# Oracle® Endeca Information Discovery Studio

Studio Installation Guide

Version 3.0.0 Rev. A • May 2013

ORACLE

# Copyright and disclaimer

# Table of Contents

**Copyright and disclaimer** . . . . . **ii**

**Preface** . . . . . **vi**
About this guide . . . . . vi
Who should use this guide . . . . . vi
Conventions used in this document . . . . . vi
Contacting Oracle Customer Support . . . . . vii

## Part I: Before You Begin

**Chapter 1: Overview of Studio and the Installation Process** . . . . . **2**
About Oracle Endeca Information Discovery Studio . . . . . 2
About the Provisioning Service . . . . . 2
Other modules in Oracle Endeca Information Discovery . . . . . 3
Recommended installation order for Studio and its associated modules . . . . . 4

**Chapter 2: System Requirements** . . . . . **5**
Studio system requirements . . . . . 5
Provisioning Service system requirements . . . . . 7

**Chapter 3: Downloading the Studio and Provisioning Service Software** . . . . . **9**

## Part II: Installing Studio

**Chapter 4: About the Studio Installation Process** . . . . . **13**

**Chapter 5: Installing Studio Using the Tomcat Bundle for Windows** . . . . . **14**
Installing the Windows Tomcat bundle . . . . . 14
Changing the context root for the Windows Tomcat bundle . . . . . 15

**Chapter 6: Installing Studio Using the Tomcat Bundle for Linux** . . . . . **17**
Installing the Linux Tomcat bundle . . . . . 17
Changing the context root in the Linux Tomcat bundle . . . . . 18

**Chapter 7: Installing Studio on Tomcat 6.0** . . . . . **20**
About installing on Tomcat 6.0 . . . . . 20
Installing Tomcat and deploying the dependency libraries . . . . . 21
Modifying the Tomcat configuration to work with Studio . . . . . 22
Deploying and starting Studio . . . . . 23

**Chapter 8: Running Studio Under Tomcat as a Windows Service** . . . . . **25**
About running Studio as a Windows service . . . . . 25
Obtaining the service installer files . . . . . 25

Configuring the service . . . . . 26
Installing and starting the service . . . . . 27
Troubleshooting the service installation . . . . . 28

**Chapter 9: Installing Studio on Oracle WebLogic Server 11gR1 (10.3.6) . . . . . 30**
About installing Studio on WebLogic Server . . . . . 30
Downloading and installing the Sun JDK . . . . . 31
Downloading and installing WebLogic Server . . . . . 34
Creating and configuring a WebLogic domain for Studio . . . . . 40
Deploying Studio to the WebLogic domain . . . . . 46
Changing to a non-root context for Studio on WebLogic Server . . . . . 49
Changing the Liferay Home directory for Studio on WebLogic Server . . . . . 50

**Chapter 10: Troubleshooting Known Studio Issues . . . . . 52**
Changing the JavaScript timeout value on Internet Explorer 8 . . . . . 52
Increasing the Tomcat connector keepAliveTimeout for Internet Explorer 8 . . . . . 53
Increasing or disabling the WebLogic Activation Timeout . . . . . 53

**Chapter 11: Getting Started with Studio . . . . . 54**
Starting and logging in to Studio . . . . . 54
Displaying the Studio Control Panel . . . . . 54
Updating the default data source to point to your Endeca Server . . . . . 55

**Chapter 12: Changing the Database Used to Store Studio Application Data . . . . . 57**
About using a different database . . . . . 57
Overview of the process for switching to a different database . . . . . 57

**Chapter 13: Using Studio with a Reverse Proxy . . . . . 59**
About reverse proxies . . . . . 59
What is a reverse proxy? . . . . . 59
Types of reverse proxies . . . . . 59
Example sequence for a reverse proxy request . . . . . 60
Recommendations for reverse proxy configuration . . . . . 60
Preserving HTTP 1.1 Host: headers . . . . . 61
Enabling the Apache ProxyPreserveHost directive . . . . . 61
Reverse proxy configuration options for Studio . . . . . 62
Simple Studio reverse proxy configuration . . . . . 62
Studio reverse proxy configuration without preserving Host: headers . . . . . 62
Configuring Studio to support an SSL-enabled reverse-proxy . . . . . 63

**Chapter 14: Creating a Studio Cluster . . . . . 64**
About Studio clustering . . . . . 64
Setting up the cluster . . . . . 65
Installing the Studio instances . . . . . 65
Configuring synchronized caching for the Studio instances . . . . . 66
About synchronized caching . . . . . 66
Updating portal-ext.properties to enable synchronized caching . . . . . 66
Customizing the clustered cache configuration files . . . . . 67

Clearing the cache for a cluster . . . . . 68

## Part III: Installing the Provisioning Service

**Chapter 15: Installing the Provisioning Service . . . . . 71**
Provisioning Service installation prerequisites . . . . . 72
Provisioning Service installation process . . . . . 72
Provisioning Service domain creation and installation . . . . . 73
Configuring the Provisioning Service . . . . . 80
Implementing SSL communication with Endeca Server and Studio . . . . . 82
Configuring the Provisioning Service for non-secure communication . . . . . 83
Setting up a production database . . . . . 83
Checking the Provisioning Service installation . . . . . 84

## Part IV: Uninstalling Oracle Endeca Information Discovery

**Chapter 16: Uninstalling Oracle Endeca Information Discovery Studio . . . . . 86**
Uninstalling Studio . . . . . 86
Uninstalling the Provisioning Service . . . . . 86

# Preface

Oracle® Endeca Information Discovery Studio is an enterprise data discovery platform for advanced, yet intuitive, exploration and analysis of complex and varied data.

Information is loaded from disparate source systems and stored in a faceted data model that dynamically supports changing data. This integrated and enriched data is made available for search, discovery, and analysis via interactive and configurable applications. Oracle Information Discovery Studio includes a Provisioning Service that allows you to upload data directly from spreadsheet files.

Oracle Endeca Information Discovery Studio enables an iterative “model-as-you-go” approach that simultaneously frees IT from the burdens of traditional data modeling and supports the broad exploration and analysis needs of business users.

# About this guide

This guide contains installation instructions for installing and setting up Oracle Endeca Information Discovery Studio on Windows and Linux.

Installing Oracle Endeca Information Discovery Studio consists of installing:

- Studio
- Provisioning Service

Oracle Endeca Information Discovery Studio requires the Oracle Endeca Server and Oracle Endeca Information Discovery Integrator.

# Who should use this guide

This guide is intended for system administrators installing Oracle Endeca Information Discovery Studio on Windows or Linux.

# Conventions used in this document

The following conventions are used in this document.

## Typographic conventions

The following table describes the typographic conventions used in this document.

| Typeface | Meaning |
|---|---|
| **User Interface Elements** | This formatting is used for graphical user interface elements such as pages, dialog boxes, buttons, and fields. |
| `Code Sample` | This formatting is used for sample code phrases within a paragraph. |

| Typeface | Meaning |
|---|---|
| *Variable* | This formatting is used for variable values.<br>For variables within a code sample, the formatting is *`Variable`*. |
| `File Path` | This formatting is used for file names and paths. |

### Symbol conventions

The following table describes symbol conventions used in this document.

| Symbol | Description | Example | Meaning |
|---|---|---|---|
| > | The right angle bracket, or greater-than sign, indicates menu item selections in a graphic user interface. | File > New > Project | From the File menu, choose New, then from the New submenu, choose Project. |

# Contacting Oracle Customer Support

Oracle Customer Support provides registered users with important information regarding Oracle software, implementation questions, product and solution help, as well as overall news and updates from Oracle.

You can contact Oracle Customer Support through Oracle's Support portal, My Oracle Support at *https://support.oracle.com*.

# Part I

## Before You Begin

Chapter 1

# Overview of Studio and the Installation Process

Here is a brief overview of Oracle Endeca Information Discovery Studio and the installation process.

*About Oracle Endeca Information Discovery Studio*

*About the Provisioning Service*

*Other modules in Oracle Endeca Information Discovery*

*Recommended installation order for Studio and its associated modules*

## About Oracle Endeca Information Discovery Studio

Oracle Endeca Information Discovery Studio is a web-based application that allows business analysts to rapidly assemble dashboard applications. These applications enable analysts and other end users to explore a full range of structured and unstructured enterprise data from an Endeca Server.

Studio is easy to deploy and is ideal for the agile development of enterprise-quality applications. Studio provides a library of UI components that embody best practices in information discovery applications. Because Studio is component-based, Studio applications are simple to control, adapt, and extend.

### About the Studio Component SDK

The Studio Component SDK is a packaged development environment for components, layout templates, and other Studio elements.

For information on installing and using the Component SDK, see the *Studio Administration and Customization Guide*.

## About the Provisioning Service

Oracle Endeca Information Discovery Studio includes the Provisioning Service, which enables dynamic application creation from data uploaded from the desktop.

The Provisioning Service profiles and creates data before sending it to Endeca Server for ingest.

**Note:** In Version 3.0 the Provisioning Service only supports the upload of Excel files.

The Provisioning Service runs as a web application in a WebLogic Server container.

# Other modules in Oracle Endeca Information Discovery

Oracle Endeca Information Discovery Studio includes the web-based Studio application and the Provisioning Service, used to upload data from spreadsheet files. Studio and the Provisioning Service are contained in the same download.

Oracle Endeca Information Discovery 3.0.0 also includes the following separately-downloaded modules:

| Module and Version | Description |
|---|---|
| **Oracle Endeca Server 7.5.1** | Oracle Endeca Server provides the query engine that serves as the foundation for Studio applications. These applications can answer queries from and provide business analytics information application end users.<br><br>Oracle Endeca Server also allows you to administer Endeca data domains, which store the source data records that are loaded by Integrator or the Provisioning Service.<br><br>Each Studio application is connected to an Endeca data domain. |
| **Oracle Endeca Information Discovery Integrator 3.0.0** | Oracle Endeca Information Discovery Integrator is a high-performance data integration platform that allows you to extract source records from a different types of sources (from flat files to databases), then load that data into an Endeca Server data domain.<br><br>The Integrator suite includes:<br><br>• **Integrator.** Used to build graphs to load source data into and configure the schema for an Endeca Server data domain.<br><br>• **Information Discovery connectors.** Integrator components used to perform various data ingest operations on Endeca data domains.<br><br>• **Integrator Server.** Used to run graphs in an enterprise-wide environment. In this environment, different users and user groups can access and run the graphs.<br><br>• **Integrator Acquisition System.** Optional component. The Integrator Acquisition System, or IAS, is a set of components that crawl source data stored in a variety of formats including: file systems, Content Management Systems, Web servers, and custom data sources.<br><br>IAS transforms the data, if necessary, and outputs the data to an XML file or a Record Store that can be accessed by Integrator for use in the Endeca Server. |

Oracle Endeca Server and Oracle Endeca Information Discovery Integrator have separate documentation sets, including separate installation guides.

Note that Studio 3.0.0 is only compatible with the above versions of Oracle Endeca Server and Integrator. It is not compatible with earlier versions of these modules.

# Recommended installation order for Studio and its associated modules

Following the recommended order of installation helps you minimize dependencies among the different products.

Note that if you are installing all of the modules on the same WebLogic server, make sure that you create a separate WebLogic domain for each module, and that each domain has a unique port.

Oracle recommends the following installation order:

1. Oracle Endeca Server. For instructions on installing Oracle Endeca Server, including system requirements, see the *Oracle Endeca Server Installation Guide*.
2. Oracle Endeca Information Discovery Integrator. For instructions on installing Oracle Endeca Information Discovery Integrator, including system requirements, see the *Integrator Installation Guide*.
3. Studio (see *About the Studio Installation Process on page 13*). After installing, to verify the installation, log in to Studio. Use a created Endeca data domain to provide the data source.
4. Provisioning Service (See *Installing the Provisioning Service on page 71*).
5. Integrator Server. For instructions about installing Integrator Server, see the *Integrator Installation Guide*.
6. Integrator Acquisition System. For instructions about installing IAS, see the *Integrator Acquisition System Installation Guide*.

Chapter 2

# System Requirements

Oracle Endeca Information Discovery Studio has the following system requirements. For details on the Oracle Endeca Server system requirements, see the *Oracle Endeca Server Installation Guide*. For details on the Oracle Endeca Information Discovery Integrator system requirements, see the *Integrator Installation Guide*.

*Studio system requirements*

*Provisioning Service system requirements*

## Studio system requirements

Here are the system requirements for Studio.

The installation instructions also assume that you have a running Oracle Endeca Server at which to point Studio. This version of Studio is compatible with the 7.5.x version of Oracle Endeca Server.

### Hardware requirements

Minimum hardware requirements:

- x64 processor, minimum 1.8 GHz
- At least 2 GB of RAM, depending on the size of the application data set
- 80 GB hard drive, depending on the size of the application data set

Recommended hardware requirements:

- x64 3.0+ GHz processors; we recommend Intel Xeon (including Nehalem) or AMD Opteron processors
- 8 GB of RAM or more, depending on the size of the application data set
- High performance network-attached storage (for example, attached via a dedicated iSCSI or fibre channel network) or high performance locally-attached RAID storage (for example, a RAID 6 or RAID 0+1 array with battery-backed write caching, operating on 72GB or 146 GB spindles at 10k or 15k RPM spindle speed)
- Gigabit Ethernet

**Note:** In this guide, the term "x64" refers to any processor compatible with the AMD64/EM64T architecture. You might need to upgrade your hardware, depending on the data you are processing. All run-time code must fit entirely in RAM. Likewise, hard disk capacity must be sufficient based on the size of your data set. Please contact your Oracle representative if you need more information on sizing your hardware.

## Supported operating systems

| Platform | Description |
|---|---|
| **Oracle Linux 5** | Oracle Linux 5 (version 5 for x64) running on x64 processors. Only the Red Hat Compatible Kernel is supported. |
| **Linux RHEL 5** | • Red Hat Enterprise Linux Server (version 5 for x64) running on x64 processors.<br>• Red Hat Enterprise Linux Advanced Platform (version 5 for x64) running on x64 processors.<br>For best performance on Red Hat Linux version 5 (Server and Advanced), we recommend the latest version of RHEL 5. |
| **Windows 2008** | Windows Server 2008 R2 Enterprise running on x64 processors. |

## Software requirements

Studio is a Web-based application that runs in an application server. It supports the following software:

| Software | Supported Versions |
|---|---|
| Application server | • Tomcat 6. The Studio Tomcat bundle includes Tomcat 6.0.36.<br>• Oracle WebLogic Server 11gR1 (10.3.6) |
| Java | Sun Java 6, update 18 or greater. We recommend that you use the latest Java 6 patch. The Studio Tomcat bundle includes version 1.6.0_38. |
| Database system | • MySQL 5.1<br>• Oracle 11g |
| Browser | • Firefox ESR on Windows<br>• Internet Explorer 8 (with compatibility mode disabled) on Windows<br>• Internet Explorer 9 on Windows<br>Firefox is recommended.<br>  **Important:** Running Internet Explorer 8 in compatibility mode is not supported. |
| Browser plugin | Adobe Flash 10.0 or greater. We recommend using the most recent version of Flash supported by the browser. |

### Alternative database support

The Studio server uses a relational database to store configuration and state. Hypersonic is used by default, but is not recommended for production use due to performance issues. Studio has been tested on MySQL and Oracle 11g. However, many other databases are expected to work.

Customers should feel free to use any database, including shared systems they may already have in place. As with application servers, customers who choose to deploy on un-tested databases will always be supported on any issue that can be traced back to core Studio code and can be reproduced on a supported database.

For information on changing the database used, see *Changing the Database Used to Store Studio Application Data on page 57*.

## Provisioning Service system requirements

Here are the system requirements for the Provisioning Service.

The installation instructions assume a running Endeca Server is available to which Provisioning Service can connect and which Provisioning Service will provision with data uploaded by the user. This version of Provisioning Service is compatible with Oracle Endeca Server Version 7.5.1.

### Hardware requirements

Minimum hardware requirements:

- x64 1.8 GHz processor
- 4 GB of RAM

Recommended hardware specifications:

- x64 3.0+ GHz processors; we recommend Intel Xeon (including Nehalem) or AMD Opteron processors
- 8 GB
- Gigabit Ethernet

**Note:** In this guide, the term "x64" refers to any processor compatible with the AMD64/EM64T architecture. You might need to upgrade your hardware, depending on the data you are processing. All run-time code must fit entirely in RAM. Likewise, hard disk capacity must be sufficient based on the size of your data set. Please contact your Oracle representative if you need more information on sizing your hardware.

### Supported operating systems

| Platform | Description |
|---|---|
| **Oracle Linux 5** | Oracle Linux 5 (version 5 for x64) running on x64 processors. Only the Red Hat Compatible Kernel is supported. |

| Platform | Description |
|---|---|
| **Linux RHEL 5** | • Red Hat Enterprise Linux Server (version 5 for x64) running on x64 processors.<br>• Red Hat Enterprise Linux Advanced Platform (version 5 for x64) running on x64 processors.<br>For best performance on Red Hat Linux version 5 (Server and Advanced), we recommend the latest version of RHEL 5. |
| **Windows 2008** | Windows Server 2008 R2 Enterprise running on x64 processors. |

## Software requirements

Provisioning Service is a Web-based application that runs in a WebLogic Server web application server. It supports the following software:

| Software | Supported Versions |
|---|---|
| Application server | Oracle WebLogic Server 11gR1 (10.3.6) |
| Java | Sun Java 6, update 18 or greater, 64-bit version |
| RDBMS | Oracle 11g |

Chapter 3

# Downloading the Studio and Provisioning Service Software

You download Studio and the Provisioning Service from the Oracle Software Delivery Cloud.

For details on downloading the Oracle Endeca Server, see the *Oracle Endeca Server Installation Guide*.

For details on downloading Oracle Endeca Information Discovery Integrator, see the *Integrator Installation Guide*.

To download the Studio software:

1. Log in to `https://edelivery.oracle.com`.
2. Accept the terms and restrictions.
3. On the **Media Pack Search** page:
   (a) From the **Select a Product Pack** drop-down list, select **Oracle Endeca**.
   (b) From the **Platform** drop-down list, select the platform you are installing on.
   (c) Click **Go**.

   The list of media packs for the selected product and platform is displayed.

   For the Windows platform, the media pack is **Oracle Endeca Information Discovery Studio (3.0) Media Pack v1 for Microsoft Windows x64 (64-bit)**.

   For the Linux platform, the media pack is **Oracle Endeca Information Discovery Studio (3.0) Media Pack v1 for Linux x86-64**.
4. In the media pack list, click the name of the media pack.

   The **Download** page displays, listing the downloads for the selected media pack.
5. From the **Download** page, click the **Download** link for the appropriate version of Studio for your application server, and for the Provisioning Service.

   The download file names use the part number associated with that product.

   For Windows, the available downloads are:

| Title | Description |
|---|---|
| **Oracle Endeca Information Discovery Studio (3.0) for Microsoft Windows x64 (64-bit) and Tomcat** | Studio Tomcat bundle. |
| **Oracle Endeca Information Discovery Studio (3.0) for Microsoft Windows x64 (64-bit) and WebLogic** | Studio to install on a WebLogic server. |

| Title | Description |
|---|---|
| **Oracle Endeca Information Discovery Studio Standalone (3.0) for Microsoft Windows x64 (64-bit)** | Studio to install as a standalone application on Tomcat. |
| **Oracle Endeca Information Discovery Provisioning Service (3.0) for Microsoft Windows x64 (64-bit)** | Provisioning Service installation files. |
| **Oracle Endeca Information Discovery Studio SDK (3.0) for Microsoft Windows x64 (64-bit)** | Studio Component SDK. Used for development of custom components.<br>For information on developing custom components, see the *Studio Administration and Customization Guide*. |
| **Oracle Endeca Information Discovery Sample Application (3.0) for Microsoft Windows x64 (64-bit)** | Sample application files.<br>For information on importing and viewing the sample application, see the *Oracle Endeca Information Discovery Getting Started Guide*. |
| **Oracle Endeca Information Discovery Studio (3.0) Documentation** | Installation and migration documentation for Studio and the Provisioning Service.<br>The full documentation set is available from the *Oracle Technology Network*. |

For Linux, the available downloads are:

| Title | Description |
|---|---|
| **Oracle Endeca Information Discovery Studio (3.0) for Linux x86-64 and Tomcat** | Studio Tomcat bundle. |
| **Oracle Endeca Information Discovery Studio (3.0) for Linux x86-64 and WebLogic** | Studio to install on a WebLogic server. |
| **Oracle Endeca Information Discovery Studio Standalone (3.0) for Linux x86-64** | Studio to install as a standalone application on Tomcat. |
| **Oracle Endeca Information Discovery Provisioning Service (3.0) for Linux x86-64** | Provisioning Service installation files. |
| **Oracle Endeca Information Discovery Studio SDK (3.0) for Linux x86-64** | Studio Component SDK. Used for development of custom components.<br>For information on developing custom components, see the *Studio Administration and Customization Guide*. |

| Title | Description |
| --- | --- |
| **Oracle Endeca Information Discovery Studio (3.0) Documentation** | Installation and migration documentation for Studio and the Provisioning Service.<br>The full documentation set is available from the *Oracle Technology Network*. |

# Part II

## Installing Studio

Chapter 4

# About the Studio Installation Process

You install Studio using one of the available installation options. For production environments, you must change the default Studio database. You may also need to update the default data source.

## Available options for installing Studio

The available options for installing Studio are:

- Studio Tomcat bundle for Windows. This is based on Tomcat 6 and Java 1.6. See *Installing Studio Using the Tomcat Bundle for Windows on page 14*.
- Studio Tomcat bundle for Linux Tomcat bundle. This is based on Tomcat 6 and Java 1.6. See *Installing Studio Using the Tomcat Bundle for Linux on page 17*.
- Studio as a standalone application on a Tomcat application server. See *Installing Studio on Tomcat 6.0 on page 20*.
- Studio as a standalone application on Oracle WebLogic Server 11g. WebLogic can be used on both Linux and Windows. See *Installing Studio on Oracle WebLogic Server 11gR1 (10.3.6) on page 30*.

## Note on changing the default Studio database

The installation instructions deploy Studio using the default embedded Hypersonic database, which is not intended for production use.

In production, you must deploy using an alternate database. For more information about this process, see *Changing the Database Used to Store Studio Application Data on page 57*.

Briefly, to deploy an alternate database, you modify the `portal-ext.properties` file to specify the appropriate JDBC connection information for the desired database.

## Note on configuring the default data source

The data sources in your Studio application must always include a default data source. This data source is automatically assigned to all data-source-backed components when they are initially added to a page.

When you install a new instance of Studio, it creates a placeholder `default` data source. For details on configuring this data source to point to your server, see *Updating the default data source to point to your Endeca Server on page 55*.

Chapter 5

# Installing Studio Using the Tomcat Bundle for Windows

In the Tomcat bundle version of Studio, Tomcat 6 and the JVM 1.6 are embedded. You can, however, use your own version of Java 6, as long as it's update 18 or greater.

*Installing the Windows Tomcat bundle*

*Changing the context root for the Windows Tomcat bundle*

## Installing the Windows Tomcat bundle

To install the Tomcat bundle, you extract the Studio application, then add the Studio components.

The Studio Tomcat bundle download for Windows contains the following files:

- `EID_3.0.x_Studio_portal.zip`
- `EID_3.0.x_Studio_components.zip`

To install the Studio Tomcat bundle:

1. Unzip `EID_3.0.x_Studio_portal.zip` to the directory of your choice.

   Studio creates a directory called `endeca-portal` in that directory. For example, if you unzip into `C:`, Information Discovery Studio installs into `C:\endeca-portal`.

   It is recommended that you create a directory `C:\Oracle\Endeca\Discovery\3.0.0\` to extract the file into.

2. Extract the `.war` files from `EID_3.0.x_Studio_components.zip` and place them into the `endeca-portal\deploy` directory.

   The `.war` files go in the root of `endeca-portal\deploy`. There should be no subdirectories.

   **Note:** This directory already contains hooks and layouts required by the portal. It is safe to overwrite these files with the versions in `EID_3.0.x_Studio_Components.zip`.

3. If the environment variables `CATALINA_HOME` or `JAVA_HOME` are already set, update them to point to your newly installed Tomcat directory and a valid 1.6 JRE.

   For example, set `CATALINA_HOME` to `C:\<path_to_endeca-portal>\tomcat-6.0.36`, and set `JAVA_HOME` to `C:\<path_to_endeca-portal>\tomcat-6.0.36\jre1.6.0_21\win`.

   If you do not have these environment variables set, you can leave them un-set.

4. To start the portal's Tomcat instance, run `endeca-portal\tomcat-6.0.36\bin\startup.bat`.

   **Note:** Server startup can take several minutes. You can follow the log messages to ascertain when the process is complete. Do not shut down the Tomcat window while Studio is running.

5. To test that the application is running, go to the portal (`http://localhost:8080/`) in your browser. Log in using the following default credentials:

| Field | Value |
|---|---|
| **Login:** | `admin@oracle.com` |
| **Password:** | `Welcome123` |

   You are immediately prompted to change the password. The new password must contain:

   - At least 6 characters
   - At least one non-alphabetic character

6. Optionally, you can set up *log4j* logging. `log4j` provides configurable, Java-based logging in an open-source utility.

   **Note:** For more information about Studio logging, see the *Studio Administration and Customization Guide*.

# Changing the context root for the Windows Tomcat bundle

Optionally, after installing the Windows Tomcat bundle, you can change the context root.

When you first install the bundle, it uses the default context root of "/", and you get access to Studio by going to `http://localhost:8080/`. If you change the context root to "sales", then the URL would be `http://localhost:8080/sales`.

Note that any change to the context root only affects the Studio application. Components and other hooks still use their default context paths.

To change the context root:

1. Stop the server.
2. Rename `endeca-portal\tomcat-6.0.36\conf\Catalina\localhost\ROOT.xml` file to `<`*`context root`*`>.xml`.

   For example, if your context root is `sales`, the file name should be `sales.xml`.

   For multi-level context paths, separate the name with #. For example, for a context path of `/sales/east`, the file name should be `sales#east.xml`.

3. Modify the XML file created in the previous step as needed:
   - For a root context: `<Context path="" />`
   - For a context of `/sales`: `<Context path="/sales"/>`
   - For a context of `/sales/east`: `<Context path="/sales/east"/>`
4. Rename the `endeca-portal\tomcat-6.0.36\webapps\ROOT` directory to `endeca-portal\tomcat-6.0.36\webapps\<`*`context root`*`>`.

   For multi-level context paths, use a multi-level path such as: `endeca-portal\tomcat-6.0.36\webapps\sales#east`.

5. Edit the `endeca-portal\portal-ext.properties` file.

   Find the `portal.ctx` property at the beginning of the file, and change the value to be the same context root value you used above.

   Do not include a trailing slash in the `portal.ctx` value. For example, use this value:

   ```
   portal.ctx=/mycompany/portal
   ```

   Do not use this value:

   ```
   portal.ctx=/mycompany/portal/
   ```

6. Restart the server.

Chapter 6

# Installing Studio Using the Tomcat Bundle for Linux

In the Linux Tomcat bundle version of Studio, Tomcat 6 is embedded.

*Installing the Linux Tomcat bundle*

*Changing the context root in the Linux Tomcat bundle*

## Installing the Linux Tomcat bundle

To install the Tomcat bundle, you extract the Studio application, then add the Studio components.

The Studio Tomcat bundle download for Linux contains the following files:

- `EID_3.0.x_Studio_portal.tgz`
- `EID_3.0.x_Studio_components.zip`

To install the Studio Linux Tomcat bundle:

1. Extract `EID_3.0.x_Studio_portal.tgz` to the directory of your choice.

   Studio creates a directory called `endeca-portal` in that directory.

   It is recommended that you create a directory `Oracle/Endeca/Discovery/3.0.0/` to extract the file into.
2. Extract the `.war` files from `EID_3.0.x_Studio_components.zip` and place them into the `endeca-portal/deploy` directory. The `.war` files go in the root of `endeca-portal/deploy`. There should be no subdirectories.

   **Note:** This directory already contains hooks and layouts required by the portal. It is safe to overwrite these files with the versions in `EID_3.0.x_Studio_components.zip`.
3. If the environment variable `CATALINA_HOME` is already set, update it to point to your newly installed Tomcat directory.
4. Make sure that the `JAVA_HOME` environment variable is set to point to a valid 1.6 JRE.
5. To start the portal's Tomcat instance, run `endeca-portal/tomcat-6.0.36/bin/startup.sh`.

   **Note:** Server startup can take several minutes. You can follow the log messages to ascertain when the process is complete.

6. To test that the application is running, go to the portal (`http://localhost:8080/`) in your browser. Log in using the following default credentials:

| Field | Value |
|---|---|
| **Login:** | `admin@oracle.com` |
| **Password:** | `Welcome123` |

   You are immediately prompted to change the password. The new password must contain:

   - At least 6 characters
   - At least one non-alphabetic character

7. Optionally, you can set up *log4j* logging. `log4j` provides configurable, Java-based logging in an open-source utility.

   **Note:** For more information about Studio logging, see the *Studio Administration and Customization Guide*.

# Changing the context root in the Linux Tomcat bundle

Optionally, you can change the context root used by your Studio application.

When you first install the bundle, it uses the default context root of "/", and you get access to Studio by going to `http://localhost:8080/`. If you change the context root to "sales", then the URL would be `http://localhost:8080/sales`.

Note that any change to the context root only affects the Studio application. Components and other hooks still use their default context paths.

To change the context root:

1. Stop the server.
2. Rename `endeca-portal/tomcat-6.0.36/conf/Catalina/localhost/ROOT.xml` file to `<context root>.xml`.

   For example, if your context root is `sales`, the file name should be `sales.xml`.

   For multi-level context paths, separate the name with #. For example, for a context path of `/sales/east`, the file name should be `sales#east.xml`.
3. Modify the XML file created in the previous step as needed:
   - For a root context: `<Context path="" />`
   - For a context of `/sales`: `<Context path="/sales"/>`
   - For a context of `/sales/east`: `<Context path="/sales/east"/>`
4. Rename the `endeca-portal/tomcat-6.0.36/webapps/ROOT` directory to `endeca-portal/tomcat-6.0.36/webapps/<context root>`.

   For multi-level context paths, use a multi-level path such as: `endeca-portal/tomcat-6.0.36/webapps/sales#east`.

5. Edit the `endeca-portal/portal-ext.properties` file.

   Find the `portal.ctx` property at the beginning of `portal-ext.properties`.

   Change the value of this setting to be the same context root value you used above.

   Do not include a trailing slash in the `portal.ctx` value. For example, use this value:

   ```
   portal.ctx=/sales/east
   ```

   Do not use this value:

   ```
   portal.ctx=/sales/east/
   ```

6. Restart the server.

Chapter 7

# Installing Studio on Tomcat 6.0

You can also install Studio on the most recent version of the Tomcat application server, without using the Studio Tomcat bundle.

*About installing on Tomcat 6.0*

*Installing Tomcat and deploying the dependency libraries*

*Modifying the Tomcat configuration to work with Studio*

*Deploying and starting Studio*

## About installing on Tomcat 6.0

You can deploy Studio as a standalone application on Tomcat 6.0.

These instructions assume that you have obtained the `apache-tomcat-6.0.x.zip` or `tar.gz` file from the Apache Foundation, but that you have not yet installed it.

The rest of these instructions refer to the installation directory as `apache-tomcat-6.0.x`, leaving off the minor version number.

Your also must have downloaded the Studio standalone package for Windows or Linux. The standalone package contains the following files:

- `EID_3.0.x_Studio_portal_dependencies.zip`
- `EID_3.0.x_Studio_portal_war.zip`
- `EID_3.0.x_Studio_components.zip`

At a high level, to install and deploy Studio on Tomcat 6.0:

1. Install Tomcat and deploy the Studio dependency libraries.
2. Modify the Tomcat configuration to work with Studio.
3. Deploy and start Studio.

**Note:** The examples in these instructions are based on a Windows server Tomcat deployment. If you install on Linux, the steps are similar, although you must substitute Linux binaries and paths. Significant differences are called out.

# Installing Tomcat and deploying the dependency libraries

Studio requires the deployment of several Java libraries.

To install the Tomcat software and deploy the Studio dependency libraries:

1. Create an `endeca-portal` directory.

   This is the home directory for your Studio installation.

2. Unzip `apache-tomcat-6.0.x.zip` into `endeca-portal/apache-tomcat-<version>`.

   Unzipping this file creates much of the directory structure mentioned below.

3. Unzip `EID_3.0.x_Studio_portal_dependencies.zip` into a temporary directory.

   This zip file contains a collection of .jar files and other dependency files.

4. Create the `endeca-portal/apache-tomcat-<version>/endorsed` directory.

5. From the temporary directory, copy the following .jar files into `endeca-portal/apache-tomcat-<version>/endorsed`:

   - `commons-logging.jar`
   - `log4j.jar`
   - `log4j.properties.jar`

6. Under the `endeca-portal/apache-tomcat-<version>/lib` directory, create an `ext` directory.

7. From the temporary directory, copy the following .jar files into `endeca-portal/apache-tomcat-<version>/lib/ext`:

   - `activation.jar`
   - `annotations.jar`
   - `ccpp.jar`
   - `commons-lang.jar`
   - `container.jar`
   - `cxf-2.2.8.jar`
   - `cxf-orawsdl-adapter.jar`
   - `endeca-images.jar`
   - `endeca-portal.jar`
   - `hsql.jar`
   - `jabsorb.jar`
   - `jackson-core-lgpl-1.7.2.jar`
   - `jackson-mapper-lgpl-1.7.2.jar`
   - `jms.jar`
   - `jta.jar`
   - `jutf7.jar`

- mail.jar
- mdex_bindings.jar
- mysql-connector-java-commercial-5.1.17-bin.jar
- ojdbc6.jar
- oracle.http_client_11.1.1.jar
- orawsdl.jar
- portal-kernel.jar
- portal-service.jar
- portlet-container.jar
- portlet.jar
- provisioningServerWSStubs.jar
- saw-api.jar
- support-tomcat.jar
- XmlSchema-1.4.5.jar

# Modifying the Tomcat configuration to work with Studio

Before you can deploy Studio, you must modify some of the Tomcat configuration files.

To complete the Tomcat configuration updates:

1. In the `endeca-portal/apache-tomcat-<version>/bin/` directory, modify `catalina.bat` (on Windows) or `catalina.sh` (on Linux) by adding the `JAVA_OPTS` (for Linux) or `set JAVA_OPTS` (for Windows) line.

   This line should be added under the line `Execute The Requested Command` as follows (for Windows, the line needs to start with `set`):

   ```
   JAVA_OPTS=%JAVA_OPTS% -Xmx1024m -XX:MaxPermSize=256m -Dfile.encoding=UTF8
   -Duser.timezone=GMT -Djava.security.auth.login.config="%CATALINA_HOME%/conf/jaas.config"
   -Dorg.apache.catalina.loader.WebappClassLoader.ENABLE_CLEAR_REFERENCES=false
   -Djavax.xml.transform.TransformerFactory
   =com.sun.org.apache.xalan.internal.xsltc.trax.TransformerFactoryImpl
   -Djavax.xml.parsers.DocumentBuilderFactory
   =com.sun.org.apache.xerces.internal.jaxp.DocumentBuilderFactoryImpl
   -Djavax.xml.parsers.SAXParserFactory
   =com.sun.org.apache.xerces.internal.jaxp.SAXParserFactoryImpl
   ```

   Adding this line increases the memory size for the server and establishes security configuration for Studio.

2. Modify the file `endeca-portal/apache-tomcat-<version>/conf/catalina.properties` as follows to add the `ext` directory to the common class loader:

   ```
   common.loader=
       ${catalina.base}/lib,\
       ...\
       ${catalina.home}/lib/ext/*.jar
   ```

3. To deploy Studio in the root context, create a new file called `ROOT.xml` in `endeca-portal/apache-tomcat-<version>/conf/Catalina/localhost/`.

   To deploy Studio into any other context, create a new file called `<context root>.xml` in `endeca-portal/apache-tomcat-<version>/conf/Catalina/localhost/`.

   For multi-level context paths, separate the name with #. For example, for a context path of `/sales/`east, the file name should be `sales#east.xml`.

4. Modify the XML file created in the previous step as needed:
   - For a root context: `<Context path="" crossContext="true" />`
   - For a context of `/sales`: `<Context path="/sales" crossContext="true" />`
   - For a context of `/sales/east`: `<Context path="/sales/east" crossContext="true" />`

5. To support UTF-8 URI encoding, edit the `server.xml` file, located in the `endeca-portal/apache-tomcat-<version>/conf` directory, as follows:

```
<!-- Define a non-SSL HTTP/1.1 Connector on port 8080 -->
    <Connector port="8080" protocol="HTTP/1.1"
        connectionTimeout="20000"
        redirectPort="8443"
        URIEncoding="UTF-8"
    />
```

# Deploying and starting Studio

Once the Tomcat configuration is complete, you can deploy and start Studio.

To deploy and start Studio:

1. Delete the contents of the `endeca-portal/apache-tomcat-<version>/webapps/ROOT` directory.

   This directory contains the standard Web application that is installed with Tomcat by default. We will replace this standard web application with Studio in the next step.

2. Unzip `EID_3.0.x_Studio_portal_war.zip` into a temporary directory.

   This zip file contains the Studio .war file and the `copyright.txt` file.

3. Read the `copyright.txt` file and then save it to the location of your choice.

4. Unzip the contents of the .war file into `endeca-portal/apache-tomcat-<version>/webapps/ROOT`.

5. If using a non-root context for your deployment, rename the `endeca-portal\apache-tomcat-<version>\webapps\ROOT` directory to `endeca-portal\apache-tomcat-<version>\webapps\<context root>`.

   For multi-level context paths, use the multi-level path.

   For example: `endeca-portal\apache-tomcat-<version>\webapps\ROOT` to `endeca-portal\apache-tomcat-<version>\webapps\mycompany\sales`.

6. Copy the `portal-ext.properties` file from the temporary directory you created for the `EID_3.0.x_Studio_portal_dependencies.zip` to the `endeca-portal` directory.

7. Edit `endeca-portal\portal-ext.properties`.

   Find the `portal.ctx` property at the beginning of `portal-ext.properties`.

   Change the value of this setting to be the same context root value you set earlier. However, do not include a trailing slash in the `portal.ctx` value.

   For example, use this value: `portal.ctx=/mycompany/portal`

   Do not use this value: `portal.ctx=/mycompany/portal/`

8. Under the `endeca-portal` directory, create a `data` directory, and then create an `endeca-data-sources` directory below that.
9. In the `endeca-portal` directory, create a `deploy` directory.
10. Extract the .war files from `EID_3.0.x_Studio_components.zip`. Place the files into the `endeca-portal\deploy` directory.

    The .war files are placed at the root of `endeca-portal\deploy`. There should be no subdirectories.

11. To start the portal's Tomcat instance, run `endeca-portal\tomcat<version>\bin\startup.bat`.

    **Note:** Server startup can take several minutes. You can follow the log messages to ascertain when the process is complete. Do not shut down the Tomcat window while Studio is running.

12. To test that the application is running, go to the portal (`http://localhost:8080/`) in your browser. Log in using the following default credentials:

| Field | Value |
|---|---|
| **Login:** | `admin@oracle.com` |
| **Password:** | `Welcome123` |

    You are immediately prompted to change the password. The new password must contain:

    - At least 6 characters
    - At least one non-alphabetic character

13. Optionally, you can set up *log4j* logging. `log4j` provides configurable, Java-based logging in an open-source utility.

    **Note:** For more information about Studio logging, see the *Studio Administration and Customization Guide*.

Chapter 8

# Running Studio Under Tomcat as a Windows Service

If you have installed the Studio Tomcat Windows bundle or have installed Studio as a standalone application on a Windows version of Tomcat, then you can run Studio as a Windows service.

*About running Studio as a Windows service*

*Obtaining the service installer files*

*Configuring the service*

*Installing and starting the service*

*Troubleshooting the service installation*

## About running Studio as a Windows service

Running Studio as a Windows service requires the Tomcat service installer files. You also should install the Tomcat service monitor, which is used to configure and monitor the Windows service, and is useful for troubleshooting.

If you have installed the Studio Tomcat Windows bundle, then the server installer and monitor executable files are provided for you. You just need to install the service.

If you have installed Studio as a standalone application on your own instance of Tomcat, then you need to obtain the files from the Tomcat download, which is available from the Apache web site. After you obtain the files, you then configure and install the service.

## Obtaining the service installer files

If you have installed Studio as a standalone application on your own version of Tomcat, then you need to obtain the service installer and monitor files for that version. These files are part of the Tomcat download.

If you have installed the Studio Tomcat bundle for Windows, then the files are provided as part of the bundle. You can skip this step and go straight to *Installing and starting the service on page 27*.

You must use the files for Tomcat version 6.0.36.

To obtain the files and add them to Studio:

1. From the Apache Tomcat website (*http://tomcat.apache.org*), download the Tomcat file (`apache-tomcat-6.0.x-windows-x86.zip`) .

   A sample URL for the archive directory is: *http://archive.apache.org/dist/tomcat/tomcat-6/v6.0.36/bin/*

   If you are using a 64-bit JVM, then you need to download the 64-bit version of the installer (`apache-tomcat-6.0.x-windows-x64.zip`).

2. Extract the file to a temporary directory.
3. In the `bin` subdirectory of the temporary download directory, locate the following files:
   - `service.bat`
   - `tomcat6.exe`
   - `tomcat6w.exe`. This is the Tomcat service monitor.
4. Copy these files to the `bin` directory of Information Discovery Studio:

   `endeca-portal\tomcat-6.0.`*`x`*`\bin`

# Configuring the service

If you downloaded the service installer files directly from the Apache web site, then before you install the service, you must make some configuration changes to the `service.bat` file.

For the Studio Tomcat bundle for Windows, the file is already configured. You can skip this step and go directly to *Installing and starting the service on page 27*.

After you download the `service.bat` and copy it to the `bin` directory, you must update the file to:

- Edit the service name and descriptions to reflect your Studio installation
- Add additional JVM startup arguments
- Increase the memory allocation. Studio requires more memory than is set in the default values.

If you change the service name, then you also must change the name of the Tomcat monitor executable.

To update the configuration:

1. Open the file `service.bat`.
2. In the file, find the following lines:

```
set SERVICE_NAME=Tomcat6
set PR_DISPLAYNAME=Apache Tomcat 6
```

3. Change the name and display name to reflect your Studio installation. For example:

```
set SERVICE_NAME=Studio30
set PR_DISPLAYNAME=Studio 3.0
```

4. Next, find the following line:

```
set PR_DESCRIPTION=Apache Tomcat 6.0.x Server -
http://tomcat.apache.org/
```

5. Change the service description to reflect your Studio installation. For example:

```
set PR_DESCRIPTION=Studio server, version 3.0
```

6. Next, find the following line:

```
"%EXECUTABLE%" //US//%SERVICE_NAME% ++JvmOptions "-Djava.io.tmpdir=%CATALINA_BASE%\temp
;-Djava.util.logging.manager=org.apache.juli.ClassLoaderLogManager
;-Djava.util.logging.config.file
=%CATALINA_BASE%\conf\logging.properties" --JvmMs 128 --JvmMx 256
```

7. Replace the last part of the line:

```
-Djava.util.logging.config.file
=%CATALINA_BASE%\conf\logging.properties" --JvmMs 128 --JvmMx 256
```

with:

```
-Djava.util.logging.config.file=%CATALINA_BASE%\conf\logging.properties;-XX:MaxPermSize
=256m" --JvmMs 256 --JvmMx 1024
```

Make sure that there are no manual line breaks or extra spaces.

8. Immediately below that line, add the following line:

```
"%EXECUTABLE%" //US//%SERVICE_NAME% ++JvmOptions "-Dfile.encoding=UTF8
;-Djava.net.preferIPv4Stack=true;-Duser.timezone=GMT;-Djavax.xml.transform.TransformerFactory
=com.sun.org.apache.xalan.internal.xsltc.trax.TransformerFactoryImpl
;-Djavax.xml.parsers.DocumentBuilderFactory
=com.sun.org.apache.xerces.internal.jaxp.DocumentBuilderFactoryImpl
;-Djavax.xml.parsers.SAXParserFactory
=com.sun.org.apache.xerces.internal.jaxp.SAXParserFactoryImpl"
```

Make sure there are no manual line breaks or extra spaces.

9. Save and close the file.
10. Create a copy of the Tomcat monitor file (`tomcat6w.exe`). Rename the copy to be <*value of SERVICE_NAME*>`w.exe`.

    For example, if you set `SERVICE_NAME=Studio30` in `service.bat`, then you must rename the copy of the `tomcat6w.exe` to `Studio30w.exe`.

# Installing and starting the service

To install the service, you run the `service.bat` file. You also must update the Tomcat monitor to point to the JVM.

If you are not using the bundled JVM, then make sure that the `JAVA_HOME` environment variable is set to the location of your JDK. By default, `service.bat` looks for `%JAVA_HOME%\jre\server\jvm.dll`.

To install and start the Studio service:

1. From the command line, navigate to the Studio Tomcat `bin` directory.

   `endeca_portal\tomcat-6.0.`*`x`*`\bin\`

2. Run the following command:

   `service.bat install`

3. Configure the Tomcat monitor to point to the JVM:
   (a) Double-click the monitor executable (for example, `Studio23w.exe`).
   (b) On the properties dialog, click the **Java** tab.
   (c) Uncheck **Use default**.
   (d) In the **Java Virtual Machine** field, set the full path to `jvm.dll`.

   For the bundled JVM, the file is `endeca-portal\tomcat-6.0.`*`x`*`\jre1.6.0_`*`x`*`\win\bin\client\jvm.dll`.

   If you are not using the bundled JVM, then set the path to your JVM.

   (e) Click **OK**.

4. When you install the service, it is set up to be started manually. To configure the service to start automatically:
   (a) Display the **Services** list (**Control Panel>Administrative Tools>Services**).
   (b) In the list, double-click the Studio service.

   The properties dialog for the service is displayed.

   (c) From the **Startup type** drop-down list, select **Automatic**.
   (d) Click **OK**.
5. From the **Services** list, to start the service for the first time, right click the service, then click **Start**.

# Troubleshooting the service installation

If the service installs properly, but fails to start, you can use the steps provided here to troubleshoot.

For additional details on using the Tomcat service and service monitor, see *http://tomcat.apache.org/tomcat-6.0-doc/windows-service-howto.html*.

As you are troubleshooting, check the log files (`endeca-portal\tomcat-6.0.36\logs\jakarta_service*.log`) for the relevant messages.

If the service will not start:

1. If you downloaded the service installer and monitor files manually, then make sure that you have used the correct version of the Tomcat download:
   - Tomcat version 6.0.*x*
   - For a 32-bit JVM (including the bundled JVM), `apache-tomcat-6.0.x-windows-x86.zip`
   - For a 64-bit JVM, `apache-tomcat-6.0.x-windows-x64.zip`
2. If you are not using the bundled JVM, make sure that the `JAVA_HOME` environment variable is set to the location of your JDK.

   By default, `service.bat` looks for `%JAVA_HOME%\jre\server\jvm.dll`.

   To change `JAVA_HOME` after the service is installed:

   (a) Uninstall the service. To uninstall the service, run the following command:
   `service.bat remove`
   (b) Update `JAVA_HOME`.
   (c) Reinstall and restart the service.
3. Make sure the Tomcat service monitor is configured to point to the location of your JVM.
   (a) Double-click the monitor executable.
   (b) On the properties dialog, click the **Java** tab.
   (c) Uncheck **Use default**.
   (d) In the **Java Virtual Machine** field, specify the path to `jvm.dll`.

   For the bundled JVM, the file is `endeca-portal\tomcat-6.0.x\jre1.6.0_x\win\bin\client\jvm.dll`.

   If you are not using the bundled JVM, then set the path to your JVM.

   (e) Click **OK.**

4. Use the Tomcat service monitor to set the startup and shutdown modes to Java.
   (a) Double-click the monitor executable.
   (b) On the properties dialog, click the **Startup** tab.
   (c) From the **Mode** drop-down list, select **Java**.
   (d) Click the **Shutdown** tab.
   (e) From the **Mode** drop-down list, select **Java**.
   (f) Click **OK**.

Chapter 9

# Installing Studio on Oracle WebLogic Server 11gR1 (10.3.6)

You can deploy Studio as a standalone application on Oracle WebLogic Server.

*About installing Studio on WebLogic Server*

*Downloading and installing the Sun JDK*

*Downloading and installing WebLogic Server*

*Creating and configuring a WebLogic domain for Studio*

*Deploying Studio to the WebLogic domain*

*Changing to a non-root context for Studio on WebLogic Server*

*Changing the Liferay Home directory for Studio on WebLogic Server*

## About installing Studio on WebLogic Server

On WebLogic, you deploy Studio into its own WebLogic domain. These instructions assume you are installing Studio onto a new WebLogic Server instance, but for development purposes you may install Studio on the same WebLogic Server instance as Endeca Server.

These instructions for installing Studio on WebLogic Server include:

1. Downloading and installing the Sun JDK.
2. Downloading and installing WebLogic Server.
3. Creating and configuring the WebLogic domain for Studio.

   Even if you are installing Studio on the same instance of WebLogic Server as Endeca Server, you must deploy it to its own domain.
4. Deploying Studio into the WebLogic domain.

The above instructions use the default configuration. Optionally, you can deploy Studio to a different context path (see *Changing to a non-root context for Studio on WebLogic Server on page 49*) or use a different directory for Liferay Home (see *Changing the Liferay Home directory for Studio on WebLogic Server on page 50*).

For full instructions on installing all of Oracle Endeca Information Discovery (Endeca Server, Integrator, Studio, and Provisioning Service) on a single WebLogic Server in a Windows environment, see the *Oracle Endeca Information Discovery Getting Started Guide*.

# Downloading and installing the Sun JDK

Studio requires the Sun Java 6 JDK.

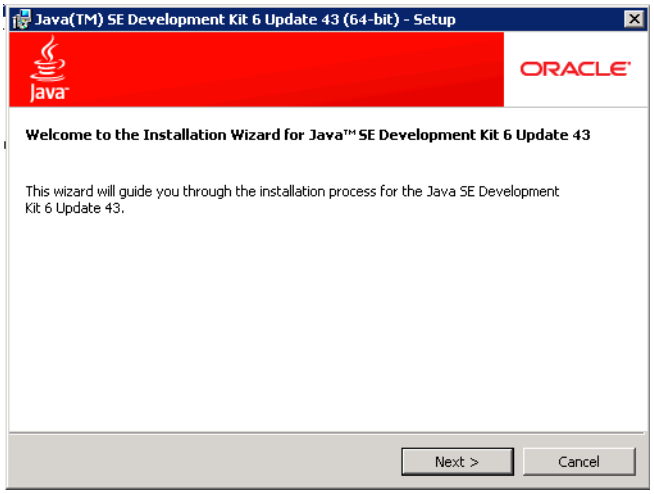
To download and install the JDK:

1. To download Sun Java 6:
   (a) Go to *http://www.oracle.com/technetwork/java/javase/downloads/index.html*.
   (b) On the **Downloads** tab, click **Previous Releases**.
   (c) On the **Oracle Java Archive** page, click **Java SE 6**.
   (d) On the **Java SE 6 Downloads** page, click the link for the most recent version of the Java SE Development Kit.

   The page scrolls to the list of downloads for that version of the JDK.

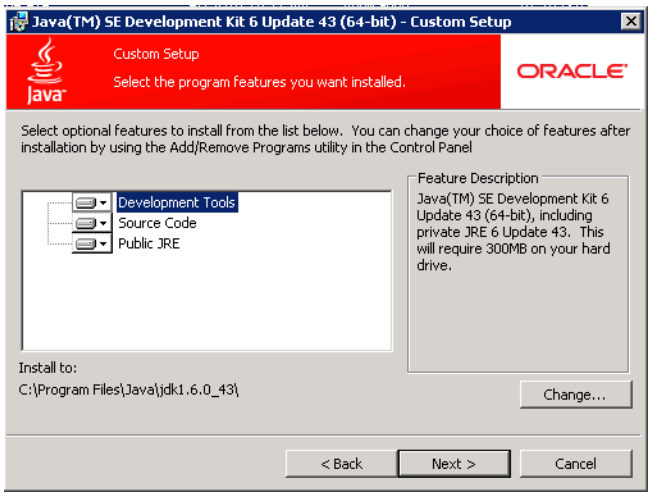
   (e) At the top of the list, click the **Accept License Agreement** radio button.
   (f) In the list, click the **Download** link for Windows x64 or Linux x64.
   (g) Save the file (`jdk-6u43-windows-x64.exe` or `jdk-6u43-linux-x64.bin`) to your machine.
2. To install the JDK on Windows:
   (a) Double-click the JDK installer file.

   The welcome page of the installation wizard is displayed.

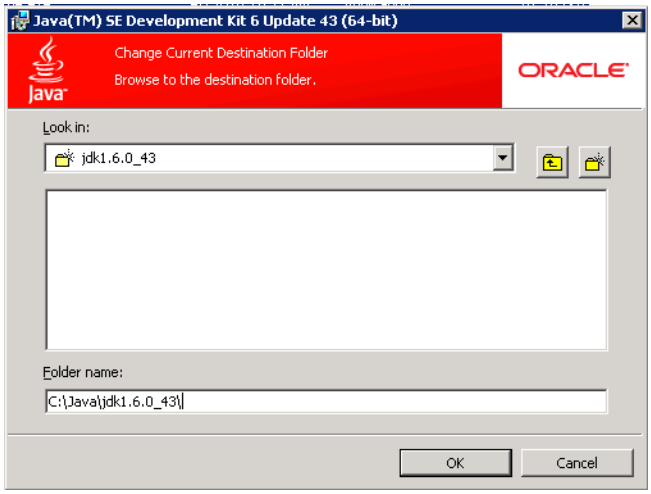



   (b) On the welcome page, click **Next**.

(c) Because the default installation location (under `Program Files`) has a space, you must change it. On the **Custom Setup** page, click the **Change** button.

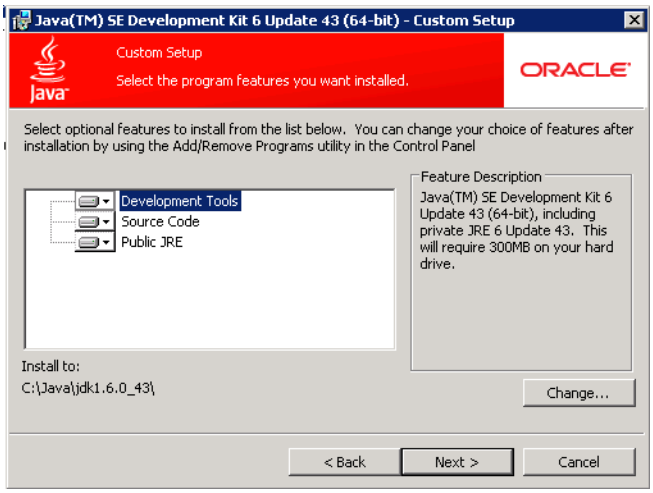



(d) In the **Folder name** field, remove the `Program Files` directory, to change the location to: `C:\Java\jdk1.6.0_43\`




(e) Click **OK**.

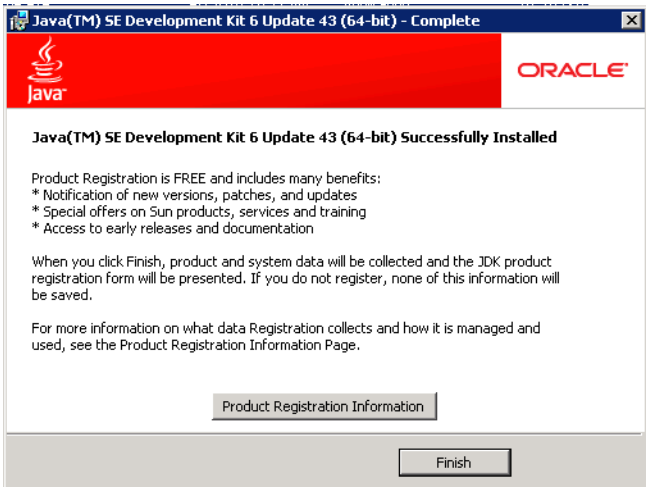
(f) On the **Custom Setup** page, verify that the installation location has been updated, then click **Next**.




(g) The **Destination Folder** page sets the location of the public JRE. Because this JRE is not used by Studio, you can leave the default directory. Click **Next**.

The installation process begins.

(h) When the installation is completed, click **Finish**.




3. To install the JDK on Linux:

(a) Select a location for installation. For example, `$HOME/Oracle/Endeca`.

(b) Place the JDK .bin file (`jdk-6u43-linux-x64.bin`) into `$HOME/Oracle/Endeca`, and ensure that it has execute permissions,

(c) Execute the file.

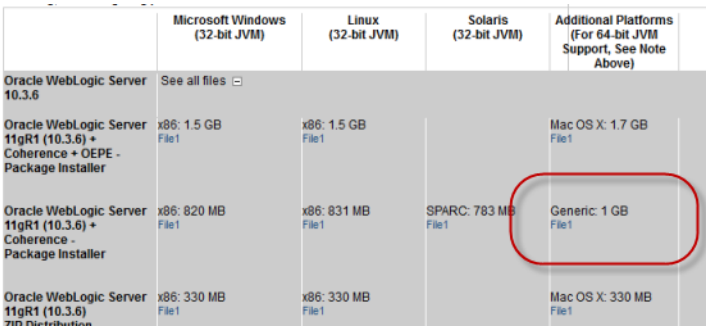
The JDK is extracted into the directory. For example, `$HOME/Oracle/Endeca/jdk1.6.0_43`.

# Downloading and installing WebLogic Server

If you are not installing Studio on the same server as Endeca Server, then you need to download and install WebLogic Server.

To download and install WebLogic Server:

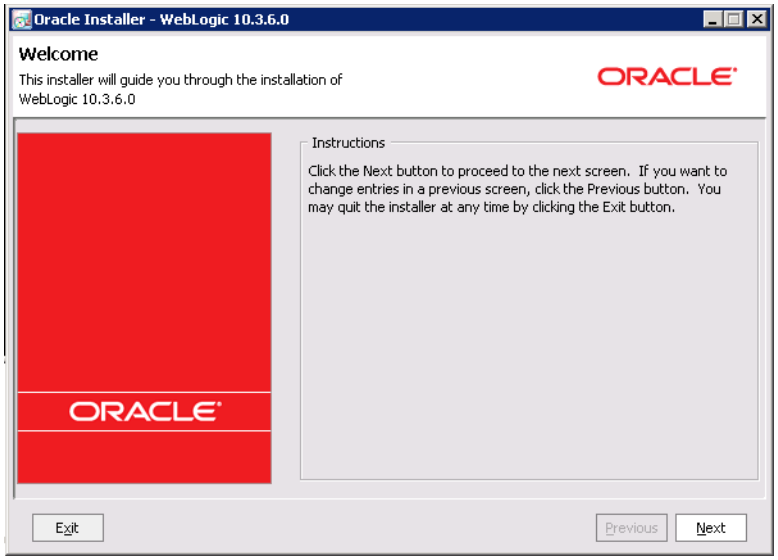
1. To download the WebLogic Server installation package:
   (a) Go to *http://www.oracle.com/technetwork/middleware/weblogic/downloads/wls-for-dev-1703574.html*.
   (b) Under **Oracle WebLogic Server Previous Releases**, expand the list for Oracle WebLogic Server 10.3.6.
   (c) Download the **Generic** version of the installer (`wls1036_generic.jar`).




2. From a command prompt, navigate to the directory where you placed `wls1036_generic.jar`.
3. To launch the installer, issue the following command:

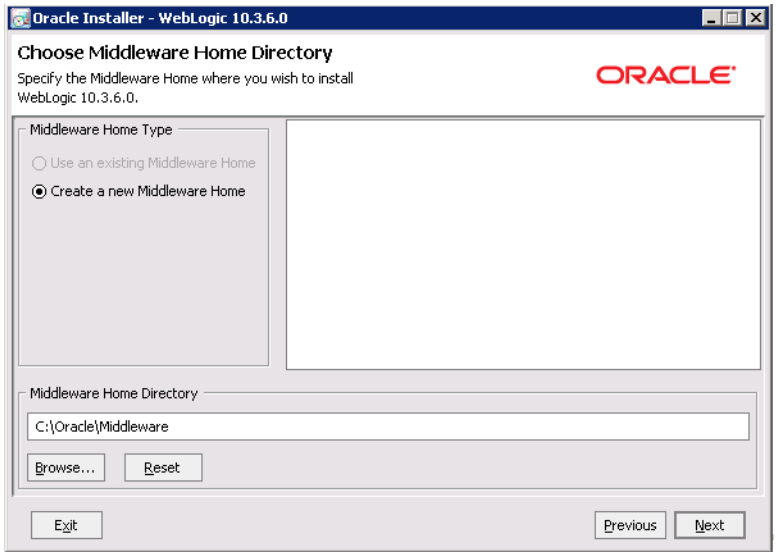
```
java -jar wls1036_generic.jar
```

   The **Welcome** page of the installation wizard is displayed.

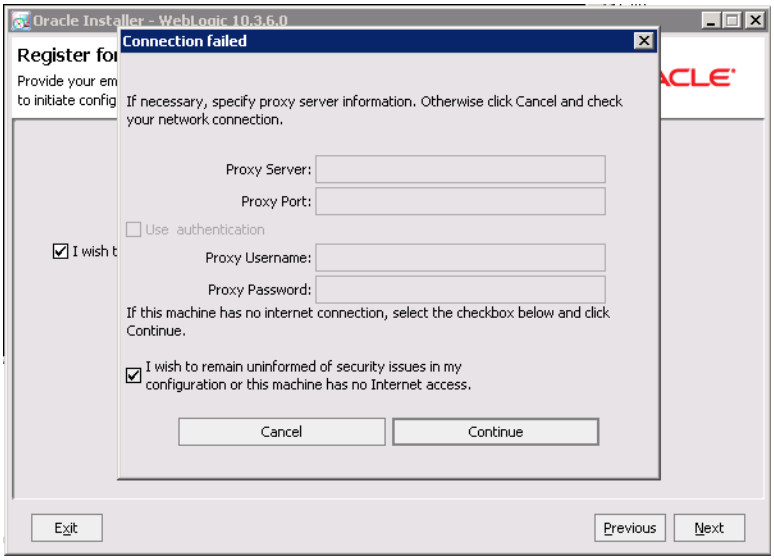



4. On the **Welcome** page, click **Next**.

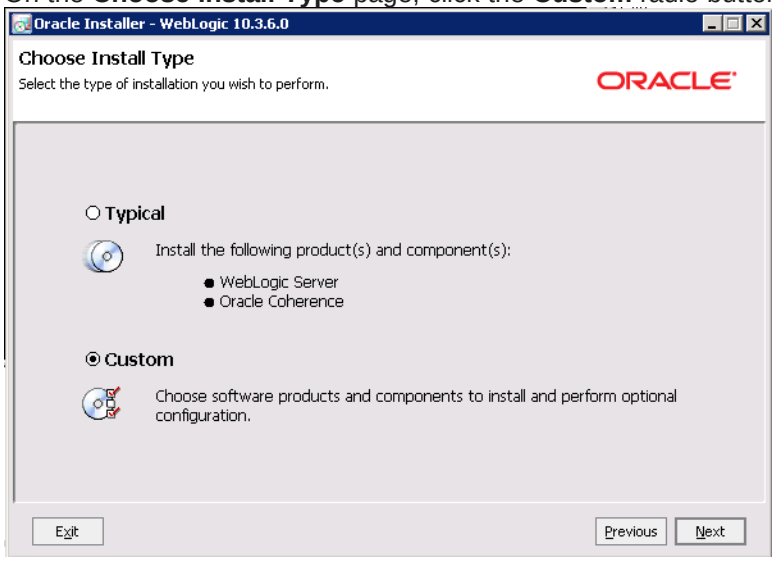
5. On the **Choose Middleware Home Directory** page:
   (a) Click the **Create a new Middleware Home** radio button.
   (b) Accept the default location for the Middleware Home Directory.
   (c) Click **Next**.



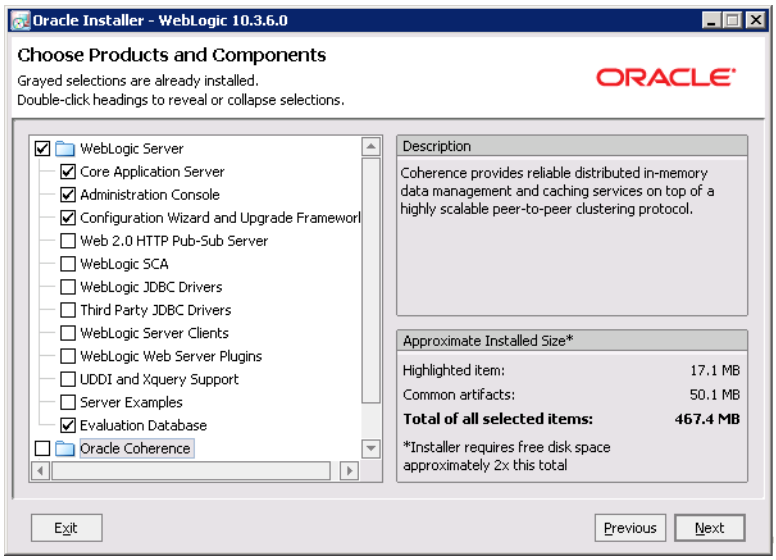

6. To bypass the **Register for Security Updates** page:
   (a) Click **Next** without entering an email address.
   (b) On the **Email Address Not Specified** dialog, click **Yes**.
   (c) On the **Are You Sure?** dialog, click **Yes**.
   (d) On the **Connection failed** dialog, check the **I wish to remain uninformed of security issues in my configuration or the machine has no Internet access** checkbox, then click **Continue**.



7. On the **Choose Install Type** page, click the **Custom** radio button, then click **Next**.




8. On the **Choose Products and Components** page, check the following products, then click **Next**:
   - **Core Application Server**
   - **Administration Console**
   - **Configuration Wizard and Upgrade Framework**
   - **Evaluation Database**

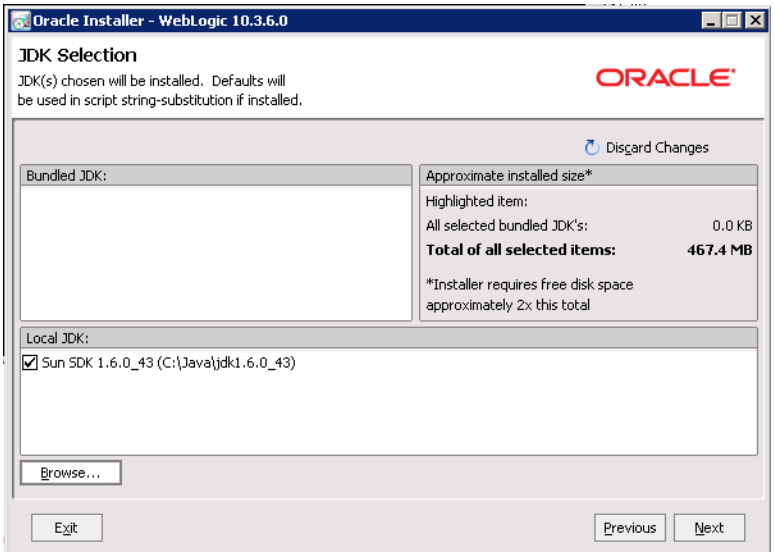



9. On the **JDK Selection** page, verify that the JDK you installed is selected. If it is, click **Next**. If that version of the JDK is not selected:

   (a) Under **Local JDK**, click the **Browse** button.

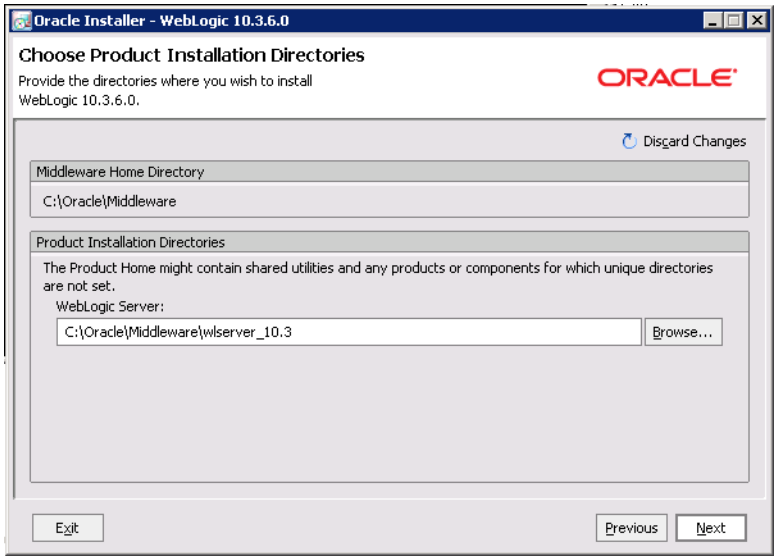
(b) Browse to and select the directory that contains the JDK (`C:\Java\jdk1.6.0_43\` or `$HOME/Oracle/Endeca/jdk1.6.0_43`)

The **Local JDK** section is updated to display the selected JDK with a checked box next to it.

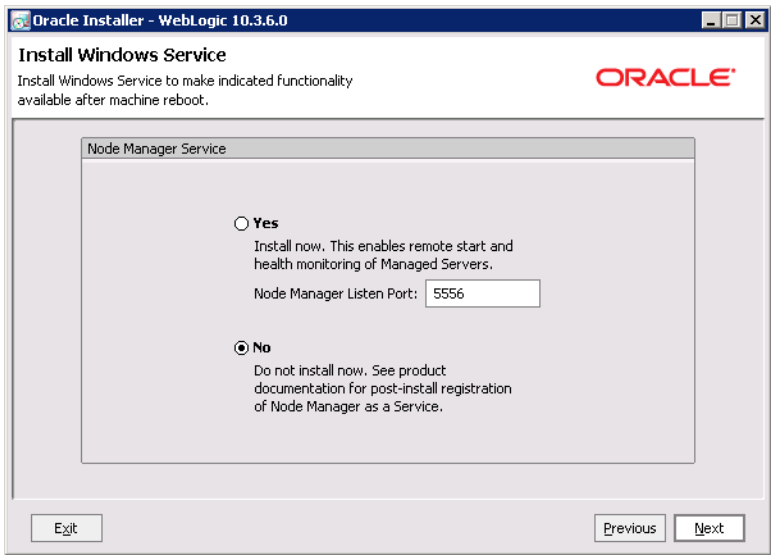



(c) Click **Next**.

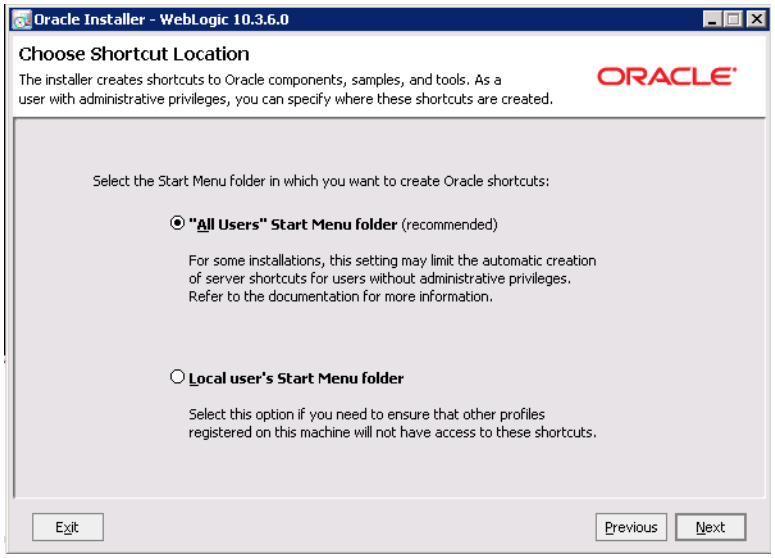
10. On the **Choose Product Installation Directories** page, accept the default installation directory, then click **Next**.



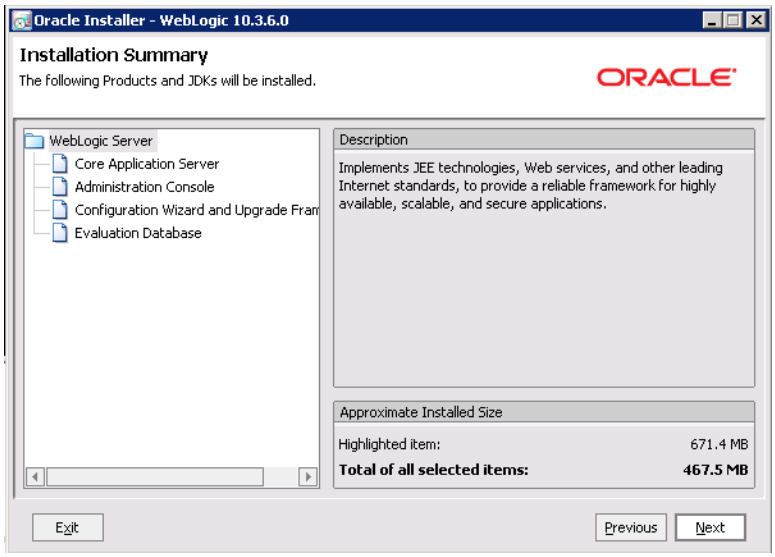
11. If the **Install Windows Service** page is displayed, click the **No** radio button, then click **Next**.



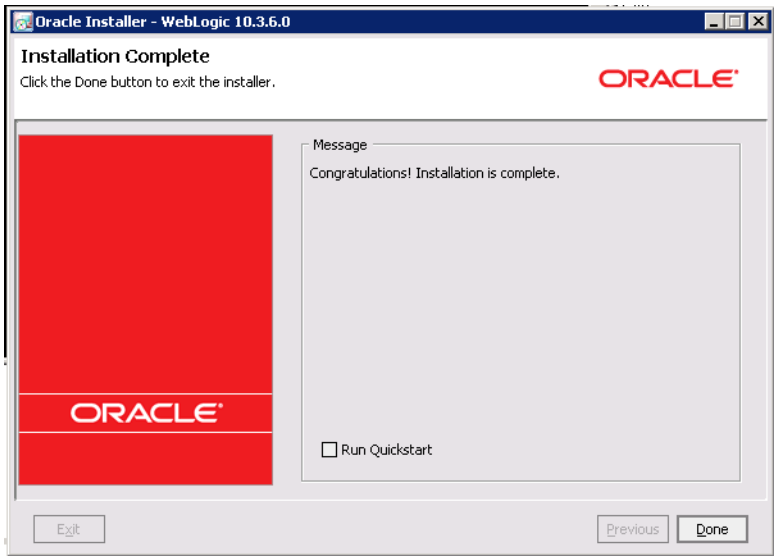

12. If the **Choose Shortcut Location** page is displayed, click a radio button to select the location of the Windows shortcut, then click **Next**.



13. On the **Installation Summary** page, verify that you are installing the correct products, then click **Next**.




14. On the **Installation Complete** page, uncheck the **Run Quickstart** checkbox, then click **Done**.




For Windows, an **Oracle WebLogic** shortcut is created in the **Start Menu**.

# Creating and configuring a WebLogic domain for Studio

Studio must be installed in its own WebLogic domain. To avoid any conflicts, the domain cannot contain any other applications. Even if you are installing Studio on the same instance of WebLogic Server as Endeca Server, you must create a separate domain for Studio.

To create and configure the WebLogic domain, and verify that the domain is not already in use:

1. From a command prompt, change to the directory that contains the Configuration Wizard start-up program.

   For example, for Windows:

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

   For Linux:

   ```
   $MW_HOME/wlserver_10.3/common/bin
   ```

2. To start the Configuration Wizard , run the following command:

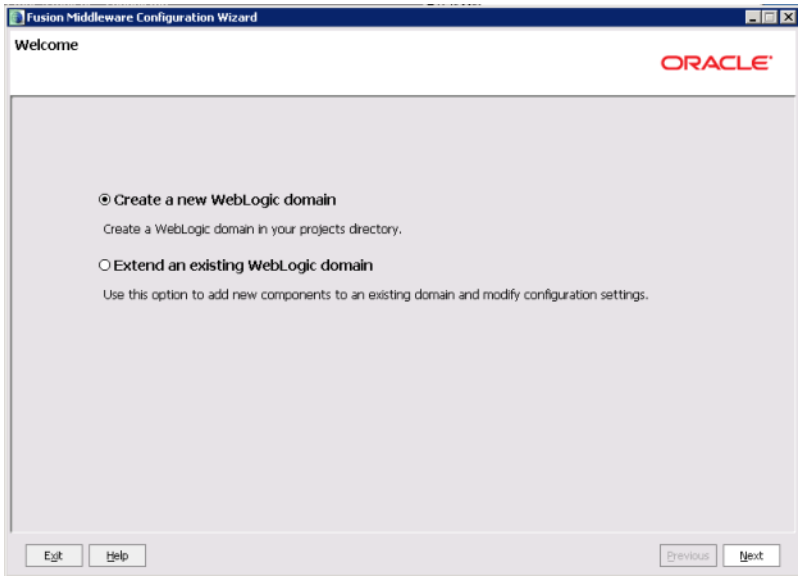
   For Windows:

   ```
   config.cmd
   ```

   For Linux:

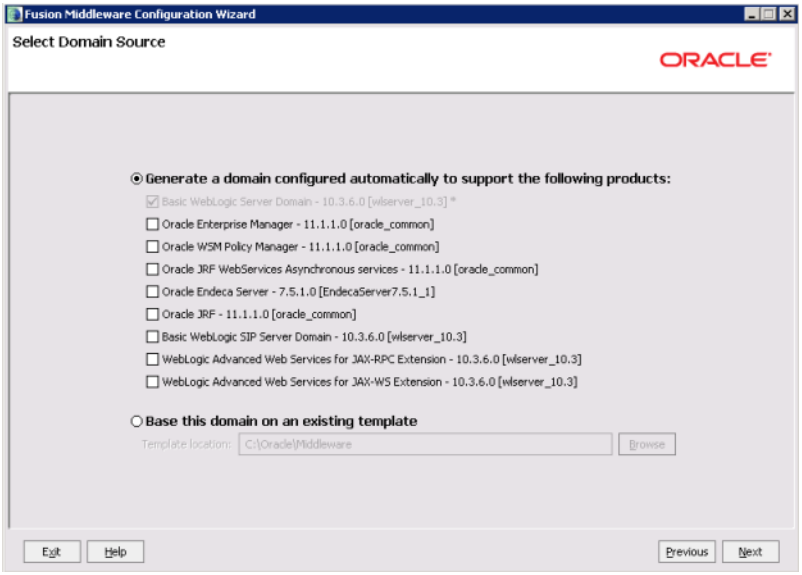
   ```
   ./config.sh
   ```

   The Configuration Wizard **Welcome** page is displayed.

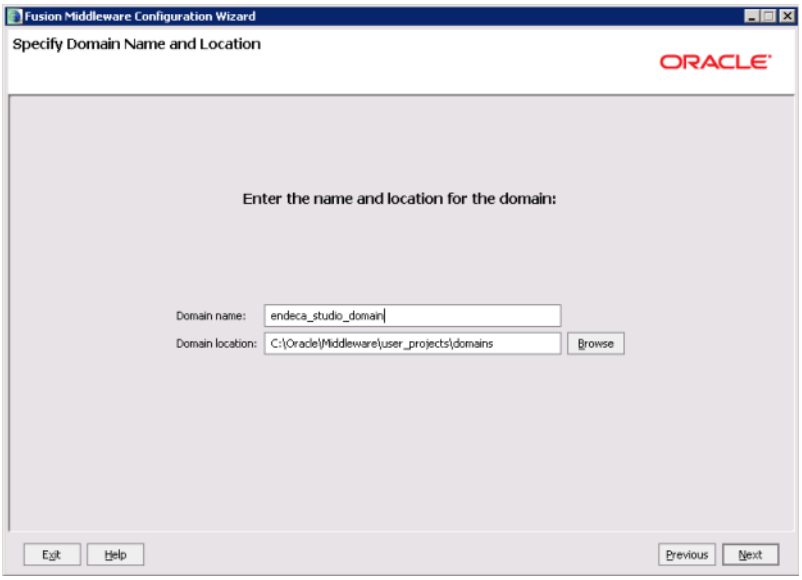
   
   

3. On the **Welcome** page, click the **Create a new WebLogic domain** radio button, then click **Next**.

4. On the **Select Domain Source** page, the **Basic WebLogic Server Domain** checkbox is checked by default. Click **Next**.




5. On the **Specify Domain Name and Location** page:
   (a) Set the domain name to `endeca_studio_domain`
   (b) Keep the default domain location.
   (c) Click **Next**.



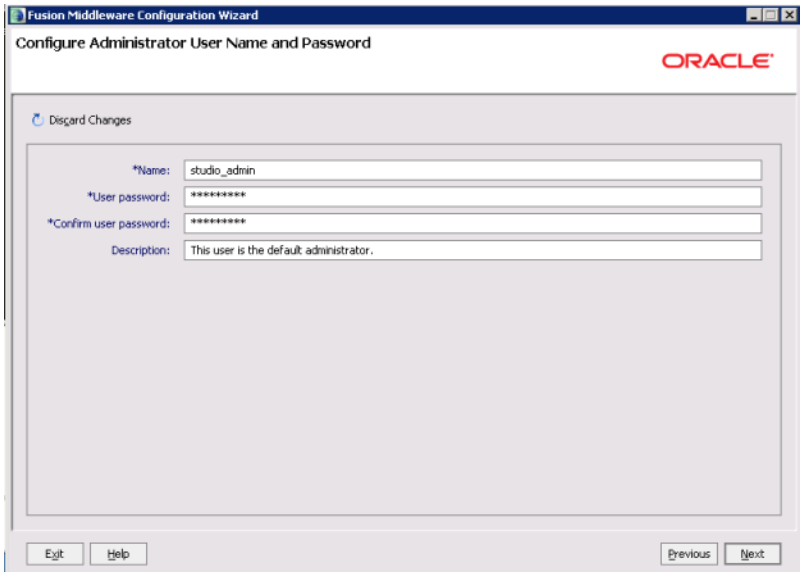

6. On the **Configure Administrator User Name and Password** page, to create the administrator credentials for starting the Studio domain:
   (a) In the **Name** field, type the name for the administrator user.
   (b) In the **User password** field, type the user password.

   The password must be at least eight characters long.

(c) In the **Confirm user password** field, retype the password.

(d) Optionally, in the **Description** field, type a description for the user account.

(e) Click **Next**.



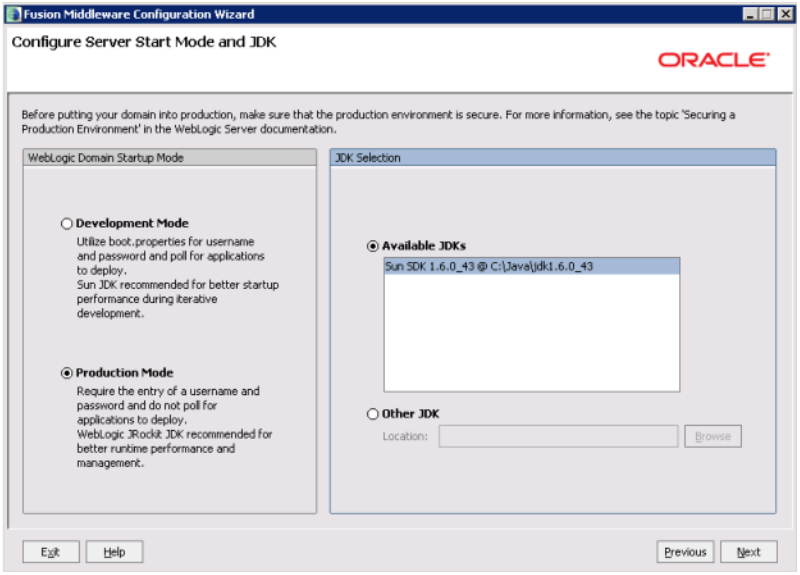

7. On the **Configure Server Start Mode and JDK** page:

(a) Under **WebLogic Domain Startup Mode**, click the **Production Mode** radio button.

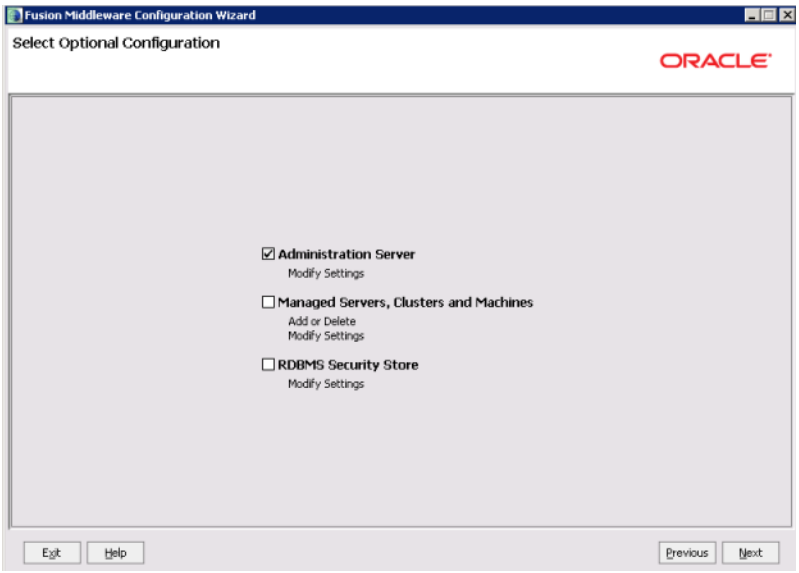
(b) Under **JDK Selection**, click the **Available JDKs** radio button.

(c) In the JDK list, click the Sun Java JDK that you installed earlier. If the JDK is not displayed, click the **Other JDK** button, and then browse to it.

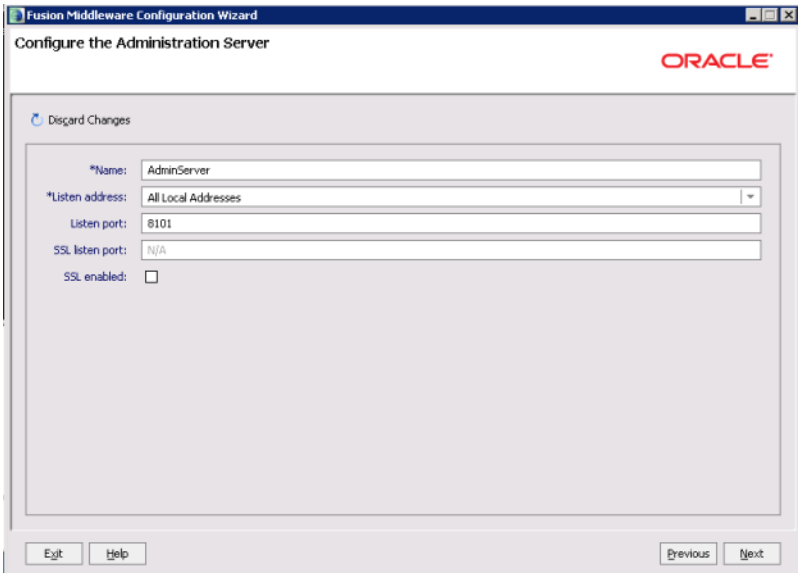
(d) Click **Next**.



8. On the **Select Optional Configuration** page, check the **Administration Server** checkbox, then click **Next**.

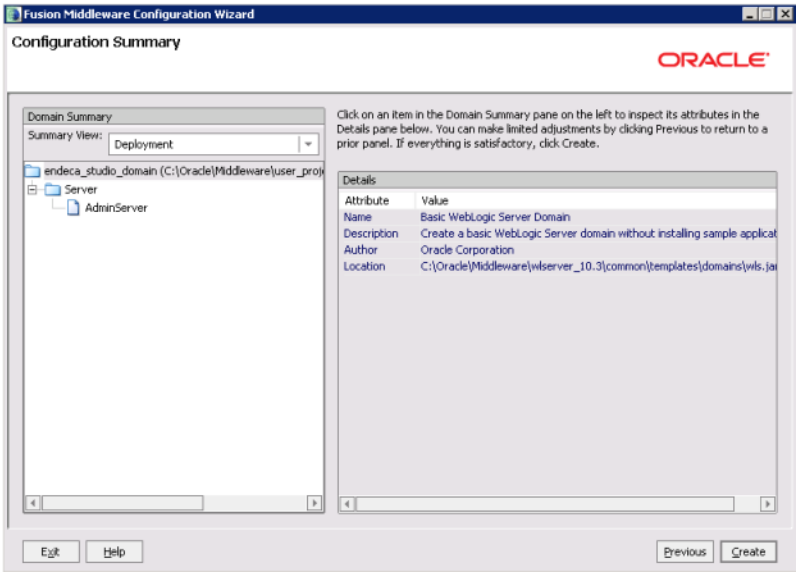



9. On the **Configure the Administration Server** page:
   (a) Use the default values for the **Name** and **Listen address** fields.
   (b) In the **Listen port** field, type `8101`.

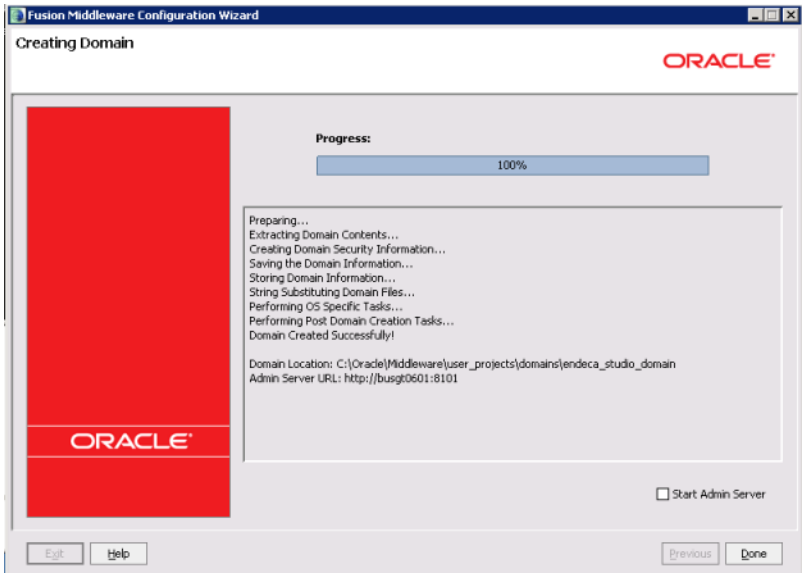



   (c) If you want to enable SSL, then check the **SSL enabled** checkbox, and in the **SSL listen port** field, type `8102`.
   (d) Click **Next**.

10. On the **Configuration Summary** page, to create the domain, click **Create**.




On the **Creating Domain** page, the message "Domain Created Successfully!" signifies the end of the domain creation process.




11. To exit the Configuration Wizard, click **Done**.
12. Update the `setDomainEnv` script file (`setDomainEnv.cmd` for Windows and `setDomainEnv.sh` for Linux).

    The file is located in the `bin` subdirectory of the domain directory (`<MiddlewareHomeDirectory>/user_projects/domains/endeca_studio_domain/bin/`) .

    (a) Add the `JAVA_OPTIONS` argument close to the top of the file.

    For `setDomainEnv.cmd` (Windows):

```
set JAVA_OPTIONS=-DUseSunHttpHandler=true -Djavax.xml.transform.TransformerFactory
=com.sun.org.apache.xalan.internal.xsltc.trax.TransformerFactoryImpl
-Djavax.xml.parsers.DocumentBuilderFactory
=com.sun.org.apache.xerces.internal.jaxp.DocumentBuilderFactoryImpl
-Djavax.xml.parsers.SAXParserFactory
=com.sun.org.apache.xerces.internal.jaxp.SAXParserFactoryImpl
```

For `setDomainEnv.sh` (Linux):

```
JAVA_OPTIONS="-DUseSunHttpHandler=true -Djavax.xml.transform.TransformerFactory
=com.sun.org.apache.xalan.internal.xsltc.trax.TransformerFactoryImpl
-Djavax.xml.parsers.DocumentBuilderFactory
=com.sun.org.apache.xerces.internal.jaxp.DocumentBuilderFactoryImpl
-Djavax.xml.parsers.SAXParserFactory
=com.sun.org.apache.xerces.internal.jaxp.SAXParserFactoryImpl"
export JAVA_OPTIONS
```

Note that if you're copying and pasting out of this guide, make sure to remove any line breaks after pasting the text into the file.

(b) Update all of the Java perm size arguments in the file to replace all occurrences of `-XX:MaxPermSize=128m` and `-XX:MaxPermSize=256m` with `-XX:MaxPermSize=512m`.

(c) Update all of the memory arguments in the file to replace all occurrences of `-Xmx512m` with `-Xmx1024m`.

13. To verify that there are no conflicting applications on the domain's root context:

(a) Start the domain.

In `<MiddlwareHomeDirectory>\user_projects\domains\endeca_studio_domain`, run `startWebLogic.cmd` (for Windows) or `startWebLogic.sh` (for Linux).

(b) When prompted, enter the domain user name and password.

(c) After the domain is started, in a Web browser, go to `http://<serverName>:<portNumber>`.

If you get a "page not found" (404) error, then there are no applications on the domain's root context, and you can continue the Studio installation process.

If you see the login page for a previously deployed Studio application, then you should un-deploy this application.

If you see another application, then you should log in to the WebLogic Console, and then either remove the application or untarget it from the WebLogic Server instance.

For example, one common conflicting application is Oracle Enterprise Manager, which deploys an application to `/em`, but also installs the FMW Welcome Page Application to the root context.

After you create the WebLogic domain for Studio, then you can deploy the Studio .ear file. See *Deploying Studio to the WebLogic domain on page 46*.

# Deploying Studio to the WebLogic domain

To install Studio, you use the WebLogic Server **Administration Console** to deploy `endeca-portal-weblogic-3.0.x.ear` into the WebLogic domain.

When you deploy using these instructions, the following default locations are used:

| Location | Default Value |
|---|---|
| Context path | Studio is deployed to the domain's root context.<br>When you created the domain, you should already have verified that there are no conflicts on the root context.<br>To a non-root context, follow the instructions in *Changing to a non-root context for Studio on WebLogic Server on page 49*. |
| Liferay Home directory | The Liferay Home directory is `<MiddlewareHomeDirectory>/user_projects/domains`.<br>Before installing, make sure that this directory does not already contain the following subdirectories:<br>• `/data/endeca-data-sources`<br>• `deploy`<br>• `weblogic-deploy`<br>If these directories exist, then it means there is an instance of Studio installed. For the installation to work correctly, these existing directories must be removed. Before moving these directories, also make sure that the other instance has been removed.<br>You can, if needed, change the location of the Liferay Home directory. If you want to use a different directory for Liferay Home, then you need to follow the instructions in *Changing the Liferay Home directory for Studio on WebLogic Server on page 50*. |

To deploy `endeca-portal-weblogic-3.0.x.ear` into a production environment:

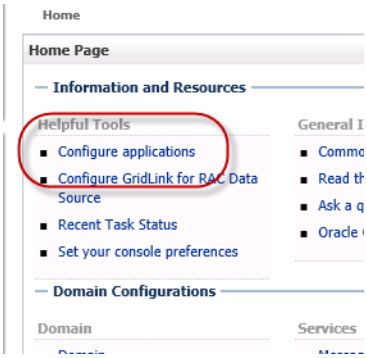
1. Set up the Liferay Home directory (the default is `<MiddlewareHomeDirectory>/user_projects/domains`):
   (a) Extract the file `portal-ext.properties` from the top level of `EID_3.0.x_Studio_portal_weblogic.zip`.
   (b) Place `portal-ext.properties` in the Liferay Home directory.
   (c) In the Liferay Home directory, create the following subdirectories:
   - `/data/endeca-data-sources`
   - `deploy`
   - `weblogic-deploy`
2. If WebLogic Server isn't already started, then to start it, run the WebLogic Server startup script.

   The startup script is in the domain directory.

For Windows, the script name is `startWebLogic.cmd`.

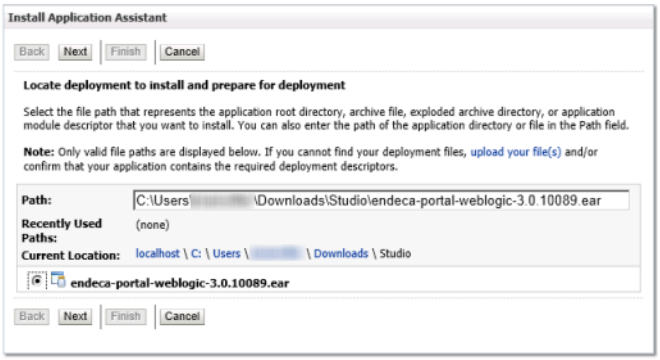
For Linux, the script name is `startWebLogic.sh`.

3. Extract `endeca-portal-weblogic-3.0.x.ear` from `EID_3.0.x_Studio_portal_weblogic.zip`.
4. Use the WebLogic Server **Administration Console** to deploy `endeca-portal-weblogic-3.0.x.ear` into the domain.
   (a) Go to the **Administration Console** at `<hostname>:8101/console`.
   (b) Under **Helpful Tools**, click **Configure applications**.



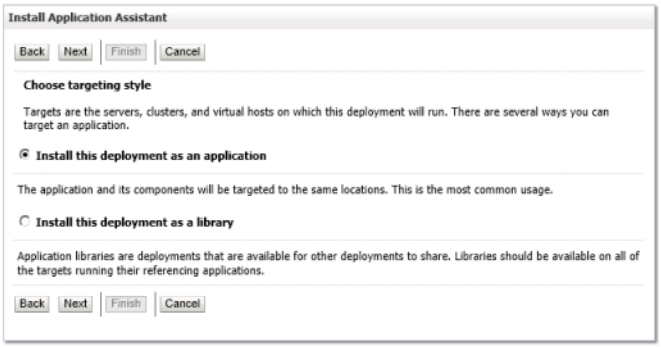

   (c) If necessary, click **Lock & Edit** at the top left of the page.
   (d) Click the **Install** button.


   (e) Use the file browser to find the `endeca-portal-weblogic-3.0.x.ear` file, then click the radio button to the left of it.

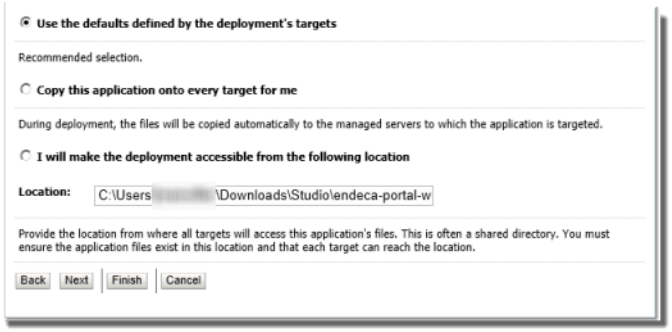



   (f) Click **Next**.

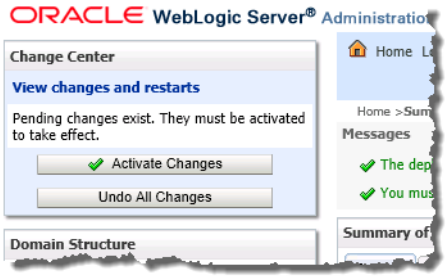
(g) On the next page, under **Choose targeting style**, make sure the **Install this deployment as an application** radio button is selected, then click **Next**.




(h) On the next page, accept the default selections, then click **Finish**.



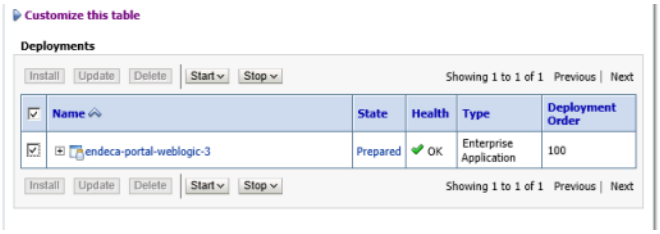

(i) If necessary, click **Activate Changes**.




The deployment may take several minutes. Make sure you do not try to log in to Studio before the deployment is complete.

5. When the deployment process is completed, the deployment status is Prepared. To start the deployment:

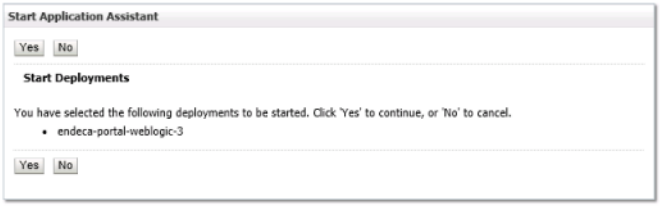
(a) In the **Deployments** list, check the checkbox for the Studio deployment.



(b) Click the **Start** button.

(c) From the drop-down list, select **Servicing all requests**.

(d) Under **Start Deployments**, click **Yes**.




6. To verify that Studio has deployed successfully, in a Web browser, go to `http://<hostName>:<portNumber>`, where <*hostName*> is the name of the server, and <*portNumber*> is the port number.

   For a non-SSL implementation, the port number is 8101.

   You should see the Studio login page.

# Changing to a non-root context for Studio on WebLogic Server

By default, Studio uses the root context path, but you can change to a non-root context.

Note that any change to the context root only affects the Studio application. Components and other hooks still use their default context paths.

To use a non-root context:

1. Before deploying `endeca-portal-weblogic-3.0.x.ear`, you must update the `context-root` property in `application.xml`. To do this:

   (a) Use a .zip utility to open `EID_3.0.x_Studio_portal_weblogic.zip`.

   (b) From `EID_3.0.x_Studio_portal_weblogic.zip`, extract `endeca-portal-weblogic-3.0.x.ear` to a temporary directory.

   (c) In the extracted directory, open the file `META-INF/application.xml`

   (d) In the file, find the following XML snippet:

```
<module>
  <web>
    <web-uri>endeca-portal.war</web-uri>
    <context-root>/</context-root>
  </web>
</module>
```

   (e) Update the value of the `context-root` element to your chosen context root path.

   The value should have a trailing slash. For example:

```
<module>
  <web>
    <web-uri>endeca-portal.war</web-uri>
    <context-root>/mycompany/portal/</context-root>
  </web>
</module>
```

   (f) Repack the extracted directory into `endeca-portal-weblogic-3.0.x.ear`.

(g) Use this updated version of `endeca-portal-weblogic-3.0.x.ear` to replace the existing file in `EID_3.0.x_Studio_portal_weblogic.zip`.

2. Start the deployment process. The first step in the deployment process is to place `portal-ext.properties` in the Liferay Home directory. If you are using a non-root context, then you must also update the file to use the correct context path:
   (a) Extract the file `portal-ext.properties` from the top level of `EID_3.0.x_Studio_portal_weblogic.zip`.
   (b) Place `portal-ext.properties` in the Liferay Home directory.
   (c) Open the file.
   (d) Find the `portal.ctx` at the beginning of the file.
   (e) Set the value of `portal.ctx` to be the same value you used for the `context-root` property in `application.xml`.

   Unlike the `context-root` property, when setting the `portal.ctx` value, do NOT include a trailing slash. For example, use this value:

   ```
   portal.ctx=/mycompany/portal
   ```

   Do not use this value:

   ```
   portal.ctx=/mycompany/portal/
   ```

3. After setting `portal.ctx`, you can then continue with step 2c of the Studio deployment process.

   See *Deploying Studio to the WebLogic domain on page 46*.

# Changing the Liferay Home directory for Studio on WebLogic Server

For a Studio installation on WebLogic Server, the default location of the Liferay Home directory is `<MiddlewareHomeDirectory>/user_projects/domains`, where `<MiddlewareHomeDirectory>` is the directory where you installed WebLogic Server. You can, if needed, use a different directory as the Liferay Home directory.

To create and configure a different location for the Liferay Home directory:

1. Create the directory you want to use as the Liferay Home directory.
2. Next, you need to update the version of `portal-ext.properties` located in the `web-inf/classes` directory of the `endeca-portal-3.0.x.war` file.

   This version of `portal-ext.properties` is only used to provide a pointer to the Liferay Home directory.

   To update this file:

   (a) Use a .zip utility to open `EID_3.0.x_Studio_portal_weblogic.zip`.
   (b) From `EID_3.0.x_Studio_portal_weblogic.zip`, copy out the file `endeca-portal-weblogic-3.0.x.ear`.
   (c) From `endeca-portal-weblogic-3.0.x.ear`, extract `endeca-portal-3.0.x.war` to a temporary directory.
   (d) In the `web-inf/classes` subdirectory, open `portal-ext.properties`.

(e) Add the `liferay.home` parameter to the top of the file.

For example:

```
liferay.home=/localdisk/liferay
```

Note that for Windows, you must escape the back slashes. For example:

```
liferay.home=C:\\Oracle\\Middleware\\user_projects\\domains
```

(f) Repackage `endeca-portal-3.0.x.war`.

(g) Use the updated version of `endeca-portal-3.0.x.war` to replace the version in `endeca-portal-weblogic-3.0.x.ear`.

(h) Use the updated version of `endeca-portal-weblogic-3.0.x.ear` to replace the version in `EID_3.0.x_Studio_portal_weblogic.zip`.

3. Start the deployment process. The first step in the deployment process is to place `portal-ext.properties` in the Liferay Home directory. If you have changed the Liferay Home directory from the default, then you must add the `liferay.home` parameter to the top of the file:

(a) Extract the file `portal-ext.properties` from the top level of `EID_3.0.x_Studio_portal_weblogic.zip`.

(b) Place `portal-ext.properties` in the Liferay Home directory you created.

(c) Open the file.

(d) Add the `liferay.home` parameter to the top of the file.

For example:

```
liferay.home=/localdisk/liferay
```

Remember that for Windows, you must escape the back slashes. For example:

```
liferay.home=C:\\Oracle\\Middleware\\user_projects\\domains
```

4. After adding the `liferay.home` property, you can then continue with step 2c of the Studio deployment process.

See *Deploying Studio to the WebLogic domain on page 46*.

Chapter 10

# Troubleshooting Known Studio Issues

Here are some troubleshooting tips for some general issues that can occur in a Studio instance.

*Changing the JavaScript timeout value on Internet Explorer 8*

*Increasing the Tomcat connector keepAliveTimeout for Internet Explorer 8*

*Increasing or disabling the WebLogic Activation Timeout*

## Changing the JavaScript timeout value on Internet Explorer 8

Internet Explorer 8 keeps track of the number of JavaScript lines executed. After a fixed value, the browser issues an error message, prompting the user to decide whether he or she would like to continuing running the script. Because Studio is a rich Internet application that leverages JavaScript heavily in all components, it can trigger this error during normal usage.

Microsoft describes this issue in Knowledge Base Article 175500 and specifies a fix. More detail can be found in the Knowledge Base Article (*http://support.microsoft.com/kb/175500*). The following is a condensed version of Microsoft's fix for the Internet Explorer 8 issue.

**Important:** If you intend to run Studio on Internet Explorer 8, Endeca strongly recommends that you apply the fix outlined here.

To change the script timeout value:

1. Using a registry editor such as `Regedt32.exe`, open this key:

   ```
   HKEY_CURRENT_USER\Software\Microsoft\Internet Explorer\Styles
   ```

   **Note:** If the `Styles` key is not present, create a new key that is called `Styles`.

2. Create a new `DWORD` value called "MaxScriptStatements" under this key, and set the value to the desired number of script statements. You will have to try different values for your application environment, but the suggested starting point is a `DWORD` value of 0x1CFFFFF.

   **Note:** You can turn off this Internet Explorer 8 feature using a `DWORD` value of 0xFFFFFFFF.

# Increasing the Tomcat connector keepAliveTimeout for Internet Explorer 8

If your users are using Internet Explorer 8 on a slower network, and experience intermittent timing out of requests to the Endeca Server, you may need to increase the value of `keepAliveTimeout` for Tomcat's HTTP 1.1 connector.

Note that increasing this setting can introduce the risk of lower throughput, or require configuration of additional server connection threads.

# Increasing or disabling the WebLogic Activation Timeout

When deploying the WebLogic .ear file, if the change activation fails with a timeout error, you can increase or disable the **Activation Timeout**.

To do this:

1. On the **Administration Console**, to return to the **Home Page**, click **Home**.
2. On the **Home Page**, under **Helpful Tools**, click **Set your console preferences**.
3. Under **Change Center Preferences**, find the **Activation Timeout** field.
4. Either increase the timeout value or, to disable the timeout, set the value to `-1`.

Chapter 11

# Getting Started with Studio

After installing Studio, use the following steps to launch and configure Studio and begin to work with it.

*Starting and logging in to Studio*

*Displaying the Studio Control Panel*

*Updating the default data source to point to your Endeca Server*

## Starting and logging in to Studio

After you complete the Studio installation, you can start and log in to the application.

To start Studio:

1. Start your application server.
2. In your Web browser, go to the portal.
3. Log in using the default login and password:

| Field | Value |
|---|---|
| **Login:** | `admin@oracle.com` |
| **Password:** | `Welcome123` |

   You are immediately prompted to change the password. The new password must contain:

   - At least 6 characters
   - At least one non-alphabetic character

## Displaying the Studio Control Panel

The **Control Panel** provides access to Studio administrative functions, including functions to configure Studio settings, manage users, and monitor performance.

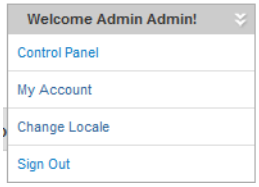
For information on the using the **Control Panel** to create data sources for and manage Studio applications, see the *Studio User's Guide*.

For information on using the **Control Panel** to administer and monitor Studio, see the *Studio Administration and Customization Guide*.

To display the **Control Panel**:

1. Click the Studio menu in the upper-right corner of the page.

   The menu is labeled "Welcome <*user name*>!"

2. From the Studio menu, select **Control Panel**.

# Updating the default data source to point to your Endeca Server

When you first install Studio, Studio creates a placeholder `default` data source.

If you want to be able to use this `default` data source for Studio applications, then you need to update it to point to the correct server, port, and data domain. If the Endeca Server is secured using SSL, then you also need to install the Endeca Server certificate files and update the certificate file and password settings.

To update the `default` data source:

1. If you are using a secured Endeca Server:
   (a) Stop Studio.
   (b) From the Endeca Server `$DOMAIN_HOME/config/ssl` directory, copy the following files:
      - `endecaServerClientCert.ks`
      - `endecaServerTrustStore.ks`
   (c) Place the files into the `endeca-data-sources` directory.

      If Studio was installed using the Tomcat bundle, the directory is `endeca-portal\data\endeca-data-sources`.

      If Studio was installed on a standalone instance of Tomcat, without using the bundle, then you will need to create the `endeca-portal\data\endeca-data-sources` directory.

      If Studio was installed on WebLogic, then the directory is the `data\endeca-data-sources` directory in the Liferay Home directory.
2. Start and log in to Studio.
3. From the Studio menu, select **Control Panel**.
4. In the **Control Panel** menu, in the **Information Discovery** section, click **Data Sources**.
5. On the **Data Sources** page, click the **Edit** button for the `default` data source.
6. In the data source definition, edit the connection and certificate information to reflect your Endeca Server.

   For details on configuring data sources, see the *Studio User's Guide*.

   For example:

```
{
 "server":"server01.lab.acme.com",
 "port":"7002",
 "dataDomainName":"acmeDB",
 "sslConfig": {
   "caFile":"endecaServerTrustStore.ks",
   "caPassword":"************",
   "certFile":"endecaServerClientCert.ks",
   "certPassword":"************"
 }
}
```

7. Click **Save**.

Chapter 12

# Changing the Database Used to Store Studio Application Data

For production instances, you need to switch from the default Hypersonic (HSQL) database.

*About using a different database*

*Overview of the process for switching to a different database*

## About using a different database

Studio uses a relational database to store configuration and state, such as component preferences, user permissions, and system settings.

By default, Studio uses Hypersonic (HSQL), which is an embedded database running inside the Java virtual machine. HSQL is useful for standing up a Studio instance very quickly, but must NOT be used in production due to performance issues and its inability to support clustered instances.

Keep the following details in mind:

- Studio ships with a `portal-ext.properties` file (in the portal distribution's root directory). You can modify this file instead of creating a new one.
- Studio has been tested on MySQL and Oracle 11g. Other databases are expected to work but have not been explicitly tested.
- When migrating to a new database, make sure that the current data does not conflict with the new database's schema. For example, different databases may have different constraints for column sizes.

## Overview of the process for switching to a different database

Here is a high-level overview of the steps involved in switching from the default Hypersonic database to the production RDBMS of your choice.

The details vary from database to database.

To switch to a different database:

1. Install and verify that your database is working.
2. Create a new empty database or schema for the application.

   When you create the new database, make sure to use UTF-8 encoding.
3. Create a database user for the application.

4. Grant that user access to the appropriate database/schema, with privileges to create tables, alter schemas, and so on in that database.

   Ensure that the user has remote access from the application servers.

5. Stop Studio if it is running.

6. Next, edit the JDBC section of the `portal-ext.properties` file to change the database connection from Hypersonic to your database.

   (a) The default version of the file has the Hypersonic settings enabled. Comment out these settings.

   ```
   # Hypersonic
   #
   jdbc.default.driverClassName=org.hsqldb.jdbcDriver
   jdbc.default.url=jdbc:hsqldb:${liferay.home}/data/hsql/lportal
   jdbc.default.username=sa
   jdbc.default.password=
   ```

   (b) To configure the connection to your database, the recommended option is to uncomment and configure the `jdbc.default.jndi.name` parameter.

   ```
   # Set the JNDI name to lookup the JDBC data source. If none is set,
   # then the portal will attempt to create the JDBC data source based on the
   # properties prefixed with "jdbc.default.".
   #
   #jdbc.default.jndi.name=jdbc/LiferayPool
   ```

   Using the JNDI name is more secure than providing the connection settings, which include the user name and password, in `portal-ext.properties`.

   If you do not use the JNDI name, Studio logs a warning.

   Before you can use the JNDI name, you must first configure a JDBC data source within your application server. The steps to configure a JDBC data source will vary based on your application server. See your application server documentation for details.

   Make sure that the value of `jdbc.default.jndi.name` matches exactly the JNDI name you assign to the data source.

7. Start Studio. Monitor the logs to check for error messages while Studio connects to the database and creates the tables.

8. After the tables have been created and you have verified that Studio is running, you may remove the user's alter table privileges.

   Note that you may have to restore these later if you upgrade Studio or install components that require schema changes.

Chapter 13

# Using Studio with a Reverse Proxy

Studio can be configured to use a reverse proxy.

*About reverse proxies*

*Example sequence for a reverse proxy request*

*Recommendations for reverse proxy configuration*

## About reverse proxies

A reverse proxy provides a more secure way for users to get access to application servers.

*What is a reverse proxy?*

*Types of reverse proxies*

### What is a reverse proxy?

A reverse proxy retrieves resources on behalf of a client from one or more servers, and then returns these resources to the client as though they came from the server itself.

A reverse proxy is located between the client and the proxied server(s). Clients access content through the proxy server. The reverse proxy server assumes the public hostname of the proxied server. The hostname(s) of the actual/proxied servers are often internal and unknown to the client browser.

Some common reasons for implementing a reverse proxy include:

- Security or firewalling
- SSL termination
- Load balancing and failover
- Resource caching/acceleration
- URL partitioning

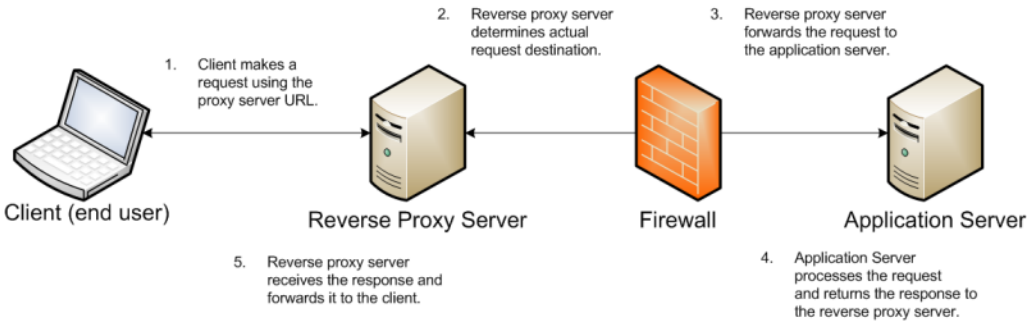
### Types of reverse proxies

Reverse proxies may be either be devices/appliances or specially configured web servers.

A very popular software-based reverse proxy is the Apache HTTP Server configured with the `mod_proxy` module. Many commercial web servers and reverse proxy solutions are built on top of Apache HTTP Server, including Oracle HTTP Server.

# Example sequence for a reverse proxy request

Here is an example of the typical sequence for a request processed using a reverse proxy server.




1. The client makes a request to the public URL.

   For this example, for a Studio application, the request URL might be something like `http://mystudio/web/myapp`, using the default port 80.

   The hostname resolves to the address of the reverse proxy server. The reverse proxy is listening on this address and receives the request.

2. The reverse proxy server analyzes the URL to determine where the request needs to be proxied to.

   A reverse proxy might use any part of the URL to route the request, such as the protocol, host, port, path, or query-string. Typically the path is the main data used for routing.

   The reverse proxy configuration rules determine the outbound URL to send the request to. This destination is usually the end server responsible for serving the content. The reverse proxy server may also rewrite parts of the request. For example, it may change or make additions to path segments.

   Reverse proxies can also add standard or custom headers to the request.

   For example, the URL `http://mystudio/web/myapp` might be proxied to `http://studioserver1:8080/studio/web/myapp`. In this case:

   - The hostname of the target server is studioserver1
   - The port is changed to 8080
   - The request URI path has `/studio` added in front of it

3. The reverse proxy server sends the request to the target server.
4. The target server sends the response to the reverse proxy server.
5. The reverse proxy server reads the request and returns it to the client.

# Recommendations for reverse proxy configuration

Here are some general configuration recommendations for setting up a reverse proxy.

*Preserving HTTP 1.1 Host: headers*

*Enabling the Apache ProxyPreserveHost directive*

## Preserving HTTP 1.1 Host: headers

HTTP 1.1 requests often include a `Host:` header, which contains the hostname from the client request. This is because a server may use a single IP address or interface to accept requests for multiple DNS hostnames.

The `Host:` header identifies the server requested by the client. When a reverse proxy proxies an HTTP 1.1 request between a client and a target server, when it makes the request, it must add the `Host:` header to the outbound request. The `Host:` header it sends to the target server should be the same as the Host: header it received from the client. It should not be the Host: header that would be sent if accessing the target server directly.

When the application server needs to create an absolute, fully-qualified URL, such as for a redirect URL or an absolute path to an image or CSS file, it must provide the correct hostname to the client to use in a subsequent request.

For example, a Java application server sends a client-side redirect to a browser (HTTP 302 Moved). It uses the `ServletRequest.getServerName()` method to fetch the hostname in the request, then constructs a `Host:` header.

The URL sent by the client is `http://mystudio/web/myapp`. The actual internal target URL generated by the reverse proxy will be `http://studioserver1:8080/studio/web/myapp`.

If there is no specific configuration for the target server, then if the reverse proxy retains the `Host:` header, the header is:

```
Host: http://mystudio
```

If the reverse proxy does not retain the `Host:` header, the result is:

```
Host: http://studioserver1:8080
```

In the latter case, where the header uses the actual target server hostname, the client may not have access to `studioserver1`, or may not be able to resolve the hostname. It also will bypass the reverse proxy on the next request, which may cause security issues.

If the `Host:` header cannot be relied on as correct for the client, then it must be configured specifically for the web or application server, so that it can render correct absolute URLs.

Most reverse proxy solutions should have a configuration option to allow the `Host:` header to be preserved.

## Enabling the Apache ProxyPreserveHost directive

The `ProxyPreserveHost` directive is used to instruct Apache `mod_proxy`, when acting as a reverse proxy, to preserve and retain the original `Host:` header from the client browser when constructing the proxied request to send to the target server.

The default setting for this configuration directive is Off, indicating to not preserve the `Host:` header and instead generate a `Host:` header based on the target server's hostname.

Because this is often not what is wanted, you should add the `ProxyPreserveHost On` directive to the Apache HTTPD configuration, either in `httpd.conf` or related/equivalent configuration files.

# Reverse proxy configuration options for Studio

Here are some options for configuring reverse proxy for Studio.

*Simple Studio reverse proxy configuration*

*Studio reverse proxy configuration without preserving Host: headers*

*Configuring Studio to support an SSL-enabled reverse-proxy*

## Simple Studio reverse proxy configuration

Here is a brief overview of a simple reverse proxy configuration for Studio. The configuration preserves the `Host:` header, and does not use SSL or path remapping.

In this simple configuration:

- A reverse proxy server is in front of a single Studio application server.
- The reverse proxy server is configured to preserve the `Host:` header.
- Neither the reverse proxy nor the application server is configured for SSL.

With this setup, Studio should be able to be accessed correctly using the reverse proxy without additional configuration.

For the bundled Tomcat configuration (running on port 8080), a sample `mod_proxy` would be:

```
ProxyRequests Off
ProxyPreserveHost On
ProxyPass / http://studioserver1:8080
ProxyPassReverse / http://studioserver1:8080
```

## Studio reverse proxy configuration without preserving Host: headers

If a reverse proxy fronting Studio does not preserve the `Host`: header, and instead makes a request with a `Host:` header referring to the target application server, Studio and its application server receive an incorrect hostname. This causes Studio to generate absolute URLs that refer to the proxied application server instead of to the reverse proxy server.

If the reverse proxy cannot be configured to preserve the `Host:` header, you must configure a fixed hostname and port. To do this, you can either:

- Configure the application server to have a fixed hostname and port
- Use `portal-ext.properties` to configure Studio with a fixed hostname and port

### Configuring a fixed hostname for the application server

For Tomcat, in `server.xml`, in the `<Connector>` element, set the `scheme`, `proxyName` and `proxyPort` attributes.

For WebLogic, you set up a virtual host with the fixed hostname and port.

### Configuring Studio with a fixed hostname

To configure Studio with a fixed hostname and port, add the following properties to `portal-ext.properties`:

```
web.server.host=<hostname of reverse proxy server>
web.server.http.port=<port of reverse proxy server>
```

## Configuring Studio to support an SSL-enabled reverse-proxy

If Studio is installed behind a reverse proxy that has SSL capabilities, and the client SSL is terminated on the reverse proxy, you must configure Studio to set the preferred protocol to HTTPS.

To do this, in `portal-ext.properties`, set `web.server.protocol` to `https`.

```
web.server.protocol=https
```

Chapter 14

# Creating a Studio Cluster

For a larger production environment, you may want to configure a cluster of Studio instances.

*About Studio clustering*

*Setting up the cluster*

## About Studio clustering

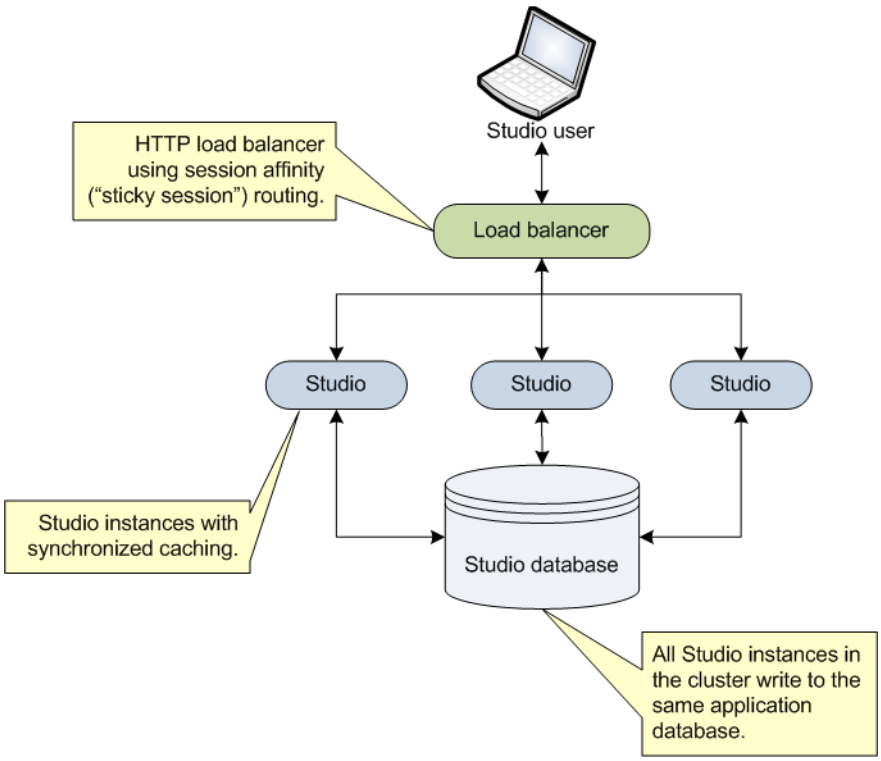
Studio allows you to create clusters of Studio instances. In a cluster, changes made to one instance are automatically made to the other instances. For a large production environment, using clustering provides redundancy and support for higher throughput, allowing for more concurrent users.

The cluster is made up of Studio instances configured to write to the same application database. For a clustered implementation, you must use a database other than the default Hypersonic database.

The Studio instances also must be configured to use synchronized caching, so that information cached on one instance is available to all of the other instances in the cluster. Studio uses Ehcache (*www.ehcache.org*), which uses RMI (Remote Method Invocation) multicast to notify each member of the cluster when the cache has been updated.

While there are multiple ways to cluster an application, based on the application server, Studio supports using an HTTP load balancer in front of the Studio instances. The load balancer must use session affinity (also

known as "sticky session") load balancing. If a member of the cluster is down, the load balancer routes requests to another instance in the cluster.




# Setting up the cluster

To configure a cluster, you connect each instance to the same database, and then configure a clustered cache for those instances.

*Installing the Studio instances*

*Configuring synchronized caching for the Studio instances*

## Installing the Studio instances

Each instance in the cluster is first installed as a standalone instance.

### Connecting each instance to the same Studio database

Each instance in the Studio cluster must be connected to the same Studio application database. For a clustered configuration, you must use a database system other than the default Hypersonic database. For details on changing the database system used to store the Studio application database, see *Changing the Database Used to Store Studio Application Data on page 57*. Optionally, you could use a clustered database configuration. For clustering, Oracle 11g uses RAC and MySQL has MySQL Cluster. For details on setting up a clustered database configuration, see the documentation for your database system.

### Using the same configuration for each instance

In a clustered configuration, each instance should have the same configuration, to ensure that users have the same experience no matter which instance in the cluster they are connected to.

Most of the application settings are stored in the database. Because each instance writes to the same database, those settings remain constant among the cluster instances.

Also make sure that each instance has the same settings in **portal-ext.properties**. This includes any **Framework Settings** that you set in the file instead of from the **Control Panel** user interface.

## Configuring synchronized caching for the Studio instances

Studio instances in a cluster must use synchronized caching.

### About synchronized caching

Synchronized caching ensures that the information cached by one Studio instance is available to all of the instances in the cluster.

This reduces the number of times each instance needs to query the Studio database, which allows for faster response times and better performance. Studio uses Ehcache (*www.ehcache.org*) for caching synchronization.

### Updating portal-ext.properties to enable synchronized caching

The `portal-ext.properties` file for each instance includes commented-out settings for synchronizing the caches.

For each instance in the cluster, uncomment the following clustering settings in `portal-ext.properties`. You should be able to use the default values provided.

```
##
## Cluster
##
# Uncomment the following properties to enable clustering
# Note: Clustering will not work with Hypersonic.  Configure a common database for all cluster nodes.

#net.sf.ehcache.configurationResourceName=/ehcache/hibernate-clustered.xml
#ehcache.multi.vm.config.location=/ehcache/liferay-multi-vm-clustered.xml
#org.quartz.jobStore.isClustered=true
```

The settings are:

| Setting | Description |
|---|---|
| `net.sf.ehcache.configurationResourceName` | The name and location of the XML configuration file for Hibernate caching. Hibernate is used by Studio to read from and write to the Studio application database.<br>In the default `portal.properties` file, the configuration file is set to `hibernate.xml`, to implement caching in a non-clustered implementation.<br>When you uncomment this property in `portal-ext.properties`, which changes the configuration file to `hibernate-clustered.xml`, then Hibernate synchronizes the cache with the other members of the cluster. |
| `ehcache.multi.vm.config.location` | The name and location of the XML configuration file for Ehcache.<br>In the default `portal.properties` file, the file is set to `liferay-multi-vm.xml`, to implement caching in a non-clustered implementation.<br>When you uncomment this property in `portal-ext.properties`, which changes the configuration file to `liferay-multi-vm-clustered.xml`, then the cache is synchronized with the other members of the cluster. |
| `org.quartz.jobStore.isClustered` | Enables clustering on the built-in Quartz job scheduling engine. |

Both configuration files are configured to automatically detect the other members of the cluster, and to use IP address 233.0.0.1 and port 4446 to send the updated cache information.

## Customizing the clustered cache configuration files

The default versions of the clustered cache configuration files should work in most cases. However, you can if needed create and deploy customized versions.

The most likely customization that might be needed would be to the IP address and port number configured near the top of each file:

```
<cacheManagerPeerProviderFactory
        class="net.sf.ehcache.distribution.RMICacheManagerPeerProviderFactory"
```

```
        properties="peerDiscovery=automatic,multicastGroupAddress=230.0.0.1,multicastGroupPort
=4446,timeToLive=1"
        propertySeparator=","
    />
```

If you make any changes to these configuration files, make sure to make the same changes for all of the instances in the cluster.

To customize the clustered cache configuration files:

1. Extract the default files from the `ehcache` directory in `portal-impl.jar`.

   The file is in the `WEB-INF\lib` directory, which is located:

   - For Tomcat, in `webapps\ROOT`
   - For WebLogic, in `endeca-portal-`*`versionNumber`*`.war`, which is in `endeca-portal-weblogic-`*`versionNumber`*`.ear`

2. Make the necessary updates to the files.

   To ensure that Studio uses the correct files, you may want to rename the customized files to something like:

   - `hibernate-clustered-custom.xml`
   - `liferay-multi-vm-clustered-custom.xml`

3. For Tomcat, copy the customized files into `webapps\ROOT\WEB-INF\classes\ehcache`.
4. To deploy the customized files in Weblogic:
   (a) Undeploy `endeca-portal-weblogic-`*`versionNumber`*`.ear`.

   Use the appropriate method to undeploy the file based on whether you auto-deployed the .ear file or installed it.

   (b) Update `endeca-portal-weblogic-`*`versionNumber`*`.ear` to add a subdirectory `APP-INF/classes/ehcache/` that contains the customized XML files.

   (c) Redeploy the updated .ear file.
5. If needed, update `portal-ext.properties` to reflect the customized file names:

   ```
   net.sf.ehcache.configurationResourceName=/ehcache/hibernate-clustered-custom.xml
   ehcache.multi.vm.config.location=/ehcache/liferay-multi-vm-clustered-custom.xml
   ```

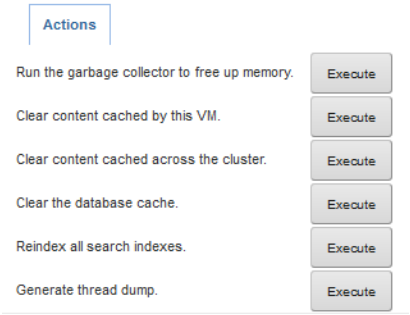
## Clearing the cache for a cluster

As part of troubleshooting issues with a clustered implementation, you can clear the cache. From the Studio **Control Panel**, you can clear the cache for either the current instance or for the entire cluster.

To clear the cache:

1. From the Studio menu, select **Control Panel**.
2. On the **Control Panel** menu, in the **Server** section, click **Server Administration**.

3. At the bottom of the page, on the **Actions** tab:




- To clear the cache for the current instance only, click the **Execute** button next to **Clear content cached by this VM**.
- To clear the cache for the entire cluster, click the **Execute** button next to **Clear content cached across the cluster**.

# Part III

## Installing the Provisioning Service

Chapter 15

# Installing the Provisioning Service

The Provisioning Service is installed as an application on Oracle WebLogic Server.

Separate Provisioning Service WebLogic installation packages are available for Windows and Linux. Both packages contain the following files and subdirectories:

- `eidProvisioningTemplate.jar`

  This file is the domain template used in the basic installation process. See *Provisioning Service domain creation and installation on page 73*.
- `eidProvisioning.ear`
- `eidProvisioningConfig`

  This directory contains the following files:
  - `config.py.sample`
  - `createDDL.sql`
  - `logging.properties`
  - `plan.xml`
- `doc`

  This directory contains the release notes and *Licensing Guide* for the Provisioning Service.

Recommended practice is to install the Provisioning Service after installing Endeca Server and Studio.

In production environments, the Provisioning Service should be installed on a separate machine from other Oracle Endeca Information Discovery products, such as Endeca Server and Studio. In other environments (evaluation, training, development, and demonstration), the Provisioning Service can be installed in the same machine and WebLogic Server as other Oracle Endeca Information Discovery components, although the Provisioning Service still requires a unique WebLogic domain.

*Provisioning Service installation prerequisites*

*Provisioning Service installation process*

*Provisioning Service domain creation and installation*

*Configuring the Provisioning Service*

*Implementing SSL communication with Endeca Server and Studio*

*Configuring the Provisioning Service for non-secure communication*

*Setting up a production database*

*Checking the Provisioning Service installation*

# Provisioning Service installation prerequisites

Before installing the Provisioning Service, create a data domain profile.

Oracle recommends that you create a data domain profile to use when creating data domains to support Provisioning Service. The following configurations are recommended:

- allowQueriesOnLeader

  See "Data Domain profile parameters" in the *Oracle Endeca Server Cluster Guide* for recommendations.

- readOnly

  Set to `false`. Also see "Data Domain profile parameters" in the *Oracle Endeca Server Cluster Guide* for further information.

- numFollowers

  If you want to use a single dgraph process, set to `0`; otherwise, see "Data Domain profile parameters" in the *Oracle Endeca Server Cluster Guide* for recommendations.

- allowOversubscribe

  Set to `true`.

- numComputeThreads

  Determine the correct value using the following formula: `(core count of the Endeca Server host/ anticipated number of user-created applications) * 2`

- computeCacheSizeMB

  See "Data Domain profile parameters" in the *Oracle Endeca Server Cluster Guide* for recommendations.

- sessionIdType

  Set to `header`. (This is the default value.)

- sessionIdKey

  Set to `X-Endeca-Session-ID`. (This is the default value.)

# Provisioning Service installation process

Install the Provisioning Service into its own WebLogic domain.

The process described below assumes you are installing the Provisioning Service into a new WebLogic instance for production. For evaluation, training, and development, you can install the Provisioning Service on the same WebLogic Server as Endeca Server and Studio. Note that in this case, the Provisioning Service still requires its own WebLogic domain.

To install the Provisioning Service:

1. Download and install the Oracle Sun JDK.

   For details about installing the Oracle Java JDK, see *Downloading and installing the Sun JDK on page 31*.

2. Download and install WebLogic Server.

   For details about installing the WebLogic Server, see *Downloading and installing WebLogic Server on page 34*.

3. Create a WebLogic domain and install the Provisioning Service.

   A domain template is provided to create the WebLogic domain for the Provisioning Service. The domain template also installs the Provisioning Service. For details about using this domain template, see *Provisioning Service domain creation and installation on page 73*.

4. Configure the Provisioning Service. For details, see *Configuring the Provisioning Service on page 80*.

5. Configure secure communication for the Provisioning Service (for details, see *Implementing SSL communication with Endeca Server and Studio on page 82*), or configure non-secure communication for the Provisioning Service (for details, see *Configuring the Provisioning Service for non-secure communication on page 83*).

# Provisioning Service domain creation and installation

Use this procedure to create the WebLogic domain and install the Provisioning Service

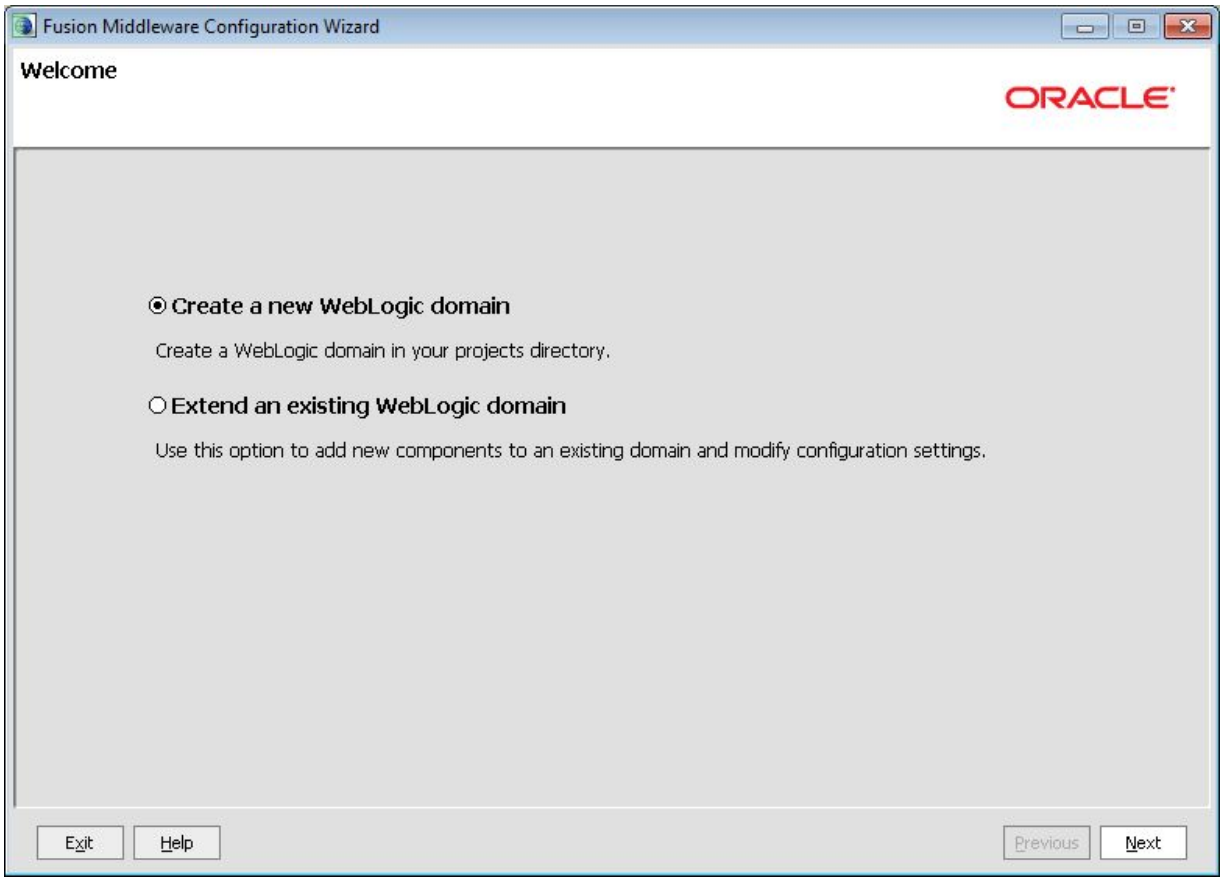
For prerequisites, see *Provisioning Service installation prerequisites on page 72*.

To create the WebLogic domain and install the Provisioning Service:

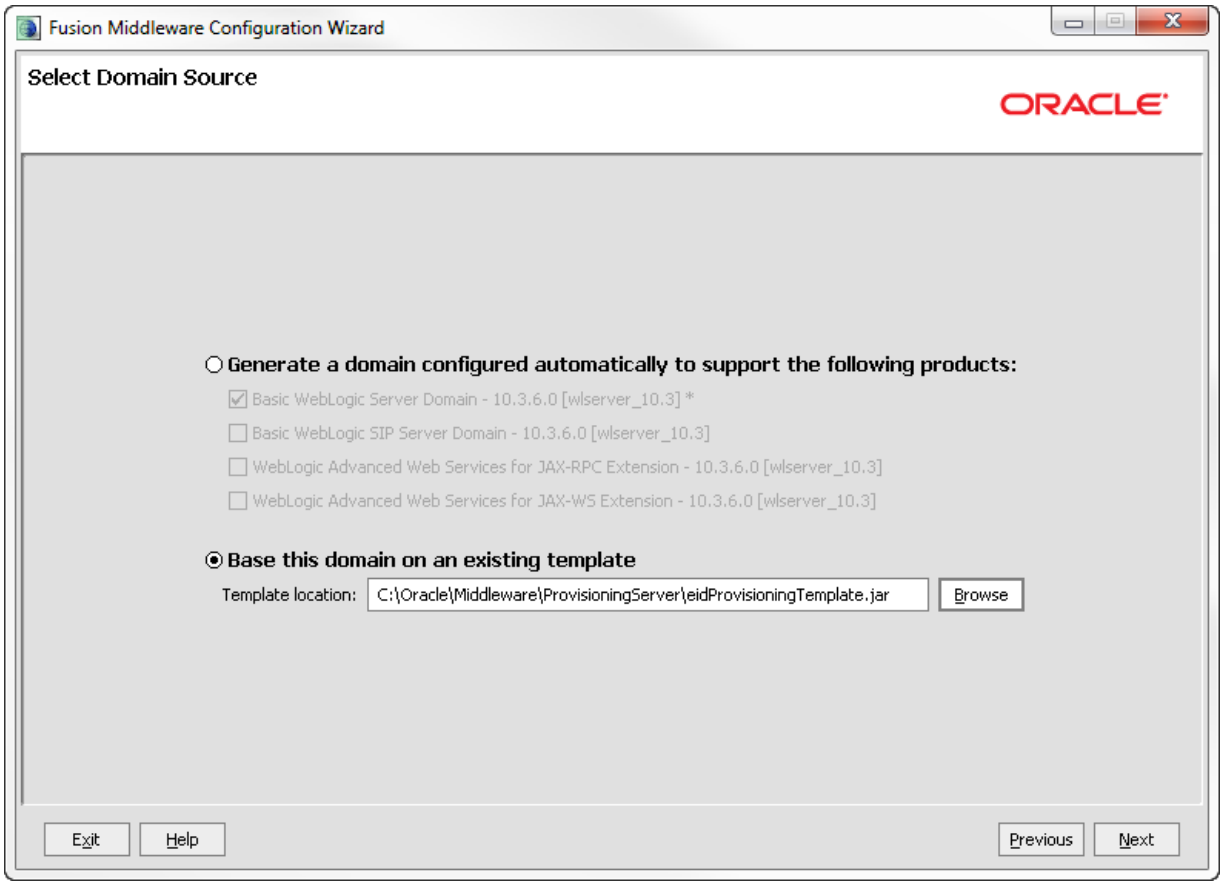
1. Unzip the Provisioning Service installation package into a convenient directory.
   For example, you could create a directory called eid_ProvisioningService at the root of your WebLogic Server installation and unzip the Provisioning Service installation package to that location.
2. From a command prompt, change to the directory that contains the Configuration Wizard start-up program:
   - For Linux: `$MW_HOME/wlserver_10.3/common/bin`
   - For Windows: `$MW_HOME\wlserver_10.3\common\bin`
3. Run the command to start the Configuration Wizard:
   - For Linux: `./config.sh`
   - For Windows: `config.cmd`

   The Configuration Wizard Welcome screen is displayed.

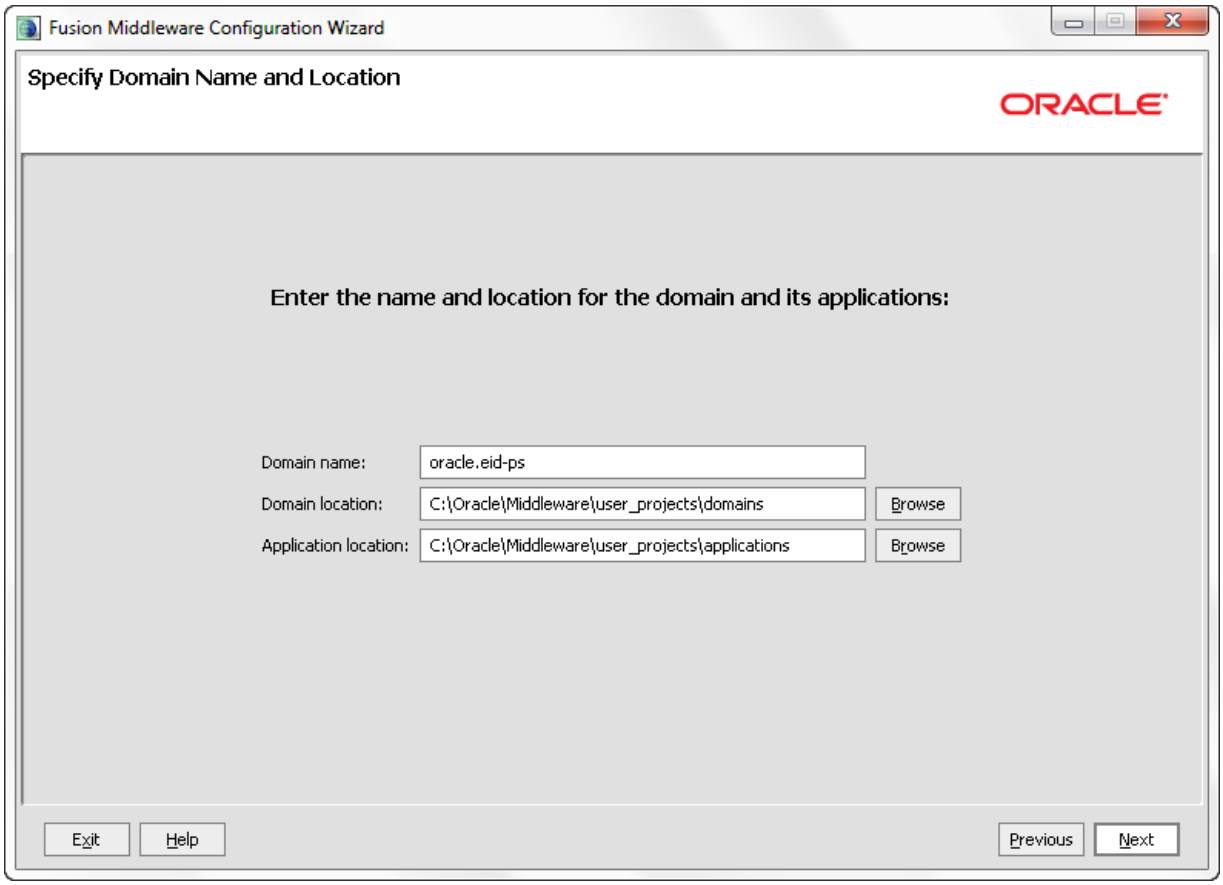
4. On the Configuration Wizard's Welcome screen, select **Create a new WebLogic domain** and click **Next**.



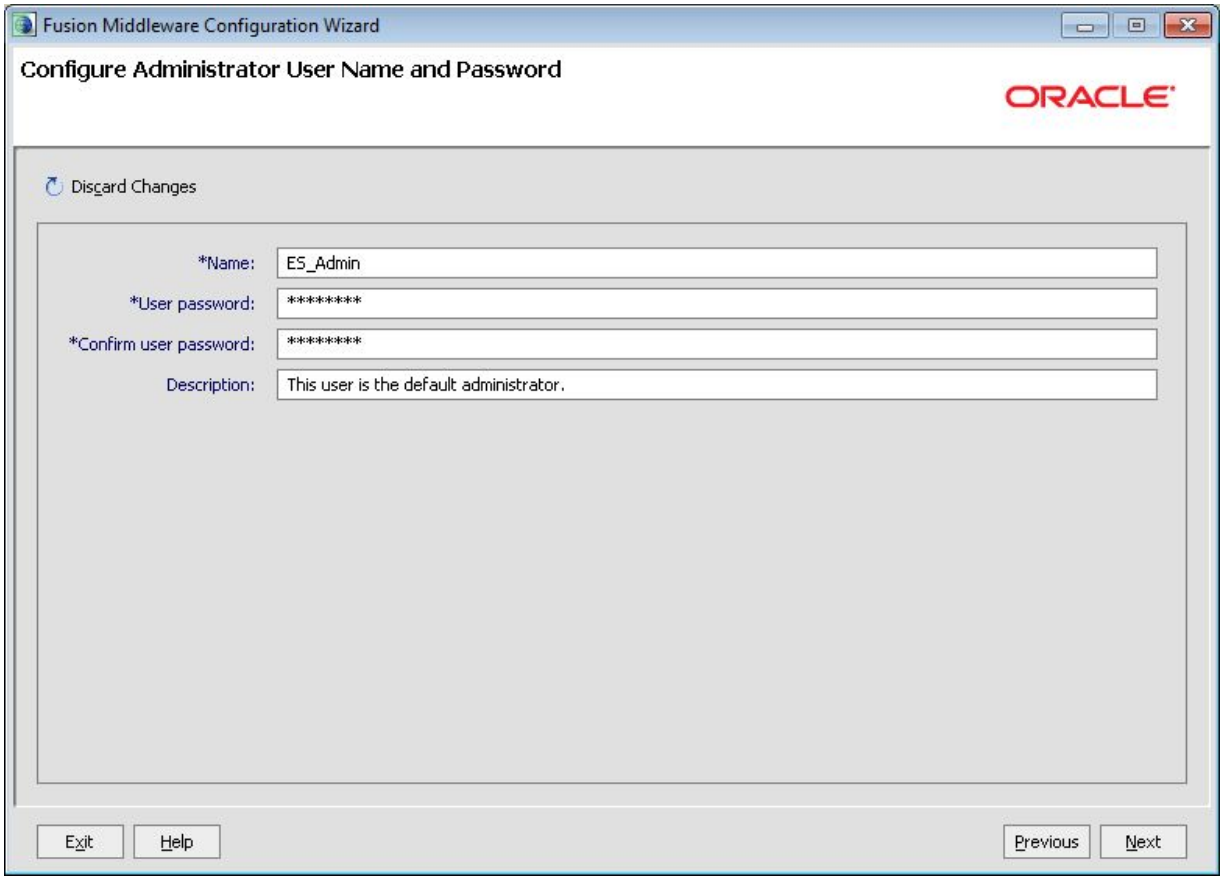
5. On the Select Domain Source screen, select **Base this domain on an existing template**. Click the browse button and browse to the directory where you unzipped the Provisioning Service installation package. Choose the file `eidProvisioningTemplate.jar`. Then click **Next**.



6. On the Specify Domain Name and Location screen, the domain name defaults to **oracle.eid-ps**. You can keep this name or change it to a different name. Do not change the **Domain location** or **Application location.** Click **Next**.

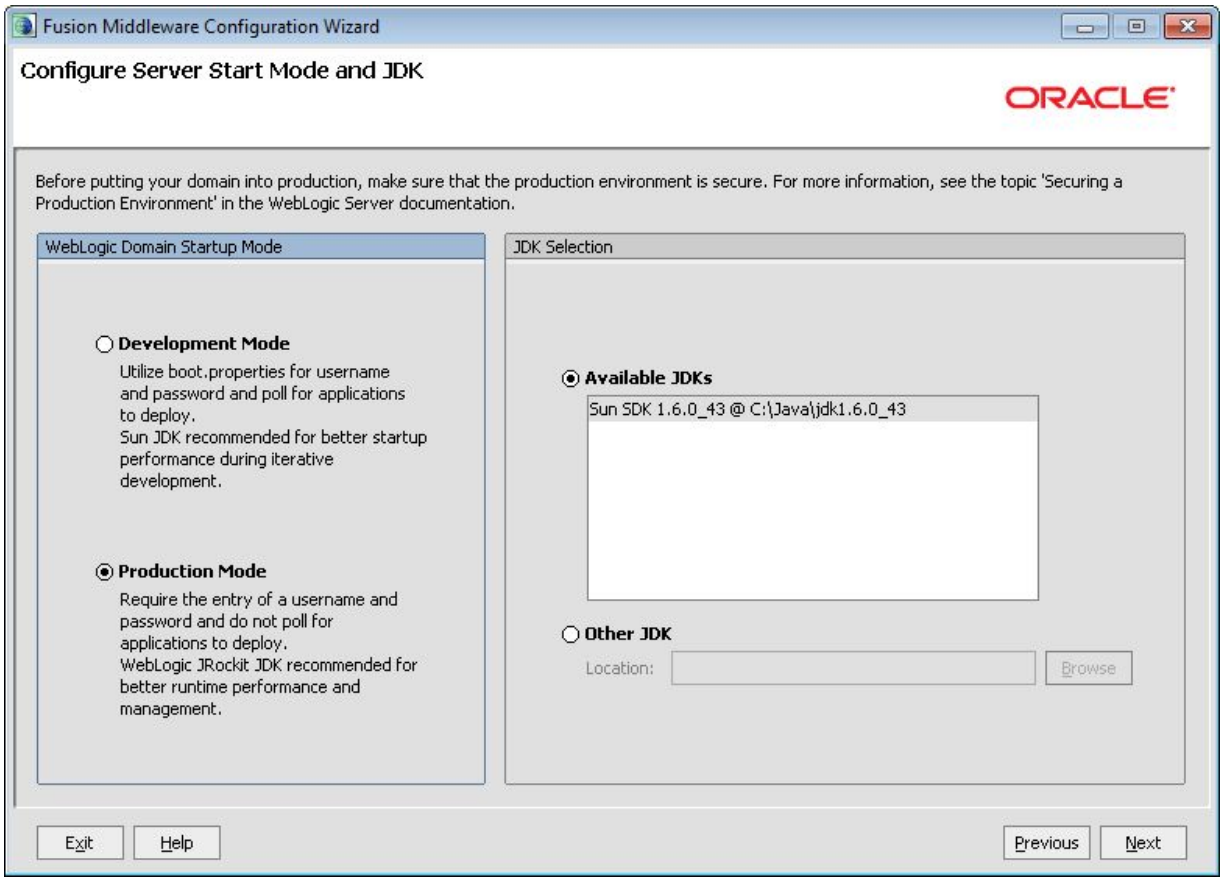


7. On the Configure Administrator User Name and Password screen, enter the name and password you want to use for the administrator. Optionally, you can enter a description. Then click **Next**.
   You will be using this name and password to start the WebLogic Server. Note that the password has a minimum length of eight characters.



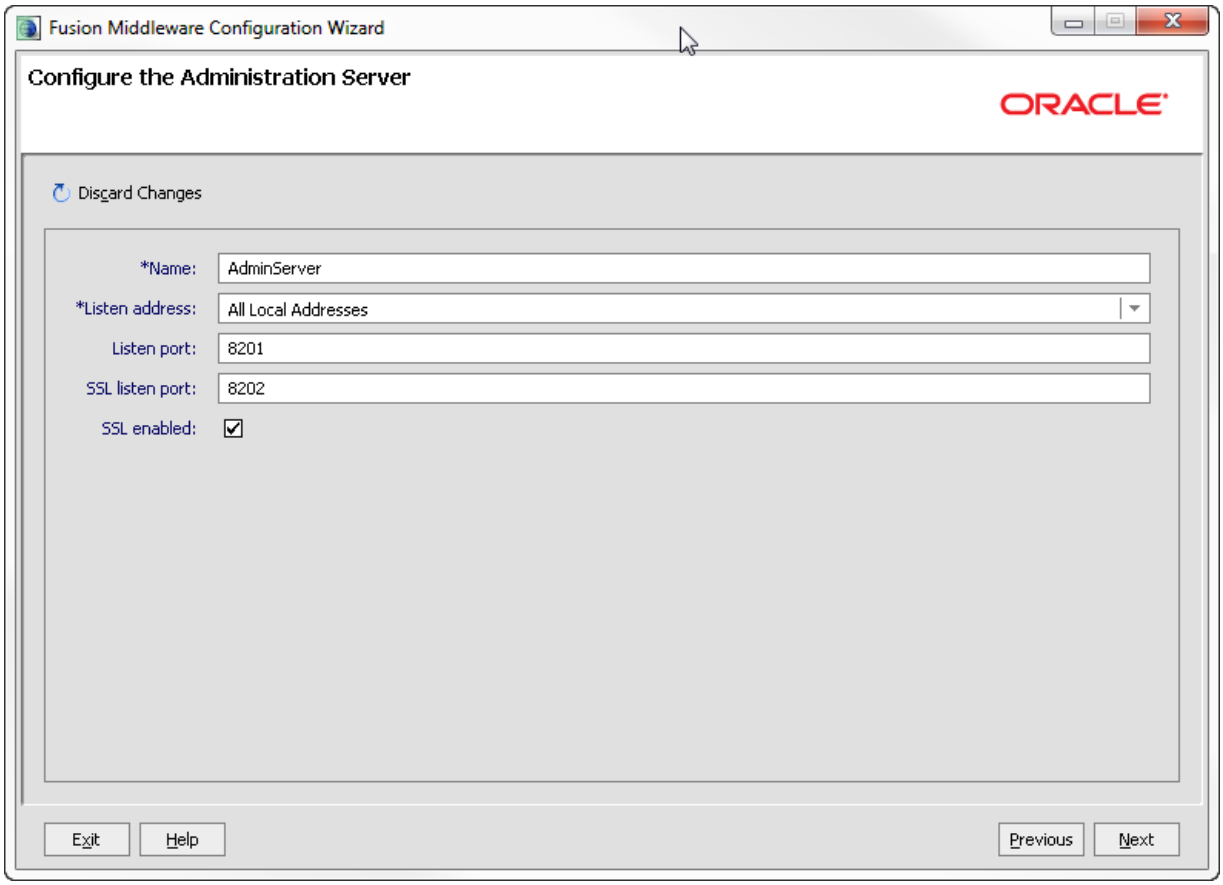

8. On the Configure Server Start Mode and JDK screen, do the following:
   (a) In the **WebLogic Domain Startup Mode** pane, select **Production Mode**.
   (b) In the **JDK Selection** pane, click **Available JDKs** and select the **Sun SDK**. (Note that if the Sun JDK is not displayed in this pane, click the **Other JDK** button and browse for the Sun JDK home.)

(c) Click **Next**.




9. On the **Select Optional Configuration** dialog, check **Administration Server**. Click **Next.**

10. On the **Configure the Administration Server** dialog, change the value of the **Listen port** field to *8201*. Change the value of the **SSL listen port** to *8202*. Click **Next**.




11. On the Configuration Summary screen, click **Create** to create the domain.
12. On the Creating Domain screen, the message "Domain Created Successfully!" signifies the end of the create process. Click **Done** to exit the Configuration Wizard.

The Weblogic domain *oracle.eid-ps* is created with a WebLogic Server port of 8201 and an SSL port of 8202.

After completing the installation:

- Configure the Provisioning Service. For details, see *Configuring the Provisioning Service on page 80*.
- Configure the Provisioning Service for secure communication (for details, see *Implementing SSL communication with Endeca Server and Studio on page 82*), or for non-secure communication (for details, see *Configuring the Provisioning Service for non-secure communication on page 83*).

After installation, you may also want to do the following:

- Update the Java heap space for the Provisioning Service.

  The default heap space in the Provisioning Service configuration is 6 GB. Heap space is set in the domain environment variables script:`$DOMAIN_HOME/bin/setDomainEnv.sh` in Linux or `$DOMAIN_HOME/bin/setDomainEnv.cmd` in Windows. In the line

  ```
  USER_MEM_ARGS="-Xms256m -Xmx6144m -XX:PermSize=64M -XX:MaxPermSize=256M"
  ```

  change the `-Xmx6144m` to the amount of heap space you want to use.

- Implement logging. For details about implementing logging, see "Changing the logging level for the Provisioning Service" in the *Oracle Endeca Information Discovery Studio Provisioning Service Administration Guide*.

# Configuring the Provisioning Service

You must configure Provisioning Service before running it.

To configure the Provisioning Service, in `wls_home/user_projects/domains/oracle.eid-ps/eidProvisioningConfig/plan.xml`, modify the configuration variables in the `<variable-definition>` as illustrated in the following code:

```
<variable-definition>
    <variable>
      <name>endeca-server-host-name</name>
      <value>${endecaServerHost}</value>
    </variable>
    <variable>
      <name>endeca-server-ws-port</name>
      <value>${endecaServerPort}</value>
    </variable>
    <variable>
      <name>endeca-server-context-root</name>
      <value>${endecaServerContextRoot}</value>
    </variable>
    <variable>
      <name>endeca-server-data-domain-profile</name>
      <value>${dataDomainProfile}</value>
    </variable>
    <variable>
      <name>endeca-server-security-enabled</name>
      <value>${endecaServerSSL}</value>
    </variable>
    <variable>
      <name>upload-file-directory</name>
      <value/>
    </variable>
          <variable>
               <name>message-driven-bean-limit</name>
      <value>2</value>
    </variable>
 </variable-definition>
```

**Table 15.1: Provisioning Service configurations**

| Configuration Parameter | Description | Value |
|---|---|---|
| endeca-server-host-name | Specifies the name of the host for the Endeca Server where you want to provision data. The default value assumes the Endeca Server is installed on the same machine as the Provisioning Service. | Name or IP address of the machine that hosts the Endeca Server you want to provision. |

| Configuration Parameter | Description | Value |
|---|---|---|
| endeca-server-ws-port | Specifies the web services port for the Endeca Server where you want to provision data. The default value is the default Endeca Server https port. | Port of the Endeca Server you want to provision. |
| endeca-server-context-root | Specifies the WebLogic application root context of the Endeca Server. | Typically, the value will be `/endeca-server`, but if you install Endeca Server to a different context root, enter your Endeca Server's context root in this configuration property. |
| endeca-server-data-domain-profile | Specifies the name of the Endeca data domain profile that will be used to create data domains when data is uploaded through the Provisioning Server. | Name of the Endeca Server data domain profile that you want to use to create data domains when users upload data. Endeca Server is installed with a default data domain profile, but Oracle recommends that you define one or more data domain profiles that match your needs. For details, see *Provisioning Service installation prerequisites on page 72*. |
| endeca-server-security-enabled | Specifies whether the Provisioning Service should use SSL to communicate with Endeca Server. | Specify `true` if you have SSL configured on Endeca Server and want to use SSL to communicate with Endeca Server. Otherwise, specify `false`.<br><br>If you do not want to use SSL, see *Configuring Provisioning Service for non-secure communication on page 83*. |
| upload-file-directory | Specifies the directory on the Provisioning Service host machine where uploaded files will be stored. The directory is relative to the domain home directory. | By default, the Provisioning Service uses the operating system temporary directory. This directory is not very secure, and often scripts modify its contents. It is safer to specify a directory to store the uploaded files.<br><br>The directory you specify must exist, and the user that runs the Provisioning Service must have write permissions on the directory. |

| Configuration Parameter | Description | Value |
|---|---|---|
| message-driven-bean-limit | Specifies the maximum number of message-driven beans allowed in the pool on the WebLogic Server host of the Provisioning Service. The number of message-driven beans defines the number of Excel files that the Provisioning Service can process at each stage of the processing workflow. The Provisioning Service workflow consists of three stages. | For recommendations about controlling memory, see "Managing memory usage" in the *Oracle Endeca Information Discovery Studio Provisioning Service Administration Guide*. |

# Implementing SSL communication with Endeca Server and Studio

By default, the Provisioning Service is configured to support SSL. Additional implementation is required to enable SSL communication with Endeca Server and Studio.

To implement secure communication with Endeca Server and Studio over SSL:

1. Copy the keystore cert files from the home directory of the Endeca Server installation to the domain home directory of the Provisioning Service.
   Copy the following cert files:
   - endecaServerClientCert.ks
   - endecaServerTrustStore.ks
2. Configure the identity keystore and trust keystore on the WebLogic Server.
   (a) Start WebLogic Server. Go to the Administration Console for your server and log in.
   (b) On Keystores tab, change value in the **Keystores** field to `Custom Identity and Custom Trust`. Save this change.
   (c) In the **Custom Identity Keystore** field, enter the path to the identity keystore.
   (d) In the **Custom Identity Keystore Type** field, enter `JKS`.
   (e) In the **Custom Identity Keystore Passphrase** and **Confirm Custom Identity Keystore Passphrase** fields, enter the keystore passphrase used to generate the key on the Endeca Server.
   (f) In the **Custom Trust Keystore** field, enter the path to the trust keystore.
   (g) In the **Custom Trust Keystore Type** field, enter `JKS`.
   (h) In the **Custom Trust Keystore Passphrase** and **Confirm Custom Trust Keystore Passphrase** fields, enter the keystore passphrase used to generate the key on the Endeca Server.
3. Configure the WebLogic Server SSL parameters:
   (a) On the Administration Console of your WebLogic Server, click the SSL tab.
   (b) The value of the **Private Key Alias** must be `Oracle Endeca Server Client Certificate`. (This value should default from the keystore.)

(c) In the **Private Key Passphrase** and **Confirm Private Key Passphrase** fields, enter the passphrase you entered when creating the certificates on the Endeca Server installation.

(d) Click **Save**.

# Configuring the Provisioning Service for non-secure communication

The default configuration of the Provisioning Service assumes secure communication with Endeca Server and with Studio over SSL. You must modify this configuration if you want to permit non-SSL communication.

To configure non-secure communication, in plan.xml:

- Change the value of the `endeca-server-security-enabled` variable to `false`.
- Change the value of the `transport-guarantee` variable to `NONE`.
- Change the value of the `protected-url-pattern` variable to `/DISABLED`.

Also, confirm that the value of the `endeca-server-ws-port` variable is `7001` (the non-secure http port); the value of this parameter defaults to `7002`, which is the secure http port.

# Setting up a production database

When using the Provisioning Service in a production environment, the Weblogic Server container should be configured to use an Oracle 11g database.

The Oracle database schema you want to use to support the Provisioning Service must be set up before you can begin to use the Provisioning Service. You can use the `createDDL.sql` script in the Provisioning Service installation package (`eidProvisioning.zip`) to create the schema.

The domain template used in the basic installation uses a Derby database installed with WebLogic server for its internal database. This database is not suitable for use in a production environment.

To configure WebLogic Server to use an Oracle database:

1. Log in to WebLogic Administration Console with the user used to install the Provisioning Service. Lock and edit the domain.
2. In the Domain Structure box, expand the Services node and click on **Data Sources**

   Administrative Console displays the Summary of JDBC Data Sources dialog.
3. Select the jdbc/oracle.eid-ps datasource. Click **Delete**.

   WebLogic Server deletes the selected datasource.
4. Create a new datasource with the JNDI name `jdbc/oracle.eid-ps`.
   (a) Click **New** and from the drop menu, choose `Generic Data Source`.

   The Administration Console displays the **JDBC Data Source** wizard.
   (b) Enter a **Name** for the datasource. You can use any name you like.
   (c) In the **JNDI Name** field, enter `jdbc/oracle.eid-ps`. You must use this name.
   (d) In the **Database Type** drop list, choose `Oracle.`
   (e) Click **Next**.

(f) In the **Database Driver** drop list, choose the appropriate driver for your database. You must use an XA-type driver.

(g) Consult your database administrator for the appropriate data for the rest of the wizard.

(h) Target the datasource to the Admin Server when asked.

5. Disable Derby in the WebLogic Server domain environment variables. The environment variables are stored in the file `WLSDOMAINHOME/bin/setDomainEnv.cmd` on Windows or `WLSDOMAINHOME/bin/setDomainEnv.sh` on Linux. Change `DERBY_FLAG="true"` to `DERBY_FLAG="false"`.
6. Restart WebLogic Server.

# Checking the Provisioning Service installation

After you finish installing the Provisioning Service, you may want to confirm that it runs.

To confirm your installation:

1. Start the Provisioning Service domain in WebLogic.
2. Open a web browser, then connect and log in to the Provisioning Service domain as a user with administrative privileges.
3. In the **Domain Structure** section of the page, click on **Deployments**.
   WebLogic displays the **Summary of Deployments**.
4. Check the State and Health of the `oracle.eid-ps` application.
   The value in the **State** column should be `Active` and the value in the **Health** column should be `OK`.

# Part IV

## Uninstalling Oracle Endeca Information Discovery

Chapter 16

# Uninstalling Oracle Endeca Information Discovery Studio

This chapter describes how to uninstall the Studio and Provisioning Service modules of an Oracle Endeca Information Discovery Studio installation. For information on uninstalling Oracle Endeca Server, see the *Oracle Endeca Server Installation Guide*. For information on uninstalling Oracle Endeca Information Discovery Integrator, see the *Integrator Installation Guide*.

*Uninstalling Studio*

*Uninstalling the Provisioning Service*

## Uninstalling Studio

To uninstall Studio, remove the packages and directories that you installed.

## Uninstalling the Provisioning Service

To uninstall Provisioning Service, follow standard WebLogic procedures for uninstalling a Web application.

For details, see "Delete Web applications" in the ***WebLogic Server Administration Console Online Help*** .

# Index


## C

clustering, Studio
- about 64
- clearing the cache 68
- customizing the cache configuration 67
- enabling synchronized caching 66
- installing instances 65

context root
- changing for the Linux Tomcat bundle 18
- changing for the Windows Tomcat bundle 15

Control Panel, displaying in Studio 54

## D

database
- changing from Hypersonic in Studio 57
- reconfiguring for Provisioning Server 83

data source, updating default in Studio 55

## I

installation
- Provisioning Service 71
- Provisioning Service basic process 73
- Provisioning Service prerequisites 72
- Studio Linux Tomcat bundle 17
- Studio on Oracle WebLogic Server 30
- Studio on Tomcat 6.0 20
- Studio Windows Tomcat bundle 14

## P

prerequisites, Provisioning Service 72

Provisioning Service
- basic installation process 73
- configuring 80
- creating a data domain profile 72
- installation prerequisites 72
- installing 71
- system requirements 7
- uninstalling 86

## R

reverse proxy, using with Studio 59

## S

Studio
- changing from the Hypersonic database 57
- clustering, about 64
- clustering, about synchronized caching 66
- clustering, clearing the cache 68
- clustering, customizing the cache configuration 67
- clustering, enabling synchronized caching 66
- clustering, installing instances 65
- configuring the default data source 55
- displaying the Control Panel 54
- downloading 9
- installing the Sun JDK for WebLogic Server 31
- installing WebLogic Server 34
- Linux Tomcat bundle installation steps 17
- Oracle WebLogic Server installation steps 30
- running as Windows service 25
- starting 54
- system requirements 5
- Tomcat 6.0 installation 20
- uninstalling 86
- using a reverse proxy 59
- Windows Tomcat bundle installation steps 14

Studio troubleshooting
- JavaScript timeout 52
- starting the Windows service 28
- Tomcat connector keepAliveTimeout 53
- WebLogic timeout 53

Sun JDK
- downloading for WebLogic Server 31
- installing for WebLogic Server 31

system requirements
- Provisioning Service 7
- Studio 5

## T

Tomcat 6.0
- about the Studio install 20
- configuring Tomcat 22
- deploying Studio 23
- deploying Studio dependency libraries 21
- installing Tomcat 21
- starting Studio 23

Tomcat bundle (Linux)
- changing the context root 18
- installing 17

Tomcat bundle (Windows)
- changing the context root 15
- installing 14

## U

uninstalling
- Provisioning Service 86
- Studio 86

## W

WebLogic Server
- changing the Liferay Home directory for Studio 50

creating the Provisioning Service domain 73
creating the Studio domain 40
deploying Studio 46
downloading 34
installing 34
installing the Provisioning Service 73
installing the Sun JDK 31
setting non-root context for Studio 49
Windows service for Studio
about 25
configuring 26
installing 27
installing Tomcat monitor 25
obtaining installer files 25
starting 27
troubleshooting 28