Name

create-virtual-server – creates the named virtual server

Synopsis

```
create-virtual-server [--help]
--hosts hosts
[--httplisteners http-listeners]
[--networklisteners network-listeners]
[--defaultwebmodule default-web-module]
[--state={on|off}]
[--logfile log-file]
[--property (name=value)[:name=value]*]
[--target target]
virtual-server-id
```

Description

The `create-virtual-server` subcommand creates the named virtual server. Virtualization in the GlassFish Server allows multiple URL domains to be served by a single HTTP server process that is listening on multiple host addresses. If the application is available at two virtual servers, they still share the same physical resource pools.

This subcommand is supported in remote mode only.

Options

`--help`
`-?`
Displays the help text for the subcommand.

`--hosts`
A comma-separated (,) list of values allowed in the host request header to select the current virtual server. Each virtual server that is configured to the same connection group must have a unique host for that group.

`--httplisteners`
A comma-separated (,) list of HTTP listener IDs. Required only for a virtual server that is not the default virtual server. HTTP listeners are converted to network listeners. This option is deprecated but maintained for backward compatibility. Use `--networklisteners` instead. If `--networklisteners` is used, this option is ignored.

`--networklisteners`
A comma-separated (,) list of network listener IDs. Required only for a virtual server that is not the default virtual server.

`--defaultwebmodule`
The standalone web module associated with this virtual server by default.

`--state`
Determines whether a virtual server is active (`on`) or inactive (`off` or disabled). Default is `on`. When inactive, the virtual server does not service requests.

`--logfile`
Name of the file where log entries for this virtual server are to be written. By default, this is the server log. The file and directory in which the access log is kept must be writable by the user account under which the server runs.

`--property`

Optional property name/value pairs for configuring the virtual server. The following properties are available:

`sso-max-inactive-seconds`

Specifies the number of seconds after which a user's single sign-on record becomes eligible for purging if no client activity is received. Since single sign-on applies across several applications on the same virtual server, access to any of the applications keeps the single sign-on record active. The default value is 300 seconds (5 minutes). Higher values provide longer single sign-on persistence for users, but at the expense of more memory use on the server.

`sso-reap-interval-seconds`

Specifies the number of seconds between purges of expired single sign-on records. The default value is 60.

`setCacheControl`

Specifies a comma-separated list of `Cache-Control` response directives. For a list of valid directives, see section 14.9 of the document at http://www.ietf.org/rfc/rfc2616.txt.

`allowLinking`

If the value of this property is `true`, resources that are symbolic links will be served for all web applications deployed on this virtual server. Individual web applications may override this setting by using the property `allowLinking` under the `sun-web-app` element in the `sun-web.xml` file:

```
<sun-web-app>
<property name="allowLinking" value="[true|false]"/>
</sun-web-app>
```

The default value is `true`.

`accessLogWriteInterval`

Indicates the number of seconds before the log will be written to the disk. The access log is written when the buffer is full or when the interval expires. If the value is 0 (zero), then the buffer is always written even if it is not full. This means that each time the server is accessed, the log message is stored directly to the file.

`accessLogBufferSize`

Specifies the size, in bytes, of the buffer where access log calls are stored.

`allowRemoteAddress`

This is a comma-separated list of regular expression patterns to which the remote client's IP address is compared. If this property is specified, the remote address must match for this request to be accepted. If this property is not specified, all requests will be accepted unless the remote address matches a `denyRemoteAddress` pattern. The default value for this property is null.

`denyRemoteAddress`

This is a comma-separated list of regular expression patterns to which the remote client's IP address is compared. If this property is specified, the remote address must not match for this request to be accepted. If this property is not specified, request acceptance is governed solely by the `allowRemoteAddress` property. The default value for this property is null.

`allowRemoteHost`

This is a comma-separated list of regular expression patterns to which the remote client's host name (as returned by `java.net.Socket.getInetAddress().getHostName()`) is compared. If this property is specified, the remote host name must match for this request to be accepted. If this property is not specified, all requests will be accepted unless the remote host name matches a denyRemoteHost pattern. The default value for this property is null.

`denyRemoteHost`

This is a comma-separated list of regular expression patterns to which the remote client's host name (as returned by `java.net.Socket.getInetAddress().getHostName()`) is compared. If this property is specified, the remote host name must not match for this request to be accepted. If this property is not specified, request acceptance is governed solely by the `allowRemoteHost` property. The default value for this property is null.

`authRealm`

Specifies the `name` attribute of an `auth-realm`, which overrides the server instance's default realm for standalone web applications deployed to this virtual server. A realm defined in a standalone web application's `web.xml` file overrides the virtual server's realm.

`securePagesWithPragma`

Set this property to `false` to ensure that for all web applications on this virtual server file downloads using SSL work properly in Internet Explorer.

You can set this property for a specific web application. For details, see "sun-web-app" in *Sun GlassFish Enterprise Server v3 Preview Application Deployment Guide*.

`contextXmlDefault`

Specifies the location, relative to *domain-dir*, of the `context.xml` file for this virtual server, if one is used. For more information about the `context.xml` file, see "Using a context.xml File" in *Sun GlassFish Enterprise Server v3 Preview Application Development Guide* and The Context Container (`http://tomcat.apache.org/tomcat-5.5-doc/config/context.html`). Context parameters, environment entries, and resource definitions in `context.xml` are supported in the GlassFish Server.

`alternatedocroot_`*n*

Specifies an alternate document root (docroot), where *n* is a positive integer that allows specification of more than one. Alternate docroots allow web applications to serve requests for certain resources from outside their own docroot, based on whether those requests match one (or more) of the URI patterns of the web application's alternate docroots.

If a request matches an alternate docroot's URI pattern, it is mapped to the alternate docroot by appending the request URI (minus the web application's context root) to the alternate docroot's physical location (directory). If a request matches multiple URI patterns, the alternate docroot is determined according to the following precedence order:

- Exact match
- Longest path match
- Extension match

For example, the following properties specify three alternate docroots. The URI pattern of the first alternate docroot uses an exact match, whereas the URI patterns of the second and third alternate docroots use extension and longest path prefix matches, respectively.

```
<property name="alternatedocroot_1"
   value="from=/my.jpg dir=/srv/images/jpg"/>
<property name="alternatedocroot_2"
   value="from=*.jpg dir=/srv/images/jpg"/>
<property name="alternatedocroot_3"
   value="from=/jpg/* dir=/src/images"/>
```

The `value` of each alternate docroot has two components: The first component, `from`, specifies the alternate docroot's URI pattern, and the second component, `dir`, specifies the alternate docroot's physical location (directory). Spaces are allowed in the `dir` component.

You can set this property for a specific web application. For details, see “sun-web-app” in *Sun GlassFish Enterprise Server v3 Preview Application Deployment Guide*.

`send-error_`*n*

Specifies custom error page mappings for the virtual server, which are inherited by all web applications deployed on the virtual server. A web application can override these custom error page mappings in its `web.xml` deployment descriptor. The value of each `send-error_`*n* property has three components, which may be specified in any order:

The first component, `code`, specifies the three-digit HTTP response status code for which the custom error page should be returned in the response.

The second component, `path`, specifies the absolute or relative file system path of the custom error page. A relative file system path is interpreted as relative to the *domain-dir*`/config` directory.

The third component, `reason`, is optional and specifies the text of the reason string (such as `Unauthorized` or `Forbidden`) to be returned.

For example:

```
<property name="send-error_1"
   value="code=401 path=/myhost/401.html reason=MY-401-REASON"/>
```

This example property definition causes the contents of `/myhost/401.html` to be returned with 401 responses, along with this response line:

```
HTTP/1.1 401 MY-401-REASON
```

`redirect_`*n*

Specifies that a request for an old URL is treated as a request for a new URL. These properties are inherited by all web applications deployed on the virtual server. The value of each `redirect_`*n* property has two components, which may 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 simply replaced by this URL prefix.

For example:

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

`valve_`*n*

Specifies a fully qualified class name of a custom valve, where *n* is a positive integer that allows specification of more than one. The valve class must implement the `org.apache.catalina.Valve` interface from Tomcat or previous GlassFish Server releases, or the `org.glassfish.web.valve.GlassFishValve` interface from the current GlassFish Server release. For example:

```
<property name="valve_1"
   value="org.glassfish.extension.Valve"/>
```

You can set this property for a specific web application. For details, see “sun-web-app” in *Sun GlassFish Enterprise Server v3 Preview Application Deployment Guide*.

`listener_`*n*

Specifies a fully qualified class name of a custom Catalina listener, where *n* is a positive integer that allows specification of more than one. The listener class must implement the `org.apache.catalina.ContainerListener` or `org.apache.catalina.LifecycleListener` interface. For example:

```
<property name="listener_1"
   value="org.glassfish.extension.MyLifecycleListener"/>
```

You can set this property for a specific web application. For details, see “sun-web-app” in *Sun GlassFish Enterprise Server v3 Preview Application Deployment Guide*.

`docroot`
Absolute path to root document directory for server. Deprecated. Replaced with a `virtual-server` attribute, `docroot`, that is accessible using the `get`, `set`, and `list` subcommands.

`accesslog`
Absolute path to server access logs. Deprecated. Replaced with a `virtual-server` attribute , `access-log`, that is accessible using the `get`, `set`, and `list` subcommands.

`accessLoggingEnabled`
If `true`, access logging is enabled for this virtual server. Deprecated. Replaced with a `virtual-server` attribute, `access-logging-enabled`, that is accessible using the `get`, `set`, and `list` subcommands.

`sso-enabled`
If `true`, single sign-on is enabled for web applications on this virtual server that are configured for the same realm. Deprecated. Replaced with a `virtual-server` attribute, `sso-enabled`, that is accessible using the `get`, `set`, and `list` subcommands.

`ssoCookieSecure`
Sets the `Secure` attribute of any `JSESSIONIDSSO` cookies associated with the web applications deployed to this virtual server. Deprecated. Replaced with a `virtual-server` attribute, `sso-cookie-secure`, that is accessible using the `get`, `set`, and `list` subcommands.

`errorReportValve`
Specifies a fully qualified class name of a custom valve that produces default error pages for applications on this virtual server. Specify an empty string to disable the default error page mechanism for this virtual server.

`--target`
Creates the virtual server only on the specified target. Valid values are as follows:

`server`
Creates the virtual server on the default server instance. This is the default value.

configuration-name
Creates the virtual server in the specified configuration.

cluster-name
Creates the virtual server on all server instances in the specified cluster.

standalone-instance-name
Creates the virtual server on the specified standalone server instance.

Operands *virtual-server-id*
Identifies the unique ID for the virtual server to be created. This ID cannot begin with a number.

Examples

EXAMPLE 1 Creating a Virtual Server

The following command creates a virtual server named `sampleServer`:

```
asadmin> create-virtual-server --hosts pigeon,localhost
--property authRealm=ldap sampleServer
Command create-virtual-server executed successfully.
```

Exit Status

0 command executed successfully

1 error in executing the command

See Also

`delete-virtual-server`(1), `list-virtual-servers`(1), `create-http-listener`(1), `create-network-listener`(1)

`get`(1), `list`(1), `set`(1)

`asadmin`(1M)