Oracle® Fusion Middleware

Administering Server Startup and Shutdown for Oracle WebLogic Server

12*c* (12.2.1.3.0)
E80419-05
July 2020

ORACLE

Oracle Fusion Middleware Administering Server Startup and Shutdown for Oracle WebLogic Server, 12*c* (12.2.1.3.0)

E80419-05

Contents

5 Understanding Server Life Cycle

Preface

This preface describes the document accessibility features and conventions used in this guide—*Administering Server Startup and Shutdown for Oracle WebLogic Server* .

Documentation Accessibility

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

Access to Oracle Support

Oracle customers 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 organization of this guide—*Administering Server Startup and Shutdown for Oracle WebLogic Server*.
This chapter includes the following sections:

- Document Scope and Audience
- Guide to This Document
- Related Documentation
- New and Changed Features for Managing Server Life Cycle

1.1 Document Scope and Audience

This document describes how you manage Oracle WebLogic Server startup, shutdown, and server life cycle. It also describes WebLogic features that help you prevent and recover from server failure.

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.

It is assumed that the reader is familiar with Java Platform, Enterprise Edition (Java EE) and Web technologies, object-oriented programming techniques, and the Java programming language.

1.2 Guide to This Document

The document is organized as follows:

- Introduction and Roadmap describes the scope of the guide and lists related documentation.
- Starting and Stopping Servers describes several ways to start and stop server instances.
- Setting Up a WebLogic Server Instance as a Windows Service describes setting up a WebLogic Server instance as a Windows service on a Windows host computer.
- Avoiding and Recovering From Server Failure describes failover procedures for WebLogic Server instances.
- Understanding Server Life Cycle describes the operational phases of a WebLogic Server instance, from startup to shutdown.
- Starting and Stopping Servers: Quick Reference provides simple procedures for starting and stopping WebLogic Server instances.

1.3 Related Documentation

- *Creating WebLogic Domains Using the Configuration Wizard*
- *Understanding Domain Configuration for Oracle WebLogic Server*
- *Oracle WebLogic Server Administration Console Online Help*

1.4 New and Changed Features for Managing Server Life Cycle

For a comprehensive listing of the new WebLogic Server features introduced in this release, see *What's New in Oracle WebLogic Server 12.2.1.3.0*.

2
Starting and Stopping Servers

There are many different ways you can start and stop WebLogic Server instances. You can use the WebLogic Server Administration Console, a command window, a script, or Node Manager. No matter how you start a server, the end result passes a set of configuration options to initialize a Java Virtual Machine (JVM). The server instance runs within the JVM, and the JVM can host only one server instance.

For a concise overview of starting and stopping servers, see Starting and Stopping Servers: Quick Reference.

> **Note:**
>
> For procedures that require the WebLogic Server Administration Console, see Start and stop servers and various startup and shutdown procedures in the Cluster section of the *Oracle WebLogic Server Administration Console Online Help*. For information on restarting failed server instances and clusters, see Section 4, Avoiding and Recovering From Server Failure.

2.1 Starting Servers: Before You Begin

Before starting a server instance, you must perform the prerequisite steps to set up the server environment.

Depending on the method you choose to manage server startup and what setup tasks you have already performed, you might need to complete the following procedures before you can start server instances:

- Meet version requirements — Version Requirements for a Domain.
- Create a domain — Choosing the Appropriate Technology for Your Administrative Tasks in *Understanding Oracle WebLogic Server.*
- Provide user credentials — Provide User Credentials to Start and Stop Servers.
- Set up Node Manager — Node Manager Overview in the *Administering Node Manager for Oracle WebLogic Server*.
- Configure Managed Server connections to the Administration Server — Configuring Managed Server Connections to the Administration Server.
- Specify Java startup options — Specifying Java Options for a WebLogic Server Instance.

2.2 Version Requirements for a Domain

All WebLogic Server instances within the same administrative domain must be at the same major and minor version. You cannot mix server versions within a domain.

Server instances within a domain can be at different patch set levels as long as the Administration Server is at the same patch set level or higher than its Managed Servers. For example, if the Managed Servers are at version 10.3.0, then the Administration Server can be either version 10.3.0, 10.3.1, or higher. However, if the Managed Servers are at 10.3.1, then the Administration Server must be at 10.3.1 or higher. Also, all server instances within a cluster must be at the same patch set level.

2.3 Starting an Administration Server with a Startup Script

Use the `startWebLogic` script to start an Administration Server. It sets the environment variables and starts a Java Virtual Machine (JVM) to run a WebLogic Server instance.

An Administration Server is a WebLogic Server instance that maintains configuration data for a domain. In a development environment, it is usually sufficient to start an Administration Server and deploy your applications directly onto the Administration Server. In a production environment, you create Managed Servers to run applications. See Understanding WebLogic Server Domains in *Understanding Domain Configuration for Oracle WebLogic Server*.

You can start an Administration Server with a default startup script or create your own. To start an Administration Server with the WebLogic Server-included startup script:

1. If you have not already done so, use the Configuration Wizard or WebLogic Scripting Tool (WLST) to create a domain.

   See Creating WebLogic Domains Using Configuration Wizard or Creating Domains Using WLST Offline in *Understanding the WebLogic Scripting Tool*.

2. Open a shell (command prompt) on the computer on which you created the domain.
3. Change to the directory in which you located the domain.

   By default, this directory is *ORACLE_HOME*`\user_projects\domains\`*`DOMAIN_NAME`*, where *`DOMAIN_NAME`* is

the root directory of the domain. (The name of this directory is the name of the domain.)

4. Run one of the following scripts:
   - `bin/startWebLogic.cmd` (Windows)
   - `bin\startWebLogic.sh` (UNIX and Windows. On Windows, this script supports the MKS and Cygnus BASH UNIX shell emulators.)

> **Note:**
>
> If you use a Configuration Wizard template that is provided by WebLogic Server, your domain directory includes a start script named startWebLogic. If you use a domain template from another source, the wizard might not create a start script, or it might create a script with a different name. The template designer determines whether the wizard creates a start script and the name of the script.
>
> The following error occurs when using `startWebLogic.cmd` to boot WebLogic Server:
>
> ```
> Enter username to boot WebLogic server:weblogic
> <Error> <Security> <BEA-090782> <Server is Running in Production Mode
> and Native Library(terminalio)
>  to read the password securely from commandline is not found>
> <Notice> <WebLogicServer> <BEA-000388> <JVM called WLS shutdown hook.
> The server will force shutdown now>
> <Alert> <WebLogicServer> <BEA-000396> <Server shutdown has been
> requested by <WLS Kernel>>
> <Notice> <WebLogicServer> <BEA-000365> <Server state changed to
> FORCE_SHUTTING_DOWN>
> This error is caused by installing a 32-bit JDK on a 64-bit OS. To
> avoid this error, install a 64-bit JDK on a 64-bit OS.
> ```

The startWebLogic script does the following:

1. Sets environment variables by invoking *DOMAIN_NAME*`\bin\setDomainEnv.cmd` (`setDomainEnv.sh` on UNIX), where *DOMAIN_NAME* is the directory in which you located the domain; for example, *ORACLE_HOME*`\user_projects\domains\`*DOMAIN_NAME*, and where *ORACLE_HOME* is the directory you specified as Oracle Home when you installed Oracle WebLogic Server.

   > **Note:**
   >
   > `setDomainEnv` is designed to be sourced from other scripts, such as the `startWebLogic` script. `setDomainEnv` should not be called directly from within an interactive shell. Doing so can cause unpredictable issues in the domain.

2. Invokes the `java weblogic.Server` command, which starts a JVM that is configured to run a WebLogic Server instance.

When the server successfully completes its startup process, it writes the following message to standard out (which, by default, is the command window):

```
<Notice> <WebLogicServer> <BEA-000360> <Server started in RUNNING mode>
```

2.4 Starting an Administration Server with the java weblogic.Server Command

To start an Administration Server from the command line, use the `java weblogic.Server` command.

The `weblogic.Server` class is the main class for a WebLogic Server instance. You start a server instance by directly invoking `weblogic.Server` in a Java command.

> **Note:**
>
> Oracle recommends using `java weblogic.Server` primarily for initial development but not as a standard mechanism for starting production systems for the following reasons:
>
> - `java weblogic.Server` will not function if you select a product directory outside of the Oracle Middleware Home directory.
> - When executing `java weblogic.Server`, patches will not be recognized by the WebLogic Server run time.

See weblogic.Server Command-Line Reference and Using the weblogic.Server Command Line to Start a Server Instance in *Command Reference for Oracle WebLogic Server*.

2.5 Starting an Administration Server Using WLST and Node Manager

Node Manager is a utility for the remote control of WebLogic Server instances. Use the WLST `nmStart` command to start an Administration Server.

If you use the nmStart command with WLST connected to a Node Manager, Node Manager supports monitoring, stopping, and restarting the Administration Server.

Using WLST and Node Manager to Manage Servers in *Understanding the WebLogic Scripting Tool* describes how to start the Administration Server with WLST and Node Manager. How Node Manager Starts an Administration Server in *Administering Node Manager for Oracle WebLogic Server* describes how Node Manager accomplishes this process.

Oracle recommends running Node Manager as an operating system service so that it automatically restarts in the event of system failure or reboot, and using Node Manager to start and restart both Administration and Managed Servers.

See Running Node Manager as a Startup Service in the *Administering Node Manager for Oracle WebLogic Server*.

2.6 Starting an Administration Server Using WLST Without Node Manager

Use the WLST `startServer` command, to start the Administration Server without using Node Manager.

The WLST `startServer` command starts the Administration Server without using Node Manager. The server runs in a separate process from WLST; exiting WLST does not shut down the server. See Starting an Administration Server Without Node Manager in *Understanding the WebLogic Scripting Tool*.

2.7 Starting Managed Servers with a Startup Script

Use the `startManagedWebLogic` script to start Managed Servers. The script uses a Java command to set the environment variables and start a Java Virtual Machine (JVM) that runs a WebLogic Server instance.

Managed Servers host business applications, components, Web services, and their associated resources. A Managed Server is a WebLogic Server instance that runs deployed applications. It refers to the Administration Server for all of its configuration and deployment information. Usually, you use Managed Servers to run applications in a production environment.

See Understanding WebLogic Server Domains in *Understanding Domain Configuration for Oracle WebLogic Server*.

If you use one of the Configuration Wizard templates that WebLogic Server provides, your domain directory includes a start script named `startManagedWebLogic` that you can use to start Managed Servers. You can use this script to start all the Managed Servers in a cluster.

See Domain Configuration Files in *Understanding Domain Configuration for Oracle WebLogic Server*.

This script does not use the Node Manager to start and manage the server. Instead, it uses a Java command to invoke the `weblogic.Server` class, which is the main class for a WebLogic Server instance. For information about invoking `weblogic.Server` in a Java command, see weblogic.Server Command-Line Reference in *Command Reference for Oracle WebLogic Server*.

To use the WebLogic Server scripts to start Managed Servers:

1. Refer to Starting Servers: Before You Begin for prerequisite tasks.
2. If you have not already done so, create one or more Managed Servers.

   See *Creating WebLogic Domains Using the Configuration Wizard* or Create Managed Servers in the *Oracle WebLogic Server Administration Console Online Help*.
3. If you have not already done so, start the domain's Administration Server.
4. In a shell (command prompt) on the computer that hosts the Managed Server, change to the directory that contains the `startManagedWebLogic` script:
   - *`DOMAIN_NAME`*`\bin\startManagedWebLogic.cmd` (Windows)

- *DOMAIN_NAME*/bin/startManagedWebLogic.sh (UNIX)

where *DOMAIN_NAME* is the directory in which you located the domain. By default, this directory is *ORACLE_HOME*\user_projects\domains*DOMAIN_NAME*.

5. Enter one of the following commands:
   - startManagedWebLogic.cmd *managed_server_name admin_url* (Windows)
   - startManagedWebLogic.sh *managed_server_name admin_url* (UNIX)

   where *managed_server_name* specifies the name of the Managed Server and *admin_url* specifies the listen address (host name, IP address, or DNS name) and port number of the domain's Administration Server.

   For example, the following command uses startManagedWebLogic.cmd to start a Managed Server named myManagedServer. The listen address for the domain's Administration Server is AdminHost:7001:

   ```
   c:\Oracle\Middleware\user_projects\domains\mydomain\bin\startManagedWebLogic.
   cmd myManagedServer http://AdminHost:7001
   ```

6. For each Managed Server that you want to start, open a separate command shell and follow steps 4 and 5. If you are starting Managed Servers on another machine, log in to that machine (remotely or locally) and then follow steps 4 and 5.

For information on running Managed Servers on a remote WebLogic Server host, see Creating and Starting a Managed Server on a Remote Machine in *Creating Templates and Domains Using the Pack and Unpack Commands*.

See Configuring Managed Server Connections to the Administration Server.

The startManagedWebLogic script does the following:

1. Calls the startWebLogic script, which sets the environment variables by invoking *ORACLE_HOME*\user_projects\domains*DOMAIN_NAME*\bin\setDomainEnv.cmd (setDomainEnv.sh on UNIX), where *ORACLE_HOME* is the directory you specified as Oracle Home when you installed Oracle WebLogic Server.

   > **Note:**
   >
   > setDomainEnv is designed to be sourced from other scripts, such as the startWebLogic script. setDomainEnv should not be called directly from within an interactive shell. Doing so can cause unpredictable issues in the domain.

2. Invokes the java weblogic.Server command, which starts a JVM that is configured to run a WebLogic Server instance.

   When the server successfully completes its startup process, it writes the following message to standard out (which, by default, is the command window):

   ```
   <Notice> <WebLogicServer> <000360> <Server started in RUNNING mode>
   ```

2.8 Starting Managed Servers from the Administration Console

Use the WebLogic Server Administration Console to start an Administration Server and its Managed Servers.

See Start Managed Servers from the Administration Console in the *Oracle WebLogic Server Administration Console Online Help*.

2.9 Starting Managed Servers and Clusters with WLST and Node Manager

Use WLST and Node Manager to start Managed Server instances in a WebLogic cluster.

To start Managed Servers and clusters using WLST and Node Manager, see Using Node Manager to Start Managed Servers in a Domain or Cluster in *Understanding the WebLogic Scripting Tool*. See Setting up WebLogic Clusters in *Administering Clusters for Oracle WebLogic Server*.

2.10 Starting Managed Servers with the java weblogic.Server Command

Use the `java weblogic.Server` command to start Managed Servers.

The `weblogic.Server` class is the main class for a WebLogic Server instance. You start a server instance by directly invoking `weblogic.Server` in a Java command. See weblogic.Server Command-Line Reference and Using the weblogic.Server Command Line to Start a Server Instance in *Command Reference for Oracle WebLogic Server*.

2.11 Starting a Managed Server When the Administration Server Is Unavailable

You can start a Managed Server in Managed Server Independence (MSI) mode if it is unable to connect to the Administration Server during startup.

A Managed Server contacts the Administration Server during its startup sequence to retrieve its configuration information. If a Managed Server cannot connect to the Administration Server during startup, it can retrieve its configuration by reading its locally cached configuration data from the `config` directory.

> **Note:**
>
> The first time you start a Managed Server instance, it must be able to contact the Administration Server. Thereafter, the Managed Server instance can start even if the Administration Server is unavailable.

See Starting a Managed Server When the Administration Server Is Not Accessible.

2.12 Provide User Credentials to Start and Stop Servers

To start and stop WebLogic Server instances, you must provide the credentials of a user who is permitted to start and stop servers for the domain.

See Users, Groups, and Security Roles in *Securing Resources Using Roles and Policies for Oracle WebLogic Server*.

Table 2-1 describes providing user credentials when starting a WebLogic Server instance.

Table 2-1 Providing User Credentials

If you specify this...	The server instance does this...
User Name and password on the command line.	Uses them and does not prompt you for either credential.
User Name and password in `boot.properties`.	Uses them and does not prompt you for either credential.
Neither user name nor password on the command line.	• Prompts you for the user name. • Prompts you for the password twice.
User Name but no password on the command line.	• Uses the user name from the command line. • Prompts you for the password twice.
Password but no user name on the command line.	• Prompts you for the user name. • Ignores the password from the command line and prompts you for the password twice.

These sections describe the following tasks:

- Specifying an Initial Administrative User for a Domain
- Boot Identity Files
- Specifying User Credentials for Starting a Server with Node Manager
- Changing the Credentials Used for Starting a Server

2.12.1 Specifying an Initial Administrative User for a Domain

When you create a domain, the Configuration Wizard prompts you to provide the user name and password for an initial administrative user. The Configuration Wizard does the following with this information:

1. Assigns the user to the Administrators security group.

   The Administrators group grants the highest level of privileges for starting and managing WebLogic Server. See Users, Groups, And Security Roles in *Securing Resources Using Roles and Policies for Oracle WebLogic Server*.

2. Adds the user to the `myrealm` security realm.

   A security realm is a collection of components (providers) that authenticate user names, determine the type of resources that the user can access, and provide other security-related services for WebLogic resources. WebLogic Server installs the `myrealm` security realm and uses it by default.

   You can use the Administration Console to add users to security realms. If you use an Authentication provider other than the one that WebLogic Server installs, you must use the provider's administration tools to create at least one user with administrative privileges.

3. If you are creating a domain in development mode, the wizard creates a boot identity file in the `security` directory of the Administration Server's root directory. The boot identity file contains an encrypted version of the user name and password which lets you bypass the login prompt during subsequent instantiations of the server. See Boot Identity Files.

   In production domains, you are prompted to enter user credentials on the command line when booting the server.

2.12.2 Boot Identity Files

A boot identity file is a text file that contains user credentials for starting and stopping an instance of WebLogic Server. An Administration Server can refer to this file for user credentials instead of prompting you to provide them. Because the credentials are encrypted, using a boot identity file is much more secure than storing unencrypted credentials in a startup or shutdown script. If there is no boot identity file when starting a server, the server instance prompts you to enter a user name and password.

If you start a Managed Server from a script that invokes the `java weblogic.Server` command (or if you invoke the `java weblogic.Server` command directly), a Managed Server can also refer to a boot identity file. If the Managed Server and Administration Server use the same root directory, the Managed Server can refer to the Administration Server's `boot.properties` file. If a Managed Server's `security` directory contains a valid `boot.properties` file, it uses this file during its startup process by default. The `boot.properties` file can be different for each server instance in the domain.

If you use the Node Manager to start a Managed Server, the Node Manager encrypts and saves the credentials with which it started the server in a server-specific `boot.properties` file for use in automatic restarts. This file is located in *`DOMAIN_NAME`*`/servers/`*`SERVER_NAME`*`/data/nodemanager`, where *`DOMAIN_NAME`* is the name of the directory in which you located the domain and *`SERVER_NAME`* is the name of the server. See Node Manager Configuration and Log Files in the *Administering Node Manager for Oracle WebLogic Server*.

The following sections describe working with boot identity files:

- Creating a Boot Identity File for an Administration Server
- Using java weblogic.Server to Create a Boot Identity File for an Administration Server

- Creating Boot Identity Files for Managed Servers
- How a Server Uses a Boot Identity File at Startup
- Removing Boot Identity Files After Startup
- Limitation Regarding User weblogic

2.12.2.1 Creating a Boot Identity File for an Administration Server

If you use the Configuration Wizard to create a domain in development mode, the Configuration Wizard creates an encrypted boot identity file in the `security` directory of the Administration Server's root directory. See Domain Directory Contents in *Understanding Domain Configuration for Oracle WebLogic Server*.

If a boot identity file for an Administration Server does not already exist, and if you want to bypass the prompt for user name and password, create one as follows.

1. Start the Administration Server at least once and provide the user credentials on the command line.

   During the Administration Server's initial startup process, it generates security files that must be in place before a server can use a boot identity file.

2. Place the following two lines in a text file:

   ```
   username=username
   password=password
   ```

   The user name and password values must match an existing user account in the Authentication provider for the default security realm and must belong to a role that has permission to start and stop a server. See Users, Groups, And Security Roles in *Securing Resources Using Roles and Policies for Oracle WebLogic Server*.

3. Save the file.

   If you save the file as `boot.properties` and locate it in the `security` directory of the server's root directory, the server automatically uses this file during its subsequent startup cycles. See How a Server Uses a Boot Identity File at Startup.

   The first time you use this file to start a server, the server reads the file and then overwrites it with an encrypted version of the user name and password.

2.12.2.2 Using java weblogic.Server to Create a Boot Identity File for an Administration Server

> **Note:**
>
> Use this technique only if you invoke the `java weblogic.Server` command from the command line. If you use a script to start an Administration Server, Oracle recommends that you do not use the technique described in this section for the following reasons:
>
> - It requires you to store an unencrypted password in the startup script.
> - Each time you run the script, the server boots with the supplied user credentials and then creates a new boot identity file.

Instead of following the steps in the previous section, Creating a Boot Identity File for an Administration Server, you can create a boot identity file by invoking the `weblogic.Server` class directly on the command line and including the following options in the Java command:

```
-Dweblogic.management.username=username
-Dweblogic.management.password=password
-Dweblogic.system.StoreBootIdentity=true
```

These options cause the server instance to boot with the supplied user credentials and then store them in a file named `boot.properties`.

For example, the following command starts an Administration Server named `myAdminServer` and creates a boot identity file:

```
java -Dweblogic.management.username=weblogic
-Dweblogic.management.password=password
-Dweblogic.system.StoreBootIdentity=true
-Dweblogic.Name=myAdminServer weblogic.Server
```

For more information about invoking the `weblogic.Server` class directly from a command line, see weblogic.Server Command-Line Reference in *Command Reference for Oracle WebLogic Server*.

2.12.2.3 Creating Boot Identity Files for Managed Servers

If a Managed Server uses the same root directory as the Administration Server, it can use the same boot properties file as the Administration Server. If you use a Node Manager to start a Managed Server, you do not need to create a boot identity file. See Node Manager Configuration and Log Files in *Administering Node Manager for Oracle WebLogic Server*.

To create a boot identity file for a Managed Server instance:

1. Start the domain's Administration Server to make sure that the required security files are in the `security` directory of the Administration Server's domain and root directories. If the files are not present, the Administration Server generates them.

See Domain Configuration Files in *Understanding Domain Configuration for Oracle WebLogic Server*.

2. Place the following two lines in a text file:

```
username=username
password=password
```

The user name and password values must match an existing user account in the Authentication provider for the default security realm and must belong to a role that has permission to start a server. See Users, Groups, And Security Roles in *Securing Resources Using Roles and Policies for Oracle WebLogic Server*.

3. Save the file.

If you save the file as `boot.properties` and locate it in the `security` directory of the server's root directory, the server automatically uses this file during its subsequent startup cycles. See How a Server Uses a Boot Identity File at Startup.

4. Repeat steps 2 and 3 for each Managed Server in the domain for which you want to create a boot identity file.

The first time you use this file to start a server, the server reads the file and then overwrites it with an encrypted version of the user name and password.

2.12.2.4 How a Server Uses a Boot Identity File at Startup

A server instance uses a boot identity file during its startup process as follows:

- If a server's `security` directory contains a valid `boot.properties` file, it uses this file during its startup process by default. See A Server's Root Directory in *Understanding Domain Configuration for Oracle WebLogic Server*.
- If you want to specify a different file (or if you do not want to store boot identity files in a server's `security` directory), you can include the following argument in the server's `weblogic.Server` startup command:

```
-Dweblogic.system.BootIdentityFile=filename
```

where *`filename`* is the fully qualified pathname of a valid boot identity file.

To specify this argument in the `startWebLogic` script, add `-Dweblogic.system.BootIdentityFile` as a value of the `JAVA_OPTIONS` variable. For example:

```
set JAVA_OPTIONS=-
Dweblogic.system.BootIdentityFile=C:\Oracle\user_domains\mydomain\myidentity.
prop
```

- If you do *not* want a server instance to use a boot identity file during its startup cycle, include the following options in the server's `weblogic.Server` startup command:

```
-Dweblogic.management.username=username
-Dweblogic.management.password=password
```

These options cause a server instance to ignore any boot identity files and override other startup options that cause a server to use boot identity files during it startup cycle.

> **Note:**
>
> If you use a script to start a server instance, Oracle recommends that you do not use this technique because it requires you to store an unencrypted password in the startup script. Use this technique only if you invoke the `weblogic.Server` class directly from the command line. See weblogic.Server Command-Line Reference in *Command Reference for Oracle WebLogic Server*.

- If a server is unable to access its boot identity file during its startup cycle, it displays the user name and password prompt in its command shell and writes a message to the log file.

For a given server instance, use only the boot identity file that the instance has created. WebLogic Server does not support copying a boot identity file from one server root directory to another.

For example, if you use `ServerA` to generate a boot identity file, use only that boot identity file with `ServerA`. Do not copy `ServerA`'s boot identity file into the `security` directory of `ServerB`. Instead, create a boot identity file for `ServerB` as described in Creating a Boot Identity File for an Administration Server or Creating Boot Identity Files for Managed Servers.

2.12.2.5 Removing Boot Identity Files After Startup

If you want to remove the boot identity file after a server starts, you can include the following argument in the server's `weblogic.Server` startup command:

```
-Dweblogic.system.RemoveBootIdentity=true
```

This argument removes only the file that the server used to start. For example, if you specify `-Dweblogic.system.BootIdentityFile=c:\secure\boot.MyServer`, only `boot.MyServer` is removed, even if the server's root directory contains a file named `boot.properties`. Open a separate command shell and include the `-Dweblogic.system.RemoveBootIdentity=true` argument in each Managed Server's `weblogic.Server` startup command to remove its boot identity file.

To specify this argument in the `startWebLogic` script, add `-Dweblogic.system.RemoveBootIdentity=true` as a value of the `JAVA_OPTIONS` variable. For example:

```
set JAVA_OPTIONS=-Dweblogic.system.RemoveBootIdentity=true
```

2.12.2.6 Limitation Regarding User weblogic

If you install the WebLogic Server Examples component, the default user `weblogic` is created that has permission to start and stop WebLogic Server. When you set the password of the `weblogic` user, WebLogic Server does not automatically update this password in the `boot.properties` file, which is located in the *`DOMAIN_NAME`*`/servers/AdminServer/security` directory.

If you change the password for user `weblogic`, you can use either of the following workarounds so that you can continue to boot a WebLogic Server instance using that user name and its new password:

- Remove the `boot.properties` file. Subsequently each time you start WebLogic Server, you are prompted for the user name and password. The changed password for the `weblogic` user will be accepted.
- Modify the existing `boot.properties` file, changing the user name and password as follows:

  ```
  username=weblogic
  password=password
  ```

  Subsequently during the server startup process, the `boot.properties` file is encrypted again.

> **Note:**
>
> Do not install the WebLogic Server Examples on a production machine. Keeping the samples software and other development tools off the production machine reduces the leverage intruders potentially have if they were to get partial access to a WebLogic Server production machine.

2.12.3 Specifying User Credentials for Starting a Server with Node Manager

If you use the Node Manager to start a Managed Server, you must provide user credentials on the server's **Configuration** > **Server Start** page of the WebLogic Server Administration Console. If you do not provide these credentials, the Node Manager throws an exception when it tries to start the server.

When you use the WebLogic Server Administration Console or the Configuration Wizard to create a Managed Server, WebLogic Server adds the user credentials to the server's **Configuration** > **Server Start** page. If you want the server instance to run under a different WebLogic Server user account, see Configure startup arguments for Managed Servers in the *Oracle WebLogic Server Administration Console Online Help*.

2.12.4 Changing the Credentials Used for Starting a Server

To change the user credentials that you provided to permit a user to start and stop WebLogic Server instances for the domain:

1. If using the WebLogic Authentication provider (also called `DefaultAuthenticator`), then change the user password using the Administration Console or WLST.

   If you are not using the WebLogic Authentication provider, then use the appropriate tool for the external authentication store to change the user password.
2. If using the `boot.properties` files, then edit the file to set the new password, as described in Boot Identity Files.
3. If using Node Manager to start server instances, and the user name and password are set in the server start element (`ServerStartMBean`), then update the server start password using the Administration Console or WLST.

If using Node Manager to start server instances, and the user name and password are not set in the server start element, then restart the Administration Server before attempting to start Managed Servers.

2.13 Other Startup Tasks

Consider the miscellaneous WebLogic Server startup tasks.

- Making Java Classfiles Globally Available
- Configuring Managed Server Connections to the Administration Server
- Specifying Java Options for a WebLogic Server Instance
- Changing the JVM That Runs Servers
- Configuring Server Level Startup and Shutdown Classes
- Customizing Domain Wide Server Parameters

2.13.1 Making Java Classfiles Globally Available

Use the `$DOMAIN_DIR/lib` environment variable or the `-Dweblogic.ext.dirs` start-up option to make Java class files globally available to all servers in the domain.

There are two methods for making java classes globally available to WebLogic Server:

- Setting the `$DOMAIN_DIR/lib` environment variable.
- Specifying the `-Dweblogic.ext.dirs` startup option.

You can specify either or both of these methods. When specifying both, classes defined using the startup option take precedence.

In both cases, you must ensure that your classes are packaged into `.jar` files.

2.13.2 Configuring Managed Server Connections to the Administration Server

To configure Managed Server connections, use the `protocol://]Admin-host:port` format to specify the listen address of the Administration Server.

If you will be starting a Managed Server from a script that invokes the `java weblogic.Server` command, or if you invoke the `java weblogic.Server` command directly, you must make sure that the Managed Server specifies the correct listen address of the Administration Server. A Managed Server uses this address to retrieve its configuration from the Administration Server.

Use the following format to specify the listen address:

`protocol://]Admin-host:port`

1. For protocol, specify any of the following:
   - `t3`
   - `t3s`
   - `http`

- `https`

  If you will be using the domain-wide administration port, you must specify either `T3S` or `HTTPS`. If you do not specify a value, the servers use `T3`.

  > **Note:**
  >
  > Regardless of which protocol you use, the initial download of a Managed Server's configuration is over `HTTP` or `HTTPS`. After the RMI subsystem initializes, the server instance can use the `T3` or `T3S` protocol.

2. For *`Admin-host`*, specify any of the following:

   - `localhost`

     Valid only if you are starting the Managed Server on the same computer as the Administration Server.

   - The DNS name of the computer that is hosting the Administration Server.

     Configuring a DNS name for the Administration Server that maps to multiple IP addresses is particularly useful for Managed Servers to trying to reconnect after an Administration Server is restarted on a different IP address. See Managed Servers and the Re-started Administration Server.

     If you are using the demo certificates in a multi-server domain, Managed Server instances will fail to boot if you specify the fully-qualified DNS name. For information about this limitation and suggested workarounds, see Limitation on CertGen Usage in *Administering Security for Oracle WebLogic Server*.

   - The IP address of the computer that is hosting the Administration Server.

     Because of the following security issue, Oracle recommends that you do not use IP addresses for *`Admin-host`* in a production environment:

     To connect to the Administration Server through an SSL port, the Managed Server verifies that the Administration Server's host name matches the host name that is specified in the URL. If you specify an IP address, and if host name verification is enabled, the connection fails because the IP address, which is a series of numbers, does not match the name of the host, which is a string of characters.

     In a development environment, where security is less of a concern, you can disable host name verification on the Managed Server so SSL connections that specify an IP address will succeed. See Using Host Name Verification in *Administering Security for Oracle WebLogic Server*.

     If the Administration Server has been configured to use some other listen address, you must specify the configured listen address.

3. For *`port`*, specify any of the following:

   - The domain-wide administration port.

     When configured, the administration port is used by each Managed Server in the domain exclusively for communication with the domain's Administration Server. See Configure the domain-wide administration port in the *Oracle WebLogic Server Administration Console Online Help*.

If you have enabled the domain-wide administration port, you must specify this port. You must specify either the T3S or HTTPS protocol to use this port.

- The non-SSL listen port for the Administration Server's default network configuration (`7001` by default).

  If this listen port has been disabled for the Administration Server, you must use one of the other listen ports described in this list. You must specify either the T3 or HTTP protocol to use this port.

- The SSL listen port for the Administration Server's default network configuration (`7002` by default).

  If this listen port has been disabled for the Administration Server, you must use one of the other listen ports described in this list. You must specify either the T3S or HTTPS protocol to use this port.

- The port number that is associated with an optional, custom network channel.

  If the port is secured with SSL, you must specify either the T3S or HTTPS protocol.

4. To verify the host IP address, name, and default listen port of the Administration Server, start the Administration Server in a shell (command prompt). When the server successfully finishes its startup cycle, it prints to standard out messages that are similar to the following (among other messages):

```
<Nov 5, 2004 12:16:04 PM EST> <Notice> <Server> <BEA-002613> <Channel
"DefaultSecure[2]" is now
listening on 127.0.0.1:7012 for protocols iiops, t3s, ldaps, https.>
...
<Nov 5, 2004 12:16:04 PM EST> <Notice> <WebLogicServer> <BEA-000331>
<Started WebLogic Admin Server
"MedRecServer" for domain "medrec" running in Development Mode>
```

For information on enabling SSL, see Set up SSL in the *Oracle WebLogic Server Administration Console Online Help*. See Understanding Network Channels in *Administering Server Environments for Oracle WebLogic Server*.

2.13.3 Specifying Java Options for a WebLogic Server Instance

Use Java options to configure operating parameters for the JVM that runs a WebLogic Server instance and to override a server configuration temporarily.

The Java options apply only to the current instance of the server. They are not saved in the domain's `config.xml` file and they are not visible from the WebLogic Server Administration Console. For example, if a server is configured to listen on port 7201, you can use a Java option to start the server so that it listens on port 7555. The WebLogic Server Administration Console will still indicate that the server is configured to listen on port 7201. If you do not use the Java option the next time you start the server, it will listen on port 7201.

If you use a WebLogic Server script to start servers, do the following. If you use the Node Manager to start servers, see Set Java options for servers started by Node Manager in the *Oracle WebLogic Server Administration Console Online Help*.

1. Create a backup copy of the WebLogic Server start scripts:

   - For scripts that start an Administration Server, back up *`DOMAIN_NAME`*`\bin\startWebLogic.cmd` (`startWebLogic.sh` on UNIX)

- For scripts that start a Managed Server, back up *DOMAIN_NAME*\bin\startManagedWebLogic.cmd (startManagedWebLogic.sh on UNIX)

  where *DOMAIN_NAME* is the name of the directory in which you located the domain. By default, this directory is *ORACLE_HOME*\user_projects\domains*DOMAIN_NAME*.

2. Open the start script in a text editor.
3. Edit the set JAVA_OPTIONS command to specify the Java options. If you specify multiple options, separate each option by a space, and place quotes around the entire set of options. For example:

   ```
   set JAVA_OPTIONS="-Xgc:gencopy -Xns:30"
   ```

   See:

   - weblogic.Server Command-Line Reference for information on the Java options that set run-time behavior of a WebLogic Server instance.
   - Using Start Scripts in *Administering Node Manager for Oracle WebLogic Server* for detailed information on how JAVA_OPTIONS are combined and how duplicate values are handled.
   - The documentation that the JVM vendor provides for information on the Java options that other JVMs support.
4. Save the start script.
5. Start the server.

2.13.4 Changing the JVM That Runs Servers

To use a different JDK in WebLogic Server, modify the values for the JAVA_HOME and JAVA_VENDOR variables in the Configuration Wizard.

When you create a domain, the Configuration Wizard lists the JDK used when you installed WebLogic Server. The default JDK is Oracle HotSpot, but you may have installed and used another JDK during installation.

After you create a domain, if you want to use a different JVM, you can modify the scripts as follows:

1. Change the value for the JAVA_HOME variable.

   Specify an absolute pathname to the top directory of the JDK that you want to use.

   On a Windows or Linux platform, Oracle recommends Oracle HotSpot.
2. Change the value for the JAVA_VENDOR variable.

   Specify the vendor of the JDK. Valid values depend on the platform on which you are running. See Oracle Fusion Middleware Supported System Configurations on Oracle Technology Network.

   For example:

   - Oracle or Sun indicates that you are using the Oracle HotSpot JDK. It is valid only on platforms that support Oracle HotSpot.

- `HP` and `IBM` indicate that you are using JDKs that Hewlett Packard or IBM have provided. These values are valid only on platforms that support HP or IBM JDKs.

3. Restart any servers that are currently running.

2.13.5 Configuring Server Level Startup and Shutdown Classes

To provide custom, system-wide services for your applications, add the Startup and Shutdown (SU/SD) Java programs to the WebLogic Server classpath and configure them to load and run when a server starts or shuts down.

You must deploy each class on one or more specific servers. By default, startup classes are loaded and run after all other server subsystems have initialized and after the server deploys modules. See Ordering Startup Class Execution and Deployment in *Deploying Applications to Oracle WebLogic Server*. Shutdown classes are loaded and run when you gracefully shut down a server.

WebLogic Server 9.0 introduced a new and simpler POJO-based approach for SU/SD classes. This approach requires only that SU/SD classes have a `static main(String args[])` method, which the server invokes after instantiating the class. Any arguments that you configure for SU/SD classes are passed via the `String args []` parameter.

The POJO-based SU/SD classes follow the same deployment and configuration steps used in earlier versions-the classes needs to be made available on the server classpath, SU/SD classes are configured using the WebLogic Server Administration Console, and ultimately stored as an entry in `config.xml`.

See Example 2-1, Example 2-2, and Use custom classes to configure servers in the *Oracle WebLogic Server Administration Console Online Help*.

Example 2-1 Startup Classes

```
package sab.demo.utils;
public class StartupMain {
/**
* @param args
*/
public static void main(String[] args) {

log(StartupMain.class.getName() + "::main");
log("Arguments::");
for(int i=0;args!=null && i<args.length;i++) {
log("  arg[" + i + "]: " + args[i]);
}
}

private static void log(String msg) {
System.out.printf("  --> [SAB]: %s\n", msg);
}
}
```

Example 2-2 config.xml

```
<startup-class>
<name>StartupMain</name>
<target>AdminServer</target>
<deployment-order>1000</deployment-order>
<class-name>sab.demo.utils.StartupMain</class-name>
<arguments>arg1 arg2 arg3</arguments>
```

```
<failure-is-fatal>false</failure-is-fatal>
<load-before-app-deployments>false</load-before-app-deployments>
<load-before-app-activation>true</load-before-app-activation>
</startup-class>
```

2.13.6 Customizing Domain Wide Server Parameters

To customize domain wide server parameters in WebLogic Server, configure `setUserOverrides.cmd` (Windows) or `setUserOverrides.sh` (UNIX) libraries by specifying the required Java command line options and environment variables to it.

Every domain includes dynamically generated domain and server startup scripts, such as `setDomainEnv`. Oracle recommends that you do not modify these startup scripts, as any changes you make to them will be overwritten during subsequent domain upgrade operations.

To enable you to customize server startup parameters that apply to the servers in a domain, there are two scripts that you can provide: `setUserOverrides.cmd` and `setUserOverridesLate.cmd` (Windows) or `setUserOverrides.sh` and `setUserOverridesLate.sh` (UNIX). The `setUserOverrides` script will be executed early in the `setDomainEnv` script, before any domain extension template properties have been set. The `setUserOverridesLate` script will be executed later after the domain extension template properties have been set. These scripts can be configured to, for example, add custom libraries to the WebLogic Server classpath, specify additional java command line options for running the servers, or specify additional environment variables. Any customizations you add to these files are preserved during domain upgrade operations and are carried over to remote servers when using the pack and unpack commands.

To assure your customizations effectively override all product defaults, add your customizations to the `setUserOverridesLate` script, otherwise your customizations may be overridden by product extensions when using `setUserOverrides`.

> **Note:**
>
> If you move customizations from `setUserOverrides` to `setUserOverridesLate`, it is recommended to re-validate your application's functionality, as some customizations may not have taken effect previously.

During server startup, if this file exists, it is included in the startup sequence and any overrides it defines take effect. The file must be stored in the *`domain_home`*`/bin` directory.

> **Note:**
>
> Node Manager does not detect `setUserOverrides`. If you start Administration Server using Node Manager, `setUserOverrides` does not create startup Java logs and takes the default path even when you mention the arguments `-Dweblogic.Stdout` and `-Dweblogic.Stderr` in Server start tab. You must follow Configuring Remote Startup Arguments in *Administering Node Manager for Oracle WebLogic Server*.

The following example shows how to add custom libraries and specify additional java command line options.

Example 2-3 Example Startup Customizations in a setUserOverridesLate or setEnvUserOverrides file

```
echo ""
echo "*****************************************************"
echo "** Executing setUserOverridesLate.sh"
echo "*****************************************************"

# add custom libraries to the WebLogic Server system classpath
  if [ "${POST_CLASSPATH}" != "" ] ; then
    POST_CLASSPATH="${POST_CLASSPATH}${CLASSPATHSEP}${HOME}/foo/fooBar.jar"
    export POST_CLASSPATH
  else
    POST_CLASSPATH="${HOME}/foo/fooBar.jar"
    export POST_CLASSPATH
  fi

# specify additional java command line options for all servers

EXTRA_JAVA_PROPERTIES="${EXTRA_JAVA_PROPERTIES} -
Dcustom.property.key=custom.value"
export EXTRA_JAVA_PROPERTIES


# Specify any server-specific java command line options by server name or
partial match
case "${SERVER_NAME} in
   AdminServer)
      echo "** AdminServer Customizations:"
      USER_MEM_ARGS="${USER_MEM_ARGS} -Xms#g -Xmx#g ";
      export USER_MEM_ARGS;
      ;;

   WLS_appA*)
      echo "** Application A Managed Servers Customizations:"
      USER_MEM_ARGS="${USER_MEM_ARGS} -Xms#g -Xmx#g ";
      export USER_MEM_ARGS;
      EXTRA_JAVA_PROPERTIES="${EXTRA_JAVA_PROPERTIES} -
Dcustom.property.key=custom.value"
      export EXTRA_JAVA_PROPERTIES
      ;;
   *)
      echo "** WARNING - No server match - VERIFY case condition coding."
      ;;

esac
echo "USER_MEM_ARGS=\"${USER_MEM_ARGS}\""
echo "EXTRA_JAVA_PROPERTIES=\"${EXTRA_JAVA_PROPERTIES}\""
echo ""
echo "*****************************************************"
echo "** End of setUserOverrideLate.sh"
echo "*****************************************************"
echo ""
```

2.14 Shutting Down Instances of WebLogic Server

Shut down a WebLogic Server instance using a stop script in the console or an operating system command to kill the Java Virtual Machine (JVM) of the specific server instance. These procedures help to stop the server instance smoothly without any session data loss.

It is recommended that you shut down WebLogic Server instances through the WebLogic Server Administration Console. See Shut Down a Server Instance, Control Graceful Shutdowns, and Shutdown servers in a cluster in the *Oracle WebLogic Server Administration Console Online Help*.

2.14.1 Shutting Down Servers with a Stop Script

If you use a Configuration Wizard template that is provided by WebLogic Server, the bin directory under your domain directory includes a stop script named `stopWebLogic` that you can use to stop an Administration Server and one named `stopManagedWebLogic` for stopping Managed Servers. To use the scripts, you must set `SERVER_NAME`, `ADMIN_URL`, `USERID`, and `PASSWORD` as environment variables or specify them on the command line. When using the `stopWebLogic` script, if you do not specify `SERVER_NAME`, the Administration Server name is used by default.

- For an Administration Server, invoke:

  `DOMAIN_NAME``\bin\stopWeblogic.cmd` *`username password admin_url`* (Windows)

  `DOMAIN_NAME``/bin/stopWeblogic.sh` *`username password admin_url`* (UNIX)

- For Managed Servers, invoke:

  `DOMAIN_NAME``\bin\stopManagedWeblogic.cmd` *`managed_server_name admin_url username password`* (Windows)

  `DOMAIN_NAME``/bin/stopManagedWeblogic.sh` *`managed_server_name admin_url username password`* (UNIX)

> **Note:**
>
> On the command line, specify parameters in the order shown. User credentials come before the `ADMIN_URL` with `stopWebLogic.cmd` and after the `ADMIN_URL` with `stopManagedWebLogic.cmd`.

2.14.2 Killing the JVM

Each WebLogic Server instance runs in its own JVM. If you are unable to shut down a server instance using the methods described in the previous sections, you can use an operating system command to kill the JVM.

> **Note:**
>
> If you kill the JVM, the server immediately stops all processing. Any session data is lost. If you kill the JVM for an Administration Server while the server is writing to the `config.xml` file, you can corrupt the `config.xml` file.

Some common ways to kill the JVM are as follows:

- If the shell (command prompt) in which you start the server is still open, you can type `Ctrl-C`.
- On a Windows computer, you can use the Task Manager to kill a JVM.
- On a UNIX computer, you can use the `ps` command to list all running processes. Then you can use the `kill` command to kill the JVM.

3

Setting Up a WebLogic Server Instance as a Windows Service

Windows service enables Oracle WebLogic Server to start automatically when you boot a Windows host computer. Using aWindows service, you can stop or restart the server instance smoothly without any data loss and manage multiple user credentials for a single instance. In Windows, the Microsoft Management Console (MMC), specifically Services, is where you start, stop, and configure Windows services. For each server instance that you set up as a Windows service, WebLogic Server creates a key in the Windows Registry under `HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services`. The registry entry contains such information as the name of the server and other startup arguments.

When you start the Windows host, the Microsoft Management Console uses the information in the Windows Registry key to invoke the `weblogic.Server` main class. The MMC cannot be configured to use Node Manager to start Managed Servers, therefore the Node Manager's monitoring and automatic restart features cannot be used for servers that run as a Windows service.

The following tasks set up and manage WebLogic Server instances that run as Windows services:

- Setting Up a Windows Service: Main Steps
- Verifying the Setup
- Using the Services Window to Stop or Restart a Server Instance
- Removing a Server as a Windows Service
- Changing Startup Credentials for a Server Set Up as a Windows Service

3.1 Setting Up a Windows Service: Main Steps

Windows services enable you to create long-running executable applications that run in the background without any user interface. You can set up a Windows service by running a server-specific script that invokes the weblogic.Server main class by using the information in the Windows registry key. Before running the server-specific script, you must be sure to configure or enable the required connections and java classes for the Administration Server and Managed Servers.

To set up a Windows service:

1. Create a script that sets values for server-specific variables and then calls a WebLogic Server master script. See Creating a Server-Specific Script.
2. If you are installing a Managed Server as a Windows service, add a variable to the server specific script that specifies the location of the domain's Administration Server. See Configuring a Connection to the Administration Server.
3. If you set up both an Administration Server and a Managed Server to run as Windows services on the same computer, modify the WebLogic Server

master script so that the Managed Server starts only after the Administration Server finishes its startup cycle. See Requiring Managed Servers to Start After Administration Servers.

4. If you want a server instance to shut down gracefully when you stop the Windows service, create a Java class and modify the master script so that the Microsoft Management Console will invoke the class. See Enabling Graceful Shutdowns.
5. If you want to see the messages that a server instance prints to standard out and standard error (including stack traces and thread dumps), redirect standard out and standard error to a file. See Redirecting Standard Out and Standard Error to a File.
6. If you have created additional Java classes that you want the WebLogic Server instance to invoke, add them to the server's classpath. See Adding Classes to the Classpath.
7. Run the server-specific script. See Run the Server-Specific Script.

3.1.1 Creating a Server-Specific Script

The script that you create must set values for variables that identify the name of the server instance and other server-specific information. Then it must call a master script, *WL_HOME*`\server\bin\installSvc.cmd`, where *WL_HOME* is the directory in which you installed WebLogic Server. The master scripts invokes the `wlsvc` utility, which adds a key to the Windows Registry.

> **Note:**
>
> You can specify a service description when installing a Windows service in the `wlsvc` utility by using the `-svcdescription:<`*user_specified_service_description*`>` option. You can also specify a service description by setting the environment variable `SERVICE_DESCRIPTION` prior to running your `installSvc.cmd` script.
>
> For more information about `wlsvc`, enter the following command at a command prompt: *WL_HOME*`\server\bin\wlsvc -help`, where *WL_HOME* is the directory in which you installed WebLogic Server.

To see an example of a server-specific script, refer to Example 3-1.

To create a server-specific script:

1. In the root directory for the domain's Administration Server create a text file.
2. Add the following, **required** `batch` commands to the text file, each command on a separate line:
   - `SETLOCAL`

     This is a `batch` command that begins the localization of environment variables in a batch file.
   - `set DOMAIN_NAME=`*domain-name*

     where *domain-name* is the name of your WebLogic Server domain.
   - `set USERDOMAIN_HOME=`*absolute-pathname*

where *absolute-pathname* is the absolute pathname of the Administration Server's root directory (the directory that contains the domain's configuration file). See Specifying a Server Root Directory in *Understanding Domain Configuration for Oracle WebLogic Server*.

- `set SERVER_NAME=`*server-name*

  where *server-name* is the name of an existing server instance that you want set up as a Windows service.

- `set WL_HOME=` *wlserver*

  where *wlserver* is the name of the WebLogic installation root directory.

3. Add the following **optional** `batch` commands to the text file. Place each command on a separate line:

   - `set WLS_USER=`*username*
   - `set WLS_PW=`*password*

     where *username* is the name of an existing user with privileges to start a server instance and *password* is the user's password. The `wlsvc` utility encrypts the login credentials and stores them in the Windows registry.

     This is one of two possible methods for avoiding the user name/password prompt when a server instance starts. The disadvantage to this method is that changing the user name or password for the server instance requires you to delete the Windows service and set up a new one with the new user name and password. Instead of this method, you can use a boot identity file. With a boot identity file, you can change the login credentials without needing to modify the Windows service. See Boot Identity Files.

   - `set PRODUCTION_MODE=[true]`

     When the `PRODUCTION_MODE` variable is set to `true`, the server instance starts in production mode. When not specified, or when set to `false`, the server starts in development mode. See Configure Server Start Mode in *Creating WebLogic Domains Using the Configuration Wizard*.

   - `set JAVA_OPTIONS=`*java-options*

     where *java-options* is one or more Java arguments that you want to pass to the Java Virtual Machine (JVM). Separate multiple arguments with a space. For a list of Java options that are specific to WebLogic Server, refer to weblogic.Server Command-Line Reference in the *Command Reference for Oracle WebLogic Server*. The JVM that you use supports additional options and are documented by the JVM vendor.

   - `set JAVA_VM=-`*JVM-mode*

     where *JVM-mode* is a text string that indicates the mode in which you want the JVM to run. The values that you supply depend on the JVM that you are using.

   - `set MEM_ARGS=[-`*XmsNumberm*`] [-`*XmxNumberm*`]`

     where *Numberm* is a numerical value in megabytes (MB). The *-XmsNumberm* argument establishes a minimum heap size for the JVM and the *-XmxNumberm* sets a maximum heap size. By default, the minimum heap size is 23 MB and the maximum heap size is 200 MB.

> **Note:**
>
> To specify a non-default JVM heap size, set the *MEM_ARGS* values in *WL_HOME*\common\bin\commEnv.cmd; however, this change affects all the domains under the same *WL_HOME*.

- `set MAX_CONNECT_RETRIES=`*number_of_attempts*
- `set HOST=`*ip_address*
- `set PORT=`*port_number*

  where *number_of_attempts* is the number of times that the Windows service will perform a status check to determine if a WebLogic Server instance is started. If you specify this variable along with `HOST` and `PORT`, the Windows service will wait until the WebLogic Server instance is started. *ip_address* is the IP address of the WebLogic Server instance and `port_number` is the port on which the WebLogic Server instance is listening for requests.

4. Add the following **required** commands to the end of the script:
   - `call "`*WL_HOME*`\user_projects\domains\base_domain\bin\setDomainEnv.cmd"`

     where *WL_HOME* is an absolute pathname for the directory in which you installed WebLogic Server. This command sets the domain-level environment variables.
   - `call "`*WL_HOME*`\server\bin\installSvc.cmd"`

     where *WL_HOME* is an absolute pathname for the directory in which you installed WebLogic Server. This command calls the WebLogic Server master script.
   - `ENDLOCAL`

     This is a `batch` command that ends the localization of environment variables in a batch file.
5. Save the text file with a `.cmd` extension. By default, the Windows command prompt associates the `.cmd` extension with `batch` files.

3.1.2 Configuring a Connection to the Administration Server

If you want to install a Managed Server as a Windows service, you must include a variable that specifies the location of the domain's Administration Server. The Managed Server must contact the Administration Server to retrieve its configuration data.

The Administration Server (which is not a service) must be started before installing and starting Managed Server as a Windows service.

To configure a connection to the Administration Server:

1. In a text editor, open the server-specific script.
2. In the text file, between the `SETLOCAL` command and the `call` command, create the following command:

   `set ADMIN_URL=`*protocol*`://`*listen-address*`:`*listen-port*

   where:

- *protocol* is `http` or `https`
- *listen-address* is a listen address of the Administration Server
- *listen-port* is a port of the Administration Server

  See Configuring Managed Server Connections to the Administration Server.

  For an example, refer to Example 3-1.

3. Save your modifications to the server-specific script.

Example 3-1 Example Script for Setting Up a Managed Server as a Windows Service

```
echo off
SETLOCAL
set MW_HOME=C:\Oracle\Middleware\Oracle_Home
set DOMAIN_NAME=base_domain
set USERDOMAIN_HOME=C:\Oracle\Middleware\Oracle_Home\user_projects\domains\base_domain
set SERVER_NAME=ManagedServer
set ADMIN_URL=http://10.xxx.xx.xx:7001
set WL_HOME=C:\Oracle\Middleware\Oracle_Home\wlserver
set PRODUCTION_MODE=true
set MEM_ARGS=-Xms128m -Xmx512m
call "C:\Oracle\Middleware\Oracle_Home\user_projects\domains\base_domain\bin\setDomainEnv.cmd"
call "C:\Oracle\Middleware\Oracle_Home\wlserver\server\bin\installSvc.cmd"
ENDLOCAL
```

Note:

To use this example script to set up an Administration Server as a Windows service, remove the `ADMIN_URL` variable.

If you copy and paste examples, make sure that there are no trailing spaces in them.

`WLS_USER` and `WLS_PW` are optional and can be added to the script only if boot identity file is not being used. See Creating a Server-Specific Script.

3.1.3 Requiring Managed Servers to Start After Administration Servers

If you set up both an Administration Server and a Managed Server to run as Windows services on the same computer, you can specify that the Managed Server starts only after the Administration Server.

To require a Managed Server to start after the Administration Server Windows service:

1. Create a backup copy of the *WL_HOME*`\server\bin\installSvc.cmd` master script.
2. If you have already installed the Administration Server as a Windows service, remove the service. See Removing a Server as a Windows Service.
3. Before you install (or reinstall) the Administration Server as a Windows service, do the following:
   a. In a text editor, open the *WL_HOME*`\server\bin\installSvc.cmd` master script.

      The last command in this script invokes `wlsvc`, which is the WebLogic Server utility that modifies the Windows Registry.

b. In `installSvc.cmd`, add the following argument to the command that invokes the `wlsvc` utility:

   `-delay:`*`delay_milliseconds`*

   This specifies the number of milliseconds to wait before the Microsoft Management Console changes the service status from `SERVER_START_PENDING` to `STARTED`.

   For example, if your Administration Server requires 2 minutes to complete its startup cycle and begin listening for requests, then specify `-delay=120000`. When you boot the Windows host computer, the Microsoft Management Console reports a status of `SERVER_START_PENDING` for 2 minutes. Then it changes the status to `STARTED`.

   The modified `wlsvc` invocation for the Administration Server will resemble the following:

```
"%WL_HOME%\server\bin\wlsvc" -install
-svcname:"%DOMAIN_NAME%_%SERVER_NAME%"
-delay:120000
-javahome:"%JAVA_HOME%" -execdir:"%USERDOMAIN_HOME%"
-extrapath:"%WL_HOME%\server\bin" -password:"%WLS_PW%"
-cmdline:%CMDLINE%
```

   For more information about `wlsvc`, enter the following command at a command prompt: *`WL_HOME`*`\server\bin\wlsvc -help`, where *`WL_HOME`* is the directory in which you installed WebLogic Server.

4. Install the Administration Server Windows service.
5. Before you install the Managed Server as a Windows service, do the following:
   a. In a text editor, open the *`WL_HOME`*`\server\bin\installSvc.cmd` master script.
   b. In `installSvc.cmd`, add the following argument to the command that invokes the `wlsvc` utility:

      `-depend:`*`Administration-Server-service-name`*

      where *`Administration-Server-service-name`* is the name of the Administration Server Windows service. To verify the service name, look in the Microsoft Management Console under Services.

      With this option, the Microsoft Management Console will wait for the Administration Server Windows service to report a status of `STARTED` before it starts the Managed Server Windows service.

      For example, the modified `wlsvc` invocation for the Managed Server will resemble the following:

```
"%WL_HOME%\server\bin\wlsvc" -install
-svcname:"%DOMAIN_NAME%_%SERVER_NAME%"
-depend:"wlsvc myDomain_myAdminServer"
-javahome:"%JAVA_HOME%" -execdir:"%USERDOMAIN_HOME%"
-extrapath:"%WL_HOME%\server\bin" -password:"%WLS_PW%"
-cmdline:%CMDLINE%
```

      You can also add the `-delay:delay_milliseconds` option to a Managed Server Windows service if you want to configure when the Microsoft Management Console reports a status of `STARTED` for the service.

3.1.4 Enabling Graceful Shutdowns

By default, if you use the Microsoft Management Console to stop a server instance, it kills the server's Java Virtual Machine (JVM). If you kill the JVM, the server immediately stops all processing. Any session data is lost. If you kill the JVM for an Administration Server while the server is writing to the `config.xml` file, you can corrupt the `config.xml` file.

To enable graceful shutdowns from Microsoft Management Console:

1. Create a Java class that invokes the `weblogic.management.runtime.ServerRuntime.shutdown()` method.

   This method gracefully shuts down a server after the server has completed all in-flight work. For an example of such a class, refer to Java Class that Shuts Down a Server Instance.

2. Create a backup copy of the *`WL_HOME`*`\server\bin\installSvc.cmd` master script.

3. In a text editor, open the *`WL_HOME`*`\server\bin\installSvc.cmd` master script and do the following:

   a. Add the class that you created to the `set CLASSPATH` statement.

      For example if you archived your class in a file named `c:\myJar`, the modified statement will be as follows:

      ```
      set
      CLASSPATH=%JAVA_HOME%\lib\tools.jar;%WL_HOME%\server\lib\weblogic_sp
      .jar;%WL_HOME%\server\lib\weblogic.jar;c:\myJar;%CLASSPATH%
      ```

   b. Add the following argument to the last line of the script, which calls the `wlsvc` utility:

      ```
      -stopclass:javaclass
      ```

      where *`javaclass`* is the full classpath name of the class that you created. This argument loads `javaclass` and then invokes its `public void static stop()` method.

      For example, if you packaged the class in Example 3-2 in `com.myClasses`, the modified `wlsvc` command will be as follows:

      ```
      "%WL_HOME%\server\bin\wlsvc" -install
      -svcname:"%DOMAIN_NAME%_%SERVER_NAME%"
      –stopclass:com.myClasses.ServerStopper
      -javahome:"%JAVA_HOME%" -execdir:"%USERDOMAIN_HOME%"
      -extrapath:"%WL_HOME%\server\bin" -password:"%WLS_PW%"
      -cmdline:%CMDLINE%
      ```

      For more information about `wlsvc`, enter the following command at a command prompt: *`WL_HOME`*`\server\bin\wlsvc -help`, where *`WL_HOME`* is the directory in which you installed WebLogic Server.

4. In the WebLogic Server Administration Console, on the server's **Control** page, configure the Managed Server's graceful shutdown behavior.

   You can determine whether a graceful shutdown operation drops all HTTP sessions immediately and you can configure the amount of time that a graceful shutdown operation waits before forcing a shut down. See Control graceful shutdowns in the *Oracle WebLogic Server Administration Console Online Help*.

5. Consider modifying the default timeout value that the Windows service specifies.

   By default, when you use the Microsoft Management Console to stop a Windows service, it waits 30 seconds for the service to stop before it kills the service and prints a timeout message to the System event log.

   If you use `-stopclass` to gracefully shut down a server, 30 seconds might not be enough time for the server to gracefully end its processing.

   To configure a timeout period, create a `REG_DWORD` registry value named `ServicesPipeTimeout` under the following registry key:

   `HKEY_LOCAL_MACHINE\System\CurrentControlSet\Control`

   The key value must be in milliseconds.

   This value is read from the registry during the startup of the Windows operating system and it affects all services that are installed.

6. Save your changes to the WebLogic Server master script.

3.1.4.1 Java Class that Shuts Down a Server Instance

The following Java class uses Java Management Extensions (JMX) to shut down a server instance. Each server uses JMX Managed Beans (MBeans) to expose its management attributes and operations. One such MBean, `ServerRuntime`, exposes a `shutdown()` method that gracefully shuts down a server.

The class in Example 3-2 uses the Administration `MBeanHome` interface, which can retrieve and call `ServerRuntime` MBean operations for all server instances in a domain.

See *Developing Custom Management Utilities Using JMX for Oracle WebLogic Server* and `Javadoc`.

Example 3-2 Java Class that Shuts Down a Server Instance

```
package com.myClasses;

import java.net.*;
import java.io.*;
import java.util.Hashtable;
import java.util.logging.Level;
import java.util.logging.Logger;
import javax.management.MBeanServerConnection;
import javax.management.ObjectName;
import javax.management.remote.JMXConnector;
import javax.management.remote.JMXConnectorFactory;
import javax.management.remote.JMXServiceURL;
import javax.naming.Context;

public class ServerStopper {
  public static void stop() {
    try {
      String servername = "AdminServer";
      String hostname = "localhost";
      int port = 7001;
      String username = "weblogic";
      String password = "password";
```

```
      //Select t3 or iiop as required
      //String protocol = "t3";
      String protocol = "iiop";
      String jndiroot = "/jndi/";
      String mserver = "weblogic.management.mbeanservers.domainruntime";

      // To avoid AssertionError
      URL[] urls = { new File("/").toURL() };
      Thread.currentThread().setContextClassLoader(new
         URLClassLoader(urls));

      JMXServiceURL serviceURL = new JMXServiceURL(protocol, hostname,
port, jndiroot + mserver);
      Hashtable h = new Hashtable();
      h.put(Context.SECURITY_PRINCIPAL, username);
      h.put(Context.SECURITY_CREDENTIALS, password);
      h.put(JMXConnectorFactory.PROTOCOL_PROVIDER_PACKAGES,
"weblogic.management.remote");
      JMXConnector connector = JMXConnectorFactory.connect(serviceURL,
h);
      MBeanServerConnection connection =
connector.getMBeanServerConnection();
      ObjectName service = new
ObjectName("com.bea:Name=DomainRuntimeService,Type=weblogic.management.m
beanservers.domainruntime.DomainRuntimeServiceMBean");
      ObjectName[] serverRuntimes = (ObjectName[])
connection.getAttribute(service, "ServerRuntimes");

    for (ObjectName serverRuntime : serverRuntimes) {
      String name = (String) connection.getAttribute(serverRuntime,
"Name");

      if (name.equals(servername)) {
        String state = (String) connection.getAttribute(serverRuntime,
"State");

        if (state.equals("RUNNING") || state.equals("ADMIN")) {
           connection.invoke(serverRuntime, "shutdown", new Object[0],
new String[0]);
           System.out.println("Stopped the server: " + servername);
        }
        else
          System.out.println("The server is not in RUNNING or ADMIN
state.  The current state is: " + state);
        return;
      }
    }
    System.out.println("The server, " + servername + " does not exist");
    connector.close();
  }
  catch (Exception ex) {
    Logger.getLogger(ServerStopper.class.getName()).log(Level.SEVERE,
null, ex);
    }
```

```
  }
}
```

3.1.5 Redirecting Standard Out and Standard Error to a File

By default, when you install a WebLogic Server instance as a Windows service, you cannot see the messages that the server or its JVM print to standard out and standard error.

To view these messages, you must redirect standard out and standard error to a file:

1. Create a backup copy of the *WL_HOME*\server\bin\installSvc.cmd master script.
2. In a text editor, open the *WL_HOME*\server\bin\installSvc.cmd master script.
3. In installSvc.cmd, the last command in the script invokes the wlsvc utility. At the end of the wlsvc command, append the following command option:

   ```
   -log:"pathname"
   ```

   where *pathname* is a fully qualified path and filename of the file that you want to store the server's standard out and standard error messages.

   The modified wlsvc command will resemble the following command:

   ```
   "%WL_HOME%\server\bin\wlsvc" -install
   -svcname:"%DOMAIN_NAME%_%SERVER_NAME%"
   -javahome:"%JAVA_HOME%" -execdir:"%USERDOMAIN_HOME%"
   -extrapath:"%WL_HOME%\server\bin" -password:"%WLS_PW%"
   -cmdline:%CMDLINE%

   -
   log:"d:\Oracle\Middleware\user_projects\domains\myWLSdomain\myWLSserv
   er-stdout.txt"
   ```

4. By default, every 24 hours the Windows service archives messages to a file named *pathname-yyyy_mm_dd-hh_mm_ss*. New messages collect in the file that you specified in the previous step.

   For information on changing the default behavior, see Changing the Default Rotation Criteria.

After you install the service and restart the Windows host, to view the messages that the server writes to standard out or standard error, do one of the following:

- Make a copy of the file that you specified and view the copy. The Windows file system cannot write to files that are currently opened.
- To view the messages as they are being printed to the file, open a command prompt and, using a DOS utility that supports the tail command, enter `tail -f` *stdout-filename*.

3.1.5.1 Changing the Default Rotation Criteria

By default, every 24 hours the Windows service archives messages to a file named *pathname-yyyy_mm_dd-hh_mm_ss*. New messages collect in the file that you specified when you set up the service.

You can change the time interval or you can set up rotation to occur based on the size of the message file instead of a time interval.

To change the default criteria at which the Windows service rotates message files:

1. If the Windows service is running, shut it down.
2. Edit the file you specified in the `-log: pathname` argument. If a file does not exist, create one.

   For example, if you issued the example command in step 3 in the previous section, create a file named `d:\Oracle\Middleware\user_projects\domains\myWLSdomain\myWLSserver-stdout.txt`.
3. Do one of the following:
   - If you want the Windows service to rotate the message file at a specific time interval regardless of file size, add the following statements at the top of the file, each statement on a separate line (make sure to press the Enter or Return key after typing the last line):

     ```
     # ROTATION_TYPE = TIME
     # TIME_START_DATE = date-in-required-format
     # TIME_INTERVAL_MINS = number-of-minutes
     ```

     where `TIME_START_DATE` specifies when the first rotation should take place. If the specified time has already passed, the first rotation occurs when the time interval specified in `TIME_INTERVAL_MINS` expires. You must use the following format to specify the start time: *`Month Day Year Hour:Minutes:Seconds`*

     where `Month` is the first 3 letters of a Gregorian-calendar month as written in English

     `Day` is the 2-digit day of the Gregorian-calendar month

     `Year` is the 4-digit year of the Gregorian calendar

     `Hour:Minutes:Seconds` expresses time in a 24-hour format

     `TIME_INTERVAL_MINS` specifies how frequently (in minutes) the Windows service rotates the file.

     Optionally, you can add the `MAX_FILE_COUNT` attribute to specify a rotation limit for the output files that are generated from WebLogic Server as a service. By default, `MAX_FILE_COUNT=-1`, meaning no limit on the number of files exists. Defining `MAX_FILE_COUNT` with a value greater than `0` sets a limit on the number of files to rotate.

     For example:

     ```
     # ROTATION_TYPE = TIME
     # TIME_START_DATE = Jul 17 2003 05:25:30
     # TIME_INTERVAL_MINS = 1440

     # MAX_FILE_COUNT = 999
     ```

     When the time interval expires, the Windows service saves the file as *`pathname-yyyy_mm_dd-hh_mm_ss`*. It then creates a new file named *`pathname`*. This new file, which contains all of the headers that you specified originally, collects new standard out and standard error messages.

If you specify `# ROTATION_TYPE = TIME` but do not include the other lines, the Windows service rotates the message file every 24 hours.

- If you want the Windows service to rotate the message file after the file grows beyond a specified size, add the following statements at the top of the file, each statement on its own line (make sure to press the Enter or Return key after typing the last line):

  ```
  # ROTATION_TYPE = SIZE
  # SIZE_KB = file-size-in-kilobytes
  # SIZE_TRIGGER_INTERVAL_MINS = polling-interval
  ```

  where `SIZE_KB` specifies the minimal file size (in kilobytes) that triggers the Windows service to move messages to a separate file.

  `SIZE_TRIGGER_INTERVAL_MINS` specifies (in minutes) how frequently the Windows service checks the file size. If you do not include this header, the Windows service checks the file size every 5 minutes.

  For example:

  ```
  # ROTATION_TYPE = SIZE
  # SIZE_KB = 1024
  # SIZE_TRIGGER_INTERVAL_MINS = 3

  # MAX_FILE_COUNT = 999
  ```

  When the Windows service checks the file size, if the file is larger than the size you specify, it saves the file as *`pathname-yyyy_mm_dd-hh_mm_ss`*. It then creates a new file named *`pathname`*. This new file, which contains all of the headers that you specified originally, collects new standard out and standard error messages.

  If you specify `# ROTATION_TYPE = SIZE` but do not include the other lines, the Windows service checks the size of the message file every 5 minutes. If the file is larger than 1 megabytes, it rotates the file.

To cause the WebLogic Server instance to print a thread dump to standard out, do either of the following:

- Use the WLST `threadDump` command.
- Open a command prompt and enter the following command:

  `WL_HOME``\bin\wlsvc -dump -svcname:`*`service-name`*

  where *`WL_HOME`* is the directory in which you installed WebLogic Server and *`service-name`* is the Windows service that is running a server instance.

For example:

```
D:\Oracle\Middleware\wlserver_12.1\server\bin\wlsvc -dump -
svcname:mydomain_myserver
```

3.1.6 Adding Classes to the Classpath

The classpath is a declaration of the location of Java classes that a JVM can invoke. When you use the WebLogic Server master script to install a server instance as a Windows service, the master script specifies all classes required to run a server instance. If you want to extend WebLogic Server by adding your own Java classes, you must add them to the classpath.

To add classes to the classpath:

1. Create a backup copy of the *WL_HOME*\server\bin\installSvc.cmd master script.
2. In a text editor, open the *WL_HOME*\server\bin\installSvc.cmd master script.
3. Add your class to the `set CLASSPATH` statement.

   For example if you archived your class in a file named `c:\myJar`, the modified statement will be as follows:

   ```
   set
   CLASSPATH=%JAVA_HOME%\lib\tools.jar;%WL_HOME%\server\lib\weblogic_sp.
   jar;%WL_HOME%\server\lib\weblogic.jar;c:\myJar;%CLASSPATH%
   ```

   > **Note:**
   >
   > Win32 systems have a 2K limitation on the length of the command line. If the classpath setting for the Windows service startup is very long, the 2K limitation could be exceeded.

   To work around this limitation:

   a. Place the value of the `set CLASSPATH` command in a separate text file.
   b. In the *WL_HOME*\server\bin\installSvc.cmd master script, find the `set CMDLINE` command.
   c. Within the `set CMDLINE` command, replace the `-classpath \"%CLASSPATH%\"` option with the following option:

      ```
      -classpath @pathname\filename
      ```

      where *pathname**filename* is the absolute path and name of the file that contains the classpath values.

      For example:

      ```
      set CMDLINE="%JAVA_VM% %MEM_ARGS% %JAVA_OPTIONS%
      -classpath @c:\myClasspath.txt -Dweblogic.Name=%SERVER_NAME%
      -Dbea.home=\"D:\bea_70sp2\" -Dweblogic.management.username=%WLS_USER%
      -Dweblogic.management.server=\"%ADMIN_URL%\"
      -Dweblogic.ProductionModeEnabled=%STARTMODE%
      -Djava.security.policy=\"%WL_HOME%\server\lib\weblogic.policy\"
      weblogic.Server"
      ```

4. Save your changes to the WebLogic Server master script.

3.1.7 Run the Server-Specific Script

> **Note:**
>
> To run the server-specific script, you must log in to the Windows computer with a user account that has privileges to modify the Windows registry.

If you install the Windows service in a production environment, Oracle recommends that you do *not* run the service under an operating-system user account that has administrator-level privileges. See Verifying the User Account Under Which the Service Runs.

To run the server-specific script:

1. Open a command prompt and change to Administration Server's root directory, which is the directory that contains the server-specific script.
2. Enter the name of the server-specific script.

   The command prompt runs the script as a `batch` file.

   If the script runs successfully, it creates a Windows service named *`DOMAIN_NAME_SERVER_NAME`* and prints a line to standard out that is similar to the following:

   ```
   mydomain_myserver installed.
   ```

   By default, standard out is the command prompt in which you run the server-specific `batch` file.
3. If you modified the *`WL_HOME`*`\server\bin\installSvc.cmd` master script, consider undoing your modifications so the script can be used to set up other server instances.

3.2 Verifying the Setup

Use the `wlsvcx64 -debug` command to verify that the Windows service is configured correctly.

To verify that you successfully set up a WebLogic Server as a Windows service, do the following:

1. Open a command window and enter the following command:

   ```
   set PATH=WL_HOME\server\bin;%PATH%
   ```
2. Navigate to the directory immediately above your domain directory. For example, to verify the setup for *`ORACLE_HOME`*`\user_projects\domains\mydomain`, navigate to *`ORACLE_HOME`*`\user_projects\domains`.
3. Enter:

   ```
   wlsvcx64 -debug "yourServiceName"
   ```

   For example, `wlsvcx64 -debug "mydomain_myserver"`.

   If your setup was successful, the `wlsvcx64 -debug` command starts your server. If the script returns an error similar to the following, make sure that you specified the correct service name:

   ```
   Unable to open Registry Key ....... System\CurrentControlSet\Services\wlsvc
   example_examplesServer\Parameters
   ```

3.2.1 Verifying the User Account Under Which the Service Runs

In a production environment, WebLogic Server Windows services should run under a special operating-system user account that has limited access privileges. For example,

the OS user should have access privileges only to Oracle files and to your domain files. This should be the only user account that has access to these files.

To ensure that the WebLogic Server instance runs under the special OS user account:

1. Open Services in the Microsoft Management Console.

   For example, from the Windows XP desktop:

   a. Select the Start menu.

   b. On the Start menu, select **Programs** > **Administrative Tools** > **Services**.

2. In the Services window, right click the WebLogic Server Windows service and select **Properties**.
3. In the Properties window, select the **Log On** tab.
4. Under Log on, select **This account**. Then enter the user name and password of the special OS user account.
5. Click OK.

> **Note:**
>
> When accessing network drives, the Windows service must run under the same user name as the one who shared the network drive.

3.3 Using the Services Window to Stop or Restart a Server Instance

Use the Microsoft Management Console Services window to stop and restart a server instance.

By default, if you use the Microsoft Management Console to stop a server instance, it kills the server's Java Virtual Machine (JVM). If you kill the JVM, the server immediately stops all processing. Any session data is lost. If you kill the JVM for an Administration Server while the server is writing to the `config.xml` file, you can corrupt the `config.xml` file. See Enabling Graceful Shutdowns.

To stop or restart a WebLogic Server instance that is installed as a Windows service:

1. Select **Start** > **Programs** > **Administrative Tools** > **Services**.
2. In the Services window, find the service that you created. By default, the service name starts with `wlsvc`.
3. Right-click the service name and select commands from the menu.

3.4 Removing a Server as a Windows Service

You can remove a Windows service by using a script that causes the wlsvc utility to remove the associated key from the Windows Registry. It does not affect the server instance's configuration that is saved in the domain's configuration file. After removing

the Windows service, you can restart the WebLogic server instance with start scripts, for Managed Servers or Node Manager.

The script sets values for variables that identify the name of the server instance and other server-specific information. Then the script calls a master uninstall script, *WL_HOME*\server\bin\uninstallSvc.cmd, where *WL_HOME* is the directory in which you installed WebLogic Server. The master scripts invokes the wlsvc utility, which removes a key from the Windows Registry.

To see an example of a server-specific uninstaller script, refer to Example 3-3.

To create a script for removing a Windows service that runs a WebLogic Server instance:

1. In the root directory for the domain's Administration Server (the directory that contains the domain's config.xml file), create a text file.
2. Add the following, **required** batch commands to the text file, each command on a separate line:
   - SETLOCAL

     This is a batch command that begins the localization of environment variables in a batch file.
   - set DOMAIN_NAME=*domain-name*

     where *domain-name* is the name of your WebLogic Server domain.
   - set SERVER_NAME=*server-name*

     where *server-name* is the name of an existing server instance that you want remove as a Windows service.
   - set MW_HOME=*ORACLE_HOME*

     where *ORACLE_HOME* is the absolute path to the directory you specified as Oracle Home when you installed Oracle WebLogic Server.
   - call "*WL_HOME*\server\bin\uninstallSvc.cmd"

     where *WL_HOME* is the absolute path to the directory in which you installed WebLogic Server. This command calls the WebLogic Server master uninstall script.
   - ENDLOCAL

     This is a batch command that ends the localization of environment variables in a batch file.
3. Save the text file with a .cmd extension. By default, the Windows command prompt associates the .cmd extension with batch files.
4. Enter the name of the server-specific script.

   The command prompt runs the script as a batch file.

If the removal script runs successfully, it prints a line similar to the following to standard out:

```
mydomain_myserver removed
```

By default, standard out is the command prompt in which you run the batch file.

Example 3-3 Script to Remove a Windows Service

```
echo off
SETLOCAL
set DOMAIN_NAME=myWLSdomain
set SERVER_NAME=myWLSserver
set MW_HOME=D:\Oracle\Middleware\Oracle_Home
call "D:\Oracle\Middleware\Oracle_Home\wlserver\server\bin\uninstallSvc.cmd"
ENDLOCAL
```

3.5 Changing Startup Credentials for a Server Set Up as a Windows Service

To make a WebLogic Server instance run under different user credentials, you can change a Windows service configuration to support various startup credentials.

Do one of the following:

- If you set up the Windows service to retrieve user names and passwords from a boot identity file, you can overwrite the existing file with a new one that contains the new user name and password. You must specify the name of an existing user in the WebLogic Server default security realm. See Boot Identity Files.
- If you set up the Windows service to retrieve user names and passwords from the Windows registry, then you must remove the Windows service and create a new one that uses your new user name or password:
  1. Uninstall the Windows service that runs the WebLogic Server instance. See Removing a Server as a Windows Service.
  2. In a text editor, open the script that you used to install the service and enter the new user name and password as the value for the `set WLS_USER` and `set WLS_PW` commands. WebLogic Server encrypts these values in the Windows Registry.
  3. Save your modifications to the script.
  4. Enter the name of the server-specific script.

     The command prompt runs the script as a `batch` file.

     If the script runs successfully, it creates a Windows service named

     `DOMAIN_NAME_SERVER_NAME` and prints a line to standard out that is similar to the following:

     ```
     mydomain_myserver installed
     ```

     By default, standard out is the command prompt in which you run the server-specific `batch` file.
  5. (Optional) Remove the user name and password from the script file.

4

Avoiding and Recovering From Server Failure

Server instances may fail periodically even in a clustered environment. Events including loss of power, hardware malfunction, operating system crashes, network partitions, and unexpected application behavior, can lead to the failure of a server instance. For high availability requirements, implement a clustered architecture that helps to minimize the impact of failure events and recover the failed server.

See Failover and Replication in a Cluster in *Administering Clusters for Oracle WebLogic Server*.

4.1 Failure Prevention and Recovery Features

WebLogic server provides several recovery features that protect the servers by avoiding conflicts resulting from unanticipated levels of application and resource utilization. The features include automatic restart, server-level migration, service-level migration, and so on.

4.1.1 Overload Protection

WebLogic Server detects increases in system load that can affect application performance and stability, and allows administrators to configure failure prevention actions that occur automatically at predefined load thresholds.

Overload protection helps you avoid failures that result from unanticipated levels of application traffic or resource utilization.

WebLogic Server attempts to avoid failure when certain conditions occur:

- Workload manager capacity is exceeded
- HTTP session count increases to a predefined threshold value
- Impending out of memory conditions

4.1.2 Failover for Clustered Services

You can increase the reliability and availability of your applications by hosting them on a WebLogic Server cluster. Clusterable services, such as EJBs and Web applications,

can be deployed uniformly—on each Managed Server—in a cluster, so that if the server instance upon which a service is deployed fails, the service can fail over to another server in the cluster, without interruption in service or loss of state.

See Failover and Replication in a Cluster in *Administering Clusters for Oracle WebLogic Server*.

4.1.3 Automatic Restart for Failed Server Instances

WebLogic Server self-health monitoring improves the reliability and availability of server instances in a domain. Selected subsystems within each WebLogic Server instance monitor their health status based on criteria specific to the subsystem. For example, the JMS subsystem monitors the condition of the JMS thread pool while the core server subsystem monitors default and user-defined execute queue statistics. If an individual subsystem determines that it can no longer operate in a consistent and reliable manner, it registers its health state as "failed" with the host server.

Each WebLogic Server instance, in turn, checks the health state of its registered subsystems to determine its overall viability. If one or more of its critical subsystems have reached the `FAILED` state, the server instance marks its own health state `FAILED` to indicate that it cannot reliably host an application.

Using Node Manager, server self-health monitoring enables you to automatically reboot servers that have failed. This improves the overall reliability of a domain, and requires no direct intervention from an administrator.

See Node Manager and System Crash Recovery in the *Administering Node Manager for Oracle WebLogic Server*.

4.1.4 Server-Level Migration

WebLogic Server provides the capability to migrate clustered server instances. A clustered server that is configured to be migratable can be moved in its entirety from one machine to another, at the command of an administrator, or automatically, in the event of failure. The migration process makes all of the services running on the server instance available on a different machine, but not the state information for the singleton services that were running at the time of failure. See Whole Server Migration in *Administering Clusters for Oracle WebLogic Server*.

4.1.5 Service-Level Migration

WebLogic Server supports migration of a individual singleton service as well as the server-level migration capability described in the previous section. Singleton services are services that run in a cluster but must run on only a single instance at any given time, such as JMS and the JTA transaction recovery system.

An administrator can migrate a JMS server or the JTS transaction recovery from one server instance to another in a cluster, either in response to a server failure or as part of regularly-scheduled maintenance. This capability improves the availability of pinned services in a cluster, because those services can be quickly restarted on a redundant server should the host server fail.

See Service Migration in *Administering Clusters for Oracle WebLogic Server*.

4.1.6 Managed Server Independence Mode

Managed Servers maintain a local copy of the domain configuration. When a Managed Server starts, it contacts its Administration Server to retrieve any changes to the domain configuration that were made since the Managed Server was last shut down. If a Managed Server cannot connect to the Administration Server during startup, it can use its locally cached configuration information—this is the configuration that was current at the time of the Managed Server's most recent shutdown. A Managed Server that starts up without contacting its Administration Server to check for configuration updates is running in *Managed Server Independence (MSI)* mode. By default, MSI mode is enabled. See Disable Managed Server independence in *Oracle WebLogic Server Administration Console Online Help*.

4.2 Directory and File Backups for Failure Recovery

Backup creates a copy of existing files, folders, directories, and restore them in case of data loss. WebLogic server performs few backups automatically and also, encourages the administrators to perform some backup procedures. These procedures include domain configuration directory, LDAP repository, and other security certificates associated with server applications.

This section describes file backups that WebLogic Server performs automatically, and recommended backup procedures that an administrator should perform.

Recovery from the failure of a server instance requires access to the domain configuration and security data. The WebLogic Security service stores its configuration data in the `config.xml` file, and also in an LDAP repository and other files.

See Domain Configuration Files in *Understanding Domain Configuration for Oracle WebLogic Server*.

4.2.1 Back Up Domain Configuration Directory

By default, an Administration Server stores the domain configuration data in the *`domain_name`*`\config` directory, where *`domain_name`* is the root directory of the domain.

Back up the `config` directory to a secure location in case a failure of the Administration Server renders the original copy unavailable. If an Administration Server fails, you can copy the backup version to a different machine and restart the Administration Server on the new machine.

Each time a Managed Server starts up, it contacts the Administration Server and if there are changes in to the domain configuration, the Managed Server updates its local copy of the domain `config` directory.

During operation, if changes are made to the domain configuration, the Administration Server notifies the Managed Servers which update their local `/config` directory. So, each Managed Server always has an current copy of its configuration data cached locally.

Do not add non-configuration files in the `config` directory or subdirectories. Non-configuration files include log (.log) and lock (.lck) files. Administration Server replicates the `config` directory in all Managed Server instances. Storing non-configuration files in the `config` directory can cause performance issues in the domain.

4.2.2 Back Up LDAP Repository

The default Authentication, Authorization, Role Mapper, and Credential Mapper providers that are installed with WebLogic Server store their data in an LDAP server. Each WebLogic Server instance contains an embedded LDAP server. The Administration Server contains the master LDAP server which is replicated on all Managed Servers. If any of your security realms use these installed providers, you should maintain an up-to-date backup of the following directory tree:

domain_name\servers\ *adminServer*\data\ldap

where *domain_name* is the domain root directory and *adminServer* is the directory in which the Administration Server stores run time and security data.

Each WebLogic Server instance has an LDAP directory, but you only need to back up the LDAP data on the Administration Server—the master LDAP server replicates the LDAP data from each Managed Server when updates to security data are made. WebLogic security providers cannot modify security data while the domain Administration Server is unavailable. The LDAP repositories on Managed Servers are replicas and cannot be modified.

The `ldap\ldapfiles` subdirectory contains the data files for the LDAP server. The files in this directory contain user, group, group membership, policies, and role information. Other subdirectories under the `ldap` directory contain LDAP server message logs and data about replicated LDAP servers.

Do not update the configuration of a security provider while a backup of LDAP data is in progress. If a change is made—for instance, if an administrator adds a user—while you are backing up the `ldap` directory tree, the backups in the `ldapfiles` subdirectory could become inconsistent. If this does occur, consistent, but potentially out-of-date, LDAP backups are available, because once a day, a server suspends write operations and creates its own backup of the LDAP data. It archives this backup in a `ZIP` file below the `ldap\backup` directory and then resumes write operations. This backup is guaranteed to be consistent, but it might not contain the latest security data.

See Configure backups for embedded LDAP servers in *Oracle WebLogic Server Administration Console Online Help*.

4.2.3 Back Up SerializedSystemIni.dat and Security Certificates

Each server instance creates a file named `SerializedSystemIni.dat` and locates it in the `/security` directory. This file contains encrypted security data that must be present to boot the server. You must back up this file.

If you configured a server to use SSL, you must also back up the security certificates and keys. The location of these files is user-configurable.

4.3 WebLogic Server Exit Codes and Restarting After Failure

When a server instance stops, it issues an exit code. The value of the exit code provides information about the conditions under which the server process ended. Each

server exit code has significant meaning and specific restart recommendation to be followed.

When a server instance under Node Manager control exits, Node Manager uses the exit code to determine whether or not to restart the server instance. The server exit code can be used by other high-availability agents or scripts to determine what, if any action, to take after a server instance exits. Server exit codes are defined in the following table:

Table 4-1 WebLogic Server Exit Codes

Exit Code Value	Meaning	Restart Recommendation
Less than 0	A negative value indicates that the server instance failed during a state transition, and did not terminate in a stable condition. Example: If a Start in Standby command is issued for a server instance whose configuration is invalid, the server instance fails in the transitional `STARTING` state, and does not achieve the `STANDBY` state.	Do not attempt to restart the server. Diagnose the problem that caused the server process to exit.
0	Indicates that the server process terminated normally, as a result of a shutdown command, either graceful or forced.	None.
Greater than 0	A positive value indicates that the server instance stopped itself after determining that one or more of its subsystems were unstable. Example: A server instance detects an out of memory condition or stuck threads, and shuts itself down.	The server instance can be restarted.

4.4 Restarting a Failed Administration Server

You can restart a failed Administration Server either by using Node Manager or by considering the listen address scenarios depending on the same or different machines sharing the same IP addresses.

The following sections describe how to start an Administration Server after a failure.

> **Note:**
>
> You can use Node Manager to automatically restart a failed Administration Server. See Restart Administration and Managed Servers in the *Administering Node Manager for Oracle WebLogic Server*.

4.4.1 Restarting an Administration Server

See Starting and Stopping Servers.

4.4.1.1 Restarting Administration Server Scenarios

Table 4-2 Administration Server Restart Scenarios

Listen Address Definition	Same Machine or Different Machine with Same IP Address	Different Machine with Different IP Address
Not defined	1. If you are starting AS on a different machine with the same IP address: a. Install WebLogic Server. b. Move data. 2. Start the Administration Server. Running MSs will reconnect automatically at the next `AdminReconnectIntervalSecs`. 3. To start an MS that was not running when AS failed, no change in command is required.	1. Install WebLogic Server. 2. Move data. 3. Start the Administration Server. MSs that were running when AS went down, will not reconnect because they know only the previous Administration Server URL. See Managed Servers and the Re-started Administration Server. Restart MSs supplying the new AS listen address on the command line. 4. To start an MS that was not running when AS failed, supply the new AS listen address on the command line.
DNS name or IP address of the host	1. If you are starting AS on a different machine with the same IP address: a. Install WebLogic Server. b. Move data. 2. Start the Administration Server. Running MSs will reconnect automatically at the next `AdminReconnectIntervalSecs`. 3. To start an MS that was not running when AS failed, no change in command is required.	1. Install WebLogic Server. 2. Move data. 3. Update the `config.xml` with the IP address of the new host machine. See Restarting an Administration Server on Another Machine. 4. Start the Administration Server. MSs that were running when AS went down, will not reconnect because they know only the previous Administration Server URL. See Managed Servers and the Re-started Administration Server. Restart MSs supplying the new AS listen address or DNS name on the command line. 5. To start an MS that was not running when AS failed, supply the new AS listen address or DNS name on the command line.

Table 4-2 (Cont.) Administration Server Restart Scenarios

Listen Address Definition	Same Machine or Different Machine with Same IP Address	Different Machine with Different IP Address
DNS name mapped to multiple hosts	1. If you are starting AS on a different machine with the same IP address: a. Install WebLogic Server. b. Move data. 2. Start the Administration Server. Running MSs will reconnect automatically at the next `AdminReconnectIntervalSecs`. 3. To start an MS that was not running when AS failed, no change in command is required.	1. Install WebLogic Server. 2. Move data. 3. Update the `config.xml` with the IP address of the new host machine. See Restarting an Administration Server on Another Machine. 4. Start the Administration Server. Running MSs that were started with a DNS name for the Administration Server URL that maps to multiple IPs, will attempt reconnection to AS on all the available URLs. MSs can then locate the AS that has been restarted at any of the URLs. 5. To start an MS that was not running when AS failed, supply the DNS name on the command line.

4.4.1.2 Restarting an Administration Server on Another Machine

If a machine crash prevents you from restarting the Administration Server on the same machine, you can recover management of the running Managed Servers as follows:

1. Install the WebLogic Server software on the new administration machine (if this has not already been done).
2. Make your application files available to the new Administration Server by restoring them from backups or by using a shared disk. Your application files should be available in the same relative location on the new file system as on the file system of the original Administration Server.
3. Make your configuration and security data available to the new administration machine by restoring them from backups or by using a shared disk. See Directory and File Backups for Failure Recovery.
   - Update the `config.xml` with the IP address of the new host machine. If the listen address was set to blank, you do not need to change it. For example:

     ```
     <server>
        <name>AdminServer</name>
        ...
        <listen-address></listen-address>
     </server>
     ```

   - You can edit `config.xml` manually or use WLST offline to update the listen address.
4. Restart the Administration Server on the new machine.

4.4.1.3 Managed Servers and the Re-started Administration Server

If an Administration Server stops running while the Managed Servers in the domain continue to run, each Managed Server periodically attempts to reconnect to the Administration Server, at the interval specified by the `ServerMBean` attribute `AdminReconnectIntervalSeconds`. By default, `AdminReconnectIntervalSeconds` is ten seconds.

In order for Managed Servers to reconnect after an Administration Server is restarted on a different IP address, you must have:

- Configured a DNS name for the Administration Server URL that maps to multiple IP addresses. For example, a DNS server named **`wlsadminserver`** which maps to 10.10.10.1 and 10.10.10.2
- Provided the DNS name for the Administration Server URL when starting the Managed Servers. For example:

  `-Dweblogic.management.server=`*`protocol://`*`wlsadminserver`*`:port`*

  or

  `startManagedWebLogic.cmd` *`managed_server_name protocol://`* **`wlsadminserver`***`:port`*

  If the Administration Server goes down, Managed Servers will attempt to reconnect to the Administration Server on all the available URLs. When the Administration Server comes up on any of these URLs, Managed Servers connect to the Administration Server and stop attempting to reconnect on the other URLs. If the Administration Server goes down again, they attempt to reconnect again.

4.5 Restarting a Failed Managed Server

WebLogic server provides various methods to restart a failed Managed Server regardless of the Administration Server accessibility. If the Managed Server cannot connect to the Administration Server during server startup, it can retrieve its configuration by reading its locally cached configuration from the config directory. If the Administration Server is reachable by the failed Managed Server, you can restart it manually or automatically using Node Manager, or using a command script. Also, you can restart the Managed Server in MSI mode.

The following sections describe how to start Managed Servers after failure. For recovery considerations related to transactions and JMS, see Additional Failure Topics.

4.5.1 Starting a Managed Server When the Administration Server Is Accessible

If the Administration Server is reachable by a Managed Server that failed, you can:

- Restart it manually or automatically using Node Manager—You must configure Node Manager and the Managed Server to support this behavior. See Start, Shut Down, Suspend, and Restart Managed Servers in the *Administering Node Manager for Oracle WebLogic Server*.
- Start it manually with a command or script. See Starting and Stopping Servers.

4.5.2 Starting a Managed Server When the Administration Server Is Not Accessible

If a Managed Server cannot connect to the Administration Server during startup, it can retrieve its configuration by reading its locally cached configuration data from the `config` directory. A Managed Server that starts in this way is running in *Managed Server Independence (MSI)* mode.

4.5.3 Understanding Managed Server Independence Mode

When a Managed Server starts, it tries to contact the Administration Server to retrieve its configuration information. If a Managed Server cannot connect to the Administration Server during startup, it can retrieve its configuration by reading configuration and security files directly. A Managed Server that starts in this way is running in Managed Server Independence (MSI) mode. By default, MSI mode is enabled. For information about disabling MSI mode, see Disable Managed Server independence in *Oracle WebLogic Server Administration Console Online Help*.

In Managed Server Independence mode, a Managed Server:

- Looks in its local `config` directory for `config.xml`—a replica of the domain `config.xml`.
- Looks in its `security` directory for `SerializedSystemIni.dat` and for `boot.properties`, which contains an encrypted version of your user name and password. See Provide User Credentials to Start and Stop Servers.

If `config.xml` and `SerializedSystemIni.dat` are not in these locations in the server domain directory, you can copy them from the Administration Server domain directory.

4.5.3.1 MSI Mode and the Security Realm

A Managed Server must have access to a security realm to complete its startup process.

If you use the security realm that WebLogic Server installs, then the Administration Server maintains an LDAP server to store the domain security data. All Managed Servers replicate this LDAP server. If the Administration Server fails, Managed Servers running in MSI mode use the replicated LDAP server for security services.

If you use a third party security provider, then the Managed Server must be able to access the security data before it can complete its startup process.

4.5.3.2 MSI Mode and SSL

If you set up SSL for your servers, each server requires its own set of certificate files, key files, and other SSL-related files. Managed Servers do not retrieve SSL-related files from the Administration Server though the domain configuration file does store the pathnames to those files for each server. Starting in MSI Mode does not require you to copy or move the SSL-related files unless they are located on a machine that is inaccessible.

4.5.3.3 MSI Mode and Deployment

A Managed Server that starts in MSI mode deploys its applications from its staging directory: *server_root*\stage*appName*.

4.5.3.4 MSI Mode and the Domain Log File

Each WebLogic Server instance writes log messages to its local log file and a domain-wide log file. The domain log file provides a central location from which to view messages from all servers in a domain.

Usually, a Managed Server forwards messages to the Administration Server, and the Administration Server writes the messages to the domain log file. However, when a Managed Server runs in MSI mode, it continues to write messages to its local server log file but does not forward messages to the domain log file.

See How a Server Instance Forwards Messages to the Domain Log in *Configuring Log Files and Filtering Log Messages for Oracle WebLogic Server*.

4.5.3.5 MSI Mode and Managed Server Configuration Changes

If you start a Managed Server in MSI mode, you cannot change its configuration until it restores communication with the Administration Server.

4.5.4 Starting a Managed Server in MSI Mode

> **Note:**
>
> If the Managed Server that failed was a clustered Managed Server that was the active server for a migratable service at the time of failure, perform the steps described in Migrating When the Currently Active Host is Unavailable in *Administering Clusters for Oracle WebLogic Server*. Do not start the Managed Server in MSI mode.

To start up a Managed Server in MSI mode:

1. Ensure that the Managed Server's root directory contains the `config` subdirectory.

   If the `config` directory does not exist, copy it from the Administration Server's root directory or from a backup to the Managed Server's root directory.

   > **Note:**
   >
   > Alternatively, you can use the `-Dweblogic.RootDirectory=`*path* startup option to specify a root directory that already contains these files.

2. Start the Managed Server at the command line or using a script. See Starting and Stopping Servers.

The Managed Server will run in MSI mode until it is contacted by its Administration Server. See Restarting a Failed Administration Server.

4.6 Additional Failure Topics

You can choose other related failure topics to understand the server failure and its recovery process.

For information related to recovering JMS data from a failed server instance, see Configuring WebLogic JMS Clustering in *Administering JMS Resources for Oracle WebLogic Server*.

For information about transaction recovery after failure, see Transaction Recovery After a Server Fails in *Developing JTA Applications for Oracle WebLogic Server*.

For information about recovering from a corrupt or unusable embedded LDAP server file, which prevents the Administration Server from starting, see Backup and Recovery in *Administering Security for Oracle WebLogic Server 12.2.1*.

5
Understanding Server Life Cycle

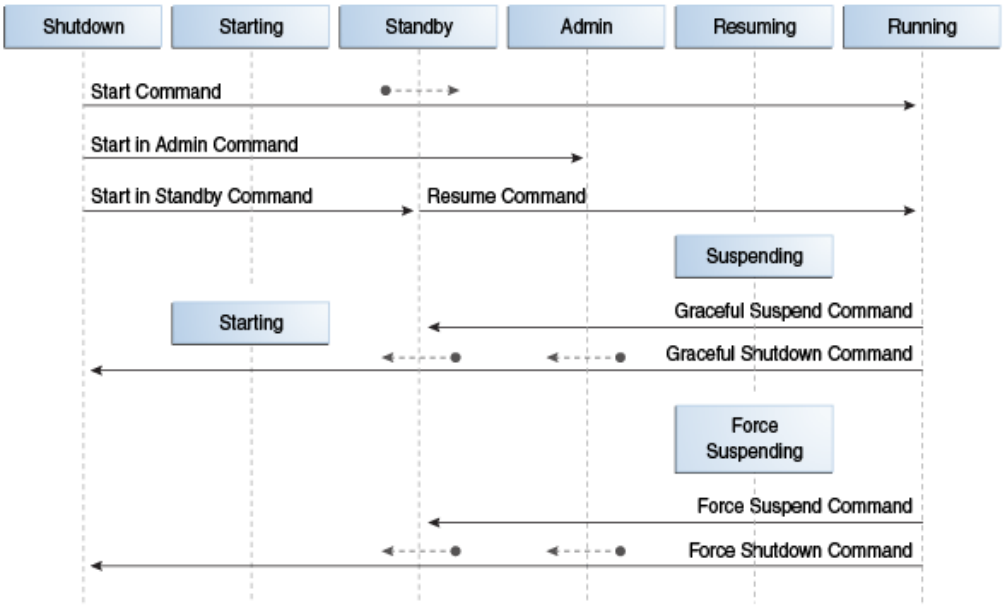
Server life cycle is the series of states through which a WebLogic Server instance can transition. These states cause specific changes to the operational state of a server instance and help to identify the accurate status of the running server. Use the server life cycle commands to track the progress of a booting server at a granular level.

5.1 Diagram of the Server Life Cycle

Server life cycle consists of several components including startup parameters and state transitions.

Figure 5-1 illustrates the server life cycle and the relationships between states and life cycle commands.

Figure 5-1 State Transitions for Server Life Cycle Commands



To understand each state and the relationships among states and server life cycle commands, see Understanding Server States in the Server Life Cycle and Using Server Life Cycle Commands.

5.2 Getting and Using Server State

Server state signifies the specific condition of a server in the life cycle management. System administrators use the server state information to plan the administration tasks related to the application services. You can get the server state using Administration Console or command prompt scripts.

WebLogic Server displays and stores information about the current state of a server instance, and state transitions that have occurred since the server instance started up. This information is useful to administrators who:

- Monitor the availability of server instances and the applications they host
- Perform day-to-day operations tasks, including startup and shutdown procedures
- Diagnose problems with application services
- Plan corrective actions, such as migration of services, when a server instance fails or crashes

Get server state as follows:

- WebLogic Server Administration Console—Multiple pages display state information:
  - On the Summary of Servers page (**Environment** > **Servers**), the Servers table displays the current state of each server instance in the current domain.
  - The ***SERVER_NAME*** > **Monitoring** page displays the state of the currently running server instance, and the date and time it entered the state.
  - **Diagnostics** > **Log Files**, includes timestamped messages for state transitions that have occurred since the server instance was last started.
- Programmatically—Use the `getState()` method on the server's `weblogic.management.runtime.ServerRuntimeMBean`. For example, to monitor the progress of a long-running graceful shutdown process, issue a `getstate` inquiry on a separate thread. For more information, see `ServerRuntimeMBean` in *MBean Reference for Oracle WebLogic Server*.
- WebLogic Scripting Tool—See Getting Run-Time Information in *Understanding the WebLogic Scripting Tool*.

5.3 Understanding Server States in the Server Life Cycle

A server instance works independently in each server state enabling the administrator to track the current status of the server. The states include SHUTDOWN, STARTING, STANDBY, ADMIN, and more.

These sections describe each state in the WebLogic Server life cycle.

5.3.1 SHUTDOWN State

In the `SHUTDOWN` state, a WebLogic Server instance is configured but inactive.

A server instance enters the SHUTDOWN state as result of a Shutdown or Force Shutdown command. In addition, a server instance can kill itself when it detects, as a result of self-health monitoring, that it has become unstable. Only a server instance with its Auto Kill If Failed attribute is true will kill itself when it detects that it is failed. See Automatic Restart for Failed Server Instances.

You can transition a server instance in the SHUTDOWN state to the STARTING state with the Start, Start in Admin, or Start in Standby commands.

5.3.2 STARTING State

During the STARTING state, a WebLogic Server instance transitions from SHUTDOWN to STANDBY, as a result of a Start, Start in Admin, or Start in Standby command.

In the STARTING state, a server instance cannot accept any client or administrative requests.

The server instance obtains its configuration data:

- An Administration Server retrieves domain configuration data, including the domain security configuration, from its config directory.
- A Managed Server contacts the Administration Server for its configuration and security data. If the Managed Server is configured for SSL communications, it uses its own certificate files, key files, and other SSL-related files and contacts the Administration Server for the remaining configuration and security data.

> **Note:**
>
> If the Managed Server cannot contact its Administration Server, by default, it starts up in Managed Server Independence mode, using its locally cached copy of the domain config directory. See Understanding Managed Server Independence Mode.
>
> Do not add non-configuration files in the config directory or subdirectories. Non-configuration files include log (.log) and lock (.lck) files. Administration Server replicates the config directory in all Managed Server instances. Storing non-configuration files in the config directory can cause performance issues in the domain.

The server instance starts the services listed in Table 5-1, in the order listed.

Table 5-1 Services Started in STARTING State

Service	Function
weblogic.management.provider.internal.BeanInfoAccessService	
weblogic.management.PropertyService	
weblogic.management.internal.DomainDirectoryService	
weblogic.upgrade.domain.DomainUpgradeServerService	

Table 5-1 (Cont.) Services Started in STARTING State

Service	Function
`weblogic.management.upgrade.ConfigurationMigrationService`	
`weblogic.deploy.service.internal.DeploymentService`	
`weblogic.management.provider.internal.RuntimeAccessDeploymentReceiverService`	
`weblogic.management.provider.internal.RuntimeAccessService`	
`weblogic.diagnostics.lifecycle.DiagnosticInstrumentationService`	
`weblogic.t3.srvr.BootService`	Includes basic services such as kernel, execute queues, and the server run time.
`weblogic.management.provider.internal.DomainAccessService`	The root for Administration Server-only services.
`weblogic.diagnostics.lifecycle.DiagnosticFoundationService`	The container service for logging and debugging.
`weblogic.nodemanager.NMService`	The Node Manager service, responsible for reporting changes to server status to Node Manager via the server output stream.
`weblogic.timers.internal.TimerService`	
`weblogic.rjvm.RJVMService`	During shutdown, closes all RJVMs except the Administration Server connection.
`weblogic.protocol.ProtocolService`	
`weblogic.t3.srvr.DomainLibService`	Registers configured protocols, making them available for outbound traffic and inbound configuration. Managed Servers require this service to be available early in the startup sequence, to allow them to provide correct addressing information to the Administration Server.
`weblogic.server.channels.ChannelService`	This service is dependent on consistent configuration, and protocols being registered. By this point in the startup sequence, all protocols should have been registered. After this service starts, addressing information, such as `ServerChannelManager.findDefaultLocalServerChannel()`, is available.
`weblogic.server.channels.AdminPortService`	
`weblogic.t3.srvr.ListenerService`	
`weblogic.transaction.internal.PrimordialTransactionService`	The transaction helper is initialized, providing utilities that associate transactions with threads, obtaining the Transaction Manager, obtain the `UserTransaction` object, and perform other tasks. The transaction service itself is not enabled at this point in the startup sequence.
`weblogic.rmi.internal.RMIServerService`	The RMI boot service that is used for initialization only.

Table 5-1 (Cont.) Services Started in STARTING State

Service	Function
weblogic.jndi.internal.NamingService	
weblogic.iiop.IIOPClientService	Installs VM-wide delegates.
weblogic.management.PrimordialManagementService	
weblogic.ldap.EmbeddedLDAP	
weblogic.security.SecurityService	
weblogic.jndi.internal.RemoteNamingService	
weblogic.security.acl.internal.RemoteSecurityService	
weblogic.rmi.cluster.RemoteBinderFactoryService	
weblogic.cluster.ClusterService	
weblogic.iiop.IIOPService	
weblogic.protocol.ProtocolHandlerService	
weblogic.management.internal.AdminService	
weblogic.xml.registry.XMLService	
weblogic.messaging.interception.MessageInterceptionService	
weblogic.cluster.migration.rmiservice.MigratableRMIService	
weblogic.messaging.interception.configuration.Configurator	
weblogic.drs.internal.DataReplicationServic e	
weblogic.management.provider.internal.EditAccessService	Start the Management Edit Service.
weblogic.health.HealthMonitorService	
weblogic.cluster.migration.MigrationService	
weblogic.t3.srvr.T3InitializationService	Initializes deprecated T3 server services such as `BootServicesImpl`.
weblogic.server.channels.ChannelRuntimeService	Addressing information, such as `ServerRuntime.getListenAddress()`, and dynamic updates are available after this point in the startup sequence.
weblogic.store.admin.DefaultStoreService	
weblogic.transaction.internal.TransactionService	
weblogic.jdbc.common.internal.JDBCService	
weblogic.connector.common.ConnectorService	
weblogic.store.admin.StoreDeploymentService	

Table 5-1 (Cont.) Services Started in STARTING State

Service	Function
weblogic.jms.JMSServiceServerLifeCycleImpl	
weblogic.jms.BridgeService	
weblogic.application.ApplicationShutdownService	Checks pending application work during graceful shutdown. Applications are also shutdown here.
weblogic.messaging.saf.internal.SAFServerService	
weblogic.ejb20.deployer.EJB20Service	
weblogic.io.common.internal.FileService	
weblogic.time.server.TimerService	Cancels application triggers during shutdown.
weblogic.rmi.internal.HeartbeatHelperService	Supports heartbeats in protocol-only clients.
weblogic.servlet.internal.WebService	
weblogic.webservice.conversation.internal.ConversationServiceImpl	
weblogic.wtc.gwt.WTCServerLifeCycleImpl	
com.beasys.CORBA.pool.weblogic.WLECService	
weblogic.management.service.ManagedServerNotificationService	
weblogic.webservice.WSServerService	
weblogic.management.mbeanservers.runtime.internal.RuntimeServerService	Run-time JMX services.
weblogic.management.mbeanservers.edit.internal.EditServerService	
weblogic.management.mbeanservers.compatability.internal.CompatabilityMBeanServerService	
weblogic.management.snmp.SNMPService	
weblogic.management.deploy.classdeployment.ClassDeploymentService	Adds handling of startup and shutdown classes.
weblogic.server.ServerLifeCycleService	Handles creation of the server life cycle run-time MBeans to allow for control of the domain.
weblogic.server.channels.EnableAdminListenersService	Enables Admin port before server goes into ADMIN state.
domainweblogic.diagnostics.lifecycle.DiagnosticSystemService	

5.3.3 STANDBY State

A server instance in STANDBY does not process any request—its regular listen port is closed. The administration port is open, and accepts life cycle commands that

transition the server instance to either the RUNNING or the SHUTDOWN state. Other Administration requests are not accepted.

Starting a server instance in standby is a method of keeping it available as a "hot" backup, a useful capability in high-availability environments.

The only life cycle command that causes a server instance to enter the STANDBY state and remain in that state is the Start in Standby command. A server instance transitions through the STANDBY state when you issue a Start or a Start in Admin command.

5.3.4 ADMIN State

In the ADMIN state, WebLogic Server is up and running, but available only for administration operations, allowing you to perform server and application-level administration tasks. When a server instance is in the ADMIN state:

- The Administration Console is available.
- The server instance accepts requests from users with the admin role. Requests from non-admin users are refused.
- Applications are activated in the application ADMIN state. They accept requests from users with the admin and AppTester roles. Users with these roles, accessing an application in the application ADMIN state, have access to all application functionality, not just administrative functions.
- The JDBC, JMS, and JTA subsystems are active, and administrative operations can be performed upon them. However, you do not have to have administrator-level privileges to access these subsystems when the server is in the ADMIN state.
- Deployments or re-deployments are allowed, and take effect when you transition the server instance from the ADMIN to the RUNNING state (using the Resume command).
- ClusterService is active and listens for heartbeats and announcements from other cluster members. It can detect that other Managed Servers have joined the cluster, but is invisible to other cluster members.

You can transition a server instance to the ADMIN state using the Start in Admin, Suspend, or Force Suspend commands.

A server instance transitions through the ADMIN state as a result of Start, Shutdown, and Force Shutdown commands.

You can transition a server instance in the ADMIN state to RUNNING with the Resume command, or to SHUTDOWN, with the Shutdown or Force Shutdown command.

5.3.5 RESUMING State

During this transitional state, WebLogic Server performs the operations required to move itself from the STANDBY or ADMIN state to the RUNNING state.

A server instance transitions to the RESUMING state when you issue the Resume command. A server instance transitions through the RESUMING state when you issue the Start command.

5.3.6 RUNNING State

In the RUNNING state, WebLogic Server is fully functional, offers its services to clients, and can operate as a full member of a cluster.

A server instance transitions to the RUNNING state as a result of the Start command, or the Resume command from the ADMIN or STANDBY states.

You can transition a server instance in the RUNNING state to the SUSPENDING state or the FORCE_SUSPENDING state using graceful and force Suspend and Shutdown commands.

> **Note:**
>
> You can use ReadyApp to determine when applications on the server instance are ready to accept requests. See Using the ReadyApp Framework in *Deploying Applications to Oracle WebLogic Server*.

5.3.7 SUSPENDING State

During this transitional state, WebLogic Server performs the operations required to place itself in the ADMIN state, suspending a subset of WebLogic Server subsystems and services in an ordered fashion, and completing a predefined portion of the application work currently in process ("in-flight" work).

A server instance transitions to the SUSPENDING state when you issue the Suspend command. A server instance transitions through the SUSPENDING state when you issue a Shutdown command.

For information about in-flight work, see Processing In-Flight Work During Suspend and Shutdown.

> **Note:**
>
> While in the SUSPENDING state, Work Managers complete in-flight processing for pending work in application threads. However, while in this state, you cannot schedule new work to Work Managers. See Understanding Work Managers in *Administering Server Environments for Oracle WebLogic Server*.

5.3.8 FORCE_SUSPENDING State

During this transitional state, WebLogic Server performs the operations required to place itself in the ADMIN state, suspending a subset of WebLogic Server subsystems and services in an ordered fashion. During the FORCE_SUSPENDING state, WebLogic Server does not complete in-flight work gracefully; application work in progress is abandoned.

A server instance transitions through the FORCE_SUSPENDING state when you issue the Force Suspend or Force Shutdown command.

5.3.9 SHUTTING_DOWN State

During this transitional state, WebLogic Server completes the suspension of subsystems and services and does not accept application or administration requests.

A server instance transitions to the `SHUTTING_DOWN` state when you issue a Shutdown or Force Shutdown command.

5.3.10 FAILED State

A running server instance can fail as a result of out-of-memory exceptions or stuck application threads, or if one or more critical services become dysfunctional. A server instance monitors its health, and upon detecting that one or more critical subsystems are unstable, it declares itself `FAILED`.

When a server instance enters the `FAILED` state, it attempts to return to a non-failed state. If it failed prior to reaching the `ADMIN` state, the server instance shuts itself down with an exit code that is less than zero. See WebLogic Server Exit Codes and Restarting After Failure.

If the server instance fails after reaching the `ADMIN` state, but before reaching the `RUNNING` state, by default, it returns to the `ADMIN` state, if the administration port is enabled.

> **Note:**
>
> If desired, you can configure a server instance that fails after reaching the `ADMIN` state, to shut itself down, rather than return to the `ADMIN` state.

A server instance can enter the `FAILED` state from any other state. However, once a server instance has entered the `FAILED` state, it cannot return to a running state directly. The `FAILED` state is fatal and a server must go into the `ADMIN` or `SHUTDOWN` state before returning to the `RUNNING` state.

> **Note:**
>
> It is theoretically possible that a server could become available again once the `FAILED` state is entered, for example if hung threads causing a failed state become un-hung.
>
> However, once the `FAILED` state is entered, a server must go into the `ADMIN` or `SHUTDOWN` states before returning to `RUNNING`.

You can use the `OverlaodProtectionMBean` to configure automatic actions, such as `SHUTDOWN` or move to the `ADMIN` state, for when a server instance enters the `FAILED` state.

5.3.11 FAILED_NOT_RESTARTABLE State

This state indicates that the Managed Server has failed and one of the following scenarios has occurred:

- The Node Manager `AutoRestart` attribute is set to false, and Node Manager will not restart the Managed Server.
- Node Manager has exceeded the configured `RestartMax` value, which specifies the number of times Node Manager will attempt to restart a failed server within the interval defined by `RestartInterval`. See Server Startup Properties in *Administering Node Manager for Oracle WebLogic Server*.

If a Managed Server is in the `FAILED_NOT_RESTARTABLE` state, the root failure should be investigated. However, you can perform the same lifecycle operations as if the server instance was in the `SHUTDOWN` state (such as Start, Start in Admin, or Start in Standby commands) without any additional actions.

5.4 Using Server Life Cycle Commands

Server life cycle commands help to transition a server instance from one state to another.

This section describes each life cycle command, how to issue it, and its effect on the state of a server instance. For more information on:

- How to issue life cycle commands, see:
  - Life Cycle Commands and Managing Servers and Server Life Cycle in *Understanding the WebLogic Scripting Tool*.
  - Start and stop servers in the *Oracle WebLogic Server Administration Console Online Help*.
  - Starting and Stopping Servers.
- Node Manager processing related to key life cycle events in environments that use Node Manager, see How Node Manager Works in the WebLogic Server Environment in the *Administering Node Manager for Oracle WebLogic Server*.
- The processing that occurs during each life cycle state, see Understanding Server States in the Server Life Cycle.

For an illustration of the relationship between server states and server life cycle, see Figure 5-1.

5.4.1 Start

The Start command transitions a server instance from the `SHUTDOWN` state to the `RUNNING` state. Depending on the initial state of a server instance, the Start command causes these state transitions:

```
SHUTDOWN > STARTING > STANDBY > ADMIN > RESUMING > RUNNING
```

The `ServerMBean.StartupMode` attribute lets you specify the state in which a server instance should be started. Its values are displayed and configurable in the WebLogic Server Administration Console, using WLST, or when specified as a `java weblogic.Server` startup option. If you do not specify a startup mode value (either on

the command line, in the WebLogic Server Administration Console, or in `config.xml`), the default is to start in the RUNNING state.

See Specify a startup mode in the *Oracle WebLogic Server Administration Console Online Help* and Other Server Configuration Options in *Command Reference for Oracle WebLogic Server*.

Command Usage

See `start`, `startServer`, and `nmStart` in *WLST Command Reference for WebLogic Server* and Start Managed Servers from the Administration Console in the *Oracle WebLogic Server Administration Console Online Help*.

5.4.2 Start in Standby

The Start command, with Standby mode enabled, transitions a server instance from the `SHUTDOWN` state to the `STANDBY` state, with this sequence of state transitions:

```
SHUTDOWN > STARTING > STANDBY
```

Command Usage

See `-Dweblogic.management.startupmode` in *Command Reference for Oracle WebLogic Server* and Start Managed Servers in Standby mode in the *Oracle WebLogic Server Administration Console Online Help*.

5.4.3 Start in Admin

The Start command, with Admin mode enabled, transitions a server instance from the `SHUTDOWN` state to the `ADMIN` state, with this sequence of state transitions:

```
SHUTDOWN > STARTING > STANDBY > ADMIN
```

Command Usage

See `-Dweblogic.management.startupmode` in *Command Reference for Oracle WebLogic Server* and Start Managed Servers in Admin mode in the *Oracle WebLogic Server Administration Console Online Help*.

5.4.4 Resume

The Resume command transitions a server instance from the `STANDBY` or `ADMIN` state to the `RUNNING` state, with this sequence of state transitions:

```
STANDBY > ADMIN > RESUMING > RUNNING
```

Command Usage

See `resume` in *WLST Command Reference for WebLogic Server* and Resume a server in the *Oracle WebLogic Server Administration Console Online Help*.

5.4.5 Graceful Suspend

The Graceful Suspend command transitions a server instance from the `RUNNING` state to the `ADMIN` state, allowing work in process to be handled gracefully, with this sequence of state transitions:

```
RUNNING > SUSPENDING > ADMIN
```

Command Usage

See `suspend` in *WLST Command Reference for WebLogic Server* and Suspend a server in the *Oracle WebLogic Server Administration Console Online Help*.

5.4.6 Force Suspend

The Force Suspend command transitions a server instance from the `RUNNING` state to the `ADMIN` state, without handling work in process gracefully, with this sequence of state transitions:

```
RUNNING > FORCE_SUSPENDING > ADMIN
```

Command Usage

See Suspend a server in the *Oracle WebLogic Server Administration Console Online Help*.

5.4.7 Graceful Shutdown

The Graceful Shutdown command transitions a server instance from the `RUNNING` state to the `SHUTDOWN` state, allowing work in process to be handled gracefully, with this sequence of state transitions:

```
RUNNING > SUSPENDING > ADMIN > SHUTTING_DOWN > SHUTDOWN
```

Command Usage

See `shutdown` in *WLST Command Reference for WebLogic Server* and Shut down a server instance in the *Oracle WebLogic Server Administration Console Online Help*.

5.4.7.1 Controlling Graceful Shutdown

`ServerMBean` has two attributes for controlling the length of the graceful shutdown process. Their values are displayed and configurable on the ***SERVER_NAME*** > **Control** > **Start/Stop** page:

- Ignore Sessions During Shutdown—If you enable this option WebLogic Server will drop all HTTP sessions immediately, rather than waiting for them to complete or timeout. Waiting for abandoned sessions to timeout can significantly lengthen the graceful shutdown process, because the default session timeout is one hour.
- Graceful Shutdown Timeout—Specifies a time limit for a server instance to complete a graceful shutdown. If you supply a timeout value, and the server instance does not complete a graceful shutdown within that period, WebLogic Server performs a forced shutdown on the server instance.

See Control graceful shutdowns and Shut down servers in a cluster in the *Oracle WebLogic Server Administration Console Online Help*.

5.4.7.2 Shutdown Operations and Application Undeployment

During both graceful and forced shutdown, subsystems undeploy applications as appropriate. This processing can result in invocation of application code, such as Servlet `destroy()` or `ejbRemove()` during shutdown. During the shutdown sequence,

JMS, JDBC, and transactions are shutdown *after* applications are shutdown, allowing application code to access JMS, JDBC, and transaction services.

5.4.8 Force Shutdown

The Force Shutdown command transitions a server instance from the any state to the `SHUTDOWN` state, without allowing work in process to be handled gracefully. When run for a server instance in the `RUNNING` state, the Force Shutdown command results in these state transitions:

```
RUNNING > FORCE_SUSPENDING > ADMIN > STANDBY > SHUTDOWN
```

Command Usage

See `shutdown` in *WLST Command Reference for WebLogic Server* and Shutdown a server instance in the *Oracle WebLogic Server Administration Console Online Help*.

A forced shutdown is immediate—WebLogic Server subsystems stop all application processing currently in progress. A forced shutdown can be performed on a server instance in any state.

If a fatal exception causes the forced shutdown to fail, the server will exit after the number of seconds specified by the `ServerLifecycleTimeoutVal` attribute in `ServerMBean`.

> **Note:**
>
> When you force shutdown a server instance in a cluster, a clustered service will fail over to another server instance in the cluster, if its state is replicated on another server instance. However:
>
> - If you issue a Forced Shutdown command on a server instance that hosts an HTTP session for which a secondary session has not yet been created, the session will be lost.
> - If you issue a Forced Shutdown command on a server instance that hosts the replicated state of a stateful session EJB, and the server instance that hosts the EJB fails (the primary), the EJB will not fail over, because its replicated state no longer exists.

For information about undeployment processes during a forced shutdown, and related programming considerations, see Shutdown Operations and Application Undeployment.

5.5 Processing In-Flight Work During Suspend and Shutdown

Subsystems are self-contained systems and responsible for performing each work item in WebLogic Server. The subsystem includes the web containers and the various services associated with the server. Each subsystem follows a specific approach to handle the work during Suspend and Shutdown operations.

The following sections describe how each subsystem handles work in process during `SUSPENDING` and `SHUTTING_DOWN` operations.

5.5.1 RMI Subsystem

The Remote Method Invocation (RMI) subsystem suspends in three steps. Each step in this process completes before the following step commences.

1. Non-transaction remote requests are rejected by the Non-Transaction RMI Service.
2. The Client Initiated Transaction Service waits for pending client transactions to complete.
3. The Remote RMI Service rejects all remote requests with or without transactions.

   After these steps are completed, no remote client requests are allowed. Requests with administrative privileges and internal system calls are accepted.

   When a clustered server instance is instructed to prepare to suspend, the RMI system refuses any in-memory replication calls, to allow other cluster members to choose new hosts for replicated sessions.

5.5.2 Web Container

After the Web Container subsystem is instructed to prepare to suspend, it rejects new sessions requests. Existing sessions are handled according to the persistence method:

- No persistence—Pending sessions with no persistence are allowed to complete.
- In-memory replication in a cluster—Sessions with secondary sessions are immediately suspended. If a primary session does not have a secondary session, the Web Container waits until a secondary session is created, or until the session times out, whichever occurs first.
- JDBC persistence and file persistence—The Web Container immediately suspends sessions that are replicated in a database or file store.

The completion of pending sessions is optional. To drop all sessions immediately, use the Ignore Sessions During Shutdown option on the ***SERVER_NAME*** > **Control** > **Start/Stop** page in the WebLogic Server Administration Console, or the `-ignoreSessions` option with the WLST `shutdown` command.

In a cluster, when a primary session is dropped, the corresponding replicated sessions on another clustered instance will be also destroyed, in addition to the primary session on the server that is being gracefully shut down.

5.5.3 Timer Service

The Timer Service cancels all triggers running on application execute queues. Application execute queues include the default queue and queues configured through the `ExecuteQueueMBean`.

5.5.4 Application Service

The Application Service completes pending work in the application queues before suspending. Application execute queues include the default queue and queues configured through the `ExecuteQueueMBean`.

5.5.5 EJB Container

The EJB Container suspends Message Drive Beans (MDBs).

5.5.6 JMS Service

The Java Messaging Service (JMS) marks itself as suspending, which causes new requests to be rejected. The JMS system suspends gracefully in this fashion:

If the server instance being shut down has a JMS server:

- Any send requests that are waiting because of message quotas are returned immediately.
- All consumers on destinations belonging the JMS Server are closed.
- The persistent store is closed.

If the server instance being shutdown has a JMS connection factory:

- Client connections are closed.

Generally each step in the graceful suspend of the JMS subsystem occurs quickly—in less than a second. Potentially, completion of a client request could take longer, if the request requires higher than normal disk I/O, for example, a request for a persistent "send" of a 100-megabyte message.

You can monitor the number of connections to a JMS server, the number of consumers to a JMS connection factory, and related run-time information using JMS run-time MBeans, including `JMSRuntimeMbean`, `JMSConnectionRuntimeMBean`, `JMSConsumerRuntimeMBean`.

5.5.7 JDBC Service

The JDBC Service closes idle connections in the connection pools.

> **Note:**
>
> If connections are still in use, the shutdown of the JDBC service will fail, and the graceful shutdown will not complete. To shut down a server instance while applications still hold connections, use a forced shutdown command, described in Force Shutdown.

5.5.8 Transaction Service

The Transaction Service waits for the pending transaction count in the Transaction Manager to drop to zero before suspending. Completing all pending transactions can be a lengthy process, depending on the configured transaction timeout.

If a graceful shutdown takes too long because of pending transactions, you can halt it with a forced shutdown command. Force Shutdown suspends all pending work in all subsystems.

A

Starting and Stopping Servers: Quick Reference

Learn simple and frequently used ways to start and shut down instances of WebLogic Server.
This appendix includes the following sections:

- Starting Instances of WebLogic Server
- Shutting Down Instances of WebLogic Server

See Starting and Stopping Servers.

A.1 Starting Instances of WebLogic Server

Learn how to to start Administration Server instances in various domains such as `medrec`, `medrec-spring`, `wl_server`, and more.

In the following table, *`WL_HOME`* refers to the top-level installation directory for WebLogic Server, such as `c:\Oracle\Middleware\Oracle_Home\wlserver\`.

Table A-1 Starting Server Instances

To Start	Do The Following
The MedRec server	Invoke: *`ORACLE_HOME`*`\user_projects\domains\medrec\bin\startWebLogic.cmd` (Windows) *`ORACLE_HOME`*`/user_projects/domains/medrec/bin/startWebLogic.sh` (UNIX) The server starts as an Administration Server in the medrec domain. See Sample Applications and Code Examples in *Understanding Oracle WebLogic Server*.
The MedRec server (Spring version)	Invoke: *`ORACLE_HOME`*`\user_projects\domains\medrec-spring\bin\startWebLogic.cmd` (Windows) *`ORACLE_HOME`*`/user_projects/domains/medrec-spring/bin/startWebLogic.sh` (UNIX) The server starts as an Administration Server in the medrec-spring domain. See Sample Applications and Code Examples in *Understanding Oracle WebLogic Server*.

Table A-1 (Cont.) Starting Server Instances

To Start	Do The Following
The Examples server	Invoke: *ORACLE_HOME*\user_projects\domains\wl_server\bin\startWebLogic.cmd (Windows) *ORACLE_HOME*/user_projects/domains/wl_server/bin/startWebLogic.sh (UNIX) The server starts as an Administration Server in the wl_server domain. See Sample Applications and Code Examples in *Understanding Oracle WebLogic Server*.
An Administration Server that you have created	Invoke: *DOMAIN_NAME*\bin\startWebLogic.cmd (Windows) *DOMAIN_NAME*/bin/startWebLogic.sh (UNIX) where *DOMAIN_NAME* is the name of the directory in which you located the domain, typically *ORACLE_HOME*\user_projects\domains*DOMAIN_NAME*. If the server prompts you to enter a user name and password, enter the name of a WebLogic Server user who has permission to start servers. See Provide User Credentials to Start and Stop Servers. **NOTE:** In a development environment, it is usually sufficient to start an Administration Server and deploy your applications directly on the Administration Server. In a production environment, you typically create Managed Servers to run applications.
Managed Servers	1. Start the domain's Administration Server. 2. Start the Node Manager on the computer that will host the Managed Server you want to start. If it's not already running, you can start Node Manager manually at a command prompt or with a script. See Starting and Stopping Node Manager in the *Administering Node Manager for Oracle WebLogic Server*. 3. Start the domain's Administration Console. See Starting the Administration Console in *Understanding Oracle WebLogic Server*. 4. Associate Managed Servers with Node Manager by assigning them to a Machine upon which Node Manager runs. See Create and configure machines and Assign servers instances to machines in the *Oracle WebLogic Server Administration Console Online Help*. 5. In the left pane of the WebLogic Server Administration Console, expand **Environment** and select **Servers**. 6. In the right pane, select the Control tab. 7. In the Server Status table, select the check box next to the name of the server you want to start and click **Start**. 8. Click **Yes** to confirm. See Starting and Stopping Servers.
A cluster of Managed Servers	To start clustered Managed Servers with Node Manager, see Start Managed Servers in a cluster in the *Oracle WebLogic Server Administration Console Online Help*.

A.2 Shutting Down Instances of WebLogic Server

Use the Administration Console or `stopWeblogic` script to shut down WebLogic Server instances.

It is recommended that you shut down WebLogic Server instances through the Administration Console:

- See Shut down a server instance and Control graceful shutdowns in the *Oracle WebLogic Server Administration Console Online Help*.
- For information on gracefully shutting down the Managed Servers in a cluster, see Shutdown servers in a cluster in the *Oracle WebLogic Server Administration Console Online Help*.

Alternatively, invoke a WebLogic Server stop script to shutdown the server. See Shutting Down Servers with a Stop Script.