Oracle® Fusion Middleware

Introduction to Oracle WebLogic Server 10.3.6

11*g* Release 1 (10.3.6)

E13752-07

July 2015

This document provides an overview of Oracle WebLogic Server 10.3.6 features and describes how you can use them to create enterprise-ready solutions.

ORACLE®

Oracle Fusion Middleware Introduction to Oracle WebLogic Server 10.3.6, 11*g* Release 1 (10.3.6)

E13752-07

Contents

Preface

This preface describes the document accessibility features and conventions used in this guide—*Introduction to Oracle WebLogic Server 10.3.6*.

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.

Conventions

The following text conventions are used in this document:

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

1

Introduction and Roadmap

This chapter describes the contents and audience for this guide—*Introduction to Oracle WebLogic Server 10.3.6*.

This chapter includes the following sections:

- Section 1.1, "Document Scope and Audience"
- Section 1.2, "Guide to This Document"
- Section 1.3, "New and Changed Features in this Release"

1.1 Document Scope and Audience

This document is a resource for system administrators and operators responsible for monitoring and managing a WebLogic Server installation. It is relevant to all phases of a software project, from development through test and production phases.

1.2 Guide to This Document

This document is organized as follows:

- This chapter, Chapter 1, "Introduction and Roadmap," describes the scope and organization of this guide.
- Chapter 2, "Introduction to Oracle WebLogic Server," provides an overview of Oracle WebLogic Server features and describes how you can use them to create enterprise-ready solutions.

1.3 New and Changed Features in this Release

For a comprehensive listing of the new WebLogic Server features introduced in this release, see *What's New in Oracle WebLogic Server*.

2

Introduction to Oracle WebLogic Server

This chapter provides an overview of Oracle WebLogic Server 10.3.6 features and describes how you can use them to create enterprise-ready solutions, as well as an overview of system administration and the Administration Console.

This chapter includes the following sections:

- Introduction to Oracle WebLogic Server
- Overview of WebLogic Server System Administration
- Overview of the Administration Console

2.1 Introduction to Oracle WebLogic Server

The following sections provide a brief overview of Oracle WebLogic Server features and describe how you can use them to create enterprise-ready solutions:

- Section 2.1.1, "The WebLogic Server Solution"
- Section 2.1.2, "Programming Models"
- Section 2.1.3, "High Availability"
- Section 2.1.4, "System Administration"
- Section 2.1.5, "Diagnostic Framework"
- Section 2.1.6, "Security"
- Section 2.1.7, "Oracle JRockit JVM"
- Section 2.1.8, "Client Options"
- Section 2.1.9, "Upgrade"
- Section 2.1.10, "Integration with Oracle WebLogic Suite"
- Section 2.1.11, "Integration with Other Systems"
- Section 2.1.12, "Integration with Web Servers"
- Section 2.1.13, "WebLogic Server API Examples and Sample Application"

2.1.1 The WebLogic Server Solution

Oracle WebLogic Server is a scalable, enterprise-ready Java Platform, Enterprise Edition (Java EE) application server. The WebLogic Server infrastructure supports the deployment of many types of distributed applications and is an ideal foundation for building applications based on Service Oriented Architectures (SOA). SOA is a design

methodology aimed at maximizing the reuse of application services. See http://www.oracle.com/technetwork/middleware/soasuite/overview/index.html.

The WebLogic Server complete implementation of the Java EE 5.0 specification provides a standard set of APIs for creating distributed Java applications that can access a wide variety of services, such as databases, messaging services, and connections to external enterprise systems. End-user clients access these applications using Web browser clients or Java clients. It also supports the Spring Framework, a programming model for Java applications which provides an alternative to aspects of the Java EE model. See Section 2.1.2, "Programming Models."

In addition to the Java EE implementation, WebLogic Server enables enterprises to deploy mission-critical applications in a robust, secure, highly available, and scalable environment. These features allow enterprises to configure clusters of WebLogic Server instances to distribute load, and provide extra capacity in case of hardware or other failures. New diagnostic tools allow system administrators to monitor and tune the performance of deployed applications and the WebLogic Server environment itself. You can also configure WebLogic Server to monitor and tune application throughput automatically without human intervention. Extensive security features protect access to services, keep enterprise data secure, and prevent malicious attacks.

2.1.2 Programming Models

WebLogic Server provides complete support for the Java EE 5.0 specification at http://java.sun.com/javaee/5/docs/api/. For more information, see the following WebLogic Server programming guides:

- Web Applications provide the basic Java EE mechanism for deployment of dynamic Web pages based on the Java EE standards of servlets and Java ServerPages (JSP). Web applications are also used to serve static Web content such as HTML pages and image files.

- Web Services provide a shared set of functions that are available to other systems on a network and can be used as a component of distributed Web-based applications.

- XML capabilities include data exchange, and a means to store content independent of its presentation, and more.

- Java Messaging Service (JMS) enables applications to communicate with one another through the exchange of messages. A message is a request, report, and/or event that contains information needed to coordinate communication between different applications.

- Java Database Connectivity (JDBC) provides pooled access to DBMS resources.

- Resource Adapters provide connectivity to legacy and other external enterprise systems.

- Enterprise JavaBeans (EJB) provide Java objects to encapsulate data and business logic.

- Remote Method Invocation (RMI) is the Java standard for distributed object computing, allowing applications to invoke methods on a remote objects locally.

- Security APIs allow you to integrate authentication and authorization into your Java EE applications. You can also use the Security Provider APIs to create your own custom security providers.

- WebLogic Tuxedo Connectivity (WTC) provides interoperability between WebLogic Server applications and Tuxedo services. WTC allows WebLogic Server

clients to invoke Tuxedo services and Tuxedo clients to invoke EJBs in response to a service request.

- Overview of WebLogic Server Application Development describes developer tools and best practices for coding WebLogic Server applications.

- For additional programming topics, see the Developing Applications on Oracle WebLogic Server topic page.

In addition, WebLogic Server supports applications developed using the Spring Framework, an open source application framework for the Java platform. *Spring Support in Oracle WebLogic Server* provides an overview of Spring and the WebLogic Server support for developing and deploying Spring applications. It also gives examples of how to write Spring applications for WebLogic Server. See also SpringSource at `http://www.springsource.org/`.

2.1.3 High Availability

The following WebLogic Server features and tools support the deployment of highly-available and scalable applications:

- WebLogic Server clusters provide scalability and reliability for your applications by distributing the work load among multiple instances of WebLogic Server. Incoming requests can be routed to a WebLogic Server instance in the cluster based on the volume of work being processed. In case of hardware or other failures, session state is available to other cluster nodes that can resume the work of the failed node. In addition, you can implement clusters so that services may be hosted on a single machine with options to migrate the service to another node in the event of failure.

  In addition to replicating HTTP session state across servers within a cluster, WebLogic Server can also replicate HTTP session state across multiple clusters, thereby expanding availability and fault tolerance in multiple geographic regions, power grids, and Internet service providers.

- Work Managers prioritize work based on rules you define and by monitoring actual run time performance statistics. This information is then used to optimize the performance of your application. Work Mangers may be applied globally to a WebLogic Server domain or to a specific application or component.

- Overload protection gives WebLogic Server the ability to detect, avoid, and recover from overload conditions.

- Network channels facilitate the effective use of network resources by segregating network traffic into channels based on the type of traffic.

- WebLogic Server persistent store is a built-in, high-performance storage solution for WebLogic Server subsystems and services that require persistence. For example, it can store persistent JMS messages or temporarily store messages sent using the Store-and-Forward feature. The persistent store supports persistence to a file-based store or to a JDBC-enabled database.

- Store-and-forward services enables WebLogic Server to deliver messages reliably between applications that are distributed across WebLogic Server instances. If the message destination is not available at the moment the messages are sent, either because of network problems or system failures, then the messages are saved on a local server instance, and are forwarded to the remote destination once it becomes available.

- Enterprise-ready deployment tools facilitate deployment and migration of applications from the development phase to a production environment.

■ Production redeployment enables enterprises to deploy a new version of their application without interrupting work in progress on the older version.

2.1.4 System Administration

System administration of a WebLogic Server environment includes tasks such as creating WebLogic Server domains; deploying applications; migrating domains from development environments to production environments; monitoring and configuring the performance of the WebLogic Server domain; and diagnosing and troubleshooting problems. WebLogic Server provides many tools for system administrators to help with these tasks, including a browser-based Administration Console, the WebLogic Scripting Tool (WLST), a scripting language for automation of WebLogic system administration tasks based on Jython, SNMP, the Configuration Wizard, and command-line utilities. See Section 2.2.1, "Choosing the Appropriate Technology for Your Administrative Tasks" and Section 2.2.2, "Summary of System Administration Tools and APIs."

Because WebLogic Server's management system is based on Java EE and other standards, it integrates well with tools used to manage other enterprise software and hardware components. In addition, WebLogic Server implements the Java EE Java Management Extension (JMX) specification, which allows programmatic access to the WebLogic Server management system. Using this API you can create your own administration utilities or automate frequent tasks using Java classes.

2.1.5 Diagnostic Framework

The WebLogic Diagnostic Framework is a monitoring and diagnostic service that lets you create, collect, analyze, archive, and access diagnostic data generated by a running server and its deployed applications. This data provides insight into the run-time performance of WebLogic Server instances and deployed applications and lets you isolate and diagnose faults and performance bottlenecks.

2.1.6 Security

The WebLogic Server security architecture provides a comprehensive, flexible security infrastructure designed to address the security challenges of making applications available on the Web. WebLogic security can be used standalone to secure WebLogic Server applications or as part of an enterprise-wide, security management system that represents a best-in-breed security management solution. See Overview of the WebLogic Security Service.

2.1.7 Oracle JRockit JVM

The Oracle JRockit JVM is a high performance JVM optimized for Intel architectures and developed to ensure reliability, scalability, manageability, and flexibility for Java applications. Oracle JRockit JVM optimizes performance of your Java applications on either the Windows or Linux operating system platforms with either 32-bit or 64-bit architectures. Oracle JRockit JVM is included with your WebLogic Server installation. See Oracle JRockit JVM.

2.1.8 Client Options

In addition to support for browser-based Web application clients, WebLogic Server also supports a variety of client types for creating rich GUI applications or simple command-line utilities. These client types include: RMI-IIOP, T3, Java SE clients, Java

EE thin clients, CORBA/IDL clients, and C++ clients that communicate with BEA Tuxedo. See Programming Stand-alone Clients for Oracle WebLogic Server.

2.1.9 Upgrade

Tools and documentation are provided to help you migrate applications implemented on earlier versions of WebLogic Server to the current WebLogic Server environment. See the Upgrade Guide for Oracle WebLogic Server.

2.1.10 Integration with Oracle WebLogic Suite

WebLogic Server provides the core application server run time within the integrated Oracle WebLogic Suite Java infrastructure. WebLogic Suite contains the following server-side components:

- Oracle WebLogic Server

- Oracle JRockit

- Oracle Coherence

- Oracle TopLink

This integrated infrastructure enhances application performance, improves application availability, and enables predictable and reliable application scalability with high quality of service. WebLogic Suite includes highly productive development tools based on Oracle JDeveloper and Oracle Enterprise pack for Eclipse, and fully integrated management for large-scale administration and operations with Oracle Enterprise Manager. Taken together, the development, run time and management capabilities of WebLogic Suite provide the foundation for implementing mission-critical enterprise applications.

2.1.11 Integration with Other Systems

WebLogic Server provides a variety of tools to integrate your applications with disparate systems. These tools include Web Services, Resource Adapters, the JMS .NET client, Messaging Bridge, and RMI.

2.1.12 Integration with Web Servers

Plug-ins are provided with your WebLogic Server installation that allow WebLogic Server to operate with Web servers from Apache, Microsoft, and Sun Microsystems. Typically, these Web servers serve static HTML content while requests for dynamic Web content such as JSPs are directed to the WebLogic Server environment. See Using Web Server Plug-Ins with WebLogic Server.

2.1.13 WebLogic Server API Examples and Sample Application

Code examples demonstrating Java EE APIs and other WebLogic Server features are provided with your WebLogic Server installation. To work with these examples, select the custom installation option when installing WebLogic Server, and select to install the Server Examples. To access the code examples, start the WebLogic Server Examples domain using the **WebLogic Examples Server** entry in the Windows Start menu, or launch the `startWebLogicEx.cmd` or `startWebLogicEx.sh` script from `WL_HOME`/samples/domains/wl_server, where `WL_HOME` is the directory where you installed WebLogic Server. As they become available, you can also download additional examples.

Along with the code examples, two versions of a complete sample application, called Avitek Medical Records (or MedRec), are installed when you install the examples, as described above.

The original MedRec (which was included in previous versions of WebLogic Server) is a WebLogic Server sample application suite that concisely demonstrates all aspects of the Java EE platform. MedRec is designed as an educational tool for all levels of Java EE developers. It showcases the use of each Java EE component, and illustrates best practice design patterns for component interaction and client development. MedRec also illustrates best practices for developing applications on WebLogic Server.

The Spring version of MedRec, called MedRec-Spring is MedRec recast using the Spring Framework. If you are developing Spring applications on WebLogic Server, you should review the MedRec-Spring sample application. In order to illustrate how Spring can take advantage of the enterprise features of WebLogic Server, MedRec was rearchitected to replace core Java EE components with their Spring counterparts. The functionality in the original version of MedRec is reimplemented using Spring in MedRec-Spring. Refer to the MedRec-Spring sample for details.

To launch MedRec, select **Start Medical Records Server** from the Windows Start menu or run `startWebLogic.cmd` or `startWebLogic.sh` script from *WL_HOME*/samples/domains/medrec, where *WL_HOME* is the directory where you installed WebLogic Server.

To launch MedRec-Spring, select **Start Medical Records Server (Spring Version)** from the Windows Start menu or run the `startWebLogic.cmd` or `startWebLogic.sh` script from *WL_HOME*/samples/domains/medrec-spring, where *WL_HOME* is the directory where you installed WebLogic Server.

2.2 Overview of WebLogic Server System Administration

System administration of WebLogic Server includes a wide range of tasks: creating WebLogic Server domains; deploying applications; migrating domains from development environments to production environments; monitoring and managing the performance of the run-time system; and diagnosing and troubleshooting problems. (A WebLogic Server **domain** is a collection of WebLogic Server services designed for a specific purpose. For example, you might create one domain to provide an employee portal and another domain to provide business services to your customers.)

Because the WebLogic Server management system is based on Java EE and other standards, it integrates with systems that are frequently used to manage other software and hardware components. In addition, WebLogic Server includes several of its own standards-based, extensible utilities. Alternatively, you can use APIs to create custom management utilities.

The following sections provide an overview of system administration for the WebLogic Server component of your development or production environments:

- Section 2.2.1, "Choosing the Appropriate Technology for Your Administrative Tasks"

- Section 2.2.2, "Summary of System Administration Tools and APIs"

For information about installing WebLogic Server, see the *Oracle WebLogic Server Installation Guide*.

For information about using Fusion Middleware administration tools, such as the Oracle Enterprise Manager Fusion Middleware Control, Oracle Fusion Middleware command-line tools, and the Fusion Middleware Control MBean Browser, see

"Overview of Oracle Fusion Middleware Administration Tools" in *Administering Oracle Fusion Middleware*.

2.2.1 Choosing the Appropriate Technology for Your Administrative Tasks

Table 2–1 describes common system administration tasks and associated technologies.

Table 2–1   Choosing the Appropriate Management Technology

To do this...	Use this technology...
Create domains	The Configuration Wizard guides you through the process of creating or extending a domain for your target environment. See Creating WebLogic Domains Using the Configuration Wizard.
	To automate the creation of domains, use the WebLogic Scripting Tool, which is a command-line scripting interface based on Jython. See "Creating Domains Using WLST Offline" in *Oracle WebLogic Scripting Tool*.
	Or create domain configuration XML files that conform to the WebLogic Server schema. See "Domain Configuration Files" in *Understanding Domain Configuration*.
Migrate domains from development environments to production environments	Domain Template Builder's `pack` command archives a snapshot of a domain into a JAR file. The `unpack` command expands the archive and creates the necessary start scripts and certain security and configuration files. See *Creating Templates and Domains Using the Pack and Unpack Commands*.
Track changes in a domain's configuration	In environments that you allow configuration changes to active domains, WebLogic Server automatically maintains a versioned archive of configuration files. See "Configuration File Archiving" in *Understanding Domain Configuration for Oracle WebLogic Server*.
	To receive real-time notifications that a domain's configuration has been modified, enable the configuration auditing feature. See "Configuring the WebLogic Auditing Provider" in *Securing Oracle WebLogic Server*.
	For tightly controlled production environments, configure the run-time domain to be read-only (see "Restricting Configuration Changes" in *Understanding Domain Configuration for Oracle WebLogic Server*). You can change the read-only setting if you need to roll in changes that have been tested and approved in a staging environment, or you can modify and test your staging environment, and then use a Web server to re-route requests from your production environment to the staging environment.
Configure connections to Web servers	Web server plug-ins. See "Understanding Using Web Server Plug-Ins With WebLogic Server" in *Using Web Server Plug-Ins with Oracle WebLogic Server*.
Configure connections to databases or other systems	Within individual applications, you can define your own data sources or database connections using JDBC, or connect to external systems using resource adapters. When you deploy such an application, WebLogic Server creates the data sources and connections for you. See:
	■ "Configuring WebLogic JDBC Resources" in *Configuring and Managing JDBC for Oracle WebLogic Server*
	■ "Understanding Resource Adapters" in *Programming Resource Adapters for Oracle WebLogic Server*
	If you have not defined your own data sources or connections within an application, you can use the Administration Console or the WebLogic Scripting Tool to create the resources. See *Oracle WebLogic Server Administration Console Help* or "Using the WebLogic Scripting Tool" in *Oracle WebLogic Scripting Tool*.

Table 2–1   (Cont.) Choosing the Appropriate Management Technology

To do this...	Use this technology...
Manage the server life cycle	The Node Manager is a utility for remote control of Administration Servers and Managed Servers. It runs separately from WebLogic Server and lets you start up and shut down Administration Servers and Managed Servers. While use of Node Manager is optional, it provides additional life cycle benefits if your WebLogic Server environment hosts applications with high availability requirements. See Using Node Manager to Control Servers in the *Node Manager Administrator's Guide for Oracle WebLogic Server*.
	To start Administration Servers or Managed Servers without using Node Manager, use the WebLogic Scripting Tool or scripts that WebLogic Server installs. See "Starting and Stopping Servers" in *Managing Server Startup and Shutdown for Oracle WebLogic Server*.
Modify or add services to an active domain	The WebLogic Server Administration Console provides a graphical user interface for modifying or adding services to an active domain. See *Administration Console Help*.
	If you prefer a command-line interface, use the WebLogic Scripting Tool in interactive mode. See "Using the WebLogic Scripting Tool" in *Oracle WebLogic Scripting Tool*.
Monitor application server services and resources	Monitor the performance of services such as the EJB container, servlet container, and JDBC data sources from the WebLogic Server Administration Console.
	Configure watch rules and notifications in the WebLogic Diagnostics Framework to automatically notify administrators of monitoring data events or integrate automated systems through JMX or JMS. See "Configuring Watches and Notifications" in *Configuring and Using the Diagnostics Framework for Oracle WebLogic Server*.
	If you use SNMP in your operations center, you can enable WebLogic Server to send SNMP notifications for run-time events that you define. See *SNMP Management Guide for Oracle WebLogic Server*.
Deploy applications	The WebLogic Server Administration Console provides a series of Web-based deployment assistants that guide you through the deployment process. See *Administration Console Help*.
	To automate the deployment of applications, use the WebLogic Scripting Tool. See "Deployment Commands" in *WebLogic Scripting Tool Command Reference*. You can also use the deployment API to write Java programs that deploy applications. See *Programming Deployment for Oracle WebLogic Server*.
	For information about additional deployment utilities and APIs, see "Deployment Tools" in *Deploying Applications to Oracle WebLogic Server*.
Modify applications in an active domain	To modify the configuration of a deployed application, use a text editor or IDE to modify the deployment descriptor. Then either redeploy the application or use the deployment API to upload the modified deployment descriptor and cause the application container to re-read the deployment descriptor.
	See *Deploying Applications to Oracle WebLogic Server*.
Monitor activity within applications	Determine which data points you want to monitor and then instrument one or more beans to expose this data through JMX. See *Developing Manageable Applications With JMX for Oracle WebLogic Server*.
	Alternatively, use the WebLogic Server Diagnostics Service to insert instrumentation code into a running application and monitor its methods or monitor transactions that involve the application. Use this technology to discover the cause of problems that cannot otherwise be discovered by scanning the available monitoring metrics. If you determine that the problem is within your application, you can prevent the problem from recurring by using JMX to expose attributes that indicate the application's health state is degrading. See *Configuring and Using the Diagnostics Framework for Oracle WebLogic Server*.

Table 2–1   (Cont.) Choosing the Appropriate Management Technology

To do this...	Use this technology...
Optimize the performance of your application and maintain service level agreements.	Work Managers configure how your application prioritizes the execution of its work. Based on rules you define and by monitoring actual run-time performance, WebLogic Server can optimize the performance of your application and maintain service level agreements.
	See "Using Work Managers to Optimize Scheduled Work" in *Configuring Server Environments for Oracle WebLogic Server*.
Configure and secure administration communications	You can separate administration traffic from application traffic in your domain by enabling the administration port. In production environments, separating the two forms of traffic ensures that critical administration operations (starting and stopping servers, changing a server's configuration, and deploying applications) do not compete with high-volume application traffic on the same network connection.
	The administration port only accepts communications that use SSL, and therefore secures your administrative requests. See "Administration Port and Administrative Channel" in *Configuring Server Environments for Oracle WebLogic Server*.
Configure logging and view log files	Many WebLogic Server operations generate logs of their activity. Each server has its own log as well as a standard HTTP access log. These log files can be configured and used in a variety of ways to monitor the health and activity of your servers and applications.
	By default, WebLogic Server uses the standard JDK logging APIs to filter and write the messages to log files. See "Understanding WebLogic Logging Services" in *Configuring Log Files and Filtering Log Messages for Oracle WebLogic Server*.
	Alternatively, you can configure WebLogic Server to use the Jakarta Project Log4j APIs to distribute log messages. See Log4j and the Commons Logging API in *Configuring Log Files and Filtering Log Messages*.

2.2.2 Summary of System Administration Tools and APIs

WebLogic Server includes several of its own standards-based, extensible utilities that you can use to create, manage, and monitor domains, or you can use WebLogic Server's management APIs to create custom management utilities.

Table 2–2 describes the utilities that are included with WebLogic Server.

Table 2–2   Management Utilities

Utility	Description
Administration Console	The Administration Console is a Web application hosted by the Administration Server. Use it to manage and monitor an active domain. The management capabilities include:
	■ Configuring active domains
	■ Stopping and starting servers
	■ Monitoring server health and performance
	■ Monitoring application performance
	■ Viewing server logs
	Through the Administration Console, system administrators can easily perform all WebLogic Server management tasks without having to learn the JMX API or the underlying management architecture. The Administration Server persists changes to attributes in the config.xml file for the domain you are managing.
	See:
	■ Section 2.3.2, "Starting the Administration Console"
	■ Administration Console Online Help (The online help is also available from the Administration Console by clicking on the **Help** link located in the tool bar at the top of the Console.)
WebLogic Scripting Tool	The WebLogic Scripting Tool (WLST) is a command-line scripting interface that you use to manage and monitor active or inactive WebLogic Server domains. The WLST scripting environment is based on the Java scripting interpreter Jython. In addition to WebLogic scripting functions, you can use common features of interpreted languages, including local variables, conditional variables, and flow control statements. You can extend the WebLogic scripting language by following the Jython language syntax. See http://www.jython.org.
	See *Oracle WebLogic Scripting Tool*.
Configuration Wizard	The Configuration Wizard creates the appropriate directory structure for a WebLogic Server domain, a config.xml file, and scripts you can use to start the servers in your domain. The wizard uses templates to create domains, and you can customize these templates to duplicate your own domains.
	You can also use the Configuration Wizard to add or remove services from an existing, inactive domain.
	You can run the Configuration Wizard through a graphical user interface (GUI) or in a text-based command-line environment. This command-line environment is called *console mode*—do not confuse this mode with the Administration *Console*. You can also create user-defined domain configuration templates for use by the Configuration Wizard.
	See *Creating Domains Using the Configuration Wizard*.

Table 2–2   (Cont.) Management Utilities

Utility	Description
Configuration Template Builder	The Configuration Template Builder provides the capability to easily create your own domain templates, to enable, for example, the definition and propagation of a standard domain across a development project, or to enable the distribution of a domain along with an application that has been developed to run on that domain. The templates you create with the Configuration Template Builder are used as input to the Configuration Wizard as the basis for creating a domain that is customized for your target environment. See *Creating Domain Templates Using the Domain Template Builder*.
Apache Ant tasks	You can use two Ant tasks provided with WebLogic Server to help you perform common configuration tasks in a development environment. Ant is a Java-based build tool similar to Make. The configuration tasks let you start and stop WebLogic Server instances as well as create and configure WebLogic Server domains. When combined with other WebLogic Ant tasks, you can create powerful build scripts for demonstrating or testing your application with custom domains.
	See "Using Ant Tasks to Configure a WebLogic Server Domain" in *Developing Applications for Oracle WebLogic Server*.
SNMP Agents	WebLogic Server includes the ability to communicate with enterprise-wide management systems using Simple Network Management Protocol (SNMP). WebLogic Server SNMP agents let you integrate management of WebLogic Servers into an SNMP-compliant management system that gives you a single view of the various software and hardware resources of a complex, distributed system.
	See *SNMP Management Guide for Oracle WebLogic Server*.

Table 2–3 describes APIs that you can use to create your own management utilities.

Table 2–3   Management APIs

API	Description
JMX	Java Management Extensions (JMX) is the Java EE solution for monitoring and managing resources on a network. Like SNMP and other management standards, JMX is a public specification and many vendors of commonly used monitoring products support it.
	The Administration Console, WebLogic Scripting Tool, and other WebLogic Server utilities use the JMX APIs.
	See *Developing Custom Management Utilities With JMX for Oracle WebLogic Server*.
Java EE Management API	The Java EE Management APIs (JSR-77) enable a software developer to create a single Java program that can discover and browse resources, such as JDBC connection pools and deployed applications, on any Java EE Web application server. The APIs are part of the Java EE Management Specification, which requires all Java EE Web application servers to describe their resources in a standard data model.
	See *Monitoring and Managing With the Java EE Management APIs for Oracle WebLogic Server*.

Table 2–3 (Cont.) Management APIs

API	Description
Deployment API	The WebLogic Server deployment API implements and extends the JSR-88 deployment specification. All WebLogic Server deployment tools, such as the Administration Console and wldeploy Ant task, use the deployment API to configure, deploy, and redeploy applications in a domain. You can use the deployment API to build your own WebLogic Server deployment tools, or to integrate WebLogic Server configuration and deployment operations with an existing JSR-88-compliant tool. See *Programming Deployment for Oracle WebLogic Server*.
WebLogic Diagnostic Service APIs	The WebLogic Diagnostic Service includes a set of standardized APIs that enable dynamic access and control of diagnostic data, as well as improved monitoring that provides visibility into the server. The interfaces are standardized to facilitate future enhancement and integration of third-party tools, while maintaining the integrity of the server code base. The service is well suited to the server and the server's stack product components and targets operations and administrative staff as primary users. See *Configuring and Using the Diagnostics Framework for Oracle WebLogic Server*.
Logging APIs	By default, WebLogic Server uses the standard JDK logging APIs to filter and write the messages to log files. See Understanding WebLogic Logging Services in *Configuring Log Files and Filtering Log Messages for Oracle WebLogic Server*. Alternatively, you can configure WebLogic Server to use the Jakarta Project Log4j APIs to distribute log messages. For more information, see Log4j and the Commons Logging API in *Configuring Log Files and Filtering Log Messages for Oracle WebLogic Server*.

2.3 Overview of the Administration Console

For detailed information on using the Administration Console, see the Administration Console Online Help, or click Help from any Administration Console page.

The following sections provide an overview of the Administration Console:

- Section 2.3.1, "About the Administration Console"
- Section 2.3.2, "Starting the Administration Console"
- Section 2.3.3, "Elements of the Administration Console"
- Section 2.3.4, "Using the Change Center"

2.3.1 About the Administration Console

The Administration Console is a Web browser-based, graphical user interface that you use to manage a WebLogic Server domain. A WebLogic Server domain is a logically related group of WebLogic Server resources that you manage as a unit. A domain includes one or more WebLogic Servers and may also include WebLogic Server clusters. Clusters are groups of WebLogic Servers instances that work together to provide scalability and high-availability for applications. You deploy and manage your applications as part of a domain.

One instance of WebLogic Server in each domain is configured as an Administration Server. The Administration Server provides a central point for managing a WebLogic Server domain. All other WebLogic Server instances in a domain are called Managed Servers. In a domain with only a single WebLogic Server instance, that server functions both as Administration Server and Managed Server. The Administration Server hosts the Administration Console, which is a Web application accessible from

any supported Web browser with network access to the Administration Server. Managed Servers host applications.

Use the Administration Console to:

- Configure, start, and stop WebLogic Server instances

- Configure WebLogic Server clusters

- Configure WebLogic Server services, such as database connectivity (JDBC) and messaging (JMS)

- Configure security parameters, including managing users, groups, and roles

- Configure and deploy your applications

- Monitor server and application performance

- View server and domain log files

- View application deployment descriptors

- Edit selected run-time application deployment descriptor elements

2.3.1.1 Administration Console Online Help

The Administration Console includes a complete help system. It has two parts:

- How do I...?, which documents procedures for tasks you can perform through using the Console.

- Administration Console Reference, which provides reference information for each page in the Console, including descriptions of the attributes you can set using the Console.

You can access the Administration Console online help either through the Console itself, or online at `http://download.oracle.com/docs/cd/E15523_01/apirefs.1111/e13952/core/index.html`.

2.3.1.2 Console Errors

Messages (including information, warning, and error messages) can be generated and logged in the course of using the Administration Console. You can view WebLogic Server logs from the **Diagnostics** > **Log Files** page of the Console.

2.3.2 Starting the Administration Console

This section contains instructions for starting the Administration Console.

To use the Administration Console, use one of the supported Web browsers for your environment. For more information, see "Oracle Fusion Middleware Supported System Configurations" at `http://www.oracle.com/technetwork/middleware/ias/downloads/fusion-certification-100350.html`. If your Web browser is not on this list of supported browsers, you may experience functional or formatting problems when using the Administration Console.

To start the Administration Console:

1. Start an Administration Server.

2. Open one of the supported Web browsers to the following URL:

   ```
   http://hostname:port/console
   ```

where *hostname* is the DNS name or IP address of the Administration Server and *port* is the listen port on which the Administration Server is listening for requests (port 7001 by default). If you have configured a domain-wide administration port, use that port number. If you configured the Administration Server to use Secure Socket Layer (SSL) you must add *s* after `http` as follows:

```
https://hostname:port/console
```

> **Note:** A domain-wide administration port always uses SSL.

3. When the login page appears, enter the user name and the password you used to start the Administration Server (you may have specified this user name and password during the installation process) or enter a user name that belongs to one of the following security groups: Administrators, Operators, Deployers, or Monitors. These groups provide various levels of access to system administration functions in the Administration Console.

   Using the security system, you can add or delete users to one of these groups to provide controlled access to the Console.

   > **Note:** If you have your browser configured to send HTTP requests to a proxy server, then you may need to configure your browser to not send Administration Server HTTP requests to the proxy. If the Administration Server is on the same machine as the browser, then ensure that requests sent to localhost or 127.0.0.1 are not sent to the proxy.

2.3.2.1 Enabling the Administration Console

By default, the Administration Console is enabled. If you disable it, you can re-enable it using the WebLogic Scripting Tool (WLST). Start the Administration Server, then invoke WLST and use the following commands:

Example 2–1   Using WLST to Re-enable the Console

```
connect("username","password")
edit()
startEdit()
cmo.setConsoleEnabled(true)
save()
activate()
The following attribute(s) have been changed on MBeans which require server
re-start.
MBean Changed : com.bea:Name=mydomain,Type=Domain Attributes changed :
ConsoleEnabled
Activation completed
disconnect()
exit()
```

2.3.3 Elements of the Administration Console

The Administration Console user interface includes the following panels.

2.3.3.1 Change Center

This is the starting point for using the Administration Console to make changes in WebLogic Server. See Section 2.3.4, "Using the Change Center."

Figure 2–1   Change Center
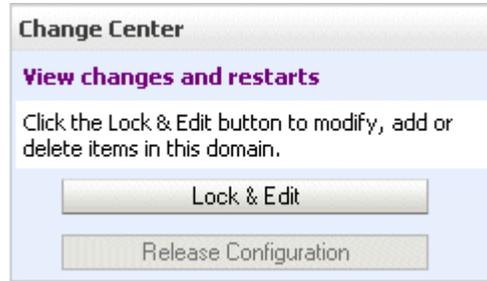
2.3.3.2 Domain Structure

This panel contains a tree structure you can use to navigate to pages in the Administration Console. Select any of the nodes in the Domain Structure tree to view that page. Click a + (plus) icon in the Domain Structure to expand a node and a - (minus) icon to collapse the node.

Figure 2–2   Domain Structure
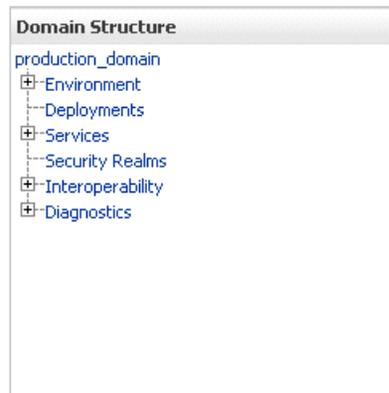
2.3.3.3 How do I...

This panel includes links to online help tasks that are relevant to the current Console page.

Figure 2–3   How do I...

2.3.3.4 Tool Bar

Tool Bar Element	Description
Welcome message	Indicates user name with which you have logged into the Console.
Connected to:	The IP address and port you used to connect to the Console.
Home	A link to the top page of the Console.
Log Out	Click to log out of the Console.
Preferences	A link to a page where you can change some Console behavior.
Record	Starts recording your configuration actions as a series of WebLogic Scripting Tool (WLST) commands. Writes the commands to a separate file that you can replay in WLST.
	See "Record WLST Scripts" in *Oracle WebLogic Server Administration Console Help*.
Help	A link to the Administration Console Online Help.
Search	A text field in which you can enter a string to find any WebLogic Server Configuration MBeans that contain the string you specified in their name.

The tool bar at the top of the Console includes the following elements:

Figure 2–4   Tool Bar

2.3.3.5 Breadcrumb Navigation

A series of links that show the path you have taken through the Administration Console's pages. You can click on any of the links to return to a previously-visited page.

Figure 2–5   Breadcrumb Navigation

2.3.3.6 System Status

The System Status panel reports on the number of information, error, and warning messages that have been logged. You can view these messages in the server log files, which you can access from the Administration Console at **Diagnostics** > **Log Files**.

Figure 2–6   System Status
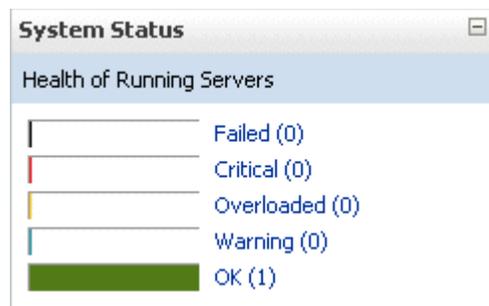
2.3.4 Using the Change Center

The starting point for using the Administration Console to make changes in your WebLogic Server domain is the Change Center. The Change Center provides a way to lock a domain configuration so you can make changes to the configuration while preventing other accounts from making changes during your edit session. See Section 2.3.4.3, "How Change Management Works."

The domain configuration locking feature is always enabled in production domains. It can be enabled or disabled in development domains. It is disabled by default when you create a new development domain. See "Enable and disable the domain configuration lock" in *Oracle WebLogic Server Administration Console Help*.

To change a production domain's configuration, you must:

1. Locate the Change Center in the upper left of the Administration Console screen.

2. Click the **Lock & Edit** button to lock the configuration edit hierarchy for the domain.

3. Make the changes you desire on the relevant page of the Console. Click **Save** on each page where you make a change.

4. When you have finished making all the desired changes, click **Activate Changes** in the Change Center.

As you make configuration changes using the Administration Console, you click Save (or in some cases Finish) on the appropriate pages. This does not cause the changes to take effect immediately. The changes take effect when you click Activate Changes in the Change Center. At that point, the configuration changes are distributed to each of the servers in the domain. If the changes are acceptable to each of the servers, then they take effect. If any server cannot accept a change, then all of the changes are rolled back from all of the servers in the domain. The changes are left in a pending state; you can then either edit the pending changes to resolve the problem or revert the pending changes.

2.3.4.1 Undoing Changes

You can revert any pending (saved, but not yet activated) changes by clicking Undo All Changes in the Change Center. You can revert any individual change by going to the appropriate page in the Administration Console and restoring the attribute to its previous value.

2.3.4.2 Releasing the Configuration Lock

You release the configuration lock as follows:

- Before you make changes, click **Release Configuration** in the Change Center to release the lock explicitly.

- After you save changes, click **Activate Changes** or **Undo All Changes** in the Change Center to release the lock implicitly.

Stopping the Administration Server does not release the configuration lock. When the Administration Server starts again, the configuration lock is in the same state it was in when the Administration Server was shut down, and any pending changes are preserved.

2.3.4.3 How Change Management Works

To provide a secure, predictable means for distributing configuration changes in a domain, WebLogic Server imposes a change management process that loosely resembles a database transaction. The configuration of a domain is represented on the file system by a set of XML configuration files, centralized in the `config.xml` file, and at run time by a hierarchy of Configuration MBeans. When you edit the domain configuration, you edit a separate hierarchy of Configuration MBeans that resides on the Administration Server. To start the edit process, you obtain a lock on the edit hierarchy to prevent other people from making changes. When you finish making changes, you save the changes to the edit hierarchy. The changes do not take effect, however, until you activate them, distributing them to all server instances in the domain. When you activate changes, each server determines whether it can accept the change. If all servers are able to accept the change, they update their working configuration hierarchy and the change is completed.

For more information about change management, see "Managing Configuration Changes" in *Understanding Domain Configuration for Oracle WebLogic Server*.

2.3.4.4 Dynamic and Non-Dynamic Changes

Some changes you make in the Administration Console take place immediately when you activate them. Other changes require you to restart the server or module affected by the change. These latter changes are called *non-dynamic changes*. Non-dynamic changes are indicated in the Administration Console with this warning icon:

Changes to dynamic configuration attributes become available once they are activated, without restarting the affected server or system restart. These changes are made available to the server and run-time hierarchies once they are activated. Changes to non-dynamic configuration attributes require that the affected servers or system resources be restarted before they become effective.

If a change is made to a non-dynamic configuration setting, no changes to dynamic configuration settings will take effect until after restart. This is to assure that a batch of updates having a combination of dynamic and non-dynamic attribute edits will not be partially activated.

Note that WebLogic Server's change management process applies to changes in domain and server configuration data, not to security or application data.

2.3.4.5 Viewing Changes

You can view any changes that you have saved, but not yet activated, by clicking the View Changes and Restarts link in the Change Center. The View Changes and Restarts link presents two tabs, Change List and Restart Checklist:

- The Change List page presents all changes that have been saved, but not yet activated.

- The Restart Checklist lists all servers for which non-dynamic changes have been activated, but which require restarts before the changes become effective.

2.3.4.6 Deploying Multiple Applications

When you use the Administration Console to deploy multiple applications, upon installing the applications, they are listed in the Console's Deployments page in the "distribute Initializing" state. After activating changes, they are listed in the "Prepared" state. To deploy the applications, select the application names on the Deployments page and click **Start**.