Oracle® Fusion Middleware

What's New in Oracle WebLogic Server 12.2.1

12*c* (12.2.1)

E55209-08

April 2016

This document describes new features for all components of
Oracle WebLogic Server 12.2.1.

ORACLE®

Oracle Fusion Middleware What's New in Oracle WebLogic Server 12.2.1, 12*c* (12.2.1)

E55209-08

Contents

Preface

This preface describes the document accessibility features and conventions used in this guide—*What's New in Oracle WebLogic Server 12.2.1*.

Documentation Accessibility

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

Access to Oracle Support

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

Conventions

The following text conventions are used in this document:

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

1

Introduction and Roadmap

This chapter describes the contents and audience for this guide—*What's New in Oracle WebLogic Server 12.2.1*.

This chapter includes the following sections:

- Document Scope and Audience
- Guide to This Document

1.1 Document Scope and Audience

This document describes new features for all components of Oracle WebLogic Server 12.2.1. This document is a resource for system administrators and operators responsible for monitoring and managing a WebLogic Server installation. It is relevant to all phases of a software project, from development through test and production phases.

1.2 Guide to This Document

This document is organized as follows:

- This chapter, "Introduction and Roadmap", describes the scope and organization of this guide.
- Chapter 2, "What's New in Oracle WebLogic Server 12.2.1", describes the new and changed functionality in this WebLogic Server release.

2

What's New in Oracle WebLogic Server 12.2.1

This chapter describes the following new and changed functionality in Oracle WebLogic Server 12.2.1:

- Multitenancy Support

- Continuous Availability

- Java EE 7 Support

- JDK 8 Certification

- Docker Certification

- Runtime Improvements

- Manageability Improvements

- Documentation Update History for Release 12.2.1

- Standards Support, Supported Configurations, and WebLogic Server Compatibility

- Deprecated Functionality (WebLogic Server 12.2.1)

- Removed Functionality and Components

Revision History of this Document

Date	Description of Change
November 2015	■ Added a section summarizing updates made to the Oracle WebLogic Server 12.2.1 documentation library. See Section 2.8, "Documentation Update History for Release 12.2.1".
	■ Added documentation for additional JDBC data source runtime features provided in Oracle WebLogic Server 12.2.1. For more information, see Section 2.6.2, "JDBC Data Sources".
December 2015	■ Updated JDK support information in Section 2.4, "JDK 8 Certification".
	■ Updated Section 2.9.1, "Standards Support" to clarify the specific versions of Java EE JSF that are supported in Oracle WebLogic Server 12.2.1.
January 2016	■ Updated Section 2.9.2, "Supported Configurations" to clarify restrictions and advice regarding the use of Java SE 8 with Oracle WebLogic Server 12.2.1.
	■ Added the deprecation notice in Section 2.10.14, "SSLMBean.ExportKeyLifespan Attribute".

Date	Description of Change
February 2016	■ Added descriptions of new log monitoring feature that prevents excessive messaging, and new log format compatibility switch. See Section 2.7.8, "Logging Enhancements".
	■ Corrected JMX support information in Section 2.9.1.1, "Java Standards".
March 2016	Updated Section 2.8, "Documentation Update History for Release 12.2.1" to summarize updates made this month to *Configuring Elasticity in Dynamic Clusters for Oracle WebLogic Server* and *Configuring and Using the Diagnostics Framework for Oracle WebLogic Server*.
April 2016	Added Compatibility MBean Server and Type-Safe MBean Interfaces to Section 2.11, "Removed Functionality and Components".

2.1 Multitenancy Support

Multitenancy in WebLogic Server provides a sharable infrastructure for use by multiple organizations. These organizations are a conceptual grouping of your own choosing, which you can think of as **tenants**. By allowing one domain to support multiple tenants, Weblogic Server Multitenant improves density and achieves a more efficient use of resources while eliminating the hurdles typically present when trying to share multiple applications: runtime cross-application impact, security differences, data co-mingling, and administrative challenges.

WebLogic Server Multitenant provides resource isolation within **domain partitions**, an administrative and runtime slice of a WebLogic domain that is dedicated to running application instances and related resources for a tenant. Domain partitions achieve greater density by allowing application instances and related resources to share the domain, WebLogic Server itself, the Java virtual machine, and the operating system while isolating tenant-specific application data, configuration, and runtime traffic.

WebLogic Server MT enables an end to end multitenant infrastructure, including multitenancy from the load balancer to the middle tier and cache tier, and to the database tier. WebLogic Server MT extends the Oracle WebLogic Server Enterprise Edition and Oracle WebLogic Suite products, and includes the following components:

- Oracle WebLogic Server MT, which enables consolidation of applications into fewer domains (by allowing partitions within domains) while maintaining secure isolation

- WebLogic MT extensions to Java SE Advanced, which enables memory, CPU and I/O isolation, monitoring, and management for applications within a JVM

- Oracle WebLogic Coherence Enterprise Edition to Grid Edition option, which enables consolidation of caches into fewer Oracle Coherence clusters while maintaining secure isolation

- Oracle Traffic Director which provides WebLogic Server MT-aware and fully integrated tenant-aware local load balancing

See *Using WebLogic Server Multitenant* for additional information.

2.1.1 WLST Changes to Support Multitenancy

To support multitenancy, new WLST commands have been added to WebLogic Server 12.2.1 and existing commands have been modified to support new arguments for multitenancy environments.

The following WLST commands have been added for multitenancy:

- `importPartition`—Imports a domain partition from a partition archive.

- `exportPartition`—Exports a domain partition to a partition archive.

- `startPartitionWait`—Starts a domain partition and waits until the partition has started.

- `migrateResourceGroup`—Migrates a domain partition resource group from one target to another target.

The following arguments have been added to existing WLST commands to support multitenancy:

- The `resourceGroup`, `resourceGroupTemplate`, and `partition` arguments have been added to the `deploy` command. Use these to specify the resource group, resource group template, and partition to which a deployment is scoped.

- The `resourceGroupTemplate`, `partition`, and `removePlanOverride` arguments have been added to the `redeploy` command. Use these to specify the resource group template and partition to which the redeployment is scoped, and to specify whether or not the previous plan override at the resource group level should be removed.

- The `resourceGroupTemplate` and `partition` arguments have been added to the `undeploy` command. Use these to specify the resource group template and partition to which the deployment is scoped.

- The `resourceGroupTemplate`, `partition`, and `removePlanOverride` arguments have been added to the `updateApplication` command. Use these to specify the resource group template and partition to which the new deployment is scoped, and to specify whether or not the previous plan override at the resource group level should be removed.

- The `resourceGroup`, `resourceGroupTemplate`, and `partition` arguments have been added to the `distributeApplication` command. Use these to specify the resource group, resource group template, and partition to which the copied deployment bundle is scoped.

- The `partition` argument has been added to the `startApplication` and `stopApplication` commands. Use this argument to specify the partition to which the deployment is scoped.

- The `partition` argument has been added to the `exportDiagnosticDataFromServer` command. Use this argument to specify the partition from which diagnostics data will be retrieved.

- The `partition` argument has been added to the `saveDiagnosticImageCaptureFile` and `saveDiagnosticImageCaptureEntry File` commands. Use this argument to specify the partition from which the image or image entry will be retrieved.

- The `partition` argument has been added to the `captureAndSaveDiagnosticImage` command. Use this argument to specify the partition from which the image will be retrieved.

The `partition` argument is also supported for the following new WLST diagnostic commands. Use this argument to restrain the scope of the diagnostic command to a particular partition within a domain:

- `activateDebugPatch`

- `deactivateDebugPatches`

- `exportHarvestedTimeSeriesData`

- `exportHarvestedTimeSeriesDataOffline`

- `getAvailableDiagnosticDataAccessornames`

- `purgeCapturedImages`

For information about these and other new WLST commands, see Section 2.7.12, "WLST".

2.1.2 Deployment Support for Multitenancy

Using WebLogic Server Multitenant, you can deploy applications and libraries to resource groups templates and resource groups at the domain and partition levels. You can perform deployment operations to these scopes using the new attributes added to existing deployment clients. The following deployment clients support multitenant application deployment:

- weblogic.Deployer

- WLST deployment commands

- JSR-88 API for deployment

- JMX Deployment API

- WLDeploy ant task

- Maven goals for deployment

- WebLogic Server Administration Console

- Fusion Middleware Control

For more information, see "Deploying Applications" in *Using WebLogic Server Multitenant* and "Deploying Applications to Resource Groups and Templates" in *Deploying Applications to Oracle WebLogic Server*.

Additional multitenant deployment features include:

- Partition-Specific Deployment Plans

- Concurrent Deployment

2.1.2.1 Partition-Specific Deployment Plans

You can use partition-specific deployment plans to customize any application deployed to a resource group template or resource group within a partition. Partition-specific deployment plans are specified at the resource group level within a partition, not at the partition level. When WebLogic Server applies a partition-specific deployment plan to a specified application, the changes prescribed by the plan affect only the application deployment within that partition.

To configure partition-specific application deployment plans, use the `redeploy` or `updateApplication` WLST commands. The new WLST options `resourceGroupTemplate` and `partition` identify the scope of the application deployment, and the existing WLST options `planPath` and `appName` identify the location of the partition-specific deployment plan and the name of the application deployment to modify.

For more information, see "Using Partition-Specific Deployment Plans" in *Using WebLogic Server Multitenant*.

2.1.2.2 Concurrent Deployment

WebLogic Server 12.2.1 supports concurrent deployment operations across multiple edit sessions. Concurrent deployment allows different partition administrators to deploy applications at the same time, without one administrator having to wait for another administrator's application to complete the deployment process. Concurrent deployment improves performance and prevents administrators from being blocked by other administrators. Concurrent deployment occurs automatically; there are no attributes to configure.

2.1.3 JNDI Support for Multitenancy

To support multitenancy, JNDI will be partition-aware and provide partition isolation internally. Partition-aware JNDI includes the following features:

- **Partition-scoped and domain-scoped global JNDI resources**—There will be one global JNDI tree for every partition and domain on a node. For example, when deploying an EJB to multiple partitions and domains, the EJB's JNDI name is bound to the global JNDI tree for each partition and domain. When looking up the JNDI name, the JNDI tree that is used depends on the scope of the lookup (partition or domain).

- **Object-based partition association**—When a JNDI context is created within a domain partition, the context object sticks to the partition namespace so that all subsequent JNDI operations are done within the context of the partition.

- **Cross-partition access**—Partition JNDI resources can be accessed from remote standalone Java code using the WebLogic Server client, or by code that resides in a remote WebLogic Server instance. They can also be accessed from another partition on the same server.

- **Clustered JNDI**—The JNDI tree that represents a cluster appears to the client as a single global tree. The tree containing cluster-wide services is replicated across each WebLogic Server instance or partition in the cluster.

- **Foreign JNDI**—You can use foreign JNDI to access another partition whether it is local or remote. By setting up a foreign JNDI provider with the properties of the other partitions, you can look up and use an object that exists outside of the partition.

- **New provider URL patterns**—For local access across partitions, the following new provider URL patterns have been introduced:

  - `local://?partitionName=DOMAIN` creates the context on the domain.

  - `local://?partitionName=partitionName` creates the context on the specified partition.

- **Partition JNDI resource lifecycle**—Partition JNDI resources are maintained based on the partition lifecycle. When a partition is created and started, the partition JNDI tree is created with the partition root node and becomes available. When the partition is shut down, the partition JNDI tree is deleted.

- **Partition information binding**—When initializing the partition tree, partition information is bound to the partition global JNDI tree. Therefore, you can obtain the partition information of the context by looking up:

  - `weblogic.partitionName`, which returns the context-based partition's partition name or "Domain" if it is a domain.

  - `weblogic.partitionID`, which returns the current partition's partition ID or "0" if it is a domain.

For more information, see "Configuring and Programming JNDI" in *Using WebLogic Server Multitenant*.

2.1.4 Partition Security

To support multitenancy and partitions, the Weblogic Security Service includes the following new features in release 12.2.1:

- Multiple active realms

  To provide isolation of the security configuration and metadata, WebLogic Server now supports the ability to have multiple active security realms in a domain. Support for multiple active realms enables per-partition configuration for realm-based services by allowing each partition to execute against a different realm. Partitions may also share a security realm, which may be appropriate when the security configuration between two partitions does not require isolation.

- Identity domains

  An identity domain is a logical name space for users and groups, typically representing a discrete set of users and groups in a physical store. Identity domains are used to identify the users associated with particular partitions.

- Partition-aware security services

  Partition-aware security services contain context about the partition in which they execute so that, for example, they can control access to a resource based on the partition to which the resource belongs. Partition-aware services are also identity domain aware.

  To support partition-aware security services, WebLogic Server adds several new predicates that can be used in security policies to scope access at the partition level.

For more information, see "Configuring Security" in *Using WebLogic Server Multitenant*.

2.1.5 Partition Work Managers

Partition-specific Work Managers provide fairness of thread usage and prioritization of work requests among partitions that share the same WebLogic Server instance.

Partition Work Managers:

- Allow system administrators to set thread usage policies among partitions and configure customized QoS levels.

- Allow partition administrators to configure Work Manager policies for their partitions.

- Support the prioritization of thread usage by work requests from various partitions.

- Ensure the fairness in thread resource usage among partitions.

For more information, see "Configuring Partition Work Managers" in *Using WebLogic Server Multitenant*.

2.1.6 Partition Scope Monitoring and Debugging

The WebLogic Diagnostics Framework (WLDF) provides the following partition scope diagnostic capabilities:

- Partition scope logging

- Partition scope debugging

- Partition scope log file and diagnostic data access

- Monitoring of partition resources

- Partition scope application instrumentation

- Partition scope diagnostic image capture

For information, see "Monitoring and Debugging Partitions" in *Using WebLogic Server Multitenant*.

2.2 Continuous Availability

Oracle WebLogic Server Continuous Availability provides an integrated solution for building maximum availability architectures (MAA) that span data centers across distributed geographical locations. Integrated components include Oracle WebLogic Server, Oracle Coherence, Oracle Traffic Director, Oracle SiteGuard, and Oracle Database. The major benefits of this integrated solution are faster failover or switchover, increased overall application availability, data integrity, reduced human error and risk, recovery of work, and local access of real-time data.

The key features in Continuous Availability include:

- **Automated cross-domain transaction recovery**— Provides automatic recovery of XA transactions across an entire domain, or across an entire site with servers running in a different domain or at a different site.

- **Zero Downtime Patching**—Provides an automated mechanism to orchestrate the rollout of patches while avoiding downtime or loss of sessions.

- **WebLogic Server Multitenant live resource group migration**— Provides the ability to migrate partition resource groups that are running from one cluster/server to another within a domain without impacting the application users.

- **Coherence federated caching**—Replicates cache data asynchronously across multiple geographically distributed clusters.

- **Coherence GoldenGate HotCache**—Detects and reflects database changes in cache in real time.

- **Oracle Traffic Director**—Routes HTTP, HTTPS, and TCP traffic to application servers and web servers on the network.

- **Oracle Site Guard**—Enables administrators to automate complete site switchover or failover.

For more information about the features in Continuous Availability and the supported MAA architectures, see *Continuous Availability for Oracle WebLogic Server*.

2.3 Java EE 7 Support

Oracle WebLogic Server 12.2.1 is a fully compatible implementation of the Java Platform, Enterprise Edition (Java EE) Version 7.0. Java EE 7 enables developers to make use of the latest innovations in the Java Enterprise APIs, which include new programming models, as well as consolidating, enhancing, and in some cases simplifying existing specifications.

The Java EE 7 APIs and related capabilities simplify development of server applications accessed by "rich" clients using lightweight web-based protocols such as

REST, WebSocket, and Server-Sent Events. Improvements to development tooling and open source support expand developer choices and simplify creation of development environments.

New Java EE 7 support updates provided in WebLogic Server 12.2.1 are described in the following sections:

- Batch Application Processing (JSR 352)
- Concurrent Managed Objects (JSR 236)
- Default Data Source
- JMS 2.0 Support for Simplified JMS Application Development (JSR 343)
- Java EE Connector Architecture 1.7 (JSR 322)
- Enterprise JavaBeans 3.2 (JSR-345)
- Clustering and High Availability Support for WebSocket 1.1 Applications
- GZIP Compression Support
- Java EE 7 Security Standards
- Sample Applications

2.3.1 Batch Application Processing (JSR 352)

WebLogic Server 12.2.1 adds support of a Batch Processing Runtime for the Java EE 7 Platform (JSR 352), which provides support for defining, implementing, and running batch jobs.

The batch runtime in WebLogic Server uses a data source, also known as the JobRepository, and a managed executor service to execute asynchronous batch jobs. The executor service processes the jobs and the JobRepository data source stores the status of current and past jobs. The default batch runtime in each WebLogic domain can be used without any configuration using the Derby demo database. For environments that use an enterprise-level database schema, you can configure a dedicated JobRepository data source and executor service for each WebLogic domain.

For more information about configuring and managing the batch runtime, see "Using the Batch Runtime" in *Administering Server Environments for Oracle WebLogic Server*.

2.3.2 Concurrent Managed Objects (JSR 236)

WebLogic Server 12.2.1 adds support of the Concurrency Utilities for Java EE 1.0 (JSR 236), which is a standard API for providing asynchronous capabilities to Java EE application components, such as servlets and EJBs.

WebLogic Server provides concurrency capabilities to Java EE applications by associating the Concurrency Utilities API with the Work Manager to make threads container-managed. You configure concurrent managed objects (CMOs) and then make them available for use by application components. Similar to Work Managers, CMOs can be defined at the domain level, application level, and module level, by using the Administration Console, MBeans, or deployment descriptors:

For more information about configuring and managing concurrent resources, see "Configuring Concurrent Resources" in *Administering Server Environments for Oracle WebLogic Server*.

2.3.3 Default Data Source

Oracle provides a Default Data Source required by a Java EE 7-compliant runtime. This pre-configured data source can be used by an application to access the Derby Database that can be installed with WebLogic Server. See "Using the Default Data Source" in *Administering JDBC Data Sources for Oracle WebLogic Server*.

2.3.4 JMS 2.0 Support for Simplified JMS Application Development (JSR 343)

Simplified API defined by the Java Message Service (JMS) 2.0 specification. For more information, see "Understanding the Simplified API Programming Model" in *Developing JMS Applications for Oracle WebLogic Server*.

2.3.5 Java EE Connector Architecture 1.7 (JSR 322)

Oracle WebLogic Server 12.2.1 supports the Java EE Connector Architecture (JCA) 1.7 specification. For more information, see "Understanding Resource Adapters" in *Developing Resource Adapters for Oracle WebLogic Server*.

2.3.6 Enterprise JavaBeans 3.2 (JSR-345)

WebLogic Server 12.2.1 supports the Enterprise Java Beans (EJBs) 3.2 specification (JSR 352). For more information, see "Understanding Enterprise JavaBeans" in *Developing Enterprise JavaBeans for Oracle WebLogic Server*.

2.3.7 Clustering and High Availability Support for WebSocket 1.1 Applications

WebLogic Server 12.2.1 adds support for clustering and high availability with WebSocket applications. WebSocket clustering uses Coherence as part of its implementation to establish communication among all cluster members. WebSocket clustering enables horizontal scaling, allows you to send messages to all members of the cluster, increases the maximum number of connected clients, and decreases broadcast execution time.

2.3.8 GZIP Compression Support

WebLogic Server 12.2.1 adds support for GZIP compression in the WebLogic Web container, which you can enable at the domain or Web application level. With GZIP compression enabled, you can configure attributes, such as minimum content length and compression content types, and monitor related statistics. For more information, see "Enabling GZIP Compression for Web Applications" in *Developing Web Applications, Servlets, and JSPs for Oracle WebLogic Server*.

2.3.9 Java EE 7 Security Standards

WebLogic Server 12.2.1 includes support for the following security standards:

- Java Authorization Contract for Containers 1.5
- Java Authentication Service Provider Interface for Containers 1.1 (JASPIC)
- Packaged Permissions
- Uncovered HTTP Methods (JSR 340 for Servlet 3.1

For more information, see Security.

2.3.10 Sample Applications

The sample applications that can optionally be installed with WebLogic Server have been updated for Java EE 7, as described in the following sections:

- Avitek Medical Records — MedRec
- New Java EE 7 Examples

2.3.10.1 Avitek Medical Records — MedRec

Avitek Medical Records (or "MedRec") is a comprehensive educational sample application that demonstrates WebLogic Server and Java EE features, as well as best practices. In WebLogic Server 12.2.1, MedRec has been upgraded to demonstrate the following Java EE 7 features:

- Java Persistence 2.1
- Simplified API of JMS 2.0
- Contexts and Dependency Injection 1.1
- Batch 1.0
- JAX-RS 2.0
- JavaServer Faces 2.2
- JSON Processing 1.0
- HTML5

2.3.10.2 New Java EE 7 Examples

New sample applications have been added to show the following Java EE 7 features:

- JSON Processing 1.0
- Servlet 3.1
- JavaServer Faces 2.2
- Expression Language 3.0
- Batch Processing
- Concurrency Utilities
- Contexts and Dependency Injection 1.1
- Java EE Connector Architecture 1.7
- Java Persistence 2.1
- Java Message Service API 2.0
- Enterprise JavaBeans 3.2
- Java API for RESTful Web Services (JAX-RS) 2.0 asynchronous processing, filters and interceptors, and server-sent events (SSE) Jersey support.

2.3.11 Enterprise JavaBeans 3.2 (JSR-345)

WebLogic Server 12.2.1 supports the Enterprise Java Beans (EJBs) 3.2 specification (JSR 352). For more information, see "Understanding Enterprise JavaBeans" in *Developing Enterprise JavaBeans for Oracle WebLogic Server*.

2.4 JDK 8 Certification

Oracle WebLogic Server 12.2.1 is certified for use with JDK 8.0. Supported Oracle WebLogic Server 12.2.1 clients are certified for use with JDK 7.0 and JDK 8.0 Update 40. A certified JDK is required for running the WebLogic Server installation program. For more information, see the following:

- The Oracle Fusion Middleware Supported System Configurations page on Oracle Technology Network.

- "Roadmap for Verifying Your System Environment" in *Installing and Configuring Oracle WebLogic Server and Coherence*

2.5 Docker Certification

Oracle WebLogic Server 12.2.1 is certified to run inside a Docker container. Docker is a Linux-based container technology that enables you to quickly create lightweight clustered and non-clustered WebLogic Server domain configurations on a single host OS or virtual machines, for either development or production environments.

As part of this certification, Oracle provides Docker files and supporting scripts for building images of Oracle WebLogic Server. These images are built as an extension of existing Oracle Linux images. These scripts and build images are available on GitHub at the following location:

https://github.com/oracle/docker/tree/master/OracleWebLogic

For more information, see "Getting Started" in *Running Oracle WebLogic Server 12.2.1 on Docker*.

2.6 Runtime Improvements

Oracle WebLogic Server 12.2.1 builds on support from prior WebLogic Server versions to improve the reliability, availability, scalability and performance of WebLogic Server applications in clustered environments, when using new Oracle database features, in multi data center architectures, and on Exalogic Systems.

These support improvements are described in the following topics:

- Deployment
- JDBC Data Sources
- JTA
- Messaging
- Monitoring WebSocket Applications
- Policy Classloader
- ReadyApp Integration with WebLogic Server
- RESTful Web Services
- Simple WLST APIs for Dynamic Clusters
- ThreadLocal Clean Out Support for Work Managers

2.6.1 Deployment

WebLogic Server 12.2.1 includes the following new and changed deployment features:

- [Overriding Resource Group Template Application Configuration](#)

- [Parallel Deployment](#)

- [FastSwap Enhancements](#)

- [Deployment Performance Enhancements](#)

- [Behavior Change for Application Names](#)

2.6.1.1 Overriding Resource Group Template Application Configuration

When a resource group references a resource group template, it inherits the application configuration defined in the resource group template. You can customize a specific application in a resource group by overriding the default application configuration in the resource group template.

To override the application configuration defined in a resource group template, specify a different deployment plan that the application should use for its configuration. You can apply application overrides or remove existing overrides using the Administration Console, Fusion Middleware Control, or by using the update or redeploy command with one of the supported deployment clients.

For more information, see "Overriding Application Configuration" in *Deploying Applications to Oracle WebLogic Server* and "Overriding Application Configuration" in *Using WebLogic Server Multitenant*.

2.6.1.2 Parallel Deployment

WebLogic Server 12.2.1 adds support for parallel deployment. For use cases involving the deployment of multiple applications, the deployment of a single application with multiple modules, or the deployment of one or more applications across multiple partitions, parallel deployment improves startup and post-running deployment time. In multitenant environments, parallel deployment helps avoid cross-tenant performance impact.

For more information about parallel deployment, see "Enabling Parallel Deployment for Applications and Modules" in *Deploying Applications to Oracle WebLogic Server* and "Enabling Parallel Deployment in Multitenant Environments" in *Using WebLogic Server Multitenant*.

2.6.1.3 FastSwap Enhancements

WebLogic Server 12.2.1 enhances FastSwap to work with the Java EE Contexts and Dependency Injection (CDI) specification.

Used in development mode, FastSwap helps developers avoid redeployment of an application and reduce turnaround time during development iterations. To improve the developer experience, you can now use FastSwap with CDI. For more information about FastSwap, see "Using FastSwap Deployment to Minimize Redeployment" in *Deploying Applications to Oracle WebLogic Server*.

2.6.1.4 Deployment Performance Enhancements

Oracle WebLogic Server 12.2.1 adds several deployment performance enhancements. Improvements include:

- Application class loading in parallel.

- Indexing of class finder data to locate classes and resources faster.

- Deployment factory caching during identification of a deployment, helping large deployments process faster.

- Annotation scanning caching for libraries and applications, benefiting server restart and resulting in faster deployment time.

- Annotation scanning in parallel so that each JAR file in the class path of a module is handled in parallel.

2.6.1.5 Behavior Change for Application Names

As of Oracle WebLogic Server 12.2.1, application names must be unique within each deployment scope. When deploying an application globally to a domain, if that application name is already in use in the current domain, the application deployment fails. This is a behavior change from previous WebLogic Server versions, when specifying the same application name caused WebLogic Server to automatically derive a unique name based on the specified name.

2.6.2 JDBC Data Sources

WebLogic Server 12.2.1 includes the following new and changed features:

2.6.2.1 Simplified Driver Installation/Update

In previous releases, adding a new JDBC driver or updating a JDBC driver where the replacement JAR has a different name than the original JAR required updating the WebLogic Server's classpath to include the location of the JDBC driver classes. In 12.2.1, you can simply put the drivers in the `$DOMAIN_HOME/lib` directory and not have to change the classpath. For more information, see "Adding Third-Party JDBC Drivers Not Installed with WebLogic Server" in *Administering JDBC Data Sources for Oracle WebLogic Server*.

2.6.2.2 Proxy Data Source Support

Oracle provides a new data source type which provides the ability to switch between databases in a WebLogic Server Multitenant environment. For more information, see "Using Proxy Data Sources" in *Administering JDBC Data Sources for Oracle WebLogic Server*.

2.6.2.3 JDBC Data Source Support for Multitenancy

Data Source configuration and monitoring support for WebLogic Server Multitenant. For more information, see "Configuring JDBC" in *Using WebLogic Server Multitenant*.

2.6.2.4 Universal Connection Pool Data Sources

Universal Connection Pool (UCP) data sources are now available as an option for users who wish to use UCP to connect to Oracle Databases. UCP provides an alternative connection pooling technology to WebLogic Server connection pooling. For more information, see "Using Universal Connection Pool Data Sources" in *Administering JDBC Data Sources for Oracle WebLogic Server*.

2.6.2.5 Connection Leak Profiling Enhancements

Connection leak profiling enhancements include:

- A new attribute, `Set Connection Leak Timeout Seconds`, that can be used instead of `Inactive Connection Timeout Seconds` to specify the length of time before a reserved connection is considered leaked.

- Two new profile records:

- JDBC Object Closed Usage—Collect profile information about application components that close a connection, statement, or result set.

- Local Transaction Connection Leak—Collect profile information about application components that leak a local transaction (start it but don't commit or rollback the transaction).

For more information, see "Collecting Profile Information" in *Administering JDBC Data Sources for Oracle WebLogic Server*.

2.6.2.6 Enhanced Connection-Based System Properties

You can set connection-based system properties using variables based on your environment. For more information, see "Enabling Connection-Based System Properties" in *Administering JDBC Data Sources for Oracle WebLogic Server*.

2.6.2.7 Application Continuity Runtime Statistics

Application Continuity (or Replay) statistics are available using the `JDBCReplayStatisticsRuntimeMBean` for Generic and Active GridLink data sources. For more information, see "Viewing Runtime Statistics for Application Continuity" in *Administering JDBC Data Sources for Oracle WebLogic Server*.

2.6.2.8 ONS Node List Configuration Enhancements

You can now use a property node list to configure the ONS node list. The property node list, which can be used instead of a single node list, is a string composed of multiple records, with each record consisting of a key=value pair. For more information, see "ONS Client Configuration" in *Administering JDBC Data Sources for Oracle WebLogic Server*.

2.6.2.9 DRCP Network Timeout Property

A system property `weblogic.jdbc.attachNetworkTimeout` is provided that, after an attach to the server, specifies a network timeout that forces a round-trip to the database (using an Oracle ping database operation). The timeout is then unset. For more information, see "Database Resident Connection Pooling" in *Administering JDBC Data Sources for Oracle WebLogic Server*.

2.6.2.10 Enhanced Edition-Based Redefinition (EBR) Documentation

The documentation that describes this feature has been enhanced to include details about using EBR with JDBC connections, and describes how to configure WebLogic data sources to use Editions. For more information, see "Using Edition-Based Redefinition" in *Administering JDBC Data Sources for Oracle WebLogic Server*.

2.6.2.11 Guidelines for Planned Maintenance and Database Outages

Procedures and guidelines have been included for planning and managing database maintenance and downtimes for Multi Data Source and Active Gridlink data sources. For more information, see the following topics in *Administering JDBC Data Sources for Oracle WebLogic Server*:

- Planned Database Maintenance with a Multi Data Source

- Managing Database Downtime with AGL Data Sources

2.6.2.12 Enabling ONS and JDBC Debugging Changes

To enable ONS and JDBC replay debugging, you must configure Java Util Logging. For more information, see the following topics in *Administering JDBC Data Sources for Oracle WebLogic Server*:

- Enabling JDBC Driver Debugging
- Debugging ONS

2.6.2.13 Support for Encrypted Passwords in a Data Source Definition

In previous releases, the `PasswordEncrypted` attribute was not supported in data source definitions. This restriction has been removed. For more information, see "Using an Encrypted Password in a DataSourceDefinition" in *Developing JDBC Applications for Oracle WebLogic Server*.

2.6.3 JTA

WebLogic Server 12.2.1 includes the following new and changed JTA features:

- Transactions without TLog Write

2.6.3.1 Transactions without TLog Write

Oracle improves XA transaction performance by providing the option to eliminate the writing of XA transactions to the TLog. XA transaction resources (Determiners) are used during transaction recovery when a TLog is not present. See "XA Transactions without Transaction TLog Write" in *Developing JTA Applications for Oracle WebLogic Server*.

2.6.4 Messaging

In WebLogic Server 12.2.1, JMS includes the following:

2.6.4.1 Messaging Support for Multitenancy

Support for the WebLogic Messaging Service in multitenant environments, including:

- Core WebLogic Messaging components including Modules, JMS Resources, Path Service, Stores, and external helpers such as JMS Destination Availability Helper.
- Integration solutions, including the Messaging Bridge, JMS pools, and Foreign JMS Servers.
- Store-and-Forward (SAF) agents, including JMS SAF and WSRM JAX RPC.
- AQ JMS using Foreign JMS Servers.

For more information, see "Configuring Messaging" in *Using WebLogic Server Multitenant*.

2.6.4.2 Simplified JMS Cluster Configuration and High Availability Enhancements

WebLogic Server 12.2.1 provides new configuration options that simplify messaging configurations, improve high availability, and remove limitations from previous releases. Support includes:

JMS cluster targeted JMS servers and persistent stores have new cluster targeting features. They eliminate the need to individually configure many JMS resource artifacts for every server in a cluster by targeting of the resource artifacts directly to the

cluster.Additional features are available when Migratable Services, Distribution-Policy, and Migration-Policy are all configured.

- Automatic Service Migration: Automatically restart a failed JMS instance on a different WebLogic Server instance.

- Restart-In-Place: Automatically restart a failed JMS instance on its running WebLogic Server instance.

- Unit-of-Order and Unit-of-Work

- Singleton Destinations

- Cluster Targeted SAF Agents, Bridges, and Path Services

- Fail-back: Return an instance to its original host server when the host server restarts.

2.6.4.3 Improved JDBC Store Retry Handling

In previous releases, the JDBC Store's retry mechanism would make one reconnect attempt and throw a `JDBCStoreException` if it was not able to reconnect to the database. In WebLogic Server 12.2.1, Oracle allows you to configure multiple retry attempts over a specified time period. `reconnectRetryPeriodMillis` specifies the duration of the reconnect retry period in milliseconds. `reconnectRetryIntervalMillis` to specifies the interval between retry connection attempts during `reconnectRetryPeriodMillis`.

2.6.5 Monitoring WebSocket Applications

In WebLogic Server 12.2.1, you can monitor message statistics and runtime properties for WebSocket applications and endpoints. Endpoint-level monitoring collects information per individual endpoint, while application-level monitoring aggregates information from all endpoints deploying in the given application.

The following MBeans have been added or modified to support WebSocket monitoring:

- `WebAppComponentRuntimeMBean`

- `WebsocketApplicationRuntimeMBean`

- `WebsocketEndpointRuntimeMBean`

- `WebsocketMessageStatisticsRuntimeMBean`

- `WebsocketBaseRuntimeMBean`

You can also use the Administration Console or Fusion Middleware Control to monitor WebSocket applications.

For more information, see "Monitoring WebSocket Applications" in *Developing Applications for Oracle WebLogic Server*.

2.6.6 Policy Classloader

WebLogic Server 12.2.1 integrates the policy classloader implementation as the default system class loader when using WebLogic start scripts. The policy classloader improves class loader performance and server startup time and is supported in all WebLogic modes (development and production).

2.6.7 ReadyApp Integration with WebLogic Server

WebLogic Server 12.2.1 integrates the ReadyApp framework. At times, applications are not fully initialized when WebLogic Server completed its startup process. By using the ReadyApp framework, applications can register with the WebLogic Server ReadyApp during the deployment process and influence the true 'ready' state of the server instance. Applications notify ReadyApp of their application state so server instances can determine if an application is fully initialized and ready to accept requests. ReadyApp also allows load balancers to detect server readiness by providing a reliable health-check URL.

2.6.8 RESTful Web Services

Oracle WebLogic Server 12.2.1 provides the following new and changed features for RESTful web services:

- Provides support for Jersey 2.x (JAX-RS 2.0 RI) by default in this release. Registration as a shared library with WebLogic Server is no longer required.

- Provides enhanced monitoring of RESTful web services in the WebLogic Administration Console, including enhanced runtime statistics for your RESTful applications and resources, detailed deployment and configuration data, global execution statistics, and resource and resource method execution statistics. The following runtime MBeans have been modified or added to support enhanced monitoring:

    - `JaxRsApplicationRuntimeBean`

    - `JaxRSExceptionMapperStatisticsRuntimeMbean`

    - `JaxRsExecutionStatisticsRuntimeMBean`

    - `JaxRsResourceMethodBaseRuntimeMBean`

    - `JaxRsResourceMethodRuntimeMBean`

    - `JaxRsResourceRuntimeMBean`

    - `JaxRsResponseStatisticsRuntimeMBean`

    - `JaxRsSubResourceLocatorRuntimeMBean`

    - `JaxRsUriRuntimeMBean`

- Includes the ability to disable RESTful web services monitoring at the individual application level, or globally at the domain level.

- Reflects support for the Jersey 2.21.1 (JAX-RS 2.0 RI).

- Supports securing Jersey 2.x (JAX-RS 2.0 RI) web services using Oracle Web Services Manager (OWSM) security policies.

- Adds support for Java EE 7.

For more information, see *Developing and Securing RESTful Web Services for Oracle WebLogic Server*.

2.6.9 Simple WLST APIs for Dynamic Clusters

WebLogic Server 12.2.1 adds WLST commands to improve usability for dynamic cluster lifecycle operations. By using the WLST `scaleUp` and `scaleDown` commands, you can easily start and stop dynamic servers in a dynamic cluster and expand or shrink the size of a dynamic cluster.

For more information, see "Starting and Stopping Servers in Dynamic Clusters" and "Expanding or Reducing Dynamic Clusters" in *Administering Clusters for Oracle WebLogic Server*.

2.6.10 ThreadLocal Clean Out Support for Work Managers

Oracle WebLogic Server 12.2.1 enhances TheadLocal clean out support in Work Managers. To clean up stray ThreadLocal use by applications and third-party libraries, configure the `eagerThreadLocalCleanup` attribute in the `KernelMBean`. By default, the self-tuning thread pool only cleans up ThreadLocal storage when a thread returns to a standby pool and after an application is undeployed. For more information, see "ThreadLocal Clean Out" in *Administering Server Environments for Oracle WebLogic Server*.

2.7 Manageability Improvements

Oracle WebLogic Server 12.2.1 continues to provide new management features which simplify the configuration, monitoring and ongoing management of WebLogic Server domains and applications. These new features are described in the following sections:

- Elasticity Support for Dynamic Clusters
- Resource Groups
- Resource Group Templates and Resource Overrides
- Named Concurrent Edit Sessions
- REST Resources for WebLogic Server Management
- Fusion Middleware Control
- Security
- Logging Enhancements
- WebLogic Diagnostic Framework
- WebLogic Server Development and Supplemental Distribution
- Zero Downtime Patching
- WLST
- Resource Consumption Management

2.7.1 Elasticity Support for Dynamic Clusters

WebLogic Server 12.2.1 introduces elasticity. Elasticity enables the automatic scaling of dynamic clusters and re-provisioning of associated resources based on demand. The Elasticity Framework leverages the WebLogic Diagnostic Framework (WLDF) policies and actions system.

For more information, see *Configuring Elasticity in Dynamic Clusters for Oracle WebLogic Server*.

2.7.2 Resource Groups

WebLogic Server Multitenant introduces resource groups as a convenient way to group together Java EE applications and the resources they use into a distinct administrative unit within the domain. The resources and applications in a resource group are "fully qualified" in that the administrator provides all the information

needed to start or connect to those resources, including credentials for connecting to a data source and targeting information for Java EE applications. A resource group will either contain these deployable resources directly or refer to a resource group template which contains the resources. Resource groups can be defined at the domain level, or be specific to a domain partition.

For more information, see "Configuring Resource Groups" in *Using WebLogic Server Multitenant*.

2.7.3 Resource Group Templates and Resource Overrides

Resource group templates are a named, domain-level collection of deployable resources intended to be used as a pattern by (usually) multiple resource groups. Each resource group that refers to a given template will have its own runtime copies of the resources defined in the template. A resource group template is a convenient way to define and replicate resources for multiple tenants. Resource group templates make it very easy to deploy the same collection of applications and resources to multiple domain partitions.

Resource group templates are particularly useful in SaaS environments where WebLogic Server Multitenant activates the same applications and resources multiple times, once per domain partition. Some of the information about such resources is the same across all domain partitions, while some of it, such as JMS queues and database connections, varies from one partition to the next. WebLogic Server Multitenant provides several methods for overriding resource definitions:

- Resource Override Configuration MBeans
- Resource deployment plans
- Partition-specific application deployment plans

Administrators can employ and combine any of these techniques.

For more information, see "Configuring Resource Group Templates" and "Configuring Resource Overrides" in *Using WebLogic Server Multitenant*.

2.7.4 Named Concurrent Edit Sessions

Prior to this release, WebLogic Server supported only one active configuration edit session at a time. The system administrator got a global edit lock, made changes, and then activated them. Other administrators could not make changes at the same time. However, Oracle WebLogic Server 12.2.1 enables multiple, named concurrent edit sessions, which allows more than one administrator to make configuration changes at the same time. This is typically useful when multiple administrators work in different parts of the system. Also, when configuring a system takes a long time because of the serial execution of configuration commands, a single administrator can open multiple named edit sessions. This saves time by running the configuration edit sessions in parallel.

In a multitenant environment, more than one administrator will need to make configuration changes concurrently. A multitenant WebLogic domain contains multiple partitions each with its own administrator. Partition administrators must be able to make configuration changes to their partitions and the resources deployed in them without affecting other partition administrators or the WebLogic system administrator. Multiple, named concurrent edit sessions support one or more configuration edit sessions per partition plus global configuration edit sessions.

For more information, see "Managing Named Concurrent Edit Sessions" in *Using WebLogic Server Multitenant*.

2.7.5 REST Resources for WebLogic Server Management

RESTful Management Services are a publicly documented programming interface to Oracle WebLogic Server. In each release of WebLogic Server, the availability of REST resources for WebLogic Server administration has been enhanced and extended. In Oracle WebLogic Server 12.2.1, WebLogic RESTful management resources provide a comprehensive public interface for configuring, monitoring, deploying and administering WebLogic Server in all supported environments.

For information about the RESTful management resources provided in WebLogic Server 12.2.1, see *Administering Oracle WebLogic Server with RESTful Management Services*.

2.7.6 Fusion Middleware Control

Fusion Middleware Control provides management support for all Fusion Middleware components, including WebLogic Server. Use Fusion Middleware Control to manage WebLogic Server when using other Fusion Middleware products in addition to WebLogic Server.

In WebLogic Server 12.2.1, the following subsets of functionality are now available in Fusion Middleware Control:

- Create WebLogic Server clusters, server instances, domains, machines, and server templates

- Configure applications and libraries

- Create and configure UCP and proxy data sources

- Create and configure JMS servers, Store-and-Forward agents, JMS modules, JMS resources, path services, messaging bridges, and messaging bridge destinations

- Create and configure security realms

- Manage WebLogic Server diagnostics

- Configure elasticity for dynamic clusters

- Manage WebLogic Server in a multitenant environment

For more information, see *Administering Oracle WebLogic Server with Fusion Middleware Control*.

2.7.7 Security

The new security features provided in WebLogic Server 12.2.1 are described in the following sections:

- Java EE 7 Standards Support

- LDAP Authentication Provider Manageability Enhancements

- Support for weblogic-jwt-token

2.7.7.1 Java EE 7 Standards Support

WebLogic Server 12.2.1 supports the following Java EE 7 standards and features for security:

- Java Authorization Contract for Containers 1.5 (JSR 115)

- Java Authentication Service Provider Interface for Containers (JASPIC) 1.1 (JSR 196)

- Packaged Permissions (Java EE 7 Platform Specification)

- Uncovered HTTP methods for Servlet 3.1 (JSR 340)

2.7.7.2 LDAP Authentication Provider Manageability Enhancements

The following enhancements have been added to the LDAP Authentication provider improve the configuration process:

- LDAP Authentication provider performance enhancements for improved caching, searching, and LDAP server connection handling, such as:

  - The ability to collect hit/miss metrics on user and group caching, allowing you to determine the best settings for user and group caching to maximize response time and throughput.

  - Support for specifying a timeout on the LDAP server connection.

- Support for testing the LDAP server connection prior activating the LDAP Authentication provider, similar to the way JDBC connections can be tested during data source configuration. Testing occurs automatically at the time you activate the this provider: if the test succeeds, the provider is activated.

For more information, see "Improving the Performance of WebLogic and LDAP Authentication Providers" in *Administering Security for Oracle WebLogic Server 12c (12.2.1)*.

2.7.7.3 Default Minimum TLS Protocol Version

The default minimum version of the Transport Layer Security (TLS) protocol configured in WebLogic Server 12.2.1 is Version 1.1.

2.7.7.4 Support for weblogic-jwt-token

The WebLogic Identity Assertion and WebLogic Credential mapping providers have been enhanced to include support for the JSON web token, `weblogic-jwt-token`. This token type, which is configured by default in these security providers, is used internally for propagating identity among web applications in the domain.

2.7.8 Logging Enhancements

In WebLogic Server 12.2.1, the WebLogic logging services include the following changes:

- Partition scope logging — The logs for several WebLogic Server components, such as partition scope JMS, SAF, and servlet resources, are kept in partition-specific log files. The logs for server and domain scope resources, such as the server scope HTTP access log, the Harvester component, the Instrumentation component, and also the server and domain logs, can be tagged with partition-specific information to enable logging that is performed on behalf of a partition to be identified and made available to partition users.

  For details, see "Monitoring and Debugging Partitions" in *Using WebLogic Server Multitenant*.

> **Note:** To revert the format of generated log messages to that they are compatible with the format used in versions of WebLogic Server prior to 12.2.1, you can enable the `DomainMBean.LogFormatCompatibilityEnabled` attribute. For information, see "Log File Format Compatibility with Previous WebLogic Server Versions" in *Configuring Log Files and Filtering Log Messages for Oracle WebLogic Server*.

- Monitoring for excessive logging — When enabled, the logging service monitors the domain for excessive rates of logging and, when present, suppresses messages that are being generated repeatedly.

  For details, see "Preventing Excessive Logging" in *Configuring Log Files and Filtering Log Messages for Oracle WebLogic Server*.

2.7.9 WebLogic Diagnostic Framework

In WebLogic Server 12.2.1, the WebLogic Diagnostic Framework (WLDF) includes the following changes:

- The terms *watch* and *notification* are replaced by *policy* and *action*, respectively. However, the definition of these terms has not changed.

- Four new action types are introduced as part of the Policy and Action component of WLDF. Actions are triggered when a policy expression evaluates to `true`. In addition to JMX notification actions, JMS message actions, SMTP (e-mail) actions, SNMP trap actions, and diagnostic image actions, WebLogic Server now supports the following new action types:

  – Elastic actions — scale a dynamic cluster up or down

  – REST notification — sends a notification to a REST endpoint

  – Script — executes an external command line script

  – Log — sends a custom message to the server log

  In addition, WLDF enhances the SMTP action to allow you to send custom subject and body elements in an email message.

  For more information, see "Configuring Actions" in *Configuring and Using the Diagnostics Framework for Oracle WebLogic Server*.

- WebLogic Server 12.2.1 introduces dynamic debug patches. Dynamic debug patches allow you to capture diagnostic information using a patch that is activated and deactivated without requiring a server restart. Dynamic debug patching requires target WebLogic Server instances to be started with the WLDF instrumentation agent. For more information, see "Using Debug Patches" in *Configuring and Using the Diagnostics Framework for Oracle WebLogic Server*.

- Oracle WebLogic Server 12.2.1 introduces smart rules. Smart rules are prepackaged policy expressions with a set of configurable parameters that allow the end user to create a complex policy expression just by specifying the values for these configurable parameters. For more information, see "Configuring Smart Rule Based Policies" in *Configuring and Using the Diagnostics Framework for Oracle WebLogic Server*.

- When you initiate a diagnostic image capture, the images produced by the different server subsystems are captured and combined into a single `.zip` file. In previous releases of WebLogic Server, the components of a diagnostics image

capture file all used the `.img` extension even though these files are all in text format and can be viewed in a text editor. In WebLogic Server 12.2.1, the file extensions have been updated to either `.txt` or `.xml` to clarify that these are text files.

For more information, see "Data Included in the Diagnostics Image Capture File" in *Configuring and Using the Diagnostics Framework for Oracle WebLogic Server*.

■ WLST command changes, described in Section 2.7.12, "WLST".

2.7.10 WebLogic Server Development and Supplemental Distribution

As of WebLogic Server 12.2.1, the WebLogic Server Development and Supplemental distributions are available as JAR files and are installed using the `java` command. The installation uses the Oracle Universal Installer (OUI) and is automatically done in silent mode; you only need to specify the `ORACLE_HOME` location for the installation.

2.7.11 Zero Downtime Patching

WebLogic Zero Downtime Patching (ZDT Patching) automates the rollout of out-of-place patching or updates across a domain while allowing your applications to continue servicing requests. To use ZDT Patching, you create a workflow that orchestrates how updates are rolled out, and then you execute the workflow using use either WLST or the WebLogic Server Administration Console.

ZDT Patching supports the following workflow types:

Workflow	Description
Move servers to a patched Oracle Home	Transitions the Administration Server or clusters, or both, to another Oracle Home that has already been patched using OPatch.
Update to a new Java version	Updates the Administration Server or clusters, or both, to use a newly installed Java Home.
Deploy updated applications	Deploys updated applications to the selected clusters.
Perform a rolling restart of servers	Sequentially and safely restarts the Administration Server or servers in the selected clusters, or both, including graceful shutdown and restart.

For more information, see *Administering Zero Downtime Patching Workflows*.

2.7.12 WLST

This section describes new WLST commands for WebLogic Server and changes to existing WLST commands in WebLogic Server 12.2.1. Additional new WLST commands and arguments for multitenancy are described in Section 2.1.1, "WLST Changes to Support Multitenancy".

2.7.12.1 New Domain Commands

The following WLST commands related to domain creation and domain extension are added in WebLogic Server 12.2.1:

■ `setTopologyProfile`—Sets the topology profile at domain creation to either `Compact` or `Expanded`.

■ `selectTemplate`—Selects an existing domain or extension template for creating or extending a domain.

- `selectCustomTemplate`—Selects an existing custom domain or extension template for creating or extending a domain.

- `loadTemplates`—Loads all templates that were selected using `selectTemplate` or `selectCustomTemplate`.

- `readTemplateForUpdate`—Opens an existing domain template for template update.

- `unselectTemplate`—Deselects a currently selected template.

- `unselectCustomTemplate`—Deselects a currently selected custom template.

- `showTemplates`—Displays all currently selected and loaded templates.

- `showAvailableTemplates`—Displays all currently selected templates for loading.

2.7.12.2 New Diagnostic Commands

The following diagnostics commands were added in WebLogic Server 12.2.1:

- `purgeCapturedImages`—Purges the diagnostic image files on the server as per the specified age criteria.

- `listDebugPatches`—Lists active and available debug patches on the specified targets.

- `showDebugPatchInfo`—Displays details about a debug patch on the specified targets.

- `activateDebugPatch`—Activates a debug patch on the specified targets.

- `deactivateDebugPatches`—Deactivates debug patches on the specified targets.

- `deactivateAllDebugPatches`—Deactivates all debug patches on the specified targets.

- `listDebugPatchTasks`—Lists debug patch tasks from the specified targets.

- `purgeDebugPatchTasks`—Purges debug patch tasks from the specified targets.

- `getAvailableDiagnosticDataAccessorNames`—Gets the diagnostic data accessor names that are currently available on a server or partition.

- `exportHarvestedTimeSeriesData`—Exports the harvested metric data within the specified internal, in CSV format.

- `exportHarvestedTimeSeriesDataOffline`—Exports the harvested metric data in offline mode within the specified internal, in CSV format.

In addition, a new optional parameter, `last`, is available to the following commands:

- `exportDiagnosticData`

- `exportDiagnosticDataFromServer`

- `exportHarvestedTimeSeriesData`

- `exportHarvestedTimeSeriesDataOffline`

The `last` option allows you to specify the timestamp range specification for the last *n* records. When specified, the `beginTimestamp` and `endTimestamp` options are ignored. The format is XXd YYh ZZm. For example, `1d 5h 30m` specifies data that is one day, five hours and 30 minutes old. You can specify any combination of day, hour, and minute components in any order.

2.7.12.3 New Node Manager Commands

The following Node Manager WLST commands were added in WebLogic Server 12.2.1:

- `nmrestart`—Restarts the Node Manager instance.

- `nmExecScript`—Executes the named script using the connected Node Manager.

2.7.12.4 New Edit Session Commands

The following WLST edit session management commands were added in WebLogic Server 12.2.1:

- `createEditSession`—Creates a new WLST edit session.

- `showEditSession`—Displays information about the specified edit sessions.

- `destroyEditSession`—Removes an open edit session.

- `edit(editSessionName)`—Creates a new edit session with the specified name or navigates to an existing edit session with the specified name.

- `resolve`—Detects any external modifications and conflicts, and resolves them.

2.7.12.5 New System Component Commands

The following system component WLST commands were added in WebLogic Server 12.2.1:

- `resync`—Re-synchronizes configuration files for a system component.

- `resyncAll`—Re-synchronizes configuration files for all system components.

- `showComponentChanges`—Displays changes to a system component's configuration files on a remote node.

- `pullComponentChanges`—Removes changes to a system component's configuration files on a remote node.

- `setOverWriteComponentChanges`—Forces changes to all system components during activation.

2.7.12.6 Other New Commands

Other WLST commands that were added in WebLogic Server 12.2.1 are:

- `setShowLSResult`—Specifies whether `ls()` should log its output to standard output.

- `consolidateDataSources`—Consolidates multiple data sources into a single data source.

- `scaleUp`—Increases the number of running dynamic servers in the specified dynamic cluster.

- `scaleDown`—Decreases the number of running dynamic servers in the specified dynamic cluster.

2.7.12.7 Modifications to Existing Commands

The following modifications were made to existing WLST commands in WebLogic Server 12.2.1.

- The `format` argument has been added to the `exportDiagnosticData` and `exportDiagnosticDataFromServer` commands. Use this argument to specify the format in which data is exported.

- The `last` argument has been added to the `exportDiagnosticData`, `exportDiagnosticDataFromServer`, `exportHarvestedTimeSeriesData`, and `exportHarvestedTimeSeriesDataOffline` commands. This argument is a timestamp range specification for the last *n* seconds.

- The `Server` argument has been added to the `getAvailableCapturedImages` command. Use this argument to specify the server from which to obtain a list of available images.

- The `waitForAllSessions` argument has been added to the `shutdown` command. Use this argument to specify whether WLST should wait for all HTTP sessions to complete while shutting down.

- The following arguments were added to the `startNodeManager` command:

  - `block`—Specifies whether WLST should block until it successfully connects to Node Manager or fails to connect within the specified timeout.

  - `nmConnectOptions`—When `block` is `true`, use this argument to specify a list of Node Manager connection options.

  - `timeout`—The number of milliseconds to wait for Node Manager to connect.

2.7.12.8 idd Variable and Argument

The idd variable has been added to WLST. This WLST variable is the Identity Domain of the user who is currently connected to WLST.

In addition, the idd argument has been added to the `connect` command to specify the Identity Domain of the user who is connecting.

2.7.13 Resource Consumption Management

Resource Consumption Management allows WebLogic system administrators to specify resource consumption management policies (such as constraints, recourse actions, and notifications) on JDK-managed resources such as CPU, Heap, File, and Network.

For more information, see "Configuring Resource Consumption Management" in *Using WebLogic Server Multitenant*.

2.8 Documentation Update History for Release 12.2.1

The following table summarizes updates made to the Oracle WebLogic Server 12.2.1 documentation library since its initial release in October 2015:

Date	Description of Updates
October 26, 2015	Updated the following documents:
	■ *Using WebLogic Server Multitenant* — Miscellaneous updates to several chapters
	■ *Configuring and Using the Diagnostics Framework for Oracle WebLogic Server* — Revised "Configuring Policies" to clarify distinctions among the various policy types, and added more details regarding scheduled policies.

Date	Description of Updates
November 2, 2015	■ Updated *Release Notes for Oracle WebLogic Server*. ■ Updated "Configuring Actions" in *Configuring and Using the Diagnostics Framework for Oracle WebLogic Server* to add descriptions of the log, REST, scale up, scale down, and script actions, which are new in Oracle WebLogic Server 12.2.1.
November 9, 2015	■ Corrected the description of the new parallel deployment feature in *Deploying Applications to Oracle WebLogic Server* and *Upgrading Oracle WebLogic Server*. For information, see "Parallel Deployment" in *Upgrading Oracle WebLogic Server*. ■ Added the section "Accessing Diagnostic Data" to *Using WebLogic Server Multitenant*, which explains how partition administrators can access partition scope log and diagnostic data.
November 16, 2015	■ Fixed access to *MBean Reference for Oracle WebLogic Server*. ■ Corrected the instructions for running the elasticity demo, which is described in "Policy Based Scaling" in *Configuring Elasticity in Dynamic Clusters for Oracle WebLogic Server*. ■ Added the section "Best Practices" in *Using WebLogic Server Multitenant* that provides advice and best practices for beginning JMS users as well as advanced JMS users in an MT environment.
November 23, 2015	■ Updated *Release Notes for Oracle WebLogic Server*. ■ Updated "Supported FIPS Standards and Cipher Suites" in *Administering Security for Oracle WebLogic Server 12c (12.2.1)* to identify the specific versions of RSA Cert-J, Crypto-J, and SSL-J that are supported in Oracle WebLogic Server 12.2.1.
December 2015	■ Added the new chapter "Configuring Oracle Traffic Director" to *Using WebLogic Server Multitenant*. This chapter explains how OTD distributes incoming client requests to WebLogic Server MT partitions by coordinating its configuration with WebLogic Server MT partition management. It also describes an initial, one-time OTD configuration that you must perform to enable OTD multitenancy features.

Date	Description of Updates
March 2016	Made the following updates to *Configuring Elasticity in Dynamic Clusters for Oracle WebLogic Server*:

- "Requirements for Configuring Elasticity" clarifies that elastic operations for a dynamic cluster must be configured in a domain-scope diagnostic module that is targeted to the Administration Server.

- "Configuring the Script Interceptor" states that if a preprocessor script fails, neither any subsequently configured interceptors nor the associated scaling operation is invoked. Likewise, if a post-processor script fails, the scaling action that preceded the invocation of that script cannot be reverted or otherwise cancelled.

- The section "Reserved Environment Variables" has been added that explains how script interceptors can pass both computed and dynamic parameters to scripts as environment variables. This section also lists and summarizes the set of reserved environment variables that can also be used in scripts.

Made the following updates to *Configuring and Using the Diagnostics Framework for Oracle WebLogic Server*:

- The section "Configuring Log Policies" notes that log policies that search for the RUNNING state message ID should search for message ID `BEA-000365`, not `BEA-000360`.

- The section "Configuring Elastic Actions" now explains that elastic actions cannot be cancelled after they have been initiated, and also includes best practices about limiting the duration of server shutdown time during scale down operations.

- The `instrumentationEvent` and `log` bean descriptions have been updated to list their attributes and to give examples of their use in EL policy expressions.

2.9 Standards Support, Supported Configurations, and WebLogic Server Compatibility

The following sections describe WebLogic Server standards support, supported system configuration, and WebLogic Server compatibility:

- Standards Support

- Supported Configurations

- Licensing Information

- WebLogic Server Compatibility

- Database Interoperability

2.9.1 Standards Support

WebLogic Server 12.2.1 supports the following standards and versions.

- Java Standards

- Web Services Standards

- Other Standards

2.9.1.1 Java Standards

Table 2–1 lists currently supported Java standards.

> **Note:**   See "WebLogic Server Security Standards" in *Administering Security for Oracle WebLogic Server 12c (12.2.1)* for the currently supported security standards, such as JAAS, JASPIC, JACC, JCE, and so forth.

Table 2–1   Java Standards Support

Standard	Version
Batch Application Processing (JSR 352)	1.0
Contexts and Dependency Injection for Java EE	1.1
Dependency Injection for Java EE	1.0
Concurrent Managed Objects (JSR 236)	1.0
Expression Language (EL)	3.0, 2.2, 2.1, 2.0
	Only JSP 2.0 and greater supports Expression Language 2.x.
Java API for JSON Processing (JSR-353)	1.0
Java API for XML-Based Web Services (JAX-WS)	2.2, 2.1, 2.0
Java API for RESTful Web Services (JAX-RS)	2.0
Java API for WebSocket	1.1
JavaBeans Activation Framework	1.1
Java EE	7.0
Java EE Application Deployment	1.2
Java EE Bean Validation	1.1
Java EE Common Annotations	1.2
Java EE Connector Architecture	1.7
Java EE EJB	3.2, 3.1, 3.0, 2.1, 2.0, and 1.1
Java EE Enterprise Web Services	1.3, 1.2, 1.1
Jave EE Interceptors	1.1
Java EE JDBC	4.0, 3.0
Java EE JMS	2.0, 1.1, 1.0.2b
Java EE JNDI	1.2
Java EE JSF	2.2, 2.1.*, 2.0, 1.2, 1.1
Java EE JSP	2.3, 2.2, 2.1, 2.0, 1.2, and 1.1
	JSP 1.2. and 1.1 include Expression Language (EL), but do not support EL 2.x or greater.
Java EE Managed Beans	1.0
Java EE Servlet	3.1, 3.0, 2.5, 2.4, 2.3, and 2.2
Java RMI	1.0
JavaMail	1.4
Java Transaction API	1.2
JAX-B	2.2, 2.1, 2.0
JAX-P	1.3, 1.2, 1.1

Table 2–1   (Cont.) Java Standards Support

Standard	Version
JAX-R	1.0
JAX-RPC	1.1
JDKs	8.0 (8.0 and 7.0 for clients)
	See Section 2.4, "JDK 8 Certification" for details.
JMX	2.0
JPA	2.1, 2.0., 1.0
JSR 77: Java EE Management	1.1
JSTL	1.2
Managed Beans	1.0
OTS/JTA	OTS 1.2 and JTA 1.2
RMI/IIOP	1.0
SOAP Attachments for Java (SAAJ)	1.3, 1.2
Streaming API for XML (StAX)	1.0
Web Services Metadata for the Java Platform	2.0, 1.1

2.9.1.2 Web Services Standards

For the current list of standards supported for WebLogic web services, see "Features and Standards Supported by WebLogic Web Services" in *Understanding WebLogic Web Services for Oracle WebLogic Server*.

2.9.1.3 Other Standards

Table 2–2 lists other standards that are supported in WebLogic Server 12.2.1.

> **Note:**   See "WebLogic Server Security Standards" in *Administering Security for Oracle WebLogic Server 12c (12.2.1)* for additional information on standards relating to security, such as SSL, TLS, and XACML, and so forth.

Table 2–2   Other Standards

Standard	Version
X.509	v3
LDAP	v3
TLS	v1.1, v1.2
HTTP	1.1
SNMP	SNMPv1, SNMPv2, SNMPv3
xTensible Access Control Markup Language (XACML)	2.0
Partial implementation of Core and Hierarchical Role Based Access Control (RABC) Profile of XACML	2.0

Table 2–2 (Cont.) Other Standards

Standard	Version
Internet Protocol (IP)	Versions:
	■ v6
	■ v4

For more information about IPv6 support for all Fusion Middleware products, see the Oracle Fusion Middleware Supported System Configurations page on Oracle Technology Network.

2.9.2 Supported Configurations

For the most current information on supported configurations, see the Oracle Fusion Middleware Supported System Configurations page on Oracle Technology Network.

Please note the following restrictions and advice when running Oracle WebLogic Server 12.2.1, and Oracle WebLogic Server 12.2.1 applications, on Java SE 8:

■ Oracle WebLogic Server 12.2.1 does not support applications using the new Java SE 8 fork/join and parallel streams features. Avoid these features when building Oracle WebLogic Server 12.2.1 applications using Java SE 8. The reason for this restriction is that the threads used by the fork/join thread pool will not be WebLogic Server managed threads. Any of the work performed in these threads may not be able to make use of WebLogic Server or Java EE facilities because the state of these threads, including security and transaction state, may not be created properly. Further, these threads will not be controlled by WebLogic Server Work Manager thread management facilities, possibly resulting in excessive thread usage.

■ Check all third party vendor software you are using for Java SE 8 compatibility. It may be necessary to upgrade to a later version of the software that correctly handles Java SE 8 classes, and some software may not yet be compatible. For example, the current version of the open source tool "jarjar" does not work correctly with Java SE 8 yet.

■ Java SE 8 has new APIs for JDBC 4.2 that are supported for versions of WebLogic Server 12.1.3 and later that are running on Java SE 8 with a JDBC driver that supports JDBC 4.2. However, although the Oracle JDBC thin driver bundled with WebLogic Server is certified on Java SE 8, the Oracle JDBC thin driver does not support JDBC 4.2. The Derby 10.10 driver that is shipped with Oracle WebLogic Server 12.2.1 has been tested with JDBC 4.2 and may be used. The corresponding Derby documentation is available at
http://db.apache.org/derby/docs/10.10/ref/rrefjdbc4_2summary.html.

■ When running using SSL connections with JCE on JDK 8, it may be necessary to install the Java Cryptography Extension (JCE) Unlimited Strength Jurisdiction Policy Files 8. You can download the JCE Unlimited Strength Jurisdiction Policy Files for JDK 8 at
http://www.oracle.com/technetwork/java/javase/downloads/jce8-download-2133166.html.

2.9.3 Licensing Information

For the most current information on Oracle Fusion Middleware Licensing, see Licensing Information.

2.9.4 WebLogic Server Compatibility

For the most current information on compatibility between current version of WebLogic Server and previous releases, see "WebLogic Server Compatibility" in *Understanding Oracle WebLogic Server*.

2.9.5 Database Interoperability

The certification matrices and My Oracle Support Certifications define the following terms to differentiate between types of database support:

- Application Data Access
- Database Dependent Features

2.9.5.1 Application Data Access

Application Data Access refers to those applications that use the database for data access only and do not take advantage of WebLogic Server features that are Database dependant. WebLogic Server support of databases used for application data access only are less restrictive than for database dependent features.

WebLogic Server provides support for application data access to databases using JDBC drivers that meet the following requirements:

- The driver must be thread safe.
- The driver must implement standard JDBC transactional calls, such as `setAutoCommit()` and `setTransactionIsolation()`, when used in transactional aware environments.

Note the following restrictions:

- JDBC drivers that do not implement serializable or remote interfaces cannot pass objects to an RMI client application.
- Simultaneous use of automatic database connection failover and load balancing and global transactions (XA) with a highly-available (HA) DBMS architecture is supported with Oracle DB RAC only, and only for the Oracle DB RAC versions indicated on the **System** worksheet. These HA capabilities are only supported by Active GridLink for RAC and Multi Data Sources with RAC. These HA capabilities are not supported on other Oracle DB RAC versions or with other HA DBMS technologies on other non-Oracle DB products. Multi Data Sources are supported on other Oracle DB versions, and with non-Oracle DB technologies, but not with simultaneous use of automatic failover and load balancing and global transactions.
- Application data access to databases meeting the restrictions articulated above is supported on other Oracle DB versions, in addition to those documented in the certification matrix.
- WebLogic Type 4 JDBC drivers also support the following databases. For these databases, WebLogic Server supports application data access only, and does not support WebLogic Server database dependent features:
  - DB2 for z/OS 10.1
  - Informix 11.7+

2.9.5.2 Database Dependent Features

When WebLogic Server features use a database for internal data storage, database support is more restrictive than for application data access. The following WebLogic Server features require internal data storage:

- Container Managed Persistence (CMP)

- Rowsets

- JMS/JDBC Persistence and use of a WebLogic JDBC Store

- JDBC Session Persistence

- RDBMS Security Providers

- Database Leasing (for singleton services and server migration)

- JTA Logging Last Resource optimization

- JDBC TLog

2.10 Deprecated Functionality (WebLogic Server 12.2.1)

Information about deprecated functionality for WebLogic Server 12.2.1 can be found on My Oracle Support at `https://support.oracle.com/`. Search for "Deprecated Features" in the **Search Knowledge Base** field.

2.10.1 WebLogic Full Client

The WebLogic full client, `wlfullclient.jar`, is deprecated as of WebLogic Server 12.1.3.

2.10.2 Log4j

The use of Log4j with the WebLogic logging service, as an alternative to Java logging, is deprecated as of WebLogic Server 12.1.3.

2.10.3 LogMBean.ServerLoggingBridgeUserParentLoggersEnabled Attribute

The `ServerLoggingBridgeUserParentLoggersEnabled` attribute on the `LogMBean` is deprecated as of WebLogic Server 12.1.3.

2.10.4 Username and Password in boot.properties File

As of WebLogic Server 12.1.1, the boot username and password system properties `weblogic.management.username` and `weblogic.management.password` have been deprecated and will be removed in a future release, and you will no longer be able to specify the username and password in the command for starting WebLogic Server in production mode.

As an alternative, Oracle recommends that you use the `boot.properties` file to specify the boot username and password for WebLogic Server. For more information about the `boot.properties` file, see "Boot Identity Files" in *Administering Server Startup and Shutdown for Oracle WebLogic Server*.

For information about other methods you can use to provide user credentials, see "Provide User Credentials to Start and Stop Servers" in *Administering Server Startup and Shutdown for Oracle WebLogic Server*.

2.10.5 Maven 11x Plug-In Deprecated

The `weblogic-maven-plugin` plug-in delivered in WebLogic Server 11*g* Release 1 is deprecated as of release 12.1.2. Oracle recommends that you instead use the WebLogic Server Maven plug-in introduced in version 12.1.2. See "Using the WebLogic Development Maven Plug-in" in *Developing Applications for Oracle WebLogic Server* for complete documentation.

2.10.6 JSP Tags for XML Processing

As of WebLogic Server 12.1.2, XSLT JSP tags and the WebLogic XSLT JSP Tag Library have been deprecated and will be removed in a future release. You can use JAXP to transform XML data. For more information, see "Transforming XML Documents" in *Developing XML Applications for Oracle WebLogic Server*.

2.10.7 Deprecated Functionality in WLST

The following functionality in WLST has been deprecated in WebLogic Server 12.2.1.

2.10.7.1 Server Argument to Diagnostics Commands

The `Server` argument to the following WLST diagnostics commands has been deprecated:

- `captureAndSaveDiagnosticImage`

- `createSystemResourceControl`

- `destroySystemResourceControl`

- `disableSystemResource`

- `enableSystemResource`

- `listSystemResourceControls`

The `Server` argument is being replaced by the `Target` argument. For more information, see "Diagnostics Commands" in *WLST Command Reference for WebLogic Server*.

2.10.7.2 The `addTemplate` and `readTemplate` Control Commands

The `addTemplate` and `readTemplate` commands have been deprecated in WebLogic Server 12.2.1 and will be removed in a future release. Use the `selectTemplate` and `loadTemplates` commands instead. For more information and examples, see "Creating and Updating a WebLogic Domain" in *Understanding the WebLogic Scripting Tool*.

2.10.7.3 Implicit Importing of Modules Using WLST

Support for using WLST to implicitly import modules into an application has been deprecated. When using WLST to import modules, Oracle recommends doing the operation explicitly.

The following WLST snippet shows an explicit import of the module `EJBResource` from `weblogic.security.service`:

```
@ from weblogic.security.service import EJBResource
ejbRes = EJBResource('DDPoliciesEar', 'DDPolinEarMiniAppBean.jar',
'DDRolesAndPolicies', 'getSubject', 'Remote', None)
```

2.10.8 URL Format for REST Management

The URL format for REST management APIs has changed in 12.1.3. The URL format introduced in 12.1.2 will continue to work, but is deprecated in WebLogic Server 12.2.1.

2.10.9 WebLogic Server/Spring Integration Features

Integration features for WebLogic Server/Spring are deprecated in Oracle WebLogic Server 12.2.1, and it is not required to support the Oracle WebLogic Server 12.2.1 Spring console extension when using JRF or restricted-JRF.

2.10.10 Deprecated Diagnostics Exceptions

The following exceptions in the Harvester component of the WebLogic Diagnostics Framework are deprecated:

```
weblogic.diagnostics.harvester.HarvesterException
weblogic.diagnostics.harvester.HarvesterException.AmbiguousInstanceName
weblogic.diagnostics.harvester.HarvesterException.AmbiguousTypeName
weblogic.diagnostics.harvester.HarvesterException.HarvestableInstancesNotFoundException
weblogic.diagnostics.harvester.HarvesterException.HarvestableTypesNotFoundException
weblogic.diagnostics.harvester.HarvesterException.HarvestingNotEnabled
weblogic.diagnostics.harvester.HarvesterException.MissingConfigurationType
weblogic.diagnostics.harvester.HarvesterException.TypeNotHarvestable
```

2.10.11 RESTful Web Services

The following JAX-RS functionality has been deprecated in Oracle WebLogic Server 12.2.1.

2.10.11.1 Jersey 1.18 (JAX-RS 1.1 RI) Client APIs

Support for the Jersey 1.18 (JAX-RS 1.1RI) client APIs, including the `com.sun.jersey` and its nested packages, and the `weblogic.jaxrs.api.client` packages, are deprecated in this release of WebLogic Server but are maintained for backward compatibility. It is recommended that you update your RESTful client applications to use the JAX-RS 2.0 RI client APIs at your earliest convenience. For more information, see *Developing and Securing RESTful Web Services for Oracle WebLogic Server*.

2.10.11.2 Runtime Monitoring MBeans

The following runtime MBeans have been deprecated:

■ `JaxRsMonitoringInfoRuntimeMBean`

■ `JaxRsResourceConfigTypeRuntimeMBean`

> **Note:** The functionality provided by these MBeans has been replaced by new or updated MBeans. For more information, see "Monitoring RESTful Web Services and Clients" in *Developing and Securing RESTful Web Services for Oracle WebLogic Server*.

2.10.12 CacheFilter API

The WebLogic Server API `weblogic.cache.filter.CacheFilter` has been deprecated in Oracle WebLogic Server 12.2.1.

2.10.13 JAX-RPC API

The JAX-RPC API is deprecated in WebLogic Server 12.2.1.

2.10.14 SSLMBean.ExportKeyLifespan Attribute

The `SSLMBean.ExportKeyLifespan` attribute is deprecated in WebLogic Server 12.2.1. This attribute was used by the Certicom-based SSL implementation, which was removed from WebLogic Server in version 12.1.1 and replaced by JSSE. The JSSE implementation in WebLogic Server does not use the `SSLMBean.ExportKeyLifespan` attribute. For more information about JSSE, see "Using the JSSE-Based SSL Implementation" in *Administering Security for Oracle WebLogic Server 12c (12.2.1)*.

2.11 Removed Functionality and Components

The following deprecated components have been removed from release 12.2.1 of WebLogic Server.

2.11.1 Compatibility Security

Compatibility security, which is used for running security configurations developed with WebLogic Server 6.*x*, is no longer supported. The following components that provided Compatibility security in previous releases are removed:

- `CompatibilityRealm`

- `CachingRealm`

- `LDAPRealm`

- `NTRealm`

- `FileRealm`

- Custom security realm

- RDBMS security realm

- Realm Adapter provider

- Realm Adapter Auditing provider

- Realm Adapter Authentication provider

- Realm Adapter Authorization provider

- Realm Adapter Adjudication provider

The following classes that provided support for Compatibility security were removed:

- `weblogic.management.configuration.Acl`

- `weblogic.management.configuration.BasicRealmMBean`

- `weblogic.management.configuration.CachingRealmMBean`

- `weblogic.management.configuration.CustomRealmMBean`

- `weblogic.management.configuration.FileRealmMBean`

- `weblogic.management.configuration.Group`

- `weblogic.management.configuration.LDAPRealmMBean`

- `weblogic.management.configuration.ListResults`

- `weblogic.management.configuration.NTRealmMBean`
- `weblogic.management.configuration.PasswordPolicyMBean`
- `weblogic.management.configuration.Principal`
- `weblogic.management.configuration.RDBMSRealmMBean`
- `weblogic.management.configuration.RealmException`
- `weblogic.management.configuration.RealmIterator`
- `weblogic.management.configuration.RealmMBean`
- `weblogic.management.configuration.RealmManager`
- `weblogic.management.configuration.RemoteEnumeration`
- `weblogic.management.configuration.SecurityMBean`
- `weblogic.management.configuration.UnixRealmMBean`
- `weblogic.management.configuration.User`
- `weblogic.management.mbeans.custom.LDAPRealm`
- `weblogic.management.mbeans.custom.NTRealm`
- `weblogic.management.mbeans.custom.Realm`
- `weblogic.management.mbeans.custom.Security`
- `weblogic.management.mbeans.custom.UnixRealm`
- `weblogic.management.internal.BatchedEnumeration`
- `weblogic.management.internal.RemoteEnumerationImpl`
- `weblogic.management.internal.RemoteRealmException`
- `weblogic.management.internal.RemoteRealmManager`
- `weblogic.management.internal.RemoteRealmManagerImpl`
- `weblogic.security.acl.AbstractListableRealm`
- `weblogic.security.acl.AbstractManageableRealm`
- `weblogic.security.acl.AclEntryImpl`
- `weblogic.security.acl.AclImpl`
- `weblogic.security.acl.AdminPermissions`
- `weblogic.security.acl.CachingRealm`
- `weblogic.security.acl.CertAuthentication`
- `weblogic.security.acl.CertAuthenticator`
- `weblogic.security.acl.ClosableEnumeration`
- `weblogic.security.acl.CredentialChanger`
- `weblogic.security.acl.DebuggableRealm`
- `weblogic.security.acl.DefaultGroupImpl`
- `weblogic.security.acl.DefaultUserImpl`
- `weblogic.security.acl.DynamicUserAcl`
- `weblogic.security.acl.Everyone`

- `weblogic.security.acl.ExplicitlyControlled`
- `weblogic.security.acl.FlatGroup`
- `weblogic.security.acl.GroupImpl`
- `weblogic.security.acl.InvalidLogin`
- `weblogic.security.acl.ListableRealm`
- `weblogic.security.acl.LoginFailureRecord`
- `weblogic.security.acl.ManageableRealm`
- `weblogic.security.acl.OwnerImpl`
- `weblogic.security.acl.PasswordGuessing`
- `weblogic.security.acl.PasswordGuessingWrapper`
- `weblogic.security.acl.PermissionImpl`
- `weblogic.security.acl.PrivilegedAction`
- `weblogic.security.acl.PrivilegedExceptionAction`
- `weblogic.security.acl.Realm`
- `weblogic.security.acl.RealmProxy`
- `weblogic.security.acl.RefreshableRealm`
- `weblogic.security.acl.SSLUserInfo`
- `weblogic.security.acl.Security`
- `weblogic.security.acl.SecurityMessage`
- `weblogic.security.acl.SecurityMulticastRecord`
- `weblogic.security.acl.TTLCache`
- `weblogic.security.acl.UnlockUserRecord`
- `weblogic.security.acl.internal.AuthenticationDelegate`
- `weblogic.security.acl.internal.ClusterRealm`
- `weblogic.security.acl.internal.DefaultRealmImpl`
- `weblogic.security.audit.Audit`
- `weblogic.security.audit.AuditProvider`
- `weblogic.security.internal.RealmTest`
- `weblogic.security.ldaprealm.LDAPRealm`
- `weblogic.security.ldaprealmv1.LDAPDelegate`
- `weblogic.security.ldaprealmv1.LDAPException`
- `weblogic.security.ldaprealmv1.LDAPGroup`
- `weblogic.security.ldaprealmv1.LDAPRealm`
- `weblogic.security.ldaprealmv1.LDAPUser`
- `weblogic.security.ldaprealmv2.LDAPDelegate`
- `weblogic.security.ldaprealmv2.LDAPEntity`
- `weblogic.security.ldaprealmv2.LDAPGroup`

- `weblogic.security.ldaprealmv2.LDAPRealm`

- `weblogic.security.ldaprealmv2.LDAPRealmException`

- `weblogic.security.ldaprealmv2.LDAPUser`

- `weblogic.security.ntrealm.NTDelegate`

- `weblogic.security.ntrealm.NTRealm`

- `weblogic.security.unixrealm.SubprocessException`

- `weblogic.security.unixrealm.UnixDelegate`

- `weblogic.security.unixrealm.UnixGroup`

- `weblogic.security.unixrealm.UnixRealm`

- `weblogic.security.unixrealm.UnixUser`

- `weblogic.security.providers.realmadapter.AdjudicationProviderImpl`

- `weblogic.security.providers.realmadapter.AuditProviderImpl`

- `weblogic.security.providers.realmadapter.AuthenticationProviderImpl`

- `weblogic.security.providers.realmadapter.AuthorizationProviderImpl`

- `weblogic.security.providers.realmadapter.IdentityAsserterImpl`

- `weblogic.security.providers.realmadapter.LoginModuleImpl`

- `weblogic.security.providers.realmadapter.RealmAdapterAdjudicatorImpl`

- `weblogic.security.providers.realmadapter.RealmAdapterAuditorImpl`

- `weblogic.security.providers.realmadapter.RealmAdapterAuthenticatorImpl`

- `weblogic.security.providers.realmadapter.RealmAdapterAuthorizerImpl`

- `RealmAdapterAdjudicatorMBean`

- `RealmAdapterAuditorMBean`

- `RealmAdapterAuthenticatorMBean`

- `RealmAdapterAuthorizerMBean`

2.11.2 6.x Realms

The 6.x realm configuration and associated APIs have been removed from WebLogic Server 12.2.1. The following deprecated configuration MBeans and associated elements have been removed from the DomainMBean configuration element:

Configuration MBean	Associated DomainMBean Configuration Element
`SecurityMBean`	`<security>`
`FileRealmMBean`	`<file-realm>`
`CachingRealmMBean`	`<caching-realm>`
`PasswordPolicyMBean`	`<password-policies>`
`BasicRealmMBean`	`<basic-realm>`
`CustomRealmMBean`	`<custom-realm>`
`LDAPRealmMBean`	`<ldap-realm>`
`NTRealmMBean`	`<nt-realm>`

Configuration MBean	Associated DomainMBean Configuration Element
RDBMSRealmMBean	<rdbms-realm>
RealmMBean	<realm>
UnixRealmMBean	<unix-realm>

2.11.3 Certificate Request Generator Servlet

The Certificate Request Generator servlet has been removed from Oracle WebLogic Server 12.2.1, including the `weblogic.servlet.security.CertificateServlet` class.

2.11.4 weblogic.Admin

The `weblogic.Admin` utility has been removed from Oracle WebLogic Server 12.2.1.

2.11.5 JAXR

The JAVA API for XML Registries (JAXR) has been removed from Oracle WebLogic Server 12.2.1.

2.11.6 Jersey 1.18 (JAX-RS 1.1. RI) Server APIs

The Jersey 1.18 (JAX-RS 1.1 RI) server APIs have been removed from Oracle WebLogic Server 12.2.1. You should use the corresponding standard JAX-RS 2.0 or Jersey 2.x APIs instead. For more information, see *Developing and Securing RESTful Web Services for Oracle WebLogic Server*.

2.11.7 WebLogic Keystore Provider

The WebLogic Keystore provider, which was deprecated in previous releases, has been removed from WebLogic Server 12.2.1.

2.11.8 weblogic.security.provider.PrincipalValidatorImpl

The `weblogic.security.provider.PrincipalValidatorImpl` class, which was deprecated in the previous release, is removed from WebLogic Server 12.2.1.

2.11.9 weblogic.xml.stream.util.XMLPullReaderBase

The `weblogic.xml.stream.util.XMLPullReaderBase` class, which was deprecated in a previous release, has been removed from Oracle WebLogic Server 12.2.1.

2.11.10 Connect-Time Failover

Oracle Connect-Time Failover was deprecated in an earlier release. This functionality and the supporting documentation has been removed from Oracle WebLogic Server 12.2.1.

2.11.11 Compatibility MBean Server and Type-Safe MBean Interfaces

In Oracle WebLogic Server 12.2.1, the compatibility MBean server and all type-safe interfaces to WebLogic Server MBeans have been removed.