Oracle® GlassFish Server 3.1 Quick Start Guide

Part No: 821–2432–12
July 2011

ORACLE

111130@25097

Contents

◆ ◆ ◆ CHAPTER 1

Quick Start for Basic Features

Oracle GlassFish Server provides a server for the development and deployment of Java Platform, Enterprise Edition (Java EE platform) applications and web technologies based on Java technology. GlassFish Server 3.1 provides the following:

- A lightweight and extensible core based on OSGi Alliance standards
- A web container
- An easy-to-use Administration Console for configuration and management
- Update Tool connectivity for updates and add-on components
- Support for high availability clustering and load balancing

The following topics are addressed here:

About This Quick Start Guide

Oracle GlassFish Server 3.1 Quick Start Guide demonstrates key features of the GlassFish Server product and enables you to quickly learn the basics. Step-by-step procedures introduce you to product features and enable you to use them immediately.

This guide assumes that you have already obtained and installed the GlassFish Server 3.1 software. For more information about installing GlassFish Server 3.1, see the *Oracle GlassFish Server 3.1 Installation Guide*.

Instructions and examples in this guide that apply to all supported operating systems use the forward slash character (`/`) as path separators in all file names and commands. Ensure that you use the correct character for the system on which GlassFish Server is installed. For example:

- **UNIX, Linux, or Mac OS X systems:** *as-install*`/bin/asadmin`
- **Windows systems:** *as-install*`\bin\asadmin`

This guide provides basic information only. For comprehensive information about GlassFish Server and other entities mentioned in this guide, see "For More Information" on page 18.

To review additional details about this release before you begin using the software, see the *Oracle GlassFish Server 3.1-3.1.1 Release Notes*. The *Release Notes* provide important information about the GlassFish Server 3.1 release, including details about new features, information about known issues and possible workarounds, and tips for installing and working with GlassFish Server 3.1 software.

Default Paths and File Names

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

TABLE 1–1 Default Paths and File Names

Placeholder	Description	Default Value
as-install	Represents the base installation directory for GlassFish Server. In configuration files, *as-install* is represented as follows: `${com.sun.aas.installRoot}`	Installations on the Oracle Solaris operating system, Linux operating system, and Mac OS operating system: *user's-home-directory*`/glassfish3/glassfish` Windows, all installations: *SystemDrive*`:\glassfish3\glassfish`
as-install-parent	Represents the parent of the base installation directory for GlassFish Server.	Installations on the Oracle Solaris operating system, Linux operating system, and Mac operating system: *user's-home-directory*`/glassfish3` Windows, all installations: *SystemDrive*`:\glassfish3`
domain-root-dir	Represents the directory in which a domain is created by default.	*as-install*`/domains/`

TABLE 1–1 Default Paths and File Names *(Continued)*

Placeholder	Description	Default Value
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*

Default Administration Values

The following table lists default administration values for GlassFish Server. See "Default Paths and File Names" on page 6 for more information about the *as-install* and *domain-dir* placeholders.

TABLE 1–2 Default Administration Values

Item	Default Value or Location
Domain name	`domain1`
Master password	`changeit`
`asadmin(1M)` command-line utility	*as-install*`/bin`
Configuration files	*domain-dir*`/config`
Log files	*domain-dir*`/logs`
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`

Starting and Stopping the Default Domain

When you install GlassFish Server, a default domain named `domain1` is created. The following procedures describe how to start and stop `domain1` when it is the only domain. For information about starting and stopping a domain when there are multiple domains, see Chapter 3, "Administering Domains," in *Oracle GlassFish Server 3.1 Administration Guide*.

▼ To Start the Default Domain

Before You Begin GlassFish Server software must be installed before you start the domain.

- **Run the `asadmin start-domain` command without an operand:**

  as-install`/bin/asadmin start-domain`

  The command starts the default domain, `domain1`.

▼ To Stop the Default Domain

- **Run the `asadmin stop-domain` command without an operand:**

  as-install`/bin/asadmin stop-domain`

  The command stops the default domain, `domain1`.

Tip – To determine whether a domain is running, use the `asadmin list-domains` command:

as-install`/bin/asadmin list-domains`

Starting and Stopping the Database Server

A database server is not started by default when you start the GlassFish Server domain. If your applications require a database back end, you must start and stop the database server manually.

The following procedures describe how to start and stop the Java DB server that is bundled with GlassFish Server. For information about starting and stopping other database servers, see the documentation for your specific product.

For the list of database products supported in this release, see the *Oracle GlassFish Server 3.1-3.1.1 Release Notes*.

For more information about database connectivity, see Chapter 12, "Administering Database Connectivity," in *Oracle GlassFish Server 3.1 Administration Guide*.

▼ To Start the Java DB Server

Before You Begin At least one GlassFish Server domain must be started before you start the database server.

- **Run the `asadmin start-database` command.**

  The general form for the command is as follows:

  as-install`/bin/asadmin start-database --dbhome` *directory-path*

  For example, to start the Java DB server from its default location:

  as-install`/bin/asadmin start-database --dbhome` *as-install-parent*`/javadb`

▼ To Stop the Java DB Server

- **Run the `asadmin stop-database` command:**

  as-install`/bin/asadmin stop-database`

Starting the Administration Console

The GlassFish Server Administration Console provides a browser interface for configuring, administering, and monitoring GlassFish Server.

▼ To Start the Administration Console

Before You Begin At least one GlassFish Server domain must be started.

1. **Type the URL in your browser.**

   The default URL for the Administration Console on the local host is as follows:

   `http://localhost:4848`

2. **If prompted, log in to the Administration Console.**

   You will be prompted to log in if you chose to require an administration password at the time GlassFish Server was installed.

See Also For more information, see the Administration Console online help.

Deploying and Undeploying Applications

The process of configuring and enabling applications to run within the GlassFish Server framework is referred to as *deployment*.

This section explains how to deploy, list, and undeploy applications. The procedures in this section use the `hello.war` sample application. The following topics are addressed here:

▼ To Obtain the Sample Application

1 **Download a copy of the `hello.war` sample application from `http://glassfish.java.net/downloads/quickstart/hello.war`.**

2 **Save the `hello.war` file in the directory of your choice.**

This directory is referred to as *sample-dir*.

Deploying and Undeploying the Sample Application From the Command Line

GlassFish Server provides `asadmin` subcommands for performing the following deployment-related tasks:

▼ To Deploy the Sample Application From the Command Line

Before You Begin The sample application must be available before you start this task. To download the sample, see "To Obtain the Sample Application" on page 10. At least one GlassFish Server domain must be started before you deploy the sample application.

1 **Run the `asadmin deploy` command.**

The general form for the command is as follows:

as-install`/bin/asadmin deploy` *war-name*

To deploy the `hello.war` sample, the command is as follows:

as-install`/bin/asadmin deploy` *sample-dir*`/hello.war`

2. **Access the `hello` application by typing the following URL in your browser:**

   `http://localhost:8080/hello`

   The application's start page is displayed, and you are prompted to type your name.

   ```
   Hi, my name is Duke. What's yours?
   ```

3. **Type your name and click Submit.**

   The application displays a customized response, giving you a personal `Hello`.

See Also For more information about the `deploy`subcommand, see `deploy(1)`.

For more information about deploying applications from the command line, see *Oracle GlassFish Server 3.1 Application Deployment Guide*.

▼ To List Deployed Applications From the Command Line

- **Run the `asadmin list-applications` command:**

  as-install`/bin/asadmin list-applications`

▼ To Undeploy the Sample Application From the Command Line

- **Run the `asadmin undeploy` command.**

  The general form for the command is as follows:

  as-install`/bin/asadmin undeploy` *war-name*

  For *war-name*, use the literal `hello`, not the full `hello.war` name.

  For the `hello.war` example, the command is as follows:

  as-install`/bin/asadmin undeploy hello`

See Also For more information about the `undeploy` subcommand, see `undeploy(1)`.

Deploying and Undeploying Applications by Using the Administration Console

The graphical Administration Console of GlassFish Server enables you to perform the following deployment-related tasks:

▼ To Deploy the Sample Application by Using the Administration Console

Before You Begin The sample application must be available before you start this task. To download the sample, see "To Obtain the Sample Application" on page 10. At least one GlassFish Server domain must be started before you deploy the sample application.

1. **Launch the Administration Console by typing the following URL in your browser:**

   `http://localhost:4848`

2. **Click the Applications node in the tree on the left.**

   The Applications page is displayed.

3. **Click the Deploy button.**

   The Deploy Applications or Modules page is displayed.

4. **Select Packaged File to be Uploaded to the Server, and click Browse.**

5. **Navigate to the location in which you saved the `hello.war` sample, select the file, and click Open.**

   You are returned to the Deploy Applications or Modules page.

6. **Specify a description in the Description field, for example:**

   `hello`

7. **Accept the other default settings, and click OK.**

   You are returned to the Applications page.

8. **Select the check box next to the `hello` application and click the Launch link to run the application.**

   The default URL for the application is as follows:

   `http://localhost:8080/hello/`

See Also For more information, see the Administration Console online help.

▼ To View Deployed Applications in the Administration Console

1. **Launch the Administration Console by typing the following URL in your browser:**

   `http://localhost:4848`

2. **Click the Applications node in the tree on the left.**

   Expand the node to list deployed applications. Deployed applications are also listed in the table on the Applications page.

▼ To Undeploy the Sample Application by Using the Administration Console

1. **Launch the Administration Console by typing the following URL in your browser:**

   `http://localhost:4848`

2. **Click the Applications node in the tree on the left.**

   The Applications page is displayed.

3. **Select the check box next to the `hello` sample application.**

4. **Remove or disable the application.**

   - **To remove the application, click the Undeploy button.**
   - **To disable the application, click the Disable button.**

See Also For more information, see the Administration Console online help.

Deploying and Undeploying the Sample Application Automatically

GlassFish Server enables you to performing the following deployment-related tasks automatically:

- "To Deploy the Sample Application Automatically" on page 14
- "To Undeploy the Sample Application Automatically" on page 14

▼ To Deploy the Sample Application Automatically

You can deploy applications automatically by placing them in the *as-install*`/domains/`*domain-name*`/autodeploy` directory, where *domain-name* is the name of the domain for which you want to configure automatic deployment. For this example, use the default domain, `domain1`:

as-install`/domains/domain1/autodeploy`

Before You Begin The sample application must be available before you start this task. To download the sample, see "To Obtain the Sample Application" on page 10.

- **Copy the application WAR file to the *as-install*`/domains/`*domain-name*`/autodeploy` directory.**

  - **On UNIX, Linux, and Mac OS X systems, type this command:**

    `cp` *sample-dir*`/hello.war` *as-install*`/domains/`*domain-name*`/autodeploy`

  - **On Windows systems, type this command:**

    `copy` *sample-dir*`\hello.war` *as-install*`\domains\`*domain-name*`\autodeploy`

  GlassFish Server automatically discovers and deploys the application. The default URL for the application is as follows:

  `http://localhost:8080/hello/`

▼ To Undeploy the Sample Application Automatically

1. **Change to the domain's `autodeploy` directory.**

   `cd` *as-install*`/domains/`*domain-name*`/autodeploy`

2. **Delete the sample application's WAR file to undeploy and remove the application.**

   - **On UNIX, Linux, and Mac OS X systems, type this command:**

     `rm hello.war`

   - **On Windows systems, type this command:**

     `del hello.war`

High Availability Clustering and Load Balancing

GlassFish Server enables multiple GlassFish Server instances to be clustered to provide high availability through failure protection, scalability, and load balancing. The subsections that follow provide an overview of high availability clustering and load balancing for GlassFish Server. For a complete example of setting up high availability clustering and load balancing, see "Deploying an Application to a Two-Instance Cluster" on page 21.

Clusters of GlassFish Server Instances

A *cluster* is a collection of GlassFish Server instances that work together as one logical entity. A cluster provides a runtime environment for one or more Java Platform, Enterprise Edition (Java EE) applications. A cluster provides high availability through failure protection, scalability, and load balancing.

A GlassFish Server *instance* is a single Virtual Machine for the Java platform (Java Virtual Machine or JVM machine) on a single node in which GlassFish Server is running. A node defines the host where the GlassFish Server instance resides. The JVM machine must be compatible with the Java Platform, Enterprise Edition (Java EE).

GlassFish Server instances form the basis of an application deployment. An instance is a building block in the clustering, load balancing, and session persistence features of GlassFish Server. Each instance belongs to a single domain and has its own directory structure, configuration, and deployed applications. Every instance contains a reference to a node that defines the host where the instance resides.

For more information, see the following documentation:

- Chapter 3, "Administering GlassFish Server Nodes," in *Oracle GlassFish Server 3.1-3.1.1 High Availability Administration Guide*
- Chapter 4, "Administering GlassFish Server Clusters," in *Oracle GlassFish Server 3.1-3.1.1 High Availability Administration Guide*
- Chapter 5, "Administering GlassFish Server Instances," in *Oracle GlassFish Server 3.1-3.1.1 High Availability Administration Guide*

Session Persistence and Failover

Storing session state data enables the session state to be recovered after the failover of an instance in a cluster. Recovering the session state enables the session to continue without loss of information. GlassFish Server supports in-memory session replication on other servers in the cluster for maintaining HTTP session and stateful session bean data.

For more information, see Chapter 10, "Configuring High Availability Session Persistence and Failover," in *Oracle GlassFish Server 3.1-3.1.1 High Availability Administration Guide*.

Load Balancing for Clustered Configurations

GlassFish Server supports web server and hardware-based load balancing for clustered configurations. A load balancer is deployed with a cluster, and provides the following features:

- Allows an application or service to be scaled horizontally across multiple physical (or logical) hosts yet still present the user with a single URL
- Insulates the user from host failures or server crashes when used with session persistence
- Enhances security by hiding the internal network from the user

Oracle GlassFish Server includes a Load Balancer Plug-in for popular web servers such as Oracle HTTP Server, Oracle iPlanet Web Server, Apache HTTP Server, and Microsoft Windows IIS. The Load Balancer Plug-in includes a graphical Load Balancer Configurator installation wizard that makes it easy to configure the plug-in to work with your particular GlassFish Server and web server installations.

GlassFish Server load balancing configurations can vary widely depending on the needs of your enterprise. For complete information about configuring load balancing in GlassFish Server, see the following documentation:

- Chapter 7, "Configuring Web Servers for HTTP Load Balancing," in *Oracle GlassFish Server 3.1-3.1.1 High Availability Administration Guide*
- Chapter 8, "Configuring HTTP Load Balancing," in *Oracle GlassFish Server 3.1-3.1.1 High Availability Administration Guide*
- Chapter 12, "RMI-IIOP Load Balancing and Failover," in *Oracle GlassFish Server 3.1-3.1.1 High Availability Administration Guide*

Updating and Extending an Existing Installation

GlassFish Server provides an administrative tool called Update Tool that enables you to install updates and add-on components to your existing GlassFish Server installation. Update Tool can be accessed as a standalone graphical tool from the command line or as a browser-based graphical tool from the Administration Console. For more information about Update Tool, see "Update Tool" in *Oracle GlassFish Server 3.1 Administration Guide*.

▼ To Access the Graphical Update Tool From the Command Line

- **Run the `updatetool` command:**

  as-install-parent`/bin/updatetool`

If Update Tool is not installed, you will be prompted to install it. Install the tool if desired, then use the `updatetool` command to start the tool. Extensive online help is available from the tool's Help menu.

See Also A command-line interface is also available for Update Tool. The command-line interface uses the `pkg` command and enables you to perform most of the tasks provided by the graphical version. For more information about the `pkg` command, see Chapter 11, "Extending and Updating GlassFish Server," in *Oracle GlassFish Server 3.1 Administration Guide*.

▼ To Access Update Tool by Using the Administration Console

Before You Begin At least one GlassFish Server domain must be started before you launch the Administration Console.

1. **Launch the Administration Console by typing the following URL in your browser:**

   `http://localhost:4848`

2. **Click the Update Tool node in the tree on the left.**

See Also For more information, see the Administration Console online help.

Removing GlassFish Server 3.1 Software

Before removing the GlassFish Server software, stop the following processes:

- All domains and other related processes
- Command prompts that use the installation directory or its subdirectories
- The Update Tool notifier process if present
- Any applications that use files that are part of the Java Platform, Standard Edition (Java SE)

For more information about performing these tasks, see Chapter 2, "Uninstalling GlassFish Server 3.1," in *Oracle GlassFish Server 3.1 Installation Guide*.

▼ To Remove GlassFish Server Software on UNIX, Linux, and Mac OS X Systems

1. **Change to the *as-install-parent* directory, which contains the uninstallation program.**
2. **If necessary, grant execute permissions to the uninstallation program file.**
   ```
   chmod +x ./uninstall.sh
   ```
3. **Run the uninstallation program.**
   ```
   ./uninstall.sh
   ```
4. **Examine the contents of the remaining installation directories and remove any files or directories that you do not want, including hidden directories prefixed with a dot.**

▼ To Remove GlassFish Server Software on Windows Systems

1. **Change to the *as-install-parent* directory, which contains the uninstallation program.**
2. **Run the uninstallation program.**
   ```
   uninstall.exe
   ```
3. **Examine the contents of the remaining installation directories and remove any files or directories that you do not want, including hidden directories prefixed with a dot.**

For More Information

Additional resources are available to help you learn more about GlassFish Server 3.1 and related technologies.

The following resources are described here:

Product Documentation

Comprehensive product documentation is available and includes the following.

- *Oracle GlassFish Server 3.1-3.1.1 Release Notes*: Latest details about new features, known issues, and tips for installing and working with GlassFish Server software.
- GlassFish Server Documentation Library (`http://download.oracle.com/docs/cd/E18930_01/index.htm`): Collection of guides that document GlassFish Server features and functions.
- GlassFish Server Screencasts (`http://wikis.sun.com/display/GlassFish/Screencasts+and+Other+Videos`): Collection of video recordings that demonstrate various features and provide examples for working with GlassFish Server and related technologies.
- GlassFish Server FAQs (`http://wikis.sun.com/display/GlassFish/GlassFishFAQIndex`): Frequently asked questions covering a variety of GlassFish Server topics.

GlassFish Communities

The following resources will help you connect with other users, learn more about GlassFish Server, and get help if needed.

- GlassFish Forum (`http://www.java.net/forums/glassfish/glassfish`): Public online discussion forum that provides community support and tips for working with GlassFish Server.
- GlassFish Wiki (`http://wikis.sun.com/display/GlassFish`): Community site that provides a wide range of information related to GlassFish Server.
- GlassFish Documentation Project (`http://glassfish.java.net/docs/project.html`): Documentation community site that provides details about GlassFish Server documentation and how you can participate.
- GlassFish Quality Community (`http://glassfish.java.net/quality/portal`): Quality community site focused on testing and improving GlassFish Server.

Tutorials

The following tutorials provide working examples and detailed instructions for creating enterprise applications for the Java EE 6 platform.

- *Your First Cup: An Introduction to the Java EE Platform*: Provides a short tutorial for beginning Java EE programmers that shows how to develop a simple enterprise application from scratch. The sample application consists of four main components: a JAX-RS RESTful web service, an enterprise bean, a Java Persistence API entity, and a web application created with JavaServer Faces Facelets technology.
- *The Java EE 6 Tutorial*: Provides a beginner's guide to developing enterprise applications for GlassFish Server. The tutorial includes working examples and instructions for creating applications with Java EE 6 technologies, including Java Servlets, JavaServer Faces, Facelets,

RESTful Web Services, Enterprise JavaBeans, Java Persistence API, Contexts and Dependency Injection for the Java EE platform, and more. The document is also available through Update Tool.

Java EE 6 Samples

The sample applications demonstrate Java EE 6 technologies. The samples are available through Update Tool and also as part of the Java EE 6 SDK distributions. The SDK distributions are available from the Java EE downloads page (`http://www.oracle.com/technetwork/java/javaee/downloads/index.html`).

◆ ◆ ◆ CHAPTER 2

Use Cases for Production Deployments

The examples and procedures in Chapter 1, "Quick Start for Basic Features," introduce the features and capabilities of GlassFish Server, but are not intended for production deployments. This chapter provides examples of how GlassFish Server can be used in production.

The following topics are addressed here:

Deploying an Application to a Two-Instance Cluster

This example provides all the steps for configuring a cluster in which iPlanet Web Server is used with the load balancer plug-in for load balancing of two GlassFish Server instances. A simple web application is deployed to this cluster. Users of this application access the application through a virtual server.

Note – In this example, line breaks are included for enhanced readability. These line breaks are not part of the syntax of the commands.

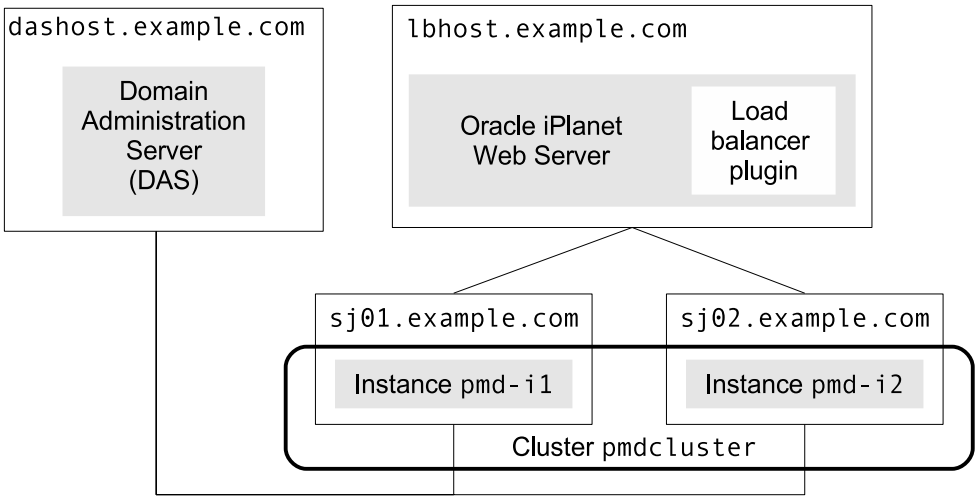
The assumptions for this example are as follows:

- The domain administration server (DAS) will run on the host `dashost.example.com`.
- The web server and load balancer plug-in will run on the host `lbhost.example.com`.
- The instances will run on the hosts `sj01.example.com` and `sj02.example.com`
- All steps are performed by the user `gfuser`.
- On the DAS host, the parent of the base installation directory of the Oracle GlassFish Server software is `/home/gfuser/glassfish3`.

- The path of user `gfuser` contains the directory `/home/gfuser/glassfish3/bin`.
- Secure shell (SSH) software is installed and configured on each host.
- The SSH Server Daemon `sshd` is running on each host.

The configuration of the cluster in this example is shown in the following figure.

FIGURE 2–1 Sample Two-Instance Cluster



The following topics are addressed here:

- "To Install and Configure the GlassFish Server Cluster" on page 22
- "To Install and Configure iPlanet Web Server for Load Balancing" on page 25
- "To Install the Load Balancer" on page 26
- "To Deploy the Application and Configure the Load Balancer" on page 27

▼ To Install and Configure the GlassFish Server Cluster

All steps in this procedure are performed from the DAS host.

1 Start the `asadmin` utility in multiple command mode (multimode).

```
dashost$ asadmin
Use "exit" to exit and "help" for online help.
```

2 Set up public key authentication without encryption on the hosts where the instances will run.

```
asadmin> setup-ssh sj01.example.com sj02.example.com
SSH key not found for user gfuser
Would you like to generate a SSH key pair (without a key passphrase) for gfuser to
```

```
access [sj01.example.com, sj02.example.com]? [y/n]: yes
Enter SSH password for gfuser@sj01.example.com>
Created directory /home/gfuser/.ssh
/usr/bin/ssh-keygen successfully generated the identification /home/gfuser/.ssh/id_rsa
Copied keyfile /home/gfuser/.ssh/id_rsa.pub to gfuser@sj01.example.com
Successfully connected to gfuser@sj01.example.com using keyfile
/home/gfuser/.ssh/id_rsa
Successfully connected to gfuser@sj02.example.com using keyfile
/home/gfuser/.ssh/id_rsa
SSH public key authentication is already configured for gfuser@sj02.example.com
Command setup-ssh executed successfully.
```

3 Copy the installation of GlassFish Server software from the DAS host to the hosts where the instances will run.

```
asadmin> install-node --installdir /export/glassfish3
sj01.example.com sj02.example.com
Created installation zip /home/gfuser/glassfish3033977962688704206.zip
Successfully connected to gfuser@sj01.example.com using keyfile
/home/gfuser/.ssh/id_rsa
Copying /home/gfuser/glassfish3033977962688704206.zip (90012883 bytes) to
sj01.example.com:/export/glassfish3
Installing glassfish3033977962688704206.zip into sj01.example.com:/export/glassfish3
Removing sj01.example.com:/export/glassfish3/glassfish3033977962688704206.zip
Fixing file permissions of all files under sj01.example.com:/export/glassfish3/bin
Successfully connected to gfuser@sj02.example.com using keyfile
/home/gfuser/.ssh/id_rsa
Copying /home/gfuser/glassfish3033977962688704206.zip (90012883 bytes) to
sj02.example.com:/export/glassfish3
Installing glassfish3033977962688704206.zip into sj02.example.com:/export/glassfish3
Removing sj02.example.com:/export/glassfish3/glassfish3033977962688704206.zip
Fixing file permissions of all files under sj02.example.com:/export/glassfish3/bin
Command install-node executed successfully.
```

4 Start the domain `domain1`.

```
asadmin> start-domain domain1
Waiting for domain1 to start .........................
Successfully started the domain : domain1
domain  Location: /home/gfuser/glassfish3/glassfish/domains/domain1
Log File: /home/gfuser/glassfish3/glassfish/domains/domain1/logs/server.log
Admin Port: 4848
Command start-domain executed successfully.
```

5 Enable secure administration for the DAS host.

```
asadmin> enable-secure-admin
Command enable-secure-admin executed successfully.
```

6 Restart the domain `domain1`.

```
asadmin> restart-domain domain1
Successfully restarted the domain
Command restart-domain executed successfully.
```

7 Add the cluster `pmdcluster` to the DAS configuration.

```
asadmin> create-cluster pmdcluster
Command create-cluster executed successfully.
```

8 Create a node for each host on which the instances will run.

a. Create the node `sj01` to represent the host `sj01.example.com`.

```
asadmin> create-node-ssh --nodehost sj01.example.com
--installdir /export/glassfish3 sj01
Command create-node-ssh executed successfully.
```

b. Create the node `sj02` to represent the host `sj02.example.com`.

```
asadmin> create-node-ssh --nodehost sj02.example.com
--installdir /export/glassfish3 sj02
Command create-node-ssh executed successfully.
```

9 Add the instances `pmd-i1` and `pmd-i2` to the cluster `pmdcluster`.

a. Add the instance `pmd-i1` on the node `sj01`.

```
asadmin> create-instance --node sj01 --cluster pmdcluster pmd-i1
Command _create-instance-filesystem executed successfully.
Port Assignments for server instance pmd-i1:
JMX_SYSTEM_CONNECTOR_PORT=28686
JMS_PROVIDER_PORT=27676
HTTP_LISTENER_PORT=28080
ASADMIN_LISTENER_PORT=24848
JAVA_DEBUGGER_PORT=29009
IIOP_SSL_LISTENER_PORT=23820
IIOP_LISTENER_PORT=23700
OSGI_SHELL_TELNET_PORT=26666
HTTP_SSL_LISTENER_PORT=28181
IIOP_SSL_MUTUALAUTH_PORT=23920
The instance, pmd-i1, was created on host sj01.example.com
Command create-instance executed successfully.
```

b. Add the instance `pmd-i2` on the node `sj02`.

```
asadmin> create-instance --node sj02 --cluster pmdcluster pmd-i2
Command _create-instance-filesystem executed successfully.
Port Assignments for server instance pmd-i2:
JMX_SYSTEM_CONNECTOR_PORT=28686
JMS_PROVIDER_PORT=27676
HTTP_LISTENER_PORT=28080
ASADMIN_LISTENER_PORT=24848
JAVA_DEBUGGER_PORT=29009
IIOP_SSL_LISTENER_PORT=23820
IIOP_LISTENER_PORT=23700
OSGI_SHELL_TELNET_PORT=26666
HTTP_SSL_LISTENER_PORT=28181
IIOP_SSL_MUTUALAUTH_PORT=23920
The instance, pmd-i2, was created on host sj02.example.com
Command create-instance executed successfully.
```

10 Start the cluster `pmdcluster`.

```
asadmin> start-cluster pmdcluster
Command start-cluster executed successfully.
```

11 Confirm that the instances in the cluster `pmdcluster` are running.

```
asadmin> list-instances
pmd-i1   running
pmd-i2   running
Command list-instances executed successfully.
```

12 End the multimode session for the `asadmin` utility.

```
asadmin> exit
Command multimode executed successfully.
```

▼ To Install and Configure iPlanet Web Server for Load Balancing

All steps in this procedure are performed from the host where the web server and load balancer plug-in will run.

Before You Begin This example assumes that the path of user `gfuser` contains the `/home/gfuser/webserver7/bin` directory.

1 Download the web server software from the Oracle iPlanet Web Server 7.0.9 download page.

2 Extract the contents of the download file.

```
lbhost$ unzip Oracle-iPlanet-Web-Server-7.0.9-solaris-sparc.zip
```

3 Start the iPlanet Web Server installation wizard.

```
lbhost$ setup &
```

4 Follow the onscreen instructions in the installation wizard to install iPlanet Web Server.

Other steps in this procedure assume the following option settings:

- Installation Directory: `/home/gfuser/webserver7`
- Type of Installation: Express
- Administrator User Name: `admin`
- Start Administration Server: checked

5 Start the `wadm` utility in multimode.

```
lbhost$ wadm --user=admin
Please enter admin-user-password>
Connected to localhost:8989
Oracle iPlanet Web Server 7.0.9 B07/04/2010 02:15
```

6 **Create a self-signed certificate for secure communication between the load balancer plug-in and the DAS.**

```
wadm> create-selfsigned-cert --server-name=lbhost.example.com
--nickname cert-lbhost --token=internal --config=lbhost
CLI201 Command 'create-selfsigned-cert' ran successfully
```

7 **Create a secure HTTP listener for the iPlanet Web Server instance `lbhost`.**

```
wadm> create-http-listener --server-name lbhost.example.com
--default-virtual-server-name=lbhost --listener-port 8082
--config lbhost http-listener-ssl
CLI201 Command 'create-http-listener' ran successfully
```

8 **Enable SSL with optional client authentication and assign the certificate for the HTTP listener.**

```
wadm> set-ssl-prop --http-listener http-listener-ssl --config lbhost enabled=true
client-auth=optional server-cert-nickname=cert-lbhost
CLI201 Command 'set-ssl-prop' ran successfully
```

9 **Deploy the configuration for the iPlanet Web Server instance `lbhost`.**

```
wadm> deploy-config lbhost
CLI201 Command 'deploy-config' ran successfully
```

10 **End the multimode session for the `wadm` utility.**

```
wadm> exit
```

▼ To Install the Load Balancer

Unless otherwise stated, all steps in this procedure are performed from the host where the web server and load balancer plug-in will run.

1 **From the DAS host, export the self-signed certificate of the DAS to enable secure communication between the HTTP listener and the DAS.**

```
dashost$ keytool -export -rfc -alias s1as
-keystore /home/gfuser/glassfish3/glassfish/domains/domain1/config/keystore.jks
-file ./s1as.rfc
Enter keystore password:
Certificate stored in file <./s1as.rfc>
```

2 **Transfer the DAS certificate file `s1as.rfc` from the DAS host to the host where the web server and load balancer plug-in will run.**

3 **Download the GlassFish Loadbalancer Configurator 3.1 from the Components section of the Oracle GlassFish Downloads page.**

4 **Extract the contents of the download file.**

```
lbhost$ unzip glassfish-lbconfigurator-3_1.zip
Archive:  glassfish-lbconfigurator-3_1.zip
  inflating: glassfish-lbconfigurator-3_1.jar
```

5 **Start the GlassFish Loadbalancer Configurator.**

```
lbhost$ java -jar glassfish-lbconfigurator-3_1.jar
```

6 **Follow the onscreen instructions in the GlassFish Loadbalancer Configurator to install and configure the load balancer plug-in.**

Other steps in this procedure assume the following option settings:

- Web Server Instance Directory: `/home/gfuser/webserver7/https-lbhost`
- DAS Certificate File: `s1as.rfc`

7 **Start the iPlanet Web Server instance `lbhost`, to which the load balancer plug-in was installed.**

```
lbhost$ wadm start-instance --user=admin --config=lbhost
Please enter admin-user-password>
CLI204 Successfully started the server instance.
```

▼ To Deploy the Application and Configure the Load Balancer

All steps in this procedure are performed from the DAS host.

Before You Begin This example assumes that the host name `pmd.example.com` is registered with the DNS server for the `example.com` domain.

1 **Start the `asadmin` utility in multimode.**

```
dashost$ asadmin
Use "exit" to exit and "help" for online help.
```

2 **Create the virtual server `pmdserver` to represent the virtual host `pmd.example.com`.**

```
asadmin> create-virtual-server --hosts pmd.example.com
--networklisteners http-listener-1 --target pmdcluster pmdserver
Command create-virtual-server executed successfully.
```

3 **Deploy the `hello` application to the cluster `pmdcluster`.**

```
asadmin> deploy --availabilityenabled=true --target pmdcluster
--virtualservers pmdserver /home/gfuser/apps/hello.war
Application deployed with name hello.
Command deploy executed successfully.
```

4 **Create the HTTP load balancer configuration `pmdcluster-lb-config` for the cluster `pmdcluster`.**

```
asadmin> create-http-lb --devicehost lbhost.example.com
--deviceport 8082 --target pmdcluster  --lbenableallinstances
--lbenableallapplications=hello pmdcluster-lb-config
Command create-http-lb executed successfully.
```

5 **Apply the HTTP load balancer configuration `pmdcluster-lb-config`.**

```
asadmin> apply-http-lb-changes pmdcluster-lb-config
Command apply-http-lb-changes executed successfully.
```

6 **End the multimode session for the `asadmin` utility.**

```
asadmin> exit
Command multimode executed successfully.
```

7 **To access the application, open the location `https://pmd.example.com:8082/hello/` in a web browser.**

Configuring an Oracle Data Source

This example demonstrates how to configure an Oracle 11 database as a JDBC resource for an application. The information in this example is based on Chapter 12, "Administering Database Connectivity," in *Oracle GlassFish Server 3.1 Administration Guide*, which explains how to configure any database that is supported by GlassFish Server as a JDBC resource.

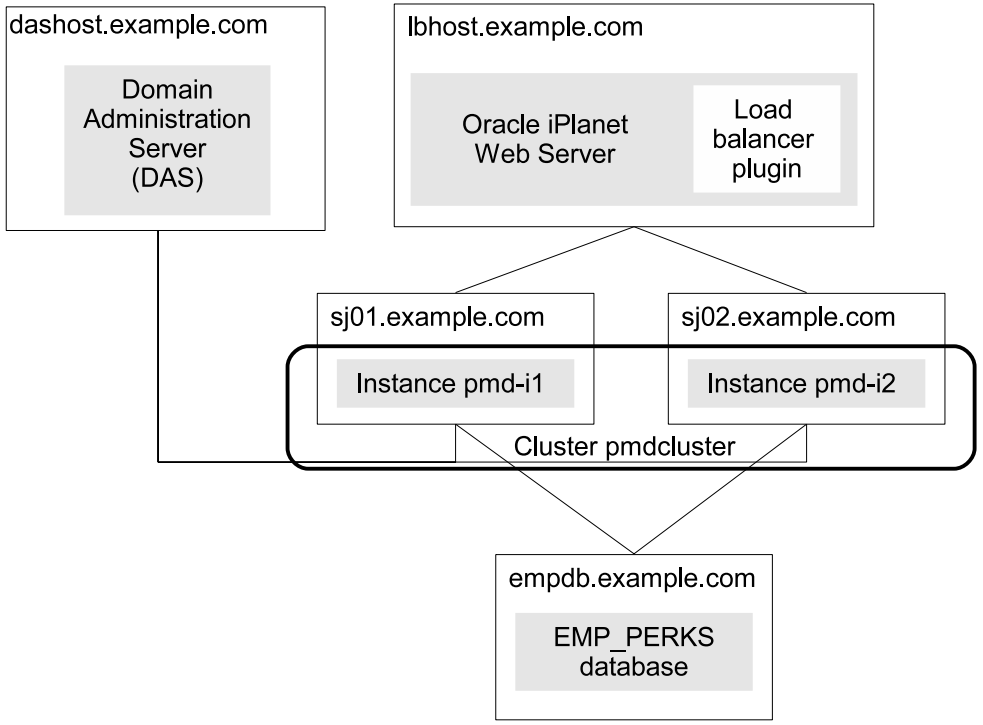
The database in this example is used by the HR application perk-olator, which provides information to employees about special savings the company has arranged for its employees.

The assumptions for this example are as follows:

- GlassFish Server has been installed and configured according to the example "Deploying an Application to a Two-Instance Cluster" on page 21.
- The details about the database that the application uses are as follows:
  - The Oracle 11 database is running on `empdb.example.com` and listening for connections on the default port (1521).
  - The name of the database is `EMP_PERKS`.
  - The database user name is `perk_app`, and this user's password is `perks4emps`.
  - The perk-olator application looks up the JNDI name `jdbc/emp_perks` to access the data source.

The configuration of the components in this example is shown in the following figure.

FIGURE 2–2 Sample Two-Instance Cluster Accessing an Oracle Database



Configuring the Oracle 11 database as a JDBC resource for the perk-olator application involves the following tasks:

1. Integrating the JDBC driver for Oracle 11 into GlassFish Server.
2. Creating a JDBC connection pool for the resource.
3. Creating the JDBC resource.

Note – In this example, line breaks are included for enhanced readability. These line breaks are not part of the syntax of the commands.

▼ To Integrate the JDBC Driver into GlassFish Server

To integrate the JDBC driver, you copy its JAR file into the domain and then restart the domain and instances to make the driver available.

1. **On `dashost.example.com`, copy the JAR file for the JDBC driver into the domain's `lib` subdirectory.**

```
dashost$ cd /home/gfuser/glassfish3
dashost$ cp oracle-jdbc-drivers/ojdbc6.jar glassfish/domains/domain1/lib
```

2. **Start the `asadmin` utility in multiple command mode (multimode).**

```
dashost$ asadmin
Use "exit" to exit and "help" for online help.
```

3. **Restart the domain to make the JDBC driver available to the domain administration server (DAS).**

```
asadmin> restart-domain domain1
Command restart-domain executed successfully.
```

4. **Restart instances in the domain to make the JDBC driver available to them.**

```
asadmin> list-instances
pmd-i1   running
pmd-i2   running
Command list-instances executed successfully.
asadmin> restart-instance pmd-i1
Command restart-instance executed successfully.
asadmin> restart-instance pmd-i1
Command restart-instance executed successfully.
```

5. **Exit the multimode session for the `asadmin` utility.**

```
asadmin> exit
Command multimode executed successfully.
```

▼ To Create a JDBC Connection Pool

Use the `create-jdbc-connection-pool(1)` subcommand to create the JDBC connection pool, specifying the database connectivity values provided to you.

1. **On `dashost.example.com`, start the `asadmin` utility in multiple command mode (multimode).**

```
dashost$ asadmin
Use "exit" to exit and "help" for online help.
```

2. **Create the JDBC connection pool.**

```
asadmin> create-jdbc-connection-pool --restype javax.sql.DataSource
--datasourceclassname oracle.jdbc.pool.OracleDataSource
--property "user=perk_app:password=perks4emps:
url=jdbc\\:oracle\\:thin\\:@empdb.example.com\\:1521\\:EMP_PERKS"
Emp_Perks-Pool
JDBC connection pool Emp_Perks-Pool created successfully.
pmd-i1:
JDBC connection pool Emp_Perks-Pool created successfully.

pmd-i2:
```

```
JDBC connection pool Emp_Perks-Pool created successfully.

Command create-jdbc-connection-pool executed successfully.
```

In this command, note the use of two backslashes (\\) preceding the colons in the `url` property value. These backslashes cause the colons to be interpreted as part of the property value instead of as separators between *property*=*value* pairs.

3 Verify connectivity to the database.

```
asadmin> ping-connection-pool Emp_Perks-Pool
Command ping-connection-pool executed successfully.
```

4 Exit the multimode session for the `asadmin` utility.

```
asadmin> exit
Command multimode executed successfully.
```

▼ To Create a JDBC Resource

Use the `create-jdbc-resource(1)` subcommand to create the JDBC resource, making sure to name it so that the perk-olator application can discover it using JNDI lookup.

- **On `dashost.example.com`, create the JDBC resource and target it to the `pmdcluster` cluster.**

```
dashost$ asadmin create-jdbc-resource --connectionpoolid Emp_Perks-Pool
--target pmdcluster jdbc/emp_perks
JDBC resource jdbc/emp_perks created successfully.
pmd-i1:
JDBC resource jdbc/emp_perks created successfully.

pmd-i2:
JDBC resource jdbc/emp_perks created successfully.

Command create-jdbc-resource executed successfully.
```

Next Steps

After creating the Oracle data source for the perk-olator application, you need to deploy the application itself. Then, as the application is used over time, you can set a variety of JDBC connection pool features to optimize performance. For information about these features, see "Configuring Specific JDBC Connection Pool Features" in *Oracle GlassFish Server 3.1 Administration Guide*.

Configuring Transport Layer Security (TLS)

As described in "Certificates and SSL" in *Oracle GlassFish Server 3.1 Security Guide*, 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. The newest version of the SSL standard is called Transport Layer Security (TLS). GlassFish Server supports the SSL 3.0 and the TLS 1.0 encryption protocols.

The following procedure lists the major tasks for configuring GlassFish Server for TLS/SSL. The procedure also provides cross-references to detailed instructions for performing each task.

▼ To Configure GlassFish Server for TLS/SSL

1 Set up the keystore and truststore for a domain.

By default, the keystore (`keystore.jks`) and truststore (`cacerts.jks`) for a domain are created in the *domain-dir*`/config` directory when you create the domain. The domain creation process creates a primary (private) key and a self-signed certificate for the DAS, and a separate private key and self-signed certificate for remote instances.

When you create a domain you can use the `create-domain(1)` subcommand `--keytooloptions` to specify the common name (CN) of the host that is to be used for the self-signed certificate. By default, the name is the fully-qualified name of the host where you run the `create-domain` subcommand.

GlassFish Server generates self-signed certificates suitable for internal testing. The self-signed certificates that GlassFish Server generates are typically not trusted by clients by default because a certificate authority does not vouch for the authenticity of the certificate. For example, browsers will warn you, let you view the certificate, and ask you to reject the certificate, accept it once, or accept it indefinitely.

You can use your tool of choice, such as keytool, to list the default self-signed certificates in the keystore, similar to the following:

Note – You can list some limited contents of the keystore without supplying a password. However, for a request that affects the private key, such as the `keytool.exe --certreq` option, the keystore password is required.

```
keytool.exe -list -v -keystore keystore.jks


Enter keystore password:

Keystore type: JKS
Keystore provider: SUN
```

```
Your keystore contains 2 entries

Alias name: glassfish-instance
Creation date: Apr 14, 2011
Entry type: PrivateKeyEntry
Certificate chain length: 1
Certificate[1]:
Owner: CN=system01.somedomain-instance, OU=GlassFish, O=Oracle Corporation,
L=Santa Clara, ST=California, C=US
Issuer: CN=system01.somedomain-instance, OU=GlassFish, O=Oracle Corporation,
 L=Santa Clara, ST=California, C=US
Serial number: 4da74a98
Valid from: Thu Apr 14 15:27:20 EDT 2011 until: Sun Apr 11 15:27:20 EDT 2021
Certificate fingerprints:
         MD5:  00:FA:CF:65:19:7B:B2:02:62:66:DE:68:7B:BA:AE:93
         SHA1: 11:E2:06:54:84:B3:67:8C:2E:AD:B6:4C:E9:E1:B9:A0:07:A7:CE:B9
         Signature algorithm name: SHA1withRSA
         Version: 3

Extensions:

#1: ObjectId: 2.5.29.14 Criticality=false
SubjectKeyIdentifier [
KeyIdentifier [
0000: 5D 37 CB 75 70 B8 52 4B   91 C6 A7 D3 FB BF 22 3F  ]7.up.RK......"?
0010: 5D AE D7 74                                        ]..t
]
]



*******************************************
*******************************************


Alias name: s1as
Creation date: Apr 14, 2011
Entry type: PrivateKeyEntry
Certificate chain length: 1
Certificate[1]:
Owner: CN=system01.somedomain, OU=GlassFish, O=Oracle Corporation, L=Santa C
lara, ST=California, C=US
Issuer: CN=system01.somedomain, OU=GlassFish, O=Oracle Corporation, L=Santa
Clara, ST=California, C=US
Serial number: 4da74a94
Valid from: Thu Apr 14 15:27:16 EDT 2011 until: Sun Apr 11 15:27:16 EDT 2021
Certificate fingerprints:
         MD5:  23:EA:3F:89:E6:34:31:21:C8:D6:47:88:30:05:3B:50
         SHA1: 8B:9E:86:AE:E4:71:C4:8E:70:99:DB:3E:93:6C:BC:E3:DB:15:D1:B6
         Signature algorithm name: SHA1withRSA
         Version: 3

Extensions:

#1: ObjectId: 2.5.29.14 Criticality=false
SubjectKeyIdentifier [
KeyIdentifier [
0000: 87 BB 44 61 54 3C 09 15   8C 4F 1E 13 8B 37 46 AB  ..DaT<...O...7F.
```

```
0010: 66 27 F9 A2                                        f'..
]
]
```

You can instead have GlassFish Server use trusted certificates for this purpose by adding one or more valid certificates and CA root in the keystore and truststore, respectively. Oracle strongly recommends that you use certificates signed by an accepted CA in a production environment.

Keep the following points in mind:

- If GlassFish Server uses self-signed certificates, you need to include them directly in the client-side truststore.
- If GlassFish Server uses certificates that are signed using a CA, import the CA root certificate into the client's truststore.
- If you use certificates other than the GlassFish Server defaults, take note of the alias names you use. You will need the alias names later in this procedure when you configure the HTTP Listener for SSL.

For more information on using your own certificates, see the following documentation:

- "To Generate a Certificate by Using keytool" in *Oracle GlassFish Server 3.1 Security Guide*
- "To Sign a Certificate by Using keytool" in *Oracle GlassFish Server 3.1 Security Guide*

2 Optionally, configure the client for two-way SSL.

With two-way SSL (SSL with client authentication), GlassFish Server presents a certificate to the client and the client presents a certificate to GlassFish Server.

In this case, you must ensure that GlassFish Server is able to validate the certificate that the client uses to digitally sign its request, and that GlassFish Server in turn uses to encrypt its responses to the client. Do one of the following:

- Make sure the client uses a digital certificate that GlassFish Server automatically trusts because it has been issued by a trusted certificate authority.
- Make sure the client uses an individual certificate that is already in the GlassFish Server keystore and therefore already trusted.

3 Create a Listener Port

See "To Create an Internet Connection" in *Oracle GlassFish Server 3.1 Administration Guide*.

Note – 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 HTTP listener has an associated HTTP protocol.

---

a. Ensure that the server is running.

Remote subcommands require a running server.

b. Create an HTTPS protocol by using the `create-protocol(1)` subcommand with the `--securityenabled` option.

(The listener named `http-listener-2` has security (SSL) enabled by default. To use this built-in `http-listener-2` HTTPS protocol, skip this step.)

See "To Create a Protocol" in *Oracle GlassFish Server 3.1 Administration Guide*.

c. Create an HTTP configuration for this protocol by using the `create-http(1)` subcommand.

(If you used the built-in `http-listener-2` HTTPS protocol, skip this step.)

See "To Create an HTTP Configuration" in *Oracle GlassFish Server 3.1 Administration Guide*.

d. Optionally, create a transport by using the `create-transport(1)` subcommand.

To use the built-in `tcp` transport, skip this step. You generally do not need another transport in addition to the default `tcp` transport.

See "To Create a Transport" in *Oracle GlassFish Server 3.1 Administration Guide*.

e. Optionally, 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," in *Oracle GlassFish Server 3.1 Administration Guide*.

f. Create an HTTP listener by using the `create-network-listener(1)` subcommand.

Specify the previously chosen protocol, and optionally a transport and thread pool.

```
asadmin> create-network-listener --listenerport 7272
protocol http-listener-2 --enabled=true sampleListener
Command create-network-listener executed successfully.
```

For more information, see "To Create an HTTP Network Listener" in *Oracle GlassFish Server 3.1 Administration Guide*.

g. Configure the HTTP Listener for SSL

The `create-ssl(1)` subcommand creates and configures the SSL element in the selected HTTP listener, IIOP listener, or IIOP service to enable secure communication on that listener/service.

You use the `create-ssl` subcommand to specify SSL2, SSL3, TLS, to set cipher suites, to enable two-way (client-auth) SSL, and so forth. By default, SSL3 and TLS are enabled and all cipher suites are enabled.

If you enabled two-way SSL for the client, you must also enable it for GlassFish Server by setting the `-clientauthenabled` option.

You must specify the alias (`--certname`) of the certificate in this subcommand.

For example, this example enables the HTTP listener named *sampleListener* for SSL with client authentication enabled. The alias name `s1as` identifies the default GlassFish Server certificate.

```
asadmin> create-ssl --type http-listener
--certname  s1as  --clientauthenabled  sampleListener
Command create-ssl executed successfully.
```

4 To activate your changes, restart GlassFish Server.

Shortcut for Configuring GlassFish Server for TLS/SSL

You can use the `create-http-listener(1)` subcommand to create a network listener configured for SSL that uses the HTTPS protocol without having to first create a protocol, transport, or HTTP con figuration. This subcommand is a convenient shortcut, but it gives access to only a limited number of options.

▼ Shortcut: To Configure GlassFish Server for TLS/SSL

1 Set up the keystore and truststore for a domain.

2 Optionally, configure the client for two-way SSL.

3 Create an HTTP Network Listener with the create-http-listener subcommand

a. Ensure that the server is running.

Remote subcommands require a running server.

b. Run create-http-listener with the `--securityenabled` option.

If the `--securityenabled` options is set to true, the HTTP listener runs SSL. The security setting globally enables or disables SSL by making certificates available to the server instance. The default value is false.

For example:

```
asadmin> create-http-listener --listeneraddress 0.0.0.0
--listenerport 443 --securityenabled=true --enabled=true
--default-virtual-server server sampleListener
Command create-http-listener executed successfully.
```

You cannot use the `create-http-listener` subcommand to specify SSL2, SSL3, TLS, to set cipher suites, or to enable client authentication. Instead, when you set `--securityenabled` to true, both SSL3 are TLS are enabled, all cipher suites are chosen, and client authentication is not enabled. With the exception of the client authentication case, these defaults should be acceptable in most cases.

To change the defaults, you need to explicitly set these elements. For example:

```
asadmin> get configs.config.server-config.network-con
fig.protocols.protocol.sampleListener.ssl.client-auth-enabled
configs.config.server-config.network-config.protocols.protocol.sampleListener.ss
l.client-auth-enabled=false
Command get executed successfully.

asadmin> set configs.config.server-config.network-con
fig.protocols.protocol.sampleListener.ssl.client-auth-enabled=true
configs.config.server-config.network-config.protocols.protocol.sampleListener.ss
l.client-auth-enabled=true
Command set executed successfully.
```

For more information, see "To Create an HTTP Network Listener" in *Oracle GlassFish Server 3.1 Administration Guide*.

4 To activate your changes, restart GlassFish Server.