Name

deploydir– deploys an exploded format of application archive

Synopsis

```
deploydir
[--help]
[ --virtualservers  virtual-servers]
[--contextroot context-root] [--force={false|true}]
[ --verify ={false|true}][ --precompilejsp ={false|true}]
[ --name  component-name] [--uniquetablenames={true|false}]
[ --dbvendorname  dbvendorname]
[--createtables={false|true}| --dropandcreatetables ={false|true}]
[ --generatermistubs ={false|true}] [ --availabilityenabled ={false|true}]
[ --libraries  jar-file[,jar-file]*] [--target target]
[--type pkg-type]
[--properties(name=value)[:name=value]*]
dirpath
```

Description

Note – The `deploydir` subcommand is deprecated. Use the `deploy(1)` subcommand instead.

The `deploydir` subcommand deploys an application directly from a development directory. The appropriate directory hierarchy and deployment descriptors conforming to the Java™ EE specification must exist in the deployment directory.

Directory deployment is for advanced developers only. Do not use it in production environments. In production environments, use the `deploy` subcommand. Directory deployment is only supported on localhost, that is, the client and server must reside on the same machine. For this reason, the only values for the `--host` option are:

- `localhost`
- The value of the `$HOSTNAME` environment variable
- The IP address of the machine

The `--force` option makes sure the component is forcefully (re)deployed even if the specified component has already been deployed or already exists. Set the `--force` option to false for an initial deployment. If the specified application is running and the `--force` option is set to false, the subcommand fails.

This subcommand is supported in remote mode only.

Options

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

`--virtualservers`
Comma-separated list of virtual server IDs.

`--contextroot`
Valid only if the archive is a web module. It is ignored for other archive types; defaults to filename without extension.

`--force`
Makes sure the component is forcefully (re)deployed even if the specified component has already been deployed or already exists. The default is `false`.

`--verify`
Do not specify this option. This option is retained for compatibility with other releases. If you specify this option, a syntax error does not occur. Instead, the subcommand runs successfully and the option is silently ignored.

`--precompilejsp`
By default, this option is set to false, which does not allow the JSP to precompile during deployment. Instead, JSPs are compiled during runtime.

`--name`
Name of the deployable component.

`--uniquetablenames`
Do not specify this option. This option is retained for compatibility with other releases. If you specify this option, a syntax error does not occur. Instead, the subcommand runs successfully and the option is silently ignored.

`--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 CMP beans. Default is the `create-tables-at-deploy` entry in the `cmp-resource` element of the `sun-ejb-jar.xml` file.

`--dropandcreatetables`
Drops existing tables and creates tables during deployment for application using unmapped CMP beans. If not specified, 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` is set to true. When the component is redeployed, the tables created by the previous deployment are dropped before creating the new tables.

`--generatermistubs`
Do not specify this option. This option is retained for compatibility with other releases. If you specify this option, a syntax error does not occur. Instead, the subcommand runs successfully and the option is silently ignored.

`--availabilityenabled`

Do not specify this option. This option is retained for compatibility with other releases. If you specify this option, a syntax error does not occur. Instead, the subcommand runs successfully and the option is silently ignored.

`--libraries`

Specify the library JAR files by their relative or absolute paths. Specify relative paths relative to *instance-root*`/lib/applibs`. The JAR files are separated by a colon on UNIX and Linux systems and by a semicolon on Windows systems. The libraries are made available to the application in the order specified. Place the dependent JAR files in the *domain-dir*`/lib` directory.

`--target`

Do not specify this option. This option is retained for compatibility with other releases. If you specify this option, a syntax error does not occur. Instead, the subcommand runs successfully and the option is silently ignored.

`--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.

Operands

dirpath

Path to the directory containing the exploded format of the deployable archive.

Examples

EXAMPLE 1 Deploying an Application From a Directory

In this example, the exploded application to be deployed is in the `/home/temp/sampleApp` directory. Because the `--force` option is set to true, if an application of that name already exists, the application is redeployed.

```
asadmin> deploydir --force=true --precompilejsp=true /home/temp/sampleApp
Command deploydir executed successfully
```

Exit Status

0	subcommand executed successfully
1	error in executing the subcommand

See Also

`deploy`(1), , `redeploy`(1), `undeploy`(1)

`asadmin`(1M)