Sun GlassFish Enterprise Server v3 Administration Guide

Sun Microsystems, Inc.
4150 Network Circle
Santa Clara, CA 95054
U.S.A.

Contents

Figures

Tables

Examples

Preface

The *Sun GlassFish Enterprise Server v3 Administration Guide* provides instructions for configuring and administering the Sun GlassFish Enterprise Server.

This preface contains information about and conventions for the entire Sun GlassFish™ Enterprise Server (Enterprise Server) documentation set.

Enterprise Server v3 is developed through the GlassFish project open-source community at `https://glassfish.dev.java.net/`. The GlassFish project provides a structured process for developing the Enterprise Server platform that makes the new features of the Java EE platform available faster, while maintaining the most important feature of Java EE: compatibility. It enables Java developers to access the Enterprise Server source code and to contribute to the development of the Enterprise Server. The GlassFish project is designed to encourage communication between Sun engineers and the community.

The following topics are addressed here:

Enterprise Server Documentation Set

The Enterprise Server documentation set describes deployment planning and system installation. The Uniform Resource Locator (URL) for Enterprise Server documentation is `http://docs.sun.com/coll/1343.9`. For an introduction to Enterprise Server, refer to the books in the order in which they are listed in the following table.

TABLE P–1 Books in the Enterprise Server Documentation Set

Book Title	Description
Release Notes	Provides late-breaking information about the software and the documentation. Includes a comprehensive, table-based summary of the supported hardware, operating system, Java™ Development Kit (JDK™), and database drivers.
Quick Start Guide	Explains how to get started with the Enterprise Server product.
Installation Guide	Explains how to install the software and its components.
Upgrade Guide	Explains how to upgrade to the latest version of Enterprise Server. This guide also describes differences between adjacent product releases and configuration options that can result in incompatibility with the product specifications.
Administration Guide	Explains how to configure, monitor, and manage Enterprise Server subsystems and components from the command line by using the asadmin(1M) utility. Instructions for performing these tasks from the Administration Console are provided in the Administration Console online help.
Application Deployment Guide	Explains how to assemble and deploy applications to the Enterprise Server and provides information about deployment descriptors.
Your First Cup: An Introduction to the Java EE Platform	Provides a short tutorial for beginning Java EE programmers that explains the entire process for developing a simple enterprise application. The sample application is a web application that consists of a component that is based on the Enterprise JavaBeans™ specification, a JAX-RS web service, and a JavaServer™ Faces component for the web front end.
Application Development Guide	Explains how to create and implement Java Platform, Enterprise Edition (Java EE platform) applications that are intended to run on the Enterprise Server. These applications follow the open Java standards model for Java EE components and APIs. This guide provides information about developer tools, security, and debugging.
Add-On Component Development Guide	Explains how to use published interfaces of Enterprise Server to develop add-on components for Enterprise Server. This document explains how to perform *only* those tasks that ensure that the add-on component is suitable for Enterprise Server.
Embedded Server Guide	Explains how to run applications in embedded Enterprise Server and to develop applications in which Enterprise Server is embedded.
Scripting Framework Guide	Explains how to develop scripting applications in languages such as Ruby on Rails and Groovy on Grails for deployment to Enterprise Server.
Troubleshooting Guide	Describes common problems that you might encounter when using Enterprise Server and how to solve them.

TABLE P–1 Books in the Enterprise Server Documentation Set   *(Continued)*

Book Title	Description
Error Message Reference	Describes error messages that you might encounter when using Enterprise Server.
Reference Manual	Provides reference information in man page format for Enterprise Server administration commands, utility commands, and related concepts.
Domain File Format Reference	Describes the format of the Enterprise Server configuration file, domain.xml.
Java EE 6 Tutorial, Volume I	Explains how to use Java EE 6 platform technologies and APIs to develop Java EE applications.
Message Queue Release Notes	Describes new features, compatibility issues, and existing bugs for Sun GlassFish Message Queue.
Message Queue Administration Guide	Explains how to set up and manage a Sun GlassFish Message Queue messaging system.
Message Queue Developer's Guide for JMX Clients	Describes the application programming interface in Sun GlassFish Message Queue for programmatically configuring and monitoring Message Queue resources in conformance with the Java Management Extensions (JMX).
System Virtualization Support in Sun Java System Products	Summarizes Sun support for Sun Java System products when used in conjunction with system virtualization products and features.

Related Documentation

The Java EE 6 Tutorial, Volume II (`https://www.sun.com/offers/details/java_ee6_tutorial.xml`) contains all the topics in *Java EE 6 Tutorial, Volume I* and adds advanced topics, additional technologies, and case studies. The document is available to registered users of Enterprise Server.

Javadoc™ tool reference documentation for packages that are provided with Enterprise Server is available as follows:

- The API specification for version 6 of Java EE is located at `http://java.sun.com/javaee/6/docs/api/`.
- The API specification for Enterprise Server v3, including Java EE 6 platform packages and nonplatform packages that are specific to the Enterprise Server product, is located at: `https://glassfish.dev.java.net/nonav/docs/v3/api/`.

Additionally, the following resources might be useful:

- The Java EE Specifications (`http://java.sun.com/javaee/technologies/index.jsp`)
- The Java EE Blueprints (`http://java.sun.com/reference/blueprints/index.html`)

For information about creating enterprise applications in the NetBeans™ Integrated Development Environment (IDE), see `http://www.netbeans.org/kb/60/index.html`.

For information about the Java DB for use with the Enterprise Server, see
http://developers.sun.com/javadb/.

The sample applications demonstrate a broad range of Java EE technologies. The samples are
bundled with the Java EE Software Development Kit (SDK).

Typographic Conventions

The following table describes the typographic changes that are used in this book.

TABLE P–2   Typographic Conventions

Typeface	Meaning	Example
AaBbCc123	The names of commands, files, and directories, and onscreen computer output	Edit your .login file.
		Use ls -a to list all files.
		machine_name% you have mail.
AaBbCc123	What you type, contrasted with onscreen computer output	machine_name% **su**
		Password:
AaBbCc123	A placeholder to be replaced with a real name or value	The command to remove a file is rm *filename*.
AaBbCc123	Book titles, new terms, and terms to be emphasized (note that some emphasized items appear bold online)	Read Chapter 6 in the *User's Guide*.
		A *cache* is a copy that is stored locally.
		Do *not* save the file.

Symbol Conventions

The following table explains symbols that might be used in this book.

TABLE P–3   Symbol Conventions

Symbol	Description	Example	Meaning
[ ]	Contains optional arguments and command options.	ls [-l]	The -l option is not required.
{ | }	Contains a set of choices for a required command option.	-d {y|n}	The -d option requires that you use either the y argument or the n argument.
${ }	Indicates a variable reference.	${com.sun.javaRoot}	References the value of the com.sun.javaRoot variable.

TABLE P–3 Symbol Conventions *(Continued)*

Symbol	Description	Example	Meaning
-	Joins simultaneous multiple keystrokes.	Control-A	Press the Control key while you press the A key.
+	Joins consecutive multiple keystrokes.	Ctrl+A+N	Press the Control key, release it, and then press the subsequent keys.
→	Indicates menu item selection in a graphical user interface.	File → New → Templates	From the File menu, choose New. From the New submenu, choose Templates.

Default Paths and File Names

The following table describes the default paths and file names that are used in this book.

TABLE P–4 Default Paths and File Names

Placeholder	Description	Default Value
as-install	Represents the base installation directory for Enterprise Server. In configuration files, *as-install* is represented as follows: `${com.sun.aas.installRoot}`	Installations on the Solaris™ operating system, Linux operating system, and Mac operating system: *user's-home-directory*/`glassfishv3/glassfish` Windows, all installations: *SystemDrive*:`\glassfishv3\glassfish`
as-install-parent	Represents the parent of the base installation directory for Enterprise Server.	Installations on the Solaris operating system, Linux operating system, and Mac operating system: *user's-home-directory*/`glassfishv3` Windows, all installations: *SystemDrive*:`\glassfishv3`
domain-root-dir	Represents the directory in which a domain is created by default.	*as-install*/`domains/`
domain-dir	Represents the directory in which a domain's configuration is stored. In configuration files, *domain-dir* is represented as follows: `${com.sun.aas.instanceRoot}`	*domain-root-dir*/*domain-name*

Documentation, Support, and Training

The Sun web site provides information about the following additional resources:

- Documentation (http://www.sun.com/documentation/)
- Support (http://www.sun.com/support/)
- Training (http://www.sun.com/training/)

Searching Sun Product Documentation

Besides searching Sun product documentation from the docs.sun.com^{SM} web site, you can use a search engine by typing the following syntax in the search field:

search-term site:docs.sun.com

For example, to search for "broker," type the following:

broker site:docs.sun.com

To include other Sun web sites in your search (for example, java.sun.com, www.sun.com, and developers.sun.com), use sun.com in place of docs.sun.com in the search field.

Third-Party Web Site References

Third-party URLs are referenced in this document and provide additional, related information.

Note – Sun is not responsible for the availability of third-party web sites mentioned in this document. Sun does not endorse and is not responsible or liable for any content, advertising, products, or other materials that are available on or through such sites or resources. Sun will not be responsible or liable for any actual or alleged damage or loss caused or alleged to be caused by or in connection with use of or reliance on any such content, goods, or services that are available on or through such sites or resources.

Sun Welcomes Your Comments

Sun is interested in improving its documentation and welcomes your comments and suggestions. To share your comments, go to http://docs.sun.com and click Send Comments. In the online form, provide the full document title and part number. The part number is a 7-digit or 9-digit number that can be found on the book's title page or in the document's URL. For example, the part number of this book is 820-4495.

1

Overview of Enterprise Server Administration

Sun GlassFish™ Enterprise Server v3 provides an environment for developing and deploying Java™ applications and web services.

As an Enterprise Server administrator, your main responsibilities are to establish a secure Enterprise Server environment and to oversee the services, resources, and users that participate in that environment. Your key tasks include configuring resources and services, managing Enterprise Server at runtime, and fixing problems that are associated with the server. You might also be involved in installing software, integrating add-on components, and deploying applications.

The following topics are addressed here:

- "Default Settings and Locations" on page 31
- "Configuration Tasks" on page 32
- "Administration Tools" on page 38
- "Instructions for Administering Enterprise Server" on page 43

Default Settings and Locations

After installation, you might need to perform some immediate configuration tasks to make your installation function as intended. If configuration defaults have been accepted, some features are enabled and some not. For an overview of initial configuration tasks for Enterprise Server services and resources, see "Initial Configuration Tasks" on page 33.

In addition, you might want to reset default passwords, change names or locations of files, and so on. The following tables list the default administration values.

Note – For the zip bundle of Enterprise Server v3, the default administrator login is admin, with no password, which means that no login is required.

TABLE 1–1 Default Administration Values

Item	Default
Domain Name	domain1
Master Password	changeit
Administration Password	admin
Administration Server Port	4848
HTTP Port	8080
HTTPS Port	8181
Pure JMX Clients Port	8686
Message Queue Port	7676
IIOP Port	3700
IIOP/SSL Port	3820
IIOP/SSL Port With Mutual Authentication	3920

TABLE 1–2 Default Locations

Item	Default
Command-line Utility (asadmin)	*as-install*/bin
Configuration Files	*domain-dir*/config
Log Files	*domain-dir*/logs
Upgrade Tool (asupgrade Command)	*as-install*/bin
Update Tool and pkg Command	*as-install-parent*/bin

For information on replaceables and default paths and files, see "Default Paths and File Names" on page 29.

Configuration Tasks

Some configuration tasks must be performed directly after installation for your Enterprise Server environment to work as intended. For example, if you are using a database with Enterprise Server, you need to set up database connectivity right away.

Some configuration situations are ongoing and will require you to make changes many times during the life of your installation. You can use either the Administration Console or the asadmin utility to modify the configuration. Changes are automatically applied to the appropriate configuration file.

The following topics are addressed here:

- "Initial Configuration Tasks" on page 33
- "How Dotted Names Work for Configuration" on page 35
- "Configuration Files" on page 36
- "Impact of Configuration Changes" on page 36

Initial Configuration Tasks

This section maps the common configuration tasks to the command–line procedures in this guide. In some situations, the resource or service is automatically enabled and your configuration tasks involve adjusting or changing the default settings to suit your specific needs.

The following resources and services frequently require configuration immediately after installation:

System Properties
See "Administering System Properties" on page 54.

Domains
The initial domain1 is created during installation. Additional configuration tasks might include such tasks as configuring additional domains or setting up automatic restart. See Chapter 3, "Administering Domains."

JVM
The initial tasks for configuring the JVM include creating JVM options and profilers. See Chapter 4, "Administering the Virtual Machine for the Java Platform."

Logging
By default, logging is enabled, so basic logging works without additional configuration. However, you might want to change log levels, property values, or the location of log files. See Chapter 7, "Administering the Logging Service."

Monitoring
By default, the monitoring service is enabled. However, monitoring for the individual modules is not enabled, so your first monitoring task is to enable monitoring for the modules that you want to monitor. See Chapter 8, "Administering the Monitoring Service."

Life Cycle Modules
See Chapter 9, "Administering Life Cycle Modules."

Security

- **System Security.** Initial configuration tasks might include setting up passwords, audit modules, and certificates. See Chapter 11, "Administering System Security."

- **User Security.** Initial configuration tasks might include creating authentication realms and file users. See Chapter 12, "Administering User Security."

- **Message Security.** Initial configuration tasks might include configuring a Java Cryptography Extension (JCE) provider, enabling default and non-default security providers, and configuring message protection policies. See Chapter 13, "Administering Message Security."

Database Connectivity

The initial tasks involved in configuring Enterprise Server to connect to the Java DB database include creating a Java Database Connectivity (JDBC) connection pool, creating a JDBC resource, and integrating a JDBC driver. See Chapter 14, "Administering Database Connectivity ."

EIS Connectivity

The initial tasks involved in configuring Enterprise Server to connect to an enterprise information system (EIS) include creating a connector connection pool, creating a connector resource, editing a resource adapter configuration, creating a connector security map, creating a connector work security map, and creating an administered object (if needed). See Chapter 15, "Administering EIS Connectivity."

Internet Connectivity

The initial tasks involved in making deployed web applications accessible by internet clients include creating HTTP network listeners and virtual servers, and configuring the HTTP listeners for SSL (if needed). See Chapter 16, "Administering Internet Connectivity."

Object Request Broker (ORB)

An initial configuration task might involve creating an IIOP listener. See Chapter 17, "Administering the Object Request Broker (ORB)."

JavaMail Service

An initial configuration task might involve creating a JavaMail resource. See Chapter 18, "Administering the JavaMail Service."

Java Message Service (JMS)

Initial configuration tasks might include creating a physical destination, creating connection factories or destination resources, creating a JMS host (if the default JMS host is not adequate), adjusting connection pool settings (if needed), and configuring resource adapters for JMS. See Chapter 19, "Administering the Java Message Service (JMS) ."

JNDI Service

An initial configuration task might involve creating a JNDI resource. See Chapter 20, "Administering the Java Naming and Directory Interface (JNDI) Service."

Information and instructions for accomplishing the tasks by using the Administration Console are contained in the Administration Console online help.

How Dotted Names Work for Configuration

After the initial configuration is working, you will continue to manage ongoing configuration for the life of your Enterprise Server installation. You might need to adjust resources to improve productivity, or issues might arise that require settings to be modified or defaults to be reset. In some situations, an asadmin subcommand is provided for updating, such as the update-connector-work-security-map subcommand. However, most updating is done by using the list, get, and set subcommands with dotted names. For detailed information about dotted names, see the dotted-names(5ASC) help page.

Note – Dotted names also apply to monitoring, but the method is different. For information on using dotted names for monitoring, see "How the Monitoring Tree Structure Works" on page 124.

The general process for working with configuration changes on the command line is as follows:

1. List the modules for the component of interest.

   The following single mode example uses the | (pipe) character and the grep command to narrow the search:

   asadmin list "*" | grep http | grep listener

   Information similar to the following is returned:

```
configs.config.server-config.network-config.network-listeners.network-listener.http-listener-1
configs.config.server-config.network-config.network-listeners.network-listener.http-listener-2
configs.config.server-config.network-config.protocols.protocol.admin-listener.http
configs.config.server-config.network-config.protocols.protocol.admin-listener.http.file-cache
configs.config.server-config.network-config.protocols.protocol.http-listener-1
configs.config.server-config.network-config.protocols.protocol.http-listener-1.http
configs.config.server-config.network-config.protocols.protocol.http-listener-1.http.file-cache
configs.config.server-config.network-config.protocols.protocol.http-listener-2
configs.config.server-config.network-config.protocols.protocol.http-listener-2.http
configs.config.server-config.network-config.protocols.protocol.http-listener-2.http.file-cache
configs.config.server-config.network-config.protocols.protocol.http-listener-2.ssl
```

2. Get the attributes that apply to the module you are interested in.

   The following multimode example gets the attributes and values for http-listener-1:

```
asadmin> get server-config.network-config.network-listeners.network-listener.http-listener-1.*
```

   Information similar to the following is returned:

```
server.http-service.http-listener.http-listener-1.acceptor-threads = 1
server.http-service.http-listener.http-listener-1.address = 0.0.0.0
server.http-service.http-listener.http-listener-1.blocking-enabled = false
server.http-service.http-listener.http-listener-1.default-virtual-server = server
server.http-service.http-listener.http-listener-1.enabled = true
server.http-service.http-listener.http-listener-1.external-port =
server.http-service.http-listener.http-listener-1.family = inet
server.http-service.http-listener.http-listener-1.id = http-listener-1
server.http-service.http-listener.http-listener-1.port = 8080
server.http-service.http-listener.http-listener-1.redirect-port =
server.http-service.http-listener.http-listener-1.security-enabled = false
server.http-service.http-listener.http-listener-1.server-name =
server.http-service.http-listener.http-listener-1.xpowered-by = true
```

3. Modify an attribute by using the set subcommand.

   This example sets the security-enabled attribute of http-listener-1 to true:

```
asadmin> set server.http-service.http-listener.http-listener-1.security-enabled = true
```

Configuration Files

The bulk of the configuration information about Enterprise Server resources, applications, and
server instances is stored in the domain.xml configuration file. This file is the central repository
for a given administrative domain and contains an XML representation of the Enterprise Server
domain model. Default location for the domain.xml file is
as-install/glassfish3/glassfish/domains/*domain-name*/config. For details on the
domain.xml file, see *Sun GlassFish Enterprise Server v3 Domain File Format Reference*.

The logging.properties file is used to configure logging levels for individual modules. The file
is located in the same directory as the domain.xml file. For further information on the
logging.properties file, see "Setting Log Levels" on page 118.

The asenv.conf file is located in the *as-install*/glassfishv3/glassfish/config. It's purpose is
to store the GlassFish-specific environment variables, such as the installation location of the
database, Message Queue, and so on.

Note – Changes are automatically applied to the appropriate configuration file. Do not edit the
configuration files directly. Manual editing is prone to error and can have unexpected results.

Impact of Configuration Changes

Configuration changes often require that you restart Enterprise Server for the changes to take
effect. In other cases, changes are applied dynamically without requiring that Enterprise Server
be restarted. The procedures in this guide indicate when you need to restart the server.

Configuration Changes That Require Server Restart

When making any of the following configuration changes, you must restart the server for the changes to take effect:

- Changing JVM options

- Changing port numbers

- Changing log handler elements

- Configuring certificates

- Managing HTTP, JMS, IIOP, JNDI services

- Creating or deleting resources (Exception: Some JDBC, JMS, or connector resources do not require restart.)

- Modifying the following JDBC connection pool properties:

  - `datasource-classname`
  - `associate-with-thread`
  - `lazy-connection-association`
  - `lazy-connection-enlistment`
  - JDBC driver vendor-specific properties

- Modifying the following connector connection pool properties:

  - `resource-adapter-name`
  - `connection-definition-name`
  - `transaction-support`
  - `associate-with-thread`
  - `lazy-connection-association`
  - `lazy-connection-enlistment`
  - Vendor-specific properties

Dynamic Configuration Changes

With *dynamic configuration*, changes take effect while the server is running. To make the following configuration changes, you do not need to restart the server:

- Adding or deleting add-on components

- Adding or removing JDBC, JMS, and connector resources and pools (Exception: Some connection pool properties require restart.)

- Adding file realm users

- Changing logging levels

- Enabling and disabling monitoring

- Changing monitoring levels for modules
- Enabling and disabling resources and applications
- Deploying, undeploying, and redeploying applications

Administration Tools

For the most part, you can perform the same tasks by using either the graphical Administration Console or the `asadmin` command-line utility, however, there are exceptions.

The following Enterprise Server administration tools are described here:

- "Administration Console" on page 38
- "asadmin Utility" on page 39
- "REST Interfaces" on page 39
- "Update Tool" on page 40
- "OSGi Module Management Subsystem" on page 40
- "keytool Utility" on page 42
- "Java Monitoring and Management Console (JConsole)" on page 43
- "Application Server Management Extension (AMX)" on page 43

Administration Console

The Administration Console is a browser-based utility that features an easy-to-navigate graphical interface that includes extensive online help for the administrative tasks.

To use the Administration Console, the domain administration server (DAS) must be running. Each domain has its own DAS, which has a unique port number. When Enterprise Server was installed, you chose a port number for the DAS, or used the default port of 4848. You also specified a user name and password if you did not accept the default login (`admin` with no password).

When specifying the URL for the Administration Console, use the port number for the domain to be administered. The format for starting the Administration Console in a web browser is `http://`*hostname*`:`*port*. For example:

`http://kindness.sun.com:4848`

If the Administration Console is running on the host where Enterprise Server was installed, specify `localhost` for the host name. For example:

`http://localhost:4848`

For Microsoft Windows, an alternate way to start the Enterprise Server Administration Console is by using the Start menu.

You can display the help material for a page in the Administration Console by clicking the Help button on the page. The initial help page describes the functions and fields of the page itself. Associated task instructions can be accessed on additional pages by clicking a link in the See Also list.

asadmin **Utility**

The asadmin utility is a command-line tool that runs subcommands for identifying the operation or task that you want to perform. You can run asadmin subcommands either from a command prompt or from a script. Running asadmin subcommands from a script is helpful for automating repetitive tasks. Basic information about how the asadmin utility works can be found in the asadmin(1M) help page. For instructions on using the asadmin utility, see "Using the asadmin Utility" on page 47.

To issue an asadmin subcommand in the standard command shell (single mode), go to the *as-install*/bin directory and type the asadmin command followed by a subcommand. For example:

asadmin list-jdbc-resources

You can invoke multiple command mode (multimode) by typing asadmin at the command prompt, after which the asadmin> prompt is presented. The asadmin utility continues to accept subcommands until you exit multimode and return to the standard command shell. For example:

asadmin> **list-jdbc-resources**

You can display a help page for any asadmin subcommand by typing help before the subcommand name. For example:

asadmin> **help restart-domain**

or

asadmin help restart-domain

A collection of the asadmin help pages is available in HTML and PDF format in the *Sun GlassFish Enterprise Server v3 Reference Manual*.

REST Interfaces

Enterprise Server provides representational state transfer (REST) interfaces to enable you to access monitoring and configuration data for Enterprise Server, including data that is provided by newly installed add-on components. For more information, see "Using REST Interfaces to Administer Enterprise Server" on page 62.

Update Tool

Enterprise Server provides a set of image packaging system (IPS) tools for updating software on a deployed Enterprise Server. Typical updates include new releases of Enterprise Server, and new or revised releases of Enterprise Server add-on components or modules.

- The Update Tool graphical utility can either be run in the Administration Console, or invoked from the command line by using the updatetool command. You can use either tool to add components. However, to update or remove existing components, you must use the standalone version. Instructions for using the graphical versions of the Update Tool are contained in the Administration Console online help and the standalone Update Tool online help.

- The pkg command is the command-line version of Update Tool. Instructions for using the pkg command with add-on components are contained in Chapter 10, "Extending Enterprise Server."

Two distributions are supported for GlassFish Enterprise Server v3: the Web Profile and the Full Platform Profile. After installation, you can view the modules on your system by using the graphical Update Tool or the pkg command.

Note – If you chose the Web Profile, you can change to the Full Platform Profile by selecting the comparable Full Platform Profile package in Update Tool. All dependent modules are automatically added.

You can add and delete individual modules from a distribution, but such configurations are not supported.

If you need information on upgrading your domain configuration data to work with a new version of Enterprise Server, see *Sun GlassFish Enterprise Server v3 Upgrade Guide*.

OSGi Module Management Subsystem

The OSGi module management subsystem that is provided with Enterprise Server is the Apache Felix OSGi framework. To enable you to administer this framework, the Apache Felix Remote Shell (http://felix.apache.org/site/apache-felix-remote-shell.html) is enabled by default in Enterprise Server. This shell uses the Felix shell service to interact with the OSGi module management subsystem, and enables you to perform administrative tasks such as:

- Browsing installed OSGi bundles
- Viewing the headers of installed OSGi bundles
- Installing OSGi bundles
- Controlling the life cycle of installed bundles

The Apache Felix Remote Shell is accessible to telnet clients from anywhere in the network. To connect to the Apache Felix Remote Shell through the telnet service, use the telnet(1) command as follows:

telnet *host felix-remote-shell-port*

host
> The host where the DAS is running.

felix-remote-shell-port
> The port for connecting to the Apache Felix Remote Shell through the telnet service. Enterprise Server is preconfigured to use port 6666 for this purpose.

To see a list of the commands that are available in the Apache Felix Remote Shell, type help at the Apache Felix Remote Shell prompt.

To exit the Apache Felix Remote Shell, type exit at the Apache Felix Remote Shell prompt.

EXAMPLE 1–1 Connecting to the Apache Felix Remote Shell

This example connects to the Apache Felix Remote Shell for a domain that is running on the local host and that uses the preconfigured port for connecting to this shell through the telnet service.

telnet localhost 6666

After the connection is established, the following information is displayed:

```
Connected to localhost.
Escape character is '^]'.

Felix Remote Shell Console:
===========================

->
```

EXAMPLE 1–2 Listing All Installed OSGi Bundles

This example runs the Felix Remote Shell Command ps without any arguments to list all installed OSGi bundles. For better readability, some bundles that would be listed by this example are not shown.

```
-> ps
START LEVEL 1
   ID   State         Level  Name
[   0] [Active      ] [    0] System Bundle (2.0.2)
[   1] [Active      ] [    1] HK2 OSGi Main Bundle (1.0.0)
```

EXAMPLE 1–2 Listing All Installed OSGi Bundles *(Continued)*

```
[    2] [Installed  ] [    1] AMX V3 Core (3.0.0.SNAPSHOT)
[    3] [Active     ] [    1] GlassFish Rest Interface (3.0.0.SNAPSHOT)
...
[ 217] [Installed  ] [    1] Admin Console JDBC Plugin (3.0.0.SNAPSHOT)
[ 218] [Resolved   ] [    1] stats77 (3.0.0.SNAPSHOT)
[ 219] [Active     ] [    1] Apache Felix Declarative Services (1.0.8)
[ 220] [Active     ] [    1] GlassFish Web Container (rfc #66) for OSGi Enabled
Web Applications (3.0.0.SNAPSHOT)
->
```

EXAMPLE 1–3 Finding an OSGi Bundle With a Specified Name

This example runs the Felix Remote Shell Command `find` to find all OSGi bundles whose names contain the text `rfc`.

```
-> find rfc
START LEVEL 1
   ID    State         Level  Name
[ 220] [Active     ] [    1] GlassFish Web Container (rfc #66) for OSGi Enabled
Web Applications (3.0.0.SNAPSHOT)
->
```

EXAMPLE 1–4 To Determine the Services That an OSGi Bundle Provides

This example runs the Felix Remote Shell Command `inspect` with the `service` option and the `capability` option to determine the services that OSGi bundle 220 provides.

```
-> inspect service capability 220
GlassFish Web Container (rfc #66) for OSGi Enabled Web Applications (220) provides services:
--------------------------------------------------------------------------------
objectClass = org.glassfish.osgiweb.Extender
service.id = 30
----
objectClass = org.osgi.service.url.URLStreamHandlerService
service.id = 31
url.handler.protocol = webbundle
->
```

`keytool` **Utility**

The `keytool` utility is used to set up and work with Java Security Socket Extension (JSSE) digital certificates. See "Administering JSSE Certificates " on page 199 for instructions on using `keytool`.

Java Monitoring and Management Console (JConsole)

Java SE provides tools to connect to an MBean server and view the MBeans that are registered with the server. JConsole is one such popular JMX Connector Client and is available as part of the standard Java SE distribution. For instructions on implementing JConsole in the Enterprise Server environment, see "Configuring JConsole to View Enterprise Server Monitoring Data" on page 164.

Application Server Management Extension (AMX)

The application server management eXtension (AMX) API exposes all of the Enterprise Server configuration and monitoring JMX managed beans as easy-to-use client-side dynamic proxies implementing the AMX interfaces.

Instructions for Administering Enterprise Server

Information and instructions on performing most of the administration tasks from the command line are provided in this document and in the asadmin utility help pages. For instructions on accessing asadmin online help, see "To Display Help Information for the asadmin Utility or a Subcommand" on page 50).

Information and instructions for accomplishing the tasks by using the Administration Console are contained in the Administration Console online help.

Note – Instructions written for the Enterprise Server tools use standard UNIX® forward slashes (/) for directory path separators in commands and file names. If you are running Enterprise Server on a Microsoft Windows system, use backslashes (\) instead. For example:

- UNIX: *as-install*/bin/asadmin
- Windows: *as-install*\bin\asadmin

The following additional documents address specific administration areas:

- Installing Enterprise Server software; updating add-on components using the Update Tool

  Sun GlassFish Enterprise Server v3 Installation Guide

- Verifying and deploying applications

  Sun GlassFish Enterprise Server v3 Application Deployment Guide

- Diagnosing and resolving problems

  Sun GlassFish Enterprise Server v3 Troubleshooting Guide

PART I

Runtime Administration

C H A P T E R 2

2

General Administration

This chapter provides instructions for performing general administration tasks in the Sun GlassFish™ Enterprise Server v3 environment by using the `asadmin` command-line utility.

The following topics are addressed here:

- "Using the `asadmin` Utility" on page 47
- "Administering System Properties" on page 54
- "Administering Resources" on page 56
- "Listing Various System Elements" on page 57
- "Using REST Interfaces to Administer Enterprise Server" on page 62

Instructions for accomplishing the tasks in this chapter by using the Administration Console are contained in the Administration Console online help.

Using the `asadmin` Utility

Use the `asadmin` utility to perform administrative tasks for Sun GlassFish Enterprise Server from the command line or from a script. You can use this utility instead of the Administration Console interface.

The following topics are addressed here:

- "Path to the `asadmin` Utility" on page 48
- "`asadmin` Utility Syntax" on page 48
- "To Run an `asadmin` Utility Subcommand in Single Mode" on page 49
- "To Display Help Information for the `asadmin` Utility or a Subcommand" on page 50
- "To Start a Multimode Session" on page 51
- "To End a Multimode Session" on page 53
- "To Run a Set of `asadmin` Subcommands From a File" on page 53

Path to the asadmin **Utility**

The asadmin utility is located in the *as-install*/bin directory. To run the asadmin utility without specifying the path, ensure that this directory is in your path.

asadmin **Utility Syntax**

The syntax for running the asadmin utility is as follows:

asadmin [*asadmin-util-options*] [*subcommand* [*subcommand-options*] [*operands*]]

The replaceable items in this syntax are described in the subsections that follow. For full details of this syntax, see the asadmin(1M) help page.

Subcommands of the asadmin **Utility**

The *subcommand* identifies the operation or task that you are performing. Subcommands are case-sensitive. Each subcommand is either a local subcommand or a remote subcommand.

- A *local subcommand* can be run without a running domain administration server (DAS). However, to run the subcommand and have access to the installation directory and the domain directory, the user must be logged in to the machine that hosts the domain.

- A *remote subcommand* is always run by connecting to a DAS and running the subcommand there. A running DAS is required.

For a list of the subcommands for this release of Enterprise Server, see Section 1 of *Sun GlassFish Enterprise Server v3 Reference Manual*.

asadmin **Utility Options and Subcommand Options**

Options control the behavior of the asadmin utility and its subcommands. Options are case-sensitive.

The asadmin utility has the following types of options:

- asadmin **utility options.** These options control the behavior of the asadmin utility, not the subcommand. The asadmin utility options may precede or follow the subcommand, but asadmin utility options after the subcommand are deprecated. All asadmin utility options must either precede or follow the subcommand. If asadmin utility options are specified both before and after the subcommand, an error occurs. For a description of the asadmin utility options, see the asadmin(1M) help page.

- **Subcommand Options.** These options control the behavior of the subcommand, not the asadmin utility. Subcommand options must follow the subcommand. For a description of a subcommand's options, see the entry for the subcommand in *Sun GlassFish Enterprise Server v3 Reference Manual*.

> **Note** – Not all subcommand options are supported for this release of Enterprise Server. If you specify an unsupported option, a syntax error does not occur. Instead, the command runs successfully and the unsupported option is silently ignored.

A subcommand option may have the same name as an asadmin utility option, but the effects of the two options are different.

Options have a long form and a short form.

- The short form of an option has a single dash (-) followed by a single character.
- The long form of an option has two dashes (- -) followed by an option word.

For example, the short form and the long form of the option for specifying terse output are as follows:

- Short form: -t
- Long form: --terse

Most options require argument values, except Boolean options, which toggle to enable or disable a feature.

Operands of asadmin **Utility Subcommands**

Operands specify the items on which the subcommand is to act. Operands must follow the argument values of subcommand options, and are set off by a space, a tab, or double dashes (- -). The asadmin utility treats anything that follows the subcommand options and their values as an operand.

▼ To Run an asadmin **Utility Subcommand in Single Mode**

In single mode, you must type a separate asadmin command for each subcommand that you want to use. After the subcommand has run, you are returned to the operating system's command shell. Any asadmin utility options must be specified in each separate asadmin command that you run. If you require the same asadmin utility options for multiple subcommands, use the asadmin utility in multimode. For more information, see .

● **In the operating system's command shell, run the** asadmin **utility, specifying the subcommand.**

If necessary, also specify any required asadmin utility options, subcommand options, and operands.

Example 2–1   Running an asadmin Utility Subcommand in Single Mode

This example runs the list-applications(1) subcommand in single mode. In this example, the default values for all options are used.

The example shows that the application hello is deployed on the local host.

```
asadmin list-applications
hello <web>
Command list-applications executed successfully.
```

Example 2–2   Specifying an asadmin Utility Option With a Subcommand in Single Mode

This example specifies the --host asadmin utility option with the list-applications subcommand in single mode. In this example, the DAS is running on the host srvr1.example.com.

The example shows that the applications basic-ezcomp, scrumtoys, ejb31-war, and automatic-timer-ejb are deployed on the host srvr1.example.com.

```
asadmin --host srvr1.example.com list-applications
basic-ezcomp <web>
scrumtoys <web>
ejb31-war <ejb, web>
automatic-timer-ejb <ejb>
Command list-applications executed successfully.
```

Example 2–3   Specifying an asadmin Utility Option and a Subcommand Option in Single Mode

This example specifies the --host asadmin utility option and the --type subcommand option with the list-applications subcommand in single mode. In this example, the DAS is running on the host srvr1.example.com and applications of type web are to be listed.

```
asadmin --host srvr1.example.com list-applications --type web
basic-ezcomp <web>
scrumtoys <web>
ejb31-war <ejb, web>
Command list-applications executed successfully.
```

▼ To Display Help Information for the asadmin Utility or a Subcommand

Enterprise Server provides help information about the syntax, purpose, and options of the asadmin utility and its subcommands. This help information is written in the style of UNIX® platform man pages. This help information is also available in *Sun GlassFish Enterprise Server v3 Reference Manual*.

1 If you are displaying help information for a remote subcommand, ensure that the server is running.

Remote subcommands require a running server.

2 Specify the subcommand of interest as the operand of the help **subcommand.**

If you run the help subcommand without an operand, help information for the asadmin utility is displayed.

Example 2–4 Displaying Help Information for the asadmin Utility

This example displays the help information for the asadmin utility.

asadmin help

Example 2–5 Displaying Help Information for an asadmin Utility Subcommand

This example displays the help information for the create-jdbc-resource subcommand.

asadmin help create-jdbc-resource

See Also To display the available subcommands, use the list-commands(1) subcommand. Local subcommands are displayed before remote subcommands. If the server is not running, only local subcommands are displayed.

▼ To Start a Multimode Session

The asadmin utility can be used in multiple command mode, or *multimode*. In multimode, you run the asadmin utility once to start a multimode session. During the session, the asadmin utility continues to accept subcommands until you end the session and return to the operating system's command shell. Any asadmin utility options that you set for your multimode session are used for all subsequent subcommands in the session.

Note – Starting a multimode session does *not* require a running DAS.

● **Do one of the following:**

■ **Run the** asadmin **utility without a subcommand.**

■ **Use the** multimode**(1) subcommand.**

If necessary, also specify any asadmin utility options that will apply throughout the multimode session.

In a multimode session, the asadmin> prompt is displayed on the command line. You can now type asadmin subcommands at this prompt to administer Enterprise Server.

Example 2–6   Starting a Multimode Session With asadmin Utility Options

This example starts a multimode session in which the asadmin utility options --user and --passwordfile are set for the session.

```
asadmin --user admin1 --passwordfile pwd.txt multimode
```

Example 2–7   Starting a Multimode Session by Using the multimode Subcommand

This example uses the multimode subcommand to start a multimode session in which the default asadmin utility options are used.

```
asadmin multimode
```

The asadmin> prompt is displayed on the command line.

Example 2–8   Running a Subcommand in a Multimode Session

This example starts a multimode session and runs the list-domains subcommand in the session.

```
asadmin
Enter commands one per "line", ^D to quit
asadmin> list-domains
Name: domain1 Status: Running
Command list-domains executed successfully.
asadmin>
```

More Information   Starting a Multimode Session From Within an Existing Multimode Session

You can start a multimode session from within an existing session by running the multimode subcommand from within the existing session. After you end the second multimode session, you return to your original multimode session.

See Also   You can also view the full syntax and options of the subcommand by typing asadmin help multimode at the command line.

▼ To End a Multimode Session

● **At the** asadmin> **prompt, type one of the following commands or key combinations:**

- **exit**
- **quit**
- UNIX and Linux systems: Ctrl-D
- Windows systems: Ctrl-Z

You are returned to the operating system's command shell and the asadmin> prompt is no longer displayed. If the asadmin> prompt is still displayed, you might have opened a multimode session within a multimode session. In this situation, repeat this procedure to end the remaining multimode session.

▼ To Run a Set of asadmin Subcommands From a File

Running a set of asadmin subcommands from a file enables you to automate repetitive tasks.

1 Create a plain text file that contains the sequence of subcommands that you want to run.

2 Run the multimode**(1) subcommand, specifying the file that you created.**

If necessary, also specify any asadmin utility options that are required to enable subcommands in the file to run.

Example 2–9 Running a Set of asadmin Subcommands From a File

This example contains the following:

- A listing of a file that is named commands_file.txt, which contains a sequence of asadmin subcommands
- The command to run the subcommands in the file commands_file.txt

The commands_file.txt file contains the asadmin utility subcommands to perform the following sequence of operations:

1. Creating the domain customdomain
2. Starting the domain customdomain
3. Listing all available subcommands
4. Stopping the domain customdomain
5. Deleting the domain customdomain

The content of the commands_file.txt file is as follows:

```
create-domain --portbase 9000 customdomain
start-domain customdomain
```

```
list-commands
stop-domain customdomain
delete-domain customdomain
```

This example runs the sequence of subcommands in the commands_file.txt file. Because the --portbase option is specified for the create-domain subcommand in the file, the --port asadmin utility option must also be set.

asadmin --port 9048 multimode --file commands_file.txt

See Also   For more information about the subcommands in the preceding example, see the following help pages:

- create-domain(1)
- delete-domain(1)
- list-commands(1)
- multimode(1)
- start-domain(1)
- stop-domain(1)

Administering System Properties

Shared server instances will often need to override attributes defined in their referenced configuration. Any configuration attribute can be overridden through a system property of the corresponding name.

The following topics are addressed here:

- "To Create System Properties" on page 54
- "To List System Properties" on page 55
- "To Delete a System Property" on page 55

▼ To Create System Properties

Use the create-system-properties subcommand in remote mode to create or update one or more system properties of the domain or configuration. Any configuration attribute can be overwritten through a system property of the corresponding name.

1   Ensure that the server is running.

Remote subcommands require a running server.

2   Create system properties by using the create-system-properties(1) **subcommand.**

Information about properties for the subcommand is included in this help page.

Example 2–10 Creating a System Property

This example creates a system property associated with `http-listener-port=1088` on `localhost`.

```
asadmin> create-system-properties http-listener-port=1088
Command create-system-properties executed successfully.
```

See Also You can also view the full syntax and options of the subcommand by typing `asadmin help create-system-properties` at the command line.

▼ To List System Properties

Use the `list-system-properties` subcommand in remote mode to list the system properties that apply to a domain or configuration.

1 Ensure that the server is running.

Remote subcommands require a running server.

2 List system properties by using the `list-system-properties`**(1) subcommand.**

The existing system properties are displayed, including predefined properties such as `HTTP_LISTENER_PORT` and `HTTP_SSL_LISTENER_PORT`.

Example 2–11 Listing System Properties

This example lists the system properties on host `localhost`.

```
asadmin> list-system-properties
http-listener-port=1088
Command list-system-properties executed successfully.
```

See Also You can also view the full syntax and options of the subcommand by typing `asadmin help list-system-properties` at the command line.

▼ To Delete a System Property

Use the `delete-system-property` subcommand in remote mode to delete system properties.

1 Ensure that the server is running.

Remote subcommands require a running server.

2 List the existing system properties by using the `list-system-properties`**(1) subcommand.**

3   Delete the system property by using the `delete-system-property`**(1) subcommand.**

4   If necessary, notify users that the system property has been deleted.

Example 2–12   Deleting a System Property

This example deletes a system property named `http-listener-port` from `localhost`.

```
asadmin> delete-system-property http-listener-port
Command delete-system-property executed successfully.
```

See Also   You can also view the full syntax and options of the subcommand by typing `asadmin help` `delete-system-property` at the command line.

Administering Resources

This section contains instructions for integrating resources into the Enterprise Server environment. Information about administering specific resources, such as JDBC, is contained in other chapters.

The following topics are addressed here:

- "To Add Resources From an XML File" on page 56

▼ To Add Resources From an XML File

Use the `add-resources` subcommand in remote mode to create the resources named in the specified XML file. The following resources are supported: JDBC connection pool and resource, JMS, JNDI, and JavaMail resources, custom resource, connector resource and work security map, admin object, and resource adapter configuration.

The XML file must reside in the *as-install*/`domains/domain1/config` directory. If you specify a relative path or simply provide the name of the XML file, this subcommand will prepend *as-install*/`domains/domain1/config` to this operand.

1   Ensure that the server is running.

Remote subcommands require a running server.

2   Add resources from an XML file by using the `add-resources`**(1) subcommand.**

Information about properties for the subcommand is included in this help page.

3   Restart Enterprise Server.

See "To Restart a Domain" on page 90.

Example 2–13   Adding Resources

This example creates resources using the contents of the resource.xml file on localhost.

```
asadmin> add-resources c:\tmp\resource.xml
Command : JDBC resource jdbc1 created successfully.
Command : JDBC connection pool poolA created successfully.
Command add-resources executed successfully.
```

See Also   You can also view the full syntax and options of the subcommand by typing asadmin help add-resources at the command line.

Listing Various System Elements

- "To Display the Enterprise Server Version" on page 57
- "To List Applications" on page 58
- "To List Containers" on page 58
- "To List Modules" on page 59
- "To List Subcommands" on page 60
- "To List Timers" on page 61
- "To Show Component Status" on page 61

▼ To Display the Enterprise Server Version

Use the version subcommand in remote mode to display information about the Enterprise Server version for a particular server. If the subcommand cannot communicate with the server by using the specified login (user/password) and target (host/port) information, then the local version is displayed along with a warning message.

1   **Ensure that the server is running.**

Remote subcommands require a running server.

2   **Display the version by using the** version**(1)** **subcommand.**

Example 2–14   Displaying Version Information

This example displays the version of Enterprise Server on the local host.

```
asadmin> version
Version = GlassFish v3
Command version executed successfully.
```

You can also view the full syntax and options of the subcommand by typing `asadmin help version` at the command line.

▼ To List Applications

Use the `list-applications` subcommand in remote mode to list the deployed Java™ applications. If the `--type` option is not specified, all applications are listed.

1 Ensure that the server is running.

Remote subcommands require a running server.

2 List applications by using the `list-applications`**(1) subcommand.**

Example 2–15 Listing Applications

This example lists the web applications on `localhost`.

```
asadmin> list-applications --type web
hellojsp <web>
Command list-applications executed successfully.
```

See Also You can also view the full syntax and options of the subcommand by typing `asadmin help list-applications` at the command line.

▼ To List Containers

Use the `list-containers` subcommand in remote mode to list application containers.

1 Ensure that the server is running.

Remote subcommands require a running server.

2 List containers by using the `list-containers`**(1) subcommand.**

Example 2–16 Listing Containers

This example lists the containers on `localhost`.

```
asadmin> list-containers
List all known application containers
Container : grizzly
Container : ejb
```

```
Container : webservices
Container : ear
Container : appclient
Container : connector
Container : jpa
Container : web
Container : jruby
Container : security
Container : webbeans
Command list-containers executed successfully.
```

See Also   You can also view the full syntax and options of the subcommand by typing asadmin help list-containers at the command line.

▼ To List Modules

Use the list-modules subcommand in remote mode to list the modules that are accessible to the Enterprise Server module subsystem. The status of each module is included. Possible statuses include NEW and READY.

1   Ensure that the server is running.

Remote subcommands require a running server.

2   List modules by using the list-modules**(1) subcommand.**

Example 2–17   Listing Modules

This example lists the accessible modules.

```
asadmin> list-modules
```

Information similar to the following is displayed (partial output):

```
List Of Modules
Module : org.glassfish.web.jstl-connector:10.0.0.b28
    properties=(visibility=public,State=READY,Sticky=true)
    Module Characteristics : List of Jars implementing the module
        Jar : file:/C:/Preview/v3_Preview_release/distributions/web/target/glass
fish/modules/web/jstl-connector.jar
    Module Characteristics : List of imported modules
    Module Characteristics : Provides to following services
Module : org.glassfish.admingui.console-common:10.0.0.b28
    properties=(visibility=public,State=NEW,Sticky=true)
Module : org.glassfish.admin.launcher:10.0.0.b28
```

```
        properties=(visibility=public,State=NEW,Sticky=true)
Module : org.glassfish.external.commons-codec-repackaged:10.0.0.b28
        properties=(visibility=public,State=NEW,Sticky=true)
Module : com.sun.enterprise.tiger-types-osgi:0.3.32.Preview-b28
        properties=(visibility=public,State=READY,Sticky=true)
        Module Characteristics : List of imported modules
        Module Characteristics : Provides to following services
        Module Characteristics : List of Jars implementing the module
            Jar : file:/C:/Preview/v3_Preview_release/distributions/web/target/glass
fish/modules/tiger-types-osgi.jar.
...
Command list-modules executed successfully.
```

See Also   You can also view the full syntax and options of the subcommand by typing asadmin help
list-modules at the command line.

▼ To List Subcommands

Use the list-commands subcommand in remote mode to list the deployed asadmin
subcommands. You can specify that only remote subcommands or only local subcommands are
listed. By default, this subcommand displays a list of local subcommands followed by a list of
remote subcommands.

1   Ensure that the server is running.

Remote subcommands require a running server.

2   List subcommands by using the list-commands**(1) subcommand.**

Example 2–18   Listing Subcommands

This example lists only local subcommands.

```
asadmin> list-commands --localonly
create-domain
delete-domain
list-commands
list-domains
login
monitor
start-database
start-domain
stop-domain
stop-database
version
Command list-commands executed successfully.
```

See Also   You can also view the full syntax and options of the subcommand by typing asadmin help list-commands at the command line.

▼ To List Timers

The timer service is a persistent and transactional notification service that is provided by the enterprise bean container and is used to schedule notifications or events used by enterprise beans. All enterprise beans except stateful session beans can receive notifications from the timer service. Persistent timers set by the service are not destroyed when the server is shut down or restarted.

Use the list-timers subcommand in remote mode to list the persistent timers owned by a specific server instance. You can use this information to decide whether to do a timer migration, or to verify that a migration has been completed successfully.

1   Ensure that the server is running.

Remote subcommands require a running server.

2   List timers by using the list-timers**(1) subcommand.**

Example 2–19   Listing Timers

This example lists the timers in a particular standalone server instance. There is one currently active timer set.

```
asadmin> list-timers server
1
The list-timers command was executed successfully.
```

▼ To Show Component Status

Use the show-component-status subcommand in remote mode to get the status (either enabled or disabled) of the specified deployed component.

1   Ensure that the server is running.

Remote subcommands require a running server.

2   Show component status by using the show-component-status**(1) subcommand.**

Example 2–20   Showing Status of a Component

This example shows the status of the MEjbApp component.

```
asadmin> show-component-status MEjbApp
Status of MEjbApp is enabled
Command show-component-status executed successfully.
```

Using REST Interfaces to Administer Enterprise Server

Enterprise Server provides representational state transfer (REST) interfaces to enable you to access monitoring and configuration data for Enterprise Server, including data that is provided by newly installed add-on components.

You can access the Enterprise Server REST interfaces through client applications such as:

- Web browsers
- cURL (http://curl.haxx.se/)
- GNU Wget (http://www.gnu.org/software/wget/)

You can also use the Enterprise Server REST interfaces in REST client applications that are developed in languages such as:

- JavaScript
- Ruby
- Perl
- Java
- JavaFX

The implementation of the Enterprise Server REST interfaces is based on project Jersey (https://jersey.dev.java.net/). Project Jersey is the reference implementation of Java ™ Specification Request (JSR) 311: JAX-RS: The Java API for RESTful Web Services (http://jcp.org/en/jsr/summary?id=311). Information about JSR 311 is also available from the JSR 311 project home page (https://jsr311.dev.java.net/).

The following topics are addressed here:

- "Using REST URLs to Administer Enterprise Server" on page 62
- "Using REST Resource Methods to Administer Enterprise Server" on page 65
- "Child Resources for Non-CRUD Operations" on page 75
- "Securing Enterprise Server REST Interfaces" on page 75
- "Formats for Resource Representation" on page 76

Using REST URLs to Administer Enterprise Server

Each node in the configuration and monitoring object trees is represented as a REST resource that is accessible through an HTTP uniform resource locator (URL). Access to REST resources for Enterprise Server monitoring and configuration data requires a running DAS.

The formats of the URLs to resources that represent nodes in the configuration and monitoring object trees are as follows:

- **Configuration:** `http://host:port/management/domain/path`
- **Monitoring:** `http://host:port/monitoring/domain/path`

The replaceable items in these URLs are as follows:

host
> The host where the DAS is running.

port
> The HTTP port or HTTPS port for administration.

path
> The path to the node. The path is the dotted name of the node in which each dot (.) is replaced with a slash (/). For more information, see the following documentation:
>
> - The `dotted-names(5ASC)` help page
> - "How the Monitoring Tree Structure Works" on page 124
> - "How Dotted Names Work for Configuration" on page 35
> - "Element Hierarchy" in *Sun GlassFish Enterprise Server v3 Domain File Format Reference*

If the URL to a REST resource for Enterprise Server monitoring or configuration data is opened in a web browser, the browser displays a web page that contains the following information about the resource:

- A list of the attributes of the resource and their values. If the resource represents a node in the configuration tree, these attributes are presented in an HTML form that you can use to update the resource. Attributes of a resource for a node in the monitoring tree are read only.
- A list of hypertext links to the children of the resource. This list of links enables you to traverse the tree that contains the resource and to discover the all resources in the tree.

The following figure shows the web page for the REST resource for managing a domain.
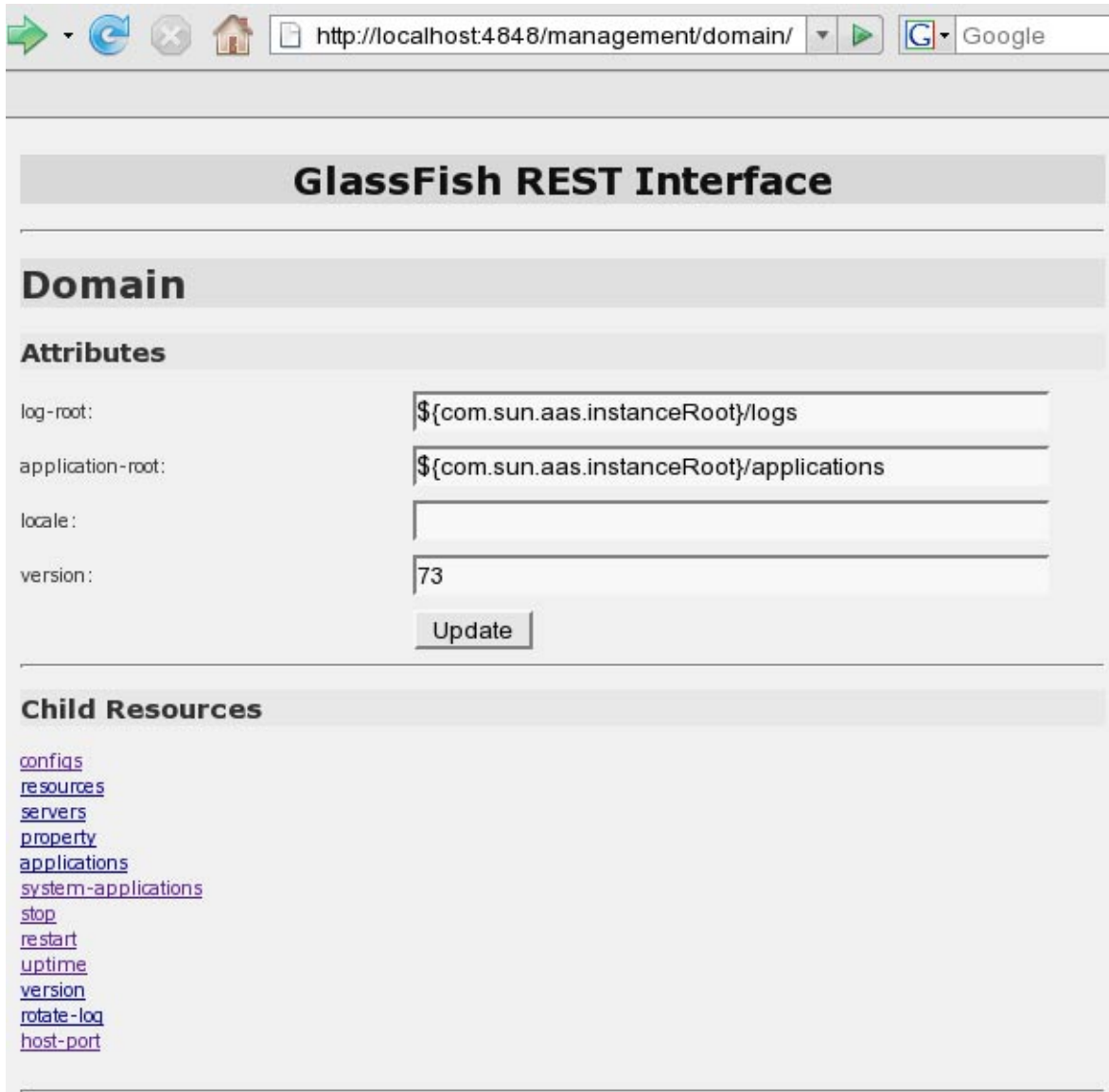
FIGURE 2–1   Web Page for the REST Resource for Managing a Domain

Using REST Resource Methods to Administer Enterprise Server

The Enterprise Server REST interfaces support methods for accessing nodes in the monitoring and configuration object trees.

The following table shows the REST methods for administering monitoring and configuration data and the tasks that you can perform with each method. These methods are HTTP 1.1 primitives. For the detailed specification of these primitives, see Hypertext Transfer Protocol -- HTTP/1.1 (http://www.w3.org/Protocols/rfc2616/rfc2616.html).

TABLE 2–1 REST Resource Methods for Administering Monitoring and Configuration Data

Task	REST Method
Determine the methods and method parameters that a node in the tree supports	OPTIONS or GET
Retrieve data for a node in the tree	GET
Add a node to the tree	POST
Update a node in the tree	POST
Delete a node from the tree	DELETE

Note – The GET method can be used instead of the OPTIONS method to determine the methods and method parameters that a node in the tree supports. The GET method also provides additional information about the node. For details, see "To Retrieve Data for a Node in the Tree" on page 67.

▼ To Determine the Methods and Method Parameters That a Node in the Tree Supports

The methods and method parameters that a node in the tree supports depend on the REST resource that represents the node:

- REST resources for monitoring support only the GET method.
- All REST resources for configuration support the GET method and the OPTIONS method. However, only some REST resources for configuration also support the POST method and the DELETE method.

Before performing any operations on a node in the tree, determine the methods and method parameters that the node supports.

You can specify the format in which this information is presented. For more information, see "Formats for Resource Representation" on page 76.

1 Ensure that the server is running.

Operations on REST resources for Enterprise Server data require a running server.

2 Use the appropriate method on the REST resource that represents the node.

■ **If the node is in the monitoring object tree, use the** GET **method.**

■ **If the node is in the configuration object tree, use the** OPTIONS **method or the** GET **method.**

The GET method and the OPTIONS method return the list of methods that the resource supports. For each method, the list of acceptable message parameters or the list of acceptable query parameters are returned.

Example 2–21 Determining the Methods and Method Parameters That a Node in the Tree Supports

This example uses the cURL utility to determine the methods and method parameters that the resource for the domain supports. The example uses the following options of the cURL utility:

■ -X to specify that the OPTIONS method is used

■ -H to specify that the resource is represented in JavaScript Object Notation (JSON)

In this example, the DAS is running on the local host and the HTTP port for administration is 4848. In addition to the OPTIONS method, the resource supports the POST method and the GET method.

```
curl -X OPTIONS -H "Accept: application/json" http://localhost:4848/management/domain
{"Domain":
  {
    "Method":{
      "Name":"POST",
      "Message Parameters":{
        "log-root":{"Key":"false", "Type":"string", "Optional":"true"},
        "application-root":{"Key":"false", "Type":"string", "Optional":"true"},
        "locale":{"Key":"false", "Type":"string", "Optional":"true"},
        "version":{"Key":"false", "Type":"string", "Optional":"true"}
      }
    },
    "Method":{
      "Name":"GET"
    }
  }
}
```

▼ To Retrieve Data for a Node in the Tree

Retrieving data for a node in the tree obtains the following information about the REST resource that represents the node:

- A list of the REST methods that the resource supports
- A list of the attributes of the resource and their values
- A list of URLs to the children of the resource

You can specify the format in which this information is presented. For more information, see "Formats for Resource Representation" on page 76.

1 Ensure that the server is running.

Operations on REST resources for Enterprise Server data require a running server.

2 Use the GET **method on the REST resource that represents the node.**

Example 2–22 Retrieving Data for a Node in the Tree

This example uses the cURL utility to retrieve data for the resource for a domain. The example uses the following options of the cURL utility:

- -X to specify that the GET method is used
- -H to specify that the resource is represented in JavaScript Object Notation (JSON)

In this example, the DAS is running on the local host and the HTTP port for administration is 4848.

Line breaks are added to enhance readability.

```
curl -X GET -H "Accept: application/json" http://localhost:4848/management/domain
{

  "Domain":{"log-root":"${com.sun.aas.instanceRoot}/logs",
"application-root":"${com.sun.aas.instanceRoot}/applications",
"locale":"", "version":"74.1"},

  "Methods":{
    "Method":{
      "Name":"POST",
      "Message Parameters":{
        "log-root":{"Key":"false", "Type":"string", "Optional":"true"},
        "application-root":{"Key":"false", "Type":"string", "Optional":"true"},
        "locale":{"Key":"false", "Type":"string", "Optional":"true"},
        "version":{"Key":"false", "Type":"string", "Optional":"true"}
      }
    },
```

```
      "Method":{
        "Name":"GET"
      }
    },

    "Child Resources":[
      "http://localhost:4848/management/domain/configs",
      "http://localhost:4848/management/domain/resources",
      "http://localhost:4848/management/domain/servers",
      "http://localhost:4848/management/domain/property",
      "http://localhost:4848/management/domain/applications",
      "http://localhost:4848/management/domain/system-applications",
      "http://localhost:4848/management/domain/stop",
      "http://localhost:4848/management/domain/restart",
      "http://localhost:4848/management/domain/uptime",
      "http://localhost:4848/management/domain/version",
      "http://localhost:4848/management/domain/rotate-log",
      "http://localhost:4848/management/domain/host-port"
    ]
```

▼ To Add a Node to the Tree

1   Ensure that the server is running.

Operations on REST resources for Enterprise Server data require a running server.

2   Determine the acceptable message parameters for the POST method of the resource that represents the parent of the node.

For information about how to perform this step, see "To Determine the Methods and Method Parameters That a Node in the Tree Supports" on page 65.

3   Use the POST method on the REST resource that represents the parent of the node that you are adding.

4   Confirm that the node has been added.

Perform this step on the resource that represents the node that you have just added, *not* the parent. For information about how to perform this step, see "To Retrieve Data for a Node in the Tree" on page 67.

Example 2–23   Adding a Node to the Tree

This example uses the cURL utility to add a JDBC resource node to the tree by creating a REST resource to represent the JDBC resource.

In this example, the DAS is running on the local host and the HTTP port for administration is 4848.

Line breaks are added to enhance readability.

1. This step determines the acceptable message parameters for the POST method of the resource jdbc-resource.

```
curl -X OPTIONS -H "Accept: application/json"
http://localhost:4848/management/domain/resources/jdbc-resource
{"JdbcResource":
  {
    "Method":{
      "Name":"POST",
      "Message Parameters":{
        "id":{"Acceptable Values":"", "Default Value":"", "Type":"string",
            "Optional":"false"},
        "enabled":{"Acceptable Values":"", "Default Value":"true",
            "Type":"boolean", "Optional":"true"},
        "description":{"Acceptable Values":"", "Default Value":"",
            "Type":"string", "Optional":"true"},
        "target":{"Acceptable Values":"", "Default Value":"", "Type":"string",
            "Optional":"true"},
        "property":{"Acceptable Values":"", "Default Value":"",
            "Type":"string", "Optional":"true"},
        "connectionpoolid":{"Acceptable Values":"", "Default Value":"",
            "Type":"string", "Optional":"false"}
      }
    },
    "Method":{
      "Name":"GET"
    }
  }
}
```

2. This step adds a resource as a child of the jdbc-resource resource. The -d option of the cURL utility sets the required message parameters as follows:

- id is set to jdbc/myjdbcresource.
- connectionpoolid is set to DerbyPool.

```
curl -X POST -d "id=jdbc/myjdbcresource&connectionpoolid=DerbyPool"
http://localhost:4848/management/domain/resources/jdbc-resource
"http://localhost:4848/management/domain/resources/jdbc-resource/
jdbc/myjdbcresource" created successfully.
```

3. This step confirms that the node has been added by retrieving data for the REST resource that represents the node.

```
curl -X GET -H "Accept: application/json"
http://localhost:4848/management/domain/resources/
jdbc-resource/jdbc-myjdbcresource
{
```

```
"JdbcMyjdbcresource":{"enabled":"true", "pool-name":"DerbyPool",
    "description":"", "jndi-name":"jdbc/myjdbcresource", "object-type":"user"},

"Methods":{
  "Method":{
    "Name":"POST",
    "Message Parameters":{
      "enabled":{"Key":"false", "Default Value":"true",
          "Type":"boolean", "Optional":"true"},
      "pool-name":{"Key":"false", "Type":"string", "Optional":"true"},
      "description":{"Key":"false", "Type":"string", "Optional":"true"},
      "jndi-name":{"Key":"true", "Type":"string", "Optional":"true"},
      "object-type":{"Key":"false", "Default Value":"user",
          "Type":"string", "Optional":"true"}
    }
  },
  "Method":{
    "Name":"GET"
  },
  "Method":{
    "Name":"DELETE",
    "Message Parameters":{
      "target":{"Acceptable Values":"", "Default Value":"",
          "Type":"string", "Optional":"true"}
    }
  }
}

}
```

▼ To Update a Node in the Tree

1   Ensure that the server is running.

Operations on REST resources for Enterprise Server data require a running server.

2   Determine the acceptable message parameters for the POST **method of the resource that represents the node.**

For information about how to perform this step, see

3   Use the POST **method on the REST resource that represents the node that you are updating.**

4   Confirm that the node has been updated.

For information about how to perform this step, see

Example 2–24   Updating a Node in the Tree

This example uses the cURL utility to update a JDBC resource in the tree by modifying the REST resource that represents the JDBC resource.

In this example, the DAS is running on the local host and the HTTP port for administration is 4848.

Line breaks are added to enhance readability.

1. This step determines the acceptable message parameters for the POST method of the resource jdbc-myjdbcresource.

    ```
    curl -X OPTIONS -H "Accept: application/json"
    http://localhost:4848/management/domain/resources/
    jdbc-resource/jdbc-myjdbcresource
    {"JdbcMyjdbcresource":
      {
        "Method":{
          "Name":"POST",
          "Message Parameters":{
            "enabled":{"Key":"false", "Default Value":"true",
                "Type":"boolean", "Optional":"true"},
            "pool-name":{"Key":"false", "Type":"string", "Optional":"true"},
            "description":{"Key":"false", "Type":"string", "Optional":"true"},
            "jndi-name":{"Key":"true", "Type":"string", "Optional":"true"},
            "object-type":{"Key":"false", "Default Value":"user",
                "Type":"string", "Optional":"true"}
          }
        },
        "Method":{
          "Name":"GET"
        },
        "Method":{
          "Name":"DELETE",
          "Message Parameters":{
            "target":{"Acceptable Values":"", "Default Value":"",
                "Type":"string", "Optional":"true"}
          }
        }
      }
    }
    ```

2. This step updates the REST resource jdbc-myjdbcresource to disable the JDBC resource that jdbc-myjdbcresource represents. The -d option of the cURL utility sets the enabled message parameter to disabled.

    ```
    curl -X POST -d "enabled=false"
    http://localhost:4848/management/domain/resources/
    jdbc-resource/jdbc-myjdbcresource
    ```

```
"http://localhost:4848/management/domain/resources/jdbc-resource/
jdbc-myjdbcresource" updated successfully.
```

3. This step confirms that the node has been updated by retrieving data for the REST resource that represents the node.

```
curl -X GET -H "Accept: application/json"
http://localhost:4848/management/domain/resources/
jdbc-resource/jdbc-myjdbcresource
{

  "JdbcMyjdbcresource":{"enabled":"false", "pool-name":"DerbyPool",
       "description":"", "jndi-name":"jdbc/myjdbcresource", "object-type":"user"},

  "Methods":{
    "Method":{
      "Name":"POST",
      "Message Parameters":{
        "enabled":{"Key":"false", "Default Value":"true",
            "Type":"boolean", "Optional":"true"},
        "pool-name":{"Key":"false", "Type":"string", "Optional":"true"},
        "description":{"Key":"false", "Type":"string", "Optional":"true"},
        "jndi-name":{"Key":"true", "Type":"string", "Optional":"true"},
        "object-type":{"Key":"false", "Default Value":"user",
            "Type":"string", "Optional":"true"}
      }
    },
    "Method":{
      "Name":"GET"
    },
    "Method":{
      "Name":"DELETE",
      "Message Parameters":{
        "target":{"Acceptable Values":"", "Default Value":"",
            "Type":"string", "Optional":"true"}
      }
    }
  }

}
```

▼ To Delete a Node From the Tree

1 Ensure that the server is running.

Operations on REST resources for Enterprise Server data require a running server.

2 Confirm that the node can be deleted.

For information about how to perform this step, see "To Determine the Methods and Method Parameters That a Node in the Tree Supports" on page 65.

3 Confirm that the node has been deleted.

Perform this step on the resource that represents the parent of the node that you have just deleted. For information about how to perform this step, see "To Retrieve Data for a Node in the Tree" on page 67.

Example 2–25 Deleting a Node From the Tree

This example uses the cURL utility to delete a JDBC resource from the tree by deleting the REST resource that represents the JDBC resource.

In this example, the DAS is running on the local host and the HTTP port for administration is 4848.

Line breaks are added to enhance readability.

1. This step confirms that the node can be deleted by retrieving the REST methods that the resource jdbc-myjdbcresource supports.

```
curl -X OPTIONS -H "Accept: application/json"
http://localhost:4848/management/domain/resources/
jdbc-resource/jdbc-myjdbcresource
{"JdbcMyjdbcresource":
  {
    "Method":{
      "Name":"POST",
      "Message Parameters":{
        "enabled":{"Key":"false", "Default Value":"true",
            "Type":"boolean", "Optional":"true"},
        "pool-name":{"Key":"false", "Type":"string", "Optional":"true"},
        "description":{"Key":"false", "Type":"string", "Optional":"true"},
        "jndi-name":{"Key":"true", "Type":"string", "Optional":"true"},
        "object-type":{"Key":"false", "Default Value":"user",
            "Type":"string", "Optional":"true"}
      }
    },
    "Method":{
      "Name":"GET"
    },
    "Method":{
      "Name":"DELETE",
      "Message Parameters":{
        "target":{"Acceptable Values":"", "Default Value":"",
            "Type":"string", "Optional":"true"}
```

```
        }
      }
    }
  }
```

2. This step deletes the jdbc-myjdbcresource resource.

```
curl -X DELETE http://localhost:4848/management/domain/resources/
jdbc-resource/jdbc-myjdbcresource
```

3. This step confirms that the node has been deleted by retrieving data for the REST resource that represents the parent of the node.

```
curl -X GET -H "Accept: application/json"
http://localhost:4848/management/domain/resources/jdbc-resource/
{

  "JdbcResource":{},

  "Methods":{
    "Method":{
      "Name":"POST",
      "Message Parameters":{
        "id":{"Acceptable Values":"", "Default Value":"", "Type":"string",
            "Optional":"false"},
        "enabled":{"Acceptable Values":"", "Default Value":"true",
            "Type":"boolean", "Optional":"true"},
        "description":{"Acceptable Values":"", "Default Value":"",
            "Type":"string", "Optional":"true"},
        "target":{"Acceptable Values":"", "Default Value":"", "Type":"string",
            "Optional":"true"},
        "property":{"Acceptable Values":"", "Default Value":"", "Type":"string",
            "Optional":"true"},
        "connectionpoolid":{"Acceptable Values":"", "Default Value":"",
            "Type":"string", "Optional":"false"}
      }
    },
    "Method":{
      "Name":"GET"
    }
  },

  "Child Resources":[
    "http://localhost:4848/management/domain/resources/jdbc-resource/
        jdbc-__TimerPool",
    "http://localhost:4848/management/domain/resources/jdbc-resource/
        jdbc-__default"
  ]
```

Child Resources for Non-CRUD Operations

The Enterprise Server REST interfaces also support operations other than create, read, update, and delete (CRUD) operations, for example:

- State management
- Queries
- Application deployment

These operations are supported through child resources of the resource on which the operation is performed. The child resources do *not* represent nodes in the configuration object tree.

For example, the resource for managing a domain provides child resources for non-CRUD operations as shown in the following table.

TABLE 2–2   Child Resources for Non-CRUD Operations on a Domain

Resource	Action
host-port	Displays the host on which the DAS is running and the port on which the DAS listens for HTTP requests.
restart	Stops and then restarts the DAS of the domain.
rotate-log	Rotates the server log file by renaming the file with a timestamp name in the format server.log_*date-and-time*, and creating an empty log file.
stop	Stops the DAS of the domain.
uptime	Displays the length of time that the DAS has been running since it was last restarted.
version	Displays version information for Enterprise Server.

Securing Enterprise Server REST Interfaces

The Enterprise Server REST interfaces support basic authentication over a secure connection. When security is enabled, you must specify https as the protocol in the URLs to REST resources and provide a username and password.

Securing Enterprise Server REST Interfaces involves the following sequence of tasks:

1. Adding an admin-realm user to the asadmin user group
2. Enabling Secure Sockets Layer (SSL)

For information about how to perform these tasks from the command line, see the following documentation:

- "To Create an Authentication Realm" on page 206
- "To Create a File User" on page 211

- "To Configure an HTTP Listener for SSL" on page 284

For information about how to perform these tasks by using the Administration Console, see the following topics in the Administration Console online help:

- To Add a User to the Admin Realm
- To Edit SSL Settings for a Protocol

Formats for Resource Representation

The Enterprise Server REST interfaces represent resources in the following formats:

- JSON (http://www.json.org/)
- XML
- HTML

How to specify the resource representation depends on how you are accessing the Enterprise Server REST interfaces. For example, if you are using the cURL utility, specify the resource representation through the -H option as follows:

- For JSON, specify -H "Accept: application/json".
- For XML, specify -H "Accept: application/xml".
- For HTML, omit the -H option.

JSON Resource Representation

The general format for the JSON representation of a resource is as follows:

```
{
    "resource":{attributes},

    "Methods": {
        method-list
    }

    "Child Resources":[urls]
}
```

The replaceable items in this format are as follows:

resource
   The name of the resource.

attributes
   Zero or more name-value pairs separated by a comma (,). Each name-value pair is specified as "*name*":*value*.

method-list
    One or more metadata sets separated by a comma (,) that represent the methods that the
    resource supports. For the format of each metadata set, see "JSON Representation of a
    Method in a Method List" on page 77.

urls
    Zero or more URLs to child resources separated by a comma (,).

JSON Representation of a Method in a Method List

The JSON representation of a method in a method list is as follows:

```
Method":{
    "Name":"method-name",

    "Message Parameters":{
        message-parameter-list
    }

    "Query Parameters":{
        queryparameter- list
    }
}
```

The replaceable items in this format are as follows:

method-name
    The name of the method, which is GET, POST, or DELETE.

message-parameter-list
    Zero or more metadata sets separated by a comma (,) that represent the message parameters
    that are allowed for the method. For the format of each metadata set, see "JSON
    Representation of a Message Parameter or a Query Parameter" on page 77.

query-parameter-list
    Zero or more metadata sets separated by a comma (,) that represent the query parameters
    that are allowed for the method. For the format of each metadata set, see "JSON
    Representation of a Message Parameter or a Query Parameter" on page 77.

JSON Representation of a Message Parameter or a Query Parameter

The JSON representation of a message parameter or a query parameter is as follows:

"*parameter-name*":{*attribute-list*}

The replaceable items in this format are as follows:

parameter-name
    The name of the parameter.

attribute-list

A comma-separated list of name-value pairs of attributes for the parameter. Each pair is in the following format:

"*name*":"*value*"

Possible attributes are as follows:

Default Value

The default value of the parameter.

Acceptable Values

The set or range of acceptable values for the parameter.

Type

The data type of the parameter, which is one of the following types:

- boolean
- int
- string

Optional

Indicates whether the parameter is optional. If true, the parameter is optional. If false, the parameter is required.

Key

Indicates whether the parameter is key. If true, the parameter is key. If false, the parameter is not key.

Example JSON Resource Representation

This example shows the JSON representation of the resource for managing a domain. In this example, the DAS is running on the local host and the HTTP port for administration is 4848. The URL to the resource in this example is http://localhost:4848/management/domain.

Line breaks are added to enhance readability.

```
{

  "Domain":{"log-root":"${com.sun.aas.instanceRoot}/logs",
      "application-root":"${com.sun.aas.instanceRoot}/applications",
      "locale":"", "version":"73"},

  "Methods":{
    "Method":{
      "Name":"POST",
      "Message Parameters":{
        "log-root":{"Key":"false", "Type":"string", "Optional":"true"},
        "application-root":{"Key":"false", "Type":"string", "Optional":"true"},
        "locale":{"Key":"false", "Type":"string", "Optional":"true"},
```

```
      "version":{"Key":"false", "Type":"string", "Optional":"true"}
    }
  },
  "Method":{
    "Name":"GET"
  }
},

"Child Resources":[
  "http://localhost:4848/management/domain/configs",
  "http://localhost:4848/management/domain/resources",
  "http://localhost:4848/management/domain/servers",
  "http://localhost:4848/management/domain/property",
  "http://localhost:4848/management/domain/applications",
  "http://localhost:4848/management/domain/system-applications",
  "http://localhost:4848/management/domain/stop",
  "http://localhost:4848/management/domain/restart",
  "http://localhost:4848/management/domain/uptime",
  "http://localhost:4848/management/domain/version",
  "http://localhost:4848/management/domain/rotate-log",
  "http://localhost:4848/management/domain/host-port"
]

}
```

XML Resource Representation

The general format for the XML representation of a resource is as follows:

<resource attributes>

```
    <Methods>
        method-list
    </Methods>
children
</type>
```

The replaceable items in this format are as follows:

resource
   The name of the resource.

attributes
   Zero or more name-value pairs separated by a space. Each name-value pair is specified as
   name="*value*".

method-list
   One or more XML elements that represent the methods that the resource supports. For the
   format of each element, see "XML Representation of a Resource Method" on page 80.

children
Zero or more XML elements that specify the URLs of child resources. Each element is specified as *<child-resource>url</child-resource>*, where *child-resource* is the name of the child resource and *url* is the URL to the child resource.

XML Representation of a Resource Method

The XML representation of a method in a method list is as follows:

```
<Method name="method-name">
    <Message-Parameters>
        message-parameter-list
    </Message-Parameters>
    <Query-Parameters>
        query-parameter-list
    </Query-Parameters>
</Method>
```

The replaceable items in this format are as follows:

method-name
The name of the method, which is GET, POST, or DELETE.

message-parameter-list
Zero or more XML elements separated by a line feed that represent the message parameters that are allowed for the method. For the format of each element, see "XML Representation of a Message Parameter or a Query Parameter" on page 80.

query-parameter-list
Zero or more XML elements separated by a line feed that represent the query parameters that are allowed for the method. For the format of each element, see "XML Representation of a Message Parameter or a Query Parameter" on page 80.

XML Representation of a Message Parameter or a Query Parameter

The XML representation of a message parameter or a query parameter is as follows:

<parameter-name attribute-list/>

The replaceable items in this format are as follows:

parameter-name
The name of the parameter.

attribute-list
A space-separated list of name-value pairs of attributes for the parameter. Each pair is in the following format:

name="value"

Possible attributes are as follows:

Default Value
  The default value of the parameter.

Acceptable Values
  The set or range of acceptable values for the parameter.

Type
  The data type of the parameter, which is one of the following types:

  - boolean
  - int
  - string

Optional
  Indicates whether the parameter is optional. If true, the parameter is optional. If false, the parameter is required.

Key
  Indicates whether the parameter is key. If true, the parameter is key. If false, the parameter is not key.

Example XML Resource Representation

This example shows the XML representation of the resource for managing a domain. In this example, the DAS is running on the local host and the HTTP port for administration is 4848. The URL to the resource in this example is http://localhost:4848/management/domain.

Line breaks are added to enhance readability.

```
<Domain log-root="${com.sun.aas.instanceRoot}/logs"
application-root="${com.sun.aas.instanceRoot}/applications" locale="" version="73">

  <Methods>
    <Method name="POST">
      <Message-Parameters>
        <log-root Key="false" Type="string" Optional="true"/>
        <application-root Key="false" Type="string" Optional="true"/>
        <locale Key="false" Type="string" Optional="true"/>
        <version Key="false" Type="string" Optional="true"/>
      </Message-Parameters>
    </Method>
    <Method name="GET">
    </Method>
  </Methods>

  <Child-Resources>
    <Child-Resource>http://localhost:4848/management/domain/configs</Child-Resource>
```

```
    <Child-Resource>http://localhost:4848/management/domain/resources</Child-Resource>
    <Child-Resource>http://localhost:4848/management/domain/servers</Child-Resource>
    <Child-Resource>http://localhost:4848/management/domain/property</Child-Resource>
    <Child-Resource>http://localhost:4848/management/domain/applications</Child-Resource>
    <Child-Resource>http://localhost:4848/management/domain/system-applications</Child-Resource>
    <Child-Resource>http://localhost:4848/management/domain/stop</Child-Resource>
    <Child-Resource>http://localhost:4848/management/domain/restart</Child-Resource>
    <Child-Resource>http://localhost:4848/management/domain/uptime</Child-Resource>
    <Child-Resource>http://localhost:4848/management/domain/version</Child-Resource>
    <Child-Resource>http://localhost:4848/management/domain/rotate-log</Child-Resource>
    <Child-Resource>http://localhost:4848/management/domain/host-port</Child-Resource>
  </Child-Resources>

</Domain>
```

HTML Resource Representation

The format for the HTML representation of a resource is a web page that provides the following information about the resource:

- A list of the attributes of the resource and their values.
- A list of the methods and method parameters that the resource supports. Each method and its parameters are presented as a field of the appropriate type in an HTML form.
- A list of hypertext links to the children of the resource.

For a sample web page, see Figure 2–1. In this example, the DAS is running on the local host and the HTTP port for administration is 4848. The URL to the resource in this example is http://localhost:4848/management/domain.

3

Administering Domains

This chapter provides procedures for administering domains in the Sun GlassFish™ Enterprise Server v3 environment by using the `asadmin` command-line utility.

The following topics are addressed here:

- "About Administering Domains (or Servers)" on page 83
- "Creating, Logging In To, and Deleting a Domain" on page 84
- "Starting and Stopping a Domain" on page 88
- "Additional Domain Tasks" on page 93

Instructions for accomplishing the tasks in this chapter by using the Administration Console are contained in the Administration Console online help.

About Administering Domains (or Servers)

A *domain* is a group of instances that are administered together. The domain provides a preconfigured runtime for user applications. In addition to providing an administration boundary, a domain provides the basic security structure whereby separate administrators can administer specific groups of server instances. By grouping the server instances into separate domains, different organizations and administrators can share a single installation of Enterprise Server. A domain has its own configuration, log files, and application deployment areas that are independent of other domains. If the configuration is changed for a domain, the configurations for other domains are not affected.

The Enterprise Server installer creates a default administrative domain named `domain1`, as well as an associated domain administration server (DAS) named `server`. The DAS is a specially-designated instance that authenticates the administrator, accepts requests from administration tools, and communicates with server instances in the domain to carry out requests. The DAS is sometimes referred to as the *default server* because it is the only server instance created during Enterprise Server installation that can be used for deployment.

The default administration port is 4848, but a different port can be specified during installation. When a domain is created, you are prompted for the administration user name and password, but you can accept the default in which case user name is admin and there is no password. To reset the administration password, see "To Change the Administration Password" on page 192.

The graphical Administration Console communicates with a specific DAS to administer the domain associated with the DAS. Each Administration Console session enables you to configure and manage the specific domain. If you create multiple domains, you must start a separate Administration Console session to manage each domain.

Creating, Logging In To, and Deleting a Domain

The following topics are addressed here:

- "To Create a Domain" on page 84
- "To List Domains" on page 85
- "To Log In to a Domain" on page 86
- "To Delete a Domain" on page 87

▼ To Create a Domain

After installing Enterprise Server and creating the default domain (domain1), you can create additional domains by using the local create-domain subcommand. This subcommand creates the configuration of a domain. Any user who has access to the asadmin utility on a given system can create a domain and store the domain configuration in a folder of choice. By default, the domain configuration is created in the default directory for domains. You can override this location to store the configuration elsewhere.

You are required to specify an administrative user when you create a domain, or you can accept the default login identity which is username admin with no password.

Before You Begin   Determine which profile will apply to the domain.

1   Select a name for the domain that you are creating.

You can verify that a name is not already in use by using the list-domains(1) subcommand

2   Create a domain by using the create-domain(1) subcommand.

Information about the options for this subcommand is included in this help page.

3   Type an admin user name and password for the domain.

To avoid setting up an admin login, you can accept the default admin, with no password. Pressing Return also selects the default.

Example 3–1   Creating a Domain

This example creates a domain named domain1 . When you type the command, you might be prompted for login information.

```
asadmin> create-domain --adminport 4848 domain1
Enter admin user name[Enter to accept default]>
Using port 4848 for Admin.
Default port 8080 for HTTP Instance is in use. Using 1161
Using default port 7676 for JMS.
Using default port 3700 for IIOP.
Using default port 8081 for HTTP_SSL.
Using default port 3820 for IIOP_SSL.
Using default port 3920 for IIOP_MUTUALAUTH.
Default port 8686 for JMX_ADMIN is in use. Using 1162
Distinguished Name of the self-signed X.509 Server Certificate is:
[CN=moonbeam.gateway.2wire.net,OU=GlassFish,O=Sun Microsystems,L=Santa Clara,ST
California,C=US]
Domain domain1 created.
Command create-domain executed successfully.
```

To start the Administration Console in a browser, enter the URL in the following format:

```
http://hostname:5000
```

For this example, the domain's log files, configuration files, and deployed applications now reside in the following directory:

domain-root-dir/mydomain

See Also   You can also view the full syntax and options of the subcommand by typing asadmin help create-domain at the command line.

▼ To List Domains

Use the list-domains subcommand to display a list of domains and their statuses. If the domain directory is not specified, the contents of the default *as-install*/domains directory is listed. If there is more than one domain, the domain name must be specified.

To list domains that were created in other directories, specify the --domaindir option.

● **List domains by using the** list-domains**(1) subcommand.**

Example 3–2   Listing Domains

This example lists the domains in the default *as-install*/domains directory:

```
asadmin> list-domains
Name: domain1 Status: Running
Name: domain4 Status: Not Running
Name: domain6 Status: Not Running
Command list-domains executed successfully.
```

See Also   You can also view the full syntax and options of the subcommand by typing asadmin help list-domain at the command line.

▼ To Log In to a Domain

All remote subcommands require that credentials be specified in terms of an administration user name and its password. By default, the domain is created with an identity that allows an asadmin user to perform administrative operations when no identity is explicitly or implicitly specified. The *default identity* is in the form of a user whose name is admin and has no password. If you specify no user name on the command line or on prompt, and specify no password in the --passwordfile option or on prompt, and you have never logged in to a domain using either the login subcommand or the create-domain subcommand with the ----savelogin option, then the asadmin utility will attempt to perform a given administrative operation without specifying any identity. A server (domain) allows administrative operations to be run using this default identity if the following conditions are true:

1. The server (domain) uses file realm for authentication of admin users.

2. The file realm has one and only one user (what the user name is does not matter).

3. That one user has no password.

By default, all of these conditions are true, unless you have created the domain with a specific user name and password. Thus, by default, the only administration user is admin with no password. If #3 is not true, you will need to specify the password. If #2 is not true, you will also need to specify the user name. If #1 is not true, you will need to specify the user name and password.

Use the login subcommand in local mode to authenticate yourself (log in to) a specific domain. After such login, you do not need to specify the administration user or password for subsequent operations on the domain. The login subcommand can only be used to specify the administration password. For other passwords that remote subcommands require, use the --passwordfile option, or specify the password at the command prompt. You are always prompted for the administration user name and password.

There is no logout subcommand. If you want to log in to another domain, invoke asadmin login with new values for --host and --port.

1 Determine the name of the domain that you are logging in to.

To list the existing domains:

asadmin list-domains

2 Log in to the domain by using the login**(1)** command.

Example 3–3 Logging In To a Domain on a Remote Machine

This example logs into a domain located on another machine. Options are specified before the login subcommand.

```
asadmin> --host foo --port 8282 login
Please enter the admin user name>admin Please enter the admin password>
Trying to authenticate for administration of server at host [foo] and port [8282] ...
Login information relevant to admin user name [admin]
for host [foo] and admin port [8282] stored at [/.asadminpass] successfully.
Make sure that this file remains protected. Information stored in this
file will be used by asadmin commands to manage associated domain.
```

Example 3–4 Logging In to a Domain on the Default Port of Localhost

This example logs into a domain on myhost on the default port. Options are specified before the login subcommand.

```
asadmin> --host myhost login
Please enter the admin user name>admin Please enter the admin password>
Trying to authenticate for administration of server at host [myhost] and port [4848] ...
An entry for login exists for host [myhost] and port [4848], probably from
an earlier login operation.
Do you want to overwrite this entry (y/n)?y
Login information relevant to admin user name [admin] for host [myhost]
and admin port [4848] stored at [/home/joe/.asadminpass] successfully.
Make sure that this file remains protected. Information stored in this file will be used by
asadmin commands to manage associated domain.
```

See Also You can also view the full syntax and options of the subcommand by typing asadmin help login at the command line. For additional information about passwords, see "Administering Passwords" on page 190.

▼ To Delete a Domain

Use the delete-domain subcommand to delete an existing domain from a server. Only the root user or the operating system user who is authorized to administer the domain can run this subcommand.

Before You Begin   A domain must be stopped before it can be deleted.

1   **List domains by using the** `list-domains`**(1) subcommand.**

2   **If necessary, notify domain users that the domain is being deleted.**

3   **Ensure that the domain you want to delete is stopped.**
  If needed, see "To Stop a Domain" on page 89.

4   **Delete the domain by using the** `delete-domain`**(1) subcommand.**

Example 3–5   Deleting a Domain

  This example deletes a domain named domain1 from the location specified.

```
asadmin> delete-domain --domaindir ..\domains domain1
Domain domain1 deleted.
Command delete-domain executed successfully.
```

See Also   You can also view the full syntax and options of the subcommand by typing `asadmin help delete-domain` at the command line.

Starting and Stopping a Domain

The following topics are addressed here:

- "To Start a Domain" on page 88
- "To Stop a Domain" on page 89
- "To Restart a Domain" on page 90
- "Automatically Restarting a Domain" on page 91

▼ To Start a Domain

When you start a domain or server, the domain administration server (DAS) is started. After startup, the DAS runs constantly, listening for and accepting requests.

If the domain directory is not specified, the domain in the default *as-install*/domains directory is started. If there are two or more domains, the domain_name operand must be specified. Each domain must be started separately.

If you want the domain you are starting to be able to use the restart-domain subcommand, the --watchdog option must be set to true (true is the default). If you have set the --watchdog option to false, the domain will not be able to use the restart-domain subcommand.

> **Note –** For Microsoft Windows, you can use an alternate method to start a domain. From the Windows Start menu, select Programs → Sun Microsystems →Enterprise Server → Start Admin Server.

This subcommand is supported in local mode only.

● **Start a domain by using the** `start-domain`**(1) subcommand.**

Example 3–6 Starting a Domain

This example starts `domain2` in the default domain directory.

```
asadmin> start-domain domain2
```

If there is only one domain, you can omit the domain name. If you do not include the password, you might be prompted to supply it.

```
Name of the domain started: [domain1] and its location:
[C:\prelude\v3_prelude_release\distributions\web\target\glassfish
domains\domain1].
Admin port for the domain: [4848].
```

See Also You can also view the full syntax and options of the subcommand by typing `asadmin help start-domain` at the command line.

▼ To Stop a Domain

Stopping a domain or server shuts down its domain administration server (DAS). When stopping a domain, the DAS stops accepting new connections and then waits for all outstanding connections to complete. This shutdown process takes a few seconds. While the domain is stopped, the Administration Console and most of the `asadmin` subcommands cannot be used. This subcommand is particularly useful in stopping a runaway server. For more controlled situations, you can use the `restart-domain`(1) subcommand.

> **Note –** For Microsoft Windows, you can use an alternate method to stop a domain. From the Start menu, select Programs → Sun Microsystems → Enterprise Server → Stop Admin Server.

1 **If necessary, notify users that you are going to stop the domain.**

2 **Stop the domain by using the** `stop-domain`**(1) subcommand.**

Example 3–7   Stopping a Domain (or Server)

This example stops domain1 in the default directory, where domain1 is the only domain present in the directory.

```
asadmin> stop-domain
Waiting for the domain to stop ..........
Command stop-domain executed successfully.
```

See Also   You can also view the full syntax and options of the subcommand by typing asadmin help stop-domain at the command line.

▼ To Restart a Domain

Use the restart-domain subcommand in remote mode to restart the Domain Administration Server (DAS) of the specified host. When restarting a domain, the DAS stops accepting new connections and then waits for all outstanding connections to complete. This shutdown process takes a few seconds. Until the domain has restarted, the Administration Console and most of the asadmin subcommands cannot be used.

This subcommand is particularly useful for environments where the server machine is secured and difficult to get to. With the right credentials, you can restart the server from a remote location as well as from the same machine.

If the server will not restart, use the stop-domain(1) subcommand followed by the start-domain(1) subcommand.

Before You Begin   For the restart-domain subcommand to work, the --watchdog option of the start-domain subcommand must have been set to true (the default) when the domain was started. If this option was set to false and you attempt to restart the domain, the domain is stopped and a warning message is logged. If the --watchdog option is set to false, you can only restart the domain by using the stop-domain and start-domain subcommands.

1   **Ensure that the server is running.**

Remote subcommands require a running server.

2   **Restart the domain by using the** restart-domain**(1) subcommand.**

Example 3–8   Restarting a Domain (or Server)

This example restarts mydoimain4 in the default directory.

```
asadmin> restart-domain mydomain4
Waiting for the domain to restart ..........
Command restart-domain executed successfully.
```

Example 3–9   Restarting a Domain in a Browser

This example invokes the restart-domain subcommand in a browser.

```
http://yourhost:4848/__asadmin/restart-domain
```

See Also   You can also view the full syntax and options of the subcommand by typing asadmin help restart-domain at the command line.

Automatically Restarting a Domain

This section provides instructions for configuring your system to automatically restart the domain on Solaris.

The following topics are addressed here:

- "To Automatically Restart a Domain on Solaris 10" on page 91
- "To Restart Automatically on Linux" on page 92
- "To Prevent Service Shutdown When User Logs Out on Windows" on page 93

▼ To Automatically Restart a Domain on Solaris 10

The create-service subcommand is supported on the Solaris and Windows platforms, however, only Solaris instructions are provided here.

On Solaris 10, you can use the asadmin create-service subcommand to create a Solaris Service Management Facility (SMF) service that restarts a Domain Administration Server (DAS). The service grants to the process the privileges of the user that runs the process. When you create an SMF service, the default user is the superuser. If you require a different user to run the process, specify the user in method_credential.

If your process is to bind to a privileged port of Solaris 10, the process requires the net_privaddr privilege. The privileged ports of the Solaris operating system have port numbers less than 1024.

To determine if a user has the net_privaddr privilege, log in as that user and type the command ppriv -l | grep net_privaddr.

After you create and enable the SMF service, if the domain goes down, SMF restarts it.

Before You Begin   To run the asadmin create-service subcommand, you must have solaris.smf.* authorization. See the useradd and usermod man pages to find out how to set the authorizations. You must also have write permission in the directory tree: /var/svc/manifest/application/SUNWappserver. Usually, the superuser has both of these permissions. Additionally, the Solaris 10 administration commands such as svccfg, svcs, and auths must be available in the PATH.

If a particular Enterprise Server domain should not have default user privileges, modify the manifest of the service and reimport the service.

1   **Create the service by using the** `create-service`**(1) subcommand.**

Information about properties for the subcommand is included in this help page.

2   **After the service is created, enable the service by using the** svacdm enable **subcommand.**

For example:

svacdm enable /appserver/domains/domain1

Example 3–10   Creating a Service to Restart a Domain Automatically on Solaris 10

This example creates a service .

```
asadmin> create-service
The Service was created successfully. Here are the details:
Name of the service:application/GlassFish/domain1
Type of the service:Domain
Configuration location of the service:/home/gfuser/glassfish-installations
/glassfishv3/glassfish/domains
Manifest file location on the system:/var/svc/manifest/application
/GlassFish/domain1_home_gfuser_glassfish-installations_glassfishv3
_glassfish_domains/Domain-service-smf.xml.
You have created the service but you need to start it yourself.
Here are the most typical Solaris commands of interest:
* /usr/bin/svcs -a | grep domain1 // status
* /usr/sbin/svcadm enable domain1 // start
* /usr/sbin/svcadm disable domain1 // stop
* /usr/sbin/svccfg delete domain1 // uninstall
Command create-service executed successfully
```

See Also   As you administer your service, the following Solaris commands are useful: auths, smf_security, svcadm, svccfg, rbac, useradd, and usermod.

▼ **To Restart Automatically on Linux**

To set up automatic restart on Linux, you edit the /etc/inittab file. If you use /etc/rc.local, or your system's equivalent, place a line in /etc/rc.local that calls the desired asadmin subcommand.

●   **Add a line of text to the** /etc/inittab **file.**

For example:

```
das:3:respawn:/opt/SUNWappserver/bin/asadmin start-domain --user admin
--passwordfile /opt/SUNWappserver/password.txt domain1
```

The text must be on a single line. The first three letters are a unique designator for the process and can be altered.

▼ To Prevent Service Shutdown When User Logs Out on Windows

By default, the Java Virtual Machine (JVM) receives signals from Windows that indicate that Windows is shutting down, or that a user is logging out of Windows, which causes the system to shut itself down cleanly. This behavior causes the Enterprise Server service to shut down. To prevent the service from shutting down when a user logs out, you must set the -Xrs Java VM option (http://java.sun.com/j2se/1.3/docs/tooldocs/solaris/java.html#Xrs).

1 **Add the following line to the section of the**
as-install\domains*domain-name*\config\domain.xml **file that defines Java VM options:**

```
<jvm-options>-Xrs</jvm-options>
```

2 **If the Enterprise Server service is running, restart the service for your changes to take affect.**

Additional Domain Tasks

The following topics are addressed here:

▼ To Display Domain Uptime

Use the uptime subcommand in remote mode to display the length of time that the domain administration server (DAS) has been running since it was last started.

1 **Ensure that the server is running.**

Remote subcommands require a running server.

2 **Display uptime by using the** uptime**(1) subcommand.**

Example 3–11 Displaying the DAS Uptime

This example displays the length of time that the DAS has been running.

```
asadmin> uptime
Uptime: 1 Weeks, 4 days, 0 hours, 17 minutes, 14 seconds, Total milliseconds: 951434595
Command uptime executed successfully.
```

See Also   You can also view the full syntax and options of the subcommand by typing `asadmin help uptime` at the command line.

▼ To Switch a Domain to Another Supported Java Version

Enterprise Server v3 requires Version 6 Java SE platform as the underlying virtual machine for the Java™ platform (Java Virtual Machine or JVM™ machine).

Note – Do not downgrade to an earlier Java version after a domain has been created with a newer JVM machine. If you must downgrade your JVM machine, downgrade it only for individual domains.

---

1   If you have not already done so, download the desired Java SDK (not the JRE) and install it on your system.

The Java SDK can be downloaded from `http://java.sun.com/j2se`.

2   Start the domain for which you are changing the JDK.

Use the following format:

as-install/bin/asadmin start-domain *domain-name*

For a valid JVM installation, locations are checked in the following order:

a. `domain.xml` **(**`java-home` **inside** `java-config`**)**

b. `asenv.conf` **(setting** `AS_JAVA="path to java home"`**)**

If a legal JDK is not found, a fatal error occurs and the problem is reported back to you.

3   If necessary, change the JVM machine attributes for the domain.

In particular, you might need to change the `JAVA_HOME` environment variable. For example, to change the `JAVA_HOME` variable, type:

as-install/bin/asadmin set "server.java-config.java-home=path-to-java-home"

4

Administering the Virtual Machine for the Java Platform

This chapter provides procedures for administering the Virtual Machine for the Java™ platform (Java Virtual Machine) or JVM™ machine) in the Sun GlassFish™ Enterprise Server v3 environment by using the `asadmin` command-line utility.

The following topics are addressed here:

Instructions for accomplishing these tasks by using the Administration Console are contained in the Administration Console online help.

Administering JVM Options

The Java Virtual Machine is an interpretive computing engine responsible for running the byte codes in a compiled Java program. The virtual machine translates the Java byte codes into the native instructions of the host machine. Enterprise Server, being a Java process, requires a virtual machine to run and support the Java applications running on it. JVM settings are part of an Enterprise Server configuration.

The following topics are addressed here:

▼ To Create JVM Options

Use the create-jvm-options subcommand in remote mode to create JVM options in the Java configuration or the profiler elements of the domain.xml file. If JVM options are created for a profiler, these options are used to record the settings that initiate the profiler.

1   Ensure that the server is running.

Remote subcommands require a running server.

2   Create JVM options by using the create-jvm-options**(1) subcommand.**

To create more than one JVM option, use a colon (:) to separate the options. If the JVM option itself contains a colon (:), use the backslash (\) to offset the colon delimiter.

Information about properties for the subcommand is included in this help page.

3   To apply your changes, restart Enterprise Server. See "To Restart a Domain" on page 90.

Example 4–1   Creating JVM Options

This example sets multiple Java system properties.

```
asadmin> create-jvm-options -Dunixlocation=/root/example:
-Dvariable=\$HOME:
-Dwindowslocation=d\\:\\\sun\\\appserver:
-Doption1=-value1
created 4 option(s)
Command create-jvm-options executed successfully.
```

See Also   You can also view the full syntax and options of the subcommand by typing asadmin help create-jvm-options at the command line.

▼ To List JVM Options

Use the list-jvm-options subcommand in remote mode to list the existing JVM options.

1   Ensure that the server is running.

Remote subcommands require a running server.

2   List JVM options by using the list-jvm-options**(1) subcommand.**

Example 4–2   Listing JVM Options

This example lists all JVM options.

```
asadmin> list-jvm-options
-Djava.security.auth.login.config=${com.sun.aas.instanceRoot}/config/login.conf
-XX: LogVMOutput
-XX: UnlockDiagnosticVMOptions
-Dcom.sun.enterprise.config.config_environment_factory_class=com.sun.enterprise.
config.serverbeans.AppserverConfigEnvironmentFactory
-Djavax.net.ssl.keyStore=${com.sun.aas.instanceRoot}/config/keystore.jks
-XX:NewRatio=2
-Djava.security.policy=${com.sun.aas.instanceRoot}/config/server.policy
-Djdbc.drivers=org.apache.derby.jdbc.ClientDriver
-Djavax.net.ssl.trustStore=${com.sun.aas.instanceRoot}/config/cacerts.jks
-client
-Djava.ext.dirs=${com.sun.aas.javaRoot}/lib/ext${path.separator}${com.sun.aas.ja
vaRoot}/jre/lib/ext${path.separator}${com.sun.aas.instanceRoot}/lib/ext${path.se
parator}${com.sun.aas.derbyRoot}/lib
-Xmx512m
-XX:LogFile=${com.sun.aas.instanceRoot}/logs/jvm.log
-Djava.endorsed.dirs=${com.sun.aas.installRoot}/lib/endorsed
Command list-jvm-options executed successfully.
```

See Also   You can also view the full syntax and options of the subcommand by typing asadmin help
list-jvm-options at the command line.

▼ **To Delete JVM Options**

Use the delete-jvm-options subcommand in remote mode to delete JVM options from the
Java configuration or profiler elements of the domain.xml file.

1   Ensure that the server is running.
Remote subcommands require a running server.

2   List JVM options by using the list-jvm-options**(1) subcommand.**

3   If necessary, notify users that the JVM option is being deleted.

4   Delete JVM options by using the delete-jvm-options**(1) subcommand.**
To remove more than one JVM option, use a colon (:) to separate the options. If the JVM option
itself contains a colon, use the backslash (\) to offset the colon delimiter.

5   To apply your changes, restart Enterprise Server. See "To Restart a Domain" on page 90**.**

Example 4–3   Deleting a JVM Option

This example removes a single JVM option.

```
asadmin> delete-jvm-options -Dopt1=A
deleted 1 option(s)
Command delete-jvm-options executed successfully.
```

Example 4–4   Deleting Multiple JVM Options

This example removes multiple JVM options.

```
asadmin> delete-jvm-options -Doption1=-value1:-Dvariable=\$HOME
deleted 2 option(s)
Command delete-jvm-options executed successfully.
```

See Also   You can also view the full syntax and options of the subcommand by typing asadmin help delete-jvm-options at the command line.

▼ To Generate a JVM Report

Use the generate-jvm-report subcommand in remote mode to generate a JVM report showing the threads (dump of a stack trace), classes, memory, and loggers for a specified domain administration server (DAS). You can generate the following types of reports: summary (default), class, thread, log.

1   **Ensure that the server is running.**

Remote subcommands require a running server.

2   **Generate the report by using the** generate-jvm-report**(1) subcommand.**

Example 4–5   Generating a JVM Report

This example displays summary information about the threads, classes, and memory.

```
asadmin> generate-jvm-report --type summary
Operating System Information:
Name of the Operating System: Windows XP
Binary Architecture name of the Operating System: x86, Version: 5.1
Number of processors available on the Operating System: 2
System load on the available processors for the last minute: NOT_AVAILABLE.
(Sum of running and queued runnable entities per minute).
.
,
.
user.home = C:\Documents and Settings\Jennifer
user.language = en
user.name = Jennifer
```

```
user.timezone = America/New_York
user.variant =
variable = \$HOME
web.home = C:\Preview\v3_Preview_release\distributions\web\target\
glassfish\modules\web
Command generate-jvm-report executed successfully.
```

See Also You can also view the full syntax and options of the subcommand by typing asadmin help
generate-jvm-report at the command line.

Administering the Profiler

A *profiler* generates information used to analyze server performance. The following topics are
addressed here:

- "To Create a Profiler" on page 99
- "To Delete a Profiler" on page 100

▼ To Create a Profiler

A server instance is tied to a particular profiler by the profiler element in the Java configuration.
If JVM options are created for a profiler, the options are used to record the settings needed to
activate a particular profiler. Use the create-profiler subcommand in remote mode to create
the profiler element in the Java configuration.

Only one profiler can exist. If a profiler already exists, you receive an error message that directs
you to delete the existing profiler before creating a new one.

1 Ensure that the server is running.

Remote subcommands require a running server.

2 Create a profiler by using the create-profiler**(1) subcommand.**

Information about properties for the subcommand is included in this help page.

3 To apply your changes, restart Enterprise Server.

See "To Restart a Domain" on page 90.

Example 4–6 Creating a Profiler

This example creates a profiler named sample_profiler.

```
asadmin> create-profiler --classpath=/home/appserver/ --nativelibrarypath=/u/home/lib
--enabled=false --property=defaultuser=admin:password=adminadmin sample_profiler
Command create-profiler executed successfully.
```

See Also You can also view the full syntax and options of the subcommand by typing asadmin help create-profiler at the command line.

▼ To Delete a Profiler

Use the delete-profiler subcommand in remote mode to delete the profiler element from the Java configuration. You can then create a new profiler.

1 Ensure that the server is running.

Remote subcommands require a running server.

2 Delete the profiler by using the delete-profiler**(1) subcommand.**

3 To apply your changes, restart Enterprise Server.

See "To Restart a Domain" on page 90.

Example 4–7 Deleting a Profiler

This example deletes the profiler named sample_profiler.

```
asadmin> delete-profiler sample_profiler
Command delete-profiler executed successfully.
```

See Also You can also view the full syntax and options of the subcommand by typing asadmin help delete-profiler at the command line.

Administering Thread Pools

This chapter provides procedures for administering thread pools in the Sun GlassFish™ Enterprise Server v3 environment by using the asadmin command-line utility.

The following topics are addressed here:

- "About Thread Pools" on page 101
- "Configuring Thread Pools" on page 102

Instructions for accomplishing these tasks by using the Administration Console are contained in the Administration Console online help.

About Thread Pools

The Virtual Machine for the Java™ platform (Java Virtual Machine) or JVM™ machine) can support many threads of execution simultaneously. To help performance, Enterprise Server maintains one or more thread pools. It is possible to assign specific thread pools to connector modules, to network listeners, or to the Object Request Broker (ORB).

One thread pool can serve multiple connector modules and enterprise beans. *Request threads* handle user requests for application components. When Enterprise Server receives a request, it assigns the request to a free thread from the thread pool. The thread executes the client's requests and returns results. For example, if the request needs to use a system resource that is currently busy, the thread waits until that resource is free before allowing the request to use that resource.

Configuring Thread Pools

You can specify the minimum and maximum number of threads that are reserved for requests from applications. The thread pool is dynamically adjusted between these two values.

The following topics are addressed here:

▼ To Create a Thread Pool

Use the create-threadpool subcommand in remote mode to create a thread pool.

The minimum thread pool size that is specified signals the server to allocate at least that many threads in reserve for application requests. That number is increased up to the maximum thread pool size that is specified. Increasing the number of threads available to a process allows the process to respond to more application requests simultaneously.

If one resource adapter or application occupies all the Enterprise Server threads, thread starvation might occur. You can avoid this by dividing the Enterprise Server threads into different thread pools.

1   Ensure that the server is running.

Remote subcommands require a running server.

2   Create a new thread pool by using the create-threadpool**(1) subcommand.**

Information about options for the subcommand is included in this help page.

3   To apply your changes, restart Enterprise Server.

See "To Restart a Domain" on page 90.

Note – Restart is not necessary for thread pools used by the web container.

Example 5–1   Creating a Thread Pool

This example creates threadpool-l.

```
asadmin> create-threadpool --maxthreadpoolsize 100
      --minthreadpoolsize 20 --idletimeout 2 --workqueues 100 threadpool-1
Command create-threadpool executed successfully
```

See Also You can also view the full syntax and options of the subcommand by typing asadmin help create-threadpool at the command line.

▼ **To List Thread Pools**

Use the list-threadpools subcommand in remote mode to list the existing thread pools.

1 Ensure that the server is running.

Remote subcommands require a running server.

2 List the existing thread pools by using the list-threadpools**(1) subcommand.**

Example 5–2 Listing Thread Pools

This example lists the existing thread pools.

```
asadmin> list-threadpools
threadpool-1
Command list-threadpools executed successfully
```

See Also You can also view the full syntax and options of the subcommand by typing asadmin help list-threadpools at the command line.

▼ **To Update a Thread Pool**

Use the set subcommand to update the values for a specified thread pool.

1 List the existing thread pools by using the list-threadpools**(1) subcommand.**

2 Modify the values for a thread pool by using the set**(1) subcommand.**

The thread pool is identified by its dotted name.

3 To apply your changes, restart Enterprise Server.

See "To Restart a Domain" on page 90.

Note – Restart is not necessary for thread pools used by the web container.

Example 5–3 Updating a Thread Pool

This example sets the max-thread-pool-size from its previous value to 8.

```
asadmin> set server.thread-pools.thread-pool.http-thread-pool.max-thread-pool-size=8
Command set executed successfully
```

See Also   You can also view the full syntax and options of the subcommand by typing `asadmin help set` at the command line.

▼ To Delete a Thread Pool

Use the `delete-threadpool` subcommand in remote mode to delete an existing thread pool. Deleting a thread pool will fail if that pool is referenced by a network listener.

1   Ensure that the server is running.

Remote subcommands require a running server.

2   List the existing thread pools by using the `list-threadpools`**(1) subcommand.**

3   Delete the specified thread pool by using the `delete-threadpool`**(1) subcommand.**

4   To apply your changes, restart Enterprise Server.

See "To Restart a Domain" on page 90.

Note – Restart is not necessary for thread pools used by the web container.

Example 5–4   Deleting a Thread Pool

This example deletes `threadpool-1`.

```
asadmin> delete-threadpool threadpool-1
Command delete-threadpool executed successfully
```

See Also   You can also view the full syntax and options of the subcommand by typing `asadmin help` `delete-threadpool` at the command line.

Administering Web Applications

This chapter explains how to administer web applications in the Sun GlassFish™ Enterprise Server v3 environment.

The following topics are addressed here:

- "Invoking a Servlet by Alternate Means" on page 105
- "Changing Log Output for a Servlet" on page 106
- "Defining Global Features for Web Applications " on page 107
- "Redirecting a URL" on page 108
- "Administering `mod_jk`" on page 108

Instructions for accomplishing some of these tasks by using the Administration Console are contained in the Administration Console online help.

Invoking a Servlet by Alternate Means

You can call a servlet deployed to Enterprise Server by using a URL in a browser or embedded as a link in an HTML or JSP file. The format of a servlet invocation URL is as follows:

`http://`*server*`:`*port*`/`*context-root*`/`*servlet-mapping*`?`*name=value*

The following table describes each URL section.

TABLE 6–1   URL Fields for Servlets Within an Application

URL element	Description
server:*port*	The IP address (or host name) and optional port number.
	To access the default web module for a virtual server, specify only this URL section. You do not need to specify the *context-root* or *servlet-name* unless you also wish to specify name-value parameters.

TABLE 6–1   URL Fields for Servlets Within an Application   *(Continued)*

URL element	Description
context-root	For an application, the context root is defined in the context-root element of the application.xml, sun-application.xml, or sun-web.xml file. For an individually deployed web module, the context root is specified during deployment.
	For both applications and individually deployed web modules, the default context root is the name of the WAR file minus the .war suffix.
servlet-mapping	The servlet-mapping as configured in the web.xml file.
?name=value...	Optional request parameters.

EXAMPLE 6–1   Invoking a Servlet With a URL

In this example, localhost is the host name, MortPages is the context root, and calcMortgage is the servlet mapping.

```
http://localhost:8080/MortPages/calcMortgage?rate=8.0&per=360&bal=180000
```

EXAMPLE 6–2   Invoking a Servlet From Within a JSP File

To invoke a servlet from within a JSP file, you can use a relative path. For example:

```
<jsp:forward page="TestServlet"/><jsp:include page="TestServlet"/>
```

Changing Log Output for a Servlet

ServletContext.log messages are sent to the server log. By default, the System.out and System.err output of servlets are sent to the server log. During startup, server log messages are echoed to the System.err output. Also by default, there is no Windows-only console for the System.err output.

You can change these defaults using the Administration Console Write to System Log box. If this box is checked, System.out output is sent to the server log. If it is unchecked, System.out output is sent to the system default location only.

Defining Global Features for Web Applications

You can use the default-web.xml file to define features such as filters and security constraints that apply to all web applications.

For example, directory listings are disabled by default for added security. To enable directory listings in your domain's default-web.xml file, search for the definition of the servlet whose servlet-name is equal to default, and set the value of the init-param named listings to true. Then restart the server.

```
<init-param>
    <param-name>listings</param-name>
    <param-value>true</param-value>
</init-param>
```

If listings is set to true, you can also determine how directory listings are sorted. Set the value of the init-param named sortedBy to NAME, SIZE, or LAST_MODIFIED. Then restart the server.

```
<init-param>
    <param-name>sortedBy</param-name>
    <param-value>LAST_MODIFIED</param-value>
</init-param>
```

The mime-mapping elements in default-web.xml are global and inherited by all web applications. You can override these mappings or define your own using mime-mapping elements in your web application's web.xml file. For more information about mime-mapping elements, see the Servlet specification.

You can use the Administration Console to edit the default-web.xml file, or edit the file directly using the following steps.

▼ To Use the default-web.xml File

1 **Place the JAR file for the filter, security constraint, or other feature in the** *domain-dir*/lib **directory.**

2 **Edit the** *domain-dir*/config/default-web.xml **file to refer to the JAR file.**

3 **To apply your changes, restart Enterprise Server.**
See .

Redirecting a URL

You can specify that a request for an old URL be treated as a request for a new URL. This is called *redirecting* a URL.

To specify a redirected URL for a virtual server, use the redirect_*n* property, where *n* is a positive integer that allows specification of more than one. Each of these redirect_*n* properties is inherited by all web applications deployed on the virtual server.

The value of each redirect_*n* property has two components which can be specified in any order:

- The first component, from, specifies the prefix of the requested URI to match.
- The second component, url-prefix, specifies the new URL prefix to return to the client. The from prefix is replaced by this URL prefix.

EXAMPLE 6–3   Redirecting a URL

This example redirects from dummy to etude:

```
<property name="redirect_1" value="from=/dummy url-prefix=http://etude"/>
```

Administering mod_jk

The Apache Tomcat Connector mod_jk can be used to connect the web container with web servers such as Apache HTTP Server. By using mod_jk, which comes with Enterprise Server, you can front Enterprise Server with Apache HTTP Server.

You can also use mod_jk directly at the JSP/servlet engine for load balancing.

Supported versions of the software referred to in this section include Apache HTTP Server 2.2.11 (UNIX), mod_ssl 2.2.11, OpenSSL 0.9.8a, and mod_jk 1.2.27.

The following topics are addressed here:

▼ To Enable mod_jk

You can front Enterprise Server with Apache HTTP Server by enabling the mod_jk protocol for one of Enterprise Server's network listeners, as described in this procedure. A typical use for

mod_jk would be to have Apache HTTP Server handle requests for static resources, while having requests for dynamic resources, such as servlets and JavaServer™ Pages (JSPs), forwarded to, and handled by the Enterprise Server back-end instance.

When you use the jk-enabled attribute of the network listener, you do not need to copy any additional JAR files into the /lib directory. You can also create JK connectors under different virtual servers by using the network listener attribute jk-enabled.

1 Install Apache HTTP Server and mod_jk.

- For information on installing Apache HTTP Server, see http://httpd.apache.org/docs/2.0/install.html.
- For information on installing mod_jk, see http://tomcat.apache.org/connectors-doc/webserver_howto/apache.html.

2 Configure the following files:

- apache2/conf/httpd.conf, the main Apache configuration file

- apache2/config/workers.properties or *domain-dir*/config/glassfish-jk.properties (to use non-default values of attributes described at http://tomcat.apache.org/tomcat-5.5-doc/config/ajp.html)

If you use both the workers.properties file and the glassfish-jk.properties file, the file referenced by httpd.conf first takes precedence.

3 Start Apache HTTP Server (httpd).

4 Start Enterprise Server with at least one web application deployed.

In order for the mod_jk–enabled network listener to start listening for requests, the web container must be started. Normally, this is achieved by deploying a web application.

5 Create an HTTP listener by using the create-http-listener(1) subcommand.

Use the following format:

```
asadmin> create-http-listener --listenerport 8009
--listeneraddress 0.0.0.0 --defaultvs server listener-name
```

where *listener-name* is the name of the new listener.

6 Enable mod_jk by using the set(1) subcommand.

Use the following format:

```
asadmin> set server-config.network-config.network-listeners.
network-listener.listener-name.jk-enabled=true
```

where *listener-name* is the ID of the network listener for which mod_jk is being enabled.

7 **If you are using the** `glassfish-jk.properties` **file and not referencing it in the** `httpd.conf` **file, point to the properties file by using the** `create-jvm-options`**(1) subcommand.**

Use the following format:

```
asadmin> create-jvm-options -Dcom.sun.enterprise.web.connector.enableJK.propertyFile=
domain-dir/config/glassfish-jk.properties
```

8 **To apply your changes, restart Enterprise Server.**

See "To Restart a Domain" on page 90.

Example 6–4 `httpd.conf` File for `mod_jk`

This example shows an `httpd.conf` file that is set for `mod_jk`.

```
LoadModule jk_module /usr/lib/httpd/modules/mod_jk.so
JkWorkersFile /etc/httpd/conf/worker.properties
# Where to put jk logs
JkLogFile /var/log/httpd/mod_jk.log
# Set the jk log level [debug/error/info]
JkLogLevel debug
# Select the log format
JkLogStampFormat "[%a %b %d %H:%M:%S %Y] "
# JkOptions indicate to send SSL KEY SIZE,
JkOptions +ForwardKeySize +ForwardURICompat -ForwardDirectories
# JkRequestLogFormat set the request format
JkRequestLogFormat "%w %V %T"
# Send all jsp requests to GlassFish
JkMount /*.jsp worker1
# Send all glassfish-test requests to GlassFish
JkMount /glassfish-test/* worker1
```

Example 6–5 `workers.properties` File for `mod_jk`

This example shows a `workers.properties` or `glassfish-jk.properties` file that is set for `mod_jk`.

```
# Define 1 real worker using ajp13
worker.list=worker1
# Set properties for worker1 (ajp13)
worker.worker1.type=ajp13
worker.worker1.host=localhost
worker.worker1.port=8009
```

See Also For more information on Apache, see http://httpd.apache.org/.

For more information on Apache Tomcat Connector, see http://tomcat.apache.org/connectors-doc/index.html.

▼ To Load Balance Using mod_jk and Enterprise Server

Load balancing is the process of dividing the amount of work that a computer has to do between two or more computers so that more work gets done in the same amount of time. Load balancing can be configured with or without security.

In order to support stickiness, the Apache mod_jk load balancer relies on a jvmRoute system property that is included in any JSESSIONID received by the load balancer. This means that every Enterprise Server instance that is front-ended by the Apache load balancer must be configured with a unique jvmRoute system property.

1 On each of the instances, perform the steps in "To Enable mod_jk" on page 108.

If your instances run on the same machine, you must choose different JK ports. The ports must match worker.worker*.port in your workers.properties file. See the properties file in Example 6–5.

2 On each of the instances, create the jvmRoute system property of Enterprise Server by using the create-jvm-options(1) subcommand.

Use the following format:

asadmin> **create-jvm-options "-DjvmRoute=**/*instance-worker-name*"/

where *instance-worker-name* is the name of the worker that you defined to represent the instance in the workers.properties file.

3 To apply your changes, restart Apache HTTP Server and Enterprise Server.

Example 6–6   httpd.conf File for Load Balancing

This example shows an httpd.conf file that is set for load balancing.

```
LoadModule jk_module /usr/lib/httpd/modules/mod_jk.so
JkWorkersFile /etc/httpd/conf/worker.properties
# Where to put jk logs
JkLogFile /var/log/httpd/mod_jk.log
# Set the jk log level [debug/error/info]
JkLogLevel debug
# Select the log format
JkLogStampFormat "[%a %b %d %H:%M:%S %Y] "
# JkOptions indicate to send SSL KEY SIZE,
JkOptions +ForwardKeySize +ForwardURICompat -ForwardDirectories
# JkRequestLogFormat set the request format
JkRequestLogFormat "%w %V %T"
# Send all jsp requests to GlassFish
JkMount /*.jsp worker1
# Send all glassfish-test requests to GlassFish
JkMount /glassfish-test/* loadbalancer
```

Example 6–7   `workers.properties` File for Load Balancing

This example shows a `workers.properties` or `glassfish-jk.properties` file that is set for load balancing. The `worker.worker*.port` should match with JK ports you created.

```
worker.list=worker1,worker2,loadbalancer
worker.worker1.type=ajp13
worker.worker1.host=localhost
worker.worker1.port=8009
worker.worker1.lbfactor=1
worker.worker1.socket_keepalive=1
worker.worker1.socket_timeout=300
worker.worker2.type=ajp13
worker.worker2.host=localhost
worker.worker2.port=8010
worker.worker2.lbfactor=1
worker.worker2.socket_keepalive=1
worker.worker2.socket_timeout=300
worker.loadbalancer.type=lb
worker.loadbalancer.balance_workers=worker1,worker2
```

▼ To Enable SSL Between the `mod_jk` Load Balancer and the Browser

To activate security for `mod_jk` on Enterprise Server, you must first generate a Secure Socket Layer (SSL) self-signed certificate on the Apache HTTP Server with the `mod_ssl` module. The tasks include generating a private key, a Certificate Signing Request (CSR), a self-signed certificate, and configuring SSL-enabled virtual hosts.

Before You Begin   The `mod_jk` connector must be enabled.

1   Generate the private key as follows:

`openssl genrsa -des3 -rand file1:file2:file3:file4:file5 -out server.key 1024`

where `file1:file2:` and so on represents the random compressed files.

2   Remove the pass-phrase from the key as follows:

`openssl rsa -in server.key -out server.pem`

3   Generate the CSR is as follows:

`openssl req -new -key server.pem -out server.csr`

Enter the information you are prompted for.

4 **Generate a temporary certificate as follows:**

```
openssl x509 -req -days 60 -in server.csr -signkey server.pem -out server.crt
```

This temporary certificate is good for 60 days.

5 **Create the** ssl.conf **file under the** /etc/apache2/conf.d **directory.**

6 **In the** ssl.conf **file, add one of the following redirects:**

- Redirect a web application, for example, JkMount /hello/* worker1.
- Redirect all requests, for example, JkMount /* worker1.

```
# Send all jsp requests to GlassFish
JkMount /*.jsp worker1
# Send all glassfish-test requests to GlassFish
JkMount /glassfish-test/* loadbalancer
```

Example 6–8 ssl.conf File for mod_jk Security

A basic SSL-enabled virtual host will appear in the ssl.conf file. In this example, all requests are redirected.

```
Listen 443
<VirtualHost _default_:443>
SSLEngine on
SSLCipherSuite ALL:!ADH:!EXP56:RC4+RSA:+HIGH:+MEDIUM:+LOW:+SSLv2:+EXP:+eNULL
SSLCertificateFile "/etc/apache2/2.2/server.crt"
SSLCertificateKeyFile "/etc/apache2/2.2/server.pem"
JkMount /* worker1
</VirtualHost>
```

▼ To Enable SSL Between the mod_jk Load Balancer and Enterprise Server

Before You Begin The self-signed certificate must be configured.

1 **Perform the steps in**

2 **Start another Enterprise Server with at least one web application deployed.**

In order for the mod_jk–enabled network listener to start listening for requests, the web container must be started. Normally, this is achieved by deploying a web application.

3 Follow instructions from "To Configure an HTTP Listener for SSL" on page 284 on the mod_jk
connector.

Use the following format:

```
asadmin> create-ssl --type http-listener --certname sampleCert new-listener
```

4 Add the following directives in the httpd.conf **file under the** /etc/apache2/conf.d **directory:**

```
# Should mod_jk send SSL information (default is On)
JkExtractSSL On
# What is the indicator for SSL (default is HTTPS)
JkHTTPSIndicator HTTPS
# What is the indicator for SSL session (default is SSL_SESSION_ID)
JkSESSIONIndicator SSL_SESSION_ID
# What is the indicator for client SSL cipher suit (default is SSL_CIPHER)
JkCIPHERIndicator SSL_CIPHER
# What is the indicator for the client SSL certificated? (default is SSL_CLIENT_CERT)
JkCERTSIndicator SSL_CLIENT_CERT
```

5 (Optional) To support both SSL and non-SSL connection, create two mod_jk–**enabled network
listeners and workers.**

Example workers.properties file:

```
#This example shows a worker.properties
worker.list=ajp13unsecure, ajp13secure

worker.ajp13secure.type=ajp13
worker.ajp13secure.host=localhost
worker.ajp13secure.port=8009

worker.ajp13unsecure.type=ajp13
worker.ajp13unsecure.host=localhost
worker.ajp13unsecure.port=8010
```

Secure and unsecure HTTP listeners with ports 8009 and 8010 are created and enabled for
mod_jk. For instructions, see "To Enable mod_jk" on page 108.

6 To apply your changes, restart Apache HTTP Server and Enterprise Server.

◆ ◆ ◆   **C H A P T E R   7**

7

Administering the Logging Service

This chapter provides instructions on how to configure logging and how to view log information in the Sun GlassFish™ Enterprise Server v3 environment.

The following topics are addressed here:

- "About Logging" on page 115
- "Setting Log Levels" on page 118
- "Rotating the Server Log" on page 121
- "Viewing Log Information" on page 121

Instructions for accomplishing these tasks and also for editing properties by using the Administration Console are contained in the Administration Console online help.

About Logging

Logging is the process by which Enterprise Server captures information about events that occur during server operation, such as configuration errors, security failures, or server malfunction. This data is recorded in a log file, and is usually the first source of information when problems occur. Analyzing the log files can help you to determine the health of the server.

Although application components can use the Apache Commons Logging Library to record messages, the platform standard JSR 047 API is recommended for better log configuration.

The following topics are addressed here:

- "Log File " on page 116
- "Logger Namespaces" on page 117

Log File

Enterprise Server log records are captured in the server log. The server log is named
`server.log` by default and is typically located in *domain-dir*/`logs`. You can change the default
name or location of the server log by using the Administration Console.

In addition to the server log, the *domain-dir*/`logs` directory contains the following additional
logs:

- HTTP service access logs, located in the `/access` subdirectory
- Transaction service logs, located in the `/tx` subdirectory

When the server log reaches the specified size in bytes, the log is rotated and renamed with a
timestamp name to `server.log_`*date*, where *date* is the date and time that the file was rotated.
You can also rotate this log manually by following instructions in "Rotating the Server Log" on
page 121.

Enterprise Server log records follow a uniform format:

[#|*yyyy-mm-dd*T*hh:mm:ss.SSS-Z*|*Log Level*|*ProductName-Version*|*LoggerName*|*Key Value Pairs*|*Message*|#]

- [# and #] mark the beginning and end of the record.
- The vertical bar (|) separates the fields of the record.
- *yyyy-mm-ddThh:mm:ss.SSSS-Z* specifies the date and time that the record was created. For
  example: `2006-10-21T13:25:53.852-0400`
- *Log Level* specifies the desired log level. You can select any of the following values: SEVERE,
  WARNING, INFO, CONFIG, FINE, FINER, and FINEST. The default is INFO.
- *ProductName-Version* refers to the current version of the Enterprise Server. For example:
  `glassfish`
- *LoggerName* is a hierarchical logger namespace that identifies the source of the log module.
  For example: `javax.enterprise.system.core`
- *Key Value Pairs* refers to pairs of key names and values, typically a thread ID. For example:
  `_ThreadID=14;`
- *Message* is the text of the log message. For all Enterprise Server SEVERE and WARNING
  messages and for many INFO messages, the message begins with a message ID that consists
  of a module code and a numerical value. For example: `CORE5004`

An example log record might look like this:

```
[#|2006-10-21T13:25:53.852-0400|INFO|GlassFish10.0|javax.enterprise.
system.core|_ThreadID=13;|CORE5004: Resource Deployed:
[cr:jms/DurableConnectionFactory].|#]
```

The Administration Console presents log records in a more readable display.

Logger Namespaces

You can use the list-logger-levels(1) subcommand to list the existing loggers for the modules. Example loggers:

```
javax.enterprise.system.container.cmp: INFO
javax.enterprise.system.tools.admin: INFO
javax.enterprise.system.container.web: INFO
javax.enterprise.system.util: INFO
javax.enterprise.resource.webcontainer.jsf.timing: INFO
javax: INFO
javax.enterprise.resource.corba: INFO
javax.enterprise.system.core.naming: INFO
javax.enterprise.system.core.selfmanagement: INFO
javax.enterprise.system.container.ejb: INFO
javax.enterprise.resource.webcontainer.jsf.config: INFO
javax.enterprise.resource.javamail: INFO
org.apache.catalina: INFO
javax.enterprise.system.core.config: INFO
javax.enterprise.system.webservices.rpc: INFO
javax.enterprise.system.webservices.registry: INFO
javax.enterprise.system.tools.deployment: INFO
javax.enterprise.resource.jms: INFO
javax.enterprise.system: INFO
javax.enterprise.system.webservices.saaj: INFO
org.apache.jasper: INFO
javax.enterprise.resource.webcontainer.jsf.lifecycle: INFO
javax.enterprise.resource.jta: INFO
javax.enterprise.resource.jdo: INFO
javax.enterprise.resource.resourceadapter: INFO
javax.enterprise.system.core.transaction: INFO
javax.enterprise.resource.webcontainer.jsf.resource: INFO
javax.enterprise.system.core.security: INFO
javax.enterprise.resource.webcontainer.jsf.application: INFO
javax.enterprise.system.core.classloading: INFO
org.apache.coyote: INFO
javax.enterprise.resource.webcontainer.jsf.managedbean: INFO
javax.enterprise.system.container.ejb.mdb: INFO
javax.enterprise.resource.webcontainer.jsf.context: INFO
javax.enterprise.resource.webcontainer.jsf.renderkit: INFO
javax.enterprise.resource.webcontainer.jsf.facelets: INFO
javax.enterprise.resource.webcontainer.jsf.taglib: INFO
```

Setting Log Levels

The *log level* determines the granularity of the message that is logged, from error only (SEVERE) to detailed debug (FINEST). The following values apply: SEVERE, WARNING, INFO, CONFIG, FINE, FINER, and FINEST. These log levels are hierarchically inclusive, which means that if you set a particular log level, such as INFO, the messages that have log levels above that level (SEVERE and WARNING) are also included. If you set the log level to the lowest level, FINEST, your output will include all the messages in the file. The default setting is INFO.

There are two levels of log settings available: global and logger-specific. If you have chosen a logger-specific setting that is different from the global setting, the logger-specific setting takes precedence.

Setting the global log level is done by editing the logging.properties file. Logging levels for the individual modules are set by using the asadmin subcommands as explained in this section

Setting Log Levels

Because setting log levels is a dynamic operation, you do not need to restart Enterprise Server for changes to take effect.

The following topics are addressed here:

▼ To List the Logger Levels

Use the list-logger-levels subcommand in remote mode to list the modules and their current log levels.

1 Ensure that the server is running.

Remote subcommands require a running server.

2 List the existing module loggers by using the list-logger-levels**(1) subcommand.**

Example 7–1 Listing Logger Levels for Modules

This example shows a partial list of the existing loggers and indicates how their log levels are set.

```
asadmin> list-logger-levels
javax.enterprise.system.container.cmp: INFO
javax.enterprise.system.tools.admin: INFO
```

```
java.util.logging.ConsoleHandler: FINEST
javax.enterprise.system.container.web: INFO
javax.enterprise.system.util: INFO
javax.enterprise.resource.webcontainer.jsf.timing: INFO
javax: INFO
javax.enterprise.resource.corba: INFO
...
Command list-logger-levels executed successfully.
```

See Also You can also view the full syntax and options of the subcommand by typing asadmin help
list-logger-levels at the command line.

▼ To Set the Global Log Level

The *global log level* specifies which kinds of events are logged across all loggers. The default level
for message output to the console is INFO (which also includes SEVERE and WARNING messages).

You configure global logging by editing the logging.properties file. The default
logging.properties file is located in the same directory as the domain.xml file, typically
domain-dir/config. You can choose a different file name by using the
java.util.logging.config.file system property to specify a file name. For example:

java -Djava.util.logging.config.file=myfile

The ConsoleHandler has a separate log level setting that limits the messages that are displayed.
For example:

```
java.util.logging.ConsoleHandler.level = INFO
java.util.logging.ConsoleHandler.formatter =
com.sun.enterprise.server.logging.UniformLogFormatter
```

1 In a text editor, find the ConsoleHandler **log level line and make your changes.**

2 Save the file.

Example 7–2 Changing the Global Log Level for All Loggers

If you set the log level at the root level, you are setting the level of all loggers. This example sets
the log level for all loggers to INFO:

```
.level= INFO
```

▼ To Set Module Logger Levels

A *module log level* specifies which kinds of events are logged for a particular logger. The default
level for message output to the console is INFO (which also includes SEVERE and WARNING
messages). The global log level is overridden by a module-specific log level.

By default, the module log level is set to FINE. The lines for the loggers might look like this (the modules are indicated in bold):

```
#javax.enterprise.system.tools.level=FINE
#javax.enterprise.system.container.ejb.level=FINE
#javax.enterprise.system.core.security.level=FINE
#javax.enterprise.system.tools.admin.level=FINE
#javax.enterprise.level=FINE
#javax.enterprise.system.container.web.level=FINE
```

Because setting log levels is a dynamic operation, you do not need to restart Enterprise Server for changes to take effect.

1 Ensure that the server is running.

Remote subcommands require a running server.

2 List the existing module loggers by using the list-logger-levels**(1) subcommand.**

3 Set the log level for a module by using the set-log-level**(1) subcommand.**

Your choices are SEVERE, WARNING, INFO, CONFIG, FINE, FINER, and FINEST.

Example 7–3   Setting the Log Level for a Module Logger

This example sets the log level for the web container logger to FINE.

```
asadmin> set-log-level javax.enterprise.system.container.web.level=FINE
Command set-log-level executed successfully.
```

Example 7–4   Setting Log Levels for Multiple Loggers

This example sets the log level for security and web container loggers.

```
asadmin> set-log-level javax.enterprise.system.core.security.level=FINE
javax.enterprise.system.container.web=WARNING
Command set-log-level executed successfully.
```

See Also   You can also view the full syntax and options of the subcommand by typing asadmin help set-log-level at the command line.

Rotating the Server Log

Logs are rotated automatically based on settings in the logging.properties file. You can change these settings by using the Administration Console.

▼ To Rotate a Log File Manually

You can rotate the server log file manually by using the rotate-log subcommand in remote mode. The server log in the default location is immediately moved to a time-stamped file and a new server log is created.

Because log rotation is a dynamic operation, you do not need to restart Enterprise Server for changes to take effect.

1   Ensure that the server is running.

Remote subcommands require a running server.

2   Rotate a log by using the rotate-log(1) **subcommand.**

Example 7–5   Rotating a Log File Manually

This example moves the server.log file to yyyy-mm-dd_server.log and creates a new server.log file in the default location.

```
asadmin> rotate-log
Command rotate-log executed successfuly.
```

See Also   You can also view the full syntax and options of the subcommand by typing asadmin help rotate-log at the command line.

Viewing Log Information

By default, all logging information is captured in the server.log file, typically located in *domain-dir*/logs. You can view logging information by using the Log Viewer in the Administration Console. Instructions for using the Administration Console logging functions are contained in the Administration Console online help.

To view information that has been collected for a module, you can open the server.log file in a text editor and search for the module that you are interested in.

8

CHAPTER 8

Administering the Monitoring Service

This chapter explains how to monitor the Sun GlassFish™ Enterprise Server v3 components and services by using the `asadmin` command-line utility. Instructions for configuring JConsole to monitor Enterprise Server resources are also provided.

The following topics are addressed here:

Instructions for monitoring by using the Administration Console are contained in the Administration Console online help.

For information on using REST interfaces for monitoring, see "Using REST Interfaces to Administer Enterprise Server" on page 62.

About Monitoring

Monitoring is the process of reviewing the statistics of a system to improve performance or solve problems. The monitoring service can track and display operational statistics, such as the number of requests per second, the average response time, and the throughput. By monitoring the state of various components and services deployed in Enterprise Server, you can identify performance bottlenecks, predict failures, perform root cause analysis, and ensure that everything is functioning as expected. Data gathered by monitoring can also be useful in performance tuning and capacity planning.

For this release of Enterprise Server, monitoring is exposed in a modular way so that many client modules can access and display the monitoring statistics. These clients include the Administration Console, the `asadmin` utility, AMX, and REST interfaces.

The following topics are addressed here:

How the Monitoring Tree Structure Works

A *monitorable object* is a component, subcomponent, or service that can be monitored. Enterprise Server uses a tree structure to track monitorable objects. Because the tree is dynamic, the tree changes as Enterprise Server components are added or removed.

In the tree, a monitorable object can have child objects (nodes) that represent exactly what can be monitored for that object. All child objects are addressed using the dot (.) character as a separator. These constructed names are referred to as *dotted names*. Detailed information on dotted names is available in the dotted-names(5ASC) help page.

The following command lists the monitorable child objects of the instance server:

```
asadmin> list --monitor "server.*"

server.applications
server.connector-service
server.http-service
server.jms-service
server.containers.jruby
server.jvm
server.network
server.orb
server.resources
server.security
server.thread-pool
server.transaction-service
server.web
```

Each object is represented by a dotted name. Dotted names can also address specific attributes in monitorable objects. For example, the jvm object has a memory attribute with a statistic called maxheapsize. The following dotted name addresses the attribute:

```
server.jvm.memory.maxheapsize
```

Although an object is monitorable, it is not necessarily being actively monitored. For instructions on activating monitoring, see "Configuring Monitoring" on page 131.

Tree Structure of Monitorable Objects

Each monitorable object has a hierarchical tree structure. In the tree, a replaceable such as *statistics represents the name of the attribute that you can show statistics for.

The following node tree hierarchies are addressed here:

Applications Tree Hierarchy

The applications tree contains the following nodes:

```
server.applications
   |--- application1
   |    |--- ejb-module-1
   |    |        |--- ejb1 *
   |    |                    |--- bean-cache (for entity/sfsb) *
   |    |                    |--- bean-pool (for slsb/mdb/entity) *
   |    |                    |--- bean-methods
   |    |                            |---method1 *
   |    |                            |---method2 *
   |    |                    |--- timers (for s1sb/entity/mdb) *
   |    |--- web-module-1
   |    |        |--- virtual-server-1 *
   |    |                        |---servlet1  *
   |    |                        |---servlet2  *
   |--- standalone-web-module-1
   |    |        |----- virtual-server-2 *
   |    |                        |---servlet3 *
   |    |                        |---servlet4 *
   |    |        |----- virtual-server-3 *
   |    |                        |---servlet3 *(same servlet on different vs)
   |    |                        |---servlet5 *
   |--- standalone-ejb-module-1
```

```
|   |           |--- ejb2 *
|   |                   |--- bean-cache (for entity/sfsb) *
|   |                   |--- bean-pool (for slsb/mdb/entity) *
|   |                   |--- bean-methods
|   |                           |--- method1 *
|   |                           |--- method2 *
|   |                   |--- timers (for s1sb/entity/mdb) *
|--- jersey-application-1
|   |--- jersey
|   |       |--- resources
|                   resource-0
|                       hitcount
|                           *statistic
|--- application2
```

An example dotted name might be `server.applications.hello.server.request.maxtime`.

An example dotted name under the EJB `method` node might be
`server.applications.ejbsfapp1.ejbsfapp1ejbmod1\.jar.SFApp1EJB1`.

An example Jersey dotted name might be
`server.applications.helloworld-webapp.jersey.resources.resource-0.hitcount.resourcehitcount`

For available statistics, see , , and .

Connector Service Tree Hierarchy

The `connector-service` tree holds monitorable attributes for pools such as the connector connection pool. The `connector-service` tree contains the following nodes:

```
server.connector-service
        resource-adapter-1
            connection-pools
                pool-1
            work-management
```

An example dotted name might be
`server.connector-service.resource-adapter-1.connection-pools.pool-1`. For available statistics, see .

HTTP Service Tree Hierarchy

The `http-service` tree contains the following nodes:

```
server.http-service
        virtual-server
            request
```

```
                        *statistic
        _asadmin
            request
                *statistic
```

An example dotted name under the *virutal-server* node might be
server.http-service.virtual-server1.request.requestcount. For available statistics, see
.

JMS/Container Service Tree Hierarchy

The jms-service tree holds monitorable attributes for connection factories (connection pools
for resource adapters) and work management (for Message Queue resource adapters). The
jms-service tree contains the following nodes:

```
server.jms-service
        connection-factories
                connection-factory-1
        work-management
```

An example dotted name under the connection-factories node might be
server.jms-service.connection-factories.connection-factory-1 which shows all the
statistics for this connection factory. For available statistics, see
.

JRuby Tree Hierarchy

The jruby tree contains the following nodes:

```
server.containers.jruby.applications
        jruby-application
                *statistic
                http
                    *statistic
                runtime-pool
                    *statistic
```

For available statistics, see .

JVM Tree Hierarchy

The jvm tree contains the following nodes:

```
server.jvm
        class-loading-system
        compilation-system
        garbage-collectors
```

```
            memory
            operating-system
            runtime
```

An example dotted name under the `memory` node might be `server.jvm.memory.maxheapsize`.
For available statistics, see "JVM Statistics" on page 149.

Network Tree Hierarchy

The network statistics apply to the network listener, such as `admin-listener`,
`http-listener-1`, `ttp-listener-2`. The `network` tree contains the following nodes:

```
server.network
        type-of-listener
            keep-alive
                    *statistic
            file-cache
                    *statistic
            thread-pool
                    *statistic
            connection-queue
                     *statistic
```

An example dotted name under the `network` node might be
`server.network.admin-listener.keep-alive.maxrequests-count`. For available statistics,
see "Network Statistics" on page 153.

ORB Tree Hierarchy

The `orb` tree holds monitorable attributes for connection managers. The `orb` tree contains the
following nodes:

```
server.orb
    transport
        connectioncache
            inbound
                *statistic
            outbound
                *statistic
```

An example dotted name might be
`server.orb.transport.connectioncache.inbound.connectionsidle-count`. For available
statistics, see "ORB Statistics (Connection Manager)" on page 156.

Resources Tree Hierarchy

The `resources` tree holds monitorable attributes for pools such as the JDBC connection pool
and connector connection pool. The `resources` tree contains the following nodes:

```
server.resources
        connection-pool
            request
                *statistic
```

An example dotted name might be
server.resources.jdbc-connection-pool1.numconnfree.count. For available statistics, see
"Resource Statistics (Connection Pool)" on page 157.

Security Tree Hierarchy

The security tree contains the following nodes:

```
server.security
        ejb
            *statistic
        web
            *statistic
        realm
            *statistic
```

An example dotted name might be server.security.realm.realmcount-starttime. For
available statistics, see "Security Statistics" on page 158.

Thread Pool Tree Hierarchy

The thread-pool tree holds monitorable attributes for connection managers, and contains the
following nodes:

```
server.thread-pool
                orb
                    threadpool
                            thread-pool-1
                                *statistic
```

An example dotted name might be
server.thread-pool.orb.threadpool.thread-pool-1.averagetimeinqueue-current. For
available statistics, see "Thread Pool Statistics" on page 159.

Transactions Service Tree Hierarchy

The transaction-service tree holds monitorable attributes for the transaction subsystem for
the purpose of rolling back transactions. The transaction-service tree contains the following
nodes:

```
server.transaction-service
          statistic
```

An example dotted name might be `server.tranaction-service.activeids`. For available statistics, see "Transaction Service Statistics" on page 161.

Web Tree Hierarchy

The `web` tree contains the following nodes:

```
server.web
        jsp
            *statistic
        servlet
            *statistic
        session
            *statistic
        request
            *statistic
```

An example dotted name for the `servlet` node might be `server.web.servlet.activeservletsloadedcount`. For available statistics, see "Web Module Common Statistics" on page 135.

About Monitoring for Add-on Components

An add-on component typically generates statistics that Enterprise Server can gather at runtime. Adding monitoring capabilities enables an add-on component to provide statistics to Enterprise Server in the same way as components that are supplied in the Enterprise Server distributions. As a result, you can use the same administrative interfaces to monitor statistics from any installed Enterprise Server component, regardless of the origin of the component.

Tools for Monitoring Enterprise Server

The following `asadmin` subcommands are provided for monitoring the services and components of Enterprise Server:

- The `enable-monitoring`, `disable-monitoring`, or the `get` and `set` subcommands are used to turn monitoring on or off. For instructions, see "Configuring Monitoring" on page 131.
- The `monitor --type` subcommand is used to display basic data for a particular type of monitorable object. For instructions, see "Viewing Common Monitoring Data" on page 133.
- The `list --monitor` subcommand is used to display the objects that can be monitored with the `monitor` subcommand. For guidelines and instructions, see "Guidelines for Using the `list` and `get` Subcommands for Monitoring" on page 136.

- The get subcommand is used to display comprehensive data, such as the attributes and values for a dotted name. The get subcommand used with a wildcard parameter displays all available attributes for any monitorable object. For additional information, see "Guidelines for Using the list and get Subcommands for Monitoring" on page 136.

Configuring Monitoring

By default, the monitoring service is enabled for Enterprise Server, but monitoring for the individual modules is not. To enable monitoring for a module, you change the monitoring level for that module to LOW or HIGH, You can choose to leave monitoring OFF for objects that do not need to be monitored.

- **LOW.** Simple statistics, such as create count, byte count, and so on
- **HIGH.** Simple statistics plus method statistics, such as method count, duration, and so on
- **OFF.** No monitoring, no impact on performance

The following tasks are addressed here:

▼ To Enable Monitoring

Use the enable-monitoring subcommand to enable the monitoring service itself, or to enable monitoring for individual modules. Monitoring is immediately activated, without restarting Enterprise Server.

You can also use the set(1) subcommand to enable monitoring for a module. Using the set command is not a dynamic procedure, so you need to restart Enterprise Server for your changes to take effect.

1 Determine which services and components are currently enabled for monitoring.

```
asadmin> get server.monitoring-service.module-monitoring-levels.*
```

This example output shows that the HTTP service is not enabled (OFF for monitoring), but other objects are enabled:

```
configs.config.server-config.monitoring-service.module-monitoring-levels.web-container=HIGH
       configs.config.server-config.monitoring-service.module-monitoring-levels.http-service=OFF
          configs.config.server-config.monitoring-service.module-monitoring-levels.jvm=HIGH
```

2 Enable monitoring by using the enable-monitoring(1) subcommand.

Server restart is not required.

Example 8–1 Enabling the Monitoring Service Dynamically

This example enables the monitoring service without affecting monitoring for individual modules.

```
asadmin> enable-monitoring
Command enable-monitoring executed successfully
```

Example 8–2 Enabling Monitoring for Modules Dynamically

This example enables monitoring for the ejb-container module.

```
asadmin> enable-monitoring --level ejb-container=HIGH
Command enable-monitoring executed successfully
```

Example 8–3 Enabling Monitoring for Modules by Using the set Subcommand

This example enables monitoring for the HTTP service by setting the monitoring level to HIGH (you must restart the server for changes to take effect).

```
asadmin> set server.monitoring-service.module-monitoring-levels.http-service=HIGH
Command set executed successfully
```

See Also You can also view the full syntax and options of the subcommand by typing asadmin help enable-monitoring at the command line.

▼ To Disable Monitoring

Use the disable-monitoring subcommand to disable the monitoring service itself, or to disable monitoring for individual modules. Monitoring is immediately stopped, without restarting Enterprise Server.

You can also use the set(1) subcommand to disable monitoring for a module. Using the set command is not a dynamic procedure, so you need to restart Enterprise Server for your changes to take effect.

1 Determine which services and components currently are enabled for monitoring.

asadmin get server.monitoring-service.module-monitoring-levels.*

This example output shows that monitoring is enabled for web-container, http-service, and jvm:

```
configs.config.server-config.monitoring-service.module-monitoring-levels.web-container=HIGH
     configs.config.server-config.monitoring-service.module-monitoring-levels.http-service=HIGH
          configs.config.server-config.monitoring-service.module-monitoring-levels.jvm=HIGH
```

2 Disable monitoring for a service or module by using the `disable-monitoring`**(1) subcommand.**

Server restart is not required

Example 8–4 Disabling the Monitoring Service Dynamically

This example disables the monitoring service without changing the monitoring levels for individual modules.

```
asadmin> disable-monitoring
Command disable-monitoring executed successfully
```

Example 8–5 Disabling Monitoring for Modules Dynamically

This example disables monitoring for specific modules. Their monitoring levels are set to OFF.

```
asadmin> disable-monitoring --modules web-container,ejb-container
Command disable-monitoring executed successfully
```

Example 8–6 Disabling Monitoring by Using the `set` Subcommand

This example disables monitoring for the HTTP service (you must restart the server for changes to take affect).

```
asadmin> set server.monitoring-service.module-monitoring-levels.http-service=OFF
Command set executed successfully
```

See Also You can also view the full syntax and options of the subcommand by typing `asadmin help disable-monitoring` at the command line.

Viewing Common Monitoring Data

Use the `monitor` subcommand to display basic data on commonly-monitored objects.

▼ To View Common Monitoring Data

Use the `--type` option of the `monitor` subcommand to specify the object for which you want to display data, such as `httplistener`, `jvm`, `webmodule`. If you use the `monitor` subcommand without specifying a type, an error message is displayed.

Output from the subcommand is displayed continuously in a tabular format. The `--interval` option can be used to display output at a particular interval (the default is 30 seconds).

Before You Begin A monitorable object must be configured for monitoring before you can display data on the object. See "To Enable Monitoring " on page 131.

1 **Determine which type of monitorable object you want to monitor.**

Your choices for v3 are `jvm`, `httplistener`, and `webmodule`.

2 **Request the monitoring data by using the** `monitor(1)` **subcommand.**

Example 8–7 Viewing Common Monitoring Data

This example requests common data for type `jvm` on instance `server`.

```
asadmin> monitor --type jvm server

UpTime(ms)                      Heap and NonHeap Memory(bytes)
current               min       max       low     high     count

9437266               8585216   619642880 0       0        93093888
9467250               8585216   619642880 0       0        93093888
```

See Also You can also view the full syntax and options of the subcommand by typing `asadmin help monitor` at the command line.

Common Monitoring Statistics

Common monitoring statistics are described in the following sections:

- "HTTP Listener Common Statistics" on page 134
- "JVM Common Statistics" on page 135
- "Web Module Common Statistics" on page 135

HTTP Listener Common Statistics

The statistics available for the `httplistener` type are shown in the following table.

TABLE 8–1 HTTP Listener Common Monitoring Statistics

Statistic	Description
ec	Error count. Cumulative value of the error count
mt	Maximum time. Longest response time for a request; not a cumulative value, but the largest response time from among the response times
pt	Processing time. Cumulative value of the times taken to process each request, with processing time being the average of request processing times over request

TABLE 8–1 HTTP Listener Common Monitoring Statistics *(Continued)*

Statistic	Description
rc	Request count. Cumulative number of requests processed so far

JVM Common Statistics

The statistics available for the jvm type are shown in the following table.

TABLE 8–2 JVM Common Monitoring Statistics

Statistic	Description
count	Amount of memory (in bytes) that is guaranteed to be available for use by the JVM machine
high	Retained for compatibility with other releases
low	Retained for compatibility with other releases
max	The maximum amount of memory that can be used for memory management.
min	Initial amount of memory (in bytes) that the JVM machine requests from the operating system for memory management during startup
UpTime	Number of milliseconds that the JVM machine has been running since it was last started

Web Module Common Statistics

The statistics available for the webmodule type are shown in the following table.

TABLE 8–3 Web Module Common Monitoring Statistics

Statistic	Description
ajlc	Number of active JavaServer Pages™ (JSP™) technology pages that are loaded
asc	Current active sessions
aslc	Number of active servlets that are loaded
ast	Total active sessions
mjlc	Maximum number of JSP pages that are loaded
mslc	Maximum number of servlets that are loaded
rst	Total rejected sessions
st	Total sessions
tjlc	Total number of JSP pages that are loaded
tslc	Total number of servlets that are loaded

Viewing Comprehensive Monitoring Data

By applying the list and get subcommands against the tree structure using dotted names, you can display more comprehensive monitoring data, such as a description of each of the statistics and its unit of measurement.

The following topics are addressed here:

- "Guidelines for Using the list and get Subcommands for Monitoring" on page 136
- "To View Comprehensive Monitoring Data" on page 137
- "Comprehensive Monitoring Statistics" on page 139

Guidelines for Using the list and get Subcommands for Monitoring

The underlying assumptions for using the list and get subcommands with dotted names are:

- A list subcommand that specifies a dotted name that is *not* followed by a wildcard (*) lists the current node's immediate children. For example, the following subcommand lists all immediate children belonging to the server node:

  ```
  list --monitor server
  ```

- A list subcommand that specifies a dotted name followed by a wildcard of the form .* lists a hierarchical tree of child nodes from the specified node. For example, the following subcommand lists all children of the applications node, their subsequent child nodes, and so on:

  ```
  list --monitor server.applications.*
  ```

- A list subcommand that specifies a dotted name preceded or followed by a wildcard of the form *dottedname* or *dotted * name* or *dottedname **lists all nodes and their children that match the regular expression created by the specified matching pattern.

- A get subcommand followed by a .* or a * gets the set of attributes and their values that belong to the node specified.

For example, the following table explains the output of the list and get subcommands used with the dotted name for the resources node.

TABLE 8–4   Example Resources Level Dotted Names

Subcommand	Dotted Name	Output
list --monitor	server.resources	List of pool names.

TABLE 8–4 Example Resources Level Dotted Names   *(Continued)*

Subcommand	Dotted Name	Output
list --monitor	server.resources.connection-pool1	No attributes, but a message saying "Use get subcommand with the --monitor option to view this node's attributes and values."
get --monitor	server.resources.connection-pool1.*	List of attributes and values corresponding to connection pool attributes.

For detailed information on dotted names, see the dotted-names(5ASC) help page.

▼ To View Comprehensive Monitoring Data

Although the monitor subcommand is useful in many situations, it does not offer the complete list of all monitorable objects. To work with comprehensive data for an object type, use the list --monitor and the get --monitor subcommands followed by the dotted name of a monitorable object.

Before You Begin   A monitorable object must be configured for monitoring before you can display information about the object. See "To Enable Monitoring " on page 131 if needed.

1   **List the objects that are enabled for monitoring by using the** list(1) **subcommand.**

For example, the following subcommand lists all components and services that have monitoring enabled for instance server.

```
asadmin> list --monitor "*"
server.web
server.connector-service
server.orb
server.jms-serviceserver.jvm
server.applications
server.http-service
server.thread-pools
```

2   **Get data for a monitored component or service by using the** get(1) **subcommand.**

Example 8–8   Viewing Attributes for a Specific Type

This example gets information about all the attributes for object type jvm on instance server.

```
asadmin> get --monitor server.jvm.*
server.jvm.class-loading-system.loadedclasscount = 3715
server.jvm.class-loading-system.totalloadedclasscount = 3731
```

```
server.jvm.class-loading-system.unloadedclasscount = 16
server.jvm.compilation-system.name-current = HotSpot Client Compiler
server.jvm.compilation-system.totalcompilationtime = 769
server.jvm.garbage-collectors.Copy.collectioncount = 285
server.jvm.garbage-collectors.Copy.collectiontime = 980
server.jvm.garbage-collectors.MarkSweepCompact.collectioncount = 2
server.jvm.garbage-collectors.MarkSweepCompact.collectiontime = 383
server.jvm.memory.committedheapsize = 23498752
server.jvm.memory.committednonheapsize = 13598720
server.jvm.memory.initheapsize = 0
server.jvm.memory.initnonheapsize = 8585216
server.jvm.memory.maxheapsize = 66650112
server.jvm.memory.maxnonheapsize = 100663296
server.jvm.memory.objectpendingfinalizationcount = 0
server.jvm.memory.usedheapsize = 19741184
server.jvm.memory.usednonheapsize = 13398352
server.jvm.operating-system.arch-current = x86
server.jvm.operating-system.availableprocessors = 2
server.jvm.operating-system.name-current = Windows XP
server.jvm.operating-system.version-current = 5.1
server.jvm.runtime.classpath-current = glassfish.jar
server.jvm.runtime.inputarguments-current = []
server.jvm.runtime.managementspecversion-current = 1.0
server.jvm.runtime.name-current = 4372@ABBAGANI_WORK
server.jvm.runtime.specname-current = Java Virtual Machine Specification
server.jvm.runtime.specvendor-current = Sun Microsystems Inc.
server.jvm.runtime.specversion-current = 1.0
server.jvm.runtime.uptime = 84813
server.jvm.runtime.vmname-current = Java HotSpot(TM) Client VM
server.jvm.runtime.vmvendor-current = Sun Microsystems Inc.
server.jvm.runtime.vmversion-current = 1.5.0_11-b03
```

Example 8–9   Viewing Monitorable Applications

This example lists all the monitorable applications for instance server.

```
asadmin> list --monitor server.applications.*
server.applications.app1
server.applications.app2
server.applications.app1.virtual-server1
server.applications.app2.virtual-server1
```

Example 8–10   Viewing Attributes for an Application

This example gets information about all the attributes for application hello.

```
asadmin> get --monitor server.applications.hello.*
server.applications.hello.server.activatedsessionstotal = 0
server.applications.hello.server.activejspsloadedcount = 1
server.applications.hello.server.activeservletsloadedcount = 1
server.applications.hello.server.activesessionscurrent = 1
server.applications.hello.server.activesessionshigh = 1
server.applications.hello.server.errorcount = 0
server.applications.hello.server.expiredsessionstotal = 0
server.applications.hello.server.maxjspsloadedcount = 1
server.applications.hello.server.maxservletsloadedcount = 0
server.applications.hello.server.maxtime = 0
server.applications.hello.server.passivatedsessionstotal = 0
server.applications.hello.server.persistedsessionstotal = 0
server.applications.hello.server.processingtime = 0.0
server.applications.hello.server.rejectedsessionstotal = 0
server.applications.hello.server.requestcount = 0
server.applications.hello.server.sessionstotal =
server.applications.hello.server.totaljspsloadedcount = 0
server.applications.hello.server.totalservletsloadedcount = 0
```

Example 8–11   Viewing a Specific Attribute

This example gets information about the jvm attribute runtime.vmversion-current on instance server.

```
asadmin> get --monitor server.jvm.runtime.vmversion-current
server.jvm.runtime.vmversion-current = 10.0-b23
```

Comprehensive Monitoring Statistics

You can get comprehensive monitoring statistics by forming a dotted name that specifies the statistic you are looking for. For example, the following dotted name will display the cumulative number of requests for the HTTP service on virtual-server1:

```
server.http-service.virtual-server1.request.requestcount
```

The tables in the following sections list the statistics that are available for each monitorable object:

- "EJB Statistics" on page 140
- "HTTP Service Statistics" on page 143
- "Jersey Statistics" on page 145
- "JMS/Connector Service Statistics" on page 145
- "JRuby Statistics" on page 147
- "JVM Statistics" on page 149
- "Network Statistics" on page 153

EJB Statistics

EJBs fit into the tree of objects as shown in "Applications Tree Hierarchy" on page 125. Use the following dotted name pattern to get applications statistics:

`server.applications.`*appname*`.`*ejbmodulename*`.`*ejbname*`.bean-cache.`*statistic*

Statistics available for applications are shown in the following sections:

EJB Cache Statistics

Use the following dotted name pattern for EJB cache statistics:

`server.applications.`*appname*`.`*ejbmodulename*`.bean-cache.`*ejbname*`.`*statistic*

The statistics available for EJB caches are listed in the following table.

TABLE 8–5   EJB Cache Monitoring Statistics

Statistic	Data Type	Description
cachemisses	RangeStatistic	The number of times a user request does not find a bean in the cache.
cachehits	RangeStatistic	The number of times a user request found an entry in the cache.
numbeansincache	RangeStatistic	The number of beans in the cache. This is the current size of the cache.
numpassivations	CountStatistic	Number of passivated beans. Applies only to stateful session beans.
numpassivationerrors	CountStatistic	Number of errors during passivation. Applies only to stateful session beans.

TABLE 8–5 EJB Cache Monitoring Statistics *(Continued)*

Statistic	Data Type	Description
numexpiredsessionsremoved	CountStatistic	Number of expired sessions removed by the cleanup thread. Applies only to stateful session beans.
numpassivationsuccess	CountStatistic	Number of times passivation completed successfully. Applies only to stateful session beans.

EJB Container Statistics

Use the following dotted name pattern for EJB container statistics:

server.applications.*appname.ejbmodulename*.container.*ejbname*

The statistics available for EJB containers are listed in the following table.

TABLE 8–6 EJB Container Monitoring Statistics

Statistic	Data Type	Description
createcount	CountStatistic	Number of times an EJB's create method is called.
messagecount	CountStatistic	Number of messages received for a message-driven bean.
methodreadycount	RangeStatistic	Number of stateful or stateless session beans that are in the MethodReady state.
passivecount	RangeStatistic	Number of stateful session beans that are in Passive state.
pooledcount	RangeStatistic	Number of entity beans in pooled state.
readycount	RangeStatistic	Number of entity beans in ready state.
removecount	CountStatistic	Number of times an EJB's remove method is called.

EJB Method Statistics

Use the following dotted name pattern for EJB method statistics:

server.applications.*appname.ejbmodulename*.bean-methods.*ejbname.statistic*

The statistics available for EJB method invocations are listed in the following table.

TABLE 8–7   EJB Method Monitoring Statistics

Statistic	Data Type	Description
executiontime	CountStatistic	Time, in milliseconds, spent executing the method for the last successful/unsuccessful attempt to run the operation. This is collected for stateless and stateful session beans and entity beans if monitoring is enabled on the EJB container.
methodstatistic	TimeStatistic	Number of times an operation is called; the total time that is spent during the invocation, and so on.
totalnumerrors	CountStatistic	Number of times the method execution resulted in an exception. This is collected for stateless and stateful session beans and entity beans if monitoring is enabled for the EJB container.
totalnumsuccess	CountStatistic	Number of times the method successfully executed. This is collected for stateless and stateful session beans and entity beans if monitoring enabled is true for EJB container.

EJB Pool Statistics

Use the following dotted name pattern for EJB pool statistics:

`server.applications.`*appname.ejbmodulename*`.bean-pool.`*ejbname.statistic*

The statistics available for EJB pools are listed in the following table.

TABLE 8–8   EJB Pool Monitoring Statistics

Statistic	Data Type	Description
jmsmaxmessagesload	CountStatistic	The maximum number of messages to load into a JMS session at one time for a message-driven bean to serve. Default is 1. Applies only to pools for message driven beans.
numbeansinpool	RangeStatistic	Number of EJBs in the associated pool, providing information about how the pool is changing.
numthreadswaiting	RangeStatistic	Number of threads waiting for free beans, giving an indication of possible congestion of requests.
totalbeanscreated	CountStatistic	Number of beans created in associated pool since the gathering of data started.

TABLE 8–8	EJB Pool Monitoring Statistics	*(Continued)*	
Statistic	Data Type	Description
totalbeansdestroyed	CountStatistic	Number of beans destroyed from associated pool since the gathering of data started.

Timer Statistics

Use the following dotted name pattern for timer statistics:

server.applications.*appname.ejbmodulename*.timers.*ejbname.statistic*

The statistics available for timers are listed in the following table.

TABLE 8–9   Timer Monitoring Statistics

Statistic	Data Type	Description
numtimerscreated	CountStatistic	Number of timers created in the system.
numtimersdelivered	CountStatistic	Number of timers delivered by the system.
numtimersremoved	CountStatistic	Number of timers removed from the system.

HTTP Service Statistics

The HTTP service fits into the tree of objects as shown in "HTTP Service Tree Hierarchy" on page 126.

The HTTP service statistics are described in the following sections:

- "HTTP Service Virtual Server Statistics" on page 143

HTTP Service Virtual Server Statistics

Use the following dotted name pattern for HTTP service virtual server statistics:

server.http-service.*virtual-server*.request.*statistic*

The HTTP service statistics for virtual servers are shown in the following table.

TABLE 8–10   HTTP Service Virtual Server Monitoring Statistics

Statistic	Data Type	Description
count200	CountStatistic	Number of responses with a status code equal to 200
count2xx	CountStatistic	Number of responses with a status code in the 2xx range

TABLE 8–10   HTTP Service Virtual Server Monitoring Statistics   *(Continued)*

Statistic	Data Type	Description
count302	CountStatistic	Number of responses with a status code equal to 302
count304	CountStatistic	Number of responses with a status code equal to 304
count3xx	CountStatistic	Number of responses with a status code equal in the 3xx range
count400	CountStatistic	Number of responses with a status code equal to 400
count401	CountStatistic	Number of responses with a status code equal to 401
count403	CountStatistic	Number of responses with a status code equal to 403
count404	CountStatistic	Number of responses with a status code equal to 404
count4xx	CountStatistic	Number of responses with a status code equal in the 4xx range
count503	CountStatistic	Number of responses with a status code equal to 503
count5xx	CountStatistic	Number of responses with a status code equal in the 5xx range
countother	CountStatistic	Number of responses with a status code outside the 2xx, 3xx, 4xx, and 5xx range
errorcount	CountStatistic	Cumulative value of the error count, with error count representing the number of cases where the response code was greater than or equal to 400
hosts	StringStatistic	The host (alias) names of the virtual server
maxtime	CountStatistic	Longest response time for a request; not a cumulative value, but the largest response time from among the response times
processingtime	CountStatistic	Cumulative value of the times taken to process each request, with processing time being the average of request processing times over the request count
requestcount	CountStatistic	Cumulative number of requests processed so far

TABLE 8–10   HTTP Service Virtual Server Monitoring Statistics   *(Continued)*

Statistic	Data Type	Description
state	StringStatistic	The state of the virtual server

Jersey Statistics

Jersey fits into the tree of objects as shown in "Applications Tree Hierarchy" on page 125.

Use the following dotted name pattern for Jersey statistics:

```
server.applications.jersey-application.jersey.resources.resource-0.hitcount.statistic
```

The statistics available for Jersey are shown in the following table.

TABLE 8–11   Jersey Statistics

Statistic	Data Type	Description
resourcehitcount	CountStatistic	Number of hits on this resource class
rootresourcehitcount	CountStatistic	Number of hits on this root resource class

JMS/Connector Service Statistics

The JMS/Connector Service fits into the tree of objects as shown in "JMS/Container Service Tree Hierarchy" on page 127.

JMS/Connector Service statistics are shown in the following sections:

- "Connector Connection Pool Statistics (JMS)" on page 145
- "Connector Work Management Statistics (JMS) " on page 146

Connector Connection Pool Statistics (JMS)

Use the following dotted name pattern for JMS/Connector Service connection pool statistics:

```
server.connector-service.resource-adapter-1.connection-pool.statistic
```

JMS/Connector Service statistics available for the connector connection pools are shown in the following table.

TABLE 8–12   Connector Connection Pool Monitoring Statistics (JMS)

Statistic	Data Type	Description
averageconnwaittime	CountStatistic	Average wait time of connections before they are serviced by the connection pool.

TABLE 8–12 Connector Connection Pool Monitoring Statistics (JMS)   *(Continued)*

Statistic	Data Type	Description
connectionrequestwaittime	RangeStatistic	The longest and shortest wait times of connection requests. The current value indicates the wait time of the last request that was serviced by the pool.
numconnfailedvalidation	CountStatistic	Total number of connections in the connection pool that failed validation from the start time until the last sample time.
numconnused	RangeStatistic	Total number of connections that are currently being used, as well as information about the maximum number of connections that were used (the high water mark).
numconnfree	RangeStatistic	Total number of free connections in the pool as of the last sampling.
numconntimedout	CountStatistic	Total number of connections in the pool that timed out between the start time and the last sample time.
numconncreated	CountStatistic	Number of physical connections, in milliseconds, that were created since the last reset.
numconndestroyed	CountStatistic	Number of physical connections that were destroyed since the last reset.
numconnacquired	CountStatistic	Number of logical connections acquired from the pool.
numconnreleased	CountStatistic	Number of logical connections released to the pool.
waitqueuelenght	CountStatistic	Number of connection requests in the queue waiting to be serviced.

Connector Work Management Statistics (JMS)

Use the following dotted name pattern for JMS/Connector Service work management statistics:

`server.connector-service.`*resource-adapter-1*`.work-management.`*statistic*

JMS/Connector Service statistics available for connector work management are listed in the following table.

TABLE 8–13 Connector Work Management Monitoring Statistics (JMS)

Statistic	Data Type	Description
activeworkcount	RangeStatistic	Number of work objects executed by the connector.
completedworkcount	CountStatistic	Number of work objects that were completed.
rejectedworkcount	CountStatistic	Number of work objects rejected by the Enterprise Server.
submittedworkcount	CountStatistic	Number of work objects submitted by a connector module.
waitqueuelength	RangeStatistic	Number of work objects waiting in the queue before executing.
workrequestwaittime	RangeStatistic	Longest and shortest wait of a work object before it gets executed.

JRuby Statistics

JRuby fits into the tree of objects as show in "JRuby Tree Hierarchy" on page 127.

The statistics that are available for JRuby are shown in the following sections:

- "JRuby Container Statistics" on page 147
- "JRuby Runtime Statistics" on page 148
- "JRuby HTTP Service Statistics" on page 148

JRuby Container Statistics

Use the following dotted name pattern for JRuby container statistics:

`server.containers.jruby.applications.`*jruby-application*`.`*statistic*

The statistics that are available for the JRuby container are shown in the following table.

TABLE 8–14 JRuby Container Statistics

Statistic	Data Type	Description
environment	StringStatistic	JRuby application environment
appname	StringStatistic	Ruby application name
contextpath	StringStatistic	Context path of Ruby application
jrubyversion	StringStatistic	JRuby version
rubyframework	StringStatistic	Ruby application framework

JRuby Runtime Statistics

Use the following dotted name pattern for JRuby runtime statistics:

`server.containers.jruby.applications.`*jruby-application*`.runtime.`*statistic*

The statistics that are available for the JRuby runtime are shown in the following table.

TABLE 8–15 JRuby Runtime Statistics

Statistic	Data Type	Description
`activeruntimes`	CountStatistic	Currently active runtimes
`appname`	StringStatistic	Ruby application name
`hardmaximum`	CountStatistic	Maximum active runtimes
`hardminimum`	CountStatistic	Minimum active runtimes

JRuby HTTP Service Statistics

Use the following dotted name pattern for JRuby HTTP service statistics:

`server.containers.jruby.applications.`*jruby-application*`.http.`*statistic*

The statistics that are available for the JRuby HTTP service are shown in the following table.

TABLE 8–16 JRuby HTTP Service Statistics

Statistic	Data Type	Description
`address`	StringStatistic	Server address
`appname`	StringStatistic	Ruby application name
`averageprocessingtime`	CountStatistic	Average request processing time in milliseconds
`contextpath`	StringStatistic	Context path of Ruby application
`count2xx`	CountStatistic	Number of responses with a status code in the 2xx range
`count200`	CountStatistic	Number of responses with a status code equal to 200
`count3xx`	CountStatistic	Number of responses with a status code in the 3xx range
`count302`	CountStatistic	Number of responses with a status code equal to 302

TABLE 8–16 JRuby HTTP Service Statistics   *(Continued)*

Statistic	Data Type	Description
Count304	CountStatistic	Number of responses with a status code equal to 304
count4xx	CountStatistic	Number of responses with a status code in the 4xx range
count400	CountStatistic	Number of responses with a status code equal to 400
count401	CountStatistic	Number of responses with a status code equal to 401
count403	CountStatistic	Number of responses with a status code equal to 403
count404	CountStatistic	Number of responses with a status code equal to 404
count5xx	CountStatistic	Number of responses with a status code in the 5xx range
count503	CountStatistic	Number of responses with a status code equal to 503
countother	CountStatistic	Number of responses with other status codes
errorcount	CountStatistic	Number of responses with a status code greater than 400
requests/seconds	CountStatistic	Requests per second

JVM Statistics

The JVM fits into the tree of objects as show in "JVM Tree Hierarchy" on page 127.

The statistics that are available for the Virtual Machine for Java platform (Java™ Virtual Machine) or JVM machine are shown in the following sections:

JVM Class Loading System Statistics

Use the following dotted name pattern for JVM class loading system statistics:

`server.jvm.class-loading-system.`*statistic*

With Java SE, additional monitoring information can be obtained from the JVM. Set the monitoring level to LOW to enable the display of this additional information. Set the monitoring level to HIGH to also view information pertaining to each live thread in the system. More information about the additional monitoring features for Java SE is available in *Monitoring and Management for the Java Platform*, which is available from `http://java.sun.com/javase/6/docs/technotes/guides/management/`.

The Java SE monitoring tools are discussed at `http://java.sun.com/javase/6/docs/technotes/tools/#manage`.

The statistics that are available for class loading in the JVM for Java SE are shown in the following table.

TABLE 8–17 JVM Monitoring Statistics for Java SE Class Loading

Statistic	Data Type	Description
`loadedclasscount`	CountStatistic	Number of classes that are currently loaded in the JVM
`totalloadedclasscount`	CountStatistic	Total number of classes that have been loaded since the JVM began execution
`unloadedclasscount`	CountStatistic	Number of classes that have been unloaded from the JVM since the JVM began execution

The statistics available for threads in the JVM in Java SE are shown in the following table.

TABLE 8–18 JVM Monitoring Statistics for Java SE - Threads

Statistic	Data Type	Description
`allthreadids`	StringStatistic	List of all live thread ids.
`currentthreadcputime`	CountStatistic	CPU time for the current thread (in nanoseconds) if CPU time measurement is enabled. If CPU time measurement is disabled, returns -1.
`daemonthreadcount`	CountStatistic	Current number of live daemon threads.
`monitordeadlockedthreads`	StringStatistic	List of thread ids that are monitor deadlocked.

TABLE 8–18   JVM Monitoring Statistics for Java SE - Threads   *(Continued)*

Statistic	Data Type	Description
peakthreadcount	CountStatistic	Peak live thread count since the JVM started or the peak was reset.
threadcount	CountStatistic	Current number of live daemon and non-daemon threads.
totalstartedthreadcount	CountStatistic	Total number of threads created and/or started since the JVM started.

JVM Compilation System Statistics

Use the following dotted name pattern for JVM compilation system statistics:

server.jvm.compilation-system.*statistic*

The statistics that are available for compilation in the JVM for Java SE are shown in the following table.

TABLE 8–19   JVM Monitoring Statistics for Java SE Compilation

Statistic	Data Type	Description
name-current	StringStatistic	Name of the current compiler
totalcompilationtime	CountStatistic	Accumulated time (in milliseconds) spent in compilation

JVM Garbage Collectors Statistics

Use the following dotted name pattern for JVM garbage collectors statistics:

server.jvm.garbage-collectors.*statistic*

The statistics that are available for garbage collection in the JVM for Java SE are shown in the following table.

TABLE 8–20   JVM Monitoring Statistics for Java SE Garbage Collectors

Statistic	Data Type	Description
collectioncount	CountStatistic	Total number of collections that have occurred
collectiontime	CountStatistic	Accumulated time (in milliseconds) spent in collection

JVM Memory Statistics

Use the following dotted name pattern for JVM memory statistics:

`server.jvm.memory.`*statistic*

The statistics that are available for memory in the JVM for Java SE are shown in the following table.

TABLE 8–21 JVM Monitoring Statistics for Java SE Memory

Statistic	Data Type	Description
`committedheapsize`	CountStatistic	Amount of heap memory (in bytes) that is committed for the JVM to use
`committednonheapsize`	CountStatistic	Amount of non-heap memory (in bytes) that is committed for the JVM to use
`initheapsize`	CountStatistic	Size of the heap initially requested by the JVM
`initnonheapsize`	CountStatistic	Size of the non-heap area initially requested by the JVM
`maxheapsize`	CountStatistic	Maximum amount of heap memory (in bytes) that can be used for memory management
`maxnonheapsize`	CountStatistic	Maximum amount of non-heap memory (in bytes) that can be used for memory management
`objectpendingfinalizationcount`	CountStatistic	Approximate number of objects that are pending finalization
`usedheapsize`	CountStatistic	Size of the heap currently in use
`usednonheapsize`	CountStatistic	Size of the non-heap area currently in use

JVM Operating System Statistics

Use the following dotted name pattern for JVM operating system statistics:

`server.jvm.operating-system.`*statistic*

The statistics that are available for the operating system for the JVM machine in Java SE are shown in the following table.

TABLE 8–22 JVM Statistics for Java SE Operating System

Statistic	Data Type	Description
`arch-current`	StringStatistic	Operating system architecture

TABLE 8–22   JVM Statistics for Java SE Operating System   *(Continued)*

Statistic	Data Type	Description
`availableprocessors`	CountStatistic	Number of processors available to the JVM
`name-current`	StringStatistic	Operating system name
`version-current`	StringStatistic	Operating system version

JVM Runtime Statistics

Use the following dotted name pattern for JVM runtime statistics:

`server.jvm.runtime.`*statistic*

The statistics that are available for the runtime in the JVM runtime for Java SE are shown in the following table.

TABLE 8–23   JVM Monitoring Statistics for Java SE Runtime

Statistic	Data Type	Description
`classpath-current`	StringStatistic	Classpath that is used by the system class loader to search for class files
`inputarguments-current`	StringStatistic	Input arguments passed to the JVM; not including arguments to the `main` method
`managementspecversion-current`	StringStatistic	Management specification version implemented by the JVM
`name-current`	StringStatistic	Name representing the running JVM
`specname-current`	StringStatistic	JVM specification name
`specvendor-current`	StringStatistic	JVM specification vendor
`specversion-current`	StringStatistic	JVM specification version
`uptime`	CountStatistic	Uptime of the JVM (in milliseconds)
`vmname-current`	StringStatistic	JVM implementation name
`vmvendor-current`	StringStatistic	JVM implementation vendor
`vmversion-current`	StringStatistic	JVM implementation version

Network Statistics

Network fits into the tree of objects as shown in "Network Tree Hierarchy" on page 128.

Network statistics are described in the following sections:

- "Network Keep Alive Statistics" on page 154

Network Keep Alive Statistics

Use the following dotted name pattern for network keep alive statistics:

`server.network.`*type-of-listener*`.keep-alive.`*statistic*

Statistics available for network keep alive are shown in the following table.

TABLE 8–24 Network Keep Alive Statistics

Statistic	Data Type	Description
countconnections	CountStatistic	Number of connections in keep-alive mode.
counttimeouts	CountStatistic	Number of keep-alive connections that timed out.
secondstimeouts	CountStatistic	Keep-alive timeout value in seconds.
maxrequests	CountStatistic	Maximum number of requests allowed on a single keep-alive connection.
countflushes	CountStatistic	Number of keep-alive connections that were closed.
counthits	CountStatistic	Number of requests received by connections in keep-alive mode.
countrefusals	CountStatistic	Number of keep-alive connections that were rejected.

Network Connection Queue Statistics

Use the following dotted name pattern for network connection queue statistics:

`server.network.`*type-of-listener*`.connection-queue.`*statistic*

Statistics available for network connection queue are shown in the following table.

TABLE 8–25 Network Connection Queue Statistics

Statistic	Data Type	Description
countopenconnections	CountStatistic	The number of open/active connections
countoverflows	CountStatistic	Number of times the queue has been too full to accommodate a connection

TABLE 8–25 Network Connection Queue Statistics   *(Continued)*

Statistic	Data Type	Description
countqueued	CountStatistic	Number of connections currently in the queue
countqueued15minutesaverage	CountStatistic	Average number of connections queued in the last 15 minutes
countqueued1minuteaverage	CountStatistic	Average number of connections queued in the last 1 minute
countqueued5minutesaverage	CountStatistic	Average number of connections queued in the last 5 minutes
counttotalconnections	CountStatistic	Total number of connections that have been accepted
counttotalqueued	CountStatistic	Total number of connections that have been queued
maxqueued	CountStatistic	Maximum size of the connection queue
peakqueued	CountStatistic	Largest number of connections that were in the queue simultaneously
tickstotalqueued	CountStatistic	(Unsupported) Total number of ticks that connections have spent in the queue

Network File Cache Statistics

Use the following dotted name pattern for network file cache statistics:

`server.network.`*type-of-listener*`.file-cache.`*statistic*

Statistics available for network file cache are shown in the following table.

TABLE 8–26 Network File Cache Statistics

Statistic	Data Type	Description
contenthits	CountStatistic	Number of hits on cached file content
contentmisses	CountStatistic	Number of misses on cached file content
heapsize	CountStatistic	Current cache size in bytes
hits	CountStatistic	Number of cache lookup hits
infohits	CountStatistic	Number of hits on cached file info
infomisses	CountStatistic	Number of misses on cached file info
mappedmemorysize	CountStatistic	Size of mapped memory used for caching in bytes

TABLE 8–26 Network File Cache Statistics *(Continued)*

Statistic	Data Type	Description
maxheapsize	CountStatistic	Maximum heap space used for cache in bytes
maxmappedmemorysize	CountStatistic	Maximum memory map size used for caching in bytes
misses	CountStatistic	Number of cache lookup misses data type
opencacheentries	CountStatistic	Number of current open cache entries

Network Thread Pool Statistics

Use the following dotted name pattern for network thread pool statistics:

`server.network.`*type-of-listener*`.thread-pool.`*statistic*

Statistics available for network thread pool are shown in the following table.

TABLE 8–27 Network Thread Pool Statistics

Statistic	Data Type	Description
corethreads	CountStatistic	Core number of threads in the thread pool
currentthreadcount	CountStatistic	Provides the number of request processing threads currently in the listener thread pool
currentthreadsbusy	CountStatistic	Provides the number of request processing threads currently in use in the listener thread pool serving requests
maxthreads	CountStatistic	Maximum number of threads allowed in the thread pool
totalexecutedtasks	CountStatistic	Provides the total number of tasks, which were executed by the thread pool

ORB Statistics (Connection Manager)

The ORB fits into the tree of objects as shown in "ORB Tree Hierarchy" on page 128.

Use the following dotted name patterns for ORB statistics:

`server.orb.transport.connectioncache.inbound.`*statistic*
`server.orb.transport.connectioncache.outbound.`*statistic*

The statistics available for the connection manager in an ORB are listed in the following table.

TABLE 8–28   ORB Monitoring Statistics (Connection Manager)

Statistic	Data Type	Description
connectionsidle	CountStatistic	Total number of connections that are idle to the ORB
connectionsinuse	CountStatistic	Total number of connections in use to the ORB
totalconnections	BoundedRangeStatistic	Total number of connections to the ORB

Resource Statistics (Connection Pool)

By monitoring connection pool resources you can measure performance and capture resource usage at runtime. Connections are expensive and frequently cause performance bottlenecks in applications. It is important to monitor how a connection pool is releasing and creating new connections and how many threads are waiting to retrieve a connection from a particular pool.

The connection pool resources fit into the tree of objects as shown in "Resources Tree Hierarchy" on page 128.

Use the following dotted name pattern for connection pool statistics:

`server.resources.`*connection-pool*`.`*statistic*

The connection pool statistics are shown in the following table.

TABLE 8–29   Resource Monitoring Statistics (Connection Pool)

Statistic	Data Type	Description
averageconnwaittime	CountStatistic	Average wait-time-duration per successful connection request
connrequestwaittime	RangeStatistic	Longest and shortest wait times, in milliseconds, of connection requests since the last sampling. current value indicates the wait time of the last request that was serviced by the pool
numconnfailedvalidation	CountStatistic	Number of connections in the connection pool that failed validation from the start time until the last sampling time
numconnused	RangeStatistic	Number of connections that are currently being used, as well as information about the maximum number of connections that were used (high water mark)
numconnfree	RangeStatistic	Number of free connections in the pool as of the last sampling

TABLE 8–29 Resource Monitoring Statistics (Connection Pool)   *(Continued)*

Statistic	Data Type	Description
numconntimedout	CountStatistic	Number of connections in the pool that timed out between the start time and the last sampling time
numconncreated	CountStatistic	Number of physical connections that were created by the pool since the last reset
numconndestroyed	CountStatistic	Number of physical connections that were destroyed since the last reset
numconnacquired	CountStatistic	Number of logical connections acquired from the pool since the last sampling
numconnreleased	CountStatistic	Number of connections released back to the pool since the last sampling
numconnnotsuccessfullymatched	CountStatistic	Number of connections rejected during matching
numconnsuccessfullymatched	CountStatistic	Number of connections successfully matched
numpotentialconnleak	CountStatistic	Number of potential connection leaks
waitqueuelength	CountStatistic	Number of connection requests in the queue waiting to be serviced

Security Statistics

Security fits into the tree of objects as shown in "Security Tree Hierarchy" on page 129.

Statistics available for security are shown in the following sections:

- "EJB Security Statistics" on page 158
- "Web Security Statistics" on page 159
- "Realm Security Statistics" on page 159

EJB Security Statistics

Use the following dotted name pattern for EJB security statistics:

server.security.ejb.*statistic*

The statistics available for EJB security are listed in the following table.

TABLE 8–30 EJB Security Monitoring Statistics

Statistic	Data Type	Description
policyconfigurationcount	CountStatistic	Number of policy configuration

TABLE 8–30   EJB Security Monitoring Statistics   *(Continued)*

Statistic	Data Type	Description
`securitymanagercount`	CountStatistic	Number of EJB security managers

Web Security Statistics

Use the following dotted name pattern for web security statistics:

`server.security.web.`*statistic*

The statistics available for web security are listed in the following table.

TABLE 8–31   Web Security Monitoring Statistics

Statistic	Data Type	Description
`websecuritymanagercount`	CountStatistic	Number of security managers
`webpolicyconfigurationcount`	CountStatistic	Number of policy configuration objects

Realm Security Statistics

Use the following dotted name pattern for realm security statistics:

`server.security.realm.`*statistic*

The statistics available for realm security are listed in the following table.

TABLE 8–32   Realm Security Monitoring Statistics

Statistic	Data Type	Description
`realmcount`	CountStatistic	Number of realms

Thread Pool Statistics

The thread pool fits into the tree of objects as shown in "Thread Pool Tree Hierarchy" on page 129.

The statistics available for thread pools are shown in the following sections:

- "Thread Pool Monitoring Statistics" on page 160
- "JVM Statistics for Java SE-Thread Information" on page 160

Thread Pool Monitoring Statistics

Use the following dotted name pattern for thread pool statistics:

`server.thread-pool.`*thread-pool*`.`*statistic*

The statistics available for the thread pool are shown in the following table.

TABLE 8–33 Thread Pool Monitoring Statistics

Statistic	Data Type	Description
`averagetimeinqueue`	BoundedRangeStatistic	Average amount of time (in milliseconds) a request waited in the queue before being processed
`averageworkcompletiontime`	BoundedRangeStatistic	Average amount of time (in milliseconds) taken to complete an assignment
`currentbusythreads`	CountStatistic	Number of busy threads
`currentnumberofthreads`	BoundedRangeStatistic	Current number of request processing threads
`numberofavailablethreads`	CountStatistic	Number of available threads
`numberofworkitemsinqueue`	BoundedRangeStatistic	Current number of work items waiting in queue
`totalworkitemsadded`	CountStatistic	Total number of work items added to the work queue as of last sampling

JVM Statistics for Java SE-Thread Information

The statistics available for `ThreadInfo` in the JVM in Java SE are shown in the following table.

TABLE 8–34 JVM Monitoring Statistics for Java SE - Thread Info

Statistic	Data Type	Description
`blockedcount`	CountStatistic	Total number of times that the thread entered the `BLOCKED` state.
`blockedtime`	CountStatistic	Time elapsed (in milliseconds) since the thread entered the `BLOCKED` state. Returns -1 if thread contention monitoring is disabled.
`lockname`	StringStatistic	String representation of the monitor lock that the thread is blocked to enter or waiting to be notified through the `Object.wait` method.
`lockownerid`	CountStatistic	ID of the thread that holds the monitor lock of an object on which this thread is blocking.

TABLE 8–34   JVM Monitoring Statistics for Java SE - Thread Info   *(Continued)*

Statistic	Data Type	Description
lockownername	StringStatistic	Name of the thread that holds the monitor lock of the object this thread is blocking on.
stacktrace	StringStatistic	Stack trace associated with this thread.
threadid	CountStatistic	ID of the thread.
threadname	StringStatistic	Name of the thread.
threadstate	StringStatistic	State of the thread.
waitedtime	CountStatistic	Elapsed time (in milliseconds) that the thread has been in a WAITING state. Returns -1 if thread contention monitoring is disabled.
waitedcount	CountStatistic	Total number of times the thread was in WAITING or TIMED_WAITING states.

Transaction Service Statistics

The transaction service allows the client to freeze the transaction subsystem in order to roll back transactions and determine which transactions are in process at the time of the freeze. The transaction service fits into the tree of objects as shown in "Transactions Service Tree Hierarchy" on page 129.

Use the following dotted name pattern for transaction service statistics:

`server.transaction-service.`*statistic*

The statistics available for the transaction service are shown in the following table.

TABLE 8–35   Transaction Service Monitoring Statistics

Statistic	Data Type	Description
activecount	CountStatistic	Number of transactions currently active.
activeids	StringStatistic	The ID's of the transactions that are currently active. Every such transaction can be rolled back after freezing the transaction service.
committedcount	CountStatistic	Number of transactions that have been committed.
rolledbackcount	CountStatistic	Number of transactions that have been rolled back.
state	StringStatistic	Indicates whether or not the transaction has been frozen.

Web Statistics

The web module fits into the tree of objects as shown in

The available web statistics shown in the following sections:

Web Module Servlet Statistics

Use the following dotted name pattern for web module servlet statistics:

```
server.applications.web-module.virtual-server.servlet.statistic
server.applications.application.web-module.virtual-server.servlet.statistic
```

The available web module servlet statistics are shown in the following table.

TABLE 8–36   Web Module Servlet Statistics

Statistic	Data Type	Description
errorcount	CountStatistic	Cumulative number of cases where the response code is greater than or equal to 400.
maxtime	CountStatistic	Maximum amount of time the web container waits for requests.
processingtime	CountStatistic	Cumulative value of the amount of time required to process each request. The processing time is the average of request processing times divided by the request count.
requestcount	CountStatistic	The total number of requests processed so far.
servicetime	CountStatistic	Aggregate response time in milliseconds.

Web JSP Statistics

Use the following dotted name pattern for web JSP statistics:

```
server.applications.web-module.virtual-server.statistic
server.applications.application.web-module.virtual-server.statistic
```

The available web JSP statistics are shown in the following table.

TABLE 8–37   Web JSP Monitoring Statistics

Statistic	Data Type	Description
jspcount-current	RangeStatistic	Number of active JSP pages
jsperrorcount	CountStatistic	Total number of errors triggered by JSP page invocations
jspreloadedcount	CountStatistic	Total number of JSP pages that were reloaded
totaljspcount	CountStatistic	Total number of JSP pages ever loaded

Web Request Statistics

Use the following dotted name pattern for web request statistics:

```
server.applications.web-module.virtual-server.statistic
server.applications.application.web-module.virtual-server.statistic
```

The available web request statistics are shown in the following table.

TABLE 8–38   Web Request Monitoring Statistics

Statistic	Data Type	Description
errorcount	CountStatistic	Cumulative value of the error count, with error count representing the number of cases where the response code was greater than or equal to 400
maxtime	CountStatistic	Longest response time for a request; not a cumulative value, but the largest response time from among the response times
processingtime	CountStatistic	Average request processing time, in milliseconds
requestcount	CountStatistic	Cumulative number of the requests processed so far

Web Servlet Statistics

Use the following dotted name pattern for web servlet statistics:

```
server.applications.web-module.virtual-server.statistic
server.applications.application.web-module.virtual-server.statistic
```

The available web servlet statistics are shown in the following table.

TABLE 8–39   Web Servlet Monitoring Statistics

Statistic	Data Type	Description
`activeservletsloadedcount`	RangeStatistic	Number of currently loaded servlets
`servletprocessingtimes`	CountStatistic	Cumulative servlet processing times , in milliseconds
`totalservletsloadedcount`	CountStatistic	Cumulative number of servlets that have been loaded into the web module

Web Session Statistics

Use the following dotted name pattern for web session statistics:

```
server.applications.web-module.virtual-server.statistic
server.applications.application.web-module.virtual-server.statistic
```

The available web session statistics are shown in the following table.

TABLE 8–40   Web Session Monitoring Statistics

Statistic	Data Type	Description
`activatedsessionstotal`	CountStatistic	Total number of activated sessions
`activesessionscurrent`	RangeStatistic	Number of currently active sessions
`activesessionhigh`	CountStatistic	Maximum number of concurrently active sessions
`expiredsessionstotal`	CountStatistic	Total number of expired sessions
`passivatedsessionstotal`	CountStatistic	Total number of passivated sessions
`persistedsessionstotal`	CountStatistic	Total number of persisted sessions
`rejectedsessionstotal`	CountStatistic	Total number of rejected sessions
`sessionstotal`	CountStatistic	Total number of sessions created

Configuring JConsole to View Enterprise Server Monitoring Data

Java SE provides tools to connect to an MBean Server and view the MBeans registered with the server. JConsole is one such popular JMX Connector Client and is available as part of the standard Java SE distribution. When you configure JConsole for use with Enterprise Server, Enterprise Server becomes the JMX Connector's server end and JConsole becomes the JMX connector's client end.

▼ To Connect JConsole to Enterprise Server

Java SE 6 enhances management and monitoring of the virtual machine by including a Platform MBean Server and by including managed beans (MBeans) to configure the virtual machine.

To view all MBeans, Enterprise Server provides a configuration of the standard JMX connector server called System JMX Connector Server. As part of Enterprise Server startup, an instance of this JMX Connector Server is started. Any compliant JMX connector client can connect to the server using the JMX Connector Server.

By default, Enterprise Server is configured with a non-secure System JMX Connector Server. If this is an issue, the JMX connector can be removed. However, access can be restricted to a specific IP address (for example, the loopback address) by setting address to localhost.

1   Start the domain.

For instructions, see .

2   Start JConsole using this format: *JDK_HOME*/bin/jconsole

For example:

/usr/java/bin/jconsole

The JConsole Connect to Agent window is displayed.

3   Click the Remote tab and type the host name and port.

Always connect remotely with JConsole, otherwise MBeans will not load automatically.

4   Click Connect.

5   In the Remote Process text box, specify the JMX Service URL.

For example:

service:jmx:rmi:///jndi/rmi://localhost:8686/jmxrmi

The JMX Service URL is emitted by the server at startup, looking something like this:

```
[#|2009-12-03T10:25:17.737-0800|INFO|glassfishv3.0|
javax.enterprise.system.tools.admin.org.glassfish.server|_ThreadID=20;
_ThreadName=Thread-26;|JMXStartupService: Started JMXConnector, JMXService
URL = service:jmx:rmi://localhost:8686/jndi/rmi://localhost:8686/jmxrmi|#]
```

However, in most cases, simply entering host:port is fine, such as, 192.168.1.150:8686. The long Service URL is not needed.

> **Note –** Another host name can be substituted for localhost. The default port number (8686)
> could change if the jmx-connector configuration has been modified.

6 Click Connect.

In the JConsole window you will see all your MBeans, JVM information, and so on, in various
tabs. Most of the useful MBeans are to be found in the amx and java.lang domains.

See Also For more information about JConsole, see http://java.sun.com/javase/6/docs/
technotes/guides/management/jconsole.html.

9

Administering Life Cycle Modules

This chapter provides procedures for administering life cycle modules in the Sun GlassFish™Enterprise Server v3 environment.

The following topics are addressed here:

- "About Life Cycle Modules" on page 167
- "Configuring Life Cycle Modules" on page 168

Instructions for accomplishing the tasks in this chapter by using the Administration Console are contained in the Administration Console online help.

About Life Cycle Modules

Life cycle modules, also known as initialization services, provide a means of running short or long duration Java-based tasks within the Enterprise Server environment. These modules are automatically initiated at server startup and are notified at various phases of the server life cycle. Configured properties for a life cycle module are passed as properties during server initialization.

All life cycle module classes and interfaces are in the *as-install*/glassfish/modules/glassfish-api.jar file.

A life cycle module listens for and performs its tasks in response to the following Enterprise Server sequence of events:

1. **Initialization.** The server reads the configuration, initializes built-in subsystems (such as security and logging services), and creates the containers.

2. **Startup.** The server loads and initializes deployed applications.

3. **Ready.** The server begins servicing requests.

4. **Shutdown.** The server shuts down the applications and stops.

5. **Termination.** The server closes the containers, the built-in subsystems, and the server runtime environment.

These events are defined in the LifecycleEvent class. For information on creating life cycle modules, see Chapter 13, "Developing Lifecycle Listeners," in *Sun GlassFish Enterprise Server v3 Application Development Guide*.

Note – If the is-failure-fatal setting is set to true (the default is false), life cycle module failure prevents server initialization or startup, but not shutdown or termination.

Configuring Life Cycle Modules

The following topics are addressed here:

▼ To Create a Life Cycle Module

Use the create-lifecycle-module subcommand in remote mode to create a life cycle module.

1 Ensure that the server is running.

Remote subcommands require a running server.

2 Create a new life cycle modules by using the create-lifecycle-module**(1) subcommand.**

Information about options and properties for the subcommand are included in this help page.

3 Restart the server for your changes to take effect.

See "To Restart a Domain" on page 90.

Example 9–1 Creating a Life Cycle Module

This example creates the customSetup life cycle module :

```
asadmin> create-lifecycle-module --classname "com.acme.CustomSetup"
--classpath "/export/customSetup" --loadorder 1 --failurefatal=true
--description "this is a sample customSetup"
--property rmi="Server\=acme1\:7070":timeout=30 customSetup
Command create-lifecycle-module executed successfully
```

See Also You can also view the full syntax and options of the subcommand by typing asadmin help
create-lifecycle-module at the command line.

▼ To List Life Cycle Modules

Use the list-lifecycle-modules subcommand in remote mode to list the existing life cycle
modules.

1 Ensure that the server is running.

Remote subcommands require a running server.

2 List life cycle modules by using the list-lifecycle-modules**(1) subcommand.**

Example 9–2 Listing Life Cycle Modules

This example lists the existing life cycle modules.

```
asadmin> list-lifecycle-modules
WSTCPConnectorLCModule
Command list-lifecycle-modules executed successfully
```

See Also You can also view the full syntax and options of the subcommand by typing asadmin help
list-lifecycle-modules at the command line.

▼ To Update a Life Cycle Module

Use the set subcommand to update an existing life cycle module.

1 List the properties that can be updated for a life cycle module by using the get**(1) subcommand.**

For example (single mode):

```
asadmin get "*" | grep sampleLCM
applications.application.sampleLCMmodule.availability-enabled=false
applications.application.sampleLCMmodule.directory-deployed=false
applications.application.sampleLCMmodule.enabled=true
applications.application.sampleLCMmodule.name=sampleLCMmodule
applications.application.sampleLCMmodule.object-type=user
applications.application.sampleLCMmodule.property.class-name=example.lc.SampleModule
applications.application.sampleLCMmodule.property.classpath=/build/lcm.jar
applications.application.sampleLCMmodule.property.is-failure-fatal=false
applications.application.sampleLCMmodule.property.isLifecycle=true
```

2 Update a life cycle module by using the set**(1) subcommand.**

3 Restart the server for your changes to take effect.

See "To Restart a Domain" on page 90.

Example 9–3 Updating a Life Cycle Module

This example updates the classpath property.

```
sadmin> set applications.application.sampleLCMmodule.
property.classpath=/build/lcm_new.jarapplications.application.
sampleLCMmodule.property.classpath=/build/lcm_new.jar
Command set executed successfully.
```

See Also You can also view the full syntax and options of the subcommand by typing asadmin help set at the command line.

▼ To Delete a Life Cycle Module

Use the delete-lifecycle-module subcommand in remote mode to delete a life cycle module.

1 Ensure that the server is running.

Remote subcommands require a running server.

2 List the current life cycle modules by using the list-lifecycle-modules**(1) subcommand.**

3 Delete a life cycle module by using the delete-lifecycle-module**(1) subcommand.**

Example 9–4 Deleting a Life Cycle Module

This example deletes the customSetup life cycle module.

```
asadmin> delete-lifecycle-module customSetup
Command delete-lifecycle-module executed successfully
```

See Also You can also view the full syntax and options of the subcommand by typing asadmin help delete-lifecycle-module at the command line.

Extending Enterprise Server

This chapter provides procedures for performing tasks associated with extending and updating the deployed Sun GlassFish™Enterprise Server v3 by using Update Tool. Procedures for using the pkg command are contained in this chapter. A set of reference pages that contain details on using the pkg command is included with Enterprise Server.

The following topics are addressed here:

- "About Add-on Components" on page 171
- "Adding Components" on page 172
- "Updating Installed Components" on page 174
- "Removing Installed Components" on page 176

Instructions for accomplishing the tasks in this chapter by using the Administration Console are contained in the Administration Console online help for Update Tool. The standalone Update Tool also contains online help. For additional information about Update Tool, see `http://wikis.sun.com/display/IpsBestPractices/Screenshots`

About Add-on Components

Enterprise Server is designed to provide its functionality in a modular form so that you can choose to include the functionality that you need and leave out the functionality that is not needed. *OSGi modules*, also called bundles, provide add-on functionality for your deployed Enterprise Server. You can add or remove an add-on component during runtime, with no need to restart the server.

Over time, new add-on components will be developed and existing components will be modified. Enterprise Server provides a set of image packaging system (IPS) tools for updating software on a deployed server. You can install new or updated add-on components by using the Update Tool in the following ways:

- As a selection in the graphical Enterprise Server Administration Console

- As a standalone graphical tool which can be started from the command line using the following format: *as-install-parent*/bin/updatetool
- As a command-line utility (the pkg command) which can perform most of the tasks offered by the graphical Update Tool. The main reasons for using the pkg command are:
    - As a basis for creating update scripts
    - As a method for working with systems that do not have a monitor, graphic card, or keyboard (headless systems)

Extensive online help is provided for both graphical versions.

Adding Components

This section provides instructions for using the pkg command to install Enterprise Server add-on components on your deployed Enterprise Server.

▼ To Install an Add-on Component

The pkg command enables you to install an add-on component on your system. If multiple versions of a package are available, the latest one is applied unless you specify otherwise. The pkg command, located in the *as-install-parent*/bin directory,

---

Note – If the pkg component, the updatetool component, or any other valid component that you try to invoke from the command line is not yet installed on your deployed Enterprise Server, you will receive a query asking if you want to install the component. Answer Y to install the component.

Before You Begin Enterprise Server v3 must be fully deployed before you can install additional components. If you need installation instructions, see *Sun GlassFish Enterprise Server v3 Installation Guide*.

1 **List your installed components:**

pkg list

Information similar to the following is displayed:

```
NAME (AUTHORITY)                          VERSION        STATE      UFIX
glassfishv3-common                        0-1            installed  ----
glassfishv3-ejb                           0-1            installed  u---
glassfishv3-nucleus                       0-1            installed  ----
glassfishv3-web                           0-1            installed  ----
grails                                    1.0-1.0        installed  ----
jersey                                    0.7-0.1        installed  u---
```

```
jmaki                                        1.8.0-1.0       installed  ----
jruby                                        1.1.1-1.0       installed  ----
metro                                        1.2-1           installed  u---
pkg                                          0.1.4-6.564     installed  u---
python2.4-minimal                            2.4.4-6.564     installed  u---
updatetool (glassfish.org)                   2.0-6.564       installed  u---
wxpython2.8-minimal                          2.8.7.1-6.564   installed  u---
```

2   List all packages that are available:

`pkg list -a`

Information similar to the following is displayed from the repository:

```
NAME (AUTHORITY)                             VERSION         STATE      UFIX
glassfishv3-common                           0-1             known      u---
glassfishv3-common                           0-1             known      u---
glassfishv3-common                           0-1             installed  ----
glassfishv3-ejb                              0-1             known      u---
glassfishv3-ejb                              0-1             known      u---
glassfishv3-ejb                              0-1             known      u---
glassfishv3-ejb                              0-1             installed  u---
glassfishv3-ejb                              0-1             known      ----
glassfishv3-nucleus                          0-1             known      u---
glassfishv3-nucleus                          0-1             known      u---
glassfishv3-nucleus                          0-1             installed  ----
glassfishv3-web                              0-1             known      u---
glassfishv3-web                              0-1             known      u---
glassfishv3-web                              0-1             installed  ----
grails                                       1.0-1.0         installed  ----
javadb                                       0-1             known      u---
javadb                                       0-1             known      u---
javadb                                       0-1             known      ----
jersey                                       0.7-0.1         installed  u---
jersey                                       0.7-0.2         known      u---
jersey                                       0.7-0.3         known      u---
jersey                                       0.8-0.1         known      ----
jmaki                                        1.8.0-1.0       installed  ----
jruby                                        1.1.1-1.0       installed  ----
metro                                        1.2-1           installed  u---
metro                                        1.2-2           known      u---
metro                                        1.2-3           known      ----
pkg                                          0.1.4-6.564     installed  u---
pkg                                          0.1.5-8.724     known      ----
python2.4-minimal                            2.4.4-6.564     installed  u---
python2.4-minimal                            2.4.4-8.724     known      ----
updatetool                                   2.0-6.564       installed  u---
updatetool                                   2.0-8.724       known      ----
wxpython2.8-minimal                          2.8.7.1-6.564   installed  u---
wxpython2.8-minimal                          2.8.7.1-8.724   known      ----
```

3 **Navigate to the** *as-install* **directory.**

4 **Install a package from the available packages list.**

Use the following syntax: pkg install *package-name*. For example:

pkg install javadb

The most recent version of the component is installed and information similar to the following is displayed:

```
DOWNLOAD                                PKGS      FILES    XFER (MB)
javadb                                  0/1      61/200    2.10/7.26

PHASE                                   ACTIONS
Install Phase                           222/222
```

5 **To apply your changes, restart Enterprise Server.**

See .

See Also Full syntax and options for using the pkg command are described in the pages in the *as-install*/pkg/man/ directory. These reference pages are not displayed by using the man command. Instead, use a command such as more or cat.

Updating Installed Components

This section provides the following instructions for updating Enterprise Server components after they have been installed:

▼ To Update an Installed Component

When you install an updated version of a component, only those files that have been modified are downloaded and installed. Files that have been removed in the updated package are removed during the update process.

1 **Obtain a list of only the installed packages that have available updates:**

pkg list -u

Information similar to the following is displayed:

```
NAME (AUTHORITY)                               VERSION          STATE     UFIX
glassfishv3-ejb                                0-1              installed  u---
jersey                                         0.7-0.1          installed  u---
```

```
metro                                         1.2-1         installed  u---
pkg                                           0.1.4-6.564   installed  u---
python2.4-minimal                             2.4.4-6.564   installed  u---
updatetool (glassfish.org)                    2.0-6.564     installed  u---
wxpython2.8-minimal                           2.8.7.1-6.564 installed  u---
```

2   Install a new version of a package.

Use the following syntax: pkg install *package-name*. For example:

pkg install metro

Information similar to the following is displayed:

```
DOWNLOAD                                 PKGS     FILES    XFER (MB)
Completed                                1/1      5/5      0.49/0.49

PHASE                                    ACTIONS
Removal Phase                              2/2
Update Phase                               7/7
Install Phase                              2/2
```

3   To apply your changes, restart Enterprise Server.

See .

See Also   Full syntax and options for using the pkg command are described in the pages in the *as-install*
/pkg/man/ directory. These reference pages are not displayed by using the man command.
Instead, use a command such as more or cat.

▼ To Update All Installed Components in an Image

Enterprise Server enables you to maintain multiple installation images on a single system.
When you update an installation image, all the components that are present in that image are
updated to new versions, if new versions are available. When you install updated versions of
components, only those files that have been modified are downloaded and installed. Files that
have been removed in the updated package are removed during the update process.

1   Install all packages for the image.

Use the following syntax: pkg install *image-name*. For example:

pkg image-update

Information similar to the following is displayed:

```
DOWNLOAD                                 PKGS      FILES    XFER (MB)
Completed                                6/6       729/729  21.59/21.59
```

```
PHASE                                ACTIONS

Removal Phase                        887/887
Update Phase                         253/253
Install Phase                        584/584
```

2   To apply your changes, restart Enterprise Server.

See "To Restart a Domain" on page 90.

See Also   Full syntax and options for using the pkg command are described in the pages in the *as-install*
/pkg/man/ directory. These reference pages are not displayed by using the man command.
Instead, use a command such as more or cat.

Removing Installed Components

If you are discontinuing use of a component and want to remove it from your system, you can
do this by using the uninstall command. If you need to revert to a prior version of a
component, you will need to uninstall the current version and install the prior version by
specifying the version number.

- "To Uninstall an Installed Component" on page 176
- "To Uninstall and Revert to an Older Version of a Component" on page 177

▼ To Uninstall an Installed Component

Before You Begin   Verify that there are no dependencies on the installed component before removing it.

1   Obtain a list of all your installed components:

pkg list

```
NAME (AUTHORITY)                     VERSION           STATE      UFIX
glassfishv3-common                   0-1               installed  ----
glassfishv3-ejb                      0-1               installed  u---
glassfishv3-nucleus                  0-1               installed  ----
glassfishv3-web                      0-1               installed  ----
grails                               1.0-1.0           installed  ----
jersey                               0.7-0.1           installed  u---
jmaki                                1.8.0-1.0         installed  ----
jruby                                1.1.1-1.0         installed  ----
metro                                1.2-1             installed  u---
pkg                                  0.1.4-6.564       installed  u---
python2.4-minimal                    2.4.4-6.564       installed  u---
updatetool (glassfish.org)           2.0-6.564         installed  u---
wxpython2.8-minimal                  2.8.7.1-6.564     installed  u---
```

2   Uninstall the component that you want removed from your system.

Use the following syntax: pkg uninstall *package-name*. For example:

pkg uninstall jruby

3   To apply your changes, restart Enterprise Server.

See "To Restart a Domain" on page 90.

See Also   Full syntax and options for using the pkg command are described in the pages in the *as-install* /pkg/man/ directory. These reference pages are not displayed by using the man command. Instead, use a command such as more or cat.

▼ To Uninstall and Revert to an Older Version of a Component

If there is a malfunction in an installed component, you might want to revert to an older version of that component. The way to restore an older version of a component is to first uninstall the current version of the component, then install the specific older version that you want to reinstate.

Before You Begin   Be sure to verify that the older version of the component is in the repository before you uninstall your current version.

1   Verify that the older version of the component is still available:

pkg list -a

2   Obtain a list of your installed components:

pkg list

3   Uninstall the currently-installed component that you want to replace.

Use the following syntax: pkg uninstall *package-name*. For example:

pkg uninstall jersey

4   Install the older version of the component.

Use the following syntax: pkg install *package-name* –version *version-number*. For example:

pkg install jersey -version 0.7-0.2

5   Verify that the older version is installed:

pkg list

See Also   Full syntax and options for using the pkg command are contained in the pages in the *as-install* /pkg/man/ directory. These reference pages are not displayed by using the man command. Instead, use a command such as more or cat.

PART II

Security Administration

11

Administering System Security

The following topics are addressed here:

- "About System Security in Enterprise Server" on page 181
- "Administering Passwords" on page 190
- "Administering Audit Modules" on page 197
- "Administering JSSE Certificates " on page 199

Instructions for accomplishing many of these tasks by using the Administration Console are contained in the Administration Console online help.

Additional instructions on configuring security is contained in Chapter 12, "Administering User Security," and Chapter 13, "Administering Message Security."

Information on application security is contained in Chapter 5, "Securing Applications," in *Sun GlassFish Enterprise Server v3 Application Development Guide*.

About System Security in Enterprise Server

Security is about protecting data, that is, how to prevent unauthorized access or damage to data that is in storage or in transit. The Enterprise Server is built on the Java security model, which uses a sandbox where applications can run safely, without potential risk to systems or users. *System security* affects all the applications in the Enterprise Server environment.

System security features include the following:

- "Authentication" on page 182
- "Authorization" on page 184
- "Auditing" on page 185
- "Firewalls" on page 185
- "Certificates and SSL" on page 186
- "Tools for Managing System Security" on page 189

Authentication

Authentication is the way in which an entity (a user, an application, or a component) determines that another entity is who it claims to be. An entity uses security *credentials* to authenticate itself. The credentials might be a user name and password, a digital certificate, or something else. Usually, servers or applications require clients to authenticate themselves. Additionally, clients might require servers to authenticate themselves. When authentication is bidirectional, it is called *mutual authentication*.

When an entity tries to access a protected resource, Enterprise Server uses the authentication mechanism configured for that resource to determine whether to grant access. For example, a user can enter a user name and password in a web browser, and if the application verifies those credentials, the user is authenticated. The user is associated with this authenticated security identity for the remainder of the session.

Authentication Types

Within its deployment descriptors, an application specifies the type of authentication that it uses. Enterprise Server supports the following types of authentication:

BASIC	Uses the server's built-in login dialog box. The communication protocol is HTTP (SSL optional). There is no user-credentialed encryption unless using SSL.
FORM	The application provides its own custom login and error pages. The communication protocol is HTTP (SSL optional). There is no user-credentialed encryption unless using SSL.
CLIENT-CERT	The server authenticates the client using a public key certificate. The communication protocol is HTTPS (HTTP over SSL). User-credentialed encryption is SSL.
DIGEST	The server authenticates a user based on a user name and a password. The authentication is performed by transmitting the password in an encrypted form which is much more secure than the simple Base64 encoding used by BASIC authentication. The communication protocol is HTTPS.

Passwords

Passwords are your first line of defense against unauthorized access to the components and data of Enterprise Server. For Information about how to use passwords for Enterprise Server, see "Administering Passwords" on page 190.

Master Password and Keystores

The *master password* is an overall shared password and is the most sensitive piece of data in the system. It is never used for authentication and is never transmitted over the network. You can choose to enter the master password manually when required, or obscure it in a file.

The master password is the password for the secure keystore. When a new application server domain is created, a new self-signed certificate is generated and stored in the relevant keystore, which is locked using the master password (default password `changeit`). If the master password is not the default (that is, you have changed it), you are prompted for the master password. After the correct master password is entered, the domain starts.

Administration Password

The administration password, also known as the admin password, is used to invoke the Administration Console and the `asadmin` utility. This password is usually set during installation, but it can be changed. For instructions, see "To Change the Administration Password" on page 192.

Encoded Passwords

Files that contain encoded passwords need to be protected using file system permissions. These files include the following:

- *domain-dir*/`master-password`

  This file contains the encoded master password and should be protected with file system permissions 600.
- Any password file created to pass as an argument by using the `--passwordfile` argument to the `asadmin` utility should be protected with file system permissions 600.

For instructions, see "To Set a Password From a File" on page 193.

Password Aliases

To avoid storing passwords in the domain configuration file in clear text, you can create an alias for a password. This process is also known as *encrypting* a password. For more information, see "Administering Password Aliases" on page 193.

Single Sign-on

With *single sign-on*, a user who logs in to one application becomes implicitly logged in to other applications that require the same authentication information. Single sign-on is based on groups. All web applications whose deployment descriptor defines the same group and uses the same authentication method (BASIC, FORM, or CLIENT-CERT) share single sign-on.

On Enterprise Server, single sign-on is enabled by default for virtual servers, allowing multiple applications in one virtual server to share the user authentication state.

Authorization

Authorization, also known as access control, is the means by which users are granted permission to access data or perform operations. After a user is authenticated, the user's level of authorization determines what operations the owner can perform. A user's authorization is based on the user's role.

Roles

A *role* defines which applications and what parts of each application users can access and what those users or groups can do with the applications. For example, in a personnel application, all employees might be able to see phone numbers and email addresses, but only managers have access to salary information. This application would define at least two roles: employee and manager. Only users in the manager role are allowed to view salary information.

A role is different from a group in that a role defines a function in an application, while a group is a set of users who are related in some way. For example, the personnel application specify groups such as full-time, part-time, and on-leave. Users in these groups are all employees (the employee role). In addition, each user has its own designation that defines an additional level of employment.

Roles are defined in the deployment descriptor for the application. The application developer or deployer maps roles to one or more groups in the deployment descriptor for each application. When the application is being packaged and deployed, the application specifies mappings between users, groups, and roles, as illustrated in the following figure.
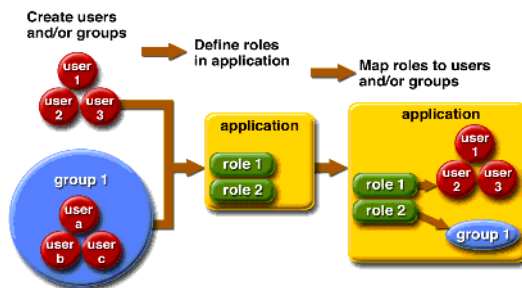
FIGURE 11–1   Role Mapping

Java Authorization Contract for Containers

Java Authorization Contract for Containers (JACC) is the part of the Java EE specification that defines an interface for pluggable authorization providers. This enables you to set up third-party plug-in modules to perform authorization. By default, the Enterprise Server provides a simple, file-based authorization engine that complies with the JACC specification. You can also specify additional third-party JACC providers.

JACC providers use the Java Authentication and Authorization Service (JAAS) APIs. JAAS enables services to authenticate and enforce access controls upon users. JAAS implements a Java technology version of the standard Pluggable Authentication Module (PAM) framework.

JSR 196 allows you to develop plugins at different layers. You can define plugins that change the way new authentication mechanism are configured, such as, `AuthConfigProvider` and `AuthConfigFactory`. You can also define new authentication mechanisms, such as `ServerAuthModule` and `ClientAuthModule`.

Auditing

Auditing is the means used to capture security-related events for the purpose of evaluating the effectiveness of security measures. Enterprise Server uses audit modules to capture audit trails of all authentication and authorization decisions. Enterprise Server provides a default audit module, as well as the ability to customize the audit modules.

For administration instructions, see "Administering Audit Modules" on page 197.

Firewalls

A *firewall* controls the flow of data between two or more networks, and manages the links between the networks. A firewall can consist of both hardware and software elements. The following guidelines pertain primarily to Enterprise Server:

- In general, firewalls should be configured so that clients can access the necessary TCP/IP ports.

  For example, if the HTTP listener is operating on port 8080, configure the firewall to allow HTTP requests on port 8080 only. Likewise, if HTTPS requests are set up for port 8081, you must configure the firewalls to allow HTTPS requests on port 8081.

- If direct Remote Method Invocations over Internet Inter-ORB Protocol (RMI-IIOP) access from the Internet to EJB modules is required, open the RMI-IIOP listener port as well.

Note – Opening the RMI-IIOP listener port is strongly discouraged because it creates security risks.

- In double firewall architecture, you must configure the outer firewall to allow for HTTP and HTTPS transactions. You must configure the inner firewall to allow the HTTP server plug-in to communicate with Enterprise Server behind the firewall.

Certificates and SSL

The following topics are addressed here:

For administration instructions, see "Administering JSSE Certificates " on page 199.

Certificates

Certificates, also called digital certificates, are electronic files that uniquely identify people and resources on the Internet. Certificates also enable secure, confidential communication between two entities. There are different kinds of certificates:

- *Personal certificates* are used by individuals.
- *Server certificates* are used to establish secure sessions between the server and clients through secure sockets layer (SSL) technology.

Certificates are based on *public key cryptography*, which uses pairs of digital keys (very long numbers) to encrypt, or encode, information so the information can be read only by its intended recipient. The recipient then decrypts (decodes) the information to read it. A *key pair* contains a public key and a private key. The owner distributes the public key and makes it available to anyone. But the owner never distributes the private key, which is always kept secret. Because the keys are mathematically related, data encrypted with one key can only be decrypted with the other key in the pair.

Certificates are issued by a trusted third party called a *Certification Authority* (CA). The CA is analogous to a passport office: it validates the certificate holder's identity and signs the certificate so that it cannot be forged or tampered with. After a CA has signed a certificate, the holder can present it as proof of identity and to establish encrypted, confidential communications. Most importantly, a certificate binds the owner's public key to the owner's identity.

In addition to the public key, a certificate typically includes information such as the following:

- The name of the holder and other identification, such as the URL of the web server using the certificate, or an individual's email address
- The name of the CA that issued the certificate
- An expiration date

Certificates are governed by the technical specifications of the X.509 format. To verify the identity of a user in the `certificate` realm, the authentication service verifies an X.509 certificate, using the common name field of the X.509 certificate as the principal name.

Certificate Chains

A *certificate chain* is a series of certificates issued by successive CA certificates, eventually ending in a root CA certificate.

Web browsers are preconfigured with a set of root CA certificates that the browser automatically trusts. Any certificates from elsewhere must come with a certificate chain to verify their validity.

When a certificate is first generated, it is a *self-signed* certificate. A self-signed certificate is one for which the issuer (signer) is the same as the subject (the entity whose public key is being authenticated by the certificate). When the owner sends a certificate signing request (CSR) to a CA, then imports the response, the self-signed certificate is replaced by a chain of certificates. At the bottom of the chain is the certificate (reply) issued by the CA authenticating the subject's public key. The next certificate in the chain is one that authenticates the CA's public key. Usually, this is a self-signed certificate (that is, a certificate from the CA authenticating its own public key) and the last certificate in the chain.

In other cases, the CA can return a chain of certificates. In this situation, the bottom certificate in the chain is the same (a certificate signed by the CA, authenticating the public key of the key entry), but the second certificate in the chain is a certificate signed by a different CA, authenticating the public key of the CA to which you sent the CSR. Then, the next certificate in the chain is a certificate authenticating the second CA's key, and so on, until a self-signed *root* certificate is reached. Each certificate in the chain (after the first) thus authenticates the public key of the signer of the previous certificate in the chain.

Certificate Files

During Enterprise Server installation, a certificate is generated in Java Secure Socket Extension (JSSE) format suitable for internal testing. By default, Enterprise Server stores its certificate information in certificate databases in the *domain-dir*/`config` directory:

Keystore file   The `key3.db` file contains Enterprise Server certificate, including its private key. The keystore file is protected with a password.

Each keystore entry has a unique alias. After installation, the Enterprise Server keystore has a single entry with an alias of s1as.

Truststore file   The cert8.db file contains the Enterprise Server trusted certificates, including public keys for other entities. For a trusted certificate, the server has confirmed that the public key in the certificate belongs to the certificate's owner. Trusted certificates generally include those of CAs.

By default, Enterprise Server is configured with a keystore and truststore that will work with the example applications and for development purposes.

Secure Sockets Layer

Secure Sockets Layer (SSL) is the most popular standard for securing Internet communications and transactions. Secure web applications use HTTPS (HTTP over SSL). The HTTPS protocol uses certificates to ensure confidential and secure communications between server and clients. In an SSL connection, both the client and the server encrypt data before sending it. Data is decrypted upon receipt.

When a Web browser (client) wants to connect to a secure site, an *SSL handshake* happens, like this:

1. The browser sends a message over the network requesting a secure session (typically, by requesting a URL that begins with https instead of http).

2. The server responds by sending its certificate (including its public key).

3. The browser verifies that the server's certificate is valid and is signed by a CA whose certificate is in the browser's database (and who is trusted). It also verifies that the CA certificate has not expired.

4. If the certificate is valid, the browser generates a one time, unique *session key* and encrypts it with the server's public key. The browser then sends the encrypted session key to the server so that they both have a copy.

5. The server decrypts the message using its private key and recovers the session key.

After the handshake, the client has verified the identity of the Web site, and only the client and the Web server have a copy of the session key. From this point forward, the client and the server use the session key to encrypt all their communications with each other. Thus, their communications are ensured to be secure.

The newest version of the SSL standard is called Transport Layer Security (TLS). The Enterprise Server supports the SSL 3.0 and the TLS 1.0 encryption protocols.

To use SSL, Enterprise Server must have a certificate for each external interface or IP address that accepts secure connections. The HTTPS service of most web servers will not run unless a certificate has been installed. For instructions on applying SSL to HTTP listeners, see "To Configure an HTTP Listener for SSL" on page 284.

Ciphers

A *cipher* is a cryptographic algorithm used for encryption or decryption. SSL and TLS protocols support a variety of ciphers used to authenticate the server and client to each other, transmit certificates, and establish session keys.

Some ciphers are stronger and more secure than others. Clients and servers can support different cipher suites. During a secure connection, the client and the server agree to use the strongest cipher that they both have enabled for communication, so it is usually sufficient to enable all ciphers.

Name-based Virtual Hosts

Using name-based virtual hosts for a secure application can be problematic. This is a design limitation of the SSL protocol itself. The SSL handshake, where the client browser accepts the server certificate, must occur before the HTTP request is accessed. As a result, the request information containing the virtual host name cannot be determined prior to authentication, and it is therefore not possible to assign multiple certificates to a single IP address.

If all virtual hosts on a single IP address need to authenticate against the same certificate, the addition of multiple virtual hosts probably will not interfere with normal SSL operations on the server. Be aware, however, that most browsers will compare the server's domain name against the domain name listed in the certificate, if any (applicable primarily to official, CA-signed certificates). If the domain names do not match, these browsers display a warning. In general, only address-based virtual hosts are commonly used with SSL in a production environment.

Tools for Managing System Security

Enterprise Server provides the following tools for managing system security:

Administration Console	The Administration Console is a browser-based utility used to configure security for the entire server. Tasks include managing certificates, users, groups, and realms, and performing other system-wide security tasks. For a general introduction to the Administration Console, see "Administration Console" on page 38.
The asadmin utility	The asadmin command-line utility performs many of the same tasks as the Administration Console. You might be able to do some things with the asadmin utility that you cannot do with the Administration Console. For a general introduction to asadmin, see "asadmin Utility" on page 39.

The keytool utility   The keytool Java 2 Platform, Standard Edition (J2SE) command-line utility is used for managing digital certificates and key pairs. For more information, see "Administering JSSE Certificates " on page 199.

The policytool utility   The policytool J2SE graphical utility is used for managing system-wide Java security policies. As an administrator, you rarely use policytool.

For more information about using keytool, policytool, and other Java security tools, see *Java 2 SDK Tools and Utilities* at http://java.sun.com/j2se/1.4.2/docs/tooldocs/tools.html#security.

Administering Passwords

There are multiple ways to administer passwords. You can rely on administrators to keep passwords secret and change the passwords regularly. You can set up files for storing passwords so that asadmin subcommands can access these files rather than having users type the commands. You can encrypt passwords by setting up aliases so that sensitive passwords are not visible in the domain.xml file.

The following topics are addressed here:

- "To Change the Master Password" on page 190
- "To Change the Administration Password" on page 192
- "To Set a Password From a File" on page 193
- "Administering Password Aliases" on page 193

▼ To Change the Master Password

The master password master gives access to the crypto store used with the domain, be that an NSS cert8.db trust store or a Java JKS keystore. This password is not tied to a UNIX user. This overall shared password is the most sensitive piece of data in your system. The master password is never used for authentication and is never transmitted over the network.

You can choose to type the password manually when required, or to obscure the password in a password file. If there is no password file, you are prompted for the master password. If there is a password file, but you want to change access to require prompting, remove the file. The default master password is changeit.

Use the change-master-password subcommand in local mode to modify the master password.

When the master password is changed, it is re-saved in the master-password keystore, which is a Java JCEKS type keystore.

Before You Begin This subcommand will not work unless the domain is stopped.

1 Stop the domain whose password you are changing.
See "To Stop a Domain" on page 89.

2 Change the master password for the domain by using the change-master-password**(1) subcommand.**
You are prompted for the old and new passwords. All dependent items are re-encrypted.

3 Start the domain.
See "To Start a Domain" on page 88.

Example 11–1 Changing the Master Password

The change-master-password subcommand is interactive in that you are prompted for the old master password as well as the new master password. This example changes the master password for domain44ps:

```
asadmin> change-master-password domain44ps
```

If you have already logged into the domain using the login login(1) subcommand, you are prompted for the new master password:

```
Please enter the new master password>
Please enter the new master password again>
```

If you are not logged into the domain, you are prompted for both the old and the new master passwords:

```
Please enter the master password again>
Please enter the new master password>
Please enter the new master password again>
```

Information similar to the following is displayed:

```
Master password changed for domain44ps
```

See Also You can also view the full syntax and options of the subcommand by typing asadmin help change-master-password at the command line.

▼ To Change the Administration Password

Use the change-admin-password subcommand in remote mode to change the administration password. The default administration password is admin. You are prompted for the old and new admin passwords, with confirmation.

Note – If you accepted the default admin user with no password during zip installation, you can add a password to this user. If there is a single user called admin that does not have a password, you are not prompted for login information. Any other situation requires login.

Encrypting the admin password is strongly encouraged.

Before You Begin   If you want to change the admin password before creating an alias for the password (encrypting), you can use the set subcommand with syntax similar to the following:

```
asadmin set --user admin server.jms-service.jms-host.default_JMS_host.admin-password=
new_pwd
```

1   Ensure that the server is running.

Remote subcommands require a running server.

2   Change the admin password by using the change-admin-password(1) **subcommand.**

3   Enter the old and new admin passwords when prompted.

Example 11–2   Changing the Admin Password

This example changes the admin password for user anonymous from adminadmin to newadmin:

```
asadmin> change-admin-password --user anonymous
```

You are prompted to enter the old and the new admin passwords:

```
Enter admin password>adminadmin
Enter new admin password>newadmin
Enter new admin password again>newadmin
```

Information similar to the following is displayed:

```
Command change-admin-password executed successfully.
```

See Also You can also view the full syntax and options of the subcommand by typing asadmin help change-admin-password at the command line.

▼ To Set a Password From a File

Instead of typing the password at the command line, you can access the password for a command from a file such as passwords.txt. The --passwordfile option of the asadmin utility takes the name of the file that contains the passwords. The entry for a password in the file must have the AS_ADMIN_ prefix followed by the password name in uppercase letters.

The following other types of passwords can be specified:

```
AS_ADMIN_MASTERPASSWORD
AS_ADMIN_USERPASSWORD
AS_ADMIN_ALIASPASSWORD
```

1 Edit the password file.

For example, to specify the password for the domain administration server (DAS), add an entry similar to the following to the password file, where adminadmin is the administrator password:

AS_ADMIN_PASSWORD=adminadmin

2 Save the password file.

You can now specify the password file in an asadmin subcommand. In This example, passwords.txt is the file that contains the password:

asadmin>**delete-jdbc-resource --user admin --password passwords.txt jdbc/DerbyPool**

Troubleshooting If AS_ADMIN_PASSWORD has been exported to the global environment, specifying the --passwordfile option will produce a warning about using the --passwordfile option. To prevent this warning situation from happening, unset AS_ADMIN_PASSWORD.

Administering Password Aliases

A password alias is used to indirectly access a password so that the password itself does not appear in the configuration file.

The following topics are addressed here:

▼ To Create a Password Alias

Use the `create-password-alias` subcommand in remote mode to create an alias for a password in the domain's keystore. The password corresponding to the alias name is stored in an encrypted form in the domain configuration file. The `create-password-alias` subcommand takes both a secure interactive form, in which users are prompted for all information, and a more script-friendly form, in which the password is propagated on the command line.

You can also use the set(1) subcommand to remove and replace the password in the configuration file. For example:

```
asadmin set --user admin server.jms-service.jms-host.default_JMS_host.
admin-password='${ALIAS=jms-password}'
```

1 Ensure that the server is running.

Remote subcommands require a running server.

2 Go to the directory where the configuration file resides.

By default, the configuration file is located in *domain-dir*/config.

3 Create the password alias by using the create-password-alias(1) **subcommand.**

4 Type the password for the alias when prompted.

5 Add the alias to a password file.

In the password file, for example, passwords.txt, add the following line: AS_ADMIN_PASSWORD=${ALIAS=*admin-password-alias*}, where *admin-password-alias* is the new password alias.

6 Stop the Enterprise Server domain.

See "To Stop a Domain" on page 89.

7 Start the domain specifying the file that contains the alias.

Use the following syntax:

```
start-domain --user admin --passwordfile /path-to/passwords.txt domain1
```

Example 11–3   Creating a Password Alias

This example creates the new jms-password alias for the admin user:

```
asadmin> create-password-alias --user admin jms-password
```

You are prompted to type the password for the alias:

```
Please enter the alias password>secret-password
Please enter the alias password again>secret-password
Command create-password-alias executed successfully.
```

See Also You can also view the full syntax and options of the subcommand by typing asadmin help create-password-alias at the command line.

▼ To List Password Aliases

Use the list-password-aliases subcommand in remote mode to list existing the password aliases.

1 Ensure that the server is running.

Remote subcommands require a running server.

2 List password aliases by using the list-password-aliases**(1) subcommand.**

Example 11–4 Listing Password Aliases

This example lists the existing password aliases:

```
asadmin> list-password aliases
jmspassword-alias
Command list-password-aliases executed successfully
```

See Also You can also view the full syntax and options of the subcommand by typing asadmin help list-password-aliases at the command line.

▼ To Delete a Password Alias

Use the delete-password-alias subcommand in remote mode to delete an existing password alias.

1 Ensure that the server is running.

Remote subcommands require a running server.

2 List all aliases by using the list-password-aliases**(1) subcommand.**

3 Delete a password alias by using the list-password-aliases**(1) subcommand.**

Example 11–5 Deleting a Password Alias

This example deletes the password alias jmspassword-alias:

```
asadmin> delete-password-alias jmspassword-alias
Command list-password-aliases executed successfully
```

See Also You can also view the full syntax and options of the subcommand by typing asadmin help delete-password-alias at the command line.

▼ To Update a Password Alias

Use the update-password-alias subcommand in remote mode to change the password for an existing password alias. The update-password-alias subcommand takes both a secure interactive form, in which the user is prompted for all information, and a more script-friendly form, in which the password is propagated on the command line.

1 Ensure that the server is running.

Remote subcommands require a running server.

2 Update an alias by using the update-password-alias(1) **subcommand.**

3 Type the password when prompted.

Example 11–6 Updating a Password Alias

This example updates the password for the jmspassword-alias alias:

```
asadmin> update-password-allias /home/password.txt jsmpassword-alias
```

You are prompted to type the new password for the alias:

```
Please enter the alias password>new-secret-password
Please enter the alias password again>new-secret-password
Command update-password-alias executed successfully
```

See Also You can also view the full syntax and options of the subcommand by typing asadmin help update-password-alias at the command line.

Administering Audit Modules

The following topics are addressed here:

▼ To Create an Audit Module

Use the `create-audit-module` subcommand in remote mode to create an audit module for the add-on component that implements the audit capabilities.

1 Ensure that the server is running.

Remote subcommands require a running server.

2 Create an audit module by using the `create-audit-module`(1) subcommand.

Information about properties for this subcommand is included in this help page.

Example 11–7   Creating an Audit Module

This example creates an audit module named `sampleAuditModule`:

```
asadmin> create-audit-module
--classname com.sun.appserv.auditmodule --property defaultuser=
admin:Password=admin sampleAuditModule
Command create-audit-module executed successfully.
```

See Also   You can also view the full syntax and options of the subcommand by typing `asadmin help create-audit-module` at the command line.

▼ To List Audit Modules

Use the `list-audit-modules` subcommand in remote mode to list the audit modules on one of the following targets:

- Server instance, `server` (the default)
- Specified server instance
- Specified configuration

1 Ensure that the server is running.

Remote subcommands require a running server.

2 List the audit modules by using the list-audit-modules**(1) subcommand.**

Example 11–8   Listing Audit Modules

This example lists the audit modules on localhost:

```
asadmin> list-audit-modules
audit-module : default
audit-module : sampleAuditModule
Command list-audit-modules executed successfully.
```

See Also   You can also view the full syntax and options of the subcommand by typing asadmin help
list-audit-modules at the command line.

▼ To Delete an Audit Module

Use the delete-audit-module subcommand in remote mode to delete an existing audit
module.

1 Ensure that the server is running.

Remote subcommands require a running server.

2 List the audit modules by using the list-audit-modules**(1) subcommand.**

3 Delete an audit module by using the delete-audit-module**(1) subcommand.**

Example 11–9   Deleting an Audit Module

This example deletes sampleAuditModule:

```
asadmin> delete-audit-module sampleAuditModule
Command delete-audit-module executed successfully.
```

Administering JSSE Certificates

In the developer profile, the Enterprise Server v3 uses the JSSE format on the server side to manage certificates and key stores. In all profiles, the client side (appclient or stand-alone) uses the JSSE format.

The J2SE SDK ships with the keytool utility, which enables you to set up and work with Java Secure Socket Extension (JSSE) digital certificates. You can administer public/private key pairs and associated certificates, and cache the public keys (in the form of certificates) of their communicating peers.

The following topics are addressed here:

- "To Generate a Certificate by Using keytool" on page 199
- "To Sign a Certificate by Using keytool" on page 201
- "To Delete a Certificate by Using keytool" on page 202

▼ To Generate a Certificate by Using keytool

By default, the keytool utility creates a keystore file in the directory where the utility is run.

Before You Begin   To run the keytool utility, your shell environment must be configured so that the J2SE /bin directory is in the path, otherwise the full path to the utility must be present on the command line.

1 **Change to the directory that contains the keystore and truststore files.**

Always generate the certificate in the directory containing the keystore and truststore files. The default is *domain-dir*/config.

2 **Generate the certificate in the keystore file,** keystore.jks**, using the following command format:**

```
keytool -genkey -alias keyAlias -keyalg RSA
 -keypass changeit
 -storepass changeit
keystore keystore.jks
```

Use any unique name as your *keyAlias*. If you have changed the keystore or private key password from the default (changeit), substitute the new password for changeit. The default key password alias is s1as.

A prompt appears that asks for your name, organization, and other information.

3   Export the generated certificate to the `server.cer` **file (or** `client.cer` **if you prefer), using the following command format:**

```
keytool -export -alias keyAlias -storepass changeit
 -file server.cer
 -keystore keystore.jks
```

4   If a certificate signed by a certificate authority is required, see "To Sign a Certificate by Using `keytool`" on page 201.

5   Create the `cacerts.jks` **truststore file and add the certificate to the truststore, using the following command format:**

```
keytool -import -v -trustcacerts
-alias keyAlias
 -file server.cer
-keystore cacerts.jks
 -keypass changeit
```

If you have changed the keystore or private key password from the default (`changeit`), substitute the new password.

Information about the certificate is displayed and a prompt appears asking if you want to trust the certificate.

6   Type `yes`**, then press Enter.**

Information similar to the following is displayed:

```
Certificate was added to keystore
[Saving cacerts.jks]
```

7   To apply your changes, restart Enterprise Server. See "To Restart a Domain" on page 90.

Example 11–10   Creating a Self-Signed Certificate in a JKS Keystore by Using an RSA Key Algorithm

RSA is public-key encryption technology developed by RSA Data Security, Inc.

```
keytool  -genkey -noprompt -trustcacerts -keyalg RSA -alias ${cert.alias}
-dname  ${dn.name} -keypass ${key.pass} -keystore ${keystore.file}
-storepass ${keystore.pass}
```

Example 11–11   Creating a Self-Signed Certificate in a JKS Keystore by Using a Default Key Algorithm

```
keytool -genkey -noprompt -trustcacerts -alias ${cert.alias} -dname
${dn.name} -keypass ${key.pass} -keystore ${keystore.file} -storepass
${keystore.pass}
```

Example 11–12   Displaying Available Certificates From a JKS Keystore

```
keytool -list -v  -keystore ${keystore.file} -storepass ${keystore.pass}
```

Example 11–13   Displaying Certificate information From a JKS Keystore

```
keytool -list -v  -alias ${cert.alias} -keystore ${keystore.file}
-storepass ${keystore.pass}
```

See Also   For more information about keytool, see the keytool documentation at
http://java.sun.com/j2se/1.5.0/docs/tooldocs/solaris/keytool.html.

▼ **To Sign a Certificate by Using** keytool

After creating a certificate, the owner must sign the certificate to prevent forgery. E-commerce
sites, or those for which authentication of identity is important, can purchase a certificate from
a well-known Certificate Authority (CA).

Note – If authentication is not a concern, for example if private secure communications are all
that is required, you can save the time and expense involved in obtaining a CA certificate by
using a self-signed certificate.

---

1   **Follow the instructions on the CA's web site for generating certificate key pairs.**

2   **Download the generated certificate key pair.**

Save the certificate in the directory containing the keystore and truststore files. The default is
domain-dir/config.

3   **In your shell, change to the directory containing the certificate.**

4   **Import the certificate into the local keystore and, if necessary, the local truststore using the
following command format:**

```
keytool -import -v -trustcacerts
-alias keyAlias
 -file server.cer
-keystore cacerts.jks
 -keypass changeit
-storepass changeit
```

If the keystore or private key password is not the default password, then substitute the new
password for the default (changeit).

5 To apply your changes, restart Enterprise Server. See

Example 11–14 Importing an RFC/Text-Formatted Certificate Into a JKS Keystore

Certificates are often stored using the printable encoding format defined by the Internet Request for Comments (RFC) 1421 standard instead of their binary encoding. This certificate format, also known as Base 64 encoding, facilitates exporting certificates to other applications by email or through some other mechanism.

```
keytool -import -noprompt -trustcacerts -alias ${cert.alias} -file
${cert.file} -keystore ${keystore.file} -storepass ${keystore.pass}
```

Example 11–15 Exporting a Certificate From a JKS Keystore in PKCS7 Format

The reply format defined by the Public Key Cryptography Standards #7, Cryptographic Message Syntax Standard, includes the supporting certificate chain in addition to the issued certificate.

```
keytool -export -noprompt -alias ${cert.alias} -file ${cert.file}
-keystore ${keystore.file} -storepass ${keystore.pass}
```

Example 11–16 Exporting a Certificate From a JKS Keystore in RFC/Text Format

```
keytool -export -noprompt -rfc -alias ${cert.alias} -file
${cert.file} -keystore ${keystore.file} -storepass ${keystore.pass}
```

See Also For more information about keytool, see the keytool documentation at
http://java.sun.com/j2se/1.5.0/docs/tooldocs/solaris/keytool.html

▼ To Delete a Certificate by Using keytool

Use the keytool -delete command to delete an existing certificate.

● **Delete a certificate using the following command format:**

```
keytool -delete
 -alias keyAlias
 -keystore keystore-name
 -storepass password
```

Example 11–17   Deleting a Certificate From a JKS Keystore

```
keytool -delete -noprompt -alias ${cert.alias}  -keystore ${keystore.file}
-storepass ${keystore.pass}
```

See Also   For more information about keytool, see the keytool documentation at
http://java.sun.com/j2se/1.5.0/docs/tooldocs/solaris/keytool.html.

Administering User Security

This chapter provides instructions for administering user security in the Sun GlassFish™ Enterprise Server environment by using the asadmin command-line utility. Enterprise Server enforces its authentication and authorization policies upon realms, users, and groups. This chapter assumes that you are familiar with security features such as authentication, authorization, and certificates. If you are not, see Chapter 11, "Administering System Security."

The following topics are addressed here:

- "Administering Authentication Realms" on page 205
- "Administering File Users" on page 210

Instructions for accomplishing these tasks by using the Administration Console are contained in the Administration Console online help.

Administering Authentication Realms

An *authentication realm*, also called a security policy domain or security domain, is a scope over which the Enterprise Server defines and enforces a common security policy. Enterprise Server is preconfigured with the file, certificate, and administration realms. In addition, you can set up LDAP, JDBC, digest, Solaris, or custom realms. An application can specify which realm to use in its deployment descriptor. If the application does not specify a realm, Enterprise Server uses its default realm (file).

File realm	Enterprise Server stores user credentials locally in a file named keyfile. The file realm is the initial default realm.
Administration realm	The administration realm is also a file realm and stores administrator user credentials locally in a file named admin-keyfile.
Certificate realm	Enterprise Server stores user credentials in a certificate database. When using the certificate realm, the server uses certificates with the HTTPS protocol to authenticate web clients.

LDAP realm	Enterprise Server gets user credentials from a Lightweight Directory Access Protocol (LDAP) server such as the Directory Server. LDAP is a protocol for enabling anyone to locate organizations, individuals, and other resources such as files and devices in a network, whether on the public Internet or on a corporate intranet. Consult your LDAP server documentation for information about managing users and groups in the LDAP realm.
JDBC realm	Enterprise Server gets user credentials from a database. The server uses the database information and the enabled JDBC realm option in the configuration file.
Digest realm	Digest Authentication authenticates a user based on a user name and a password. However, the authentication is performed by transmitting the password in an encrypted form.
Solaris realm	Enterprise Server gets user credentials from the Solaris operating system. This realm is supported on the Solaris 9 and Solaris 10 operating systems. Consult your Solaris documentation for information about managing users and groups in the Solaris realm.
Custom realm	You can create other repositories for user credentials, such as a relational database or third-party components. For more information about custom realms, see the Administration Console online help. For instructions on creating a custom realm, see "Creating a Custom Realm" in *Sun GlassFish Enterprise Server v3 Application Development Guide*.

The Enterprise Server authentication service can govern users in multiple realms.

The following tasks and information are used to administer authentication realms:

▼ To Create an Authentication Realm

Use the `create-auth-realm` subcommand in remote mode to create an authentication realm.

1 Ensure that the server is running.

Remote subcommands require a running server.

2 Create a realm by using the `create-auth-realm`**(1) subcommand.**

Information about properties for this subcommand is included in this help page.

Example 12–1 Creating a Realm

This example creates a realm named db.

```
asadmin> create-auth-realm --classname com.iplanet.ias.security.
auth.realm.DB.Database --property defaultuser=admin:Password=admin db
Command create-auth-realm executed successfully.
```

See Also You can also view the full syntax and options of the subcommand by typing asadmin help create-auth-realm at the command line.

For information on creating a custom realm, see "Creating a Custom Realm" in *Sun GlassFish Enterprise Server v3 Application Development Guide*.

▼ **To List Authentication Realms**

Use the list-auth-realms subcommand in remote mode to list the existing authentication realms.

1 Ensure that the server is running.

Remote subcommands require a running server.

2 List realms by using the `list-auth-realms`**(1) subcommand.**

Example 12–2 Listing Realms

This example lists the authentication realms on localhost.

```
asadmin> list-auth-realms
db
certificate
file
admin-realm
Command list-auth-realms executed successfully.
```

See Also You can also view the full syntax and options of the subcommand by typing asadmin help list-auth-realms at the command line.

▼ To Update an Authentication Realm

Use the set subcommand to modify an existing authentication realm.

Note – A custom realm does not require server restart.

---

1   List realms by using the list-auth-realms**(1) subcommand.**

2   Modify the values for the specified thread pool by using the set**(1) subcommand.**
The thread pool is identified by its dotted name.

3   To apply your changes, restart Enterprise Server.
See "To Restart a Domain" on page 90.

▼ To Delete an Authentication Realm

Use the delete-auth-realm subcommand in remote mode to delete an existing authentication realm.

1   Ensure that the server is running.
Remote subcommands require a running server.

2   List realms by using the list-auth-realms**(1) subcommand.**

3   If necessary, notify users that the realm is being deleted.

4   Delete the realm by using the delete-auth-realm**(1) subcommand.**

5   To apply your changes, restart Enterprise Server. See "To Restart a Domain" on page 90.

Example 12–3   Deleting a Realm

This example deletes an authentication realm named db.

```
asadmin> delete-auth-realm db
Command delete-auth-realm executed successfully.
```

See Also   You can also view the full syntax and options of the subcommand by typing asadmin help delete-auth-realm at the command line.

▼ To Configure a JDBC or Digest Authentication Realm

Enterprise Server enables you to specify a user's credentials (user name and password) in the JDBC realm instead of in the connection pool. Using the jdbc type realm instead of the connection pool prevents other applications from browsing the database tables for user credentials.

Note – By default, storage of passwords as clear text is not supported in the JDBC realm. Under normal circumstances, passwords should not be stored as clear text.

---

1 Create the database tables in which to store user credentials for the realm.

How you create the database tables depends on the database that you are using.

2 Add user credentials to the database tables that you created.

How you add user credentials to the database tables depends on the database that you are using.

3 Create a JDBC connection pool for the database.

See "To Create a JDBC Connection Pool" on page 236.

4 Create a JDBC resource for the database.

"To Create a JDBC Resource" on page 240

5 Create a realm.

For instructions, see "To Create an Authentication Realm" on page 206.

Note – The JAAS context should be jdbcDigestRealm for digest authentication or jdbcRealm for other authentication types.

---

6 Modify the deployment descriptor to specify the jdbc **realm.**

Modify the deployment descriptor that is associated with your application.

- **For an enterprise application in an Enterprise Archive (EAR) file, modify the** sun-application.xml **file.**

- **For a web application in a Web Application Archive (WAR) file, modify the** web.xml **file.**

- **For an enterprise bean in an EJB JAR file, modify the** sun-ejb-jar.xml **file.**

For more information about how to specify a realm, see "How to Configure a Realm" in *Sun GlassFish Enterprise Server v3 Application Development Guide*.

7   Assign security roles to users in the realm.

To assign a security role to a user, add a `security-role-mapping` element to the deployment descriptor that you modified.

8   Verify that the database is running.

If needed, see

9   To apply the authentication, restart the server.

See .

Example 12–4   Assigning a Security Role

This example shows a `security-role-mapping` element that assigns the security role `Employee` to user `Calvin`

```
<security-role-mapping>
    <role-name>Employee</role-name>
    <principal-name>Calvin</principal-name>
  </security-role-mapping>
```

Administering File Users

A *user* is an individual (or application program) identity that is defined in Enterprise Server. A user who has been authenticated is sometimes called a *principal*.

As the administrator, you are responsible for integrating users into the Enterprise Server environment so that their credentials are securely established and they are provided with access to the applications and services that they are entitled to use.

The following tasks are used to manage users:

-
-
-
-
-

▼ To Create a File User

Use the create-file-user subcommand in remote mode to create a new user by adding a new entry to the keyfile. The entry includes the user name, password, and any groups for the user. Multiple groups can be specified by separating the groups with colons (:).

Creating a new file realm user is a dynamic event and does not require server restart.

1 Ensure that the server is running.

Remote subcommands require a running server.

2 If the user will belong to a particular group, see the current groups by using the list-file-groups(1) subcommand.

3 Create a file user by using the create-file-user(1) subcommand.

Example 12–5   Creating a User

This example create user Jennifer on the default realm file (no groups are specified).

```
asadmin> create-file-user --user admin
--passwordfile=c:\tmp\asadminpassword.txt Jennifer
Command create-file-user executed successfully.
```

See Also   You can also view the full syntax and options of the subcommand by typing asadmin help create-file-user at the command line.

▼ To List File Users

Use the list-file-users subcommand in remote mode to list the users that are in the keyfile.

1 Ensure that the server is running.

Remote subcommands require a running server.

2 List users by using the list-file-users(1) subcommand.

Example 12–6   Listing File Users

This example lists file users on the default file realm file.

```
asadmin> list-file-users
Jennifer
Command list-file-users executed successfully.
```

See Also   You can also view the full syntax and options of the subcommand by typing asadmin help
list-file-users at the command line.

▼ To List File Groups

A *group* is a category of users classified by common traits, such as job title or customer profile.
For example, users of an e-commerce application might belong to the customer group, and the
big spenders might also belong to the preferred group. Categorizing users into groups makes it
easier to control the access of large numbers of users. A group is defined for an entire server and
realm. A user can be associated with multiple groups of users.

A group is different from a role in that a role defines a function in an application, while a group
is a set of users who are related in some way. For example, in the personnel application there
might be groups such as full-time, part-time, and on-leave. Users in these groups are all
employees (the employee role). In addition, each user has its own designation that defines an
additional level of employment.

Use the list-file-groups subcommand in remote mode to list groups for a file user, or all file
groups if the --name option is not specified.

1   Ensure that the server is running.

Remote subcommands require a running server.

2   List file groups by using the list-file-groups**(1) subcommand.**

Example 12–7   Listing Groups for a User

This example lists the groups for user joesmith.

```
asadmin> list-file-groups --name joesmith
staff
manager
Command list-file-groups executed successfully
```

▼ To Update a File User

Use the update-file-user subcommand in remote mode to modify the information in the keyfile for a specified user.

1 Ensure that the server is running.

Remote subcommands require a running server.

2 Update the user information by using the update-file-user**(1) subcommand.**

3 To apply your changes, restart Enterprise Server.

See "To Restart a Domain" on page 90.

Example 12–8 Updating a User

The following subcommand updates the groups for user Jennifer.

```
asadmin> update-file-user --passwordfile c:\tmp\asadminpassword.txt --groups
staff:manager:engineer Jennifer
Command update-file-user executed successfully.
```

See Also You can also view the full syntax and options of the subcommand by typing asadmin help update-file-user at the command line.

▼ To Delete a File User

Use the delete-file-user subcommand in remote mode to remove a user entry from the keyfile by specifying the user name. You cannot delete yourself, that is, the user you are logged in as cannot be deleted during your session.

1 Ensure that the server is running.

Remote subcommands require a running server.

2 List users by using the list-file-users**(1) subcommand.**

3 Delete the user by using the delete-file-user**(1) subcommand.**

Example 12–9 Deleting a User

This example deletes user Jennifer from the default file realm.

```
asadmin> delete-file-user Jennifer
Command delete-file-user executed successfully.
```

See Also   You can also view the full syntax and options of the subcommand by typing asadmin help
delete-file-user at the command line.

13

Administering Message Security

This chapter provides information and procedures on configuring the message layer security for web services in the Sun GlassFish™ Enterprise Server environment.

Note – Message security (JSR 196) is supported only in the Full Platform Profile of Enterprise Server, not in the Web Profile.

The following topics are addressed here:

- "About Message Security in Enterprise Server" on page 215
- "Enabling Default Message Security Providers for Web Services" on page 221
- "Configuring Message Protection Policies" on page 222
- "Administering Non-default Message Security Providers" on page 226
- "Enabling Message Security for Application Clients" on page 228
- "Additional Information About Message Security" on page 228

Some of the material in this chapter assumes a basic understanding of security and web services concepts. For more information about security, see "About System Security in Enterprise Server" on page 181.

Instructions for accomplishing the tasks in this chapter by using the Administration Console are contained in the Administration Console online help.

About Message Security in Enterprise Server

Message security enables a server to perform end-to-end authentication of web service invocations and responses at the message layer. Security information is inserted into messages so that it travels through the networking layers and arrives with the intact message at the message destination(s). Message security differs from transport layer security in that message security can be used to decouple message protection from message transport so that messages remain protected after transmission.

Web services deployed on Enterprise Server are secured by binding SOAP layer message security providers and message protection policies to the containers in which the applications are deployed, or to web service endpoints served by the applications. SOAP layer message security functionality is configured in the client-side containers of Enterprise Server by binding SOAP layer message security providers and message protection policies to the client containers or to the portable service references declared by client applications.

Message-level security can be configured for the entire Enterprise Server or for specific applications or methods. Configuring message security at the application level is discussed in the *Sun GlassFish Enterprise Server v3 Application Development Guide*.

The following topics are addressed here:

- "Security Tokens and Security Mechanisms" on page 216
- "Authentication Providers" on page 217
- "Message Protection Policies" on page 218
- "Application-Specific Web Services Security" on page 218
- "Message Security Administration " on page 219
- "Sample Application for Web Services " on page 220

Security Tokens and Security Mechanisms

WS-Security is a specification that provides a communications protocol for applying security to web services. The security mechanisms implement the specification. Web Services Interoperability Technologies (WSIT) implements WS-Security so as to provide interoperable message content integrity and confidentiality, even when messages pass through intermediary nodes before reaching their destination endpoint. WS-Security as provided by WSIT is in addition to existing transport-level security, which can still be used. The Simple Object Access Protocol (SOAP) layer message security providers installed with Enterprise Server can be used to employ username/password and X.509 certificate security tokens to authenticate and encrypt SOAP web services messages.

- **Username Tokens.** Enterprise Server uses username tokens in SOAP messages to authenticate the message sender. The recipient of a message containing a username token (within embedded password) validates that the message sender is authorized to act as the user (identified in the token) by confirming that the sender knows the password of the user.

  When using a username token, a valid user database must be configured on Enterprise Server.

- **Digital Signatures.** Enterprise Server uses XML digital signatures to bind an authentication identity to message content. Clients use digital signatures to establish their caller identity. Digital signatures are verified by the message receiver to authenticate the source of the message content (which might be different from the sender of the message.)

  When using digital signatures, valid keystore and truststore files must be configured on Enterprise Server.

- **Encryption.** The purpose of encryption is to modify the data so that it can only be understood by its intended audience. This is accomplished by substituting an encrypted element for the original content. When based on public key cryptography, encryption can be used to establish the identity of the parties who are authorized to read a message.

  When using encryption, a Java Cryptography Extension (JCE) provider that supports encryption must be installed.

Authentication Providers

The *authentication layer* is the message layer on which authentication processing must be performed. Enterprise Server enforces web services message security at the SOAP layer. The types of authentication that are supported include the following:

- Sender authentication, including username-password authentication
- Content authentication, including XML digital signatures

Enterprise Server invokes *authentication providers* to process SOAP message layer security. The message security providers provide information such as the type of authentication that is required for the request and response messages. The following message security providers are included with Enterprise Server:

- **Client-side Provider.** A client-side provider establishes (by signature or username/password) the source identity of request messages and/or protects (by encryption) request messages such that they can only be viewed by their intended recipients. A client-side provider also establishes its container as an authorized recipient of a received response (by successfully decrypting it) and validates passwords or signatures in the response to authenticate the source identity associated with the response. Client-side providers configured in Enterprise Server can be used to protect the request messages sent and the response messages received by server-side components (servlets and EJB components) acting as clients of other services.

  The *default client provider* is used to identify the client—side provider to be invoked for any application for which a specific client provider has not been bound.

- **Server-side Provider.** A server-side provider establishes its container as an authorized recipient of a received request (by successfully decrypting it), and validates passwords or signatures in the request to authenticate the source identity associated with the request. A server-side provider also establishes (by signature or username/password) the source identity of response messages and/or protects (by encryption) response messages such that they can only be viewed by their intended recipients. Server-side providers are only invoked by server-side containers.

The *default server provider* is used to identify the server—side provider to be invoked for any application for which a specific server provider has not been bound.

Message Protection Policies

A *request policy* defines the authentication policy requirements associated with request processing performed by the authentication provider. Policies are expressed in message sender order such that a requirement that encryption occur after content would mean that the message receiver would expect to decrypt the message before validating the signature. The *response policy* defines the authentication policy requirements associated with response processing performed by the authentication provider.

Message protection policies are defined for request message processing and response message processing. The policies are expressed in terms of requirements for source and/or recipient authentication. The providers apply specific message security mechanisms to cause the message protection policies to be realized in the context of SOAP web services messages.

- **Source Authentication Policy.** A source authentication policy represents a requirement that the identity of the entity that sent a message or that defined the content of a message be established in the message such that it can be authenticated by the message receiver.

- **Recipient Authentication Policy.** A recipient authentication policy represents a requirement that the message be sent such that the identity of the entities that can receive the message can be established by the message sender.

Request and response message protection policies are defined when a security provider is configured into a container. Application-specific message protection policies (at the granularity of the web service port or operation) can also be configured within the Sun-specific deployment descriptors of the application or application client. In any situation where message protection policies are defined, the request and response message protection policies of the client must be equivalent t) the request and response message protection policies of the server. For more information about defining application-specific message protection policies, see Chapter 5, "Securing Applications," in *Sun GlassFish Enterprise Server v3 Application Development Guide*

Application-Specific Web Services Security

Application-specific web services security functionality is configured (at application assembly) by defining the `message-security-binding` elements in the Sun-specific deployment descriptors of the application. These `message-security-binding` elements are used to associate a specific security provider or message protection policy with a web service endpoint or service reference, and might be qualified so that they apply to a specific port or method of the corresponding endpoint or referenced service.

For information about defining application-specific message protection policies, see Chapter 5, "Securing Applications," in *Sun GlassFish Enterprise Server v3 Application Development Guide*.

Message Security Administration

When Enterprise Server is installed, SOAP layer message security providers are configured in the client and server-side containers of Enterprise Server, where they are available for binding for use by the containers, or by individual applications or clients deployed in the containers. During installation, the default providers are configured with a simple message protection policy that, if bound to a container, or to an application or client in a container, would cause the source of the content in all request and response messages to be authenticated by XML digital signature.

Enterprise Server administrative interfaces can be used as follows:

- To modify the message protection policies enforced by the providers
- To bind the existing providers for use by the server-side containers of Enterprise Server
- To create new security provider configurations with alternative message protection policies

Analogous administrative operations can be performed on the SOAP message layer security configuration of the application client container. If you want web services security to protect all web services applications deployed on Enterprise Server. See "Enabling Message Security for Application Clients" on page 228.

By default, message layer security is disabled on Enterprise Server. To configure message layer security for the Enterprise Server see "Enabling Default Message Security Providers for Web Services" on page 221.

In most cases, you must restart Enterprise Server after performing administrative tasks. This is especially true if you want the effects of the administrative change to be applied to applications that were already deployed on Enterprise Server at the time the operation was performed.

Message Security Tasks

The general implementation tasks for message security include some or all of the following:

1. If you are using a version of the Java SDK prior to version 1.5.0, and using encryption technology, configuring a JCE provider

2. If you are using a username token, verifying that a user database is configured for an appropriate realm

   When using a username/password token, an appropriate realm must be configured and a user database must be configured for the realm.

3. Managing certificates and private keys, if necessary

4. Enabling the Enterprise Server default providers

5. Configuring new message security providers

Message Security Roles

In Enterprise Server, the administrator and the application deployer are expected to take primary responsibility for configuring message security. In some situations, the application developer might also contribute.

System Administrator

The system administrator is responsible for the following message security tasks:

- Administering server security settings and certificate databases
- Administering keystore and truststore files
- Configuring message security providers on Enterprise Server
- Turning on message security
- (If needed) Installing the samples server

Application Deployer

The application deployer is responsible for the following message security tasks:

- Specifying (at application reassembly) any required application-specific message protection policies if such policies have not already been specified by the developer/assembler.
- Modifying Sun-specific deployment descriptors to specify application-specific message protection policies information (message-security-binding elements) to web service endpoint and service references.

Application Developer/Assembler

The application developer/assembler is responsible for the following message security tasks:

- Determining if an application-specific message protection policy is required by the application

  If so, the developer ensures that the required policy is specified at application assembly time.

- Specifying how web services should be set up for message security

  Message security can be set up by the administrator so that all web services are secured, or by the application deployer when the security provider or protection policy bound to the application must be different from that bound to the container.

- Turning on message security if authorized to do so by the administrator

Sample Application for Web Services

Enterprise Server includes a sample application named xms. The xms application features a simple web service that is implemented by both a Java EE EJB endpoint and a Java servlet endpoint. Both endpoints share the same service endpoint interface. The service endpoint

interface defines a single operation, sayHello, which takes a string argument, and returns a String composed by pre-pending Hello to the invocation argument.

The xms sample application is provided to demonstrate the use of Enterprise Server WS-Security functionality to secure an existing web services application. The instructions which accompany the sample describe how to enable the WS-Security functionality of Enterprise Server such that it is used to secure the xms application. The sample also demonstrates the binding of WS-Security functionality directly to the application as described in "Application-Specific Web Services Security" on page 218 application.

For information about compiling, packaging, and running the xms sample application, Chapter 5, "Securing Applications," in *Sun GlassFish Enterprise Server v3 Application Development Guide*.

The xms sample application is installed in the following directory:
as-install/samples/webservices/security/ejb/apps/xms/

Enabling Default Message Security Providers for Web Services

By default, message security is disabled on Enterprise Server. Default message security providers have been created, but are not active until you enable them. After the providers have been enabled, message security is enabled.

The following topics are addressed here:

- "To Enable a Default Server Provider" on page 221
- "To Enable a Default Client Provider" on page 222

▼ To Enable a Default Server Provider

To enable message security for web services endpoints deployed in Enterprise Server, you must specify a security provider to be used by default on the server side. If you enable a default provider for message security, you also need to enable providers to be used by clients of the web services deployed in Enterprise Server.

1 Specify the default server provider by using the set(1) **subcommand.**

Use the following syntax:

```
asadmin  set --port admin-port
server-config.security-service.message-security-config.SOAP.
default_provider=ServerProvider
```

2 To apply your changes to applications that are already running, restart Enterprise Server.

See "To Restart a Domain" on page 90.

▼ To Enable a Default Client Provider

To enable message security for web service invocations originating from deployed endpoints, you must specify a default client provider. If you enabled a default client provider for Enterprise Server, you must ensure that any services invoked from endpoints deployed in Enterprise Server are compatibly configured for message layer security.

1 Specify the default client provider by using the set(1) subcommand.

Use the following syntax:

```
asadmin  set --port admin-port
server-config.security-service.message-security-config.SOAP.
default_client_provider=ClientProvider
```

2 To apply your changes to applications that are already running, restart Enterprise Server.

See "To Restart a Domain" on page 90.

Configuring Message Protection Policies

Message protection policies are defined for request message processing and response message processing. The policies are expressed in terms of requirements for source and/or recipient authentication. The providers apply specific message security mechanisms to cause the message protection policies to be realized in the context of SOAP web services messages.

The following topics are addressed here:

- "Message Protection Policy Mapping" on page 222
- "To Configure the Message Protection Policies for a Provider" on page 224
- "Setting the Request and Response Policy for the Application Client Configuration" on page 224

Message Protection Policy Mapping

The following table shows message protection policy configurations and the resulting message security operations performed by the WS-Security SOAP message security providers for that configuration.

TABLE 13–1   Message Protection Policy Mapping to WS-Security SOAP Operations

Message Protection Policy	Resulting WS-Security SOAP message protection operations
auth-source="sender"	The message contains a `wsse:Security` header that contains a `wsse:UsernameToken` (with password).

TABLE 13–1 Message Protection Policy Mapping to WS-Security SOAP Operations   *(Continued)*

Message Protection Policy	Resulting WS-Security SOAP message protection operations
auth-source="content"	The content of the SOAP message Body is signed. The message contains a `wsse:Security` header that contains the message Body signature represented as a `ds:Signature`.
auth-source="sender" *auth-recipient="before-content"* OR *auth-recipient="after-content"*	The content of the SOAP message Body is encrypted and replaced with the resulting `xend:EncryptedData`. The message contains a `wsse:Security` header that contains a `wsse:UsernameToken` (with password) and an `xenc:EncryptedKey`. The `xenc:EncryptedKey` contains the key used to encrypt the SOAP message body. The key is encrypted in the public key of the recipient.
auth-source="content" *auth-recipient="before-content"*	The content of the SOAP message Body is encrypted and replaced with the resulting `xend:EncryptedData`. The `xenc:EncryptedData` is signed. The message contains a `wsse:Security` header that contains an `xenc:EncryptedKey` and a `ds:Signature`. The `xenc:EncryptedKey` contains the key used to encrypt the SOAP message body. The key is encrypted in the public key of the recipient.
auth-source="content" *auth-recipient="after-content"*	The content of the SOAP message Body is signed, then encrypted, and then replaced with the resulting `xend:EncryptedData`. The message contains a `wsse:Security` header that contains an `xenc:EncryptedKey` and a `ds:Signature`. The `xenc:EncryptedKey` contains the key used to encrypt the SOAP message body. The key is encrypted in the public key of the recipient.
auth-recipient="before-content" OR *auth-recipient="after-content"*	The content of the SOAP message Body is encrypted and replaced with the resulting `xend:EncryptedData`. The message contains a `wsse:Security` header that contains an `xenc:EncryptedKey`. The `xenc:EncryptedKey` contains the key used to encrypt the SOAP message body. The key is encrypted in the public key of the recipient.
No policy specified.	No security operations are performed by the modules.

▼ To Configure the Message Protection Policies for a Provider

Typically, you would not reconfigure a provider. However, if needed for your situation, you can modify a provider's message protection policies by changing provider type, implementation class, and provider-specific configuration properties. To understand the results of different combinations, see Table 13–1.

Use the set(1) subcommand to set the response policy, then replace the word request in the following commands with the word response.

1 **Add a request policy to the client and set the authentication source by using the** set**(1)** **subcommand.**

For example:

```
asadmin> set server-config.security-service.message-security-config.SOAP.
provider-config.ClientProvider.request-policy.auth_source=[sender | content]
```

2 **Add a request policy to the server and set the authentication source by using the** set **subcommand.**

For example:

```
asadmin> set server-config.security-service.message-security-config.SOAP.
provider-config.ServerProvider.request-policy.auth_source=[sender | content]
```

3 **Add a request policy to the client and set the authentication recipient by using the** set **subcommand:**

For example:

```
asadmin> set server-config.security-service.message-security-config.SOAP.
provider-config.ClientProvider.request-policy.auth_recipient=[before-content | after-content]
```

4 **Add a request policy to the server and set the authentication recipient by using the** set **subcommand:**

For example:

```
asadmin> set server-config.security-service.message-security-config.SOAP.
provider-config.ServerProvider.request-policy.auth_recipient=[before-content | after-content]
```

Setting the Request and Response Policy for the Application Client Configuration

The request and response policies define the authentication policy requirements associated with request and response processing performed by the authentication provider. Policies are

expressed in message sender order such that a requirement that encryption occur after content would mean that the message receiver would expect to decrypt the message before validating the signature.

To achieve message security, the request and response policies must be enabled on both the server and client. When configuring the policies on the client and server, make sure that the client policy matches the server policy for request/response protection at application-level message binding.

To set the request policy for the application client configuration, modify the Enterprise Server–specific configuration for the application client container as described in "Enabling Message Security for Application Clients" on page 228.

EXAMPLE 13–1 Message Security Policy Setting for Application Clients

In the application client configuration file, the `request-policy` and `response-policy` elements are used to set the request policy, as shown in the following code snippet. (Additional code in the snippet is provided as illustration and might differ slightly in your installation. Do not change the additional code.)

```
<client-container>
  <target-server name="your-host" address="your-host"
      port="your-port"/>
  <log-service file="" level="WARNING"/>
  <message-security-config auth-layer="SOAP"
      default-client-provider="ClientProvider">
    <provider-config
        class-name="com.sun.enterprise.security.jauth.ClientAuthModule"
        provider-id="ClientProvider" provider-type="client">
      <request-policy auth-source="sender | content"
        auth-recipient="after-content | before-content"/>
      <response-policy auth-source="sender | content"
        auth-recipient="after-content | before-content"/>
       <property name="security.config"
            value="as-install/lib/appclient/wss-client-config.xml"/>
    </provider-config>
  </message-security-config>
</client-container>
```

Valid values for `auth-source` include `sender` and `content`. Valid values for `auth-recipient` include `before-content` and `after-content`. A table describing the results of various combinations of these values can be found in "Configuring Message Protection Policies" on page 222.

To not specify a request or response policy, leave the element blank, for example:

```
<response-policy/>
```

Administering Non-default Message Security Providers

The following topics are addressed here:

▼ To Create a Message Security Provider

Use the create–message–security–provider subcommand in remote mode to create a new message provider for the security service. If the message layer does not exist, the message layer is created, and the provider is created under it.

1 Ensure that the server is running.

Remote subcommands require a running server.

2 Create the message security provider by using the create-message-security-provider(1) **subcommand.**

Information about properties for this subcommand is included in this help page.

3 (Optional) If needed, restart the server.

Some properties require server restart. See "Configuration Changes That Require Server Restart" on page 37. If your server needs to be restarted, see "To Restart a Domain" on page 90.

Example 13–2 Creating a Message Security Provider

This example creates the new message security provider mySecurityProvider.

```
asadmin> create-message-security-provider
--classname com.sun.enterprise.security.jauth.ClientAuthModule
--providertype client mySecurityProvider
Command create-message-security-provider executed successfully.
```

See Also You can also view the full syntax and options of the subcommand by typing asadmin help create–message–security–provider at the command line.

▼ To List Message Security Providers

Use the list−message−security−providers subcommand in remote mode to list the message providers for the security layer.

1 Ensure that the server is running.

Remote subcommands require a running server.

2 List the message security providers by using the list-message-security-providers**(1) subcommand.**

Example 13–3 Listing Message Security Providers

This example lists the message security providers for a message layer.

```
asadmin> list-message-security-providers --layer SOAP
XWS_ClientProvider
ClientProvider
XWS_ServerProvider
ServerProvider
Command list-message-security-providers executed successfully.
```

See Also You can also view the full syntax and options of the subcommand by typing asadmin help list−message−security−providers at the command line.

▼ To Update a Message Security Provider

1 Ensure that the server is running.
Remote subcommands require a running server.

2 List the message security providers by using the list-message-security-providers**(1) subcommand.**

3 Modify the values for the specified message security provider by using the set**(1) subcommand.**
The message security provider is identified by its dotted name.

▼ To Delete a Message Security Provider

Use the delete-message-security-provider subcommand in remote mode to remove

1 Ensure that the server is running.
Remote subcommands require a running server.

2 **List the message security providers by using the** `list-message-security-providers`**(1)** **subcommand.**

3 **Delete the message security provider by using the** `delete-message-security-provider`**(1)** **subcommand.**

Example 13–4 Deleting a Message Security Provider

This example deletes the `myServerityProvider` message security provider.

```
asadmin> delete-message-security-provider --layer SOAP myServerityProvider
Command delete-message-security-provider executed successfully.
```

See Also You can also view the full syntax and options of the subcommand by typing `asadmin help delete-message-security-provider` at the command line.

Enabling Message Security for Application Clients

The message protection policies of client providers must be configured such that they are equivalent to the message protection policies of the server-side providers they will be interacting with. This is already the situation for the providers configured (but not enabled) when Enterprise Server is installed.

To enable message security for client applications, modify the Enterprise Server specific configuration for the application client container. The process is analogous to the process in "Configuring Message Protection Policies" on page 222.

Additional Information About Message Security

Additional information associated with message security:

- *Java EE 6.0 Tutorial* chapter titled *Security* http://java.sun.com/javaee/6/docs/tutorial/doc/index.html.
- Chapter 5, "Securing Applications," in *Sun GlassFish Enterprise Server v3 Application Development Guide*

Resources and Services Administration

14

Administering Database Connectivity

This chapter provides procedures for performing database connectivity tasks in the Sun GlassFish™ Enterprise Server v3 environment by using the `asadmin` command-line utility.

The following topics are addressed here:

- "About Database Connectivity" on page 231
- "Setting Up the Database" on page 232
- "Configuring Access to the Database" on page 235
- "Configuration Specifics for JDBC Drivers" on page 243

Instructions for accomplishing these tasks by using the Administration Console are contained in the Administration Console online help.

About Database Connectivity

A database management system (DBMS) provides facilities for storing, organizing, and retrieving data. The information in databases is often described as persistent data because it is saved on disk and exists after the application process ends. Most business applications store data in relational databases. Applications can access database information by using the Java Database Connectivity (JDBC) API.

The key elements of database connectivity are the following:

- **Database.** The repository where data is stored for an enterprise. Java EE applications access relational databases through the JDBC API. For administration procedures, see "Setting Up the Database" on page 232.

- **JDBC Connection Pool.** A JDBC connection pool is a group of reusable connections for a particular database. For administration procedures, see "Administering JDBC Connection Pools" on page 236.

- **JDBC Resource.** A JDBC resource (data source) provides applications with a means of connecting to a database. To create a JDBC resource, specify the connection pool with which it is associated. Multiple JDBC resources can specify a single connection pool. A JDBC resource is identified by its Java Naming and Directory Interface (JNDI) name. For administration procedures, see "Administering JDBC Resources" on page 240.

- **JDBC Driver.** A database driver is a software component that enables a Java application to interact with a database connectivity API . Each database requires its own driver. For administration procedures, see "Integrating the JDBC Driver" on page 242.

At runtime, the following sequence occurs when an application connects to a database:

1. The application gets the JDBC resource associated with the database by making a call through the JNDI API.

   Using the JNDI name of the resource, the naming and directory service locates the JDBC resource. Each JDBC resource specifies a connection pool.

2. Using the JDBC resource, the application gets a database connection.

   Enterprise Server retrieves a physical connection from the connection pool that corresponds to the database. The pool defines connection attributes such as the database name (URL), user name, and password.

3. After the database connection is established, the application can read, modify, and add data to the database.

   The application accesses the database by making calls to the JDBC API. The JDBC driver translates the application's JDBC calls into the protocol of the database server.

4. When the application is finished accessing the database, the application closes the connection and returns the connection to the connection pool.

Setting Up the Database

Most applications use relational databases to store, organize, and retrieve data. Applications access relational databases through the Java™ Database Connectivity (JDBC) API.

The following topics are addressed here:

- "To Install the Database and Database Driver" on page 233
- "To Start the Database" on page 233
- "To Stop the Database" on page 234
- "Java DB Utility Scripts" on page 234

▼ To Install the Database and Database Driver

1   Install a supported database product.

To see the current list of database products supported by Enterprise Server, refer to the *Sun GlassFish Enterprise Server v3 Release Notes*.

2   Install a supported JDBC driver for the database product.

For a list of drivers supported by Enterprise Server, see "Configuration Specifics for JDBC Drivers" on page 243.

3   Make the JDBC driver JAR file accessible to the domain administration server (DAS).

See "Integrating the JDBC Driver" on page 242.

4   Create the database.

The application provider usually delivers scripts for creating and populating the database.

Next Steps   You are now ready to create a connection pool for the database, and a JDBC resource that points to the connection pool. See "To Create a JDBC Connection Pool" on page 236 and "To Create a JDBC Resource" on page 240. The final step is to integrate the JDBC driver into an administrative domain as described in "Integrating the JDBC Driver" on page 242.

▼ To Start the Database

Enterprise Server includes an implementation of Java DB (formerly known as Derby), however, you can use any JDBC-compliant database. The database is not started automatically when you start Enterprise Server, so if you have applications that require a database, you need to start Java DB manually by using the local `start-database` subcommand.

●   **Start the database by using the** `start-database`**(1) subcommand.**

When the database server starts, or a client connects to it successfully, the following files are created at the location that is specified by the `--dbhome` option:

- The `derby.log` file contains the database server process log along with its standard output and standard error information.
- The database files contain your schema (for example, database tables).

Example 14–1   Starting a Database

This example starts Derby on the host host1 and port 5001.

```
asadmin> start-database --dbhost host1 --dbport 5001 --terse=true
Starting database in the background.
Log redirected to /opt/SUNWappserver/databases/javadb.log.
Command start-database executed successfully.
```

See Also　You can also view the full syntax and options of the subcommand by typing asadmin help
start-database at the command line.

▼ To Stop the Database

Use the local stop-database subcommand to stop Java DB on a specified port. A single host
can have multiple database server processes running on different ports.

1　**If necessary, notify users that the database is being stopped.**

2　**Stop the database by using the** stop-database**(1) subcommand.**

Example 14–2　Stopping a Database

This example stops Java DB on port 5001 of localhost.

```
asadmin> stop-database --dbhost=localhost --dbport=5001
onnection obtained for host: localhost, port number 5001.
Apache Derby Network Server - 10.2.2.1 - (538595) shutdown
at 2008-10-17 23:34:2 7.218 GMT
Command stop-database executed successfully.
```

Troubleshooting　For a laptop that roams between networks, you might have trouble shutting down the database.
If you start Java DB and then change your IP address, you will not be able to stop Java DB unless
you add a specific --dbhost argument. For example, if you run asadmin start-database
--dbhost = 0.0.0.0, and then disconnect Ethernet and switch to wifi, you should run a
command similar to the following to stop the database:

```
asadmin stop-database --dbhost localhost
```

See Also　You can also view the full syntax and options of the subcommand by typing asadmin help
stop-database at the command line.

Java DB Utility Scripts

The Java DB configuration that is available for use with Enterprise Server includes scripts that
can help you use Java DB. The following scripts are available in the
as-install/javadb/frameworks/NetworkServer/bin directory:

startNetworkServer,startNetworkServer.bat
: Script to start the network server

stopNetworkServer,stopNetworkServer.bat
: Script to stop the network server

ij,ij.bat
: Interactive JDBC scripting tool

dblook,dblook.bat
: Script to view all or part of the DDL for a database

sysinfo, sysinfo.bat
: Script to display versioning information about the Java DB environment

NetworkServerControl,NetworkServerControl.bat
: Script to execute commands on the NetworkServerControl API

▼ To Configure Your Environment to Run Java DB Utility Scripts

1 **Ensure that the JAVA_HOME environment variable specifies the directory where the JDK is installed.**

2 **Set the JAVADB_HOME environment variable to point to the** *as-install*/derby **directory.**

See Also For more information about these utilities, see the following documentation:

- *Derby Tools and Utilities Guide* (http://db.apache.org/derby/docs/10.1/tools/)
- *Derby Server and Administration Guide* (http://db.apache.org/derby/docs/10.1/adminguide/)

Configuring Access to the Database

After establishing the database, you are ready to set up access for Enterprise Server applications. The high-level steps include creating a JDBC connection pool, creating a JDBC resource for the connection pool, and integrating a JDBC driver into an administrative domain. Instructions for performing these steps are contained in the following sections:

- "Administering JDBC Connection Pools" on page 236
- "Administering JDBC Resources" on page 240
- "Integrating the JDBC Driver" on page 242

Administering JDBC Connection Pools

A *JDBC connection pool* is a group of reusable connections for a particular database. Because creating each new physical connection is time consuming, Enterprise Server maintains a pool of available connections. When an application requests a connection, it obtains one from the pool. When an application closes a connection, the connection is returned to the pool.

A JDBC resource is created by specifying the connection pool with which the resource is associated. Multiple JDBC resources can specify a single connection pool. The properties of connection pools can vary with different database vendors. Some common properties are the database name (URL), the user name, and the password.

The following tasks and information are used to administer JDBC connection pools:

- "To Create a JDBC Connection Pool" on page 236
- "To List JDBC Connection Pools" on page 237
- "To Contact (Ping) a Connection Pool" on page 238
- "To Reset (Flush) a Connection Pool" on page 238
- "To Update a JDBC Connection Pool" on page 239
- "To Delete a JDBC Connection Pool" on page 239

▼ To Create a JDBC Connection Pool

Use the `create-jdbc-connection-pool` subcommand in remote mode to register a new JDBC connection pool with the specified JDBC connection pool name. A JDBC connection pool or a connector connection pool can be created with authentication. You can either use a subcommand option to specify user, password, or other connection information using the `asadmin` utility, or specify the connection information in the XML descriptor file.

One connection pool is needed for each database, possibly more depending on the application. When you are building the connection pool, certain data specific to the JDBC driver and the database vendor is required. You can find some of the following specifics in"Configuration Specifics for JDBC Drivers" on page 243:

- Database vendor name
- Resource type, such as `javax.sql.DataSource` (local transactions only) `javax.sql.XADataSource` (global transactions)
- Data source class name
- Required properties, such as the database name (URL), user name, and password

Creating a JDBC connection pool is a dynamic event and does not require server restart. However, there are some parameters that do require server restart. See "Configuration Changes That Require Server Restart" on page 37.

Before You Begin Before creating the connection pool, you must first install and integrate the database and its associated JDBC driver. For instructions, see "Setting Up the Database" on page 232.

1 Ensure that the server is running.

Remote subcommands require a running server.

2 Create the JDBC connection pool by using the `create-jdbc-connection-pool`**(1) subcommand.**

3 (Optional) If needed, restart the server.

Some parameters require server restart. See "Configuration Changes That Require Server Restart" on page 37.

Example 14–3 Creating a JDBC Connection Pool

This example creates a JDBC connection pool named sample_derby_pool on localhost.

```
asadmin> create-jdbc-connection-pool
--datasourceclassname org.apache.derby.jdbc.ClientDataSource
--restype javax.sql.XADataSource
--property portNumber=1527:password=APP:user=APP:serverName=
localhost:databaseName=sun-appserv-samples:connectionAttribut
es=\;create\\=true sample_derby_pool
Command create-jdbc-connection-pool executed successfully.
```

See Also You can also view the full syntax and options of the subcommand by typing asadmin help create-jdbc-connection-pool at the command line.

▼ To List JDBC Connection Pools

Use the list-jdbc-connection-pools subcommand in remote mode to list all existing JDBC connection pools.

1 Ensure that the server is running.

Remote subcommands require a running server.

2 List the JDBC connection pools by using the `list-jdbc-connection-pools`**(1) subcommand.**

Example 14–4 Listing JDBC Connection Pools

This example lists the JDBC connection pools that are on localhost.

```
asadmin> list-jdbc-connection-pools
sample_derby_pool2
poolA
__TimerPool
DerbyPool
```

```
sample_derby_pool
Command list-jdbc-connection-pools executed successfully.
```

See Also   You can also view the full syntax and options of the subcommand by typing `asadmin help list-jdbc-connection-pools` at the command line.

▼ To Contact (Ping) a Connection Pool

Use the `ping-connection-pool` subcommand in remote mode to test if a connection pool is usable. For example, if you create a new JDBC connection pool for an application that is expected to be deployed later, you can test the JDBC pool with this subcommand before the application is deployed. Running a ping will force the creation of the pool if it hasn't already been created.

Before You Begin   Before you can contact a connection pool, the connection pool must be created with authentication, and the server or database must be running.

1   Ensure that the server is running.

Remote subcommands require a running server.

2   Ping a connection pool by using the `ping-connection-pool`**(1) subcommand.**

Example 14–5   Contacting a Connection Pool

This example tests to see if the `DerbyPool` connection pool is usable.

```
asadmin> ping-connection-pool DerbyPool
Command ping-connection-pool executed successfully
```

See Also   You can also view the full syntax and options of the subcommand by typing `asadmin help ping-connection-pool` at the command line.

▼ To Reset (Flush) a Connection Pool

Use the `flush-connection-pool` in remote mode to reinitialize all connections established in the specified connection pool. The JDBC connection pool or connector connection pool is reset to its initial state. Any existing live connections are destroyed, which means that the transactions associated with these connections are lost. The subcommand then recreates the initial connections for the pool, and restores the pool to its steady pool size.

1   Ensure that the server is running.

Remote subcommands require a running server.

2   Reset a connection pool by using the `flush-connection-pool`**(1) subcommand.**

Example 14–6 Resetting (Flushing) a Connection Pool

This example resets the JDBC connection pool named __TimerPool to its steady pool size.

```
asadmin> flush-connection-pool __TimerPool
Command flush-connection-pool executed successfully.
```

See Also You can also view the full syntax and options of the subcommand by typing asadmin help flush-connection-pool at the command line.

▼ To Update a JDBC Connection Pool

You can change all of the settings for an existing pool except its name. Use the get and set subcommands to view and change the values of the JDBC connection pool properties.

1 List the JDBC connection pools by using the list-jdbc-connection-pools(1) subcommand.

2 View the attributes of the JDBC connection pool by using the get subcommand.

For example:

asadmin get resources.jdbc-connection-pool.DerbyPool.property

3 Set the attribute of the JDBC connection pool by using the set subcommand.

For example:

asadmin set resources.jdbc-connection-pool.DerbyPool.steady-pool-size=9

4 (Optional) If needed, restart the server.

Some parameters require server restart. See "Configuration Changes That Require Server Restart" on page 37.

▼ To Delete a JDBC Connection Pool

Use the delete-jdbc-connection-pool subcommand in remote mode to delete an existing JDBC connection pool. Deleting a JDBC connection pool is a dynamic event and does not require server restart.

Before You Begin Before deleting a JDBC connection pool, all associations to the resource must be removed.

1 Ensure that the server is running.

Remote subcommands require a running server.

2 List the JDBC connection pools by using the list-jdbc-connection-pools(1) subcommand.

3 If necessary, notify users that the JDBC connection pool is being deleted.

4 Delete the connection pool by using the `delete-jdbc-connection-pool`**(1) subcommand.**

Example 14–7   Deleting a JDBC Connection Pool

This example deletes the JDBC connection pool named `DerbyPool`.

```
asadmin> delete-jdbc-connection-pool jdbc/DerbyPool
Command delete-jdbc-connection-pool executed successfully.
```

See Also   You can also view the full syntax and options of the subcommand by typing `asadmin help delete-jdbc-connection-pool` at the command line.

Administering JDBC Resources

A *JDBC resource*, also known as a data source, provides an application with a means of connecting to a database. Typically, you create a JDBC resource for each database that is accessed by the applications deployed in a domain. Multiple JDBC resources can be specified for a database.

A JDBC resource is created by specifying the connection pool with which the resource will be associated . Use a unique Java Naming and Directory Interface (JNDI) name to identify the resource. For example, the JNDI name for the resource of a payroll database might be `java:comp/env/jdbc/payrolldb`.

The following tasks and information are used to administer JDBC resources:

▼ To Create a JDBC Resource

Use the `create-jdbc-resource` subcommand in remote mode to create a JDBC resource. Creating a JDBC resource is a dynamic event and does not require server restart.

Because all JNDI names are in the `java:comp/env` subcontext, when specifying the JNDI name of a JDBC resource in the Administration Console, use only the `jdbc/`*name* format. For example, a payroll database might be specified as `jdbc/payrolldb`.

Before You Begin   Before creating a JDBC resource, you must first create a JDBC connection pool. For instructions, see "To Create a JDBC Connection Pool" on page 236.

1 Ensure that the server is running.

Remote subcommands require a running server.

2 Create a JDBC resource by using the `create-jdbc-resource`**(1) subcommand.**

Information about properties for the subcommand is included in this help page.

3 If necessary, notify users that the new resource has been created.

Example 14–8 Creating a JDBC Resource

This example creates a JDBC resource named `DerbyPool`.

```
asadmin> create-jdbc-resource --connectionpoolid DerbyPool jdbc/DerbyPool
Command create-jdbc-resource executed successfully.
```

See Also You can also view the full syntax and options of the subcommand by typing `asadmin help create-jdbc-resource` at the command line.

▼ **To List JDBC Resources**

Use the `list-jdbc-resources` subcommand in remote mode to list the existing JDBC resources.

1 Ensure that the server is running.

Remote subcommands require a running server.

2 List JDBC resources by using the `list-jdbc-resources`**(1) subcommand.**

Example 14–9 Listing JDBC Resources

This example lists JDBC resources for `localhost`.

```
asadmin> list-jdbc-resources
jdbc/__TimerPool
jdbc/DerbyPool
jdbc/__default
jdbc1
Command list-jdbc-resources executed successfully.
```

See Also You can also view the full syntax and options of the subcommand by typing `asadmin help list-jdbc-resources` at the command line.

▼ **To Update a JDBC Resource**

You can enable or disable a JDBC resource by using the `set` subcommand. The JDBC resource is identified by its dotted name.

1 List JDBC resources by using the `list-jdbc-resources`**(1) subcommand.**

2   Modify the values for the specified JDBC resource by using the set**(1) subcommand.**
For example:

Example 14–10   Updating a JDBC Resource

This example changes the res1 enabled setting to false.

```
asadmin>set resources.jdbc-resource.res1.enabled=false
```

▼ **To Delete a JDBC Resource**

Use the delete-jdbc-resource subcommand in remote mode to delete an existing JDBC resource. Deleting a JDBC resource is a dynamic event and does not require server restart.

Before You Begin   Before deleting a JDBC resource, all associations with this resource must be removed.

1   Ensure that the server is running.
Remote subcommands require a running server.

2   List JDBC resources by using the list-jdbc-resources**(1) subcommand.**

3   If necessary, notify users that the JDBC resource is being deleted.

4   Delete a JDBC resource by using the delete-jdbc-resource**(1) subcommand.**

Example 14–11   Deleting a JDBC Resource

This example deletes a JDBC resource named DerbyPool.

```
asadmin> delete-jdbc-resource jdbc/DerbyPool
Command delete-jdbc-resource executed successfully.
```

See Also   You can also view the full syntax and options of the subcommand by typing asadmin help delete-jdbc-resource at the command line.

Integrating the JDBC Driver

After setting up the connection pool and resources, integrate the JDBC driver in either of the following ways:

■ Make the driver accessible to the common class loader, and restart the domain.

■ Copy the driver's JAR and ZIP files into the *domain-dir*/lib directory or the glassfish_home/lib directory, or copy the driver's class files into the *domain-dir*/lib/ext directory. Identify the fully-qualified path name for the driver's JAR file.

Configuration Specifics for JDBC Drivers

Enterprise Server is designed to support connectivity to any database management system by using a corresponding JDBC driver.

- "JDBC Drivers, Full Support " on page 243
- "JDBC Drivers, Limited Support " on page 249

JDBC Drivers, Full Support

The following JDBC driver and database combinations have been tested and are supported for container-managed persistence:

- "IBM DB2 Database Type 2 JDBC Driver for Sun/DataDirect" on page 243
- "IBM DB2 Database Type 4 JDBC Driver" on page 244
- "IBM DB2 Database Type 2 JDBC Driver" on page 244
- "Java DB/Derby Type 4 JDBC Driver" on page 244
- " Microsoft SQL Server Database Type 4 JDBC Driver for Sun/DataDirect" on page 245
- " MySQL Server Database Type 4 JDBC Driver for Sun/DataDirect" on page 245
- "MySQL Server Database Type 4 JDBC Driver" on page 246
- "Oracle 11 Database JDBC Driver for Sun/DataDirect" on page 246
- "Oracle 11 Database Type 2 JDBC Driver for Sun/DataDirect" on page 247
- "Oracle 11 Database Thin Type 4 JDBC Driver" on page 247
- "PostgreSQL Type 4 JDBC Driver" on page 248
- "Sybase Database Type 4 JDBC Driver for Sun/DataDirect" on page 248

To see the most current list of supported JDBC drivers, refer to the *Sun GlassFish Enterprise Server v3 Release Notes*.

IBM DB2 Database Type 2 JDBC Driver for Sun/DataDirect

The JAR file for this driver is `smdb2.jar`. Configure the connection pool using the following settings:

- **Name:** Use this name when you configure the JDBC resource later.

- **Resource Type:** Specify the appropriate value.

- **Database Vendor:** DB2

- **DataSource Classname:** `com.sun.sql.jdbcx.db2.DB2DataSource`

  DataDirect DataSource Classname: `com.ddtek.jdbcx.db2.DB2DataSource`

- **Properties:**
    - `serverName` – Specify the host name or IP address of the database server.
    - `portNumber` – Specify the port number of the database server.
    - `databaseName` – Set as appropriate.

- **user** – Set as appropriate.
- **password** – Set as appropriate.

IBM DB2 Database Type 4 JDBC Driver

TBD

IBM DB2 Database Type 2 JDBC Driver

The JAR files for the DB2 driver are db2jcc.jar, db2jcc_license_cu.jar, and db2java.zip. Set your environment variables . For example:

```
LD_LIBRARY_PATH=/usr/db2user/sqllib/lib:${Java EE.home}/lib
DB2DIR=/opt/IBM/db2/V8.2
DB2INSTANCE=db2user
INSTHOME=/usr/db2user
VWSPATH=/usr/db2user/sqllib
THREADS_FLAG=native
```

Configure the connection pool using the following settings:

- **Name:** Use this name when you configure the JDBC resource later.

- **Resource Type:** Specify the appropriate value.

- **Database Vendor:** DB2

- **DataSource Classname:** com.ibm.db2.jcc.DB2SimpleDataSource

  DataDirect DataSource Classname: com.ddtek.jdbcx.db2.DB2DataSource

- **Properties:**
  - **databaseName** - Set as appropriate.
  - **user** – Set as appropriate.
  - **password** – Set as appropriate.
  - **driverType** – Set to 2.
  - **deferPrepares** – Set to false.

Java DB/Derby Type 4 JDBC Driver

The JAR file for the Java DB driver is derbyclient.jar. (Java DB is based upon Apache Derby.) Configure the connection pool using the following settings:

- **Name:** Use this name when you configure the JDBC resource later.

- **Resource Type:** Specify the appropriate value.

- **Database Vendor:** Java DB

- **DataSource Classname:** Specify one of the following:

  ```
  org.apache.derby.jdbc.ClientDataSource
  org.apache.derby.jdbc.ClientXADataSource
  ```

- **Properties:**
    - **serverName** – Specify the host name or IP address of the database server.
    - **portNumber** – Specify the port number of the database server if it is different from the default.
    - **databaseName** – Specify the name of the database.
    - **user** - Specify the database user.

      This is only necessary if Java DB is configured to use authentication. Java DB does *not* use authentication by default. When the user is provided, it is the name of the schema where the tables reside.
    - **password** – Specify the database password.

      This is only necessary if Java DB is configured to use authentication.

Microsoft SQL Server Database Type 4 JDBC Driver for Sun/DataDirect

The JAR file for this driver is smsqlserver.jar. Configure the connection pool using the following settings:

- **Name:** Use this name when you configure the JDBC resource later.
- **Resource Type:** Specify the appropriate value.
- **Database Vendor:** Microsoft SQL Server
- **DataSource Classname:** com.sun.sql.jdbcx.sqlserver.SQLServerDataSource

  DataDirect DataSource Classname:
  com.ddtek.jdbcx.sqlserver.SQLServerDataSource
- **Properties:**
    - **serverName** – Specify the host name or IP address and the port of the database server.
    - **portNumber** – Specify the port number of the database server.
    - **user** – Set as appropriate.
    - **password** – Set as appropriate.
    - **selectMethod** – Set to cursor.

MySQL Server Database Type 4 JDBC Driver for Sun/DataDirect

The Sun MySQL driver only works with MySQL Enterprise. The JAR file for this driver is smmysql.jar. Configure the connection pool using the following settings:

- **Name:** Use this name when you configure the JDBC resource later.
- **Resource Type:** Specify the appropriate value.
- **Database Vendor:** MySQL Server
- **DataSource Classname:** com.sun.sql.jdbcx.mysql.MySQLDataSource

DataDirect DataSource: `com.ddtek.jdbcx.mysql.MySQLDataSource`

- **Properties:**
    - `serverName` – Specify the host name or IP address and the port of the database server.
    - `portNumber` – Specify the port number of the database server.
    - `user` – Set as appropriate.
    - `password` – Set as appropriate.
    - `selectMethod` – Set to `cursor`.

MySQL Server Database Type 4 JDBC Driver

The JAR file for the MySQL™ driver is `mysql-connector-java-5.1.7-bin.jar`. Configure the connection pool using the following settings:

- **Name:** Use this name when you configure the JDBC resource later.
- **Resource Type:** Specify the appropriate value.
- **Database Vendor:** Microsoft SQL Server
- **DataSource Classname:**

```
com.mysql.jdbc.jdbc2.optional.MysqlDataSource
com.mysql.jdbc.jdbc2.optional.MysqlXADataSource
```

- **Properties:**
    - `serverName` – Specify the host name or IP address of the database server.
    - `portNumber` – Specify the port number of the database server.
    - `databaseName` – Set as appropriate.
    - `user` – Set as appropriate.
    - `password` – Set as appropriate.

Oracle 11 Database JDBC Driver for Sun/DataDirect

The JAR file for this driver is `smoracle.jar`.

Note – To make the Oracle driver behave in a Java EE-compliant manner, you musts set this system property as `true`: `oracle.jdbc.J2EE13Complient=true`.

Configure the connection pool using the following settings:

- **Name:** Use this name when you configure the JDBC resource later.
- **Resource Type:** Specify the appropriate value.
- **Database Vendor:** Oracle
- **DataSource Classname:** `com.sun.sql.jdbcx.oracle.OracleDataSource`

   DataDirect DataSource Classname: `com.ddtek.jdbcx.oracle.OracleDataSource`

- **Properties:**
  - `serverName` – Specify the host name or IP address of the database server.
  - `portNumber` – Specify the port number of the database server.
  - `user` – Set as appropriate.
  - `password` – Set as appropriate.

Oracle 11 Database Type 2 JDBC Driver for Sun/DataDirect

TBD

Note – To make the Oracle driver behave in a Java EE-compliant manner, you musts set this system property as `true`: `oracle.jdbc.J2EE13Complient=true`.

Oracle 11 Database Thin Type 4 JDBC Driver

The JAR file for the Oracle driver is `ojdbc6.jar`.

Note – When using this driver, keep in mind that you cannot insert more than 2000 bytes of data into a column. To circumvent this problem, use the OCI driver (JDBC type 2).

Note – To make the Oracle driver behave in a Java EE-compliant manner, you musts set this system property as `true`: `oracle.jdbc.J2EE13Complient=true`.

Configure the connection pool using the following settings:

- **Name:** Use this name when you configure the JDBC resource later.

- **Resource Type:** Specify the appropriate value.

- **Database Vendor:** Oracle

- **DataSource Classname:** Specify one of the following:

  ```
  oracle.jdbc.pool.OracleDataSource
  oracle.jdbc.xa.client.OracleXADataSource
  ```

  DataDirect DataSource Classname: `com.ddtek.jdbcx.oracle.OracleDataSource`

- **Properties:**
  - `user` – Set as appropriate.
  - `password` – Set as appropriate.

- **xa-driver-does-not-support-non-tx-operations** - Set to the value true. Optional: only needed if both non-XA and XA connections are retrieved from the same connection pool. Might degrade performance.

  As an alternative to setting this property, you can create two connection pools, one for non-XA connections and one for XA connections.

Note – For the Oracle thin driver, the XAResource.recover method repeatedly returns the same set of in-doubt Xids regardless of the input flag. According to the XA specifications, the Transaction Manager initially calls this method with TMSTARTSCAN and then with TMNOFLAGS repeatedly until no Xids are returned. The XAResource.commit method also has some issues.

To disable this Enterprise Server workaround, the oracle-xa-recovery-workaround property value must be set to false.

PostgreSQL Type 4 JDBC Driver

The JAR file for the PostgreSQL driver is postgresql-8.4-701.jdbc4.jar. Configure the connection pool using the following settings:

- **Name:** Use this name when you configure the JDBC resource later.

- **Resource Type:** Specify the appropriate value.

- **Database Vendor:** PostgreSQL Server

- **DataSource Classname:** org.postgresql.ds.PGSimpleDataSource

- **Properties:**
  - **serverName** – Specify the host name or IP address of the database server.
  - **portNumber** – Specify the port number of the database server.
  - **databaseName** – Set as appropriate.
  - **user** – Set as appropriate.
  - **password** – Set as appropriate.

Sybase Database Type 4 JDBC Driver for Sun/DataDirect

The JAR file for this driver is smsybase.jar. Configure the connection pool using the following settings:

- **Name:** Use this name when you configure the JDBC resource later.

- **Resource Type:** Specify the appropriate value.

- **Database Vendor:** Sybase

- **DataSource Classname:** com.sun.sql.jdbcx.sybase.SybaseDataSource

  DataDirect DataSource Classname: com.ddtek.jdbcx.sybase.SybaseDataSource

- **Properties:**

- **serverName** – Specify the host name or IP address of the database server.
- **portNumber** – Specify the port number of the database server.
- **databaseName** – Set as appropriate. This is optional.
- **user** – Set as appropriate.
- **password** – Set as appropriate.

JDBC Drivers, Limited Support

The following JDBC drivers can also be used with Enterprise Server, but have not been fully tested. Although Sun offers no product support for these drivers, Sun does offer limited support for the use of these drivers with Enterprise Server:

Note – An Oracle database user running the capture-schema command needs ANALYZE ANY TABLE privileges if that user does not own the schema. These privileges are granted to the user by the database administrator. For information about capture-schema, see *Sun GlassFish Enterprise Server v3 Reference Manual*.

IBM Informix Type 4 Driver for Sun/DataDirect

Configure the connection pool using the following settings:

- **Name:** Use this name when you configure the JDBC resource later.

- **Resource Type:** Specify the appropriate value.

- **Database Vendor:** Informix

- **DataSource Classname:** Specify one of the following:

  ```
  com.informix.jdbcx.IfxDataSource
  com.informix.jdbcx.IfxXADataSource
  ```

  DataDirect DataSource Classname: com.ddtek.jdbcx.informix.InformixDataSourcee

- **Properties:**

  - **serverName** – Specify the Informix database server name.
  - **portNumber** – Specify the port number of the database server.
  - **databaseName** – Set as appropriate. This is optional.
  - **user** – Set as appropriate.
  - **password** – Set as appropriate.

- **IfxIFXHost** – Specify the host name or IP address of the database server.

Inet Oraxo JDBC Driver for Oracle Databases

The JAR file for the Inet Oracle driver is `Oranxo.jar`. Configure the connection pool using the following settings:

- **Name:** Use this name when you configure the JDBC resource later.
- **Resource Type:** Specify the appropriate value.
- **Database Vendor:** Oracle
- **DataSource Classname:** `com.inet.ora.OraDataSource`
- **Properties:**
  - **serverName** – Specify the host name or IP address of the database server.
  - **portNumber** – Specify the port number of the database server.
  - **user** – Specify the database user.
  - **password** – Specify the database password.
  - **serviceName** – Specify the URL of the database. The syntax is as follows:

    `jdbc:inetora:`*server*`:`*port*`:`*dbname*

    For example:

    `jdbc:inetora:localhost:1521:payrolldb`

    In this example, `localhost` is the name of the host running the Oracle server, `1521` is the Oracle server's port number, and `payrolldb` is the SID of the database. For more information about the syntax of the database URL, see the Oracle documentation.

  - **streamstolob** - If the size of BLOB or CLOB data types exceeds 4 KB and this driver is used for CMP, this property must be set to `true`.

  - **xa-driver-does-not-support-non-tx-operations** - Set to the value `true`. Only needed if both non-XA and XA connections are retrieved from the same connection pool. Might degrade performance.

    As an alternative to setting this property, you can create two connection pools, one for non-XA connections and one for XA connections.

Inet Merlia JDBC Driver for Microsoft SQL Server Databases

The JAR file for the Inet Microsoft SQL Server driver is `Merlia.jar`. Configure the connection pool using the following settings:

- **Name:** Use this name when you configure the JDBC resource later.
- **Resource Type:** Specify the appropriate value.
- **Database Vendor:** Microsoft SQL Server

- **DataSource Classname:** `com.inet.tds.TdsDataSource`
- **Properties:**
  - `serverName` – Specify the host name or IP address and the port of the database server.
  - `portNumber` – Specify the port number of the database server.
  - `user` – Set as appropriate.
  - `password` – Set as appropriate.

Inet Sybelux JDBC Driver for Sybase Databases

The JAR file for the Inet Sybase driver is `Sybelux.jar`. Configure the connection pool using the following settings:

- **Name:** Use this name when you configure the JDBC resource later.
- **Resource Type:** Specify the appropriate value.
- **Database Vendor:** Sybase
- **DataSource Classname:** `com.inet.syb.SybDataSource`
- **Properties:**
  - `serverName` – Specify the host name or IP address of the database server.
  - `portNumber` – Specify the port number of the database server.
  - `databaseName` – Set as appropriate. Do not specify the complete URL, only the database name.
  - `user` – Set as appropriate.
  - `password` – Set as appropriate.

JConnect Type 4 Driver for Sybase ASE 12.5 Databases

The JAR file for the Sybase driver is `jconn4.jar`. Configure the connection pool using the following settings:

- **Name:** Use this name when you configure the JDBC resource later.
- **Resource Type:** Specify the appropriate value.
- **Database Vendor:** Sybase
- **DataSource Classname:** Specify one of the following:

  ```
  com.sybase.jdbc4.jdbc.SybDataSource
  com.sybase.jdbc4.jdbc.SybXADataSource
  ```

- **Properties:**
  - `serverName` – Specify the host name or IP address of the database server.
  - `portNumber` – Specify the port number of the database server.

- **databaseName** – Set as appropriate. Do not specify the complete URL, only the database name.
- **user** – Set as appropriate.
- **password** – Set as appropriate.
- **BE_AS_JDBC_COMPLIANT_AS_POSSIBLE** – Set to true.
- **FAKE_METADATA** – Set to true.

OCI Oracle Type 2 Driver for Oracle Databases

The JAR file for the OCI Oracle driver is ojdbc14.jar. Make sure that the shared library is available through LD_LIBRARY_PATH and that the ORACLE_HOME property is set. Configure the connection pool using the following settings:

- **Name:** Use this name when you configure the JDBC resource later.
- **Resource Type:** Specify the appropriate value.
- **Database Vendor:** Oracle
- **DataSource Classname:** Specify one of the following:

  ```
  oracle.jdbc.pool.OracleDataSource
  oracle.jdbc.xa.client.OracleXADataSource
  ```

- **Properties:**
  - **user** – Set as appropriate.
  - **password** – Set as appropriate.
  - **xa-driver-does-not-support-non-tx-operations** - Set to the value true. Only needed if both non-XA and XA connections are retrieved from the same connection pool. Might degrade performance.

    As an alternative to setting this property, you can create two connection pools, one for non-XA connections and one for XA connections.

Administering EIS Connectivity

This chapter provides information and procedures for administering connections to enterprise information system (EIS) data in the Sun GlassFish™ Enterprise Server v3 environment by using the `asadmin` command-line utility.

Note – If you installed the Web Profile, connector modules that use only outbound communication features and work-management that does not involve inbound communication features are supported. Other connector features are supported only in the Full Platform Profile.

The following topics are addressed here:

Instructions for accomplishing the tasks in this chapter by using the Administration Console are contained in the Administration Console online help.

For information about database connectivity, see Chapter 14, "Administering Database Connectivity ."

About EIS Connectivity

Enterprise information system (EIS) refers to any system that holds the data of an organization. It can be a mainframe, a messaging system, a database system, or an application. Connection resources are used by applications and modules to access EIS software.)

The key elements of EIS connectivity are the following:

- **Connector Module.** A connector module, also called a *resource adapter*, is a Java EE component that enables applications to interact with EIS software. A connector module is used by Enterprise Server to implement Java™ Message Service (JMS). Like other Java EE modules, a connector module is installed when it is deployed. For instructions on creating a connector module, see Chapter 12, "Developing Connectors," in *Sun GlassFish Enterprise Server v3 Application Development Guide*

- **Connector Connection Pool.** A connector connection pool is a group of reusable connections for a particular EIS. A connector connection pool is created when you specify the connector module that is associated with the pool. For administration procedures, see "Administering Connector Connection Pools" on page 255.

- **Connector Resource.** A connector resource is a program object that provides an application with a connection to an EIS. A connector resource is created when you specify its JNDI name and its associated connection pool. The JNDI name of a connector resource for an EIS is usually in the java:comp/env/*eis-specific* subcontext. For administration procedures, see "Administering Connector Resources" on page 258.

- **Connector Module Configuration.** A connector module configuration is the information that resides in the domain configuration file (domain.xml) for the particular connector module (resource adapter). For administration procedures, see "Administering the Resource Adapter Configuration" on page 261.

- **Connector Security Map.** A connector security map associates the caller identity of the application (principal or user group) to a suitable EIS principal or group. For administration procedures, see "Administering Connector Security Maps" on page 263.

- **Connector Work Security Map.** A connector work security map associates the caller identity of the work submitted by the connector module (resource adapter) EIS principal or EIS user group to a suitable principal or user group in the Enterprise Server security domain. For administration procedures, see "Administering Connector Work Security Maps" on page 267.

- **Administered Object.** An administered object provides specialized functionality for an application, such as providing access to a parser that is specific to the connector module and its associated EIS. For administration procedures, see "Administering Administered Objects" on page 270.

At runtime, the following sequence occurs when an application connects to an EIS:

1. The application gets the connector resource (data source) associated with the EIS by making a call through the JNDI API.

Using the JNDI name of the connector resource, the naming and directory service locates the resource. Each EIS resource specifies a connector connection pool.

2. Using the connector resource, the application gets an EIS connection.

   Enterprise Server retrieves a physical connection from the connection pool that corresponds to the EIS resource. The pool defines connection attributes such as the EIS name, user name, and password.

3. After the EIS connection is established, the application can read, modify, and add data to the EIS.

   The application accesses the EIS information by making calls to the JMS API.

4. When the application is finished accessing the EIS, the application closes the connection and returns the connection to the connection pool.

Administering Connector Connection Pools

After a connector module has been deployed, you are ready to create a connector connection pool for it.

The following topics are addressed here:

- "To Create a Connector Connection Pool" on page 255
- "To List Connector Connection Pools" on page 256
- "To Connect to (Ping) or Reset (Flush) a Connector Connection Pool" on page 257
- "To Update a Connector Connection Pool" on page 257
- "To Delete a Connector Connection Pool" on page 258

▼ To Create a Connector Connection Pool

Use the `create-connector-connection-pool` subcommand in remote mode to create a connector connection pool for a deployed connector module. When you are building the connector connection pool, certain data specific to the EIS will be required. The value in the mandatory `--connectiondefintion` option provides the EIS info.

Multiple connector resources can specify a single connection pool.

Creating a connector connection pool is a dynamic event and does not require server restart. However, there are some parameters that do require server restart. See "Configuration Changes That Require Server Restart" on page 37.

Before You Begin   Before creating the connector connection pool, the connector must be installed.

1   **Ensure that the server is running.**

Remote subcommands require a running server.

2 Create the connector connection pool by using the `create-connector-connection-pool`**(1) subcommand.**

Information about properties for the subcommand is included in this help page.

3 (Optional) If needed, restart the server.

Some properties require server restart. See "Configuration Changes That Require Server Restart" on page 37. If your server needs to be restarted, see "To Restart a Domain" on page 90.

4 (Optional) You can verify that a connection pool is usable by using the `ping-connection-pool` **subcommand.**

For instructions, see "To Contact (Ping) a Connection Pool" on page 238.

Example 15–1 Creating a Connector Connection Pool

This example creates the new jms/qConnPool pool for the javax.jms.QueueConnectionFactory connector module.

```
asadmin> create-connector-connection-pool --steadypoolsize 20 --maxpoolsize 100
--poolresize 2 --maxwait 60000 --raname jmsra --connectiondefinition
javax.jms.QueueConnectionFactory jms/qConnPool
Command create-connector-connection-pool executed successfully
```

See Also You can also view the full syntax and options of the subcommand by typing asadmin help create-connector-connection-pool at the command line.

▼ To List Connector Connection Pools

Use the list-connector-connection-pools subcommand in remote mode to list the pools that have been created.

1 Ensure that the server is running.

Remote subcommands require a running server.

2 List the connector connection pools by using the `list-connector-connection-pools`**(1) subcommand.**

Example 15–2 Listing Connector Connection Pools

This example lists the existing connector connection pools.

```
asadmin> list-connector-connection-pools
jms/qConnPool
Command list-connector-connection-pools executed successfully
```

See Also You can also view the full syntax and options of the subcommand by typing asadmin help list-connector-connection-pools at the command line.

▼ To Connect to (Ping) or Reset (Flush) a Connector Connection Pool

Use the ping-connection-pool or flush-connection-pool subcommands in remote mode to perform these tasks on a connection pools. See "To Contact (Ping) a Connection Pool" on page 238 or "To Reset (Flush) a Connection Pool" on page 238 for instructions.

1 Ensure that the server is running.

Remote subcommands require a running server.

2 Connect to or reset a connector connection pool by using the flush-connection-pool**(1) subcommand or the** ping-connection-pool**(1) subcommand.**

▼ To Update a Connector Connection Pool

Use the get and set subcommands to view and change the values of the connector connection pool properties.

1 List the connector connection pools by using the list-connector-connection-pools**(1) subcommand.**

2 View the properties of the connector connection pool by using the get**(1) subcommand.**

For example:

```
asadmin> get domain.resources.connector-connection-pool.conectionpoolname.*
```

3 Set the property of the connector connection pool by using the set**(1) subcommand.**

For example:

```
asadmin> set domain.resources.connector-connection-pool
.conectionpoolname.validate-atmost-once-period-in-seconds=3
```

4 (Optional) If needed, restart the server.

Some properties require server restart. See "Configuration Changes That Require Server Restart" on page 37. If your server needs to be restarted, see "To Restart a Domain" on page 90.

▼ To Delete a Connector Connection Pool

Use the `delete-connector-connection-pool` subcommand in remote mode to remove a connector connection pool.

1   Ensure that the server is running.

Remote subcommands require a running server.

2   List the connector connection pools by using the `list-connector-connection-pools`(1) subcommand.

3   If necessary, notify users that the connector connection pool is being deleted.

4   Delete the connector connection pool by using the `delete-connector-connection-pool`(1) subcommand.

Example 15–3   Deleting a Connector Connection Pool

This example deletes the connection pool named `jms/qConnPool`.

```
asadmin> delete-connector-connection-pool --cascade=false jms/qConnPool
Command delete-connector-connection-pool executed successfully
```

See Also   You can also view the full syntax and options of the subcommand by typing `asadmin help delete-connector-connection-pool` at the command line.

Administering Connector Resources

A connector resource provides an application or module with the means of connecting to an EIS. Typically, you create a connector resource for each EIS that is accessed by the applications deployed in the domain.

The following topics are addressed here:

- "To Create a Connector Resource" on page 258
- "To List Connector Resources" on page 259
- "To Update a Connector Resource" on page 260
- "To Delete a Connector Resource" on page 260

▼ To Create a Connector Resource

Use the `create-connector-resource` subcommand in remote mode to register a new connector resource with its JNDI name.

Creating a connector resource is a dynamic event and does not require server restart. However, there are some parameters that do require server restart. See "Configuration Changes That Require Server Restart" on page 37.

Before You Begin Before creating a connector resource, you must first create a connector connection pool. For instructions, see "To Create a Connector Connection Pool" on page 255.

1 **Ensure that the server is running.**

Remote subcommands require a running server.

2 **Create the connector resource by using the** `create-connector-resource`**(1) subcommand.**

Information about properties for the subcommand is included in this help page.

3 **(Optional) If needed, restart the server.**

Some properties require server restart. See "Configuration Changes That Require Server Restart" on page 37. If your server needs to be restarted, see "To Restart a Domain" on page 90.

Example 15–4 Creating a Connector Resource

This example creates a new resource named jms/qConnFactory for the jms/qConnPool connection pool.

```
asadmin> create-connector-resource --poolname jms/qConnPool
--description "creating sample connector resource" jms/qConnFactory
Command create-connector-resource executed successfully
```

See Also You can also view the full syntax and options of the subcommand by typing asadmin help create-connector-resource at the command line.

▼ **To List Connector Resources**

Use the list-connector-resources subcommand in remote mode to list the connector resources that have been created.

1 **Ensure that the server is running.**

Remote subcommands require a running server.

2 **List the connector connection pools by using the** list-connector-resources**(1) subcommand.**

Example 15–5 Listing Connector Resources

This example lists the existing connector resources.

```
asadmin> list-connector-resources
jms/qConnFactory
Command list-connector-resources executed successfully
```

See Also   You can also view the full syntax and options of the subcommand by typing asadmin help list-connector-resources at the command line.

▼ To Update a Connector Resource

Use the get and set subcommands to view and change the values of the connector resource properties.

1   **List the connector connection pools by using the** list-connector-resources**(1) subcommand.**

2   **View the properties of the connector resource by using the** get**(1) subcommand.**

For example

```
asadmin> get domain.resources.connector-resource.jms/qConnFactory
```

3   **Set the property of the connector resource by using the** set**(1) subcommand.**

For example:

```
asadmin> set domain.resources.connector-resource.jms/qConnFactory.enabled=true
```

4   **(Optional) If needed, restart the server.**

Some properties require server restart. See "Configuration Changes That Require Server Restart" on page 37. If your server needs to be restarted, see "To Restart a Domain" on page 90.

▼ To Delete a Connector Resource

Use the delete-connector-resource subcommand in remote mode to remove a connector resource by specifying the JNDI name.

Before You Begin   Before deleting a resource, all associations with the resource must be removed.

1   **Ensure that the server is running.**

Remote subcommands require a running server.

2   **List the connector connection pools by using the** list-connector-resources**(1) subcommand.**

3   **If necessary, notify users that the connector resource is being deleted.**

4   **Delete the connector resource by using the** delete-connector-resource**(1) subcommand.**

Example 15–6 Deleting a Connector Resource

This example deletes the jms/qConnFactory connector resource.

```
asadmin> delete-connector-resource jms/qConnFactory
Command delete-connector-resources executed successfully
```

See Also You can also view the full syntax and options of the subcommand by typing asadmin help delete-connector-resource at the command line.

Administering the Resource Adapter Configuration

The following topics are addressed here:

- "To Create Configuration Information for a Resource Adapter" on page 261
- "To List Resource Adapter Configurations" on page 262
- "To Update a Resource Adapter Configuration" on page 262
- "To Delete a Resource Adapter Configuration" on page 263

▼ To Create Configuration Information for a Resource Adapter

Use the create-resource-adapter-config subcommand in remote mode to create configuration information for a resource adapter, also known as a connector module. You can run the subcommand before deploying a resource adapter, so that the configuration information is available at the time of deployment. The resource adapter configuration can also be created after the resource adapter is deployed. In this situation, the resource adapter is restarted with the new configuration.

1 Ensure that the server is running.

Remote subcommands require a running server.

2 Create configuration information by using the create-resource-adapter-config(1) **subcommand.**

Information about properties for the subcommand is included in this help page.

Example 15–7 Creating a Resource Adapter Configuration

This example creates the configuration for resource adapter ra1.

```
asadmin> create-resource-adapter-config --property foo=bar
--threadpoolid mycustomerthreadpool ra1
Command create-resource-adapter-config executed successfully
```

See Also   You can also view the full syntax and options of the subcommand by typing asadmin help
create-resource-adapter-config at the command line.

▼ To List Resource Adapter Configurations

Use the list-resource-adapter-configs subcommand in remote mode to list the
configuration information contained in the domain configuration file (domain.xml) for the
specified resource adapter (connector module).

1   Ensure that the server is running.

Remote subcommands require a running server.

2   List the configurations for a resource adapter by using the
list-resource-adapter-configs**(1) subcommand.**

Example 15–8   Listing Configurations for a Resource Adapter

This example lists all the resource adapter configurations.

```
asadmin> list-resource-adapter-configs
ra1
ra2
Command list-resource-adapter-configs executed successfully
```

See Also   You can also view the full syntax and options of the subcommand by typing asadmin help
list-resource-adapter-configs at the command line.

▼ To Update a Resource Adapter Configuration

Use the get and set subcommands to view and change the values of the resource adapter
configuration properties.

1   List the configurations for a resource adapter by using the
list-resource-adapter-configs**(1) subcommand.**

2   View the properties of the connector resource by using the get**(1) subcommand.**

For example:

```
asadmin> get domain.resources.resource-adapter-config.ra1.*
```

3   Set the property of the connector resource by using the set**(1) subcommand.**

For example:

```
asadmin> set domain.resources.resource-adapter-config.ra1.raSpecificProperty=value
```

▼ To Delete a Resource Adapter Configuration

Use the `delete-resource-adapter-config` subcommand in remote mode to delete the configuration information contained in the domain configuration file (`domain.xml`) for a specified resource adapter (connector module).

1 Ensure that the server is running.

Remote subcommands require a running server.

2 List the configurations for a resource adapter by using the `list-resource-adapter-configs`(1) subcommand.

3 Delete the configuration for a resource adapter by using the `delete-resource-adapter-config`(1) subcommand.

Example 15–9 Deleting a Resource Adapter Configuration

This example deletes the configuration for resource adapter `ra1`.

```
asadmin> delete-resource-adapter-config ra1
Command delete-resource-adapter-config executed successfully
```

See Also You can also view the full syntax and options of the subcommand by typing `asadmin help delete-resource-adapter-config` at the command line.

Administering Connector Security Maps

The EIS is any system that holds the data of an organization. It can be a mainframe, a messaging system, a database system, or an application. The connector security map is used to map the application's credentials to the EIS credentials.

A security map applies to a particular connector connection pool. One or more named security maps can be associated with a connector connection pool.

The following topics are addressed here:

▼ To Create a Connector Security Map

Use the `create-connector-security-map` subcommand in remote mode to create a security map for the specified connector connection pool. If the security map is not present, a new one is created. You can specify back-end EIS principals or back-end EIS user groups. The connector security map configuration supports the use of the wild card asterisk (*) to indicate all users or all user groups.

You can also use this subcommand to map the caller identity of the application (principal or user group) to a suitable EIS principal in container-managed transaction-based scenarios.

Before You Begin   For this subcommand to succeed, you must have first created a connector connection pool. For instructions, see "To Create a Connector Connection Pool" on page 255.

1   Ensure that the server is running.

Remote subcommands require a running server.

2   Create a connector security map by using the `create-connector-security-map`**(1) subcommand.**

Information about the options for the subcommand is included in this help page.

3   (Optional) If needed, restart the server.

Some properties require server restart. See "Configuration Changes That Require Server Restart" on page 37. If your server needs to be restarted, see "To Restart a Domain" on page 90.

Example 15–10   Creating a Connector Security Map

This example creates a connector security map securityMap1 for connection-pool1.

```
asadmin> create-connector-security-map --poolname connector-pool1
--principals principal1, principal2 --mappedusername backend-username securityMap1
Command create-connector-security-map executed successfully
```

▼ To List Connector Security Maps

Use the `list-connector-security-maps` subcommand in remote mode to list the existing security maps belonging to the specified connector connection pool. You can get a simple listing of the connector security maps for a connector connection pool, or you can get a more comprehensive listing that shows the principals of the map.

1   Ensure that the server is running.

Remote subcommands require a running server.

2 **List existing connector connection pools by using the** `list-connector-connection-pools`**(1)**
subcommand.

3 **List the security maps for a specific connector connection pool by using the**
`list-connector-security-maps`**(1) subcommand.**

Example 15–11 Listing All Connector Security Maps for a Connector Connection Pool

This example lists the connector security maps associated with `connector-Pool1`.

```
asadmin> list-connector-security-maps connector-Pool1
securityMap1
Command list-connector-security-maps executed successfully.
```

Example 15–12 Listing Principals for a Specific Security Map for a Connector Connection Pool

This example lists the principals associated with `securityMap1`.

```
asadmin> list-connector-security-maps --securitymap securityMap1 connector-Pool1
principal1
principal1
Command list-connector-security-maps executed successfully.
```

Example 15–13 Listing Principals of All Connector Security Maps for a Connector Connection Pool

This example lists the connector security maps associated with `connector-Pool1`.

```
asadmin> list-connector-security-maps --verbose connector-Pool1
securityMap1
principal1
principal1
Command list-connector-security-maps executed successfully.
```

▼ **To Update a Connector Security Map**

Use the `update-connector-security-map` subcommand in remote mode to create or modify a
security map for the specified connector connection pool.

1 **Ensure that the server is running.**

Remote subcommands require a running server.

2 **List existing connector security maps by using the** `list-connector-security-maps`**(1)**
subcommand.

3 Modify a security map for a specific connector connection pool by using the
update-connector-security-map**(1) subcommand.**

4 (Optional) If needed, restart the server.
Some properties require server restart. See "Configuration Changes That Require Server Restart" on page 37. If your server needs to be restarted, see "To Restart a Domain" on page 90.

Example 15–14   Updating a Connector Security Map

This example adds principals to securityMap1.

```
asadmin> update-connector-security-map --poolname connector-pool1
--addprincipals principal1, principal2 securityMap1
Command update-connector-security-map executed successfully.
```

▼ To Delete a Connector Security Map

Use the delete-connector-security-map subcommand in remote mode to delete a security map for the specified connector connection pool.

1 Ensure that the server is running.
Remote subcommands require a running server.

2 List existing connector connection pools by using the list-connector-connection-pools**(1) subcommand.**

3 Delete a security map for a specific connector connection pool by using the
delete-connector-security-map**(1) subcommand.**
Information about options for this subcommand is included in this help page.

Example 15–15   Deleting a Connector Security Map

This example deletes securityMap1 from connector-pool1.

```
asadmin> delete-connector-security-map --poolname connector-pool1 securityMap1
Command delete-connector-security-map executed successfully
```

Administering Connector Work Security Maps

The EIS is any system that holds the data of an organization. It can be a mainframe, a messaging system, a database system, or an application. The connector work security map is used to is used to map the EIS credentials to the credentials of Enterprise Server security domain.

A security map applies to a particular connector connection pool. One or more named security maps can be associated with a connector connection pool.

The following topics are addressed here:

- "To Create a Connector Work Security Map" on page 267
- "To List Connector Work Security Maps" on page 268
- "To Update a Connector Work Security Map" on page 268
- "To Delete a Connector Work Security Map" on page 269

▼ To Create a Connector Work Security Map

Use the `create-connector-work-security-map` subcommand in remote mode to map the caller identity of the work submitted by the connector module (resource adapter) EIS principal or EIS user group to a suitable principal or user group in the Enterprise Server security domain. One or more work security maps can be associated with a connector module.

The connector security map configuration supports the use of the wild card asterisk (*) to indicate all users or all user groups.

Before You Begin   Before creating a connector work security map, you must first create a connector connection pool. For instructions, see "To Create a Connector Connection Pool" on page 255.

1   Ensure that the server is running.

Remote subcommands require a running server.

2   Create the connector work security map by using the
`create-connector-work-security-map`**(1) subcommand.**

Information about properties for the subcommand is included in this help page.

3   (Optional) If needed, restart the server.

Some properties require server restart. See "Configuration Changes That Require Server Restart" on page 37. If your server needs to be restarted, see "To Restart a Domain" on page 90.

Example 15–16   Creating Connector Work Security Maps

The following examples create `workSecurityMap1` and `workSecurityMap2` for `my-resource-adapter-name`.

```
asadmin> create-connector-work-security-map --raname my-resource-adapter-name
--principalsmap eis-principal-1=server-principal-1,eis-principal-2=server-principal-2,
eis-principal-3=server-principal-1 workSecurityMap1

asadmin> create-connector-work-security-map --raname my-resource-adapter-name
--groupsmap eis-group-1=server-group-1,eis-group-2=server-group-2,
eis-group-3=server-group-1 workSecurityMap2
Command create-connector-work-security-map executed successfully
```

See Also   You can also view the full syntax and options of the subcommand by typing asadmin help
create-connector-work-security-map at the command line.

▼ To List Connector Work Security Maps

Use the list-connector-work-security-maps subcommand in remote mode to list the work
security maps that belong to a specific connector module.

1   Ensure that the server is running.

Remote subcommands require a running server.

2   List the connector work security maps by using the list-connector-work-security-maps**(1)**
subcommand.

Example 15–17   Listing the Connector Work Security Maps

This example lists the generic work security maps.

```
asadmin> list-connector-work-security-maps generic-ra
generic-ra-groups-map: EIS group=eis-group, mapped group=glassfish-group
generic-ra-principals-map: EIS principal=eis-bar, mapped principal=bar
generic-ra-principals-map: EIS principal=eis-foo, mapped principal=foo
Command list-connector-work-security-maps executed successfully.
```

See Also   You can also view the full syntax and options of the subcommand by typing asadmin help
list-connector-work-security-maps at the command line.

▼ To Update a Connector Work Security Map

Use the update-connector–work-security-map subcommand in remote to modify a work
security map that belongs to a specific resource adapter (connector module).

1   Ensure that the server is running.

Remote subcommands require a running server.

2 List the connector work security maps by using the `list-connector-work-security-maps`**(1)
subcommand.**

3 If necessary, notify users that the connector work security map is being modified.

4 Update a connector work security map by using the
`update-connector-work-security-map`**(1) subcommand.**

Example 15–18 Updating a Connector Work Security Map

This example removes a principal from a work security map.

```
asadmin> update-connector-work-security-map --raname generic-ra
--removeprincipals eis-foo generic-ra-principals-map
Command update-connector-work-security-map executed successfully.
```

See Also You can also view the full syntax and options of the subcommand by typing `asadmin help`
`update-connector-work-security-map` at the command line.

▼ **To Delete a Connector Work Security Map**

Use the `delete-connector–work-security-map` subcommand in remote mode to delete a
work security map that belongs to a specific connector module (resource adapter).

1 Ensure that the server is running.
Remote subcommands require a running server.

2 List the connector work security maps by using the `list-connector-work-security-maps`**(1)
subcommand.**

3 Delete a connector work security map by using the `delete-connector-work-security-map`**(1)
subcommand.**

Example 15–19 Deleting a Connector Work Security Map

This example deletes the `worksecuritymap1` map from the `my_ra` connector module.

```
asadmin> delete-connector-work-security-map --raname my_ra worksecuritymap1
Command delete-connector-work-security-map executed successfully.
```

See Also You can also view the full syntax and options of the subcommand by typing `asadmin help`
`delete-connector-work-security-map` at the command line.

Chapter 15 • Administering EIS Connectivity 269

Administering Administered Objects

Packaged within a connector module, an administered object provides specialized functionality for an application. For example, an administered object might provide access to a parser that is specific to the connector module and its associated EIS.

The following topics are addressed here:

▼ To Create an Administered Object

Use the create-admin-object subcommand to create an administered object resource. When creating an administered object resource, name-value pairs are created, and the object is associated to a JNDI name.

Before You Begin   The resource adapter must be deployed before running this subcommand (jmsrar.rar).

1   **Create an administered object by using the** create-admin-object**(1) subcommand.**

Information about properties for the subcommand is included in this help page.

2   **(Optional) If needed, restart the server.**

Some properties require server restart. See "Configuration Changes That Require Server Restart" on page 37. If your server needs to be restarted, see "To Restart a Domain" on page 90.

Example 15–20   Creating an Administered Object

For this example, the javax.jms.Queue resource type is obtained from the ra.xml file. The JNDI name of the new administered object is jms/samplequeue.

```
asadmin> create-admin-object --restype javax.jms.Queue --raname jmsra  --description "sample administered object"
--property Name=sample_jmsqueue jms/samplequeueCommand create-admin-object executed successfully
```

See Also   You can also view the full syntax and options of the subcommand by typing asadmin help create-admin-object at the command line.

▼ To List Administered Objects

Use the list-admin-object subcommand in remote mode to list the existing administered objects.

1   Ensure that the server is running.

Remote subcommands require a running server.

2   List the administered objects by using the list-admin-objects**(1) subcommand.**

Example 15–21   Listing Administered Objects

This example lists the existing administered objects.

```
asadmin> list-admin-objects
jms/samplequeue
Command list-admin-objects executed successfully
```

See Also   You can also view the full syntax and options of the subcommand by typing asadmin help list-admin-object at the command line.

▼ To Update an Administered Object

Use the get and set subcommands to view and change the values of the administered objects properties.

1   List the administered objects by using the list-admin-objects**(1) subcommand.**

2   View the properties of the administered object by using the get**(1) subcommand.**

For example:

```
asadmin> get domain.resources.admin-object-resource.jms/samplequeue.*
```

3   Set the property of the administered object by using the set**(1) subcommand.**

For example:

```
asadmin> set domain.resources.admin-object-resource.jms/samplequeue.enabled=false
```

4   (Optional) If needed, restart the server.

Some properties require server restart. See "Configuration Changes That Require Server Restart" on page 37. If your server needs to be restarted, see "To Restart a Domain" on page 90.

▼ To Delete an Administered Object

Use the delete-admin-object subcommand to delete an administered objects.

1 List the administered objects by using the list-admin-objects**(1) subcommand.**

2 If necessary, notify users that the administered object is being deleted.

3 Delete an administered object by using the delete-admin-object**(1) subcommand.**

Example 15–22 Deleting an Administered Object

This example deletes the administered object with the JNDI name jms/samplequeue.

```
asadmin> delete-admin-object jms/samplequeue
Command delete-admin-object executed successfully
```

See Also You can also view the full syntax and options of the subcommand by typing asadmin help delete-admin-object at the command line.

Administering Internet Connectivity

This chapter provides procedures for performing internet connectivity tasks in the Sun GlassFish™ Enterprise Server v3 environment by using the `asadmin` command-line utility.

The following topics are addressed here:

Instructions for accomplishing the tasks in this chapter by using the Administration Console are contained in the Administration Console online help.

About Internet Connectivity

The HTTP service provides functionality for deploying web applications and for making deployed web applications accessible by Internet clients. HTTP services are provided by two kinds of related objects: listeners and virtual servers.

The following topics are addressed here:

About HTTP Network Listeners

An *HTTP listener*, also known as a *network listener*, is a listen socket that has an Internet Protocol (IP) address, a port number, a server name, and a default virtual server. Each virtual server provides connections between the server and clients through one or more listeners. Each listener must have a unique combination of port number and IP address. For example, an

HTTP listener can listen for a host on all configured IP addresses on a given port by specifying the IP address 0.0.0.0. Alternatively, the listener can specify a unique IP address for each listener while using the same port.

Because an HTTP listener is a combination of IP address and port number, you can have multiple HTTP listeners with the same IP address and different port numbers, or with different IP addresses and the same port number (if your host was configured to respond to these addresses). However, if an HTTP listener uses the 0.0.0.0 IP address, which listens on all IP addresses on a port, you cannot create HTTP listeners for additional IP addresses that listen on the same port for a specific IP address. For example, if an HTTP listener uses 0.0.0.0:8080 (all IP addresses on port 8080), another HTTP listener cannot use 1.2.3.4:8080. The host running the Enterprise Server typically has access to only one IP address. HTTP listeners typically use the 0.0.0.0 IP address and different port numbers, with each port number serving a different purpose. However, if the host does have access to more than one IP address, each address can serve a different purpose.

To access a web application deployed on Enterprise Server, use the URL `http://localhost:8080/` (or `https://localhost:8081/` for a secure application), along with the context root specified for the web application.

To access the Administration Console, use the URL `https://localhost:4848/` or `http://localhost:4848/asadmin/` (console default context root).

About Virtual Servers

A *virtual server*, sometimes called a virtual host, is an object that allows the same physical server to host multiple Internet domain names. All virtual servers hosted on the same physical server share the IP address of that physical server. A virtual server associates a domain name for a server (such as `www.aaa.com`) with the particular server on which Enterprise Server is running. Each virtual server must be registered with the DNS server for your network.

Note – Do not confuse an Internet domain with the administrative domain of Enterprise Server.

For example, assume that you want to host the following domains on your physical server: `www.aaa.com`, `www.bbb.com`, and `www.ccc.com`. Assume that these domains are respectively associated with web modules `web1`, `web2`, and `web3`. This means that the following URLs are handled by your physical server:

```
http://www.aaa.com:8080/web1
http://www.bbb.com:8080/web2
http://www.ccc.com:8080/web3
```

The first URL is mapped to virtual server www.aaa.com, the second URL is mapped to virtual server www.bbb.com, and the third is mapped to virtual server www.ccc.com. For this mapping to work, www.aaa.com, www.bbb.com, and www.ccc.com must all resolve to your physical server's IP address and each virtual server must be registered with the DNS server for your network. In addition, on a UNIX system, add these domains to your /etc/hosts file (if the setting for hosts in your /etc/nsswitch.conf file includes files).

Administering HTTP Network Listeners

By default, when Enterprise Server starts, the following HTTP listeners are started automatically:

- HTTP listeners associated with the virtual server named server:
  - The listener named http-listener-1 does not have security enabled.
  - The listener named http-listener-2 has security enabled
- An HTTP listener named admin-listener, associated with the virtual server named __asadmin. For this listener, security is not enabled.

The following table describes the Enterprise Server default ports for the listeners that use ports.

TABLE 16–1   Default Ports for Listeners

Listener	Default Port	Description
Administrative server	4848	A domain's administrative server is accessed by the Administration Console and the asadmin utility. For the Administration Console, specify the port number in the URL of the browser. When running an asadmin subcommand remotely, specify the port number by using the --port option.
HTTP	8080	The web server listens for HTTP requests on a port. To access deployed web applications and services, clients connect to this port.
HTTPS	8181	Web applications configured for secure communications listen on a separate port.

The following topics are addressed here:

▼ To Create an Internet Connection

Use the subcommands in this procedure to create an internet connection with the full range of listener options. A network listener is created behind the scenes. For the shortcut version of this process , see "To Create an HTTP Network Listener" on page 281.

1 Ensure that the server is running.

Remote subcommands require a running server.

2 Create an HTTP or HTTPS protocol by using the create-protocol**(1) subcommand with the** --securityenabled **option.**

To use the built-in http-listener-1 HTTP protocol, or http-listener-2 HTTPS protocol, skip this step.

3 Create an HTTP configuration by using the create-http**(1) subcommand.**

To use a built-in protocol, skip this step.

4 Create a transport by using the create-transport**(1) subcommand.**

To use the built-in tcp transport, skip this step.

5 (Optional) Create a thread pool by using the create-threadpool**(1) subcommand.**

To avoid using a thread pool, or to use the built-in http-thread-pool thread pool, skip this step.

For additional thread pool information, see Chapter 5, "Administering Thread Pools."

6 Create an HTTP listener by using the create-network-listener**(1) subcommand.**

Specify a protocol and transport, optionally a thread pool.

7 To apply your changes, restart Enterprise Server.

See "To Restart a Domain" on page 90.

See Also You can also view the full syntax and options of the subcommand by typing a command such as asadmin help create-http-listener at the command line.

Administering HTTP Protocols

Each HTTP listener has an HTTP protocol, which is created either by using the create-protocol subcommand or by using the built-in protocols that are applied when you follow the instructions in "To Create an HTTP Network Listener" on page 281.

The following topics are addressed here:

▼ To Create a Protocol

Use the create-protocol subcommand in remote mode to create a protocol.

1 Ensure that the server is running.

Remote subcommands require a running server.

2 Create a protocol by using the create-protocol**(1)**

Information about options and properties for the subcommand are included in this help page.

Example 16–1 Creating an HTTP Protocol

This example creates a protocol named http-1 with security enabled.

```
asadmin> create-protocol --securityenabled=true http-1
Command create-protocol executed successfully.
```

See Also You can also view the full syntax and options of the subcommand by typing asadmin help create-protocol at the command line.

▼ To List Protocols

Use the list-protocols subcommand in remote mode to list the existing HTTP protocols.

1 Ensure that the server is running.

Remote subcommands require a running server.

2 List the existing protocols by using the list-protocols**(1) subcommand.**

Example 16–2 Listing the Protocols

This example lists the existing protocols.

```
asadmin> list-protocols
admin-listener
http-1
http-listener-1
http-listener-2
Command list-protocols executed successfully.
```

See Also   You can also view the full syntax and options of the subcommand by typing asadmin help
list-protocols at the command line.

▼ To Delete a Protocol

Use the delete-protocol subcommand in remote mode to remove a protocol.

1   **Ensure that the server is running.**

Remote subcommands require a running server.

2   **Delete a protocol by using the** delete-protocol(1) **subcommand**

Example 16–3   Deleting a Protocol

This example deletes the protocol named http-1.

```
asadmin> delete-protocol http-1
Command delete-protocol executed successfully.
```

See Also   You can also view the full syntax and options of the subcommand by typing asadmin help
delete-protocol at the command line.

Administering HTTP Configurations

Each HTTP listener has an HTTP configuration, which is created either by using the
create-http subcommand or by using the built-in configurations that are applied when you
follow the instructions in "To Create an HTTP Network Listener" on page 281.

The following topics are addressed here:

- "To Create an HTTP Configuration" on page 278
- "To Delete an HTTP Configuration" on page 279

▼ To Create an HTTP Configuration

Use the create-http subcommand in remote mode to create a set of HTTP parameters for a
protocol. This set of parameters configures one or more network listeners,

1   **Ensure that the server is running.**

Remote subcommands require a running server.

2   **Create an HTTP configuration by using the** create-http(1) **subcommand.**

Information about options and properties for the subcommand are included in this help page.

Example 16–4 Creating an HTTP Configuration

This example creates an HTTP parameter set for the protocol named http-1.

```
asadmin> create-http --timeout-seconds 60 --default-virtual-server server http-1
Command create-http executed successfully.
```

See Also You can also view the full syntax and options of the subcommand by typing asadmin help create-http at the command line.

▼ To Delete an HTTP Configuration

Use the delete-http subcommand in remote mode to remove HTTP parameters from a protocol.

1 Ensure that the server is running.

Remote subcommands require a running server.

2 Delete the HTTP parameters from a protocol by using the delete-http**(1) subcommand.**

Example 16–5 Deleting an HTTP Configuration

This example deletes the HTTP parameter set from a protocol named http-1.

```
asadmin> delete-http http-1
Command delete-http executed successfully.
```

See Also You can also view the full syntax and options of the subcommand by typing asadmin help delete-http at the command line.

Administering HTTP Transports

Each HTTP listener has an HTTP transport, which is created either by using the create-transport subcommand or by using the built-in transports that are applied when you follow the instructions in "To Create an HTTP Network Listener" on page 281.

The following topics are addressed here:

- "To Create a Transport" on page 280
- "To List Transports" on page 280
- "To Delete a Transport" on page 281

▼ To Create a Transport

Use the create-transport subcommand in remote mode to create a transport for a network listener,

1 Ensure that the server is running.

Remote subcommands require a running server.

2 Create a transport by using the create-transport**(1) subcommand.**

Information about options and properties for the subcommand are included in this help page.

Example 16–6 Creating a Transport

This example creates a transport named http1-trans that uses a non-default number of acceptor threads.

```
asadmin> create-transport --acceptorthreads 100 http1-trans
Command create-transport executed successfully.
```

See Also You can also view the full syntax and options of the subcommand by typing asadmin help create-transport at the command line.

▼ To List Transports

Use the list-transports subcommand in remote mode to list the existing HTTP transports.

1 Ensure that the server is running.

Remote subcommands require a running server.

2 List the existing transports by using the list-transports**(1) subcommand.**

Example 16–7 Listing HTTP Transports

This example lists the existing transports.

```
asadmin> list-transports
http1-trans
tcp
Command list-transports executed successfully.
```

See Also You can also view the full syntax and options of the subcommand by typing asadmin help list-transports at the command line.

▼ To Delete a Transport

Use the delete-transport subcommand in remote mode to remove a transport.

1 Ensure that the server is running.

Remote subcommands require a running server.

2 Delete a transport by using the delete-transport**(1) subcommand.**

Example 16–8 Deleting a Transport

This example deletes he transport named http1-trans.

```
asadmin> delete-transport http1-trans
Command delete-transport executed successfully.
```

See Also You can also view the full syntax and options of the subcommand by typing asadmin help delete-transport at the command line.

Administering HTTP Network Listeners

The following topics are addressed here:

▼ To Create an HTTP Network Listener

Use the create-http-listener subcommand or the create-network-listener subcommand in remote mode to create a listener. These subcommands provide backward compatibility and also provide a shortcut for creating network listeners that use the HTTP protocol. Behind the scenes, a network listener is created as well as its associated protocol, transport, and HTTP configuration. This method is a convenient shortcut, but it gives access to only a limited number of options. If you want to specify the full range of listener options, follow the instructions in .

1 Ensure that the server is running.

Remote subcommands require a running server.

2   **Create an HTTP network listener by using the** `create-network-listener`**(1) subcommand or the** `create-http-listener`**(1) subcommand.**

3   **If needed, restart the server.**

If you edit the special HTTP network listener named `admin-listener`, you must restart the server for changes to take effect. See "To Restart a Domain" on page 90.

Example 16–9   Creating an HTTP Listener

This example creates an HTTP listener named `sampleListener` that uses a non-default number of acceptor threads. Security is not enabled at runtime.

```
asadmin> create-http-listener --listeneraddress 0.0.0.0
--listenerport 7272 --defaultvs server --servername host1.sun.com
--acceptorthreads 100 --securityenabled=false
--enabled=false sampleListener
Command create-http-listener executed successfully.
```

Example 16–10   Creating a Network Listener

This example a network listener named `sampleListener` that is not enabled at runtime:

```
asadmin> create-network-listener --listenerport 7272 protocol http-1
--enabled=false sampleListener
Command create-network-listener executed successfully.
```

See Also   You can also view the full syntax and options of the subcommand by typing `asadmin help create-http-listener` or `asadmin help create-network-listener` at the command line.

▼   **To List HTTP Network Listeners**

Use the `list-http-listeners` subcommand or the `list-network-listeners` subcommand in remote mode to list the existing HTTP listeners.

1   **Ensure that the server is running.**

Remote subcommands require a running server.

2   **List HTTP listeners by using the** `list-http-listeners`**(1) or** `list-network-listeners`**(1) subcommand.**

Example 16–11   Listing HTTP Listeners

This example lists the HTTP listeners. The same output is given if you use the `list-network-listeners` subcommand.

```
asadmin> list-http-listeners
admin-listener
http-listener-2
http-listener-1
Command list-http-listeners executed successfully.
```

See Also You can also view the full syntax and options of the subcommand by typing asadmin help list-http-listeners or asadmin help list-network-listeners at the command line.

▼ To Update an HTTP Network Listener

1 List HTTP listeners by using the list-http-listeners(1) or list-network-listeners(1) subcommand.

2 Modify the values for the specified listener by using the set(1) subcommand.
The listener is identified by its dotted name.

Example 16–12 Updating an HTTP Network Listener

This example changes security-enabled to false.

```
asadmin> set "server.network-config.protocols.protocol.
http-listener-2.security-enabled=false"server.network-config.
protocols.protocol.http-listener-2.security-enabled=false
Command set executed successfully.
```

▼ To Delete an HTTP Network Listener

Use the delete-http-listener subcommand or the delete-network-listener subcommand in remote mode to delete an existing HTTP listener. This disables secure communications for the listener.

1 Ensure that the server is running.
Remote subcommands require a running server.

2 List HTTP listeners by using the list-http-listeners(1) subcommand.

3 Delete an HTTP listener by using the delete-http-listener(1) or delete-network-listener(1) subcommand.

4 To apply your changes, restart Enterprise Server.
See "To Restart a Domain" on page 90.

Example 16–13 Deleting an HTTP Listener

This example deletes the HTTP listener named `sampleListener`:

```
asadmin> delete-http-listener sampleListener
Command delete-http-listener executed successfully.
```

See Also You can also view the full syntax and options of the subcommand by typing `asadmin help delete-http-listener` or `asadmin help delete-network-listener` at the command line.

▼ To Configure an HTTP Listener for SSL

Use the `create-ssl` subcommand in remote mode to create and configure an SSL element in the specified listener. This enables secure communication for the listener.

1 Ensure that the server is running.

Remote subcommands require a running server.

2 Configure an HTTP listener by using the `create-ssl`**(1) subcommand.**

3 To apply your changes, restart Enterprise Server.

See "To Restart a Domain" on page 90.

Example 16–14 Configuring an HTTP Listener for SSL

This example enables the HTTP listener named `http-listener-1` for SSL:

```
asadmin> create-ssl --type http-listener --certname sampleCert http-listener-1
Command create-ssl executed successfully.
```

See Also You can also view the full syntax and options of the subcommand by typing `asadmin help create-ssl` at the command line.

▼ To Delete SSL From an HTTP Listener

Use the `delete-ssl` subcommand in remote mode to delete the SSL element in the specified listener. This disables secure communications for the listener.

1 Ensure that the server is running.

Remote subcommands require a running server.

2 Delete SSL from an HTTP listener by using the `delete-ssl`**(1) subcommand.**

3 To apply your changes, restart Enterprise Server.

See "To Restart a Domain" on page 90.

Example 16–15 Deleting SSL From an HTTP Listener

This example disables SSL for the HTTP listener named http-listener-1:

```
asadmin> delete-ssl --type http-listener http-listener-1
Command delete-http-listener executed successfully.
```

See Also You can also view the full syntax and options of the subcommand by typing asadmin help delete-ssl at the command line.

▼ **To Assign a Default Virtual Server to an HTTP Listener**

tbd

1 In the Administration Console, open the HTTP Service component under the relevant configuration.

2 Open the HTTP Listeners component under the HTTP Service component.

3 Select or create a new HTTP listener.

4 Select from the Default Virtual Server drop-down list.

For more information, see "To Assign a Default Web Module to a Virtual Server" on page 288.

See Also For details, click the Help button in the Administration Console from the HTTP Listeners page.

Administering Virtual Servers

A virtual server is a virtual web server that serves content targeted for a specific URL. Multiple virtual servers can serve content using the same or different host names, port numbers, or IP addresses. The HTTP service directs incoming web requests to different virtual servers based on the URL.

When you first install Enterprise Server, a default virtual server is created. You can assign a default virtual server to each new HTTP listener you create.

Web applications and Java EE applications containing web components (web modules) can be assigned to virtual servers during deployment. A web module can be assigned to more than one virtual server, and a virtual server can have more than one web module assigned to it. If you deploy a web application and don't specify any assigned virtual servers, the web application is assigned to all currently defined virtual servers. If you then create additional virtual servers and want to assign existing web applications to them, you must redeploy the web applications. For more information about deployment, see the *Sun GlassFish Enterprise Server v3 Application Deployment Guide*.

You can define virtual server properties using the asadmin set command. For example:

```
asadmin set server-config.http-service.virtual-server.MyVS.property.sso-enabled="true"
```

Some virtual server properties can be set for a specific web application. For details, see "sun-web-app" in *Sun GlassFish Enterprise Server v3 Application Deployment Guide*.

The following topics are addressed here:

▼ To Create a Virtual Server

By default, when Enterprise Server starts, the following virtual servers are started automatically:

- A virtual server named server, which hosts all user-defined web modules.

  For development, testing, and deployment of web services in a non-production environment, server is often the only virtual server required.

- A virtual server named __asadmin, which hosts all administration-related web modules (specifically, the Administration Console). This server is restricted, which means that you cannot deploy web modules to this virtual server.

In a production environment, additional virtual servers provide hosting facilities for users and customers so that each appears to have its own web server, even though there is only one physical server.

Use the create-virtual-server subcommand in remote mode to create the named virtual server.

Before You Begin   A virtual server must specify an existing HTTP listener. Because the virtual server cannot specify an HTTP listener that is already being used by another virtual server, create at least one HTTP listener before creating a new virtual server.

1   **Ensure that the server is running.**

Remote subcommands require a running server.

2   **Create a virtual server by using the** create-virtual-server(1) **subcommand.**

Information about properties for this subcommand is included in this help page.

3 To apply your changes, restart Enterprise Server.

See "To Restart a Domain" on page 90.

Example 16–16 Creating a Virtual Server

This example creates a virtual server named sampleServer on localhost.

```
asadmin> create-virtual-server sampleServer
Command create-virtual-server executed successfully.
```

See Also You can also view the full syntax and options of the subcommand by typing asadmin help create-virutal-server at the command line.

▼ To List Virtual Servers

Use the list-virtual-servers subcommand in remote mode to list the existing virtual servers.

1 Ensure that the server is running.

Remote subcommands require a running server.

2 List virtual servers by using the list-virtual-servers(1) **subcommand.**

Example 16–17 Listing Virtual Servers

This example lists the virtual servers for localhost.

```
asadmin> list-virtual-servers
sampleListener
admin-listener
http-listener-2
http-listener-1
Command list-http-listeners executed successfully.
```

See Also You can also view the full syntax and options of the subcommand by typing asadmin help list-virutal-servers at the command line.

▼ To Update a Virtual Server

1 **List virtual servers by using the** `list-virtual-servers`**(1) subcommand.**

2 **Modify the values for the specified virtual server by using the** `set`**(1) subcommand.**
The virtual server is identified by its dotted name.

▼ To Delete a Virtual Server

Use the `delete-virtual-server` subcommand in remote mode to delete an existing virtual server.

1 **Ensure that the server is running.**
Remote subcommands require a running server.

2 **List virtual servers by using the** `list-virtual-servers`**(1) subcommand.**

3 **If necessary, notify users that the virtual server is being deleted.**

4 **Delete a virtual server by using the** `delete-virtual-server`**(1) subcommand.**

5 **To apply your changes, restart Enterprise Server.**
See "To Restart a Domain" on page 90.

Example 16–18 Deleting a Virtual Server

This example deletes the virtual server named `sampleServer` from `localhost`.

```
asadmin> delete-virtual-server sampleServer
Command delete-virtual-server executed successfully.
```

See Also You can also view the full syntax and options of the subcommand by typing `asadmin help delete-virutal-server` at the command line.

To Assign a Default Web Module to a Virtual Server

A default web module can be assigned to the default virtual server and to each new virtual server. To access the default web module for a virtual server, point the browser to the URL for the virtual server, but do not supply a context root. For example:

```
http://myvserver:3184/
```

A virtual server with no default web module assigned serves HTML or JavaServer Pages™ (JSP™) content from its document root, which is usually *domain-dir*/`docroot`. To access this HTML or JSP content, point your browser to the URL for the virtual server, do not supply a context root, but specify the target file.

For example:

```
http://myvserver:3184/hellothere.jsp
```

▼ To Assign a Virtual Server to an Application or Module

You can assign a virtual server to a deployed application or web module.

Before You Begin   The application or module must already be deployed. For more information, see *Sun GlassFish Enterprise Server v3 Application Deployment Guide*.

1   **In the Administration Console, open the HTTP Service component under the relevant configuration.**

2   **Open the Virtual Servers component under the HTTP Service component.**

3   **Select the virtual server to which you want to assign a default web module.**

4   **Select the application or web module from the Default Web Module drop-down list.**

For more information, see "To Assign a Default Web Module to a Virtual Server" on page 288.

Administering the Object Request Broker (ORB)

The Sun GlassFish™ Enterprise Server supports a standard set of protocols and formats that ensure interoperability. Among these protocols are those defined by CORBA. The Object Request Broker (ORB) is the central component of CORBA. The ORB provides the required infrastructure to identify and locate objects, handle connection management, deliver data, and request communication. This chapter describes how to configure the ORB and the IIOP listeners.

The following topics are addressed here:

- "About the ORB" on page 291
- "Configuring the ORB" on page 292
- "Administering IIOP Listeners" on page 292

Instructions for accomplishing the tasks in this chapter by using the Administration Console are contained in the Administration Console online help.

About the ORB

The Common Object Request Broker Architecture (CORBA) model is based on clients requesting services from distributed objects or servers through a well-defined interface by issuing requests to the objects in the form of remote method requests. A *remote method request* carries information about the operation that needs to be performed, including the object name (called an object reference) of the service provider and parameters, if any, for the invoked method. CORBA automatically handles network programming tasks such as object registration, object location, object activation, request de-multiplexing, error-handling, marshalling, and operation dispatching.

Configuring the ORB

A CORBA object never talks directly with another. Instead, the object makes requests through a remote stub to the Internet Inter-Orb Protocol (IIOP) running on the local host. The local ORB then passes the request to an ORB on the other host using IIOP. The remote ORB then locates the appropriate object, processes the request, and returns the results.

IIOP can be used as a Remote Method Invocation (RMI) protocol by applications or objects using RMI-IIOP. Remote clients of enterprise beans (EJB modules) communicate with Enterprise Server by using RMI-IIOP.

Administering IIOP Listeners

An *IIOP listener* is a listen socket that accepts incoming connections from the remote clients of enterprise beans and from other CORBA-based clients. Multiple IIOP listeners can be configured for Enterprise Server. For each listener, specify a port number (optional; default 1072), a network address, and security attributes (optional). If you create multiple listeners, you must assign a different port number for each listener.

The following topics are addressed here:

- "To Create an IIOP Listener" on page 292
- "To List IIOP Listeners" on page 293
- "To Update an IIOP Listener" on page 293
- "To Delete an IIOP Listener" on page 294

▼ To Create an IIOP Listener

Use the create-iiop-listener subcommand in remote mode to create an IIOP listener.

1 Ensure that the server is running.

Remote subcommands require a running server.

2 Create an IIOP listener by using the create-iiop-listener**(1) subcommand.**

Information about the properties for the subcommand is included in this help page.

3 To apply your changes, restart Enterprise Server.

See "To Restart a Domain" on page 90.

Example 17–1 Creating an IIOP Listener

This example creates an IIOP listener named sample_iiop_listener.

```
asadmin> create-iiop-listener --listeneraddress 192.168.1.100
--iiopport 1400 sample_iiop_listener
Command create-iiop-listener executed successfully.
```

See Also You can also view the full syntax and options of the subcommand by typing asadmin help create-iiop-listener at the command line.

▼ To List IIOP Listeners

Use the list-iiop-listeners subcommand in remote mode to list the existing IIOP listeners.

1 Ensure that the server is running.

Remote subcommands require a running server.

2 List the IIOP listeners by using the list-iiop-listeners(1) **subcommand.**

Example 17–2 Listing IIOP Listeners

This example lists all the IIOP listeners for the server instance.

```
asadmin> list-iiop-listeners
orb-listener-1
SSL
SSL_MUTUALAUTH
sample_iiop_listener
Command list-iiop-listeners executed successfully.
```

See Also You can also view the full syntax and options of the subcommand by typing asadmin help list-iiop-listeners at the command line.

▼ To Update an IIOP Listener

1 List the IIOP listeners by using the list-iiop-listeners(1) **subcommand.**

2 Modify the values for the specified IIOP listener by using the set(1) **subcommand.**

The listener is identified by its dotted name.

Example 17–3 Updating an IIOP Listener

This example changes SSL from enabled to disabled.

```
asadmin> set "server.iiop-service.iiop-listener.SSL.enabled"
server.iiop-service.iiop-listener.SSL.enabled=false
Command set executed successfully.
```

▼ To Delete an IIOP Listener

Use the delete-iiop-listener subcommand in remote mode to delete an IIOP listener.

1   Ensure that the server is running.

Remote subcommands require a running server.

2   List the IIOP listeners by using the list-iiop-listeners**(1) subcommand.**

3   Delete an IIOP listener by using the delete-iiop-listener**(1) subcommand.**

4   To apply your changes, restart Enterprise Server.

See "To Restart a Domain" on page 90.

Example 17–4   Deleting an IIOP Listener

This example deletes the IIOP listener named sample_iiop_listener.

```
asadmin> delete-iiop-listener sample_iiop_listener
 Command delete-iiop-listener executed successfully.
```

See Also   You can also view the full syntax and options of the subcommand by typing asadmin help delete-iiop-listener at the command line.

Administering the JavaMail Service

The Sun Java™ Enterprise Server includes the JavaMail API along with JavaMail service providers that allow an application component to send email notifications over the Internet and to read email from IMAP and POP3 mail servers.

The following topics are addressed here:

- "About JavaMail" on page 295
- "Administering JavaMail Resources" on page 296

Instructions for accomplishing the tasks in this chapter by using the Administration Console are contained in the Administration Console online help.

About JavaMail

The JavaMail API is a set of abstract APIs that model a mail system. The JavaMail API provides a platform-independent and protocol-independent framework to build mail and messaging applications and provide facilities for reading and sending electronic messages. Service providers implement particular protocols. Using the API you can add email capabilities to your applications. JavaMail provides access from Java applications to Internet Message Access Protocol (IMAP) and Simple Mail Transfer Protocol (SMTP) capable mail servers on your network or the Internet. The API does not provide mail server functionality; you must have access to a mail server to use JavaMail.

The JavaMail API is implemented as an optional package in the Java platform and is also available as part of the Java EE platform.

To learn more about the JavaMail API, consult the JavaMail web site at `http://java.sun.com/products/javamail/`.

Administering JavaMail Resources

When you create a mail session, the server-side components and applications are enabled to access JavaMail services with JNDI, using the session properties you assign for them. When creating a mail session, you can designate the mail hosts, the transport and store protocols, and the default mail user so that components that use JavaMail do not have to set these properties. Applications that are heavy email users benefit because Enterprise Server creates a single session object and makes the session available to any component that needs it.

JavaMail settings such as the following can be specified:

- **JNDI Name.** The unique name for the mail session. Use the naming sub-context prefix mail/ for JavaMail resources. For example: `mail/MySession`

- **Mail Host.** The host name of the default mail server. The connect methods of the store and transport objects use this value if a protocol-specific host property is not supplied. The name must be resolvable to an actual host name.

- **Default User.** The default user name to provide when connecting to a mail server. The connect methods of the store and transport objects use this value if a protocol-specific username property is not supplied.

- **Default Return Address.** The email address of the default user, in the form: *username@host.domain*.

- **Description.** A descriptive statement for the component.

- **Session.** Indicates whether or not mail session is enabled or disabled at this time

The following topics are addressed here:

▼ To Create a JavaMail Resource

Use the `create-javamail-resource` subcommand in remote mode to create a JavaMail session resource. The JNDI name for a JavaMail session resource customarily includes the mail/ naming subcontext, For example: `mail/MyMailSession`.

1 Ensure that the server is running.

Remote subcommands require a running server.

2 Create a JavaMail resource by using the `create-javamail-resource`(1) subcommand.

Information about the properties for the subcommand is included in this help page.

3 To apply your changes, restart Enterprise Server.

See "To Restart a Domain" on page 90.

Example 18–1 Creating a JavaMail Resource

This example creates a JavaMail resource named `mail/MyMailSession`. The escape character (\) is used in the `--fromaddress` option to distinguish the dot (.) and at sign (@).

```
asadmin> create-javamail-resource --mailhost localhost
--mailuser sample --fromaddress sample\@sun\.com mail/MyMailSession
Command create-javamail-resource executed successfully.
```

See Also You can also view the full syntax and options of the subcommand by typing `asadmin help create-javamail-resource` at the command line.

▼ To List JavaMail Resources

Use the `list-javamail-resources` subcommand in remote mode to list the existing JavaMail session resources.

1 Ensure that the server is running.

Remote subcommands require a running server.

2 List the JavaMail resources by using the `list-javamail-resources`**(1) subcommand.**

Example 18–2 Listing JavaMail Resources

This example lists the JavaMail resources on `localhost`.

```
asadmin> list-javamail-resources
mail/MyMailSession
Command list-javamail-resources executed successfuly.
```

See Also You can also view the full syntax and options of the subcommands by typing `asadmin help list-javamail-resources` at the command line.

▼ To Update a JavaMail Resource

1 List the JavaMail resources by using the `list-javamail-resources`**(1) subcommand.**

2 Modify the values for the specified JavaMail source by using the `set`**(1) subcommand.**

The resource is identified by its dotted name.

Example 18–3   Updating a JavaMail Resource

This example changes `joeserver` to `joe`.

```
asadmin> set server.resources.mail-resource.mail/
MyMailSession.user=joeserver.resources.mail-resource.mail/
MyMailSession.user=joe
Command set executed successfully.
```

▼ To Delete a JavaMail Resource

Use the `delete-javamail-resource` subcommands in remote mode to delete a JavaMail session resource.

Before You Begin   References to the specified resource must be removed before running the `delete-javamail-resource` subcommands.

1   **Ensure that the server is running.**

Remote subcommands require a running server.

2   **List the JavaMail resources by using the** `list-javamail-resources(1)` **subcommands.**

3   **Delete a JavaMail resource by using the** `delete-javamail-resource(1)` **subcommands.**

4   **To apply your changes, restart Enterprise Server.**

See "To Restart a Domain" on page 90.

Example 18–4   Deleting a JavaMail Resource

This example deletes the JavaMail session resource named `mail/MyMailSession`.

```
asadmin> delete-javamail-resource  mail/MyMailSession
Command delete-javamail-resource executed successfully.
```

See Also   You can also view the full syntax and options of the subcommand by typing `asadmin help delete-javamail-resource` at the command line.

19

Administering the Java Message Service (JMS)

Sun implements the Java™ Message Service (JMS) API by integrating the Sun GlassFish™ Message Queue software into Sun GlassFish Enterprise Server. This chapter provides procedures for administering JMS resources in the Enterprise Server environment by using the `asadmin` command-line utility.

Note – JMS resources are supported only in the Full Platform Profile of Enterprise Server, not in the Web Profile.

The following topics are addressed here:

Instructions for accomplishing the task in this chapter by using the Administration Console are contained in the Administration Console online help.

About the JMS

The JMS API is a messaging standard that allows Java EE applications and components to create, send, receive, and read messages. It enables distributed communication that is loosely coupled, reliable, and asynchronous.

Enterprise Server support for JMS messaging, in general, and for message-driven beans in particular, requires a *JMS provider*. Enterprise Server uses the Sun GlassFish Message Queue software as its native JMS provider, providing transparent JMS messaging support. This

support is known within Enterprise Server as the *JMS Service*. JMS requires only minimal administration. When a JMS client accesses a JMS administered object for the first time, the client JVM retrieves the JMS configuration from Enterprise Server.

A JMS resource is a type of connector. Message Queue is integrated with Enterprise Server by means of a *connector module*, also known as a resource adapter, which is defined by the Java EE Connector Architecture Specification 1.6. Any Java EE components that are deployed to Enterprise Server exchange JMS messages by using the JMS provider that is integrated by the connector module. When a JMS resource is created in Enterprise Server, a connector resource is created in the background. Each JMS operation invokes the connector runtime and uses the Message Queue connector module in the background. Enterprise Server pools JMS connections automatically.

You can configure properties to be used by all JMS connections. If you update these properties at runtime, only those connection factories that are created after the properties are updated will apply the updated values. The existing connection factories will continue to have the original property values. For most values to take effect, Enterprise Server must be restarted. For instructions, see "To Restart a Domain" on page 90. The only property that can be updated without restarting Enterprise Server is the default JMS host.

Message Queue Broker Modes

Message Queue can be integrated with Enterprise Server in LOCAL, REMOTE, or EMBEDDED mode. These modes are represented by the JMS type attribute.

- **LOCAL Mode.** Enterprise Server starts and stops the Message Queue broker that is specified as the default JMS host. The Message Queue process is started in a separate virtual machine from the Enterprise Server process. Enterprise Server supplies an additional port to the broker, which is used by the broker to start the RMI registry. This port number will be equal to the configured JMS port for that instance, plus 100. For example, if the JMS port number is 37676, then this additional port number is 37776.

  In LOCAL mode, use the Start Arguments attribute to specify Message Queue broker startup parameters.

- **REMOTE Mode.** When the type attribute is set to REMOTE, the Message Queue broker must be started and stopped separately from Enterprise Server. Message Queue tools must be used to configure and tune the broker. In this situation, Enterprise Server uses an externally–configured broker or broker cluster. REMOTE type is most suitable for clusters.

  In REMOTE mode, you must specify Message Queue broker startup parameters using Message Queue tools. The Start Arguments attribute is ignored.

- **EMBEDDED Mode (default).** When the JMS type attribute is set to EMBEDDED, Enterprise Server and the JMS broker are colocated in the same virtual machine. The JMS Service is started in-process and managed by Enterprise Server.

In EMBEDDED mode, the JMS operations bypass the networking stack, which leads to performance optimization.

For information about administering Message Queue, see *Sun GlassFish Message Queue 4.4 Administration Guide*.

Administering JMS Physical Destinations

Messages are delivered for routing and delivery to consumers by using *physical destinations* in the JMS provider. A physical destination is identified and encapsulated by an administered object (such as a `Topic` or `Queue` destination resource) that an application component uses to specify the destination of messages it is producing and the source of messages it is consuming. For instructions on configuring a destination resource, see "To Create a Connection Factory or Destination Resource" on page 305.

If a message-driven bean is deployed and the physical destination it listens to does not exist, Enterprise Server automatically creates the physical destination and sets the value of the `maxNumActiveConsumers` property to `-1`. However, it is good practice to create the physical destination beforehand. The first time that an application accesses a destination resource, Message Queue automatically creates the physical destination specified by the Name property of the destination resource. The physical destination is temporary and expires after a period specified by a Message Queue configuration property.

The following topics are addressed here:

- "To Create a JMS Physical Destination" on page 301
- "To List JMS Physical Destinations" on page 302
- "To Purge Messages From a Physical Destination" on page 303
- "To Delete a JMS Physical Destination" on page 303

▼ To Create a JMS Physical Destination

For production purposes, always create physical destinations. During the development and testing phase, however, this step is not required. Use the `create-jmsdest` subcommand in remote mode to create a physical destination.

Because a physical destination is actually a Message Queue object rather than a server object, you use Message Queue broker commands to update properties. For information on Message Queue properties, see *Sun GlassFish Message Queue 4.4 Administration Guide*.

1 Ensure that the server is running.

Remote subcommands require a running server.

2 Create a JMS physical destination by using the `create-jmsdest`**(1) subcommand.**

Information about the properties for the subcommand is included in this help page.

3 (Optional) If needed, restart the server.

Some properties require server restart. See "Configuration Changes That Require Server Restart" on page 37. If your server needs to be restarted, see "To Restart a Domain" on page 90.

Example 19–1 Creating a JMS Physical Destination

This example creates a queue named `PhysicalQueue`.

```
asadmin> create-jmsdest --desttype queue --property
User=public:Password=public PhysicalQueue
Command create-jmsdest executed successfully.
```

See Also You can also view the full syntax and options of the subcommand by typing `asadmin help` `create-jmsdest` at the command line.

▼ **To List JMS Physical Destinations**

Use the `list-jmsdest` subcommand in remote mode to list the existing JMS physical destinations.

1 Ensure that the server is running.

Remote subcommands require a running server.

2 List the existing JMS physical destinations by using the `list-jmsdest`**(1) subcommand.**

Example 19–2 Listing JMS Physical Destinations

This example lists the physical destinations for the default server instance.

```
asadmin> list-jmsdest
PhysicalQueue queue {}
PhysicalTopic topic {}
Command list-jmsdest executed successfully.
```

See Also You can also view the full syntax and options of the subcommand by typing `asadmin help` `list-jmsdest` at the command line.

▼ To Purge Messages From a Physical Destination

Use the flush-jmsdest subcommand in remote mode to purge the messages from a physical destination in the specified target's JMS service configuration.

1   Ensure that the server is running.

Remote subcommands require a running server.

2   Purge messages from the a JMS physical destination by using the flush-jmsdest**(1) subcommand.**

3   (Optional) If needed, restart the server.

Some properties require server restart. See "Configuration Changes That Require Server Restart" on page 37. If your server needs to be restarted, see "To Restart a Domain" on page 90.

Example 19–3   Flushing Messages From a JMS Physical Destination

This example purges messages from the queue named PhysicalQueue.

```
asadmin> flush-jmsdest --desttype queue PhysicalQueue
Command flush-jmsdest executed successfully
```

See Also   You can also view the full syntax and options of the subcommand by typing asadmin help flush-jmsdest at the command line.

▼ To Delete a JMS Physical Destination

Use the delete-jmsdest subcommand in remote mode to remove the specified JMS physical destination.

1   Ensure that the server is running.

Remote subcommands require a running server.

2   List the existing JMS physical destinations by using the list-jmsdest**(1) subcommand.**

3   Delete the physical resource by using the delete-jmsdest**(1) subcommand.**

Example 19–4   Deleting a Physical Destination

This example deletes the queue named PhysicalQueue.

```
asadmin> delete-jmsdest --desttype queue PhysicalQueue
Command delete-jmsdest executed successfully
```

See Also You can also view the full syntax and options of the subcommand by typing `asadmin help delete-jmsdest` at the command line.

Administering JMS Connection Factories and Destinations

The JMS API uses two kinds of administered objects. *Connection factory objects* allow an application to create other JMS objects programmatically. *Destination objects* serve as repositories for messages. How these objects are created is specific to each implementation of JMS. In Enterprise Server, JMS is implemented by performing the following tasks:

- Creating a connection factory
- Creating a destination, which requires creating a physical destination and a destination resource that refers to the physical destination

JMS applications use the Java Naming and Directory Interface (JNDI) API to access the connection factory and destination resources. A JMS application normally uses at least one connection factory and at least one destination. By studying the application or consulting with the application developer, you can determine what resources must be created. The order in which the resources are created does not matter.

Enterprise Server provides the following types of connection factory objects:

- `QueueConnectionFactory` objects, used for point-to-point communication
- `TopicConnectionFactory` objects, used for publish-subscribe communication
- `ConnectionFactory` objects, which can be used for both point-to-point and publish-subscribe communications (recommended for new applications)

Enterprise Server provides the following types of destination objects:

- `Queue` objects, used for point-to-point communication
- `Topic` objects, used for publish-subscribe communication

The following topics are addressed here:

- "To Create a Connection Factory or Destination Resource" on page 305
- "To List JMS Resources" on page 306
- "To Delete a Connection Factory or Destination Resource" on page 307

The subcommands in this section can be used to administer both the connection factory resources and the destination resources. For instructions on administering physical destinations, see "Administering JMS Physical Destinations" on page 301.

▼ To Create a Connection Factory or Destination Resource

For each JMS connection factory that you create, Enterprise Server creates a connector connection pool and connector resource. For each JMS destination that you create, Enterprise Server creates a connector admin object resource. If you delete a JMS resource, Enterprise Server automatically deletes the connector resources.

Use the `create-jms-resource` command in remote mode to create a JMS connection factory resource or a destination resource.

Tip – To specify the `addresslist` property (in the format host:mqport,host2:mqport,host3:mqport) for the asadmin `create-jms-resource` command, escape the : by using \\. For example, host1\\:mqport,host2\\:mqport,host3\\:mpqport. For more information about using escape characters, see the asadmin(1M) concepts page.

To update a JMS connection factory, use the `set` subcommand for the underlying connector connection pool, See "To Update a Connector Connection Pool" on page 257.

To update a destination, use the `set` subcommand for the admin object resource. See "To Update an Administered Object" on page 271.

1 Ensure that the server is running.

Remote subcommands require a running server.

2 Create a JMS resource by using the create-jms-resource(1) **command.**

Information about the properties for the subcommand is included in this help page.

3 (Optional) If needed, restart the server.

Some properties require server restart. See "Configuration Changes That Require Server Restart" on page 37. If your server needs to be restarted, see "To Restart a Domain" on page 90.

Example 19–5 Creating a JMS Connection Factory

This example creates a connection factory resource of type `javax.jms.ConnectionFactory` whose JNDI name is `jms/DurableConnectionFactory`. The `ClientId` property sets a client ID on the connection factory so that it can be used for durable subscriptions. The JNDI name for a JMS resource customarily includes the `jms/` naming subcontext.

```
asadmin> create-jms-resource --restype javax.jms.ConnectionFactory
--description "connection factory for durable subscriptions"
```

```
--property ClientId=MyID jms/DurableConnectionFactory
Command create-jms-resource executed successfully.
```

Example 19–6 Creating a JMS Destination

This example creates a destination resource whose JNDI name is jms/MyQueue.

```
asadmin> create-jms-resource --restype javax.jms.Queue
--property Name=PhysicalQueue jms/MyQueue
Command create-jms-resource executed successfully.
```

See Also You can also view the full syntax and options of the subcommand by typing asadmin help create-jms-resource at the command line.

▼ **To List JMS Resources**

Use the list-jms-resources subcommand in remote mode to list the existing connection factory and destination resources.

1 **Ensure that the server is running.**

Remote subcommands require a running server.

2 **List the existing JMS resources by using the** list-jms-resources(1) **subcommand.**

Example 19–7 Listing All JMS Resources

This example lists all the existing JMS connection factory and destination resources.

```
asadmin> list-jms-resources
jms/Queue
jms/ConnectionFactory
jms/DurableConnectionFactory
jms/Topic
Command list-jms-resources executed successfully
```

Example 19–8 Listing a JMS Resources of a Specific Type

This example lists the resources for the resource type javax.

```
asadmin> list-jms-resources --restype javax.jms.TopicConnectionFactory
jms/DurableTopicConnectionFactory
jms/TopicConnectionFactory
Command list-jms-resources executed successfully.
```

See Also   You can also view the full syntax and options of the subcommand by typing `asadmin help list-jms-resources` at the command line.

▼ To Delete a Connection Factory or Destination Resource

Use the `delete-jms-resource` subcommand in remote mode to remove the specified connection factory or destination resource.

Before You Begin   Ensure that you remove all references to the specified JMS resource before running this subcommand.

1   **Ensure that the server is running.**

Remote subcommands require a running server.

2   **List the existing JMS resources by using the** `list-jms-resources`**(1) subcommand.**

3   **Delete the JMS resource by using the** `delete-jms-resource`**(1) subcommand.**

Example 19–9   Deleting a JMS Resource

This example deletes the `jms/Queue` resource.

```
asadmin> delete-jms-resource jms/Queue
Command delete-jms-resource executed successfully
```

See Also   You can also view the full syntax and options of the subcommand by typing `asadmin help delete-jms-resource` at the command line.

Administering JMS Hosts

A *JMS host* represents a Message Queue broker. JMS contains a *JMS hosts list* (the `AddressList` property) that contains all the JMS hosts that are used by Enterprise Server. The JMS hosts list is populated with the hosts and ports of the specified Message Queue brokers and is updated whenever a JMS host configuration changes. When you create JMS resources or deploy message driven beans, the resources or beans inherit the JMS hosts list.

One of the hosts in the JMS hosts list is designated the default JMS host. Enterprise Server starts the default JMS host when the Message Queue broker mode is configured as type LOCAL.

The following topics are addressed here:

▼ To Create a JMS Host

A default JMS host, default_JMS_host, is provided by Enterprise Server. The default JMS host is used by Enterprise Server to perform all Message Queue broker administrative operations, such as creating and deleting JMS destinations.

Creating a new JMS host is not often necessary and is a task for advanced users. Use the create-jms-host subcommand in remote mode to create an additional JMS host.

Because a JMS is actually a Message Queue object rather than a server object, you use Message Queue broker commands to update properties. For information on Message Queue properties, see *Sun GlassFish Message Queue 4.4 Administration Guide*.

1   Ensure that the server is running.

Remote subcommands require a running server.

2   Create the JMS host by using the create-jms-host(1) **subcommand.**

Information about the properties for this the subcommand is included in this help page.

Example 19–10   Creating a JMS Host

This example creates a JMS host named MyNewHost.

```
asadmin> create-jms-host --mqhost pigeon --mqport 7677 MyNewHost
Command create-jms-host executed successfully.
```

See Also   You can also view the full syntax and options of the subcommand by typing asadmin help create-jms-host at the command line.

▼ To List JMS Hosts

Use the list-jms-hosts subcommand in remote mode to list the existing JMS hosts.

1   Ensure that the server is running.

Remote subcommands require a running server.

2   List the JMS hosts by using the list-jms-hosts(1) **subcommand.**

Example 19–11 Listing JMS Hosts

The following subcommand lists the existing JMS hosts.

```
asadmin> list-jms-hosts
default_JMS_host
MyNewHost
Command list-jmsdest executed successfully
```

▼ To Update a JMS Host

1 **List the JMS hosts by using the** list-jms-hosts**(1) subcommand.**

2 **Use the** set**(1) subcommand to modify a JMS host.**

Example 19–12 Updating a JMS Host

This example changes the value of the host attribute of the default JMS host. By default this value is localhost.

```
asadmin> set server-config.jms-service.jms-host.default_JMS_host.host=
"archie.india.sun.com"
```

See Also You can also view the full syntax and options of the subcommand by typing asadmin help set at the command line.

▼ To Delete a JMS Host

Use the delete-jms-host subcommand in remote mode to delete a JMS host from the JMS service. If you delete the only JMS host, you will not be able to start the Message Queue broker until you create a new JMS host.

1 **Ensure that the server is running.**

Remote subcommands require a running server.

2 **List the JMS hosts by using the** list-jms-hosts**(1) subcommand.**

3 **Delete a JMS host by using the** delete-jms-host**(1) subcommand.**

Example 19–13 Deleting a JMS Host

This example deletes a JMS host named MyNewHost.

```
asadmin> delete-jms-host MyNewHost
Command delete-jms-host executed successfully.
```

See Also   You can also view the full syntax and options of the subcommand by typing asadmin help
delete-jms-host at the command line.

Administering Connection Addressing

Certain JMS resources use the JMS host list (AddressList) configuration, which is populated
with the hosts and ports of the JMS hosts defined in Enterprise Server. The JMS host list is
updated whenever a JMS host configuration changes. The JMS host list is inherited by any JMS
resource when it is created, and by any message-driven bean when it is deployed.

In the Message Queue software, the AddressList property is called imqAddressList.

The following topics are addressed here:

-
-

Setting JMS Connection Pooling

Enterprise Server pools JMS connections automatically. When a JMS connection pool is
created, there is one ManagedConnectionFactory instance associated with it. If you configure
the AddressList property as a ManagedConnectionFactory property, the AddressList
configuration in the ManagedConnectionFactory value takes precedence over the value defined
in Enterprise Server.

Use the create-connector-connection-pool subcommand to manage an existing pool. For
instructions, see "Administering Connector Connection Pools" on page 255.

By default, the addresslist-behavior JMS service attribute is set to random. This means that
each physical connection (ManagedConnection) created from the ManagedConnectionFactory
selects its primary broker in a random way from the AddressList property.

To specify whether Enterprise Server tries to reconnect to the primary broker if the connection
is lost, set the reconnect-enabled attribute in the JMS service by using the set(1)
subcommand. To specify the number of retries and the time between retries, set the
reconnect-attempts and reconnect-interval-in-seconds attributes, respectively.

If reconnection is enabled and the primary broker fails, Enterprise Server tries to reconnect to
another broker in the JMS host list (AddressList). The logic for scanning is decided by two
JMS service attributes, addresslist-behavior and addresslist-iterations. You can
override these settings by using JMS connection factory settings. The Sun GlassFish Message
Queue software transparently transfers the load to another broker when the failover occurs.
JMS semantics are maintained during failover.

Accessing Remote Servers

Changing the provider and host to a remote system causes all JMS applications to run on the remote server. To use both the local server and one or more remote servers, create a connection factory resource with the AddressList property. This creates connections that access remote servers.

Configuring Resource Adapters for JMS

Enterprise Server implements JMS by using a system resource adapter named jmsra. When you create JMS resources, Enterprise Server automatically creates connector resources. The resource adapter can be configured to indicate whether the JMS provider supports XA or not. It is possible to indicate what mode of integration is possible with the JMS provider.

Two modes of integration are supported by the resource adapter. The first one uses JNDI as the means of integration. In this situation, administered objects are set up in the JMS provider's JNDI tree and will be looked up for use by the generic resource adapter. If that mode is not suitable for integration, it is also possible to use the Java reflection of JMS administered object javabean classes as the mode of integration.

Generic resource adapter 1.6 for JMS is a Java EE connector 2.0 resource adapter that can wrap the JMS client library of external JMS providers such as IBM WebSphere MQ, Tibco EMS, and Sonic MQ among others. This integrates any JMS provider with a Java EE 6 application server, such as the Sun GlassFish Enterprise Server. The adapter is a .rar archive that can be deployed and configured using Java EE 6 application server administration tools.

▼ To Configure the Generic Resource Adapter

Before deploying the generic resource adapter, JMS client libraries must be made available to Enterprise Server. For some JMS providers, client libraries might also include native libraries. In such cases, these native libraries must be made available to any Enterprise Server JVMs.

1 **Deploy the generic resource adapter the same way you would deploy a connector module.**

2 **Create a connector connection pool.**
See "To Create a Connector Connection Pool" on page 255.

3 **Create a connector resource.**
See "To Create a Connector Resource" on page 258.

4 **Create an administered object resource.**
See "To Create an Administered Object" on page 270.

5   Make the following changes to the security Enterprise Server policy files:

- Modify the `sjsas_home/domains/domain1/config/server.policy` file to add the following:

    ```
    java.util.logging.LoggingPermission "control"
    ```

- Modify the `sjsas_home/lib/appclient/client.policy` file to add permission:

    ```
    javax.security.auth.PrivateCredentialPermission
    "javax.resource.spi.security.PasswordCredential ^ \"^\"","read":
    ```

Troubleshooting JMS

When you start Enterprise Server, the JMS service is available but is not loaded until it is needed (for example, when you create a JMS resource). Use the `jms-ping(1)`subcommand to check if the JMS service is running or, if it is not yet running, to start it. If the `jms-ping` subcommand is unable to contact a built-in JMS service, an error message is displayed.

If you encounter problems, consider the following:

- View the Enterprise Server log file, typically located at *domain-dir*/`logs/server.log`.

    If a the log file indicates that a Message Queue broker did not respond to a message, stop the broker and then restart it.

- View the broker log, typically available at
    as-install/`domains/domain1/imq/instances/imqbroker/log/log.txt`.

- For JMS REMOTE mode, be sure to start Message Queue brokers first, then Enterprise Server.

- If all Message Queue brokers are down, it takes 30 minutes for Enterprise Server to go down or up when you are using the default values in JMS. You can change the default values for this timeout. For example:

    ```
    asadmin set domain1.jms-service.reconnect-interval-in-seconds=5
    ```

20

Administering the Java Naming and Directory Interface (JNDI) Service

The Java Naming and Directory Interface (JNDI) API is used for accessing different kinds of naming and directory services. Java EE components locate objects by invoking the JNDI lookup method.

The following topics are addressed here:

- "About JNDI" on page 313
- "Administering JNDI Resources" on page 315

Instructions for accomplishing the tasks in this chapter by using the Administration Console are contained in the Administration Console online help.

About JNDI

By making calls to the JNDI API, applications locate resources and other program objects. A *resource* is a program object that provides connections to systems, such as database servers and messaging systems. A JDBC resource is sometimes referred to as a data source. Each resource object is identified by a unique, people-friendly name, called the *JNDI name*. A resource object and its JNDI name are bound together by the naming and directory service, which is included with the Enterprise Server.

When a new name-object binding is entered into the JNDI, a new resource is created.

The following topics are addressed here:

- "J2EE Naming Environment" on page 314
- "How the Naming Environment and the Container Work Together" on page 314
- "Naming References and Binding Information" on page 315

J2EE Naming Environment

JNDI names are bound to their objects by the naming and directory service that is provided by a J2EE server. Because J2EE components access this service through the JNDI API, the object usually uses its JNDI name. For example, the JNDI name of the PointBase database is `jdbc/Pointbase`. At startup, the Enterprise Server reads information from the configuration file and automatically adds JNDI database names to the name space, one of which is `jdbc/Pointbase`.

Java EE application clients, enterprise beans, and web components must have access to a JNDI naming environment.

The application component's naming environment is the mechanism that allows customization of the application component's business logic during deployment or assembly. This environment allows you to customize the application component without needing to access or change the source code off the component. A Java EE container implements the provides the environment to the application component instance as a *JNDI naming context*.

How the Naming Environment and the Container Work Together

The application component's environment is used as follows:

- The application component's business methods access the environment using the JNDI interfaces. In the deployment descriptor, the application component provider declares all the environment entries that the application component expects to be provided in its environment at runtime.
- The container provides an implementation of the JNDI naming context that stores the application component environment. The container also provides the tools that allow the deployer to create and manage the environment of each application component.
- A deployer uses the tools provided by the container to initialize the environment entries that are declared in the application component's deployment descriptor. The deployer sets and modifies the values of the environment entries.
- The container makes the JNDI context available to the application component instances at runtime. These instances use the JNDI interfaces to obtain the values of the environment entries.

Each application component defines its own set of environment entries. All instances of an application component within the same container share the same environment entries. Application component instances are not allowed to modify the environment at runtime.

Naming References and Binding Information

A *resource reference* is an element in a deployment descriptor that identifies the component's coded name for the resource. For example, jdbc/SavingsAccountDB. More specifically, the coded name references a connection factory for the resource.

The JNDI name of a resource and the resource reference name are not the same. This approach to naming requires that you map the two names before deployment, but it also decouples components from resources. Because of this decoupling, if at a later time the component needs to access a different resource, the name does not need to change. This flexibility makes it easier for you to assemble J2EE applications from preexisting components.

The following table lists JNDI lookups and their associated resource references for the J2EE resources used by the Enterprise Server.

TABLE 20–1   JNDI Lookup Names and Their Associated References

JNDI Lookup Name	Associated Resource Reference
java:comp/env	Application environment entries
java:comp/env/jdbc	JDBC DataSource resource manager connection factories
java:comp/env/ejb	EJB References
java:comp/UserTransaction	UserTransaction references
java:comp/env/mail	JavaMail Session Connection Factories
java:comp/env/url	URL Connection Factories
java:comp/env/jms	JMS Connection Factories and Destinations
java:comp/ORB	ORB instance shared across application components

Administering JNDI Resources

Within Enterprise Server, you can configure your environment for custom and external JNDI resources. A custom resource accesses a local JNDI repository; an external resource accesses an external JNDI repository. Both types of resources need user-specified factory class elements, JNDI name attributes, and so on.

- "Administering Custom JNDI Resources" on page 316
- "Administering External JNDI Resources" on page 318

Administering Custom JNDI Resources

A custom resource specifies a custom server-wide resource object factory that implements the `javax.naming.spi.ObjectFactory` interface. The following topics are addressed here:

▼ To Create a Custom JNDI Resource

Use the `create-custom-resource` subcommand in remote mode to create a custom resource.

1   Ensure that the server is running.

Remote subcommands require a running server.

2   Create a custom resource by using the `create-custom-resource`**(1) subcommand.**

Information on properties for the subcommand is contained in this help page.

3   Restart Enterprise Server.

See "To Restart a Domain" on page 90.

Example 20–1   Creating a Custom Resource

This example creates a custom resource named `sample-custom-resource`.

```
asadmin> create-custom-resource --restype topic --factoryclass com.imq.topic
sample_custom_resource
Command create-custom-resource executed successfully.
```

See Also   You can also view the full syntax and options of the subcommand by typing `asadmin help create-custom-resource` at the command line.

▼ To List Custom JNDI Resources

Use the `list-custom-resources` subcommand in remote mode to list the existing custom resources.

1   Ensure that the server is running.

Remote subcommands require a running server.

2   List the custom resources by using the `list-custom-resources`**(1) subcommand.**

Example 20–2   Listing Custom Resources

This example lists the existing custom resources.

```
asadmin> list-custom-resources
sample_custom_resource01
sample_custom_resource02
Command list-custom-resources executed successfully
```

See Also   You can also view the full syntax and options of the subcommand by typing asadmin help list-custom-resources at the command line.

▼ To Update a Custom JNDI Resource

1 **List the custom resources by using the** list-custom-resources**(1) subcommand.**

2 **Use the** set**(1) subcommand to modify a custom JNDI resource.**

Example 20–3   Updating a Custom JNDI Resource

This example modifies a custom resource.

```
asadmin> set server.resources.custom-resource.custom
/my-custom-resource.property.value=2010server.resources.custom-resource.custom
/my-custom-resource.property.value=2010
```

▼ To Delete a Custom JNDI Resource

Use the delete-custom-resource subcommand in remote mode to delete a custom resource.

1 **Ensure that the server is running.**

Remote subcommands require a running server.

2 **List the custom resources by using the** list-custom-resources**(1) subcommand.**

3 **Delete a custom resource by using the** delete-custom-resource**(1) subcommand.**

Example 20–4   Deleting a Custom Resource

This example deletes a custom resource named sample-custom-resource.

```
asadmin> delete-custom-resource sample_custom_resource
Command delete-custom-resource executed successfully.
```

Chapter 20 • Administering the Java Naming and Directory Interface (JNDI) Service 317

See Also   You can also view the full syntax and options of the subcommand by typing asadmin help
delete-custom-resource at the command line.

Administering External JNDI Resources

Applications running on Enterprise Server often require access to resources stored in an
external JNDI repository. For example, generic Java objects might be stored in an LDAP server
according to the Java schema. External JNDI resource elements let you configure such external
resource repositories.

The following topics are addressed here:

▼ To Register an External JNDI Resource

Use the create-jndi-resource subcommand in remote mode to register an external JNDI
resource.

Before You Begin   The external JNDI factory must implement the javax.naming.spi.InitialContextFactory
interface.

1   Ensure that the server is running.

Remote subcommands require a running server.

2   Register an external JNDI resource by using the create-jndi-resource(1) **subcommand.**

Information on properties for the subcommand is contained in this help page.

3   Restart Enterprise Server.

See "To Restart a Domain" on page 90.

Example 20–5   Registering an External JNDI Resource

In This example sample_jndi_resource is registered.

```
asadmin> create-jndi-resource --jndilookupname sample_jndi
--restype queue --factoryclass sampleClass --description "this is a sample jndi
resource" sample_jndi_resource
Command create-jndi-resource executed successfully
```

See Also You can also view the full syntax and options of the subcommand by typing asadmin help create-jndi-resource at the command line.

▼ To List External JNDI Resources

Use the list-jndi-resources subcommand in remote mode to list all existing JNDI resources.

1 Ensure that the server is running.

Remote subcommands require a running server.

2 List the existing JNDI resources by using the list-jndi-resources**(1) subcommand.**

Example 20–6 Listing JNDI Resources

This example lists the JNDI resources.

```
asadmin> list-jndi-resources
jndi_resource1
jndi_resource2
jndi_resource3
Command list-jndi-resources executed successfully
```

See Also You can also view the full syntax and options of the subcommand by typing asadmin help list-jndi-resources at the command line.

▼ To List External JNDI Entries

Use the list-jndi-entries subcommand in remote mode to browse and list the entries in the JNDI tree. You can either list all entries, or you can specify the JNDI context or subcontext to list specific entries.

1 Ensure that the server is running.

Remote subcommands require a running server.

2 List the JNDI entries for a configuration by using the list-jndi-entries**(1) subcommand.**

Example 20–7 Listing JNDI Entries

This example lists all the JNDI entries for the naming service.

```
asadmin> list-jndi-entries
jndi_entry03
jndi_entry72
jndi_entry76
Command list-jndi-resources executed successfully
```

See Also   You can also view the full syntax and options of the subcommand by typing asadmin help
list-jndi-entries at the command line.

▼ To Update an External JNDI Resource

1   **List the existing JNDI resources by using the** list-jndi-resources**(1) subcommand.**

2   **Use the** set**(1) subcommand to modify an external JNDI resource.**

Example 20–8   Updating an External JNDI Resource

This example modifies an external resource.

```
asadmin> set server.resources.external-jndi-resource.my-jndi-resource.
jndi-lookup-name=bar server.resources.external-jndi-resource.my-jndi-resource.jndi-lookup-name=bar
```

▼ To Delete an External JNDI Resource

Use the delete-jndi-resource subcommand in remote mode to remove a JNDI resource.

1   **Ensure that the server is running.**

Remote subcommands require a running server.

2   **Remove an external JNDI entry by using the** delete-jndi-resource**(1) subcommand.**

Example 20–9   Deleting an External JNDI Resource

This example deletes an external JNDI resource:

```
asadmin> delete-jndi-resource jndi_resource2
Command delete-jndi-resource executed successfully.
```

See Also   You can also view the full syntax and options of the subcommand by typing asadmin help
delete-jndi-resource at the command line.

Example of Using an External JNDI Resource

```
<resources>
 <!-- external-jndi-resource element specifies how to access J2EE resources
 -- stored in an external JNDI repository. This example
 -- illustrates how to access a java object stored in LDAP.
 -- factory-class element specifies the JNDI InitialContext factory that
 -- needs to be used to access the resource factory. property element
 -- corresponds to the environment applicable to the external JNDI context
```

```
      -- and jndi-lookup-name refers to the JNDI name to lookup to fetch the
      -- designated (in this case the java) object.
      -->
       <external-jndi-resource jndi-name="test/myBean"
           jndi-lookup-name="cn=myBean"
           res-type="test.myBean"
           factory-class="com.sun.jndi.ldap.LdapCtxFactory">
         <property name="PROVIDER-URL" value="ldap://ldapserver:389/o=myObjects" />
         <property name="SECURITY_AUTHENTICATION" value="simple" />
         <property name="SECURITY_PRINCIPAL", value="cn=joeSmith, o=Engineering" />
         <property name="SECURITY_CREDENTIALS" value="changeit" />
       </external-jndi-resource>
    </resources>
```

Administering Transactions

This chapter discusses how to manage the transaction service for the Sun GlassFish™ Enterprise Server environment by using the `asadmin` command-line utility. Instructions for manually recovering transactions are also included.

The following topics are addressed here:

- "About Transactions" on page 323
- "Managing the Transaction Service" on page 324
- "Recovering Transactions" on page 326

Instructions for accomplishing the tasks in this chapter by using the Administration Console are contained in the Administration Console online help. For additional information on configuring the transaction service, transaction logging, and distributed transaction recovery, see Chapter 15, "Using the Transaction Service," in *Sun GlassFish Enterprise Server v3 Application Development Guide*.

About Transactions

A *transaction* is a series of discreet actions in an application that must all complete successfully. By enclosing one or more actions in an indivisible unit of work, a transaction ensures data integrity and consistency. If all actions do not complete, the changes are rolled back.

For example, to transfer funds from a checking account to a savings account, the following steps typically occur:

1. Check to see if the checking account has enough money to cover the transfer.
2. Debit the amount from the checking account.
3. Credit the amount to the savings account.
4. Record the transfer to the checking account log.
5. Record the transfer to the savings account log.

These steps together are considered a single transaction.

If all the steps complete successfully, the transaction is *committed*. If any step fails, all changes from the preceding steps are rolled back, and the checking account and savings account are returned to the states they were in before the transaction started. This type of event is called a *rollback*. A normal transaction ends in either a committed state or a rolled back state.

The following elements contribute to reliable transaction processing by implementing various APIs and functionalities:

- **Transaction Manager.** Provides the services and management functions required to support transaction demarcation, transactional resource management, synchronization, and transaction context propagation.

- **Enterprise Server.** Provides the infrastructure required to support the application runtime environment that includes transaction state management.

- **Resource Manager.** Through a resource adapter, the resource manager provides the application access to resources. The resource manager participates in distributed transactions by implementing a transaction resource interface used by the transaction manager to communicate transaction association, transaction completion, and recovery work. An example of such a resource manager is a relational database server.

- **Resource Adapter.** A system-level software library is used by Enterprise Server or a client to connect to a resource manager. A resource adapter is typically specific to a resource manager. The resource adapter is available as a library and is used within the address space of the client using it. An example of such a resource adapter is a Java™ Database Connectivity (JDBC) driver. For information on supported JDBC drivers, see "Configuration Specifics for JDBC Drivers" on page 243.

- **Transactional User Application.** In the Enterprise Server environment, the transactional user application uses Java Naming and Directory Interface (JNDI) to look up transactional data sources and, optionally, the user transaction). The application might use declarative transaction attribute settings for enterprise beans, or explicit programmatic transaction demarcation.

Managing the Transaction Service

You can roll back a single transaction by using the asadmin subcommands described in this section. To do so, the transaction service must be stopped (and later restarted), allowing you to see the active transactions and correctly identify the one that needs to be rolled back.

For instructions on configuring the transaction service and setting up automatic recovery, see Chapter 15, "Using the Transaction Service," in *Sun GlassFish Enterprise Server v3 Application Development Guide*.

The following topics are addressed here:

▼ To Stop the Transaction Service

Use the freeze-transaction-service subcommand in remote mode to stop the transaction service. When the transaction service is stopped, all in-flight transactions are immediately suspended. You must stop the transaction service before rolling back any in-flight transactions.

Running this subcommand on a stopped transaction subsystem has no effect. The transaction service remains suspended until you restart it by using the unfreeze-transaction-service subcommand.

1 Ensure that the server is running.

Remote subcommands require a running server.

2 Stop the transaction service by using the freeze-transaction-service**(1) subcommand.**

Example 21–1 Stopping the Transaction Service

This example stops the transaction service.

```
asadmin> freeze-transaction-service
Command freeze-transaction-service executed successfully
```

See Also You can also view the full syntax and options of the subcommand by typing asadmin help freeze-transaction-service at the command line.

▼ To Roll Back a Transaction

In some situations, you might want to roll back a particular transaction. Before you can roll back a transaction, you must first stop the transaction service so that transaction operations are suspended. Use the rollback-transaction subcommand in remote mode to roll back a specific transaction.

Before You Begin Stop the transaction service before rolling back an in-flight transaction.

1 Ensure that the server is running.

Remote subcommands require a running server.

2 Identify the ID of the transaction you want to roll back.

To see a list of IDs of active transactions, use the get subcommand to get the monitoring data for the activeids statistic. See "Transaction Service Statistics" on page 161.

3 Roll back the transaction by using the `rollback-transaction`**(1) subcommand.**

Example 21–2 Rolling Back a Transaction

This example rolls back the transaction with transaction ID `0000000000000001_00`.

```
asadmin> rollback-transaction 0000000000000001_00
Command rollback-transaction executed successfully
```

See Also You can also view the full syntax and options of the subcommand by typing `asadmin help` `rollback-transaction` at the command line.

▼ To Restart the Transaction Service

Use the `unfreeze-transaction-service` subcommand in remote mote to resume all the suspended in-flight transactions. Run this subcommand to restart the transaction service after it has been frozen.

1 Ensure that the server is running.

Remote subcommands require a running server.

2 Restart the suspended transaction service by using the `unfreeze-transaction-service`**(1) subcommand.**

Example 21–3 Restarting the Transaction Service

This example restarts the transaction service after it has been frozen.

```
asadmin> unfreeze-transaction-service
Command unfreeze-transaction-service executed successfully
```

See Also You can also view the full syntax and options of the subcommand by typing `asadmin help` `unfreeze-transaction-service` at the command line.

Recovering Transactions

There are some situations where the commit or rollback operations might be interrupted, typically because the server crashed or a resource manager crashed. Crash situations can leave some transactions stranded between steps. Enterprise Server is designed to recover from these failures and complete the transactions upon server startup. If the failed transaction spans multiple servers, the server that started the transaction can contact the other servers to get the

outcome of the transaction. If the other servers are unreachable, the transaction uses heuristic decision information to determine the outcome. The transactions are resolved upon server startup.

In a failure situation that is not a crash, you can perform manual recovery as described in the following procedure

▼ To Manually Recover Transactions

Use the recover-transactions subcommand in remote mode to manually recover transactions that were pending when a resource on the server failed. Manual transaction recovery cannot be used to recover transactions after a server crash. Manual operations are intended for the cases when a resource fails unexpectedly, but the server is still running. If the server crashes, only the full startup recovery process can recover transactions that are in doubt.

1 Ensure that the server is running.

Remote subcommands require a running server.

2 Manually recover transactions by using the recover-transactions**(1) subcommand.**

Example 21–4 Manually Recovering Transactions

This example performs manual recovery of transactions on sampleserver.

```
asadmin recover-transactions sampleserver
Transaction recovered.
```

See Also You can also view the full syntax and options of the subcommand by typing asadmin help recover-transactions at the command line.

PART IV

Appendixes

A

Subcommands for the `asadmin` Utility

This appendix lists the `asadmin` subcommands that are included with this release of the Sun GlassFish™ Enterprise Server v3 software.

For information and instructions on using the `asadmin` application deployment subcommands, see *Sun GlassFish Enterprise Server v3 Application Deployment Guide*.

Online help for the `asadmin` subcommands can be invoked on the command line, for example, `asadmin create-domain --help`. The *Sun GlassFish Enterprise Server v3 Reference Manual* also provides a collection of these help pages.

Note – The common options used with remote subcommands are described in the `asadmin(1M)` help page.

General Administration Subcommands

add-resources(1)
: Creates the resources named in the specified XML file. Supported in remote mode only. For procedural information in this guide, see "To Add Resources From an XML File" on page 56.

asadmin(1M)
: Describes how the asadmin utility works.

create-service(1)
: Configures the starting of a domain administration server (DAS) on an unattended boot. On Solaris 10, this subcommand uses the Service Management Facility (SMF). For procedural information in this guide, see "To Automatically Restart a Domain on Solaris 10" on page 91.

create-system-properties(1)
: Creates or updates system properties. Supported in remote mode only. For procedural information in this guide, see "To Create System Properties" on page 54.

delete-system-property(1)
: Deletes system properties of a domain or configuration, or server instance. Supported in remote mode only. For procedural information in this guide, see "To Delete a System Property" on page 55.

get(1)
: Gets an attribute of an element in the domain.xml file. With the -m option, gets the names and values of the monitorable or configurable attributes. For procedural information in this guide, see "Guidelines for Using the list and get Subcommands for Monitoring" on page 136.

list(1)
: Lists the configurable element. On Solaris, quotes are needed when running subcommands with * as the option value or operand. For procedural information in this guide, see "Guidelines for Using the list and get Subcommands for Monitoring" on page 136.

list-commands(1)
: Lists all the asadmin subcommands, local subcommands first, then remote subcommands. You can specify that only remote subcommands or only local subcommands be displayed. Supported in remote mode only. For procedural information in this guide, see "To List Subcommands" on page 60.

list-containers(1)
: Lists application containers and the status of each container. Supported in remote mode only. For procedural information in this guide, see "To List Containers" on page 58.

list-modules(1)   Lists modules that are accessible to the Enterprise Server
                                   subsystem. The status of each module is included.
                                   Supported in remote mode only. For procedural
                                   information in this guide, see "To List Modules" on
                                   page 59.

list-system-properties(1)   Lists the system properties of a domain or configuration.
                                   Supported in remote mode only. For procedural
                                   information in this guide, see "To List System Properties"
                                   on page 55.

list-timers(1)   List the timers owned by a specific server instance.
                                   Supported in remote mode only. For procedural
                                   information in this guide, see "To List Timers" on page 61.

multimode(1)   Provides an asadmin> prompt for running multiple
                                   subcommands while preserving options and environment
                                   settings. Supported in local mode only. For procedural
                                   information, see "Using the asadmin Utility" on page 47.

set(1)   Sets the values of one or more configurable attributes. For
                                   procedural information in this guide, see "Configuring
                                   Monitoring" on page 131.

show-component-status(1)   Lists the status of existing components. Supported in
                                   remote mode only. For procedural information in this
                                   guide, see "To Show Component Status" on page 61.

start-database(1)   Starts the Java DB server. Use this subcommand only for
                                   working with applications deployed to the Enterprise
                                   Server. For procedural information in this guide, see "To
                                   Start the Database" on page 233.

stop-database(1)   Stops a process of the Java DB database server. For
                                   procedural information in this guide, see "To Stop the
                                   Database" on page 234.

version(1)   Displays the version information for the option specified
                                   in archive or folder format. Supported in remote mode
                                   only. For procedural information in this guide, see "To
                                   Display the Enterprise Server Version" on page 57.

Connectivity Subcommands

`create-admin-object(1)`	Creates an administered object. For procedural information in this guide, see "To Create an Administered Object" on page 270.
`create-connector-connection-pool(1)`	Adds a new connector connection pool with the specified connection pool name. For procedural information in this guide, see "To Create a Connector Connection Pool" on page 255.
`create-connector-resource(1)`	Creates a connector resource. For procedural information in this guide, see "To Create a Connector Resource" on page 258.
`create-connector-security-map(1)`	Creates a connector security map for the specified connector connection pool. For procedural information, see "To Create a Connector Security Map" on page 264.
`create-connector-work-security-map(1)`	Creates a connector work security map for the specified resource adapter. Supported in remote mode only. For procedural information in this guide, see "To Create a Connector Work Security Map" on page 267.
`create-jdbc-resource(1)`	Creates a new JDBC resource. Supported in remote mode only. For procedural information in this guide, see "To Create a JDBC Resource" on page 240.
`create-jdbc-connection-pool(1)`	Registers a new JDBC connection pool with the specified JDBC connection pool name. Supported in remote mode only. For procedural information in this guide, see "To Create a JDBC Connection Pool" on page 236.
`create-resource-adapter-config(1)`	Creates configuration information for the connector module. Supported in remote mode only. For procedural information in this guide, see "To Create Configuration Information for a Resource Adapter" on page 261.
`delete-admin-object(1)`	Deletes an administered object. For procedural information in this guide, see "To Delete an Administered Object" on page 272.

delete-connector-connection-pool(1)	Removes the connector connection pool specified using the connector_connection_pool_name operand. For procedural information in this guide, see "To Delete a Connector Connection Pool" on page 258.
delete-connector-resource(1)	Deletes connector resource. For procedural information in this guide, see "To Delete a Connector Resource" on page 260.
delete-connector-security-map(1)	Deletes a specified connector security map. Supported in remote mode only. For procedural information in this guide, see "To Delete a Connector Security Map" on page 266.
delete-connector-work-security-map(1)	Deletes a specified connector work security map. Supported in remote mode only. For procedural information in this guide, see "To Delete a Connector Work Security Map" on page 269.
delete-jdbc-connection-pool(1)	Deletes the specified JDBC connection pool. Supported in remote mode only. For procedural information in this guide, see "To Delete a JDBC Connection Pool" on page 239.
delete-jdbc-resource(1)	Deletes a JDBC resource. The specified JNDI name identifies the resource to be deleted. Supported in remote mode only. For procedural information in this guide, see "To Delete a JDBC Resource" on page 242.
delete-resource-adapter-config(1)	Deletes configuration information for the connector module. Supported in remote mode only. For procedural information in this guide, see "To Delete a Resource Adapter Configuration" on page 263.
flush-connection-pool(1)	Reinitializes all connections established in the specified connection. For procedural information in this guide, see "To Reset (Flush) a Connection Pool" on page 238.

list-admin-objects(1)   Lists administered objects. For procedural information in this guide, see "To List Administered Objects" on page 271.

list-connector-connection-pools(1)   Lists the connector connection pools that have been created. For procedural information in this guide, see "To List Connector Connection Pools" on page 256.

list-connector-resources(1)   Creates connector resources. For procedural information in this guide, see "To List Connector Resources" on page 259.

llist-connector-security-maps(1)   Lists the connector security maps belonging to a specified connector connection pool. For procedural information in this guide, see "To List Connector Security Maps" on page 264.

list-connector-work-security-maps(1)   Lists the existing connector work security maps for a resource adapter. Supported in remote mode only. For procedural information in this guide, see "To List Connector Work Security Maps" on page 268.

list-jdbc-connection-pools(1)   Lists the existing JDBC connection pools. Supported in remote mode only. For procedural information in this guide, see "To List JDBC Connection Pools" on page 237.

list-jdbc-resources(1)   Lists the existing JDBC resources. Supported in remote mode only. For procedural information in this guide, see "To List JDBC Resources" on page 241.

list-resource-adapter-configs(1)   Lists configuration information for the connector modules. Supported in remote mode only. For procedural information in this guide, see "To List Resource Adapter Configurations" on page 262.

ping-connection-pool(1)   Tests if a JDBC connection pool is usable. Supported in remote mode only. For procedural information in this guide, see "To Contact (Ping) a Connection Pool" on page 238.

update-connector-security-map(1)   Modifies a security map for the specified connector connection pool. For procedural

information in this guide, see "To Update a Connector Security Map" on page 265.

update-connector-work-security-map(1)   Modifies a work security map that belongs to a specific resource adapter (connector module). For procedure information in this guide, see "To Update a Connector Work Security Map" on page 268.

Domain Subcommands

create-domain(1)   Creates the configuration of a domain. A domain can exist independent of other domains. Any user who has access to the asadmin utility on a given host can create a domain and store its configuration in a location of choice. For procedural information in this guide, see "To Create a Domain" on page 84.

delete-domain(1)   Deletes the specified domain. The domain must be stopped before it can be deleted. For procedural information in this guide, see "To Delete a Domain" on page 87.

list-domains(1)   Lists the existing domains and their statuses. If the domain directory is not specified, the domains in the default *as-install*/domains directory is displayed. For procedural information in this guide, see "To List Domains" on page 85.

login(1)   Allows you to log in to a domain. For procedural information in this guide, see "To Log In to a Domain" on page 86.

restart-domain(1)   Restarts the Domain Administration Server (DAS) of the specified domain. Supported in remote mode only. For procedural information in this guide, see "To Restart a Domain" on page 90.

start-domain(1)   Starts a domain. If the domain directory is not specified, the default domain1 in the default *as-install*/domains directory is started. If there are two or more domains, the *domain_name* operand must be specified. For procedural information in this guide, see "To Start a Domain" on page 88.

stop-domain(1)   Stops the domain administration server (DAS) of the specified domain. Supported in remote mode only. For procedural information in this guide, see "To Stop a Domain" on page 89.

uptime(1)   Displays the length of time that the domain administration server (DAS) has been running since the last restart. Supported in remote mode only. For procedural information in this guide, see "To Display

Domain Uptime" on page 93.

Internet Connectivity Subcommands

create-http(1)   Creates a set of HTTP parameters for a protocol, which in
                                  turn configures one or more network listeners. Supported in
                                  remote mode only. For procedural information in this
                                  guide, see "To Create an HTTP Configuration" on page 278.

create-http-listener(1)   Creates a new HTTP listener socket. Supported in remote
                                  mode only. For procedural information in this guide, see
                                  "To Create an Internet Connection" on page 276.

create-network-listener(1)   Creates a new HTTP listener socket. Supported in remote
                                  mode only. For procedural information in this guide, see
                                  "To Create an Internet Connection" on page 276.

create-protocol(1)   Creates a protocol for a listener. Supported in remote mode
                                  only. For procedural information in this guide, see "To
                                  Create a Protocol" on page 277.

create-transport(1)   Creates a transport for a listener. Supported in remote mode
                                  only. For procedural information in this guide, see "To
                                  Create a Transport" on page 280.

create-virtual-server(1)   Creates the specified virtual server element. Supported in
                                  remote mode only. For procedural information in this
                                  guide, see "To Create a Virtual Server" on page 286.

create-ssl(1)   Creates and configures the SSL element in the selected
                                  HTTP listener to enable secure communication on that
                                  listener/service. Supported in remote mode only. For
                                  procedural information in this guide, see "To Configure an
                                  HTTP Listener for SSL" on page 284.

delete-http(1)   Deletes an existing HTTP configuration. Supported in
                                  remote mode only. For procedural information in this
                                  guide, see "To Delete an HTTP Configuration" on page 279.

delete-http-listener(1)   Deletes the specified HTTP listener. Supported in remote
                                  mode only. For procedural information in this guide, see
                                  "To Delete an HTTP Network Listener" on page 283.

delete-network-listener(1)   Deletes the specified HTTP listener. Supported in remote
                                  mode only. For procedural information in this guide, see
                                  "To Delete an HTTP Network Listener" on page 283.

delete-protocol(1) Deletes and existing HTTP protocol. Supported in remote mode only. For procedural information in this guide, see "To Delete a Protocol" on page 278.

delete-ssl(1) Deletes the SSL element in the selected HTTP listener. Supported in remote mode only. For procedural information in this guide, see "To Delete SSL From an HTTP Listener" on page 284.

delete-transport(1) Deletes and existing HTTP transport. Supported in remote mode only. For procedural information in this guide, see "To Delete a Transport" on page 281.

delete-virtual-server(1) Deletes the specified virtual server element. Supported in remote mode only. For procedural information in this guide, see "To Delete a Virtual Server" on page 288.

list-http-listeners(1) Lists the existing HTTP listeners. Supported in remote mode only. For procedural information in this guide, see "To List HTTP Network Listeners" on page 282.

list-network-listeners(1) Lists the existing HTTP listeners. Supported in remote mode only. For procedural information in this guide, see "To List HTTP Network Listeners" on page 282.

list-protocols(1) Lists the existing HTTP protocols. Supported in remote mode only. For procedural information in this guide, see "To List Protocols" on page 277.

list-transports(1) Lists the existing HTTP transports. Supported in remote mode only. For procedural information in this guide, see "To List Transports" on page 280.

list-virtual-servers(1) Lists the existing virtual servers. Supported in remote mode only. For procedural information in this guide, see "To List Virtual Servers" on page 287.

JavaMail Subcommands

create-javamail-resource(1) Creates a JavaMail session resource. Supported in remote mode only. For procedural information in this guide, see "To Create a JavaMail Resource" on page 296.

delete-javamail-resource(1) Deletes a JavaMail session resource. Supported in remote mode only. For procedural information in this guide, see "To Delete a JavaMail Resource" on page 298.

list-javamail-resources(1)   Creates JavaMail session resources. Supported in remote
                               mode only. For procedural information in this guide, see
                               "To List JavaMail Resources" on page 297.

JMS Subcommands

create-jmsdest(1)   Creates a JMS physical destination. Along with the physical
                              destination, you use the create-jms-resource subcommand to
                              create a JMS destination resource that has a Name property that
                              specifies the physical destination. Supported in remote mode
                              only. For procedural information in this guide, see "To Create a
                              JMS Physical Destination" on page 301.

create-jms-host(1)   Creates a JMS host within the JMS service. Supported in remote
                              mode only. For procedural information in this guide, see "To
                              Create a JMS Host" on page 308.

create-jms-resource(1)   Creates a JMS connection factory resource or JMS destination
                              resource. Supported in remote mode only. Supported in remote
                              mode only. For procedural information in this guide, see "To
                              Create a Connection Factory or Destination Resource" on
                              page 305.

delete-jmsdest(1)   Removes the specified JMS destination. Supported in remote
                              mode only. For procedural information in this guide, see "To
                              Delete a JMS Physical Destination" on page 303.

delete-jms-host(1)   Deletes a JMS host within the JMS service. Supported in remote
                              mode only. For procedural information in this guide, see "To
                              Delete a JMS Host" on page 309.

delete-jms-resource(1)   Deletes a JMS connection factory resource or JMS destination
                              resource. Supported in remote mode only. For procedural
                              information in this guide, see "To Delete a Connection Factory
                              or Destination Resource" on page 307.

flush-jmsdest(1)   Purges the messages from a physical destination in the specified
                              JMS Service configuration of the specified target. Supported in
                              remote mode only. For procedural information in this guide, see
                              "To Purge Messages From a Physical Destination" on page 303.

jms-ping(1)   Checks if the JMS service (also known as the JMS provider) is up
                              and running. Supported in remote mode only. For procedural
                              information in this guide, see "Troubleshooting JMS" on
                              page 312.

list-jmsdest(1)	Lists the JMS physical destinations. Supported in remote mode only. For procedural information in this guide, see "To List JMS Physical Destinations" on page 302.
list-jms-hosts(1)	Lists the existing JMS hosts. Supported in remote mode only. For procedural information in this guide, see "To List JMS Hosts" on page 308.
list-jms-resources(1)	Lists the existing JMS connection factory or destination resources. Supported in remote mode only. For procedural information in this guide, see "To List JMS Resources" on page 306.

JNDI Subcommands

create-custom-resource(1)	Creates a custom JNDI resource. Supported in remote mode only. For procedural information in this guide, see "To Create a Custom JNDI Resource" on page 316.
create-jndi-resource(1)	Creates an external JNDI resource. Supported in remote mode only. For procedural information in this guide, see "To Register an External JNDI Resource" on page 318.
delete-custom-resource(1)	Deletes a custom JNDI resource. Supported in remote mode only. For procedural information in this guide, see "To Delete a Custom JNDI Resource" on page 317.
delete-jndi-resource(1)	Deletes an external JNDI resource. Supported in remote mode only. For procedural information in this guide, see "To Delete an External JNDI Resource" on page 320.
list-custom-resources(1)	Lists the existing custom JNDI resources. Supported in remote mode only. For procedural information in this guide, see "To List Custom JNDI Resources" on page 316.
list-jndi-entries(1)	Lists the entries in the JNDI tree. Supported in remote mode only. For procedural information in this guide, see "To List External JNDI Entries" on page 319,
list-jndi-resources(1)	Lists the existing external JNDI resources. Supported in remote mode only. For procedural information in this guide, see "To List External JNDI Resources" on page 319.

JVM Subcommands

create-jvm-options(1)	Creates a JVM option in the Java configuration or profiler elements of the domain.xml file. Supported in remote mode only. For procedural information in this guide, see "To Create JVM Options" on page 96.
create-profiler(1)	Creates a profiler element. Supported in remote mode only. For procedural information in this guide, see "To Create a Profiler" on page 99.
delete-jvm-options(1)	Deletes the specified JVM option from the Java configuration or profiler elements of the domain.xml file. Supported in remote mode only. For procedural information in this guide, see "To Delete JVM Options" on page 97.
delete-profiler(1)	Deletes the specified profiler element. Supported in remote mode only. For procedural information in this guide, see "To Delete a Profiler" on page 100.
generate-jvm-report(1)	Generates a report showing the threads, classes, and memory for the virtual machine that runs Enterprise Server. For procedural information in this guide, see "To Generate a JVM Report" on page 98.
list-jvm-options(1)	Lists the command-line options that are passed to the Java application launcher when Enterprise Server is started. Supported in remote mode only. For procedural information in this guide, see "To List JVM Options" on page 96.

Life Cycle Module Subcommands

create-lifecycle-module(1)	Creates a new life cycle module. Supported in remote mode only. For procedural information in this guide, see "To Create a Life Cycle Module" on page 168.
list-lifecycle-modules(1)	Lists life cycle modules. Supported in remote mode only. For procedural information in this guide, see "To List Life Cycle Modules" on page 169.
delete-lifecycle-module(1)	Deletes an existing life cycle module. Supported in remote mode only. For procedural information in this guide, see "To Delete a Life Cycle Module" on page 170.

Logging and Monitoring Subcommands

disable-monitoring(1)   Disables the monitoring service. Supported in remote mode only. For procedural information in this guide, see "To Disable Monitoring" on page 132.

enable-monitoring(1)   Enables the monitoring service. Supported in remote mode only. For procedural information in this guide, see "To Enable Monitoring" on page 131.

monitor(1)   Displays monitoring information for the common Enterprise Server resources. Supported in remote mode only. For procedural information in this guide, see "To View Common Monitoring Data" on page 133.

list-logger-levels(1)   Lists the existing loggers. Supported in remote mode only. For procedural information in this guide, see "To Set Module Logger Levels" on page 119.

rotate-log(1)   Rotates the server.log file and stores the old data in a time-stamped file. Supported in remote mode only. For procedural information in this guide, see "To Rotate a Log File Manually" on page 121.

set-log-level(1)   Sets the log level for a module. Supported in remote mode only. For procedural information in this guide, see "To Set Module Logger Levels" on page 119.

ORB Subcommands

create-iiop-listener(1)   Creates an IIOP listener. Supported in remote mode only. For procedural information in this guide, see "To Create an IIOP Listener" on page 292.

delete-iiop-listener(1)   Deletes an IIOP listener. Supported in remote mode only. For procedural information in this guide, see "To Delete an IIOP Listener" on page 294.

list-iiop-listeners(1)   Lists the existing IIOP listeners. Supported in remote mode only. For procedural information in this guide, see "To List IIOP Listeners" on page 293.

Security Subcommands

change-admin-password(1)
Modifies the administration password. You are prompted for the old and new admin password (with confirmation). For procedural information in this guide, see "To Change the Administration Password" on page 192.

change-master-password(1)
Changes the administration master password. This password is used to access the secure store where primary keys are stored. This subcommand will not work unless the server is stopped. For procedural information in this guide, see "To Change the Master Password" on page 190.

configure-ldap-for-admin(1)
Configures the authentication realm named admin-realm for the given LDAP. Supported in remote mode only.

create-audit-module(1)
Adds the named audit module for the plug-in that implements the audit capabilities. Supported in remote mode only. For procedural information in this guide, see "To Create an Audit Module" on page 197.

create-message-security-provider(1)
Creates a provider-config subelement for the given message layer (the message-security-config element of domain.xml, the file that specifies parameters and properties). Supported in remote mode only. For procedural information in this guide, see "To Create a Message Security Provider" on page 226.

create-password-alias(1)
Creates an alias for a password; aliases are not stored as clear text in the domain.xml file. Supported in remote mode only. For procedural information in this guide, see "To Create a Password Alias" on page 194.

delete-audit-module(1)
Removes the named audit module. Supported in remote mode only. For procedural information in this guide, see "To Delete an Audit Module" on page 198.

delete-message-security-provider(1)
Deletes a provider-config subelement for the given message layer. Supported in remote mode

only. For procedural information in this guide, see "To Delete a Message Security Provider" on page 227.

delete-password-alias(1)   Deletes an alias for a password. Supported in remote mode only. For procedural information in this guide, see "To Delete a Password Alias" on page 195.

list-audit-modules(1)   Lists all audit modules. Supported in remote mode only. For procedural information in this guide, see "To List Audit Modules" on page 197.

list-message-security-providers(1)   Lists the security message providers for the given message layer. Supported in remote mode only. For procedural information in this guide, see "To List Message Security Providers" on page 227.

list-password-aliases(1)   Lists existing password aliases. Supported in remote mode only. For procedural information, see "To List Password Aliases" on page 195.

update-password-alias(1)   Changes the alias for a specified password. Supported in remote mode only. For procedural information, see "To Update a Password Alias" on page 196.

Thread Pool Subcommands

create-threadpool(1)   Creates a new thread pool. Supported in remote mode only. For procedural information in this guide, see "To Create a Thread Pool" on page 102.

delete-threadpool(1)   Deletes the specified thread pool. Supported in remote mode only. For procedural information in this guide, see "To Delete a Thread Pool" on page 104.

list-threadpools(1)   Lists the existing thread pools. Supported in remote mode only. For procedural information in this guide, see "To List Thread Pools" on page 103.

Transaction Service Subcommands

freeze-transaction-service(1) Freezes the transaction subsystem during which time all the in-flight transactions are suspended. Supported in remote mode only. For procedural information, see "To Stop the Transaction Service" on page 325.

recover-transactions(1) Manually recovers pending transactions. Supported in remote mode only. For procedural information, see "To Manually Recover Transactions" on page 327.

rollback-transaction(1) Rolls back the named transaction. Supported in remote mode only. For procedural information, see "To Roll Back a Transaction" on page 325.

unfreeze-transaction-service(1) Resumes all the suspended in-flight transactions. Invoke this subcommand on an already frozen transaction. Supported in remote mode only. For procedural information, see "To Restart the Transaction Service" on page 326.

User Management Subcommands

create-auth-realm(1) Adds the specified authentication realm. Supported in remote mode only. For procedural information in this guide, see "To Create an Authentication Realm" on page 206.

create-file-user(1) Creates a file user in a given file-based authentication realm. An entry is added to the keyfile with the specified user name, password, and groups. Multiple groups can be created by separating each one with a colon (:). Supported in remote mode only. For procedural information in this guide, see "To Create a File User" on page 211.

delete-auth-realm(1) Deletes the specified authentication realm. Supported in remote mode only. For procedural information in this guide, see "To Delete an Authentication Realm" on page 208.

delete-file-user(1) Deletes the specified user entry in the keyfile. Supported in remote mode only. For procedural information in this guide, see "To Delete a File User" on page 213.

list-auth-realms(1) Lists the existing authentication realms. Supported in remote mode only. For procedural information in this guide, see "To List Authentication Realms" on page 207.

list-file-users(1)   Lists the file users supported by the file realm authentication
                          method. Supported in remote mode only. For procedural
                          information in this guide, see "To List File Users" on page 211.

list-file-groups(1)   Lists groups for a file user, or all groups if the --name option is not
                          specified. For procedural information in this guide, see "To List File
                          Groups" on page 212.

update-file-user(1)   Updates an existing entry in the keyfile using the specified user
                          name, password, and groups. Supported in remote mode only. For
                          procedural information in this guide, see "To Update a File User"
                          on page 213.

Index

K