DZone Refcardz

CONTENTS INCLUDE:

GlassFish Application Server

By Masoud Kalali

ABOUT GLASSFISH

The use of GlassFish application server is growing, and having a reference card for day-to-day jobs is inevitable. Looking for a sample command to perform a specific job can take time but by using this refcard, you won't need to look for any commands or lose time searching in countless pages of manuals and administration references. This refcard covers administration, security, and performance management topics.

WHAT IS GLASSFISH?

GlassFish is a Java EE application server which is hosted on Java.net and mainly sponsored by Sun Microsystems. GlassFish usually has full support of the latest Java EE related JSRs. It is accessible both under GPL and CDDL licenses.

What makes it different from other products:

- Support for the latest web services specification in addition to proven interoperability with Microsoft WCF.
- Support for both older and newer versions of EJBs.
- Cluster wide management and deployment.
- High availability both with in-memory replication and persisted replication using HADB.
- Fine grained monitoring API based on JMX in addition to already available monitoring facilities in its administration console.
- Support for Ruby on Rails, in addition to PHP support, using Quercus makes GlassFish a suitable application server for hosting heterogeneous applications.

WHERE TO GET GLASSFISH

Usually more than one version of GlassFish is available for download, which includes the current stable version, the previous stable version with all patch and post ported features, and development builds of the next GlassFish version. You should choose the current version unless you are looking for a maintenance release for past versions, or you are eager to check out new JSRs or functionalities.

Downloading the right version of GlassFish: If you are going to deploy some applications in a production environment you should choose a current stable version. And if you have a very mission critical application, but you do not require to have the latest standards, go with the previous stable version. Development versions are only suitable for experimental tasks and not for production. The current version of GlassFish is GlassFish Version 2, update release 2, which is available at: https://glassfish.dev.java.net/downloads/v2ur2-b04.html.

Distribution packages are available for six different operating systems as follows:

Operating Systems	English Language Distribution Package	Multilingual Distribution Package
Solaris SPARC Platform	glassfish-installer-v2ur2-b04-sunos.jar	glassfish-installer-v2ur2-b04-sunos-ml.jar
Solaris x86 Platform	glassfish-installer-v2ur2-b04-sunos_x86.jar	glassfish-installer-v2ur2-b04-sunos_x86-ml.jar
Windows Platform	glassfish-installer-v2ur2-b04-windows.jar	glassfish-installer-v2ur2-b04-windows-ml.jar
Linux Platform	glassfish-installer-v2ur2-b04-linux.jar	glassfish-installer-v2ur2-b04-linux-ml.jar
MacOS Platform	glassfish-installer-v2ur2-b04-darwin.jar	glassfish-installer-v2ur2-b04-darwin-ml.jar
AIX Platform	glassfish-installer-v2ur2-b04-aix.jar	glassfish-installer-v2ur2-b04-aix-ml.jar

Selecting a GlassFish version: There are several downloads available for each version of GlassFish. Each of these packages is suitable for one operating system so you need to ensure to get the package designated for your own OS. Go to https://glassfish.dev.java.net/public/downloadsindex.html to access the GlassFish download page.

Sun Microsystems' distribution of GlassFish: Sun Microsystems provides an alternate package of GlassFish application server. Sun's package comes with an installer, integrated HADB for the enterprise version, an integrated Open ESB, Portlet container, access manager, and some other projects. Sun's distribution is located at http://java.sun.com/javaee/downloads/index.jsp.

GlassFish documentation: There is plenty of documentation which will help you to get started or to continue with using GlassFish. Some of the most important:

- GlassFish tech tips master index: https://glassfish.dev.java.net/public/TipsandBlogs.html
- Master index for all versions of GlassFish documentation: https://glassfish.dev.java.net/javaee5/docs/DocsIndex.html
- GlassFish Wiki: http://wiki.glassfish.java.net/

INSTALLING GLASSFISH

How to install GlassFish: In order to install GlassFish you will only need to execute the following commands in the console window:

1. *java -Xmx256m -jar glassfish-installer-vx-bx-osname.jar*
   It will extract the packed file, while vx and bx will be replaced by version and build number.
2. *ant -f setup.xml*

If you do not have ANT in your environment then you can navigate to the extracted GlassFish directory and execute:

lib/ant/bin/ant -f setup.xml

The first installer will install GlassFish in the developer profile and the second one will setup GlassFish in the enterprise profile, which is a cluster aware configuration.

GlassFish Directory Structure

Each GlassFish installation can host multiple domains that work independently with independent configuration areas like security, access port, and libraries. You deploy your applications inside these domains, and the domains are the entities which span your server. Figure 1 shows typical content of a GlassFish installation directory.

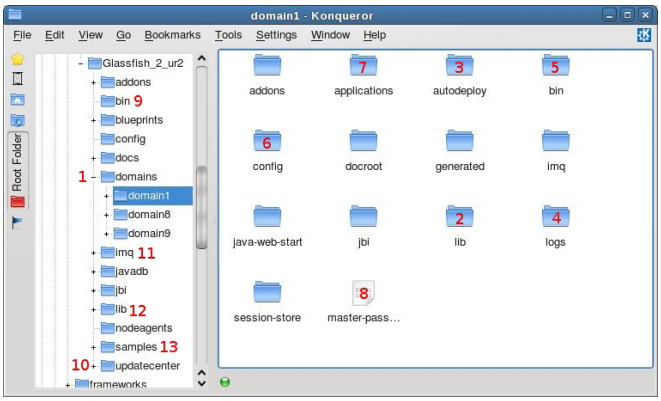


Figure 1: GlassFish installation directory structure.

1. Domains parent folder is typically where a created domain will reside.
2. A designated domain lib folder is where you should put your common libraries like JDBC driver, cache libraries, and web framework library.
3. Copy any deployable package into this folder, and GlassFish will pick it up and deploy it using default configuration.
4. All logs that you may need for troubleshooting are inside this directory.
5. Scripts to directly start and stop the domain without using *asadmin* reside inside this folder.
6. Domain configuration files like *domain.xml* and *default-web.xml* are inside this directory.
7. Deployed applications reside in the sub-directory of this directory.
8. Domain master password which protects domain certification store files, is stored inside this file.
9. *asadmin* and other commands like *wsgen*, *wsimport*, *xjc*, and *jspc* reside inside this folder.
10. Inside this folder resides one of the most important features of GlassFish. It lets you add or update your application server functionalities.
11. JMS implementation of GlassFish applicaton server resides in this folder. There are some administration tools for JMS broker administration inside its bin folder. GlassFish uses Open MQ as the default JMS implementation.
12. The library folder of the application server which is shared between all domains that are hosted in the instance and certainly all applications that are deployed inside those domains.
13. Sample application which will help you learn Java EE faster and easier.

GLASSFISH ADMINISTRATION CHANNEL

GlassFish provides you with several administration channels which can be used in different situations.

Web-based administration console: You can access the Administration console using a browser. GlassFish uses a separate http listener for the administration console in order to provide a detailed, and completely separate configuration for it.

You may access the web-based administration console using http://127.0.0.1:4848/. In case you are trying to access the administration console from a remote place, change the IP address to the address which resembles your server address. The default administration credentials are ***admin*** as username and ***adminadmin*** as password.

Command line administration console: This is the preferred way for the experienced administrators who can't quit the command line habit. However, some tasks like creating/removing/backing-up and restoring a domain is only possible using the CLI (Command Line Interface). To enter the command line administration console, navigate to GlassFish_home/bin and execute *./asadmin*. There are two ways to execute a command using *./asadmin* or *asadmin.bat*.

1. Pass the command as a parameter to the asadmin when you want to execute it.

   ./asadmin help

2. Execute the asadmin and then execute the command in the asadmin console.

   asadmin>help

Hot Tip

You should know that you can get general help by executing the *help* command in the administration console, and you can get detailed help for each command by executing *help command-name*.

GlassFish Administration Channel, continued

JMX (Java Management eXtension) / AMX (Application Server Management eXtension): This channel is mostly preferred by developers who need to interact with the GlassFish administration core by code. These APIs allow you to extend monitoring and management facilities of the GlassFish application server. In order to connect to the JMX administration console, you can use any JMX console like JDK's JConsole or MC4J, and then connect to the GlassFish server using *IP: 8686* and its administration credentials. The port may differ based on your selected port number during domain creation.

JConsole is a part of the JDK (Java Development Kit). You can run it by issuing the *jconsole* command in your operating system command line interface.

MC4J is a JMX management console built on top of NetBeans RCP. It is highly modular and easy to use. MC4J can be found at http://www.mc4j.org.

GLASSFISH DOMAINS PROFILE

Each GlassFish domain can be created in one of the following three profiles. Each profile has its own characteristics, and therefore, its own benefits and drawbacks.

Profile	Characteristics
Developer	Suitable for developers who develop and test applications; embedded JMS implementation; no clustering; heart-beat support
Clustered	Can join a cluster, support heartbeat, has both embedded and local JMS implementation and can also support in-memory session replication.
Enterprise	This domain is comprised of all the Cluster profile features, in addition to utilizing NSS for security store, replication using HADB, and enabled platform security manager by default

You should consider that when you have no replication, you have the best possible performance provided by the application server, and using any kind of replication either in memory or HADB degrades the performance. *--profile* lets you determine your domain profile during domain creation. Here is a detailed profiles comparison:

Profiles			
Features	**Developer**	**Cluster**	**Enterprise**
Security Manager	Disabled	Disabled	Enabled
Security Store	JKS	JKS	NSS
HTTP Access Log	Disabled	Disabled	Enabled
JVM Software	Client	Client	JDK
Session Replication	No	In Memory	HADB
Cluster Support	No	Yes	Yes
GMS Heartbeat	No	Yes	Yes
Quick Startup	Yes	No	No

Hot Tip

If you would like to change the default 60 minutes time out of the web-based administration console to something smaller or bigger, you can use the following command in *asadmin* CLI:

set server.admin-service.das-config.admin-session-timeout-in-minutes= NUMBER

COMMON ADMINISTRATION TASKS

Some common administration tasks that you may encounter in performing your daily jobs are categorized as follows:

Task	Command	Description
To create a domain	*create-domain --adminuser admin --adminport 4848 --instanceport 8080 --profile cluster --domainproperties domain.jmxPort=8686:http.ssl.port=8181 domain8*	Although the command gives you much more control, the ones mentioned are enough.
To start a domain	*start-domain --verbose=true domain8*	By using ***--verbose=true*** you can see detailed output about what application server is doing and spot any possible problem it may encounter.
To stop a domain	*stop-domain domain8*	It will stop the ***domain8***; you will need to include domain name if you have more than one domain in the domains directory.
To delete a domain	*delete-domain domain8*	It will delete the given domain (***domain8***)
To backup a domain	*backup-domain domain8*	This command will create a backup from a domain named domain8. The domain is in the default domains directory of GlassFish. Otherwise you will need to pass ***--domaindir***, which points to a parent directory of the designated domain. Backups will reside inside a directory named backups, which is inside the domain8 directory.
Restore a domain backup	*restore-domain --filename /opt/dev/apps/Glassfish_2_ur2/domains/domain8/backups/sjsas_backup_v00001.zip domain8*	As you can see, we pass the complete path of the backup file in order to allow the asadmin to process and restore the given domain based on its content.

Following are some additional commands related to domain management:

Command	Description
list-domains	Lists all available domains. You can pass *--domaindir <PATH_TO_DOMAINS_DIRECTORY>* in order to list domains that are not in the default domains directory
verify-domain-xml	Verify the domain.xml file to check its consistency. You may issue this command after you edit the *domain.xml* manually. It takes domain name as its required operand

DEPLOYING, UNDEPLOYING AND MONITORING APPLICATIONS, AND MANAGED RESOURCES

GlassFish provides web-based administration console and CLI for performing administrative tasks. Supporting both CLI and web-based console will let administrators with different tastes manage their GlassFish installation from their favorite channel. For example, a veteran administrator may write some scripts for performing daily jobs while another administrator may prefer to use web-based console to manage the application server. In this refcard I will use CLI path as it is easier and saves time.

Deploying and undeploying an application: To deploy an application independently from its type (WAR, EAR, RAR, etc.), the following command will do the job. The command has many options which are omitted to show the common use case. Application server should be running to execute the command.

deploy ***--name samplename*** *--contextroot samplecontext --upload=true --dbvendorname derby --createtables=true /home/masoud/customer-cmp-ear.ear*

Deploying, Undeploying and Monitoring Applications and Managed Resources, continued

```
asadmin> deploy --host 127.0.0.1 --port 4848 --user admin -name samplename --con
textroot samplecontext --upload=true --dbvendorname derby --createtables=true
/home/masoud/customer-cmp-ear.ear
```

The above image shows how a command will look in the console (cmd in windows, terminal or gnome-terminal in Linux).

However, when you are deploying into a cluster of domains you should use *--target* parameter to determine that you want to deploy it into the cluster instead of the default server. Required JDBC resources should already be created in order to automatically create and drop tables.

To undeploy:

undeploy --droptables=true ***samplename***

You may omit host, port and user parameters if your target server is the default local server and you are not going to provide alternative user name and passwords.

Create and delete a resource. (JMS, JDBC)

To create the JDBC connection pool:

create-jdbc-connection-pool --datasourceclassname com.mysql.jdbc.jdbc2.optional.MysqlConnectionPoolDataSource --restype javax.sql.DataSource --steadypoolsize 8 --maxpoolsize 32 --maxwait 60000 --poolresize 2 --idletimeout 300 --isolationlevel read-committed --isisolationguaranteed --isconnectvalidatereq=true --validationmethod auto-commit --property User=root:Password=passwd:URL="jdbc\:mysql\://192.168.100.9/sampledatabase" ***sample-mysql-pool***

Make sure that you take careful look at how we escape ":" in each property, like URL, it is also the sign that we should use to separate different properties. Many of the above parameters are not required if you are going to use default ones. Usually the common attributes *--steadypoolsize* and *--maxpoolsize* have default values.

In order to delete the created connection pool you will only need to execute:

delete-jdbc-connection-pool ***sample-mysql-pool***

Now to create a JDBC resource which your applications typically will look up and use in order to interact with the database:

create-jdbc-resource --connectionpoolid ***sample-mysql-pool*** *--host 127.0.0.1* ***--port 4849 --secure*** *--user admin* ***jdbc/sample-jdbc-resource***

Take a closer look at what is written in bold. If you want to connect to the server using a secure connection then you should include *--secure* parameter, and also use the https port of the application server administration console.

To delete the created resource:

delete-jdbc-resource ***jdbc/sample-jdbc-resource***

Here are some commands related to JDBC resource management:

Command	Description
list-jdbc-connection-pools	Lists all JDBC connection pools.
list-jdbc-resources	Lists all JDBC resources.

GlassFish provides a rich set of commands to manage JMS resources like JMS hosts, destinations and resources. I will demonstrate creating JMS resource and JMS destinations.

CREATE A JMS RESOURCE

create-jms-resource --restype javax.jms.Topic jms/sample-topic

Other resource types:

- javax.jms.Queue
- javax.jms.TopicConnectionFactory
- javax.jms.QueueConnectionFactory

Here are some related commands:

Command	Description
delete-jms-resource	Deletes a JMS resource by using its JNDI name.
jms-ping	Checks the default JMS server status, whether it is running or not.
list-jms-resources	Lists all JMS resources.

MONITOR A RESOURCE

GlassFish provides very good monitoring facilities. You can monitor the application server using all three administration channels mentioned earlier. Monitoring is a bit different than other commands, because first you should know what you want to monitor, then you need to change the level of monitoring, and finally you can monitor the element.

1. Find the monitoring status of all components that can be monitored using the following command:

   *get server.monitoring-service.module-monitoring-levels.**

2. Change the level of monitoring to HIGH or LOW (by default it is OFF)

   set server.monitoring-service.module-monitoring-levels.jdbc-connection-pool=HIGH

3. Monitor the element that you like. In our case, the *sample-jdbc-resource*:

   monitor --type jdbcpool --filter **sample-mysql-pool** *--filename /opt/out2.csv --interval 5 server*

The target instance that we monitor is the "server" instance, and we only like to view monitoring data for one connection pool. The command stores the monitoring data in a csv file.

SECURE THE GLASSFISH APPLICATION SERVER

1. Make sure to change both the administration password and the master password of your domain after you create it. The following commands will change the passwords. They don't need any parameters if you work with the default domain.

   change-admin-password
   change-master-password

2. Make sure to configure listeners to only listen on a specific IP address, and not all available addresses. For example, for a http listener, the steps are as follows:
   - Open the web-based administration console and use your administration credentials: http://127.0.0.1:4848/
   - From the left panel, navigate to *Configuration > HTTP Service > HTTP Listeners > http-listener-1*
   - Change the *Network Address* field to your designated address.
   - Run GlassFish under its own user, and provide only the required permission for the GlassFish user.

Secure the GlassFish Application Server, continued

3. Enable and configure "Security Manager" to ensure that GlassFish, or any deployed application, won't do anything unorthodox. There are two levels of policy configuration possible:

Policy Configuration	Policy File Path
Domain wide	*domain_dir/config/server.policy*
Single application level	*domain_dir/generated/policy/application-name/applicationname/granted.policy*

4. Don't save any password in plain text. Use an alias instead. If you are concerned about your database password being accessible in *domain.xml* file you can use an alias to store the password, and then use that alias when you create the JDBC connection pool:

   create-password-alias sample-alias

   Anywhere that you need to use the above password just enter *${ALIAS=sample-alias}* as the password placeholder

   Related commands:

Command	Description
delete-password-alias	Delete a password alias
list-password-aliases	List all password aliases
update-password-alias	Update a password alias with a new password

5. Do not type your administration password when using CLI. Instead, save the password into a file and pass the file to the command using *--passwordfile* parameter. To do this, the following steps are required:
   - Create a plain text file somewhere accessible by a GlassFish user. For example, *passfile.txt*.
   - Create a password alias. For example, ***admin-alias*** using ***create-password-alias***
   - Add *AS_ADMIN PASSWORD=${ALIAS=admin-alias}* to *passfile.txt*.

Now you can use this file to avoid typing your administration password.

Hot Tip

Remote debugging

You can start GlassFish in debug mode and then attach a debugger to its JVM. Usually the port number for remote debugger attachment is 9009. To start GlassFish in debug mode you can use the following command in the *asadmin* environment:

start-domain --debug DOMAIN_NAME

Some debuggers which can be used:

NetBeans IDE: http://www.netbeans.org
JSwat: http://jswat.sourceforge.net/
JDebugTool: http://www.debugtools.com/

GLASSFISH AND PERFORMANCE

Usually there are several places that we should check and fine tune in order to get better performance from our available resources. Application Server is one of the places which can be tuned without affecting the application code. There are several places which we may fine tune to get better performance, such as:

- Memory. JVM has several memory sections and each one has its own purpose. You should fine tune how much memory should be assigned to each section. You may use GC Portal, a product developed by Sun Microsystems to help you fine tune the JVM's Garbage Collection configuration parameters, or you can simply view the memory consumption rate by using a simple JMX client like JDK's *jconsole*. GC Portal is available at java.sun.com/developer/technicalArticles/Programming/GCPortal/
- Change the JVM Type to server by going to Application Server > JVM Settings > JVM Options. You can also change all JVM parameters for memory management, such as:

  -XX:MaxPermSize=192m
  -Xmx512m
  -XX:NewRatio=2

- Request Processing Threads number:

  Configuration > HTTP Service > Request Processing

Usually this number is close to your number of CPUs or CPU cores.

- HTTP listener Acceptor Threads number:

  Configuration> HTTP Service> HTTP Listeners> A-Listener-Name

Changing this number depends on what kind of resources each request needs to be fulfilled. For example, it only needs CPU or database and process management is involved too.

- Connection pool attributes like Maximum Pool Size, Maximum Wait Time, and vendor specific attributes such as caching.
- Thread pools configuration, if you use any.
- Make sure to change some development related attributes of *default-web.xml* to ensure better performance. The file is located inside the directory numbered 6 in Figure 1.

```
<init-param>
  <param-name>development</param-name>
  <param-value>false</param-value>
</init-param>
<init-param>
  <param-name>genStrAsCharArray</param-name>
  <param-value>true</param-value>
</init-param>
```

Definitions should be included in *JspServlet* definition.

TROUBLESHOOTING

The most important source for troubleshooting is in the log files. GlassFish stores log files in directory #4 in Figure 1 on page 2.

Hot Tip

server.log is the most important log file that GlassFish creates, so it will be the source for finding possible problems.

Troubleshooting, continued

Configuring the logging behavior and logging level :

Application Server> Logging> General

You may use rotation to have old log files stored for further review or you may change the logging level of a component, which you think is the source of a problem, by looking at:

Application Server> Logging>Log Levels

Make sure that you never use extensive logging in a production environment.

DIAGNOSTIC REPORTS FOR YOUR DOMAIN

GlassFish can generate a diagnostic report from the application server state. The report can be generated with different levels of detail:

- To configure which components are required to be present in a diagnostic report, navigate to *Configuration > Diagnostic Service.* Here you can choose both a diagnostic report details level and components that need to be included in the report.
- Now you can generate the report either from *Application Server > Diagnostics* or by using the following CLI command:
  generate-diagnostic-report --outputfile /opt/reports/gf/domain1.jar domain1

This report gives you a snapshot of your domain with all related log files and configuration for later review. The report is compressed as a jar file and contains some interrelated HTML files for each type of information you may need.

CLUSTERING AND LOAD BALANCING

A cluster is a logical entity of several homogeneous server instances. Instances can be on a single machine and single subnet, or they can be scattered throughout several machines and subnets. Usually each machine is called a node and has at least one *Node Agent*. The *Node Agent* is a process responsible for the life cycle of the instances (create, stop, start, delete, watchdog) that is running on that machine. To be clear, a server instance is a running domain in a GlassFish installation.

GlassFish application server has clustering capabilities and is compatible with several web servers, or hardware appliances, as the load balancer.

A cluster is usually controlled by a domain that is usually served as the single point of cluster administration and not for serving applications. This domain is called DAS (Domain Administration Server). Administrators use DAS to configure and manage the clusters that are created under it.

GlassFish clusters can:

- Have in-memory replication
- Have persisted replication using HADB

When using HADB, the performance will degrade but the reliability will be the highest because all replicate information will get persisted in multiple instances of HADB.

A cluster needs to have a load balancer to distribute the load (Incoming Requests) between all instances in order to prevent a server from saturation.

Several load balancers such as Sun Java Web server, Tomcat with Mode_JK, Apache HTTPD with Mode_JK or Mode_proxy, F5 traffic managers, or others can be used. However, the only load balancer that can be configured from the DAS is Sun Java Web Server.

Figure 2 shows a complete anatomy of a clustered system.

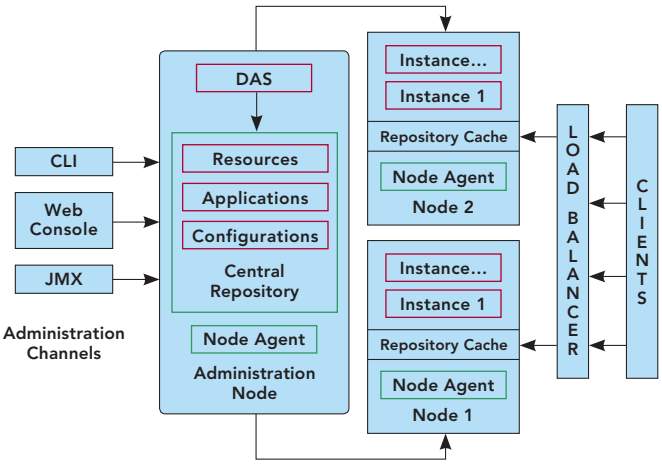


Figure 2.

To set up a cluster you will need GlassFish Application Server, Version 2: https://glassfish.dev.java.net/downloads/v2ur2-b04.html. Make sure that you choose the correct distribution for your operating system.

To set up the DAS, install GlassFish in the cluster or enterprise profile:

ant -f setup-cluster.xml

Now you need to have a (in-memory replicated) cluster in your DAS that will be the single point of management for all of its instances.

*Create-cluster **cluster-01***

Related commands:

Command	Description
delete-cluster	Delete the given cluster—the logical entity not the instance members
list-clusters	List all clusters
start-cluster	Start the given cluster (start all of the cluster instances)
stop-cluster	Stop the given cluster (stop all of the cluster instances)

Install GlassFish in Cluster's nodes:

In each one of the cluster's node, GlassFish should be installed in the Cluster or Enterprise profile in order to be able to create domains with clustering capabilities. To set up GlassFish in a node, use the following command:

ant -f setup-cluster.xml

Create a node agent on the node that will host some instances. Using the *--port* and *--host* is necessary in order to let the Node Agent understand where the DAS is, and that it should communicate and join.

create-node-agent –host <DAS_ADDRESS>
*--port <DAS_ADMINISTRATION_PORT> **node-agent-01***

Now start the node agent

*start-node-agent **node-agent-01***

Related commands:

Command	Description
delete-node-agent	Delete the given node agent
list-node-agents	List all node agents
stop-node-agent	Stop the given node agent

Clustering and Load Balancing, continued

Creating instances is possible both from the DAS administration console or CLI, and from the node's CLI itself. To use DAS CLI to create some instances assigned to a specific cluster:

create-instance --nodeagent ***node-agent-01*** *--* *cluster* ***cluster-01 instance-01***

By starting the cluster, all in-memory information of every instance member will be the same in order to have a homogeneous farm of servers.

start-cluster ***cluster-01***

To set up the load balancer, visit https://glassfish.dev.java.net/javaee5/build/GlassFish_LB_Cluster.html.

Hot Tips

An instance can only be a member of one cluster and not more.

IP multi cast should be allowed between subnets when we have instances on multiple subnets.

Instances which form the cluster virtually shape a ring topology network. Each instance gets a replication date from the previous one and sends its data, in addition to received data, to the next member in the ring. Instances are sorted in alphanumerical order. So the best bet for maximizing the availability, and having all data, is sorting cluster members in a way that none of the instances which resides in a single node come one after another. You can do this by selecting proper names for instances.

ABOUT THE AUTHOR

Masoud Kalali

Masoud Kalali is a GlassFish application server contributor, community award winner, and has been spotlighted as a developer of the Glassfish application server. He holds a software engineering degree and has been working on software development projects since 1999. He is experienced with .Net but his platform of choice is Java. His experience is in software architecture, design and server side development. Masoud's main area of research and interest is Web Services and Service Oriented Architecture.

Blog
http://weblogs.java.net/blog/kalali/

RECOMMENDED BOOK

The complete guide to installing and configuring the GlassFish Application Server and developing Java EE 5 applications to be deployed to this server.

BUY NOW
books.dzone.com/books/glassfish

Get More FREE Refcardz. Visit refcardz.com now!

Upcoming Refcardz:

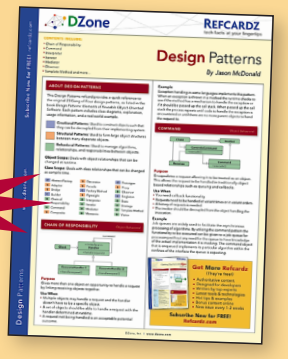
Core Seam
Core CSS: Part III
Hibernate Search
Equinox
EMF
XML
JSP Expression Language
ALM Best Practices
HTML and XHTML

Available:

Essential Ruby
Essential MySQL
JUnit and EasyMock
Getting Started with MyEclipse
Spring Annotations
Core Java
Core CSS: Part II
PHP
Getting Started with JPA
JavaServer Faces
Core CSS: Part I
Struts2
Core .NET
Very First Steps in Flex
C#
Groovy
NetBeans IDE 6.1 Java Editor
RSS and Atom
GlassFish Application Server
Silverlight 2

Visit **refcardz.com** for a complete listing of available Refcardz.

FREE



Design Patterns
Published June 2008

DZone

DZone communities deliver over 4 million pages each month to more than 1.7 million software developers, architects and decision makers. DZone offers something for everyone, including news, tutorials, cheatsheets, blogs, feature articles, source code and more. **"DZone is a developer's dream,"** says PC Magazine.

DZone, Inc.
1251 NW Maynard
Cary, NC 27513

888.678.0399
919.678.0300

Refcardz Feedback Welcome
refcardz@dzone.com

Sponsorship Opportunities
sales@dzone.com

ISBN-13: 978-1-934238-14-1
ISBN-10: 1-934238-14-7
50795
9 781934 238141
$7.95

Version 1.0