode

To reconfigure a WebLogic domain by using the Reconfiguration Wizard in graphical mode, start the wizard as follows. You can start the Reconfiguration Wizard only from a DOS command prompt window or UNIX shell.

Caution: Once the domain reconfiguration process starts, it is irreversible. Prior to running the Reconfiguration Wizard, ensure that you have backed up the domain as described in Section 3.1.3, "Backing Up the Domain." If an error or other interruption occurs while running the Reconfiguration Wizard, you must restore the domain by copying the files and directories from the backup location to the original domain directory. This is the only way to ensure that the domain has been returned to its original state prior to reconfiguration.

Note: In situations where you cannot run the Reconfiguration Wizard in GUI mode, Oracle recommends that you use a WLST script to reconfigure your domain. For more information, see Section 3.2.2, "Reconfiguring a WebLogic Domain Using WebLogic Scripting Tool."

To start the Reconfiguration Wizard in graphical mode from a Windows command prompt or on UNIX systems:

1. Log in to the system on which the domain resides.
2. Open an MS-DOS command prompt window (on Windows) or a command shell (on UNIX).
3. Go to the following directory:

   On Windows: *ORACLE_HOME*`\oracle_common\common\bin`

   On UNIX: *ORACLE_HOME*`/oracle_common/common/bin`

   where *ORACLE_HOME* is your Oracle home directory.
4. Execute the following command:

   On Windows: `reconfig.cmd`

   On UNIX: `sh reconfig.sh`

Note: When you run the `reconfig.cmd` or `reconfig.sh` command, the following error message might be displayed to indicate that the default cache directory is not valid:

```
*sys-package-mgr*: can't create package cache dir
```

You can change the cache directory by including the `-Dpython.cachedir=`*`valid_directory`* option in the command.

The Select Domain screen is displayed.

The Reconfiguration Wizard displays a sequence of screens, in the order listed in Table 3–1. For more information on each screen, refer to the related section in Chapter 5, "Reconfiguration Wizard Screens," or click the link in the **Screen** column.

Notes: Depending on the applications in your domain and other factors, additional configuration screens may also be displayed in addition to the screens shown in the following table. For information on these screens, click the **Help** button on the screen or refer to Chapter 5.

If the Advanced Configuration screen is displayed during reconfiguration, do not select any options to skip all advanced configuration. If necessary, you can use WLST, the Configuration Wizard or the Administration Console at a later time to perform advanced configuration such as adding additional servers and clusters or changing deployment targeting.

Table 3–1 Reconfiguring an Existing WebLogic Domain

Screen	When Does This Screen Appear?	Perform the Following Action
Select Domain	Always	Enter the full path to the domain directory or click **Browse** to navigate to and select the domain directory. Click **Next** to continue.
Reconfiguration Setup Progress	Always	Shows the progress of the application of reconfiguration templates. When the process completes, click **Next** to continue.
Domain Mode and JDK	Always	Domain mode cannot be changed. Select the JDK to use in the domain or click **Browse** to navigate to the JDK you want to use. Click **Next** to continue.
Additional domain configuration screens may appear at this point	Additional screens depend on the domain configuration	Click the **Help** button on the screen or refer to Chapter 5, which describes all of the screens in the order in which they are displayed.
Advanced Configuration	Always	Select the check box for each category (if any) for which you want to perform advanced configuration tasks The available check boxes depend on the domain configuration. Click **Next** to continue.
Additional domain configuration screens may appear at this point	Additional screens depend on the Advanced Configuration options you selected	Click the **Help** button on the screen or refer to Chapter 5, which describes all of the screens in the order in which they are displayed.

Table 3–1 (Cont.) Reconfiguring an Existing WebLogic Domain

Screen	When Does This Screen Appear?	Perform the Following Action
Configuration Summary	Always	Review the configuration. Click the **Back** button to change the configuration or click the **Reconfig** button to complete the domain reconfiguration process.
Reconfiguration Success	Always	Shows the final status of the reconfiguration process. Click **Finish** to exit the Configuration Wizard.

3.2.2 Reconfiguring a WebLogic Domain Using WebLogic Scripting Tool

This section describes how to reconfigure a domain using WebLogic Scripting Tool (WLST) in offline mode, using the `readDomainForUpgrade` command.

Caution: Once the domain reconfiguration process starts, it is irreversible. Prior to running using WLST to reconfigure the domain, ensure that you have backed up the domain as described in Section 3.1.3, "Backing Up the Domain." If an error or other interruption occurs during reconfiguration, you must restore the domain by copying the files and directories from the backup location to the original domain directory. This is the only way to ensure that the domain has been returned to its original state prior to reconfiguration.

The following example script shows how to reconfigure a domain called `my_domain` using WLST offline.

Example 3–1 Reconfiguring a WebLogic Domain

```
# Open the domain for upgrade.
wls:/offline> readDomainForUpgrade('c:/domains/my_domain')

# Save the updated domain.
wls:/offline/my_domain> updateDomain()

# Close the domain.
wls:/offline/my_domain> closeDomain()
```

3.2.3 Completing the Node Manager Configuration

After reconfiguring your domains, perform the following steps to complete the Node Manager configuration:

Note: The following steps describe how to configure Node Manager to run as it did in your previous installation, using one Node Manager to control all of your domains.

---

1. In the new WebLogic Server installation, create the `nodemanager` directory in *ORACLE_HOME*/oracle_common/common.

2. Copy the nodemanager.properties and nodemanager.domains files from the *WL_HOME*/common/nodemanager directory of your previous WebLogic Server installation to the directory you created in Step 1.
3. If your previous installation includes an nm_data.properties, SerializedSystemIni.data, or security/SerializedSystemIni.dat file, copy it to the directory you created in Step 1. If copying SerializedSystemIni.dat, you must create a `security` directory under the `nodemanager` directory and store the file there.
4. Make the following edits to the nodemanager.properties file, where *ORACLE_HOME* is the Oracle home directory for your WebLogic Server 12.1.2 installation:
   - Update `DomainsFile` to point to *ORACLE_HOME*/oracle_common/common/nodemanager/nodemanager.domains file.
   - If the file contains a `javaHome` property setting, remove it as it is not needed.
   - Update `JavaHome` to point to the `jre` directory for the JDK that you are using for WebLogic Server 12.1.2.
   - Update `NodeManagerHome` to point to *ORACLE_HOME*/oracle_common/common/nodemanager.
   - Update `LogFile` to point to *ORACLE_HOME*/oracle_common/common/nodemanager/nodemanager.log
5. If you are using your own security certificates, verify that the location of those certificates is correct in nodemanager.properties. You may have to update the path if you moved the certificates to another location.

   If you were using the WebLogic Server demo certificate in your previous installation, you must run `CertGen` to create a demo keystore for this installation:

   a. Run `setWLSEnv`:

      `cd` *`WL_HOME`*`/server/bin`

      `. ./setWLSEnv.sh` (UNIX)

      `setWLSEnv.cmd` (Windows)

      Note: On UNIX operating systems, the `setWLSEnv.sh` command does not set the environment variables in all command shells. Oracle recommends that you execute this command using the Korn shell or bash shell.

   b. Change to the *ORACLE_HOME*/oracle_common/common/nodemanager/ directory and create a security directory if it does not already exist.

   c. Change to the security directory and enter the following command:

      ```
      java utils.CertGen -certfile democert -keyfile demokey
      -keyfilepass DemoIdentityPassPhrase
      ```

   d. Enter the following command to generate the DemoIdentity.jks file:

      ```
      java utils.ImportPrivateKey -certfile democert.pem
      -keyfile demokey.pem -keyfilepass DemoIdentityPassPhrase
      -keystore DemoIdentity.jks -storepass
      DemoIdentityKeyStorePassPhrase -alias demoidentity
      ```

6. From the *ORACLE_HOME*/wlserver/server/bin directory, run `startNodeManager.cmd` (Windows) or `startNodeManager.sh` (UNIX).
7. Verify that you can start servers using Node Manager. For more information, see "Using Node Manager to Control Servers" in *Administering Node Manager for Oracle WebLogic Server*. To ensure that your `permgen` settings are adequate for starting the servers, you can use any one of the following methods:
   - Start the Managed Servers using the `startManagedWebLogic` script.
   - Set the `StartScriptEnabled` value in `nodemanager.properties` to `true`, which causes the `StartManagedWebLogic` script to be invoked when starting Managed Servers.
   - Set `permgen` space as described in Section A.5.2, "Setting permgen space."
   - Use a `setUserOverrides` script to specify `permgen` settings for server startup. For more information, see "Customizing Domain Wide Server Parameters" in *Administering Server Startup and Shutdown for Oracle WebLogic Server*.

3.3 Updating a Managed Server Domain on a Remote Machine

If your WebLogic domain contains multiple Managed Servers, and each Managed Server domain directory is located on a remote machine on which the Administration Server does not reside, you can use either of the following methods to update the domain on the remote machine:

- Run the `pack -managed=true` command to generate the domain template JAR, move the JAR to the remote machine, and then use `unpack` on the remote machine to create the Managed Server domain. For more information, see *Creating Templates and Domains Using the Pack and Unpack Commands*.
- Use the WLST `writeTemplate` command in online mode. When you execute the `writeTemplate` command while connected to the Administration Server from a remote machine, it dynamically packs the domain on the Administration Server into a template JAR file and transfers the template JAR to the specified directory.

  The following sample WLST script demonstrates how to use `writeTemplate` to create or update a Managed Server domain on a remote machine. Run the script on each remote machine in the domain. This script:
  - logs in to the Administration Server
  - packs the Administration Server domain into a JAR file and writes it to the specified template directory on the remote machine.
  - disconnects from the Administration Server
  - reads the template JAR
  - creates the domain on the remote machine

```
import os

wlsHome = os.getenv('WL_HOME')
mwHome = os.path.join(wlsHome, '..')

#Substitute the administrator user name and password values below as needed
connect('adminuser','adminpassword','admin_server_url')

#Create the path on the local machine where the template will be stored,
#The specified template JAR should not already exist. The timeout value
```

```
#specifies the number of milliseconds to elapse before the connection between
#the Administration Server and remote machine times out (default is 120000).
templatePath = '/user_templates/myTemplate.jar'
timeout = '180000'

#get the packed template from the Administration Server
writeTemplate(templatePath, timeout)

#disconnect from online WLST connection to the Administration Server
disconnect()

#read the template that was downloaded from the Administration Server
readTemplate(templatePath)

#specify the domain directory where the domain needs to be created
domainPath = 'domains/myDomain')

#create the domain
writeDomain(domainPath)
```

3.4 Important Notes About the Domain Upgrade Process

Please note the following important information about the upgrade process:

- It is not necessary for WebLogic Server applications to be undeployed. In most cases, WebLogic Server applications can be run without modifications in the new WebLogic Server 12.1.2 application environment. Review the compatibility information in Appendix A, "WebLogic Server 12.1.2 Compatibility with Previous Releases," to determine whether any features changes affect the applications in your environment. Note that if APIs that have been deprecated or removed are used in the application, then you may encounter warnings or exceptions at run time.
- At a minimum, the domain directory must contain the following files:
  - `config.xml`
  - Security-related files, including `SerializedSystemIni.dat`, `DefaultAuthenticatorInit.ldift`, `DefaultAuthorizerInit.ldift`, and `DefaultRoleMapperInit.ldift`

    If the security-related files are not available, the server fails to start and an authentication error message is logged.
  - Any transaction log (`.tlog`) files that reside in the domain. For more information, see "Transaction Log Files" in *Developing JTA Applications for Oracle WebLogic Server*.
- All contents of the domain directory on the target computer are updated during this process.
- You must upgrade the domain on every computer in the application environment.
- The reconfiguration wizard does not upgrade your own applications that may exist in the domain during a WebLogic domain upgrade.
- Domains that contain resources for WebLogic Liquid Data, or AquaLogic Data Services Platform cannot be upgraded to WebLogic Server 12.1.2.

- When you upgrade a domain on a remote Managed Server, a message similar to the following may be displayed to indicate that the referenced application path does not reside on the system:

  ```
  <Apr 12, 2009 6:42:06 PM EDT> <INFO> <Upgrade> <BEA-800000> <An invalid
  path, 'C:\bea\wlserver_10.3\user_projects\mydomain\medrecEar.ear', was
  specified for application, 'medrecEar'.>
  ```

  This message can be ignored.

- If you upgraded the Avitek Medical Records application from 8.1 to 12.1.2 on a Solaris computer (only), before starting the server, you must edit the setDomainEnv.sh file to remove `-Xverify:none` from the start command by setting `JAVA_OPTIONS=""` after the following line:

  ```
  . ${WL_HOME}/common/bin/commEnv.sh
  ```

  Otherwise, the server start fails with a JVM error.

3.5 Completing Post-Upgrade Tasks

After you upgrade the application environment, it may be necessary to perform the following tasks:

- Re-apply Customizations to Startup Scripts
- Verify File Permissions
- Verify Remote Server Startup Options
- Promote the Application Environment to Production

Not all of these steps are required for all situations. Review the sections to determine which, if any, of these steps are appropriate for your environment. In addition, you should review the compatibility issues in Appendix A, "WebLogic Server 12.1.2 Compatibility with Previous Releases," to determine if any of the compatibility issues apply to your environment].

3.5.1 Re-apply Customizations to Startup Scripts

To complete the upgrade of your application environment to 12.1.2, it might be necessary to re-apply any customizations to startup scripts. The following sections describe how to customize the default startup scripts as well as any custom startup scripts.

3.5.1.1 Default Startup Scripts

The Reconfiguration Wizard does not carry forward any customizations that have been made to the default startup scripts, such as the setting of the `JAVA_OPTIONS` environment variable. After the upgrade process is complete, you must customize the default scripts again.

If you are upgrading your domain to 12.1.2 and you want to continue using PointBase, you must add the PointBase JAR files to the beginning of the CLASSPATH environment variable definition. To do so, update the `set CLASSPATH` statement in your setDomainEnv files.

Note: WebLogic Server 12.1.2 supports PointBase 5.7; however, the use of any version of PointBase with WebLogic Server 10.3.3 or later requires a PointBase license, available at `http://www.pointbase.com`.

3.5.1.2 Custom Startup Scripts

If you have created custom startup scripts, you must update them manually, as follows:

- Set the JDK version to the JDK that you are using with WebLogic Server.
- Update the CLASSPATH variable, as follows:
  - Add WebLogic Server 12.1.2 classes to the beginning of the variable.
  - Remove all *unused* pre-10.3 WebLogic classes.
  - To continue using PointBase, include the PointBase database JARs at the beginning of the CLASSPATH environment variable definition.

    For more information about upgrading a domain that uses an evaluation database, see Section 3.6, "Upgrading a Domain that Uses an Evaluation Database."

3.5.2 Verify File Permissions

Verify the file permissions, as follows:

- If you backed up the domain directory as part of the upgrade, you must make your backup files secure because they might contain confidential information.
- During the upgrade process, file permissions are not preserved. If nondefault file permissions are set on files, they must be verified and reset.
- On a UNIX system, ownership and permissions for any new files created during the upgrade process are assigned to the user performing the upgrade. For example, if the upgrade is performed by root, then root is assigned ownership of any new files. As a result, any user who subsequently wants to update these files in the domain must have root privileges. You may want to review or modify the permissions on files created during the upgrade process.

3.5.3 Verify Remote Server Startup Options

When you start the Administration Server, verify that the remote server start options, such as `JAVA_HOME`, `BEA_HOME`, and `CLASSPATH`, reference the WebLogic Server 12.1.2 installation on the target Managed Server. This can be accomplished using the Administration Console, as described in "Configure startup arguments for Managed Servers" in *Oracle WebLogic Server Administration Console Online Help*.

Note: If the remote server startup options are not set correctly, when attempting to start a Managed Server using Node Manager, messages similar to the following may be written to the log file. Because these messages may be sent recursively, they may eventually consume all space available on the drive.

```
No config.xml was found.
```

```
Would you like the server to create a default configuration and boot? (y/n):
java.io.IOException: The handle is invalid
```

3.5.4 Promote the Application Environment to Production

Execute standard procedures for quality assurance and performance tuning before promoting an application environment to production. You should test the execution of your applications (including external client applications) in your test application environment. If your applications use APIs that have been deprecated or removed, then you may encounter warnings or exceptions at run time. If you do, you can make any required modifications before promoting your applications to production.

When all test criteria have been met, you can promote the application environment to production, as outlined in your upgrade plan (defined previously in Section 2.1.4, "Step 4: Create an Upgrade Plan").

When the new 12.1.2 application environment is deployed into production, you can start redirecting requests to the new environment from the existing environment. Gradually, you can bring the existing environment to a safe state for shutdown. This might be accomplished using a load balancer, for example.

3.6 Upgrading a Domain that Uses an Evaluation Database

As of WebLogic 10.3.3, the evaluation database that is available from the installation program is changed from PointBase to Derby. The evaluation database is provided for use by the sample applications and code examples and may be used as a demonstration database. If you are upgrading an examples or demonstration domain that was originally based on PointBase, you have the option to continue using PointBase in the domain.

To continue using PointBase as the database for a domain being upgraded to WebLogic 12.1.2, complete the following steps:

1. When installing WebLogic Server 12.1.2, as described in Section 2.2, "Prepare to Upgrade," you must use the full installer. The full installer does not preserve the PointBase installation from the previous WebLogic Server installation.
2. Complete the steps described in Section 2.3, "Upgrade Your Application Environment," and Section 3.5, "Completing Post-Upgrade Tasks."

   The domain's configuration settings for PointBase are preserved.
3. Obtain a PointBase license, available from `http://www.pointbase.com`.
4. Restore your PointBase installation. PointBase is installed in the *WL_HOME*/common/eval/pointbase directory.
5. Add the PointBase database JARs at the beginning of the CLASSPATH environment variable definition.

4 Upgrading WebLogic Web Services

The following sections describe the procedures for upgrading WebLogic Web services from WebLogic Server 9.x or 10.x to the WebLogic Server 12.1.*x* release. It also describes how to upgrade 8.1 WebLogic Web services to 12.1.*x* WebLogic Web services.

Topics include:

- Section 4.1, "Upgrading a 10.3.x RESTful Web Service (JAX-RS) to 12.1.x"
- Section 4.2, "Upgrading a 10.x WebLogic Web Service (JAX-WS or JAX-RPC) to 12.1.x"
- Section 4.3, "Upgrading an 8.1 WebLogic Web Service to 12.1.x"

Note: 10.3.*x* WebLogic Web services (JAX-WS or JAX-RPC) will continue to run, without any changes, on version 12.1.*x* of WebLogic Server because the associated Web services run time is still supported in this release, although they are deprecated and will be removed from the product in future releases. For this reason, Oracle highly recommends that you follow the instructions in this chapter to upgrade your 10.3.*x* Web service to 12.1.*x*.

4.1 Upgrading a 10.3.*x* RESTful Web Service (JAX-RS) to 12.1.*x*

In 10.3.*x*, a set of pre-built shared libraries were delivered with WebLogic Server to support Jersey 1.9 and 1.1.5.1 JAX-RS RI. In order to use the shared libraries, you needed to register them with the WebLogic Server instance, and modify the `web.xml` and `weblogic.xml` deployment descriptors to use the Jersey servlet and reference the shared libraries, respectively.

The following sections described the steps required to upgrade your environment so that the 10.3.*x* RESTful Web services run in the 12.1.*x* environment.

- Section 4.1.1, "Upgrading a 10.3.x RESTful Web Service That Uses the Jersey 1.9 Shared Libraries."
- Section 4.1.2, "Upgrading a 10.3.x RESTful Web Service That Uses the Jersey 1.1.5.1 Shared Libraries."

4.1.1 Upgrading a 10.3.*x* RESTful Web Service That Uses the Jersey 1.9 Shared Libraries

If your 10.3.*x* RESTful WebLogic Web service uses Jersey 1.9 shared libraries, then no additional steps are required to run the application in the 12.1.*x* environment. In this case, the Jersey 1.9 JAR file that is present in the system classpath is loaded by the WebLogic Server system classloader.

Note: If the RESTful Web service application is packaged with a servlet, ensure that it is packaged appropriately based on whether your Web application is using Servlet 3.0 or earlier. For more information, see "Packaging With a Servlet" in *Developing and Securing RESTful Web Services for Oracle WebLogic Server*.

4.1.2 Upgrading a 10.3.*x* RESTful Web Service That Uses the Jersey 1.1.5.1 Shared Libraries

Note: The Jersey 1.1.5.1 JAX-RS RI shared libraries are provided for backwards compatibility only. It is recommended that you upgrade your RESTful Web service applications to use the Jersey 1.9 JAX-RS RI APIs that are available on the WebLogic Server system classpath.

For backwards compatibility, the Jersey 1.1.5.1 JAX-RS RI shared libraries are delivered with the 12.1.*x* release of WebLogic Server. The shared libraries are located in the following directory: `WL_HOME/common/deployable-libraries`.

Table 4–1 summarizes the pre-built shared libraries that support Jersey JAX-RS RI Version 1.1.5.1 Web services, organized by the functionality that they support. The table also indicates whether the shared library is required or optional.

Table 4–1 Shared Libraries for Jersey JAX-RS RI 1.1.5.1

Functionality	Description	Required?
Jersey	■ Shared Library Name: jersey-bundle ■ JAR Filename: jersey-bundle-1.1.5.1.jar ■ WAR Filename: jersey-bundle-1.1.5.1.war ■ Version: 1.1.5.1 ■ License: SUN CDDL+GPL	Required
JAX-RS API	■ Shared Library Name: jsr311 ■ JAR Filename: jsr311-api-1.1.1.jar ■ WAR Filename: jsr311-api-1.1.1.war ■ Version: 1.1.1 ■ License: JSR311 license	Required
JSON processing	■ Shared Library Name: jackson-core-asl ■ JAR Filename: jackson-core-asl-1.1.1.jar ■ WAR Filename: jackson-core-asl-1.1.1.war ■ Version: 1.1.1 ■ License: Apache 2.0	Optional

Table 4–1 (Cont.) Shared Libraries for Jersey JAX-RS RI 1.1.5.1

Functionality	Description	Required?
JSON processing	■ Shared Library Name: jackson-jaxrs ■ JAR Filename: jackson-jaxrs-1.1.1.jar ■ WAR Filename: jackson-jaxrs-1.1.1.war ■ Version: 1.1.1 ■ License: Apache 2.0	Optional
JSON processing	■ Shared Library Name: jackson-mapper-asl ■ JAR Filename: jackson-mapper-asl-1.1.1.jar ■ WAR Filename: jackson-mapper-asl-1.1.1.war ■ Version: 1.1.1 ■ License: Apache 2.0	Optional
JSON streaming	■ Shared Library Name: jettison ■ JAR Filename: jettison-1.1.jar ■ WAR Filename: jettison-1.1.war ■ Version: 1.1 ■ License: Apache 2.0	Optional
ATOM processing	■ Shared Library Name: rome ■ JAR Filename: rome-1.0.jar ■ WAR Filename: rome-1.0.war ■ Version: 1.0 ■ License: Apache 2.0	Optional

To upgrade your environment so that the 10.3.*x* RESTful Web services run in the 12.1.*x* environment and continue to use the Jersey 1.1.5.1 shared libraries, you need to perform the following steps:

1. Register the Jersey 1.1.5.1 shared libraries with one or more WebLogic Server instances.

   Shared Java EE libraries are registered with one or more WebLogic Server instances by deploying them to the target servers and indicating that the deployments are to be shared. Shared Java EE libraries must be targeted to the same WebLogic Server instances you want to deploy applications that reference the libraries.

   The following shows an example of how to deploy the shared libraries that provide support for the basic Jersey JAX-RS RI functionality and JAX-RS API.

   ```
   weblogic.Deployer -verbose -noexit -source C:\myinstall\wlserver_
   10.3\common\deployable-libraries\jersey-bundle-1.1.5.1.war -targets myserver
   -adminurl t3://localhost:7001 -user system -password ******** -deploy -library

   weblogic.Deployer -verbose -noexit -source C:\myinstall\wlserver_
   10.3\common\deployable-libraries\jsr311-api-1.1.1.war -targets myserver
   -adminurl t3://localhost:7001 -user system -password ******** -deploy -library
   ```

   For more information about the weblogic.Deployer, see "weblogic.Deployer Command-Line Reference" in *Deploying Applications to Oracle WebLogic Server*.

2. If the RESTful Web service application is packaged with a servlet, ensure that it is packaged appropriately based on whether your Web application is using Servlet

3.0 or earlier. For more information, see "Packaging With a Servlet" in *Developing and Securing RESTful Web Services for Oracle WebLogic Server*.

3. Deploy your RESTful Web service application.

4.2 Upgrading a 10.x WebLogic Web Service (JAX-WS or JAX-RPC) to 12.1.*x*

No steps are required to upgrade a 10.*x* WebLogic Web service to 12.1.*x*; you can redeploy the JAX-WS or JAX-RPC Web service to WebLogic Server 12.1.*x* without making any changes or recompiling.

4.3 Upgrading an 8.1 WebLogic Web Service to 12.1.*x*

In the 12.1.2 release, the 8.1 WebLogic Web services run time has been removed. If you are using 8.1 WebLogic Web services, you need to upgrade the 8.1 WebLogic Web service applications to Web service stacks available in this release. The 8.1 WebLogic Web services rely on Apache XMLBeans for the purpose of mapping XML elements in SOAP payloads into Java objects and vice versa. XMLBeans are not supported in 12.1.2. The following upgrade paths are available:

- Upgrade to the WebLogic JAX-RPC stack: This is the simplest upgrade path in terms of level of effort required for upgrade. XMLBeans support in the WebLogic JAX-RPC stack is compatible with 8.1 WebLogic Web services. For more information, see "Upgrading an 8.1 WebLogic Web Service to the WebLogic JAX-RPC Stack" on page 4-4.
- Upgrade to the JAX-WS stack: This is the best upgrade path from the standpoint of taking advantage of latest technologies and standard support in WebLogic Server. This path requires a manual upgrade process, and the level of effort is determined by the nature of the existing 8.1 Web service applications. For example, if the applications have little XMLBeans usage, then the upgrade process is relatively easy. For 8.1 Web Service applications with heavy XMLBeans dependencies, you must modify the business logic in the service implementation in order to use JAXB classes instead of XMLBeans classes. Please note that JAX-WS does not support RPC-encoded style, and 8.1 Web Service applications with RPC-encoded style will need to adopt more interoperable literal style service contracts. For more information, see "Upgrading an 8.1 WebLogic Web Service to the WebLogic JAX-WS Stack" on page 4-20.

4.3.1 Upgrading an 8.1 WebLogic Web Service to the WebLogic JAX-RPC Stack

This section describes how to upgrade an 8.1 WebLogic Web service to use the WebLogic JAX-RPC run-time environment. The WebLogic JAX-RPC run time is based on the *JSR 109: Implementing Enterprise Web Services* specification at `http://www.jcp.org/en/jsr/detail?id=109`. The 12.1.*x* programming model uses standard JDK metadata annotations, as specified by the *JSR 181: Web Services Metadata for the Java Platform* specification (JSR-181) at `http://www.jcp.org/en/jsr/detail?id=181`.

Upgrading your 8.1 Web service includes the following high-level tasks; the procedures in later sections go into more detail:

- Update the 8.1 Java source code of the Java class or stateless session EJB that implements the Web service so that the source code uses JWS annotations.

In 12.1.*x*, WebLogic Web services are implemented using JWS files, which are Java files that contains JWS annotations. The `jwsc` Ant Task always implements the Web service as a plain Java file unless you explicitly implement `javax.ejb.SessionBean` in your JWS file. This latter case is not typical. This programming model differs from that of 8.1, where you were required to specify the type of back-end component (Java class or EJB).

- Update the Ant build script that builds the Web service to call the 12.1.*x* WebLogic Web service Ant task `jwsc` instead of the 8.1 `servicegen` task.

In the sections that follow it is assumed that:

- You previously used `servicegen` to generate your 8.1 Web service and that, more generally, you use Ant scripts in your development environment to iteratively develop Web services and other Java Platform, Enterprise Edition (Java EE) Version 5 artifacts that run on WebLogic Server. The procedures in this section direct you to update existing Ant `build.xml` files.
- You have access to the Java class or EJB source code for your 8.1 Web service.

This section does *not* discuss the following topics:

- Upgrading a JMS-implemented 8.1 Web service, because the WebLogic Web services JAX-RPC run time does not support JMS-implemented services.
- Upgrading Web services from versions previous to 8.1.
- Upgrading a client application that invokes an 8.1 Web service to one that invokes a 12.1.*x* Web service. For details on how to write a client application that invokes a 12.1.*x* Web service, see "Invoking Web Services" in *Developing JAX-RPC Web Services for Oracle WebLogic Server*.

4.3.1.1 Upgrading an 8.1 Java Class-Implemented WebLogic Web Service to 12.1.*x*: Main Steps

To upgrade an 8.1 Java class-implemented Web service to use the WebLogic Web services JAX-RPC run time:

1. Open a command window and set your WebLogic Server 12.1.*x* environment by executing the `setDomainEnv.cmd` (Windows) or `setDomainEnv.sh` (UNIX) script, located in the `bin` subdirectory of your 12.1.*x* domain directory.

   The default location of WebLogic Server domains is *`ORACLE_HOME`*`/user_projects/domains/`*`domainName`*, where *`ORACLE_HOME`* is the Oracle home directory you specified at installation and *`domainName`* is the name of your domain.

2. Create a project directory:

   ```
   prompt> mkdir /myExamples/upgrade_pojo
   ```

3. Create a `src` directory under the project directory, as well as sub-directories that correspond to the package name of the new 12.1.*x* JWS file (shown later in this procedure) that corresponds to the old 8.1 Java class:

   ```
   prompt> cd /myExamples/upgrade_pojo
    prompt> mkdir src/examples/webservices/upgrade_pojo
   ```

4. Copy the old Java class that implements the 8.1 Web service to the `src/examples/webservices/upgrade_pojo` directory of the working directory. Rename the file, if desired.

5. Edit the Java file, as described in the following steps. See the old and new sample Java files in Section 4.3.1.1.1, "Example of an 8.1 Java File and the Corresponding 12.1.x JWS File" for specific examples.

   a. If needed, change the package name and class name of the Java file to reflect the new 12.1.*x* source environment.

   b. Add `import` statements to import both the standard and WebLogic-specific JWS annotations.

   c. Add, at a minimum, the following JWS annotation:

      – The standard `@WebService` annotation at the Java class level to specify that the JWS file implements a Web service.

      Oracle recommends you also add the following annotations:

      – The standard `@SOAPBinding` annotation at the class-level to specify the type of Web service, such as document-literal-wrapped or RPC-encoded.

      – The WebLogic-specific `@WLHttpTransport` annotation at the class-level to specify the context and service URIs that are used in the URL that invokes the deployed Web service.

      – The standard `@WebMethod` annotation at the method-level for each method that is exposed as a Web service operation.

      See "Programming the JWS File" in *Developing JAX-RPC Web Services for Oracle WebLogic Server* for general information about using JWS annotations in a Java file.

   d. You might need to add additional annotations to your JWS file, depending on the 8.1 Web service features you want to carry forward to 12.1.*x*. In 8.1, many of these features were configured with attributes of `servicegen`. See Section 4.3.1.3, "Mapping of servicegen Attributes to JWS Annotations or jwsc Attributes" for a table that lists equivalent JWS annotation, if available, for features you enabled in 8.1 using `servicegen` attributes.

6. Copy the old `build.xml` file that built the 8.1 Web service to the 12.1.*x* working directory.

7. Update your Ant `build.xml` file to execute the `jwsc` Ant task, along with other supporting tasks, instead of `servicegen`.

   Oracle recommends that you create a new target, such as `build-service`, in your Ant build file and add the `jwsc` Ant task call to compile the new JWS file you created in the preceding steps. Once this target is working correctly, you can remove the old `servicegen` Ant task.

   The following procedure lists the main steps to update your `build.xml` file; for details on the steps, see the standard iterative development process outlined in "Developing WebLogic Web Services" in *Developing JAX-RPC Web Services for Oracle WebLogic Server*.

   See Section 4.3.1.1.2, "Example of an 8.1 and Updated 12.1.x Ant Build File for Java Class-Implemented Web Services" for specific examples of the steps in the following procedure.

   1. Add the `jwsc` taskdef to the `build.xml` file.

   2. Create a `build-service` target and add the tasks needed to build the 12.1.*x* Web service, as described in the following steps.

3. Add the `jwsc` task to the build file. Set the `srdir` attribute to the src directory (`/myExamples/upgrade_pojo/src`, in this example) and the `destdir` attribute to the root Enterprise application directory you created in the preceding step.

   Set the `file` attribute of the `<jws>` child element to the name of the new JWS file, created earlier in this procedure.

   You may need to specify additional attributes to the `jwsc` task, depending on the 8.1 Web service features you want to carry forward to 12.1.*x*. In 8.1, many of these features were configured using attributes of `servicegen`. See Section 4.3.1.3, "Mapping of servicegen Attributes to JWS Annotations or jwsc Attributes" for a table that describes if there is an equivalent `jwsc` attribute for features you enabled using `servicegen` attributes.

8. Execute the `build-service` Ant target. Assuming all the tasks complete successfully, the resulting Enterprise application contains your upgraded 12.1.*x* Web service.

See "Deploying and Undeploying WebLogic Web Services" and "Browsing to the WSDL of the Web Service" in *Developing JAX-RPC Web Services for Oracle WebLogic Server* for additional information about deploying and testing your Web service.

Based on the sample Java code shown in the following sections, the URL to invoke the WSDL of the upgraded 12.1.*x* Web service is:

```
http://host:port/upgradePOJO/HelloWorld?WSDL
```

4.3.1.1.1 Example of an 8.1 Java File and the Corresponding 12.1.*x* JWS File Assume that the following sample Java class implemented a 8.1 Web service:

```
package examples.javaclass;
/**
  * Simple Java class that implements the HelloWorld Web service.   It takes
  * as input an integer and a String, and returns a message that includes these
  * two parameters.
  */
public final class HelloWorld81 {
  /**
    * Returns a text message that includes the integer and String input
    * parameters.
    *
    */
   public String sayHello(int num, String s) {
    System.out.println("sayHello operation has been invoked with arguments " + s +
" and " + num);
    String returnValue = "This message brought to you by the letter "+s+" and the
number "+num;
    return returnValue;
   }
}
```

An equivalent JWS file for a 12.1.*x* Java class-implemented Web service is shown below, with the differences shown in bold. Note that some of the JWS annotation values are taken from attributes of the 8.1 `servicegen` Ant task shown in Section 4.3.1.1.2, "Example of an 8.1 and Updated 12.1.x Ant Build File for Java Class-Implemented Web Services."

```
package examples.webservices.upgrade_pojo;
// Import standard JWS annotations
import javax.jws.WebService;
 import javax.jws.WebMethod;
 import javax.jws.soap.SOAPBinding;
// Import WebLogic JWS anntoation
import weblogic.jws.WLHttpTransport;
/**
  * Simple Java class that implements the HelloWorld92 Web service.   It takes
  * as input an integer and a String, and returns a message that includes these
  * two parameters.
  */
@WebService(name="HelloWorld92PortType", serviceName="HelloWorld",
             targetNamespace="http://example.org")
@SOAPBinding(style=SOAPBinding.Style.DOCUMENT,
              use=SOAPBinding.Use.LITERAL,
              parameterStyle=SOAPBinding.ParameterStyle.WRAPPED)
@WLHttpTransport(contextPath="upgradePOJO", serviceUri="HelloWorld",
                  portName="HelloWorld92Port")
public class HelloWorld92Impl {
  /**
    * Returns a text message that includes the integer and String input
    * parameters.
    *
    */
  @WebMethod()
   public String sayHello(int num, String s) {
    System.out.println("sayHello operation has been invoked with arguments " + s +
" and " + num);
    String returnValue = "This message brought to you by the letter "+s+" and the
number "+num;
    return returnValue;
   }
 }
```

4.3.1.1.2 Example of an 8.1 and Updated 12.1.*x* Ant Build File for Java Class-Implemented Web Services

The following simple `build.xml` file shows the 8.1 way to build a WebLogic Web service using the `servicegen` Ant task; in the example, the Java file that implements the 8.1 Web service has already been compiled into the examples.javaclass.HelloWorld81 class:

```
<project name="javaclass-webservice" default="all" basedir=".">
  <!-- set global properties for this build -->
   <property name="source" value="."/>
   <property name="build" value="${source}/build"/>
   <property name="war_file" value="HelloWorldWS.war" />
   <property name="ear_file" value="HelloWorldApp.ear" />
   <property name="namespace" value="http://examples.org" />
  <target name="all" depends="clean, ear"/>
  <target name="clean">
     <delete dir="${build}"/>
   </target>
  <!-- example of old 8.1 servicegen call to build Web Service -->
  <target name="ear">
     <servicegen
        destEar="${build}/${ear_file}"
        warName="${war_file}">
        <service
          javaClassComponents="examples.javaclass.HelloWorld81"
          targetNamespace="${namespace}"
          serviceName="HelloWorld"
          serviceURI="/HelloWorld"
          generateTypes="True"
          expandMethods="True">
        </service>
      </servicegen>
   </target>
</project>
```

An equivalent `build.xml` file that calls the `jwsc` Ant task to build a 12.1.*x* Web service is shown below, with the relevant tasks discussed in this section in bold. In the example, the new JWS file that implements the 12.1.*x* Web service is called `HelloWorld92Impl.java`:

```
<project name="webservices-upgrade_pojo" default="all">
  <!-- set global properties for this build -->
  <property name="wls.username" value="weblogic" />
   <property name="wls.password" value="weblogic" />
   <property name="wls.hostname" value="localhost" />
   <property name="wls.port" value="7001" />
   <property name="wls.server.name" value="myserver" />
  <property name="ear.deployed.name" value="upgradePOJOEar" />
   <property name="example-output" value="output" />
   <property name="ear-dir" value="${example-output}/upgradePOJOEar" />
  <taskdef name="jwsc"
     classname="weblogic.wsee.tools.anttasks.JwscTask" />
  <taskdef name="wldeploy"
     classname="weblogic.ant.taskdefs.management.WLDeploy"/>
  <target name="all" depends="clean,build-service,deploy" />
  <target name="clean" depends="undeploy">
     <delete dir="${example-output}"/>
   </target>
  <target name="build-service">
    <jwsc
       srcdir="src"
       destdir="${ear-dir}">
      <jws file="examples/webservices/upgrade_pojo/HelloWorld92Impl.java" />
    </jwsc>
  </target>
  <target name="deploy">
     <wldeploy action="deploy" name="${ear.deployed.name}"
       source="${ear-dir}" user="${wls.username}"
       password="${wls.password}" verbose="true"
       adminurl="t3://${wls.hostname}:${wls.port}"
       targets="${wls.server.name}" />
   </target>
  <target name="undeploy">
     <wldeploy action="undeploy" name="${ear.deployed.name}"
       failonerror="false"
       user="${wls.username}" password="${wls.password}" verbose="true"
       adminurl="t3://${wls.hostname}:${wls.port}"
       targets="${wls.server.name}" />
   </target>
</project>
```

4.3.1.2 Upgrading an 8.1 EJB-Implemented WebLogic Web Service to 12.1.*x*: Main Steps

The following procedure describes how to upgrade an 8.1 EJB-implemented Web service to use the WebLogic Web services JAX-RPC run time.

The 12.1.*x* Web services programming model is quite different from the 8.1 model in that it hides the underlying implementation of the Web service. Rather than specifying up front that you want the Web service to be implemented by a Java class or an EJB, the `jwsc` Ant task always picks a plain Java class implementation, unless you have explicitly implemented `javax.ejb.SessionBean` in the JWS file, which is not typical. For this reason, the following procedure does not show how to import EJB classes or use EJBGen, even though the 8.1 Web service was explicitly implemented with an EJB. Instead, the procedure shows how to create a standard JWS file that is the 12.1.*x* equivalent to the 8.1 EJB-implemented Web service.

1. Open a command window and set your 12.1.*x* WebLogic Server environment by executing the `setDomainEnv.cmd` (Windows) or `setDomainEnv.sh` (UNIX) script, located in the `bin` subdirectory of your 12.1.*x* domain directory.

   The default location of WebLogic Server domains is *`ORACLE_HOME`*`/user_projects/domains/`*`domainName`*, where *`ORACLE_HOME`* is the Oracle home you specified at installation and *`domainName`* is the name of your domain.

2. Create a project directory:

   ```
   prompt> mkdir /myExamples/upgrade_ejb
   ```

3. Create a `src` directory under the project directory, as well as sub-directories that correspond to the package name of the new 12.1.*x* JWS file (shown later on in this procedure) that corresponds to your 8.1 EJB implementation:

   ```
   prompt> cd /myExamples/upgrade_ejb
    prompt> mkdir src/examples/webservices/upgrade_ejb
   ```

4. Copy the 8.1 EJB Bean file that implemented `javax.ejb.SessionBean` to the `src/examples/webservices/upgrade_ejb` directory of the working directory. Rename the file, if desired.

   Note: You do not need to copy over the 8.1 Home and Remote EJB files.

   ---

5. Edit the EJB Bean file, as described in the following steps. See the old and new sample Java files in Section 4.3.1.2.1, "Example of 8.1 EJB Files and the Corresponding 12.1.x JWS File" for specific examples.

   a. If needed, change the package name and class name of the Java file to reflect the new 12.1.*x* source environment.

   b. Optionally remove the `import` statements that import the EJB classes (`javax.ejb.*`). These classes are no longer needed in the upgraded JWS file.

   c. Add `import` statements to import both the standard and WebLogic-specific JWS annotations.

   d. Ensure that the JWS file does *not* implement `javax.ejb.SessionBean` anymore by removing the `implements SessionBean` code from the class declaration.

   e. Remove all the EJB-specific methods:

      – `ejbActivate()`

      – `ejbRemove()`

      – `ejbPassivate()`

      – `ejbCreate()`

   f. Add, at a minimum, the following JWS annotation:

- The standard `@WebService` annotation at the Java class level to specify that the JWS file implements a Web service.

  Oracle recommends you also add the following annotations:

- The standard `@SOAPBinding` annotation at the class-level to specify the type of Web service, such as document-literal-wrapped or RPC-encoded.
- The WebLogic-specific `@WLHttpTransport` annotation at the class-level to specify the context and service URIs that are used in the URL that invokes the deployed Web service.
- The standard `@WebMethod` annotation at the method-level for each method that is exposed as a Web service operation.

See "Programming the JWS File" in *Developing JAX-RPC Web Services for Oracle WebLogic Server* for general information about using JWS annotations in a Java file.

g. You might need to add additional annotations to your JWS file, depending on the 8.1 Web service features you want to carry forward to 12.1.*x*. In 8.1, many of these features were configured using attributes of `servicegen`. See Section 4.3.1.3, "Mapping of servicegen Attributes to JWS Annotations or jwsc Attributes" for a table that lists equivalent JWS annotation, if available, for features you enabled in 8.1 using `servicegen` attributes.

6. Copy the old `build.xml` file that built the 8.1 Web service to the 12.1.*x* working directory.
7. Update your Ant `build.xml` file to execute the `jwsc` Ant task, along with other supporting tasks, instead of `servicegen`.

   Oracle recommends that you create a new target, such as `build-service`, in your Ant build file and add the `jwsc` Ant task call to compile the new JWS file you created in the preceding steps. Once this target is working correctly, you can remove the old `servicegen` Ant task.

   The following procedure lists the main steps to update your `build.xml` file; for details on the steps, see the standard iterative development process outlined in

   See Section 4.3.1.2.6, "Example of an 8.1 and Updated 12.1.x Ant Build File for an 8.1 EJB-Implemented Web Service" for specific examples of the steps in the following procedure.

   a. Add the `jwsc` taskdef to the `build.xml` file.

   b. Create a `build-service` target and add the tasks needed to build the 12.1.*x* Web service, as described in the following steps.

   c. Add the `jwsc` task to the build file. Set the `srdir` attribute to the src directory (`/myExamples/upgrade_ejb/src`, in this example) and the `destdir` attribute to the root Enterprise application directory you created in the preceding step.

   Set the `file` attribute of the `<jws>` child element to the name of the new JWS file, created earlier in this procedure.

   You may need to specify additional attributes to the `jwsc` task, depending on the 8.1 Web service features you want to carry forward to 12.1.*x*. In 8.1, many of these features were configured using attributes of `servicegen`. See Section 4.3.1.3, "Mapping of servicegen Attributes to JWS Annotations or jwsc Attributes" for a table that indicates whether there is an equivalent `jwsc` attribute for features you enabled using `servicegen` attributes.

8. Execute the `build-service` Ant target. Assuming all tasks complete successfully, the resulting Enterprise application contains your upgraded 12.1.*x* Web service.

See "Deploying and Undeploying WebLogic Web Services" and "Browsing to the WSDL of the Web Service" in *Developing JAX-RPC Web Services for Oracle WebLogic Server* for additional information about deploying and testing your Web service.

Based on the sample Java code shown in the following sections, the URL to invoke the WSDL of the upgraded 12.1.*x* Web service is:

```
http://host:port/upgradeEJB/HelloWorldService?WSDL
```

4.3.1.2.1 Example of 8.1 EJB Files and the Corresponding 12.1.*x* JWS File

Assume that the Bean, Home, and Remote classes and interfaces, shown in the next three sections, implemented the 8.1 stateless session EJB which in turn implemented an 8.1 Web service.

The equivalent 12.1.*x* JWS file is shown in Section 4.3.1.2.5, "Equivalent 12.1.x JWS File." The differences between the 8.1 and 12.1.*x* classes are shown in bold. Note that some of the JWS annotation values are taken from attributes of the 8.1 `servicegen` Ant task shown in Section 4.3.1.2.6, "Example of an 8.1 and Updated 12.1.x Ant Build File for an 8.1 EJB-Implemented Web Service."

4.3.1.2.2 8.1 SessionBean Class

```
package examples.statelessSession;
import javax.ejb.CreateException;
 import javax.ejb.SessionBean;
 import javax.ejb.SessionContext;
/**
  * HelloWorldBean is a stateless session EJB.  It has a single method,
  * sayHello(), that takes an integer and a String and returns a String.
  * <p>
  * The sayHello() method is the public operation of the Web service based on
  * this EJB.
   */
public class HelloWorldBean81 implements SessionBean {
  private static final boolean VERBOSE = true;
   private SessionContext ctx;
  // You might also consider using WebLogic's log service
   private void log(String s) {
     if (VERBOSE) System.out.println(s);
   }
  /**
    *   Single EJB business method.
    */
   public String sayHello(int num, String s) {
    System.out.println("sayHello in the HelloWorld EJB has "+
       "been invoked with arguments " + s + " and " + num);
    String returnValue = "This message brought to you by the "+
       "letter "+s+" and the number "+num;
    return returnValue;
   }
  /**
    * This method is required by the EJB Specification,
    * but is not used by this example.
    *
    */
   public void ejbActivate() {
     log("ejbActivate called");
   }
  /**
```

```
    * This method is required by the EJB Specification,
    * but is not used by this example.
    *
    */
   public void ejbRemove() {
     log("ejbRemove called");
   }
  /**
    * This method is required by the EJB Specification,
    * but is not used by this example.
    *
    */
   public void ejbPassivate() {
     log("ejbPassivate called");
   }
  /**
    * Sets the session context.
    *
    * @param ctx SessionContext Context for session
    */
   public void setSessionContext(SessionContext ctx) {
     log("setSessionContext called");
     this.ctx = ctx;
   }
  /**
    * This method is required by the EJB Specification,
    * but is not used by this example.
    *
    */
   public void ejbCreate () throws CreateException {
     log("ejbCreate called");
   }
}
```

4.3.1.2.3 8.1 Remote Interface

```
package examples.statelessSession;
import java.rmi.RemoteException;
 import javax.ejb.EJBObject;
/**
  * The methods in this interface are the public face of HelloWorld.
  * The signatures of the methods are identical to those of the EJBean, except
  * that these methods throw a java.rmi.RemoteException.
   */
public interface HelloWorld81 extends EJBObject {
  /**
    * Simply says hello from the EJB
    *
    * @param num               int number to return
    * @param s                 String string to return
    * @return                  String returnValue
    * @exception               RemoteException if there is
    *                          a communications or systems failure
    */
   String sayHello(int num, String s)
     throws RemoteException;
 }
```

4.3.1.2.4 8.1 EJB Home Interface

```
package examples.statelessSession;
import java.rmi.RemoteException;
 import javax.ejb.CreateException;
 import javax.ejb.EJBHome;
```

```
/**
  * This interface is the Home interface of the HelloWorld stateless session EJB.
  */
 public interface HelloWorldHome81 extends EJBHome {
  /**
    * This method corresponds to the ejbCreate method in the
    * HelloWorldBean81.java file.
    */
   HelloWorld81 create()
     throws CreateException, RemoteException;
}
```

4.3.1.2.5 Equivalent 12.1.*x* JWS File The differences between the 8.1 and 12.1.*x* files are shown in bold. The value of some JWS annotations are taken from attributes of the 8.1 `servicegen` Ant task shown in Section 4.3.1.2.6, "Example of an 8.1 and Updated 12.1.x Ant Build File for an 8.1 EJB-Implemented Web Service."

```
package examples.webservices.upgrade_ejb;
// Import the standard JWS annotations
import javax.jws.WebMethod;
 import javax.jws.WebService;
 import javax.jws.soap.SOAPBinding;
// Import the WebLogic specific annotation
import weblogic.jws.WLHttpTransport;
// Class-level annotations
@WebService(name="HelloWorld92PortType", serviceName="HelloWorldService",
             targetNamespace="http://example.org")
@SOAPBinding(style=SOAPBinding.Style.DOCUMENT,
              use=SOAPBinding.Use.LITERAL,
              parameterStyle=SOAPBinding.ParameterStyle.WRAPPED)
@WLHttpTransport(contextPath="upgradeEJB", serviceUri="HelloWorldService",
                  portName="HelloWorld92Port")
/**
  * HelloWorld92Impl is the JWS equivalent of the HelloWorld81 EJB that
  * implemented the 8.1 Web Service.  It has a single method,
  * sayHello(), that takes an integer and a String and returns a String.
  */
public class HelloWorld92Impl {
  /** the sayHello method will become the public operation of the Web
    *  Service.
    */
  @WebMethod()
   public String sayHello(int num, String s) {
    System.out.println("sayHello in the HelloWorld92 Web Service has "+
       "been invoked with arguments " + s + " and " + num);
    String returnValue = "This message brought to you by the "+
       "letter "+s+" and the number "+num;
    return returnValue;
  }
}
```

4.3.1.2.6 Example of an 8.1 and Updated 12.1.*x* Ant Build File for an 8.1 EJB-Implemented Web Service The following simple `build.xml` file shows the 8.1 way to build an EJB-implemented WebLogic Web service using the `servicegen` Ant task. Following this example is an equivalent `build.xml` file that calls the `jwsc` Ant task to build a 12.1.*x* Web service.

```
<project name="ejb-webservice" default="all" basedir=".">
  <!-- set global properties for this build -->
   <property name="source" value="."/>
   <property name="build" value="${source}/build"/>
   <property name="ejb_file" value="HelloWorldWS.jar" />
   <property name="war_file" value="HelloWorldWS.war" />
   <property name="ear_file" value="HelloWorldApp.ear" />
   <property name="namespace" value="http://examples.org" />
  <target name="all" depends="clean,ear"/>
  <target name="clean">
     <delete dir="${build}"/>
   </target>
  <!-- example of old 8.1 servicegen call to build Web Service -->
  <target name="ejb">
     <delete dir="${build}" />
     <mkdir dir="${build}"/>
     <mkdir dir="${build}/META-INF"/>
     <copy todir="${build}/META-INF">
       <fileset dir="${source}">
         <include name="ejb-jar.xml"/>
       </fileset>
     </copy>
     <javac srcdir="${source}" includes="HelloWorld*.java"
            destdir="${build}" />
     <jar jarfile="${ejb_file}" basedir="${build}" />
     <wlappc source="${ejb_file}" />
   </target>
 <target name="ear" depends="ejb">
    <servicegen
       destEar="${build}/${ear_file}"
       warName="${war_file}">
       <service
         ejbJar="${ejb_file}"
         targetNamespace="${namespace}"
         serviceName="HelloWorldService"
         serviceURI="/HelloWorldService"
         generateTypes="True"
         expandMethods="True">
       </service>
     </servicegen>
  </target>
</project>
```

An equivalent `build.xml` file that calls the `jwsc` Ant task to build a 12.1.*x* Web service is shown below, with the relevant tasks discussed in this section in bold:

```
<project name="webservices-upgrade_ejb" default="all">
  <!-- set global properties for this build -->
  <property name="wls.username" value="weblogic" />
   <property name="wls.password" value="weblogic" />
   <property name="wls.hostname" value="localhost" />
   <property name="wls.port" value="7001" />
   <property name="wls.server.name" value="myserver" />
  <property name="ear.deployed.name" value="upgradeEJB" />
   <property name="example-output" value="output" />
   <property name="ear-dir" value="${example-output}/upgradeEJB" />
  <taskdef name="jwsc"
     classname="weblogic.wsee.tools.anttasks.JwscTask" />
  <taskdef name="wldeploy"
     classname="weblogic.ant.taskdefs.management.WLDeploy"/>
  <target name="all" depends="clean,build-service,deploy" />
  <target name="clean" depends="undeploy">
     <delete dir="${example-output}"/>
   </target>
```

```
    <target name="build-service">
      <jwsc
         srcdir="src"
         destdir="${ear-dir}">
        <jws file="examples/webservices/upgrade_ejb/HelloWorld92Impl.java" />
      </jwsc>
    </target>
    <target name="deploy">
       <wldeploy action="deploy" name="${ear.deployed.name}"
         source="${ear-dir}" user="${wls.username}"
         password="${wls.password}" verbose="true"
         adminurl="t3://${wls.hostname}:${wls.port}"
         targets="${wls.server.name}" />
     </target>
    <target name="undeploy">
       <wldeploy action="undeploy" name="${ear.deployed.name}"
         failonerror="false"
         user="${wls.username}" password="${wls.password}" verbose="true"
         adminurl="t3://${wls.hostname}:${wls.port}"
         targets="${wls.server.name}" />
     </target>
  </project>
```

4.3.1.3 Mapping of servicegen Attributes to JWS Annotations or jwsc Attributes

The following table maps the attributes of the 8.1 `servicegen` Ant task to their equivalent 12.1.*x* JWS annotation or `jwsc` attribute.

The attributes listed in the first column are a mixture of attributes of the main `servicegen` Ant task and attributes of the four child elements of `servicegen` (`<service>`, `<client>`, `<handlerChain>`, and `<security>`)

See "JWS Annotation Reference" and "jwsc" in the *WebLogic Web Services Reference for Oracle WebLogic Server* for more information about the 12.1.*x* JWS annotations and `jwsc` Ant task.

Table 4–2 Mapping of servicegen Attributes to JWS Annotations or jwsc Attributes

servicegen or Child Element of servicegen Attribute	Equivalent JWS Annotation or jwsc Attribute
`contextURI`	`contextPath` attribute of the WebLogic-specific `@WLHttpTransport` annotation. **Note**: Because this is a WebLogic-specific annotation, you can use it to generate *only* a JAX-RPC Web service, and not a JAX-WS Web service.
`destEAR`	`destdir` attribute of the `jwsc` Ant task.
`keepGenerated`	`keepGenerated` attribute of the `jwsc` Ant task.
`mergeWithExistingWS`	No equivalent.
`overwrite`	No equivalent.
`warName`	`name` attribute of the `<jws>` child element of the `jwsc` Ant task.
`ejbJAR` (attribute of the `service` child element)	No direct equivalent, because the `jwsc` Ant task generates Web service artifacts from a JWS file, rather than a compiled EJB or Java class. Indirect equivalent is the `file` attribute of the `<jws>` child element of the `jwsc` Ant task that specifies the name of the JWS file.

Table 4–2 (Cont.) Mapping of servicegen Attributes to JWS Annotations or jwsc Attributes

servicegen or Child Element of servicegen Attribute	Equivalent JWS Annotation or jwsc Attribute
`excludeEJBs` (attribute of the `service` child element)	No equivalent.
`expandMethods` (attribute of the `service` child element)	No equivalent.
`generateTypes` (attribute of the `service` child element)	No equivalent.
`ignoreAuthHeader` (attribute of the `service` child element)	No equivalent.
`includeEJBs` (attribute of the `service` child element)	No equivalent.
`javaClassComponents` (attribute of the `service` child element)	No direct equivalent, because the `jwsc` Ant task generates Web service artifacts from a JWS file, rather than a compiled EJB or Java class. Indirect equivalent is the `file` attribute of the `<jws>` child element of the `jwsc` Ant task that specifies the name of the JWS file.
`JMSAction` (attribute of the `service` child element)	No equivalent because JMS-implemented Web services are not supported in the 12.1.*x* release.
`JMSConnectionFactory` (attribute of the `service` child element)	No equivalent because JMS-implemented Web services are not supported in the 12.1.*x* release.
`JMSDestination` (attribute of the `service` child element)	No equivalent because JMS-implemented Web services are not supported in the 12.1.*x* release.
`JMSDestinationType` (attribute of the `service` child element)	No equivalent because JMS-implemented Web services are not supported in the 12.1.*x* release.
`JMSMessageType` (attribute of the `service` child element)	No equivalent because JMS-implemented Web services are not supported in the 12.1.*x* release.
`JMSOperationName` (attribute of the `service` child element)	No equivalent because JMS-implemented Web services are not supported in the 12.1.*x* release.
`protocol` (attribute of the `service` child element)	One of the following WebLogic-specific annotations: ■ `@WLHttpTransport` ■ `@WLJmsTransport` **Note**: Because these are WebLogic-specific annotations, you can use them to generate *only* a JAX-RPC Web service, and not a JAX-WS Web service.
`serviceName` (attribute of the `service` child element)	`serviceName` attribute of the standard `@WebService` annotation.
`serviceURI` (attribute of the `service` child element)	`serviceUri` attribute of the WebLogic-specific `@WLHttpTransport` or `@WLJmsTransport` annotations. **Note**: Because these are WebLogic-specific annotations, you can use them to generate *only* a JAX-RPC Web service, and not a JAX-WS Web service.

Table 4–2 (Cont.) Mapping of servicegen Attributes to JWS Annotations or jwsc Attributes

servicegen or Child Element of servicegen Attribute	Equivalent JWS Annotation or jwsc Attribute
`style` (attribute of `service` child element)	`style` attribute of the standard `@SOAPBinding` annotation.
`typeMappingFile` (attribute of the `service` child element)	No equivalent.
`targetNamespace` (attribute of the `service` child element)	`targetNamespace` attribute of the standard `@WebService` annotation.
`userSOAP12` (attribute of the `service` child element)	`value` attribute of the WebLogic-specific `@weblogic.jws.Binding` JWS annotation **Note**: Because this is a WebLogic-specific annotation, you can use it to generate *only* a JAX-RPC Web service, and not a JAX-WS Web service.
`clientJarName` (attribute of `client` child element)	No equivalent.
`packageName` (attribute of the `client` child element)	No direct equivalent. Use the `packageName` attribute of the `clientgen` Ant task to generate client-side Java code and artifacts.
`saveWSDL` (attribute of the `client` child element)	No equivalent.
`userServerTypes` (attribute of the `client` child element)	No equivalent.
`handlers` (attribute of the `handlerChain` child element)	Standard `@HandlerChain` or `@SOAPMessageHandlers` annotation.
`name` (attribute of the `handlerChain` child element)	Standard `@HandlerChain` or `@SOAPMessageHandlers` annotation.
`duplicateElimination` (attribute of the `reliability` child element)	No direct equivalent. Use WebLogic-specific `@Policy` attribute to specify a WS-Policy file that contains Web service reliable messaging policy assertions. See "Using Web Service Reliable Messaging" in *Developing JAX-RPC Web Services for Oracle WebLogic Server*.
`persistDuration` (attribute of the `reliability` child element)	No direct equivalent. Use WebLogic-specific `@Policy` attribute to specify a WS-Policy file that contains Web service reliable messaging policy assertions. In this release, the equivalent is valid for JAX-RPC Web services only. See "Using Web Service Reliable Messaging" in *Developing JAX-RPC Web Services for Oracle WebLogic Server*.
`enablePasswordAuth` (attribute of the `security` child element)	No direct equivalent. Use WebLogic-specific `@Policy` attribute to specify a WS-Policy file that contains message-level security policy assertions. **Note**: Because this is a WebLogic-specific annotation, you can use it to generate *only* a JAX-RPC Web service, and not a JAX-WS Web service. See "Configuring Message-Level Security" in *Securing WebLogic Web Services for Oracle WebLogic Server*.

Table 4–2 (Cont.) Mapping of servicegen Attributes to JWS Annotations or jwsc Attributes

servicegen or Child Element of servicegen Attribute	Equivalent JWS Annotation or jwsc Attribute
`encryptKeyName` (attribute of the `security` child element)	No direct equivalent. Use WebLogic-specific `@Policy` attribute to specify a WS-Policy file that contains message-level security policy assertions. **Note**: Because this is a WebLogic-specific annotation, you can use it to generate *only* a JAX-RPC Web service, and not a JAX-WS Web service. See "Configuring Message-Level Security" in *Securing WebLogic Web Services for Oracle WebLogic Server*.
`encryptKeyPass` (attribute of the `security` child element)	No direct equivalent. Use WebLogic-specific `@Policy` attribute to specify a WS-Policy file that contains message-level security policy assertions. **Note**: Because this is a WebLogic-specific annotation, you can use it to generate *only* a JAX-RPC Web service, and not a JAX-WS Web service. See "Configuring Message-Level Security" in *Securing WebLogic Web Services for Oracle WebLogic Server*.
`password` (attribute of the `security` child element)	No direct equivalent. Use WebLogic-specific `@Policy` attribute to specify a WS-Policy file that contains message-level security policy assertions. See "Configuring Message-Level Security" in *Securing WebLogic Web Services for Oracle WebLogic Server*.
`signKeyName` (attribute of the `security` child element)	No direct equivalent. Use WebLogic-specific `@Policy` attribute to specify a WS-Policy file that contains message-level security policy assertions. **Note**: Because this is a WebLogic-specific annotation, you can use it to generate *only* a JAX-RPC Web service, and not a JAX-WS Web service. See "Configuring Message-Level Security" in *Securing WebLogic Web Services for Oracle WebLogic Server*.
`signKeyPass` (attribute of the `security` child element)	No direct equivalent. Use WebLogic-specific `@Policy` attribute to specify a WS-Policy file that contains message-level security policy assertions. **Note**: Because this is a WebLogic-specific annotation, you can use it to generate *only* a JAX-RPC Web service, and not a JAX-WS Web service. See "Configuring Message-Level Security" in *Securing WebLogic Web Services for Oracle WebLogic Server*.
`username` (attribute of the `security` child element)	No direct equivalent. Use WebLogic-specific `@Policy` attribute to specify a WS-Policy file that contains message-level security policy assertions. **Note**: Because this is a WebLogic-specific annotation, you can use it to generate *only* a JAX-RPC Web service, and not a JAX-WS Web service. See "Configuring Message-Level Security" in *Securing WebLogic Web Services for Oracle WebLogic Server*.

4.3.2 Upgrading an 8.1 WebLogic Web Service to the WebLogic JAX-WS Stack

This section summarizes how to upgrade an 8.1 WebLogic Web service to use the WebLogic JAX-WS stack.

The WebLogic JAX-WS run time is based on the JAX-WS (The Java API for XML-Based Web Services) 2.2 specification and the Web Services for Java EE v1.3 (JSR 109) specifications. These define annotations that are used in a Java Web Service (JWS) source file to define a Web service. Ant tasks are then used to compile the JWS into a Java class and generate all the associated artifacts. The Java Web Service (JWS) is the core of your JAX-WS web service.

Upgrading your 8.1 Web service includes the following high-level tasks:

- Upgrade any Web service EJBs from 2.*x* to 3.*x*.

  JAX-WS supports EJB 3.0 and 3.*x*. It does not support EJB 2.*x*.

- Rewrite the 8.1 Web service class as a JAX-WS JWS file and map any proprietary 8.x features to similar JAX-WS features.

  Note that there is not a one-to-one correspondence between 8.1 Web service features and JAX-WS 12.1.*x* features.

- Update the Ant build script that builds the Web service to call the 12.1.*x* WebLogic Web service Ant task `jwsc` instead of the 8.1 `servicegen` task.
- Generate new JAX-WS clients using the JAX-WS `clientgen` Ant task.

JAX-WS Upgrade Considerations

Before upgrading to JAX-WS, you should consider the following:

- The JAX-WS specification supports the "document-literal" and "rpc-literal" styles, but not "rpc-encoded". If you require rpc-encoded, you should consider upgrading your 8.1 Web service to the JAX-RPC model, rather than JAX-WS.
- SOAP Arrays are not supported by JAX-WS; they are available in JAX-RPC.

For more information about JAX-WS, refer to *Developing JAX-WS Web Services for Oracle WebLogic Server*.

5 Reconfiguration Wizard Screens

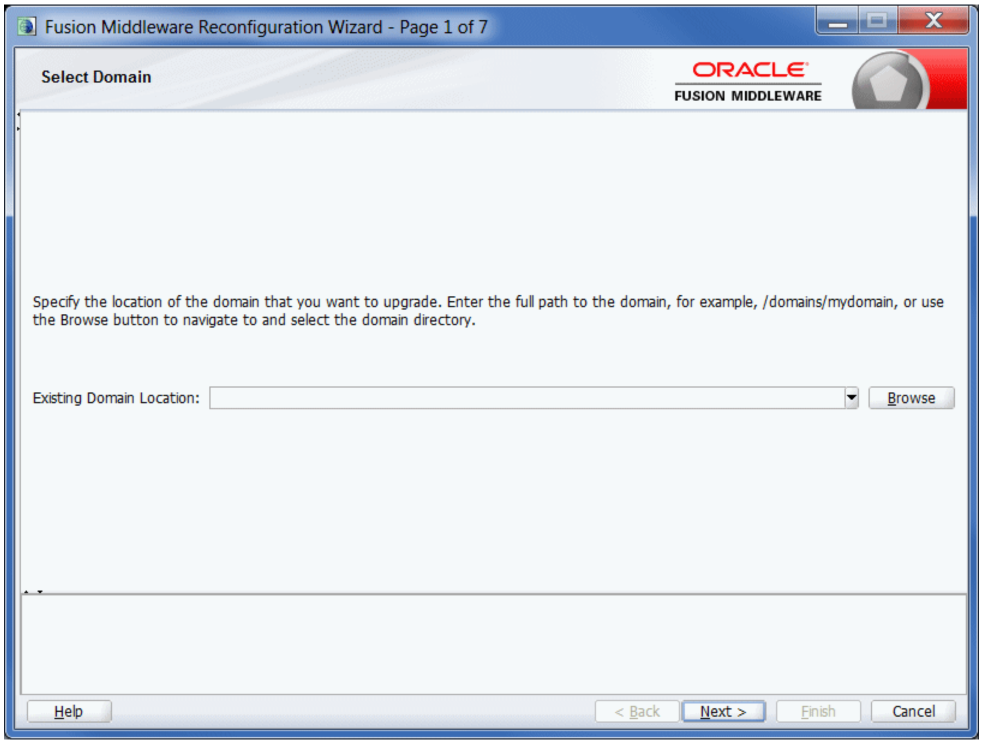
This chapter describes the screens that are displayed for the Reconfiguration Wizard when reconfiguring a domain during upgrade.

It includes the following sections:

- Section 5.1, "Select Domain"
- Section 5.2, "Reconfiguration Setup Progress"
- Section 5.3, "Domain Mode and JDK"
- Section 5.4, "Database Configuration Type"
- Section 5.5, "JDBC Component Schema"
- Section 5.6, "GridLink Oracle RAC Component Schema"
- Section 5.7, "Oracle RAC Multi Data Source Component Schema"
- Section 5.8, "JDBC Component Schema Test"
- Section 5.9, "JDBC Data Sources"
- Section 5.10, "GridLink Oracle RAC Data Sources"
- Section 5.11, "Oracle RAC Multi Data Sources"
- Section 5.12, "JDBC Data Sources Test"
- Section 5.13, "Database Scripts"
- Section 5.14, "Credentials"
- Section 5.15, "Keystore"
- Section 5.16, "Advanced Configuration"
- Section 5.17, "Managed Servers"
- Section 5.18, "Clusters"
- Section 5.19, "Assign Servers to Clusters"
- Section 5.20, "HTTP Proxy Applications"
- Section 5.21, "Coherence Clusters"
- Section 5.22, "Machines"
- Section 5.23, "Assign Servers to Machines"
- Section 5.24, "Deployments Targeting"
- Section 5.25, "Services Targeting"

- Section 5.26, "JMS File Stores"
- Section 5.27, "Configuration Summary"
- Section 5.28, "Reconfiguration Progress"
- Section 5.29, "Reconfiguration Success"

5.1 Select Domain



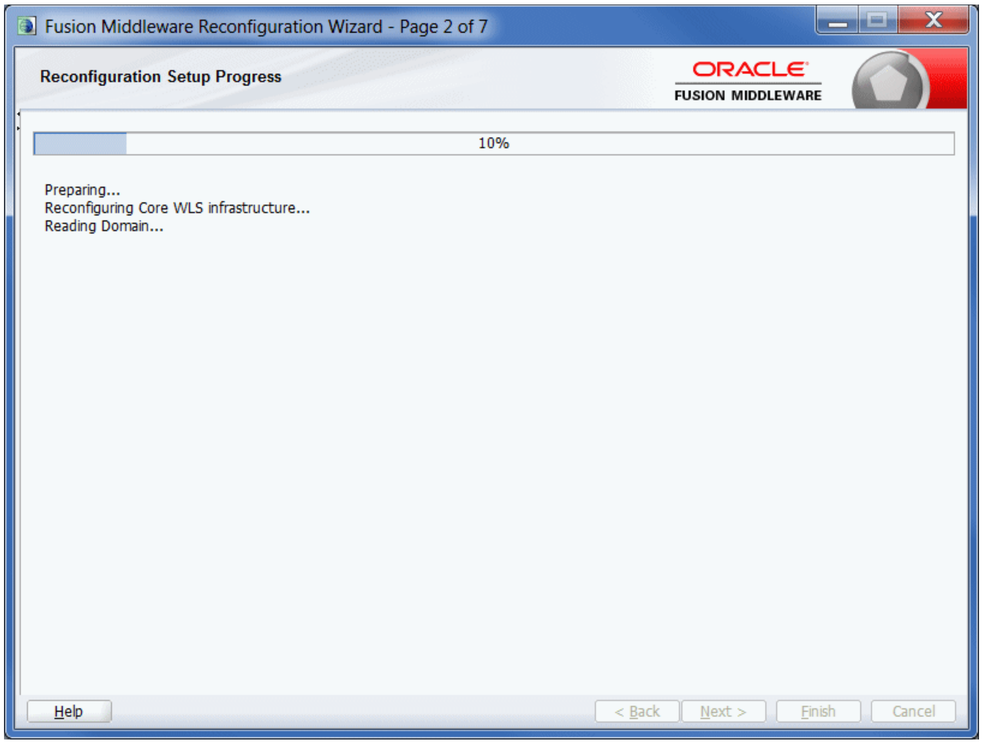
Specify the location of the domain that you want to upgrade. Enter the full path to the domain, for example, /domains/mydomain, or use the Browse button to navigate to and select the domain directory.

Existing Domain Location:

From this screen, you specify the location of the domain that you want to upgrade.

Option/Field	Description
Enter Domain Location	Enter the full path to the domain that you want to reconfigure, for example, C:\domains\mydomain. You can also use the **Browse** button to navigate to the domain directory.

5.2 Reconfiguration Setup Progress


Fusion Middleware Reconfiguration Wizard - Page 2 of 7
Reconfiguration Setup Progress
ORACLE
FUSION MIDDLEWARE
10%
Help
< Back
Next >
Finish
Cancel

```
Preparing...
Reconfiguring Core WLS infrastructure...
Reading Domain...
```

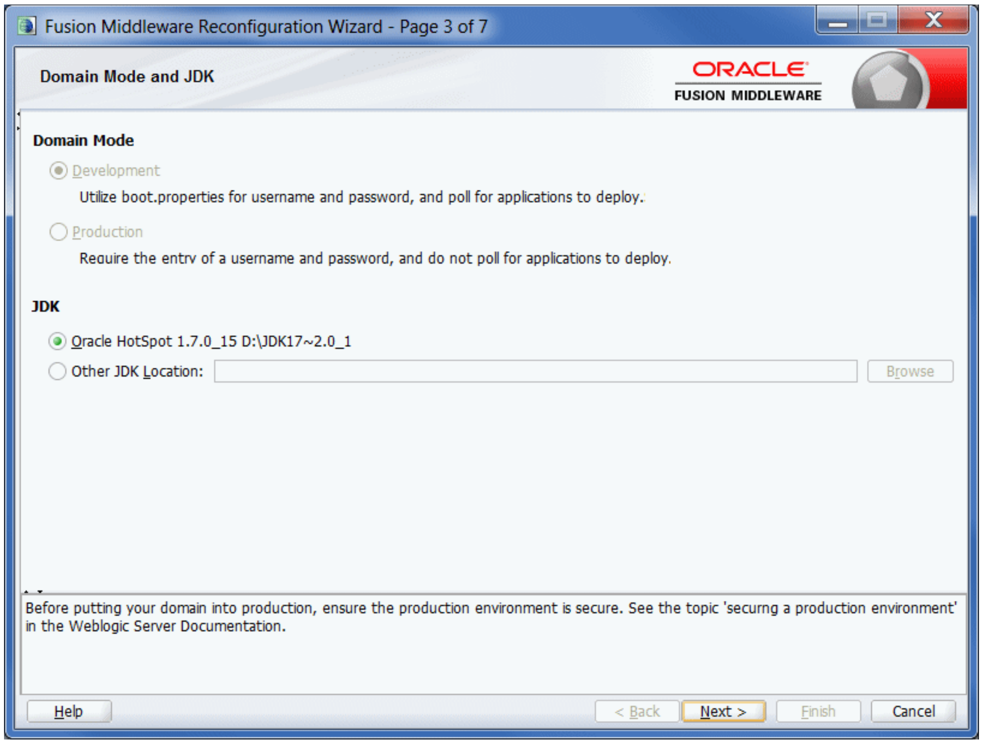
This screen displays the progress of the setup process. During this process:

- WLS core infrastructure files are updated
- The WebLogic domain configuration is read
- Reconfiguration templates for your installed products, including Fusion Middleware products, are automatically applied. This updates various domain configuration files such as config.xml, config-groups.xml, and security.xml (among others).
- Schemas, scripts, and other such files that support your Fusion Middleware products are updated.
- The domain upgrade is validated.

Click **Next** to continue. The Domain Mode and JDK screen is displayed.

After configuring the domain mode and selecting the JDK, subsequent screens depend on the contents of the domain, and differ for each domain.

5.3 Domain Mode and JDK


Fusion Middleware Reconfiguration Wizard - Page 3 of 7

Domain Mode and JDK

Domain Mode

( ) Development
Utilize boot.properties for username and password, and poll for applications to deploy.

( ) Production
Require the entry of a username and password, and do not poll for applications to deploy.

JDK

(●) Oracle HotSpot 1.7.0_15 D:\JDK17~2.0_1
( ) Other JDK Location:

Before putting your domain into production, ensure the production environment is secure. See the topic 'securng a production environment' in the Weblogic Server Documentation.

Domain Mode

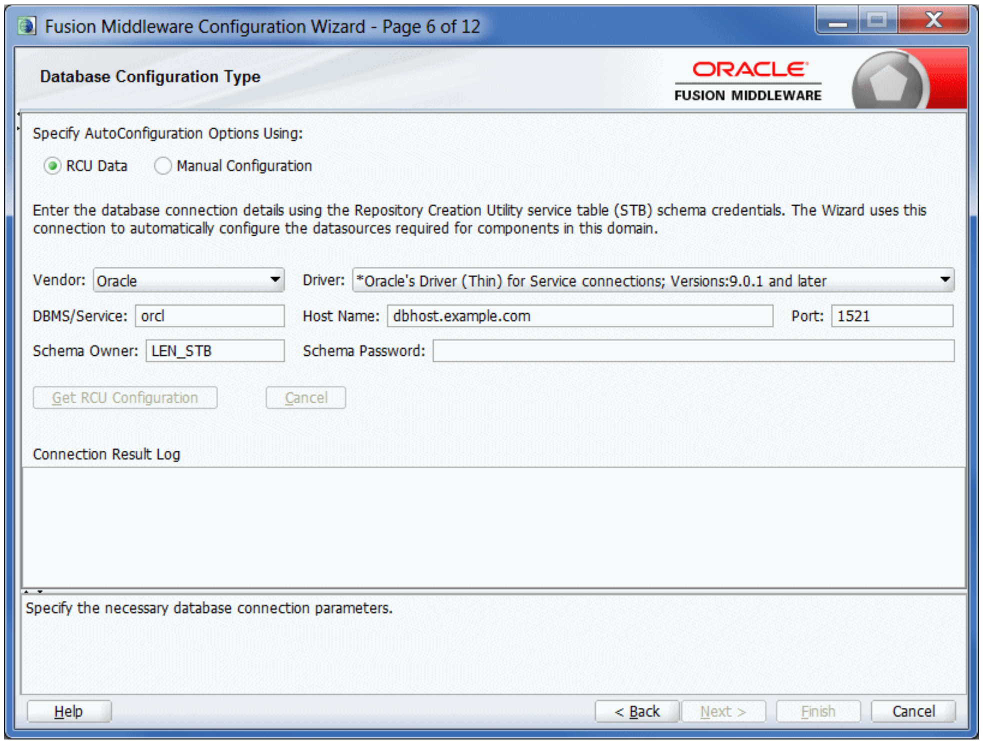
The domain mode cannot be changed during reconfiguration. It is inherited from the original domain.

JDK

Use the JDK section to select the JDK for the domain. Select only a JDK that is supported on the platform you are using. For a list of the JDKs that are supported for a specific platform, see "Oracle Fusion Middleware Supported System Configurations" on Oracle Technology Network.

Option	Description
JDK	Lists the JDK that was used when you installed WebLogic Server. The default JDK is Oracle HotSpot SDK *version*, but you may have installed and used another JDK during installation.
Other JDK	Select this option to use a JDK other than the one that you used when you installed WebLogic Server. Use the **Browse** button to navigate to the directory where the JDK resides. If you select the JDK that you used when you installed WebLogic Server, the Reconfiguration Wizard creates server startup scripts to invoke that JDK. If you select a JDK that you did not use when you installed WebLogic Server, the Reconfiguration Wizard does not configure the startup scripts; you must change the startup scripts manually. For more information about startup scripts, see *Tuning Performance of Oracle WebLogic Server*.

5.4 Database Configuration Type


Fusion Middleware Configuration Wizard - Page 6 of 12
Database Configuration Type
ORACLE
FUSION MIDDLEWARE
Get RCU Configuration
Cancel
Help
< Back
Next >
Finish
Cancel

Specify AutoConfiguration Options Using:

[✓ RCU Data] Manual Configuration

Enter the database connection details using the Repository Creation Utility service table (STB) schema credentials. The Wizard uses this connection to automatically configure the datasources required for components in this domain.

Vendor: Oracle
Driver: *Oracle's Driver (Thin) for Service connections; Versions:9.0.1 and later
DBMS/Service: orcl
Host Name: dbhost.example.com
Port: 1521
Schema Owner: LEN_STB
Schema Password:

Connection Result Log

Specify the necessary database connection parameters.

You can use this screen to specify the information for connecting to the database to retrieve schema information that will be used to populate the schema fields on subsequent component schema screens (JDBC Component Schema, GridLink Oracle RAC Component Schema, or Oracle RAC Multi Data Source Component Schema). You also have the option to skip this step and manually configure each component schema on the component schema screen.

Select one of the following options.

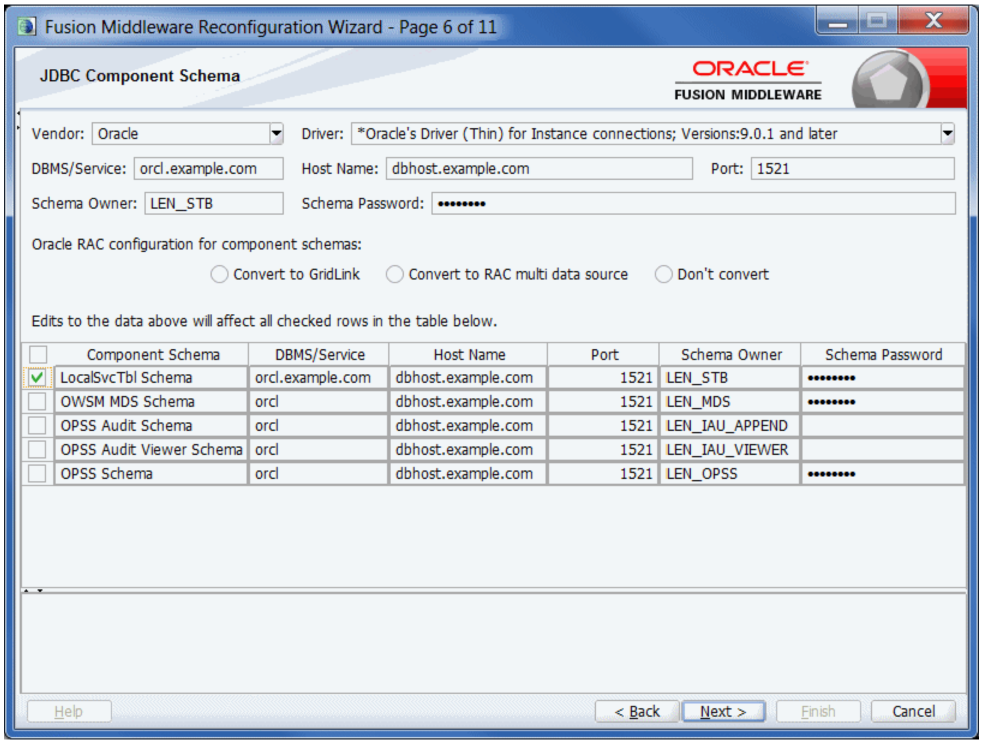
Field	Description
RCU Data	Select this option if you want to connect to the database to retrieve schema information for all schemas that are included in the domain. If you select this option, the fields on this screen are activated. Fill in each field, using the connection information that you specified for the Service Table component in the Repository Creation Utility (RCU). When done, click **Get RCU Configuration** to retrieve the schema information. For information about Service Tables, see "Understanding Service Tables" in *Administering*. After successfully retrieving the schema information, click **Next** to continue. **Note:** The only JDBC connections that are supported are the Oracle service type JDBC connections that are supported by RCU.
Manual Configuration	If you select this option, you must manually configure the settings for each schema. Note that some of the fields in the schema table on the component schema screen are populated with default Java DB values, for example, Host Name is set to dbhost.example for all schemas. After selecting this option, click **Next** to continue.

Complete the following fields for the **RCU Data** option, and then click **Get RCU Configuration**.

See "Database Connection Details" in for more information.

Field	Description
Vendor	Select the database vendor.
Driver	Select the JDBC driver that is configured for the database. The list includes common JDBC drivers for the selected database vendor.
DBMS/Service	Enter the database DBMS name, or service name if you selected a service type driver.
Host Name	Enter the name of the server hosting the database.
Port	Enter the port number to be used to connect to the server that hosts the database.
Schema Owner Schema Password	Enter the username and password for connecting to the database's service table schema. This is the schema username and password that was specified for the Service Table component on the "Schema Passwords" screen of the Repository Creation Utility (RCU). The default username is *prefix*_STB, where prefix is the prefix that you defined in RCU. The schema you specify must be unique for the domain (not being used by any other domains). See "Schema Passwords" in for more information.

5.5 JDBC Component Schema



Vendor: Oracle Driver: *Oracle's Driver (Thin) for Instance connections; Versions:9.0.1 and later

DBMS/Service: orcl.example.com Host Name: dbhost.example.com Port: 1521

Schema Owner: LEN_STB Schema Password: ••••••••

Oracle RAC configuration for component schemas:

○ Convert to GridLink ○ Convert to RAC multi data source ○ Don't convert

Edits to the data above will affect all checked rows in the table below.

☐	Component Schema	DBMS/Service	Host Name	Port	Schema Owner	Schema Password
✓	LocalSvcTbl Schema	orcl.example.com	dbhost.example.com	1521	LEN_STB	••••••••
☐	OWSM MDS Schema	orcl	dbhost.example.com	1521	LEN_MDS	••••••••
☐	OPSS Audit Schema	orcl	dbhost.example.com	1521	LEN_IAU_APPEND	
☐	OPSS Audit Viewer Schema	orcl	dbhost.example.com	1521	LEN_IAU_VIEWER	
☐	OPSS Schema	orcl	dbhost.example.com	1521	LEN_OPSS	••••••••

For some Fusion Middleware components, JDBC data sources might be defined as part of the component's database schema, which are loaded during installation of the component by using the Repository Creation Utility (RCU).

When you reconfigure a WebLogic domain for such components by using the Reconfiguration Wizard, you can configure the JDBC component schema settings, such as database driver, schema owner, password, and so on.

- If you selected the **RCU Data** option on the Database Configuration Type screen, the schema table has already been populated appropriately and you can click **Next** to continue.
- If you selected the **Manual Configuration** option on the Database Configuration Type screen, you must configure the schemas listed in this table manually before continuing.

The JDBC component schemas associated with the products for which you are creating the domain are listed in the lower half of the screen.

Select the schemas for which you want to specify data source settings by selecting the check box adjacent to each schema name.

Note: When you select multiple component schemas, the text "Varies among component schemas" might be displayed in certain fields, indicating that the current values of those fields are different across the selected component schemas. If you change the values in such fields, the new values are applied uniformly across the selected component schemas.

The default values of component schema parameters such as vendor, driver, host name, and port number depend on the values that are specified in the application templates.

Field	Description
Vendor	Select the database vendor.
Driver	Select the JDBC driver to use to connect to the database. The list includes common JDBC drivers for the selected database vendor.
DBMS/Service	Enter a database DBMS name, or service name if you selected a service type driver.
Host Name	Enter the name of the server hosting the database.
Port	Enter the port number to be used to connect to the server that hosts the database.
Schema Owner	Enter the username for connecting to the database.
Schema Password	Enter the password for this username.

The values that you specify are displayed in the appropriate columns in the schema list, for the selected schemas.

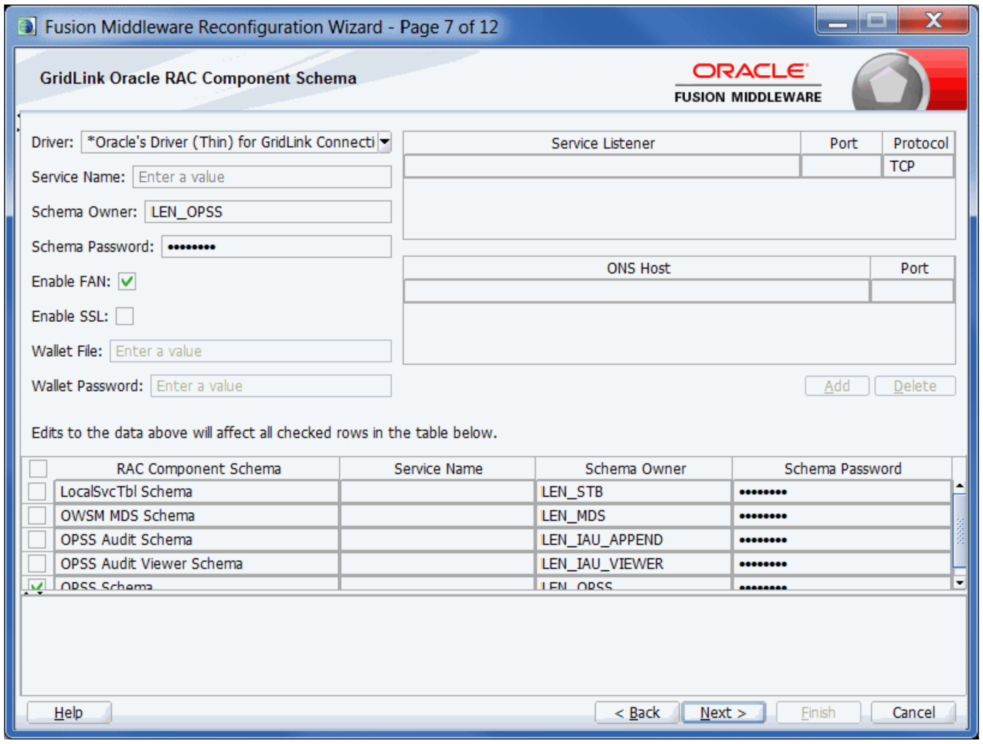
To convert one or more schemas to GridLink RAC schemas, select the check boxes adjacent to the name of the those schemas, and select the **Convert to GridLink** option. Click **Next** when done. When you click **Next**, the GridLink Oracle RAC Component Schema screen is displayed.

For more information, see "Using GridLink Data Sources" in *Administering JDBC Data Sources for Oracle WebLogic Server*

To convert one or more of the schemas to Oracle RAC multi data source schemas, select the check boxes adjacent to the name of the those schemas, and select the **Convert to RAC multi data source** option. Click **Next** when done. When you click **Next**, the Oracle RAC Multi Data Source Component Schema screen is displayed.

For more information, see "Using WebLogic Server with Oracle RAC" in *Administering JDBC Data Sources for Oracle WebLogic Server*.

5.6 GridLink Oracle RAC Component Schema


Fusion Middleware Reconfiguration Wizard - Page 7 of 12
GridLink Oracle RAC Component Schema
ORACLE
FUSION MIDDLEWARE
Add
Delete
Help
< Back
Next >
Finish
Cancel

Driver: *Oracle's Driver (Thin) for GridLink Connecti
Service Name: Enter a value
Schema Owner: LEN_OPSS
Schema Password: ••••••••
Enable FAN: ✓
Enable SSL:
Wallet File: Enter a value
Wallet Password: Enter a value

Service Listener	Port	Protocol
		TCP

ONS Host	Port

Edits to the data above will affect all checked rows in the table below.

	RAC Component Schema	Service Name	Schema Owner	Schema Password
	LocalSvcTbl Schema		LEN_STB	••••••••
	OWSM MDS Schema		LEN_MDS	••••••••
	OPSS Audit Schema		LEN_IAU_APPEND	••••••••
	OPSS Audit Viewer Schema		LEN_IAU_VIEWER	••••••••
✓	OPSS Schema		LEN_OPSS	••••••••

Use this screen to configure the component schemas that are included in your WebLogic domain as GridLink RAC data sources. A GridLink data source is a single data source that represents a service that responds to Fast Application Notification (FAN) events.

For more information on GridLink RAC data sources, see "Using GridLink Data Sources" in *Administering JDBC Data Sources for Oracle WebLogic Server*.

The component schemas that you opted to configure as GridLink RAC data sources in the JDBC Component Schema screen of the wizard are listed in the lower half of the screen.

- If you selected the **RCU Data** option on the Database Configuration Type screen, the schema table has already been populated appropriately and you can click **Next** to continue.
- If you selected the **Manual Configuration** option on the Database Configuration Type screen, you must configure the schemas listed in this table manually before continuing.

In the schema list in the lower half of the screen, select the schemas to configure as GridLink RAC data sources by selecting the check box adjacent to each schema name.

Note: When you select multiple data source schemas, the text "Varies among component schemas" might be displayed in certain fields, indicating that the current values of those fields are different across the selected schemas. If you go ahead and change the values in such fields, the new values are applied uniformly across the selected schemas.

Field	Description
Driver	Select the appropriate driver. Some or all of the following drivers are listed: ■ Oracle Driver (Thin) for GridLink Connections This is the GridLink Type 4 non-XA driver. ■ Oracle Driver (Thin XA) for GridLink Connections This is the GridLink Type 4 XA driver.
Service Name	Enter a database Oracle RAC service name.
Schema Owner	Enter the username for connecting to the database.
Schema Password	Enter the password for this username.
Enable FAN	When selected, the data source will register for and process FAN notifications.
Enable SSL	When selected, SSL is enabled, and you must specify a wallet file, wallet password, and at least one Oracle Notification Service (ONS) host/port.
Wallet File	If SSL is enabled, specify the full path to the wallet file that contains the credentials for ONS/SSL. A wallet file is an Oracle credential file that stores keys and certificates.
Wallet Password	Specify the password for the wallet file. The password will be encrypted in the module configuration file.
Service Listener	Enter the name of the GridLink database Service Listener. You must configure the Service Listener for at least one database instance.
Port	This is the listen port for the database service listener. It defaults to 1521 and typically does not need to be changed.
Protocol	Click in this field and select the protocol to use for communication between WebLogic Server and the database service listener.
ONS Host	Specify the Oracle Notification Service (ONS) host name. If SSL is enabled, you must specify at least one ONS host and port.
Port	Specify the listen port to use on the ONS host.

The values that you specify are displayed in the appropriate columns in the schema list, for the selected schemas.

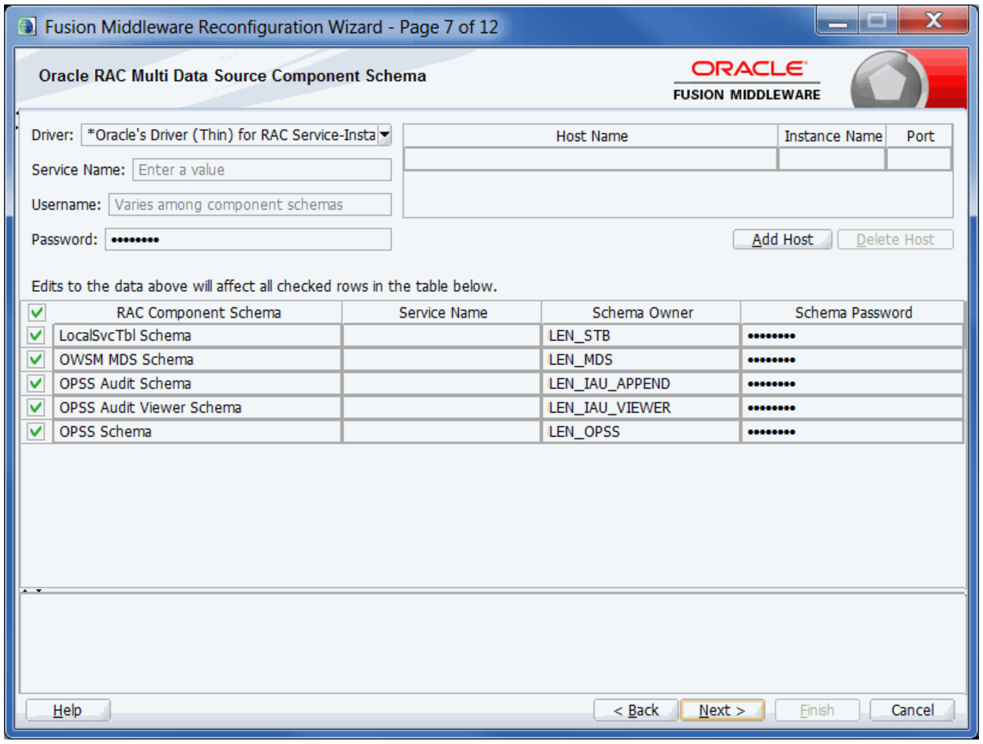
To add another row to the **Service Listener** table, click anywhere in the table, and then click **Add**.

To add another row to the **ONS Host** table, click anywhere in the table, and then click **Add**.

To delete a row from the **Service Listener** table, click anywhere in the row, and then click **Delete**.

To delete a row from the **ONS Host** table, click anywhere in the row, and then click **Delete**.

5.7 Oracle RAC Multi Data Source Component Schema


Fusion Middleware Reconfiguration Wizard - Page 7 of 12
Oracle RAC Multi Data Source Component Schema
ORACLE
FUSION MIDDLEWARE
Add Host
Delete Host
Help
< Back
Next >
Finish
Cancel

Driver: *Oracle's Driver (Thin) for RAC Service-Insta
Service Name: Enter a value
Username: Varies among component schemas
Password: ••••••••

Host Name	Instance Name	Port

Edits to the data above will affect all checked rows in the table below.

✓	RAC Component Schema	Service Name	Schema Owner	Schema Password
✓	LocalSvcTbl Schema		LEN_STB	••••••••
✓	OWSM MDS Schema		LEN_MDS	••••••••
✓	OPSS Audit Schema		LEN_IAU_APPEND	••••••••
✓	OPSS Audit Viewer Schema		LEN_IAU_VIEWER	••••••••
✓	OPSS Schema		LEN_OPSS	••••••••

Use this screen to configure the component schemas that are included in the WebLogic domain as Oracle RAC multi data sources.

For more information, see "Using WebLogic Server with Oracle RAC" in *Administering JDBC Data Sources for Oracle WebLogic Server*.

The component schemas that you opted to configure as Oracle RAC multi data sources in the JDBC Component Schema screen of the wizard are listed in the lower half of the screen.

- If you selected the **RCU Data** option on the Database Configuration Type screen, the schema table has already been populated appropriately and you can click **Next** to continue.
- If you selected the **Manual Configuration** option on the Database Configuration Type screen, you must configure the schemas listed in this table manually before continuing.

In the schema list in the lower half of the screen, select the schemas to configure as Oracle RAC multi data sources by selecting the check box adjacent to each schema name.

Note: When you select multiple data source schemas, the text "Varies among component schemas" might be displayed in certain fields, indicating that the current values of those fields are different across the selected schemas. If you go ahead and change the values in such fields, the new values are applied uniformly across the selected schemas.

Field	Description
Driver	Select the JDBC driver to use to connect to the database.
Service Name	Enter a database Oracle RAC service name.
Username	Enter the username for connecting to the database.
Password	Enter the password for the specified username.
Host Name	Enter the name of the server hosting the Oracle RAC database instances.
Instance Name	Enter the name of each Oracle database instance.
Port	Enter the port number to use to connect to the server that hosts the database.

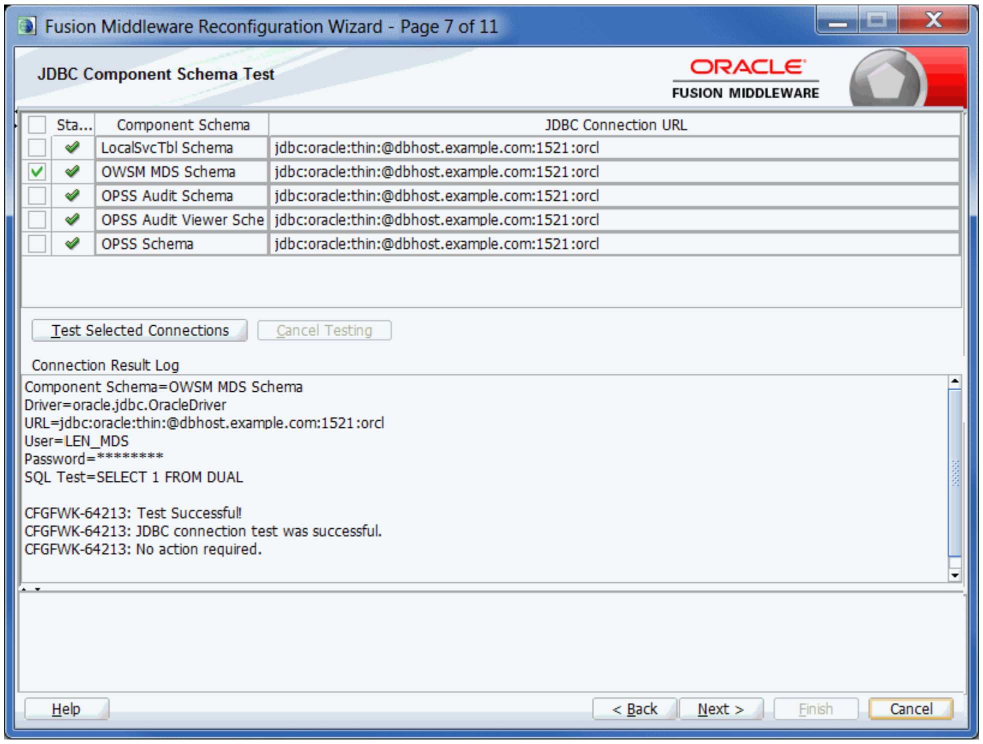
Note: You must specify the host name, instance name, and port number of at least one database instance.

To add another database instance for the currently selected schemas, click **Add Host**.

To delete a database instance, click anywhere in that row in the Host Name table, and then click **Delete**.

The values that you specify for the schema are displayed in the appropriate columns in the schema list, for the selected schemas.

5.8 JDBC Component Schema Test


Fusion Middleware Reconfiguration Wizard - Page 7 of 11
JDBC Component Schema Test
ORACLE
FUSION MIDDLEWARE
Test Selected Connections
Cancel Testing
Connection Result Log
Help
< Back
Next >
Finish
Cancel

	Sta...	Component Schema	JDBC Connection URL
	✓	LocalSvcTbl Schema	jdbc:oracle:thin:@dbhost.example.com:1521:orcl
✓	✓	OWSM MDS Schema	jdbc:oracle:thin:@dbhost.example.com:1521:orcl
	✓	OPSS Audit Schema	jdbc:oracle:thin:@dbhost.example.com:1521:orcl
	✓	OPSS Audit Viewer Sche	jdbc:oracle:thin:@dbhost.example.com:1521:orcl
	✓	OPSS Schema	jdbc:oracle:thin:@dbhost.example.com:1521:orcl

```
Component Schema=OWSM MDS Schema
Driver=oracle.jdbc.OracleDriver
URL=jdbc:oracle:thin:@dbhost.example.com:1521:orcl
User=LEN_MDS
Password=********
SQL Test=SELECT 1 FROM DUAL

CFGFWK-64213: Test Successful!
CFGFWK-64213: JDBC connection test was successful.
CFGFWK-64213: No action required.
```

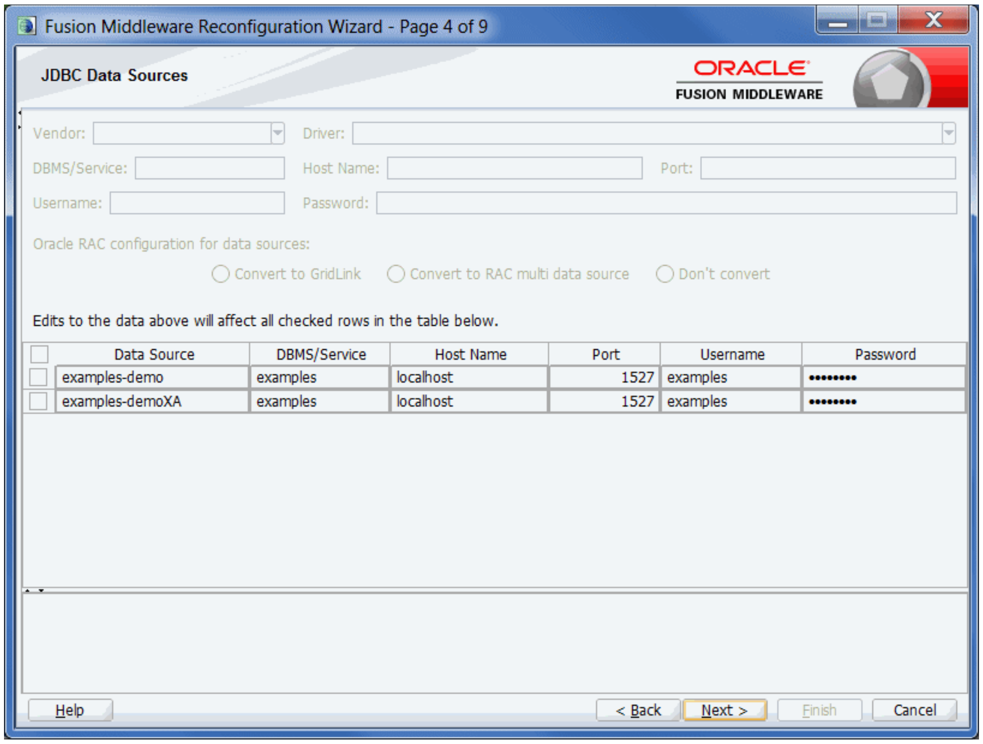
Use this screen to test the configurations that you specified for the data sources in the previous screen.

Select the check boxes adjacent to the names of the schemas to test, and then click **Test Selected Connections**.

The wizard tests the configuration for each schema by attempting to connect to a URL that is constructed by using the driver, host, port, and other information that you specified while configuring the schema.

The result of the test is indicated in the **Status** column. Details are displayed in the **Connection Result Log** section.

5.9 JDBC Data Sources



Edits to the data above will affect all checked rows in the table below.

	Data Source	DBMS/Service	Host Name	Port	Username	Password
☐	examples-demo	examples	localhost	1527	examples	••••••••
☐	examples-demoXA	examples	localhost	1527	examples	••••••••

A JDBC data source contains a pool of database connections that are created when the data source instance is created—when it is deployed or targeted, or at server startup. Applications look up a data source on the JNDI tree, and then request a connection. When the applications no longer need the connections, they return the connections to the connection pool in the data source.

Use this screen to configure the JDBC data sources defined in your domain source.

The JDBC data sources associated with the products for which you are creating the domain are listed in the lower half of the screen.

Select the data source(s) for which you want to specify settings by selecting the check box adjacent to each data source name. The values that you specify are displayed in the appropriate columns in the data source list, for the selected data source.

Notes: When you select multiple data sources, the text "Varies among component schemas" might be displayed in certain fields, indicating that the current values of those fields are different across the selected data sources. If you change the values in such fields, the new values are applied uniformly across the selected data sources.

The default values of data source parameters such as vendor, driver, host name, and port number depend on the values that are specified in the application templates.

Field	Description
Vendor	Select the database vendor.

Field	Description
Driver	Select the JDBC driver to use to connect to the database. The list includes common JDBC drivers for the selected database vendor.
DBMS/Service	Enter a DBMS SID or service name. The value that you enter depends on the driver that you selected. If the name of the Oracle driver that you selected contains the words "for Instance connections," you must enter the SID. If the name of the Oracle driver contains the words "for Service connections," you must enter the service name. For information about configuring a DataDirect driver, see the DataDirect documentation.
Host Name	Enter the name of the server hosting the database.
Port Name	Enter the port number to be used to connect to the server.
Username	Enter the username for connecting to the database.
Password	Enter the password for the specified username.

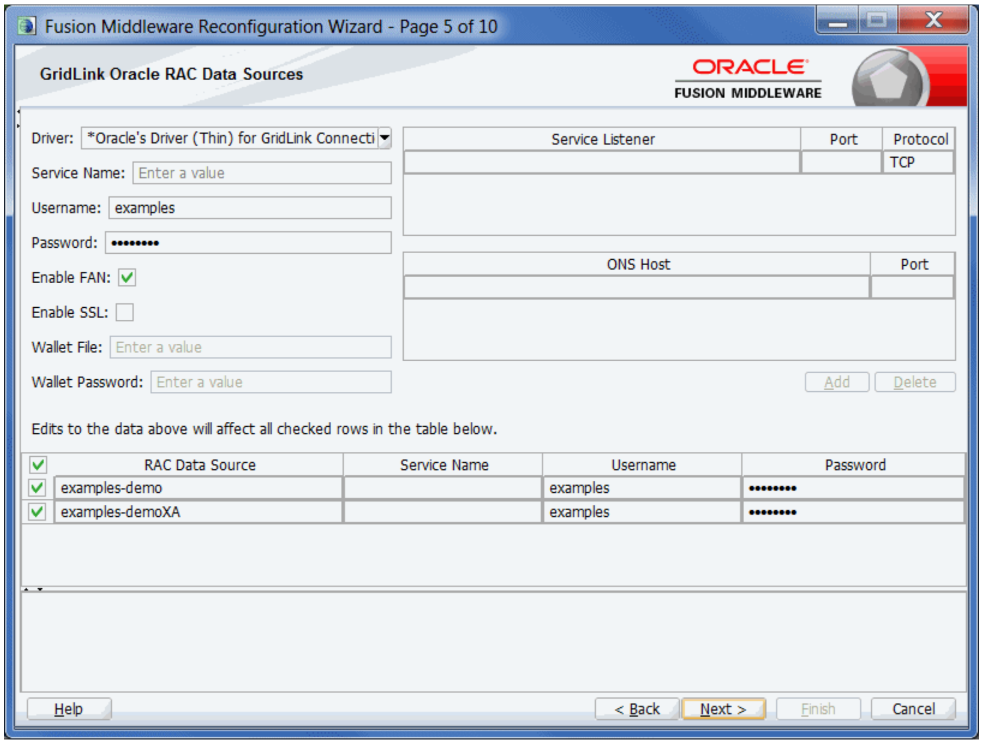
To convert one or more data sources to GridLink Oracle RAC data sources, select the check boxes adjacent to the name of the those schemas, and select the **Convert to GridLink** option. Click **Next** when done. When you click **Next**, the GridLink Oracle RAC Data Sources screen is displayed.

For more information, see "Using GridLink Data Sources" in *Administering JDBC Data Sources for Oracle WebLogic Server*

To convert one or more of the data sources to Oracle RAC multi data sources, select the check box adjacent to the name of the required data source, and select the **Convert to RAC multi data source** option. When you click **Next**, the Oracle RAC Multi Data Sources screen is displayed.

For more information, see "Using WebLogic Server with Oracle RAC" in *Administering JDBC Data Sources for Oracle WebLogic Server*.

5.10 GridLink Oracle RAC Data Sources


Fusion Middleware Reconfiguration Wizard - Page 5 of 10
GridLink Oracle RAC Data Sources
ORACLE FUSION MIDDLEWARE
Add
Delete
Help
< Back
Next >
Finish
Cancel

Driver: *Oracle's Driver (Thin) for GridLink Connecti

Service Name: Enter a value

Username: examples

Password: ••••••••

Enable FAN: ✓

Enable SSL:

Wallet File: Enter a value

Wallet Password: Enter a value

Service Listener	Port	Protocol
		TCP

ONS Host	Port

Edits to the data above will affect all checked rows in the table below.

✓	RAC Data Source	Service Name	Username	Password
✓	examples-demo		examples	••••••••
✓	examples-demoXA		examples	••••••••

Use this screen to configure the data sources that are included in your WebLogic domain as GridLink Oracle RAC data sources. A GridLink data source is a single data source that represents a service that responds to Fast Application Notification (FAN) events.

For more information, see "Using GridLink Data Sources" in *Administering JDBC Data Sources for Oracle WebLogic Server*

The data sources that you opted to configure as GridLink RAC data sources in the JDBC Data Sources screen of the wizard are listed in the lower half of the screen.

In the data source list in the lower half of the screen, select the data sources to configure as GridLink RAC data sources by selecting the check box adjacent to each data source name.

Note: When you select multiple data sources, the text "Varies among data sources" might be displayed in certain fields, indicating that the current values of those fields are different across the selected data sources. If you go ahead and change the values in such fields, the new values are applied uniformly across the selected data sources.

Field	Description
Driver	Some or all of the following drivers are listed: ■ Oracle Driver (Thin) for GridLink Connections This is the GridLink Type 4 non-XA driver. ■ Oracle Driver (Thin XA) for GridLink Connections This is the GridLink Type 4 XA driver.

Field	**Description**
Service Name	Enter a database Oracle RAC service name.
Username	Enter the username for connecting to the database.
Password	Enter the password for the specified username.
Enable FAN	When selected, the data source will register for and process FAN notifications.
Enable SSL	When selected, SSL is enabled, and you must specify a wallet file, wallet password, and at least one Oracle Notification Service (ONS) host/port.
Wallet File	If SSL is enabled, specify the full path to the wallet file that contains the credentials for ONS/SSL. A wallet file is an Oracle credential file that stores keys and certificates.
Wallet Password	Specify the password for the wallet file. The password will be encrypted in the module configuration file.
Service Listener	Enter the name of the GridLink database Service Listener. You must configure the Service Listener for at least one database instance.
Port	This is the listen port for the database service listener. It defaults to 1521 and typically does not need to be changed.
Protocol	Click in this field and select the protocol to use for communication between WebLogic Server and the database service listener.
ONS Host	Specify the Oracle Notification Service (ONS) host name. If SSL is enabled, you must specify at least one ONS host and port.
Port	Specify the listen port to use on the ONS host.

The values that you specify are displayed in the appropriate columns in the data source list, for the selected schemas.

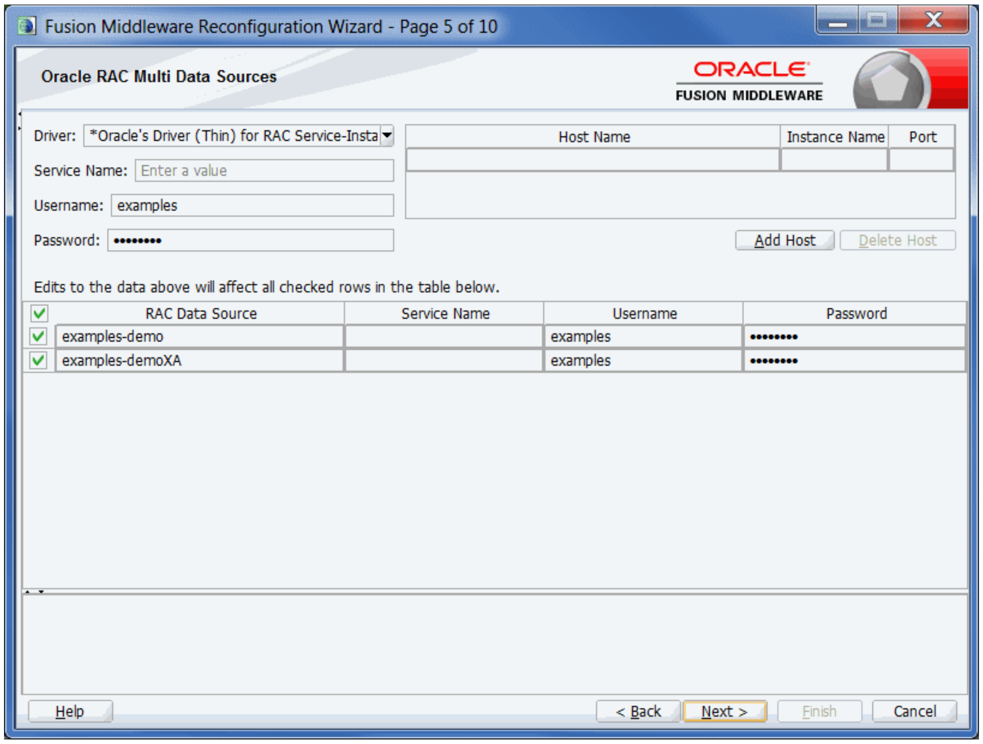
To add another row to the **Service Listener** table, click anywhere in the table, and then click **Add**.

To add another row to the **ONS Host** table, click anywhere in the table, and then click **Add**.

To delete a row from the **Service Listener** table, click anywhere in the row, and then click **Delete**.

To delete a row from the **ONS Host** table, click anywhere in the row, and then click **Delete**.

5.11 Oracle RAC Multi Data Sources


Fusion Middleware Reconfiguration Wizard - Page 5 of 10
Oracle RAC Multi Data Sources
ORACLE
FUSION MIDDLEWARE
Add Host
Delete Host
Help
< Back
Next >
Finish
Cancel

Driver: *Oracle's Driver (Thin) for RAC Service-Insta

Service Name: Enter a value

Username: examples

Password: ••••••••

Host Name	Instance Name	Port

Edits to the data above will affect all checked rows in the table below.

✓	RAC Data Source	Service Name	Username	Password
✓	examples-demo		examples	••••••••
✓	examples-demoXA		examples	••••••••

Use this screen to configure the data sources that are included in the domain as Oracle RAC data sources.

The data sources that you opted to configure as Oracle RAC data sources on the JDBC Data Sources screen are listed in the lower half of the screen.

Select the data source(s) for which you want to specify settings by selecting the check box adjacent to each data source name.

For information about Oracle RAC data sources, see "Using WebLogic Server with Oracle RAC" in *Administering JDBC Data Sources for Oracle WebLogic Server*.

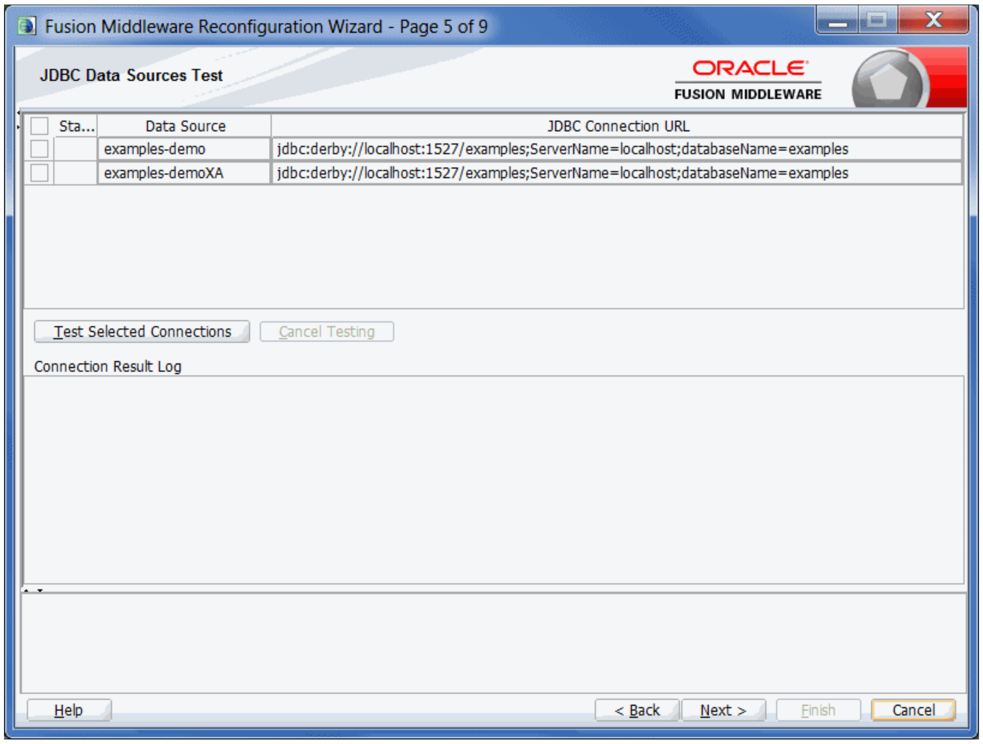
Note: When you select multiple data sources, the text "Varies among data sources" might be displayed in certain fields, indicating that the current values of those fields are different across the selected data sources. If you change the values in such fields, the new values are applied uniformly across the selected data sources.

Field/Column	Description
Driver	Select the JDBC driver to use to connect to the database.
Service Name	Enter an Oracle RAC database service name.
Username	Enter the username for connecting to the database.
Password	Enter the password for the specified user account.
Host Name	Enter the name of the server hosting the Oracle RAC database instances.
Instance Name	Enter the name of each Oracle database instance.

Field/Column	Description
Port	Enter the port numbers to be used to connect to the server that hosts the database.

To add a new database instance, click **Add**, and then specify the host name, instance name and port number.

5.12 JDBC Data Sources Test



Sta...	Data Source	JDBC Connection URL
	examples-demo	jdbc:derby://localhost:1527/examples;ServerName=localhost;databaseName=examples
	examples-demoXA	jdbc:derby://localhost:1527/examples;ServerName=localhost;databaseName=examples

Use this screen to test the data source connections you configured on the JDBC Data Sources and Oracle RAC Multi Data Sources screens.

Notes: In order to test the database connections, the database to which you are connecting must be running.

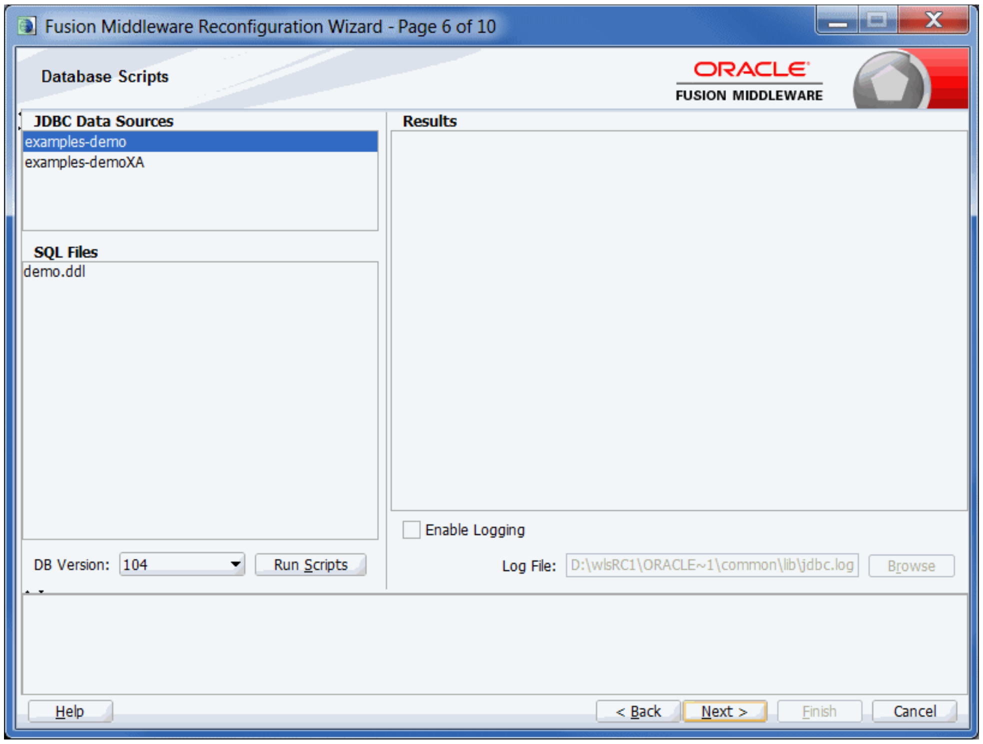
If you do not want to test the connections at this time, do not select any data sources. Click **Next** to continue.

Select the check box for each data source you want to test, and then click **Test Connections**.

The wizard tests the configuration for each selected data source by attempting to connect to a URL that is constructed by using the driver, host, port, and other information that you specified while configuring the data source.

The result of the test is indicated in the **Status** column. Details are displayed in the **Connection Result Log** section.

5.13 Database Scripts

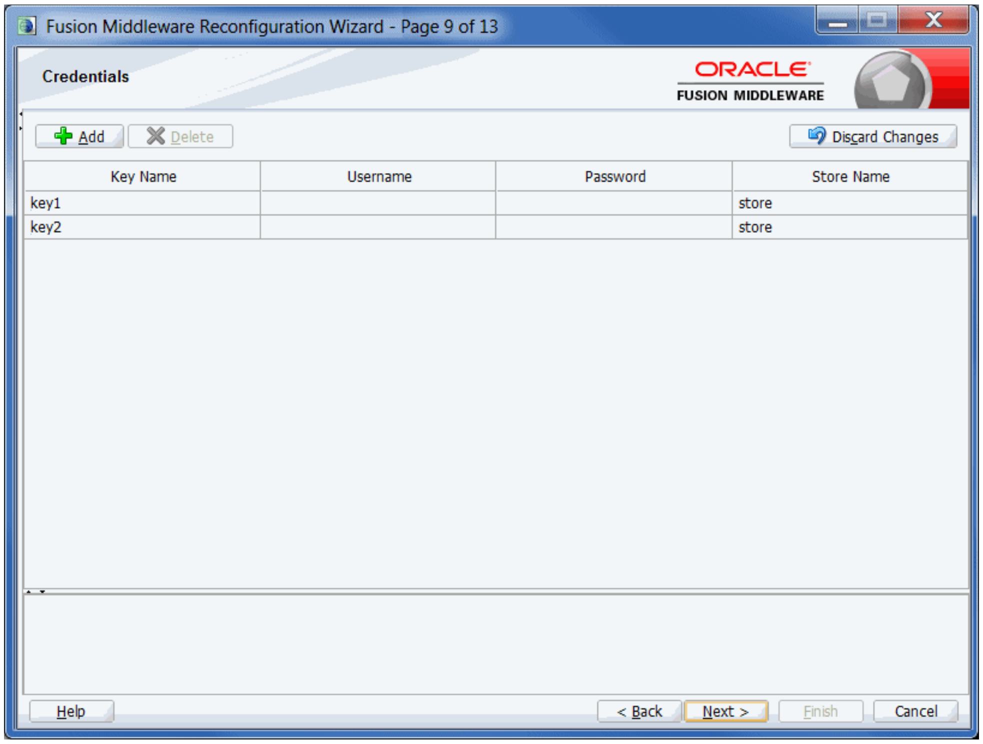


A domain template might contain a set of SQL files organized by database type. If the domain template contains SQL files, you can run them while creating the WebLogic domain, on the Database Scripts screen. Database content for each of the data sources defined in your WebLogic domain is set up by using pre-existing SQL or database loading files.

1. In the JDBC Data Sources section, select the data source for which you want to run the scripts. The scripts that can be executed are displayed in the SQL Files section.
2. Select the database version from the **DB Version** drop-down list.
3. Click **Run Scripts**.

   All the scripts displayed in the SQL Files section for the selected data source are executed, and the results are displayed in the Results section. To capture test output in a log file, select the **Enable Logging** check box and specify the full path for the log file in the **Log File** field.
4. Repeat steps 1 through 3 for each data source for which you want to execute SQL scripts.
5. Click **Next** once you have executed all scripts.

5.14 Credentials



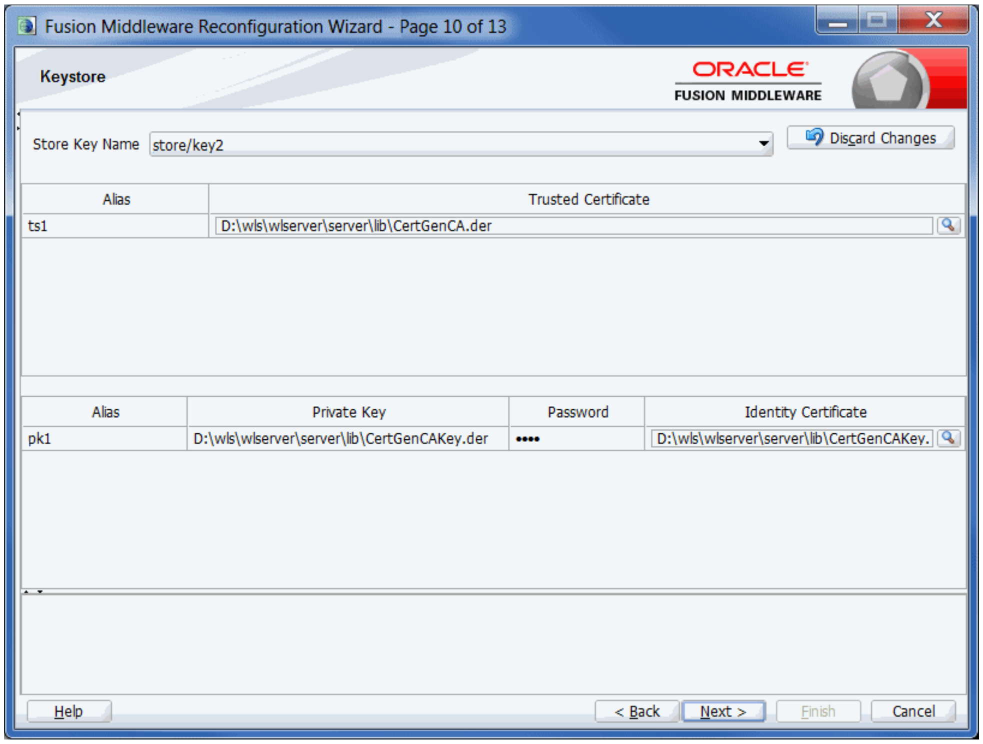
Key Name	Username	Password	Store Name
key1			store
key2			store

Use this screen to provide credentials for each key in the domain.

For more information on credentials, see "Understanding Identities, Policies, Credentials, Keys, Certificates, and Auditing" in *Application Security Guide*.

Column	Description
Key Name	This column displays the name of each key in the domain.
Username	On each row, enter the username to use for each key.
Password	On each row, enter the password to use for each key.
Store Name	This column displays the credential store that is associated with each key.

5.15 Keystore



Store Key Name: store/key2

Alias	Trusted Certificate
ts1	D:\wls\wlserver\server\lib\CertGenCA.der

Alias	Private Key	Password	Identity Certificate
pk1	D:\wls\wlserver\server\lib\CertGenCAKey.der	••••	D:\wls\wlserver\server\lib\CertGenCAKey.

Use this screen to:

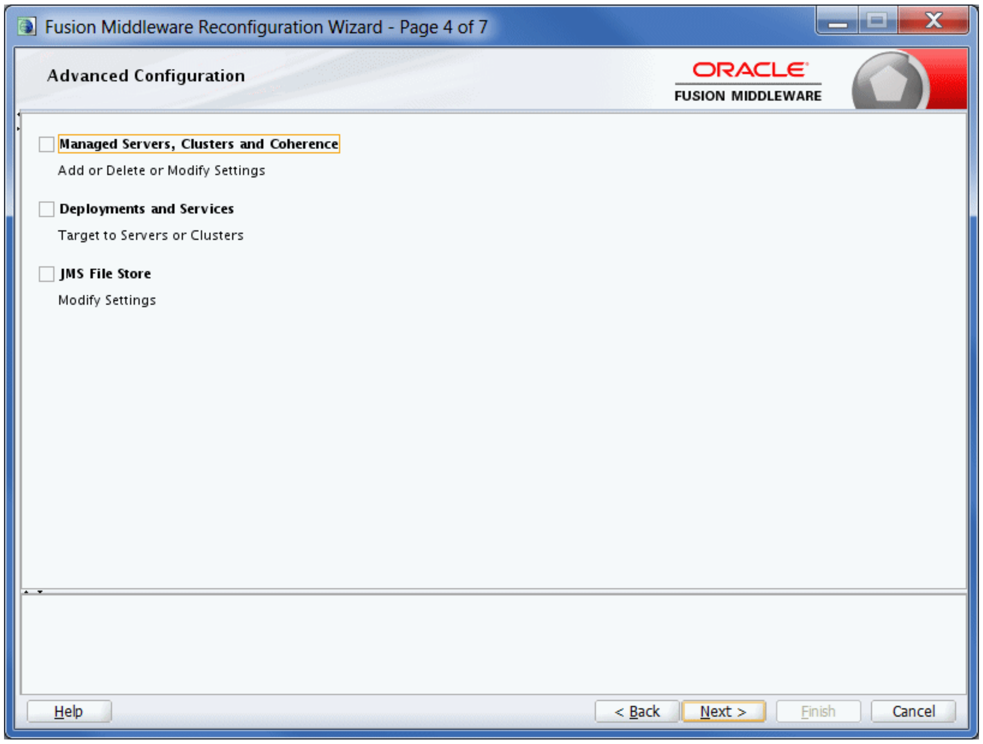
- specify the path to the trusted certificate for each keystore
- specify the path to each keystore's private key, the password for the private key and the path to the Identity Certificate for the private key.

When you click in the Trusted Certificate, Private Key, or Identity Certificate fields, a browse icon appears to the right of the field. Click this icon to browse to the appropriate file.

Option/Field	Description
Store Key Name	From this drop-down list, select the store/key that you want to configure.
Trusted Certificate table	The Trusted Certificate table contains the following two columns.
Alias	A read-only field that displays the alias for the trusted certificate as defined in the product template.
Trusted Certificate	Enter the full path and file name for the trusted certificate to use for the selected store/key, or click the icon on the far right of the row to navigate to and select the trusted certificate file.
Private Key table	The Private Key table contains the following four columns.
Alias	A read-only field that displays the alias for the private key as defined in the product template.
Private Key	Enter the full path and file name for the private key file to use for the selected store/key, or click the icon to the right of the field to navigate to and select the private key file.
Password	Enter the password to use for the private key.

Option/Field	Description
Identity Certificate	Enter the full path and file name for the identity certificate to associate with the private key, or click the icon to the right of the field to navigate to and select the identity certificate file.

5.16 Advanced Configuration



Managed Servers, Clusters and Coherence
Add or Delete or Modify Settings

Deployments and Services
Target to Servers or Clusters

JMS File Store
Modify Settings

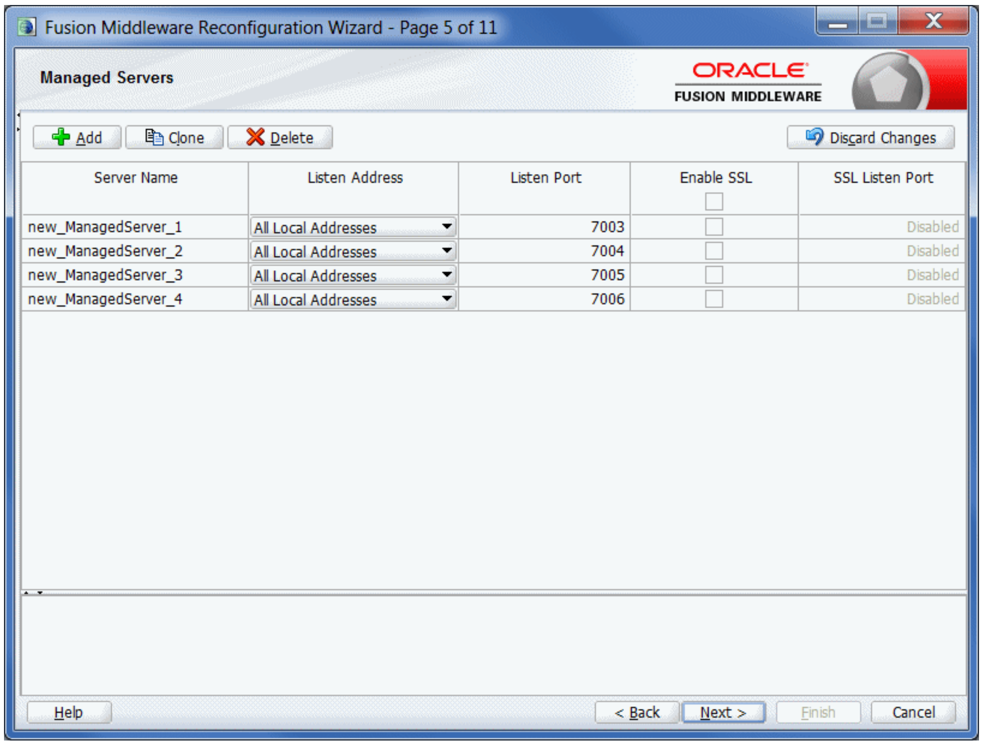
Select all categories (if any) for which you want to perform advanced configuration. For each category you select, the appropriate configuration screen is displayed to allow you to perform advanced configuration. If you do not select any items on this screen, the Configuration Summary screen is displayed next.

Notes: The categories that are listed on this screen depend on the resources defined in the templates you selected for the domain.

Option	Description
Managed Servers, Clusters, and Coherence	Select this option to ■ Add Managed Servers, clusters, or machines to the domain ■ Delete an existing Managed Server, cluster, or machine ■ Add Managed Servers to an existing cluster ■ Modify the settings for an existing Managed Server, cluster, or machine ■ Configure the default Coherence cluster See Section 5.17, "Managed Servers," through Section 5.23, "Assign Servers to Machines."

Option	Description
Deployments and Services	Select this option to customize how application deployments and services are targeted to servers and clusters. See Section 5.24, "Deployments Targeting," and Section 5.25, "Services Targeting."
JMS File Store	Select this option to change the settings for your JMS file stores. You can change the name, directory, and synchronous write policy for each file store. See Section 5.26, "JMS File Stores."

5.17 Managed Servers



Server Name	Listen Address	Listen Port	Enable SSL	SSL Listen Port
new_ManagedServer_1	All Local Addresses	7003	☐	Disabled
new_ManagedServer_2	All Local Addresses	7004	☐	Disabled
new_ManagedServer_3	All Local Addresses	7005	☐	Disabled
new_ManagedServer_4	All Local Addresses	7006	☐	Disabled

From this screen, you can add, delete, or clone Managed Servers. You can also change the settings for an existing Managed Server.

Note: You can create Managed Servers on remote machines by using the `pack` and `unpack` commands.

For more information, see "Creating and Starting a Managed Server on a Remote Machine" in *Creating Templates and Domains Using the Pack and Unpack Commands*.

Column	Description
Server Name	Valid server names are a string of characters (alphabetic and numeric). The name must be unique in the domain.
Listen Address	From the drop-down list, select a value for the listen address.

Column	Description
Listen port	Enter a valid value for the listen port to be used for regular, nonsecure requests (through protocols such as HTTP and T3). The default value is the next available listen port. If you leave this field blank, the default value is used. The valid listen port range is from 1 to 65535.
Enable SSL	Select this check box to enable the SSL listen port. By default, SSL is disabled for all new servers.
SSL listen port	This field is enabled only if you selected the SSL enabled check box for the server. Enter a valid value to be used for secure requests (through protocols such as HTTPS and T3S). The default value is the next available listen port. If you leave this field blank, the default value is used. The valid listen port range is from 1 to 65535.
Server Groups	If any of the templates you selected to create or update your domain contain a user-expandable server group definition, the **Server Groups** column is displayed. For each Managed Server, select the check box for each server group you want to assign to the server. Only server groups that are defined as user-selectable are displayed in the list. Typically, you should accept the defaults for Fusion Middleware product servers. **Note:** If you clone a Managed Server, the server group assignments are identical to the original server. Cloning is recommended for creating additional Fusion Middleware product servers. The selected server group determines the applications and services that are mapped to a given Managed Server. For example, if you select the OVAB_MAN_SVR group for a server, all applications and services that are mapped to that server group in the config-groups.xml file for the domain are automatically targeted to the server. For more information on server groups, see "config-groups.xml and startup-plan.xml" in *Domain Template Reference*.

To add a server, click **Add** and configure the settings for the new server. The default name for a new server is `new_ManagedServer_`*`n`*, where *`n`* starts at `1` and increments for each new server you add.

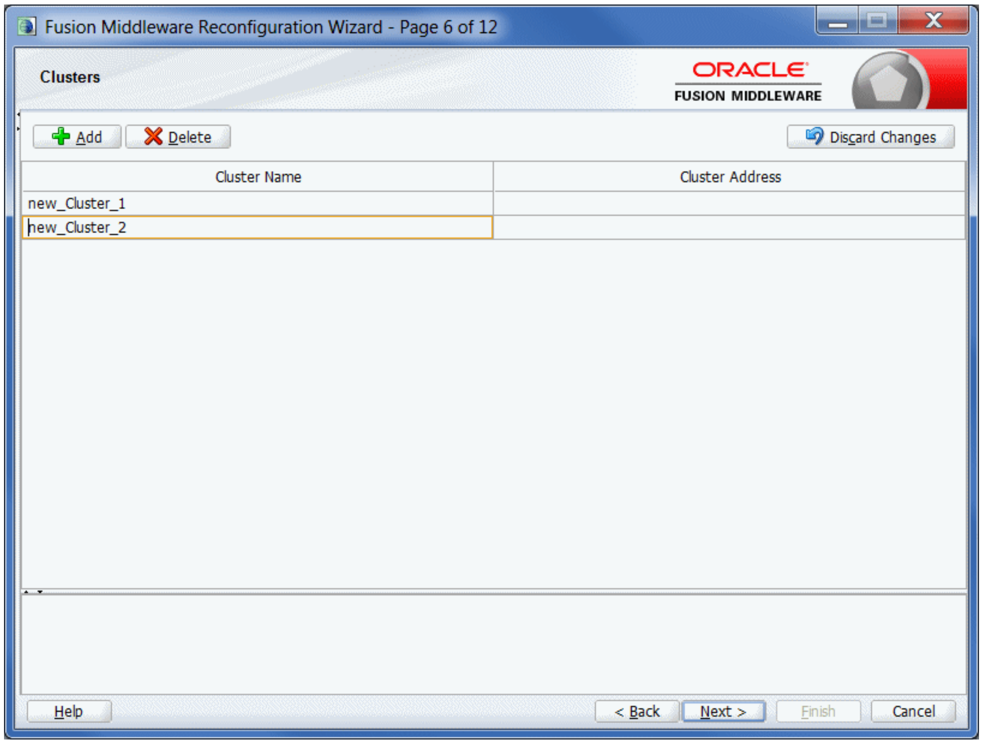
To clone a server, click in the row for the server you want to clone, and then click **Clone**. The default name for the new clone is *`original_server_name`*`_clone`*`n`*, where *`n`* starts at `1` and increments for each new server that you clone from that server. When you create a Managed Server that is a clone of an existing Managed Server, all applications and libraries that are targeted to the source server are also deployed to the clone. The cloned server is also assigned to all server groups to which the source server is assigned (if any). In addition, any of the following services that are targeted to the source server are automatically targeted to the clone:

- connectionFactory
- queueConnectionFactory
- topicConnection
- Queue
- Topic
- activationSpec
- Data source

- URLProvider
- workManager
- busMember
- customService
- resourceAdapter

To delete a server, select the server and click **Delete**. You can delete only one server at a time.

5.18 Clusters



Cluster Name	Cluster Address
new_Cluster_1	
new_Cluster_2	

A cluster is a group of WebLogic Server instances that work together to provide scalability and high-availability for applications. By creating clusters, you can group Managed Servers such that they operate as a single unit for hosting applications and resources.

Use this screen to add or delete clusters. You can also change the settings for an existing cluster.

Column	Description
Cluster Name	Enter a valid name. The name of the cluster must be unique among all component names within the WebLogic domain.

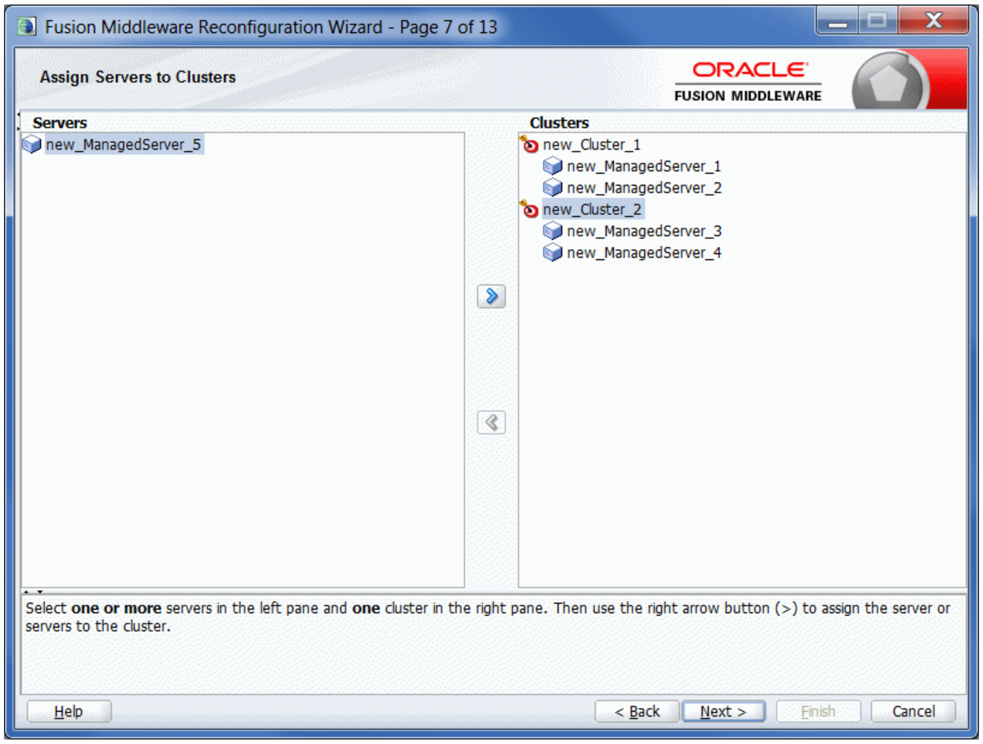
Column	Description
Cluster address	Enter the addresses for identifying the Managed Servers in the cluster. A cluster address can be one of the following: ■ Comma-separated list of IP addresses or DNS names and ports (for example: `dns_name:port`, `dns_name:port`) ■ DNS name that maps to multiple IP addresses ■ `localhost`, DNS name, or IP address if the listen address of all Managed Servers is listening to the same address with unique port numbers

To delete a cluster, select the server and click **Delete**. When you delete a cluster, you do not delete the servers assigned to it; the servers are merely removed from the cluster and can then be added to another cluster. You can delete only one cluster at a time.

To add a cluster, click **Add** and configure the settings for the new server. The default name for a new cluster is `new_Cluster_`*n*, where *n* starts at `1` and increments for each new cluster you add. To change the default, type the desired name in the **Name** column.

For more information about clusters, see "Setting Up WebLogic Clusters" in *Administering Clusters for Oracle WebLogic Server*.

5.19 Assign Servers to Clusters


Fusion Middleware Reconfiguration Wizard - Page 7 of 13
Assign Servers to Clusters
ORACLE
FUSION MIDDLEWARE
Servers
Clusters
< Back
Next >
Finish
Cancel
Help

Servers:
- new_ManagedServer_5

Clusters:
- new_Cluster_1
  - new_ManagedServer_1
  - new_ManagedServer_2
- new_Cluster_2
  - new_ManagedServer_3
  - new_ManagedServer_4

Select **one or more** servers in the left pane and **one** cluster in the right pane. Then use the right arrow button (>) to assign the server or servers to the cluster.

Use this screen to assign Managed Servers to clusters.

Note: Only Managed Servers are displayed in the **Server** list box. The Administration Server is not listed because it cannot be assigned to a cluster.

To assign one or more servers to a cluster:

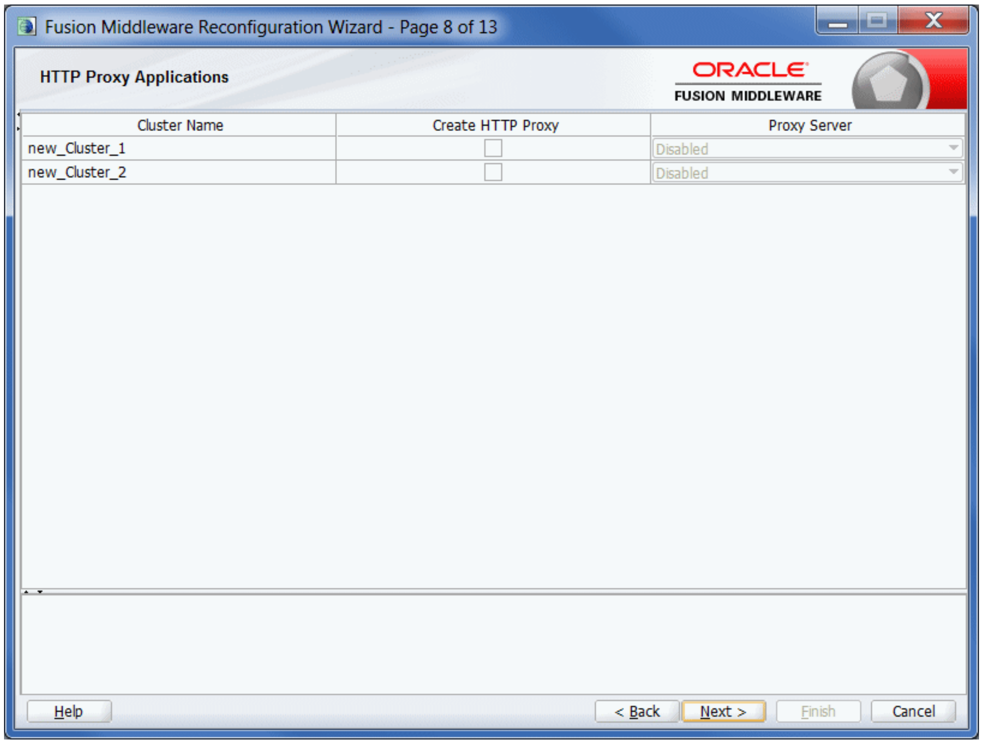
1. In the **Clusters** list box, select the cluster to which you want to assign a Managed Server.
2. Assign Managed Servers to the selected cluster in one of the following ways:
   - Double-click the name of the Managed Server in the **Servers** list box.
   - Select the Managed Server and click the right arrow.
   - Shift+click to select multiple Managed Servers; then, click the right arrow.

   The name of the Managed Server is removed from the **Servers** list box and added below the name of the target cluster in the **Clusters** list box.
3. Repeat steps 1 and 2 for each Managed Server to assign to a cluster.
4. Review the cluster assignments.

   If necessary, you can remove a Managed Server from a cluster in one of the following ways:
   - Double-click the name of the Managed Server in the **Clusters** list box.
   - Select the Managed Server and click the left arrow.

   The name of the Managed Server is removed from the **Clusters** list box and restored to the **Servers** list box.

5.20 HTTP Proxy Applications



Cluster Name	Create HTTP Proxy	Proxy Server
new_Cluster_1	☐	Disabled
new_Cluster_2	☐	Disabled

An HTTP proxy application acts as an intermediary for HTTP requests.

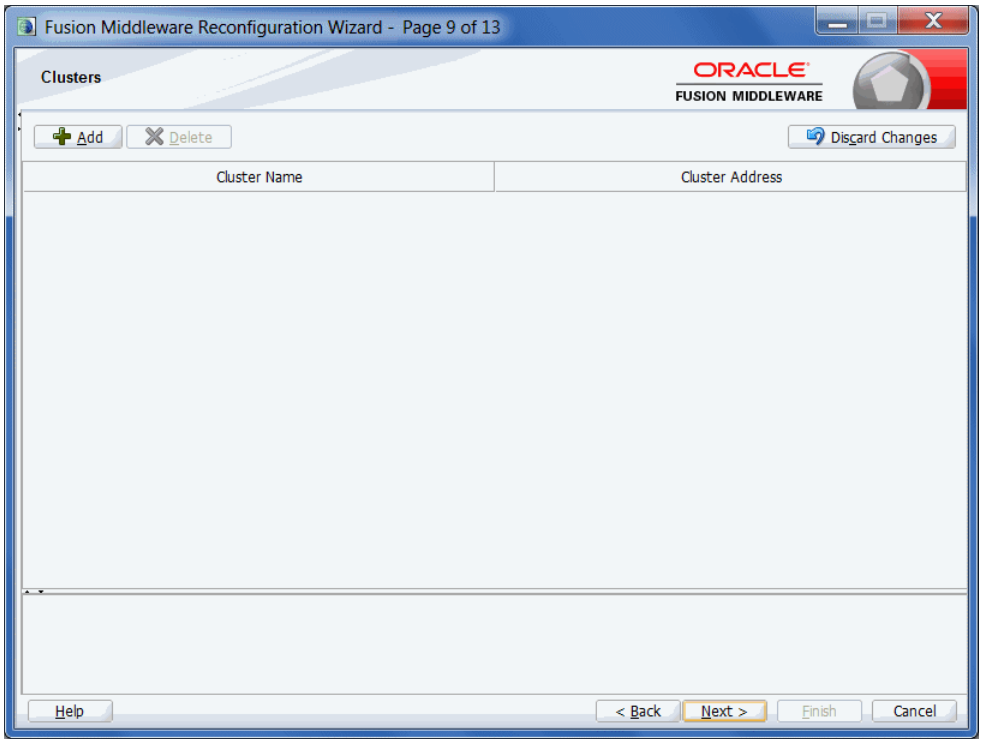
Use this screen to create an HTTP proxy application for each cluster, and specify the Managed Server on which the proxy application must be deployed.

This screen is displayed only if both of the following statements are true:

- At least one Managed Server is assigned to a cluster.
- At least one Managed Server is not assigned to any cluster.

Column	Description
Cluster Name	This column lists each cluster in the domain.
Create HTTP Proxy	Select this check box for each cluster on which you want to deploy the HTTP proxy application.
Proxy Server	This drop-down list contains all Managed Servers that are not assigned to a cluster. Select the Managed Server on which to deploy the proxy application. A proxy application named `OracleProxy4_`*`clustername_ servername`* is created and deployed on the Managed Server.

5.21 Coherence Clusters



This screen is displayed only if you included Coherence in the WebLogic Server installation. It lists the Coherence cluster that is automatically added to the domain.

Column	Description
Name	Accept the default cluster name or type a new name for the Coherence cluster. When updating a domain, if you have added additional Coherence clusters to the domain via WLST or the Administration Console, they are also listed here.
Coherence Listen Port	Enter the port number to use as the Coherence cluster listen port.

When including a Coherence cluster in a domain:

- All Managed Servers and clusters that you configure in the domain during the current Reconfiguration Wizard session or future Configuration Wizard sessions are automatically added to the Coherence cluster.
- Servers and clusters that already exist in the domain prior to running the Reconfiguration Wizard are automatically assigned to the Coherence cluster.
- When reconfiguring a domain, if the domain contains a only an Administration Server and no Managed Servers, the Administration Server is automatically assigned to the Coherence cluster. If, however, at least one Managed Server exists in the domain, the Administration Server is not assigned to the Coherence cluster.
- If there are multiple Coherence clusters in the domain (for example, you added a second Coherence cluster via WLST or the Administration Console), any servers or clusters that you subsequently add to the domain via the Reconfiguration Wizard are automatically assigned to the first Coherence cluster that is listed on the Coherence Clusters screen.

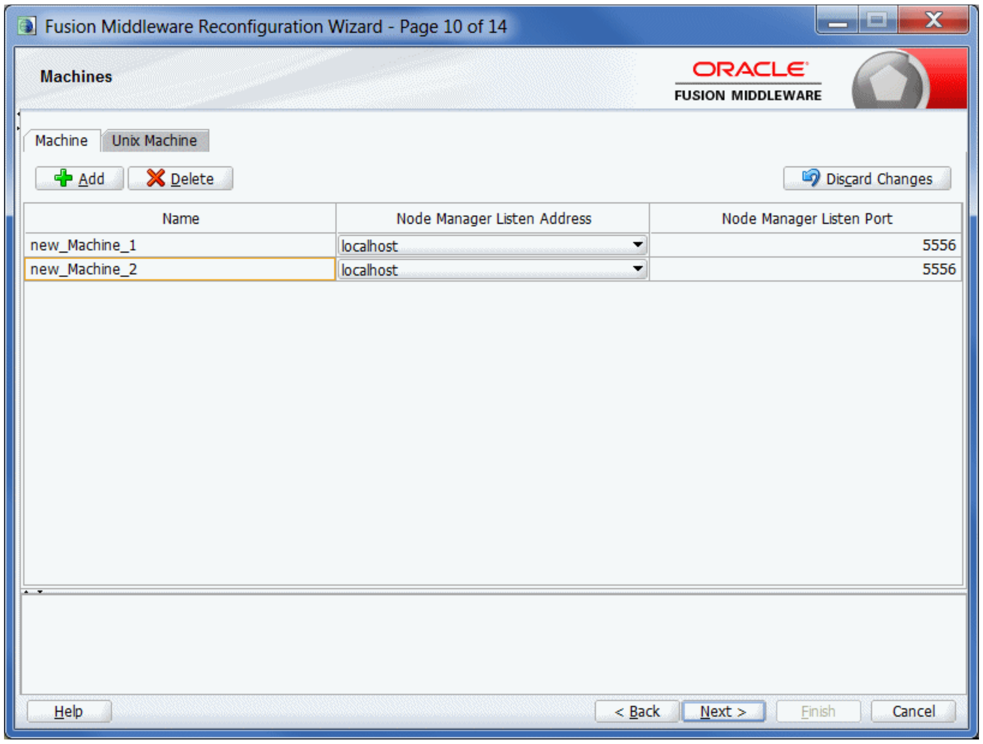
Related Topics

Managed Servers

Clusters

Assign Servers to Clusters

5.22 Machines



Name	Node Manager Listen Address	Node Manager Listen Port
new_Machine_1	localhost	5556
new_Machine_2	localhost	5556

In a WebLogic domain, the machine definitions identify physical units of hardware and are associated with the WebLogic Server instances or system components (such as OHS servers) that they host.

Use this screen to add or delete machines, or to modify the settings for an existing machine. Each machine has the following configuration settings.

Select the **Machine** tab (for Windows) or the **UNIX Machine** tab (for UNIX).

Column	Description
Name	Enter a valid machine name. The machine name is used to identify the machine within the WebLogic domain; it does not have to match the network name for the machine. The name must be unique among all component names within the domain.
Node Manager Listen Address	Select a value from the drop-down list for the listen address used by Node Manager to listen for connection requests. By default, the IP addresses defined for the local system and localhost are shown in the drop-down list. The default value is localhost. If you specify an IP address for a machine that hosts the Administration Server and you need to access the WebLogic Server Node Manager, you must disable host name verification. For more information, see "Using Host Name Verification" in *Administering Security for Oracle WebLogic Server*.
Node Manager Listen Port	Enter a valid value for the listen port used by Node Manager to listen for connection requests. The valid Node Manager listen port range is from 1 to 65535. The default value is 5556.

Column	Description
Post bind GID enabled	(UNIX machines only) Select this check box to enable a server running on this machine to bind to a UNIX group ID (GID) after it finishes all privileged startup actions. By default, this check box is not selected.
Post bind GID	(UNIX machines only) Enter the UNIX group ID (GID) under which a server on this machine will run after it finishes all privileged startup actions. Otherwise, the server continues to run under the group from which it was started. For this setting to take effect, you must select the **Post bind GID enabled** check box.
Post bind UID enabled	(UNIX machines only) Select this check box to enable a server running on this machine to bind to a UNIX user ID (UID) after it finishes all privileged startup actions. By default, this check box is not selected.
Post bind UID	(UNIX machines only) Enter the UNIX user ID (UID) under which a server on this machine will run after it finishes all privileged startup actions. Otherwise, the server continues to run under the account from which it was started. For this setting to take effect, you must select the **Post bind UID enabled** check box.

You might want to create machine definitions in situations such as the following:

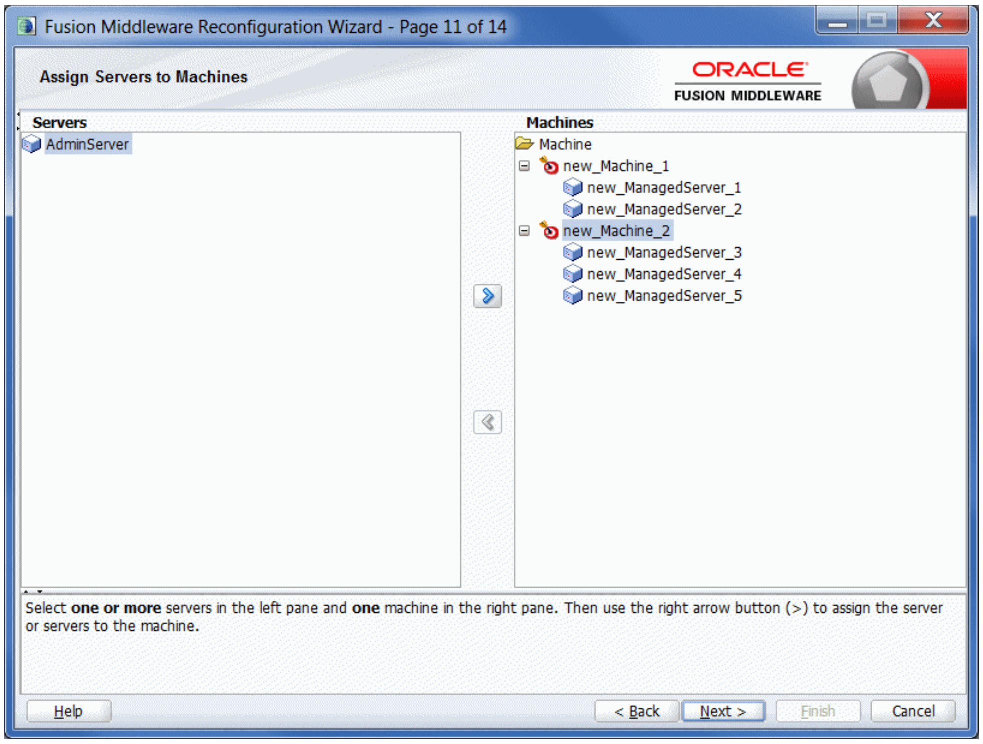
- The Administration Server uses the machine definition, with the Node Manager application, to start remote servers.
- WebLogic Server or other system components such as OHS use configured machine names when determining the server in a cluster that is best able to handle certain tasks, such as HTTP session replication. Those tasks are then delegated to the identified server.

Note: You must configure machines for each product installation that runs a Node Manager process. The machine configuration must include values for the listen address and port number parameters.

Click **Add** to add a new machine. The default name for a new machine is `new_[Unix]Machine_n`, where *n* starts at 1 and increments by 1 for each machine that you add.

Click **Delete** to delete an existing machine.

5.23 Assign Servers to Machines



Select **one or more** servers in the left pane and **one** machine in the right pane. Then use the right arrow button (>) to assign the server or servers to the machine.

Use this screen to assign WebLogic Server instances to each of the machines you defined.

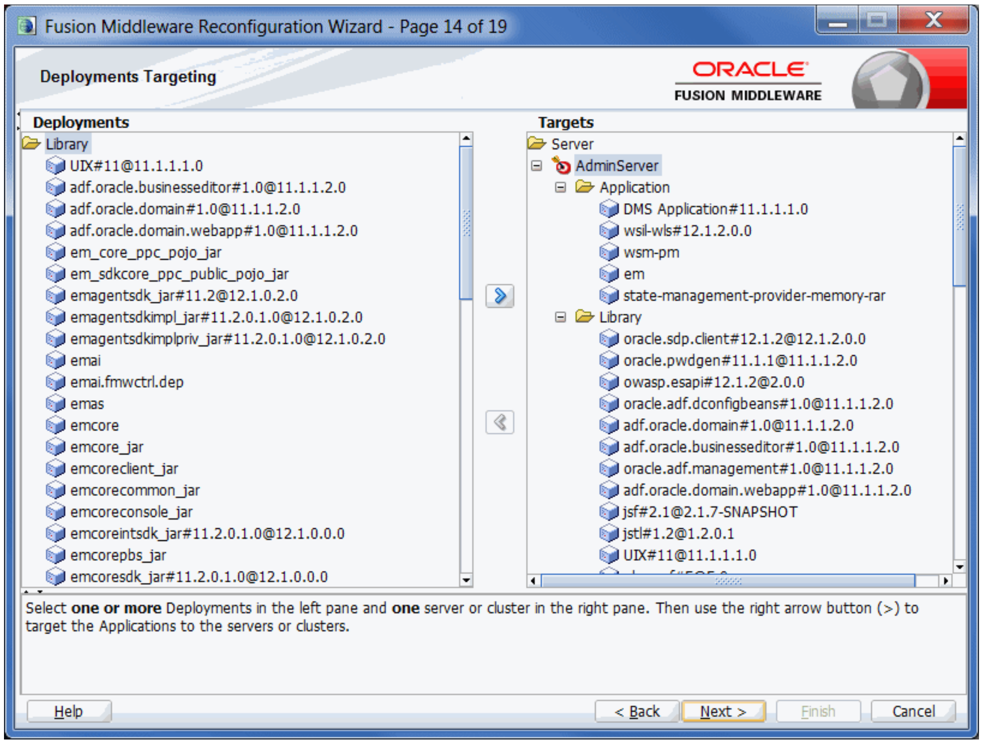
1. In the **Machine** list box, select the Windows or UNIX machine to which you want to assign a WebLogic Server instance.
2. Assign WebLogic Server instances to the selected machine in one of the following ways:
   - Double-click the WebLogic Server instance in the **Server** list box.
   - Select the appropriate WebLogic Server instance in the **Server** list box and click the right arrow.
   - Shift+click to select multiple servers in the **Server** list box; then, click the right arrow.

   The name of the WebLogic Server instance is removed from the **Server** list box and added, below the name of the target machine, in the **Machine** list box.
3. Repeat steps 1 and 2 for each WebLogic Server instance to assign to a machine.
4. Review the machine assignments.

   If necessary, you can remove a WebLogic Server instance from a machine in one of the following ways:
   - Double-click the name of the appropriate WebLogic Server instance in the **Machine** list box.
   - Select the appropriate WebLogic Server instance in the **Machine** list box and click the left arrow.

   The name of the WebLogic Server instance is removed from the **Machine** list box and restored to the **Server** list box.

5.24 Deployments Targeting


Fusion Middleware Reconfiguration Wizard - Page 14 of 19
Deployments Targeting
ORACLE
FUSION MIDDLEWARE
Deployments
Library
UIX#11@11.1.1.1.0
adf.oracle.businesseditor#1.0@11.1.1.2.0
adf.oracle.domain#1.0@11.1.1.2.0
adf.oracle.domain.webapp#1.0@11.1.1.2.0
em_core_ppc_pojo_jar
em_sdkcore_ppc_public_pojo_jar
emagentsdk_jar#11.2@12.1.0.2.0
emagentsdkimpl_jar#11.2.0.1.0@12.1.0.2.0
emagentsdkimplpriv_jar#11.2.0.1.0@12.1.0.2.0
emai
emai.fmwctrl.dep
emas
emcore
emcore_jar
emcoreclient_jar
emcorecommon_jar
emcoreconsole_jar
emcoreintsdk_jar#11.2.0.1.0@12.1.0.0.0
emcorepbs_jar
emcoresdk_jar#11.2.0.1.0@12.1.0.0.0
Targets
Server
AdminServer
Application
DMS Application#11.1.1.1.0
wsil-wls#12.1.2.0.0
wsm-pm
em
state-management-provider-memory-rar
Library
oracle.sdp.client#12.1.2@12.1.2.0.0
oracle.pwdgen#11.1.1@11.1.1.2.0
owasp.esapi#12.1.2@2.0.0
oracle.adf.dconfigbeans#1.0@11.1.1.2.0
adf.oracle.domain#1.0@11.1.1.2.0
adf.oracle.businesseditor#1.0@11.1.1.2.0
oracle.adf.management#1.0@11.1.1.2.0
adf.oracle.domain.webapp#1.0@11.1.1.2.0
jsf#2.1@2.1.7-SNAPSHOT
jstl#1.2@1.2.0.1
UIX#11@11.1.1.1.0
Help
< Back
Next >
Finish
Cancel

Select **one or more** Deployments in the left pane and **one** server or cluster in the right pane. Then use the right arrow button (>) to target the Applications to the servers or clusters.

Use this screen to target applications for deployment on servers or clusters.

Applications associated with the product for which you are configuring the domain are targeted automatically to the Managed Server created for that product or to the cluster to which that Managed Server is assigned. In this screen, you can target applications to additional servers and clusters.

To target an application deployment to a cluster or server:

1. In the Target list box, select the cluster or server on which you want to deploy applications.

   The name of the selected target is displayed as the title of the list box on the right.

2. In the *target_name* list box, select the check boxes corresponding to the applications to deploy on the selected target.

   The applications displayed here vary, depending on the products that you selected in the Select Domain Source screen, earlier in the wizard.

   Note: When you select a Managed Server in the Target list box, some of the check boxes in the *target_name* list box might be disabled, indicating applications that are already targeted at the cluster that contains the selected Managed Server.

   After you select applications, the names of the targeted clusters and servers are displayed in the Target column in the *target_name* list box.

3. Repeat steps 1 and 2 for the other clusters and servers, as required.

4. After making the required selections, click **Next**.

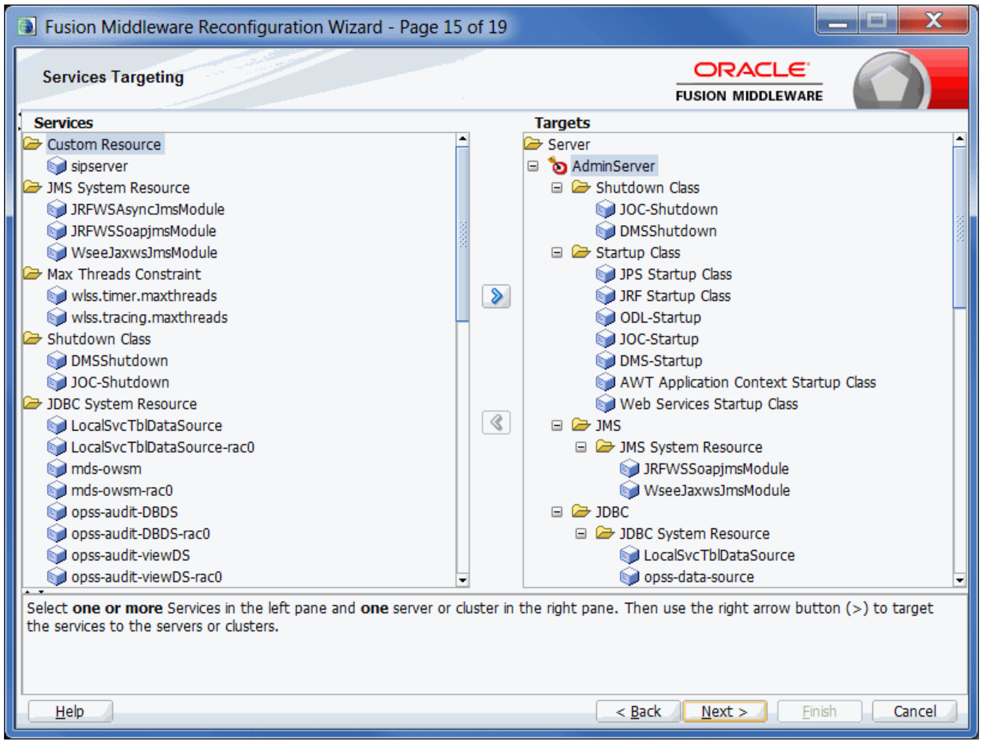
When you reconfigure a domain, if you delete a Managed Server or cluster to which applications are currently targeted, the Reconfiguration Wizard automatically retargets those applications as follows:

- If the applications were originally targeted solely to the Managed Server or cluster that you are now deleting (that is, after you delete the Managed Server or cluster, the applications would become untargeted in the modified domain), then the Reconfiguration Wizard automatically retargets the applications to all *eligible* targets.

  An eligible target is any cluster or Managed Server that is not defined in the configuration groups specification (`config-groups.xml` file) of an included template. Servers or clusters that are specified in `config-groups.xml` are essentially owned by the template and, therefore, are not eligible for automatic targeting.

- If the applications were originally targeted to multiple targets (including Managed Servers, clusters, and the Administration Server), and one of the targeted Managed Servers or clusters is deleted, then, in the extended domain, the Reconfiguration Wizard leaves the remaining target associations intact and does not attempt to retarget the applications.

5.25 Services Targeting



Select **one or more** Services in the left pane and **one** server or cluster in the right pane. Then use the right arrow button (>) to target the services to the servers or clusters.

Use this screen to target services to the appropriate Managed Servers or clusters.

Services that are associated with the product for which you are configuring the domain are targeted automatically, to the Managed Server created for that product or to the cluster to which that Managed Server is assigned. In this screen, you can target services to additional servers and clusters.

To target services to Managed Servers or clusters:

1. In the **Target** list box, select the cluster or server on which you want to deploy services.

   The name of the selected target is displayed as the title of the list box on the right.

2. In the *target_name* list box, select the check boxes corresponding to the services to deploy on the selected target.

   The services displayed here vary, depending on the products that you selected in the Select Domain Source screen earlier in the wizard.

   Note: When you select a Managed Server in the Target list box, some of the check boxes in the *target_name* list box might be disabled, indicating services that are already targeted at the cluster that contains the selected Managed Server.

   After you select services, the names of the targeted clusters and servers are displayed in the **Target** column in the *target_name* list box.

3. Repeat steps 1 and 2 for the other clusters and servers, as required.

4. After making the required selections, click **Next**.

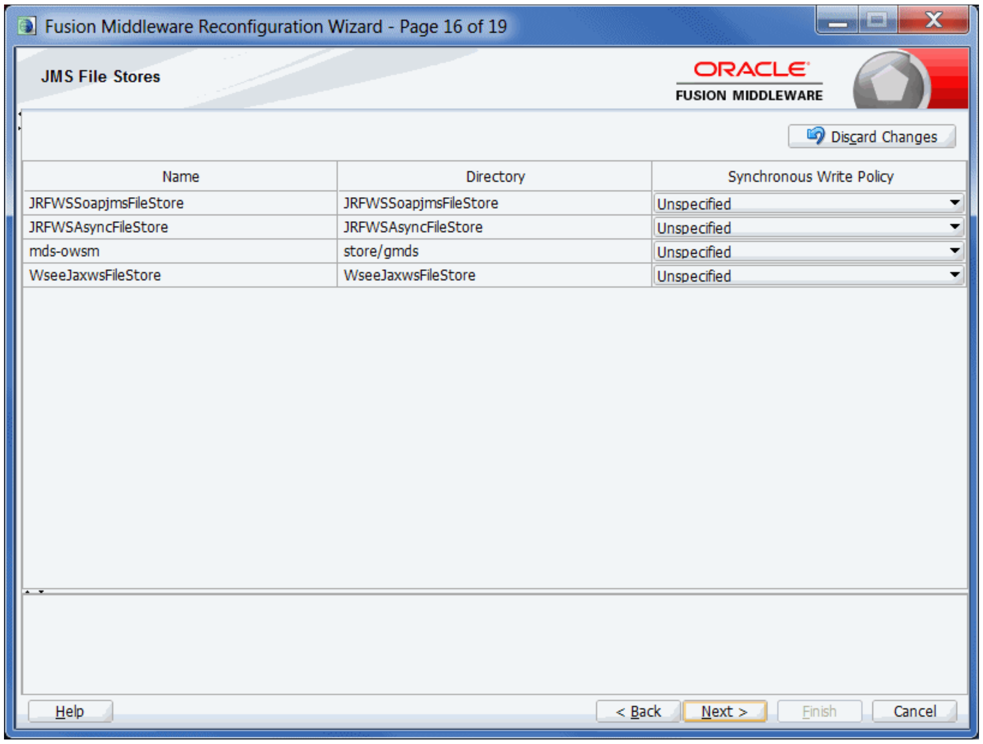
When you reconfigure a domain, if you delete a Managed Server or cluster to which services are currently targeted, the Reconfiguration Wizard automatically retargets those services as follows:

- If the services were originally targeted solely to the Managed Server or cluster that you are now deleting (that is, after you delete the Managed Server or cluster, the services would become untargeted in the modified domain), then the Reconfiguration Wizard automatically retargets the services to all *eligible* targets.

  An eligible target is any cluster or Managed Server that is not defined in the configuration groups specification (`config-groups.xml` file) of an included template. Servers or clusters that are specified in `config-groups.xml` are essentially owned by the template and, therefore, are *not* eligible for automatic targeting.

- If the services were originally targeted to multiple targets (including Managed Servers, clusters, and the Administration Server), and one of the targeted Managed Servers or clusters is deleted, then, in the extended domain, the Reconfiguration Wizard leaves the remaining target associations intact and does not attempt to retarget the services.

5.26 JMS File Stores



Name	Directory	Synchronous Write Policy
JRFWSSoapjmsFileStore	JRFWSSoapjmsFileStore	Unspecified
JRFWSAsyncFileStore	JRFWSAsyncFileStore	Unspecified
mds-owsm	store/gmds	Unspecified
WseeJaxwsFileStore	WseeJaxwsFileStore	Unspecified

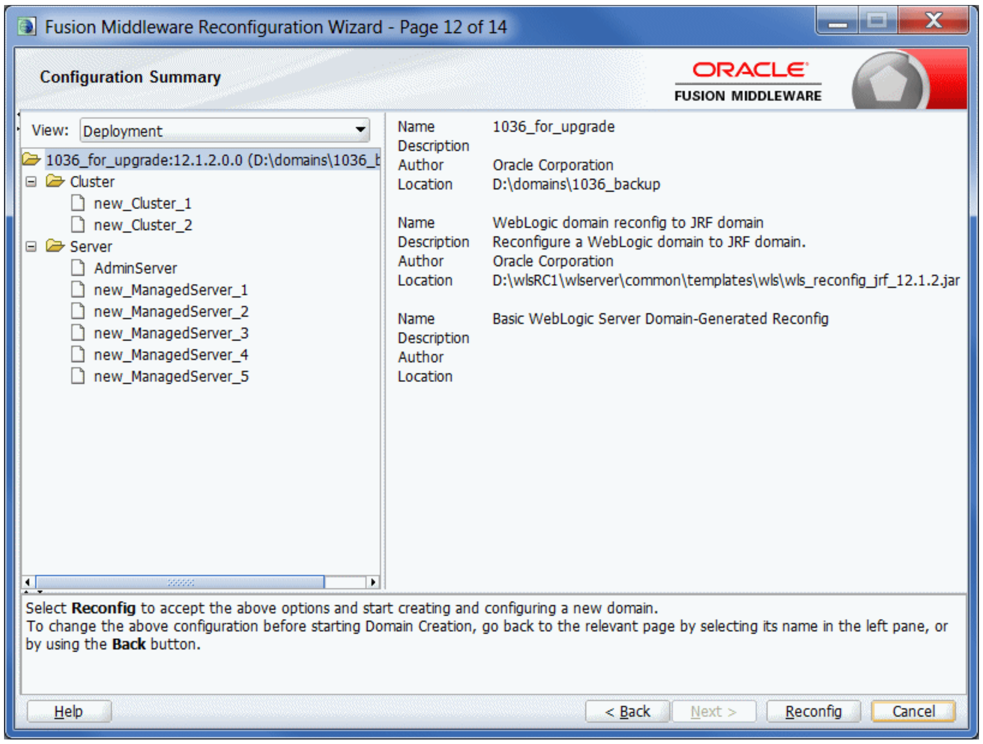
A JMS file store is a disk-based file in which persistent messages can be saved.

Use this screen to modify the JMS file stores that are configured in your domain. It contains the following fields.

Column	**Description**
Name	Enter a valid name for the JMS file store. The name must be a string of characters. The name of the JMS file store must be unique among all component names within the domain.
Directory	Enter the path of the directory (in your system) in which the JMS file store resides.
Synchronous write policy	From the drop-down list, select one of the following synchronous write policies to determine how the file store writes data to the disk. If the JMS file store is used exclusively for paging non-persistent messages to the disk, the synchronous write policy is ignored.
Cache-Flush	WebLogic Server enables the default file write behavior of the operating system and storage device, which typically includes caching and scheduling file writes, but forces a flush of the cache to disk before completing a transaction. For more information, see "Cache-Flush Policy" in*Administering Server Environments for Oracle WebLogic Server*.
Direct Write	WebLogic Server writes synchronously to a primary set of files in the location defined by the Directory attribute of the file store configuration using a native I/O wlfileio driver. For more information, see "Direct-Write Policy" in *Administering Server Environments for Oracle WebLogic Server*

Column	Description
Direct-Write-With-Cache	For most scenarios, Oracle recommends using the Direct-Write-With-Cache policy. When this policy is selected, WebLogic Server writes synchronously to a primary set of files in the location defined by the Directory attribute of the file store configuration using a native I/O wlfileio driver. For more information, see "Direct-Write-With-Cache Policy" in *Administering Server Environments for Oracle WebLogic Server*.
Disabled	WebLogic Server relies on the default file write behavior of the operating system and storage device. For more information, see "Disabled Policy" in *Administering Server Environments for Oracle WebLogic Server*.
Direct-Write	Write operations are performed directly to the disk. This policy is supported on Solaris and Windows. If this policy is active on an unsupported platform, the file store switches automatically to the cache-flush policy. For more information, see "Direct-Write Policy" in *Administering Server Environments for Oracle WebLogic Server*
Disabled	Transactions are complete as soon as the writes are cached in memory. When this policy is active, completion of transactions does not depend on waiting for writes to reach the disk. This setting affects performance, scalability, and reliability.

5.27 Configuration Summary



Name	1036_for_upgrade
Description	
Author	Oracle Corporation
Location	D:\domains\1036_backup
Name	WebLogic domain reconfig to JRF domain
Description	Reconfigure a WebLogic domain to JRF domain.
Author	Oracle Corporation
Location	D:\wlsRC1\wlserver\common\templates\wls\wls_reconfig_jrf_12.1.2.jar
Name	Basic WebLogic Server Domain-Generated Reconfig
Description	
Author	
Location	

Select **Reconfig** to accept the above options and start creating and configuring a new domain.
To change the above configuration before starting Domain Creation, go back to the relevant page by selecting its name in the left pane, or by using the **Back** button.

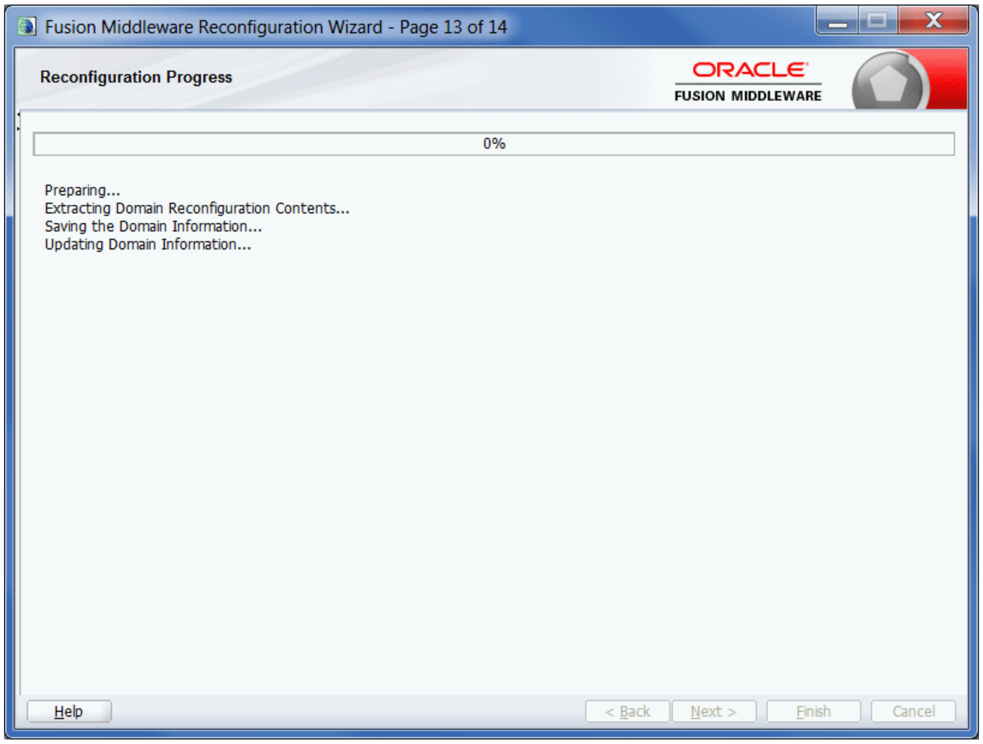
Review the detailed configuration settings of the domain before continuing.

You can limit the items that are displayed in the right-most panel by selecting a filter option from the **View** drop-down list.

If you need to change the configuration, click **Back** to return to the appropriate screen.

When done, click **Reconfig** to reconfigure the domain. The location of the domain does not change.

5.28 Reconfiguration Progress



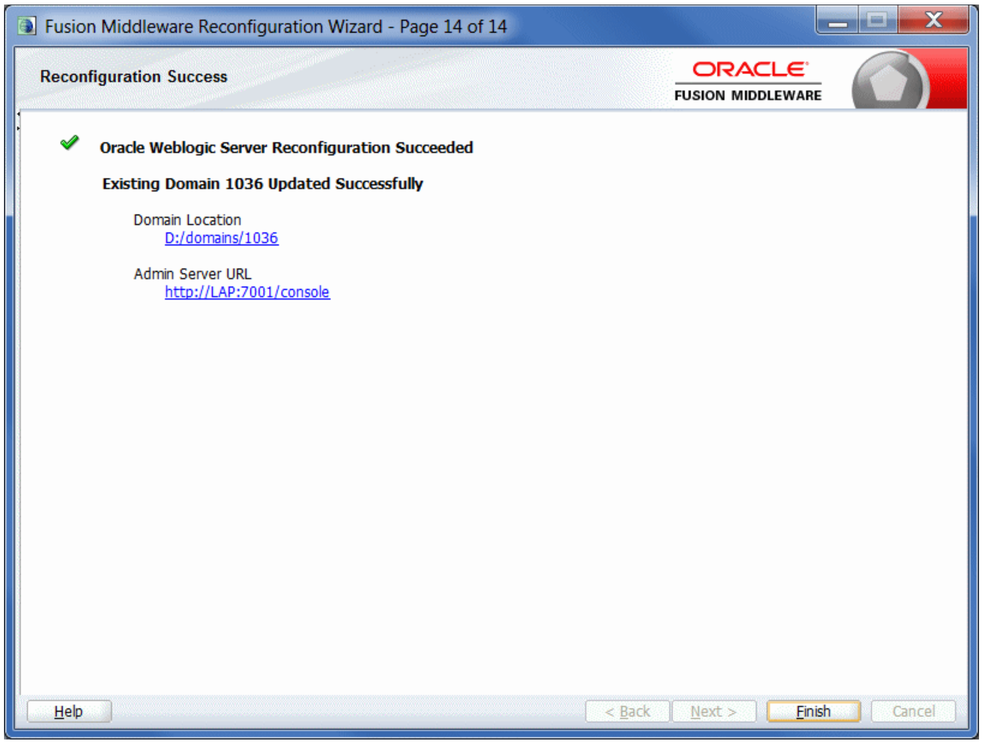
```
Preparing...
Extracting Domain Reconfiguration Contents...
Saving the Domain Information...
Updating Domain Information...
```

This screen displays the progress of the reconfiguration process. During this process:

- Domain information is extracted, saved, and updated
- Schemas, scripts, and other such files that support your Fusion Middleware products are updated.

When the process completes, click **Finish**.

5.29 Reconfiguration Success



Fusion Middleware Reconfiguration Wizard - Page 14 of 14

Reconfiguration Success

Oracle Weblogic Server Reconfiguration Succeeded

Existing Domain 1036 Updated Successfully

Domain Location
D:/domains/1036

Admin Server URL
http://LAP:7001/console

This screen indicates whether the reconfiguration process completed successfully or failed. It also displays the location of the domain that was reconfigured as well as the Administration Server URL (including the listen port).

If the reconfiguration process did not complete successfully, an error message is displayed to indicate the reason. Take appropriate action to resolve the issue. If you cannot resolve the issue, contact My Oracle Support.

A

WebLogic Server 12.1.2 Compatibility with Previous Releases

This section describes important compatibility information that you should consider before upgrading to WebLogic Server 12.1.2 from a WebLogic Server 10.3.*x* release.

See also "WebLogic Server Compatibility" in *Understanding Oracle WebLogic Server* and *What's New in Oracle WebLogic Server* for this and prior releases.

Compatibility considerations are provided in the following sections. The sections that apply to your situation depend on the WebLogic Server version from which you are upgrading to WebLogic Server 12.1.2. Refer to Table A–1 for a list of sections to which you should refer based on your current WebLogic Server version.

Table A–1 Sections Applying to Upgrades From Each WebLogic Server Version

Updating From This WebLogic Server Version	Refer to These Sections
12.1.1	Section A.1, "Maximum POST Size"
	Section A.2, "WLDF Schema Upgrade"
	Section A.3, "jdbc-connection-timeout-secs Element"
	Section A.4, "Commitment of Local Transactions"
10.3.5 and 10.3.6	All sections in the above row, plus:
	Section A.5, "JVM Settings"
	Section A.6, "Node Manager startScriptEnabled Default"
	Section A.7, "Enterprise Java Beans (EJBs)"
	Section A.8, "WebLogic Server 8.1 Web Services Stack Has Been Removed"
	Section A.9, "Universal Description and Discover (UDDI) Registry Has Been Removed"
	Section A.10, "Certicom SSL Implementation"
	Section A.11, "Coherence Version"
	Section A.12, "Deprecated and Obsolete Web Application Features"
	Section A.13, "Evaluation Database Changed From PointBase to Derby"
	Section A.14, "Data Source Profile Logging"
	Section A.15, "ONS Debugging"
	Section A.16, "Oracle Type 4 JDBC drivers from DataDirect"
	Section A.17, "Default Message Mode Has Changed"

Table A–1 (Cont.) Sections Applying to Upgrades From Each WebLogic Server Version

Updating From This WebLogic Server Version	Refer to These Sections
10.3.3 and 10.3.4	All sections in the above rows, plus: Section A.18, "Modifications to SSLMBean"
10.3.2	All sections in the above rows, plus: Section A.19, "New Web Services Features" Section A.20, "Introduction of JSSE" Section A.21, "Performance Enhancements for Security Policy Deployment" Section A.22, "ActiveCache" Section A.23, "Class Caching" Section A.24, "Deprecated JDBC Drivers" Section A.25, "Changes to weblogic.jms.extension API" Section A.26, "Persistent Store Updates"
10.3.1	All sections in the above rows, plus: Section A.27, "Oracle Internet Directory and Oracle Virtual Directory Authentication Providers"
10.3.0	All sections in the above rows, plus: Section A.28, "capacityIncrement Attribute" Section A.29, "Middleware Home Directory" Section A.30, "Resource Registration Name" Section A.31, "Servlet Path Mapping"

A.1 Maximum POST Size

A new session descriptor, `max-post-save-size`, has been added in WebLogic Server 12.1.2, which may impact existing applications. This descriptor sets the maximum size, in bytes, of the POST that will be saved or buffered by the application container during FORM authentication. The default value is 4096 bytes.

If your application posts a form for which the size exceeds 4096 bytes during FORM authentication, you must increase `max-post-save-size` to an appropriate value. Otherwise, a `MaxPostSizeExceededException` will occur in the browser.

A.2 WLDF Schema Upgrade

If you are using a JDBC-based store for WLDF event and harvester data, you must update or recreate the WLDF tables in your database. In the wls_events table, change the `THREADNAME` column from `varchar(128)` to `varchar(250)`. In the wls_hvst table, add the column `WLDFMODULE varchar(250) default NULL`. For more information, see "Configuring a JDBC-Based Store" in *Configuring and Using the Diagnostics Framework for Oracle WebLogic Server*.

This applies only to WLS standalone installations. For installations that include Fusion Middleware products, the schema upgrade process is done via the Oracle Upgrade Assistant.

A.3 jdbc-connection-timeout-secs Element

As of WebLogic Server 12.1.2, The `jdbc-connection-timeout-secs` element in the `weblogic.xml` deployment descriptor has been removed in WebLogic Server 12.1.2. If your application configures the `jdbc-connection-timeout-secs` element, you must remove it from the `weblogic.xml` deployment descriptor to prevent deployment of the application from failing.

A.4 Commitment of Local Transactions

As of WebLogic Server 12.1.2, local transactions on non-XA connections that were not committed or rolled back by the application are now explicitly committed by default when the connection is returned to the pool. In addition, the following two parameters have been added to set whether or not local transactions on non-XA and XA connections are committed when the connection pool is closed:

- `-Dweblogic.datasource.endLocalTxOnNonXAConWithCommit=false` can be used to avoid one extra DBMS round-trip with non-XA connections, for applications that are trusted to always complete their transaction explicitly. If this parameter is set to `false`, local transactions on non-XA connections will be implicitly committed or rolled back when a connection pool is closed, according to what the particular JDBC driver being used does when `setAutoCommit(true)` is called. Per the JDBC specification, that is to commit the transaction, but there is varied compliance among drivers. By default, or if the property is set to `true`, these transactions are now committed.
- `-Dweblogic.datasource.endLocalTXOnXAConWithCommit=true` can be used to commit local transactions on XA connections when a connection pool is closed. By default, these transactions are rolled back.

A.5 JVM Settings

When upgrading a WebLogic Server 10.3.*x* domain to a WebLogic Server 12.1.2 domain, you may have to:

- manually set the location of the Java endorsed directory (*`JRE_HOME`*`/lib/endorsed`) or directories.
- manually increase the permgen space and maximum permgen space

A.5.1 Setting the Location of the Java Endorsed Directory

In the following situations, you *do not* need to manually set the location of the Java endorsed directory or directories:

- you are using JDK7.
- you are using one of the JDKs that is installed with WebLogic Server 12.1.1.
- you are using WLS 12*c* domains and start scripts that were generated by domain creation via the WebLogic Server 12*c* Configuration Wizard, or your start scripts reference `commEnv.cmd/sh` as installed by the WebLogic Server installer, or both.

If none of these situations apply, and any one of the following situations apply, you must manually set the location of the Java endorsed directory in the command you use to start your Managed Servers:

- you are using Node Manager to start your Managed Servers, but you are not using a start script, that is `startScriptEnabled=false`. Note that as of WebLogic Server 12.1.1, the default value for `startScriptEnabled` is `true`.
- you are using custom start scripts, that is, start scripts that are not provided by Oracle.
- you are trying to create an empty domain using `java.weblogic.Server`.

In any of these cases, include the `java.endorsed.dirs` parameter in the Managed Server startup command.

```
startWeblogic.sh -Djava.endorsed.dirs=WL_HOME/endorsed
```

To specify multiple Java endorsed directories, separate each directory path with a colon (:).

Note: In all of the options described in this section, you must replace *`WL_HOME`* with the full path to your WebLogic Server installation.

You can also specify this value when calling `startServer` by passing the values as `jvmArgs`, or when calling `nmstart` by passing them as properties, such as:

```
wls:/nm/mydomain> prps =
makePropertiesObject("Arguments=-Djava.endorsed.dirs=/WL_
HOME/endorsed")
```

```
wls:/nm/mydomain> nmStart("AdminServer",props=prps)
```

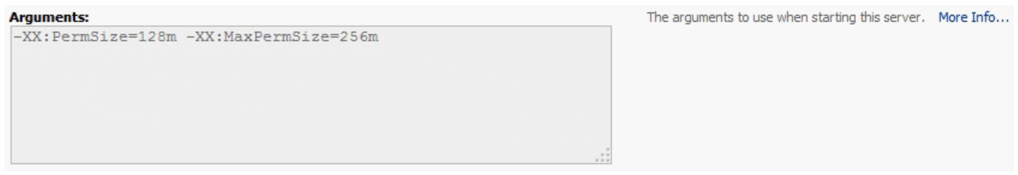
If you are using Node Manager to start the Managed Server, you can include the `-Djava.endorsed.dirs=/WL_HOME/endorsed")` parameter in the ServerStartMBean's `arguments` attribute, either using WLST or the Administration Console. If using the Administration Console, enter this parameter in the **Arguments** field on the server's **Configuration > Server Start** tab. This attribute will be applied when you call `start(server_name 'Server')` from a WLST client that is connected to the Administration Server or when you click on the **Start** button for the server in the Administration Console.

A.5.2 Setting permgen space

If you receive an `OutOfMemory: PermGen Space` error when starting a Managed Server, you have to manually set the permgen space to at least 128M and increase the maximum permgen space to at least 256M.

Note: In all of the options described here, you must replace *`WL_HOME`* with the full path to your WebLogic Server installation.

This can be done by specifying the following in the ServerStartMBean's `arguments` attribute, either using WLST or the Administration Console. If using the Administration Console, enter `-XX:PermSize=128m -XX:MaxPermSize=256m` in the **Arguments** field on the server's **Configuration > Server Start** tab.


Arguments:
```
-XX:PermSize=128m -XX:MaxPermSize=256m
```
The arguments to use when starting this server. More Info...

Note: If you plan to start the server via the Administration Console, you must apply the permgen settings prior to starting the server from the Administration Console. Otherwise the server may go into an unrecoverable state.

This attribute will be applied when you call `start(server_name 'Server')` from a WLST client that is connected to the Administration Server or when you click on the **Start** button for the server in the Administration Console.

Another method you can use is to start the Managed Server via the command line and specify the correct settings, as shown here:

(UNIX) `startManagedWebLogic.sh` *`server_name`* `-XX:PermSize=128m -XX:MaxPermSize=256m`

(Windows) `startManagedWebLogic.cmd` *`server_name`* `-XX:PermSize=128m -XX:MaxPermSize=256m`

You can also specify these values when calling `startServer` by passing the values as `jvmArgs`, or when calling `nmstart` by passing them as properties, such as:

```
wls:/nm/mydomain> prps = makePropertiesObject("Arguments=
-XX:PermSize=128m -XX:MaxPermSize=256m")
```

```
wls:/nm/mydomain> nmStart("AdminServer",props=prps)
```

A.6 Node Manager startScriptEnabled Default

As of WebLogic Server 12.1.1, the default value for `startScriptEnabled` has been changed to `true`. In all previous releases, the default was `false`. If you do not want to use a start script with Node Manager, change this value to `false` after upgrading.

A.7 Enterprise Java Beans (EJBs)

Oracle Kodo has been deprecated as of WebLogic Server 10.3.1. As of WebLogic Server 12.1.1, EclipseLink is the default JPA provider, replacing Kodo. Applications that continue to use Kodo as the persistence provider with WebLogic Server 12.1.2 will need to be updated. For more information, see "Updating Applications to Overcome Conflicts" in *Developing Enterprise JavaBeans for Oracle WebLogic Server*.

As of WebLogic Server 12.1.1, support for JPA 2.0 is built in. JPA 2.0 includes improvements and enhancements to domain modeling, object/relational mapping, `EntityManager` and `Query` interfaces, and the Java Persistence Query Language (JPQL), and more. For more information, see "Using JPA 2.0 with TopLink in WebLogic Server" in *Developing Enterprise JavaBeans for Oracle WebLogic Server*.

A.8 WebLogic Server 8.1 Web Services Stack Has Been Removed

The WebLogic Server 8.1 Web services stack has been removed in the WebLogic Server 12.1.1 release. Therefore, WebLogic Server 8.1 Web services applications will no longer work. Oracle recommends that you upgrade such applications to the WebLogic JAX-RPC or JAX-WS stacks, per the instructions in "Upgrading an 8.1 WebLogic Web Service to 12.1.x" on page 4-4.

A.9 Universal Description and Discover (UDDI) Registry Has Been Removed

The Universal Description and Discovery (UDDI) registry has been removed as of WebLogic Server 12.1.1. If you are still using UDDI and want to upgrade to WebLogic Server 12.1.1, Oracle recommends that you migrate to the Oracle Service Registry (OSR), which is UDDI 3.0 compliant.

A.10 Certicom SSL Implementation

As of WebLogic Server 12.1.1, the Certicom SSL Implementation has been removed. This change may require you to update system properties and debug switches as described in "Configuring SSL" in *Administering Security for Oracle WebLogic Server*.

A.11 Coherence Version

The WebLogic Server 12.1.1 installer includes Coherence 3.7.1. All servers in a cluster must use the same version of Coherence. Therefore, all cache servers in the cluster must be upgraded to Coherence 3.7.1.

A.12 Deprecated and Obsolete Web Application Features

For a list of Web application features that are deprecated or are not supported as of WebLogic Server 12.1.1, refer to the following:

- Information about deprecated functionality for WebLogic Server 11*g* Release 1 can be found on My Oracle Support at `https://support.oracle.com/`.

  In the Search Knowledge Base field, enter document ID `888028.1`.

- Information about functionality that is deprecated in WebLogic Server 12.1.1 can be found on My Oracle Support at `https://support.oracle.com/`. Search for **Deprecated Features**.

A.13 Evaluation Database Changed From PointBase to Derby

As of WebLogic Server 10.3.3, the evaluation database available from the WebLogic Server installation program has been changed from PointBase to Apache Derby. If you select the **Evaluation Database** option on the Choose Products and Components screen, the Derby database is installed in the *`WL_HOME`*`\common\derby` directory. If you select a Typical installation, Derby is installed by default.

If you have a domain based on PointBase and you want to continue using PointBase after upgrading the domain to WebLogic Server 10.3.3 or later, you must obtain a PointBase license from `http://www.pointbase.com`. Note that the full WLS installer does not preserve the PointBase installation directory. As an alternative to using PointBase, you can migrate the domain database to Derby.

For more information, see Section 3.6, "Upgrading a Domain that Uses an Evaluation Database."

A.14 Data Source Profile Logging

To provide better usability and performance, WebLogic Server 10.3.6 and higher uses a data source profile log to store events. See "Monitoring WebLogic JDBC Resources" in *Administering JDBC Data Sources for Oracle WebLogic Server*.

A.15 ONS Debugging

For WebLogic Server release 10.3.6 and higher, the package names for UCP and ONS are no longer repackaged. For information on how to set UPC and ONS debugging, see "Setting Debugging for UCP/ONS" in *Administering JDBC Data Sources for Oracle WebLogic Server*.

A.16 Oracle Type 4 JDBC drivers from DataDirect

As of WebLogic Server 10.3.6, Oracle Type 4 JDBC drivers from DataDirect are referred to as WebLogic-branded DataDirect drivers. Oracle has retired the documentation in *Type 4 JDBC Drivers for Oracle WebLogic Server* and no longer provides detailed information on DataDirect drivers. Oracle continues to provide information on how WebLogic-branded drivers are configured and used in WebLogic Server environments at "Using WebLogic-branded DataDirect Drivers" in *Developing JDBC Applications for Oracle WebLogic Server*. Oracle recommends reviewing DataDirect documentation for detailed information on driver behavior, see "Progress DataDirect for JDBC User's Guide Release 5.1" and "Progress DataDirect for JDBC Reference Release 5.1" at `http://www.datadirect.com/index.html`.

A.17 Default Message Mode Has Changed

As of WebLogic Server 12.1.1, the default messaging mode has been changed from multicast to unicast.

A.18 Modifications to SSLMBean

The SSLMBean has been modified as of WebLogic Server 10.3.5 to support additional SSL configuration capabilities, including the ability to enable or disable the JSSE adapter.

For more information, see the following documents:

- For a list of the differences in the way the JSSE SSL implementation handles the WebLogic system properties, see "System Property Differences Between the JSSE and Certicom SSL Implementations" in *Administering Security for Oracle WebLogic Server*.
- For more information about SSL support in WebLogic Server, see "Secure Sockets Layer (SSL)" in *Understanding Security for Oracle WebLogic Server*.
- For more information on JSEE, see "Java Secure Socket Extension (JSEE) Reference Guide" at `http://download.oracle.com/javase/7/docs/technotes/guides/security/jsse/JSSERefGuide.html`.

A.19 New Web Services Features

The following new features have been added in WebLogic Server as of release 10.3.3:

- Support for Web services atomic transactions—WebLogic Web services enable interoperability with other external transaction processing systems, such as WebSphere, JBoss, Microsoft .NET.
- Enhanced support for Web services in a clustered environment
- Enhanced monitoring of Web services and clients

- Attachment of Oracle WSM policies to WebLogic Web services using Fusion Middleware Control
- EclipseLink DBWS support for declarative Web service solution for accessing relational databases
- Method-Level policy attachment behavior change—Before WebLogic Server 10.3.3, if a policy was attached, through the Administration Console, to a method of one Web service, the policy was also attached to all methods of the same name for all Web services in that module. As of WebLogic Server 10.3.3, the policy is attached only to the method of the appropriate Web service.
- `policy:` prefix now removed from OWSM policy names
- Web services WSDL tab now removed—Before WebLogic Server 10.3.3, you could view the WSDL for the current Web service by selecting the **Configuration** > **WSDL** tab. The WSDL tab has been removed as of WebLogic Server 10.3.3.
- New development tools—Oracle JDeveloper and Oracle Enterprise Pack for Eclipse (OEPE)
- Integration with Oracle Enterprise Manager Fusion Middleware Control
- Support for Oracle WebLogic Services Manager (WSM) security policies
- Support for WS-SecureConversation 1.3 on JAX-WS and MTOM with WS-Security on JAX-WS

For more information, see "Web Services" in *What's New in Oracle WebLogic Server*.

A.20 Introduction of JSSE

As of WebLogic Server 10.3.3, Java Secure Socket Extension (JSSE) was introduced as an SSL implementation. JSSE is the Java standard framework for SSL and TLS and includes both blocking-I/O and non-blocking-I/O APIs, and a reference implementation including several commonly-trusted CAs.

A.21 Performance Enhancements for Security Policy Deployment

As of release 10.3.3, WebLogic Server includes a deployment performance enhancement for Deployable Authorization providers and Role Mapping providers that are thread safe. WebLogic Server by default supports thread-safe parallel modification to security policy and roles during application and module deployment. For this reason, deployable Authorization and Role Mapping providers configured in the security realm should support parallel calls. The WebLogic deployable XACML Authorization and Role Mapping providers meet this requirement.

However, if your custom deployable Authorization or Role Mapping providers do not support parallel calls, you must disable the parallel security policy and role modification and instead enforce a synchronization mechanism that results in each application and module being placed in a queue and deployed sequentially. You can turn on this synchronization enforcement mechanism from the Administration Console or by using the `DeployableProviderSynchronizationEnabled` and `DeployableProviderSynchronizationTimeout` attributes of the RealmMBean.

See "Enabling Synchronization in Security Policy and Role Modification at Deployment" in *Administering Security for Oracle WebLogic Server* for additional information.

A.22 ActiveCache

As of WebLogic Server 10.3.3, applications deployed on WebLogic Server can easily use Coherence data caches, and seamlessly incorporate Coherence*Web for session management and TopLink Grid as an object-to-relational persistence framework. Collectively, these features are called ActiveCache.

ActiveCache provides replicated and distributed data management and caching services that you can use to reliably make an application's objects and data available to all servers in a Coherence cluster.

For more information, see *Deploying Applications with Oracle WebLogic Server ActiveCache*.

A.23 Class Caching

As of release 10.3.3, WebLogic Server allows you to enable class caching. The advantages of using class caching are:

- Reduces server startup time.
- The package level index reduces search time for all classes and resources.

Class caching is supported in development mode when starting the server using a startWebLogic script. Class caching is disabled by default and is not supported in production mode. The decrease in startup time varies among different JRE vendors. For more information, see "Configuring Class Caching" in *Developing Applications for Oracle WebLogic Server*.

A.24 Deprecated JDBC Drivers

The following JDBC drivers are deprecated:

- WebLogic Type 4 JDBC driver for Oracle

  This driver was deprecated in WebLogic Server 10.3 and is now removed. Instead of using this deprecated driver, you should use the Oracle Thin Driver that is provided with WebLogic Server. For details about the Oracle Thin Driver, see "JDBC Drivers Installed with WebLogic Server" in *Administering JDBC Data Sources for Oracle WebLogic Server*.
- The Sybase JConnect 5.5 and 6.0 drivers 5.5 and 6.0 are removed from WebLogic Server as of release 10.3.3 due to an Oracle security policy regarding default installation of code samples. You can download the driver from Sybase or you can use the Oracle-branded JDBC driver for Sybase that is packaged with WebLogic Server.

A.25 Changes to weblogic.jms.extension API

As of WebLogic Server 10.3.3, the following internal methods of the `weblogic.jms.extensions.WLMessage` interface have been removed from the *Java API Reference for Oracle WebLogic Server*:

```
public void setSAFSequenceName(String safSequenceName);
public String getSAFSequenceName();
public void setSAFSeqNumber(long seqNumber);
public long getSAFSeqNumber();
```

Your applications should not use these internal methods. Internal methods may change or be removed in a future release without notice.

A.26 Persistent Store Updates

As of WebLogic Server 10.3.3, WebLogic File Store behavior and tuning have changed for default file stores and custom file stores.

A.27 Oracle Internet Directory and Oracle Virtual Directory Authentication Providers

Two new LDAP authentication providers were added to WebLogic Server 10.3.2: the Oracle Internet Directory Authentication Provider and the Oracle Virtual Directory Authentication Provider. These authentication providers can store users and groups in, and read users and groups from, the Oracle Internet Directory and Oracle Virtual Directory LDAP servers, respectively.

For information about configuring and using these new security providers, see "Configuring LDAP Authentication Providers" in *Administering Security for Oracle WebLogic Server*.

A.28 capacityIncrement Attribute

In WebLogic Server 10.3.1 and higher releases, the `capacityIncrement` attribute is no longer configurable and is set to a value of 1.

A.29 Middleware Home Directory

As of WebLogic Server 10.3.1, the notion of the BEA Home directory is replaced by the Middleware Home. The default path of this directory is <*drive:*>Oracle/Middleware. This change has the following impact on WebLogic Server:

- A new environment variable is introduced in several WebLogic scripts in 10.3.1 to represent the Middleware Home directory: *`MW_HOME`*. The directory to which this variable is set generally is the same as *`BEA_HOME`*, which is also still used in WebLogic Server scripts.
- By default, the WebLogic Server installation program selects <*drive:*>`Oracle/Middleware` as the root product installation directory. However, if a directory containing an existing WebLogic Server installation is detected, that directory is selected instead by default.
- The WebLogic Server 10.3.1 documentation now uses the term *Middleware Home*, instead of BEA Home. However, this revision is functionally only a change in terminology and does not imply that any WebLogic software, custom domains, or applications must be moved, or that any existing environment variables that represent those locations must be changed.

This change does not affect any existing WebLogic Server installations, custom domains, applications, or scripts on your computer. You may continue to use the *`BEA_ HOME`* environment variable as before.

A.30 Resource Registration Name

As of WebLogic Server 10.3.1, the behavior of the resource registration name for XA data source configurations has changed. In previous releases, the JTA registration name was simply the name of the data source. Now, the registration name is a combination of data source name and domain.

For more information, see "Registering an XAResource to Participate in Transactions" in *Developing JTA Applications for Oracle WebLogic Server*.

A.31 Servlet Path Mapping

As of version 2.3 of the Java Servlet Specification, the following syntax is used to define mappings:

- A servlet path string that contains only the `/` (slash) character indicates the default servlet of the application. The servlet path resolves to the request URI minus the context path; in this case, the path resolves to `null`.
- A String that begins with an `*` (asterisk) specifies an extension mapping.

These changes introduce a change in behavior with the following `HttpServletRequest` methods:

- `getPathInfo`
- `getServletPath`

To better illustrate the change in behavior, consider the request `/abc/def.html` that resolves to ServletA:

- If `/` maps to ServletA, then `servletPath="abc/def.html"` and `pathInfo=null`.
- If `/*` maps to ServletA, then `servletPath=""` and `pathInfo="abc/def.html"`.

To ensure that the path info returned is non-null, replace all occurrences of the `/` (slash) servlet mapping string with `/*`.

The Java Servlet Specification can be downloaded from the following location:

`http://www.oracle.com/technetwork/java/javaee/servlet/index.html`