

**Oracle® Fusion Middleware**

SmartUpgrade User's Guide

11g Release 1 (11.1.1)

**E15878-01**

October 2009

Oracle Fusion Middleware SmartUpgrade User's Guide, 11g Release 1 (11.1.1)

E15878-01

Copyright © 2009, Oracle and/or its affiliates. All rights reserved.

Primary Author: Peter LaQuerre

Contributors: John Bracken, Pascal Filion, Clyde Iwamoto, Nilesh Junnarkar, Jason Minton, Olga Peschansky, Wendy Puckett, Michael Rubino, Reza Shafii, Robert St. Jean, Gavin Steyn, Velmurugan Subramanian, Sitaraman Swaminathan, Frances Zhao, Lixin Zheng

This software and related documentation are provided under a license agreement containing restrictions on use and disclosure and are protected by intellectual property laws. Except as expressly permitted in your license agreement or allowed by law, you may not use, copy, reproduce, translate, broadcast, modify, license, transmit, distribute, exhibit, perform, publish, or display any part, in any form, or by any means. Reverse engineering, disassembly, or decompilation of this software, unless required by law for interoperability, is prohibited.

The information contained herein is subject to change without notice and is not warranted to be error-free. If you find any errors, please report them to us in writing.

If this software or related documentation is delivered to the U.S. Government or anyone licensing it on behalf of the U.S. Government, the following notice is applicable:

U.S. GOVERNMENT RIGHTS Programs, software, databases, and related documentation and technical data delivered to U.S. Government customers are "commercial computer software" or "commercial technical data" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, the use, duplication, disclosure, modification, and adaptation shall be subject to the restrictions and license terms set forth in the applicable Government contract, and, to the extent applicable by the terms of the Government contract, the additional rights set forth in FAR 52.227-19, Commercial Computer Software License (December 2007). Oracle USA, Inc., 500 Oracle Parkway, Redwood City, CA 94065.

This software is developed for general use in a variety of information management applications. It is not developed or intended for use in any inherently dangerous applications, including applications which may create a risk of personal injury. If you use this software in dangerous applications, then you shall be responsible to take all appropriate fail-safe, backup, redundancy, and other measures to ensure the safe use of this software. Oracle Corporation and its affiliates disclaim any liability for any damages caused by use of this software in dangerous applications.

Oracle is a registered trademark of Oracle Corporation and/or its affiliates. Other names may be trademarks of their respective owners.

This software and documentation may provide access to or information on content, products, and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third-party content, products, and services. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services.

---

---

# Contents

|   |      |
|---|------|
| <b>Preface</b> .....  | vii  |
| Audience .....  | vii  |
| Documentation Accessibility .....   | vii  |
| Related Documents .....   | viii |
| Conventions .....   | viii |
| <br>  |      |
| <b>1 Introduction to Oracle WebLogic Server SmartUpgrade</b>  |      |
| 1.1 What Is Oracle WebLogic Server SmartUpgrade? .....  | 1-1  |
| 1.2 Using SmartUpgrade As Part of Your Oracle Fusion Middleware Upgrade .....                             | 1-1  |
| 1.3 About the SmartUpgrade Report .....   | 1-2  |
| 1.4 About SmartUpgrade Artifact Generation.....   | 1-2  |
| 1.4.1 Generation of Deployment Descriptors.....   | 1-3  |
| 1.4.2 Generation of Web Services Artifacts .....  | 1-3  |
| 1.5 SmartUpgrade Oracle JDeveloper Integration and Command-Line Interface .....                           | 1-4  |
| 1.6 Downloading and Installing SmartUpgrade .....   | 1-4  |
| 1.6.1 Deinstalling SmartUpgrade Release 1.0 .....   | 1-4  |
| 1.6.2 Obtaining the SmartUpgrade ZIP Files .....  | 1-4  |
| 1.6.3 Installing the Oracle JDeveloper SmartUpgrade Extension .....                                       | 1-5  |
| 1.6.4 Installing the SmartUpgrade Command-Line Interface .....  | 1-6  |
| 1.6.5 Obtaining the Latest Documentation.....   | 1-7  |
| <br>  |      |
| <b>2 Using SmartUpgrade with Oracle JDeveloper</b>  |      |
| 2.1 Verifying the Oracle JDeveloper SmartUpgrade Extension Installation .....                             | 2-1  |
| 2.2 Starting and Using the Java EE Upgrade Wizard .....   | 2-2  |
| 2.2.1 Starting the Java EE Upgrade Wizard .....   | 2-2  |
| 2.2.1.1 Using the Java EE Upgrade Application Wizard.....   | 2-2  |
| 2.2.1.2 Using the Java EE Upgrade Project Wizard.....   | 2-3  |
| 2.2.2 Using the Java EE Upgrade Wizard.....   | 2-3  |
| 2.2.2.1 Summary of the Java EE Upgrade Wizard Pages .....   | 2-3  |
| 2.2.2.2 Types of Archives and Projects Supported by the SmartUpgrade<br>Oracle JDeveloper Extension ..... | 2-6  |
| 2.2.2.3 Limiting the Analysis to Specific SmartUpgrade Rule Categories.....                               | 2-6  |
| 2.2.2.4 Analyzing Applications Deployed on Oracle Application Server 10g<br>Release 2 (10.1.2) .....      | 2-7  |
| 2.2.3 Setting Properties When Upgrading OC4J Web Services with SmartUpgrade.....                          | 2-7  |

|         |   |      |
|---------|---|------|
| 2.2.3.1 | Learning About Each of the Web Services Artifact Generation Options .....                 | 2-7  |
| 2.2.3.2 | Setting the Mandatory Target Server Home Directory Option .....                           | 2-8  |
| 2.2.3.3 | Generating Instrumented Code for Performance Analysis.....                                | 2-8  |
| 2.2.3.4 | Packaging the Upgraded Web Services Artifacts as a Deployable<br>Enterprise Archive ..... | 2-8  |
| 2.2.4   | Continuing with the Upgrade of Web Services Artifacts .....                               | 2-8  |
| 2.3     | Using a SmartUpgrade Report.....  | 2-9  |
| 2.3.1   | Viewing a SmartUpgrade Report .....   | 2-10 |
| 2.3.2   | Using the Upgrade Findings Toolbar to Filter SmartUpgrade Findings.....                   | 2-10 |
| 2.3.3   | Using the Structure Window to Sort and Filter SmartUpgrade Findings.....                  | 2-11 |

### 3 Using the SmartUpgrade Command Line

|         |   |      |
|---------|---|------|
| 3.1     | Using the SmartUpgrade Java Command-Line.....   | 3-1  |
| 3.1.1   | Verifying Prerequisites and Locating the smartupgrade.jar File.....                                     | 3-1  |
| 3.1.2   | Basic Syntax of the SmartUpgrade Command-Line Interface .....   | 3-2  |
| 3.1.3   | Getting Help on the SmartUpgrade Command-Line Options.....  | 3-2  |
| 3.1.4   | Summary of the SmartUpgrade Command-Line Options .....  | 3-2  |
| 3.1.4.1 | List of SmartUpgrade Command-Line Interface Options .....   | 3-2  |
| 3.1.4.2 | Identifying a SmartUpgrade Locator.....   | 3-2  |
| 3.1.4.3 | Specifying More Than One Locator on the SmartUpgrade Command Line .....                                 | 3-4  |
| 3.1.4.4 | Summary of the SmartUpgrade Optional Command-Line Options .....   | 3-4  |
| 3.1.5   | Summary of the SmartUpgrade Command-Line Options Specific to Web<br>Services Artifact Generation.....   | 3-5  |
| 3.1.6   | Examples of Using the SmartUpgrade Command-Line Interface .....   | 3-11 |
| 3.1.6.1 | Using the SmartUpgrade Command-Line Interface to analyze an<br>Enterprise Archive (EAR) File .....      | 3-11 |
| 3.1.6.2 | Using the SmartUpgrade Command-Line Interface to Generate<br>Oracle WebLogic Server Artifacts .....     | 3-12 |
| 3.1.6.3 | Using the SmartUpgrade Command-Line Interface to Generate an<br>HTML Report.....                        | 3-12 |
| 3.1.6.4 | Limiting the Findings to Specific Rule Categories from the<br>SmartUpgrade Command-Line Interface ..... | 3-13 |
| 3.1.6.5 | Using the SmartUpgrade Command-Line Interface to Analyze 10g<br>Release 2 (10.1.2) Applications .....   | 3-14 |
| 3.2     | Integrating SmartUpgrade with Apache Ant .....  | 3-14 |

### 4 Using SmartUpgrade Generated Artifacts

|       |  |     |
|-------|--|-----|
| 4.1   | Locating the Artifacts Generated by SmartUpgrade .....                                   | 4-1 |
| 4.1.1 | Locating the Generated Artifacts and Output Folders.....                                 | 4-1 |
| 4.1.2 | Locating the Generated Artifacts in Oracle JDeveloper.....                               | 4-2 |
| 4.2   | Using Web Application Deployment Descriptor Artifacts Generated by<br>SmartUpgrade ..... | 4-3 |
| 4.2.1 | Differences Between OC4J and Oracle WebLogic Server Deployment Descriptors..             | 4-3 |
| 4.2.2 | List of Deployment Descriptor Elements Generated by SmartUpgrade .....                   | 4-4 |
| 4.2.3 | Using the Generated Web Application Deployment Descriptor Elements.....                  | 4-4 |
| 4.3   | Using Web Services Artifacts Generated by SmartUpgrade .....                             | 4-6 |
| 4.3.1 | Differences Between OC4J and Oracle WebLogic Server Web Services.....                    | 4-6 |

|       |   |      |
|-------|---|------|
| 4.3.2 | How Web Services Artifact Generation Differs From Deployment<br>Descriptor Generation.....              | 4-6  |
| 4.3.3 | Capabilities of the SmartUpgrade Web Services Upgrade .....   | 4-7  |
| 4.3.4 | About the Content of the Generated Web Services Output Directories.....                                 | 4-7  |
| 4.3.5 | Using Artifacts Generated for a Dedicated Web Services Application .....                                | 4-8  |
| 4.3.6 | Using Artifacts Generated for an Application with Both Web Services and<br>Other Java EE Elements ..... | 4-10 |
| 4.4   | Analyzing the Performance Impact of the Web Services Glue Code<br>Generated by SmartUpgrade .....       | 4-13 |

## Index



---

---

# Preface

This preface contains the following sections:

- [Audience](#)
- [Documentation Accessibility](#)
- [Related Documents](#)
- [Conventions](#)

## Audience

This manual is intended for Java EE application developers who are using Oracle JDeveloper to upgrade applications previously deployed on Oracle Containers for Java EE applications and who want to redeploy the applications on Oracle WebLogic Server and Oracle Fusion Middleware 11g.

## Documentation Accessibility

Our goal is to make Oracle products, services, and supporting documentation accessible to all users, including users that are disabled. To that end, our documentation includes features that make information available to users of assistive technology. This documentation is available in HTML format, and contains markup to facilitate access by the disabled community. Accessibility standards will continue to evolve over time, and Oracle is actively engaged with other market-leading technology vendors to address technical obstacles so that our documentation can be accessible to all of our customers. For more information, visit the Oracle Accessibility Program Web site at <http://www.oracle.com/accessibility/>.

### **Accessibility of Code Examples in Documentation**

Screen readers may not always correctly read the code examples in this document. The conventions for writing code require that closing braces should appear on an otherwise empty line; however, some screen readers may not always read a line of text that consists solely of a bracket or brace.

### **Accessibility of Links to External Web Sites in Documentation**

This documentation may contain links to Web sites of other companies or organizations that Oracle does not own or control. Oracle neither evaluates nor makes any representations regarding the accessibility of these Web sites.

## Deaf/Hard of Hearing Access to Oracle Support Services

To reach Oracle Support Services, use a telecommunications relay service (TRS) to call Oracle Support at 1.800.223.1711. An Oracle Support Services engineer will handle technical issues and provide customer support according to the Oracle service request process. Information about TRS is available at

<http://www.fcc.gov/cgb/consumerfacts/trs.html>, and a list of phone numbers is available at <http://www.fcc.gov/cgb/dro/trsphonebk.html>.

## Related Documents

For more information, see the following related documentation available in the Oracle Fusion Middleware 11g documentation library:

- Related Upgrade Documentation
  - *Oracle Fusion Middleware Upgrade Planning Guide*
  - *Oracle Fusion Middleware Upgrade Guide for Oracle SOA Suite, WebCenter, and ADF*
  - *Oracle Fusion Middleware Upgrade Guide for Oracle Portal, Forms, Reports, and Discoverer*
  - *Oracle Fusion Middleware Upgrade Guide for Oracle Identity Management*
- *Oracle Fusion Middleware Installation Planning Guide*
- *Oracle Fusion Middleware Administrator's Guide*

## Conventions

The following text conventions are used in this document:

| Convention      | Meaning  |
|-----------------|--|
| <b>boldface</b> | Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.         |
| <i>italic</i>   | 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. |



---

---

# Introduction to Oracle WebLogic Server SmartUpgrade

This chapter introduces the Oracle WebLogic Server SmartUpgrade software. It includes the following sections:

- [What Is Oracle WebLogic Server SmartUpgrade?](#)
- [Using SmartUpgrade As Part of Your Oracle Fusion Middleware Upgrade](#)
- [About the SmartUpgrade Report](#)
- [About SmartUpgrade Artifact Generation](#)
- [SmartUpgrade Oracle JDeveloper Integration and Command-Line Interface](#)
- [Downloading and Installing SmartUpgrade](#)

## 1.1 What Is Oracle WebLogic Server SmartUpgrade?

Oracle WebLogic Server SmartUpgrade is an Oracle JDeveloper extension and command-line utility that analyzes the applications you deployed previously on OC4J. It then offers advice and performs actions that can help you successfully redeploy the applications on Oracle WebLogic Server.

You can analyze an application archive, or you can analyze an application or project you have opened in Oracle JDeveloper. In addition, SmartUpgrade can analyze the OC4J server where you deployed your applications and provide advice on how to set up a similar configuration in Oracle WebLogic Server.

In addition to providing a comprehensive report with specific findings about each application, SmartUpgrade can also automate some of the upgrade tasks. For Web services applications, SmartUpgrade can upgrade your OC4J Web services automatically and optionally package the upgraded Web services as enterprise archives that can be deployed on Oracle WebLogic Server.

For Web-based applications, SmartUpgrade can generate specific types of artifacts that can speed up your application upgrade tasks.

## 1.2 Using SmartUpgrade As Part of Your Oracle Fusion Middleware Upgrade

Oracle WebLogic Server SmartUpgrade is one several Oracle software tools that can help you upgrade your entire Oracle Application Server or Oracle WebLogic Server environment to Oracle Fusion Middleware 11g.

As a result, you can use SmartUpgrade as part of an overall approach to upgrading your environment. Oracle recommends that you use SmartUpgrade in conjunction with the other upgrade-related Oracle Fusion Middleware documentation resources.

Refer to the following for more information:

- For information about upgrading all the components of your application server and middleware environment, refer to the *Oracle Fusion Middleware Upgrade Planning Guide*.
- For specific information about upgrading your Java EE applications and Oracle Application Server 10g environment to Oracle WebLogic Server, see the *Oracle Fusion Middleware Upgrade Guide for Java EE*.

## 1.3 About the SmartUpgrade Report

After SmartUpgrade analyzes a selected enterprise archive or Oracle JDeveloper application or project, SmartUpgrade generates an application upgrade report. You can then browse the report to identify potential issues to address before you can deploy it on Oracle WebLogic Server. Each issue is referred to as a "finding."

Each finding is assigned various attributes, including the level of priority and complexity. The priority and complexity attributes can help you plan the work required to modify and redeploy an application on Oracle WebLogic Server.

SmartUpgrade provides the following information within each finding:

- A reason for the finding.  
The reason defines why the finding exists. It often identifies a particular element or attribute in the deployment descriptors of the application archive or in the configuration files of the OC4J server. It also identifies the dependency APIs used in the application, which need to be changed before the application can be deployed to Oracle WebLogic Server.
- Advice for how to remedy the problem.  
The advice identifies the actions you must take to resolve the issues identified in the finding. The advice often includes instructions or references to documentation that can help you modify the application or the Oracle WebLogic Server environment appropriately.
- The implications of the finding. The implication is optional (not always shown) and indicates the differences in behavior (if any) that will be noticed in the upgraded application as a result of following the finding's advice.

## 1.4 About SmartUpgrade Artifact Generation

In addition to the upgrade report, SmartUpgrade can optionally generate specific types of application artifacts, such as Oracle WebLogic Server deployment descriptors, data source configurations, and Web services.

Instead of reviewing the report findings and creating these artifacts yourself, you can use SmartUpgrade to generate the artifacts for you, which can save you time and effort when upgrading your applications for Oracle WebLogic Server.

For more information, see the following sections:

- [Generation of Deployment Descriptors](#)
- [Generation of Web Services Artifacts](#)

## 1.4.1 Generation of Deployment Descriptors

If you configure SmartUpgrade to generate artifacts for a typical Java EE application, then SmartUpgrade analyzes the OC4J deployment descriptors within the application and, for specific elements within the OC4J-specific deployment descriptor, generates sample files that contain the equivalent deployment descriptor elements that can be used to deploy the application on Oracle WebLogic Server.

You can then use the generated deployment descriptors as a starting point for creating the require Oracle WebLogic Server deployment descriptors for your upgraded application.

For more information, see [Section 4.2, "Using Web Application Deployment Descriptor Artifacts Generated by SmartUpgrade"](#).

## 1.4.2 Generation of Web Services Artifacts

SmartUpgrade makes it possible to quickly and efficiently upgrade and deploy your Web services to Oracle WebLogic Server, while preserving the existing URLs and interfaces with remote applications in your environment.

When you configure SmartUpgrade to generate artifacts for a Web services application, SmartUpgrade analyzes the OC4J Web services and performs three distinct tasks automatically:

1. Uses Oracle Weblogic Server service generation tools to generate the service skeleton artifacts for each WSDL in the application.

The skeleton includes:

- A new set of value types or data transfer objects
- A new service endpoint interface (SEI) and a skeleton Web service
- Oracle WebLogic Server deployment descriptors

2. Provides the implementation of the skeleton Web service. This implementation is called "glue code" and is generated with required annotations supported by Oracle WebLogic Server.
3. As part of the implementation, dispatches requests to the original Web service implementation class, which is part of the OC4J application being upgraded. It performs this task by converting the new value types into the original value types.

SmartUpgrade ensures that the existing clients of the original OC4J Web services are not affected by the upgrade. The original WSDL contract is preserved and existing clients can continue to interoperate with the new Web service generated for Oracle WebLogic Server.

Note that the generated Web services glue code contains the required annotations.

For dedicated Web services applications, you can configure SmartUpgrade to package the generated Web services configuration into an archive that you can deploy directly to Oracle WebLogic Server.

For more information, see [Section 4.3, "Using Web Services Artifacts Generated by SmartUpgrade"](#).

## 1.5 SmartUpgrade Oracle JDeveloper Integration and Command-Line Interface

SmartUpgrade is available as an Oracle JDeveloper extension that can be installed using the Oracle JDeveloper **Check for Updates** feature. It is also available as a command-line tool.

Refer to the following resources for details:

- For information on using the Oracle JDeveloper extension, see [Chapter 2, "Using SmartUpgrade with Oracle JDeveloper"](#).
- For information about using the command-line interface, see [Chapter 3, "Using the SmartUpgrade Command Line"](#)

## 1.6 Downloading and Installing SmartUpgrade

Refer to the following sections for more information about downloading and installing SmartUpgrade:

- [Deinstalling SmartUpgrade Release 1.0](#)
- [Obtaining the SmartUpgrade ZIP Files](#)
- [Installing the Oracle JDeveloper SmartUpgrade Extension](#)
- [Installing the SmartUpgrade Command-Line Interface](#)
- [Obtaining the Latest Documentation](#)

### 1.6.1 Deinstalling SmartUpgrade Release 1.0

If you have previously installed SmartUpgrade Release 1.0, then you must remove the existing SmartUpgrade installation before you install SmartUpgrade Release 1.1.

To remove SmartUpgrade Release 1.0:

1. Stop Oracle JDeveloper.
2. Delete the `oracle.smartupgrade.weblogic` folder and its contents from the Middleware home where you installed Oracle JDeveloper:  

```
MW_HOME/jdeveloper/modules/oracle.smartupgrade.weblogic
```
3. Delete the `oracle.jdeveloper.smartupgrade.weblogic.jar` file from the Middleware home:  

```
MW_HOME/jdeveloper/jdev/extensions/oracle.jdeveloper.smartupgrade.weblogic.jar
```
4. Delete the `toplink_patch.jar` file from the Middleware home:  

```
MW_HOME/jdeveloper/jdev/lib/patches/toplink_patch.jar
```
5. Start Oracle JDeveloper.

### 1.6.2 Obtaining the SmartUpgrade ZIP Files

You can obtain SmartUpgrade in one of two ways:

- From the Oracle Fusion Middleware Companion CD-ROM, which is part of the Oracle Fusion Middleware CD-ROM pack
- From the Oracle Technology Network (OTN)

For example, to download and unpack the SmartUpgrade ZIP file from OTN:

1. Use your Web browser to navigate to the following URL on OTN:  
<http://www.oracle.com/technology/products/middleware/upgrade/index.html>

Note that the first time you access OTN, you will have to register. Registration is free and provides you with access to all the resources on OTN.

2. Click the link to download Oracle WebLogic Server SmartUpgrade.  
 At the time this document was published, the SmartUpgrade download link was located under the Free Downloads image on the right side of the page.
3. Follow the instructions on the screen to download a ZIP file that contains all the required SmartUpgrade files.
4. Unpack the ZIP file in a temporary directory on your local disk.

[Table 1–1](#) describes the contents of the ZIP file.

**Table 1–1 Contents of the SmartUpgrade ZIP File**

| File                        | Description   |
|-----------------------------|---|
| readme.txt                  | A text file that describes the files in the ZIP file.   |
| smartupgrade.zip            | A ZIP file containing the files required to use the SmartUpgrade command-line interface.          |
| releasenotes.txt            | A text file that describes late-breaking information about this SmartUpgrade download.            |
| jdeveloper_smartupgrade.zip | A ZIP file containing the files required to install the SmartUpgrade Oracle JDeveloper extension. |

### 1.6.3 Installing the Oracle JDeveloper SmartUpgrade Extension

To install the Oracle JDeveloper SmartUpgrade extension from the contents of the downloadable ZIP file:

1. Verify that you are currently running Oracle JDeveloper 11g.  
 If necessary, download and install Oracle JDeveloper 11g from the following location on OTN:  
<http://www.oracle.com/technology/products/jdev/index.html>  
 For Oracle JDeveloper installation instructions, refer to the *Oracle Fusion Middleware Installation Guide for Oracle JDeveloper*.
2. If you have previously installed SmartUpgrade Release 1.0, then deinstall Release 1.0, using the instructions in [Section 1.6.1, "Deinstalling SmartUpgrade Release 1.0"](#).
3. Start Oracle JDeveloper.
4. Select **Check for Updates** from the **Help** menu.
5. Click **Next** on the Welcome page of the Check for Updates wizard.
6. On the Source page, select **Install From Local File**.
7. Click **Browse** to locate and select the `jdeveloper_smartupgrade.zip` file you obtained and unpacked in [Section 1.6.2, "Obtaining the SmartUpgrade ZIP Files"](#).
8. Click **Finish** to install the SmartUpgrade extension.

9. Restart Oracle JDeveloper.

After you restart Oracle JDeveloper, refer to [Chapter 2, "Using SmartUpgrade with Oracle JDeveloper"](#).

---



---

**Note:** Future updates to Oracle JDeveloper will be available by selecting one of the **Search Update Centers** options in the Check for Updates wizard as they are available.

---



---

## 1.6.4 Installing the SmartUpgrade Command-Line Interface

When you install the SmartUpgrade Oracle JDeveloper extension, the files required to run the command-line interface are installed automatically.

To verify that the command-line interface has been installed, locate the `smartupgrade.jar` file in the following directory after you install the SmartUpgrade extension:

```
MW_HOME\jdeveloper\jdev\extensions\oracle.jdeveloper.smartupgrade.weblogic\
```

If you do not install the Oracle JDeveloper extension, and you want to install only the SmartUpgrade command-line interface, you can do so from the contents of the downloadable ZIP file or the SmartUpgrade files available on the Oracle Fusion Middleware 11g Release 1 (11.1.1.2.0) Companion CD-ROM.

To install and configure the SmartUpgrade command-line interface without using Oracle JDeveloper:

1. Verify that you have installed and configured one of the following prerequisites:
  - Java 2 Standard Edition or Enterprise Edition Version 1.6 or later  
For more information, refer to the following Web site for information on downloading the Java 1.6 Software Development Kit (SDK) or Java Runtime (JRE):  
<http://java.sun.com/>
  - Apache Ant Version 1.7 or later  
Apache Ant 1.7 is available is installed as part of any Oracle WebLogic Server 11g installation. You can also download it from the following URL:  
<http://ant.apache.org/>
2. Unpack the `smartupgrade.zip` file into a permanent directory where you will run the program.  
Oracle includes the `smartupgrade.zip` file as part of the SmartUpgrade download from the Oracle Technology Network (OTN), as well as on the Oracle Fusion Middleware 11g Release 1 (11.1.1.2.0) Companion CD-ROM.
3. Verify that the following files are unpacked into the directory:
  - `smartupgrade.jar`
  - `Manifest.mf`
  - `readme.txt`
4. For get started using the command-line interface, refer to [Chapter 3, "Using the SmartUpgrade Command Line"](#).

## 1.6.5 Obtaining the Latest Documentation

For the most up-to-date information about SmartUpgrade, including tutorials, data sheet, and the most recent version of this guide, refer to the Oracle Fusion Middleware Upgrade page on the Oracle Technology Network (OTN):

<http://www.oracle.com/technology/products/middleware/upgrade/index.html>





---

---

# Using SmartUpgrade with Oracle JDeveloper

This chapter describes how to use SmartUpgrade within Oracle JDeveloper. Refer to the following sections for more information:

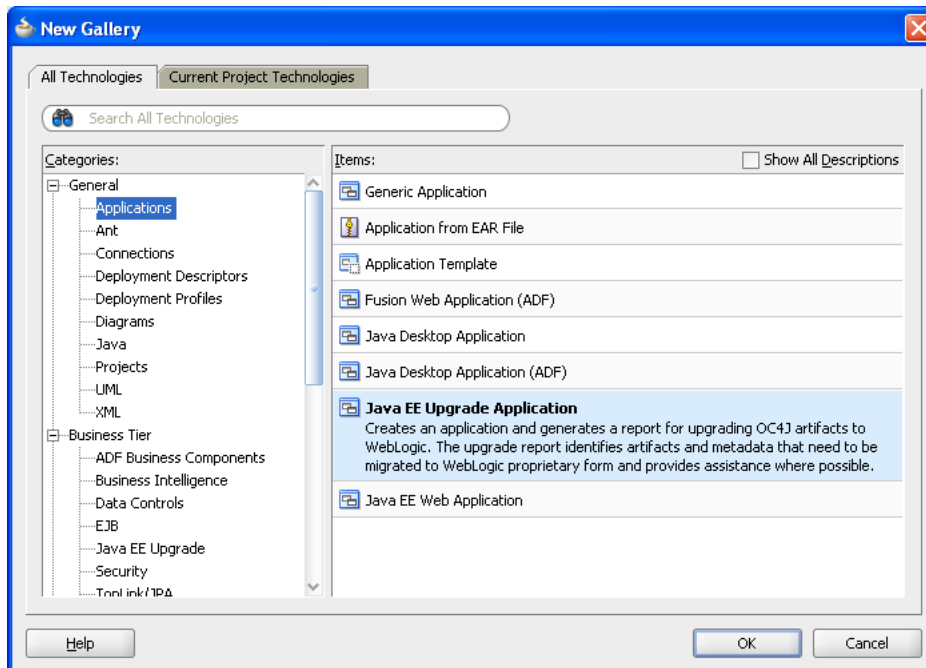
- [Verifying the Oracle JDeveloper SmartUpgrade Extension Installation](#)
- [Starting and Using the Java EE Upgrade Wizard](#)
- [Using a SmartUpgrade Report](#)

## 2.1 Verifying the Oracle JDeveloper SmartUpgrade Extension Installation

Before you begin, verify that the Oracle JDeveloper SmartUpgrade extension has been installed and configured:

1. Start Oracle JDeveloper.
2. Select **File > New**.
3. In the Categories tree of the New Gallery, select **General** then **Applications**.
4. In the Items list, verify that the **Java EE Upgrade Application** option appears in the list of Items, as shown in [Figure 2-1](#).

If this option does not appear in the list, then refer to [Section 1.6, "Downloading and Installing SmartUpgrade"](#).

**Figure 2–1 Java EE Upgrade Application Item in the New Gallery Dialog**

## 2.2 Starting and Using the Java EE Upgrade Wizard

To generate a SmartUpgrade report or to optionally generate artifacts to help you upgrade your applications, you use the Java EE Upgrade Wizard.

Refer to the following sections for more information:

- [Starting the Java EE Upgrade Wizard](#)
- [Using the Java EE Upgrade Wizard](#)
- [Setting Properties When Upgrading OC4J Web Services with SmartUpgrade](#)
- [Continuing with the Upgrade of Web Services Artifacts](#)

### 2.2.1 Starting the Java EE Upgrade Wizard

There are two ways to start Java EE Upgrade wizard, which guides you through the process of analyzing your application with SmartUpgrade.

The following sections describe these two options:

- [Using the Java EE Upgrade Application Wizard](#)
- [Using the Java EE Upgrade Project Wizard](#)

#### 2.2.1.1 Using the Java EE Upgrade Application Wizard

To start the Java EE Upgrade wizard and save the results to a new Oracle JDeveloper application:

1. Choose **File > New** to display the New Gallery.
2. In the Categories tree, select **General** then **Applications**.
3. In the Items list, double-click **Java EE Upgrade Application**([Figure 2–1](#)).

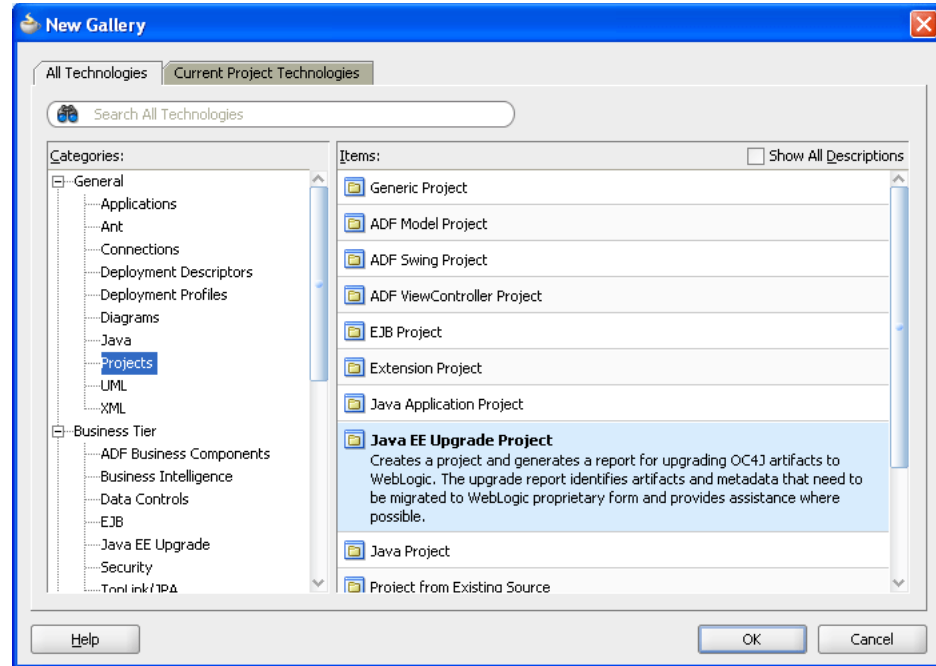
### 2.2.1.2 Using the Java EE Upgrade Project Wizard

To start the Java EE Upgrade wizard and save the results to a new project within an existing Oracle JDeveloper application:

1. Choose **File > New** to display the New Gallery.
 

You can also right-click an existing project and select **New** from the context menu.
2. In the Categories tree, select **General** then **Projects**.
3. In the Items list, double-click **Java EE Upgrade Project** (Figure 2–2).

**Figure 2–2** Java EE Upgrade Project Item in the New Gallery Dialog



## 2.2.2 Using the Java EE Upgrade Wizard

Refer to the following sections for information about using the Java EE Upgrade wizard:

- [Summary of the Java EE Upgrade Wizard Pages](#)
- [Types of Archives and Projects Supported by the SmartUpgrade Oracle JDeveloper Extension](#)
- [Limiting the Analysis to Specific SmartUpgrade Rule Categories](#)
- [Analyzing Applications Deployed on Oracle Application Server 10g Release 2 \(10.1.2\)](#)

### 2.2.2.1 Summary of the Java EE Upgrade Wizard Pages

Table 2–1 describes the pages of the Java EE Upgrade wizard. Note that the pages that appear will vary depending on how you started the wizard and the type of report you want to generate.

**Table 2–1 Summary of the Java EE Upgrade Application Wizard Pages**

| Page Name                              | Description   | More Information   |
|--|---|--|
| Application Name                       | <p>This page appears only if you use the Java EE Upgrade Application wizard.</p> <p>Enter a name for the new Oracle JDeveloper application that SmartUpgrade creates after analyzing your OC4J application.</p> <p>The resulting upgrade report will be saved in a project inside the application.</p>  | Press F1 or click <b>Help</b> on the wizard page.  |
| Project Name                           | <p>Enter a name for the default project that SmartUpgrade creates after analyzing the selected application or project.</p> <p>The resulting upgrade report will be saved in this project.</p>   | Press F1 or click <b>Help</b> on the wizard page.  |
| Upgrade Report Type (Figure 2–3)       | <p>Select the type of upgrade report you want to create.</p> <p>Note that if you are creating a new application, then the option to analyze an existing application does not appear on the page.</p> <p>To analyze one or more existing Oracle JDeveloper projects, you must use the Java EE Upgrade Project wizard.</p>  | Press F1 or click <b>Help</b> on the wizard page.  |
| OC4J Artifacts (Figure 2–4)            | <p>The content of this page varies, depending upon the upgrade report type you select on the previous page.</p> <p>Select the enterprise archives, Oracle JDeveloper projects, or OC4J server directory you want to analyze, and specify any related options available on the page.</p> <p>In addition, you can:</p> <ul style="list-style-type: none"> <li>■ Indicate whether or not you want to generate artifacts in addition to the upgrade report.</li> <li>■ Click <b>Advanced</b> to limit the analysis to specific SmartUpgrade rules and to indicate the version of OC4J you are using.</li> </ul> | <p>Press F1 or click <b>Help</b> on the wizard page.</p> <p><a href="#">Section 2.2.2.2, "Types of Archives and Projects Supported by the SmartUpgrade Oracle JDeveloper Extension"</a></p> <p><a href="#">Section 2.2.2.3, "Limiting the Analysis to Specific SmartUpgrade Rule Categories"</a></p> <p><a href="#">Section 2.2.2.4, "Analyzing Applications Deployed on Oracle Application Server 10g Release 2 (10.1.2)"</a></p> |
| OC4J Web Services Upgrade (Figure 2–5) | <p>This page appears if the following are true:</p> <ul style="list-style-type: none"> <li>■ You select <b>Generate Artifacts</b> on the OC4J Artifacts wizard page.</li> <li>■ The application contains any Web services</li> <li>■ The "web-services" rule category is selected in Rules Management dialog, which you can display by clicking <b>Advanced</b> on the OC4J Artifacts page.</li> <li>■ You selected an enterprise archive; you can analyze or generate artifacts for Web services only when you select an existing EAR file as the input to the analysis.</li> </ul>                        | <p><a href="#">Section 2.2.3, "Setting Properties When Upgrading OC4J Web Services with SmartUpgrade"</a></p> <p><a href="#">Section 2.2.2.2, "Types of Archives and Projects Supported by the SmartUpgrade Oracle JDeveloper Extension"</a></p>   |

Figure 2-3 Java EE Upgrade Application Wizard - Upgrade Report Type Page

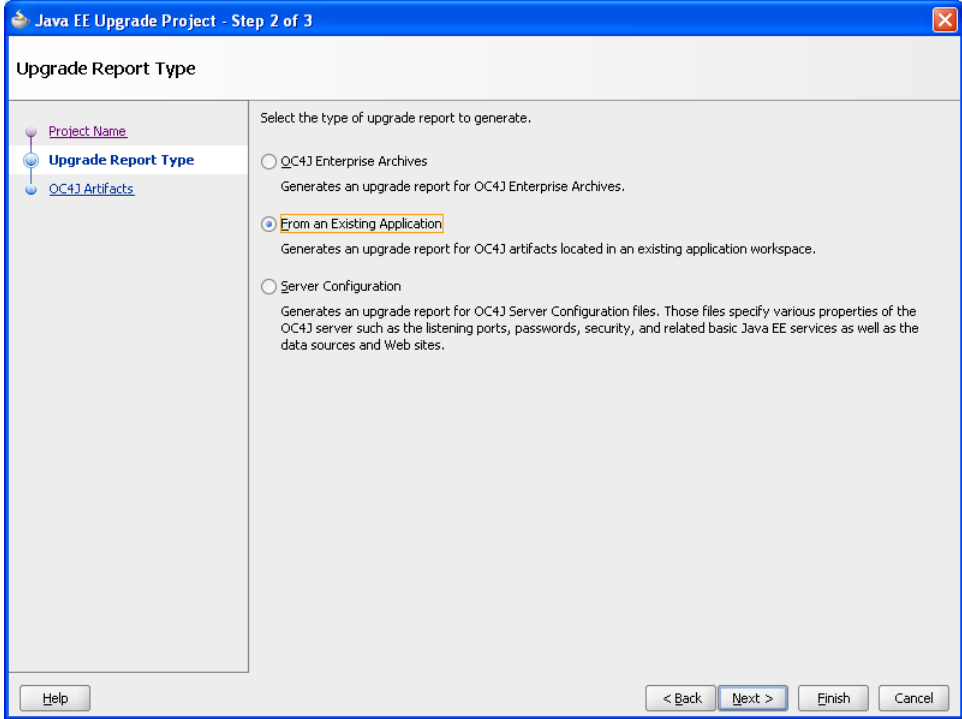
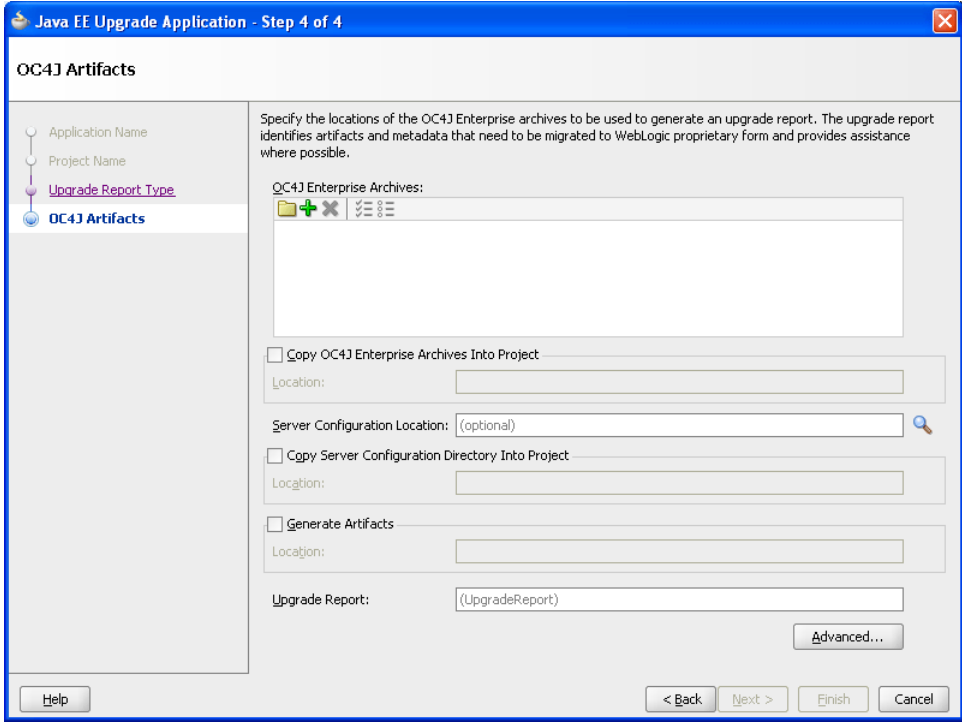
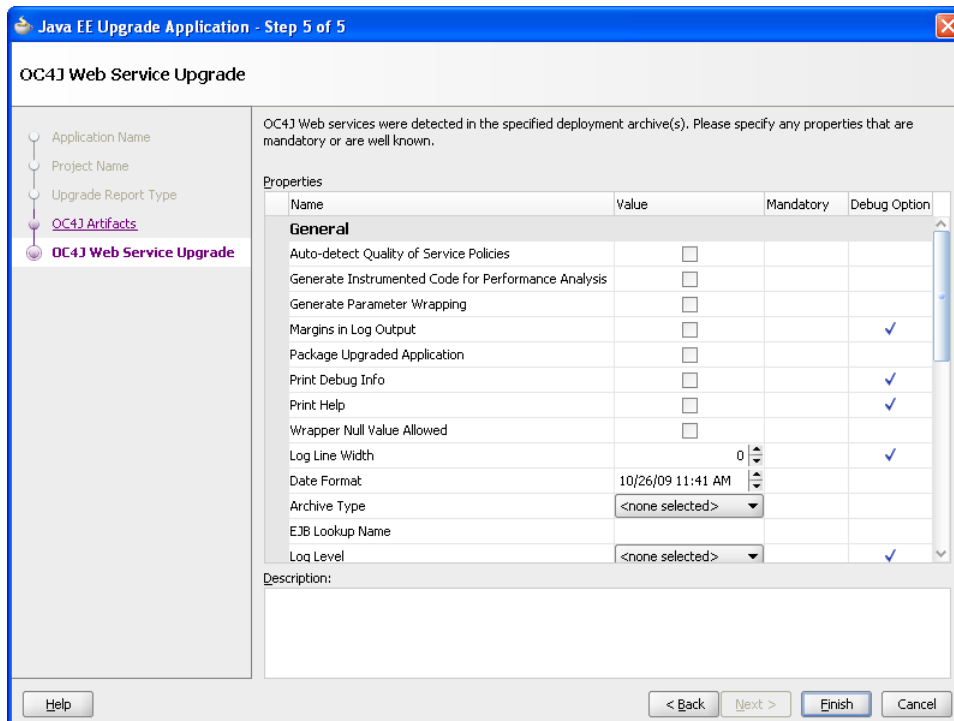


Figure 2-4 Java EE Upgrade Application Wizard - OC4J Artifacts Page When Analyzing an Enterprise Archive



**Figure 2–5 Java EE Upgrade Application Wizard - OC4J Web Services Upgrade Page**

### 2.2.2.2 Types of Archives and Projects Supported by the SmartUpgrade Oracle JDeveloper Extension

When running SmartUpgrade from Oracle JDeveloper, you can analyze the following:

- An existing enterprise archive (EAR) file
- One or more Oracle JDeveloper projects within the current Oracle JDeveloper application

If you want to analyze other types of archives, such as WAR, RAR, or JAR files, you have two options:

- Package the WAR, RAR, or JAR file in an EAR file and select the EAR file for processing.
- Use the SmartUpgrade command-line interface.

If you want to analyze Web services within an application, you must select an EAR file. You cannot analyze or generate artifacts for Web services within an open Oracle JDeveloper application or project.

### 2.2.2.3 Limiting the Analysis to Specific SmartUpgrade Rule Categories

By default, SmartUpgrade generates a report and optionally generates artifacts by applying all rule categories to the selected archive or OC4J server configuration.

However, if you want to limit the size of the report, or if you want to focus on a particular aspect of your application, you can limit the analysis to a specific set of SmartUpgrade rule categories.

For example, to analyze only the data-source configuration of the `myApp.ear` application:

1. Start the Java EE Upgrade wizard, as described in [Section 2.2.1, "Starting the Java EE Upgrade Wizard"](#).
2. On the OC4J Artifacts page in the wizard, click **Advanced** to display the Rules Management dialog box.
3. Use the check boxes to select the rules categories you want SmartUpgrade to use during the current application analysis.

For a description of the SmartUpgrade rule categories you can apply when generating your reports and, optionally, your application artifacts, refer to [Table 3-5](#).

#### 2.2.2.4 Analyzing Applications Deployed on Oracle Application Server 10g Release 2 (10.1.2)

By default, SmartUpgrade is configured to analyze applications that were previously deployed on Oracle Application Server 10g Release 3 (10.1.3). However, if you are upgrading from 10g Release 2 (10.1.2), you can indicate the specific version of OC4J you are using:

1. Start the Java EE Upgrade wizard, as described in [Section 2.2.1, "Starting the Java EE Upgrade Wizard"](#).
2. On the OC4J Artifacts page in the wizard, click **Advanced** to display the Rules Management dialog box.
3. Select **10.1.3** or **10.1.2** from the OC4J Server Version section of the dialog box.

### 2.2.3 Setting Properties When Upgrading OC4J Web Services with SmartUpgrade

If the application or project you are upgrading contains any Web services, then the Java EE Upgrade wizard displays the OC4J Web Services Upgrade wizard page ([Figure 2-5](#)).

For information about setting some the options on this page, refer to the following sections:

- [Learning About Each of the Web Services Artifact Generation Options](#)
- [Setting the Mandatory Target Server Home Directory Option](#)
- [Generating Instrumented Code for Performance Analysis](#)
- [Packaging the Upgraded Web Services Artifacts as a Deployable Enterprise Archive](#)

#### 2.2.3.1 Learning About Each of the Web Services Artifact Generation Options

You can learn more about the options on the page by referring to the following resources:

- Select a option on the page and review the text in the **Description** field.
- [Table 3-2, "Summary of Optional SmartUpgrade Command-Line Options"](#)
- [Table 3-3, "Summary of the General Command-Line Options Specific to Generating Web Services Artifacts"](#)
- [Table 3-4, "Summary of the Advanced Command-Line Options Specific to Generating Web Services Artifacts"](#)

### 2.2.3.2 Setting the Mandatory Target Server Home Directory Option

The only mandatory option on the OC4J Web Services Upgrade wizard page is the **Target Server Home Directory** option. Select this option to specify the target WebLogic Server home directory.

For example:

```
C:\Oracle\Middleware\wlserver_10.3
```

If you are migrating a large number of applications, you can avoid repeatedly specifying this property by setting the WL\_HOME environment variable before you start Oracle JDeveloper. The value you set in WL\_HOME environment variable will be applied to all applications during the upgrade process.

---

**Important Note:** The Target Server Home Directory must be for a WebLogic Server home that is installed separately from Oracle JDeveloper. Do not select the WebLogic Server home that is installed with the Oracle JDeveloper installer.

---

### 2.2.3.3 Generating Instrumented Code for Performance Analysis

Select the **Generate Instrumented Code for Performance Analysis** property on the OC4J Web Services Upgrade wizard page if you want SmartUpgrade to collect performance data and generate an HTML report that displays the performance impact of the additional code generated for the upgraded application.

SmartUpgrade generates an HTML report that summarizes the performance impact of the glue code. If there are multiple Web services, multiple report files will be generated.

For more information, see [Section 4.4, "Analyzing the Performance Impact of the Web Services Glue Code Generated by SmartUpgrade"](#).

### 2.2.3.4 Packaging the Upgraded Web Services Artifacts as a Deployable Enterprise Archive

Select the **Package Upgraded Application** option if you want SmartUpgrade to package the upgraded application into an archive you can then deploy on Oracle WebLogic Server.

The archive will be written to a directory called `Final`, which SmartUpgrade creates inside the output directory you defined as the location of generated artifacts.

You can use this option for applications that are dedicated Web services applications that do not contain other features that need to be upgraded to run on Oracle WebLogic Server.

For more information, see [Section 4.3, "Using Web Services Artifacts Generated by SmartUpgrade"](#).

## 2.2.4 Continuing with the Upgrade of Web Services Artifacts

In some cases, when SmartUpgrade is analyzing the Web services artifacts of your application, it might find that some manual steps are required before the Web services can be upgraded. In other cases, it might find that some properties that you did not provide are required.

In those cases, a warning appears in the SmartUpgrade log window at the bottom of the Oracle JDeveloper window. [Example 2-1](#) is a typical example of a warning that



alerts you that the Web services analysis and artifact generation could not be completed and that you need to continue the Web services upgrade process.

In other cases, the warning or error message will describe additional steps you must take before continuing with the Web services upgrade.

**Example 2–1 Web Services Upgrade Warning in the SmartUpgrade Log Window**

```
15:36 WARNING - -----Attention Required - Start-----
15:36 WARNING - Action Recommended: Please rerun the command, passing all the
                  arguments used in this invocation, and ensure the following
                  additional arguments are also passed:
15:36 WARNING - 1: -acceptDuplicates
15:36 WARNING - 2: -skipSourcePlan
15:36 WARNING - -----Attention Required - End-----
```

To run SmartUpgrade again and continue with the Web services upgrade process:

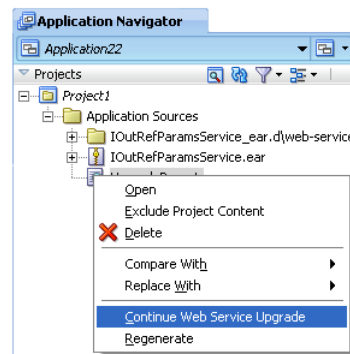
1. Right-click the upgrade report in the Oracle JDeveloper Application Navigator and select **Continue Web Service Upgrade** from the context menu (Figure 2–6).

Oracle JDeveloper displays the OC4J Web Services Properties dialog, which is similar to the OC4J Web Services Upgrade wizard page, except that it contains only the properties you originally supplied, plus the additional properties that are required to continue the upgrade process.

2. Provide the missing property values and click **OK**.

SmartUpgrade analyzes the Web services in the application again and attempts to complete the artifact generation. If additional properties are required, you might be prompted to continue the upgrade process again.

**Figure 2–6 Selecting Continue Web Service Upgrade from the Upgrade Report Context Menu**



## 2.3 Using a SmartUpgrade Report

After you generate a SmartUpgrade upgrade report in Oracle JDeveloper, you can sort and filter the findings. You can use this feature to focus on the most important findings first.

Refer to the following sections for more information:

- [Viewing a SmartUpgrade Report](#)
- [Using the Upgrade Findings Toolbar to Filter SmartUpgrade Findings](#)
- [Using the Structure Window to Sort and Filter SmartUpgrade Findings](#)

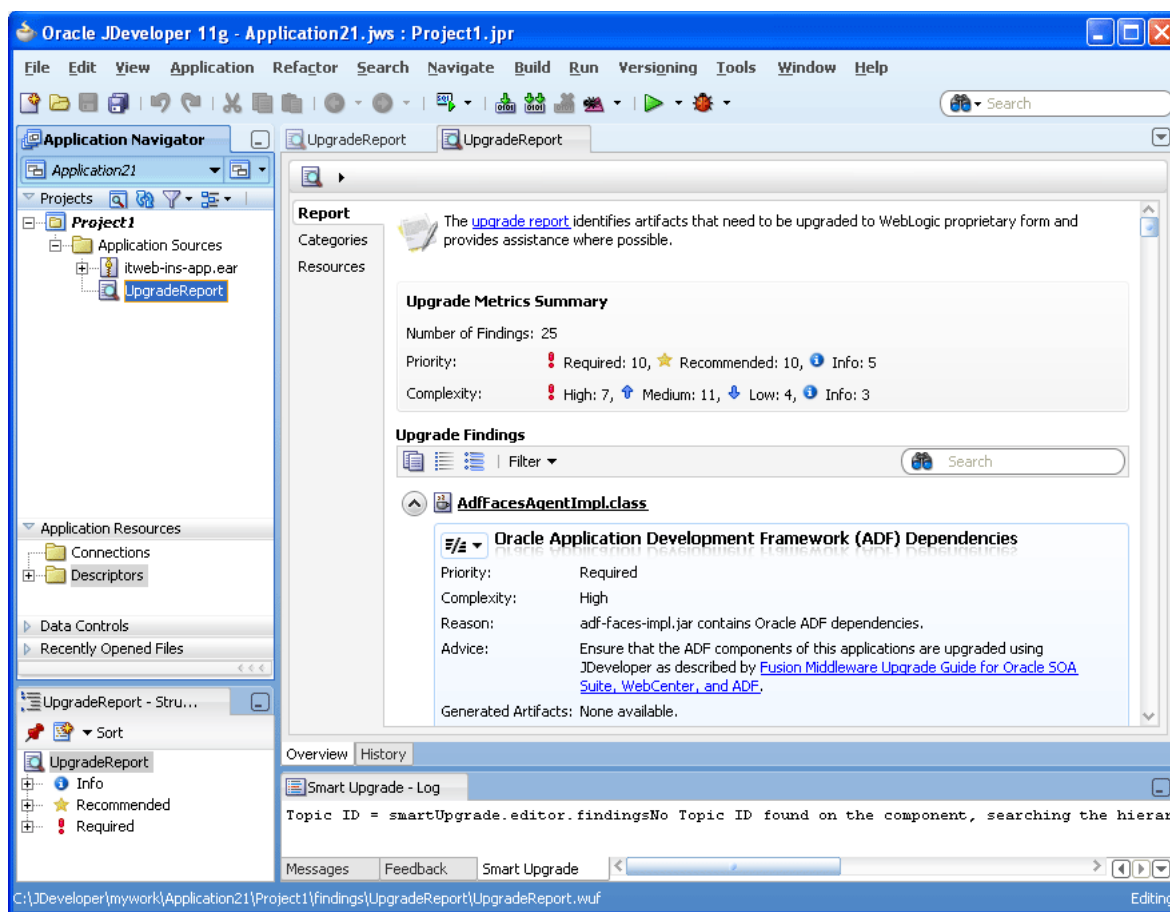
## 2.3.1 Viewing a SmartUpgrade Report

To view the upgrade report, use the Application Navigator to open the project you created with the Java EE Upgrade wizard. Locate the upgrade report under the Application Sources folder in the project and select the upgrade report (Figure 2–7).

Note that the report is divided into three tabs:

- The Report tab, which contains a summary of the report and a list containing each finding generated by SmartUpgrade for the selected application
- The Categories tab, which lists the categories of SmartUpgrade rules that were applied when generating the report.
- The Resources tab, which lists the archives or projects that were analyzed by SmartUpgrade to generate the report.

Figure 2–7 SmartUpgrade Report in Oracle JDeveloper



## 2.3.2 Using the Upgrade Findings Toolbar to Filter SmartUpgrade Findings

If your report contains a long list of findings, you can easily sort or filter the list using the Upgrading Findings toolbar (Figure 2–8).

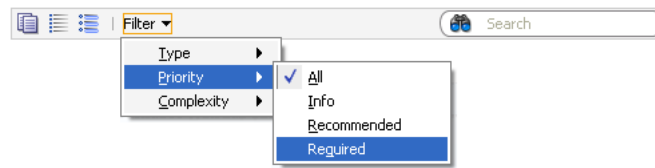



**Figure 2–8 Using the Upgrade Findings Toolbar**

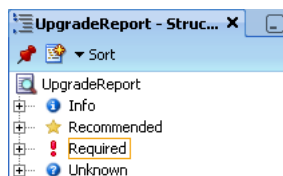
Table 2–2 describes each of the icons and menus available in the Upgrade Findings toolbar.

**Table 2–2 Summary of the Upgrade Findings Toolbar Options**

| Icon or Menu  | Description  |
|---|--|
|  | Click this icon to copy the contents of the report to the clipboard in regular text format. You can then paste it into a separate document or program.   |
|  | Click this icon to collapse all the findings so you can scroll down and see the main headings for each finding, but not the details.   |
|  | Click this icon to expand all the findings so the detail is displayed for every finding in the report.   |
| Type  | Use this submenu to filter the findings by finding type.<br>The finding types are organized by the types of archive associated with each finding. For example, you can filter out all but the findings related to WAR files. |
| Priority  | Use this submenu to filter the findings by priority. For example, you can filter out all but the required findings.  |
| Complexity  | Use this submenu to filter the findings by complexity. For example, you can filter out all but the findings with the highest complexity.   |

### 2.3.3 Using the Structure Window to Sort and Filter SmartUpgrade Findings

The structure window (Figure 2–9) under the Oracle JDeveloper navigation pane provides a view of the report structure. You can use this structure window to filter the findings in your report.

**Figure 2–9 SmartUpgrade Report Structure Window**

In this default view, you can quickly view all findings of a particular priority. For example, click **Required** to show only the findings that are marked as "Required."

You can also use the **Sort** menu in the Structure view to change the list. For example, select **Complexity** from the **Sort** menu to show a list of the complexity categories. You can then select one of the categories to view only the findings of that category in the Report window.

At any time, you can click the root node in the structure view (UpgradeReport in Figure 2–9) to view all the findings in the report.



---

---

## Using the SmartUpgrade Command Line

In addition to using SmartUpgrade as an integrated Oracle JDeveloper extension, you can also use the SmartUpgrade command-line interface.

The command line interface can be run using the Java command line or as an Apache Ant task.

With the SmartUpgrade command-line, you can consider automating the analysis and generation of artifacts for multiple applications using scripts.

For more information, see the following sections:

- [Using the SmartUpgrade Java Command-Line](#)
- [Integrating SmartUpgrade with Apache Ant](#)

### 3.1 Using the SmartUpgrade Java Command-Line

The following sections describe how to use the command-line interface for SmartUpgrade:

- [Verifying Prerequisites and Locating the smartupgrade.jar File](#)
- [Basic Syntax of the SmartUpgrade Command-Line Interface](#)
- [Getting Help on the SmartUpgrade Command-Line Options](#)
- [Summary of the SmartUpgrade Command-Line Options](#)
- [Summary of the SmartUpgrade Command-Line Options Specific to Web Services Artifact Generation](#)
- [Examples of Using the SmartUpgrade Command-Line Interface](#)

#### 3.1.1 Verifying Prerequisites and Locating the smartupgrade.jar File

Before you can run the SmartUpgrade command-line interface:

1. Verify that you have installed the required Java software, as described in [Section 1.6.4, "Installing the SmartUpgrade Command-Line Interface"](#).

Note that you should also make sure that the Java bin directory is defined as part of the current PATH variable, so you can run the `java` command from any location on your system. Otherwise, you must include the path to the `java` command every time you run the SmartUpgrade software.

2. Locate the `smartupgrade.jar` file, which you installed using the instructions in [Section 1.6, "Downloading and Installing SmartUpgrade"](#).

Note that the instructions and examples in this chapter assume you are running SmartUpgrade from the directory where the `smartupgrade.jar` resides.

### 3.1.2 Basic Syntax of the SmartUpgrade Command-Line Interface

To use the SmartUpgrade command-line interface, simply navigate to the directory where you unpacked the contents of the downloaded `smartupgrade.zip` file and use the following syntax:

```
java -jar smartupgrade.jar options
```

For more information, see [Section 3.1.4, "Summary of the SmartUpgrade Command-Line Options"](#)

### 3.1.3 Getting Help on the SmartUpgrade Command-Line Options

To display a list of the available options, enter one of the following commands:

```
java -jar smartupgrade.jar -help
java -jar smartupgrade.jar -help locator
java -jar smartupgrade.jar -help category
java -jar smartupgrade.jar -help option
```

For detailed information, see [Section 3.1.4, "Summary of the SmartUpgrade Command-Line Options"](#).

### 3.1.4 Summary of the SmartUpgrade Command-Line Options

Refer to the following sections for detailed information about the options you can use when running SmartUpgrade from the command line:

- [List of SmartUpgrade Command-Line Interface Options](#)
- [Identifying a SmartUpgrade Locator](#)
- [Specifying More Than One Locator on the SmartUpgrade Command Line](#)
- [Summary of the SmartUpgrade Optional Command-Line Options](#)

#### 3.1.4.1 List of SmartUpgrade Command-Line Interface Options

The following example shows the options you can use when running the SmartUpgrade command-line utility:

```
java -jar smartupgrade.jar --LOCATOR_NAME path_or_list_file_names
                        -category
                        -generate
                        -html
                        -target
```

All but the `LOCATOR_NAME` value are optional.

In the previous example, replace `LOCATOR_NAME` with a valid Locator that SmartUpgrade upgrade can analyze. For more information, see [Section 3.1.4.2, "Identifying a SmartUpgrade Locator"](#).

#### 3.1.4.2 Identifying a SmartUpgrade Locator

A locator is a general term to identify the object or objects that you want SmartUpgrade to analyze. The locator can be one or more application archives (EAR,

WAR, JAR, or RAR files). It can also be a directory path where archives are stored, or the configuration directory of an OC4J server.

[Table 3–1](#) describes the values you can use for the LOCATOR\_NAME command-line option.

**Table 3–1 Supported Values for the SmartUpgrade LOCATOR\_NAME Option**

| <b>LOCATOR_NAME value</b> | <b>Description</b>  | <b>Examples</b>  |
|---------------------------|---|--|
| --ears                    | Identifies one or more enterprise archive (EAR) files to analyze.<br><br>If you are providing the path to more than one EAR file, then use a space-delimited list after the -ear option.                            | java -jar smartupgrade.jar<br>--ears myApp.ear<br>java -jar smartupgrade.jar<br>--ears C:\samples\App3.ear<br>java -jar smartupgrade.jar<br>--ears myApp.ear App3.ear                      |
| --wars                    | Identifies one or more Web archive (WAR) files to analyze.<br><br>If you are providing the path to more than one WAR file, then use a space-delimited list after the -ear option.                                   | java -jar smartupgrade.jar<br>--wars payroll.war<br>java -jar smartupgrade.jar<br>--wars C:\samples\Webapp3.war<br>java -jar smartupgrade.jar<br>--wars payroll.war C:\samples\Webapp3.war |
| --jars                    | Identifies one or more Java archive (JAR) files to analyze.<br><br>If you are providing the path to more than one JAR file, then use a space-delimited list after the -ear option.                                  | java -jar smartupgrade.jar<br>--jars myProj.jar<br>java -jar smartupgrade.jar<br>--jars C:\samples\App3.jar<br>java -jar smartupgrade.jar<br>--jars myApp.jar C:\samples\App3.jar          |
| --rars                    | Identifies one or more RAR archive files to analyze.<br><br>If you are providing the path to more than one RAR file, then use a space-delimited list after the -ear option.   | java -jar smartupgrade.jar<br>--rars myApp.rar<br>java -jar smartupgrade.jar<br>--rars C:\samples\App3.rar<br>java -jar smartupgrade.jar<br>--rars myApp.rar C:\samples\App3.rar           |
| --server-config           | Identifies the configuration directory of an existing OC4J server.<br><br>SmartUpgrade analyzes the configuration of the OC4J server and provide advice and configuring Oracle WebLogic Server in a similar manner. | java -jar smartupgrade.jar --server-config<br>C:\Oracle\AppServ1\j2ee\home\config<br>java -jar smartupgrade.jar --server-config<br>/dual/Oracle/AppServ1/j2ee/home/config                  |
| --archive-homes           | Identifies one or more directories that contain archive files (EAR, WAR, RAR, or JAR files) that you want to analyze.<br><br>SmartUpgrade scans the directory and analyzes all the archives in the directory.       | java -jar smartupgrade.jar --archive-home<br>C:\projects\myEARfiles\<br>java -jar smartupgrade.jar --archive-home<br>/dual/projects/myEARfiles/  |

**Table 3–1 (Cont.) Supported Values for the SmartUpgrade LOCATOR\_NAME Option**

| <b>LOCATOR_NAME value</b> | <b>Description</b>   | <b>Examples</b>   |
|---------------------------|--|---|
| --application-jars        | Identifies the location of jar files that required or referenced by an application you are analyzing. The specified file is not analyzed by SmartUpgrade.<br><br>You can use this feature to identify third-party libraries required by the application you are analyzing. | java -jar smartupgrade.jar<br>--application-jars C:\projects\myApp\lib\<br>java -jar smartupgrade.jar<br>--application-jars /dual/projects/myApp/lib/ |

### 3.1.4.3 Specifying More Than One Locator on the SmartUpgrade Command Line

You can specify more than one LOCATOR\_NAME on the command line.

For example, to analyze an enterprise archive and the configuration of the OC4J server on Linux where the archive was deployed, use the following command:

```
java -jar smartupgrade.jar --ears myApp.ear
--server-config /dual/Oracle/AppServ1/j2ee/home/config
```

### 3.1.4.4 Summary of the SmartUpgrade Optional Command-Line Options

In addition to identifying a SmartUpgrade locator, you can also control the behavior of SmartUpgrade by using various optional command-line options.

Table 3–2 describes the optional command-line options you can use.

**Table 3–2 Summary of Optional SmartUpgrade Command-Line Options**

| <b>Options</b> | <b>Description</b>  | <b>For More Information</b>   |
|----------------|---|---|
| -category      | Use this option to limit the analysis of the application to specific categories of SmartUpgrade rules.<br><br>For example, you can limit the report to findings about data-source configurations in the selected application archive.   | <a href="#">Section 3.1.6.4, "Limiting the Findings to Specific Rule Categories from the SmartUpgrade Command-Line Interface"</a> |
| -generate      | Use this option to generate specific types of Oracle WebLogic Server artifacts, such as Oracle WebLogic Server deployment descriptors for the application.<br><br>The artifacts can be used as a starting point for the deploying your OC4J applications on Oracle WebLogic Server. | <a href="#">Section 3.1.6.2, "Using the SmartUpgrade Command-Line Interface to Generate Oracle WebLogic Server Artifacts"</a>     |
| -html          | Use this option to generate output formatted in HTML. You can use this option to redirect the resulting report to an HTML file.   | <a href="#">Section 3.1.6.3, "Using the SmartUpgrade Command-Line Interface to Generate an HTML Report"</a>                       |
| -target        | Use this option to specify that you are analyzing an Oracle Application Server 10g Release 2 (10.1.2) application.<br><br>By default, SmartUpgrade assumes you are analyzing an application that was previously deployed on 10g Release 3 (10.1.3)                                  | <a href="#">Section 3.1.6.5, "Using the SmartUpgrade Command-Line Interface to Analyze 10g Release 2 (10.1.2) Applications"</a>   |



### 3.1.5 Summary of the SmartUpgrade Command-Line Options Specific to Web Services Artifact Generation

[Example 3–1](#) shows the options you can use when running the SmartUpgrade command-line utility to generate Web services artifacts.

For complete information about using the artifacts generated by SmartUpgrade, refer to [Chapter 4, "Using SmartUpgrade Generated Artifacts"](#).

For a description of each option, refer to:

- [Table 3–3, "Summary of the General Command-Line Options Specific to Generating Web Services Artifacts"](#)
- [Table 3–4, "Summary of the Advanced Command-Line Options Specific to Generating Web Services Artifacts"](#)

When generating Web services artifacts, the `LOCATOR_NAME` value, `-generate` option, and `-targetStackHome` option are mandatory.

Replace `LOCATOR_NAME` with a valid Locator that SmartUpgrade upgrade can analyze. For more information, see [Section 3.1.4.2, "Identifying a SmartUpgrade Locator"](#).

***Example 3–1 List of Command-Line Options When Generating Web Services Artifacts***

```
java -jar smartupgrade.jar --LOCATOR_NAME path_to_application_archive_or_directory
-generate
-category web-services
    -acceptDuplicates
    -additionalClassPath
    -archiveType
    -autoPack
    -autoWrap
    -dateFormat
    -debug
    -debugControl
    -debugWrap
    -debugWrapLength
    -ejbLookupName
    -ejbNewWarBase
    -ejbNewWarContextRoot
    -evaluate
    -javaHome
    -packLIBs
    -plansOnly
    -processTimeout
    -qos
    -resolveMapAmiguity
    -skipGlueCode
    -skipSourcePlan
    -skipTargetPlan
    -sourceStack
    -targetStackHome
    -wrapperNullAllowed
```

**Table 3–3 Summary of the General Command-Line Options Specific to Generating Web Services Artifacts**

| Command-Line Option  | Equivalent Option in the Java EE Upgrade Wizard | Description  |
|----------------------|---|--|
| -archiveType         | Archive Type                                    | <p>The type of archive. The value can be one of the following:</p> <ul style="list-style-type: none"> <li>▪ EJBJAR</li> <li>▪ WAR</li> <li>▪ EAR</li> </ul> <p>If you do not provide a value for this option, SmartUpgrade determines the archive type automatically, by analyzing the archive and assigning the value as follows:</p> <ul style="list-style-type: none"> <li>▪ WAR - If the archive contains a WEB-INF directory</li> <li>▪ EJBJAR - If the archive contains a META-INF directory with an ejb-jar.xml file.</li> <li>▪ EAR - If the archive contains a WAR or EJBJAR archive</li> </ul>   |
| -additionalClassPath | Additional Classpath                            | An additional classpath containing JAR files and class directories. This option is required when the input application does not contain all the libraries that a Web service depends on.   |
| -autoPack            | Package Upgraded Application                    | <p>Enable this option by passing <code>-autoPack</code> (or <code>-autoPack true</code>) on command line or by selecting the check box on the OC4J Web Services Upgrade wizard page.</p> <p>When enabled, SmartUpgrade packages the target upgraded application into a deployable archive.</p> <p>The default value is <code>true</code> when you use the command-line interface and <code>false</code> when you use Oracle JDeveloper (the check box is clear by default).</p> <p><b>Note:</b> You can disable this option if the input archive has other JEE artifacts that need to be upgraded. After the wrapper code is generated, and you have upgraded other Java EE artifacts in the application, you can pack the application in a new command by specifying <code>-skipGlueCode</code> and <code>-autoPack</code> options.</p> <p>For more information, see <a href="#">Section 4.3.6, "Using Artifacts Generated for an Application with Both Web Services and Other Java EE Elements"</a>.</p> |
| -autoWrap            | Generate Parameter Wrapping                     | <p>Enable this option by passing <code>-autoWrap</code> (or <code>-autoWrap true</code>) on command line or by selecting the check box on the OC4J Web Services Upgrade wizard page.</p> <p>When this option is enabled, SmartUpgrade enables parameter wrapping for the internal types that are generated.</p> <p>The existing business methods will not be aware of any of new interfaces and value types generated.</p> <p>When this option is not specified, its value is defined by whether the existing business parameters are wrapped.</p>   |
| -dateFormat          | Format for Date-String Conversion               | The date format to be used by glue code for conversion between <code>java.lang.String</code> and <code>java.util.Date</code> .   |

**Table 3–3 (Cont.) Summary of the General Command-Line Options Specific to Generating Web Services**

| Command-Line Option            | Equivalent Option in the Java EE Upgrade Wizard     | Description   |
|--------------------------------|---|---|
| <code>-ejbLookupName</code>    | EJB Lookup Name                                     | <p>The EJB 2.x lookup name to be used in the POJO-based Web service to which an EJB web service will be upgraded.</p> <p>This option should be used only when there is a single EJB Web service in the application. If multiple EJB Web services are present, edit the upgrade plans generated for each service.</p> <p>For information about locating the upgrade plans, see <a href="#">Section 4.3.4, "About the Content of the Generated Web Services Output Directories"</a>.</p>  |
| <code>-ejbNewWarBase</code>    | WAR Base Name                                       | <p>The base name of the WAR file to be generated.</p> <p>When an EJB-based Web service is upgraded to a POJO-based Web service, SmartUpgrade generates a new WAR file. The value of this property will be the base name of the new WAR file.</p> <p>If there are more than one EJB Web service, edit the source upgrade plan to add this property with different values for each EJB Web service.</p> <p>For information about locating the upgrade plans, see <a href="#">Section 4.3.4, "About the Content of the Generated Web Services Output Directories"</a>.</p>                                       |
| <code>-ejbNewWarContext</code> | WAR Context Root                                    | <p>The context path of the WAR file to be generated.</p> <p>When an EJB-based Web service is upgraded to POJO-based Web service, SmartUpgrade generates a new WAR file. The value of this property will be the context root of the new WAR file.</p> <p>If there is more than one EJB Web service, then modify the source upgrade plan to add this property with different values for each EJB Web service.</p> <p>For information about locating the upgrade plans, see <a href="#">Section 4.3.4, "About the Content of the Generated Web Services Output Directories"</a>.</p>                             |
| <code>-evaluate</code>         | Generate Instrumented Code for Performance Analysis | <p>Enable this option by passing <code>-evaluate</code> (or <code>-evaluate true</code>) on command line, or by selecting the check box on the OC4J Web Services Upgrade wizard page.</p> <p>When you enable this option, the generated wrapper ("glue") code is instrumented for performance analysis.</p> <p>The default value of the option is <code>false</code>.</p> <p>This option is appropriate only for testing the application, and not for production environments.</p> <p>For more information, see <a href="#">Section 2.2.3.3, "Generating Instrumented Code for Performance Analysis"</a>.</p> |
| <code>-javaHome</code>         | JDK Home for Target Server Tools                    | <p>The JDK home that SmartUpgrade uses to run the Web Service tools on the target server. The system automatically attempts to locate JDK installed with the target server.</p> <p>You can override this behavior by specifying a JDK home as a value for this option.</p> <p>If this option is not specified and if no JDK home is installed as part of the target server, the system will use the current JDK home used to run this upgrade tool.</p>   |

**Table 3–3 (Cont.) Summary of the General Command-Line Options Specific to Generating Web Services**

| Command-Line Option | Equivalent Option in the Java EE Upgrade Wizard | Description  |
|---------------------|---|--|
| -packLIBs           | Packaging Includes                              | <p>A list of JAR files, separated by a path separator.</p> <p>On Windows, the separator is a semicolon (;); on UNIX systems, the separator is colon (:).</p> <p>The identified JAR files are automatically packaged into the final generated archive, if the final archive is an EAR or WAR file.</p> <p>This value is also added into the value of <code>additionalClassPath</code> option.</p>   |
| -processTimeout     | Generate Tasks Timeout (seconds)                | <p>The number of seconds SmartUpgrade will continue trying to process WSDL documents and generate artifacts.</p> <p>The tasks timeout default is 300 seconds. A different value may be set by using this command-line option, by providing a positive number on the wizard page in Oracle JDeveloper, or by setting the following environment variable prior to starting SmartUpgrade:</p> <p><code>GENERATION_TASKS_TIMEOUT</code></p>  |
| -qos                | Quality of Service Policies                     | <p>This option indicates the types of policies (Quality Of Services) that need to be enabled in the target Web service that is generated.</p> <p>The list of possible tokens is MTOM, WSS_UNT, or STATEFUL.</p> <p>Multiple values can be passed with a comma(,) as the separator.</p>   |
| -sourceStack        | N/A   | <p>The name and version of the server that hosted the original Web services application.</p> <p>Possible values are:</p> <p>OC4J1013<br/>OC4J1012</p> <p>By default, SmartUpgrade attempts to determine the server version from the application. When there is an ambiguity, this option is mandatory.</p>   |
| -targetStackHome    | Target Server Home Directory                    | <p>Mandatory property used to specify the target WebLogic Server home. For example:</p> <p><code>C:\Oracle\Middleware\wlserver_10.3</code></p> <p>If you are migrating a large number of applications, you can avoid repeatedly specifying this property by setting the environment variable <code>WL_HOME</code> prior to starting Oracle JDeveloper.</p> <p>The value you set in <code>WL_HOME</code> environment variable will be applied to all applications during the upgrade process.</p> <p><b>Important Note:</b> The Target Server Home Directory must be for a WebLogic Server home that is installed separately from Oracle JDeveloper. Do not select the WebLogic Server home that is installed with the Oracle JDeveloper installer.</p> |

**Table 3–3 (Cont.) Summary of the General Command-Line Options Specific to Generating Web Services**

| Command-Line Option | Equivalent Option in the Java EE Upgrade Wizard | Description   |
|---------------------|---|---|
| -wrapperNullAllowed | Wrapper Null Value Allowed                      | <p>Enable this option by passing <code>-wrapperNullAllowed</code> (or <code>-wrapperNullAllowed true</code>) on the command line or by selecting the check box on the OC4J Web Services Upgrade wizard page.</p> <p>When enabled, null values are allowed for wrapper types already present in the application before upgrade.</p> <p>The default value is <code>false</code>. In this case, the null value of a wrapper type will be represented by an empty object.</p> <p>Note that WebLogic Server JAX-RPC Web Services do not allow null values for wrapper types.</p> |

**Table 3–4 Summary of the Advanced Command-Line Options Specific to Generating Web Services Artifacts**

| Option            | Equivalent Option in the Java EE Upgrade Wizard | Description   |
|-------------------|---|---|
| -acceptDuplicates | Accept Duplicate Classes                        | <p>Enable this option by passing <code>-acceptDuplicates</code> (or <code>-acceptDuplicates true</code>) on the command line or by selecting the check box on the OC4J Web Services Upgrade wizard page.</p> <p>This option indicates whether the system should ignore the duplicate class error and continue the glue code generation. You should enable this option only after ensuring that the duplicate classes are identical.</p> |
| -debug            | Print Debug Info                                | <p>Enable this option by passing <code>-debug</code> (or <code>-debug true</code>) on the command line or by selecting the check box on the OC4J Web Services Upgrade wizard page.</p> <p>When enabled, indicates that debug level messages should be printed. The debug messages are more detailed than INFO level messages.</p> <p>The default value is <code>false</code>.</p>   |
| -debugControl     | Logging Level                                   | <p>Use this option to set the level of logging output.</p> <p>The value is one of the following:<br/>FINE   DEBUG   INFO   WARNING   ERROR   TRACE</p> <p>The default value is INFO.</p> <p><b>Note:</b> This option is not validated; any invalid value will be ignored.</p>   |
| -debugWrap        | Margins in Log Output                           | <p>Enable this option by passing <code>-debugWrap</code> (or <code>-debugWrap true</code>) on the command line or by selecting the check box on the OC4J Web Services Upgrade wizard page.</p> <p>When enabled, this option causes SmartUpgrade to align the diagnostic output for easy reading.</p> <p>The default value is <code>true</code>. You can adjust column width using the <code>debugWrapLength</code> option.</p>          |

**Table 3–4 (Cont.) Summary of the Advanced Command-Line Options Specific to Generating Web Services Artifacts**

| Option               | Equivalent Option in the Java EE Upgrade Wizard | Description  |
|----------------------|---|--|
| -debugWrapLength     | Log Line Width                                  | <p>Use this option to set the maximum length of the text that is logged without breaking the text into multiple lines. Use this option to preserve alignment.</p> <p>This is applicable only when <code>debugWrap</code> option is set to <code>true</code>. The default value is 80.</p>  |
| -plansOnly           | Generate Upgrade Plans Only                     | <p>Enable this option by setting the value to <code>true</code> or selecting the check box in the OC4J Web Services Upgrade wizard page.</p> <p>When this option is enabled, SmartUpgrade does not generate Web services glue code.</p> <p>The default value is <code>false</code> (glue code generation should be done).</p> <p>When this option is enabled, SmartUpgrade exits immediately after the target upgrade plan is generated.</p> <p>You can use this option when you want to edit the target upgrade plan and then generate glue code in a separate step. This option takes priority over options such as <code>autoPack</code>, which are functional only after the glue code is generated. Such options will be ignored when this option is enabled.</p> |
| -resolveMapAmbiguity | Resolve Mapping Ambiguity                       | <p>Enable this option by setting the value to <code>true</code> or selecting the check box in the OC4J Web Services Upgrade wizard page.</p> <p>Enable this option to allow the system to resolve mapping ambiguities automatically.</p> <p>This option is applicable only when SmartUpgrade does not find mapping information either in mapping file or from annotations.</p> <p>The default value is <code>false</code>.</p> <p>If you want to rely on SmartUpgrade to resolve ambiguity, then enable this option.</p>   |
| -skipGlueCode        | Skip Glue Code Generation                       | <p>Enable this option by setting the value to <code>true</code> or selecting the check box in the OC4J Web Services Upgrade wizard page.</p> <p>When enabled, this option indicates that glue code generation needs to be skipped. When enabled, the target plan generation is also skipped. Use this option when the you have modified the generated glue code and you need to compile and package the application.</p>   |
| -skipSourcePlan      | Skip Source Upgrade Plan Generation             | <p>Enable this option by setting the value to <code>true</code> or selecting the check box in the OC4J Web Services Upgrade wizard page.</p> <p>When enabled, SmartUpgrade does not generate a source migration plan generation and uses an existing plan.</p>   |

**Table 3–4 (Cont.) Summary of the Advanced Command-Line Options Specific to Generating Web Services Artifacts**

| Option          | Equivalent Option in the Java EE Upgrade Wizard | Description  |
|-----------------|---|--|
| -skipTargetPlan | Skip Target Upgrade Plan Generation             | <p>Enable this option by setting the value to <code>true</code> or selecting the check box in the OC4J Web Services Upgrade wizard page.</p> <p>When enabled, SmartUpgrade does not generate a target upgrade plan generation and instead uses an existing plan.</p> <p>When this option is enabled, the source upgrade plan generation is also skipped.</p> |

### 3.1.6 Examples of Using the SmartUpgrade Command-Line Interface

The following sections provide some examples of how you can use the SmartUpgrade command-line options to help you upgrade your applications to Oracle WebLogic Server:

- [Using the SmartUpgrade Command-Line Interface to analyze an Enterprise Archive \(EAR\) File](#)
- [Using the SmartUpgrade Command-Line Interface to Generate Oracle WebLogic Server Artifacts](#)
- [Using the SmartUpgrade Command-Line Interface to Generate an HTML Report](#)
- [Limiting the Findings to Specific Rule Categories from the SmartUpgrade Command-Line Interface](#)
- [Using the SmartUpgrade Command-Line Interface to Analyze 10g Release 2 \(10.1.2\) Applications](#)

#### 3.1.6.1 Using the SmartUpgrade Command-Line Interface to analyze an Enterprise Archive (EAR) File

To generate a SmartUpgrade report for a specific enterprise archive (EAR) file, use the following command syntax:

```
java -jar smartupgrade.jar --ears path_of_ear_file.ear
```

For example, if you have an EAR file called `myApp.ear` and it resides in a directory called `my C:\MyApps`, then use the following command:

```
java -jar smartupgrade.jar --ears C:\MyApps\myApp.ear
```

SmartUpgrade generates a report that is written to the terminal window. To save the report findings, redirect the output to a file. For example, the following examples show how to redirect the output to a file called `report.txt`.

- On the Windows operating system:
 

```
java -jar smartupgrade.jar --ears C:\MyApps\myApp.ear > account_mgmt_report.txt
```
- On the Linux operating system:
 

```
java -jar smartupgrade.jar --ears C:\MyApps\myApp.ear > account_mgmt_report.txt
```

### 3.1.6.2 Using the SmartUpgrade Command-Line Interface to Generate Oracle WebLogic Server Artifacts

In addition to generating a report, SmartUpgrade can also generate a limited number of artifacts that can make it easier to update your applications so they can be deployed successfully on Oracle WebLogic Server.

At a minimum, you must:

- Use the `-generate` option to generate artifacts
- Specify the mandatory `-targetStackHome` option that identifies the target server home directory

For example, to generate artifacts in addition to a SmartUpgrade report for an archive called `myApp.ear`, use the following command:

```
java -jar smartupgrade.jar
      --ears archive_name
      -generate
      -category web-services
      -targetStackHome c:\middleware\wlserver_10.3
```

For another example, suppose you want to do the following:

- Specify the Oracle WebLogic Server home
- Specify the classpath to additional libraries not contained in the application archive but required for the loading of classes
- Indicate that you want SmartUpgrade to generate code instrumented for performance analysis

In that case, you would enter the following command:

```
java -jar smartupgrade.jar
      --ears archive_name
      -generate
      -category web-services
      -targetStackHome c:\middleware\wlserver_10.3
      -additionalClassPath path_to_libraries
      -evaluate
```

For information on the results of this analysis, see [Section 4.1, "Locating the Artifacts Generated by SmartUpgrade"](#).

### 3.1.6.3 Using the SmartUpgrade Command-Line Interface to Generate an HTML Report

By default, the SmartUpgrade command-line interface generates a report in text format and the report is written to the standard output of the current machine. In most cases, this means the output is written to the terminal window.

You can optionally generate a report in HTML format, which includes headings and lists that can make the output easier to read. Further, you can use your operating system commands to redirect the output to a file. The resulting HTML file can be read by any Web browser or other tool that can read HTML.

For example, if you have an EAR file called `account_mgmt.ear` and it resides in a directory called `my C:\MyApps`, then use the following command to generate an HTML file called `account_mgmt_report.html`:

```
java -jar smartupgrade.jar --ears myApp.ear -html > account_mgmt_report.html
```



### 3.1.6.4 Limiting the Findings to Specific Rule Categories from the SmartUpgrade Command-Line Interface

By default, SmartUpgrade generates a report and optionally generates artifacts by applying all rule categories to the selected archive or OC4J server configuration.

However, if you want to limit the size of the report, or if you want to focus on a particular aspect of your application, you can limit the analysis to a specific set of SmartUpgrade rule categories.

For example, to analyze only the data-source configuration of the `myApp.ear` application:

```
java -jar smartupgrade.jar --ears myApp.ear -category data-sources
```

To apply multiple categories, separate the list of categories with a space. For example:

```
java -jar smartupgrade.jar --ears myApp.ear -category data-sources web-app
```

[Table 3–5](#) describes the SmartUpgrade rule categories you can apply when generating your reports and, optionally, your application artifacts.

**Table 3–5 List of SmartUpgrade Rule Categories**

| Rule Category | Use this category to analyze...  |
|---------------|--|
| adf           | Artifacts specific to the Oracle Application Development Framework (Oracle ADF).   |
| api           | Standard application programming interfaces (APIs) used in the application. This category of rules checks for OC4J and third-party APIs that may not be supported in Oracle WebLogic Server. |
| app-client    | Client interfaces within the application.  |
| classloading  | Classloading configurations and shared libraries used by the application.  |
| cluster       | Cluster-specific configuration settings in the application.  |
| data-sources  | OC4J-specific data source configuration settings.  |
| ejb           | Enterprise Java Beans being used by the application.   |
| jca           | J2EE Connector Architecture (JCA) configuration settings and artifacts.  |
| jms           | Java Messaging Server configuration settings and artifacts, including the use of Oracle Enterprise Messaging Service (OEMS).   |
| jmx           | Java Management Extensions (JMX) used by the application.  |
| jndi          | Java Naming and Directory Interface (JNDI) configuration settings and artifacts.   |
| jta           | Java Transaction API (JTA) configuration settings and artifacts.   |
| rmi           | Remote Method Invocation (RMI) configuration settings and artifacts.   |
| security      | Security configuration settings and artifacts.   |
| soa           | Service-Oriented Architecture (SOA) configuration settings and artifacts.  |
| web-app       | Web application configuration settings and artifacts.  |
| web-services  | Web services configuration settings and artifacts.   |
| webcache      | Oracle Web Cache configuration settings.   |

### 3.1.6.5 Using the SmartUpgrade Command-Line Interface to Analyze 10g Release 2 (10.1.2) Applications

By default, SmartUpgrade assumes the applications you are upgrading were previously deployed on Oracle Application Server 10g Release 3 (10.1.3).

However, you can use the `-target` option to specify that SmartUpgrade analyze your application for any features, configuration settings, or artifacts that are specific to 10g Release 2 (10.1.2).

If you are upgrading an application that was previously deployed on OC4J as part of an Oracle Application Server 10g Release 2 (10.1.2) installation, use the `-target` option as follows:

```
java -jar smartupgrade.jar --LOCATOR_NAME -target 10.1.2
```

For example:

```
java -jar smartupgrade.jar --ears C:\myApps\my1012App.ear -target 10.1.2
```

## 3.2 Integrating SmartUpgrade with Apache Ant

If you use Apache Ant in your development environment, you can use the custom Ant task shown in [Example 3–2](#) to integrate SmartUpgrade with your existing Ant environment:

### **Example 3–2 Custom Ant Task for SmartUpgrade**

```
<taskdef
  name="SmartUpgrade"
  classname="oracle.smartupgrade.UpgradeTask"
  classpath="${basedir}/smartupgrade.jar"/>
```

After you define the `taskdef`, as shown in [Example 3–2](#), then you can execute SmartUpgrade from within an Ant script.

[Example 3–3](#) shows a typical example, which recursively locates all EAR files in the demo directory and executes SmartUpgrade to examine each file. The valid values used for the `upgrade locator` element in [Example 3–3](#) are identical to those used for the `LOCATOR_NAME` on the Java command line.

For more information, see [Section 3.1.4.2, "Identifying a SmartUpgrade Locator"](#).

### **Example 3–3 Using the SmartUpgrade Custom Ant Task Within Apache Ant**

```
<target name="test" depends="declare">
  <SmartUpgrade flags="-quiet"
    <upgrade locator="ears">
      <fileset dir="${basedir}/demo">
        <include name="**/*.ear" />
      </fileset>
    </upgrade>
  </SmartUpgrade>
</target>
```

---

## Using SmartUpgrade Generated Artifacts

In addition to generating an upgrade report, SmartUpgrade can generate artifacts, such as Web application deployment descriptor files and the mapping files and type classes required to run Web Services on a WebLogic server.

For the generated deployment descriptors, you can review the generated Oracle WebLogic Server elements and use them as a starting point for upgrading your application so you can deploy it successfully on Oracle WebLogic Server.

In the case of Web services, you can configure SmartUpgrade to create an archive of the generated Web services artifacts and the archive can be deployed directly to Oracle WebLogic Server.

The following sections describe how to use artifacts generated by SmartUpgrade to upgrade your applications so they can be deployed on Oracle WebLogic Server:

- [Locating the Artifacts Generated by SmartUpgrade](#)
- [Using Web Application Deployment Descriptor Artifacts Generated by SmartUpgrade](#)
- [Using Web Services Artifacts Generated by SmartUpgrade](#)

### 4.1 Locating the Artifacts Generated by SmartUpgrade

When SmartUpgrade generates artifacts, it saves them in a separate, predefined location. None of the source files of your original application are modified.

For more information, see the following:

- [Locating the Generated Artifacts and Output Folders](#)
- [Locating the Generated Artifacts in Oracle JDeveloper](#)

#### 4.1.1 Locating the Generated Artifacts and Output Folders

By default, SmartUpgrade saves the generated artifacts to a series of directories relative to the project directory if you're using Oracle JDeveloper or relative to the directory where you are running the command-line interface:

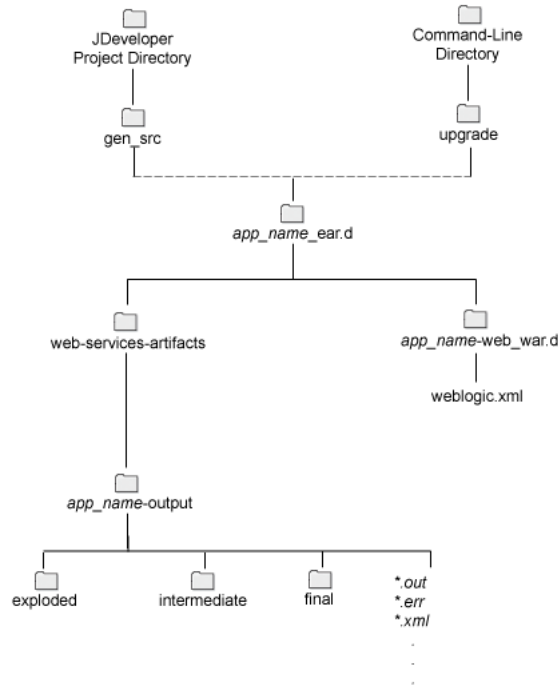
- If you are using Oracle JDeveloper, then the output directories are by default under the `/gen_src/` directory inside your new project directory. However, you can define a different location from the OC4J Artifacts page of the Java EE Upgrade wizard. The location is relative to the project directory where SmartUpgrade saves the upgrade report.

- If you are using the SmartUpgrade command-line interface, the output directories are created in the upgrade directory inside the directory where you invoked the command-line tool.

Figure 4–1 illustrates the organization of the output files on disk.

For information about the Web services artifact directories, including the **exploded**, **intermediate**, and **final** directories, and the location of the source and target upgrade plans, see Section 4.3.4, "About the Content of the Generated Web Services Output Directories" and Table 4–2.

**Figure 4–1 Output Directories for Generated Artifacts**

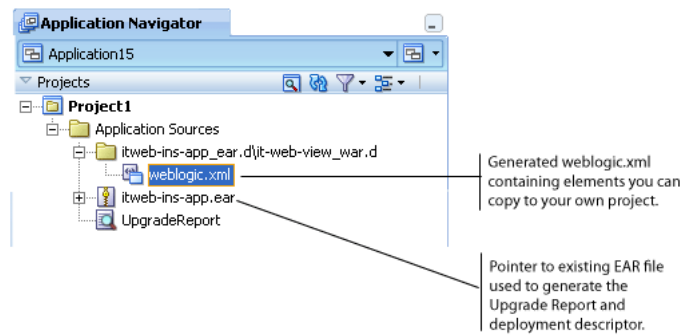
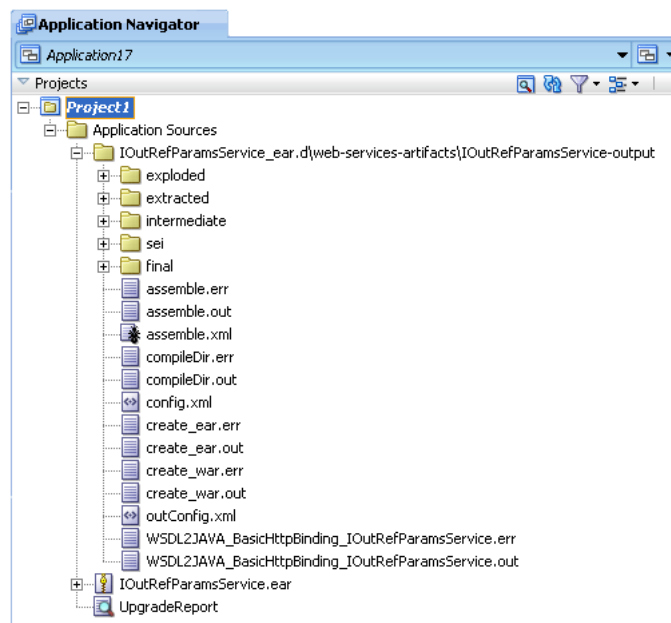


### 4.1.2 Locating the Generated Artifacts in Oracle JDeveloper

If you are using Oracle JDeveloper, you can also access the generated artifacts by using the Application Navigator to expand the folders of the new SmartUpgrade project.

Figure 4–2 shows how you can expand the Application Sources folder of the SmartUpgrade project to view the generated `weblogic.xml` file, as well as a pointer to the existing EAR file that was used to generate the Upgrade Report and the sample Oracle WebLogic Server deployment descriptor.

Figure 4–3 shows how you can expand the project to view the generated Web services output folders. For more information, see Section 4.3.4, "About the Content of the Generated Web Services Output Directories".

**Figure 4–2 Locating Generated Deployment Descriptor Artifacts in Oracle JDeveloper****Figure 4–3 Locating the Web Services Output Folders in Oracle JDeveloper**

## 4.2 Using Web Application Deployment Descriptor Artifacts Generated by SmartUpgrade

For information about using the deployment descriptor artifacts that are generated by SmartUpgrade, refer to the following sections:

- [Differences Between OC4J and Oracle WebLogic Server Deployment Descriptors](#)
- [List of Deployment Descriptor Elements Generated by SmartUpgrade](#)
- [Using Web Application Deployment Descriptor Artifacts Generated by SmartUpgrade](#)

### 4.2.1 Differences Between OC4J and Oracle WebLogic Server Deployment Descriptors

One of the key differences between OC4J and Oracle WebLogic Server is the fact that OC4J supports a set of OC4J-specific deployment descriptor elements (for example, orion-web.xml), and Oracle WebLogic Server supports a set of WebLogic-specific

deployment descriptor elements, which are stored in its equivalent proprietary files (for example, `weblogic.xml`).

For more information, see "Comparison of OC4J and Oracle WebLogic Server Deployment Descriptors" in the *Oracle Fusion Middleware Upgrade Guide for Java EE*.

One of the important tasks you must perform when upgrading your applications from OC4J to Oracle WebLogic Server is creating a `weblogic.xml` file that contains a set of elements equivalent to those previously defined in the `orion-web.xml` file.

To help you perform this task, Oracle provides the following resources:

- The information available in the *Oracle Fusion Middleware Upgrade Guide for Java EE*
- The advice provided in the SmartUpgrade report that is generated when you analyze your application with SmartUpgrade

## 4.2.2 List of Deployment Descriptor Elements Generated by SmartUpgrade

Besides reviewing the upgrade documentation and reviewing the advice available in the upgrade report findings, you can also configure SmartUpgrade to generate a specific set of Oracle WebLogic Server Web application deployment descriptor elements for you.

Specifically, if you configure SmartUpgrade to generate artifacts, and the software identifies an `orion-web.xml` file within the application, SmartUpgrade automatically generates the equivalent Oracle WebLogic Server deployment descriptor elements in a sample `weblogic.xml` file:

- `virtual-directory`
- `resource-eve-ref-mapping`
- `ejb-ref-mapping`
- `default-char-set`
- `jsp-print-nulls`
- `default-mime-type`
- `directory-browsing`
- `persistence-path`

## 4.2.3 Using the Generated Web Application Deployment Descriptor Elements

If your application contains a Web application, then SmartUpgrade can generate the deployment descriptor elements listed [Section 4.2.2, "List of Deployment Descriptor Elements Generated by SmartUpgrade"](#). The generated elements are saved in a sample `weblogic.xml` file.

[Figure 4-4](#) shows a flow chart of the steps you can take to generate and use the generated deployment descriptor elements. [Figure 4-4](#) describes each of the steps in the flow chart in more detail.

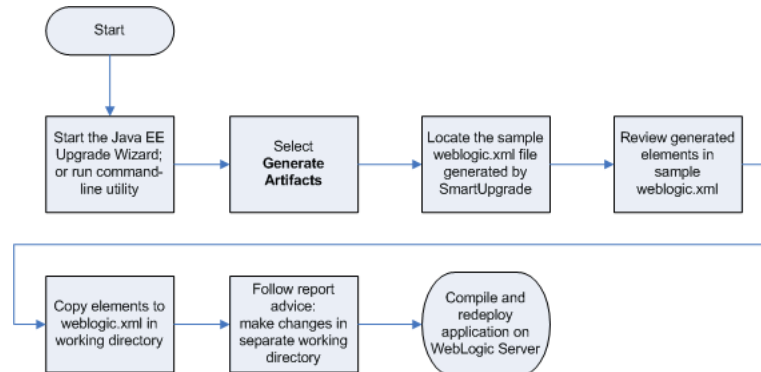
To use the deployment descriptor artifacts generated SmartUpgrade:

1. Locate the generated `weblogic.xml` file and review the elements that SmartUpgrade created in the file.

For more information, see [Section 4.1, "Locating the Artifacts Generated by SmartUpgrade"](#).

2. If you are unfamiliar with any of the elements, refer to the deployment descriptor information in the *Oracle Fusion Middleware Upgrade Guide for Java EE*.
3. Copy the elements from the generated `weblogic.xml` file and paste them into the `weblogic.xml` file you created in the project where you are modifying your application for redeployment on Oracle WebLogic Server.

**Figure 4–4 Flow Chart of Upgrading a Java EE Application and Generating Web Application Deployment Descriptor Artifacts**



**Table 4–1 Description of the Steps Required to Generate and Use Deployment Descriptor Elements Generated by SmartUpgrade**

| Step   | Description  |
|--|--|
| Start the Java EE Upgrade Wizard; or run command-line utility              | From Oracle JDeveloper, start the Java EE Upgrade wizard, as described in <a href="#">Section 2.2, "Starting and Using the Java EE Upgrade Wizard"</a><br>From the command-line interface, run the command-line arguments, as described in <a href="#">Chapter 3, "Using the SmartUpgrade Command Line"</a> .  |
| Select <b>Generate Artifacts</b>   | From Oracle JDeveloper, select Generate Artifacts on the OC4J Artifacts screen in the Java EE Upgrade wizard.<br>From the command-line interface, use the <code>-generate</code> command-line option.  |
| Locate the sample <code>weblogic.xml</code> file generated by SmartUpgrade | For more information, see <a href="#">Section 4.1, "Locating the Artifacts Generated by SmartUpgrade"</a> .  |
| Review generated elements in sample <code>weblogic.xml</code>              | If you are unfamiliar with any of the generated elements, refer to the Oracle WebLogic Server deployment descriptor information in the <i>Oracle Fusion Middleware Upgrade Guide for Java EE</i>   |
| Copy elements to <code>weblogic.xml</code> in working directory            | The <code>weblogic.xml</code> generated by SmartUpgrade upgrade is a sample file.<br>To upgrade your application, you should edit the files in a working project directory that is a copy of your original application project directory. As you follow the advice in the upgrade report findings, copy the appropriate elements in the generated sample into the working copy of your <code>weblogic.xml</code> file. |
| Follow report advice and make changes in separate working directory        | For each finding in the report, follow the advice and make the necessary modifications to the files in your working directory.<br>Your working directory should be a separate project directory where you are upgrading your application. This directory is typically separate from the application and project created for the upgrade report findings and for the artifacts generated by SmartUpgrade.               |

**Table 4–1 (Cont.) Description of the Steps Required to Generate and Use Deployment Descriptor Elements Generated by SmartUpgrade**

| Step  | Description   |
|---|---|
| Compile and redeploy application on WebLogic Server | <p>After you make the changes identified in the upgrade report, you can then compile your application and check for compilation errors. If the compilation is successful, redeploy your application on Oracle WebLogic Server.</p> <p>For information on deploying applications to Oracle WebLogic Server, refer to the following resource:</p> <ul style="list-style-type: none"> <li>▪ <i>Oracle Fusion Middleware Deploying Applications to Oracle WebLogic Server</i></li> <li>▪ "Deploying Applications" in the <i>Oracle Fusion Middleware Administrator's Guide</i></li> </ul> |

## 4.3 Using Web Services Artifacts Generated by SmartUpgrade

For information about using the Web services artifacts that are generated by SmartUpgrade, refer to the following sections:

- [Differences Between OC4J and Oracle WebLogic Server Web Services](#)
- [How Web Services Artifact Generation Differs From Deployment Descriptor Generation](#)
- [Capabilities of the SmartUpgrade Web Services Upgrade](#)
- [About the Content of the Generated Web Services Output Directories](#)
- [Using Artifacts Generated for a Dedicated Web Services Application](#)
- [Using Artifacts Generated for an Application with Both Web Services and Other Java EE Elements](#)

### 4.3.1 Differences Between OC4J and Oracle WebLogic Server Web Services

For general information about the differences between OC4J and Oracle WebLogic Server Web services, refer to the following resources:

- "Task 6: Upgrade the Application Web Services" in the *Oracle Fusion Middleware Upgrade Guide for Java EE*
- "Overview of WebLogic Web Services" in *Oracle Fusion Middleware Introducing WebLogic Web Services for Oracle WebLogic Server*

### 4.3.2 How Web Services Artifact Generation Differs From Deployment Descriptor Generation

Unlike other features of OC4J, SmartUpgrade automates much of the upgrade process for Web services.

For example, for most of the other application features and technologies, you analyze the application or project with SmartUpgrade and then follow the advice in each finding within the SmartUpgrade upgrade report.

Alternatively, for Web services applications, you can let SmartUpgrade do most of the work. If you configure SmartUpgrade to generate artifacts, and the application contains Web services, then SmartUpgrade does not generate findings for the Web services in your application. Instead, it automatically generates a set of artifacts that in most cases are ready to be deployed on Oracle WebLogic Server.

In fact, if the application is a dedicated Web services application (with no other features to upgrade) you can configure SmartUpgrade to generate an archive of the



generated Web services artifacts. The resulting archive can be deployed directly on Oracle WebLogic Server. For more information, see [Section 2.2.3, "Setting Properties When Upgrading OC4J Web Services with SmartUpgrade"](#).

### 4.3.3 Capabilities of the SmartUpgrade Web Services Upgrade

When you generate Oracle WebLogic Server Web services artifacts, SmartUpgrade upgrades the following OC4J Web services to Oracle WebLogic Server 10.3 JAX-RPC Web services:

- OC4J 10g Release 3 (10.1.3) Web services
- OC4J 10g Release 3 (10.1.3) EJB 2.1 Web services
- OC4J 10g Release 3 (10.1.3) EJB 3.0 Web services
- OC4J 10g Release 2 (10.1.2) Web services

The upgraded Web services use the same `contextpath` and the context URI as the OC4J 10g Web services. For example, suppose the URI for your OC4J 10g Release 3 (10.1.3) Web services were as follows:

```
http://localhost:8888/contextpath/soap11
http://localhost:8888/contextpath/soap12
```

After the upgrade, you should be able to access the same Web services via the following URI on the Oracle WebLogic Server domain:

```
http://localhost:7001/contextpath/soap11
http://localhost:7001/contextpath/soap12
```

During the upgrade process, SmartUpgrade automatically merges Web services related deployment descriptors (such as `web.xml` and `weblogic-webservices.xml`) and generates "glue code" that allows your upgraded Web services to communicate with your existing remote endpoints.

For more information about the Web services upgrade tasks that SmartUpgrade performs for you, see [Section 1.4.2, "Generation of Web Services Artifacts"](#).

You can optionally configure SmartUpgrade to analyze your application specifically to determine the performance impact of the generated glue code. For more information, see [Section 4.4, "Analyzing the Performance Impact of the Web Services Glue Code Generated by SmartUpgrade"](#).

### 4.3.4 About the Content of the Generated Web Services Output Directories

By default, SmartUpgrade saves all generated artifacts, including the Web services artifacts, in the `/gen_src/` directory inside your new project directory (if you are using Oracle JDeveloper) or in the upgrade folder (if you are using the command-line interface). For more information, see [Section 4.1, "Locating the Artifacts Generated by SmartUpgrade"](#).

The Web services artifacts are output to a subdirectory called `web-services-artifacts`, which is created under the default application-specific output directory ([Figure 4-1](#)).

For example, if your application archive is called `myWSapplication.ear`, then the Web services artifacts are created in the following directory structure:

```
project_directory/gen_src/myWSapplication.d/web-services-artifacts
    /myWSapplication-output/
```

Within the output directory, you will find:

- A set of log files with the extension `.out` and `.err`, which identify the output and error log files for each phase of the Web services upgrade.
- The source upgrade plan (`config.xml`) and target upgrade plan (`outConfig.xml`).
- The configuration files used to assemble the Web services.
- The subdirectories described in [Table 4-2](#).

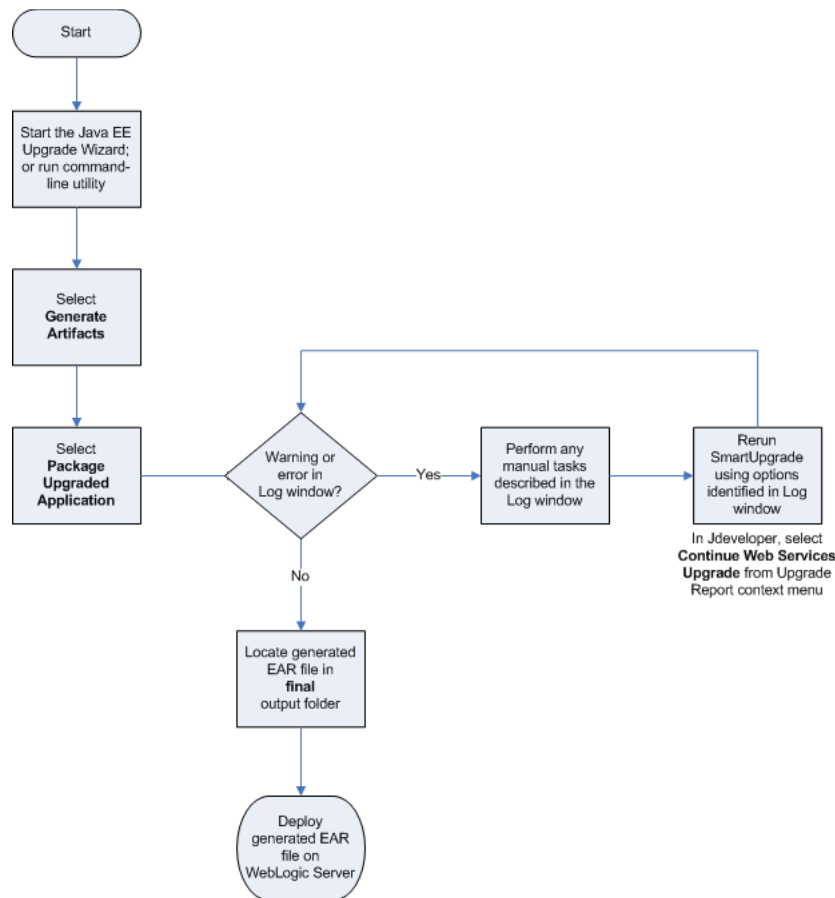
**Table 4-2 Summary of the Web Services Output Directories**

| Directory    | Description  | When is it created or overwritten?  |
|--------------|--|---|
| final        | <p>If you configured SmartUpgrade to package the upgraded application into an archive, then this folder contains the final, upgraded, and deployable Web services application.</p> <p>The application is expanded on disk, as well as in the form of a packaged enterprise archive file.</p>   | <p>This directory gets overwritten and recreated each time you run SmartUpgrade with the options for generating Web services artifacts and packaging the upgrade results in an EAR file.</p> <p>Specifically, the directory is overwritten when you select <b>Package Upgraded Application</b> on the OC4J Web Service Upgrade wizard page or use the <code>-autoPack</code> command-line option.</p> <p>The directory is recreated by merging the content of the <b>exploded</b> and <b>intermediate</b> directories.</p>  |
| intermediate | <p>This directory contains the upgraded Web services artifacts and glue code that are not yet packaged in the final deployable application archive.</p> <p>It contains intermediate information only for Web service artifacts.</p>  | <p>This directory gets overwritten and recreated each time you run SmartUpgrade when it is configured to generate Web services glue code.</p> <p>You can prevent SmartUpgrade from overwriting this directory by enabling the option to skip the generation of glue code.</p>   |
| exploded     | <p>This directory contains the original, input application archive, as-is, exploded on disk, with no changes.</p> <p>You can perform manual changes in this directory to manually upgrade non-Web services components.</p> <p>However, note that if you run SmartUpgrade again, you must enable the option to skip upgrade plan generation; otherwise, the directory and your changes are overwritten.</p> | <p>This directory gets overwritten and is recreated each time you run SmartUpgrade when it is configured to generate a source upgrade plan.</p> <p>You can prevent SmartUpgrade from overwriting this directory by selecting <b>Skip Source Upgrade Plan Generation</b> on the OC4J Web Services Upgrade wizard page or by using the <code>-skipSourcePlan</code> command-line option.</p> <p><b>Note:</b> If you select <b>Skip Glue Code Generation</b> (or use the <code>-skipGlueCode</code> command-line option), then SmartUpgrade also enables the <code>-skipTargetPlan</code> option, which in turn enables the <code>-skipSourcePlan</code> option.</p> |

### 4.3.5 Using Artifacts Generated for a Dedicated Web Services Application

If the application you are upgrading is a dedicated Web services application, then you can configure SmartUpgrade to package the contents of the output folder into a deployable archive.

[Figure 4-5](#) shows a flow chart of the steps required to generate a deployable EAR file that contains the upgraded Web services artifacts. [Table 4-3](#) describes each of the steps in the flow chart.

**Figure 4–5 Flow Chart of Upgrading a Dedicated Web Services Application****Table 4–3 Description of the Flow Chart Steps When Upgrading a Dedicated Web Services Application**

| Step  | Description  |
|---|--|
| Start the Java EE Upgrade Wizard; or run command-line utility | From Oracle JDeveloper, start the Java EE Upgrade wizard, as described in <a href="#">Section 2.2, "Starting and Using the Java EE Upgrade Wizard"</a><br>From the command-line interface, run the command-line arguments, as described in <a href="#">Chapter 3, "Using the SmartUpgrade Command Line"</a> .                        |
| Select <b>Generate Artifacts</b>                              | From Oracle JDeveloper, select Generate Artifacts on the OC4J Artifacts screen in the Java EE Upgrade wizard.<br>From the command-line interface, use the <code>-generate</code> command-line option.  |
| Select <b>Package Upgraded Application</b>                    | From Oracle JDeveloper, select <b>Package Upgraded Application</b> from the OC4J Web Services Upgrade page in the Java EE Upgrade wizard.<br>From the command-line interface, use the <code>-autoPack</code> command-line option.  |
| Warning or error in Log window?                               | Check the SmartUpgrade log output to see if an error or warning appears, such as the one shown in <a href="#">Example 2–1</a> .  |
| Perform any manual tasks described in the Log window          | The warning or error in the Log window will include specific instructions for how to address any issues found with the Web services in your application.<br>Follow the instructions in the warning or error. Some of these actions will involve manipulating files in the <b>intermediate</b> or <b>exploded</b> output directories. |

**Table 4–3 (Cont.) Description of the Flow Chart Steps When Upgrading a Dedicated Web Services**

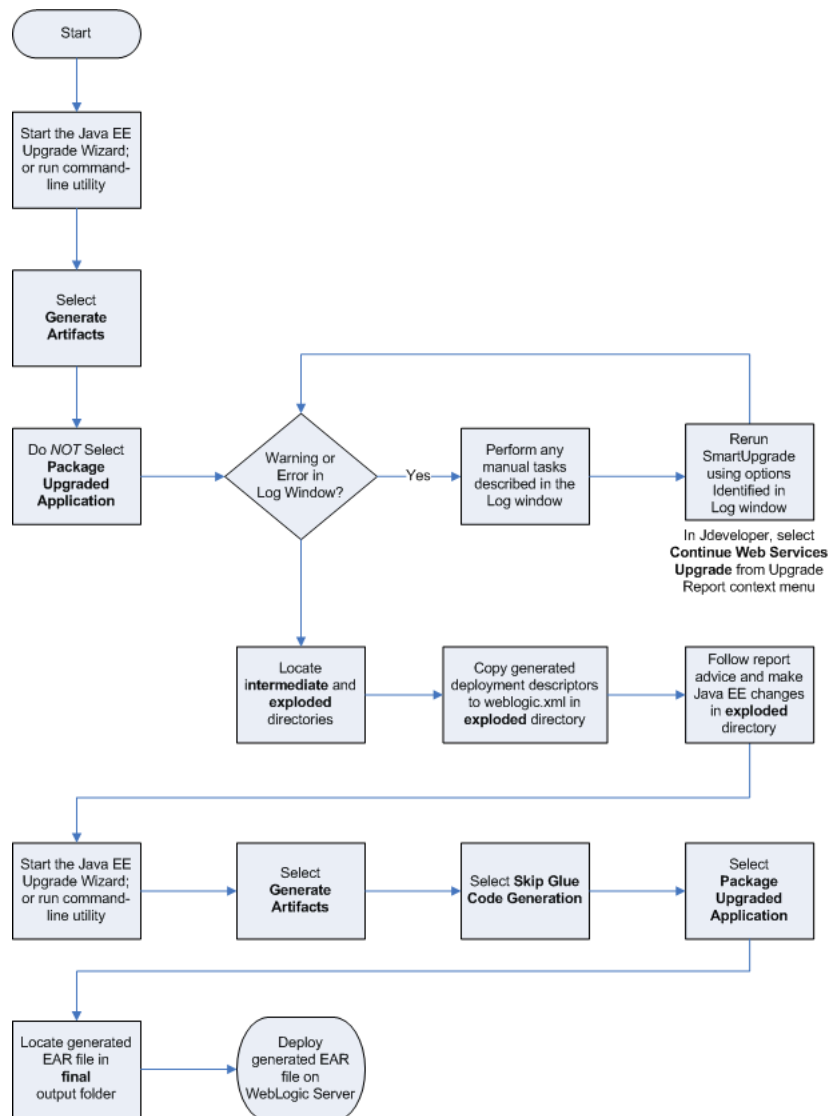
| Step  | Description   |
|---|---|
| Rerun SmartUpgrade using options identified in Log window | In Oracle JDeveloper, follow the instructions in <a href="#">Section 2.2.4, "Continuing with the Upgrade of Web Services Artifacts"</a> .<br>If you are using the command-line interface, then run the SmartUpgrade command again, using the options indicated in the warning or error message.   |
| Locate generated EAR file in final output folder          | Refer to <a href="#">Section 4.1, "Locating the Artifacts Generated by SmartUpgrade"</a> for more information about the SmartUpgrade output folders.  |
| Deploy generated EAR file on WebLogic Server              | For information on deploying applications to Oracle WebLogic Server, refer to the following resource: <ul style="list-style-type: none"> <li>▪ <i>Oracle Fusion Middleware Deploying Applications to Oracle WebLogic Server</i></li> <li>▪ "Deploying Applications" in the <i>Oracle Fusion Middleware Administrator's Guide</i></li> </ul> |

### 4.3.6 Using Artifacts Generated for an Application with Both Web Services and Other Java EE Elements

If your application contains OC4J Web services and also contains other Java EE features that must be updated, then you can generate the Web services artifacts, update the application with your additional required changes, and then repackage the application.

[Figure 4–6](#) shows a flow chart of the steps required to perform this task. XXX provides an explanation of each step.

**Figure 4–6 Flow Chart of Upgrading an Application That Contains Web Services and Other Java EE Features That Require Upgrade**



**Table 4–4 Description of the Steps in the Web Services and Java EE Application Upgrade Flow Chart**

| Step  | Description   |
|---|---|
| Start the Java EE Upgrade Wizard; or run command-line utility | From Oracle JDeveloper, start the Java EE Upgrade wizard, as described in <a href="#">Section 2.2, "Starting and Using the Java EE Upgrade Wizard"</a><br>From the command-line interface, run the command-line arguments, as described in <a href="#">Chapter 3, "Using the SmartUpgrade Command Line"</a> . |
| Select <b>Generate Artifacts</b>                              | From Oracle JDeveloper, select Generate Artifacts on the OC4J Artifacts screen in the Java EE Upgrade wizard.<br>From the command-line interface, use the <code>-generate</code> command-line option.   |
| Do <b>NOT</b> select <b>Package Upgraded Application</b>      | From Oracle JDeveloper, clear the <b>Package Upgraded Application</b> check box on the OC4J Web Services Upgrade page in the Java EE Upgrade wizard.<br>From the command-line interface, do not use the <code>-autoPack</code> command-line option.   |
| Warning or Error in Log Window?                               | Check the SmartUpgrade log output to see if an error appears, such as the one shown in <a href="#">Example 2–1</a> .  |

**Table 4–4 (Cont.) Description of the Steps in the Web Services and Java EE Application Upgrade Flow**

| Step  | Description   |
|---|---|
| Perform any manual tasks described in the Log window  | The warning or error in the Log window will include specific instructions for how to address any issues found with the Web services in your application.<br>Follow the instructions in the warning or error. Some of these actions will involve manipulating files in the <b>intermediate</b> or <b>exploded</b> output directories.  |
| Rerun SmartUpgrade using options identified in Log window                                       | In Oracle JDeveloper, follow the instructions in <a href="#">Section 2.2.4, "Continuing with the Upgrade of Web Services Artifacts"</a> .<br>If you are using the command-line interface, then run the SmartUpgrade command again, using the options indicated in the warning message.  |
| Locate <b>intermediate</b> and <b>exploded</b> directories                                      | For more information, see <a href="#">Section 4.1, "Locating the Artifacts Generated by SmartUpgrade"</a> .   |
| Copy generated deployment descriptors to <code>weblogic.xml</code> in <b>exploded</b> directory | As you update the deployment descriptors in the <b>exploded</b> directory, copy any generated elements from the sample <code>weblogic.xml</code> file in the new SmartUpgrade project to a <code>weblogic.xml</code> deployment descriptor structure in the exploded directory.<br>For more information, see <a href="#">Section 4.2.3, "Using the Generated Web Application Deployment Descriptor Elements"</a> .  |
| Follow report advice and make Java EE changes in exploded directory                             | For each finding in the report, follow the advice and make the necessary modifications to the files in the <b>exploded</b> directory.<br>Modifications you make to the files in this directory will be saved and can be packaged into the final EAR file later, as long as you do not run SmartUpgrade and generate a source upgrade plan.<br><b>CAUTION:</b> To be sure your changes to this directory are not overwritten, after you make changes to the directory, be sure to select <b>Skip Glue Code Generation</b> in Oracle JDeveloper or use the <code>-skipGlueCode</code> option when running the SmartUpgrade command-line interface; otherwise, your changes will be overwritten by SmartUpgrade. |
| Start the Java EE Upgrade Wizard; or run command-line utility                                   | From Oracle JDeveloper, start the Java EE Upgrade wizard, as described in <a href="#">Section 2.2, "Starting and Using the Java EE Upgrade Wizard"</a><br>From the command-line interface, run the command-line arguments, as described in <a href="#">Chapter 3, "Using the SmartUpgrade Command Line"</a> .   |
| Select <b>Generate Artifacts</b>  | From Oracle JDeveloper, select <b>Generate Artifacts</b> on the OC4J Artifacts screen in the Java EE Upgrade wizard.<br>From the command-line interface, use the <code>-generate</code> command-line option.  |
| Select <b>Skip Glue Code Generation</b>   | From Oracle JDeveloper, select <b>Skip Glue Code Generation</b> on the OC4J Artifacts page in the Java EE Upgrade wizard.<br>From the command-line interface, use the <code>-skipGlueCode</code> option.<br><b>CAUTION:</b> If you made changes to the <b>exploded</b> directory, it is important that you select this option; otherwise your changes will be overwritten.  |
| Select <b>Package Upgraded Application</b>  | From Oracle JDeveloper, select <b>Package Upgraded Application</b> from the OC4J Web Services Upgrade page in the Java EE Upgrade wizard.<br>From the command-line interface, use the <code>-autoPack</code> command-line option.   |
| Locate generated EAR file in final output folder  | Refer to <a href="#">Section 4.1, "Locating the Artifacts Generated by SmartUpgrade"</a> for more information about the SmartUpgrade output folders.  |
| Deploy generated EAR file on WebLogic Server  | For information on deploying applications to Oracle WebLogic Server, refer to the following resource: <ul style="list-style-type: none"> <li>■ <i>Oracle Fusion Middleware Deploying Applications to Oracle WebLogic Server</i></li> <li>■ "Deploying Applications" in the <i>Oracle Fusion Middleware Administrator's Guide</i></li> </ul>   |

## 4.4 Analyzing the Performance Impact of the Web Services Glue Code Generated by SmartUpgrade

To upgrade your OC4J Web services to Oracle WebLogic Server, SmartUpgrade generates Java programming code called "glue code," which allows your upgraded Web services to continue to communicate with your existing external partners and URIs.

To determine the performance impact of the generated glue code:

1. Select one of the following options when you use SmartUpgrade to upgrade your application and generate Web services artifacts:
  - If you are using Oracle JDeveloper, select **Generate Instrumented Code for Performance Analysis** on the OC4J Web Services Upgrade page of the Java EE Upgrade wizard.
  - If you are using the SmartUpgrade command-line interface, use the `-evaluate` command-line option.

If you enable this option, then SmartUpgrade instruments the generated wrapper ("glue") code for performance analysis.

Note that this option is appropriate only for testing the application, and not for production environments.

2. Perform the upgrade of your application and deploy the application on Oracle WebLogic Server.
3. After the application is deployed and running successfully on Oracle WebLogic Server, locate the following HTML file in the Oracle WebLogic Server domain directory:

```
MW_HOME/user_projects/domains/domain_name/webservice_port_name.html
```

4. Open the HTML file in a browser.

The browser displays the Glue Code Performance Analysis page. By default, no data is displayed in each column of the table.

5. Exercise the Web service to generate some performance data.

You can test the Web service using the Oracle WebLogic Server Administration Console or the Oracle Enterprise Manager Fusion Middleware Control.

For more information, see "Testing Your Web Services" in the *Oracle Fusion Middleware Security and Administrator's Guide for Web Services*.

6. Refresh the browser where the Glue Code Performance Analysis page is displayed.

Each time you test the Web service, an entry is added to the Glue Code Performance page. Use the data on this page to measure the impact of the glue code on the performance of the Web service.

**Note:** When you are finished analyzing the performance metrics, repackage the application for production environment, by running SmartUpgrade again with the following options:

- If you are using Oracle JDeveloper, then do not select **Generate Instrumented Code for Performance Analysis** and select **Skip Target Upgrade Plan Generation**.
- If you are using the command-line interface, then use the `-evaluate false` and `-skipTargetPlan true` options when you run SmartUpgrade.

**Figure 4-7 SmartUpgrade Web Services Upgrade - Glue Code Performance Analysis Page**

| S.NO | Operation Name    | Argument Glue Overhead | Business Call Time | Return Type Glue Overhead | Operation Total Time |
|------|-------------------|------------------------|--------------------|---------------------------|----------------------|
| 1.   | retStringArrayOut | 0 s, 4 ms              | 1 ms               | 0 ms                      | 0 s, 5 ms            |
| 2.   | retStringArrayOut | 0 ms                   | 0 ms               | 0 ms                      | 0 ms                 |
| 3.   | retStringArrayOut | 0 ms                   | 0 ms               | 0 ms                      | 0 ms                 |
| 4.   | retStringArrayOut | 0 ms                   | 0 ms               | 0 ms                      | 0 ms                 |



## A

---

Accept Duplicate Classes  
option on the OC4J Web Services Upgrade wizard page, 3-9

acceptDuplicates  
Web services command-line option, 3-9

Additional Classpath  
option on the OC4J Web Services Upgrade wizard page, 3-6

additionalClassPath  
Web services command-line option, 3-6, 3-12

adf  
SmartUpgrade rule category  
definition, 3-13

Advanced  
button on OC4J Artifacts page, 2-7

advice  
category within a finding in a SmartUpgrade report, 1-2

Apache Ant  
integrating with SmartUpgrade, 3-14

Apache Ant Version 1.7, 1-6

api  
SmartUpgrade rule category  
definition, 3-13

app-client  
SmartUpgrade rule category  
definition, 3-13

Application Name  
page in the Java EE Upgrade wizard, 2-4

application-jars  
command-line locator, 3-4

Archive Type  
option on the OC4J Web Services Upgrade wizard page, 3-6

archive-homes  
command-line locator, 3-3

archiveType  
Web services command-line option, 3-6

artifact generation  
about, 1-2  
deployment descriptors, 1-3  
Web services artifacts, 1-3

autoPack  
Web services command-line option, 3-6, 4-9, 4-11,

4-12

autoWrap  
Web services command-line option, 3-6

## C

---

category  
command-line option, 3-2, 3-4, 3-13

Check for Updates  
feature in JDeveloper, 1-4  
JDeveloper feature, 1-5

classloading  
SmartUpgrade rule category  
definition, 3-13

cluster  
SmartUpgrade rule category  
definition, 3-13

command-line interface, 3-1  
basic syntax of, 3-2  
examples of using, 3-11  
getting help on, 3-2  
installation of SmartUpgrade command-line, 1-6  
list of options, 3-5  
summary of options, 3-2  
using, 3-1  
verifying prerequisites, 3-1

Companion CD-ROM, 1-6

Complexity  
filtering report findings by, 2-11

context URI  
preserving during upgrade, 4-7

Continue Web Service Upgrade  
option on context menu, 2-9

## D

---

data-sources  
SmartUpgrade rule category  
definition, 3-13

dateFormat  
Web services command-line option, 3-6

debug  
Web services command-line option, 3-9

debugControl  
Web services command-line option, 3-9

debugWrap

- Web services command-line option, 3-9
- debugWrapLength
  - Web services command-line option, 3-10
- default-char-set
  - deployment descriptor element, 4-4
- default-mime-type
  - deployment descriptor element, 4-4
- deinstallation
  - of SmartUpgrade Release 1.1, 1-4
- deployment descriptors
  - differences between OC4J and WebLogic Server, 4-3
  - elements generated by SmartUpgrade, 4-4
  - generation of, 1-3
  - OC4J-specific, 1-3
  - using generated elements, 4-4
- directory-browsing
  - deployment descriptor element, 4-4
- documentation
  - obtaining the latest, 1-7
- downloading
  - of SmartUpgrade, 1-4

## E

---

- ears
  - command-line locator, 3-3, 3-11
- ejb
  - SmartUpgrade rule category
    - definition, 3-13
- EJB 2.1 Web services, 4-7
- EJB 3.0 Web services, 4-7
- EJB Lookup Name
  - option on the OC4J Web Services Upgrade wizard page, 3-7
- ejbLookupName
  - Web services command-line option, 3-7
- ejbNewWarBase
  - Web services command-line option, 3-7
- ejbNewWarContext
  - Web services command-line option, 3-7
- ejb-ref-mapping
  - deployment descriptor element, 4-4
- evaluate
  - Web services command-line option, 3-7, 3-12, 4-13
- examples of using command-line interface, 3-11
- exploded
  - Web services artifact output directory, 4-8, 4-12
- extension
  - installing the SmartUpgrade JDeveloper, 1-5
  - verifying installation of JDeveloper, 2-1

## F

---

- final
  - Web services artifact output directory, 4-8
- finding
  - within a SmartUpgrade report, 1-2
- findings

## Index-2

- sorting and filtering within a report, 2-11
- flow chart
- upgrading a dedicated Web services application, 4-9
- upgrading a Java EE application and generating deployment descriptors, 4-5
- upgrading an application with Web services and other Java EE features, 4-11

Format for Date-String Conversion

- option on the OC4J Web Services Upgrade wizard page, 3-6

## G

---

- /gen\_src/
  - default output directory, 4-7
- generate
  - command-line option, 3-2, 3-4, 3-5, 3-12, 4-5, 4-9, 4-11, 4-12
- Generate Artifacts
  - option on OC4J Web Services Upgrade wizard page, 4-9
  - option on the OC4J Web Services Upgrade wizard page, 4-5, 4-11, 4-12
- Generate Instrumented Code for Performance Analysis
  - option on OC4J Web Services Upgrade wizard page, 2-8
  - option on the OC4J Web Services Upgrade wizard page, 3-7, 4-13
- Generate Parameter Wrapping
  - option on the OC4J Web Services Upgrade wizard page, 3-6
- Generate Tasks Timeout (seconds)
  - option on the OC4J Web Services Upgrade wizard page, 3-8
- Generate Upgrade Plans Only
  - option on the OC4J Web Services Upgrade wizard page, 3-10
- generated artifacts
  - differences between deployment descriptor and Web services artifacts, 4-6
  - list of artifacts generated by SmartUpgrade, 4-4
  - locating, 4-1
    - in JDeveloper, 4-2
    - on disk, 4-1
  - using as part of application upgrade, 4-1
  - using Web services artifacts, 4-6
- glue code, 4-7, 4-8
- Glue Code Performance Analysis Page, 4-14

## H

---

- html
  - command-line option, 3-2, 3-4
  - Web services command-line option, 3-12

## I

---

- implication
  - category within a finding in a SmartUpgrade

- report, 1-2
- installation
  - of SmartUpgrade, 1-4
  - verifying in JDeveloper, 2-1
- intermediate
  - Web services artifact output directory, 4-8, 4-12

## J

---

- jars
  - command-line locator, 3-3
- Java 2 Enterprise Edition Version 1.6, 1-6
- Java 2 Standard Edition, 1-6
- Java EE Upgrade Application Wizard
  - OC4J Artifacts page, 2-5
  - OC4J Web Services Upgrade wizard page, 2-6
  - Upgrade Report Type page, 2-5
- Java EE Upgrade Application wizard, 2-1
- Java EE Upgrade Project wizard in JDeveloper, 2-3
- Java EE Upgrade Wizard
  - starting, 2-2
  - starting and using, 2-2
  - summary of pages in, 2-3
  - wizard, 2-3
- javaHome
  - Web services command-line option, 3-7
- jca
  - SmartUpgrade rule category
    - definition, 3-13
- jdeveloper\_smartupgrade.zip, 1-5
- JDK Home for Target Server Tools
  - option on the OC4J Web Services Upgrade wizard page, 3-7
- jms
  - SmartUpgrade rule category
    - definition, 3-13
- jmx
  - SmartUpgrade rule category
    - definition, 3-13
- jndi
  - SmartUpgrade rule category
    - definition, 3-13
- jsp-print-nulls
  - deployment descriptor element, 4-4
- jta
  - SmartUpgrade rule category
    - definition, 3-13

## L

---

- LOCATOR\_NAME
  - in command-line interface, 3-2

## N

---

- New Gallery Dialog, 2-2

## O

---

- OC4J Artifacts

- page in the Java EE Upgrade wizard, 2-4
- OC4J Web services
  - generating an archive deployable on WebLogic Server, 2-8
  - setting properties when upgrading, 2-7
- OC4J Web Services Upgrade page
  - in the Java EE Upgrade wizard, 2-4
  - learning about the properties on, 2-7
  - setting properties on, 2-7
- Oracle Application Server 10g Release 2 (10.1.2)
  - analyzing applications deployed on, 2-7
  - analyzing with command-line interface, 3-14
- Oracle Application Server 10g Release 3 (10.1.3)
  - analyzing applications deployed on, 2-7
- Oracle JDeveloper
  - Check for Updates feature, 1-4
  - Java EE Upgrade Application wizard, 2-1
  - using SmartUpgrade with, 2-1
  - verifying installation of SmartUpgrade extension, 2-1
- Oracle Technology Network (OTN)
  - obtaining SmartUpgrade from, 1-4
  - upgrade page on, 1-7
- Oracle WebLogic Server SmartUpgrade
  - See SmartUpgrade
- oracle.jdeveloper.smartupgrade.weblogic.jar, 1-4
- oracle.smartupgrade.weblogic
  - folder in JDeveloper directory, 1-4
- output directories
  - about the content of, 4-7
  - generated by SmartUpgrade, 4-2

## P

---

- Package Upgraded Application
  - option on OC4J Web Services Upgrade wizard page, 2-8
  - option on the OC4J Web Services Upgrade wizard page, 3-6, 4-9, 4-11, 4-12
- Packaging Includes
  - option on the OC4J Web Services Upgrade wizard page, 3-8
- packLIBs
  - Web services command-line option, 3-8
- performance
  - analyzing the performance impact of SmartUpgrade glue code, 4-13
- Performance Analysis
  - generating instrumented code for, 2-8
- persistence-path
  - deployment descriptor element, 4-4
- plansOnly
  - Web services command-line option, 3-10
- Priority
  - filtering report findings by, 2-11
- processTimeout
  - Web services command-line option, 3-8
- Project Name
  - page in the Java EE Upgrade wizard, 2-4

## Q

---

- qos
  - Web services command-line option, 3-8
- Quality of Service Policies
  - option on the OC4J Web Services Upgrade wizard page, 3-8

## R

---

- rars
  - command-line locator, 3-3
- readme.txt, 1-5, 1-6
- reason
  - category within a finding in a SmartUpgrade report, 1-2
- releasenotes.txt, 1-5
- Resolve Mapping Ambiguity
  - option on the OC4J Web Services Upgrade wizard page, 3-10
- resolveMapAmbiguity
  - Web services command-line option, 3-10
- resource-eve-ref-mapping
  - deployment descriptor element, 4-4
- rmi
  - SmartUpgrade rule category
    - definition, 3-13
- rule categories
  - limiting analysis to select, 2-6
  - limiting to specific
    - using the command-line, 3-13

## S

---

- security
  - SmartUpgrade rule category
    - definition, 3-13
- server-config
  - command-line locator, 3-3
- Skip Glue Code Generation
  - option on the OC4J Web Services Upgrade wizard page, 3-10, 4-12
- Skip Source Upgrade Plan Generation
  - option on the OC4J Web Services Upgrade wizard page, 3-10
- Skip Target Upgrade Plan Generation
  - option on the OC4J Web Services Upgrade wizard page, 3-11
- skipGlueCode
  - Web services command-line option, 3-10, 4-12
- skipSourcePlan
  - Web services command-line option, 3-10
- skipTargetPlan
  - Web services command-line option, 3-11
- SmartUpgrade
  - artifact generation
    - introduction to, 1-2
  - command-line interface, 3-1
  - definition, 1-1
  - deinstalling Release 1.1, 1-4
  - downloading and installing, 1-4

- how it works, 1-1
- installation of command-line interface, 1-6
- installing the JDeveloper extension, 1-5
- introduction
- limiting analysis to specific rule categories, 2-6
- obtaining installation ZIP files, 1-4
- output directories, 4-2
- using a SmartUpgrade report, 1-2
- using an upgrade report, 2-9
- using as part of overall Fusion Middleware upgrade, 1-1
  - using with JDeveloper, 2-1
- smartupgrade.jar, 1-6, 3-1
- smartupgrade.zip, 1-5, 1-6
- soa
  - SmartUpgrade rule category
    - definition, 3-13
- sourceStack
  - Web services command-line option, 3-8
- structure window
  - for upgrade report, 2-11

## T

---

- target
  - command-line option, 3-2, 3-4, 3-14
- Target Server Home Directory
  - option on the OC4J Web Services Upgrade wizard page, 2-8, 3-8
- targetStackHome
  - Web services command-line option, 3-8, 3-12
- toplink\_patch.jar, 1-4
- Type
  - filtering report findings by, 2-11

## U

---

- Upgrade Findings Toolbar, 2-11
  - summary of options on, 2-11
- upgrade report
  - in JDeveloper, 2-10
  - introduction to, 1-2
  - structure window, 2-11
  - using, 2-9
  - viewing, 2-10
- Upgrade Report Type
  - page in the Java EE Upgrade wizard, 2-4

## V

---

- virtual-directory
  - deployment descriptor element, 4-4

## W

---

- WAR Base Name
  - option on the OC4J Web Services Upgrade wizard page, 3-7
- WAR Context Root
  - option on the OC4J Web Services Upgrade wizard page, 3-7

- wars
  - command-line locator, 3-3
- Web services
  - capabilities of SmartUpgrade upgrade, 4-7
  - differences between OC4J and Web services, 4-6
  - generation of artifacts during upgrade, 1-3
  - output directories
    - locating in JDeveloper, 4-3
  - summary of output directories during
    - upgrade, 4-8
  - upgrading, 1-3
  - upgrading dedicated Web services
    - applications, 4-8
- Web services artifacts
  - continuing with upgrade of, 2-8
  - sample warning message in log file, 2-9
- web-app
  - SmartUpgrade rule category
    - definition, 3-13
- webcache
  - SmartUpgrade rule category
    - definition, 3-13
- weblogic-webservices.xml, 4-7
- weblogic.xml, 4-4, 4-5
  - locating generated file in JDeveloper, 4-3
- web-services
  - SmartUpgrade rule category
    - definition, 3-13
- web.xml, 4-7
- Wrapper Null Value Allowed
  - option on the OC4J Web Services Upgrade wizard
    - page, 3-9
- wrapperNullAllowed
  - Web services command-line option, 3-9

## **Z**

---

- ZIP files
  - obtaining SmartUpgrade installation files, 1-4

