Name

redeploy– redeploys the specified component

Synopsis

```
redeploy
[--help]
 --name  component-name [--upload={true|false}]
[ --retrieve  local-dirpath] [--dbvendorname dbvendorname]
[--createtables={true|false} |  --dropandcreatetables ={true|false}]
[ --uniquetablenames ={true|false}] [ --deploymentplan  deployment-plan]
[--enabled={true|false}]
[ --contextroot  context-root]
[--precompilejsp={true|false}]
[ --virtualservers  virtual-servers]
[--libraries jar-file[,jar-file]*]
[--type pkg-type]
[--properties(name=value)[:name=value]*][archive-path]
```

Description

The `redeploy` subcommand redeploys an enterprise application, web application, module based on the Enterprise JavaBeans$^{\text{TM}}$ (EJB$^{\text{TM}}$) specification (EJB module), connector module, or application client module that is already deployed or already exists. The `redeploy` subcommand preserves the settings and other options with which the application was originally deployed. The application must already be deployed. Otherwise, an error occurs.

This subcommand is supported in remote mode only.

Options

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

`--name`
The name of the application that is being redeployed. This option must specify an application that has previously been deployed. If the application does not exist, an error occurs.

`--upload`
When set to true (the default), uploads the deployable file to the administration server. The deployable file must be accessible from the client. If the file is accessible to both server and client, set the `--upload` option to false.

`--retrieve`
Retrieves the client stub JAR file from the server machine to the local directory.

`--dbvendorname`
Specifies the name of the database vendor for which tables are created. Supported values include `db2`, `mssql`, `oracle`, `derby`, `javadb`, `postgresql`, `pointbase`, and `sybase`, case-insensitive. If not specified, the value of the `database-vendor-name` attribute in `sun-ejb-jar.xml` is used. If no value is specified, a connection is made to the resource specified by the `jndi-name` subelement of the `cmp-resource` element in the `sun-ejb-jar.xml` file, and the database vendor name is read. If the connection cannot be established, or if the value is not recognized, SQL-92 compliance is presumed.

`--createtables`
Creates tables at deployment of an application with unmapped container-managed persistence (CMP) beans. Default is the `create-tables-at-deploy` entry in the `cmp-resource` element of the `sun-ejb-jar.xml` file.

`--dropandcreatetables`
If set to true, when the component is redeployed, the tables created by the previous deployment are dropped before creating the new tables. Applies to already deployed applications with unmapped CMP beans. If not set to true, the tables are dropped if the `drop-tables-at-undeploy` entry in the `cmp-resource` element of the `sun-ejb-jar.xml` file is set to true. The new tables are created if the `create-tables-at-deploy` entry in the `cmp-resource` element of the `sun-ejb-jar.xml` file is set to true.

`--uniquetablenames`
Guarantees unique table names for all the beans and results in a hashcode added to the table names. This is useful if you have an application with case-sensitive bean names.

`--deploymentplan`
Takes the deployment plan, which is a JAR containing Sun-specific descriptors, and deploys it. This should be passed along when deploying a pure EAR file. A pure EAR file is an EAR without Sun-specific descriptors.

`--enabled`
If set to true (default), allows users to access the application. If set to false, users will not be able to access the application.

`--contextroot`
Valid only if the archive is a web module. It is ignored for other archive types; defaults to filename without extension. The context root of the application that is being redeployed. The context root identifies the application in the server. The default is the name, without the extension, of the archive file that contains the application.

`--precompilejsp`
Specifies whether pages that are created with the JavaServer Pages™ technology ("JSP™ pages") are precomplied during deployment. Possible values are as follows:

`false`
JSP pages are *not* precompiled during deployment, but are complied at runtime (default).

`true`
JSP pages are precompiled during deployment.

`--virtualservers`
One or more virtual server IDs. Multiple IDs are separated by commas.

`--libraries`
A comma-separated list of library JAR files. Specify the library JAR files by their relative or absolute paths. Specify relative paths relative to *instance-root*`/lib/applibs`. The libraries are made available to the application in the order specified.

`--type`

The packaging archive type of the component that is being deployed. Possible values are as follows:

`osgi`

The component is packaged as an OSGi Alliance bundle.

The `--type` option is optional. If the component is packaged as a regular archive, omit this option.

`--properties`

`--property`

Optional keyword-value pairs that specify additional properties for the deployment. The available properties are determined by the implementation of the component that is being deployed. The `--properties` option and the `--property` option are equivalent. You can use either option regardless of the number of properties that you specify. You can specify the following properties for a deployment:

`default-EE6—app-name`

The default Java EE 6 name of the Java EE application (EAR file). The default Java EE 6 name is not always the same as the name attribute. According to the Java EE 6 specification, the default Java EE 6 name is the archive name minus the suffix.

`java-web-start-enabled`

Specifies whether Java Web Start access is permitted for an application client module. Default is true.

`class-name`

The fully qualified name of a lifecycle module class file. A lifecycle module class must implement the `com.sun.appserv.server.LifecycleListener` interface.

`classpath`

The classpath for a lifecycle module. Specifies where the module is located. Default is the value of `application-root` attribute of the `domain` element.

`load-order`

Determines the order in which lifecycle modules are loaded at startup. Modules with smaller integer values are loaded sooner. Values can range from 101 to the operating system's `MAXINT`. Values from 1 to 100 are reserved.

`is-failure-fatal`

Determines whether the server is shut down if a lifecycle module fails. Default is false.

`jruby-home`

Specifies the directory where the JRuby container is installed. Overrides the `jruby-home` attribute of the JRuby container. Default is *as-install*`/jruby`.

`jruby-runtime`
Specifies the initial number of JRuby runtimes to start. Must be at greater than zero, at least `jruby.runtime.min`, and `jruby.runtime.max` or less. Overrides the `jruby-runtime` attribute of `jruby-runtime-pool`. Default is 1.

`jruby-runtime-min`
Specifies the minimum number of JRuby runtimes in the pool. Must be greater than zero, `jruby.runtime` or less, and `jruby.runtime.max` or less. Overrides the `jruby-runtime-min` attribute of `jruby-runtime-pool`. Default is 1.

`jruby-runtime-max`
Specifies the maximum number of JRuby runtimes in the pool. Must be greater than zero, at least `jruby.runtime.min`, and at least `jruby.runtime`. Overrides the `jruby-runtime-max` attribute of `jruby-runtime-pool`. Default is 1.

`jruby-rackEnv`
Specifies the environment in which a JRuby application such as Rails or Merb runs. Allowed values are development, production, or test. Default is development.

`jruby-applicationtype`
Specifies the name of a supported framework or the path to a script that initializes the user's framework. Allowed values corresponding to supported frameworks are Rails, Merb, or Sinatra. Setting this property bypasses the normal, and potentially lengthy, auto-detection process and forces deployment on the specified framework. If the deployed application is not written for the specified framework, errors result. Default is computed through auto-detection.

`jruby-MTSafe`
If true, specifies that a framework being started using `jruby.applicationType` is thread-safe and therefore does not need a pool created for it. This property affects applications started using an auto-detected user-provided startup script. If `jruby.applicationType` is set and `jruby.MTsafe` is not set or is set to false, the application starts with a pool of application instances, and each instance of the application is accessed by one thread at a time. This property only affects frameworks being launched where the thread safety cannot be automatically determined. Setting `jruby.MTsafe` to true does not cause an auto-detected Rails 2.1.x application to be launched in thread-safe mode, nor can it be used to force a thread-safe framework to start in pooled mode. Default is computed through auto-detection.

`compatibility`
Specifies the Enterprise Server release with which to be backward compatible in terms of JAR visibility requirements for application clients. The only allowed value is `v2`, which refers to GlassFish version 2 or Enterprise Server version 9.1 or 9.1.1. The Java EE 6 platform specification imposes stricter requirements than Java EE 5 did on which JAR files can be visible to various modules within an EAR file. In particular, application clients must not have access to EJB JAR files or other JAR files in the EAR file unless

references use the standard Java SE mechanisms (extensions, for example) or the Java EE library-directory mechanism. Setting this property to `v2` removes these Java EE 6 restrictions.

`keepSessions={false|true}`

Specifies whether active sessions of the application that is being redeployed are preserved and then restored when the redeployment is complete. Possible values are as follows:

`false`

Active sessions of the application are *not* preserved and restored (default).

`true`

Active sessions of the application are preserved and restored.

If any active session of the application fails to be preserved or restored, *none* of the sessions will be available when the redeployment is complete. However, the redeployment continues and a warning is logged.

To preserve active sessions, the Enterprise Server serializes the sessions and saves them in memory. To restore the sessions, the class loader of the newly redeployed application deserializes any sessions that were previously saved.

Other available properties are determined by the implementation of the component that is being redeployed.

Note – The `--properties` option and the `--property` option are equivalent. You can use either option regardless of the number of properties that you specify.

Operands

archive-path

The path to the archive that contains the application that is being redeployed. This path can be a relative path or an absolute path.

The archive can be in either of the following formats:

- An archive file, for example, `/export/JEE_apps/hello.war`
- A directory that contains the exploded format of the deployable archive

Whether this operand is required depends on how the application was originally deployed:

- If the application was originally deployed from a file, the *archive-path* operand is required. The operand must specify an archive file.
- If the application was originally deployed from a directory, the *archive-path* operand is optional.

  If this operand is omitted, the path is retrieved from the `domain.xml` file. Otherwise, the operand can specify a directory or an archive file.

Examples

EXAMPLE 1 Redeploying a Web Application From a File

This example redeploys the web application `hello` from the `hello.war` file in the current working directory. The application was originally deployed from a file. Active sessions of the application are to be preserved and then restored when the redeployment is complete.

```
asadmin> redeploy --name hello --properties keepSessions=true hello.war
Command redeploy executed successfully
```

EXAMPLE 2 Redeploying a Web Application From a Directory

This example redeploys the web application `hellodir`. The application was originally deployed from a directory. The path is retrieved from the `domain.xml` file.

```
asadmin> redeploy --name hellodir
Command redeploy executed successfully
```

Exit Status

0	subcommand executed successfully
1	error in executing the subcommand

See Also

`deploy`(1), `undeploy`(1), `list-components`(1)

`asadmin`(1M)

Sun GlassFish Enterprise Server v3 Preview Application Deployment Guide