Oracle® WebLogic Remote Console

Oracle WebLogic Remote Console Online Help

F89910-04
March 2025

Oracle WebLogic Remote Console Oracle WebLogic Remote Console Online Help,

F89910-04

Copyright © 2024, 2025, Oracle and/or its affiliates.

Primary Author: Oracle Corporation

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

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

If this is software, software documentation, data (as defined in the Federal Acquisition Regulation), or related documentation that is delivered to the U.S. Government or anyone licensing it on behalf of the U.S. Government, then the following notice is applicable:

U.S. GOVERNMENT END USERS: Oracle programs (including any operating system, integrated software, any programs embedded, installed, or activated on delivered hardware, and modifications of such programs) and Oracle computer documentation or other Oracle data delivered to or accessed by U.S. Government end users are "commercial computer software," "commercial computer software documentation," or "limited rights data" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, the use, reproduction, duplication, release, display, disclosure, modification, preparation of derivative works, and/or adaptation of i) Oracle programs (including any operating system, integrated software, any programs embedded, installed, or activated on delivered hardware, and modifications of such programs), ii) Oracle computer documentation and/or iii) other Oracle data, is subject to the rights and limitations specified in the license contained in the applicable contract. The terms governing the U.S. Government's use of Oracle cloud services are defined by the applicable contract for such services. No other rights are granted to the U.S. Government.

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

Oracle®, Java, MySQL, and NetSuite are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

Intel and Intel Inside are trademarks or registered trademarks of Intel Corporation. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. AMD, Epyc, and the AMD logo are trademarks or registered trademarks of Advanced Micro Devices. UNIX is a registered trademark of The Open Group.

This software or hardware and documentation may provide access to or information about content, products, and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third-party content, products, and services unless otherwise set forth in an applicable agreement between you and Oracle. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services, except as set forth in an applicable agreement between you and Oracle.

Contents

Preface

Oracle WebLogic Remote Console Online Help describes how to use WebLogic Remote Console to administer WebLogic Server domains, whether through their Administration Server or as WebLogic Deploy Tooling metadata model files.

Audience

This document is intended for users who are responsible for administering WebLogic Server domains.

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.

Diversity and Inclusion

Oracle is fully committed to diversity and inclusion. Oracle respects and values having a diverse workforce that increases thought leadership and innovation. As part of our initiative to build a more inclusive culture that positively impacts our employees, customers, and partners, we are working to remove insensitive terms from our products and documentation. We are also mindful of the necessity to maintain compatibility with our customers' existing technologies and the need to ensure continuity of service as Oracle's offerings and industry standards evolve. Because of these technical constraints, our effort to remove insensitive terms is ongoing and will take time and external cooperation.

Related Resources

For more information, see these Oracle resources:

- *Understanding Oracle WebLogic Server* - provides an overview of Oracle WebLogic Server features and describes how you can use them to create enterprise-ready solutions.
- *Installing and Configuring Oracle WebLogic Server and Coherence* - describes how to install and configure WebLogic Server and Coherence.
- *Understanding Domain Configuration for Oracle WebLogic Server* - describes WebLogic Server domains and how they are configured.

- *Administering Oracle WebLogic Server with RESTful Management Services* - describes how to use WebLogic Server RESTful management interfaces for administration, monitoring, deploying, and configuration tasks which are exposed for developing RESTful clients, including WebLogic Remote Console.
- *WebLogic Remote Console Accessibility Notes for Oracle WebLogic Server* - describes the accessibility features available in WebLogic Remote Console.

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.

Part I
About WebLogic Remote Console

Oracle WebLogic Remote Console is a graphical user interface that administers Oracle WebLogic Server domains.

WebLogic Remote Console relies on REST APIs to provide flexibility, enabling it to connect to domains in varied environments: on physical or virtual machines, in containers or Kubernetes pods, in the cloud. Additionally, you can use WebLogic Remote Console to manage WebLogic Deploy Tooling (WDT) metadata models - meaning you can build domain templates and then manage them within the same environment.

WebLogic Remote Console is compatible with WebLogic Server 12.2.1.4.0 or later and adjusts its user interface to match the features available to the relevant WebLogic Server release.

Use WebLogic Remote Console to:

- Manage WebLogic Server instances and clusters
- Create and modify WDT metadata models
- Deploy, manage, and monitor applications
- Configure security parameters, including managing users, groups, and roles
- Configure WebLogic Server services, such as database connectivity (JDBC), and messaging (JMS)

See what's new in the latest release of WebLogic Remote Console. Visit WebLogic Remote Console GitHub Repository - Releases.

The latest available release of Desktop WebLogic Remote Console may be ahead of the latest release of Hosted WebLogic Remote Console, which is released with the WebLogic Server Patch Set Update (PSU) on a quarterly basis. Always confirm that you are looking at the appropriate release notes for your current release. You can see the release number of your WebLogic Remote Console in the top-left corner of its user interface.

Related Content

WebLogic Remote Console is part of the thriving WebLogic Server community. Explore the following related products so you can use the console to its full capacity.

- Read general Oracle WebLogic Server documentation in the Oracle Help Center for comprehensive descriptions of the concepts and features of WebLogic Server.
- Read the WebLogic Deploy Tooling documentation to learn how to transform WDT model files into fully functional WebLogic Server domains.
- Read the WebLogic Kubernetes Toolkit User Interface documentation to understand how to deploy WebLogic Server to Kubernetes clusters.

1 Get Started

WebLogic Remote Console is easy to install and you can quickly start managing your domains.

WebLogic Remote Console is available in two formats:

- **Desktop WebLogic Remote Console**, a desktop application installed on your computer.
- **Hosted WebLogic Remote Console**, a web application deployed to an Administration Server and accessed through a browser.

Generally, the two formats have similar functionality, though the desktop application offers certain conveniences that are not possible when using a browser.

If your environment has restrictions on installing applications, consider enabling Hosted WebLogic Remote Console. See Deploy Hosted WebLogic Remote Console.

System Requirements

Review the system requirements needed to run WebLogic Remote Console.

WebLogic Remote Console is compatible with WebLogic Server 12.2.1.4.0 and later.

Note:

Hosted WebLogic Remote Console is only supported on Administration Servers running WebLogic Server 14.1.2.0.0 or later, and is subject to the same system requirements as its associated WebLogic Server release. To see the applicable certification matrix for your release of WebLogic Server, refer to Oracle Fusion Middleware Supported System Configurations.

Desktop WebLogic Remote Console is supported on the following platforms.

Table 1-1 Requirements

Platform	Minimum Requirement
Linux	64 bit only. • Debian 11 and later • Fedora 40 and later – Oracle Linux 7 and later • Ubuntu 16.04 and later
macOS	13 (Ventura) or later **Note**: Intel machines must be 64 bit.
Windows	64 bit only. Windows 10 and later.

We recommend setting the viewport of WebLogic Remote Console to 1300 px or wider. On narrower viewports, UI elements may overlap or disappear.

Install Desktop WebLogic Remote Console

Desktop WebLogic Remote Console is a version of WebLogic Remote Console that is based on the Electron framework and installed as a desktop application.

1. Download the latest version of WebLogic Remote Console from the WebLogic Remote Console GitHub Repository releases page. Choose the appropriate installer for your operating system.
2. Follow the typical process for installing applications on your operating system.
3. Launch WebLogic Remote Console.

If you are using WebLogic Server 14.1.1.0 or earlier, you can enhance the functionality for managing Administration Servers by installing the WebLogic Remote Console extension. See Install the WebLogic Remote Console Extension. The extension comes pre-installed in WebLogic Server 14.1.2.0 and later.

Deploy Hosted WebLogic Remote Console

Hosted WebLogic Remote Console is a version of WebLogic Remote Console that is hosted on the Administration Server of the domain and accessed through a browser.

Hosted WebLogic Remote Console is useful for situations where it is not feasible to install external applications in your environment.

Note:

Deploying Hosted WebLogic Remote Console is only supported on Administration Servers running WebLogic Server 14.1.2.0.0 or later.

1. Start the Administration Server.
2. Deploy the Hosted WebLogic Remote Console application using the WebLogic Scripting Tool (WLST).
    a. Open a command line terminal and go to `ORACLE_HOME/oracle_common/common/bin`.
    b. Invoke WLST with the following options:

       For Unix operating systems, enter:

       ```
       wlst.sh WL_HOME/server/bin/remote_console_deployment.py t3://
       hostname:port username < password.txt
       ```

       For Windows operating systems, enter:

       ```
       wlst.cmd WL_HOME\server\bin\remote_console_deployment.py t3://
       hostname:port username < password.txt
       ```

       Where:

       - *WL_HOME* is the top-level installation directory for WebLogic Server.

- *`hostname`* is the host name of the Administration Server.
- *`port`* is the port number of the Administration Server.
- *`username`* is a user account capable of deploying applications.
- *`password.txt`* is the path to a file containing the password for the specified user account.

Eamples:

On Unix operating systems:

```
wlst.sh /Users/smithdoe/Oracle/Middleware/Oracle_Home/wlserver/
server/bin/remote_console_deployment.py t3://localhost:7001 admin < /
Users/smithdoe/password.txt
```

On Windows operating systems:

```
wlst.cmd
C:\Oracle\Middleware\Oracle_Home\wlserver\server\bin\remote_console_depl
oyment.py t3://localhost:7001 admin < C:\Users\smithdoe\password.txt
```

Hosted WebLogic Remote Console is now deployed and will remain active as long as the `weblogic-remote-console-app` application is deployed and the Administration Server is running.

Do not stop or delete the Hosted WebLogic Remote Console application, `weblogic-remote-console-app`, if you want to continue to use Hosted WebLogic Remote Console.

To use Hosted WebLogic Remote Console to manage the running Administration Server, see Start Hosted WebLogic Remote Console.

Start Hosted WebLogic Remote Console

Hosted WebLogic Remote Console is a version of WebLogic Remote Console that is accessible using a browser.

Before you can start Hosted WebLogic Remote Console, you must have deployed it. See Deploy Hosted WebLogic Remote Console.

1. Start the Administration Server.
2. Open a browser and enter `http://`*`hostname`*`:`*`port`*`/rconsole` (or for HTTPS, `https://`*`hostname`*`:`*`port`*`/rconsole`).

   Where *`hostname`* and *`port`* match the values you set when you deployed Hosted WebLogic Remote Console.

> **Note:**
>
> If your Administration Server is behind a firewall or load balancer, or otherwise externally unavailable, you must manually expose the `rconsole/*` endpoint to make it accessible, similar to exposing `console/*` for the WebLogic Server Administration Console.
>
> This is in addition to exposing the `management/*` endpoint which is required for general domain configuration.

3. Log in to Hosted WebLogic Remote Console.
4. In the **Providers** drawer, select **This Server**.

Hosted WebLogic Remote Console is active as long as the `weblogic-remote-console-app` application is deployed and the Administration Server is running.

> **Note:**
>
> You can use Hosted WebLogic Remote Console to connect to and manage other providers beyond the Administration Server it is hosted on. See Connect to a Provider.

Install the WebLogic Remote Console Extension

The WebLogic Remote Console extension enhances the management capabilities of WebLogic Remote Console for Administration Server connections.

For an overview of the functionality provided by the WebLogic Remote Console extension, see Features of the WebLogic Remote Console Extension.

> **Note:**
>
> You only need to perform this procedure if you are running WebLogic Server 14.1.1.0.0 or earlier. As of WebLogic Server 14.1.2.0.0, the WebLogic Remote Console extension is automatically installed in your domain and updated with each Patch Set Update (PSU).

1. Under *`DOMAIN_HOME`*`/`, create a folder and name it `management-services-ext`.
2. Download the latest WebLogic Remote Console extension, `console-rest-ext-2.4.16.war`, from the WebLogic Remote Console GitHub Repository releases page.
3. Save the extension under `management-services-ext`.

   You do not need to deploy `console-rest-ext-2.4.16.war` as an application in your domain.
4. Restart the Administration Server.
5. In WebLogic Remote Console, disconnect and then reconnect to the Administration Server.

The WebLogic Remote Console extension is updated with WebLogic Remote Console. Whenever you update WebLogic Remote Console, make sure you also update the extension to match.

If you need to update the WebLogic Remote Console extension separately from the console, follow the instructions outlined in step 2 of Upgrade Desktop WebLogic Remote Console. Note that if you manually install the WebLogic Remote Console extension in domains running WebLogic Server 14.1.2.0.0 or later, under *`DOMAIN_HOME`*`/`, it will supersede the version of the extension that is automatically installed and updated.

Note:

With the release of WebLogic Remote Console 2.4.11, the versioning scheme of the WebLogic Remote Console *extension* was updated to match that of WebLogic Remote Console. There are no versions between `console-rest-ext-9.0.war` (released with WebLogic Remote Console 2.4.10) and `console-rest-ext-2.4.11.war` (released with WebLogic Remote Console 2.4.11).

Features of the WebLogic Remote Console Extension

The WebLogic Remote Console extension adds many features to WebLogic Remote Console that are useful for the administration of WebLogic Server domains.

When the WebLogic Remote Console extension is installed, it grants the following capabilities to WebLogic Remote Console:

- **View pending changes** - Identify any changes that are saved to the domain but not yet committed. Pending changes are listed in the Shopping Cart.
- **Access the Security Data Tree perspective** - Edit security data that is stored in the embedded LDAP server, including, but not limited to, managing:
  - Users and Groups
  - Roles and Policies
  - Credential Mappings

  You can also view (but not edit) the users and groups for any authentication provider that supports it.
- **Create and edit application deployment plans** - Use deployment plans to extend or override an application's deployment descriptors.
- **Manage JMS messages** - Import, export, or delete messages.
- **Manage JTA transactions** - Import, export, or delete transactions.
- **Test data source connections** - Verify that the domain can successfully connect to the configured database.
- **View objects in JNDI structure** - Examine objects such as Java EE services and components such as RMI, JMS, EJBs, and JDBC data sources.
- **Upload applications or database client data for redeployment** - Update and redeploy applications or database client data with new versions that were not already deployed to the Administration Server.

  The redeployment of applications or database client data that are already on the server is supported without the extension.

- **Analyze server connection issues -** Test connections between the Administration Server and Managed Servers or clusters.

For installation instructions, see Install the WebLogic Remote Console Extension.

Provider Types

Use WebLogic Remote Console to connect to the following provider types, each of which offers a different approach for managing your WebLogic Server domains.

Provider	Description
Administration Server	Connect to a running WebLogic Server domain through its Administration Server.
WDT Model Files	Edit WebLogic Deploy Tooling (WDT) metadata models of WebLogic Server domains.
WDT Composite Models	Combine and compare settings across multiple WDT model files.
Property Lists	Edit the key-value pairs that enable the use of variables in WDT model files.

Connect to a Provider

You can connect WebLogic Remote Console to a WebLogic Administration Server, WDT model file, or another provider.

1. Open the **Providers** drawer and click **More** ⋮ .
2. Choose a provider type from the list:
   - Add Admin Server Connection Provider
   - Add WDT Model File Provider
   - Add WDT Composite Model File Provider
   - Add Property List Provider
   - Create Provider for new WDT Model File
   - Create Provider for New Property List

   For information on the different providers, see Provider Types.
3. Fill in any required connection details for the selected provider.
4. Click **OK** to establish the connection.

   The Administration Server must be running for the connection to succeed.

Upgrade Desktop WebLogic Remote Console

Upgrade to the latest version of Desktop WebLogic Remote Console and take advantage of the latest features in both the console and WebLogic Server.

If a newer version of Desktop WebLogic Remote Console is available, an alert will appear in the menu bar.

1. In Desktop WebLogic Remote Console, open **Updates Available**, then **Download and install**.

Desktop WebLogic Remote Console update runs in the background so you can continue working in WebLogic Remote Console with no interruptions. The next time the console is launched, the updates will apply.

You can also download the update from the WebLogic Remote Console GitHub Repository releases page and apply the update manually.

2. *For WebLogic Server 14.1.1.0.0 and earlier:* Update the WebLogic Remote Console *extension*.

   In WebLogic Server 14.1.2.0.0 and later, the WebLogic Remote Console extension is included with the initial installation and is updated by subsequent Patch Set Updates (PSU) and should not be updated manually.

   For the best experience, you should keep the versions of WebLogic Remote Console and its extension in sync with each other. That is, when you are running WebLogic Remote Console 2.4.16, you should have `console-rest-ext-2.4.16.war` installed.

   Note:

   With the release of WebLogic Remote Console 2.4.11, the versioning scheme of the extension was updated to match WebLogic Remote Console. Due to this change, the extension version jumps from `console-rest-ext-9.0.war` to `console-rest-ext-2.4.11.war`.

   a. While connected to the domain, open the **Providers** drawer and beside the provider connection, click the **Get Info** icon to see the **Console Extension Version**. If it matches WebLogic Remote Console version, you can skip the rest of the steps. Otherwise, continue.

   b. Go to *`DOMAIN_HOME`*`/management-services-ext` and delete the existing WebLogic Remote Console extension.

   c. Download the WebLogic Remote Console extension that matches your WebLogic Remote Console from the WebLogic Remote Console GitHub Repository. It will be within the matching WebLogic Remote Console release section.

   d. Save the extension to *`DOMAIN_HOME`*`/management-services-ext`.

   e. Restart the Administration Server.

2 Get to Know the Console

WebLogic Remote Console is your portal to managing WebLogic Server domains from a unified interface, whether they are running domains or model templates.

When you launch WebLogic Remote Console for the first time, it opens a **Startup Tasks** dialog box with options to connect to a new provider, such as an Administration Server or WDT model file. WebLogic Remote Console remembers your connection details so on subsequent visits, it will open on your last project, letting you continue where you left off.

The structure of WebLogic Remote Console remains similar across provider types, with variances as appropriate for the provider type.

Key Elements of the Interface

Open the **Providers** drawer to see and manage your saved provider connections.

Browse the **Navigation Tree** to see the structure of the provider as a node tree. In providers based on domains, such as Administration Servers or WDT model files, the node tree builds upon the MBean structure that defines a WebLogic Server domain. Beside the Navigation Tree is the **NavStrip**. On providers with multiple perspectives, you can also use the icons in the NavStrip to move between perspectives.

Click ☰ above the NavStrip to hide the Navigation Tree when you want a bigger editing area. Click it again to bring the Navigation Tree back.

The **Content Pane** is the main editing area for managing your domains or property lists. As you move through the nodes of the Navigation Tree, the Content Pane updates to match your new position.

Tools

Tools are a set of quick actions that you can use to navigate or manipulate the interface. Icons for these actions are scattered across the top of WebLogic Remote Console.

Tool icons only appear on pages where they are applicable. For example, the Shopping Cart, used to indicate pending changes to an active domain, only appears in Administration Server providers.

Table 2-1 Tools

Tool Icon	Description
Home	Sends you to the Home Page of the provider.
Shopping Cart	Indicates if there are pending changes ready to be committed to a WebLogic Server domain. When the **Shopping Cart** is full, you can click it to see the pending changes and then commit or discard them.
Visibility of History	Triggers a record of the pages you visit. When active, a drop-down list appears where you can see, and return to, your previous pages. History is limited to the current perspective. Click **More** ⋮ , then **Clear History Entries** to delete your navigation history.

Table 2-1 (Cont.) Tools

Tool Icon	Description
	Opens the **Providers** drawer where you can manage your provider connections.
←/→ Browse Back and Forward	Browses backward or forward through your previously visited pages in WebLogic Remote Console.
Pages History	Opens the **Pages History** dialog box, which lists the pages that you have previously visited in WebLogic Remote Console.
Landing Page	Sends you to the Landing Page of the current perspective.
Help	Shows reference information about the attributes present on the current page.
Reload	Refreshes any forms or tables on the current page with the latest available data. When **Auto Reload Interval** is enabled, this icon will bounce continuously.
Auto Reload Interval	Sets a time interval (in seconds) for how often WebLogic Remote Console should reload a page to refresh its information. If you select another tab or move to another page, automatic reload stops. To stop auto reload, click the bouncing **Reload** icon or click **Auto Reload Interval** and set the interval to `0`.
Refresh Page Values	Updates the Content Pane with the latest values.

Explore your Domain

WebLogic Server domains can be complex structures, so WebLogic Remote Console provides multiple methods to locate your objective.

Navigation Tree: Expand the nodes of the Navigation Tree to browse its children and gain a high level understanding of the hierarchical relationship of a WebLogic Server domain.

Breadcrumbs: The Content Pane provides a breadcrumb menu to orient you as you descend further into a node branch. You can also click on a node's page title in the Breadcrumb menu to find links to related nodes that take you directly to the related content.

Search: Use the search bar to find any MBeans in the domain that match your query. Searches are restricted to within the current provider and perspective. Previous search queries can be found as the last node in the Navigation Tree, nested under the **Recent Searches** node.

Dashboards: If you find yourself regularly reviewing specific data within your domain, you may want to make use of Dashboards, which let you curate complex search filters and compare disparate information about your domain. For more information, see Dashboards.

Change Indicators

When you modify an MBean attribute, WebLogic Remote Console adds a black outline to the attribute's field to indicate that it is no longer set to the default value.

To return an MBean attribute to its default setting, right-click inside the attribute's field and click **Restore to default**. Save and commit the change.

Note:

The domain configuration file, `config.xml`, only describes MBean attributes if they are set to non-default values. After you use Restore to default, the (now unnecessary) attribute will be removed from the file.

Change Console Preferences

You can customize Desktop WebLogic Remote Console to suit your needs.

1. Go to **File**, then **Preferences**. (On macOS, go to **WebLogic Remote Console**, then **Application Preferences**).
2. Choose a section tab and make your changes as needed.
3. Close the **Preferences** dialog box to apply your changes.

Customize a Table

You can choose which columns are visible in a table, hiding irrelevant data so you can focus on the important details.

1. Find a table that you want to customize. For example, in the **Edit Tree**, go to **Environment**, then **Servers**.
2. Click **Customize Table** to display the list of Available Columns.

   The available column options differ by table as not all columns (that is, properties) are applicable to all tables.
3. Use the arrows to move your wanted columns from **Available Columns** to **Selected Columns**. Move any unwanted columns back to **Available Columns**. Move the columns to **Selected Columns** in the order that you want them to appear in the table. There must be at least one column under **Selected Columns**.
4. Click **Apply** to save your changes.

Changes to table columns will persist indefinitely and apply to that specific table across all applicable providers. If you change the table columns for Server Templates in wdt-model_1, then those changes will also affect the Server Templates table in wdt-model_2 and admin-server_1 (Edit Tree).

To return a table to its default set of columns, open **Customize Table** and click **Reset**.

Projects

Use projects to organize providers in a way that makes sense for your workflow.

You can add as many providers to a project as you want. You can even add the same provider to multiple projects.

For simplicity, we recommend that if you use property lists with your WDT model files, then you should keep the property list provider in the same project as the WDT model file.

Create a Project

To arrange providers into a collection:

1. Open **File**, then **New Project**.
2. Enter a name for the new project.
3. Open the **Providers** drawer and beside the project's name, click **More** ⋮. Select one of the following options to add a provider:
   - Add Admin Server Connection Provider
   - Add WDT Model File Provider
   - Add WDT Composite Model File Provider
   - Add Property List Provider
   - Create Provider for New WDT Model File
   - Create Provider for New Property List

   For more information on the different provider types, see Provider Types
4. Fill in any required information.
5. Click **OK** to add the provider to your project.
6. Optional: Add more providers to the project.

You can add or remove providers from the project at any point. To switch to another project, open **File**, then **Switch to project** and choose the project that you want to switch to.

Deleting a project will not affect the providers within the project.

Export a Project

If you use WebLogic Remote Console on multiple computers, exporting a project is a convenient way to share a project across multiple computers.

Note:

If the project contains a WDT model file, make sure the location of the WDT model file is accessible to all the computers where this project will be imported to.

1. In the Providers drawer, click **More** ⋮ and select **Export Providers as Project**.
2. Enter a name for the exported project. When imported into another WebLogic Remote Console, this will be the name of the project.
3. Enter a name for the file of the exported project.
4. Click **OK** and choose a file location to save the exported project file.

WebLogic Remote Console creates a JSON file with the project details that you can import into other instances of WebLogic Remote Console.

Import a Project

You can import the project details of an existing project to rapidly ramp up productivity in a new installation of WebLogic Remote Console.

1. Export a project and save the file to the computer where you want to import it. See Export a Project.
2. On the computer where you want to import the file, start WebLogic Remote Console.
3. In the Providers drawer, click **More** ⋮ and select **Import Project**.
4. Click **Upload File** and browse to the exported project file.
5. Click **Import**.

The imported project automatically becomes the current project loaded in WebLogic Remote Console.

Enable Keyboard Navigation on macOS

On devices running macOS, you must specifically enable keyboard navigation before you can use the Tab key (or Shift + Tab keys) to navigate between the controls in WebLogic Remote Console.

1. On your device running macOS, open the Apple main menu and select **System Settings**.
2. Click the **Keyboard** item in the sidebar.
3. Turn on the **Keyboard Navigation** option.
4. Optional: If you plan to use the Safari browser with Hosted WebLogic Remote Console, then do the following:
   a. Open the Safari browser and open its main menu.
   b. Select **Settings**.
   c. On the **Advanced** tab, turn on the **Press Tab to highlight each item on a webpage** option.

3

Contribute to WebLogic Remote Console

Contributions from the community to the WebLogic Remote Console project help us build a better experience together. You can contribute by raising a bug or enhancement request or by submitting pull requests to address an issue yourself.

1. Open an issue in the WebLogic Remote Console GitHub repository that describes the issue you plan to address or the enhancement request.

   If you only want to raise a bug or file an enhancement request, you can stop here - you're done! Thank you for helping to improve WebLogic Remote Console.
2. If you want to fix an issue yourself, you must sign the Oracle Contributor Agreement (OCA) before you can submit a pull request. For instructions, see the Oracle Contributor Agreement. We cannot merge pull requests from contributors who have not signed the OCA.
3. Fork the WebLogic Remote Console GitHub repository.
4. Create a branch in your forked repository and implement your changes. Include the issue number in your branch's name. For example, `1234-fixlink`.
5. Test your changes. Build WebLogic Remote Console as described in Build from Source.
6. If the base image was changed, update the samples.
7. Commit your changes. Make sure to include a git commit signoff that lists your name and the email address that matches its entry in the OCA Signatories list. For example, `Signed-off-by: Your Name you@example.org`. You can add this automatically by adding the `--signoff` option to your git commit command: `git commit--signoff`.
8. Submit the pull request. Include a link to its related issue and describe what you hope to accomplish with your changes and how to validate them.

Thank you for your contribution. We will assign reviewers to your pull request.

Build from Source

If you want to contribute to the WebLogic Remote Console project or just want a better understanding of how WebLogic Remote Console works, then you can generate the project from its source code to create a local build.

1. Make sure the following software is installed in your local environment.
   - Java SE 11 or later
   - Maven 3.6.1 or later
   - Node.js 18.0.0 or later

   To verify that you have installed the correct versions of the listed software, run:

   ```
   java -version
   mvn --version
   node -v
   ```

2. Install the Oracle JET 15.1.0 client libraries.
   - In Windows environments, run:

     ```
     npm install --location=global @oracle/ojet-cli@~15.1.0
     ```

   - In Linux or macOS environments, run:

     ```
     sudo npm install --location=global @oracle/ojet-cli@~15.1.0
     ```

   You can run `ojet --version` to verify it installed correctly.
3. Fork the WebLogic Remote Console GitHub repository and then clone it to your machine.
4. Open a command-line interface and navigate to the home directory of the cloned repository.
5. Run `mvn clean install`.

   After the build finishes, confirm that `/installer/target/console.zip` was created.
6. From the home directory of the repository, run the `build-electron.sh` script.

The WebLogic Remote Console executable file is created under `/electron/dist/`.

Run WebLogic Remote Console in Development Mode

If you want to see your changes without building the full WebLogic Remote Console, you can run a development version of WebLogic Remote Console from within a browser.

Note:

WebLogic Remote Console in development mode is distinct from Hosted WebLogic Remote Console. You should only use WebLogic Remote Console in development mode as a tool for development purposes. If you want to perform domain management tasks but your environment limits the installation of standalone applications, then you should use Hosted WebLogic Remote Console. See Deploy Hosted WebLogic Remote Console.

You must have a modern internet browser installed to run WebLogic Remote Console in the browser.

1. Build WebLogic Remote Console from its source code. Follow the instructions at Build from Source.
2. Copy `/installer/target/console.zip` to a new directory and extract its contents.
3. Open a command window and navigate to the new directory.
4. Run `java -jar console.jar`.
5. Open a browser window and enter `http://localhost:8012` in the address bar.
6. Test your changes to WebLogic Remote Console.

To end the session, enter `Ctrl+C` in the command window.

Unsaved changes made in the browser application do not persist after you close the browser tab. If you accidentally close or refresh the browser tab, you may lose all of your changes.

Changes to an Administration Server provider remain in a pending state and can generally be recovered in a new session. Changes to WDT model files, WDT Composite files, and property lists are lost if you do not click **Download File** at the end of your session.

4

Support Information

WebLogic Remote Console offers several avenues to receive support and help.

Note:

Stay up-to-date with the latest release of WebLogic Remote Console to get the latest features and fixes. If you experience issues, try upgrading to the latest release, which may address your issue. For more information, see Upgrade Desktop WebLogic Remote Console.

Oracle Documentation

In addition to the current guide, you can retrieve information on WebLogic Remote Console from within the user interface. Click the **Help** icon in the Tool Bar or in dialog boxes to reveal descriptions of the fields on your current page.

You can also view general documentation for WebLogic Server at Oracle WebLogic Server in the Oracle Help Center.

Oracle Support

Support for WebLogic Remote Console is available to Oracle customers that have purchased support for WebLogic Server.

Community Support

- Join our Slack channel at #remote-console in the Oracle WebLogic workspace! Ask questions, make suggestions, or just get in touch with developers and other users of WebLogic Remote Console.
- Visit our GitHub repository where you can contribute directly to the development of WebLogic Remote Console, whether it's through raising issues or submitting your own code.

Feature Roadmap

WebLogic Remote Console is one of several tools available to manage your WebLogic Server domain. For a list of alternative administration options, see Choosing the Appropriate Technology for Your Administrative Tasks in *Understanding Oracle WebLogic Server*.

WebLogic Remote Console is an evolving product that continues to add new features with each release. As WebLogic Remote Console is an open-source product, you can view our progress in our WebLogic Remote Console GitHub repository.

We are currently considering the implementation of the following features. Note that these features are not listed in any particular order; a higher position does not indicate a higher priority.

- Add support for adding and configuring certificate registries.

- Add support for adding JASPIC authentication configuration providers.
- Add support for viewing log file contents within WebLogic Remote Console.
- Add support for viewing JMS message details within WebLogic Remote Console.
- Add support for deploying applications stored in the application installation directory.
- Add support for adding and managing SNMP credential mappings.
- Add support for adding and managing PKI credential mappings.
- Add support for securing JNDI nodes using roles and policies.
- Add support for adding and deleting durable JMS subscribers.
- Add support for importing and exporting security data.
- Add support for whole server migration.
- Add support for service migration of singleton services.
- Add support for starting and stopping all servers in a cluster simultaneously.
- Add support for validating RDBMS security store configuration.
- Add support for notifying users of required system resource restarts.
- Add support for configuring JDBC data sources with an unlisted (or Other) database type.
- Improve editing of transaction support variable assignments.
- Improve configuration of custom security providers.
- Improve server cloning process to include server children, such as channels.

Frequently Asked Questions

This section provides answers to frequently asked questions about WebLogic Remote Console.

My domain is behind a firewall. How do I connect to WebLogic Remote Console?

For WebLogic Remote Console to connect to a domain's Administration Server, the management endpoint of the domain, `management/*`, must be publicly accessible. If your Administration Server is behind a firewall or load balancer, or otherwise externally unavailable, you will need to expose the endpoint manually.

If you use the Hosted WebLogic Remote Console, you will also need to expose `rconsole/*`.

Tip:

If you needed to expose `console/*` to access the WebLogic Server Administration Console for your domain, simply follow the same procedure to expose `management/*` for WebLogic Remote Console.

Can I connect WebLogic Remote Console to domains running in WebLogic on Oracle Cloud Infrastructure?

Yes, you can. First, make sure that the domain’s Administration Server is publicly accessible so WebLogic Remote Console can establish a connection to it. Then, in WebLogic Remote

Console, enter your credentials and the publicly accessible URL for the WebLogic Administration Server.

Can I connect WebLogic Remote Console to domains running on other cloud providers such as Amazon Web Services, Google Cloud, Microsoft Azure and so on?

Yes, you can connect to other cloud providers. You'll need to make sure the domain's Administration Server is publicly accessible to allow a connection between WebLogic Remote Console and the cloud provider. Then, in WebLogic Remote Console, enter your credentials and the publicly accessible URL for the WebLogic Administration Server.

Visit your cloud providers' documentation for more specific instructions on how to expose the `management/*` endpoint of the WebLogic Server.

Can I connect WebLogic Remote Console to domains running in Kubernetes with the WebLogic Kubernetes Operator?

Yes. For details on how to set up access to WebLogic Server domains running on Kubernetes, see Use the WebLogic Remote Console in the *WebLogic Kubernetes Operator User Guide*.

Which versions of WebLogic Server can I use with WebLogic Remote Console?

WebLogic Remote Console is compatible with WebLogic Server 12.2.1.4 or later.

Can I use both WebLogic Remote Console and WebLogic Server Administration Console to manage WebLogic Server Administration Servers?

It depends on the WebLogic Server release that you have installed. The Administration Console was removed in WebLogic Server 14.1.2.0.0 and therefore is not available as of that release.

However, if you're running WebLogic Server 14.1.1.0 or earlier, then you can continue to use the Administration Console for domain management alongside WebLogic Remote Console. Any changes you make in WebLogic Remote Console are reflected in the Administration Console.

As with any combination of system administration tools, avoid using them simultaneously as it can cause configuration conflicts and unexpected behavior.

To see a full list of the tools that you can use to administer WebLogic Server, see Overview of WebLogic Server System Administration in *Understanding Oracle WebLogic Server*.

Do I need to upgrade my installation of WebLogic Remote Console whenever I upgrade or patch WebLogic Server?

No, older releases of WebLogic Remote Console will continue to work with newer releases of WebLogic Server. However, we recommend that you update WebLogic Remote Console (and its extension) as frequently as possible to take advantage of any fixes and improvements, both to WebLogic Remote Console itself and those added in WebLogic Server patches.

For example, if new fields or WebLogic MBeans were added in a WebLogic Server update, then outdated versions of WebLogic Remote Console will not detect or display those fields.

Are there any security risks associated with WebLogic Remote Console?

No, WebLogic Remote Console accesses WebLogic Administration Servers through its standard WebLogic REST API, which is available out of the box. However, since your computer will be accessing potentially sensitive data, you need to make sure it is protected and secure.

Why does WebLogic Remote Console disagree with attributes that I set at the command line?

When you make changes to the domain configuration using system properties, the properties override but do *not* change the `config.xml` file. The **Edit Tree** perspective, which is generated from `config.xml`, will only show the state of the domain as specified in the configuration file. In contrast, the **Configuration View Tree** perspective shows the current and effective state of the domain, including command line options.

See Perspectives in the Administration Server Provider.

As an example, if your domain is set to secured production mode and you choose to temporarily disable secured production mode with the following system property, `-Dweblogic.securemode.SecureModeEnabled=false`, then only the Configuration View Tree will accurately report the domain mode as production mode. The Edit Tree (and `config.xml`) will continue to report that secured production mode is enabled, even though it is not.

See Verifying Attribute Values That Are Set on the Command Line in *Command Reference for Oracle WebLogic Server*.

Part II
Administration Server

WebLogic Remote Console connects directly to a WebLogic Administration Server to view and edit its domain.

WebLogic Remote Console administers your domain by connecting to its Administration Server through REST APIs generated by WebLogic Server RESTful management services. The WebLogic RESTful management services dynamically generate REST resources mapped from WebLogic Server MBeans. WebLogic Remote Console builds upon these REST resources to produce a graphical user interface that reflects the MBean structure of your domain and that can connect to domains running remotely.

See About the WLS RESTful Management Interface in *Administering Oracle WebLogic Server with RESTful Management Services* for more information.

For an overview of WebLogic Server domains, their contents, and general guidance on domain configuration, see Understanding Oracle WebLogic Server Domains in *Understanding Domain Configuration for Oracle WebLogic Server*.

WebLogic Remote Console is one of several tools available to administer your WebLogic Server domain, each with its own advantages. See Choosing the Appropriate Technology for Your Administrative Tasks in *Understanding Oracle WebLogic Server* for a list of other system administration options.

5

Domain Configuration

Use WebLogic Remote Console to make configuration changes to your WebLogic Server domain through its Administration Server.

Connect to an Administration Server

You can connect to a running WebLogic Server domain through its Administration Server and then manage its configurations using WebLogic Remote Console.

1. Start the Administration Server.

   See Starting and Stopping Servers in *Administering Server Startup and Shutdown for Oracle WebLogic Server* for instructions.
2. Open the **Providers** drawer and click **More** ⋮.
3. From the list, select **Add Admin Server Connection Provider**.
4. Enter a name for the Administration Server.

   This name appears in the Project list of providers so you can identify which domain you're connected to.
5. Optional: *For domains running WebLogic Server 14.1.2.0.0 or later only:* If you have configured WebLogic Server to delegate authentication to an external service using a browser, enable the **Use Web Authentication** option. Otherwise, leave it unselected and enter credentials for a local user account as described in step 6.

   For more information, see Configure Web Authentication.
6. Enter a username and password for a user account in the domain.

   Log in with a user who has the necessary privileges for the tasks you plan to perform. To prevent unauthorized modifications, WebLogic Remote Console limits which tasks and screens are available to a user, based on their role.
7. In the **URL** field, enter the URL for the Administration Server.

   For example:

   ```
   http://localhost:7001
   ```

   Make sure that the `management/*` endpoint of your Administration Server is accessible by WebLogic Remote Console. If your Administration Server's endpoints are behind a firewall, load balancer, or otherwise externally unavailable, you will need to expose the endpoint manually.

   If you are using Hosted WebLogic Remote Console, you will also need to expose the `rconsole/*` endpoint.
8. Optional: If you want WebLogic Remote Console to ignore warnings about expired, untrusted, or missing certificates when connecting to an Administration Server, enable the **Make Insecure Connection** checkbox. We recommend that you only enable this setting for development or demonstration environments.

9. Optional: If this WebLogic Server domain resides in a different network, you can still facilitate communication between it and WebLogic Remote Console. In the **Proxy Override** field, enter a proxy server address.

   You can also set a proxy address that applies to all Administration Server connections. See Connect using a Proxy Server.

10. Click **OK**.

After you connect successfully, you can begin managing your WebLogic Server domain. To understand how WebLogic Remote Console presents a domain structure, see Perspectives in the Administration Server Provider.

You can view and update the connection details for an Administration Server provider in the **Providers** drawer.

Connect using SSL/TLS

If you want to connect to a WebLogic Server domain using SSL/TLS, you may need to perform some additional configuration steps in WebLogic Remote Console.

If you specified HTTPS for the domain URL when you connected to the Administration Server, then WebLogic Remote Console uses SSL/TLS to communicate with the domain.

The SSL/TLS connection requires trust in the WebLogic Server domain, where the trust configuration is handled by the underlying JDK JSSE support. By default, the JDK uses the `cacerts` truststore provided with the JDK. If the domain requires additional trust, separate trust, or is using the WebLogic Server demo trust, then you'll need to configure SSL/TLS trust.

You can either use WebLogic Remote Console to specify the type and location of the trust store (as described below), or use the `keytool` utility to import the required trust certificates into the `cacerts` truststore supplied with the JDK. See The keytool Command in *Java Development Kit Version 17 Tool Specifications*.

1. In WebLogic Remote Console, go to **File**, then **Settings**. (On macOS, go to **WebLogic Remote Console**, then **Settings**.)
2. Under the **Networking** section, in the **Trust Store Type** field, enter the algorithm name for your trust store. Depending on the Trust Store Type that you provide, additional fields may appear.

   See JDK Providers Documentation in *Java Security Developer's Guide* for specific algorithm names.
3. Click **Choose a trust store file** to browse to the file location of your trust store.
4. Click **Change** beside **Trust Store Key**, then enter the secret for your trust store key.
5. Click **Save** to add the secret.
6. Click **Save** to apply your changes.

Configure Web Authentication

You can delegate authentication of users from WebLogic Remote Console to an external authentication service.

By default, WebLogic Remote Console uses the Basic HTTP authentication scheme to authenticate users. If you want to replace Basic authentication with another authentication scheme, perhaps to support a single sign-on flow, you can enable the **Use Web Authentication** option to send users through an alternative login process, using the browser.

> **Note:**
>
> This functionality is only available on domains running WebLogic Server 14.1.2.0.0 or later.

When web authentication is enabled, the default Basic authentication HTTP header is replaced by using a WebLogic Server token for REST communications. Users are sent to a login endpoint that is generated by taking the domain URL of the connected Administration Server and then adding the Remote Console Helper Context Path (the default is `console`), and then `login`. For example, the full URL might look like: `https://`*`administrationServer`*`:7002/console/login`.

In the URL, notice that the protocol is `https` and the port number is `7002` (the default port number for the SSL/TLS listen port). To use web authentication, you must configure SSL/TLS.

Web authentication in WebLogic Remote Console is facilitated by the WebLogic Remote Console Helper, an application included with WebLogic Server. If necessary, you can customize settings for this application to fit your web authentication process.

For user accounts that are authenticated through the embedded LDAP server (WebLogic Authentication provider) or another supported LDAP or RDBMS authentication provider, you do not need to perform any further configuration.

However, if you want to authenticate virtual users, you must make these users known to WebLogic Server by adding them to an LDAP or RDBMS authentication provider as the REST communication from WebLogic Remote Console does not directly support the use of virtual users.

The general process for setting up web authentication is as follows:

1. Configure SSL/TLS for your domain. See Set Up SSL/TLS.
2. Configure your authentication provider.

   In *Administering Security for Oracle WebLogic Server*, see:

   - About Configuring the Authentication Providers in WebLogic Server
   - Configuring the WebLogic Authentication Provider
   - Configuring LDAP Authentication Providers
   - Configuring RDBMS Authentication Providers
3. Ensure that both the login and management endpoints (`console/*` and `management/*`, respectively) of your Administration Server are accessible by WebLogic Remote Console. If your Administration Server's endpoints are behind a firewall, load balancer, or otherwise externally unavailable, you will need to expose those endpoints manually.

> **Note:**
>
> If you want to customize the login endpoint, update the **Remote Console Helper Context Path** attribute (see step 4) with a new endpoint and restart WebLogic Server. Be aware that this only replaces `console`; you cannot change the `login` segment.
>
> Then, go to **File**, then **Settings**. (On macOS, go to **WebLogic Remote Console**, then **Settings**.) Under the **Other Java System Properties** section, click **+** to add a new row and enter `console.ssoDomainLoginUri=/`*`newEndpoint`*`/login`.
>
> If you modify the Remote Console Helper Context Path, it can prevent WebLogic Remote Console from successfully connecting to the Administration Server. Do not change it unless you understand how changing the login endpoint will affect access to your environment.

4. Optional: Customize the Remote Console Helper.
   a. In the **Edit Tree**, go to **Environment**, then **Domain**.
   b. Click **Show Advanced Fields**.
   c. Edit the following Remote Console Helper attributes as needed:
      - Remote Console Helper Enabled
      - Remote Console Helper Context Path
      - Remote Console Helper Cookie Name
      - Remote Console Helper Session Timeout
      - Remote Console Helper Protected Cookie Enabled
      - Remote Console Helper Token Timeout
   d. Click **Save**.
5. Optional: If you want to support the authentication of virtual users, then you must add each virtual user to the default authentication provider (WebLogic Authentication provider) or another configured LDAP or RDBMS provider. The REST communication from WebLogic Remote Console does not directly support the use of virtual users.

   Make sure to add the users as members of a group with permissions to accomplish their responsibilities, such as Administrators, Operators, or so on.

   If you add the virtual user to the WebLogic Authentication provider, then you can use WebLogic Remote Console to create the user as described in Create a User. Otherwise, to add a virtual user to the provider, you'll need to use an external tool specific to the provider.
6. When connecting to the Administration Server using WebLogic Remote Console:
   a. Enable the **Use Web Authentication** option.
   b. Ensure the Administration Server **URL** is using `https` and the SSL/TLS listen port (default is 7002). If the administration port is enabled, the default port value is 9002 instead.

      For example:
      - `https://`*`administrationServer`*`:7002`
      - `https://`*`administrationServer`*`:9002`

Enable SAML 2.0 SSO for WebLogic Remote Console

If you configure WebLogic Server as a SAML 2.0 Service Provider site, you can use it to implement single sign-on (SSO) functionality for WebLogic Remote Console.

For an overview of the steps required to set up web authentication in WebLogic Remote Console, see Configure Web Authentication.

Note:

As this functionality relies on web authentication, it is only supported on domains running WebLogic Server 14.1.2.0.0 or later.

1. Configure WebLogic Server as a SAML 2.0 Service Provider site as described in Configuring a Service Provider Site for SAML 2.0 Single Sign-On in *Administering Security for Oracle WebLogic Server*.

   Make sure to:

   - Add `/console/login` to the list of Identity Provider partner redirect URIs. See Configure Redirect URIs in *Administering Security for Oracle WebLogic Server*.
   - Secure the published site URL by using `https` and the SSL/TLS listen port. For example, `https://wls.example.com:7002/saml2`. See About SAML 2.0 General Services in *Administering Security for Oracle WebLogic Server*.

2. Optional: If you want to support the authentication of virtual users, then you need to configure an LDAP or RDBMS authentication provider and add each virtual user to that provider. The REST communication from WebLogic Remote Console does not directly support the use of virtual users.

   If you add the virtual user to the WebLogic Authentication provider, then you can use WebLogic Remote Console to create the user as described in Create a User. Otherwise, to add a virtual user to the provider, you'll need to use an external tool specific to the provider.

   Make sure to add the users as members of the Administrators group.

3. Update the Remote Console Helper attributes to support SAML 2.0 SSO.

   a. In WebLogic Remote Console, in the **Edit Tree**, go to **Environment**, then **Domain**.

   b. Click **Show Advanced Fields**.

   c. Update the values for the following Remote Console Helper attributes as specified:

      - **Remote Console Helper Cookie Name**: `JSESSIONID`
      - **Remote Console Helper Protected Cookie Enabled**: `false`

   d. Click **Save**.

Connect using a Proxy Server

You may need to configure settings for a proxy server to facilitate communication between WebLogic Remote Console and a WebLogic Server domain that resides in a different network.

You can configure a global proxy server that applies to all Administration Server connections or you can assign proxy server settings individually to each Administration Server connection.

You can also configure a combination of global and individual settings; the individual proxy server settings will override the global proxy server settings.

- To apply a global proxy server that affects all Administration Server connections:
  1. Go to **File**, then **Settings**. (On macOS, go to **WebLogic Remote Console**, then **Settings**.)
  2. Under the **Networking** section, in the **Proxy Address** field, enter the address of the proxy server, including both the host name and port number.
  3. Click **Save** to apply your changes.

  Although it is possible to create multiple global proxy servers using Java system properties, we do not recommend it. Configuring multiple global proxy servers can lead to unexpected and unwanted behavior. This applies even if the proxy servers use different protocols: if you use Java system properties to add a proxy server that uses HTTPS and then another that uses SOCKS, WebLogic Remote Console will ignore the SOCKS proxy server.

  The proxy server value in the **Settings** dialog box takes precedence over global proxy server settings that were set using Java system properties.
- To apply proxy server settings to a single Administration Server connection:
  1. Open the **Providers** drawer.
  2. Beside the Administration Server provider where you want to configure a proxy server, click **Settings**.
  3. In the **Proxy Override** field, enter the address of the proxy server, including both the host name and port number.

     Note:

     If you want to bypass a global proxy server and make a direct connection, enter `DIRECT`.
  4. Click **OK**.

Change Network Timeout Settings

You can change the default settings for the connection and read timeout limits used with a WebLogic Server domain from WebLogic Remote Console.

1. Go to **File**, then **Settings**. (On macOS, go to **WebLogic Remote Console**, then **Settings**.)
2. Under the **Networking** section, in the **Administration Server Connection Timeout** field, specify an interval (in milliseconds) to determine how long the WebLogic Remote Console should wait for a successful connection to a domain. The default is 10 seconds (10 000 milliseconds).
3. In the **Administration Server Read Timeout** field, specify an interval (in milliseconds) to determine how long the WebLogic Remote Console should wait for a response from the server. The default is 20 seconds (20 000 milliseconds).
4. Click **Save** to apply your changes.

When changing network timeout settings, the primary impact will be to the response time for Console threads, while the application will show no data when a timeout occurs. Timeouts are more likely to occur during requests where WebLogic Server experiences longer initialization or execution times, such as during runtime monitoring actions of servers.

Disable Host Name Verification in Connections to the Domain

When using WebLogic demo trust to connect to a domain, it may be necessary to disable host name verification.

When host name verification is disabled, WebLogic Remote Console does not verify that the host name in the URL to which a connection is made matches the host name in the digital certificate that the server sends back as part of the SSL connection.

1. Go to **File**, then **Settings**. (On macOS, go to **WebLogic Remote Console**, then **Settings**.)
2. Under the **Networking** section, under **Disable Host Name Verification**, select **Yes** to disable host name verification or **No** to enable host name verification.
3. Click **Save** to apply your changes.

Using Java System Properties

WebLogic Remote Console supports the use of Java system properties to customize the console if a specific setting is not available.

Note:

If the equivalent setting already exists in the **Settings** dialog box, we recommend using that configuration option instead of a Java system property. For example, use the **Proxy Address** option under **Networking** rather than `https.proxyHost` and `https.proxyPort`.

To add a Java system property to WebLogic Remote Console:

1. Go to **File**, then **Settings**. (On macOS, go to **WebLogic Remote Console**, then **Settings**.)
2. Under the **Other Java System Properties** section, click **+** to add a new row and enter the Java system property as a name-value pair, separated by `=`. For example, `server.port=8092`.

To delete a property, select the row and click **-**.

Examples

Usage	Syntax
To set the `SameSite` cookie attribute if required for web browser support. When WebLogic Remote Console establishes a connection with the WebLogic Server domain, a HTTP cookie is established with the Web Browser session. For security reasons, the `SameSite` attribute of the HTTP cookie may need to be set for the Web Browser to accept the HTTP session cookie.	Set both properties: • `console.enableSameSiteCookieValue=true` • `console.valueSameSiteCookie=Lax|Strict` The default is `Lax`.

Usage	Syntax
To specify an alternative location for the JDK.	`javaPath=`*`pathToJDK`*
To specify a custom logging configuration file so you can control the logging information that WebLogic Remote Console collects. The custom logging configuration file must follow the Java format for configuration files. You can see an example of a Java logging configuration file at `$JAVA_HOME/conf/logging.properties`. If a problem occurs with your custom logging configuration file, WebLogic Remote Console will fallback to use its default logging configuration file. `STDOUT` includes a log message indicating which file was used.	`java.util.logging.config.file=<path-to-logging.properties>`

Configuring a WebLogic Server Domain

After you connect to an Administration Server with WebLogic Remote Console, you are ready to make configuration changes to your domain.

For a comprehensive explanation of domain configuration in WebLogic Server, see Understanding Oracle WebLogic Server Domains in *Understanding Domain Configuration for Oracle WebLogic Server*.

In WebLogic Remote Console, it is a generally two step process to apply configuration changes to your domain.

Step One: Edit

When you make changes to your domain, they are saved in a Pending state. They are not active in the domain but you can make changes in other areas without losing them and even if you log out of WebLogic Remote Console, the domain will retain those changes.

You can see your pending changes in the Shopping Cart. In the Shopping Cart, click **View Changes**. If you cannot see specific changes in the Shopping Cart, Install the WebLogic Remote Console Extension.

WebLogic Remote Console protects your edits from being overwritten by other users by enforcing a configuration lock that prevents other users from making changes to the domain during your editing session. You will retain this lock until you commit (or discard) your changes.

Note:

The configuration lock only prevents conflicts from *other* users. If you use the same user account to log in to another WebLogic Server system administration tool such as the WebLogic Scripting Tool (WLST), then WebLogic Server considers both sessions as the same edit session.

Avoid using making changes using multiple tools simultaneously. When one session commits its changes, it releases the configuration lock and discards changes from the other session.

Step Two: Commit

When you are satisfied with your changes, you must commit them for the changes to apply to the domain. Open the Shopping Cart and then click **Commit Changes**.

Some configuration changes apply to the domain immediately - these are called dynamic changes. Other configuration changes require a server restart to take effect and are called non-dynamic changes. A server restart required icon ⓘ appears beside any attribute that is non-dynamic.

> **Note:**
>
> If your set of changes contains both dynamic and non-dynamic changes, then none of the changes will take effect until after a server restart. This ensures that the configuration changes are not partially, and potentially imperfectly, implemented.

WebLogic Remote Console releases the configuration lock after your changes are committed.

You may not need to restart all servers to apply non-dynamic changes. In the **Monitoring Tree**, go to **Environment**, then **Servers** to see which servers require a restart.

> **Note:**
>
> Any actions are performed within the Monitoring Tree or Security Data tree perspectives do not require a commit step. They are active immediately after the changes are saved.

Back Up Configuration Files

You can set WebLogic Server to save a domain's existing configuration state before pending changes are committed. If you ever need to reverse a change, then WebLogic Server has the previous set of configuration files (including the central configuration file, `config.xml`) available as a back up.

1. In the **Edit Tree**, go to **Environment**, then **Domain**.
2. Click **Show Advanced Fields**.
3. Turn on the **Configuration Archive Enabled** option.
4. In the **Archive Configuration Count** field, enter the number of archive files to retain.

   When the maximum number of archive files is reached, older archive files will be discarded.
5. Click **Save**.

Backups of the previous configuration files are saved in the *`DOMAIN_HOME`*`/configArchive` directory, in JAR files named `config-1.jar`, `config-2.jar`, and so on. The archived files are rotated so that the newest file has a suffix with the highest number. For more information, see configArchive in *Understanding Domain Configuration for Oracle WebLogic Server*.

Perspectives in the Administration Server Provider

The Administration Server provider is split into multiple perspectives, each of which focuses on a different management area for a WebLogic Server domain.

- **Edit Tree**: an editable view of the WebLogic Server domain. Use this perspective when you want to make changes and commit them to the domain.
- **Configuration View Tree**: a read-only view of the WebLogic Server domain. Enter this perspective if you want to see the current settings of the domain, without making any changes.
- **Monitoring Tree**: an overview of the runtime statistics for the running domain. You can view statistics per server or aggregated across all servers in the domain. For example, compare applications running on Server1 vs. applications running on one or more servers. The Monitoring Tree also provides some control operations such as starting and stopping servers or applications.
- **Security Data Tree**: an editable view of the security providers' data in the security realm. This includes users, groups, policies, and so on.

Note:

Although the Edit Tree and the Configuration View Tree perspectives look similar, they are generated from two separate collections of configuration MBeans, which results in important and distinct nuances between the two perspectives:

1. Changes made in the Edit Tree perspective do not appear in the Configuration View Tree perspective until you commit them, or, for non-dynamic changes, until you restart the server.
2. Certain dynamically computed domain contents do not appear in the Edit Tree. For example,
   - Dynamic clusters (and their servers)
   - Situational configurations
   - Changes made from the command line using system properties
   - Changes to some production mode-related settings

For more information on the differences between editable Configuration MBeans and read-only Configuration MBeans, see Managing Configuration Changes in *Understanding Domain Configuration for Oracle WebLogic Server*.

Where do I?

The majority of domain configuration is performed within the **Edit Tree** perspective. However, there are some tasks that are performed in the Monitoring Tree or the Security Data Tree perspectives. These tasks do not require a commit step and become active immediately after your changes are saved.

Note:

The Configuration View Tree perspective is read-only and no management tasks are performed within it. Therefore, it's not included in this table.

Table 5-1 Tasks in Each Perspective

Task	Edit Tree	Monitoring Tree	Security Data Tree
Start or stop a server			
Deploy an application			
Start or stop an application			
Configure servers, clusters, and machines			
Manage access control			
Manage users, groups, roles, policies			
Configure security providers			
Configure database connectivity			
Configure messaging			
Diagnose domain issues			
Configure SSL/TLS			
Create dashboards			

Access Limitations

To prevent unauthorized changes to your WebLogic Server domains, WebLogic Remote Console limits its functionality based on a user's role.

If you log in as a user whose role has limited permissions, WebLogic Remote Console automatically adjusts its user interface to conceal the areas to which you do not have access. Users with the administrator role have full access to WebLogic Remote Console.

If you want to see the full user interface of WebLogic Remote Console, regardless of your current role, then open **File** then **Settings**. (On macOS, go to WebLogic Remote Console, then **Settings**.) Under the **Role Checking** section, set **Restrict Content Based On Role** to **No** and click **Save**. A user with limited permissions can now *see* everything but they are still blocked from performing any actions beyond the scope of their role.

Note:

These restrictions are based on the default security policies assigned to each role. If you customize the policies to add or remove access beyond the default policies, those changed permissions will not be reflected in WebLogic Remote Console. It will continue to hide or show functionality based on the default security policies.

For more information, see Users, Groups, And Security Roles in *Securing Resources Using Roles and Policies for Oracle WebLogic Server*.

Table 5-2 provides some examples of the limitations that users in non-Administrator roles might encounter.

Table 5-2 Access Limitations based on Role

Role	Limitations
Deployer	User can view but not edit server or domain configurations. They can also modify areas related to application deployment including some JDBC and JMS resources.
Monitor	User access to WebLogic Remote Console is entirely read-only.
Operator	User can start and stop servers but cannot see the Edit Tree perspective.

Configure Managed Servers

Managed Servers are subordinate servers that are managed indirectly through the domain's Administration Server.

In a typical production environment, you should create one or more Managed Servers in the domain to host business applications and only use the Administration Server to configure and monitor the Managed Servers.

You can configure Managed Servers as standalone instances or organize them into clusters. See Managed Servers and Managed Server Clusters in *Understanding Oracle WebLogic Server*.

Create a Managed Server

1. In the **Edit Tree**, go to **Environment**, then **Servers**.
2. Click **New**.
3. Enter a name for the new server.

   See Domain and Server Name Restrictions in *Understanding Domain Configuration for Oracle WebLogic Server*.
4. Optional: You can copy the settings from an existing server onto your new server. Select a server from the **Copy settings from another server** drop-down list.

   Only the server's settings will be copied. Children, such as channels, are not copied. Any settings that are not supported by the WebLogic Server REST API are also not copied.
5. Click **Create**.

This process creates a standalone Managed Server. If you want to add the Managed Server to a cluster, follow the instructions in Assign a Managed Server to a Cluster.

Start a Managed Server

Start a Managed Server from WebLogic Remote Console.

1. Configure Node Manager for use with Managed Servers.

   If you have already configured Node Manager, you can skip to the next step.

   a. Choose a Node Manager implementation. See Node Manager Implementations in *Administering Node Manager for Oracle WebLogic Server*.

   b. Configure your Node Manager implementation. See either Configuring Java Node Manager or Configuring Script-Based Node Manager in *Administering Node Manager for Oracle WebLogic Server*.

   c. Start Node Manager on the computer that you want to host the Managed Server.

2. Configure the Managed Server to communicate with Node Manager.

   If you have already configured communication between your Managed Servers and the Node Manager, you can skip to the next step.

   a. Configure Machines.
   b. Assign a Server Instance to a Machine.
   c. Configure Startup Arguments for a Managed Server.

3. Optional: Change the startup mode for the Managed Server. The default is `RUNNING`. See Specify a Startup Mode.

4. Start Node Manager on the computer that you want to host the Managed Server.

   The WebLogic Server custom installation process optionally installs and starts Node Manager as a Windows service on Windows systems. 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 *Administering Node Manager for Oracle WebLogic Server*.

5. In the **Monitoring Tree**, go to **Environment**, then **Servers**.

6. Select the server you want to start, then click **Start**.

Node Manager starts the server on the target machine. When the Node Manager finishes its start sequence, the server's state is indicated in the **State** column.

Configure Startup Arguments for a Managed Server

In most environments, Node Manager can start a server without requiring you to specify startup options. However, if you have modified your environment, such as adding classes to the WebLogic Server class path, you must specify startup options before you can use WebLogic Remote Console to start a server.

1. In the **Edit Tree**, go to **Environment**, then **Servers**.

2. Select the Managed Server that you want to configure startup arguments for.

3. On the **Advanced** tab, select the **Node Manager** subtab.

4. If you want the server instance to run under a different WebLogic Server user account, enter the username and password of an existing user in the **User Name** and **Password** fields. The user must have a role with permission to start servers.

   The existing username and password are the values that you supplied when you used WebLogic Remote Console or the Configuration Wizard to create the server.

5. Update any other fields where you want to override the default values provided by the Node Manager.

   WebLogic Remote Console *replaces* the Node Manager default values, it does not append the new values to the default values. For more information, see Reviewing nodemanager.properties in *Administering Node Manager for Oracle WebLogic Server*.

   All paths refer to paths on the Node Manager machine.

   - If you provide values for the **Class Path** field, make sure that you provide the full class path required to start the Managed Server.
   - If you want to add **Arguments**, make sure you prepend `-D` before each argument, like so `-Dweblogic.management.startupMode=`*`MODE`* .

     See weblogic.Server Command-Line Reference in *Command Reference for Oracle WebLogic Server* for information on the Java options that set runtime behavior of a WebLogic Server instance and Configuring Node Manager to Use Start and Stop

Scripts in *Administering Node Manager for Oracle WebLogic Server* for information on how `JAVA_OPTIONS` are combined and how duplicate values are handled.

6. Click **Save**.
7. Repeat for every applicable Managed Server.

Specify a Startup Mode

The startup mode specifies the state in which a server instance should be started. The default is to start in the `RUNNING` state.

1. In the **Edit Tree**, go to **Environment**, then **Servers**, then *myServer*.
2. Under the **General** tab, click **Show Advanced Fields**.
3. In the **Startup Mode** field, enter one of the following startup modes (in uppercase as shown):
   - `RUNNING` (default): a server offers its services to clients and can operate as a full member of a cluster.
   - `ADMIN`: the server is up and running, but available only for administration operations, allowing you to perform server and application-level administration tasks without risk to running applications.
   - `STANDBY`: the server listens for administrative requests only on the domain-wide administration port and only accepts life cycle commands that transition the server instance to either the `RUNNING` or `SHUTDOWN` state. Other administration requests are not accepted. If you specify `STANDBY`, you must also enable the domain-wide administration port. See Configure the Domain-Wide Administration Port.

   See Understanding Server Life Cycle in *Administering Server Startup and Shutdown for Oracle WebLogic Server*.
4. Click **Save**.

Configure Clusters

A WebLogic Server cluster consists of multiple WebLogic Server server instances running simultaneously and working together to provide increased scalability and reliability.

A cluster appears to clients as a single WebLogic Server instance. The server instances that constitute a cluster can run on the same machine or be located on different machines. You can increase a cluster’s capacity by adding additional server instances to the cluster on an existing machine or you can add machines to the cluster to host the incremental server instances. Each server instance in a cluster must run the same version of WebLogic Server. See Understanding WebLogic Server Clustering in *Administering Clusters for Oracle WebLogic Server*.

You can also create dynamic clusters which consist of server instances that can be dynamically scaled up or down to meet the resource needs of your applications. A dynamic cluster uses a single server template to define configuration for a specified number of generated (dynamic) server instances. For more information, see Dynamic Clusters in *Administering Clusters for Oracle WebLogic Server*.

Create a Cluster

Create a WebLogic Server cluster.

1. In the **Edit Tree**, go to **Environment**, then **Clusters**.

2. Click **New**.
3. Enter a name for the cluster.
4. Click **Create**.

Assign a Managed Server to a Cluster

You can add an existing Managed Server to clusters.

1. If you haven't already, create a cluster. See Create a Cluster.
2. Shut down the Managed Server. You cannot change the cluster of a running server.
3. In the **Edit Tree**, go to **Environment**, then **Servers**.
4. Select the Managed Server you want to add to a cluster.
5. From the **Cluster** drop-down list, select the cluster where you want to assign this Managed Server.
6. Click **Save** and commit your changes.
7. Start the Managed Server.

Create a Dynamic Cluster

Create a cluster that dynamically scales depending on the resource needs of your applications.

1. If you haven't already, create a cluster and a server template. See Create a Cluster and Create a Server Template.
2. Go to the **Environment**, then **Clusters**, then *myCluster* and select the **Dynamic** tab.
3. Select a server template from the **Server Template** drop-down list.

   If you do not have a server template or if you want to create a new one, click **More** ⋮ beside the **Server Template** drop-down list and select **Create New Server Template**. Enter a unique name for the new server template.
4. In the **Dynamic Cluster Size** field, enter the maximum number of running dynamic server instances allowed for scale up operations in this dynamic cluster. This number is the sum of the Dynamic Cluster Size and the additional number of dynamic server instances allowed for scale up operations.
5. In the **Server Name Prefix**, specify the naming convention you want to use for the dynamic servers in your cluster.
6. Optional: If you want to specify how the dynamic servers in your cluster are distributed across machines, turn on **Enable Calculated Machine Associations** and then configure **Machine Name Match Expression**.

   Machine distribution is determined by the pattern set in **Machine Name Match Expression**. If a machine in the domain matches the expression, then it will be included in the set of machines used by these dynamic servers. The expression is a comma-separated set of values that specifies the machines to match. The value can either match a machine name exactly or, you can use a trailing `*` to match multiple machine names. If you do not enter a pattern, then all machines in the domain are available to the dynamic servers.
7. Optional: If you want to specify whether listen ports are calculated, turn on **Enable Calculated Listen Ports**.
8. Click **Save**.

Consider configuring elasticity on the cluster to control how the cluster scales up or down. See Configuring Dynamic Clusters in *Configuring Elasticity in Dynamic Clusters for Oracle WebLogic Server* for more information.

Remove a Server from a Cluster

You can remove a Managed Server from a cluster without deleting the server instance from the domain.

1. In the **Edit Tree**, go to **Environment**, then **Servers**.
2. Select the Managed Server you want to remove from a cluster to make it a standalone server instance.
3. From the **Cluster** drop-down list, select **None**.
4. Click **Save**.

Define a Singleton Service

A singleton service is a service running on a Managed Server that is available on only one member of a cluster at a time. WebLogic Server lets you automatically monitor and migrate singleton services from one server to another.

> **Note:**
>
> Singleton services can only be migrated automatically within a cluster. Ensure that you have created and configured a cluster and its member server instances.

1. In the **Edit Tree**, go to **Environment**, then **Singleton Services**.
2. Click **New**.
3. Enter a name for the singleton service.
4. Click **Create**.
5. From the **Cluster** drop-down list, select the cluster in which you want to apply this singleton service.
6. In the **Class Name** field, define the fully qualified name of the class. The class must also be contained in the server class path.

   The class specified in **Class Name** must implement the singleton service interface to function as a migratable singleton service. See Service Migration in *Administering Clusters for Oracle WebLogic Server*.
7. In the **Additional Migration Attempts** field, enter the number of additional attempts that should be tried to reach a migratable service after the service has failed on every possible configured server instance at least once.
8. In the **Sleep Time Between Attempts** field, enter the interval between migration attempts.
9. Click **Save**.
10. Optional: If you want to specify the migration behavior of this target, go to the **Migration** tab.
    a. From the **User Preferred Server** drop-down list, select the preferred server instance for this singleton service.

   b. In the **Constrained Candidate Servers** field, choose which server instances in the cluster to use as a backup for services on this singleton service. Move selected server instances from **Available** to **Chosen**.

11. Click **Save**.

By default, the singleton service iterates through all servers in the cluster to determine the list of candidate servers for migration.

Create a Server Template

Server templates define non-default settings and attributes that you can apply to a set of server instances.

When you create a server template and then apply it to Managed Servers, you can specify configuration attributes once and then propagate the new attribute settings to server instances without manually configuring each one. When you need to update an attribute, you can simply change the value in the server template, and the new value takes effect in all of the server instances that use that server template. You can also add a server template to a cluster, then all of the servers within the cluster will inherit the server template. See Server Templates in *Understanding Domain Configuration for Oracle WebLogic Server*.

If you need to define any server-specific attributes, you can easily override the server template at the individual server level.

1. In the **Edit Tree**, go to **Environment**, then **Server Templates**.
2. Click **New**.
3. Enter a unique name for the server template.
4. Click **Create**

After you create a server template, you can configure its attributes as you would configure a traditional Managed Server.

Apply a Server Template

Server templates define common configuration attributes for a set of server instances in one centralized location. When you apply a server template to a Managed Server or a cluster, the server instances inherit the configurations from the server template.

- To apply a server template to a standalone Managed Server:
  1. In the **Edit Tree**, go to **Environment**, then **Servers**.
  2. Select the Managed Server where you would like to apply the server template.
  3. From the **Template** drop-down list, select a template. You can also click **More** ⋮ to create and apply a new server template.
  4. Click **Save**.
- To apply a server template to a cluster:
  1. In the **Edit Tree**, go to **Environment**, then **Clusters**.
  2. Select the cluster where you would like to apply the server template.
  3. From the **Template** drop-down list, select a template. You can also click **More** ⋮ to create and apply a new server template.
  4. Click **Save**.

Configure Machines

A machine is the logical representation of the computer that hosts one or more WebLogic Server instances. Each Managed Server must be assigned to a machine.

As part of the machine configuration process, you'll also configure each machine in your domain to communicate with Node Manager, a program used to control WebLogic Server instances. A single Node Manager instance is used to control all of the server instances running on the same physical machine. These instances can reside in different clusters, domains, and so on. See About Node Manager in *Administering Node Manager for Oracle WebLogic Server*.

1. In the **Edit Tree**, navigate to **Environment**, then **Machines**.
2. Click **New**.
3. Enter a name for the new machine.
4. Optional: If you are creating a machine that will run on a UNIX platform, select **UNIX Machine** from the **Type** drop-down list.
5. Click **Create**.
6. Configure the settings of your new machine under **Environment**, then **Machines**, then *myMachine*.
7. On the **Node Manager** tab, configure the following properties:
   a. in the **Type** drop-down list, select the Node Manager type.
   b. In the **Listen Address** field, enter the DNS name or IP address upon which Node Manager listens for incoming requests.

      If you identify the Listen Address by IP address, you must disable Host Name Verification on Administration Servers that will access Node Manager. For more information and instructions, see Using Host Name Verification in *Administering Security for Oracle WebLogic Server* and Enable Host Name Verification.
   c. In the **Listen Port** field, enter the port value where Node Manager listens for incoming requests.
   d. Optional: If you set **Type** to `SSH` or `RSH`, you should also specify values in the **Node Manager Home** and **Shell Commands** fields.
   e. Turn on **Debug Enabled** if you want to enable Node Manager debugging.
   f. Click **Save**.

The new machine entry now specifies the attributes required to connect to the Node Manager process running on the machine, as well as identify which WebLogic Server instances reside on the machine.

Next, see Assign a Server Instance to a Machine.

Assign a Server Instance to a Machine

As part of the process for setting up Node Manager to administer Managed Servers, you must assign a server instance to a machine.

> **Note:**
>
> You cannot change the machine of a server that's running, therefore you cannot use WebLogic Remote Console to change the machine of the Administration Server. Use an offline tool, such as WLST Offline instead.

1. Shut down the Managed Server. You cannot change the machine of a running server.
2. In the **Edit Tree**, go to **Environment**, then **Servers**.
3. Select the Managed Server that you want to assign to a machine.
4. In the **Machine** drop-down list, select the machine to which you want to assign this server.
5. Click **Save**.

Configure a Virtual Host

A virtual host is a set of host names to which servers or clusters respond. When you use virtual hosting, you use DNS to specify one or more host names that map to the IP address of a server or cluster. You also specify which web applications are served by each virtual host.

1. In the **Edit Tree**, go to **Environment**, then **Virtual Hosts**.
2. Click **New**.
3. Enter a name for the virtual host.
4. Click **Create**.
5. In the **Virtual Host Names** field, enter the host names for which this virtual host will serve requests. Use line breaks to separate host names.
6. In the **Network Access Point Name** field, enter the dedicated server channel name (NetworkAccessPoint) for which this virtual host serves HTTP requests.
7. Click **Save**.
8. On the **HTTP** tab, configure HTTP attributes for the virtual host as needed. Click **Save**.
9. On the **Logging** tab, configure the default log file settings for the virtual host as needed. Click **Save**.
10. On the **Targets** tab, select the servers or clusters where you want to deploy this virtual host.
11. Click **Save**.

Use Custom Classes to Configure Servers

You can create custom Java classes that extend server features or that perform some task when a server instance starts or shuts down.

These startup classes or shutdown classes are loaded by the system class loader and therefore are available to all resources on a server instance even if the resources are managed by different containers. For example, EJBs and JMX clients can access startup or shutdown classes even though their containers use their own, higher level class loaders.

These are system-level classes so you must place them on the server's class path. If you want to make custom classes available to multiple applications but do not need the classes to be available at the system level, consider deploying them as application startup classes.

For more information, see Configuring Server Level Startup and Shutdown Classes in *Administering Server Startup and Shutdown for Oracle WebLogic Server*.

Configure Startup Classes

You can create custom Java classes that extend server features or that perform some task when a server instance starts.

1. In the **Edit Tree**, go to **Environment**, then **Startup Classes**.
2. Click **New**.
3. Enter a name for the startup class and click **Create**.
4. On the configuration page for your new startup class, in the **Class Name** field, enter the fully qualified name of the startup class.

   For example: `mycompany.myclasses.startupclass`.
5. Modify other startup class attributes as necessary.
6. Click **Save**.
7. On the **Targets** tab, move the server instances or clusters where you want to deploy this startup class over to **Chosen**.
8. Click **Save**.

You must add the startup class to the class path of each server where it is deployed.

9. If you use a script to start server instances, perform these steps.
   a. Open the start script in a text editor.
   b. In the command that sets the class path, add the path of the directory that contains your class root package and save the script.

      For example, if you create a startup class named `StartBrowser` in a package named `com.mycompany.startup`, and you archive the class file in a JAR file named `/myDomain/src/myJAR.jar`, then the start script for your server must add `/myDomain/src/myJAR.jar` to the server's class path.
   c. Restart the server.
10. If you use Node Manager to start server instances, perform the following steps on every server that will run the startup class.
    a. Go to **Environment**, then **Servers**, then select the server you want to modify.
    b. On the **Advanced** tab, select the **Node Manager** sub-tab.
    c. In the **Class Path** field, add the pathname for your class or for a JAR file that contains your class.

       For example, if you create a startup class named `StartBrowser` in a package named `com.mycompany.startup`, and you archive the class file in a JAR file named `/`

`myDomain/src/myJAR.jar`, then the Class Path field should contain this value: `/myDomain/src/myJAR.jar` to the server's class path.

> **Note:**
>
> Ensure that all of the classes that WebLogic Server requires are also on the class path. Use absolute paths or paths relative to the Node Manager's home directory. Separate multiple classes with either `:` (BASH) or `;` (Windows).
>
> For example, `/Oracle/Middleware/wlserver/server/lib/weblogicsp.jar:/Oracle/Middleware/wlserver/server/lib/weblogic.jar:/myDomain/src/myJAR.jar`

d. Click **Save**.

e. Repeat on any additional servers.

Configure Shutdown Classes

You can create custom Java classes that extend server features or that perform some task when a server instance shuts down.

1. In the **Edit Tree**, go to **Environment**, then **Shutdown Classes**.
2. Click **New**.
3. Enter a name for the shutdown class and click **Create**.
4. On the configuration page for your new shutdown class, in the **Class Name** field, enter the fully qualified name of the shutdown class.

   For example: `mycompany.myclasses.shutdownclass`.
5. Modify other shutdown class attributes as necessary.
6. Click **Save**.
7. On the **Targets** tab, move the server instances or clusters where you want to deploy this shutdown class over to **Chosen**.
8. Click **Save**.

You must add the shutdown class to the class path of each server where it is deployed.

9. If you use a script to stop server instances, perform these steps.
   a. Open the stop script in a text editor.
   b. In the command that sets the class path, add the path of the directory that contains your class root package and save the script.

      For example, if you create a shutdown class named `StopBrowser` in a package named `com.mycompany.shutdown`, and you archive the class file in a JAR file named `/myDomain/src/myJAR.jar`, then the stop script for your server must add `/myDomain/src/myJAR.jar` to the server's class path.
   c. Restart the server.
10. If you use Node Manager to shutdown server instances, perform the following steps on every server that will run the shutdown class.
    a. Go to **Environment**, then **Servers**, then select the server you want to modify.

b. On the **Advanced** tab, select the **Node Manager** sub-tab.

c. In the **Class Path** field, add the pathname for your class or for a JAR file that contains your class.

   For example, if you create a shutdown class named `StopBrowser` in a package named `com.mycompany.shutdown`, and you archive the class file in a JAR file named `/myDomain/src/myJAR.jar`, then the Class Path field should contain this value: `/myDomain/src/myJAR.jar` to the server's class path.

   > **Note:**
   >
   > Ensure that all of the classes that WebLogic Server requires are also on the class path. Use absolute paths or paths relative to the Node Manager's home directory. Separate multiple classes with either `:` (BASH) or `;` (Windows).
   >
   > For example, `/Oracle/Middleware/wlserver/server/lib/weblogicsp.jar:/Oracle/Middleware/wlserver/server/lib/weblogic.jar:/myDomain/src/myJAR.jar`

d. Click **Save**.

e. Repeat on any additional servers.

> **Note:**
>
> After you delete a shutdown class, it will run the first time you shut down the server. When you shut down the sever subsequently, it will no longer run.

Configure Network Connections

Configure the network resources for your domain.

As your domain increases in complexity, it is imperative to carefully manage the network resources and connections within your domain to ensure secure and reliable communication between servers and applications.

WebLogic Server uses network channels to organize and define the attributes of its network connections. Among other attributes, network channels can define:

- The protocol the connection supports
- The listen address
- The listen ports for secure and non-secure communication

See Understanding Network Channels in *Administering Server Environments for Oracle WebLogic Server*.

Each WebLogic Server instance provides default settings for the protocols, listen addresses, and listen ports through which it can be reached. These settings are referred to collectively as the default network channel. This default network channel provides two listen ports through which it receives requests: one for non-SSL/TLS requests and the other for SSL/TLS requests. You can disable one of these ports, but at least one must be enabled.

You can also configure your own custom network channels.

Configure the Domain-Wide Administration Port

An administration port restricts all administrative traffic between server instances in a WebLogic Server domain to a single port. Additionally, only secure, SSL/TLS traffic is accepted and all connections through the port require authentication by a server administrator.

Enabling the administration port imposes the following restrictions on your domain:

- The Administration Server and all Managed Servers in your domain must be configured with support for the SSL/TLS protocol.
- All servers in the domain, including the Administration Server, enable or disable the administration port at the same time.

See Configure an Administration Port for the Domain in *Securing a Production Environment for Oracle WebLogic Server*.

1. Shut down all Managed Servers in the domain. You cannot enable the administration port dynamically on a Managed Server.
2. Ensure that all servers in the domain are properly configured to use SSL/TLS. See Set Up SSL/TLS.
3. In the **Edit Tree**, go to **Environment**, then **Domain**.
4. Turn on the **Enable Administration Port** option.
5. In the **Administration Port** field, enter the SSL/TLS port number that server instances in the domain should use as the administration port.
6. Optional: If multiple server instances run on the same computer in a domain that uses a domain-wide administration port, then you must perform one of the following actions:
   - Host the server instances on a multi-homed machine and assign each server instance a unique listen address
   - Override the domain-wide administration port on all but one of the servers instances on the machine. On the **Environment**: **Servers**: *myServer* page for each Managed Server, enter a unique port value in the **Local Administration Port Override** field.
7. Click **Save** and then commit your changes.
8. Restart the Administration Server and start all the Managed Server instances in the domain.

Make sure the connection between the Managed Servers and the Administration Server uses the administration port.

Configure Custom Network Channels

A network channel is a configurable resource that defines the attributes of a network connection to WebLogic Server.

1. In the **Edit Tree**, go to **Environment**, then **Servers**, then *myServer*, then **Channels**.
2. Click **New**.
3. Enter a name for the new network channel.
4. Click **Create**.

   A node for the new network channel will appear under the **Channels** node.
5. Go to the new node to define the configuration for the new network channel.

At minimum, you should define the following information:

- Listen address
- Listen port
- External listen address
- External listen port

The external listen address and port are used to support Network Address Translation (NAT) firewalls. These should match the IP address or DNS name that clients use to access application on the server.

6. Click **Save**.

Specify a Listen Address

Define the listen address that a server uses for incoming connections.

1. In the **Edit Tree**, go to **Environment**, then **Servers**, then *myServer*.
2. In the **Listen Address** field, enter an IP address or DNS name for this server to use to listen for incoming connections.

   Note that the value you specify for the listen address is not the URL to the host machine and it does not include the communication protocol, listen port, or channel.

   Servers can be reached through the following URL: `*protocol*://*listen-address*:*listen-port*`
3. Click **Save**.

Specify Listen Ports

Define the listen ports for secure and non-secure communication.

> **Note:**
>
> You cannot disable both **Listen Port Enabled** or **SSL Listen Port Enabled**. At least one type of listen port must be active.

1. In the **Edit Tree**, go to **Environment**, then **Servers**, then *myServer*.
2. Enable **Listen Port Enabled** and enter a port number.
3. Enable **SSL Listen Port Enabled** and enter a port number.
4. Click **Save**.

If you plan to configure a domain-wide administration port, make sure the SSL Listen Port and the Administration Port are set to different port numbers.

Configure General Protocol Settings

You can configure each WebLogic Server instance to communicate over a variety of protocols, such as HTTP, T3, and IIOP. You can also configure a group of communication settings that apply to all protocols.

The general protocol settings apply to connections that use the server's default listen port and listen address. If you create network channels for this server, each channel can override these settings. For more information, see Configuring Network Resources in *Administering Server Environments for Oracle WebLogic Server*.

1. In the **Edit Tree**, go to **Environment**, then **Servers**, then *myServer*.
2. On the **Protocols** tab, on the **General** tab, modify the default settings for protocols as necessary.
3. Click **Save**.
4. Repeat on applicable servers.

Configure HTTP

You can define settings for HTTP connections.

1. In the **Edit Tree**, go to **Environment**, then **Servers**, then *myServer*.
2. On the **Protocols** tab, go to the **HTTP** subtab and update the settings for HTTP as necessary.
3. Optional: If you want to enable the tunneling of connections, turn on the **Enable Tunneling** option and then enter values in the **Tunneling Client Ping** and **Tunneling Client Timeout** fields.

   Note:

   These settings apply to all protocols in the server's default network configuration that support tunneling. See Setting Up WebLogic Server for HTTP Tunneling in *Administering Server Environments for Oracle WebLogic Server*.

4. Click **Save**.

Configure T3 Protocol

T3 is an Oracle proprietary remote network protocol that implements the Remote Method Invocation (RMI) protocol.

1. In the **Edit Tree**, go to **Environment**, then **Servers**, then *myServer*.
2. On the **Protocols** tab, in the **Complete Message Timeout** field, enter the maximum number of seconds that this server should wait for a complete message to be received.
3. In the **Maximum Message Size** field, enter the maximum number of bytes allowed in messages that are received over all supported protocols, unless overridden by a protocol-specific setting or a custom channel setting.
4. Optional: Turn on the **Enable Tunneling** option and then enter values in the **Tunneling Client Ping** and **Tunneling Client Timeout** fields.

> **Note:**
>
> These settings apply to all protocols in the server's default network configuration that support tunneling. See Setting Up WebLogic Server for HTTP Tunneling in *Administering Server Environments for Oracle WebLogic Server*.

5. Click **Save**.

Configure IIOP

The IIOP (Internet Inter-ORB Protocol) makes it possible for distributed programs written in different programming languages to communicate over the internet.

For information about using RMI-IIOP in your applications, see Understanding WebLogic RMI in *Developing RMI Applications for Oracle WebLogic Server*.

1. In the **Edit Tree**, go to **Environment**, then **Servers**, then *myServer*.
2. On the **Protocols** tab, go to the **IIOP** subtab.
3. Turn on the **Enable IIOP**.
4. If you want to modify the default configuration of IIOP (including the default IIOP user credentials), click **Show Advanced Fields** and update the options as necessary.
5. Click **Save**.

Change the Domain Mode

The domain mode of your WebLogic Server domain determines its default security configuration. Select the domain mode that best meets the security requirements of the environment in which WebLogic Server runs.

The domain modes, in order from least to most secure, are Development Mode, Production Mode, and Secured Production Mode. We recommend that you only use development mode for domains in development or demonstration environments.

The default values for various security configurations will change to match the selected domain mode. See Understand How Domain Mode Affects the Default Security Configuration in *Securing a Production Environment for Oracle WebLogic Server*.

> **Note:**
>
> As of WebLogic Server 14.1.2.0.0, production mode now turns on secured production mode by default. All *new* installations of WebLogic Server 14.1.2.0.0 and later follow this behavior. In previous releases, when you enabled production mode, secured production mode was disabled by default and you had to enable it explicitly.
>
> If you *upgrade* from WebLogic Server 14.1.1.0.0 and earlier, the behavior of your domain mode will not change. For example, when a domain in production mode is upgraded from 14.1.1.0.0 to 14.1.2.0.0 or later, it will remain in production mode with secured production mode *disabled*. However, if you upgrade your domain to 14.1.2.0.0 or later, and *then* select production mode as the new domain mode, it will *enable* secured production mode by default.

1. Shut down all of the Managed Servers in the domain.

   Changes to the domain mode require a full domain restart - a rolling restart is not sufficient.
2. In the **Edit Tree**, go to **Environment**, then **Domain**.
3. Change the domain mode:

Target domain mode	Perform these steps
Development Mode	Disable the **Production Mode** option to convert your domain to development mode.
Production Mode	Enable the **Production Mode** option to convert your domain to production mode. **Note:** As of WebLogic Server 14.1.2.0.0, when you select production mode, *secured* production mode is enabled by default. If you want to apply the features of basic production mode only, then you must explicitly disable the **Secured Production Mode** option. If you use Node Manager to start your Managed Servers, then you must also add the `–Dweblogic.securemode.SecureModeEnabled=false` startup argument to each Managed Server. See Configure Startup Arguments for a Managed Server.
Secured Production Mode	Enable the **Production Mode** and **Secured Production Mode** options. **Note**: For more information, see Using Secured Production Mode in *Administering Security for Oracle WebLogic Server*.

4. Click **Save**.
5. Commit your changes and restart your Administration Server. Then, you can start your Managed Servers.

Changes to the domain mode can affect the default URL of the Administration Server, specifically the protocol and the port number. When SSL/TLS and the administration port are enabled (by default, both are enabled in secured production mode), the default URL is `https://`*`hostname`*`:9002`. When SSL/TLS and the administration port are disabled (by default, both are disabled in development mode), the default URL is `http://`*`hostname`*`:7001`.

Resume a Server

If a server in the `STANDBY` or `ADMIN` state, when you are ready for the server to receive requests other than administration requests, you can resume the server to transition it into the `RUNNING` state.

For more information on the different server states, see Understanding Server Life Cycle in *Administering Server Startup and Shutdown for Oracle WebLogic Server*.

1. In the **Monitoring Tree**, go to **Environment**, then **Servers**.
2. Select the servers that you want to transition into a `RUNNING` state and click **Resume**.

Suspend a Server

When you suspend a server, it transitions a server instance from the `RUNNING` state to the `ADMIN` state. In the `ADMIN` state, the server is running, but it is only available for administration operations. This allows you to perform server and application-level administration tasks without risk to running applications.

For more information on the different server states, see Understanding Server Life Cycle in *Administering Server Startup and Shutdown for Oracle WebLogic Server*.

1. In the **Monitoring Tree**, go to **Environment**, then **Servers**.
2. Select the servers that you want to transition into an `ADMIN` state.
3. Click **Suspend** and choose how the server completes its current tasks:
   - **When work completes**: initiates a graceful suspension, which gives WebLogic Server subsystems and services time to complete certain application processing currently in progress.
   - **Force suspend now**: initiates a forced suspension, in which the server instructs subsystems to immediately drop in-flight requests.

Shut Down a Server

You can use WebLogic Remote Console to shut down a server instance of WebLogic Server.

WebLogic Remote Console is one of several tools you can use to shut down a server instance. For other options, see Shutting Down Instances of WebLogic Server in *Administering Server Startup and Shutdown for Oracle WebLogic Server*.
If you shut down the Administration Server, you won't be able to manage the domain from WebLogic Remote Console until it is restarted.

1. Optional: If you want to shut down a server gracefully and allow some WebLogic Server subsystems to complete existing tasks, then you can configure graceful shutdown settings.

   These options only apply when a server is shutdown gracefully. A forceful shutdown ignores them.

   a. In the **Edit Tree**, go to **Environment**, then **Servers**, then select the server where you want to apply grace shutdown settings.

   b. On the **Advanced** tab, select the **Start/Stop** sub-tab.

   c. Turn on **Ignore Sessions During Shutdown** to force the server to drop all HTTP sessions immediately instead of waiting for them to complete or timeout.

      Waiting for abandoned HTTP sessions to timeout can significantly lengthen the graceful shutdown process because the default session timeout is one hour.

   d. Specify a **Graceful Shutdown Timeout** in seconds to determine how long the server should wait before forcing a shutdown.

   e. Repeat on applicable servers.
2. In the **Monitoring Tree**, go to **Environment**, then **Servers**.
3. Select the servers that you want to shut down.
4. Click **Shutdown** and choose how the server completes its tasks:

- **When work completes**: initiates a graceful shutdown, which gives WebLogic Server subsystems time to complete certain application processing currently in progress.
- **Force shutdown now**: initiates a forced shutdown, in which the server instructs subsystems to immediately drop in-flight requests.

See Understanding Server Life Cycle in *Administering Server Startup and Shutdown for Oracle WebLogic Server*.

6

Securing Domains

WebLogic Server offers a robust set of security tools to protect your WebLogic Server environment from unauthorized access. Use WebLogic Remote Console to secure your system.

For general information on security concepts in WebLogic Server, see:

- *Understanding Security for Oracle WebLogic Server*
- *Administering Security for Oracle WebLogic Server*
- *Securing Resources Using Roles and Policies for Oracle WebLogic Server*
- *Securing a Production Environment for Oracle WebLogic Server*

Address Potential Security Issues

WebLogic Server regularly checks your domain to ensure it complies with its recommended security policies. If a domain does not meet the recommendations, a security warning is logged and displayed in WebLogic Remote Console.

When there are active security warnings in your domain, a red banner appears across the top of the WebLogic Remote Console window. It will remain until you have fixed the security issue. For more information on the issues that might trigger a security warning, see Review Potential Security Issues in *Securing a Production Environment for Oracle WebLogic Server*.

Do not rely on the Security Warnings Report alone to determine the security of your domain. While these security configuration settings cover a broad set of potential security issues, other security issues that do not generate warnings may still exist in your domain.

Note:

Environments running WebLogic Server 14.1.1.0 and earlier require the July 2021 Patch Set Update (PSU) to see security warnings.

1. In the Security Warnings banner, click **View/Refresh Report** to see which issues have been flagged.

   You can also get to the Security Warnings page through the **Monitoring** perspective, then **Environment**, then **Domain Security Runtime**.
2. Review the description of the issue to understand the security warning.
3. Follow the steps suggested in Resolution Information to fix the problem.

   The same issue can affect multiple servers within your domain simultaneously. Make sure to fix the issue on every affected server.

   If you think certain policies don't apply to your environment or are not feasible to implement with your business needs, you can disable individual security checks, with the exception of the minimum JDK version check.

4. If necessary, restart the servers to apply the fix and clear the security warnings.

Security Warning Fixes

For the latest information on security warnings and their suggested steps for resolution, see the My Oracle Support article, *WebLogic Server Security Warnings (Doc ID 2788605.1)*.

> **Note:**
>
> The same issue may affect multiple servers within your domain simultaneously. As you review the Security Warnings Report, make sure that you fix the issue on every affected server. Depending on the problem and its resolution, you may need to restart servers to update the Security Warnings Report.

Security Realms

A security realm is a collection of mechanisms designed to protect WebLogic Server resources. Each security realm consists of a set of configured security providers, users, groups, security roles, and security policies. A user must be defined in a security realm in order to access any WebLogic resources belonging to that realm.

For more information, see Security Realms in *Understanding Security for Oracle WebLogic Server*.

WebLogic Server provisions each new domain with a security realm with pre-configured security configurations. See The Default Security Configuration in WebLogic Server in *Administering Security for Oracle WebLogic Server*.

If the default security configurations do not meet your requirements, you can create a new security realm with custom settings and providers and set it as the default security realm. See Customizing the Default Security Configuration in *Administering Security for Oracle WebLogic Server*.

> **Note:**
>
> You can set WebLogic Server to immediately apply non-dynamic changes to a security realm without restarting WebLogic Server. See Enable Automatic Realm Restart.

Required Security Providers

For a security realm to be valid, the following security providers must be configured:

- Authentication provider
- Authorization provider
- Adjudication provider
- Credential Mapping provider
- Certification Path provider
- Role Mapping provider

WebLogic Server includes a default option for each of the required security providers, plus alternative options if the default provider does not suit your needs. You can also create your own custom providers.

Create a Security Realm

A security realm organizes the security configurations of a domain.

Before you create a new security realm, review the considerations outlined in Before You Create a New Security Realm in *Administering Security for Oracle WebLogic Server*.

1. In the **Edit Tree**, go to **Security**, then **Realms**.
2. Click **New**.
3. Enter a name for the new security realm.
4. Turn on the **Create Default Providers** option if you want WebLogic Server to add the required security providers to the security realm, ensuring that the new security realm is valid.

   You can modify the default providers or replace them with another provider as needed.

   > **Note:**
   >
   > If you do not enable **Create Default Providers**, then you must manually configure the required security providers before you can commit the new security realm. For a list of which security providers are required, see Required Security Providers.

5. Click **Create**.

Change Default Security Realm

You can configure WebLogic Server to replace the default security configurations with a custom security realm.

> **Note:**
>
> Although you can customize the original default security configuration, Oracle recommends creating an entirely new security realm and making changes to security configurations there, which allows you to easily revert to the original default security configuration if necessary.

1. If you haven't done so already, create a new security realm. See Create a Security Realm.
2. In the **Edit Tree**, go to **Environment**, then **Domain**.
3. Select the **Security** tab.
4. From the **Default Realm** drop-down list, select the security realm that you want to make the default.
5. Click **Save**.

Consider enabling the configuration archive to save previous domain configurations so you can revert to a previous security configuration if necessary. See Back Up Configuration Files.

Revert to a Previous Security Configuration

Certain mistakes when configuring a new security realm or security providers can prevent you from booting the server. If this happens, then you can revert the configuration XML files to reinstate a previous realm configuration and recover from the error.

You can only revert to a previous configuration if configuration archiving is enabled and has been saving previous versions of the configuration. See Back Up Configuration Files.

You can also use WLST Offline to correct a mistake that prevents you from booting the server.

Note:

This process will only revert your security realm (meaning, the configuration of the realm and its providers), not the users, groups, roles, or security policies used by the realm, which are persisted in a data store and not in the configuration files.

1. Locate the `config.jar` file that contains the security configuration to which you want to revert, copy it to a temporary directory, and unpack it.
2. Copy the unpacked configuration files to the appropriate location in the *`DOMAIN_HOME`*`/config` directory. For information about which directories hold which configuration files, see Domain Directory Contents in *Understanding Domain Configuration for Oracle WebLogic Server*.
3. Restart WebLogic Server.

Enable Automatic Realm Restart

You can set WebLogic Server to immediately apply non-dynamic changes to a security realm without restarting WebLogic Server.

In the default security realm, automatic realm restart is **disabled** by default. In *new* security realms that you create, automatic realm restart is **enabled** by default.

For more information, see Using Automatic Realm Restart in *Administering Security for Oracle WebLogic Server*.

1. In the **Edit Tree**, go to **Security**, then **Realms**, then *myRealm*.
2. Enable the **Automatically Restart after Non-Dynamic Changes** option.
3. Click **Save**.

Security Providers

The WebLogic Security Service supports a wide variety of security architectures, including multiple security providers.

Before you configure security providers for your WebLogic security realm, you should have a good understanding of how the WebLogic Security Service works and what sort of security architecture you want for your WebLogic environment. See:

- Overview of the WebLogic Security Service in *Understanding Security for Oracle WebLogic Server*

- Configuring Security Providers in *Administering Security for Oracle WebLogic Server*.
- Introduction to Developing Security Providers for WebLogic Server in *Developing Security Providers for Oracle WebLogic Server*.
- Understanding WebLogic Resource Security in *Securing Resources Using Roles and Policies for Oracle WebLogic Server*.

Your security architecture can consist of a combination of the default security providers included in WebLogic Server, security providers developed by third parties, or custom security providers that you develop yourself.

Types of security providers include:

- Authentication providers
- Identity Assertion providers
- Authorization providers
- Adjudication providers
- Role Mapping providers
- Credential Mapping providers
- Auditing providers

Configure an Authentication or Identity Assertion Provider

Use authentication providers to prove the identity of users or system processes. The WebLogic Authentication provider and WebLogic Identity Assertion provider are enabled by default but you can configure additional authentication or identity assertion providers as needed.

For more information about the authentication and identity assertion providers included in WebLogic Server, see Choosing an Authentication Provider in *Administering Security for Oracle WebLogic Server*.

1. In the **Edit Tree**, go to **Security**, then **Realms**, then *myRealm*, then **Authentication Providers**.
2. Click **New**.
3. In the **Name** field, enter a name for the new provider.
4. From the **Type** drop-down list, select the type of authentication provider.
5. Click **Create**.
6. On the **Configuration** page for the new authentication provider, set the appropriate values on the **Common** and **Provider-Specific Parameters** tabs.
7. Click **Save**.

We recommend that you configure the Password Validation provider immediately after configuring a new WebLogic domain. The password validation provider, which is included with WebLogic Server, can be configured with several out-of-the-box authentication providers to manage and enforce password composition rules. Whenever a password is created or updated in the security realm, the corresponding authentication provider automatically invokes the password validation provider to ensure that the password meets the established composition requirements. For more information, see Configure the Password Validation Provider.

Set the JAAS Control Flag

If you have multiple Authentication providers in your domain, use the JAAS control flag to control how each Authentication provider is used in the login sequence.

For information on how WebLogic Server handles multiple Authentication providers, see Using More Than One Authentication Provider in *Administering Security for Oracle WebLogic Server*.

1. In the **Edit Tree**, go to **Security**, then **Realms** and choose the security realm where you want to make changes.
2. Expand the **Authentication Providers** node and choose the Authentication provider that you want to configure.
3. Select an option from the **Control Flag** drop-down list:
   - Required
   - Requisite
   - Sufficient
   - Optional

   For information on each value, see Setting the JAAS Control Flag Option in *Administering Security for Oracle WebLogic Server*.
4. Click **Save**.
5. Repeat for the rest of the Authentication providers in the domain.

Configure an Authorization Provider

Use Authorization providers to determine who has access to a resource.

For more information on Authorization providers, see Configuring an Authorization Provider in *Administering Security for Oracle WebLogic Server*.

Note:

The WebLogic Authorization provider was deprecated in WebLogic Server 14.1.1.0.0 and will be removed in a future release. The XACML Authorization provider is the default provider.

1. In the **Edit Tree**, go to **Security**, then **Realms**, then *myRealm*, then **Authorizers**.
2. Click **New**.
3. In the **Name** field, enter a name for the new provider.
4. From the **Type** drop-down list, select the type of Authorization provider.
5. Click **Create**.
6. On the new Authorization provider page, set the desired values.
7. Click **Save**.

Configure the WebLogic Adjudication Provider

An Adjudication provider resolves authorization conflicts by weighing each Authorization provider's access decision and determining whether to permit access to the requested resource.

If you only have one Authorization provider configured in the security realm and it is the WebLogic Authorization provider, then an `ABSTAIN` returned from the single Authorization provider is treated as a `DENY`.

Each security realm must have one, and only one, Adjudication provider.

WebLogic Server offers one Adjudication provider for the WebLogic Security Framework: the WebLogic Adjudication provider. Note that WebLogic Remote Console refers to the WebLogic Adjudication provider as the Default Adjudicator.

For more information on Adjudication providers, see Configuring the WebLogic Adjudication Provider in *Administering Security for Oracle WebLogic Server*.

1. In the **Edit Tree**, go to **Security**, then **Realms**, then *myRealm*, then **Adjudicator**.
2. Click **Create**.
3. In the **Name** field, enter a name for the new provider.
4. From the **Type** drop-down list, select the type of Adjudication provider.
5. Click **Create**.
6. On the new Adjudication provider page, set the appropriate values on the **Default Adjudicator Parameters** tab.
7. Click **Save**.

Configure a Role Mapping Provider

Role mapping is the process whereby principals (users or groups) are dynamically mapped to security roles at runtime. A Role Mapping provider determines what security roles apply to the principals stored in a subject when the subject is attempting to perform an operation on a WebLogic resource.

Since these operations usually involve gaining access to a WebLogic resource, Role Mapping providers are typically used with Authorization providers.

Note:

The WebLogic Role Mapping provider was deprecated in WebLogic Server 14.1.1.0.0 and will be removed in a future release. The XACML Role Mapping provider is the default provider.

For more information on Role Mapping providers, see Configuring a Role Mapping Provider in *Administering Security for Oracle WebLogic Server*.

1. In the **Edit Tree**, go to **Security**, then **Realms**, then *myRealm*, then **Role Mappers**.
2. Click **New**.
3. In the **Name** field, enter a name for the new provider.

4. From the **Type** drop-down list, select the type of Role Mapping provider.
5. Click **Create**.
6. On the new Role Mapping provider page, update values as necessary.
7. Click **Save**.

Configure an Auditing Provider

An Auditing provider collects, stores, and distributes information about operating requests and the outcome of those requests for the purposes of non-repudiation.

You can configure multiple Auditing providers in a security realm, but none are required. The default security realm does not include an Auditing provider.

WebLogic Auditing provider is the standard Auditing provider for the WebLogic Security Framework. Note that the WebLogic Remote Console refers to the WebLogic Auditing provider as the Default Auditor.

You can also configure WebLogic Server to audit configuration actions. See Enable Configuration Auditing.

For more information on Auditing providers, see Configuring the WebLogic Auditing Provider.

1. In the **Edit Tree**, go to **Security**, then **Realms**, then *myRealm*, then **Auditors**.
2. Click **New**.
3. In the **Name** field, enter a name for the new provider.
4. Click **Create**.
5. On the new Auditing provider page, set the appropriate values on the **Default Auditor Parameters** tab.
6. Click **Save**.

Configure a Credential Mapping Provider

A Credential Mapping provider allows WebLogic Server to log into a remote system on behalf of a subject that has already been authenticated. You must have one Credential Mapping provider in a security realm, and you can configure multiple Credential Mapping providers in a security realm.

For more information on Credential Mapping providers, see Configuring Credential Mapping Providers in *Administering Security for Oracle WebLogic Server*.

1. In the **Edit Tree**, go to **Security**, then **Realms**, then *myRealm*, then **Credential Mappers**.
2. Click **New**.
3. In the **Name** field, enter a name for the new provider.
4. From the **Type** drop-down list, select the type of Credential Mapper.
5. Click **Create**.
6. On the new Credential Mapping provider page, update the values on the **Default Credential Mapper Parameters** tab as necessary.
7. Click **Save**.

After you have configured a credential mapping provider, you can map credentials to WebLogic resources. You can create credential mappings for the following WebLogic Server resources:

- Application deployments
  - EJBs
  - JDBC applications
  - Resource adapters
- Data sources
- Remote resources

The general process is:

1. In the Edit Tree, configure an MBean. Commit your changes and then restart the server, if necessary.
2. In the Security Data Tree, under Credential Mappers, find the corresponding node for the MBean you configured. You may need to define the properties of the WebLogic resource to identify it, forming a connection between the MBean's configuration data and its security data. See Identify Resources for Credential Mapping.
3. Create mappings for the WebLogic resource.

Identify Resources for Credential Mapping

Certain WebLogic resources require that you manually form a connection between its MBean configuration data and its security data.

> **Note:**
>
> As part of this process, you will create the first credential mapping for the resource.

You must configure the WebLogic resource before you can add any mappings.

1. In the **Security Data Tree**, go to **Realms**, then *myRealm*, then **Credential Mappers**.
2. Continue until you reach the WebLogic resource that you want to prepare for credential mappings.

   For example:

   - EJB component nodes appear under Credential Mappers: *credentialMappingProvider*: App Deployments: *applicationName*: EJBs
   - Resource adapter nodes appear under Credential Mappers: *credentialMappingProvider*: App Deployments: *applicationName*: Resource Adapters
   - Remote resources appear under Credential Mappers: *credentialMappingProvider*: Remote Resources.

   You do not need to identify data sources.
3. Click **New** and fill in the fields as necessary.
4. Click **Create**.

A reference to the resource is added under the resource's node. The name of the resource is generated by combining its property values.

You cannot delete a reference to a resource with multiple credential mappings. To delete a resource reference, you must delete all of the credential mappings first - which will then delete the resource reference automatically.

Add a Credential Mapping

You can add a new credential mapping to associate another WebLogic resource to a remote user.

1. In the **Security Data Tree**, go to **Realms**, then *myRealm*, then **Credential Mappers**, then *credentialMappingProvider*, then *wlResourceType*, then *wlResourceName*, then **Credential Mappings**.
2. Click **New**.
3. Fill in the fields as needed.

> **Note:**
>
> If you want to map to a remote user that is already referenced in the remote resource, disable the **Create Credential** option.

4. Click **Create**.

The credential mappings and credentials for each WebLogic resource appear under the resource’s node.

When you delete a credential mapping, the remote user that the WebLogic resource was previously associated with is also deleted if it was the only credential mapping using that remote user.

Remap a WebLogic Resource

You can edit a credential mapping to associate a WebLogic user for a WebLogic resource to a different remote user.

WebLogic Remote Console must already be aware of the remote user before you can remap the WebLogic Server user. If you want to remap the WebLogic resource to a new remote user, you must first add it to WebLogic Remote Console.

1. In the **Security Data Tree**, go to **Realms**, then *myRealm*, then **Credential Mappers**, then *credentialMappingProvider*, then *wlResourceType*, then *wlResourceName*, then **Credential Mappings**.
2. Select the credential mapping that you want to edit.
3. In the **Remote User** field, replace the username of the current remote user with the username of the remote user that you want to remap the WebLogic resource to.
4. Click **Save**.

Add a Set of Credentials

After you have added the first set of credentials for a remote system to a WebLogic resource, you can add more users from that remote system.

1. In the **Security Data Tree**, go to **Realms**, then *myRealm*, then **Credential Mappers**, then *credentialMappingProvider*, then *wlResourceType*, then *wlResourceName*, then **Credentials**.
2. Click **New**.
3. Enter the username and password for the remote user.

4. Click **Create**.

You can now map a WebLogic user for the WebLogic resource to the new set of credentials.

> **Note:**
>
> If the password of the remote user changes, you'll need to update it in WebLogic Remote Console or the mapping will break and prevent the WebLogic resource from logging into the remote system.

Delete a Set of Credentials

Each WebLogic resource has its own set of credentials. Removing a set of credential from WebLogic Remote Console will not affect the user in the remote system.

You cannot delete a credential that currently has a WebLogic resource mapped to it.

1. In the **Security Data Tree**, go to **Realms**, then *myRealm*, then **Credential Mappers**, then *credentialMappingProvider*, then *wlResourceType*, then *wlResourceName*, then **Credentials**.
2. If the set of credentials that you want to delete has an active credential mapping, you can either:
   - Select any relevant mappings and update the **Remote User** field to a new remote user.
   - Delete all of the associated credential mappings. Once all credential mappings associated with this set of credential are deleted, the credential itself will also be automatically deleted and you can skip the rest of the steps.
3. Under the same WebLogic resource, expand the **Credentials** node.
4. Delete the set of credentials.
5. Click **Save**.

Configure a Certification Path Provider

A Certification Path provider completes and validates certificate chains by using trusted CA based checking.

The provider checks the signatures in the chain, ensures that the chain has not expired, and confirms that one of the certificates in the chain is issued by one of the server's trusted CAs. If any of these checks fail, the chain is not valid. Optionally, the provider checks the certificate chain's basic constraints.

For more information on certificate validation on WebLogic Server, see Configuring the Certificate Lookup and Validation Framework in *Administering Security for Oracle WebLogic Server*.

1. In the **Edit Tree**, go to **Security**, then **Realms**, then *myRealm*, then **Cert Path Providers**.
2. Click **New**.
3. In the **Name** field, enter a name for the new provider.
4. From the **Type** drop-down list, select the type of Certificate Lookup and Validation (CLV) provider.

5. Click **Create**.
6. Optional: If you want to make this CLV provider the current builder of certificate chains for this realm, perform the following steps:
   a. In the **Edit Tree**, go to **Security**, then **Realms**, then *myRealm*.
   b. On the **Cert Path Builder** tab, select the CLV provider from the **Cert Path Builder** drop-down list.
   c. Click **Save**.

Configure the Password Validation Provider

When configured in a security realm, the Password Validation provider is automatically invoked by a supported authentication provider whenever a password is created or updated for a user in that realm. The Password Validation provider then performs a check to determine whether the password meets the criteria established by the composition rules, and the password is accepted or rejected as appropriate.

The password composition rules you can configure for the Password Validation provider include:

- User name policies, such as whether the password can be the same as the username.
- Password length policies, such as a minimum or maximum length.
- Character policies, such as the minimum or maximum number of alphabetic, numeric, or non-alphanumeric characters required in each password.

Passwords cannot contain a curly brace `{` as the first character.

Note:

If the Default Authentication provider is configured in the security realm, make sure that the setting for the minimum password length is consistent in both the Default Authentication provider and the Password Validation provider.

1. In the **Edit Tree**, go to **Security**, then **Realms**, then *myRealm*, then **Password Validators**.
2. Click **New**.
3. Enter a name for the password validator.
4. Click **Create**.
5. On the configuration page of the password validator that you just created, select the **System Password Validator Parameters** tab.
6. Use the configuration options to determine the criteria for an acceptable password.
7. Click **Save**.

Configure Custom Security Providers

If the security providers provided by WebLogic Server do not meet the needs of your environment, then you can design your own custom security providers.

1. Create a custom security provider. For guidance, see Introduction to Developing Security Providers for WebLogic Server in *Developing Security Providers for Oracle WebLogic Server*.
2. Place the MBean JAR file for the custom provider in the `WL_HOME/server/lib/mbeantypes` directory.
3. Add the custom security provider to the domain's security realm using WLST or the WebLogic Server RESTful management interface.

> **Note:**
>
> - If you are adding a deployable Authorization provider and the provider does not support parallel security policy, then set the `RealmMBean.DeployableProviderSynchronizationEnabled` attribute to `true`. Next, for the `RealmMBean.DeployableProviderSynchronizationTimeout` attribute, enter a timeout value (in milliseconds) or accept the default value.
> - If you are adding a deployable Role Mapping provider and the provider does not support role modification, then set the `RealmMBean.DeployableProviderSynchronizationEnabled` attribute to `true`. Next, for the `RealmMBean.DeployableProviderSynchronizationTimeout` attribute, enter a timeout value (in milliseconds) or accept the default value.
>
> For more information, see Is Your Custom Authorization Provider Thread Safe? or Is Your Custom Role Mapping Provider Thread Safe? in *Developing Security Providers for Oracle WebLogic Server*.

After you add the custom security provider to the security realm, you can use WebLogic Remote Console to configure and manage the inherited, standard attributes of the custom security provider. However, you cannot use WebLogic Remote Console to manage the custom attributes from your MDF file.

Configure the Embedded LDAP Server

The embedded LDAP server contains user, group, group membership, security role, security policy, and credential map information. By default, each WebLogic Server domain has an embedded LDAP server configured with the default values set for each attribute.

The WebLogic Authentication, Authorization, Credential Mapping, and Role Mapping providers use the embedded LDAP server as their database. If you use any of these providers in a new security realm, you may want to change the default values for the embedded LDAP server to optimize its use in your environment.

1. In the **Edit Tree**, go to **Environment**, then **Domain**.
2. On the **Security** tab, select the **Embedded LDAP** subtab.
3. Update the values as appropriate for your environment.

   Consider updating the settings for backing up for the embedded LDAP servers. Specifically **Backup Hour**, **Backup Minute**, and **Backup Copies**. For information on how WebLogic Server backs up data for the embedded LDAP server, see Backup and Recovery in *Administering Security for Oracle WebLogic Server*.
4. Click **Save** and commit your changes.
5. Restart the server.

Note:

The WebLogic Security providers store their data in the embedded LDAP server. When you delete a WebLogic Security provider, the security data in the embedded LDAP server is not automatically deleted. The security data remains in the embedded LDAP server in case you want to use the provider again. Use an external LDAP browser to delete the security data from the embedded LDAP server.

Users and Groups

WebLogic Server employs users and groups to control access to its resources.

Users are entities that can be authenticated in a security realm. A user can be a person, such as application end user, or a software entity, such as a client application, or other instances of a WebLogic Server. As a result of authentication, a user is assigned an identity or principal. Each user is given a unique identity within the security realm. Users may be placed into groups that are associated with security roles, or can be directly associated with security roles.

Groups are logically ordered sets of users. Users are organized into groups that can have different levels of access to WebLogic resources, depending on their job functions. Managing groups is more efficient than managing large numbers of users individually. All user and group names must be unique within a security realm.

For more information, see Users, Groups, And Security Roles in *Securing Resources Using Roles and Policies for Oracle WebLogic Server*.

WebLogic Remote Console can only add, edit, or delete the users and groups within the default authentication provider (WebLogic Authentication Provider). If you are using another authentication provider that supports the required WebLogic Server APIs, you can view its users and groups in WebLogic Remote Console, but to manage them, you'll need to use an external tool specific to the provider. Providers that do not support the required WebLogic Server APIs do not appear in the Security Data tree.

Create a User

You can add users to the WebLogic Authentication provider.

1. In the **Security Data Tree**, go to **Realms**, *myRealm*, then **Authentication Providers**, then **DefaultAuthenticator**, then **Users**.
2. Click **New**.
3. Enter a **Name**, **Description**, and **Password** for the user.

   User names must be unique in the security realm and passwords must be eight characters or longer.
4. Click **Create**.
5. Optional: If you want to add the new user to a group, then under the **Membership** tab, select a group under the **Available** section and move it to the **Chosen** section.

   User groups provide an efficient way to manage multiple users with the same responsibilities.

Create a Group

You can add groups to the WebLogic Authentication provider.

1. In the **Security Data Tree**, go to **Realms**, *myRealm*, then **Authentication Providers**, then **DefaultAuthenticator**, then **Groups**.
2. Click **New**.
3. Enter a **Name** (and optionally, a **Description**) for the group.

   Group names must be unique in the security realm.
4. Click **Create**.

You can nest a group under another so it inherits policies. Under the **Membership** tab, move groups from **Available** over to **Chosen** and the current group will be nested under the Chosen groups.

Filter Users or Groups

If you have a large number of users or groups in an authentication provider, then you can reduce which users or groups are visible by applying a filter. Only users or groups that match the filter criteria are displayed.

1. In the **Security Data Tree**, go to **Realms**, *myRealm*, then **Authentication Providers**, then *myAuthenticationProvider*.
2. Click either the **Users** or the **Groups** node. You cannot filter users and groups at the same time.
3. Click **Filter** and in the **Name Filter** field, enter the match criteria.

   The filter performs an *exact* match on the value you enter. If you want to match all users or groups that *contain* the match criteria, then add the wildcard character `*` to expand the search.

   For example:

   - If you enter `Deploy`, it will not return the Deployers group. Instead, use `Deployers` or `Deploy*` to show the Deployers group.
   - *For the Default authenticator (WebLogic Authentication provider) only*: You can add two `*` characters to the match criteria. If you enter `*admin*`, it will return any user or group that contains the string `admin` such as AdminChannelUsers, Administrators, and sysadmin.

   The match is case-insensitive.
4. Click **Done**.

The filter persists until you clear it or restart WebLogic Remote Console. If a filter is active, its criteria is displayed at the top of the page. To remove the filter and see all users or groups, click **Filter** and enter `*` in the **Name Filter** field.

Set User Lockout Attributes

You can control how many login attempts can be made by the same user account before WebLogic Server locks the account.

For more information, see How Passwords Are Protected in WebLogic Server and Protecting User Accounts in *Administering Security for Oracle WebLogic Server*.

1. In the **Edit Tree**, go to **Security**, then **Realms**, then *myRealm*.
2. On the **User Lockout** tab, turn on the **Lockout Enabled** option.
3. In the **Lockout Threshold** field, set the maximum number of invalid user login attempts that you want to allow before the user account is locked.
4. Modify any additional user lockout attributes as necessary.
5. Click **Save**.

Unlock a User

You can unlock user accounts that became locked because of excessive failed login attempts on a per server basis.

1. In the **Monitoring Tree**, go to **Environment**, then **Servers**, then *myServer*.
2. Go to **Security**, then **Realm Runtimes**, then *myRealm*.
3. Click **Unlock User**.
4. Enter the username of the locked user account. If applicable to your domain, you can also enter the identity domain.
5. Click **Done**.

View Users and Groups

After you configure an authentication provider, you can view its users and groups in WebLogic Remote Console, if the provider supports the required WebLogic Server APIs. WebLogic Remote Console displays up to 1000 users or groups at a time. You cannot view group membership.

1. In the **Security Data Tree**, go to **Realms**, *myRealm*, then **Authentication Providers**.
2. Select the authentication provider whose users and groups you want to view.
3. Expand the **Users** or **Groups** nodes to view the users and groups.

For the default authentication provider (WebLogic Authentication Provider) *only*, you can also add, modify, or delete users and groups.

Security Policies and Roles

Use security policies to manage who can access a resource in a WebLogic Server domain.

A resource is an entity (such as a Web Service or a server instance) or an action (such as a method in a Web Service or the act of shutting down a server instance). For a list of resource types, see Resource Types You Can Secure with Policies in *Securing Resources Using Roles and Policies for Oracle WebLogic Server*.

A security policy specifies which users, groups, or roles can access the resource according to a set of conditions. Whenever possible, you should use security roles to determine access control. A security role, like a security group, grants an identity to a user. Unlike a group, however, membership in a role can be based on a set of conditions that are evaluated at runtime.

For most types of WebLogic resources, you can use WebLogic Remote Console to define the security policies and roles that restrict access. For Web applications and EJB resources, you can also use deployment descriptors.

The general process to secure a WebLogic resource is:

1. Create users and groups.
2. Manage default security roles or create new ones. We recommend that you use roles to secure WebLogic resources (instead of users or groups) to increase efficiency for administrators who work with many users. You can use the default roles that WebLogic Server provides or create your own.
3. Create and apply security policies.

For more information, see *Securing Resources Using Roles and Policies for Oracle WebLogic Server*.

Security Roles

A security role is an identity granted to users or groups based on specific conditions. Multiple users or groups can be granted the same security role and a user or group can be assigned more than one role. Security roles are used by policies to control access a WebLogic resource.

WebLogic Server provides a default set of roles that you can use with any policy. These are called global roles and you can create your own if the default global roles do not meet your needs.

You can also create roles that are only used by policies for a specific resource. These are called scoped roles. For example, You can create a scoped role for a specific EJB that contains highly sensitive business logic. When you create a policy for the EJB, you can specify that only the scoped role can access the EJB.

If two roles conflict, the role of a narrower scope overrides the role of the broader scope. For example, a scoped role for an EJB resource overrides a global role or a scoped role for the enterprise application that contains the EJB.

Security roles are created within a security realm, and the roles can be used only when the realm is active.

For more information, see Overview of Security Roles in *Securing Resources Using Roles and Policies for Oracle WebLogic Server*.

Create a Global Role

Create a new role that applies to all WebLogic resources deployed within a security realm (and thus the entire WebLogic Server domain).

> **Note:**
>
> Before creating a new global security role, review the Default Global Roles in *Securing Resources Using Roles and Policies for Oracle WebLogic Server* to assess if an existing role is sufficient for your needs.

1. In the **Security Data Tree**, go to **Realms**, then *myRealm*, then **Role Mappers**, then **XACMLRoleMapper**, then **Global**, then **Roles**.
2. Click **New**.
3. Enter a name for the new global role.
4. Click **Create**.

After a role is created, you can build a policy that uses conditions to determine which users or groups it encompasses. We recommend that whenever possible, you use the Group condition which grants the security role to all members of a specified group.

Create a Scoped Role

Create a new role that can only be used with policies that apply to specific resources.

1. In the **Security Data Tree**, go to **Realms**, then *myRealm*, then **Role Mappers**, then **XACMLRoleMapper**, then *myResource*, then **Roles**.

   myResource is the node path to the resource and may contain multiple node children. For example, to create a scoped role for a queue in a JMS module, your path would be … **XACMLRoleMapper**, then **JMS Modules**, then *jmsModuleName*, then **Queues**, then *queueName*, then **Roles**.
2. Click **New**.
3. Enter a name for the new scoped role.
4. Click **Create**.

After a role is created, you can build a policy that uses conditions to determine which users or groups it encompasses. We recommend that whenever possible, that you use the Group condition which grants the security role to all members of a specified group.

Security Policies

A security policy specifies the conditions that users, groups, or roles must meet to access a resource. Policies must have one or more conditions and you can combine conditions to create complex policies for more dynamic access control.

Root level policies apply to all instances of a specific resource type, for example, all JMS resources in your domain. All default security policies are root level policies.

You can also create policies that only apply to a specific resource instance. If the instance contains other resources, the policy will apply to the included resource as well. For example, you can create a policy for an entire JMS system resource, or for a particular queue or topic within that resource.

The policy of a narrower scope overrides policy of a broader scope. For example, if you create a security policy for a JMS system resource and a policy for a JMS queue within that system resource, the JMS queue will be protected by its own policy and will ignore the policy for the system resource.

For more information, see Security Policies in *Securing Resources Using Roles and Policies for Oracle WebLogic Server*.

It's recommended that you use the Role condition where possible. Basing conditions on security roles lets you create one security policy that takes into account multiple users or groups, and is a more efficient method of management.

See Security Policy Conditions in *Securing Resources Using Roles and Policies for Oracle WebLogic Server*.

Building Security Policies

You can build and edit a policy by modifying a condition's arguments or by modifying the relationships between conditions in the policy.

Note:

You can only edit the arguments of the condition. If you want to use a different predicate for the condition, you must add a new condition.

Conditions have three different types of relationships: `AND`, `OR`, and `Combination`.

- `AND`: All of the conditions joined by an `AND` operator must be met.
- `OR`: At least one of the conditions joined by an `OR` operator must be met.
- `Combination`: Two or more conditions are combined and must be evaluated as a group. Conditions *within* a combination are themselves related to each other through `AND` or `OR` operators.

By default, a new condition is added as a simple condition at the top of the list of conditions. To insert a new condition elsewhere, select an existing condition and then click **Add Condition**. You will get a drop-down list with options to add the new condition either above or below the selected condition. The order of conditions is not meaningful to how the policy is interpreted.

A policy can contain multiple simple or compound conditions or a mix of simple and compound conditions. You can also nest compound conditions.

Note:

Why should you use compound conditions? Consider the following scenario: a resource exists where you want Administrators to always have access, but to restrict Deployer access to between 9 a.m. and 5 p.m EST. The following policy would address both requirements:

- Condition 1 (simple): Role: Admin

`OR`

- Condition 2 (compound): Role: Deployer `AND` access occurs between 09:00:00 and 17:00:00 GMT -5:00

Use the actions on the Policy page to edit a policy.

Action	Description
Add Condition	Adds a new condition to the policy. You can choose to add the new condition above or below another condition.
Combine	When multiple conditions are selected, you can combine them to create a compound condition.
Uncombine	When a compound condition is selected, you can break it into independent (simple) conditions.
Remove	Deletes a simple or compound condition from the policy. When there are no conditions in a policy, the default policy applies.
Negate	Reverses the meaning of a condition. The criteria to access a resource becomes the opposite of the original condition.
Reset	Reverts the policy to its last *saved* change, not the last change. It's recommended that you save your policy frequently or you may lose several changes unintentionally using the Reset action.

You can also edit a policy from its **Advanced** tab where the policy is expressed as string. Any changes made to a policy in the Advanced tab are reflected in the main Policy tab, and vice versa.

To delete a policy, simply delete all of its conditions. Confirm that the default security policy for the resource instance will provide adequate access control before you delete a policy.

Create a Policy for Resource Instances

You can create a security policy that only applies to a specific resource instance. If the instance contains other resources, the policy will apply to the included resources as well.

1. In the **Security Data Tree**, go to **Realms**, then *myRealm*, then **Authorizers**, then **XACMLAuthorizer**, then *myResourceInstance*.

   myResourceInstance is the node path to the resource instance and may contain multiple nodes. For example, to create a policy for a queue in a JMS module, your path would be "…**XACMLAuthorizer**, then **JMS Modules**, then *jmsModuleName*, then **Queues**, then *queueName*."

2. Click **Add Condition**.

3. Select a predicate from the **Predicate List**. Depending on the predicate you choose, you may need to configure arguments for the condition.
4. Click **OK**.
5. Click **Save**.
6. Optional: Add more conditions to the policy to increase its complexity.

Identity and Trust

WebLogic Server uses private keys, digital certificates, and trusted certificates issued by certificate authorities to establish and verify server identity and trust.

WebLogic Server provides a default identity keystore and a default trust keystore. These default keystores are appropriate for testing and development purposes only.

- In WebLogic Server 14.1.1.0 and earlier, the default identity keystore and default trust keystore are `DemoIdentity.jks` and `DemoTrust.jks`, respectively.
- In WebLogic Server 14.1.2.0 and later, the default identity keystore and default trust keystore are `DemoIdentity.p12` and `DemoTrust.p12`, respectively.

Additionally, WebLogic Server trusts the certificate authorities in the `cacerts` file in the JDK.

For more information, see:

- Identity and Trust in *Understanding Security for Oracle WebLogic Server*
- Configuring Keystores in *Administering Security for Oracle WebLogic Server*

> **Note:**
>
> 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 OPSS Keystore Service (KSS) provides an alternative mechanism to manage keys and certificates for message security. See Managing Keys and Certificates in *Securing Applications with Oracle Platform Security Services*. The OPSS KSS makes using certificates and keys easier by providing central management and storage of keys and certificates for all servers in a domain. You can use the OPSS KSS to create and maintain keystores of type KSS. If the Oracle Java Required Files (JRF) template is installed on the WebLogic Server system, you have the option to use KSS keystores. The KSS keystore is available only with the JRF template and is not available with the default WebLogic Server configuration.

Configuring Identity and Trust in WebLogic Server

1. Obtain digital certificates, private keys, and trusted CA certificates from the `CertGen` utility, the `keytool` utility, the OPSS Keystore Service, or a reputable vendor that creates and signs certificates for use on the internet. You can also use the digital certificates, private keys, and trusted CA certificates provided by WebLogic Server. The demonstration digital certificates, private keys, and trusted CA certificates should be used in a development environment only.
   See Configure Keystores.

2. If using KSS, verify whether KSS is properly populated, and note the KSS URIs and aliases of the required certificates and keys. You will need the KSS URIs and key aliases when specifying keystores, keys, and certificates.
3. Store the private keys, digital certificates, and trusted CA certificates. Private keys and trusted CA certificates are stored in a keystore. If using KSS, import the required keys and certificates to KSS.
4. Configure the identity and trust keystores for a WebLogic Server instance. See Configure Keystores.
5. Configure SSL/TLS attributes for the server. These attributes describe the location of the identity key and certificate in the keystore. See Set Up SSL/TLS.

Configure Keystores

WebLogic Server uses private keys, digital certificates, and trusted certificates issues by certification authorities to establish and verify server identity and trust. You can use JKS or PKCS12 keystores for identity and trust.

For more information, see Configuring Keystores in *Administering Security for Oracle WebLogic Server*.

Note:

For testing and development purposes *only*, WebLogic Server provides a demonstration identity keystore and a demonstration trust keystore. For production environments, you should configure your own identity and trust keystores.

- In WebLogic Server 14.1.1.0 and earlier, the demo identity keystore and demo trust keystore are `*DOMAIN_NAME*/security/DemoIdentity.jks` and `*WL_HOME*/server/lib/DemoTrust.jks`, respectively.
- In WebLogic Server 14.1.2.0 and later, the demo identity keystore and demo trust keystore are `*DOMAIN_NAME*/security/DemoIdentity.p12` and `*DOMAIN_NAME*/security/DemoTrust.p12`, respectively.

Additionally, the JDK installation provides the `cacerts` truststore in JKS format at `*JDK*/lib/security/cacerts`.

1. If you are using custom identity and custom trust keystores:
   a. Obtain private keys and digital certificates from a reputable third-party certificate authority (CA).
   b. Create identity and trust keystores.
   c. Load the private keys and trusted CAs into the keystores.
2. In the **Edit Tree**, go to **Environment**, then **Servers**, then *myServer*.
3. Click the **Security** tab, then the **Keystores** tab.
4. From the **Keystores** drop-down list, select a type for storing and managing private keys/digital certificate pairs and trusted CA certificates.
   - **Demo Identity and Demo Trust** - Select this option to use the demo certificates (suitable for development and testing). This is the default setting, and uses the demonstration identity and trust keystores, and the JDK `cacerts` keystore.

To use a KSS keystore for demo identity and trust, you must first enable the **Use KSS For Demo** option (on the **Environment: Domain** page, under the **Security** tab (click **Show Advanced Fields**)) which determines whether the Demo Identity and Demo Trust key stores should be obtained from the Oracle Key Store Service (KSS).

- **Custom Identity and Java Standard Trust** - Select this option to use an identity keystore you created and the trusted CAs that are defined in the `cacerts` file in the *`JAVA_HOME`*`/jre/lib/security` directory.
- **Custom Identity and Custom Trust** - Select this option to use both identity and trust keystores that you created.
- **Custom Identity and Command Line Trust** - Select this option to use an identity keystore that you created, but the trust keystore is passed as an argument in the command that starts WebLogic Server.

5. Define attributes for the identity and trust keystores. Depending on the keystore type that you selected, different options are available.
6. Click **Save**.
7. Repeat on all applicable servers.
8. Optional: If you are updating a keystore configuration and have SSL/TLS configured already, you can check that all SSL/TLS connections exist according to the specified connection. In the **Monitoring Tree**, go to **Environment**, then **Servers**. Restart the applicable servers.
9. Optional: If you enabled custom keystores and want to use Node Manager to start Managed Servers, you must also update the `nodemanager.properties` file to match.

   At minimum, update the following node manager properties as applicable to your environment:

   - `CustomIdentityAlias`
   - `CustomIdentityKeyStoreFileName`
   - `CustomIdentityPrivateKeyPassPhrase`
   - `CustomIdentityKeyStorePassPhrase`
   - `KeyStores`

   See Node Manager Properties in *Administering Node Manager for Oracle WebLogic Server*.

Enable Certificate Revocation Checking

WebLogic Server’s JSSE implementation supports X.509 certificate revocation (CR) checking, which checks a certificate’s revocation status as part of the SSL/TLS certificate validation process. CR checking improves the security of certificate usage by ensuring that received certificates have not been revoked by the issuing certificate authority. By default, CR checking is disabled in WebLogic Server.

WebLogic Server's CR checking implementation includes both the Online Certificate Status Protocol (OCSP) and certificate revocation lists (CRLs). For more information, see X.509 Certificate Revocation Checking in *Administering Security for Oracle WebLogic Server*.

Ensure that you have configured the identity and trust keystores for WebLogic Server. See Identity and Trust.

1. In the **Edit Tree**, go to **Environment**, then **Domain**.

2. On the **Security** tab, click the **SSL Certificate Revocation Checking** subtab.
3. Turn on the **Enable Certificate Revocation Checking** option.
4. Select a revocation checking method from the **Revocation Checks** drop-down list. The default is OCSP_THEN_CRL.

   Use the **OCSP** and **CRL** tabs to customize settings for the revocation checking method. For information on the options, see Using the Online Certificate Status Protocol or Using Certificate Revocation Lists in *Administering Security for Oracle WebLogic Server*.
5. Optional: If you want a certificate whose revocation status cannot be determined to fail SSL/TLS certificate path validation, turn on the **Fail on Unknown Revocation Status** option.

   When this option disabled, if an X.509 certificate's revocation status cannot be determined, but the SSL/TLS certificate path validation is otherwise successful, the certificate will be accepted.
6. Click **Save**.

You can configure certificate authority overrides to the certificate revocation configuration. See Configure Certificate Authority Overrides.

Configure Certificate Authority Overrides

Configuring a certificate authority override allows you to specify CR checking behavior that is specific to certificates issued by a particular certificate authority (CA). A certificate authority override always supersedes the corresponding certificate revocation (CR) checking configuration that is set at the domain level.

A certificate authority override can be used to supersede, for a given CA, any domain-wide CR checking configuration settings, with the exception of the CRL local cache, which is configured on a domain-wide basis only.

1. If you haven't done so already, enable certificate revocation checking as described in Enable Certificate Revocation Checking.
2. In the **Edit Tree**, go to **Security**, then **Certificate Authority Overrides**.
3. Click **New** and then enter a name for the new certificate authority override.
4. Click **Create**.
5. In the **Distinguished Name** field, enter the distinguished name of the CA. This must be the complete issuer distinguished name (defined in RFC 2253) of the certificates for which this override applies.

   For example, `CN=CertGenCAB, OU=FOR TESTING ONLY, O=MyOrganization, L=MyTown, ST=MyState,C=US`.
6. Use the **General**, **OCSP**, and **CRL** tabs to modify the settings as necessary for your environment. For descriptions of the attributes and when you might want to configure them, see:
   - General Certificate Authority Overrides
   - Configuring OCSP Properties in a Certificate Authority Override
   - Configuring CRL Properties in a Certificate Authority Override

   in *Administering Security for Oracle WebLogic Server*.
7. Click **Save**.

SSL/TLS

Transport Layer Security (TLS) and its predecessor, Secure Sockets Layer (SSL), ensure secure connections by allowing two applications connecting over a network connection to authenticate the other's identity and by encrypting the data exchanged between the applications.

Authentication allows a server and optionally a client to verify the identity of the application on the other end of a network connection. Encryption makes data transmitted over the network intelligible only to the intended recipient.

WebLogic Server supports SSL/TLS on a dedicated listen port which defaults to 7002. To establish an SSL/TLS connection, a Web browser connects to WebLogic Server by supplying the SSL/TLS listen port and the HTTPs protocol in the connection URL, for example, `https://myserver:7002`.

You can configure SSL/TLS as either one-way or two-way.

- With one-way SSL/TLS, the server is required to present a certificate to the client but the client is not required to present a certificate to the server.
- With two-way SSL/TLS, the server presents a certificate to the client and the client presents a certificate to the server. WebLogic Server can be configured to require clients to submit valid and trusted certificates before completing the SSL/TLS handshake.

For more information on how SSL/TLS is used in WebLogic Server, see Configuring SSL in *Administering Security for Oracle WebLogic Server*.

Note:

Although the terms TLS, SSL, and SSL/TLS are used interchangeably throughout WebLogic Server documentation, it is expected and encouraged that you use a currently supported version of TLS, not SSL, to secure communication in WebLogic Server.

Set Up SSL/TLS

Establish secure communication between multiple applications connecting over a network connection.

Configure the identity and trust keystores for WebLogic Server. See Configure Keystores.

1. In the **Edit Tree**, go to **Environment**, then **Servers**, then *myServer*.
2. Click the **Security** tab, then the **SSL** subtab.
3. Set SSL/TLS attributes for the private key alias and password.
4. In the **Server Private Key Alias** field, enter the keystore attribute that defines the string alias used to store and retrieve the server's private key.
5. In the **Server Private Key Pass Phrase** field, enter the keystore attribute that defines the passphrase used to retrieve the server's private key.
6. Click **Show Advanced Fields**.
7. Turn off the **Disable Hostname Verification** option.

See Enable Host Name Verification.

8. In the **Export Key Lifespan** field, indicate the number of times WebLogic Server can use an exportable key between a domestic server and an exportable client before generating a new key. The more secure you want WebLogic Server to be, the fewer times the key should be used before generating a new key.
9. For both the **Inbound Certificate Validation** and the **Outbound Certificate Validation** drop-down lists, select a validation method.
   - **Builtin SSL Validation Only**: Uses the built-in trusted CA-based validation. This is the default.
   - **Builtin SSL Validation and Cert Path Validators**: Uses the built-in trusted CA-based validation and uses configured CertPathValidator providers to perform extra validation.

   For more information, see Using Certificate Lookup and Validation Providers in *Administering Security for Oracle WebLogic Server*.
10. Optional: Enable two-way SSL/TLS.

    By default, WebLogic Server is configured to use one-way SSL/TLS, where the server passes its identity to the client. In a two-way SSL/TLS connection, the client verifies the identity of the server and then passes its identity certificate to the server. The server then validates the identity certificate of the client before completing the SSL/TLS handshake. The server determines whether or not two-way SSL/TLS is used.
    Before enabling two-way SSL/TLS, ensure that the trust keystore for the server includes the certificate for the trusted certificate authority that signed the certificate for the client. See Identity and Trust.
    a. Turn on the **Two Way SSL Enabled** option.
    b. Choose whether to enable the **Client Certificate Enforced** option. When enabled, a client is required to present a certificate. If a certificate is not presented, the SSL/TLS connection is terminated. When disabled, the SSL/TLS connection continues even if a certificate is not presented by the client.

    You may also want to force all administration clients use two way SSL/TLS. If an administration client attempts to connect to a server on a channel that does not require two-way SSL/TLS, then the connection is rejected.
    a. Under **Environment**, then **Domain**, click the **Security** tab and enable **Show Advanced Fields**.
    b. Turn on the **Require 2 way TLS for Admin Clients** option.
11. Click **Save**.
12. Repeat on all applicable servers.

Enable Host Name Verification

A host name verifier ensures the host name in the URL to which the client connects matches the host name in the digital certificate that the server sends back as part of the SSL/TLS connection. If the host name in the certificate matches the local machine’s host name, host name verification passes if the URL specifies `localhost`, `127.0.0.1`, or the default IP address of the local machine.

A host name verifier is useful when a SSL/TLS client (or a WebLogic Server acting as an SSL/TLS client) connects to an application server on a remote host. Host name verification is performed only by a SSL/TLS client. By default, WebLogic Server has host name verification enabled and we recommend keeping it enabled for production environments.

> **Note:**
>
> The following steps only apply when a WebLogic Server instance is acting as an SSL/TLS client. Stand alone SSL/TLS clients specify the use of host name verification using command-line arguments or the API.

1. In the **Edit Tree**, go to **Environment**, then **Servers**, then *myServer*.
2. Click the **Security** tab, then **SSL**.
3. Click **Show Advanced Fields**.
4. Turn off **Disable Hostname Verification**.
5. Choose a hostname verifier from the **Hostname Verifier** drop-down list.

   As of WebLogic Server 14.1.1.0.0, the default verifier is the wildcard verifier.
6. Optional: If you want to use a custom verifier, follow these steps.

   > **Note:**
   >
   > If you write a custom host name verifier, the class that implements the host name verifier must be specified in the `CLASSPATH` of WebLogic Server (when acting as an SSL/TLS client) or a stand alone SSL/TLS client.
   >
   > When you use stand alone SSL/TLS clients, a custom host name verifier must be specified on the command line using the following argument or through the API: `-Dweblogic.security.SSL.HostnameVerifier=`*`classname`* where *classname* specifies the implementation of the `weblogic.security.SSL.HostnameVerifier` interface.

   a. Create a custom host name verifier as described in Using a Custom Hostname Verifier in *Developing Applications with the WebLogic Security Service*.
   b. From the **Hostname Verifier** drop-down list, select **Custom Verifier**.
   c. In the **Custom Hostname Verifier** field, enter the name of the implementation of the `weblogic.security.SSL.HostnameVerifier` interface.
7. Click **Save**.

All the server SSL/TLS attributes are dynamic. When modified using WebLogic Remote Console, they cause the corresponding SSL/TLS server or channel SSL/TLS server to restart and use the new settings for new connections. Old connections will continue to run with the old configuration.

Restart SSL/TLS

All server SSL/TLS attributes are dynamic. You can modify these attributes and then apply your changes without rebooting WebLogic Server. Instead, only the corresponding SSL/TLS server or channel SSL/TLS server will restart and use the new settings for new connections. Existing connections continue to run with the old configuration.

1. If you haven't done so already, turn on automatic realm restart in the default security realm. See Enable Automatic Realm Restart.

If automatic realm restart is not enabled, old connections will continue to run with the old configuration and you must restart WebLogic Server after restarting SSL/TLS to ensure that all the SSL/TLS connections exist according to the specified configuration.

2. In the **Monitoring Tree**, go to **Environment**, then **Servers**.
3. Select the checkbox for each server where you want to restart SSL/TLS.
4. Click **Restart SSL** to restart the SSL/TLS listen sockets to apply changes to your keystore.

Specify Cipher Suites

Configure cipher suites supported by WebLogic Server.

1. In the **Edit Tree**, go to **Environment**, then **Servers**, then *myServer*.
2. Click the **Security** tab, then the **SSL** subtab.
3. Click **Show Advanced Fields**.
4. In the **Cipher Suites** field, enter the cipher suites that you want to support on this server.
5. Optional: In the **Excluded Cipher Suites** field, enter any cipher suites that you want to block on this server.
6. Click **Save**.
7. Repeat for other servers as necessary.

Enable Cross Domain Security

Cross domain security establishes trust between two WebLogic Server domains by using a credential mapper to configure communication between the WebLogic Server domains.

For more information, see Configuring Cross-Domain Security in *Administering Security for Oracle WebLogic Server*.

1. In the **Edit Tree**, go to **Environment**, then **Domain**.
2. On the **Security** tab, turn on **Cross Domain Security Enabled**.

   To reduce the impact on performance, WebLogic Server caches the authenticated subject. If you want to modify the cache settings for your environment, click **Show Advanced Fields** and change the following settings:

   - To disable the cache, turn off **Cross Domain Security Cache Enabled**.
   - To change how often the cache is cleared, update the **Cross Domain Security Cache TTL** value (in seconds).
3. Click **Save** and then commit your changes.
4. Create a user for cross domain security and assign it to the `CrossDomainConnectors` group. See Create a User.

   For more information on, see Configuring Cross-Domain Users in *Administering Security for Oracle WebLogic Server*.
5. Configure a cross-domain security credential mapping for the cross-domain security user.

   For information on credential mapping, see Configure a Credential Mapping Provider.

   a. In the **Security Data Tree**, go to **Realms**, then *myRealm*, then **Credential Mappers**, then *myCredentialMapper*, then **Remote Resources**.

   b. Click **New**.

c. Turn on the **Use cross-domain protocol** option.

d. In the **Remote Domain** field, enter the name of the remote domain that needs to interact with the local domain.

e. In the **Remote User** field, enter the username of the user configured in the remote domain that is authorized to interact with the local domain.

f. In the **Remote Password** field, enter the password of the user configured in the remote domain that is authorized to interact with the local domain.

g. Click **Create**.

Enable Global Trust Between Domains

If you want multiple WebLogic Server domains to interoperate, you can specify the same domain credential for each of the domains. Typically, the domain credential is randomly generated by default and therefore, no two domains have the same domain credential.

When this feature is enabled, identity is passed between WebLogic Server domains over an RMI connection without requiring authentication in the second domain. When inter-domain trust is enabled, transactions can commit across domains. A trust relationship is established when the domain credential for one domain matches the domain credential for another domain. For more information, see Enabling Global Trust in *Administering Security for Oracle WebLogic Server*.

Instead of enabling global trust between domains, consider using the CrossDomainConnector role, as described in Enable Cross Domain Security.

1. In the **Edit Tree**, go to **Environment**, then **Domain**.
2. On the **Security** tab, click **Show Advanced Fields**.
3. In the **Security Credential** field, enter a new password for the domain. Choose a strong password.
4. Click **Save**.
5. Perform the same procedure in each domain for which you want to enable global trust, entering the exact same password.

Configure Connection Filtering

Connection filters allow you to deny access at the network level. They can be used to protect server resources on individual servers, server clusters, or an entire internal network or intranet.

For more information on connection filters, see Using Connection Filters in *Administering Security for Oracle WebLogic Server*.

1. In the **Edit Tree**, go to **Environment**, then **Domain**.
2. On the **Security** tab, select the **Filter** subtab.
3. Turn on the **Connection Logger Enabled** option to enable the logging of accepted messages. The Connection Logger logs successful connections and connection data in the server. This information can be used to debug problems relating to server connections.
4. In the **Connection Filter** field, specify the connection filter class to be used in the domain.
   - To configure the default connection filter, specify `weblogic.security.net.ConnectionFilterImpl`.

- To configure a custom connection filter, specify the class that implements the network connection filter. This class must also be present in the class path for WebLogic Server.

5. In the **Connection Filter Rules** field, enter the syntax for the connection filter rules.

   For more information about connection filter rules, see Guidelines for Writing Connection Filter Rules in *Developing Applications with the WebLogic Security Service*.
6. Optional: If you want WebLogic Server to ignore errors in filter rules during server startup, turn on the **Connection Filter Ignore Rule Errors** option.
7. Click **Save**.

Create an Allowlist for JEP 290 Filtering

To improve security, WebLogic Server uses the JDK JEP 290 mechanism to filter incoming serialized Java objects and limit the classes that can be deserialized.

For more information, see Using JEP 290 in Oracle WebLogic Server in *Administering Security for Oracle WebLogic Server*.

When using the allowlist model, WebLogic Server and the customer define a list of the acceptable classes and packages that are allowed to be deserialized, and block all other classes.

Note:

WebLogic Server also supports using blocklists for JEP 290 filtering. For instructions on using blocklists, see How WebLogic Server Uses JEP 290 Blocklists and Allowlists in *Administering Security for Oracle WebLogic Server*.

1. Configure the domain to record all of the classes and packages used in both WebLogic Server and customer application deserialization.
   a. In the **Edit Tree**, go to **Environment**, then **Domain**.
   b. On the **Security** tab, select the **Allow List** subtab.
   c. Turn on the **Recording Enabled** option.

      When recording is enabled, all classes are allowed during deserialization except for the classes specified in the blocklist.
   d. Save and commit the change.
2. Run a full set of tests to ensure that the recorded allowlist configuration file provides appropriate coverage of all packages and classes that must be allowed in order for your application to run successfully. When deserialization occurs, each class is recorded in `*DOMAIN_HOME*/config/security/jep290-recorded.serial.properties`.

   A sample `jep290-recorded.serial.properties` is shown below:

```
Wed May 19 23:55:13 UTC 2021
weblogic.oif.serialFilter=\
    com.company1.common.collections.objs.*;\
    com.company1.common.tools.Calculator;\
    com.company2.shared.tools.Converter
weblogic.oif.serialGlobalFilter=\
    com.company1.common.lists.AList;\
    com.company1.common.tools.Calculator;\
    com.company2.shared.tools.*
```

3. Turn off the **Recording Enabled** option. Save and commit the change.
4. Configure the domain to use allowlists.
   a. In the **Edit Tree**, go to **Environment**, then **Domain**. Click the **Security** tab, then the **Allow List** subtab. From the **Violation Action** drop-down list, select the appropriate setting.
      - `IGNORE` - Ignore the allowlist and use the blocklists. If any class found during deserialization is present in the blocklist, the class is blocked from being deserialized.
      - `LOG` - Log a message if a violation occurs but allow the class unless it is listed in the blocklist.
      - `DENY` - Block everything except the classes specified in the allowlist, and log a message when a class is blocked.

      Note:

      You can also set the `AllowListViolationAction` on a channel using the network access point. Doing so allows you to use an allowlist on untrusted external channels and a blocklist on internal trusted channels.

   b. Click **Save** and commit your changes.
5. By default, the directory containing the allowlist configuration file is polled every 60 seconds. If you want to change the default polling interval, do the following:
   a. In the **Edit Tree**, go to **Environment**, then **Domain**.
   b. Click the **Security** tab, then the **Allow List** subtab.
   c. In the **Serial Profile Polling Interval** field, enter the new interval.
   d. Click **Save**.
6. Configure your production domain to use allowlists by copying the recorded allowlist configuration file that you just created to the *`DOMAIN_HOME`*`/config/security` directory of the production domain.

   Note:

   Oracle recommends that you run your production domain with `AllowListViolationAction` set to `Log` for some period of time to ensure that all classes and packages were recorded.

It's important to maintain the accuracy of the allowlist configuration file. Whenever a new application is deployed to the domain, or a new version of the application is deployed, you should repeat this entire process, to recreate the allowlist or verify the allowlist with the new application to ensure that all packages and classes required by the new or updated application are included in the allowlist.

Configure SAML 2.0 Services: Main Steps

You can configure your WebLogic Server instance as either a Service Provider or Identity Provider.

For more information on using SAML 2.0 with WebLogic Server, see Configuring SAML 2.0 Services in *Administering Security for Oracle WebLogic Server*.

Note:

You cannot configure SAML 1.1 services using WebLogic Remote Console.

1. If you plan to have SAML 2.0 services running in more than one WebLogic Server instance in the domain, then create a domain in which the RDBMS security store is configured. See Use the RDBMS Security Store.

   The RDBMS security store is required by SAML 2.0 security providers in production environments so that the data they manage can be synchronized across all of the WebLogic Server instances that share that data.

   Oracle does not recommend upgrading an existing domain in place to use the RDBMS security store. If you want to use the RDBMS security store, then you should configure the RDBMS security store at the time of domain creation. If you have an existing domain with which you want to use the RDBMS security store, then create the new domain and migrate your existing security realm to it.

   See Managing the RDBMS Security Store in *Administering Security for Oracle WebLogic Server*.
2. Review the considerations described in Web Application Deployment Considerations for SAML 2.0 in *Administering Security for Oracle WebLogic Server*.
3. Configure SAML 2.0 general services. Apply the same configuration settings to each server instance. See Configure SAML 2.0 General Services.
4. If you are configuring a SAML 2.0 Identity Provider site,
   a. Create an instance of the SAML 2.0 Credential Mapping provider in the security realm. See Configure a Credential Mapping Provider.
   b. Configure SAML 2.0 Identity Provider services. Apply the same configuration settings to each server instance. See Configure SAML 2.0 Identity Provider Services.
   c. Publish the metadata file describing your site, and manually distribute it to your Service Provider partners. See Publish SAML Metadata.
   d. Create and configure your Service Provider partners. See Create a SAML 2.0 Web Service Service Provider Partner or Create a SAML 2.0 Web Single Sign-On Service Provider Partner.
5. If you are configuring a SAML 2.0 Service Provider site,
   a. Create an instance of the SAML 2.0 Identity Assertion provider in the security realm. See Configure an Authentication or Identity Assertion Provider
   b. If you are allowing virtual users to log in using SAML, create an instance of the SAML Authentication provider. See Configuring the SAML Authentication Provider in *Administering Security for Oracle WebLogic Server*.

c. Configure SAML 2.0 Service Provider services. Apply the same configuration settings to each server instance. See Configure SAML 2.0 Service Provider Services
d. Publish the metadata file describing your site, and manually distribute it to your Identity Provider partners. See Publish SAML Metadata
e. Create and configure your Identity Provider partners. See Create a SAML 2.0 Web Service Identity Provider Partner or Create a SAML 2.0 Web Single Sign-On Identity Provider Partner.

Configure SAML 2.0 General Services

Whether you configure a WebLogic Server instance as a SAML 2.0 Identity Provider or as a SAML 2.0 Service Provider, you must configure the server's general SAML 2.0 services.

For more information, see Configuring SAML 2.0 General Services in *Administering Security for Oracle WebLogic Server*.

1. Review the general steps for configuring SAML 2.0 services as described in Configure SAML 2.0 Services: Main Steps.
2. In the **Edit Tree**, go to **Environment**, then **Servers**.
3. Select the server instance where you want to configure SAML general services.
4. Click the **Security** tab, then the **SAML 2.0 General** subtab.
5. Turn on the **Replicated Cache** option to use the persistent cache for storing SAML 2.0 artifacts.

   This option is required if you are configuring SAML 2.0 services in two or more WebLogic Server instances in your domain. If you are configuring SAML 2.0 services in a cluster, you must enable this option in each Managed Server instance individually.
6. Modify the settings as necessary for your environment. For descriptions of the fields and when you might want to configure them, see About SAML 2.0 General Services in *Administering Security for Oracle WebLogic Server*.
7. Click **Save**.
8. Repeat for the rest of the servers in domain. Make sure you apply the exact same configuration settings to all servers.
9. Publish the metadata file. See Publish SAML Metadata.

Next, configure the server as an Identity Provider or as a Service Provider. See Configure SAML 2.0 Identity Provider Services or Configure SAML 2.0 Service Provider Services, respectively.

Publish SAML Metadata

The SAML metadata file contains the information about this site's SAML 2.0 services. Share this file with your federated partners to facilitate SAML 2.0 web single sign-on.

1. In the **Monitoring Tree**, go to **Environment**, then **Servers**, then *myServer*.
2. On the **SAML 2.0** tab, click **Publish Metadata**.
3. In the **Publish Metadata** dialog box, specify a file path location to save the metadata file. The file path must be relative to the Administration Server.
4. Click **Done**.

5. Distribute the metadata file to your federated partners. For distribution recommendations, see Publishing and Distributing the Metadata File in *Administering Security for Oracle WebLogic Server*.

Configure SAML 2.0 Identity Provider Services

A SAML 2.0 Identity Provider creates, maintains, and manages identity information for principals, and provides principal authentication to other Service Provider partners within a federation by generating SAML 2.0 assertions for those partners.

1. Perform the SAML 2.0 general services configuration as described in Configure SAML 2.0 General Services.
2. Configure a SAML 2.0 Credential Mapping provider if you haven't done so already. See Configure a Credential Mapping Provider.
3. In the **Edit Tree**, go to **Environment**, then **Servers**.
4. Select the server instance where you want to perform SAML configuration for servers in the role of Identity Provider.
5. Click the **Security** tab, then the **SAML 2.0 Identity Provider** subtab.
6. Turn on the **Enabled** option to activate this server's SAML 2.0 services in the role of Identity Provider.
7. Modify the settings as necessary for your environment. For descriptions of the attributes and when you might want to configure them, see Configure SAML 2.0 Identity Provider Services in *Administering Security for Oracle WebLogic Server*.
8. Click **Save**.
9. Repeat for the rest of the servers. Make sure you apply the same configuration settings to all servers.

Create and configure your Service Provider partners. See Create a SAML 2.0 Web Single Sign-On Service Provider Partner or Create a SAML 2.0 Web Service Service Provider Partner.

Coordinate with your federated partners to ensure that the SAML bindings you have enabled for this SAML authority, as well as your requirements for signed documents, are compatible with your partners.

Create a SAML 2.0 Web Service Service Provider Partner

A SAML 2.0 Service Provider partner is an entity that consumes the SAML 2.0 assertions generated by the Identity Provider site.

1. Obtain, using a trusted and secure mechanism, the metadata file from your federated partner that describes the partner, the binding support, certificates and keys, and so on. See Obtain Your Service Provider Partner's Metadata File in *Administering Security for Oracle WebLogic Server*.
2. In the **Security Data Tree**, go to **Realms**, then *myRealm*, then **Credential Mappers** and select the SAML 2.0 credential mapping provider that you configured for the server.
3. Click the **Partners** node.
4. Click **New**.
5. Enter a name for the Service Provider partner.
6. From the **Type** drop-down list, select **Web Service Service Partner**.

7. Click **Create**.
8. Turn on the **Enabled** option to enable interactions between this server and this Service Provider partner.
9. Modify the settings as necessary for your environment. For descriptions of the settings and when you might want to configure them, see Create and Configure Web Single Sign-On Service Provider Partners in *Administering Security for Oracle WebLogic Server*.
10. Click **Save**.

Create a SAML 2.0 Web Single Sign-On Service Provider Partner

A SAML 2.0 Service Provider partner is an entity that consumes the SAML 2.0 assertions generated by the Identity Provider site.

1. Obtain, using a trusted and secure mechanism, the metadata file from your federated partner that describes the partner, the binding support, certificates and keys, and so on. See Obtain Your Service Provider Partner's Metadata File in *Administering Security for Oracle WebLogic Server*.
2. In the **Security Data Tree**, go to **Realms**, then *myRealm*, then **Credential Mappers** and select the SAML 2.0 credential mapping provider that you configured for the server.
3. Click the **Partners** node.
4. Click **New**.
5. Enter a name for the Service Provider partner.
6. From the **Type** drop-down list, select **Web Single Sign-On Service Partner**.
7. In the **Meta Data File Name** field, enter the file path to the metadata partner file you received from your federated partner.
8. Click **Create**.
9. Turn on the **Enabled** option to enable interactions between this server and this Service Provider partner.
10. Modify the settings as necessary for your environment. For descriptions of the settings and when you might want to configure them, see Create and Configure Web Single Sign-On Service Provider Partners in *Administering Security for Oracle WebLogic Server*.
11. Click **Save**.

Configure SAML 2.0 Service Provider Services

A Service Provider is a SAML authority that can receive SAML assertions and extract identity information from those assertions. The identity information can then be mapped to local Subjects, and optionally groups as well, that can be authenticated.

1. Perform the SAML 2.0 general services configuration as described in Configure SAML 2.0 General Services.
2. If you haven't done so already, configure a SAML 2.0 Identity Assertion provider . See Configure an Authentication or Identity Assertion Provider.
3. In the **Edit Tree**, go to **Environment**, then **Servers**.
4. Select the server instance where you want to perform SAML configuration.
5. Click the **Security** tab, then the **SAML 2.0 Service Provider** subtab.

6. Turn on the **Enabled** option to activate this server's SAML 2.0 services in the role of Service Provider.
7. Modify the settings as necessary for your environment. For descriptions of the attributes and when you might want to configure them, see Configure SAML 2.0 Service Provider Services in *Administering Security for Oracle WebLogic Server*.
8. Click **Save**.
9. Repeat for the rest of the servers. Make sure you apply the same configuration settings to all servers.

Create and configure your Identity Provider partners. See Create a SAML 2.0 Web Single Sign-On Identity Provider Partner or Create a SAML 2.0 Web Service Identity Provider Partner.

Create a SAML 2.0 Web Service Identity Provider Partner

A SAML 2.0 Identity Provider partner is an entity that generates SAML 2.0 assertions consumed by the Service Provider site.

1. Obtain, using a trusted and secure mechanism, the metadata file from your federated partner that describes the partner, the binding support, certificates and keys, and so on. See Obtain Your Identity Provider Partner's Metadata File in *Administering Security for Oracle WebLogic Server*.
2. In the **Security Data Tree**, go to **Realms**, then *myRealm*, then **Authentication Providers** and select the SAML 2.0 Identity Assertion provider that you configured.
3. Click **Partners** node.
4. Click **New**.
5. Enter a name for the Identity provider partner.
6. From the **Type** drop-down list, select **Web Service Identity Partner**.
7. Click **Create**.
8. Turn on the **Enabled** option to enable interactions between this server and this Identity Provider partner.
9. Modify the settings as necessary for your environment. For descriptions of the settings and when you might want to configure them, see Create and Configure Web Single Sign-On Identity Provider Partners in *Administering Security for Oracle WebLogic Server*.
10. Click **Save**.
11. Click the **Assertion Signing Certificate** tab to configure the Identity Provider partner's assertion signing certificate.
    a. Coordinate with your partner to obtain the Assertion Signing Certificate in a secure manner. For more information, see Using Security Assertion Markup Language (SAML) Tokens For Identity in *Securing WebLogic Web Services for Oracle WebLogic Server*.
    b. Click **Import Certificate from File** and enter the file path to the `.pem` or `.der` file containing the X.509 certificate.
    c. Click **Done**.

       WebLogic Remote Console will populate the page with information pulled from the imported certificate.

Create a SAML 2.0 Web Single Sign-On Identity Provider Partner

A SAML 2.0 Identity Provider partner is an entity that generates SAML 2.0 assertions consumed by the Service Provider site.

1. Obtain, using a trusted and secure mechanism, the metadata file from your federated partner that describes the partner, the binding support, certificates and keys, and so on. See Obtain Your Identity Provider Partner's Metadata File in *Administering Security for Oracle WebLogic Server*.
2. In the **Security Data Tree**, go to **Realms**, then *myRealm*, then **Authentication Providers** and select the SAML 2.0 Identity Assertion provider that you configured for the server.
3. Click the **Partners** node.
4. Click **New**.
5. Enter a name for the Identity Provider partner.
6. From the **Type** drop-down list, select **Web Single Sign-On Identity Partner**.
7. In the **Meta Data File Name** field, enter the file path to the metadata partner file you received from your federated partner.
8. Click **Create**.
9. Modify the settings as necessary for your environment. For descriptions of the settings and when you might want to configure them, see Create and Configure Web Single Sign-On Identity Provider Partners in *Administering Security for Oracle WebLogic Server*.
10. Click **Save**.

Use the RDBMS Security Store

You can use an external RDBMS as a data store for the authorization, role mapping, credential mapping, and certificate registry providers. This data store, called the RDBMS security store, is strongly recommended when using SAML 2.0 services in two or more WebLogic Server instances in a domain.

If you plan to use the RDBMS security store, you should set it up when you first create the domain. If you have an existing domain, then you should create a new domain, configure it for RDBMS, and then migrate security data from the existing domain over to the new domain. We do not recommend trying to enable the RDBMS security store on an existing domain.

For information on using the RDBMS security store in WebLogic Server, see Managing the RDBMS Security Store in *Administering Security for Oracle WebLogic Server*.

For a list of security providers affected by the RDBMS, see Security Providers that Use the RDBMS Security Store in *Administering Security for Oracle WebLogic Server*.

For a list of the specific RDBMS systems supported in this release of WebLogic Server for use as an RDBMS security store, see Oracle Fusion Middleware Supported System Configurations.

1. Create a new WebLogic domain using either the Domain Configuration wizard or WLST Offline. See Creating a WebLogic Domain in *Creating WebLogic Domains Using the Configuration Wizard* or Creating WebLogic Domains Using WLST Offline in *Understanding the WebLogic Scripting Tool*, respectively.

> **Note:**
>
> Do not start the domain at this time.

2. Enable the RDBMS security store using WLST Offline. See Use WLST Offline to Create the RDBMS Security Store in *Administering Security for Oracle WebLogic Server*.
3. Create the RDBMS tables in your data store. The WebLogic Server installation directory includes a set of scripts for each supported RDBMS system.

   Typically this step is performed by the database administrator. See Create RDBMS Tables in the Security Datastore in *Administering Security for Oracle WebLogic Server* for a description of the scripts, their locations in the product installation directory, and instructions for running them.
4. Start the domain.

Configure the RDBMS Security Store

You can update the RDBMS security store settings. However, you should avoid modifying the database settings of the RDBMS security store after it has been created.

1. In the **Edit Tree**, go to **Security**, then **Realms**, then *myRealm*, then **RDBMS Security Store**.
2. Click **New**.
3. Enter values for the **Name**, **Connection URL**, **Driver Name**, and **Username** fields.

   The database name, type, and user credentials must match the values you set when you created the domain.
4. Click **Create**.
5. Specify the appropriate settings for JNDI and JMS so that the RDBMS security store can cache database information in memory correctly. If the RDBMS is running in more than one JVM, for example, the domain has multiple servers, or other Oracle products are sharing the same RDBMS store with the new domain, these caches must be synchronized to ensure the integrity of the security data. To configure server synchronization:
   a. Specify a JNDI user name and password. This can be any valid user in the security realm who has access to JNDI.
   b. Create a JMS topic. You can reuse an existing one. See Configure Resources for JMS System Modules.

   > **Note:**
   >
   > Failure to configure JMS actions in a multiserver domain in which the RDBMS security store is configured may result in a security vulnerability.
6. Click **Save**.

If the JMS topic with which the RDBMS security store is configured goes down, see Managing the RDBMS Security Store in *Administering Security for Oracle WebLogic Server* for important information about restoring it.

7 Deploying Applications

You can use WebLogic Remote Console to manage the deployment process of applications to WebLogic Server.

For general information on the application deployment process, see Understanding WebLogic Server Deployment in *Deploying Applications to Oracle WebLogic Server*.

Supported Deployment Units

A deployment unit refers to a Java EE application (an enterprise application or Web application) or a standalone Java EE module (such as an EJB or resource adapter) that has been organized according to the Java EE specification and can be deployed to WebLogic Server.

The following deployment units are supported:

- Enterprise Application
- Web Application
- Enterprise JavaBean
- Web Service
- Java EE Library
- Optional Package
- JDBC, JMS, and WLDF Modules
- Client Application Archive

For more information on each deployment unit, see Supported Deployment Units in *Deploying Applications to Oracle WebLogic Server*

Install an Application

An application can be installed as an archived EAR file or as an exploded directory. Installing an application makes its physical file or directory known to WebLogic Server.

This procedure applies to all of the deployment units listed in Supported Deployment Units, with the exception of Java EE libraries. For Java EE libraries, see Install a Java EE Library instead.

1. In the **Edit Tree**, go to **Deployments**, then **App Deployments**.
2. Select **New**.
3. Enter a name for the application.
4. Select the servers and clusters to which you want to deploy the application.
5. Make the archive file or exploded directory known to the Administration Server.
   - If the application is on your file system and you need to upload it to the Administration Server, enable the **Upload** option. Then, beside **Source**, click **Choose File** to browse to the application's location on your system.

- If the application is already in the file system of the Administration Server, disable the **Upload** option. Then, in the **Source Path** field, enter the file path to the application.

6. Optional: Add a deployment plan, choose another staging mode, or set application behavior at deployment.
7. Click **Create**.

You can view the status of running deployment tasks on the **Monitoring Tree**: **Deployments**: **Deployment Tasks** page.
Your new application appears under the App Deployment node. You can make additional changes to the application on this page.

You must start an application before it can process client requests.

Start an Application

You must start an application to make it available to WebLogic Server clients.

When you start an application, you can make it immediately available to clients, or you can start it in Administration Mode first to ensure that it is working as expected. Starting in Administration Mode allows you to perform final sanity checking of the distributed application directly in the production environment without disrupting clients.

1. In the **Monitoring Tree**, go to **Deployments**, then **Application Management**.
2. Select the application that you want to start.
3. Click **Start** and choose either:
   - **Servicing all requests**: to make the application immediately available to all clients.
   - **Servicing only administration request**: to make the application available in Administration Mode only.

Stop an Application

When you stop an application, you prevent clients from accessing it. You can choose whether no clients can use it, or transition it to Administration Mode so that only administrative tasks can be performed.

Stopping an application does not remove its source files from the server; you can redeploy a stopped application to make it available to WebLogic Server clients again.

1. In the **Monitoring Tree**, go to **Deployments**, then **Application Management**.
2. Select the application that you want to stop.
3. Click **Stop** and choose one of:
   - **When work completes**: to allow the application to finish its work and for all currently connected users to disconnect.
   - **Force stop now**: to stop the application immediately, regardless of the work that is being performed and the users that are connected.
   - **Servicing non-administration requests**: to stop the application once all its work has finished, and then put the application in Administration Mode so it remains accessible for administrative purposes.

Specify Deployment Properties

After you deploy an application, you can configure additional settings to meet the needs of your environment.

1. If you haven't already, create a deployment plan for the application. You can view the default configuration settings without creating a plan but they are read-only until you create a deployment plan. For more information, see Deployment Plans.

   > **Note:**
   >
   > You cannot create a deployment plan for applications that were auto-deployed.

   a. In the **Monitoring Tree**, go to **Deployments**, then **Application Management**, then *myApp*.
   b. Click **Create Plan**.
   c. In the **Plan Path** field, enter a file path for the new deployment plan. Deployment plans must be in XML format and should be called `plan.xml`.

      If possible, you should create the new deployment plan for a single application within a `plan/` subdirectory of the application's root directory.
   d. Click **Done**. WebLogic Server will create a basic deployment plan.

      You can see the deployment plan for an application under **Deployments**: **Application Management**: *myApplication*: **Deployment Plan (Advanced)**.
2. Go to **Deployments**, then **Application Management**, then *myApplication*, then **Configuration**. Explore the Configuration node and its children to see the available deployment configuration options.

   > **Note:**
   >
   > The contents of the Configuration node and its children differ based on application type. For example, web applications include settings for container descriptors while resource adapters include settings for outbound connection pools.
3. Click **Save** as you make changes. Any changes that you make to the Configuration node and its children are automatically reflected in the deployment plan of the application.
4. Update and possibly redeploy your application to apply your changes. See Update or Redeploy an Application for instructions.

Deployment Plans

Use a deployment plan to specify deployment property values for an application.

A deployment plan is an optional document that works with or overrides your application’s deployment descriptors to configure an application for deployment to a specific WebLogic Server environment. Deployment plans are written in XML.

See Understanding WebLogic Server Deployment Plans in *Deploying Applications to Oracle WebLogic Server*.

> **Note:**
>
> You cannot create a deployment plan for applications that were auto-deployed.

For deployment plan descriptions and examples, see Understanding Deployment Plan Contents in *Deploying Applications to Oracle WebLogic Server*.

Generally, if you need to modify the deployment properties of an application, you should use the Configuration node (as described in Specify Deployment Properties), instead of manually editing the deployment plan.

If your changes in the deployment plan include non-dynamic changes, you must redeploy the application to propagate the changes from the deployment plan to the application.

Modify a Deployment Plan

You can manually update a deployment plan with new deployment instructions for an application.

For more information, see Manually Customizing the Deployment Plan in *Deploying Applications to Oracle WebLogic Server*.

1. In the **Monitoring Tree**, go to **Deployments**, then **Application Management**, then *myApplication*, then **Deployment Plan (Advanced)**.
2. You can edit the deployment plan as a whole or the individual variable assignments within a deployment plan:
   - To edit a deployment plan document:
     a. Select the **Deployment Plan** tab.
     b. Edit the deployment plan directly within the text box field. You can also copy the plan contents into an external text editor, make your edits and then paste the changed data back into WebLogic Remote Console.
     c. Click **Save**.
   - To edit individual variable assignments:
     a. Select the **Variable Assignments** tab.
     b. Select the variable assignment module that you want to edit and click **Edit**.
     c. Enter a value and, if updating an array, choose an operation.
     d. Click **Done**.
3. Update and possibly redeploy your application to apply your changes. See Update or Redeploy an Application for instructions.

Update or Redeploy an Application

After you make changes to a deployed application or its deployment plan, you need to update the application with the new instructions and possibly redeploy the application to make the changes available to WebLogic Server clients.

> **Note:**
>
> If your changes are dynamic, then you only need to update the deployment plan. However, if your changes include non-dynamic changes, you must redeploy the application to propagate the changes from the deployment plan to the application.
>
> If you try to update a deployment plan but your changes require a redeployment, WebLogic Remote Console will prompt you to redeploy the application instead.

If the application is deployed in a production environment, review Overview of Redeployment Strategies in *Deploying Applications to Oracle WebLogic Server* for guidance on limiting downtime for the application.

1. In the **Monitoring Tree**, go to **Deployments**, then **Application Management** and select the application that you want to update.
2. Click **Update/Reploy** and choose one of the options:
   - **Update - Deployment Plan on Server**: Updates the application using a new deployment plan located on the server. Use this option if all of your changes are dynamic.
   - **Update - Deployment Plan on Local Machine**: Updates the application using a new deployment plan located on the local machine and then uploads it to the Administration Server's upload directory. Use this option if all of your changes are dynamic.
   - **Redeploy - Deployment Source and Plan on Server**: Updates a deployment plan located on the server and redeploys the application. Use this option if your changes include non-dynamic changes that require the application to restart.
   - **Redeploy - Deployment Source and Plan on Local Machine**: Updates a deployment plan located on the local machine and redeploys the application. Use this option if your changes include non-dynamic changes that the application to restart.
3. Optional: If necessary for your selection, enter the file path to the new deployment plan.
4. Click **Done** to update or redeploy the application.

You can track the status of the application deployment. While still in the **Monitoring Tree**, go to **Deployments**, then **Deployment Tasks**.

Test Application Deployment

After you deploy an application, you may want to confirm that it was deployed successfully.

1. In the **Monitoring Tree**, go to **Environment**, then **Servers**, then *myServer*, then **Deployments**, then **Application Runtimes**, then *myApp*, then **Component Runtimes**, then *myServer/myApp*, then **Servlets**.

2. Construct a URL using the target server address plus the `ContextPath` and `Name` values from the Servlets table. For example, for an application deployed on the Administration Server, a test URL might look like: `http://adminserver:7001/myapp/welcome`.

> **Note:**
>
> Remember to consider any changes you made to network channels that may affect the application deployment. For example, the application may be on a different port.

3. Enter the URL into your browser. If the page loads correctly, then your application was deployed successfully.

Remove an Application from a Domain

An installed application or module remains available in the domain to stop, restart, or update until you explicitly remove it. After you delete an application or module from a domain, you must redeploy it to make it available to a WebLogic Server client again.

To temporarily make a deployed application unavailable to WebLogic Server clients, consider stopping it instead of deleting it. See Stop an Application.

1. In the **Edit Tree**, go to **Deployments**, then **App Deployments**.
2. Select the application that you want to remove and click **Delete**.

Install a Java EE Library

A Java EE library can be a standalone EJB or Web application module, multiple EJB or Web application modules packaged in an enterprise application (EAR), or a single plain JAR file that is registered with the Java EE application container upon deployment. After the library has been installed and started, other deployed modules can reference the library.

Installing a Java EE library means making its physical file or directory known to WebLogic Server. A Java EE library can be installed as an archived EAR file or as an exploded directory. After you have installed the Java EE library, other deployed modules can start using it.

1. In the **Edit Tree**, go to **Deployments**, then **Libraries**.
2. Click **New**.
3. Enter a name for the Java EE library.
4. Select the servers and clusters to which you want to deploy the Java EE library. Make sure to target all of the servers and clusters to which modules or applications that will reference the Java EE library are deployed.
5. Make the Java EE library (an archive file or exploded directory) that you want to install known to the Administration Server.
   - Enable **Upload** if the Java EE library is on your file system and you need to upload it to the Administration Server. Then, click **Choose File** to browse to the library's location on your system.
   - If the Java EE library is already in the file system of the Administration Server, then disable the **Upload** option and enter the file path to the Java EE library.

If you specify an exploded directory, WebLogic Server installs all components it finds in and below the specified directory.

> **Note:**
>
> You can install only the following types of archive files (or their corresponding exploded directories) as Java EE libraries: EJB JARs, Web application WARs, EAR files that contain EJB JARs or WARs, or plain JAR files that contain compiled classes.

6. Optional: Select a different staging mode.
7. Click **Create**.

Your new library appears under the Libraries node. You can make additional changes to the Java library on this page.

Redeploy a Java EE Library

When you update a Java EE library, WebLogic Server redeploys the archive file or exploded directory. Update a library if you have made changes to it and you want to make the changes available to the modules and applications deployed to WebLogic Server that are using the library.

1. In the **Monitoring Tree**, go to **Deployments**, then **Library Management**.
2. Select the Java EE library that you want to update.
3. Decide how you want to update the library:
   - Choose **Redeploy** if the changes to the library are already available on the Administration Server and you only want to make those changes available to the modules and applications that are using the library.
   - Choose **Upload and Redeploy** if you want to upload a new archive file or exploded directory.
     In the **Source** field, enter the path to the new archive file or exploded directory. Make sure the library's latest archive is under the Administration Server's `upload` directory.
4. Click **Done**.

Configure JASPIC for a Web Application

If using JASPIC, you can specify which Authentication Configuration provider applies to a specific Web application.

The Java Authentication Service Provider Interface for Containers (JASPIC) specification defines a service provider interface (SPI). The JASPIC SPI is used by authentication providers that implement message authentication mechanisms that can be integrated in server Web application message processing.

You can review the JASPIC specification at JSR 196: Java Authentication Service Provider Interface for Containers.

1. If you haven't done so already, configure JASPIC in the domain.
   a. Ensure JASPIC is enabled in the domain. In the **Edit Tree**, go to **Environment**, then **Domain**. On the **Security** tab, click **Show Advanced Fields** and confirm that **JASPIC Enabled** is turned on.

   b. Configure an Authentication Configuration provider.
      - To configure a WebLogic Server Authentication Configuration Provider, see Creating a WLS Authentication Configuration Provider in *Administering Security for Oracle WebLogic Server*.
      - To configure a Custom Authentication Configuration Provider, see Creating a Custom Authentication Configuration Provider in *Administering Security for Oracle WebLogic Server*.

2. In the **Monitoring Tree**, go to **Deployments**, then **Application Management**, then *myWebApp*.
3. If the application does not have a deployment plan, click **Create Plan**.

   WebLogic Server will create a basic deployment plan for the application which you can edit using the newly created **Configuration** and **Deployment Plan (Advanced)** child nodes.

   Whenever possible, use the **Configuration** node to edit deployment properties rather than editing the deployment plan directly. See Specify Deployment Properties.
4. Expand the **Configuration** node and go to the **JASPIC Container** node.

   The **JASPIC Container** node is only available for web applications.
5. Enter the name of the Authentication Configuration provider that you configured and want to apply to this web application.
6. Click **Save**.
7. Redeploy the web application or restart the server.

Create an Application Scoped Credential Mapping

Use credential mapping to control access between WebLogic resources and remote systems. WebLogic Server allows you to restrict access on a per-application basis.

1. Determine which MBean configuration data is required to form a connection between the WebLogic resource and its security data.
   a. In the **Security Data Tree**, go to **Realms**, then *myRealm*, then **Credential Mappers**.
   b. Continue until you reach the WebLogic resource that you want to prepare for credential mappings. See Identify Resources for Credential Mapping for example node paths.
   c. Click **New** and make a note of all of the required attributes (fields marked by an asterisk) for which you will need to retrieve configuration data.
   d. Click **Cancel**.
2. In the **Monitoring Tree**, go to **Deployments**, then **Application Runtime Data**, then *myApplication*, then **Component Runtimes**, then *myApplication*.
3. In the Component Runtimes table, locate the row for the application but do not click it. Instead, right-click on the application's cell for the **EIS Resource ID** column and select **Copy Cell to Clipboard**.
4. Paste the values to a text file for use in a later step.
5. Locate any additional data (as determined in step 1) that you require to identify the resource for its credential mapping. The necessary data will differ depending on the resource type.

   For example:

- EJBs require the EJB name. Under the **Deployments** node, go to **Application Runtime Data**, then *myApplication*, then **Component Runtimes**, then *myApplication*, then **EJB Runtimes**.
- Resource Adapters require the Outbound Connection Pool Instance name. Under the **Deployments** node, go to **Application Management**, then *myApplication*, then **Configuration**, then **Outbound Connection Pool Groups**, *myOutboundConnectionPoolGroup*, then **Outbound Connection Pool Instances**.

6. Form the connection between the resource and its security data, as described in Identify Resources for Credential Mapping. To fill in the fields, use the Module, Name, and any of the other requested values that you retrieved in the previous steps.

   This process will also create the first credential mapping for the application.
7. If you want to create more credential mappings, see Add a Credential Mapping.

8
Monitoring Domains

Use WebLogic Remote Console to monitor WebLogic Server and its related resources and services.

For general information on monitoring, diagnostic, and tuning tools in WebLogic Server, see Monitoring, Diagnosing, and Troubleshooting in *Understanding Oracle WebLogic Server*.

Monitor Servers

Review the servers in your domain to ensure they are functioning as expected.

1. In the **Monitoring Tree**, go to **Environment**, then **Servers**.

   Use the **Servers** table to monitor and compare properties across the different server instances in your domain.

   Use the **Customize Table** feature to rearrange the table columns to suit your needs.

2. To see more information about a server, click on its row in the table.

   Each server has its own child node under the **Servers** node. A top-level *myServer* node provides general runtime information regarding the selected server. Expand the *myServer* node to see more specific information for other areas such as scheduling or deployments.

View Node Manager Status and Logs

You can check the current status of Node Manager and use Node Manager logs to help troubleshoot problems in starting or stopping individual Managed Servers.

For more information, see Log Files in *Administering Node Manager for Oracle WebLogic Server*.

> **Note:**
>
> Node Manager must be running to view its logs.

1. In the **Monitoring Tree**, go to **Environment**, then **Node Manager Logs**.
2. Click the name of the machine whose Node Manager logs you want to view.
3. Click **Download**.

When using Hosted WebLogic Remote Console, your browser controls the location where the log files are downloaded. If you want to change the default download location, update your browser settings accordingly.

Configure Health Monitoring

A server can monitor key aspects of its subsystems and report when a subsystem is not functioning properly.

If the server is running under a Node Manager, the Node Manager can automatically restart a server with an unhealthy subsystem. Node Manager can only perform automatic monitoring and shutdown for servers that it starts.

1. In the **Edit Tree**, go to **Environment**, then **Servers**, then *myServer*.
2. On the **Health** tab, modify the options as necessary.
3. Click **Save**.

Dashboards

Use dashboards to quickly assess and filter data about your domain. You can specify criteria for WebLogic Remote Console to match against MBeans in your domain and then review the results.

Dashboards are highly customizable. Start from any node in the **Monitoring** perspective and you can use its properties to develop precise criteria that go far beyond simple true/false statements. Dashboards let you find obscure, cross-functional data that might otherwise require comparison across multiple nodes.

Dashboards are only available on the **Monitoring Tree** perspective.

Tip:

You can expand the **Dashboards** node at the bottom of the Navigation tree to see any saved or built-in dashboards.

Dashboard Filters

In dashboards, filters are the criteria that you use to curate your results. Generally, filters are based on the properties of a node.

WebLogic Remote Console restricts which filters are available based on your current node so you can focus on the criteria that is relevant to your goals. As such, the filters that are available on the Environments: Servers node are different than those available under Deployments: Application Runtime Data.

Each filter consists of a **name** (or **property**) from the domain, a **value**, and an **operator** that determines how the name and value interact with each other. A basic dashboard filter is simply `name=value`. The following example dashboard filter returns any servers that require a restart to apply configuration changes:

```
ServerRuntime.RestartRequired == true
```

Values come in three formats:

- Text

- Numeric
- Boolean (expressed as option toggles). Set the option to `On` for True, `Off` for False.

Use operators to determine how a value should be assessed. Only the operators that are applicable to a name or property appear as options - you won't see `greater than` for text values. By default, all filters are set to `Any` to provide the broadest possible search parameters.

Dashboards consist of one or more filters and filters are *cumulative* - beans must match ALL of the defined filters to be returned as a result. Therefore, when you build a dashboard, only edit the filters that are relevant to your query and leave the rest of the filters unchanged.

Built-In Dashboards

WebLogic Remote Console provides a set of pre-defined dashboards to monitor common domain statistics.

On the **Monitoring Tree**, expand the **Dashboards** node to see both custom and built-in dashboards.

You cannot edit or delete built-in dashboards. However, you can use a built-in dashboards as a starting point for another dashboard. Select a built-in dashboard and click **Copy**, then enter a new name for your custom dashboard. Edit it as you would a custom dashboard.

Create a Dashboard

You can create your own custom dashboards to monitor the state of your domain.

1. In the **Monitoring Tree**, go to the node in the Navigation Tree that most closely matches the type of content that you want to track. For example, if you want to learn about servers, open the **Environment: Servers** node.
2. Click **New Dashboard**.
3. Enter a name for your new dashboard and then begin configuring the filters that will determine the results of the dashboard. For guidance on how to build filters, see Dashboard Filters.
4. Click **Create** to generate the dashboard.
5. Optional: Click **Customize Table** to control which columns appear in the dashboard.

All of your dashboards are available in the **Dashboards** node of the Monitoring Tree.

Dashboards are not automatically refreshed with new data. You must refresh or rerun the dashboard page to see any changes to MBeans. Click **Reload** to update the results. You can also click **Auto Reload Interval** to set the dashboard to regularly reload and update the results.

Log Messages

WebLogic Server logging services provide facilities for writing, viewing, filtering, and listening for log messages. These log messages are generated by WebLogic Server instances, subsystems, and Java EE applications that run on Oracle WebLogic Server or in client JVMs.

Use WebLogic Remote Console to configure WebLogic Server Logging services. For general information on WebLogic Server logging services, see Understanding WebLogic Logging Services in *Configuring Log Files and Filtering Log Messages for Oracle WebLogic Server*.

View Logs

Each subsystem within WebLogic Server generates log messages to communicate its status. To keep a record of the messages that its subsystems generate, WebLogic Server writes the messages to log files.

The contents of log files are generated according to the logging settings that are currently defined for a server. To manage what information is sent to a log file, see Configure Logs. For more information on WebLogic logging services, see Understanding WebLogic Logging Services in *Configuring Log Files and Filtering Log Messages for Oracle WebLogic Server*.

1. In the **Monitoring Tree**, go to **Diagnostics**, then **Logs and Archives**.
2. Each row is a type of diagnostic data. Click on the row that you're interested in to view its logs.
3. Select a log file and then click **Download Logs**.

   Logs are segregated by server instance.

Configure Logs

WebLogic Server generates log files to track and communicate the status of its various subsystems. To ensure these log files remain valuable and usable, you can configure their settings.

Each WebLogic Server instance writes all messages from its subsystems and applications to a server log file that is located on the local host computer. In addition to writing messages to the server log file, each server instance forwards a subset of its messages to a domain-wide log file. By default, servers forward only messages of severity level `Notice` or higher. See Server Log Files and Domain Log Files in *Configuring Log Files and Filtering Log Messages for Oracle WebLogic Server*.

The server log records information about events such as the startup and shutdown of servers, the deployment of new applications, or the failure of one or more subsystems. The messages include information about the time and date of the event as well as the ID of the user who initiated the event. You can view and sort these server log messages to detect problems, track down the source of a fault, and track system performance.

You can also create client applications that listen for these messages and respond automatically. For example, you can create an application that listens for messages indicating a failed subsystem and sends email to a system administrator.

For more information on WebLogic logging services, see Understanding WebLogic Logging Services in *Configuring Log Files and Filtering Log Messages for Oracle WebLogic Server*.

1. In the **Edit Tree**, go to **Environment**, then **Servers**, then *myServer*.
2. Click the **Logging** tab.
3. Modify the logging settings to suit your needs.
4. Click **Save**.
5. Repeat on the rest of the servers.
6. Go to **Environment**, then **Domain**.
7. Click the **Logging** tab.
8. Modify the settings for the domain-level log to suit your needs.

9. Click **Save**.

Define Debug Settings

Specify debugging settings for WebLogic Server.

1. In the **Edit Tree**, go to **Environment**, then **Servers**, then *myServer*.
2. Click the **Debug** tab.

   The Debug tab is split into several subtabs that group related debug flags together. Use the **All** tab to see all available debug flags.
3. Enable the debug flags that you want to generate logging messages.
4. Click **Save**.

If you haven't already, you should configure general logging settings for this server. See Configure Logs.

If you create applications to run on WebLogic Server, you can configure your applications to generate messages of severity `DEBUG`. These messages are never forwarded to the domain log and are intended to contain detailed information about the operation of an application or the server.

Filter Log Messages

WebLogic Server can generate a significant amount of data in its log messages. You can filter log messages so only certain messages are published.

For example, you can filter out messages of a certain severity level, or from a particular subsystem, or according to some other specified criteria. Only the log messages that satisfy the filter criteria get published. You can also create separate filters for the messages that each server instance writes to its server log file, standard out, memory buffer, or broadcasts to the domain-wide message log.

WebLogic Server offers multiple methods for filtering log messages and you can use these methods simultaneously. Choose the methods that work for your environment. For more information, see Filtering WebLogic Server Log Messages in *Configuring Log Files and Filtering Log Messages for Oracle WebLogic Server*.

1. In the **Edit Tree**, go to **Environment**, then *myServer*.
2. Click on the **Logging** tab.
3. Click **Show Advanced Fields**.
4. If you want to change the default severity level for all loggers or packages in the logger tree, then choose a new level from the **Minimum severity to log** drop-down list.

   The default level is `Info`. For information on message severity, see Message Severity in *Configuring Log Files and Filtering Log Messages for Oracle WebLogic Server*.
5. If you want to redirect the standard out of the JVM in which the WebLogic Server instance runs to the four log message destinations (log file, standard out, domain log, and message buffer), then turn on **Redirect stdout logging enabled**. For more information, see Redirecting JVM Output in *Configuring Log Files and Filtering Log Messages for Oracle WebLogic Server*.
6. If you want to override the setting for the root Logger (as specified by the **Minimum severity to log list** option) or its closest parent node in the logger tree, then do as follows:

   a. Click the **Severity Properties** tab.
   b. In the **Logger Severity Properties** table, click **+** to add a new key-value row, then double-click the row cells to update the key and value.

   You can also use Logger severity properties to specify severity levels for packages (if using the Commons Logging API) or for individual WebLogic Server subsystem Loggers (if using the Message Catalog Logger).

   All loggers inherit the severity level of their nearest parent node in the logger hierarchy. You can specify a severity level for a given logger that is different than its nearest parent node using key-value pairs, where the key is the logger name and the value is the severity level (Info, Critical, Warning, and so on).

   See Specifying Severity Level for Loggers in *Configuring Log Files and Filtering Log Messages for Oracle WebLogic Server*.

7. If you want to control which log messages are published, you can create log filters which use custom logic to evaluate log message content and then accept or reject it for publication based on its criteria. You can apply log filters to these log message destinations: log file, standard out, and domain log broadcaster.
   a. Create a Log Filter.
   b. To apply a log filter to the log file message destination, choose your preferred severity level from the **Log File Severity Level** drop-down list, then choose a log filter from the **Log File Filter** drop-down list.
   c. To apply a log filter to the standard out message destination, choose your preferred severity level from the **Stdout Severity Level** drop-down list, then choose a log filter from the **Stdout Filter** drop-down list.
   d. To apply a log filter to the domain log broadcaster message destination, choose your preferred severity level from the **Domain Log Broadcast Severity Level** drop-down list, then choose a log filter from the **Domain Log Broadcast Filter** drop-down list.

      You can specify the size of the message buffer which stores messages that will be forwarded to the domain log. A higher value causes more messages to be stored in the buffer before the contents are forwarded to the domain log. For performance reasons, it is recommended that this value be set to 10 or higher in production mode.

   You cannot forward `DEBUG` messages to the domain log.
8. Click **Save**.

Create a Log Filter

Log filters provide control over the log messages that get published. A filter uses custom logic to evaluate the log message content, which you use to accept or reject a log message

For example, to filter out messages of a certain severity level, from a particular subsystem, or according to specified criteria. Only the log messages that satisfy the filter criteria get published. You can create separate filters for the messages that each server instance writes to its server log file, standard out, memory buffer, or broadcasts to the domain-wide message log.

1. In the **Edit Tree**, go to **Environment**, then **Log Filters**.
2. Click **New**.
3. Enter a name for the log filter and click **Create**.
4. Enter an expression in the **Filter Expression** field.

A filter expression defines simple filtering rules to limit the volume of log messages written to a particular log destination. For information on building filter expressions, see WLDF Query Language in *Configuring and Using the Diagnostics Framework for Oracle WebLogic Server*.

5. Click **Save**.

Update your logging settings to apply this log filter. See Filter Log Messages.

Rotate Log Files

Set a rotation schedule for the log files generated by WebLogic Server.

For more information, see Rotating Log Files in *Configuring Log Files and Filtering Log Messages for Oracle WebLogic Server*.

Note:

WebLogic Server sets a threshold size limit of 2,097,152 kilobytes before it forces a hard rotation to prevent excessive log file growth.

1. In the **Edit Tree**, go to **Environment**, then **Servers**, then *myServer*.
2. On the **Logging** tab, from the **Rotation Type** drop-down list, select the criteria that triggers the server to move log messages to a separate file. Depending on your choice, configure the appropriate settings.
   - **By Size** - rotates log messages when log file size reaches the specified size.
   - **By Time** - rotates log messages after a specified time interval passes.
   - **By Size or Time** - rotates log messages if the file size reaches the specified size or if the specified time interval passes, whichever occurs first.
   - **None** - log messages are not automatically rotated. You must manually erase the contents of the file when the size becomes too large.
3. If you chose **By Size** as the Rotation Type.
   - In the **Rotation file size** field, enter the file size that triggers the server to move log messages to a separate file. After the log file reaches the specified size, the next time the server checks the file size it renames the current log file by appending a 5-digit integer; for example, `SERVER_NAME.log00007`. After the server renames the file, subsequent messages accumulate in a new file named `SERVER_NAME.log`.
4. If you chose **By Time** as the Rotation Type.
   a. In the **Begin rotation time** field, enter the start time

      Use the following format: `hh:mm`, where `hh` is the hour in a 24-hour format and `mm` is the minute. At the time that you specify, the server rotates the current log file. If the time that you specify is already past, the server rotates the log file at the next scheduled interval, as specified in **Rotation Interval**.
   b. In the **Rotation Interval** field, enter the interval, in hours, at which the server saves old messages to another file.
5. If you chose **By Size or Time** as the Rotation Type, then configure the **By Size** and **By Time** options as described in steps 3 and 4.

6. Optional: If you want to limit the number of log files that the server creates to store old log messages, enable the **Limit Number of Retained Log Files** option. Then, in the **Files to Retain** field, enter the maximum number of files. If the server receives additional log messages after reaching the capacity of the last log file, it overwrites the oldest log file.
7. In the **Log file rotation directory** field, enter the directory location where the rotated log files will be stored.

   Enter an absolute pathname or a pathname that is relative to the server's root directory. By default, the rotated files are stored in the same directory where the log file is stored.
8. Optional: If you want to add a time or date stamp to the file name when the log file is rotated, then in the **Log file name** field, add `java.text.SimpleDateFormat` variables to the file name and surround each variable with percentage (%) characters.

   For example, if you enter the following value in the Log file name field: `myserver_%yyyy%_%MM%_%dd%_%hh%_%mm%.log`, the server's log file will be named `myserver_`*`yyyy`*`_`*`MM`*`_`*`dd`*`_`*`hh`*`_`*`mm`*`.log`.

   When the server instance rotates the log file, the rotated file name contains the date stamp. For example, if the server instance rotates its local log file on 4 March 2020 at 10:15 AM, the log file that contains the old log messages will be named `myserver_2020_03_04_10_15.log`*`nnnnn`*. (The current, in-use server log file retains the name `myserver_`*`yyyy`*`_`*`MM`*`_`*`dd`*`_`*`hh`*`_`*`mm`*`.log`.)

   If you do not include a time and date stamp, the rotated log files are numbered in order of creation *`SERVER_NAME`*`.log`*`nnnnn`*, where *`SERVER_NAME`* is the name configured for the log file. For example: `myserver.log00007`.
9. Click **Save**.

Enable Configuration Auditing

You can configure the Administration Server to emit audit messages that enable auditing of configuration changes in a domain.

This provides a record of changes to the configuration of any resource within a domain or invokes management operations on any resource within a domain. Configuration audit records can be saved to a log file, sent to an Auditing provider in the security realm, or both.

If you plan to audit configuration changes, then you must configure the WebLogic Auditing Provider first. See Configure an Auditing Provider.

1. In the **Edit Tree**, go to **Environment**, then **Domain**.
2. Click **Show Advanced Fields**.
3. From the **Configuration Audit Type** drop-down list, select the method to use for auditing configuration change events.
   - **None**: No audit configuration change events are written.
   - **Log**:Configuration events will be written to the server log.
   - **Audit**: Configuration events will be directed to the Security Framework and handled by the Auditing provider.
   - **LogAudit**: Configuration events will written to the server log as well as directed to the Security Framework and handled by the Auditing provider.
4. Click **Save**.

Display Thread Stacks

You can display the current stack for an active thread.

1. In the **Monitoring Tree**, go to **Environment**, then **JVM Runtime**.
2. Click the **Thread Stack Dump** tab to see an overview of the thread stack dumps for each server in the domain.
3. To see the thread stack dump for a single server, click the server instance in the table. This sends you to the **JVM Runtime** node for the selected server. Click the **Thread Stack Dump** tab.

Tuning Performance

To ensure that your WebLogic Server environment is performing optimally, you should regularly monitor its behavior and then adjust its settings accordingly.

For guidance on the performance tuning options that are available in WebLogic Server, see *Tuning Performance of Oracle WebLogic Server*.

1. In the **Edit Tree**, go to **Environment**, then **Servers**, then *myServer*.
2. On the **Advanced** tab, select the **Tuning** subtab.
3. Modify the available options as recommended based on the needs of your environment.
4. Click **Save**.
5. Repeat as needed for the rest of the servers in your domain.

9

Configuring Services

Use WebLogic Remote Console to manage services in a WebLogic Server domain.

Data Sources

You can connect WebLogic Server to databases by adding JDBC data sources to your domain. A data source is a Java EE standard method of configuring connectivity to a database.

Each WebLogic data source contains a pool of database connections. Applications look up the data source on the JNDI tree or in the local application context and then use a database connection from the pool of connections. Data sources and their connection pools provide connection management processes that help keep your system running efficiently.

You can manage the following JDBC data source types from WebLogic Remote Console:

- JDBC Generic Data Sources
- JDBC Multi Data Sources
- Active GridLink Multi Data Sources
- Universal Connection Pool (UCP) Data Sources

For more information on using data sources with WebLogic Server, see Understanding JDBC Data Sources in *Understanding Oracle WebLogic Server* and Configure Database Connectivityin *Administering JDBC Data Sources for Oracle WebLogic Server*.

JDBC Drivers

Data sources require the use of JDBC drivers to gain access to various databases. WebLogic Server comes with a default set of JDBC drivers but you can also install third-party JDBC drivers.

For the types of JDBC drivers supported in WebLogic Server, see Types of JDBC Driver in *Administering JDBC Data Sources for Oracle WebLogic Server*.

For a list of the JDBC drivers installed in WebLogic Server, see JDBC Drivers Installed with WebLogic Server in *Administering JDBC Data Sources for Oracle WebLogic Server*.

For instructions on how to install third-party JDBC drivers, see Adding Third-Party JDBC Drivers Not Installed with WebLogic Server in *Administering JDBC Data Sources for Oracle WebLogic Server*.

Create a Generic Data Source

A generic data source provides database access and database connection management. Generic data sources and their connection pools provide connection management processes that help keep your system running efficiently.

For more information on generic data sources, see Using JDBC Generic Data Sources in *Administering JDBC Data Sources for Oracle WebLogic Server*.

Note:

Generic is the term used to distinguish a simple data source from a Multi Data Source or Active GridLink data source.

If you need a JDBC driver that is not installed with WebLogic Server, you must install it before you can set up a data source. See Adding Third-Party JDBC Drivers Not Installed with WebLogic Server in *Administering JDBC Data Sources for Oracle WebLogic Server*.

1. In the **Edit Tree**, go to **Services**, then **Data Sources**.
2. Click **New**.
3. Enter a name for the new data source.

   The name cannot contain the following characters: `@ # $`.
4. In the **JNDI Names** field, enter the JNDI path to the location where this JDBC data source will be bound. Applications look up the data source on the JNDI tree by this name when reserving a connection.
5. Choose the server instances or clusters where you want to deploy the data source.
6. Select **Generic Data Source** from the **Data Source Type** drop-down list.
7. From the **Database Type** drop-down list, select the database management system (DBMS) of the database that you want to connect to.
8. From the **Database Driver** drop-down list, select a JDBC driver.
9. Enter the connection details for the database that you want to connect to:
   - **Database Name**: Enter the name of the database you want to connect to. Exact database name requirements vary by JDBC driver and by DBMS.
   - **Host Name**: Enter the DNS name or IP address of the server that hosts the database. If you are creating an Oracle GridLink service-instance connection, this must be the same for each data source in a given multi data source.
   - **Port**: Enter the port on which the database server listens for connections requests.
   - **Database User Name**: Enter the database user account name that you want to use for connections in the data source.
   - **Password**: Enter the password for the database user account.
10. Click **Create**.

Create a Multi Data Source

A multi data source is an abstraction around a group of generic data sources. It provides failover and load balancing for connection requests between two or more data sources.

For more information on multi data sources, see Using JDBC Multi Data Sources in *Administering JDBC Data Sources for Oracle WebLogic Server*.

Before you create a multi data source, you should create the generic data sources that the multi data source will manage, and ensure they are deployed to same targets as where you plan to deploy the multi data source. See Create a Generic Data Source.

1. In the **Edit Tree**, go to **Services**, then **Data Sources**.

2. Click **New**.
3. Enter a name for the new data source.

   The name cannot contain the following characters: `@ # $`.
4. In the **JNDI Names** field, enter the JNDI path to the location where this JDBC data source will be bound. Applications look up the data source on the JNDI tree by this name when reserving a connection.
5. Choose the server instances or clusters where you want to deploy the data source.

   > **Note:**
   >
   > You must deploy the multi data source and the generic data sources to the same targets.
6. Select **Multi Data Source** from the **Data Source Type** drop-down list.
7. From the **Algorithm Type** drop-down list, choose an algorithm type to determine the connection request processing for the multi data source.
   - **Failover**: The multi data source routes connection requests to the first data source in the list; if the request fails, the request is sent to the next data source in the list, and so forth.
   - **Load-Balancing**: The multi data source distributes connection requests evenly to its member data sources.
8. Specify whether this is an XA or non-XA JDBC multi data source.
   - When the **XA Driver** option is *enabled*, the multi data source will only use data sources that use an XA JDBC driver to create database connections.
   - When the **XA Driver** option is *disabled*, the multi data source will only use data sources that use a non-XA JDBC driver to create database connections

   The option you select limits the data sources that you can select as part of the multi data source in a later step. Limiting data sources by JDBC driver type enables the WebLogic Server transaction manager to properly complete or recover global transactions that use a database connection from a multi data source.
9. Choose the data sources that you want the multi data source to use to satisfy connection requests.
10. Click **Create**.

Add or Remove Data Sources in a Multi Data Source

You can add or remove data sources to a multi data source while the data source is deployed (referred to as dynamically changing the data source list in the multi data source).

The data sources that you add to a multi data source must be deployed on the same targets on which you intend to deploy the multi data source. You cannot include data sources in a multi data source that are deployed on different servers or clusters. See Target Data Sources.

1. In the **Edit Tree**, go to **Services**, then **Data Sources**.
2. Select the multi data source whose components you want to edit.
3. On the **Data Sources** tab, use the **Data Source List** field to modify which data sources are covered by this multi data source.

- To add a new data source, enter the exact name of the data source, separating each data source with a comma.
- To remove an existing data source, delete the name of the data source.

> **Note:**
>
> The order of data sources in the list determines the order that the multi data source uses to route connection requests. For multi data sources that use the Failover algorithm, the first data source in the list is considered the primary data source. Others are considered secondary, tertiary, and so forth.

4. Click **Save**.

Create an Active GridLink Data Source

An Active GridLink data source provides connectivity between WebLogic Server and an Oracle database.

For more information on GridLink data sources, see Using Active GridLink Data Sources in *Administering JDBC Data Sources for Oracle WebLogic Server*.

If you need a JDBC driver that is not installed with WebLogic Server, you must install it before you can set up a data source. See Adding Third-Party JDBC Drivers Not Installed with WebLogic Server in *Administering JDBC Data Sources for Oracle WebLogic Server*.

1. In the **Edit Tree**, go to **Services**, then **Data Sources**.
2. Click **New**.
3. Enter a name for the new data source.

   The name cannot contain the following characters: `@ # $`.
4. In the **JNDI Names** field, enter the JNDI path to the location where this JDBC data source will be bound. Applications look up the data source on the JNDI tree by this name when reserving a connection.
5. Choose the server instances or clusters where you want to deploy the data source.
6. Select **GridLink Data Source** from the **Data Source Type** drop-down list.
7. From the **Database Driver** drop-down list, select JDBC drivers.
   - Optional: If you selected a non-XA driver, then select a **Global Transactions Protocol**.

     See JDBC Data Source Transaction Options in *Administering JDBC Data Sources for Oracle WebLogic Server*.
8. Enter the connection details for the database that you want to connect to:
   - **Listeners**: Enter the DNS name or IP address and port number (separated by a colon) of the server that hosts the database. Enter each listener on a new line.
   - **Service Name**: Specify the service name of the database to which you want to connect.
   - **Database User Name**: Enter the database user account name that you want to use for each connection in the data source.
   - **Password**: Enter the password for the database user account.

- **Protocol**: If required, change the value from **TCP** to **SDP**. To use Socket Direct Protocol (SDP), your database network must be configured to use Infiniband.

9. Configure any additional connection details that are applicable to your environment.
10. Click **Create**.

Create a UCP Data Source

A Universal Connection Pool (UCP) data source provides an option for users who want to use Oracle Universal Connection Pooling to connect to Oracle databases. UCP provides an alternative connection pooling technology to Oracle WebLogic Server connection pooling.

For more information on UCP data sources, see Using Universal Connection Pool Data Sources in *Administering JDBC Data Sources for Oracle WebLogic Server*.

If you need a JDBC driver that is not installed with WebLogic Server, you must install it before you can set up a data source. See Adding Third-Party JDBC Drivers Not Installed with WebLogic Server in *Administering JDBC Data Sources for Oracle WebLogic Server*.

1. In the **Edit Tree**, go to **Services**, then **Data Sources**.
2. Click **New**.
3. Enter a name for the new data source.

   The name cannot contain the following characters: `@ # $`.
4. In the **JNDI Names** field, enter the JNDI path to the location where this JDBC data source will be bound. Applications look up the data source on the JNDI tree by this name when reserving a connection.
5. Choose the server instances or clusters where you want to deploy the data source.
6. Select **UCP Data Source** from the **Data Source Type** drop-down list.
7. From the **Database Driver** drop-down list, select JDBC drivers.
8. Enter the connection details for the database that you want to connect to:
   - **URL**: Enter the URL for the database.
   - **Database User Name**: Enter the database user account name that you want to use for each connection in the data source.
   - **Password**: Enter the password for the database user account.
9. Configure any additional connection details that are applicable to your environment.
10. Click **Create**.

Control JDBC Data Sources

After you create a JDBC data source, you can perform administrative tasks on instances of the data source in WebLogic Remote Console.

1. In the **Monitoring Tree**, go to **Services**, then **Data Sources**, then **JDBC Data Source Runtime MBeans**.
2. Select the data source that you want to manage and then choose the control operation that you want to perform on the data source.

Table 9-1 describes the control operations that you can perform on a JDBC data source.

Table 9-1 Control Operations for JDBC Data Sources

Operation	Description
Start	Use **Start** to start an individual instance of a data source. For more information, see Starting a Data Source in *Administering JDBC Data Sources for Oracle WebLogic Server*.
Resume	Use **Resume** to resume individual data sources that are in a `Suspended` state. For more information, see Resuming a Connection Pool in *Administering JDBC Data Sources for Oracle WebLogic Server*.
Suspend	Use **Suspend** to pause individual instances of a data source. When you suspend a data source, applications can no longer get a database connection from the data source. You can choose how to handle active connections: • Choose **Gracefully** to mark the data source as disabled and block any new connection requests. If there are any reserved connections, the operation will wait for the period as specified by `InactiveTimeout`, otherwise the operation waits 60 seconds before suspending all connections. If successful, the health state is set to `Suspended`. All connections are preserved exactly as they were before the data source was suspended. When you resume the data source, clients that had reserved a connection before the data source was suspended can continue exactly where they left off. • Choose **Force Suspend** to mark the data source as disabled. Any transaction on any currently reserved connection is rolled back, and all reserved connections are destroyed. Any subsequent attempts by applications to use their reserved connections will fail. If successful, the health state is set to `Suspended`. At this time, the data source attempts to replenish the connection pool by creating as many new connections as had been destroyed. When you **Resume**, clients must reserve new connections to proceed. For more information, see Suspending a Connection Pool in *Administering JDBC Data Sources for Oracle WebLogic Server*.
Shutdown	Use **Shutdown** to shut down individual instances of a data source. You can choose how to handle active connections: • Choose **Gracefully** to shut down a data source if there are no active connections. If any connections from the data source are currently in use, the **Shutdown** operation will fail and the health state remains `Running`. • Choose **Force Shutdown** to shut down a data source and force the disconnection of all current connection users. For more information, see Shutting Down a Data Source in *Administering JDBC Data Sources for Oracle WebLogic Server*.
Shrink	Use **Shrink** to shrink the pool of database connections in individual instances of a data source to the minimum capacity or the current number of connections in use, whichever is greater. For more information, see Shrinking a Connection Pool in *Administering JDBC Data Sources for Oracle WebLogic Server*.
Reset	Use **Reset** to reset the database connections in a JDBC data source by closing and then recreating all available database connections in the pool of connections in the data source. For more information, see Resetting a Connection Pool in *Administering JDBC Data Sources for Oracle WebLogic Server*.

Table 9-1 (Cont.) Control Operations for JDBC Data Sources

Operation	Description
Clear cache	Use **Clear cache** to clear the statement cache for all connections in the instance of the data source. Statement caching must be enabled for the data source for WebLogic Server to cache prepared and callable statements that are used in each connection in the data source. Each connection has its own cache, but the caches for each connection are configured and managed as a group. For more information, see Managing the Statement Cache for a Data Source and Increasing Performance with the Statement Cache in *Administering JDBC Data Sources for Oracle WebLogic Server*.

Monitor Statistics for JDBC Data Sources

You can monitor a variety of statistics for each data source instance in your domain, such as the current number of database connections in the connection pool, current number of connections in use, and the longest wait time for a database connection.

1. In the **Monitoring Tree**, go to **Services**, then **Data Sources**, then **JDBC Data Source Runtime MBeans**.

   Note:

   For multi data sources, you can also go to **Services**, then **Data Sources**, then **JDBC Multi Data Source Runtime MBeans** to review a multi data source and its associated sub data sources.

2. Select the data source for which you want to see statistics.

By default, this data source page will display all of the statistics that are available for the selected data source. To refine the information, customize the table to show only relevant statistics. See Customize a Table.

Target Data Sources

When you target a JDBC data source, a new instance of the data source is created on the target. When you select a *server* as a target, an instance of the data source is created on the server. When you select a *cluster* as a target, an instance of the data source is created on *all* servers in the cluster.

Make sure that the JDBC drivers that you want to use to create database connections are installed on all servers on which you want to deploy the data source. See Using JDBC Drivers with WebLogic Server in *Administering JDBC Data Sources for Oracle WebLogic Server*.

1. In the **Edit Tree**, go to **Services**, then **Data Sources**.
2. Select the data source whose targets you want to edit.
3. On the **Targets** tab, select each server or cluster where you want to deploy the data source and move them under **Chosen**. Move unwanted servers or clusters under **Available**.
4. Click **Save**.

Test JDBC Data Sources

You can manually test individual instances of a data source. When you test a data source, WebLogic Server reserves a connection from the data source, tests it using the standard testing query or the query specified in Test Table Name, and then returns the database connection to the pool of connections.

It is important to regularly check that the database connections in a data source remain healthy, which helps keep your applications running properly. See Testing Data Sources and Database Connections in *Administering JDBC Data Sources for Oracle WebLogic Server*.

1. Optional: Configure testing options for the JDBC data source.

   You may want to modify the default database connection testing options to better match the needs of your environment.

   a. In the **Edit Tree**, go to **Services**, then **Data Sources**.

   b. Select the data source whose connection testing options you want to edit.

   c. Select the **Connection Pool** tab, then the **Advanced** subtab.

   d. Modify the following options as necessary for your environment.

   e. In the **Test Table Name** field, enter the name of a small table to use in a query to test database connections. The standard query is `select 1 from table_name`. If you prefer to use a different query as a connection test, enter `SQL` followed by a space and the SQL code you want to use to test database connections.

   f. Enable the **Test Connections on Reserve** option to test the database connection before giving it to your application when your application requests a connection form the data source.

      The test adds a small delay in serving the client's request for a connection from the pool, but ensures that the client receives a viable connection.

   g. In the **Test Frequency** field, specify how frequently (in seconds) WebLogic Server should perform background connection tests.

   h. In the **Seconds to Trust an Idle Pool Connection** field, enter the interval (in seconds) during which, if the database connection has been used or tested, WebLogic Server will skip the connection test. This option can help reduce the overhead of connection testing and improve application performance.

   i. Click **Save** and commit your changes.

2. In the **Monitoring Tree**, go to **Environment**, then **Servers**, then choose the server where the data source you want to test is deployed.
3. Go to **Services**, then **Data Sources**, then **JDBC Data Source Runtime MBeans**.
4. Select the data source that you want to test.
5. Click the **Test** tab.

The test will occur immediately. Test results are shown on the **Test** tab page.

Specify RMI JDBC Security

You can secure RMI JDBC communication with a data source using a check for administrator authentication.

For more information about using JDBC over RMI, see Using the WebLogic RMI Driver (Deprecated) in *Developing JDBC Applications for Oracle WebLogic Server*.

1. Optional: If you plan to choose the `Secure` option, you must first configure an SSL/TLS listen port channel. See Specify Listen Ports.
2. In the **Edit Tree**, go to **Environment**, then **Servers**. Click **Show Advanced Fields**.
3. Select an option from the **RMI JDBC Security** drop-down list:
   - `Secure`: Rejects all incoming application JDBC calls over RMI by remote clients and servers. Internal interserver JDBC calls over RMI operations are allowed for the Logging Last Resource, Emulate Two-Phase Commit and One-Phase Commit Global Transactions Protocol options. The `Secure` option requires that all the servers are configured with an SSL listen port. If not, all operations fail with an exception.
   - `Compatibility`: Allows uncontrolled access to DataSource objects for all incoming JDBC application calls over RMI. This setting should only be used when strong network security is in place.
   - `Disabled`: Disables all JDBC calls over RMI, including the internal RMI operations for Logging Last Resource, Emulate Two-PhaseCommit and One-Phase Commit Global Transactions Protocol options.

   For more information, see Security Considerations for WebLogic RMI Drivers in *Developing JDBC Applications for Oracle WebLogic Server*.

   As of WebLogic Server 14.1.2.0.0, the default value for **RMI JDBC Security** is `Secure`.
4. Click **Save**.

Configure Global Transaction Options for a JDBC Data Source

The transaction protocol for a JDBC data source determines how connections from the data source are handled during transaction processing.

1. In the **Edit Tree**, go to **Services**, then **Data Sources**.
2. Select the data source whose targets you want to edit.
3. On the **Transactions** tab, select an option for transaction processing from the **Global Transactions Protocol** drop-down list.

   To understand how transaction options differ, see JDBC Data Source Transaction Options in *Administering JDBC Data Sources for Oracle WebLogic Server*.

   > **Note:**
   >
   > If the data source uses an XA JDBC driver to create database connections, connections from the data source will support the two-phase commit transaction protocol only.
4. Click **Save**.

Messaging

WebLogic Server provides an enterprise-class messaging system that fully supports the Java Messaging Service (JMS) specification and which also provides numerous extensions that go beyond the standard JMS APIs.

It is tightly integrated into the WebLogic Server platform, allowing you to build highly secure Java EE applications that can be easily monitored and administered through WebLogic Remote Console. In addition to fully supporting XA transactions, WebLogic Server messaging also features high availability through its clustering and service migration features while also providing seamless interoperability with other versions of WebLogic Server and third-party messaging vendors.

WebLogic Server messaging is comprised of these areas:

- **JMS Servers**: act as management containers for JMS queue and topic destinations in a JMS module that are targeted to them. A JMS server's primary responsibility for its destinations is to maintain information on which persistent store is used for any persistent messages that arrive on the destinations, and to maintain the states of durable subscribers created on the destinations.
- **Store-and-Forward Agents**: provide a mechanism for reliably delivering messages between applications that are distributed across WebLogic Server subsystems, in particular the WebLogic JMS and Web Services subsystems. Using the highly available SAF service, an application can send messages to a remote endpoint that is not available at the moment when the messages are sent, either because of network problems or system failures.
- **JMS System Modules**: contain global configuration JMS resources, such as queues, topics, templates, connections factories, and JMS store-and-forward (SAF) destinations, and are defined by XML documents that conform to the `weblogic-jmsmd.xsd` schema. JMS system modules are stored in *`DOMAIN_HOME`*`/config/jms` and a reference to the module is added in the domain's configuration file as a `JMSSystemResource` element. System modules are globally available for targeting to servers and clusters configured in the domain, and therefore are available to all applications deployed on the same targets and to client applications.
- **Messaging Bridges**: provide a forwarding mechanism between any two messaging products that support the JMS API. Use a messaging bridge to provide interoperability between separate implementations of WebLogic JMS, or between WebLogic JMS and another messaging product.

Related Content

For more information, see:

- Understanding JMS Resource Configuration in *Administering JMS Resources for Oracle WebLogic Server*
- Understanding the Store-and-Forward Service in *Administering the Store-and-Forward Service for Oracle WebLogic Server*
- Understanding the Messaging Bridge in *Administering the WebLogic Messaging Bridge for Oracle WebLogic Server*
- The WebLogic Persistent Store in *Administering the WebLogic Persistent Store*

Create a JMS Server

JMS servers are environment-related configuration entities that act as management containers for the queues and topics in JMS modules that are targeted to them.

The primary responsibility of a JMS server for its destinations is to maintain information on which persistent store is used for any persistent messages that arrive on the destinations, and to maintain the states of durable subscribers created on the destinations.

JMS servers also manage message paging on destinations, and optionally, can manage message and byte thresholds, as well as server-level quota for its targeted destinations. As a container for targeted destinations, any configuration or run-time changes to a JMS server can affect all the destinations that it hosts.

For more information, see Overview of JMS Servers and Configure JMS Servers and Persistent Stores in *Administering JMS Resources for Oracle WebLogic Server*.

1. In the **Edit Tree**, go to **Services**, then **JMS Servers**.
2. Click **New**.
3. Enter a name for the new JMS Server and click **Create**.
4. From the **Persistent Store** drop-down list, choose the store where this JMS server should store its persistent messages.

   If you want to use the default persistent store provided by WebLogic Server, leave **Persistent Store** set to None.

   If you don't have a custom persistent store configured yet, click the **More** ⋮ button beside **Persistent Store** and then select either **New File Store** or **New JDBC Stores**. Then, follow the instructions at Create a File Store or Create a JDBC Store. Select the newly created store from the **Persistent Store** drop-down list.

   Note:

   When a JMS server is targeted to a:

   - Migratable target, it cannot use the default store. A custom store must be configured and targeted to the same migratable target.
   - Dynamic cluster, it requires a custom persistent store that must target the same dynamic cluster or use the default store available on each dynamic cluster member.

     Use a custom store with **Distribution Policy** set to `Singleton` to host stand-alone (non-distributed) destinations.

     Use a custom store with **Distribution Policy** set to `Distributed` to host distributed destinations.

5. Configure any additional JMS server attributes. Remember to click **Show Advanced Fields** to see all of the options.
6. Click **Save**.
7. On the **Target** tab, from the **Target** drop-down list, select the server instance, cluster, or migratable target where you want to deploy the JMS server.

You can target a JMS server to a standalone WebLogic Server instance, a cluster, or a migratable target server. Migratable targets define a set of WebLogic Server instances in a cluster that can potentially host a pinned service, such as a JMS server.

If you are using a JMS server in a clustered server environment, review the guidance provided at Configure JMS Servers and Persistent Stores in *Administering JMS Resources for Oracle WebLogic Server*.

8. Click **Save**.

Monitor a JMS Server

You can monitor runtime statistics for an active JMS server. You can also access runtime information for a JMS server's destinations, transactions, connections, and server session pools.

For more information, see Monitoring JMS Statistics and Managing Messages in *Administering JMS Resources for Oracle WebLogic Server*.

1. In the **Monitoring Tree**, go to **Environment**, then **Servers** and choose the server where the JMS server is deployed. Then proceed to **Services**, then **Messaging**, then **JMS Runtime**, then **JMS Servers**.

   If you want to compare all of the JMS servers in the domain, instead go to **Services**, then **Messaging**, then **JMS Runtime**, then **JMS Servers**.
2. Select the JMS server whose runtime information you want to view.

Create a JMS System Module

JMS resources are configured and stored as modules similar to standard Java EE modules. Such resources include queues, topics, connection factories, templates, destination keys, quota, distributed queues, distributed topics, foreign servers, and JMS store-and-forward (SAF) parameters.

System modules are globally available for targeting to servers and clusters configured in the domain, and therefore are available to all applications deployed on the same targets and to client applications.

JMS configuration resources can also be managed as deployable application modules, either with a Java EE application as a packaged module, which is available only to the enclosing application, or as a stand-alone module that provides global access to the resources defined in that module.

For more information, see Overview of JMS Modules in *Administering JMS Resources for Oracle WebLogic Server*.

1. In the **Edit Tree**, go to **Services**, then **JMS Modules**.
2. Click **New**.
3. Enter a name for the new JMS system module and click **Create**.
4. On the **Target** tab, from the **Target** drop-down list, select a server instance or cluster on which to deploy the JMS system module.
5. Click **Save**.

   A new node will appear under **JMS Modules** in the Navigation Tree for your JMS system module.

6. Expand the new node for your JMS system module to see its children which are the JMS system resources that you can configure. See Configure Resources for JMS System Modules.

Configure Resources for JMS System Modules

After creating a JMS system module, you can configure resources for the module, including stand-alone queues and topics, distributed queues and topics, connection factories, JMS templates, destination sort keys, destination quota, foreign servers, and JMS SAF (store-and-forward) parameters.

For more information, see Configurable JMS Resources in Modules in *Administering JMS Resources for Oracle WebLogic Server*.

1. In the **Edit Tree**, go to **Services**, then **JMS Modules**.
2. Select the JMS system module that you want to configure resources for.
3. In the Navigation Tree, as child nodes of JMS system module that you selected, click the resource that you want to configure.

   The following JMS system resources are available:

   - **Quota** - controls the allotment of system resources available to destinations.
   - **Template** - provides an efficient means of defining multiple queues and topics with similar configuration settings.
   - **Destination Key** - defines a sort order for the messages as they arrive on destinations.
   - **Topic** - defines a publish/subscribe (pub/sub) destination, which enables an application to send a message to multiple applications.
   - **Queue** - defines a point-to-point (PTP) destination, which enables one application to send a message to another.
   - **Connection Factory** - defines a set of connection configuration parameters that enable JMS clients to create JMS connections.
   - **Distributed Topic** - a single unit of JMS topics that are accessible as a single, logical topic to a client. The members of the distributed topic are usually distributed across multiple servers within a cluster, with each topic member belonging to a separate JMS server.
   - **Distributed Queue** - a single unit of JMS queues that are accessible as a single, logical topic to a client. The members of the distributed queue are usually distributed across multiple servers within a cluster, with each topic member belonging to a separate JMS server.
   - **Foreign Server** - represents a third-party JMS provider that is outside WebLogic Server. It contains information that allows a local server instance to reach a remote JNDI provider, thereby allowing for a number of foreign connection factory and destination objects to be defined on one JNDI directory.
   - **SAF Imported Destination** - defines a collection of imported SAF (store-and-forward) queues or topics that represent JMS destinations in a remote server instance or cluster.
   - **Remote SAF Context** - specifies SAF login context that a SAF imported queue or topic uses to connect to a remote destination.
   - **SAF Error Handling** - specifies the action to be taken when the SAF service fails to forward messages to a remote destination.

For guidance on how to configure JMS modules, see Configuring Basic JMS System Resources in *Administering JMS Resources for Oracle WebLogic Server*.

4. Enter any required information for the selected resource.

   Certain resources may encourage you to configure an appropriate subdeployment. A subdeployment is the mechanism by which targetable JMS module resources (such as queues, topics, and connection factories) are grouped and targeted to a server resource (such as JMS servers, server instances, or cluster).

   Most JMS resources have additional parameters that can be modified after they are created. For example, you can modify the default message threshold values or enable message logging for queues, topics, and templates.

5. Click **Create**.

Create a Store-and-Forward Agent

The Store-and-Forward (SAF) service enables WebLogic Server to deliver messages reliably between applications that are distributed across WebLogic Server instances.

If the destination is not available at the moment the messages are sent, either because of network problems or system failures, then the messages are saved on a local server instance, and are forwarded to the remote destination once it becomes available.

SAF agents are responsible for store-and-forwarding messages between local sending and remote receiving endpoints. A SAF agent can be configured to have only sending capabilities, receiving capabilities, or both. JMS SAF only requires a sending agent on the sending side for JMS messages. Web Services Reliable Messaging (WSRM) SAF requires both a sending agent and a receiving agent.

For more information, see SAF Service Agents in *Administering the Store-and-Forward Service for Oracle WebLogic Server*.

1. In the **Edit Tree**, go to **Services**, then **SAF Agents**.
2. Click **New**.
3. Enter a name for the new SAF agent and click **Create**.
4. From the **Persistent Store** drop-down list, choose the store where this SAF agent will store its persistent messages.

   If you want to use the default persistent store provided by WebLogic Server, leave **Persistent Store** set to `None`.

   If you don't have a custom persistent store configured yet, click the **More** ⋮ button beside **Persistent Store** and then either **New File Store** or **New JDBC Stores**. Then, follow the instructions at Create a File Store or Create a JDBC Store.

   - When a SAF agent is targeted to a cluster, a SAF Agent must use a custom store with **Distribution Policy** set to `Distributed` and is targeted to the same cluster.
   - A SAF Agent can use a default store only when targeting a configured (non-dynamic) cluster.
   - When a SAF agent is targeted to a migratable target, a custom store must be configured and targeted to the same migratable target.

5. From the **Agent Type** drop-down list, choose one of the following options:

   - **Both**: Configures an agent that has sending and receiving agent functionality.

- **Sending-only**: Configures an agent that stores messages in persistent storage, forwards messages to the receiving side, and re-transmits messages when acknowledgments do not come back in time.
- **Receiving-only**: Configures an agent that detects and eliminates duplicate messages sent by a receiving agent, and delivers messages to the final destination.

JMS SAF users should select **Sending-only** as JMS SAF doesn't require a configured receiving agent.

6. Configure any additional SAF agent attributes. Remember to click **Show Advanced Fields** to see all of the options.
7. Click **Save**.
8. On the **Target** tab, from the **Target** drop-down list, select the server instance, cluster, or migratable target where you want to deploy the SAF agent.

   When targeting a cluster, a SAF Agent must use a custom store with **Distribution Policy** set to `Distributed` and is targeted to the same cluster.

   If a SAF agent is targeted to a migratable target, it cannot be targeted to any other server targets, including an entire cluster.
9. Click **Save**.

Monitor an SAF Agent

You can view runtime information for an active SAF agent .

For more information, see Monitoring SAF Agents in *Administering the Store-and-Forward Service for Oracle WebLogic Server*.

1. In the **Monitoring Tree**, go to **Environment**, then **Servers** and choose the server where the SAF agent is deployed. Then proceed to **Services**, then **Messaging**, then **SAF Runtime**, then **Agents**.

   If you want to see all of the SAF agents in the domain, instead go to **Services**, then **Messaging**, then **SAF Runtime**, then **Agents**.
2. Select the SAF agent whose runtime information you want to view.

Create a JMS Bridge Destination

A JMS bridge destination instance defines the actual source and target JMS bridge destinations for a bridge instance.

You need to configure a JMS bridge destination instance for each source *and* each target destination to be mapped to a messaging bridge instance. Therefore, when you finish defining attributes for a source JMS bridge destination, repeat these steps to configure a target JMS bridge destination.

1. In the **Edit Tree**, go to **Services**, then **JMS Bridge Destinations**.
2. Click **New**.
3. Enter a name for the new JMS bridge destination and click **Create**.
4. In the **Adapter JNDI Name** field, specify the JNDI name of the adapter used to communicate with the bridge destinations.

   For more information on which adapter name to enter, see Resource Adapters in *Administering the WebLogic Messaging Bridge for Oracle WebLogic Server*.

5. In the **Connection URL** field, specify the connection URL for this JMS bridge destination.
6. In the **Connection Factory JNDI Name** field, specify the connection factory's JNDI name for this JMS bridge destination.
7. In the **Destination JNDI Name** field, specify the destination JNDI name for this JMS bridge destination.
8. Configure any additional attributes that are applicable to your environment.
9. Click **Save**.
10. Repeat these steps to create a matching JMS bridge destination.

    If you first created a *source* JMS bridge destination, then now you should create a *target* destination.

    If you first created a *target* JMS bridge destination, then now you should create a *source* destination.

Create a Messaging Bridge Instance

The WebLogic messaging bridge allows you to configure a forwarding mechanism between any two messaging products. You can use the messaging bridge to integrate your messaging applications. A messaging bridge instance communicates with the configured source and target bridge destinations.

For each mapping of a source destination to a target destination, whether it is another WebLogic JMS implementation, or a third-party JMS provider, you must configure a messaging bridge instance.

Each instance defines the source and target destination for the mapping, a message filtering selector, a quality of service (QOS), transaction semantics, and reconnection parameters.

For more information, see Understanding the Messaging Bridge in *Administering the WebLogic Messaging Bridge for Oracle WebLogic Server*.

1. If you haven't already, create and configure source and target JMS bridge destinations as described in Create a JMS Bridge Destination.
2. In the **Edit Tree**, go to **Services**, then **Messaging Bridge**.
3. Click **New**.
4. Enter a name for the new messaging bridge instance and click **Create**.
5. From the **Source Bridge Destination** drop-down list, select a source destination.
6. From the **Target Bridge Destination** drop-down list, select a target destination.
7. Configure any additional attributes that are applicable to your environment.
8. Click **Save**.

If you plan to use the messaging bridge to access destinations on different releases of WebLogic Server, or in remote WebLogic domains, then you may need to manually implement some of the interoperability guidelines described at Interoperating with Different WebLogic Server Releases or Foreign Providers in *Administering the WebLogic Messaging Bridge for Oracle WebLogic Server*.

Create a File Store

A file store is a file-based repository for storing subsystem data, such as persistent JMS messages or durable subscriber information.

A persistent store provides a built-in, high-performance storage solution for WebLogic Server subsystems and services that require persistence. For example, it can store persistent JMS messages or durable subscriber information, as well as temporarily store messages sent to an unavailable destination using the store-and-forward feature.

The persistent store also supports persistence to a JDBC-enabled database (JDBC store). See Create a JDBC Store instead.

1. Create a directory on your file system for the file store.

   The directory must be accessible from all candidate server members. For highest reliability, use a shared storage solution that is itself highly available. For example, a storage area network (SAN) or a dual-ported SCSI disk.
2. In the **Edit Tree**, go to **Services**, then **File Stores**.
3. Click **New**.
4. Enter a name for the new File Store and click **Create**.
5. In the **Directory** field, enter the path to the directory on the file system where the file store is kept.
6. Modify any additional settings that are applicable to your environment.
7. Click **Save**.
8. On the **Target** tab, from the **Target** drop-down list, select a server instance, dynamic cluster, or migratable target on which to deploy the file store.

   When selecting a dynamic cluster, the store must be targeted to the same dynamic cluster as the JMS server.

   When selecting a migratable target, the store must share the same migratable target as the migratable JMS server or SAF agent.
9. Click **Save**.

Create a JDBC Store

A JDBC store is a JDBC-accessible database for storing subsystem data, such as persistent JMS messages and durable subscriber information.

A persistent store provides a built-in, high-performance storage solution for WebLogic Server subsystems and services that require persistence. For example, it can store persistent JMS messages or durable subscriber information, as well as temporarily store messages sent to an unavailable destination using the store-and-forward feature.

The persistent store also supports persistence to a file-based store (file store). See Create a File Store instead.

1. If you haven't already, create a JDBC data source. See Data Sources.

   A JDBC store must use a JDBC data source that uses a non-XA JDBC driver and has **Supports Global Transactions** disabled. This limitation does not remove the XA capabilities of layered subsystems that use JDBC stores. For example, WebLogic JMS is fully XA-capable regardless of whether it uses a file store or any JDBC store.

Make sure the data source is accessible to all candidate servers.

2. In the **Edit Tree**, go to **Services**, then **JDBC Stores**.
3. Click **New**.
4. Enter a name for the new JDBC store.
5. In the **Prefix Name** field, specify a prefix name to add to the start of the table name in this JDBC store for use with multiple instances.
6. From the **Data Source** drop-down list, select a data source.
7. From the **Target** drop-down list, select a server instance, dynamic cluster, or migratable target on which to deploy the file store.

   When selecting a dynamic cluster, the store must be targeted to the same dynamic cluster as the JMS server. When selecting a migratable target, the store must share the same migratable target as the migratable JMS server or SAF agent.
8. Modify the rest of the settings as needed.
9. Click **Save**.

Create a Path Service

A path service is persistent map that can be used to store the mapping of a group of messages to a messaging resource, such as a member of a distributed destination or a store-and-forward agent.

Path services provide a way to enforce message ordering by pinning messages to a member of a cluster hosting servlets, distributed queue members, or store-and-forward agents.

For more information, see Using the WebLogic Path Service in *Administering JMS Resources for Oracle WebLogic Server*.

1. If you haven't already, create and configure at least one for each of the following: a cluster, a custom persistent store, and a store-and-forward (SAF) agent. You may also want to configure JMS system modules.

   See:
   - Configure Clusters
   - Create a File Store or Create a JDBC Store
   - Create a Store-and-Forward Agent
   - Create a JMS System Module
2. In the **Edit Tree**, go to **Services**, then **Path Services**.
3. Click **New**.
4. Enter a name for the new Path Service and click **Create**.
5. From the **Persistent Store** drop-down list, choose the store where this path service will store its persistent messages.

> **Note:**
>
> If you plan to target the path service to a migratable target, then it must use a custom store. If you plan to target the path service to a cluster, then you must specify a custom store with the same target, a **Migration Policy** set to `Always`, and a **Distribution Policy** set to `Singleton`.

If you want to use the default persistent store provided by WebLogic Server, leave **Persistent Store** set to `None`. However, it is recommended that you use a custom store instead of the default store.

6. Click **Save**.
7. On the **Target** tab, from the **Target** drop-down list, select the cluster, cluster member or migratable target where you want to deploy the path service.
8. Click **Save**.

Manage JMS Messages

You can manage JMS messages that are available on the standalone queue, distributed queue, or durable topic subscriber that you are monitoring.

> **Note:**
>
> This functionality is only available on domains running WebLogic Server 14.1.2.0.0 or later. If you want to manage JMS messages for an older release, you must use an alternative administration tool.

1. In the **Monitoring Tree**, go to **Environment**, then **Servers** and select the server hosting the messages that you want to move.
2. Under the selected server, go to **Services**, then **Messaging**, then **JMS Runtime**, then **JMS Servers**, then *myJMSServer*, then **Destinations** and select the JMS destination that currently hosts the messages that you want to move.
3. On the **Messages** tab, select all of the messages on which you want to perform the action and then click the action's button. See Table 9-2.

Table 9-2 JMS Message Actions

Action	Description
Delete	Deletes the selected JMS messages from the current queue. See Deleting Messages in *Administering JMS Resources for Oracle WebLogic Server*.
Export	Exports the selected messages from the current queue, which results in a JMS message that is converted to either XML or serialized format. See Exporting Messages in *Administering JMS Resources for Oracle WebLogic Server*.

Table 9-2 (Cont.) JMS Message Actions

Action	Description
Import	Imports the selected messages in XML format, which results in the creation or replacement of a message on the current queue. See Importing Messages in *Administering JMS Resources for Oracle WebLogic Server*.
Move	Transfers selected JMS messages from the current queue to another destination, including a destination on a different JMS server. The message identifier does not change when you move a message. If the message being moved already exists on the target destination, a duplicate message with the same identifier is added to the destination. See Moving Messages in *Administering JMS Resources for Oracle WebLogic Server*.

View Objects in the JNDI Table

You can use WebLogic Remote Console to view objects in the Java Naming and Directory Interface (JNDI) table.

WebLogic Server implements the JNDI of the Java EE platform as a means to provide a standard, unified interface to multiple naming and directory services in an enterprise. See Understanding WebLogic JNDI in *Developing JNDI Applications for Oracle WebLogic Server*.

You can load WebLogic Server Java EE services and components, such as RMI, JMS, EJBs, and JDBC Data Sources, in the JNDI table. Typically, these objects are bound in the JNDI table when you configure their **JNDI Name** attribute and deploy them to the server.

1. In the **Monitoring Tree**, go to **Environment**, then **Servers**, then *myServer*.
2. Click the **JNDI** tab to see the JNDI objects.

> **Note:**
>
> The JNDI tab is only visible on servers that are running and reachable by the Administration Server.

Create a Foreign JNDI Provider

A foreign JNDI provider represents a JNDI tree that resides outside of a WebLogic Server environment. This could be a JNDI tree in a different server environment or within an external Java program.

By setting up a foreign JNDI provider, you can look up and use a remote object with the same ease as using an object bound in your WebLogic Server instance. In other words, you can access local and remote objects using a single WebLogic Server connection.

1. In the **Edit Tree**, go to **Services**, then **Foreign JNDI Providers**.
2. Click **New**.
3. Enter a name for the new Foreign JNDI provider and click **Create**.

4. In the **Initial Context Factory** field, enter the name of the class that must be instantiated to access the JNDI provider. This class name depends on the JNDI provider and the vendor that are being used. The value corresponds to the standard JNDI property, `java.naming.factory.initial`.
5. In the **Provider URL** field, enter the URL that WebLogic Server will use to contact the JNDI provider. This value corresponds to the standard JNDI property, `java.naming.provider.url`.
6. In the **User** field, enter the name of a user authorized to access the foreign JNDI, then in **Password**, enter the user account's password.
7. Click the **Targets** tab and select the servers or clusters where you want to deploy this foreign JNDI provider.
8. Optional: If you want to specify additional properties for the JNDI provider, click the **Properties** tab.

   These properties will be passed directly to the constructor for the JNDI provider's `InitialContext` class.

   a. In the **Properties** table, click **+** to add a new row.
   b. Double-click the cell under **Properties Name** and enter a name for the property.
   c. Double-click the cell under **Properties Value** and enter a value for the property.
   d. Click **Save**.

Next, you should create foreign JNDI object links which set up a relationship between a name in your local JNDI table and the object in the foreign (remote) table.

9. Expand the node in the Navigation Tree for the Foreign JNDI Provider you created and open the **Foreign JNDI Links** child node.
10. Click **New**.
11. Enter a name for the foreign JNDI link.
12. In the **Local JNDI Name** field, specify the name that the remote JNDI object will be bound to in the local server's JNDI tree and used to look up the object on the local server.
13. In the **Remote JNDI Name** field, specify the name of the remote object that will be looked up in the foreign JNDI directory.
14. Click **Create**.

XML Resources

You can configure two different types of XML resources for WebLogic Server.

- XML registries, which you can use to specify alternative server-wide XML parsers and transformers for WebLogic Server to use when it parses and transforms XML documents. You can also use the XML registry to specify local copies of external entities and caching instructions for these entities. See Create an XML registry.
- XML entity caches, which you can use to configure the cache that WebLogic Server uses to cache external entities. See Create an XML Entity Cache.

You can create as many XML registries and entity caches as you like. However, you can only associate one of each type with a particular server instance of WebLogic Server.

For more information on how XML resources are used in WebLogic Server, see *Developing XML Applications for Oracle WebLogic Server*.

Create an XML registry

An XML Registry is a facility for configuring and administering the XML resources of WebLogic Server. XML resources include the default parser, transformer factories, and external entity resolution.

In particular, use an XML registry to specify:

- An alternative, server-wide XML parser, used by default when parsing XML documents, instead of the parser that is installed by default. You do this by specifying the names of the classes that implement the `javax.xml.parsers.DocumentBuilderFactory` and `javax.xml.parsers.SaxParserFactory` interfaces; these implementing classes are used to parse XML in DOM and SAX mode, respectively.
- A specific XML parser that should be used to parse a particular document type.
- An alternative server-wide transformer instead of the default transformer. You do this by specifying the name of the class that implements the `javax.xml.transform.TransformerFactory` interface, used to transform XML documents.
- External entities that are to be resolved by using local copies of the entities. After you specify these entities, the Administration Server stores local copies of them in the file system and automatically distributes them to the server's parser at parse time. This feature eliminates the need to construct and set SAX EntityResolvers.
- External entities to be cached by WebLogic Server after retrieval from the Web. Specify how long these external entities should be cached before WebLogic Server re-retrieves them and when WebLogic should first retrieve the entities, either at application run time or when WebLogic Server starts.

You can create as many XML Registries as you like; however, you can associate only *one* XML registry with a particular instance of WebLogic Server.

If an instance of WebLogic Server does not have an XML registry associated with it, then the default parser and transformer are used when parsing or transforming documents. The default parser and transformer are those included in the JDK.

Once you associate an XML registry with an instance of WebLogic Server, all its configuration options are available for XML applications that use that server.

1. In the **Edit Tree**, go to **Services**, then **XML Registries**.
2. Click **New**.
3. Enter a name for the XML registry and click **Create**.
4. Optional: If you don't plan to use the default parser and transformer, you must specify your alternative settings.
   a. In the **Document Builder Factory** field, enter the fully qualified name of the class that implements the `javax.xml.parsers.DocumentBuilderFactory` interface.
   b. In the **SAX Parser Factory** field, enter the fully qualified name of the class that implements the `javax.xml.parsers.SaxParserFactory` interface.
   c. In the **Transformer Factory** field, enter the fully qualified name of the class that implements the `javax.xml.transform.TransformerFactory` interface.
   d. Click **Save**.

Next, you must associate the XML registry with a WebLogic Server instance. See Target an XML Registry to a Server.

Target an XML Registry to a Server

A WebLogic Server can only have one XML registry associated with it. However, you can target the same XML registry to multiple WebLogic Server instances.

1. If you haven't done so already, create an XML registry. See Create an XML registry.
2. In the **Edit Tree**, go to **Environment**, then **Servers**.
3. Choose the server to which you want the XML registry.
4. Enable **Show Advanced Fields** and then from the **XML Registry** drop-down list, select the XML registry that you want to target to this server.
5. Click **Save**.

Create an XML Entity Cache

You can specify that WebLogic Server should cache external entities that are referenced with a URL or a pathname relative to the main directory of the EAR archive, either at server startup or when the entity is first referenced. You specify this by first creating an XML entity cache and then specifying when it should be cached for the particular entity.

Caching the external entity saves the remote access time and provides a local backup in the event that the Administration Server cannot be accessed while an XML document is being parsed, due to the network or the Administration Server being down.

1. In the **Edit Tree**, go to **Services**, then **XML Entity Caches**.
2. Click **New**.
3. Enter a name for the XML entity cache and click **Create**.
4. Update the configuration options for the new XML entity cache as needed.
5. Click **Save**.

Next, you must associate the XML entity cache with a WebLogic Server instance. See Target an XML Entity Cache to a Server.

Target an XML Entity Cache to a Server

A WebLogic Server instance can only have one XML entity cache associated with it.

1. If you haven't already done so, create an XML entity cache. See Create an XML Entity Cache.
2. In the **Edit Tree**, go to **Environment**, then **Servers**.
3. Choose the server to which you want the XML entity cache.
4. Enable **Show Advanced Fields** and then from the **XML Entity Cache** drop-down list, select the XML entity cache that you want to target to this server.
5. Click **Save**.

Configure Access to JavaMail

You can configure a mail server or establish user credentials for an existing mail server. Mail sessions and the JavaMail API do not provide mail server functions. They merely enable applications to send and receive data from an existing mail server.

JavaMail APIs provide applications and other Java EE modules with access to Internet Message Access Protocol (IMAP) and Simple Mail Transfer Protocol (SMTP) capable mail servers on your network or the internet.

In the reference implementation of JavaMail, applications must instantiate `javax.mail.Session` objects, which designate mail hosts, transport and store protocols, and a default mail user for connecting to a mail server. You can use WebLogic Remote Console to create a mail session, which configures a `javax.mail.Session` object and registers it in the WebLogic Server JNDI table. Applications access the mail session through JNDI instead of instantiating their own `javax.mail.Session` object.

1. In the **Edit Tree**, go to **Services**, then **Mail Sessions**.
2. Click **New**.
3. Enter a **Name** and a **JNDI Name** for the mail session.

   Applications use the JNDI Name to look up the mail session. For example, if you enter `myMailSession` as the JNDI name, applications perform the following look up:

   ```
   InitialContext ic = new InitialContext();
   Session session = (Session) ic.lookup("myMailSession");
   ```

4. Click **Create**.
5. In the **Session Username** field, specify the user account to use to create an authenticated JavaMail session. Then, in the **Session Password** field, enter the password for the user account.

   If you do not specify a user account, it is assumed the session is not to be authenticated.
6. On the **Targets** tab, move the servers or clusters that you want this mail session to target over to **Chosen**.

   The mail session is only registered in the JNDI table for the targeted WebLogic Server instances.

   When you target all or part of a cluster, WebLogic Remote Console initiates a two-phase deployment. In general, such a deployment ensures that if the deployment fails for one active server, it fails for all active servers.
7. Click **Save**.
8. Optional: You can specify additional properties for connecting to an existing mail server.

   > **Note:**
   >
   > Only specify a property if you want to override the default value. If you do not specify any properties, the mail session will use the JavaMail default property values.

a. On the **Java Mail Properties** tab, in the **Java Mail Properties** table, click **+** to add a new row.
b. Double-click the cell under **Properties Name** and enter a name for the property.
c. Double-click the cell under **Properties Value** and enter a value for the property.
d. Click **Save**.

Table 9-3 describes the valid properties and default values, derived from the JavaMail API Design Specification.

Table 9-3 Java Mail Properties

Property	Description	Default
`mail.store.protocol`	Protocol for retrieving email. For example, `mail.store.protocol=imap`.	`imap`
`mail.transport.protocol`	Protocol for sending email. For example, `mail.transport.protocol=smtp`	`smtp`
`mail.host`	The name of the mail host machine. For example, `mail.host=mailserver`.	Local machine
`mail.user`	Name of the default user for retrieving email. For example, `mail.user=postmaster`.	Value of the `user.name` Java system property.

`mail.protocol.host`	The mail host for a specific protocol. You can set `mail.SMTP.host` and `mail.IMAP.host` to different machine names.
	For example,
	`mail.smtp.host=mail.mydom.com`
	`mail.imap.host=localhost`
		Value of the `mail.host` property.

`mail.protocol.user`	Protocol-specific default user name for logging into a mailer server. For example, `mail.smtp.user=weblogic` `mail.imap.user=appuser`.	Value of the `mail.user` property.
`mail.protocol.user`	The default return address. For example, `mail.from=master@mydom.com`	`username@host`
`mail.debug`	Set to `True` to enable JavaMail debug output.	`False`

Note:

Applications can override any properties set in the mail session by creating a `Properties` object containing the properties you want to override. See Programming JavaMail with WebLogic Server in *Developing Applications for Oracle WebLogic Server*.

Configure Domain JTA Options

In WebLogic Server, you can set many options that affect transaction processing at the domain level so that they apply to all servers in the domain.

1. In the **Edit Tree**, go to **Services**, then **JTA**.
2. Specify new values for any or all of the available options. Click **Show Advanced Fields** to show all of the options.
3. Click **Save**.

View Transaction Statistics

You can monitor transactions and transaction statistics.

For more information, see Monitoring Transactions in *Developing JTA Applications for Oracle WebLogic Server*.

1. In the **Monitoring Tree**, go to **Services**, then **Transactions**, then **JTA Runtime** to see statistics for all transactions coordinated by server.
2. Optional: If you want to see the transaction statistics for only one server, click the server row in the table.
3. You can also expand the child nodes under **JTA Runtime** to view transaction details by transaction name or by resource, details about current transactions, or details about transaction recovery performed by the server.

View Current Transactions

You can view in-progress transactions coordinated by the selected server.

1. In the **Monitoring Tree**, go to **Environment**, then **Servers**, then choose the server whose transactions you want to view.
2. Under the server node, go to **Services**, then **Transactions**, then **JTA Runtime**.
3. Click the **Transactions** tab to see current transactions for the selected server.

Enable Local Domain Security for JTA Communication

Local domain security for JTA establishes trust between servers within a domain so that global transactions may occur across secure communication channels.

Note:

Local domain security extends the cross-domain protocol and its terminology and configuration reflect that origin. Nevertheless, local domain security is only applicable to local (intra-) domain communication.

If you need to secure JTA communication across separate domains, you should configure cross-domain security or the security interoperability mode. See How to Determine the Communication to Use for Inter-Domain Transactions in *Developing JTA Applications for Oracle WebLogic Server*.

1. In the **Edit Tree**, go to **Services**, then **JTA**.
2. Click **Show Advanced Controls**.
3. Turn on the **Local Domain Security Enabled** option.

   To reduce the impact on performance, WebLogic Server caches the authenticated subject. If you want to modify the cache settings for your environment, change the following settings:

   - To disable the cache, turn off **Local Domain Security Cache Enabled**.
   - To change how often the cache is cleared, update the **Local Domain Security Cache TTL** value (in seconds).
4. Click **Save** and then commit your changes.
5. Create a user for local domain security and assign it to the `CrossDomainConnectors` group. See Create a User.

   This user will be authorized to perform all JTA communication between servers in the domain.
6. Configure a local domain security credential mapping for the local domain security user. Use the default credential mapping provider or, if you want to configure your own credential mapping provider, see Configure a Credential Mapping Provider.

   a. In the **Security Data Tree**, go to **Realms**, then *myRealm*, then **Credential Mappers**, then **Remote Resources**.

   b. Click **New**.

   c. Turn on the **Use cross-domain protocol** option.

   d. In the **Remote Domain** field, enter the name of the *local* domain.

   e. In the **Remote User** field, enter the username of the user you configured in step 5. Then, in the **Remote Password** field, enter their password.

   f. Click **Create**.

Local domain security is now enabled for JTA communication.

Specify the JTA Security Interoperability Mode

The security interoperability (interop) mode determines the security subject for JTA communication between servers.

Note:

If local- or cross-domain security are enabled, they supersede the security interop mode.

1. In the **Edit Tree**, go to **Services**, then **JTA**.
2. Click **Show Advanced Controls**.
3. From the **Security Interoperability Mode** drop-down list, select a mode:
   - **Default**: Messages are forwarded using kernel identity *if* the **Admin** channel is also configured. Otherwise, it behaves like `performance` mode. See Configure the Domain-Wide Administration Port.
   - **Performance**: Messages are forwarded using an anonymous user.
4. Click **Save**.

10
Configuring Coherence

Oracle Coherence enables organizations to predictably scale mission-critical applications by providing fast and reliable access to frequently used data. By automatically and dynamically partitioning data in memory across multiple servers, Coherence enables continuous data availability and transactional integrity, even in the event of a server failure.

WebLogic Server domains include a Coherence container that simplifies the management and deployment of Coherence clusters and Coherence-based applications.

Note:

Coherence can also be installed in a standalone mode that does not rely on WebLogic Server. WebLogic Remote Console cannot administer standalone Coherence installations, therefore this documentation focuses solely on using Coherence with WebLogic Server.

To learn more about standalone Coherence, see Introducing the Oracle Coherence Standard Installation Topologies in *Installing Oracle Coherence*.

Coherence Clusters and Managed Coherence Servers

Coherence is integrated within WebLogic Server as a container subsystem, known as a Coherence cluster. The use of a container aligns the lifecycle of a Coherence member with the lifecycle of a Managed Server: starting or stopping a server JVM starts and stops a Coherence cluster member.

A WebLogic Server domain can contain a single Coherence cluster that can be associated with multiple WebLogic Server clusters. Within a Coherence cluster are Coherence cluster members.

Managed Coherence servers are Managed Servers that are configured to be Coherence cluster members. Managed Coherence servers provide in-memory distributed caching for applications to increase application scalability, availability, and performance.

Managed servers can be explicitly associated with a Coherence cluster or they can be associated with a WebLogic Server cluster that is associated with a Coherence cluster.

Managed Coherence servers that are part of a WebLogic Server cluster inherit their Coherence settings from the WebLogic Server cluster. WebLogic Server clusters are typically used to set up Coherence deployment tiers that organize managed Coherence servers based on their role in the Coherence cluster.

Note:

Similar to WebLogic Server clusters, Coherence clusters consist of multiple managed Coherence server instances. However, Coherence clusters use different clustering protocols and are configured separately from WebLogic Server clusters.

With Coherence clusters, a client interacts with the data in a local cache, and the distribution and backup of the data is automatically performed across cluster members.

For more information, see Configuring and Managing Coherence Clusters in *Administering Clusters for Oracle WebLogic Server*.

Note:

In previous releases, Coherence cluster members were organized into Coherence servers (also known as Coherence data nodes). This feature is deprecated and not supported by WebLogic Remote Console. Coherence tiers should be configured to use Managed Coherence servers instead.

Configure Coherence: Main Steps

For optimal performance or scalability, Coherence is typically set up using WebLogic Server clusters to represent deployment tiers.

1. Verify whether your domain's topology can support separate WebLogic Server clusters to represent Coherence deployment tiers. For guidance, see Cluster Architectures in *Administering Clusters for Oracle WebLogic Server*.

   Note:

   Alternatively, for development purposes, you can set up a single standalone Managed Server instance to act as both a cache server and a cache client. See Configure and Deploy Coherence on a Single-Server Cluster.

   Single-server topologies are not recommended for production use.

2. Configure a Coherence cluster as described in Create a Coherence Cluster.
3. Configure a data tier as described in Create a Coherence Data Tier.

   A Coherence data tier is a WebLogic Server cluster that is associated with a Coherence cluster and hosts any number of storage-enabled managed Coherence servers. For more information, see Configuring and Managing a Coherence Data Tier in *Administering Clusters for Oracle WebLogic Server*.
4. Configure an application tier as described in Create a Coherence Application Tier.

   A Coherence application tier is a WebLogic Server cluster that is associated with a Coherence cluster and hosts any number of storage-disabled managed Coherence servers. For more information, see Configuring and Managing a Coherence Application Tier in *Administering Clusters for Oracle WebLogic Server*.

5. Optional: Configure a proxy tier as described in Create a Coherence Proxy Tier.

   A Coherence proxy tier is a WebLogic Server cluster that is associated with a Coherence cluster and hosts any number of managed Coherence proxy servers. Managed Coherence proxy servers allow Coherence*Extend clients to use Coherence caches without being cluster members. For more information, see Configuring and Managing a Coherence Proxy Tier in *Administering Clusters for Oracle WebLogic Server*.

6. Create and package a Coherence Grid Archive (GAR) module for any application modules (Web application, EJB, and so on) that use Coherence. For more information, see Creating Coherence Applications for WebLogic Server in *Developing Oracle Coherence Applications for Oracle WebLogic Server*.

7. Deploy your Coherence GAR module to your WebLogic Server domain as described in Install an Application.

   Standalone GARs are deployed in the same way as other Java EE modules. For more information, see Deploying Coherence Applications in WebLogic Server in *Developing Oracle Coherence Applications for Oracle WebLogic Server*.

Create a Coherence Cluster

Coherence clusters enable applications to share data management and caching services among server instances and clusters hosting the applications that need access to them.

Configure cluster properties and then target Coherence clusters to WebLogic Server instances or clusters in the domain.

1. In the **Edit Tree**, go to **Environment**, then **Coherence Clusters**.
2. Click **New**.
3. Enter a name for the Coherence cluster and click **Create**.
4. Select a **Clustering Mode** and then adjust the Coherence general properties according to the clustering mode you selected.
5. Optional: If you want to specify operational settings that are not available through the provided MBeans, you can upload a cluster configuration file with your supplemental settings. Click **Import Configuration**, then, in the **Custom Cluster Configuration File Name** dialog box, enter the location of the cluster configuration file, relative to the domain configuration directory.

   > **Note:**
   >
   > Avoid configuring the same operational settings in both an external cluster configuration file and through the MBeans.

6. Click **Save**.

Next, you should consider updating the following configuration options:

- Target Coherence Clusters
- Configure Coherence Federation
- Configure Coherence Persistence
- Set Up the Coherence Security Framework

Configure Coherence Clusters

A Coherence cluster provides several cluster settings that can be configured for a specific domain.

You should update the default values of the Coherence cluster as needed for your environment.

Configure Coherence Federation

Use Coherence federation to federate cache data asynchronously across multiple geographically dispersed clusters. Federating cached data across clusters provides redundancy, off-site backup, and multiple points of access for application users in different geographical locations.

For more information, see Federating Caches Across Clusters in *Administering Oracle Coherence*.

1. In the **Edit Tree**, go to **Environment**, then **Coherence Clusters**.
2. Click the Coherence cluster that you want to edit, then select the **Coherence Federation** tab.
3. Select a **Federation Topology** from the drop-down list.

   When a topology is selected, a topology configuration is automatically created and named `Default-Topology`. The Default-Topology topology configuration is created and used if no other federation topology is specified in the cache configuration file.

   When using federation, complementary topologies must be configured on both the local and remote clusters. For example, if a local cluster is set to use `active-passive`, then the remote cluster must be set to use `passive-active`.
4. In the **Remote Coherence Cluster Name** field, enter the name of the remote cluster to which this cluster is being federated.
5. In the **Remote Participant Hosts** field, enter one or more hosts (separated by commas) that are running managed Coherence servers on the remote cluster.
6. If required, change the cluster port of the remote cluster in the **Remote Coherence Cluster Listen Port**. The default cluster port is typically not changed.
7. Click **Save**.

Configure Coherence Persistence

Use Coherence persistence to save and recover Coherence distributed caches. Cached data is persisted so that it can be quickly recovered after a catastrophic failure or after a cluster restart due to planned maintenance.

For more information on configuring persistence in Coherence, see Persisting Caches in *Administering Oracle Coherence*.

1. In the **Edit Tree**, go to **Environment**, then **Coherence Clusters**.
2. Click the Coherence cluster that you want to edit, then select the **Coherence Persistence** tab.
3. Choose a **Persistence Mode** from the drop-down list.

- **On-Demand**: In this mode, a cache service is manually persisted and recovered upon request using the persistence coordinator. The persistence coordinator is exposed as an MBean interface that provides operations for creating, archiving, and recovering snapshots of a cache service.
- **Active**: In this mode, cache contents are automatically persisted on all mutations and are automatically recovered on cluster/service startup. The persistence coordinator can still be used in active persistence mode to perform on-demand snapshots.
- **Active-Async**: In this mode, the storage servers can persist data asynchronously. Thus, a mutating request is successful after the primary stores the data and (if there is a synchronous backup) after the backup receives the update.
- **Active Backup**: In this mode, persistence behaves similarly to the active persistence mode but it also stores backup partitions asynchronously on a disk.

4. Optional: You can override the default locations where various persistence files are stored. Enter new file locations in the appropriate fields:
   - **Active Directory**
   - **Snapshot Directory**
   - **Trash Directory**
   - **Backup Directory**
   - **Events Directory**
5. Click **Save**.

Configure Coherence Logging

You can configure logging properties for a Coherence cluster.

1. In the **Edit Tree**, go to **Environment**, then **Coherence Clusters**.
2. Click the Coherence cluster that you want to edit, then select the **Logging** tab.
3. Make sure the **Enabled** option is turned on.
4. In the **Logger Name** field, enter the logger name for the Coherence logs.
5. In the **Message Format** field, enter the Coherence logs message format.
6. In the **Severity Level** field, enter the logging severity level for Coherence logs. The maximum level you can set is `9`.
7. In the **Character Limit** field, enter the character limit for Coherence logs.
8. Click **Save**.

Set Up the Coherence Security Framework

You can enable and configure the Coherence security framework. If you do not enable security, then any non-WebLogic Server JVM can access the Coherence cluster without going through WebLogic Server authorization.

For more information, see Securing Oracle Coherence in Oracle WebLogic Server in *Securing Oracle Coherence*.

1. In the **Edit Tree**, go to **Environment**, then **Coherence Clusters**.
2. Click the Coherence cluster that you want to edit, then select the **Security** tab.

3. Turn on the **Security Framework Enabled** option and then configure the security parameters to meet the requirements of your environment.

   > **Note:**
   >
   > If the **Secured Production** option is not set, Coherence will inherit its domain mode from WebLogic Server.

4. Click **Save**.
5. Enable an Identity Asserter for a Coherence cluster to assert a client's identity token. For local extend clients, an identity asserter is already enabled for asserting a token of type `weblogic.security.acl.internal.AuthenticatedSubject`. For remote (outside of WebLogic Server) extend clients, a custom identity asserter implementation class must be packaged in a GAR. However, an identity asserter is not required if the remote extend client passes `null` as the token. If the proxy service receives a non-null token and there is no identity asserter implementation class configured, a `SecurityException` is thrown and the connection attempt is rejected.
   a. Under the current Coherence cluster node in the navigation tree, select **Coherence Identity Asserter**.
   b. In the **Class Name** field, enter the fully qualified name of the asserter class. For example, to use the default identity asserter, enter `com.tangosol.net.security.DefaultIdentityAsserter`.
   c. Click **Save**.
   d. If there are any arguments, open the **Identity Asserter Constructor Arguments** node and click **New** to add class constructor arguments.
   e. Click **Save**.

If you want to define WebLogic Server roles and policies for authorizing access to Coherence services and caches, see Secure a Coherence Service Configuration or Secure Coherence Caches, respectively.

Target Coherence Clusters

Target Coherence clusters to WebLogic Server instances or clusters that host applications that need access to Coherence data caches and services.

1. In the **Edit Tree**, go to **Environment**, then **Coherence Clusters**.
2. Click the Coherence cluster that you want to edit, then select the **Members** tab.
3. Select the servers or clusters where you want to deploy the Coherence cluster configuration and move them under **Chosen**. Move any unwanted servers or clusters under **Available**.
4. Click **Save**.

Create a Cluster Cache Configuration

A Coherence cache configuration file defines the caches and services that are used by an application. Typically, a cache configuration file is included in a GAR module.

A GAR is deployed to all managed Coherence servers in the data tier and can also be deployed as part of an EAR to the application tier. The GAR ensures that the cache configuration is available on every Coherence cluster member.

However, there are cases that require a different cache configuration file to be used on specific managed Coherence servers. For example, a proxy tier requires access to all artifacts in the GAR but needs a different cache configuration file that defines the proxy services to start.

You can use WebLogic Remote Console to define a cluster cache configuration. A cache configuration file can be associated with WebLogic clusters or Managed Coherence Servers at runtime. In this case, the cache configuration overrides the cache configuration file that is included in a GAR. You can also omit the cache configuration file from a GAR file and assign it at runtime.

To override a cache configuration file at runtime, the cache configuration file must be bound to a JNDI name. The JNDI name is defined using the `override-property` attribute of the `<cache-configuration-ref>` element. The element is located in the `coherence-application.xml` file that is packaged in a GAR file. For more information on the `coherence-application.xml` file, see coherence-application.xml Deployment Descriptor Elements in *Developing Oracle Coherence Applications for Oracle WebLogic Server*. For more information on importing a cache configuration file, see Overriding a Cache Configuration File in *Administering Clusters for Oracle WebLogic Server*.

1. If you haven't already done so, create a Coherence cluster. See Create a Coherence Cluster.
2. In the **Edit Tree**, go to **Environment**, then **Coherence Clusters**, then *myCoherenceCluster*, then **Coherence Cache Configs**.
3. Click **New**.
4. In the **Name** field, enter a name for the Coherence cache configuration.
5. In the **JNDI Name** field, enter the JNDI name to which the cache configuration file is bound.

   This JNDI name is prefixed with `cache-config/` and must match the value in the `override-property` of your `cache-configuration-ref` entry in `coherence-application.xml`.

   For example, JNDI name of `ExamplesGAR` must match the `override-property` of `cache-config/ExamplesGAR`.
6. In the **Cache Configuration File** field, enter the full path to the cache configuration file. Alternatively, you may specify a URL.
7. Click **Create**.
8. On the **Targets** tab, select the servers or clusters where you want to target the cluster cache configuration and move them under **Chosen**. Move unwanted servers or clusters under **Available**.
9. Click **Save**.

Secure Coherence Caches

Use WebLogic Server authorization to restrict access to specific Coherence caches.

1. If you haven't already done so, create a Coherence cluster. See Create a Coherence Cluster.
2. In the **Edit Tree**, go to **Environment**, then **Coherence Clusters**, then *myCoherenceCluster*, then **Coherence Caches**.
3. Click **New**.
4. In the **Name** field, enter a name for the Coherence cache. The name of the cache must match *exactly* the name of the cache used in an application.
5. Click **Create** and commit your changes.
6. Apply a security role that is scoped to the Coherence cache. See Create a Scoped Role.

Secure a Coherence Service Configuration

You can use WebLogic Server authorization to restrict access to Coherence services. Specifying authorization on a cache service affects access to all the caches that are created by that service.

1. In the **Edit Tree**, go to **Environment**, then **Coherence Clusters**, then *myCoherenceCluster*, then **Coherence Services**.
2. Click **New**.
3. In the **Name** field, enter a name for the Coherence service. The name of the service must match *exactly* the name of the service used in an application.

   Note:

   The exact name must include the scope name as a prefix to the service name. The scope name can be explicitly defined in the cache configuration file or, more commonly, taken from the deployment module name. For example, if you deploy a GAR named `contacts.gar` that defines a service named `ContactsService`, then the exact service name is `contacts:ContactsService`.

4. Click **Create** and commit your changes.
5. Apply a security role that is scoped to the Coherence service. See Create a Scoped Role.

Specify a Coherence Well Known Address

Configure a well known address for a Coherence cluster. Other members can use this address to enroll in the cluster.

1. In the **Edit Tree**, go to **Environment**, then **Coherence Clusters**, then *myCoherenceCluster*, then **Coherence Cluster Well Known Addresses**.
2. Click **New**.
3. In the **Name** field, enter a name for the Coherence well known address.
4. In the **Listen Address** field, enter the IP address of the Coherence cluster member.

5. Click **Create**.

Configure a Coherence Address Provider

An address provider specifies the TCP listener address (IP or DNS name, and port) for a proxy service.

1. In the **Edit Tree**, go to **Environment**, then **Coherence Clusters**, then *myCoherenceCluster*, then **Coherence Address Providers**.
2. Click **New**.
3. In the **Name** field, enter a name for the Coherence address provider.
4. Click **Create**.

   A node for the new address provider will appear under the **Coherence Address Providers**.
5. Expand the new node and go to **Coherence Socket Addresses**.
6. Click **New**.
7. Enter a name for the socket address.
8. In the **Address** field, specify the listen address for your socket address.
9. In the **Port** field, specify the listen port for your socket address.
10. Click **Create**.

Coherence Deployment Tiers

In production environments, Coherence is typically set up using WebLogic Server clusters to represent deployment tiers.

There are three types of Coherence deployment tiers:

- **Data Tier** which hosts one or more cache servers
- **Application Tier** which hosts one or more cache clients
- **Proxy Tier** which hosts one or more managed Coherence proxy servers and the Coherence extend client tier that hosts extend clients

The tiered topology approach provides optimal scalability and performance.

For more information on Coherence deployment tiers, see Creating Coherence Deployment Tiers in *Administering Clusters for Oracle WebLogic Server*.

Create a Coherence Data Tier

A Coherence data tier is a WebLogic Server cluster that is associated with a Coherence cluster and hosts any number of storage-enabled managed Coherence servers.

For more information on Coherence data tiers, see Configuring and Managing a Coherence Data Tier in *Administering Clusters for Oracle WebLogic Server*.

1. If you haven't already done so, create a Coherence cluster as described in Create a Coherence Cluster.
2. Create a WebLogic Server cluster as described in Create a Cluster.

3. On the WebLogic Server cluster that you just created, go to the **Advanced** tab, then the **Coherence** subtab.
4. From the **Coherence Cluster System Resource** drop-down list, select the Coherence cluster.
5. Turn on the **Local Storage Enabled** option.
6. Click **Save**.
7. Create one or more managed Coherence servers as described in Create a Managed Coherence Server and assign them to this data tier.

You should also create a Coherence application tier and, optionally, a Coherence proxy tier. See Create a Coherence Application Tier or Create a Coherence Proxy Tier.

Create a Coherence Application Tier

A Coherence application tier is a WebLogic Server cluster that is associated with a Coherence cluster and hosts any number of storage-disabled managed Coherence servers.

For more information on Coherence application tiers, see Configuring and Managing a Coherence Application Tier in *Administering Clusters for Oracle WebLogic Server*.

1. If you haven't already done so, create a Coherence cluster as described in Create a Coherence Cluster.
2. Create a WebLogic Server cluster as described in Create a Cluster.
3. On the WebLogic Server cluster that you just created, go to the **Advanced** tab, then the **Coherence** subtab.
4. From the **Coherence Cluster System Resource** drop-down list, select the Coherence cluster.
5. Turn off the **Local Storage Enabled** option. Servers in the application tier should never be used to store cache data.
6. Click **Save**.
7. Create one or more managed Coherence servers as described in Create a Managed Coherence Server and assign them to this application tier.

You should also create a Coherence data tier and, optionally, a Coherence proxy tier. See Create a Coherence Data Tier or Create a Coherence Proxy Tier.

Create a Coherence Proxy Tier

A Coherence proxy tier is a WebLogic Server cluster that is associated with a Coherence cluster and hosts any number of managed Coherence proxy servers. Managed Coherence proxy servers allow Coherence*Extend clients to use Coherence caches without being cluster members.

The number of managed Coherence proxy servers that are required in a proxy tier depends on the number of expected clients. At least two proxy servers must be created to allow for load balancing. However, additional servers may be required when supporting a large number of client connections and requests.

For details on Coherence*Extend and creating extend clients, see Introduction to Coherence*Extend in *Developing Remote Clients for Oracle Coherence*.

1. If you haven't already done so, create a Coherence cluster as described in Create a Coherence Cluster.

2. Create a WebLogic Server cluster as described in Create a Cluster.
3. On the WebLogic Server cluster that you just created, go to the **Advanced** tab, then the **Coherence** subtab.
4. From the **Coherence Cluster System Resource** drop-down list, select the Coherence cluster.
5. Turn off the **Local Storage Enabled** option. Servers in the proxy tier should never be used to store cache data.
6. Click **Save**.
7. Create one or more managed Coherence servers as described in Create a Managed Coherence Server and assign them to this proxy tier.

You should also create a Coherence data tier and a Coherence application tier. See Create a Coherence Data Tier and Create a Coherence Application Tier.

Create a Managed Coherence Server

A Managed Coherence server is a Managed Server that is configured to be Coherence cluster member and provide in-memory distributed caching for applications.

1. Create a WebLogic Server Managed Server, as described in Create a Managed Server.
2. On the Managed Server that you just created, from the **Cluster** drop-down list, select a cluster that is acting as a Coherence deployment tier.

   For information on Coherence deployment tiers, see Coherence Deployment Tiers.
3. Click **Save**.

Configure and Deploy Coherence on a Single-Server Cluster

During development, a single-server cluster offers a quick way to start and stop a cluster. A single-server cluster is a cluster that is constrained to run on a single managed server instance and does not access the network. The server instance acts as a storage-enabled cluster member, a client, and (optionally) a proxy.

For more information on using a single-server Coherence cluster, see Using a Single-Server Cluster in *Administering Clusters for Oracle WebLogic Server*.

Note:

To set up Coherence for production use, follow the steps outlined in Configure Coherence: Main Steps instead.

1. If you haven't already done so, create a Coherence cluster as described in Create a Coherence Cluster.

   If you are using multicast communication, set **Time To Live** to `0`. For more information about communication for Coherence clusters, see Configure Cluster Communication in *Administering Clusters for Oracle WebLogic Server*.
2. Create a WebLogic Server Managed Server, as described in Create a Managed Server.

   Do not create a managed *Coherence* server.

3. On the Managed Server that you just created, go to the **Advanced** tab and then its **Coherence** subtab.
4. From the **Coherence Cluster System Resource** drop-down list, select the Coherence cluster.

   This standalone Managed Server will now inherit settings directly from the Coherence cluster.
5. Turn on the **Local Storage Enabled** option to ensure the Managed Server will be a storage-enabled Coherence member (cache server).
6. Set the **Unicast Listen Address** to an address that is routed to loop back. On most computers, setting the address to `127.0.0.1` works.

   For more information about unicast settings for Coherence cluster members, see Configure Coherence Cluster Member Unicast Settings in *Administering Clusters for Oracle WebLogic Server*.
7. Click **Save**.
8. Create and package a Coherence Grid Archive (GAR) module for any application modules (Web application, EJB, etc.) that use Coherence. See Creating Coherence Applications for WebLogic Server in *Developing Oracle Coherence Applications for Oracle WebLogic Server*.
9. Deploy your Coherence GAR module to your WebLogic Server domain. Standalone GARs are deployed in the same way as other Jakarta EE modules. See Deploying Coherence Applications in WebLogic Server in *Developing Oracle Coherence Applications for Oracle WebLogic Server*.

11
Scheduling Work

You can use Work Managers to prioritize work based on rules that you define and by monitoring actual run time performance statistics to then optimize the performance of your applications.

Work Managers can be applied globally to a WebLogic Server domain.

A Work Manager defines a set of request classes and thread constraints that manage work performed by WebLogic Server.

A request class defines a fair share thread entitlement, a response time goal, or a context policy for a particular class of application request.

Thread constraints define the maximum number of threads to allocate for requests, the minimum number of threads to use for resolving deadlocks, and the total number of requests that can be queued or running before WebLogic Server begins rejecting requests.

See Using Work Managers to Optimize Scheduled Work in *Administering Server Environments for Oracle WebLogic Server*.

Create a Global Work Manager

You can create global Work Managers that are used to prioritize thread execution.

1. In the **Edit Tree**, go to **Scheduling**, then **Work Managers**.
2. Click **New**.
3. Enter a name for the new Work Manager and click **Create**.
4. Update the default values as appropriate.
5. On the **Targets** tab, select the servers or clusters on which you plan to deploy applications that reference the Work Manager and move them under **Chosen**. Move unwanted servers or clusters under **Available**.
6. Click **Save**.

Next, you must create at least one request class or constraint. The global Work Manager uses request classes and constraints to determine how to prioritize work. See Create Request Classes and Create Constraints.

Create Request Classes

Request classes express a scheduling guideline that WebLogic Server uses to allocate threads to requests.

1. In the **Edit Tree**, go to **Scheduling**, then choose one of the request class types:
   - **Context Request Classes**
   - **Fair Share Request Classes**
   - **Response Time Request Classes**

For information on the different request class types, see Request Classes in *Administering Server Environments for Oracle WebLogic Server*.

2. Click **New**.
3. Enter a name for the new request class and, for Response Time request classes, update the **Response Time Goal** value.
4. Click **Create**.
5. On the **Targets** tab, select the servers or clusters on which you plan to deploy applications that reference this request class and move them under **Chosen**.

   The servers or clusters that you select must share at least one target with the applicable Work Manager.
6. Click **Save**.

Next, you must assign this request class to a Work Manager to use it. See Assign a Request Class or Constraints to a Work Manager.

Create Constraints

A constraint defines the minimum and maximum numbers of threads allocated to execute requests and the total number of requests that can be queued or executing before WebLogic Server begins rejecting requests.

1. In the **Edit Tree**, go to **Scheduling**, then choose one of the constraint types:
   - **Capacities**
   - **Max Threads Constraints**
   - **Min Threads Constraints**

   For information on the different request class types, see Constraints in *Administering Server Environments for Oracle WebLogic Server*.
2. Click **New**.
3. Enter a name for the new constraint and click **Create**.
4. Update the default values for the constraint as necessary. Click **Save** when you are done with your changes.
5. On the **Targets** tab, select the servers or clusters on which you plan to deploy applications that reference this constraint and move them under **Chosen**.

   The servers or clusters that you select must share at least one target with the applicable Work Manager.
6. Click **Save**.
7. Optional: Repeat to add another type of constraint, if necessary.

   Each Work Manager can contain only one constraint of each type.

Next, you must assign this constraint to a Work Manager to use it. See Assign a Request Class or Constraints to a Work Manager.

Assign a Request Class or Constraints to a Work Manager

Before you can use the settings defined in a request class or constraint, you must assign it to a Work Manager.

Note:

- **Request Classes**: Each Work Manager can contain only *one* request class, but you can share request classes among multiple Work Managers.
- **Constraints**: Each Work Manager can contain only one constraint of each *type*, but you can share constraints among multiple Work Managers.

1. If you haven't done so already, create a global Work Manager. See Create a Global Work Manager.
2. In the **Edit Tree**, go to **Scheduling**, then **Work Managers** and select the Work Manager where you want to assign the request class or constraints.
3. Use the applicable drop-down lists to choose a request class or constraints to assign to this Work Manager.
4. Click **Save**.

Create a Work Manager Shutdown Trigger

You can define how a Work Manager should handle stuck threads.

For more information, see Stuck Thread Handling in *Administering Server Environments for Oracle WebLogic Server*.

1. If you haven't done so already, create a global Work Manager. See Create a Global Work Manager.
2. In the **Edit Tree**, go to **Scheduling**, then **Work Managers**, then *myWorkManager*, then **Work Manager Shutdown Trigger**.
3. Click **Create**.
4. Update the default values as needed.
5. Click **Save**.

Concurrent Managed Object Templates

WebLogic Server provides concurrency capabilities to Jakarta EE applications by using Concurrent Managed Object (CMO) templates to make threads container-managed. You can configure CMO templates and then make them available for use by application components, such as servlets and EJBs.

You can define three types of CMO templates:

- Managed Executor Service Template - Used by applications to execute submitted tasks asynchronously. See Create a Global Managed Executor Service Template.

- Managed Scheduled Executor Service Template - Used by applications to execute submitted tasks asynchronously at specific times. See Create a Global Managed Scheduled Executor Service Template.
- Managed Thread Factory Template - Used by applications to create managed threads. See Create a Global Managed Thread Factory Template.

For more information, see Configuring Concurrent Managed Objects in *Administering Server Environments for Oracle WebLogic Server*.

Create a Global Managed Executor Service Template

You can create managed executor service (MES) templates that are used by applications to execute submitted tasks asynchronously.

1. In the **Edit Tree**, go to **Scheduling**, then **Managed Executor Service Templates**.
2. Click **New**.
3. Enter a name for the new MES template.
4. Click **Create**.
5. If you want to use a non-default Work Manager with this MES template, in the **Dispatch Policy** field, enter the name of the custom Work Manager.
6. If you want to specify the priority of the long running daemon thread, update the value in the **Long Running Threads Priority** field.
7. If you want to define the maximum number of concurrent long-running tasks submitted to this MES template in the domain, update the value in the **Max Concurrent Long Running Requests** field.
8. Click **Save**.
9. On the **Targets** tab, select the servers or clusters where this MES template will be accessible and move them under **Chosen**. Move unwanted servers or clusters under **Available**.

   Only applications that have been deployed to the selected servers and clusters can use this MES template.
10. Click **Save**.

Create a Global Managed Scheduled Executor Service Template

You can create managed scheduled executor service (MSES) templates that are used by applications to execute submitted tasks asynchronously at specific times.

1. In the **Edit Tree**, go to **Scheduling**, then **Managed Scheduled Executor Service Templates**.
2. Click **New**.
3. Enter a name for the new MSES template.
4. Click **Create**.
5. If you want to use a non-default Work Manager with this MSES template, in the **Dispatch Policy** field, enter the name of the custom Work Manager.
6. If you want to specify the priority of the long running daemon thread, update the value in the **Long Running Threads Priority** field.

7. If you want to define the maximum number of concurrent long-running tasks submitted to this MSES template in the domain, update the value in the **Max Concurrent Long Running Requests** field.
8. Click **Save**.
9. On the **Targets** tab, select the servers or clusters where this MSES template will be accessible and move them under **Chosen**. Move unwanted servers or clusters under **Available**.

   Only applications that have been deployed to the selected servers and clusters can use this MSES template.
10. Click **Save**.

Create a Global Managed Thread Factory Template

You can create managed thread factory (MTF) templates that are used by applications to create managed threads.

1. In the **Edit Tree**, go to **Scheduling**, then **Managed Thread Factory Templates**.
2. Click **New**.
3. Enter a name for the new MTF template.
4. Click **Create**.
5. If you want to specify the priority assigned to the thread, update the value in the **Priority** field. The greater the number, the higher the priority.
6. If you want to define the maximum number of threads that are created by the MTF and are still executing the `run()` method of the tasks, update the value in the **Max Concurrent New Threads** field.
7. Click **Save**.
8. On the **Targets** tab, select the servers or clusters where this MTF template will be accessible and move them under **Chosen**. Move unwanted servers or clusters under **Available**.

   Only applications that have been deployed to the selected servers and clusters can use this MTF template.
9. Click **Save**.

12
Interoperating with Oracle Tuxedo

WebLogic Server supports interoperability between WebLogic Server applications and Tuxedo services.

Use WebLogic Remote Console to manage the connections between the two services using one of the following options:

- Use the **WebLogic Tuxedo Connector (WTC)** to develop and support applications interoperating between WebLogic Server and Tuxedo by using a Java Application-to-Transaction Monitor Interface (JATMI) (similar to Tuxedo ATMI) or by using RMI over IIOP applications and Tuxedo CORBA remote objects. See Create a WTC Server.
- Use **Jolt Connection Pools** to enable Tuxedo ATMI services for the Web, using WebLogic Server as the front-end HTTP and application server. See Create a Jolt Connection Pool.

For a comparison of these options, see How WebLogic Tuxedo Connector Differs from Jolt in *Administering WebLogic Tuxedo Connector for Oracle WebLogic Server*.

Create a WTC Server

The WebLogic Tuxedo Connector (WTC) provides interoperability between WebLogic Server applications and Tuxedo services. It allows WTC clients to invoke Tuxedo services and Tuxedo clients to invoke WebLogic Server applications, such as EJBs and servlets.

To configure the WTC, you must create a WTC server, which organizes the various attributes necessary to establish a session connection between WebLogic Server and Tuxedo.

1. In the **Edit Tree**, go to **Interoperability**, then **WTC Servers**.
2. Click **New**.
3. Enter a name for the WTC Server.
4. Click **Create**.
5. On the **Targets** tab, select the servers or clusters where you want to deploy this WTC server.

Next, you must create at least one local Tuxedo access point. See Create a Local Access Point.

Configure WTC Servers

After you create an WTC server, you can configure how it to determine how WebLogic Server should interact with the Tuxedo environment.

1. In the **Edit Tree**, go to **Interoperability**, then **WTC Servers**, then *myWTCServer*, and select the configuration parameter you want to edit.

See Configuring WebLogic Tuxedo Connector for Your Applications in *Administering WebLogic Tuxedo Connector for Oracle WebLogic Server*.

Table 12-1 Tuxedo Interactions

Option	Description
Local Access Point	Local Tuxedo access points provide configuration information to connect available remote Tuxedo domains to a WTC server.
Remote Acess Point	Remote Tuxedo access points provide configuration information to connect a WTC server to available remote Tuxedo domains.
Exported Services	Exported services provide information about how to provide Java application services to remote Tuxedo application environments.
Imported Services	Imported services provide information on how to access services that are available on remote Tuxedo domains.
Passwords	Password configurations provide passwords for inter-domain authentication through access points.
Resources	Resources specify field table classes, reference view buffer structures, and provide application passwords for domains.
Queuing Bridge	A Tuxedo queuing bridge provides a bidirectional JMS interface that allows WTC applications to communicate with Tuxedo application environments.
Redirections	Redirections are used to configure a one-to-one connection between the JMS interface and Tuxedo application environment.

Create a Local Access Point

Local Tuxedo access points provide configuration information to connect available remote Tuxedo domains to a WTC server.

You must have at least one local Tuxedo access point configured to create a valid WTC server.

1. In the **Edit Tree**, go to **Interoperability**, then **WTC Servers**, then *myWTCServer*, then **Local APs**.
2. Click **New**.
3. Enter a name for the local Tuxedo access point.
4. Click **Create**.
5. Update the other local access point attributes as necessary.
6. Click **Save**.
7. If you want to define the connection configurations of a local Tuxedo access point that will be used with this WTC server, then, on the **Connections** tab, update the appropriate attributes.
8. If you want to define the security configurations of a local Tuxedo access point that will be used with this WTC server, then, on the **Security** tab, update the appropriate attributes.
9. Click **Save**.

Create a Remote Access Point

Remote Tuxedo access points provide configuration information to connect a WTC server to available remote Tuxedo domains.

1. In the **Edit Tree**, go to **Interoperability**, then **WTC Servers**, then *myWTCServer*, then **Remote APs**.
2. Click **New**.
3. Enter a name for the remote access point.
4. Click **Create**.
5. Update the other remote access point attributes as necessary.
6. Click **Save**.
7. If you want to define the connection configurations of a remote Tuxedo access point that will be used with this WTC server, then, on the **Connections** tab, update the appropriate attributes.
8. If you want to define the security configurations of a remote Tuxedo access point that will be used with this WTC server, then, on the **Security** tab, update the appropriate attributes.
9. Click **Save**.

Create a Jolt Connection Pool

Jolt is a Java-based client API that manages requests from servlets and applications on WebLogic Server to Oracle Tuxedo services.

It is recommended that you create one Jolt connection pool for each application running on WebLogic Server.

1. Make sure that you have properly configured Oracle Tuxedo to use Jolt. See Configuring the Oracle Jolt System in *Using Oracle Jolt*.
2. On each WebLogic Server instance that hosts a Jolt connection pool, you must configure Jolt classes to run as server startup up and shutdown classes. These classes establish and terminate the connection between Tuxedo and WebLogic Server.
    a. Configure the Jolt startup classes, as described in Configure Startup Classes.
        - In the **Class Name** field, enter `bea.jolt.pool.servlet.weblogic.PoolManagerStartUp`.
        - Enable **Failure is Fatal**.
        - Target the startup class to the server or clusters that will host the Jolt connection pools.

    b. Configure the Jolt shutdown classes, as described in Configure Shutdown Classes.
        - In the **Class Name** field, enter `bea.jolt.pool.servlet.weblogic.PoolManagerShutDown`.
        - Target the shutdown class to the server or clusters that will host the Jolt connection pools.

    c. Make sure you add the Jolt startup and shutdown classes to the classpath.

    d. Restart each server where you deployed the classes.
3. In the **Edit Tree**, go to **Interoperability**, then **Jolt Connection Pools**.
4. Click **New**.
5. Enter a name for the Jolt connection pool.
6. Click **Create**.

7. Update the other attributes for the Jolt connection pool as necessary.
8. Click **Save**.

Part III
WebLogic Deploy Tooling

WebLogic Deploy Tooling (WDT) lets you easily create and manage WebLogic Server environments. In WDT, a WebLogic domain and its properties are expressed as a YAML set, which creates a metadata model that describes the domain it will create or the changes it will implement.

These WDT metadata models establish a process that is easy to manipulate, verify, and repeat. For more information on WDT, see WebLogic Deploy Tooling.

WebLogic Remote Console lets you create WDT model files within its console. Although you cannot deploy or build a live domain from within WebLogic Remote Console, the inclusion of WDT within WebLogic Remote Console provides a familiar and graphical interface for the various WebLogic properties, which you can use to map to their equivalent YAML keys.

13
WDT Model Files

WDT metadata model files are descriptions of a WebLogic Server domain configuration. These models are not connected to a running WebLogic Server domain; you make edits to a model file and then use WebLogic Deploy Tooling (WDT) to build or modify live domains.

WDT model files are simplistic representations of a domain. They are generally written in `YAML` but WebLogic Remote Console also accepts models in `JSON` format. For brevity, a WDT model file only describes departures from the default configuration.

Note:

WebLogic Remote Console does not validate the data that you insert into WDT model files. It will accept changes or values that are invalid and that will present problems when the model file is used to build or update a domain.

Example 13-1 WDT Model

```
topology:
    Server:
        AdminServer:
        ManagedServer1:
            Cluster: Cluster2
        ManagedServer2:
            Cluster: Cluster2
        ManagedServer3:
    ServerTemplate:
        ServerTemplate1:
    Cluster:
        Cluster1:
            DynamicServers:
                ServerTemplate: ServerTemplate1
                DynamicClusterSize: 3
        Cluster2:
resources:
    JDBCSystemResource:
        DataSource1:
            JdbcResource:
                JDBCDriverParams:
                    DriverName: oracle.jdbc.replay.OracleXADataSourceImpl
                    URL: 'jdbc:oracle:thin:@//dbhost:1521/Database1'
                JDBCDataSourceParams:
                    JNDIName: [
                        jdbc/myDS
                    ]
                    GlobalTransactionsProtocol: TwoPhaseCommit
                DatasourceType: GENERIC
            Target: [
                Cluster2
            ]

appDeployments:
    Application:
        Application1:
            Target: [
                Cluster1
            ]
            SourcePath: /apps/benefits/benefits.war
            StagingMode: default
```

WDT Model Tokens

Use a WDT model token to increase the flexibility of a WDT model file by replacing a fixed value with a dynamic one.

With WDT model tokens, you can create WDT model files that adapt based on the applicable WDT tokens. Rather than creating multiple WDT model files whose only difference is a few fixed values, you can create a single file and update the tokens as needed.

WebLogic Remote Console supports multiple types of WDT model tokens. All tokens follow this format: `@@TYPE:KEY@@` where `TYPE` is the model token type and `KEY` is the variable value. For more information on the different types of WDT model tokens and their syntax, see Model Tokens in *WebLogic Deploy Tooling*.

WebLogic Remote Console allows you to insert WDT model tokens in two ways:

- Standalone WDT model tokens
- WDT variables

Standalone WDT Model Tokens

Standalone WDT model token are tokens that replaced when the WDT model file is passed through WebLogic Deploy Tooling or through another external process. WebLogic Remote Console knows nothing about the value of the token and cannot update it.

WDT Variables

WDT variables allow you to manage WDT model tokens and their values from within WebLogic Remote Console. First, you create a property list with a set of tokens and their values, then you assign a property list to a WDT model file. After they are linked, any properties that you create in the property list become available to insert into the associated WDT model file. By centralizing model tokens in a property lists, you can easily update their values and review available tokens.

For more information, see Property Lists.

Each WDT model file can only pull from one property list, but you can use the same property list for multiple WDT model files.

Note:

You can use both standalone WDT model tokens and WDT variables in the same WDT model file. This can be useful if you want to synchronize certain values and leave other values as slightly more static or to be changed by a later process.

Create a WDT Model File

To create a WDT model file for use with the WebLogic Deploy Tooling:

1. Open the Providers drawer and beside the project name, click **More** ⋮. Select **Create Provider for New WDT Model File**.
2. Enter a name for the WDT model file provider.
3. In the **WDT Model Filename** field, enter a name for the WDT model file. Include `.yaml` or `.json` at the end of the file name.
4. Click **Pick Directory** and browse to the directory where you want to save the new WDT model file.
5. Optional: Enable **Use Sparse Template** to create a WDT model file which does not contain any references to an Administration Server.
6. Optional: If you want to use WDT variables with this WDT model file, then from the **WDT Variables** dropdown list, choose a property list provider. If you don't have a property list provider yet, you can add one later.
7. Click **OK** to create the WDT model file.

Upload a WDT Model File

If you created a WDT model file elsewhere, you can upload it to WebLogic Remote Console and continue to edit it.

1. Open the Providers drawer and beside the project name, click **More** ⋮. Select **Add WDT Model File Provider**.
2. Enter a name for the WDT model file provider.
3. Click **Upload File** and browse to the directory where you saved the WDT model file.

   The WDT model file must be in `YAML` or `JSON` format.
4. Optional: If you want to use WDT variables with this WDT model file, then from the **WDT Variables** drop-down list, choose a property list provider. If you don't have a property list provider yet, you can edit this provider's settings later to add one.
5. Click **OK** to upload the WDT model file.

Edit a WDT Model File

To make changes to a WDT model file:

Note:

For guidance on where to find specific domain configuration options and how to apply them, review the tasks under Administration Server. The WDT Model Tree perspective is very similar to the Edit Tree perspective in an Administration Server provider.

1. Open the WDT model file that you want to edit.
2. Click **WDT Model Tree** and go to the node where you want to make your changes.
3. Beside the domain configuration that you want to modify, click **WDT Settings** to open the WDT settings dialog box.
4. Set a new value for the field, using one of the following options:

Option	Description
Default (unset)	Restore field to its default value.
Select Value	Select a reference to a component that exists in the current WDT model file.
Enter Value	Enter a fixed value.
Enter Model Token	Enter a WDT model token.
Enter Unresolved Reference	Enter a reference to a component that does not exist in the current WDT model file but will exist at a later point.
Select Model Token Variable	Select a WDT model token variable from the list of available options. The WDT model file must be connected with a property list to see this option.
Create Model Token Variable	Enter a Variable Name and Variable Value to create a new WDT model token variable. New model token variables are added to the connected property list. The WDT model file must be connected with a property list to see this option.

5. Click **Save Now** to update the `YAML` file with your changes.

If you want to restore fields to their default value, right-click on a field and click **Restore to default**.

Build a WebLogic Server Domain

When you're satisfied with the properties of your WDT model file, you can transform it into a live WebLogic Server domain with WebLogic Deploy Tooling (WDT).

1. Save the WDT model file in WebLogic Remote Console. Make a note of the location of the `YAML` file on your computer.
2. Download the latest version of WDT from the WDT GitHub Repository.
3. Follow the instructions in the WDT Documentation for creating a domain from a WDT model file.

14

WDT Composite Models

WDT composite models are multiple WDT model files that have been merged together. WDT composite models are read-only files that are useful for comparison.

Note:

If there are conflicting properties between WDT model files, then the last WDT model file added overrides the properties of the previous file.

Create a WDT Composite Model

Combine two or more WDT model files.

There must be at least two WDT model file providers in the active project.

1. Open the Providers drawer and beside the project name, click **More** ⋮ . Select **Add WDT Composite Model File Provider**.
2. Enter a name for the WDT Composite Model.
3. Click inside the **WDT Models** field and select the WDT model files that you want to appear in the WDT Composite Model in the order that you want them to appear. Only WDT model files in the current project will appear in the drop-down list.

   Choose the order that the WDT model files are added to the composite carefully. Each subsequent WDT model file overrides the properties of the previous file. If there are any conflicting properties, the properties of the last WDT model file added to the composite takes precedence.
4. Click **OK** to create the WDT composite model.

The selected WDT model files are merged and you can view the combined set in the **WDT Composite Model Tree**.
You can add or remove WDT model files from a WDT composite model as needed. Removing a WDT model file from a WDT composite model (or deleting the WDT composite model entirely) does not affect the constituent WDT model files.

15 Property Lists

Property lists are simple key-value pairs that facilitate the WDT variable feature in WDT model files. Use property lists to create a collection of key-value pairs that WebLogic Remote Console can insert into a WDT model file.

For more information on how property lists can enhance WDT model files, see WDT Model Tokens.

You may want to create a single property list that applies to multiple WDT model files or multiple, discrete files for each WDT model file, depending on your usage.

Be careful when editing property list key *names*. If you change a key name, WebLogic Remote Console registers that change as a *new* key-value pair. If that key was actively used by a WDT model file, it becomes a standalone WDT model token instead of a WDT variable. Conversely, if you add a key name that matches an existing standalone WDT model token, WebLogic Remote Console converts that the previously standalone token into a WDT variable.

You can also independently create a property list and then upload it to WebLogic Remote Console. Save the property list as `*.properties`, `*.props`, or another text file format. Place each key-value pair on a new line and separate names from their values by `=`, `:`, or a space. For example, `key=value` or `key:value` or `key value`.

If you delete a property list that's connected to a WDT model file, the WDT model file will remain unchanged but all WDT variables are converted to standalone WDT model tokens.

Create a Property List

Create a property list to collect key-value pairs for use with WDT model files.

1. Open the Providers drawer and beside the project name, click **More** ⋮. Select **Create Provider for New Property List**.
2. Enter a name for the property list provider.
3. In the **Property List Filename** field, enter a name for the property list file. Include `.properties` or `.props` at the end of the file name.
4. Click **Pick Directory** and browse to the directory where you want to save the new property list file.
5. Click **OK** to create the file.

You can now use the property list to hold WDT variables for WDT model files.

Note:

You cannot rename keys. If you change the name of a key, WebLogic Remote Console registers that action as deleting the previous key and then creating a new key. If the old key was actively being used as a WDT variable in a WDT model file, it will become a standalone WDT model token, unassociated with the property list.

A
Troubleshoot Issues with WebLogic Remote Console

Learn how to identify and recover from issues in WebLogic Server or WebLogic Remote Console.

Perform these preliminary troubleshooting steps to get started:

- Restart all servers in the domain.
- Confirm that the Administration Server can reach your Managed Servers and clusters. See Investigate Server Connection Issues.
- Check the log files for any errors. See Review Log Files.
- Review any known issues. See Known Issues.
- Update to the latest versions of Desktop WebLogic Remote Console and the WebLogic Remote Console extension. See Upgrade Desktop WebLogic Remote Console.

Investigate Server Connection Issues

If the Administration Server cannot reach a Managed Server or cluster, you may be able to determine the cause using WebLogic Remote Console.

1. In the **Monitoring Tree**, go to **Environment**, then **Servers**. Use the **State** column to determine the current life cycle state of a server instance or cluster.

   If the Administration Server cannot reach the server to determine its state, the server will be marked as `Unreachable`.
2. Click on a server or cluster whose state is `Unreachable`.
3. Click the **Troubleshoot** tab. WebLogic Remote Console will automatically attempt to send test administration traffic to the server or cluster.

   Depending on the result of the test, WebLogic Remote Console provides different information:

   - If the Administration Server cannot find the HTTP/S address for sending administration traffic to the server (typically because the server is not running), then the **Server URL** field will return `unknown`.
   - If the Administration Server receives a response from the server, then WebLogic Remote Console will populate the **Status**, **Response Headers**, and **Response Body** fields.
   - If the Administration Server receives a Java exception, then WebLogic Remote Console will populate the **Exception** and **Stack Trace** fields.

Use the information gathered from this troubleshooting test to help determine the cause of the connection issues between the Administration Server and server instances or clusters.

Review Log Files

If you experience any issues with WebLogic Remote Console, then you can check its log files to determine the cause of the issues.

1. Open **Help**, then **Toggle Developer Tools** and select the **Console** tab.
2. Read the log messages to see they provide enough information to determine the cause of your issue.
3. WebLogic Remote Console also generates a log file, `out.log`, for each session. Open the log file and review its contents.

   The location of `out.log` varies depending on your platform:

   - Linux: `$HOME/.config/weblogic-remote-console/out.log`
   - macOS: `/Users/<user>/Library/Application Support/weblogic-remote-console/out.log`
   - Windows: `C:\Users\<user>\AppData\Roaming\weblogic-remote-console\out.log`

   Log file entries from previous sessions are saved to a new file in the same directory, marked by date: `out-yyyy-mm-dd.log`.

Note:

If you want to generate log files for WebLogic Server, see Log Messages instead.

Known Issues

This page describes the known issues associated with WebLogic Remote Console.

Limitations

You cannot:

- Record WLST scripts while you configure WebLogic Server using WebLogic Remote Console.
- Reorder security providers in WDT model file providers. However, you can download the WDT model file and edit the YAML file manually to rearrange the order of the security providers.

Limited MBean Property Support

WebLogic Remote Console includes most of the MBean properties that the WebLogic Server Administration Console supported. However, due to limitations in the WebLogic REST API, some MBeans are omitted, including some that are deprecated. If you require an MBean property that is not currently available in WebLogic Remote Console, then file an enhancement request in the WebLogic Remote Console GitHub repository.

My Credentials Don't Work

If you are sure your login credentials are correct but you're receiving an Unexpected Error Response error, it might be because your credentials contain unsupported characters.

By default, WebLogic Remote Console uses HTTP Basic Authentication which supports a limited character set. Characters outside that set, for example, Japanese characters, cannot be used in either usernames or passwords.

Workaround: If you wish to continue using credentials with these characters, then when you connect to an Administration Server from WebLogic Remote Console, you must enable **Use Web Authentication** in the provider connection dialog box. The WebLogic Remote Console will send you to the external authentication portal. See Configure Web Authentication.

Hosted WebLogic Remote Console is "unable to connect to the WebLogic domain's administration server"

In Hosted WebLogic Remote Console, an orange banner appears along the top of the console that indicates it cannot reach the Administration Server.

Solution: Try logging into Hosted WebLogic Remote Console again. In the browser's address bar, enter `http://`*`hostname`*`:`*`port`*`/rconsole/signin` (or `https://`*`hostname`*`:`*`port`*`/rconsole/signin`) and re-enter your credentials.

Deleted Items Do Not Re-Appear When Change is Reverted

If you delete an item but then discard the change instead of committing it, WebLogic Remote Console will not show the restored item immediately.

Workaround: Navigate away and then back to the affected page and the previously deleted item will become visible. If the deletion affected the navigation tree, such as deleting a node, you can collapse and then re-expand the navigation tree and the node will re-appear.

Using the Hosted WebLogic Remote Console Across Multiple Browser Tabs May Cause Strange Behavior

If you open Hosted WebLogic Remote Console in multiple browser tabs and edit the domain, the different instances will interfere with each other and may result in configuration errors.

Solution: Only use Hosted WebLogic Remote Console in one browser tab at a time.

Cannot connect to the Administration Server

WebLogic Remote Console cannot connect to a WebLogic Server Administration Server.

If you experience issues connecting to an Administration Server when using WebLogic Remote Console, then you may need to update the connection settings between the application and the domain. Use the following methods to identify which setting may be causing the issue.

- Test access to the Administration Server using `curl`. Make sure to use a WebLogic user that is assigned one of the following roles: Admin, Deployer, Operator or Monitor.
  - For HTTP connections, run:

    ```
    curl -v --user username:password http://adminServerHost:adminServerPort/
    management/weblogic/latest/domainConfig
    ```

  - For HTTPS connections, run:

    ```
    curl -v -k --user username:password https://
    adminServerHost:adminServerPort/management/weblogic/latest/domainConfig
    ```

> **Note**: Adding `-k` makes this connection insecure. It tells curl to skip the verification step that it normally performs on secure connections.

If you can connect successfully over HTTPS, then the problem is likely that WebLogic Remote Console does not trust the SSL certificate of the Administration Server. You can either import the Administration Server's certificate into your client's keystore, or, if you're using demo certificates, enable the **Make Insecure Connection** option when you connect to the Administration Server.

- Make sure your Administration Server's management endpoint, `management/*`, is accessible to clients. It may be blocked if your domain is behind a load-balancer or firewall, or is in a Docker container. You will need to expose the endpoint manually.
  You should also make sure that the value of the Remote Console Helper Context Path attribute (`RemoteConsoleHelperMBean.ContextPath`) has not been changed. The default value is `console`, which WebLogic Remote Console appends to the domain URL. If you modify the context path, it may prevent WebLogic Remote Console from successfully connecting to the Administration Server. Do not change it unless you understand the possible impacts to Desktop WebLogic Remote Console. See Configure Web Authentication.
- If the Administration Server resides in a different network than WebLogic Remote Console, then make sure the proxy settings of WebLogic Remote Console are properly configured to allow communication between the two. See Connect using a Proxy Server.
  You may need to add a location match stanza for the `management/*` endpoint to the domain configuration file, `config.xml`.

Features are Missing from WebLogic Remote Console

Certain features or screens in WebLogic Remote Console are hidden if your current user is not assigned the appropriate permissions.

For security purposes, WebLogic Remote Console restricts what a user can see or do depending on their role. Only administrators have access to the full functionality of WebLogic Remote Console.

Try logging in as a user with more elevated permissions.

For more information, see Access Limitations.

> **Note:**
>
> If you believe a feature is missing and it is not a case of inadequate permissions, you can open an enhancement request in the WebLogic Remote Console GitHub repository.

REST Communication Issues

WebLogic Remote Console reports that it is experiencing REST communication issues between servers.

When REST communication is blocked, it may prevent further configuration to the domain or cause the Monitoring Tree perspective to report inaccurate statuses for Managed Servers.

You can test the REST connection using curl.

- To get the Administration Server's runtime statistics *directly*:
  - For HTTP, run:

    ```
    curl -v --user username:password http://adminServerHost:adminServerPort/
    management/weblogic/latest/serverRuntime
    ```

  - For HTTPS, run:

    ```
    curl -v -k --user username:password https://
    adminServerHost:adminServerPort/management/weblogic/latest/serverRuntime
    ```

    Note: Adding `-k` makes this connection insecure. It tells curl to skip the verification step that it normally performs on secure connections.

  If this command fails, then a more general connectivity issue is affecting your domain. See Cannot connect to the Administration Server.
- To get the Administration Server's runtime statistics *indirectly*:
  - For HTTP, run:

    ```
    curl -v --user username:password http://adminServerHost:adminServerPort/
    management/weblogic/latest/domainRuntime/serverRuntimes/adminServerName
    ```

  - For HTTPS, run:

    ```
    curl -v -k --user username:password https://
    adminServerHost:adminServerPort/management/weblogic/latest/
    domainRuntime/serverRuntimes/adminServerName
    ```

    Note: Adding `-k` makes this connection insecure. It tells curl to skip the verification step that it normally performs on secure connections.

  If this command fails, then the problem is a REST communication failure between servers.

Potential Causes

If the direct connection to the Administration Server succeeded but the indirect connection failed, this may indicate an SSL/TLS handshake error or some other SSL/TLS issue. Enable JDK SSL/TLS debugging on the Administration Server and then re-run the indirect curl command to help you identify the SSL/TLS connection issue.

If you receive a `SocketException: Permission denied` error, then check if your VPN is interfering with communication to the Administration Server. Disconnect from the VPN and try to connect again. If the connection is successful, you may be able to work around the issue by setting the listen address of the Administration Server to `localhost`, `127.0.0.1`, or another NIC specific address and re-enabling the VPN.

Certain configuration changes can block REST communication between servers, including, but not limited to the following examples:

- Disabling the default Identity Asserter provider.
- Disabling the default Credential Mapping provider.
- Removing `weblogic-jwt-token` from the default Identity Asserter provider's Active Types.

- Applying incompatible REST invocation policies to the WebLogic Server REST API. For example, using Oracle Web Services Manager (OWSM) to protect the domain may inadvertently restrict access to the REST API.
- Changing the listen port (including enabling the Administration port) without immediately restarting the servers.

You must update your configuration changes so they no longer block REST communication.

Invalid WebLogic Server Configurations

WebLogic Remote Console reports an invalid configuration in your domain.

To make sure that changes to your domain are valid, WebLogic Remote Console performs validation checks whenever you save or try to commit a change. If WebLogic Remote Console reports an invalid configuration error, then you must identify and correct the change before you can commit your changes.

- Review any recent changes for issues. You can check the Shopping Cart to see your pending changes if you have the WebLogic Remote Console extension installed.
- Check the log output from WebLogic Remote Console and the Administration Server.

If you still cannot determine the cause of the error, then you may need to discard all of your changes and then reapply them one at a time to isolate the change responsible for the error.

> **Note:**
>
> WebLogic Remote Console *does not* validate WDT model files. It will accept changes or values that are invalid and which may prevent the WDT model file from building or updating a domain. For example, if you add integer values that are invalid or out of range for a specific setting, or remove a server or target but do not update the deployments to select a different server or target, WebLogic Remote Console will not flag these errors.
>
> For information on acceptable values, refer to the WDT Documentation.

Problem reading auto-prefs.json

WebLogic Remote Console shuts down unexpectedly with a `Failure reading auto prefs` error.

The `auto-prefs.json` file saves state information about WebLogic Remote Console including details on projects and providers. Users should not touch this file unless it becomes corrupted.

If `auto-prefs.json` does become corrupted, you can reset it, but all of your data regarding your projects will be lost. The data for your domain will be unaffected.

1. Close WebLogic Remote Console.
2. Delete `auto-prefs.json`. The location of `auto-prefs.json` varies depending on your platform:
   - Linux: `$HOME/.config/weblogic-remote-console/auto-prefs.json`
   - macOS: `/Users/user/Library/Application Support/weblogic-remote-console/auto-prefs.json`

- Windows: `C:\Users\`*`user`*`\AppData\Roaming\weblogic-remote-console\auto-prefs.json`

3. Restart WebLogic Remote Console.