Oracle® Fusion Middleware

Administering Oracle ADF Applications

12*c* (12.2.1.4.0)
E80013-01
September 2019

Oracle Fusion Middleware Administering Oracle ADF Applications, 12*c* (12.2.1.4.0)

E80013-01

Primary Author: Peter Jew

Contributors: Ayonee Sengupta, Krithika Gangadhar, Lynn Munsinger, Duncan Mills, Dipankar Bajpai, Harry Hsu

Contents

Preface

Welcome to *Developing Fusion Web Applications with Oracle Application Development Framework*.

Audience

This document is intended for enterprise developers who need to create and deploy database-centric Java EE applications with a service-oriented architecture using the Oracle Application Development Framework (Oracle ADF). This guide explains how to build Fusion web applications using ADF Business Components, ADF Controller, ADF Faces, and JavaServer Faces.

Documentation Accessibility

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

Access to Oracle Support

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

Related Documents

For more information, see the following documents:

- *Understanding Oracle Application Development Framework*
- *Developing Web User Interfaces with Oracle ADF Faces*
- *Developing Applications with Oracle ADF Desktop Integration*
- *Developing Applications with Oracle ADF Data Controls*
- *Developing Applications with Oracle JDeveloper*
- *Developing ADF Skins*
- *Administering Oracle ADF Applications*
- *Tuning Performance*
- *High Availability Guide*
- *Installing Oracle JDeveloper*
- *Oracle JDeveloper Online Help*

- *Oracle JDeveloper Release Notes*, included with your JDeveloper installation, and on Oracle Technology Network
- *Java API Reference for Oracle ADF Model*
- *Java API Reference for Oracle ADF Controller*
- *Java API Reference for Oracle ADF Lifecycle*
- *Java API Reference for Oracle ADF Faces*
- *JavaScript API Reference for Oracle ADF Faces*
- *Java API Reference for Oracle ADF Data Visualization Components*
- *Java API Reference for Oracle ADF Share*
- *Java API Reference for Oracle ADF Model Tester*
- *Java API Reference for Oracle Generic Domains*
- *Java API Reference for Oracle Metadata Service (MDS)*
- *Java API Reference for Oracle ADF Resource Bundle*
- *Tag Reference for Oracle ADF Faces*
- *Tag Reference for Oracle ADF Faces Skin Selectors*
- *Tag Reference for Oracle ADF Faces Data Visualization Tools*
- *Tag Reference for Oracle ADF Data Visualization Tools Skin Selectors*

Conventions

The following text conventions are used in this document:

Convention	Meaning
boldface	Boldface type indicates graphical user interface elements (for example, menus and menu items, buttons, tabs, dialog controls), including options that you select.
italic	Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.
`monospace`	Monospace type indicates language and syntax elements, directory and file names, URLs, text that appears on the screen, or text that you enter.

What's New in This Guide

The following topics introduce the new and changed features of Oracle JDeveloper and Oracle Application Development Framework (Oracle ADF) and other significant changes, which are described in this guide.

New and Changed Features for Release 12c (12.2.1.4.0)

Oracle Fusion Middleware Release 12c (12.2.1.4.0) of Oracle JDeveloper and Oracle Application Development Framework (Oracle ADF) includes the following new and changed development features, which are described in this guide:

- Users can now view different metrics on ADF Business Components as RESTful web services. This feature update was first introduced in the 12.2.1.2 documentation library and is repeated here for users who may not have seen this information in the 12.2.1.2 release. See Chapter 5, Diagnostic Tools.

Other Significant Changes in this Document for Release 12c (12.2.1.4.0)

For Release 12c (12.2.1.4.0), this document has been updated in several ways. Following are the sections that have been added or changed.

- The *Administering ADF Desktop Integration* chapter has been moved to the following guide: *Developing Applications with Oracle ADF Desktop Integration*. This update was first introduced in the 12.2.1.2 documentation library and is repeated here for users who may not have seen this information in the 12.2.1.2 release. Here is a link to the chapter’s new location.

Part I
Understanding Oracle ADF

This part introduces you to Oracle ADF, ADF architecture, and the different administration tasks that can be performed on applications developed with Oracle ADF.

Part I contains the following chapters:

1

Introduction to Oracle ADF Administration

This chapter describes the administrative tasks you can perform and the tools you can use to deploy, manage, monitor, and configure applications developed for the Oracle Application Development Framework (Oracle ADF). Some of the tools you will be using are the Oracle Enterprise Manager Fusion Middleware Control, Oracle JDeveloper, and WebLogic Scripting Tool (WLST).
This chapter includes the following sections:

- Introducing Oracle ADF
- Oracle ADF Architecture
- Administering Oracle ADF Applications

For definitions of unfamiliar terms found in this and other books, see the Glossary.

Introducing Oracle ADF

You can use Oracle Application Development Framework (Oracle ADF) to simplify the development of interactive, databound web applications, known as Fusion web applications (or simply ADF applications). Applications built with Oracle ADF follow Java EE standards and integrate with the Oracle Fusion Middleware stack.

The Oracle ADF builds on Java Platform, Enterprise Edition (Java EE) standards and open-source technologies to provide a complete framework for implementing service-oriented applications.

You can use this framework to provide enterprise solutions across different platforms. You can build applications that search, display, create, modify, and validate data for web, web services, desktop, or mobile interfaces.

You use Oracle JDeveloper with Oracle ADF to develop applications with an environment that supports the full development lifecycle of design, test, and deployment. For more information about ADF development, see Introduction to Building Fusion Web Applications with Oracle ADF in *Developing Fusion Web Applications with Oracle Application Development Framework*.

After you have developed and tested your ADF application in test environments, you can deploy your application to production environments using the tools described in this book. You can monitor the performance of applications as they are running. You can also manage and configure properties and attributes.

Oracle ADF Architecture

Oracle ADF architecture is a model-view-controller (MVC) model that has four layers in its MVC implementation. Use these four layers and stack components of Fusion web application technology to build ADF applications.

Oracle ADF supports the industry-standard MVC architecture to achieve separation of business logic, navigation, and user interface.

The MVC architecture provides:

- A model layer that represents the data values
- A view layer that contains the UI components
- A controller layer that handles input and navigation
- A business service layer that encapsulates business logic

The Fusion web application technology stack components are:

- ADF Model, for accessing declarative data binding metadata
- ADF Business Components, for building business services
- ADF Faces rich client, for AJAX-enabled UI components for web applications built with JavaServer Faces (JSF)
- ADF Controller, for input processing, navigation, and reusable task flows

ADF Business Components

ADF Business Components are application objects you can use to implement service-oriented Java EE applications. You implement ADF Business Components for clients to query, insert, update, and delete business data. You can apply business rules to the Business Components to enforce proper usage. The key components of ADF Business Components are the entity object, the view object, and the application module.

An *entity object* represents a row in a database table. It uses data manipulation language (DML) operations to modify data. Entity objects are used with others to reflect relationships in the database schema.

A *view object* represents a SQL query. You use the SQL Language to query the database to obtain the results. You can also link a view object with other entity objects to create master-detail hierarchies.

An *application module* is the transactional component that allows UI components to access data. It presents a data model and methods to perform certain tasks.

ADF Model

ADF Model implements a service abstraction called *data control*. Data control uses metadata interfaces to abstract business services. This metadata is used to describe data collections, properties, methods, and types. In JDeveloper, data controls appear in the Data Controls panel. When you drag and drop attributes, collections, and methods onto a page, JDeveloper automatically creates the bindings from the page to the associated services.

ADF Controller

ADF Controller provides a navigation and state management model that works with JSF. You can create navigational flows called task flows that encapsulate a specific task sequence.

ADF Faces Rich Client

ADF Faces provides over 100 rich components that can be used out of the box to create web applications. ADF Faces components provide built-in AJAX functionality to allow requests to be sent to the server without fully rendering the page. JSF provides server-side control to reduce the dependency on JavaScript. The components support skinning, internationalization, and accessibility options.

ADF Faces has a large set of components, including tables, trees, dialogs, accordions, and a variety of layout components. It also includes ADF Data Visualization components, which are Flash- and SVG-enabled, for displaying graphs, charts, and gauges.

ADF Desktop Integration

ADF Desktop Integration provides a framework for Oracle ADF developers to extend the functionality provided by a Fusion web application to desktop applications, such as Excel, even when they are disconnected from the network.

Using Excel's familiar user interface, end users can undertake information management tasks, such as performing complex calculations or uploading a large amount of data.

Administering Oracle ADF Applications

You can deploy, manage, monitor, and configure applications developed with Oracle ADF. Use the different tools to perform these administration tasks.

You can perform a variety of administration tasks on ADF applications. You can deploy ADF applications using Enterprise Manager Fusion Middleware Control, WLST commands, the `ojdeploy` command, scripts, or the WebLogic Administration Console.

After the ADF application has been deployed, you can configure application properties using Enterprise Manager Fusion Middleware Control. You can also configure some properties using the MBean Browser to change values in the ADF MBeans. For example, you can use Enterprise Manager Fusion Middleware Control to change the URL connection or WebService connection endpoints or seed the production credentials.

When you run the application, you can monitor performance data on the application modules, application module pooling, and task flows.

Part II
Basic Administration

This part describes how to deploy, manage, monitor, and configure applications developed for Oracle ADF using Fusion Middleware Control and WLST commands. It also describes the diagnostic tools used to diagnose Fusion Web applications problems.

Part II contains the following chapters:

2 Getting Started with Managing Oracle ADF

This chapter describes how to use Oracle Enterprise Fusion Middleware Control to perform ADF application configuration, monitor performance, and setting up logs. It also describes how to use the ADF-specific WebLogic Scripting Tool (WLST).
This chapter contains the following sections:

- Overview of ADF Administration Tools
- Getting Started Using Fusion Middleware Control to Manage ADF
- Using the WebLogic Scripting Tool
- Using the Oracle WebLogic Administration Console
- Using the Fusion Middleware Control MBean Browsers

Overview of ADF Administration Tools

Different administration tools are available to manage and configure ADF applications. You can use Enterprise Manager Fusion Middleware Control, WLST commands, or Oracle WebLogic Administration Console to perform these administration tasks.

Oracle offers the following primary tools for managing your Oracle Fusion Middleware installations:

- Oracle Enterprise Manager Fusion Middleware Control. See Getting Started Using Fusion Middleware Control to Manage ADF.
- The Oracle Fusion Middleware command-line tools. See Using the WebLogic Scripting Tool.
- Oracle WebLogic Server Administration Console. See Using the Oracle WebLogic Administration Console.
- The Fusion Middleware Control MBean Browser. See Using the Fusion Middleware Control MBean Browsers.

Getting Started Using Fusion Middleware Control to Manage ADF

Fusion Middleware Control is a Web browser-based, graphical user interface that you can use to monitor and administer Oracle Fusion Middleware. You can navigate to Oracle ADF pages in Fusion Middleware Control to view and configure different ADF-specific properties.

For information about navigating within Fusion Middleware Control and a description of its elements and main menus, see Navigating Within Fusion Middleware Control in the *Administering Oracle Fusion Middleware*.

To navigate to Oracle Application Development Framework pages in Fusion Middleware Control:

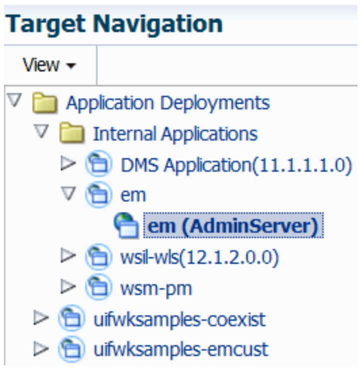
1. Enter the Fusion Middleware Control URL, which includes the name of the host and the port number assigned to Fusion Middleware Control during the installation. The following shows the format of the URL

   ```
   http://hostname.domain:port/em
   ```

   The port number is the number of the Administration Server of Oracle WebLogic Server By default, the port number is 7001.

2. Enter the Oracle Fusion Middleware administrator user name and password and click **Login**.
3. From the navigation pane, expand the farm and then Application Deployments, and select the ADF application.

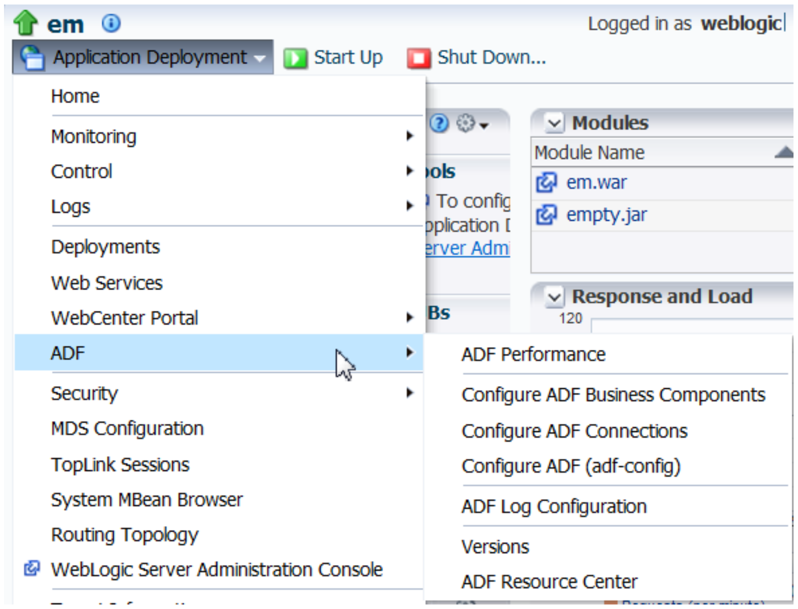
   Figure 2-1 Target Navigation

   The ADF home page displays.

4. Select the **Application Deployment** menu and then the **ADF** menu.

   Figure 2-2 ADF Menu

The **ADF** menu displays the following options.

Menu Option	Description
ADF Performance	Displays the Application Module Pool performance tab and the ADF Taskflows tab. For more information, see Viewing Application Module Pool Performance, and Viewing ADF Task Flow Performance.
Configure ADF Business Components	Configure the ADF Business Components. You can select configuration tabs for: • **Pooling and Scalability - Application Pool Properties** • **Pooling and Scalability - Connection Pool Properties** • **Core** • **Database Properties** • **Security Properties** For more information, see Modifying ADF Business Components Parameters.
Configure ADF Connections	Allows you create and edit connections that are available to ADF applications. For more information, see Modifying Connection Configurations.
Configure ADF (adf-config)	Allows you to configure ADF application properties using the System MBean Browser. For more information, see Modifying ADF Application Configurations Using MBeans.
ADF Log Configuration	Allows you to configure the logging levels for ADF Loggers. For more information, see Managing Log Files and Diagnostic Data chapter in *Administering Oracle Fusion Middleware*.
Versions	Allows you to find the version information of ADF runtime JAR files. For more information, see Finding Version Information of ADF Runtime JARs .
ADF Resource Center	Provides links to ADF resources such as documentation, knowledge base, forums, demos, samples, and software downloads.

Using the WebLogic Scripting Tool

WebLogic Scripting Tool (WLST) is a command-line scripting environment that provides a set of commands for ADF applications that are specific to WebLogic Server. You can use the ADF-specific WLST commands to administer ADF applications.

WLST is used to create, manage, and monitor Oracle WebLogic Server domains. WLST is a command-line based on Jython. ADF provides a set of custom WLST commands that you can use to perform functions specifically for ADF applications.

Custom ADF-specific WLST commands can be used with Maven.

For more information about using WLST, see Using the WebLogic Scripting Tool in *Understanding the WebLogic Scripting Tool* and Getting Started Using the Oracle WebLogic Scripting Tool (WLST) in *Administering Oracle Fusion Middleware*.

For reference information about the ADF WLST commands, see Working with WLST Commands for ADF Applications.

To access the ADF-specific WLST commands:

1. Go to the Oracle Common home directory for your installation, for example `/home/Oracle/Middleware/oracle_common`.

   For information about the Oracle Common home directory, see About the Oracle Home Directory in *Planning an Installation of Oracle Fusion Middleware*.
2. Start Oracle WebLogic Server.
3. Start WLST using the `WLST.sh/cmd` command located in the `oracle_common/common/bin` directory. For example:
   - `/home/Oracle/Middleware/oracle_common/common/bin/WLST.sh` (UNIX)
   - `C:\Oracle\Middleware\oracle_common\common\bin\wlst.cmd` (Windows)
4. Connect to the running WebLogic Server instance using the `connect()` command. For example, the following command connects WLST to the Admin Server at the URL `myAdminServer.example.com:7001` using the username/password credentials `my_username/my_password`:

   ```
   connect("my_username","my_password","t3://myAdminServer.example.com:7001")
   ```

For reference information about the ADF WLST commands, see Working with WLST Commands for ADF Applications.

Using the Oracle WebLogic Administration Console

Oracle WebLogic Server Administration Console provides a graphical user interface that you can use to configure, monitor, and manage the WebLogic Server instances, clusters, services, and security parameters.

You can use the Oracle WebLogic Administration Console to manage Oracle WebLogic Server domains. The console runs in a Web browser and has a graphical user interface.

For more information, see Getting Started Using Oracle WebLogic Server Administration Console in *Administering Oracle Fusion Middleware*.

Using the Fusion Middleware Control MBean Browsers

The Fusion Middleware Control MBean Browser is an Oracle Web application that is based on JMX MBean containers. You can use the MBean Browser to view and modify the values of MBeans for an application or a WebLogic Server.

Fusion Middleware Control provides a set of MBean browsers that allow to you view the MBeans for an application or for an Oracle WebLogic Server. You can also use the MBean browser to perform monitoring and configuration tasks.

For more information, see Getting Started Using the Fusion Middleware Control MBean Browsers in *Administering Oracle Fusion Middleware*.

3

Monitoring and Configuring ADF Applications Using Fusion Middleware Control

This chapter describes how to view ADF application performance. It also describes how to configure an ADF application's properties after it has been deployed to Oracle WebLogic Server. It describes how to use Oracle Enterprise Manager Fusion Middleware Control and its System MBean Browser to perform monitoring and configuration tasks. It also describes diagnosing problems with the Diagnostic Framework and monitoring metrics using DMS spy.
This chapter includes the following sections:

- Introduction to ADF Application Monitoring and Configuration
- Monitoring Performance Using Enterprise Manager Fusion Middleware Control
- Configuring Application Properties Using Fusion Middleware Control
- Configuring Application Properties Using the MBean Browser
- Editing Credentials Deployed with the Application

Introduction to ADF Application Monitoring and Configuration

You can deploy, manage, monitor, and configure applications developed with Oracle ADF in a WebLogic Server. Use Enterprise Manager Fusion Middleware Control to perform these tasks.

After you have deployed an ADF application to Oracle WebLogic Server, you can view the application performance and configure application properties on the server. You can use Enterprise Manager Fusion Middleware Control to perform these tasks.

Enterprise Manager Fusion Middleware Control offers a user interface for the performance tasks. Some configuration tasks can be performed either from a user interface or by configuring an MBean, as listed in Table 3-1.

Table 3-1 Configuration Tasks Using Fusion Middleware Control

Configuration tasks	Fusion Middleware Control UI	Fusion Middleware Control MBean Browser
ADF Business Components	Modifying ADF Business Components Parameters	Modifying ADF Business Components Configurations Using MBeans
ADF connections	Modifying Connection Configurations	Modifying ADF Connections Using MBeans

Table 3-1 (Cont.) Configuration Tasks Using Fusion Middleware Control

Configuration tasks	Fusion Middleware Control UI	Fusion Middleware Control MBean Browser
ADF application configuration		Modifying ADF Application Configurations Using MBeans
Metadata Services (MDS)		Modifying MDS Configuration Using MBeans
Active Data Service (ADS)		Modifying Active Data Service Configuration Using MBeans

By default, the post-deployment changes made using MBeans are stored in MDS with a layer name of `adfshare` and a layer value of `adfshare`. You can provide a specific layer name by specifying the `adfAppUId` property in the application's `adf-config.xml`.

The following example shows the `adf-properties-child` code in `adf-config.xml`.

```
<adf:adf-properties-child xmlns="http://xmlns.oracle.com/adf/config/properties">
     <adf-property name="adfAppUID" value="DeptApp.myApp"/>
</adf:adf-properties-child>
```

If you are moving data between MDS repositories (for example, from a test to a production system), use the MDS `exportMetadata` and `importMetadata` commands as described in the Managing the Metadata Repository of the *Administering Oracle Fusion Middleware* and in the Metadata Services (MDS) Custom WLST Commands of the *WLST Command Reference for Infrastructure Components*.

Monitoring Performance Using Enterprise Manager Fusion Middleware Control

You can monitor the performance of ADF applications. Use Fusion Middleware Control to view the application performance, application module pool performance, ADF task flow performance, and the ADF runtime JAR version.

You can monitor the performance of Oracle ADF applications using the Fusion Middleware Control, as described in the following topics:

- Understanding the Home Page
- Finding Version Information of ADF Runtime JARs
- Viewing Application Performance
- Viewing Application Module Pool Performance
- Viewing ADF Task Flow Performance

Understanding the Home Page

You can view performance information about application module pools and ADF task flows. Application module components can be used to support a unit of work which spans multiple browser pages.

You can:

1. View application module pool performance.
2. View task flow performance.

Finding Version Information of ADF Runtime JARs

You can use find the version information for ADF runtime JAR files and display the results in a table or export the information to an Excel file. You can narrow down your results by specifying a filter for each column.

To finding version information of runtime JARs:

1. From the navigation pane, expand **Application Deployments**, then click the application that you want to view the runtime JARs.
2. From the Application Deployment menu, choose **ADF > Versions**.

   The Log Configuration page displays.
3. You can filter the results for each column by entering the filter criteria into the input field above each column. For instance, if you want to display only JAR files associated with specification 2.2, enter "2.2" in the field above the **Specification Version** column and press Enter.
4. If you want to export the results table into an Excel file, click **Export to Excel**.

Viewing Application Performance

You can view performance information about application modules. Application module components can be used to support a unit of work which spans multiple browser pages.

To view application performance:

1. From the navigation pane, expand **Application Deployments**, then click the application that you want to view.
2. From the Application Deployment menu, choose **ADF > ADF Performance**.

   The ADF Performance page displays. It contains subtabs for viewing performance information about active application module pools and task flows.

Viewing Application Module Pool Performance

An *application module pool* is a collection of instances of a single application module type which are shared by multiple application clients. One application module pool is created for each root application module used by an ADF web application (ADF Business Components, ADF Controller, or ADF Faces) in each Java virtual machine where a root application module of that type is used by the ADF Controller layer.

To view application module pool performance:

1. From the navigation pane, expand **Application Deployments**, then click the application that you want to view pool performance.
2. From the Application Deployment menu, choose **ADF > ADF Performance**.
3. Click the **Application Module Pools** tab.
4. In the **Module** column, select an application module to display its details in the Application Module Pools table.

`No Data Available` displays in the Module column if an application has never run.

5. Click a module to display additional informations about the module, for example, Lifetime, State Management, Pool Use, and Application Module Pools Page.

   Use the Application Module Pools page to display active application module pools, a collection of application module instances of the same type. The Application Module Pools page:

   - Displays size and performance information about pool connections
   - Specifies settings that affect how application module pools behave
   - Specifies credential information for the application module pools

Viewing ADF Task Flow Performance

You can view performance information about task flows. Task flows provide a modular and transactional approach to navigation and application control. Task flows mostly contain pages that will be viewed, but they also can contain activities that call methods on managed beans, evaluate an EL expression, or call another task flow, all without invoking a particular page.

To view task flow performance:

1. From the navigation pane, expand **Application Deployments**, then click the application that you want to view task flow performance.
2. From the Application Deployment menu, choose **ADF > ADF Performance**.
3. Click the **Task Flows** tab.

   By default, Task Flow Performance charts on the tab display data for the preceding 15 minutes. To set a different interval, click the time at the top of the page or move the slider to another interval, for example, from 08:00 AM to 08:30 AM.
4. Click **Table View** below the following Task Flow Performance charts:
   - **Request Processing Time** - displays a window with the average request processing time for all ADF task flows that execute during the selected interval.
   - **Activated Taskflows** - displays a window with the number of active instances of each ADF task flow during the selected interval.

Configuring Application Properties Using Fusion Middleware Control

After you have deployed an ADF application to Oracle WebLogic Server, you can configure application properties on the server. Use Fusion Middleware Control to configure ADF application configuration parameters.

These configuration parameters are stored in ADF MBeans. Fusion Middleware Control provides a user interface to configure the ADF Business Components and ADF Connections MBeans. You can also use the System MBean Browser to directly access the underlying MBeans and configure their values. For more information about accessing the underlying MBeans, see Configuring Application Properties Using the MBean Browser.

You can use Fusion Middleware Control to configure ADF parameters, as described in the following topics:

- Modifying ADF Business Components Parameters
- Modifying Connection Configurations

Modifying ADF Business Components Parameters

You control the runtime behavior of an application module pool by setting appropriate configuration parameters. Fusion Middleware Control provides a UI to configure ADF Business Components, as described in this section. You can also configure the ADF Business Components MBeans directly using the generic MBean Browser, as described in Modifying ADF Business Components Configurations Using MBeans

To modify Business Components parameters:

1. From the navigation pane, expand **Application Deployments**, then click the application that you want to configure Business Components.
2. From the Application Deployment menu, choose **ADF > Configure ADF Business Components**.
3. Click an **Application Module.**
4. Click the **Pooling and Scalability**, **Core**, **Database**, or **Security** tabs to update configuration parameters.

   If the application module uses data sources, you can configure the data sources by clicking **Edit Datasource** from the **Core** tab.

The ADF Business Components configurations page is arranged with the following sections or tabs:

- Application Modules section
- Pooling and Scalability tab - Application Pool Properties
- Pooling and Scalability tab - Connection Pool Properties
- Core tab
- Database Properties tab
- Security Properties tab

Modifying Connection Configurations

A connection configuration contains information that a client application uses to identify the ADF application module's deployment scenario. You use Oracle Enterprise Manager Fusion Middleware Control to:

- Register and manage back-end services such as mail, discussion forums servers, and so on
- Register and manage external applications that users need access to while working with applications
- Register and manage any portlet producers that the application uses or that users may need access to

Fusion Middleware Control provides a UI to configure ADF connections, as described in this section. You can also configure the ADF connections MBean directly using the System MBean Browser, as described in Modifying ADF Connections Using MBeans.

You must already have deployed an Oracle ADF application and have Enterprise Manager Fusion Middleware Control available to access the application.

You must have MDS configured in your application before you can modify the ADF application and connection configurations. ADF connection attributes are persisted to MDS.

If you deployed an application to several nodes within a cluster, any ADF connection changes to a single node will be propagated to all the other nodes. MDS will store a single set of connection information for all versions of an application.

To modify connection configurations:

1. From the navigation pane, expand **Application Deployments**, then click the application that you want to configure connections.
2. From the Application Deployment menu, choose **ADF > Configure ADF Connections**.
3. In the **Connection Type** drop-down list, choose the type of connection you want to configure:
   - ADF BC Service
   - Enterprise Scheduling Service
   - URL
   - Web Service

   You cannot create an Essbase connection, however, you can edit an existing Essbase connection that was deployed with the application.
4. In the **Connection Name** field, enter a unique name for the connection configuration.
5. Click **Create Connection**.

   The ADF Connections Configuration page updates with a section where you can specify options for the connection type you chose.

Configuring Application Properties Using the MBean Browser

Fusion Middleware Control MBean Browser is a web application used to access and modify the values of MBeans for an ADF application deployed into Oracle WebLogic Server.

You can use the Fusion Middleware Control System MBean Browser to perform configuration tasks, as described in the following topics:

- Modifying ADF Application Configurations Using MBeans
- Modifying ADF Connections Using MBeans
- Modifying ADF Business Components Configurations Using MBeans
- Modifying MDS Configuration Using MBeans

- Modifying Active Data Service Configuration Using MBeans

Modifying ADF Application Configurations Using MBeans

You can modify ADF application configurations MBeans using the MBean Browser.

You must have MDS configured in your application before you can modify the ADF application and connection configurations. ADF application attributes are persisted to MDS.

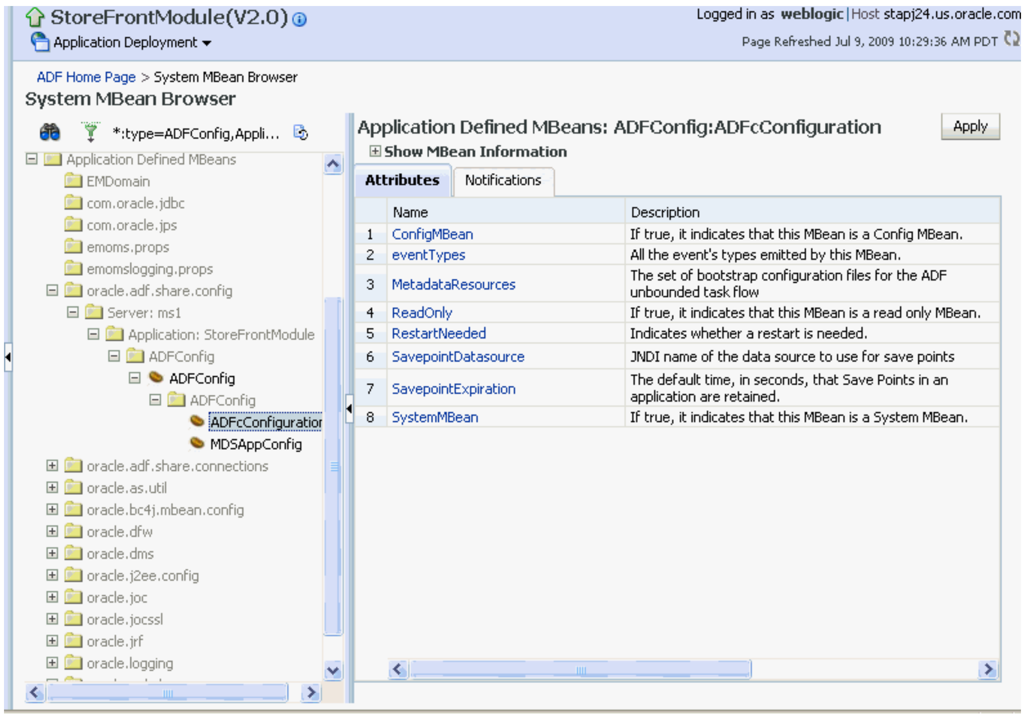
If you deployed an application to several nodes within a cluster, any ADF application configuration changes to a single node via an MBean will be propagated to all the other nodes. MDS will store a single set of ADF application configuration information for all versions of an application.

To modify ADF application configurations using the System MBean Browser:

1. From the navigation pane, expand **Application Deployments**, then click the application that you want to configure.
2. From the Application Deployment menu, choose **ADF > Configure ADF (adf-config)**.
3. In the left pane of the System MBean Browser, expand the parent ADF MBean **ADFConfig** and then the **ADFConfig** folder to expose the child ADF MBeans.

   You may see the child ADF MBeans **ADFcConfiguration** and **MDSAppConfig**.
4. In the left pane, select the **ADFcConfiguration** MBean, and in the right pane, select the attribute you want to view or modify.

Figure 3-1 ADFcConfiguration MBean Attributes



	Name	Description
1	ConfigMBean	If true, it indicates that this MBean is a Config MBean.
2	eventTypes	All the event's types emitted by this MBean.
3	MetadataResources	The set of bootstrap configuration files for the ADF unbounded task flow
4	ReadOnly	If true, it indicates that this MBean is a read only MBean.
5	RestartNeeded	Indicates whether a restart is needed.
6	SavepointDatasource	JNDI name of the data source to use for save points
7	SavepointExpiration	The default time, in seconds, that Save Points in an application are retained.
8	SystemMBean	If true, it indicates that this MBean is a System MBean.

5. Change the attribute value and click **Apply**.

6. In the left pane, select the parent ADF MBean **ADFConfig**.
7. In the right pane, click the **Operations** tab and click **save**.

   The new values you have edited are written to MDS after you click **save** from the parent MBean.

Modifying ADF Connections Using MBeans

You can modify ADF connection configurations MBean using the MBean Browser.

You can also modify ADF connections using the Fusion Middleware UI described in Modifying Connection Configurations.

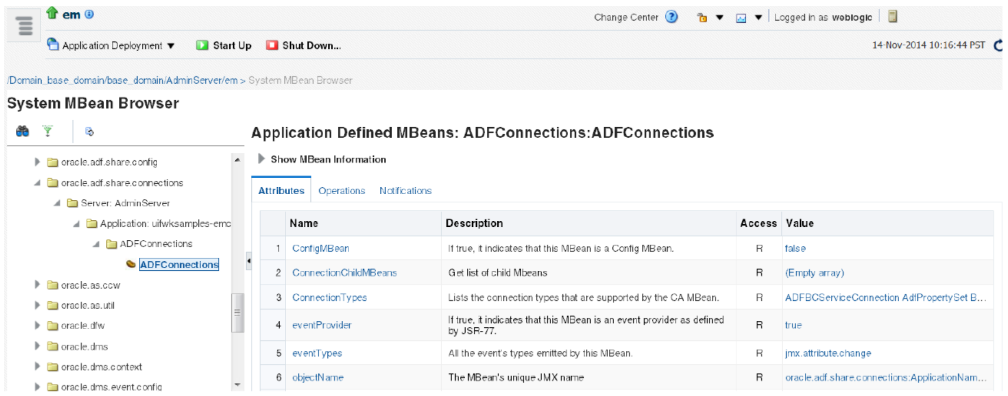
You must have MDS configured in your application before you can modify the ADF application and connection configurations. ADF application attributes are persisted to MDS.

If you deployed an application to several nodes within a cluster, any ADF connection changes to a single node via an MBean will be propagated to all the other nodes. MDS will store a single set of ADF application configuration information for all versions of an application.

To modify ADF connections configurations using the System MBean Browser:

1. From the navigation pane, expand **Application Deployments**, then click the application that you want to configure.
2. From the Application Deployment menu, choose **System MBean Browser**.
3. In the left pane of the System MBean Browser, navigate to the **ADFConnections** MBean. The MBean should be in **oracle.adf.share.connections > *server name* > *application name***.
4. In the left pane, select the ADF Connections MBean, and in the right pane, select the attribute you want to view or modify.

Figure 3-2 ADFConnections MBean Attributes



	Name	Description	Access	Value
1	ConfigMBean	If true, it indicates that this MBean is a Config MBean.	R	false
2	ConnectionChildMBeans	Get list of child Mbeans	R	(Empty array)
3	ConnectionTypes	Lists the connection types that are supported by the CA MBean.	R	ADFBCServiceConnection AdfPropertySet B...
4	eventProvider	If true, it indicates that this MBean is an event provider as defined by JSR-77.	R	true
5	eventTypes	All the event's types emitted by this MBean.	R	jmx.attribute.change
6	objectName	The MBean's unique JMX name	R	oracle.adf.share.connections:ApplicationNam...

5. Change the attribute value and click **Apply**.
6. In the right pane, click the **Operations** tab and click **save**.

   The new values you have edited are written to MDS after you click **save**.

Modifying ADF Business Components Configurations Using MBeans

You can modify ADF Business Components configurations MBeans using the MBean Browser. ADF Business Component configuration information are stored in MBeans that are specific for each application. Unlike ADF connections and ADF application configuration information which you can configure once for all versions of the same application, you will need to configure ADF Business Components for each version of the application.

You can also modify ADF Business Components configuration information using the Fusion Middleware UI described in Modifying ADF Business Components Parameters.

You must have MDS configured in your application before you can modify the ADF application and connection configurations. ADF application attributes are persisted to MDS.

If you deployed an application to several nodes within a cluster, any ADF Business Components changes to a single node via MBeans will be propagated to all the other nodes. MDS will store a single set of ADF application configuration information for all versions of an application.

To modify ADF Business Components configurations using the System MBean Browser:

1. From the navigation pane, expand **Application Deployments**, then click the application that you want to configure.
2. From the Application Deployment menu, choose **System MBean Browser**.
3. In the left pane of the System MBean Browser, navigate to the BC4J MBeans. These MBeans should be in **oracle.bc4j.mbean.share >** ***server name* >** ***application name***.
4. In the left pane, select the ADF Connections MBean, and in the right pane, select the attribute you want to view or modify.
5. Change the attribute value and click **Apply**.

Modifying MDS Configuration Using MBeans

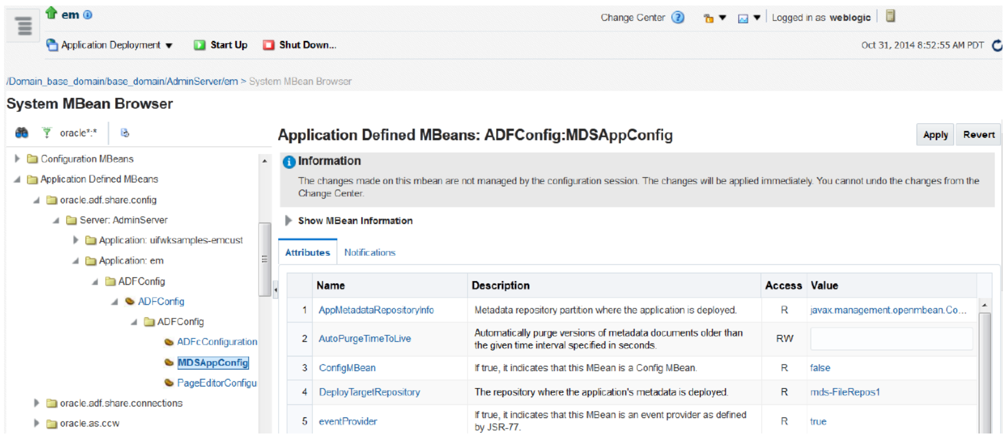
You can use the MBean Browser to perform advanced configuration of MDS parameters. For more information about configuring MDS using MBeans, see Changing the MDS Configuration Attributes Using Fusion Middleware Control in *Administering Oracle Fusion Middleware*.

You must already have deployed an Oracle ADF application and have Enterprise Manager Fusion Middleware Control available to access the application.

To modify MDS configurations using the System MBean Browser:

1. From the navigation pane, expand **Application Deployments**, then click the application that you want to configure.
2. From the Application Deployment menu, choose **MDS Configuration**.
3. Click **Configuration MBean Browser** or **Runtime MBean Browser.**
4. Select the MBean and the attribute you want to view or modify.

Figure 3-3 MDSAppConfig MBean Attributes



The changes made on this mbean are not managed by the configuration session. The changes will be applied immediately. You cannot undo the changes from the Change Center.

	Name	Description	Access	Value
1	AppMetadataRepositoryInfo	Metadata repository partition where the application is deployed.	R	javax.management.openmbean.Co...
2	AutoPurgeTimeToLive	Automatically purge versions of metadata documents older than the given time interval specified in seconds.	RW	
3	ConfigMBean	If true, it indicates that this MBean is a Config MBean.	R	false
4	DeployTargetRepository	The repository where the application's metadata is deployed.	R	mds-FileRepos1
5	eventProvider	If true, it indicates that this MBean is an event provider as defined by JSR-77.	R	true

5. Change the value and click **Apply**.
6. In the left pane, select the parent ADF MBean **ADFConfig**.
7. In the right pane, click the **Operations** tab and click **save**.

   The new values you have edited are written to MDS after you click **save** from the parent MBean.

Modifying Active Data Service Configuration Using MBeans

You can use Active Data Service (ADS) framework to control the runtime behavior of an Oracle ADF application and qualifying ADF Faces components so that whenever data changes on the server, the ADF Model layer notifies the component and the component rerenders the changed data.

You must already have deployed an Oracle ADF application and have Enterprise Manager Fusion Middleware Control available to access the application.

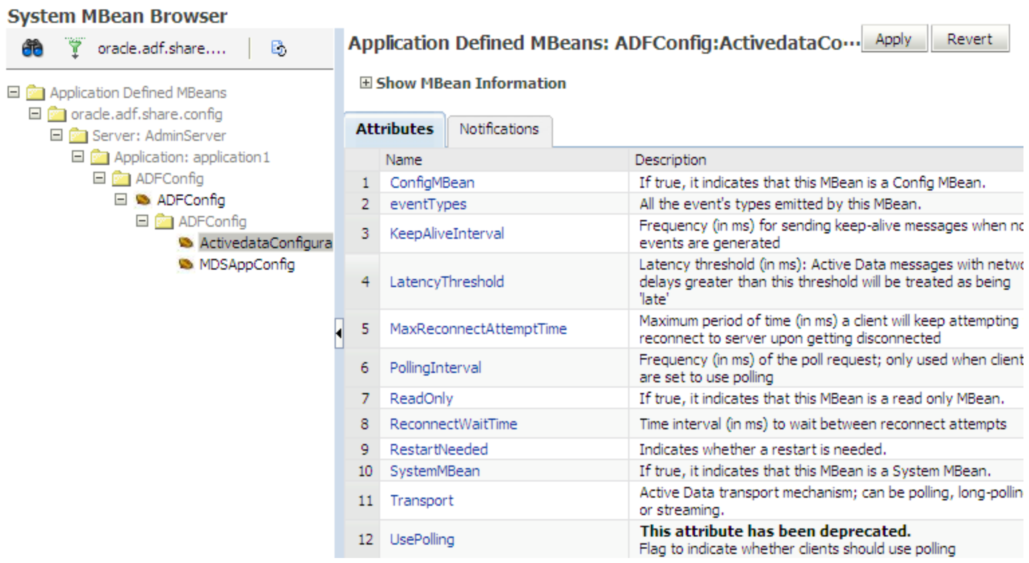
Note that the ADF Faces components of your application must be configured to use ADS. Additionally, if your application services do not support ADS, then your application must define a service proxy so that the components can display the data as it updates in the source. For details about ADS, see Using the Active Data Service in *Developing Fusion Web Applications with Oracle Application Development Framework*.

To modify Active Data Service configurations using the System MBean Browser:

1. From the navigation pane, expand **Application Deployments**, then click the application that you want to configure.
2. From the Application Deployment menu, choose **ADF > Configure ADF (adf-config)**.
3. In the left pane of the System MBean Browser, expand the parent ADF MBean **ADFConfig** and then the **ADFConfig** folder to expose the child ADF MBeans.

   You may see the child ADF MBeans **ActiveDataConfiguration** and **MDSAppConfig**.

Figure 3-4 ActiveDataConfiguration MBean Attributes



Application Defined MBeans: ADFConfig:ActivedataCo···

	Name	Description
1	ConfigMBean	If true, it indicates that this MBean is a Config MBean.
2	eventTypes	All the event's types emitted by this MBean.
3	KeepAliveInterval	Frequency (in ms) for sending keep-alive messages when nc events are generated
4	LatencyThreshold	Latency threshold (in ms): Active Data messages with netwc delays greater than this threshold will be treated as being 'late'
5	MaxReconnectAttemptTime	Maximum period of time (in ms) a client will keep attempting reconnect to server upon getting disconnected
6	PollingInterval	Frequency (in ms) of the poll request; only used when client are set to use polling
7	ReadOnly	If true, it indicates that this MBean is a read only MBean.
8	ReconnectWaitTime	Time interval (in ms) to wait between reconnect attempts
9	RestartNeeded	Indicates whether a restart is needed.
10	SystemMBean	If true, it indicates that this MBean is a System MBean.
11	Transport	Active Data transport mechanism; can be polling, long-pollin or streaming.
12	UsePolling	**This attribute has been deprecated.** Flag to indicate whether clients should use polling

4. In the left pane, select the **ActiveDataConfiguration** MBean, and in the right pane, select the attribute you want to view or modify.

Attribute	Description
Transport	The method by which data will be delivered to the client. Value values are: • `streaming (default)` • `polling` • `long-polling` For more information, see What You May Need to Know About Configuring an ADS Transport Mode in *Developing Fusion Web Applications with Oracle Application Development Framework*.
LatencyThreshold	Latency threshold in milliseconds. Active data messages with network delays greater than this threshold will be treated as being "late".
KeepAliveInterval	Frequency in milliseconds for sending keep-alive messages when no events are generated.
PollingInterval	When `transport` set to `polling`, frequency in milliseconds of the poll request.
MaxReconnectAttemptTime	Maximum period of time in milliseconds a client will attempt to reconnect the push channel upon getting disconnected
ReconnectWaitTime	Time interval in milliseconds to wait between reconnect attempts.

5. Change the attribute value and click **Apply**.
6. In the left pane, select the parent ADF MBean **ADFConfig**.
7. In the right pane, click the **Operations** tab and click **save**.

   The new values you have edited are written to MDS after you click **save** from the parent MBean.

Editing Credentials Deployed with the Application

Credentials can be provisioned, retrieved, modified, or deleted, but only by a user in the appropriate administration role. As an administrator, you can manage credentials using Fusion Middleware Control.

You can use Enterprise Manager Fusion Middleware Control to edit credentials that were deployed with an ADF application to the credential store. You can also create new credentials and delete existing credentials.

For ADF applications, the following considerations apply:

- The **Map** name is typically the `adfAppUId` property defined in the application's `adf-config.xml` file.
- The **Key** name is typically in the format `anonymous#`*`connection`*, where *`connection`* is the connection name.
- The **Credential Type** is **Generic** and it is modeled as a hash map of key-value pairs.

For more information, see Managing Credentials with Fusion Middleware Control in *Securing Applications with Oracle Platform Security Services*.

4

Deploying ADF Applications

This chapter describes how to deploy Oracle ADF applications packaged as an EAR file to a target application server. The focus is on deploying ADF applications for production and later stage testing. It shows you how to prepare the application server for deployment by installing the ADF runtime. It also describes some of the tools that can be used for deployment including Oracle Enterprise Manager Fusion Middleware Control, scripts, and Ant.
This chapter includes the following sections:

For information about deploying ADF applications for development, see the Deploying Fusion Web Applications chapter of the *Developing Fusion Web Applications with Oracle Application Development Framework*.

Introduction to Deploying ADF Applications

JDeveloper can be used to develop an ADF application and create an EAR archive file to deploy to the target. There are several tools and methods used by the administrators to deploy ADF applications to the target.

Deployment is the process of packaging application files and artifacts and transferring them to a target application server to be run. During application development using JDeveloper, developers can test the application using the Integrated WebLogic Server that is built into the JDeveloper installation, or they can use JDeveloper to directly deploy to a standalone application server.

After the application has been developed, administrators can deploy the application to production application servers. The tools that the administrators use for production-level deployment are:

- Oracle Enterprise Manager Fusion Middleware Control
- WebLogic Scripting Tool (WLST) commands
- Command scripts and Ant scripts
- Oracle WebLogic Administration Console

This chapter describes the tools and methods that administrators use to deploy ADF applications. For information about deploying ADF applications for development and testing purposes using JDeveloper, see Deploying Fusion Web Applications of the

Developing Fusion Web Applications with Oracle Application Development Framework.

If your application uses customizations, you may need to set up the MDS repository in the application server. For more information about MDS, see Managing the Metadata Repository of the *Administering Oracle Fusion Middleware*.

Note:

Developers, Test, and QA personnel may also use these tools and the methods in this chapter to deploy ADF applications to staging application servers.

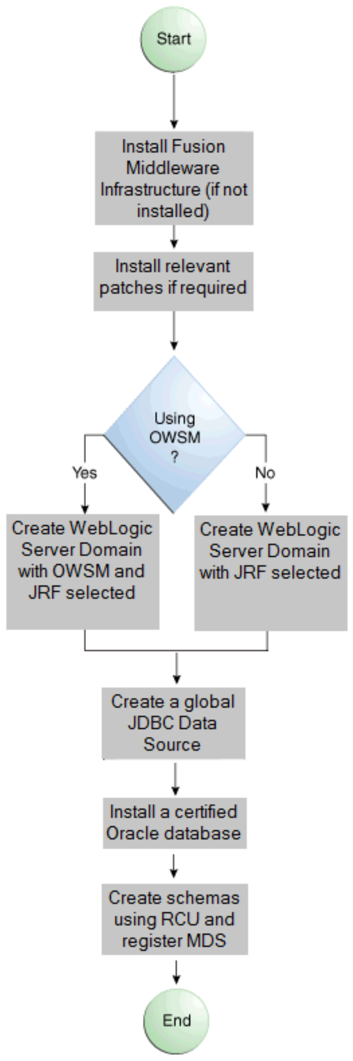
Preparing the Standalone Application Server for Deployment

To run ADF applications, you must install the standalone application server with the ADF runtime. You can include the ADF runtime during a new application server installation or you can install the ADF runtime into an existing application server installation.

Figure 4-1 shows the flow diagram for preparing a standalone application server for deployment. Note the following definitions used in the diagram:

- OWSM: Oracle Web Services Manager
- JRF: Java Required Files
- RCU: Repository Creation Utility
- MDS: Metadata Store

Figure 4-1 Preparing the Application Server Flow Diagram


Start
Install Fusion Middleware Infrastructure (if not installed)
Install relevant patches if required
Using OWSM ?
Yes
No
Create WebLogic Server Domain with OWSM and JRF selected
Create WebLogic Server Domain with JRF selected
Create a global JDBC Data Source
Install a certified Oracle database
Create schemas using RCU and register MDS
End

For WebLogic Server, the following points apply:

- After you install the Oracle Fusion Middleware Infrastructure, you can create a new WebLogic Server domain or you can extend an existing WebLogic Server domain for Oracle ADF.
- If the Managed Servers are on a different host than the Administration Server, you must perform additional configuration tasks for the Managed Servers to enable them to host ADF applications.
- An ADF application will use either a JDBC data source or a JDBC URL to access its data. You can configure WebLogic Server with the data source using the Oracle WebLogic Server Administration Console.
- For additional information about OPatch for patching your existing Oracle Fusion Middleware environment, see Patching Your Environment Using OPatch of the *Patching with OPatch*.

- For additional information about creating and loading the schemas, see Creating Schemas in *Creating Schemas with the Repository Creation Utility*.
- For the latest information about Oracle Application Development Framework and Oracle JDeveloper, see the *Oracle JDeveloper Release Notes* on Oracle Technology Network.

Installing the ADF Runtime to the Application Server Installation

The application server requires the ADF runtime to run ADF applications.

Installing the ADF runtime is not required if you are using JDeveloper to run applications in Integrated WebLogic Server.

For WebLogic Server, you can install the ADF runtime using the following installers:

- Oracle Fusion Middleware Application Developer Infrastructure Installer: Installs the ADF runtime and Oracle Enterprise Manager. You should use the Oracle Fusion Middleware Infrastructure Installer if you want to use Oracle Enterprise Manager to manage standalone ADF applications (without Oracle SOA Suite or Oracle WebCenter Portal components). For more information, see Planning the Oracle Fusion Middleware Infrastructure Installation of the *Installing and Configuring the Oracle Fusion Middleware Infrastructure*.

> **Note:**
>
> The Oracle JDeveloper Installer can also be used to install the ADF runtime to the application server installation. However, it does not include all the components that are typically needed for production and full test environments. Therefore, this installer should not be used for anything other than for development purposes. For information about the obtaining and using the installer, see Installing the Oracle JDeveloper Software of the *Installing Oracle JDeveloper*.

Using the Oracle Fusion Middleware Infrastructure Installer

You can use the Oracle Fusion Middleware Infrastructure Installer to install the ADF runtime and Enterprise Manager.

Install Oracle WebLogic Server. You must also have obtained the Oracle Fusion Middleware Infrastructure Installer.

Use the instructions in Preparing for an Oracle Fusion Middleware Installation of the *Planning an Installation of Oracle Fusion Middleware* to obtain the software, start the installer, and to complete the installation.

In the installer you will perform several tasks including:

- Adding any software updates
- Selecting the WebLogic Server directory for installation
- Verifying installation information

After you have installed the ADF runtime, follow the instructions in Creating and Extending Oracle WebLogic Server Domains to use the Oracle Fusion Middleware Configuration Wizard to create or extend the Oracle WebLogic Server domain.

Installing Relevant Patches

You may need to install patches and other updates to your software distribution. For more information, see Patching Your Environment Using OPatch of the *Patching with OPatch*.

To enable existing sessions to continue to run while OPatchAuto updates the servers, use `-plan wls-zdt-rollout` . Note that Oracle WebLogic Server domain must be configured with a Load Balancer like Oracle HTTP Server, for example. For more information, see ZDT Patching Restrictions.

If Oracle WebLogic Server Domain is not configured with Load Balancer, you can use the `-plan` rolling OpatchAuto option. Note that this option is not related to zero downtime technology and causes existing sessions to be terminated as patching proceeds.

Creating and Extending Oracle WebLogic Server Domains

You need to create and configure the Oracle WebLogic Server domain to accept ADF applications. If you do not already have a domain, you need to create one. If you already have a domain, you must extend the domain before it can run ADF applications.

If you are using Managed Servers to run your applications, you may need to configure your Managed Server. For more information about configuring a Managed Server on Oracle WebLogic Server, see Managed Servers of the *Creating WebLogic Domains Using the Configuration Wizard*.

If you are setting up Managed Servers for ADF where the Managed Servers are on the same host as the Administration Server, follow the instructions described in this section.

If you are setting up to deploy to Managed Servers that are on a different host than the Administration Server, perform the additional steps described in Setting Up Remote WebLogic Managed Servers for Oracle ADF.

Creating an Oracle WebLogic Server Domain for Oracle ADF

It may be helpful to have an understanding of the options that are available to you when you create a WebLogic Server domain for Oracle ADF. For more information, see Preparing the Standalone Application Server for Deployment.

To create a new Oracle WebLogic Server domain:

1. Start the Oracle Fusion Middleware Configuration wizard as described in Configuring Your WebLogic Domain of the *Installing and Configuring the Oracle Fusion Middleware Infrastructure*.

   Follow the directions as described in that guide but consider the following steps.
2. In the Welcome page, select **Create a New Domain** and click **Next**.
3. In the Templates page, select **Create Domain Using Product Templates**.

The option **Basic WebLogic Server Domain - 12.2.1 [wlserver]** is already selected.

Select **Oracle JRF - 12.2.1 [oracle_common]**. If you are using Oracle Web Services, select **Oracle WSM Policy Manager 12.2.1 [oracle_common]** and click **Next**.

4. Continue to follow the directions in *Installing and Configuring the Oracle Fusion Middleware Infrastructure*.

Extending the Oracle WebLogic Server Domain for Oracle ADF

It may be helpful to have an understanding of the options that are available to you when you extend the WebLogic Server domain for Oracle ADF. For more information, see Preparing the Standalone Application Server for Deployment.

You will need to create an Oracle WebLogic Server domain with the ADF runtime installed.

To extend an Oracle WebLogic Server domain for ADF:

1. Start the Oracle Fusion Middleware Configuration wizard as described in Configuring Your WebLogic Domain of the *Installing and Configuring the Oracle Fusion Middleware Infrastructure*.

   Follow the directions as described in that guide but consider the following steps.
2. In the Welcome page, select **Update an existing domain**.
3. Select the location of the domain you want to update for Oracle ADF, and click **Next**.
4. In the Templates page, select **Update Domain Using Product Templates**.

   The option **Basic WebLogic Server Domain - 12.2.1 [wlserver]** is already selected.

   Select **Oracle JRF - 12.2.1 [oracle_common]**. If you are using Oracle Web Services, select **Oracle WSM Policy Manager 12.2.1 [oracle_common]** and click **Next**.
5. Continue to follow the directions in *Installing and Configuring the Oracle Fusion Middleware Infrastructure*.

This configures the rest of the runtime `.jar` files using the `manifest` file.

Note:

Your application's EAR file must have a `weblogic-application.xml` file containing a reference to the `adf.oracle.domain` shared library.

You can now start Oracle WebLogic Server by running the command-line script `ORACLE_HOME\user_projects\domains\`*`domain_name`*`\bin\startWebLogic.cmd`, and you can stop the server using the `stopWebLogic.cmd` script in the same directory. For Linux platforms, use `\bin\startWebLogic.sh` and `stopWebLogic.sh` respectively.

Access the Oracle WebLogic Server Administration Console using the URL `http://localhost:7001/console`.

Setting Up Remote WebLogic Managed Servers for Oracle ADF

If the WebLogic Managed Servers are on a different host than the Administration Server, you need to perform additional steps.

You will need to set up Managed Servers for Oracle ADF on the host with the Administration Server, pack the JRF template, copy it to the remote host, and unpack the template.

It may be helpful to have an understanding of the options that are available to you when you create remote WebLogic Managed Servers for Oracle ADF. For more information, see Preparing the Standalone Application Server for Deployment.

For more information about using `pack` and `unpack` commands to set up managed servers, see pack and unpack Command Reference of the *Creating Templates and Domains Using the Pack and Unpack Commands*.

You will need to complete this task:

1. Set up Managed Servers for Oracle ADF on the host with the Administration Server.
2. Pack the JRF template.
3. Copy the template to the remote host.
4. Unpack the template

To set up remote Managed Servers for Oracle ADF:

1. Use the Oracle Installer for JDeveloper to install Oracle WebLogic Server installations on both the local and remote hosts, if not already installed. If you are not installing JDeveloper Studio, you need to select the **Application Development Framework Runtime** option in the installer. The local host is the host with the Administration Server.

   Or, if there are existing Weblogic Server installations, use the Oracle Installer for JDeveloper to install the ADF runtime into the WebLogic Server installations on both hosts by selecting the **Application Development Framework Runtime** option. For more information on installation, see Installing the ADF Runtime to the Application Server Installation.
2. Run the Oracle Fusion Middleware Configuration Wizard to create a new Oracle WebLogic Server domain. In the wizard, select the **Oracle JRF** option, as described in Creating an Oracle WebLogic Server Domain for Oracle ADF.
3. On the local host, run the Oracle Fusion Middleware Configuration Wizard to create Managed Servers.
4. On the local host, start the Administration Server and the Managed Server.

   For example,

   ```
   cd ORACLE_HOME/user_projects/domain/base_domain/bin
   ./startWeblogic.sh
   ./startManagedWebLogic.sh ManagedServer_1 http://localhost:7001
   ```
5. On the local host, pack the Managed Server configuration information into a JAR and then copy the JAR to the remote host. This JAR contains the JRF template information.

For example,

```
cd ORACLE_HOME/oracle_home/common/bin

./pack.sh -managed=true -domain=../../../user_projects/domains/base_domain
     -template=../../../base_domain_managed.jar -template_name=
     "Base Managed Server Domain"

cp ../../../base_domain_managed.jar remote_machine_ORACLE_HOME/
```

6. On the remote host, unpack the Managed Server configuration JAR.

   For example,

   ```
   cd ORACLE_HOME/oracle_common/common/bin
   ./unpack.sh -domain=../../../user_projects/domains/base_domain
            -template=../../../base_domain_managed.jar
   ```

   If the Managed Server was created after the domain was, you must delete the entire domain configuration directory of the Managed Server before running `unpack`.

7. On the remote host, start the Node Manager.

   For example,

   ```
   cd ORACLE_HOME/wlserver_10.3/server/bin
   ./startNodeManager.sh
   ```

8. On the remote host, if the Managed Server was not created with the JRF template applied, run the `applyJRF` WLST command to extend the Managed Server with the JRF template.

   Also, if the Managed Server was created after the domain was, you must delete the entire domain configuration directory of the Managed Server before running `applyJRF`.

9. On the both hosts, start the Managed Servers.

   For example,

   ```
   cd ORACLE_HOME/user_projects/domains/base_domain/bin./startManagedWebLogic.sh
   ManagedServer_2 http://<adminServerHost>:7001
   ```

Creating a JDBC Data Source for Oracle WebLogic Server

Use the Oracle WebLogic Server Administration Console to set up a JDBC data source in the WebLogic Server instance for your applications.

It may be helpful to have an understanding of the options that are available to you when you create a JDBC data source. For more information, see Preparing the Standalone Application Server for Deployment.

To configure Oracle WebLogic Server for a JDBC data source:

1. Start Oracle WebLogic Server (if not already started) by choosing **Oracle Fusion Middleware > User Projects > *Domain* > Start Admin Server for WebLogic Server Domain** from the Windows **Start** menu.

   For Linux, log in as the root user and navigate to:

   ```
   <ORACLE_HOME>/user_projects/domains/MYSOADomain/bin
   ```

   Run the following command:

```
./startWebLogic.sh
```

Or, from the Application Server Navigator, right-click an Oracle WebLogic Server instance and choose **Launch Admin Console**.

2. Start the Oracle WebLogic Server Administration Console by choosing **Oracle Fusion Middleware > User Projects > *Domain* > Admin Server Console** from the Windows **Start** menu.
3. Log in to the Oracle WebLogic Server Administration Console.
4. In the WebLogic Server Administration Console page, select **JDBC > Data Sources**.
5. Click **New**.
6. In the JDBC Data Source Properties page:
   - In the **Name** field, enter the name of the JDBC data source.
   - In the **JNDI** field, enter the name of the connection in the form `jdbc/`*`connection`*`DS`.
   - For the **Database Type**, select **Oracle**.
   - For the **Database Driver**, select **Oracle Driver (thin)**, and click **Next**.
7. In the Transactions Options page, accept the default options and click **Next**.
8. In the Connection Properties page:
   - For **Database Name**, enter the Oracle SID. For example, `orcl`.
   - For **Host Name**, enter the machine name of the database.
   - Enter the port number used to access the database.
   - Enter the user name and password for the database and click **Next**.
9. In the Test Database Connection page, click **Test Configuration** to test the connection.
10. In the Select Targets page, select the server for which the JDBC data source is to be deployed.
11. Click **Finish**.

Once the data source has been created in Oracle WebLogic Server, it can be used by an application module.

Deploying Using Oracle Enterprise Manager Fusion Middleware Control

You can use Oracle Enterprise Manager Fusion Middleware Control to deploy the EAR file created in JDeveloper. Fusion Middleware Control is a web browser-based, graphical user interface that you can use to monitor and administer a server farm.

For more information about deploying using Fusion Middleware Control, see Deploying Oracle ADF Applications in *Administering Oracle Fusion Middleware*.

Deploying Using Scripting Commands

Applications or modules can be deployed from JDeveloper without starting the JDeveloper IDE. You can run WLST commands from the command line or sequence them in scripts to run as a batch.

Before deploying from the command line, there must be deployment profiles for the application (EAR) or project (JAR or WAR). JDeveloper creates these deployment profiles automatically for certain types of applications, but before using commands for deployment, it is important to verify that the deployment profile(s) exist. To verify that the profiles exist, choose the **Deployment** node from either the Application Properties or Project Properties dialogs in JDeveloper. For more information about deployment profiles, see How to Create Deployment Profiles in *Developing Fusion Web Applications with Oracle Application Development Framework*.

JDeveloper can also be used to deploy an application's EAR, WAR, or JAR files. The same scripts that are used for deployment via a command line are also used to deploy via JDeveloper, but JDeveloper creates the syntax and provides a user interface for the deployment.

There are specific WLST commands (WebLogic) for working with ADF applications. For a list of these commands, see Working with WLST Commands for ADF Applications.

For more information about using WLST scripts, see the *WLST Command Reference for Infrastructure Components*.

Deploying Using Scripts and Ant

You can deploy the application using commands and scripts. You create a script to deploy the application using the `ojdeploy` command and use the `ojaudit` command to audit projects, workspaces, or source files of the application.

You can also set up the script to run automatically, for instance, whenever a developer checks in new changes.

`ojdeploy` scripts and Ant scripts can be used together or separately:

1. Create an `ojdeploy` script to compile, package, and deploy the application.
2. Create an `ojdeploy` script to compile and package the application. Then use an Ant script (such as `WLDeploy`) to deploy the application.
3. Create an Ant script to compile, package, and deploy the application. The Ant does not need to use `ojdeploy`.

For more information about the `ojdeploy` and `ojaudit` commands, see the JDeveloper online help.

You can deploy to most application servers from JDeveloper, or use tools provided by the application server vendor. You may also use Ant to package and deploy applications. The `build.xml` file, which contains the deployment commands for Ant, may vary depending on the target application server.

For deployment to other application servers, see the application server's documentation. If your application server does not provide specific Ant tasks, you may be able to use generic Ant tasks. For example, the generic `ear` task creates an EAR file for you.

For information about Ant, see `http://ant.apache.org`.

Deploying Using the Application Server Administration Tool

JDeveloper can be used to develop an ADF application and create an EAR archive file to deploy to the target. Use Oracle WebLogic Server Administration Console to deploy the EAR file to the target.

For Oracle WebLogic Server, you can use the Oracle WebLogic Server Administration Console to deploy the EAR file created in JDeveloper.

For more information, see Introduction and Roadmap of the *Deploying Applications to Oracle WebLogic Server*.

Deploying Using Oracle Java Cloud Service

There are different tools and options to deploy ADF applications to a WebLogic Server instance in Oracle Java Cloud Service. You can use JDeveloper, WebLogic Server Administration Console, or Oracle Enterprise Manager Fusion Middleware Control to deploy ADF applications.

You can deploy an application to a WebLogic Server instance in Oracle Java Cloud Service using any option listed below:

- from Oracle JDeveloper through a secure shell (SSH) tunnel
- from WebLogic Server Administration Console through a menu on the Oracle Java Cloud Service cloud portal
- from Oracle Enterprise Manager Fusion Middleware Control for WebLogic Server through a menu on the Oracle Java Cloud Service cloud portal
- from WebLogic Scripting Tool through an SSH client

For more information, see the documentation provided with Oracle Java Cloud Service.

5 Diagnostic Tools

This chapter describes the diagnostic and monitoring tools you can use to help you develop and diagnose Fusion web applications problems.
This chapter includes the following sections:

Introduction to Monitoring and Diagnostic Tools

Oracle Fusion Middleware and Oracle ADF provide tools you can use to monitor and diagnose issues with your application.

The tools and procedures include:

- Click History
- Loggers
- Diagnostics Framework
- DMS metrics

Click History

Click History is a Fusion Middleware diagnostics feature that continuously reports the physical clicks on UI elements such as buttons and tabs, scrolling events, and graph draws. Use this feature to diagnose the cause of the incidents.

Click History captures ADF Faces events generated by user activities on ADF Faces components (for instance, clicking an ADF command button from a web browser). The data is continuously captured and stored in a circular buffer in memory. The buffer size is adjustable. When a diagnostic incident occurs, the data in the buffer is automatically written to disk as part of the diagnostic data. For more information about diagnostic incidents, see the Diagnosing Problems chapter of *Administering Oracle Fusion Middleware*.You do not need to configure Click History to capture the "click" events, it is automatically enabled by default. You can use Click History data to help diagnose problems. It is particularly useful to help determine the last user action that had caused the failure or problem.

The Click History record provides information about the UI component, the view, the region, the user, the Diagnostic Session Id (DSID), and other key application data.

Click History captures its data using ADF EUM (End User Monitoring) services It has a circular buffer that is optimized specifically for EUM data. The data is packed in a single contiguous memory segment in the Java heap. It has minimal impact to Java Garbage Collection and its memory usage is strictly controlled. The default buffer size of 860 KB can hold approximately1,500 "click" events, which should be sufficient for incident analysis.

How to Obtain Click History

Click History is delivered as part of the default JRF Template. When you install Fusion Oracle Middleware or Oracle ADF, Click History will be installed as part of the Fusion Middleware framework.

How to Enable Click History for an Application

Click History is installed as part of ADF. However, the actual logging of Click History messages are performed by a separate thread independent of ADF, therefore, Click History imposes insignificant overhead for ADF.

Click History is enabled on an application by application basis. You enable an application for Click History by adding entries to its `weblogic.xml` and `web.xml` configuration files. For instructions, see Enabling the Application for RUEI and Click History in *Developing Fusion Web Applications with Oracle Application Development Framework*.

How to Enable and Disable Click History in WebLogic Server

After you have enabled Click History for an application, you can enable it for a specific WebLogic server instance by setting the `oracle.clickhistory.EUM` logger to the `NOTIFICATION` level.

You can a WLST setLogLevel command or the Enterprise Manager Cloud Control Support Workbench to set the logging level.

You do not need to restart the server for this to take effect.

To enable Click History using WLST, run the following command:

```
setLogLevel(logger='oracle.clickhistory.EUM', target='server_name', level='INFO',
persist='1')
```

To disable Click History using WLST, run the following command:

```
setLogLevel(logger='oracle.clickhistory.EUM', target='server_name', level='OFF',
persist='1')
```

For information on how to use Fusion Middleware Cloud Control Support Workbench to set the log level to enable Click History, see the Managing Log Files and Diagnostic Data section in *Administering Oracle Fusion Middleware*.

How to Adjust Click History Capacity

The Click History buffer size is defined in the `logging.xml` file. The default value is 860 KB or 880640 bytes. You can use the WLST `configureLogHandler` command to configure the buffer size (in bytes). You will need to restart the server for the change to take effect.

For example:

```
configureLogHandler(name="apps-clickhistory-handler", propertyName="bufferSize",
propertyValue="100000000", target="your_server_name")
```

You can find the existing buffer size using the WLST `listLogHandlers` command.

For example:

```
listLogHandlers(name="apps-clickhistory-handler")
```

The command returns a list of information including `bufferSize` in bytes.

For example:

```
bufferSize=880640
```

How to Dump Click History Data Explicitly

When an incident occurs, Click History automatically dumps its data into a file. However, you can also access Click History data at any time using the WLST `executeDump` command.

To dump Click History data explicitly:

```
executeDump(name="odl.quicktrace", args={"handlerName" : "apps-clickhistory-
handler"}, server="CRMCommonServer_1")
```

The Click History output file is a `QuickTrace` dump file. It has the same format as the ODL log file. This format can be parsed using the ODL `LogQuery` API as well as using the Enterprise Manager Cloud Control Support Workbench log query feature.

The Click History dump file format convention is:

```
odl_quicktrace<unique ID>_i<incident ID>.app-clickhistory-handler.dmp
```

The suffix is the `QuickTrace` handler name defined in the server's `logging.xml`.

For example:

```
odl_quicktrace502_i273.apps-clickhistory-handler.dmp
```

A sample dump file output is shown in the following example.

```
Sample Click History Dump
[2013-10-08T14:56:12.664-07:00] [AdminServer] [NOTIFICATION] [SOC]
[oracle.clickhistory.EUM] [tid: 34] [userId: CASH_MANAGER]
[ecid:a3da34f168f6c0de:-7bd21c26:14199e687e9:-8000-00000000000033d3,0]
[CH_RTY: oracle.adf.rich.Combobox ListOfValues]
[CH_CST: 1381269371002] [DSID: 0000K6PzLBrFS8G5Iz9Dif1IL7HS00000A]
[CH_CID: pt1:USma:0:MAnt2:1:pt1:Manua1:0:mrpanel1:mrbankAccountNameId]
[CH_VID: /ManualReconPage] [APP: PayablesApp]
[CH_CMP: oracle.adf.RichInputComboboxListOfValues]
[CH_CNM: Bank Account]
[CH_TYP: lovInternal
[CH_WID: 114iztpu6s_6]
```

```
[CH_RVD: /ManualReconLocalAreaFlow/ManualReconLocalAreaPF]
[CH_EID: a3da34f168f6c0de:-7bd21c26:14199e687e9:-8000-00000000000033d3]
[CH_FAM: org.apache.myfaces.trinidad.Input]

[2013-10-08T14:56:12.664-07:00] [AdminServer] [NOTIFICATION] [EOC]
[oracle.clickhistory.EUM] [tid: 34] [userId: CASH_MANAGER]
[ecid: a3da34f168f6c0de:-7bd21c26:14199e687e9:-8000-000000000000339c,0]
[CH_CET: 1381269364245]
[CH_CST: 1381269363617]
[APP: PayablesApp]
[DSID: 0000K6PzLBrFS8G5Iz9Dif1IL7HS00000A]
[CH_TTT: 628]
[CH_EID: a3da34f168f6c0de:-7bd21c26:14199e687e9:-8000-000000000000339c]
```

How to Interpret Click History Records

The Click History dump file is an ordinary ASCII file in the format of a log file. You can use any ASCII viewer to view its contents, but it would be best to use log file viewing tools to take advantage of the search and formatting functions they offer. The Click History messages written to the log file are logged by a separate thread that is independent of ADF.

The log file is located at:

```
${domain.home}/servers/${weblogic.Name}/logs/${weblogic.Name}-clickhistory.log
```

The Click History log data is automatically captured as part of a diagnostic incident. If you have an incident that includes Click History data, you can use the incident's `readme.txt` file to find the ECID (ExecutionContext ID), DSID (Diagnostics Session ID), user ID, and thread ID of the incident.You can also query Click History records based on these diagnostics flags.

The following information are logged:

- Region/View Id
- Component Id
- Server time
- Network time
- Browser render time

By default, a pair of messages are logged for each click. The message Id for the first message is `SOC` (Start of Click). The second message `EOC` (End of Click) will not be logged until the subsequent click occurs. Timing details, such as `TTT` (Total Time). can be found only in the second message.

Table 5-1 lists the Click History attributes and their corresponding ADF UserActivityInfo fields.

Table 5-1 Click History Attribute Mapping

Click History Attributes	ADF UserActivityInfo fields
CH_CST	`userActivityInfo.getClientStartTime()`
CH_CET	`userActivityInfo.getClientEndTime()`

Table 5-1 (Cont.) Click History Attribute Mapping

Click History Attributes	ADF UserActivityInfo fields
CH_VID	`userActivityInfo.getViewId()`
CH_WID	`userActivityInfo.getWindowId()`
CH_CID	`eventInfo.getComponentClientId()`
CH_CNM	`eventInfo.getComponentDisplayName()`
CH_FAM	`eventInfo.getComponentFamily()`
CH_CMP	`eventInfo.getComponentType()`
CH_RVD	`eventInfo.getRegionViewId()`
CH_RNM	`eventInfo.getRegionViewName()`
CH_RTY	`eventInfo.getRendererType()`
CH_TYP	`eventInfo.getType()`
CH_EID	`userActivityInfo.getContextId()`
CH_PEI	`userActivityInfo.getParentContextId()`
CH_TTT	Total Time = CH_CET - CH_CST
CH_RRT	`userActivityInfo.getResponseReceivedTime()`
CH_PRT	`userActivityInfo.getParentResponseReceivedTime()`
CH_RNT	Rendering Time = CH_CET - CH_RRT

Some of the viewing tools you can use to show the log file contents are:

- Enterprise Manager Cloud Control Support Workbench

  You can use Enterprise Manager Cloud Control Support Workbench to view and sort the log information.
- WLST command

  You can also use WLST command `displayLog` to view the record
- Any text editor such as `vi`.

For example, search all Click History records with a matching DSID in order to return a list of the recent "click" events of a HTTP session.

For example:

```
[2013-01-21T20:50:16.403-08:00] [ProcurementServer_1] [NOTIFICATION] []
[oracle.clickhistory.EUM] [tid: 44] [userId: FUSION]
[ecid: 004ox_GMXwy6aMGpIws1yf00018d000gvb,0:2:8:2:2]
[CLICK_COMPTYPE: oracle.adf.RichCommandButton] [APP: ProcurementApp#V2.0]
[DSID: 0000JlW_fz5Fw0HpIsO5yf1GzEoI000Dh3]
[CLICK_VIEWID: /adfp-portlet-bridge-container/container-view]
[CLICK_REGIONVIEWID: [/BrowseCategoryFlow/BrowseCategory,
 /BrowseCategoryMainAreaFlow/BrowseCategoryMainArea,
 /FSGenericTaskFlow/DynamicRegionContainer]]
[CLICK_STARTTIME: 1358830216656]
[CLICK_RENDERTYPE: oracle.adf.rich.Button]
[CLICK_TYPE: action]
[CLICK_COMPCLIENTID:_jpfcpncuivr__ns1588903364_j_id__
```

```
 ctru0:fragmentRegionStatic:0:dynamicRegion1:0:pt1:Brows1:0:
 ManageCategoryHierarchyPanel:ATT1:deleteWarningYES]
[CLICK_WINDOWID: k4coiwdvi_1]
[CLICK_REGIONVIEWNAME: [jpfcpncuivr_ns1588903364_j_id
 _ctru0:fragmentRegionStatic:0:dynamicRegion1:0:pt1:Brows1, _jpfcpncuivr
 _ns1588903364_j_id_ctru0:fragmentRegionStatic:0:dynamicRegion1, _jpfcpncuivr
 _ns1588903364_j_id__ctru0:fragmentRegionStatic]]
[CLICK_COMPDISPLAYNAME: Yes]
```

How to Get Click History DMS Metrics

You can determine Click History's buffer utilization and capacity by inspecting the DMS metrics. You can execute a WLST command to display the metrics. If multiple QuickTrace handlers are configured, look for the handler with this name: `apps-clickhistory-handler`.

For example:

```
wls:/base_domain/serverConfig>  displayMetricTables('QuickTraceInfo',
 servers=['your_server'])

QuickTraceInfo

Host:   xyz.com
Name:   apps-clickhistory-handler
Parent: /oracle/odl/quicktrace
Process:        your_server:7009
ServerName:     your_server
bufferElapsed.value:    134719    seconds
bufferRecordCount.value:        1579     records
bufferUsedPercentage.value:     100       percent
oldestTimestamp.value:  2012-06-26T18:38:48.261-0700    time
```

Viewing REST Metrics

Oracle ADF provides REST metrics that are used to analyze the system performance and monitor the status of the ADF application. Use DMS Spy Servlet to view the REST metrics.

ADF middle tier business components may be exposed as REST endpoints by building an ADF REST Application, see How to Create ADF REST Resources from View Object. When the service is deployed in a large scale clustered environment, the developers and system administrators need access to metrics that help analyze system performance or monitor system status.

Oracle Dynamic Monitoring Service calls are made from the REST servlet whenever it services a REST request, recording resource usage incurred during processing. For more information about the Oracle Dynamic Monitoring Service (DMS), see Using the Oracle Dynamic Monitoring Service.

The DMS Spy Servlet, which is optionally deployed into the container, allows you to view the state of the DMS metrics. The following is an example of an HTTP REST request from the DMS Spy Servlet: `http://10.242.60.105:8888/RESTMar2016-RESTWebService-context- root/rest/v0/Emp/7654`. For more information on how to send an HTTP REST request in JDeveloper or using the command line tool cURL, see Testing the ADF REST Resources Using Integrated WebLogic Server.

ADF REST web service returns a structured REST response. For more information on HTTP method support for ADF REST, see ADF REST Framework Reference.

At runtime, metrics are stored in memory and recorded to an ADF-REST deployment dashboard. For more information on how to get the ADF BC resource describe from which the specific HTTP REST request is made, see Retrieving the ADF REST Resource Describe.

Example 5-1   Sample Structured REST Response

```
{
"Empno" : 7654,
"Ename" : "MARTIN",
"Job" : "SALESMAN",
"Mgr" : 7098,
"Hiredate" : "1981-09-28T00:00:00-07:00",
"Sal" : 1250,
"Com" : 1400,
"Deptno" : 30,
"links" : [ {
"rel" : "self",
"href" : "http://10.242.60.105:8888/RESTMar2016-RESTWebService-
context- root/rest/v0/Emp/7654",
"name" : "Emp",
"kind" : "item"
}, {
"rel" : "canonical",
"href" : "http://10.242.60.105:8888/RESTMar2016-RESTWebService-
context- root/rest/v0/Emp/7654",
"name" : "Emp",
"kind" : "item"
} ]
}
```

Types of REST Metrics

The REST web service has several types of metrics. The metrics help you to track performance information that developers and system administrators use to help analyze system performance or monitor system status.

Different HTTP REST requests result in different metrics displayed in ADF-REST dashboard.

Table 5-2 describes metrics that are associated with the ADF-REST application.

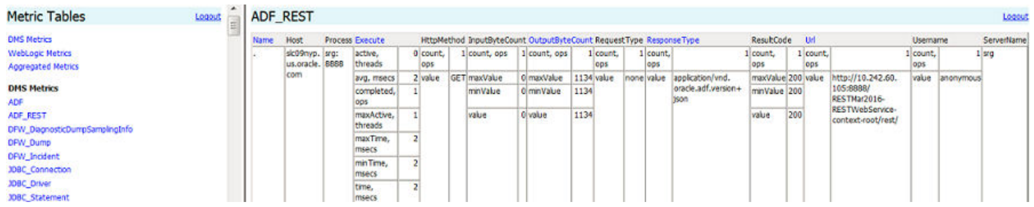
Table 5-2   DMS ADF-REST Metrics

Metric	Description
Name	Specifies the REST resource name.
Host	Specifies the host name servicing the request.
Process	Specifies the node name and port.
Execute	Specifies execution details.
HTTP Method	Specifies such HTTP methods as get, put, patch, etc.

Table 5-2 (Cont.) DMS ADF-REST Metrics

Metric	Description
InputByteCount	Specifies inbound payload size.
OutputByteCount	Specifies output payload size.
Request Type	Specifies HTTP request type.
Response Type	Specifies HTTP response type.
resultCode	Specifies HTTP result code.
Url	Specifies the URL of the REST request.
Username	Specifies the name of a user.

Figure 5-1 ADF_REST View of the DMS Spy Servlet



Metric Tables Logout

DMS Metrics
WebLogic Metrics
Aggregated Metrics

DMS Metrics
ADF
ADF_REST
DFW_DiagnosticDumpSamplingInfo
DFW_Dump
DFW_Incident
JDBC_Connection
JDBC_Driver
JDBC_Statement

ADF_REST Logout

Name	Host	Process	Execute		HttpMethod		InputByteCount		OutputByteCount		RequestType		ResponseType		ResultCode		Url		Username		ServerName
.	slc09nyp.us.oracle.com	srg:8888	active, threads	0	count, ops	1	count, ops	1	count, ops	1	count, ops	1	count, ops	1	count, ops	1	count, ops	1	count, ops	1	srg
			avg, msecs	2	value	GET	maxValue	0	maxValue	1134	value	none	value	application/vnd.oracle.adf.version+json	maxValue	200	value	http://10.242.60.105:8888/RESTMar2016-RESTWebService-context-root/rest/	value	anonymous	
			completed, ops	1			minValue	0	minValue	1134					minValue	200					
			maxActive, threads	1			value	0	value	1134					value	200					
			maxTime, msecs	2																	
			minTime, msecs	2																	
			time, msecs	2																	

Getting REST Metrics

ADF-REST metrics are shown in a table.

To view the contents of the ADF-REST metric table in the DMS Spy servlet, click the table name in the left column.

For more information on different types of REST metrics and their description see Types of REST Metrics.

ADF Logs

Oracle Fusion Middleware provides log files that record events such as error messages, warning messages, server information, and HTTP requests. Use the Fusion Middleware Control or the WLST commands to view, download, and configure settings for these log files.

Most Fusion Middleware log files are in the Oracle Diagnostic Logging (ODL) format. You can use Fusion Middleware Control or WLST commands to search, view, and download log files. You can also use these tools to perform various tasks related to logging, such as configuring settings, changing file locations, setting log levels, setting file format, and configuring tracing.

For more information about Fusion Middleware log files, see Managing Log Files and Diagnostic Data of the *Administering Oracle Fusion Middleware*.

JDeveloper also provides an ADF Logger and a log analyzer that you can use for monitoring and diagnostic purposes. JDeveloper generates the log file in the Oracle Diagnostic Logging (ODL) format that is used throughout Fusion Middleware. In JDeveloper, you use the Oracle Diagnostic Logging Configuration editor to set the log levels and other configuration parameters. Configuration information is stored in the `logging.xml` file.

After the log files have been created, you can use the Oracle Diagnostic Log Analyzer within JDeveloper to view the log entries. You can search and filter the log entries based on log levels, time frames, message content, and other parameters.

For more information about the ADF Loggers, see Using the ADF Logger in *Developing Fusion Web Applications with Oracle Application Development Framework*.

Diagnosing Problems using the Diagnostic Framework

Oracle Fusion Middleware provides a Diagnostic Framework to collect and manage information about a problem so that you can resolve it or send it to Oracle Support for resolution.

This Diagnostic Framework can help you to detect, diagnose, and resolve problems with your application. When a critical error occurs, the Diagnostic Framework immediately captures diagnostic data and associates the data and error with an incident number. Using this number, you can retrieve the data for analysis from the Automatic Diagnostic Repository (ADR).

Oracle ADF provides an ADFConfig dump which will execute when an INCIDENT_ERROR message is logged. You can also add code to invoke the dump in the application exception handlers. The following example shows a sample code you can add to your exception handler to invoke the ADFConfig dump.

```
IllegalArgumentException e = new IllegalArgumentException("test exception");
LoggerFactory.getFrameworkLogger().log(ODLLevel.INCIDENT_ERROR,
 "Test error message", e);
```

For more information about the Diagnostic Framework, see Diagnosing Problems of the *Administering Oracle Fusion Middleware*.

Finding ADF Runtime Version Information

Oracle ADF installation requires the ADF runtime JARs that must be available as a library in the target application server. Use the ADF runtime library version for diagnosing ADF application errors.

Knowing the ADF runtime JARs version numbers can be important information for debugging and diagnosing application problems. This information may be requested by Oracle Support.

You can access My Oracle Support at `https://support.oracle.com`.

You can find which versions of the ADF Runtime JARs are in effect using several methods:

- Retrieving a CSV file that is generated when the application starts.
- Using shell scripts in offline mode.
- Using WLST commands.
- Using Fusion Middleware Control.
- Retrieving the dump files when an incident occurs.

How to Find Version Information by Starting the Application

When a deployed application is started, the runtime JARs version information is dumped into a CSV file located at:

```
<DomainName>/servers/<ServerName>/logs/<MyApp>-Versions.csv
```

Note the following:

- `oracle.adf.share.diagnostics.versions` logger controls generation of runtime JARs versions at application startup. JARs versions are generated when the logger is set to `level="FINEST"`.
- On new WebLogic domains, `oracle.adf.share.diagnostics.versions` logger will be available in the `logging.xml`.
- On existing domains, `oracle.adf.share.diagnostics.versions` logger will not be present in the `logging.xml`. Use Enterprise Manager Fusion Middleware Control or the WLST command to set the logger level.

How to Find Version Information by Using Shell Scripts in Offline Mode

In offline mode, you can use the `printJarVersions.sh` shell script to print the version information or to redirect the information to a file.

In the `%MW_HOME%/oracle_common/common/bin` directory, run the script:

```
bin$./printJarVersions.sh
```

You can redirect the output to a file:

```
bin$./printJarVersions.sh  > /home/trdsouza/export-JarVersions.csv
```

The version information have a format similar to the following (trimmed):

```
Jar Path,Oracle-Version,Oracle-Label,Oracle-Builder,Oracle-BuildTimestamp,
Specification-Version,Implementation-Version/ade/userYada/oracle/oracle_
common/modules/oracle.adf.model_12.2.1/adfm.jar,12.2.1.0.40.66.68,
JDEVADF_12.2.1.0.0_GENERIC_130608.2330.6668,user=userY
host=abc123456,2013-10-02 04:55:45 -0700,,
```

How to Find Version Information by Using the WLST Scripting Console

In offline mode, you can use the WLST `exportJarVersions` command to find the version information:

```
exportJarVersions('/home/userY/export-JarVersions.csv')
```

In online mode, you can use the WLST `exportApplicationJarVersions` command to find the version information:

```
exportApplicationJarVersions('MyApp','/home/userY/export-EarCrmCommon-Versions.csv')
```

If you want to find a specific JAR version, you can use the WLST `exportApplicationSelectedJarVersions` command:

```
exportApplicationSelectedJarVersions('MyApp','/home/userY/export-EarCrmCommon-
Versions.csv', $ORACLE_HOME$/modules/oracle.adf.model_12.2.1/adfm.jar;$ORACLE_HOME$/
modules/oracle.adf.view_12.2.1/adf-richclient-impl-12.jar')
```

How to Find Version Information Using Fusion Middleware Control

You can use Enterprise Manager Fusion Middleware Control to monitor Oracle products that are built using ADF. You can also use this tool to find the version information of runtime JAR files.

For instructions, see Finding Version Information of ADF Runtime JARs.

How to Find Version Information by Inspecting Diagnostic Dump Files

When an INCIDENT_ERROR occurs when the application is running in a WebLogic Domain, the Oracle Diagnostic Framework generates dump files in the WebLogic Domain directories. The location of these files are in:

```
<DomainName>/servers/<ServerName>/adr/diag/ofm/<DomainName>/<ServerName>/incident/
```

One of the generated dump files contains the application JAR version information. It is called `adf_DiagnosticsJarsVersionDumpN.txt` where *N* is a numerical value.

For example, `<myDomainName>/servers/<myServerName>/adr/diag/ofm/<myDomainName>/<myServerName>/incident/adf_DiagnosticsJarsVersionDump45_i5.txt`.

You can examine this file to view the version information. You can also upload this file to My Oracle Support for further analysis.

You can trigger a diagnostic dump from code by logging an INCIDENT_ERROR message.

Troubleshooting Oracle ADF for High Availability

You can deploy a Fusion Web application on high availability environments. After the deployment, you can monitor and diagnose the application deployment issues and the Oracle ADF replication and failover issues.

Troubleshooting Fusion web applications that are deployed to high availability environments require additional concerns. For information about deployment issues in high availability environments, see Troubleshooting High Availability Issues After Deployment in *Developing Fusion Web Applications with Oracle Application Development Framework*. For information about Oracle ADF replication issues in high availability environments, see Troubleshooting Oracle ADF Replication and Failover Issues in *Developing Fusion Web Applications with Oracle Application Development Framework*.

6

Working with WLST Commands for ADF Applications

This chapter describes the WLST commands you can use to deploy, manage, and configure Oracle ADF applications to Oracle WebLogic Server.
This chapter includes the following sections:

Overview of Custom WSLT Commands for Oracle ADF

WLST provides a set of ADF-specific commands to administer ADF applications. Use the WLST commands to deploy, manage, and configure Oracle ADF applications running on Oracle WebLogic Server.

WLST provides a set of commands for ADF applications that are specific to WebLogic Server and allow to:

- manage URL connections for an ADF application
- manage Web Service connections for an ADF application
- export CSV format of JARs versions
- upgrade ADF Metadata of an application

For a complete list of ADF-specific WLST Commands, see *Application Development Framework (ADF) Custom WLST Commands* of the *WLST Command Reference for Infrastructure Components*.

For information about other WLST commands, such as custom Metadata Services (MDS) commands, see *Metadata Services (MDS) Custom WLST Commands* of the *WLST Command Reference for Infrastructure Components*.

Using ADF-Specific WLST Commands

Custom WLST commands for a given Oracle Fusion Middleware component are available for use only if the component is installed in the ORACLE_HOME directory. Use the different ADF-based URL Connections WLST commands to control the prompt display.

To use the custom WLST commands for Oracle ADF, you must run the WLST script from the Oracle Common home (not from the WebLogic server home).

The script is located at:

```
%ORACLE_HOME%\oracle_common\common\bin\wlst.cmd on Windows
$ORACLE_HOME/oracle_common/common/bin/wlst.sh on Unix
```

To navigate the hierarchy of configuration or runtime beans and control the prompt display using ADF-based URL Connections WLST commands:

- Use the `getADFMArchiveConfig` command to manage the `ADFMArchiveConfig` object.
- Use the `exportJarVersions`, `exportApplicationJarVersions`, and `exportApplicationSelectedJarVersions` commands to export CSV format of JARs versions to a specified location.
- Use the Web service connection-specific commands to create, list, or delete Web service connections for an ADF application.
- Use the `listUpgradeHandlers` command to list all upgrade handlers of an application.
- Use commands related to applications' registered ADF Metadata to upgrade all or selected registered ADF Metadata of an application (or all the applications).

Note:

ADF-specific WLST commands can be used with WLST either online, offline, or both. Offline WLST commands are not supported from Maven.

For detailed description, syntax, and examples of using ADF-specific WLST commands, see the *Application Development Framework (ADF) Custom WLST Commands* chapter of *WLST Command Reference for Infrastructure Components*.

For more information about using custom WLST commands, see *Getting Started Using the Oracle WebLogic Scripting Tool (WLST)* in *Administering Oracle Fusion Middleware*.

Part III
Appendices

This part contains reference information for your ADF application, including configuring GlassFish Servers, ADF runtime libraries, and audit references for Oracle ADF.

Part III contains the following chapters:

A
Configuring GlassFish Server

This appendix describes how to configure GlassFish Server for Oracle ADF Essentials. It describes how to obtain the Oracle ADF Runtime and how to install these files into the GlassFish Server
This appendix contains the following sections:

About Configuring GlassFish

GlassFish Server is an application server that can be configured to run Oracle ADF applications. You can use JDeveloper to deploy ADF applications directly to the GlassFish Server, or indirectly to an archive file as the deployment target, and then install this archive file to the target GlassFish Server.

If you do not have a GlassFish Server installation, you can download GlassFish Server from the GlassFish website. Before you can run ADF applications in a GlassFish Server, you need to configure GlassFish with the Oracle ADF Runtime libraries.

For a list of the supported Oracle ADF features for GlassFish, go to the OTN site at `http://www.oracle.com/technetwork/developer-tools/adf/overview/adfessentials-1719844.html`

For instructions on obtaining and installing GlassFish, see `https://glassfish.java.net/download.html`

For information about developing ADF applications for GlassFish, see the Deploying ADF Applications to GlassFish appendix in the *Developing Fusion Web Applications with Oracle Application Development Framework*.

Obtaining GlassFish Server and Oracle ADF Runtime

For a GlassFish Server to support Oracle ADF features, you must install Oracle ADF Essentials and ADF runtime libraries.

Oracle ADF Essentials supports the Open Source and commercial versions of GlassFish Server. With either version, you will need the Full Platform distribution. The Web Profile distribution is not supported. After you have installed the GlassFish Server, you need to obtain the ADF Essentials `adf-essentials.zip` file from OTN and follow the instructions in this appendix to install the ADF Runtime libraries.

For information on how to start and stop the server and other application server tasks, see GlassFish documentation at the GlassFish website.

You will need to perform the following tasks, as described in the following topics:

- Obtaining GlassFish Server
- Obtaining Oracle ADF Runtime

Obtaining GlassFish Server

You can download the open source version of the GlassFish Server from the GlassFish website:

`https://glassfish.java.net/download.html`

Note:

You must download Java EE 7 Full Platform (not Java EE 7 Web Profile) since it includes enterprise class features like JMS.

Follow the instructions and documentation at the GlassFish site to install and configure a GlassFish Server.

Note:

Before installing ADF Essentials your installation of GlassFish Server at `/glassfish-4.1/glassfish/modules/javax.el.jar` must be patched with `javax.el-3.0.1-b08.jar` available for download at https://java.net/projects/uel/sources/svn/show/tags/javax.el-3.0.1-b08.

Obtaining Oracle ADF Runtime

In order for a GlassFish Server to run Oracle ADF applications, you must install the ADF Runtime library files into the GlassFish installation directory.

You can download the Oracle ADF Essentials `adf-essentials.zip` file from the Oracle Technology Network at `http://www.oracle.com/technetwork/developer-tools/adf/downloads/index.html`.

After you have downloaded the `adf-essentials.zip` file, you can extract the files to a flat-structured temporary directory which you can use to copy the required files into the GlassFish installation directories.

For instance, if you are using `unzip`, you can add the `-j` option to create a flat directory structure that has no hierarchical folders.

```
unzip -j <file> -d <destination>
```

Configuring GlassFish with ADF Runtime Libraries

GlassFish Server can be configured to run Oracle ADF applications. Before you can run ADF applications on a GlassFish Server, you need to configure GlassFish with the Oracle ADF Runtime libraries.

The ADF Runtime libraries consists of the following:

- ADF Share libraries
- ADF Certified Compatible JSF Implementation
- ADF Model libraries
- ADF Controller libraries
- ADF View libraries

The ADF Share libraries and ADF Certified Compatible JSF Implementation must be manually installed into the GlassFish installation. You use your operating system commands or tools to copy the files into GlassFish. For instructions, see Installing ADF Share Libraries Manually.

The ADF Model, ADF Controller, and ADF View libraries are loaded into GlassFish with the deployed application. When you use JDeveloper to develop the application, you will reference the required libraries before you package the application into an EAR file for deployment.

The steps for installing the ADF Runtime libraries are:

1. Configure ADF Share libraries.
   - Copy the ADF Share libraries from `adf-essentials.zip` into the GlassFish installation.
2. Deploy the application as an EAR file to GlassFish Server.

Installing ADF Share Libraries Manually

It may be helpful to have an understanding of the options that are available to you when you are mapping ADF Share for GlassFish. For more information, see Configuring GlassFish with ADF Runtime Libraries.

You will need to complete these tasks:

- Install the GlassFish Server
- Obtain the `adf-essentials.zip` file and unzip it to a temporary directory
- Obtain the ADF Certified Compatible JSF implementation version 2.2.8-11 from Maven Central at https://repo.maven.apache.org/maven2/org/glassfish/javax.faces/2.2.8-11/javax.faces-2.2.8-11.jar

To install the ADF Share Runtime libraries:

1. Copy or move the ADF Share library files from the temporary directory to the `<`*`glassfish`*`>/domains/domain1/lib/applibs` folder:

   The ADF Share files should be in the temporary directory where you had unzip the `adf-essentials.zip` file as described in Obtaining Oracle ADF Runtime This directory should be `<`*`temp`*`>/oracle_common/modules`.

You must copy the following JAR files into the `<glassfish>/domains/domain1/lib/applibs` directory:

```
oracle.adf.share.ca_12.2.1/adf-share-base.jar
oracle.adf.share.ca_12.2.1/adf-share-ca.jar
oracle.adf.share_12.2.1/commons-el.jar
oracle.adf.share_12.2.1/adf-share-support.jar
oracle.adf.share_12.2.1/adfsharembean.jar
oracle.adf.share_12.2.1/jsp-el-api.jar
oracle.adf.share_12.2.1/adflogginghandler.jar
oracle.adf.share_12.2.1/oracle-el.jar
oracle.mds_12.2.1/mdsrt.jar
oracle.bali.share_12.2.1/share.jar
oracle.xmlef_12.2.1/xmlef.jar
oracle.javatools_12.2.1/resourcebundle.jar
oracle.javatools_12.2.1/javamodel-rt.jar
oracle.javatools_12.2.1/javatools-nodeps.jar
oracle.javatools_12.2.1/oicons.jar
oracle.adf.security_12.2.1/adf-share-security.jar
oracle.adf.security_12.2.1/adf-controller-security.jar
oracle.xdk_12.2.1/xmlparserv2_sans_jaxp_services.jar
oracle.xdb_11.1.0.jar
oracle.jrf_12.2.1/jrf-api.jar
oracle.jdbc_11.2/ojdbc6dms.jar
oracle.dms_12.2.1/dms.jar
oracle.odl_12.2.1/ojdl.jar
oracle.odl_12.2.1/ojdl2.jar
oracle.classloader_11.1.1.jar
oracle.logging-utils_11.1.1.jar
oracle.web-common_12.2.1.jar
org.apache.bcel_5.1.jar
oracle.nlsrtl_11.2.0/orai18n-mapping.jar
org.apache.commons.beanutils_1.8.3
org.apache.commons.logging_1.1.1
oracle.jsp_12.2.1/globaltldcache
oracle.mds_12.2.1/mdsrt
com.oracle.webservices.fmw.oc4j-ws-support-impl_12.2.1
oracle.http_client_12.2.1
oracle.web-common_12.2.1
oracle.web-common-schemas_12.2.1
```

2. Verify the libraries.

To install the ADF Certified Compatible JSF Implementation:

1. Download the `javax.faces-2.2.8-11.jar` file from Maven Central.
2. Remove the version number to rename the downloaded file to `javax.faces.jar`.
3. Copy the renamed file to the `glassfish4/glassfish/modules/` directory to overwrite the `javax.faces.jar` file in the target directory.

   The ADF Certified Compatible JSF Implementation is backward compatible with the one which is already present in GlassFish 4.1.

Additional Configuration Tasks

GlassFish Server can be configured to create a datasource and JVM cache after the installation of ADF runtime libraries. Use the GlassFish Server Administration Console to perform the additional configuration tasks.

After you have installed the ADF Runtime into the GlassFish Server, you need to perform additional configuration tasks on GlassFish.

The configuration tasks are:

- Using the GlassFish Administration Console
- Creating a Datasource for GlassFish
- Configuring the JVM Cache

Using the GlassFish Administration Console

You can use the GlassFish Administration Console to configure the GlassFish Server including managing applications, JDBC pools, and other resources. If you are using GlassFish 3.1.2, the secure console is disabled by default. You would need to enable secure console to access the admin page remotely or only access the page from the same machine where the GlassFish server is running.

The GlassFish Administration Console is at:

```
http://<machine_name>:4848/
```

Creating a Datasource for GlassFish

It may be helpful to have an understanding of the options that are available to you when you are creating a datasource for GlassFish. For more information, see Additional Configuration Tasks.

You can use the GlassFish console or use asadmin commands to create the datasource. For using the GlassFish console to create a datasource, see GlassFish documentation.

To create a datasource for GlassFish using asadmin commands:

1. Open a command line window.
2. Invoke the `asadmin` command to create a datasource for GlassFish.

   For instance, the following command creates a datasource for an application:

   ```
   asadmin> create-jdbc-connection-pool --datasourceclassname
     oracle.jdbc.pool.SampleDataSource
     --restype javax.sql.XADataSource
     --property user=xyz
     :password=xyz:url=jdbc:oracle:thin:@machine.com:1521:machine
     --target=server SampleDSPool

   asadmin> ping-connection-pool sampleDSPool

   asadmin> create-jdbc-resource --connectionpoolid SampleDSPool jdbc/OracleDS
   ```

Configuring the JVM Cache

You need to configure the JVM settings to `simple` and increase the memory size from 192 Mb to 512 Mb.

It may be helpful to have an understanding of the options that are available to you when you are configuring JVM for GlassFish. For more information, see Additional Configuration Tasks.

To configure JVM Cache for MDS:

1. Start the GlassFish Administration Console.
2. Choose **Configurations > server-config > JVM Settings**.
3. Select **JVM Options** and specify `-Doracle.mds.cache=simple` and `XX:MaxPermSize=512m`.
4. Click **Save**.
5. Or, open the `<glassfish>/domains/domain1/config/domain.xml` file and edit the following entries:

   ```
   <jvm-options>-XX:MaxPermSize=512m</jvm-options>
   <jvm-options>-Doracle.mds.cache=simple</jvm-options>
   ```

Deploying an ADF Application to GlassFish

You can deploy an ADF application to a GlassFish Server after you have referenced the libraries in the application.

For more information, see the Deploying ADF Applications to GlassFish appendix in the *Developing Fusion Web Applications with Oracle Application Development Framework*.

B

ADF Runtime Libraries

This appendix describes the contents of ADF runtime libraries (`adf.oracle.domain.webapp.war`, `adf.oracle.domain.ear`, `adf.desktopintegration.war`, `adf.desktopintegration.model.ear`, and system classpath) deployed into Oracle WebLogic Server to support ADF applications.
The following ADF runtime libraries are described:

- Using JDeveloper to Find the ADF Runtime Library
- adf.oracle.domain.webapp.war Library
- adf.oracle.domain.ear Library
- adf.desktopintegration.war Library
- adf.desktopintegration.model.ear Library
- System Classpath

Using JDeveloper to Find the ADF Runtime Library

You can develop and deploy an Oracle ADF application on the target application server. Use JDeveloper to find the ADF runtime library version deployed into Oracle WebLogic Server.

You can use JDeveloper to find a JAR’s corresponding ADF runtime library.

To find the JDeveloper library for a JAR:

1. In JDeveloper, select **Tools > Manage Libraries**.
2. In the Manage Libraries dialog Libraries tab, click the **Search** icon and select **Jar name** from the dropdown list.
3. In the search field, enter the name of the JAR and click the search icon.

adf.oracle.domain.webapp.war Library

The contents of `adf.oracle.domain.webapp.war` ADF runtime library can be deployed into Oracle WebLogic Server to support ADF applications. Different JAR files and their corresponding ADF runtime library are loaded into `adf.oracle.domain.webapp.war` file.

Table B-1 lists the JAR files that are packaged into the `adf.oracle.domain.webapp.war` file and their corresponding ADF runtime library.

Table B-1 adf.oracle.domain.webapp.war Library

JAR	ADF Library
`oracle.facesconfigdt_12.2.1/taglib.jar`	ADF Faces Change Manager Runtime 11

Table B-1   (Cont.) adf.oracle.domain.webapp.war Library

JAR	ADF Library
`oracle.facesconfigdt_12.2.1/facesconfigmodel.jar`	ADF Faces Change Manager Runtime 11
`org.apache.http.components.httpmime-4.1.2.jar`	NA
`oracle.adf.view_12.2.1/batik-anim.jar`	ADF DVT Faces Runtime
`oracle.adf.view_12.2.1/dvt-trinidad.jar`	ADF DVT Core Runtime ADF DVT Faces Runtime
`oracle.adf.view_12.2.1/batik-transcoder.jar`	ADF DVT Faces Runtime
`oracle.adf.view_12.2.1/adf-faces-changemanager-rt.jar`	ADF Faces Change Manager Runtime 11
`oracle.adf.view_12.2.1/dvt-databinding-dt-core.jar`	ADF Designtime API
`oracle.adf.view_12.2.1/dvt-jclient-adf.jar`	Oracle BI Graph
`oracle.adf.view_12.2.1/adf-faces-templating-dt-core.jar`	ADF Designtime API
`oracle.adf.view_12.2.1/adf-richclient-api-11.jar`	Trinidad Databinding Runtime ADF Faces Runtime 11
`oracle.adf.view_12.2.1/dvt-shared-js.jar`	ADF DVT Faces Runtime
`oracle.adf.view_12.2.1/adf-faces-skin-dt-core.jar`	NA
`oracle.adf.view_12.2.1/inspect4.jar`	Oracle JEWT
`oracle.adf.view_12.2.1/batik-swing.jar`	ADF DVT Faces Runtime
`oracle.adf.view_12.2.1/batik-codec.jar`	ADF DVT Faces Runtime
`oracle.adf.view_12.2.1/adf-faces-databinding-rt.jar`	Trinidad Databinding Runtime ADF Faces Databinding Runtime
`oracle.adf.view_12.2.1/adf-dt-at-rt.jar`	ADF Model Runtime ADF Designtime API
`oracle.adf.view_12.2.1/batik-script.jar`	ADF DVT Faces Runtime
`oracle.adf.view_12.2.1/jewt4.jar`	BC4J Tester Oracle Help for Java Oracle JEWT
`oracle.adf.view_12.2.1/adf-view-databinding-dt-core.jar`	ADF Designtime API
`oracle.adf.view_12.2.1/adf-richclient-impl-11.jar`	ADF Faces Runtime 11
`oracle.adf.view_12.2.1/dvt-faces.jar`	ADF DVT Faces Runtime

Table B-1 (Cont.) adf.oracle.domain.webapp.war Library

JAR	ADF Library
`oracle.adf.view_12.2.1/dvt-utils.jar`	Oracle BI Graph ADF DVT Core Runtime ADF DVT Faces Runtime
`oracle.adf.view_12.2.1/trinidad-impl.jar`	ADF Faces Runtime 11 Trinidad Runtime 11
`oracle.adf.view_12.2.1/dvt-databindings.jar`	ADF DVT Faces Databinding Runtime
`oracle.adf.view_12.2.1/trinidad-api.jar`	ADF Faces Runtime 11 Trinidad Runtime 11
`oracle.adf.view_12.2.1/batik-xml.jar`	ADF DVT Faces Runtime
`oracle.adf.view_12.2.1/batik-css.jar`	ADF DVT Faces Runtime
`oracle.adf.view_12.2.1/batik-dom.jar`	ADF DVT Faces Runtime
`oracle.adf.view_12.2.1/dvt-as.jar`	NA
`oracle.adf.view_12.2.1/batik-parser.jar`	ADF DVT Faces Runtime
`oracle.adf.view_12.2.1/batik-util.jar`	ADF DVT Faces Runtime
`oracle.adf.view_12.2.1/batik-bridge.jar`	ADF DVT Faces Runtime
`oracle.adf.view_12.2.1/batik-ext.jar`	ADF DVT Faces Runtime
`oracle.adf.view_12.2.1/adf-share-web.jar`	NA
`oracle.adf.view_12.2.1/dvt-basemaps.jar`	ADF DVT Faces Runtime
`oracle.adf.view_12.2.1/adf-faces-templating-dtrt.jar`	ADF Designtime API
`oracle.adf.view_12.2.1/batik-gui-util.jar`	ADF DVT Faces Runtime
`oracle.adf.view_12.2.1/adf-dt-at-rt-wizards.jar`	ADF Runtime Wizards
`oracle.adf.view_12.2.1/xml-apis-ext.jar`	ADF DVT Faces Runtime
`oracle.adf.view_12.2.1/adf.constants.jar`	NA
`oracle.adf.view_12.2.1/batik-svggen.jar`	ADF DVT Faces Runtime
`oracle.adf.view_12.2.1/batik-svg-dom.jar`	ADF DVT Faces Runtime
`oracle.adf.view_12.2.1/adf-dynamic-faces.jar`	ADF Faces Dynamic Components
`oracle.adf.view_12.2.1/batik-extension.jar`	ADF DVT Faces Runtime
`oracle.adf.view_12.2.1/batik-awt-util.jar`	ADF DVT Faces Runtime
`oracle.adf.view_12.2.1/dvt-facesbindings.jar`	ADF DVT Faces Databinding Runtime
`oracle.adf.view_12.2.1/dvt-jclient.jar`	Oracle BI Graph ADF DVT Core Runtime ADF DVT Faces Runtime

Table B-1   (Cont.) adf.oracle.domain.webapp.war Library

JAR	ADF Library
`oracle.adf.view_12.2.1/bundleresolver.jar`	Resource Bundle Variable Resolver
`oracle.adf.view_12.2.1/adf-view-unified-databinding-dt-core.jar`	ADF Designtime API
`oracle.adf.view_12.2.1/adf-richclient-jmx.jar`	ADF Faces JMX Runtime 11
`oracle.adf.view_12.2.1/adf-faces-databinding-dt-core.jar`	ADF Designtime API
`oracle.adf.view_12.2.1/adf-richclient-automation-11.jar`	Oracle Extended Selenium
`oracle.adf.view_12.2.1/batik-gvt.jar`	ADF DVT Faces Runtime
`oracle.adf.view_12.2.1/adf-customizationset-ui.jar`	NA
`oracle.adf.view_12.2.1/adf-customizationset-mig.jar`	NA
`oracle.xdk_12.2.1/xml.jar`	MDS Runtime Dependencies Oracle XML Parser v2 XSQL Runtime
`oracle.adf.controller_12.2.1/adf-controller.jar`	ADF Controller Runtime
`oracle.adf.controller_12.2.1/adf-controller-rt-common.jar`	ADF Controller Runtime
`oracle.adf.controller_12.2.1/adf-controller-api.jar`	ADF Controller Runtime
`oracle.adf.share.ca_12.2.1/adf-share-ca.jar`	MDS Runtime Dependencies ADF Model Generic Runtime BC4J Runtime BC4J Security ADF Common Runtime
`velocity-dep-1.4.jar`	ADF Designtime API
`org.apache.http.components.httpclient-4.1.2.jar`	NA
`oracle.adf.pageflow_12.2.1/adf-pageflow-impl.jar`	ADF Page Flow Runtime
`oracle.adf.pageflow_12.2.1/adf-pageflow-rc.jar`	ADF Page Flow Runtime
`oracle.adf.pageflow_12.2.1/adf-pageflow-fwk.jar`	ADF Page Flow Runtime
`oracle.adf.pageflow_12.2.1/adf-pageflow-dtrt.jar`	ADF Page Flow Runtime ADF Designtime API

Table B-1 (Cont.) adf.oracle.domain.webapp.war Library

JAR	ADF Library
`org.apache.http.components.httpclient-cache-4.1.2.jar`	NA

adf.oracle.domain.ear Library

The contents of `adf.oracle.domain.ear` ADF runtime library can be deployed into Oracle WebLogic Server to support ADF applications. Different JAR files and their corresponding ADF runtime library are loaded into `adf.oracle.domain.ear` file.

Table B-2 lists the JAR files that are packaged into the `adf.oracle.domain.ear` file and their corresponding ADF runtime library.

Table B-2 adf.oracle.domain.ear Library

JAR	ADF Library
`oracle.adf.model_12.2.1/ordim.jar`	Oracle Intermedia
`oracle.adf.model_12.2.1/adflibfilter.jar`	ADF Common Web Runtime
`oracle.adf.model_12.2.1/adf-faces-registration.ja`	NA
`oracle.adf.model_12.2.1/bc4jhtml.jar`	BC4J Struts Runtime
`oracle.adf.model_12.2.1/adfbcsvc-share.jar`	BC4J Service Runtime BC4J Service Client BC4J System Catalog
`oracle.adf.model_12.2.1/adfmportlet.jar`	NA
`oracle.adf.model_12.2.1/jmxdc.jar`	JMX Data Control
`oracle.adf.model_12.2.1/jr_dav.jar`	Resource Catalog Service
`oracle.adf.model_12.2.1/bc4j-mbeans.jar`	BC4J Runtime
`10 oracle.adf.model_12.2.1/bc4jimdomains.jar`	Oracle Intermedia
`oracle.adf.model_12.2.1/adfm.jar`	BC4J EJB Client ADF Model Runtime BC4J Oracle Domains ADF Model Generic Runtime BC4J Runtime SR-227 API BC4J EJB Runtime BC4J Client BC4J IAS Client
`oracle.adf.model_12.2.1/rcs-adflib-rt.jar`	NA
`oracle.adf.model_12.2.1/adfbcsvc.jar`	BC4J Service Runtime
`oracle.adf.model_12.2.1/ordhttp.jar`	Oracle Intermedia

Table B-2 (Cont.) adf.oracle.domain.ear Library

JAR	ADF Library
`15 oracle.adf.model_12.2.1/jdev-cm.jar`	BC4J EJB Client ADF Model Runtime BC4J Tester BC4J Runtime BC4J Client BC4J IAS Client Connection Manager BC4J Recorder
`oracle.adf.model_12.2.1/dvt-databindings-mds.jar`	ADF DVT Faces Databinding MDS Runtime
`oracle.adf.model_12.2.1/adflibrary.jar`	ADF Model Runtime ADFm Designtime API
`oracle.adf.model_12.2.1/adftransactionsdt.jar`	ADF Model Runtime ADFm Designtime API ADF Designtime API;
`oracle.adf.model_12.2.1/regexp.jar`	BC4J Tester BC4J Recordere
`20 oracle.adf.model_12.2.1/adf-sec-idm-dc.jar`	User and Role Data Control
`oracle.adf.model_12.2.1/adfmweb.jar`	ADF Web Runtime
`oracle.adf.model_12.2.1/mds-dc.jar`	NA
`oracle.adf.model_12.2.1/adfbcsvc-client.jar`	BC4J Service Client
`oracle.adf.model_12.2.1/adf-controller-schema.jar`	ADF Controller Schema ADF Controller Schema
`25 oracle.adf.model_12.2.1/adfdt_common.jar`	ADF Model Runtime ADFm Designtime API
`oracle.adf.model_12.2.1/db-ca.jar`	BC4J EJB Client ADF Model Runtime BC4J Tester BC4J Runtime BC4J Client BC4J IAS Client BC4J Recorder DB Runtime (db-tests)
`oracle.adf.model_12.2.1/rcsrt.jar`	Resource Catalog Service
`oracle.adf.model_12.2.1/adfm-sqldc.jar`	ADF SQL Data Control Runtimel
`oracle.adf.model_12.2.1/adfm-debugger.jar`	BC4J Tester
`30 oracle.adf.model_12.2.1/bc4jsyscat.jar`	BC4J System Catalog
`oracle.adf.model_12.2.1/adfbcsvc-registration.jar`	Kava SDO

Table B-2 (Cont.) adf.oracle.domain.ear Library

JAR	ADF Library
`oracle.adf.model_12.2.1/adftags.jar`	Oracle ADF DataTag
`oracle.adf.model_12.2.1/datatags.jar`	NA
`oracle.adf.businesseditor_12.2.1/adf-businesseditor-model.jar`	ADFm Business Editor Runtime
`35 oracle.adf.businesseditor_12.2.1/adf-businesseditor-objects.jar`	ADFm Business Editor Runtime
`oracle.adf.businesseditor_12.2.1/adf-businesseditor-settings.jar`	ADFm Business Editor Settings ADFm Business Editor Runtime
`groovy-all-2.0.5.jar`	ADF Model Runtime;ADF Model Generic Runtime;BC4J Runtime
`oracle.xdk_12.2.1/oraclexsql.jar`	NA
`oracle.xdk_12.2.1/xsqlserializers.jar`	XSQL Runtime

adf.desktopintegration.war Library

The contents of `adf.desktopintegration.war` ADF runtime library can be deployed into Oracle WebLogic Server to support ADF applications. Different JAR files and their corresponding ADF runtime library are loaded into `adf.desktopintegration.war` file.

Table B-3 lists the JAR files that are packaged into the `adf.desktopintegration.war` file and their corresponding ADF runtime library.

Table B-3 adf.desktopintegration.war Library

JAR	ADF Library
`oracle.adf.desktopintegration_12.2.1/adf-desktop-integration.jar`	ADF Desktop Integration Runtime
`oracle.adf.desktopintegration_12.2.1/adf-desktop-integration-admin-tool.jar`	NA

adf.desktopintegration.model.ear Library

The contents of `adf.desktopintegration.model.ear` ADF runtime library can be deployed into Oracle WebLogic Server to support ADF applications. Different JAR files and their corresponding ADF runtime library are loaded into `adf.desktopintegration.model.ear` file.

Table B-4 lists the JAR files that are packaged into the `adf.desktopintegration.model.ear` file and their corresponding ADF runtime library.

Table B-4 adf.desktopintegration.model.ear Library

JAR	ADF Library
`oracle.adf.desktopintegration.model_12.2.1/adf-desktop-integration-model-api.jar`	ADF Desktop Integration Model API

System Classpath

The contents of `system classpath` ADF runtime library can be deployed into Oracle WebLogic Server to support ADF applications. Different JAR files and their corresponding ADF runtime library are loaded into `system classpath`.

Table B-5 lists the JAR files that are loaded into the system classpath and their corresponding ADF runtime library.

Table B-5 System Classpath

JAR	ADF Library
`oracle.javatools_12.2.1/javatools-jndi-local.jar`	NA
`oracle.javatools_12.2.1/javamodel-rt.jar`	JAX-RPC Client
`oracle.javatools_12.2.1/javatools-nodeps.jar`	ADF Common Web Runtime MDS Runtime Dependencies ADFm Desgntime API JAX-RPC Client DB Runtime (db-tests)
`oracle.javatools_12.2.1/oicons.jar`	ADFm Designtime API
`oracle.javatools_12.2.1/resourcebundle.jar`	ADF Desktop Integration Runtime Resource Bundle Support BC4J Runtime
`oracle.adf.security_12.2.1/adf-share-security.jar`	ADF Model Runtime BC4J Security ADF Common Runtime
`oracle.adf.security_12.2.1/adf-controller-security.jar`	ADF Model Runtime ADF Common Runtime
`oracle.adf.share.ca_12.2.1/adf-share-ca.jar`	MDS Runtime Dependencies ADF Model Generic Runtime BC4J Runtime BC4J Security ADF Common Runtime

Table B-5 (Cont.) System Classpath

JAR	ADF Library
`oracle.adf.share.ca_12.2.1/adf-share-base.jar`	ADF Common Web Runtime MDS Runtime Dependencies ADF Model Generic Runtime BC4J Runtime BC4J Security ADF Common Runtime
`features/adf.security_12.2.1.jar`	NA
`features/adf.share_12.2.1.jar`	NA
`features/adf.share.ca_12.2.1.jar`	NA
`oracle.mds_12.2.1/oramds.jar`	MDS Runtime Dependencies
`oracle.mds_12.2.1/mdsrt.jar`	MDS Runtime
`oracle.mds_12.2.1/mdslcm-client.jar`	NA
`oracle.mds_12.2.1/mdslcm.jar`	NA
`oracle.xmlef_12.2.1/xmlef.jar`	MDS Runtime Dependencies ADF Faces Change Manager Runtime 11 ADF Model Generic Runtime
`oracle.adf.share_12.2.1/jsp-el-api.jar`	ADF Model Runtime MDS Runtime Dependencies ADF Model Generic Runtime BC4J Runtime
`oracle.adf.share_12.2.1/adf-share-mbeans-wlst.jar`	NA
`oracle.adf.share_12.2.1/adflogginghandler.jar`	MDS Runtime Dependencies BC4J Tester ADF Model Generic Runtime BC4J Runtime ADF Common Runtime
`oracle.adf.share_12.2.1/adf-share-wls.jar`	NA
`oracle.adf.share_12.2.1/oracle-el.jar`	ADF Model Runtime MDS Runtime Dependencies ADF Model Generic Runtime BC4J Runtime

Table B-5 (Cont.) System Classpath

JAR	ADF Library
`oracle.adf.share_12.2.1/adf-share-support.jar`	MDS Runtime Dependencies ADF Model Generic Runtime BC4J Runtime BC4J Security ADF Common Runtime
`oracle.adf.share_12.2.1/adfsharembean.jar`	BC4J Runtime ADF Common Runtime
`oracle.adf.share_12.2.1/commons-el.jar`	ADF Model Runtime MDS Runtime Dependencies ADF Model Generic Runtime BC4J Runtime
`oracle.adf.share_12.2.1/adfscripting.jar`	NA
`oracle.ons_12.2.1/ons.jar`	NA
`oracle.bali.share_12.2.1/share.jar`	MDS Runtime Dependencies BC4J Tester ADF Model Generic Runtime Oracle Help for Java Oracle JEWT

C

Audit Reference for Oracle Application Development Framework

This appendix provides reference information for auditing in Oracle Application Development Framework.
This appendix contains these sections:

About Custom and Standard Audit Reports

The Common Audit Framework in Oracle Fusion Middleware provides a set of standard reports based on your audit records. It also enables you to modify the standard reports and create your own custom audit reports.

This appendix provides details about events that can be audited in Oracle Application Development Framework. Use this information to understand the structure of each event record to develop custom reports.

The following documents provide more information to help you write custom reports:

- The Audit Schema in the *Securing Applications with Oracle Platform Security Services*

The following documents provide additional information about how to configure auditing and view standard reports:

- Configuring and Managing Auditing in the *Securing Applications with Oracle Platform Security Services*

Attributes of ADF Audit Events

You can audit ADF events using Oracle ADF. These events use different attributes to perform various operational tasks.

ADF audit events use the attributes specified in Table C-1:

Table C-1   Attributes for ADF Audit Events

Attribute Name	Description
`mds-mo-name`	Metadata object name.
`mds-mo-type`	Metadata object type.
`mds-cust-layer-name`	Customization layer name.
`mds-cust-layer-value`	Customization layer value.
`mds-sandbox-name`	Sandbox name.
`custom-operation`	The operation that triggered this event.
`custom-attribute1-name`	The name of custom attribute 1.
`custom-attribute1-value`	The value of custom attribute 1.
`custom-attribute1-old-value`	The old value of custom attribute 1.
`custom-attribute2-name`	The name of custom attribute 2.
`custom-attribute2-value`	The value of custom attribute 2.
`custom-attribute2-old-value`	The old value of custom attribute 2.
`custom-attribute3-name`	The name of custom attribute 3.
`custom-attribute3-value`	The value of custom attribute 3.
`custom-attribute3-old-value`	The old value of custom attribute 3.
`custom-attribute4-name`	The name of custom attribute 4.
`custom-attribute4-value`	The value of custom attribute 4.
`custom-attribute4-old-value`	The old value of custom attribute 4.
`custom-attribute5-name`	The name of custom attribute 5.
`custom-attribute5-value`	The value of custom attribute 5.
`custom-attribute5-old-value`	The old value of custom attribute 5.
`custom-sub-object-name`	User-recognizable name for the sub-object (XML element/attribute) that changed.
`custom-sub-object-type`	User-friendly type for the sub-object (XML element/attribute) that changed.
`mds-repository-name`	MDS repository name.
`mds-partition-name`	MDS repository partition name.
`custom-object-name`	User-friendly object name.

Table C-1 (Cont.) Attributes for ADF Audit Events

Attribute Name	Description
`custom-object-type`	User-recognizable object type.

Audit Events in Oracle ADF View

You can audit events that occur in Oracle ADF View using Oracle ADF. These events are used to perform various operational tasks.

Table C-2 lists the audit events for ADF View:

Table C-2 Oracle ADF View Audit Events

Event	Description
`CreatePage`	Create a new page.
`CreateRegion`	Add a region to a page.
`CreateDataboundComponent`	Add a databound component to a page.

Audit Events in Oracle ADF Business Components

You can audit events that occur in Oracle ADF Business Components using Oracle ADF. These events are used to perform various operational tasks.

Table C-3 lists the audit events for ADF Business Components:

Table C-3 Oracle ADF Business Components Audit Events

Event Type	Event	Description
View Object	`AddViewLinkDef`	Add a view link definition.
View Object	`RemoveViewLinkDef`	Remove a view link definition.
View Object	`AddAttributeDef`	Add an attribute.
Entity Object	`AddAttributeDef`	Add an attribute definition.
Entity Object	`AddTrigger`	Add a trigger.
Entity Object	`AddValidator`	Add a validator.
Entity Object	`RemoveTrigger`	Remove a trigger.
Entity Object	`RemoveValidator`	Remove a validator.
Application Module	`CreateViewObject`	Create a view object.
Application Module	`RemoveViewObject`	Remove a view object.

Audit Events in Oracle ADF Model

You can audit events that occur in Oracle ADF Model using Oracle ADF. These events are used to perform various operational tasks.

Table C-4 lists the audit events for ADF Model:

Table C-4 Oracle ADF Model Audit Events

Event Type	Event	Description
Page Definition	`AddControBinding`	Add a control binding.
Page Definition	`AddIterator`	Add an iterator binding.
Page Definition	`AddVariable`	Add a variable binding.
Page Definition	`AddExecutable`	Add an executable binding.
Page Definition	`AddParameter`	Add a parameter binding.
Page Definition	`AddNestedContainer`	Add a nested container binding.
Page Definition	`RemoveExecutable`	Remove an executable binding.
Page Definition	`RemoveIterator`	Remove an iterator binding.
Page Definition	`RemoveParameter`	Remove a parameter binding.
Application	`AddDataControlReference`	Add data control reference.
Application	`AddPageDefinitionUsage`	Add a page definition usage.
Application	`AddPageMapEntry`	Add a page map entry.
Application	`RemoveDataControl`	Remove data control usage.
Application	`RemovePageDefinitionUsage`	Remove page definition usage.
Application	`RemovePageMapEntry`	Remove a page map entry.
Configuration	`AddDataControlDefinition`	Add a data control definition.
Configuration	`RemoveDataControlDefinition`	Remove a data control definition.

Audit Events in Oracle ADF Controller

You can audit events that occur in Oracle ADF Controller using Oracle ADF. These events are used to perform various operational tasks.

Table C-5 lists the audit events for ADF Controller:

Table C-5 Oracle ADF Controller Audit Events

Event	Description
`setPageName`	Set a page name.
`addControlFlowCase`	Add a control flow case.
`addActivity`	Insert a new activity into a flow.

Table C-5 (Cont.) Oracle ADF Controller Audit Events

Event	Description
`getMutableTaskFlowDefinition`	Edit a task flow definition.
`addEntry`	Add a resource entry.
`updateEntry`	Update a resource entry.
`deleteEntry`	Delete a resource entry.

Audit Events in Jedi

You can audit events that occur in ADF Jedi using Oracle ADF. These events are usually used for various operational tasks such as editing an attribute, deleting an operation, etc.

Table C-6 lists the audit events for Jedi:

Table C-6 Jedi Audit Events

Event	Description
`AddAttribute`	Add an attribute.
`AddOperation`	Add an operation.
`AddValidationRule`	Add a validation rule.
`DeleteOperation`	Delete an operation.
`EditAttribute`	Edit an attribute.
`RemoveAttribute`	Remove an attribute.
`NewCustomObject`	Add new custom object.
`ChangedProperty`	Change a property.
`ChangedLabel`	Display label change.
`ChangedRequired`	Required change.
`ChangedUpdateable`	Updateable change.