

Name dotted-names– syntax for using periods to separate name elements

Description A *dotted name* is an identifier for a particular Enterprise Server element, such as a configurable or a monitorable object. A dotted name uses the period (.), known dot, as a delimiter to separate the parts of an element name. The period in a dotted name is similar to the slash (/) character that delimits the levels in the absolute path name of a file in the UNIX file system.

The subcommands of the `asadmin` utility use dotted names as follows:

- The `list` subcommand provides the fully qualified dotted names of the management components' attributes.
- The `get` subcommand provides access to the attributes.
- The `set` subcommand enables you to modify configurable attributes.

The configuration hierarchy is loosely based on the domain's schema document, and the attributes are modifiable. The attributes of the monitoring hierarchy are read-only.

The following format is used for configuration dotted names (*italic indicates replaceable*):

```
config-name.config-element-name.primary-key.attribute-name |  
instance-name.config-element-name.primary-key.attribute-name
```

The following format is used for resource dotted names (*italic indicates replaceable*):

```
server-name.resource-name.primary-key.attribute-name |  
domain.resources.resource-name.primary-key.attribute-name
```

The following rules apply to forming dotted names:

- The top-level is configuration, server, or domain name. For example, `server-config` (default configuration), `server` (default server), or `domain1` (default domain).
- A dot (.) always separates two sequential parts of the name.
- A part of the name usually identifies a server subsystem or its specific instance. For example, `web-container`, `log-service`, `thread-pool-1`.
- If any part of the name itself contains a dot (.), then the dot must be escaped with a leading \ (backslash) so that the . (dot) does not act like a delimiter.
- An * (asterisk) character can be used anywhere in the dotted name and acts like the wildcard character in regular expressions. Additionally, an * can collapse all the parts of the dotted name. For example, a long dotted name such as `this.is.really.long.hierarchy` can be abbreviated to `th*.hierarchy`. The . (dot) always delimits the parts of the dotted name.

Note – On Solaris™ systems, double quotes are required when running subcommands that use * (asterisk) as the option value or operand.

- The `--monitor` option of the `get` and `list` subcommands selects the monitoring or configuration hierarchy. If the subcommand specifies `--monitor=false` (the default), the configuration hierarchy is selected. If the subcommand specifies `--monitor=true`, the monitoring hierarchy is selected.
- If you know the *complete dotted name* and do not need to use a wildcard, the `list`, `get`, and `set` subcommands treat the name differently:
 - The `list` subcommand treats a complete dotted name as the name of a parent node in the abstract hierarchy. When you specify this name to the `list` subcommand, the names of the immediate children at that level are returned. For example, the following command lists all the web modules deployed to the domain or the default server:

```
asadmin> list server.applications.web-module
```

- The `get` and `set` subcommands treat a complete dotted name as the fully qualified name of the attribute of a node (whose dotted name itself is the name that you get when you remove the last part of this dotted name). When you specify this name to the `get` or `set` subcommand, the subcommand acts on the value of that attribute, if such an attribute exists. You will never start with this case because in order to find out the names of attributes of a particular node in the hierarchy, you must use the `*` wildcard character. For example, the following dotted name returns the context root of the web application deployed to the domain or default server:

```
server.applications.web-module.JSPWiki.context-root
```

Examples **EXAMPLE 1** Listing All Configurable Elements

This example lists all the configurable elements. (This example of the Solaris operating system. On other operating systems, the double quotes would be omitted.)

```
asadmin> list ""
```

Output similar to the following is displayed:

```
applications
configs
configs.config.server-config
configs.config.server-config.admin-service
configs.config.server-config.admin-service.das-config
configs.config.server-config.admin-service.jmx-connector.system
configs.config.server-config.admin-service.property.adminConsoleContextRoot
configs.config.server-config.admin-service.property.adminConsoleDownloadLocation
configs.config.server-config.admin-service.property.ipsRoot
configs.config.server-config.ejb-container
configs.config.server-config.ejb-container.ejb-timer-service
configs.config.server-config.http-service
configs.config.server-config.http-service.access-log
configs.config.server-config.http-service.virtual-server.__asadmin
configs.config.server-config.http-service.virtual-server.server
configs.config.server-config.iiop-service
```

EXAMPLE 1 Listing All Configurable Elements *(Continued)*

```

configs.config.server-config.iiop-service.iiop-listener.SSL
configs.config.server-config.iiop-service.iiop-listener.SSL.ssl
configs.config.server-config.iiop-service.iiop-listener.SSL_MUTUALAUTH
configs.config.server-config.iiop-service.iiop-listener.SSL_MUTUALAUTH.ssl
configs.config.server-config.iiop-service.iiop-listener.orb-listener-1
configs.config.server-config.iiop-service.orb
configs.config.server-config.java-config
configs.config.server-config.jms-service
configs.config.server-config.jms-service.jms-host.default_JMS_host
configs.config.server-config.mdb-container
configs.config.server-config.monitoring-service
configs.config.server-config.monitoring-service.module-monitoring-levels
...
property.administrative.domain.name
resources
resources.jdbc-connection-pool.DerbyPool
resources.jdbc-connection-pool.DerbyPool.property.DatabaseName
resources.jdbc-connection-pool.DerbyPool.property.Password
resources.jdbc-connection-pool.DerbyPool.property.PortNumber
resources.jdbc-connection-pool.DerbyPool.property.User
resources.jdbc-connection-pool.DerbyPool.property.connectionAttributes
resources.jdbc-connection-pool.DerbyPool.property.serverName
resources.jdbc-connection-pool.__TimerPool
resources.jdbc-connection-pool.__TimerPool.property.connectionAttributes
resources.jdbc-connection-pool.__TimerPool.property.databaseName
resources.jdbc-resource.jdbc/__TimerPool
resources.jdbc-resource.jdbc/__default
servers
servers.server.server
servers.server.server.resource-ref.jdbc/__TimerPool
servers.server.server.resource-ref.jdbc/__default
system-applications

```

Command list executed successfully.

EXAMPLE 2 Listing All the Monitorable Objects

The following example lists all the monitorable objects.

```
asadmin> list --monitor=true *
```

Output similar to the following is displayed:

```

server
server.jvm
server.jvm.class-loading-system

```

EXAMPLE 2 Listing All the Monitorable Objects *(Continued)*

```
server.jvm.compilation-system
server.jvm.garbage-collectors
server.jvm.garbage-collectors.Copy
server.jvm.garbage-collectors.MarkSweepCompact
server.jvm.memory
server.jvm.operating-system
server.jvm.runtime
server.network
server.network.admin-listener
server.network.admin-listener.connections
server.network.admin-listener.file-cache
server.network.admin-listener.keep-alive
server.network.admin-listener.thread-pool
server.network.http-listener-1
server.network.http-listener-1.connections
server.network.http-listener-1.file-cache
server.network.http-listener-1.keep-alive
server.network.http-listener-1.thread-pool
server.transaction-service
```

Command list executed successfully.

See Also [list\(1\)](#), [get\(1\)](#), [set\(1\)](#)

[asadmin\(1m\)](#)

[8 Administering the Monitoring Service, Configuration Tasks](#)

