Red Hat JBoss Fuse 6.3

Console Reference

Quick access to the Apache Karaf shell commands packaged under JBoss Fuse.

Last Updated: 2018-07-16

Red Hat JBoss Fuse 6.3 Console Reference

Quick access to the Apache Karaf shell commands packaged under JBoss Fuse.

JBoss A-MQ Docs Team
Content Services
fuse-docs-support@redhat.com

Legal Notice

Abstract

The primary interface for managing a container is the command console. This reference provides an easy way to see the commands, their syntax, and options.

Table of Contents

CHAPTER 1. USING THE COMMAND CONSOLE

OVERVIEW

The Red Hat JBoss Fuse command console is the central tool for both managing the JBoss Fuse environment and interacting with Fuse Fabric. When you start JBoss Fuse the console starts automatically.

The console provides commands that you can use to perform basic management of your JBoss Fuse environment, including deploying and configuring applications.

The console uses prefixes to group commands relating to the same functionality. For example, commands related to configuration are prefixed **`config:`**, and logging-related commands are prefixed **`log:`**.

STARTING THE COMMAND CONSOLE

To start JBoss Fuse open a command prompt in the installation directory and enter:

Windows	**`bin\fuse`**
***NIX**	**`bin/fuse`**

JBoss Fuse starts and the console is ready. You should see a prompt similar to this:

```
 _ ____ _______ | | _ \ | ____| | | |_) | ___ ___ ___ | |__ _ _ ___ ___ _ |
| _ < / _ \/ __/ __| | __| | | / __|/ _ \ | |__| | |_) | (_) \__ \__ \ | |
| |_| \__ \ __/ \____/|____/ \___/|___/___/ |_| \__,_|___/\___| JBoss Fuse
(6.0.0.redhat-xxx)
http://www.redhat.com/products/jbossenterprisemiddleware/fuse/ Hit '<tab>'
for a list of available commands and '[cmd] --help' for help on a specific
command. Hit '<ctrl-d>' or 'osgi:shutdown' to shutdown JBoss Fuse.
JBossFuse:karaf@root
```

GETTING HELP

The console provides two levels of help:

- console help—lists all of the commands along with a brief summary of the commands function
- command help—provides a detailed description of a command and its arguments

To access the console help, enter the **`help`** command from the console prompt. It displays a grouped list of all the commands available in the console. Each command in the list is followed by a description, as shown in Example 1.1, “Console Help”.

Example 1.1. Console Help

```
JBossFuse:karaf@root> help
COMMANDS activemq:browse activemq:bstat activemq:create-broker Creates a
broker instance. activemq:destroy-broker Destory a broker instance.
```

```
activemq:list activemq:purge activemq:query admin:change-opts Changes
the Java options of an existing container instance. admin:change-rmi-
registry-port Changes the RMI registry port (used by management layer)
of an existing container instance.
    ...
JBossFuse:karaf@root>
```

The help for each command includes the definition, the syntax, and the arguments and any options. To display the help for a command, type the command with the **--help** option. As shown in Example 1.2, "Help for a Command", entering **admin:start --help** displays the help for that command.

Example 1.2. Help for a Command

```
JBossFuse:karaf@root> admin:start --help
DESCRIPTION admin:start Starts an existing container instance. SYNTAX
admin:start [options] name ARGUMENTS name The name of the container
instance OPTIONS --help Display this help message -o, --java-opts Java
options when launching the instance
JBossFuse:karaf@root>
```

COMMAND COMPLETION

Pressing **Tab** at anytime provides you with a list of commands that can complete what you have already entered at the prompt. For example, if you enter **active** followed by **Tab**, a list similar to Example 1.3, "Console Commands" is shown.

Example 1.3. Console Commands

```
activemq:browse activemq:bstat activemq:create-broker activemq:destroy-
broker activemq:list activemq:purge activemq:query JBossA-MQ:karaf@root>
```

If you press **Tab** without entering anything at the prompt, the console lists all of the available commands.

COMMAND GROUPS

Commands are grouped under prefixes according to functionality. Table 1.1, "Apache ActiveMQ Command Groups" summarizes the command groups available in the console. Click on a command group name for more information.

Table 1.1. Apache ActiveMQ Command Groups

Command Group	Description
activemq	Views and manages brokers and messages.
admin	Creates, manages, and destroys containers.

Command Group	Description
camel	Manages Apache Camel contexts and routes
config	Manages configuration.
cxf	Manages Apache CXF buses and endpoints.
dev	Utilities that are useful for a developer while testing bundles in the container.
fab	Manages the dependency resolution mechanism used by Fuse Application Bundles.
fabric	Performs provisioning and configuration using Fuse Fabric.
features	Performs provisioning based on Apache Karaf feature specs.
jaas	Manages the console's security settings.
jbi	Manage deployed JBI artifacts.
log	Displays and configures logging.
nmr	Lists NMR endpoints.
obr	Accesses the OSGi Bundle Repository (OBR).
osgi	Manages OSGi bundles.
packages	Lists imported and exported packages.
patch	Manages patches.
shell	Performs basic console functions
ssh	Creates and connects to a remote SSH server
web	Lists the WARs deployed in the container.
zk	Accesses and modifies entries in the Zookeeper registry.

SHORT VERSION

Many of the console commands allow you to omit the group prefix.

If the command is only in one command groups, you can omit the group prefix. For example, you can enter **`bstat`** in place of **`activemq:bstat`** because it only exists in the activemq command group.

If the command exists in multiple command groups, you can still drop the prefix and the console will default to using the version of the command from one of the following command groups:

- shell
- osgi
- admin

For example, **`info`** is equivalent to **`shell:info`**. If you wanted to use **`osgi:info`**, you need to enter the full command.

ABOUT CONSOLE INPUT

In console input, the Karaf shell drops leading zeros when the input appears to be a number. For example:

```
FuseMQ:karaf@root> echo 0123
123
FuseMQ:karaf@root> echo 00.123
0.123
FuseMQ:karaf@root>
```

This is a problem if you define a numeric username with a leading zero. The shell drops the leading zero and login attempts fail. To avoid this, do not use a zero as the first character in a numeric username. Alternatively, include at least one alphabetic character in a username.

CHAPTER 2. SHELL CONSOLE COMMANDS

The shell command group provides a number of commands that provide basic console functions such as displaying system information and showing the contents of files.

Type **shell:** then press **Tab** at the prompt to view the commands in this group.

2.1. SHELL:CAT, CAT

Abstract

displays the contents of a file or URL

Synopsis

shell:cat [-n] [--help] {[*path*] | [*URL*]}

Arguments

Table 2.1, "**shell:cat** Arguments" describes the arguments for this command.

Table 2.1. shell:cat Arguments

Argument	Interpretation
-n	Display line numbers.
--help	Displays the online help for this command
path	The path(s) of the file to display, separated by whitespace (separated by **-** for STDIN)
URL	The URL(s) to display, separated by whitespace (separated by **-** for STDIN)

2.2. SHELL:CLEAR, CLEAR

Abstract

clears the console buffer

Synopsis

shell:clear [--help]

Arguments

Table 2.2, "**shell:clear** Arguments" describes the command's arguments.

Table 2.2. shell:clear Arguments

Argument	Interpretation
--help	Displays the online help for this command

2.3. SHELL:EACH, EACH

Abstract

execute a closure on a list of arguments

Synopsis

shell:each [--help] { *values* } { *function* }

Arguments

Table 2.3, "**shell:each** Arguments" describes the command's arguments.

Table 2.3. shell:each Arguments

Argument	Interpretation
--help	Displays the online help for this command
values	The collection of arguments to iterate over.
function	The function to execute.

2.4. SHELL:ECHO, ECHO

Abstract

prints arguments to the standard output

Synopsis

shell:echo [--help] [-n] { argument ...}

Arguments

Table 2.4, "**shell:echo** Arguments" describes the command's arguments.

Table 2.4. shell:echo Arguments

Argument	Interpretation
`--help`	Displays the online help for this command.
`-n`	Do not print the trailing newline character.
argument	Specifies a space delimited list of arguments to print.

2.5. SHELL:EXEC, EXEC

Abstract

Executes system processes. Do not use this command with interactive processes. For example, do not invoke: `exec more /path/to/fuse.log`. Doing this can lead to performance degradation.

Synopsis

shell:exec [--help] { *command* }

Arguments

Table 2.5, "**shell:exec** Arguments" describes the command's arguments.

Table 2.5. shell:exec Arguments

Argument	Interpretation
`--help`	Displays the online help for this command
command	Specifies the command, with arguments, to execute.

2.6. SHELL:GREP, GREP

Abstract

displays lines matching a regular expression

Synopsis

shell:grep [--help] [[-i] | [--ignore-case]] [[-w] | [--word-regexp]] [[-n] | [--line-number]] [[-x] | [--line-regexp]] [[-v] | [--invert-match]] { *regex* }

Arguments

Table 2.6, "**shell:grep** Arguments" describes the command's arguments.

Table 2.6. shell:grep Arguments

Argument	Interpretation
--help	Displays the online help for this command.
-i, --ignore-case	Ignore case distinctions in both the *regex* and the input files.
-w, --word-regexp	Select only lines containing matches that form whole words. A match qualifies if it meets one of the following conditions: • The matching string is at the beginning of the line. • The matching string is preceded by a non-word constituent character. • The matching string is at the end of the line. • The matching string is followed by a non-word constituent character.
-n, --line-number	Display the line number of the match within its input file.
-x, --line-regexp	Selects only those matches that exactly match the whole line.
-v, --invert-match	Select non-matching lines.
regex	Specifies the regular expression to match.

2.7. SHELL:HEAD, HEAD

Abstract

displays the first lines of a file

Synopsis

shell:head [--help] [-n *numLines*] {[*path*] | [*URL*]}

Arguments

Table 2.7, "**shell:head** Arguments" describes the command's arguments.

Table 2.7. shell:head Arguments

Argument	Interpretation
`--help`	Displays the online help for this command.
`-n`	Specifies the number of lines to display. Default is 1.
`path`	The path(s) of the file to display, separated by whitespace (separated by `-` for STDIN)
`URL`	Specifies the URL(s) to display, separated by whitespace (separated by `-` for STDIN)

2.8. SHELL:HISTORY, HISTORY

Abstract

prints the command history

Synopsis

`shell:history` [--help]

Arguments

Table 2.8, "**`shell:history`** Arguments" describes the arguments for this command.

Table 2.8. `shell:history` Arguments

Argument	Interpretation
`--help`	Displays the online help for this command

2.9. SHELL:IF, IF

Abstract

executes an if/then/else block

Synopsis

`shell:if` [--help] { *condition* } { *ifTrue* } [*ifFalse*]

Arguments

Table 2.9, "**`shell:if`** Arguments" describes the command's arguments.

Table 2.9. `shell:if` Arguments

Argument	Interpretation
--help	Displays the online help for this command
condition	Boolean condition.
ifTrue	Function to evaluate, if condition is true.
ifFalse	Function to evaluate, if condition is false.

2.10. SHELL:INFO, INFO

Abstract

displays system information and statistics about the container

Synopsis

shell:info [--help]

Arguments

Table 2.10, "**shell:info** Arguments" describes the command's arguments.

Table 2.10. shell:info Arguments

Argument	Interpretation
--help	Displays the online help for this utility

2.11. SHELL:JAVA, JAVA

Abstract

execute a Java application

Synopsis

shell:java [--help] [[-m] | [--method] ***methodName***] { *className* } [*arguments*]

Arguments

Table 2.11, "**shell:java** Arguments" describes the command's arguments.

Table 2.11. shell:java Arguments

Argument	Interpretation
`--help`	Displays the online help for this command
`-m`, `--method`	Specifies the name of a method to invoke. The default is `main()`.
`className`	Specifies the name of the class to invoke.
`arguments`	Specifies the arguments to pass to the method of the given *className*.

2.12. SHELL:LOGOUT, LOGOUT

Abstract

disconnects the shell from the current session

Synopsis

`shell:logout` [--help]

Arguments

Table 2.12, "**`shell:logout`** Arguments" describes the command's arguments.

Table 2.12. `shell:logout` Arguments

Argument	Interpretation
`--help`	Displays the online help for this command

2.13. SHELL:MORE, MORE

Abstract

displays output as pages of a specified length

Synopsis

`shell:more` [--help] [--lines *numLines*]

Arguments

Table 2.13, "**`shell:more`** Arguments" describes the command's arguments.

Table 2.13. `shell:more` Arguments

Argument	Interpretation
`--help`	Displays the online help for this command
`--lines`	Specifies the number of lines to display before pausing.

2.14. SHELL:NEW, NEW

Abstract

creates a new Java object of the specified class

Synopsis

`shell:new` [--help] { *class* } [*arg* ...]

Arguments

Table 2.14, "**`shell:new` Arguments**" describes the command's arguments.

Table 2.14. `shell:new` Arguments

Argument	Interpretation
`--help`	Displays the online help for this command
class	The class of the object to create.
args	The constructor arguments.

2.15. SHELL:PRINTF, PRINTF

Abstract

formats and prints the specified output

Synopsis

`shell:printf` [--help] { *format* } { *arguments* }

Arguments

Table 2.15, "**`shell:printf` Arguments**" describes the command's arguments.

Table 2.15. `shell:printf` Arguments

Argument	Interpretation
--help	Displays the online help for this command
format	The output format pattern to use
arguments	The arguments for the given format pattern

2.16. SHELL:SLEEP, SLEEP

Abstract

sleeps for a specified time, then wakes up

Synopsis

shell:sleep [--help] [[-s] | [--second]] { *duration* }

Arguments

Table 2.16, "**shell:sleep** Arguments" describes the command's arguments.

Table 2.16. shell:sleep Arguments

Argument	Interpretation
--help	Displays the online help for this command
-s, --second	Specify the duration in seconds (instead of milliseconds).
duration	The time to sleep in milliseconds (default) or in seconds (with the **-s** option).

2.17. SHELL:SORT, SORT

Abstract

writes a sorted concatenation of the specified files to standard output

Synopsis

shell:sort [--help] [[-t] | [--field-separator] ***sep***] [[-b] | [--ignore-leading-blanks]] [[-f] | [--ignore-case]] [[-r] | [--reverse]] [[-k] | [--key] ***keys***] [[-n] | [--numeric-sort]] [[-u] | [--unique]] { *file* ...}

Arguments

Table 2.17, "`shell:sort` Arguments" describes the command's arguments.

Table 2.17. `shell:sort` Arguments

Argument	Interpretation
`--help`	Displays the online help for this command
`-t`, **`--field-separator`**	Specifies a character to use as a field separator. The default is whitespace.
`-b`, **`--igonore-leading-blanks`**	Igonores leading blanks.
`-f`, **`--ignore-case`**	Ignores case when sorting.
`-r`, **`--reverse`**	Reverses the result of the sort.
`-k`, **`--key`**	Specifies a space delimited list of fields to use for sorting.
`-n`, **`--numeric-set`**	Compares according to string numerical value.
`-u`, **`--unique`**	Outputs only the first of an equal run.
`files`	Specifies a space delimited list of files to sort.

2.18. SHELL:SOURCE, SOURCE

Abstract

run a shell script

Synopsis

`shell:source` [--help] { *script* } [*arguments*]

Arguments

Table 2.18, "`shell:source` Arguments" describes the command's arguments.

Table 2.18. `shell:source` Arguments

Argument	Interpretation
`--help`	Displays the online help for this command

Argument	Interpretation
script	A URI pointing to the script
arguments	Arguments to pass to the script

2.19. SHELL:TAC, TAC

Abstract

captures the STDIN and returns it as a string and optionally writes the content to a file

Synopsis

shell:tac [--help] [-f *fileName*]

Arguments

Table 2.19, "**shell:tac** Arguments" describes the command's arguments.

Table 2.19. shell:tac Arguments

Option	Interpretation
--help	Displays the online help for this command
-f	Specifies the name of the file into which the output is written.

2.20. SHELL:TAIL, TAIL

Abstract

displays the last lines of a file

Synopsis

shell:head [--help] [-n *lineNum*] [-s *seconds*] [-f] {[*path*] | [*URL*]...}

Arguments

Table 2.20, "**shell:tail** Arguments" describes the command's arguments.

Table 2.20. shell:tail Arguments

Argument	Interpretation
`--help`	Displays the online help for this command.
`-n`	Specifies the number of lines to display. The default is 1.
`-s`	Specifies the interval, in seconds, to sleep before checking for changes to display.
`-f`	Follow file changes.
`path`	A space delimited list of file paths to display.
`URL`	A space delimited list of file URLs to display.

2.21. SHELL:WATCH, WATCH

Abstract

watches and refreshes the output of a command

Synopsis

`shell:watch` [--help] [[-n] | [--interval] ***`seconds`***] { *command* }

Arguments

Table 2.21, "**`shell:watch`** Arguments" describes the command's arguments.

Table 2.21. `shell:watch` Arguments

Argument	Interpretation
`--help`	Displays the online help for this command
`-n`,`--interval`	Specifies the interval, in seconds, between executions of the command. The default is 1.
`command`	Specifies the command to watch and refresh.

CHAPTER 3. ACTIVEMQ CONSOLE COMMANDS

The **activemq** commands allow you to view and manage the brokers and messages.

Type **activemq:** then press **Tab** at the prompt to view the available commands.

3.1. ACTIVEMQ:BROWSE, BROWSE

Abstract

displays messages on a specified destination

Synopsis

activemq:browse { --amqurl *brokerURL* } [--msgsel { msgsel ...}] [--factory *className*] [--passwordFactory *className*] [--user *username*] [--password *password*] [--view { *attr* ...}] [[-Vheader] | [-Vcustom] | [-Vbody]] [--version] [[--help] | [-h] | [-?]] *destName*

Arguments

Table 3.1, "**activemq:browse** Arguments" describes the command's arguments.

Table 3.1. activemq:browse Arguments

Argument	Interpretation
--amqurl *brokerURL*	Specifies the URL of the broker to which you are connecting.
--msgsel *msgsel1,msgsel2,...*	Displays messages matched by the message selector.
--factory *className*	Load className as the javax.jms.ConnectionFactory to use for creating connections.
--passwordFactory *className*	Load className as the **org.apache.activemq.console.command.PasswordFactory** for retrieving the password from a keystore.
--user *username*	Username to use for JMS connections.
--password *password*	Password to use for JMS connections.
-Vheader	Shows all the standard JMS message headers.
-Vcustom	Shows all the custom fields added to each JMS message.

Argument	Interpretation
-Vbody	Shows the body of the message.
--view *attr1,attr1,...*	Selects the specific attribute of the message to view.
--version	Displays the version information.
-h, -?, --help	Displays the online help for this command.

Message filters

Message filters specified using the **--msgsel** option take the form ***header=value***. Table 3.2, "Message Headers for Filtering" lists the headers you can use to filter messages.

Table 3.2. Message Headers for Filtering

Name	Type
JMSCorrelationID	**String**
JMSDeliveryMode	**1**-Non-Persistent, **2**-Persistent
JMSDestination	**javax.jms.Destination**
JMSExpiration	**long**
JMSMessageID	**String**
JMSPriority	**int**
JMSRedelivered	**boolean**
JMSReplyTo	**javax.jms.Destination**
JMSTimestamp	**long**
JMSType	**String**

Examples

The following command prints the JMS message header, custom message header, and message body of all the messages in the queue **TEST.FOO** on a broker:

```
JBossA-MQ:karaf@root>activemq:browse --amqurl tcp://localhost:61616
TEST.FOO
```

The following command displays the attributes from the body of the messages in the **TEST.FOO**

queue:

```
JBossA-MQ:karaf@root>activemq:browse --amqurl tcp://localhost:61616 -Vbody
TEST.FOO
```

The following command displays any messages with an ID ending in **10**:

```
JBossA-MQ:karaf@root>activemq:browse --amqurl tcp://localhost:61616 --
msgsel JMSMessaageID='*:10' TEST.FOO
```

The following command displays messages with a priority of **3**, enter:

```
JBossA-MQ:karaf@root>activemq:browse --amqurl tcp://localhost:61616 --
msgsel JMSPriority=3 TEST.FOO
```

The message selectors from the preceding two examples can be combined as follows:

```
JBossA-MQ:karaf@root>activemq:browse --amqurl tcp://localhost:61616 --
msgsel JMSMessaageID='*:10',JMSPriority=3 TEST.FOO
```

3.2. ACTIVEMQ:BSTAT, BSTAT

Abstract

summarizes the statistics for a broker

Synopsis

activemq:bstat [--jmxurl *JMXUrl*] [--pid *PID*] [-jmxuser *userName*] [-jmxpassword *password*] [-jmxlocal] [--version] [[--help] | [-h] | [-?]] { *brokerName* }

Arguments

Table 3.3, "**activemq:bstat** Arguments" describes the command's arguments.

Table 3.3. activemq:bstat Arguments

Argument	Description
--jmxurl *URL*	Sets the JMX URL used to locate brokers.
--pid *PID*	Set the pid to connect to (only on Sun JVM).
--jmxuser *user*	Sets the JMX user, used for authentication.
--jmxpassword *password*	Sets the JMX password, used for authentication.
--jmxlocal	Use the local JMX server instead of a remote server.

Argument	Description
--version	Displays the version information.
-h, -?, --help	Displays the online help for this command.
brokerName	The name of the broker

3.3. ACTIVEMQ:LIST

Abstract

lists all available brokers in the specified JMX context

Synopsis

activemq:list [--jmxurl *JMXUrl*] [--pid *PID*] [-jmxuser *userName*] [-jmxpassword *password*] [-jmxlocal] [--version] [[--help] | [-h] | [-?]]

Arguments

Table 3.4, "**activemq:list** Arguments" describes the command's arguments.

Table 3.4. activemq:list Arguments

Argument	Interpretation
--jmxurl *URL*	Sets the JMX URL to connect to
--pid *PID*	Set the pid to connect to (only on Sun JVM).
--jmxuser *user*	Sets the JMX user, used for authentication
--jmxpassword *password*	Sets the JMX password, used for authentication
--jmxlocal	Specifies to use the local JMX server instead of a remote server
--version	Displays the version information
-h, -?, --help	Displays the online help for this command

3.4. ACTIVEMQ:PURGE, PURGE

Abstract

purges messages from a destination

Synopsis

activemq:purge [--msgsel { msgsel ...}] [--pid *PID*] [--jmxurl *JMXUrl*] [-jmxuser *userName*] [-jmxpassword *password*] [-jmxlocal] [--version] [[--help] | [-h] | [-?]] { *destName* }

Arguments

Table 3.5, "**activemq:purge** Arguments" describes the command's arguments.

Table 3.5. activemq:purge Arguments

Option	Interpretation
--msgsel *msgsel1,msgsel2,...*	Purges messages matched by the message selector. See the section called "Message filters".
--jmxurl *URL*	Sets the JMX URL used to locate the broker.
--pid *PID*	Set the pid to connect to (only on Sun JVM).
--jmxuser *user*	Sets the JMX user, used for authentication.
--jmxpassword *password*	Sets the JMX password, used for authentication.
--jmxlocal	Specifies to use the local JMX server instead of a remote server
--version	Displays the version information
-h, -?, --help	Displays the online help for this command
destName	The specified message destination(s)

Examples

The following command purges all the messages in the queue **TEST.FOO** on a broker:

```
JBossA-MQ:karaf@root>activemq:purge TEST.FOO
```

The following command purges any messages with an ID ending in **10**:

```
JBossA-MQ:karaf@root>activemq:purge --msgsel JMSMessaageID='*:10' TEST.FOO
```

The following command purges messages with a priority of **3**, enter:

```
JBossA-MQ:karaf@root>activemq:purge --msgsel JMSPriority=3 TEST.FOO
```

The message selectors from the preceding two examples can be combined as follows:

```
JBossA-MQ:karaf@root>activemq:purge --msgsel
JMSMessaageID='*:10',JMSPriority=3 TEST.FOO
```

3.5. ACTIVEMQ:QUERY, QUERY

Abstract

queries the for broker information on specific objects

Synopsis

activemq:query [-Q*MBeanType=name*] [-xQ*MBeanType=name*] [--objname *query*] [--xobjname *query*] [--view { *attr* ...}] [--jmxurl *JMXUrl*] [--pid *PID*] [-jmxuser *userName*] [-jmxpassword *password*] [-jmxlocal] [--version] [[--help] | [-h] | [-?]]

Arguments

Table 3.6, "**activemq:query** Arguments" describes the command's arguments.

Table 3.6. activemq:query Arguments

Argument	Interpretation
-Q *type=name*	Adds to the search list the specific object type matched by the defined object identifier.
-xQ *type=name*	Removes from the search list the specific object type matched by the object identifier.
--objname *query*	Adds to the search list objects matched by the query similar.
--xobjname *query*	Removes from the search list objects matched by the query.
--view *attr1,attr2,...*	Selects the specific attribute of the object to view. By default, all attributes are displayed.
--jmxurl *URL*	Sets the JMX URL to connect to.
--pid *PID*	Set the pid to connect to (only on Sun JVM).
--jmxuser *user*	Sets the JMX user, used for authentication
--jmxpassword *password*	Sets the JMX password, used for authentication
--jmxlocal	Specifies to use the local JMX server instead of a remote server

Argument	Interpretation
--version	Displays the version information
-h, -?, --help	Displays the online help for this command

Examples

The following command displays all attributes and object name information for all registered MBeans in the default JMX context:

```
JBossA-MQ:karaf@root>activemq:query
```

The following command displays all attributes and object name information of the destination topic **TEST.FOO**:

```
JBossA-MQ:karaf@root>activemq:query -QTopic=TEST.FOO
```

The following command displays all the brokers in a context whose name ends in **host**:

```
JBossA-MQ:karaf@root>activemq:query -QBroker=*host
```

the Following command displays all attributes and object name information for all registered queues:

```
JBossA-MQ:karaf@root>activemq:query -QQueue=*
```

The following command displays all attributes and object name information for all topics ending with **.FOO** except those that also begin with **ActiveMQ.Advisory.**:

```
JBossA-MQ:karaf@root>activemq:query -QTopic=*.FOO -
xQTopic=ActiveMQ.Advisory.*
```

CHAPTER 4. ADMIN CONSOLE COMMANDS

The **admin** commands allow you to create, manage and destroy container instances.

Type **admin:** then press **Tab** at the **FuseMQkaraf:karaf@root>** prompt to view the available commands.

4.1. ADMIN:CHANGE-OPTS, CHANGE-OPTS

Abstract

changes the Java options of an existing container

Synopsis

admin:change-opts [--help] { *name* } { *opts* }

Arguments

Table 4.1, "**admin:change-opts** Arguments" describes the command's arguments.

Table 4.1. admin:change-opts Arguments

Argument	Interpretation
--help	Displays the online help for this command
name	The name of the container for which you want to change the Java options
opts	The Java options to change

4.2. ADMIN:CHANGE-RMI-REGISTRY-PORT, CHANGE-RMI-REGISTRY-PORT

Abstract

changes the RMI registry port used by the management layer of a container

Synopsis

admin:change-rmi-registry-port [--help] { *name* } { *port* }

Arguments

Table 4.2, "**admin:change-rmi-registry-port** Arguments" describes the command's arguments.

Table 4.2. admin:change-rmi-registry-port Arguments

Argument	Interpretation
`--help`	Displays the online help for this command
name	The name of the container instance for which you want to change the port
port	The new RMI registry port

4.3. ADMIN:CHANGE-RMI-SERVER-PORT, CHANGE-RMI-SERVER-PORT

Abstract

changes the RMI server port used by the management layer of a container

Synopsis

admin:change-rmi-server-port [--help] { *name* } { *port* }

Arguments

Table 4.3, "**admin:change-rmi-server-port** Arguments" describes the command's arguments.

Table 4.3. admin:change-rmi-server-port Arguments

Argument	Interpretation
`--help`	Displays the online help for this command
name	The name of the container instance for which you want to change the port
port	The new RMI server port

4.4. ADMIN:CHANGE-SSH-PORT, CHANGESSH-PORT

Abstract

changes the secure shell port of a container

Synopsis

admin:change-ssh-port [--help] { *name* } { *port* }

Arguments

Table 4.4, "`admin:change-ssh-port` Arguments" describes the command's arguments.

Table 4.4. `admin:change-ssh-port` Arguments

Argument	Interpretation
`--help`	Displays the online help for this command
`name`	The name of the container instance for which you want to change the port
`port`	The new secure shell port

4.5. ADMIN:CLONE, CLONE

Abstract

clones an existing container instance

Synopsis

`admin:clone` [--help] [[-l] | [--location]***`fileName`***] [[-o] | [--java-opts]***`JVMOpts`***] [[-s] | [--ssh-port]***`port`***] [[-rs] | [--rmi-server-port]***`port`***] [[-r] | [-rr] | [--rmi-port] | [--rmi-registry-port]***`port`***] [[-v] | [--verbose]] { *name* } { *cloneName* }

Arguments

Table 4.5, "`admin:clone` Arguments" describes the command's arguments.

Table 4.5. `admin:clone` Arguments

Argument	Interpretation
`--help`	Displays the online help for this command
`-l, --location`	Location of the cloned container instance in the file system.
`-o, --java-opts`	JVM options to use when launching the cloned instance.
`-s, --ssh-port`	Port number for remote secure shell connection.
`-rs, --rmi-server-port`	Port number for RMI server connection.
`-r, -rr, --rmi-port, --rmi-registry-port`	Port number for RMI registry connection.

Argument	Interpretation
-v, --verbose	Display actions performed by the command (disabled by default).
name	Name of the original container instance.
cloneName	Name of the cloned container instance.

4.6. ADMIN:CONNECT, CONNECT

Abstract

connects to an existing container

Synopsis

admin:connect [--help] [[-u] | [--username] ***userName***] [[-p] | [--password] ***password***] { *container* } [*command*]

Arguments

Table 4.6, "**admin:connect** Arguments" describes the command's arguments.

Table 4.6. admin:connect Arguments

Argument	Interpretation
--help	Displays the online help for this command
-u, --username	The remote user name; the default is **karaf**
-p, --password	The remote user password; the default is **karaf**
container	The container to connect to
command	Command to execute on connecting

4.7. ADMIN:CREATE, CREATE

Abstract

creates a new child container

Synopsis

admin:create [--help] [[-l] | [--location] ***filePath***] [[-furl] | [--featureURL] ***URL***...] [[-f] | [--feature] ***feature***...] [[-s] | [--ssh-port] ***SSHPort***] [[-rs] | [--rmi-server-port] ***RMIServPort***] [[-r] | [-rr] | [--rmi-registry-port] | [--rmi-port] ***RMIRegPort***] [[-o] | [--java-opts] ***javaOpts***] { *name* }

Arguments

Table 4.7, "**admin:create** Arguments" describes the command's arguments.

Table 4.7. admin:create Arguments

Argument	Interpretation
--help	Displays the online help for this command
-l, --location	The location of the child's data folders on the file system. By default, the child's data is added to the ***InstallDir*/instances/*name*** directory
-furl, **--featureURL**	Registers additional feature URLs with the child.
-f, **--feature**	Specifies additional features loaded by the child.
-s, --ssh-port	The port number for remote secure shell connection
-rs, --rmi-server-port	The port number for RMI server connection
-r, -rr, --rmi-registry-port, --rmi-port	The port number for RMI registry connection
-o, --java-opts	JVM options to use when launching the child
name	The name of the child

4.8. ADMIN:DESTROY, DESTROY

Abstract

destroys a child container

Synopsis

admin:destroy [--help] { *name* }

Arguments

Table 4.8, "**admin:destroy** Arguments" describes the command's arguments.

Table 4.8. admin:destroy Arguments

Argument	Interpretation
`--help`	Displays the online help for this command
name	The name of the container to destroy

4.9. ADMIN:LIST

Abstract

list all of the child containers on the current host

Synopsis

admin:list [--help] [[-l] | [--location] *filePath*] [[-o] | [--java-opts] *javaOpts*]

Arguments

Table 4.9, "**admin:list** Arguments" describes the command's arguments.

Table 4.9. admin:list Arguments

Argument	Interpretation
`--help`	Displays the online help for this command
`-l, --location`	Displays the location of the container instances
`-o, --java-opts`	Displays the options used when launching the container's JVM

4.10. ADMIN:RENAME, RENAME

Abstract

renames a child container

Synopsis

admin:rename [--help] { name } { new-name }

Arguments

Table 4.10, "**admin:rename** Arguments" describes the command's arguments.

Table 4.10. admin:rename Arguments

Argument	Interpretation
--help	Displays the online help for this command
name	Current name of the container
new-name	The new name for the container

4.11. ADMIN:START

Abstract

starts a child container

Synopsis

admin:start [--help] [[-o] | [--java-opts] ***javaOpts***] { *name* }

Arguments

Table 4.11, "**admin:start** Arguments" describes the command's arguments.

Table 4.11. admin:start Arguments

Argument	Interpretation
--help	Displays the online help for this command
-o, --java-opts	The Java options used when launching the container
name	The name of the container to start

4.12. ADMIN:STOP

Abstract

stops a child container

Synopsis

admin:stop [--help] { *name* }

Arguments

Table 4.12, "**admin:stop** Arguments" describes the command's arguments.

Table 4.12. `admin:stop` Arguments

Argument	Interpretation
`--help`	Displays the online help for this command
name	The name of the container to start

CHAPTER 5. CAMEL CONSOLE COMMANDS

The **camel** commands are used for managing Camel contexts and routes.

NOTE

Camel Karaf commands about routes no longer need the **context** as a second parameter. It is now an optional parameter. However, if you do not provide the **context**, the command is then considered as a bulk operation for all the camel contexts.

5.1. CAMEL:CONTEXT-INFO

Abstract

display detailed information about the specified Camel context

Synopsis

camel:context-info [--help] { *contextName* }

Arguments

This command takes the following arguments.

Table 5.1. camel:context-info Arguments

Argument	Interpretation
--help	Displays the online help for this command
contextName	The name of the Camel context.

5.2. CAMEL:CONTEXT-LIST

Abstract

list all active Camel contexts

Synopsis

camel:context-list [--help]

Arguments

This command takes the following arguments.

Table 5.2. camel:context-list Arguments

Argument	Interpretation
--help	Displays the online help for this command

5.3. CAMEL:CONTEXT-START

Abstract

start up the specified Camel context

Synopsis

camel:context-start [--help] { *contextName* }

Arguments

This command takes the following arguments.

Table 5.3. camel:context-start Arguments

Argument	Interpretation
--help	Displays the online help for this command
contextName	The name of the Camel context.

5.4. CAMEL:CONTEXT-STOP

Abstract

stop the specified Camel context

Synopsis

camel:context-stop [--help] { *contextName* }

Arguments

This command takes the following arguments.

Table 5.4. camel:context-stop Arguments

Argument	Interpretation
--help	Displays the online help for this command

Argument	Interpretation
contextName	The name of the Camel context.

5.5. CAMEL:ENDPOINT-LIST

Abstract

lists all deployed Camel endpoints

Synopsis

camel:endpoint-list [--help]

Arguments

This command takes the following arguments.

Table 5.5. camel:endpoint-list Arguments

Argument	Interpretation
--help	Displays the online help for this command

5.6. CAMEL:ROUTE-INFO

Abstract

display detailed information about the specified Camel route

Synopsis

camel:route-info [--help] { *route* } [*contextName*]

Arguments

This command takes the following arguments.

Table 5.6. camel:route-info Arguments

Argument	Interpretation
--help	Displays the online help for this command
route	The Camel route ID.

Argument	Interpretation
contextName	*(Optional)* The Camel context name.

5.7. CAMEL:ROUTE-LIST

Abstract

list the Camel routes

Synopsis

camel:route-list [--help] [*contextName*]

Description

You can optionally restrict the listing to show only the routes belonging to the specified Camel context.

Arguments

This command takes the following arguments.

Table 5.7. camel:route-list Arguments

Argument	Interpretation
--help	Displays the online help for this command
contextName	*(Optional)* The Camel context name.

5.8. CAMEL:ROUTE-RESUME

Abstract

resume the specified Camel route (which was previously suspended)

Synopsis

camel:route-resume [--help] { *route* } [*contextName*]

Arguments

This command takes the following arguments.

Table 5.8. camel:route-resume Arguments

Argument	Interpretation
--help	Displays the online help for this command
route	The Camel route ID.
contextName	*(Optional)* The Camel context name.

5.9. CAMEL:ROUTE-SHOW

Abstract

display the Camel route definition in XML format

Synopsis

camel:route-show [--help] { *route* } [*contextName*]

Arguments

This command takes the following arguments.

Table 5.9. camel:route-show Arguments

Argument	Interpretation
--help	Displays the online help for this command
route	The Camel route ID.
contextName	*(Optional)* The Camel context name.

5.10. CAMEL:ROUTE-START

Abstract

start the specified Camel route

Synopsis

camel:route-start [--help] { *route* } [*contextName*]

Arguments

This command takes the following arguments.

Table 5.10. `camel:route-start` Arguments

Argument	Interpretation
`--help`	Displays the online help for this command
`-n,--interval`	
`route`	The Camel route ID.

5.11. CAMEL:ROUTE-STOP

Abstract

stop the specified Camel route

Synopsis

`camel:route-stop` [--help] { *route* } [*contextName*]

Arguments

This command takes the following arguments.

Table 5.11. `camel:route-stop` Arguments

Argument	Interpretation
`--help`	Displays the online help for this command
`route`	The Camel route ID.
`contextName`	*(Optional)* The Camel context name.

5.12. CAMEL:ROUTE-SUSPEND

Abstract

suspend the specified Camel route

Synopsis

`camel:route-suspend` [--help] { *route* } [*contextName*]

Arguments

This command takes the following arguments.

Table 5.12. `camel:route-suspend` Arguments

Argument	Interpretation
`--help`	Displays the online help for this command
`route`	The Camel route ID.
`contextName`	*(Optional)* The Camel context name.

5.13. CAMEL:CONTEXT-INFLIGHT

Abstract

Displays the list of inflight exchanges.

Synopsis

`camel:context-inflight` [--help] { *contextName* }

Arguments

This command takes the following arguments.

Table 5.13. `camel:context-inflight` Arguments

Argument	Interpretation
`name`	The name of the Camel context.

Options

It provides the following options.

Table 5.14. `camel:context-inflight` Options

Option	Interpretation	
`--help`	Displays the online help for this command.	
`--list`	To limit the number of exchanges shown.	
`--sort`	When the value is true, sort by longest duration and when the value is false, sort by exchange id.	

5.14. CAMEL:COMPONENT-LIST

Abstract

Lists all Camel components that are in use in Karaf.

Synopsis

`camel:component-list` [--help] { *contextName* }

Arguments

This command takes the following arguments.

Table 5.15. `camel:component-list` Arguments

Argument	Interpretation
`name`	The name of the Camel context.

Options

It provides the following options.

Table 5.16. `camel:component-list` Options

Option	Interpretation	
`--help`	Displays the online help for this command.	

5.15. CAMEL:ROUTE-PROFILE

Abstract

Displays the profile information about Camel route(s).

Synopsis

`camel:route-profile` [--help] { *contextName* } { *route* }

Arguments

This command takes the following arguments.

Table 5.17. `camel:route-profile` Arguments

Argument	Interpretation
name	The name of the Camel context.
route	The Camel route ID.

Options

It provides the following options.

Table 5.18. `camel:route-profile` Options

Option	Interpretation	
--help	Displays the online help for this command.	

5.16. CAMEL:REST-REGISTRY-LIST

Abstract

Lists all the Camel REST services enlisted in the Rest Registry from a CamelContext.

Synopsis

camel:rest-registry-list [--help] { *contextName* }

Arguments

This command takes the following arguments.

Table 5.19. `camel:rest-registry-list` Arguments

Argument	Interpretation
name	The name of the Camel context where to look for the REST services.

Options

It provides the following options.

Table 5.20. `camel:rest-registry-list` Options

Option	Interpretation	

Option	Interpretation	
`--help`	Displays the online help for this command.	

5.17. CAMEL:ROUTE-RESET-STATS

Abstract

Reset route performance stats from a CamelContext

Synopsis

`camel:route-reset-stats` [--help] { *contextName* }

Arguments

This command takes the following arguments.

Table 5.21. `camel:route-reset-stats` Arguments

Argument	Interpretation
name	The Camel context name.

Options

It provides the following options.

Table 5.22. `camel:route-reset-stats` Options

Option	Interpretation	
`--help`	Displays the online help for this command.	

5.18. CAMEL:CONTEXT-SUSPEND

Abstract

Suspends a Camel context.

Synopsis

`camel:context-suspend` [--help] { *contextName* }

Arguments

This command takes the following arguments.

Table 5.23. `camel:context-suspend` Arguments

Argument	Interpretation
name	The Camel context name.

Options

It provides the following options.

Table 5.24. `camel:context-suspend` Options

Option	Interpretation	
--help	Displays the online help for this command.	

5.19. CAMEL:REST-SHOW

Abstract

Displays the Camel REST definition in XML.

Synopsis

camel:rest-show [--help] { *contextName* }

Arguments

This command takes the following arguments.

Table 5.25. `camel:rest-show` Arguments

Argument	Interpretation
name	The Camel context name.

Options

It provides the following options.

Table 5.26. `camel:rest-show` Options

Option	Interpretation	
`--help`	Displays the online help for this command.	

5.20. CAMEL:EIP-EXPLAIN

Abstract

Displays about the EIP in the CamelContext.

Synopsis

`camel:eip-explain` [--help] { *contextName* } { *nameOrId* }

Arguments

This command takes the following arguments.

Table 5.27. `camel:eip-explain` Arguments

Argument	Interpretation
`name`	The Camel context name.
`nameOrId`	The name of the EIP or a node id

Options

It provides the following options.

Table 5.28. `camel:eip-explain` Options

Option	Interpretation	
`--help`	Displays the online help for this command.	

5.21. CAMEL:CONTEXT-RESUME

Abstract

Resumes a Camel context.

Synopsis

camel:context-resume [--help] { *contextName* }

Arguments

This command takes the following arguments.

Table 5.29. camel:context-resume Arguments

Argument	Interpretation
name	The Camel context name.

Options

It provides the following options.

Table 5.30. camel:context-resume Options

Option	Interpretation	
--help	Displays the online help for this command.	

5.22. CAMEL:ENDPOINT-EXPLAIN

Abstract

Explain all Camel endpoints available in the CamelContext.

Synopsis

camel:endpoint-explain [--help] [--filter] { *contextName* }

Arguments

This command takes the following arguments.

Table 5.31. camel:endpoint-explain Arguments

Argument	Interpretation
name	The Camel context name.

Options

It provides the following options.

Table 5.32. camel:endpoint-explain Options

Option	Interpretation	
`--help`	Displays the online help for this command.	
`--filter`	To filter endpoints by pattern.	

CHAPTER 6. CONFIG CONSOLE COMMANDS

The config commands are used for managing container configuration. The configuration data is edited in two stages. First the changes are queued until they are dynamically loaded into the container by executing the **config:update** command. A copy of the configuration is persisted to the file system in the container's **etc** folder.

When editing a configuration the commands are used as follows:

1. Start the editing session for the specified configuration.

   config:edit

2. Edits, or creates, a configuration.
   - **config:proplist**

     Lists the properties in the configuration.
   - **config:propappend**

     Append a new property to the configuration.
   - **config:propset**

     Sets the value for a configuration property.
   - **config:propdel**

     Deletes a property from the configuration.
3. **config:update**

   Saves the changes and updates the containers using the configuration.

You can abandon an editing session using **config:cancel**.

Type **config:** then press **Tab** at the prompt to view the available commands.

6.1. CONFIG:CANCEL

Abstract

cancels the changes to the configuration being edited

Synopsis

config:cancel [--help]

Details

When editing a configuration, the changes are buffered until the editing session is closed. The **config:cancel** command clears the buffer without saving the changes and closes the editing session.

You can see a list of the buffered changes using the **jaas:pending** command.

Arguments

Table 6.1, "**config:cancel** Arguments" describes the command's arguments.

Table 6.1. config:cancel Arguments

Option	Interpretation
--help	Displays the online help for this command

6.2. CONFIG:DELETE, DELETE

Abstract

deletes a configuration from the container

Synopsis

config:delte [--help] [[-f] | [--use-file]] [--no-delete-cfg-file] { *pid* }

Details

When you delete a configuration, the change is made directly on the running container. Any properties set in the configuration are reverted to their default values and the behavior of the container will be immediate.

If you use the **--no-delete-cfg-file** argument, the original settings can be reloaded from the configuration file.

Arguments

Table 6.2, "**config:delete** Arguments" describes the command's arguments.

Table 6.2. config:delete Arguments

Option	Interpretation
--help	Displays the online help for this command.
-f, **--use-file**	Use a filename instead of the PID to locate the configuration.
--no-delete-cfg-file	Does not delete the associated configuration file from the container's **etc** folder.
pid	Specifies the configuration's persistent identifier.

6.3. CONFIG:EDIT, EDIT

Abstract

begins an editing session for a configuration. If the configuration does not exist a new configuration is created.

Synopsis

config:edit [--help] [--force] [[-f] | [--use-file]] { *pid* }

Details

The **config:edit** command is the first step in editing a container configuration. It opens the configuration so that calls to the **config:*** editing commands will update the selected configuration. The edits made by the **config:*** editing commands are placed in a buffer associated with the selected configuration and not propagated to the container, or the file system, until the editing session is ended by the **config:update** command.

If you use the **config:edit** command before saving the changes to a configuration that is open for editing, the changes to the previously open configuration are abandoned. The pending edits cleared without being saved.

Arguments

Table 6.3, "**config:edit** Arguments" describes the command's arguments.

Table 6.3. config:edit Arguments

Option	Interpretation
--help	Displays the online help for this command
--force	Forces the editing of this configuration, even if another configuration was being edited
-f, **--use-file**	Use a filename instead of the PID to locate the configuration
pid	The persistent identifier of the configuration

6.4. CONFIG:LIST

Abstract

lists the existing configurations for the container

Synopsis

config:list [--help] [*query*]

Arguments

Table 6.4, "**config:list** Arguments" describes the command's arguments.

Table 6.4. config:list Arguments

Argument	Interpretation
--help	Displays the online help for this command
query	An LDAP query

6.5. CONFIG:PROPAPPEND, PROPAPPEND

Abstract

appends the given value to an existing property or creates the property with the specified name and value

Synopsis

config:propappend [--help] [[-b] | [--bypass-storage]] [[-p *PID*] | [--pid *PID*]] { *name* } { *value* }

Details

When you append a value to a property using the **config:propappend** command, the change is stored in the buffer and not propagated to the container until the editing session is closed.

If you use the **-p** argument to specify a PID, however, the change is made immediately.

Arguments

Table 6.5, "**config:propappend** Arguments" describes the command's arguments.

Table 6.5. config:propappend Arguments

Argument	Interpretation
--help	Displays the online help for this command.
-b, **--bypass-storage**	Doe not write the change to the local file.
-p, **--pid**	Specifies the PID of the configuration in which to make the change. The default is to change the configuration currently open for editing.
name	Specifies the name of the property to change.

Argument	Interpretation
value	Specifies the value to append to the property.

6.6. CONFIG:PROPDEL, PROPDEL

Abstract

deletes a property from the configuration being edited

Synopsis

config:propdel [--help] [[-b] | [--bypass-storage]] [[-p *PID*] | [--pid *PID*]] { *name* }

Details

When you delete a property using the **config:propdel** command, the change is stored in the buffer and not propagated to the container until the editing session is closed.

If you use the **-p** argument to specify a PID, however, the change is made immediately.

Arguments

Table 6.6, "**config:propdel** Arguments" describes the command's arguments.

Table 6.6. config:propdel Arguments

Argument	Interpretation
--help	Displays the online help for this command.
-b, **--bypass-storage**	Does not write the change to the local file.
-p, **--pid**	Specifies the PID of the configuration in which to make the change. The default is to change the configuration currently open for editing.
name	Specifies the name of the property to delete.

6.7. CONFIG:PROPLIST, PROPLIST

Abstract

lists the properties in the configuration being edited

Synopsis

config:proplist [--help] [[-p *PID*] | [--pid *PID*]]

Arguments

Table 6.7, "**config:proplist** Arguments" describes the command's arguments.

Table 6.7. config:proplist Arguments

Argument	Interpretation
--help	Displays the online help for this command
-p, **--pid**	The PID of the configuration in which to make the change

6.8. CONFIG:PROPSET, PROPSET

Abstract

sets a property in the configuration being edited

Synopsis

config:propset [--help] [[-b] | [--bypass-storage]] [[-p *PID*] | [--pid *PID*]] { *name* } { *value* }

Details

When you set a property using the **config:propset** command, the change is stored in the buffer and not propagated to the container until the editing session is closed.

If you use the **-p** argument to specify a PID, however, the change is made immediately.

Arguments

Table 6.8, "**config:propset** Arguments" describes the command's arguments.

Table 6.8. config:propset Arguments

Argument	Interpretation
--help	Displays the online help for this command.
-b, **--bypass-storage**	Does not write the change to the local file.
-p, **--pid**	Specifies the PID of the configuration in which to make the change. The default is to change the configuration currently open for editing.
name	Specifies the name of the property to set.

Argument	Interpretation
value	Specifies the value to set for the property.

6.9. CONFIG:UPDATE

Abstract

saves the changes made to the configuration being edited and propagates then to the container

Synopsis

config:propset [--help] [[-b] | [--bypass-storage]]

Arguments

Table 6.9, "**config:update** Arguments" describes the command's arguments.

Table 6.9. config:update Arguments

Argument	Interpretation
--help	Displays the online help for this command
-b, **--bypass-storage**	Do not update the copy of the configuration on the file system

CHAPTER 7. CXF CONSOLE COMMANDS

The **cxf** commands are used for managing Apache CXF buses and endpoints. If these commands are not already loaded into the console, you can load them using the following console command:

```
JBossFuse:karaf@root> features:install cxf-commands
```

7.1. CXF:LIST-BUSSES

Abstract

lists all Apache CXF buses

Synopsis

cxf:list-busses [--help]

Arguments

This command takes the following arguments.

Table 7.1. cxf:list-busses Arguments

Argument	Interpretation
--help	Displays the online help for this command

7.2. CXF:LIST-ENDPOINTS

Abstract

list all active endpoints belonging to the specified Apache CXF bus

Synopsis

cxf:list-endpoints [--help] { *busID* }

Arguments

This command takes the following arguments.

Table 7.2. cxf:list-endpoints Arguments

Argument	Interpretation
--help	Displays the online help for this command

Argument	Interpretation
busID	The Apache CXF bus ID.

7.3. CXF:START-ENDPOINT

Abstract

start the specified Apache CXF endpoint belonging to the specified bus

Synopsis

cxf:start-endpoint [--help] { *busID* } { *endpointName* }

Arguments

This command takes the following arguments.

Table 7.3. cxf:start-endpoint Arguments

Argument	Interpretation
--help	Displays the online help for this command
busID	The Apache CXF bus ID.
endpointName	The name of the Apache CXF endpoint.

7.4. CXF:STOP-ENDPOINT

Abstract

stop the specified Apache CXF endpoint belonging to the specified bus

Synopsis

cxf:stop-endpoint [--help] { *busID* } { *endpointName* }

Arguments

This command takes the following arguments.

Table 7.4. cxf:stop-endpoint Arguments

Argument	Interpretation
--help	Displays the online help for this command
busID	The Apache CXF bus ID.
endpointName	The name of the Apache CXF endpoint.

CHAPTER 8. DEV CONSOLE COMMANDS

The dev commands are a collection of utilities that are useful testing bundles in the container.

Type **dev:** then press **Tab** at the prompt to view the available commands.

8.1. DEV:CLASSLOADERS, CLASSLOADERS

Abstract

displays a list of leaking bundle classloaders

Synopsis

dev:classloaders [--help]

Arguments

Table 8.1, "**dev:classloader** Arguments" describes the commands arguments.

Table 8.1. dev:classloader Arguments

Argument	Interpretation
--help	Displays the online help for this command

8.2. DEV:CREATE-DUMP, CREATE-DUMP

Abstract

creates a ZIP file containing diagnostic information

Synopsis

dev:create-dump [--help] [[-d *dumpFolder*] | [--directory *dumpFolder*]] { *dumpName* }

Arguments

Table 8.2, "**dev:create-dump** Arguments" describes the commands arguments.

Table 8.2. dev:create-dump Arguments

Argument	Interpretation
--help	Displays the online help for this command
-d, **--directory**	Specifies the folder into which to store the dump

Argument	Interpretation
dumpName	Specifies the name for the dump file

8.3. DEV:DYNAMIC-IMPORT, DYNAMIC-IMPORT

Abstract

enables/disables dynamic imports for a bundle

Synopsis

dev:dynamic-import [--help] { *bundleID* }

Arguments

Table 8.3, "**dev:dynamic-import** Arguments" describes the commands arguments.

Table 8.3. dev:dynamic-import Arguments

Argument	Interpretation
--help	Displays the online help for this command
bundleID	A bundle ID.

8.4. DEV:FRAMEWORK, FRAMEWORK

Abstract

enables/disables debugging for an OSGi framework

Synopsis

dev:framework [--help] {[[-debug] | [--enable-debug]] [[-nodebug] | [--disable-debug]]} { *framework* }

Arguments

Table 8.4, "**dev:framework** Arguments" describes the commands arguments.

Table 8.4. dev:framework Arguments

Argument	Interpretation
--help	Displays the online help for this command

Argument	Interpretation
-nodebug, --disable-debug	Disable debugging for the OSGi framework.
-debug, --enable-debug	Enable debugging for the OSGi framework.
framework	Name of the OSGi framework

8.5. DEV:PRINT-STACK-TRACES, PRINT-STACK-TRACES

Abstract

enables/disables printing of full stack traces in the console when the execution of a command throws an exception

Synopsis

dev:print-stack-traces [--help] [false]

Arguments

Table 8.5, "**dev:print-stack-traces** Arguments" describes the commands arguments.

Table 8.5. dev:print-stack-traces Arguments

Argument	Interpretation
--help	Displays the online help for this command
false	Disables stack traces

8.6. DEV:RESTART

Abstract

restart the container

Synopsis

dev:restart [--help] [[-c] | [--clean]]

Arguments

Table 8.6, "**dev:restart** Arguments" describes the commands arguments.

Table 8.6. dev:restart Arguments

Argument	Interpretation
`--help`	Displays the online help for this command
`-c, --clean`	Force a clean (cold) restart by deleting the container's `data` directory.

8.7. DEV:SHOW-TREE, SHOW-TREE

Abstract

shows the tree of bundles based on the wiring information

Synopsis

dev:show-tree [--help] { *bundleID* }

Arguments

Table 8.7, "**dev:show-tree** Arguments" describes the commands arguments.

Table 8.7. dev:show-tree Arguments

Argument	Interpretation
`--help`	Displays the online help for this command
bundleID	A bundle ID.

8.8. DEV:THREADS, THREADS

Abstract

shows the threads in the JVM

Synopsis

dev:threads [--help] [[-f] | [--flat]]

Arguments

Table 8.8, "**dev:threads** Arguments" describes the commands arguments.

Table 8.8. dev:threads Arguments

Argument	Interpretation
--help	Displays the online help for this command
-f, **--flat**	Do not show the threads in a tree

8.9. DEV:WAIT-FOR-SERVICE, WAIT-FOR-SERVICE

Abstract

wait for the specified OSGi service

Synopsis

dev:wait-for-service [--help] [[-t] | [--timeout]***timeout***] [[-e] | [--exception]] { *serviceClassOrFilter* }

Description

This command is useful when you are developing a console script and you want to wait for a specific OSGi service to start up, before proceeding with the execution of the script.

For example, the various command sets installed in the console (**shell:***, **admin:***, **features:***, and so on) are represented by OSGi services of type, **org.apache.karaf.shell.console.SubShell**. If you want to check that a sub-shell service is available, you could enter the following console command:

```
karaf@root> dev:wait-for-service -t 1000
org.apache.karaf.shell.console.SubShell
true
```

This form of the command is not very useful in this case, because there are many different instances of the **SubShell** service installed in the container. To be more specific, you can define an LDAP filter, which specifies one or more service property values. For example, you can wait specifically for the **osgi** sub-shell service by entering a command like the following:

```
karaf@root> dev:wait-for-service -t 1000 &
(objectClass=org.apache.karaf.shell.console.SubShell)(name=osgi)
true
```

Arguments

Table 8.9, "**dev:wait-for-service** Arguments" describes the commands arguments.

Table 8.9. dev:wait-for-service Arguments

Argument	Interpretation

Argument	Interpretation
--help	Displays the online help for this command
-t, --timeout	Timeout (specified in milliseconds: negative to not wait at all, zero to wait forever). Default is forever.
-e, --exception	Throw an exception if the wait command times out (the service is not found). Default is false.
serviceClassOrFilter	Specifies the OSGi service either by the service's class name or by an LDAP-style filter (which is applied to the OSGi service's properties).

8.10. DEV:WATCH, WATCH

Abstract

watches and automatically updates bundles

Synopsis

dev:watch [--help] [[--start] | [--stop]] [-i *interval*] [--list] [--remove] { *bundles* ...}

Arguments

Table 8.10, "**dev:watch** Arguments" describes the commands arguments.

Table 8.10. dev:watch Arguments

Argument	Interpretation
--help	Displays the online help for this command
--stop	Stop watching the specified bundles
--start	Start watching the specified bundles
-i	Specifies the interval, in milliseconds, to check the bundles.
--list	List the bundles being watched.
--remove	Remove the specified bundles from the watch list.
bundles...	Specifies a whitespace delimited list of bundle URLs or bundle IDs.

IMPORTANT

Only Maven URLs and Maven snapshots will be updated automatically. So, if you run

```
JBossA-MQ:karaf@root> dev:watch *
```

You are monitoring all bundles that have a location matching mvn:* that have -SNAPSHOT in their URL.

CHAPTER 9. FABRIC CONSOLE COMMANDS

This chapter describes **fabric** console commands.

9.1. FABRIC:CLUSTER-LIST

Abstract

lists the members of a cluster

Synopsis

fabric:cluster-list [--help] [*Path*]

Description

This command lists the members of the specified cluster, where the cluster can be a cluster of message brokers, a cluster of servlets, or a cluster of Web applications.

For example, to list all of the servlet services in the Fabric, enter the following console command:

```
fabric:cluster-list servlets
```

To list all of the Web application services in the Fabric, enter the following console command:

```
fabric:cluster-list webapps
```

Arguments

Table 9.1, "**fabric:cluster-list** Arguments" describes the command's arguments.

Table 9.1. fabric:cluster-list Arguments

Argument	Interpretation
--help	Displays the online help for this command
Path	Path of the fabric registry node (Zookeeper registry node) to list. Relative paths are evaluated relative to the base node, **/fabric/registry/clusters**. If not specified, all clusters are listed

Related topics

Section 9.32, "fabric:mq-create"

9.2. FABRIC:CLOUD-FIREWALL-EDIT

Abstract

manage a cloud container's firewall

Synopsis

`fabric:cloud-firewall-edit` [--help] [--owner *owner*] [--option *key=value*]

Arguments

Table 9.2, "**`fabric:cloud-firewall-edit` Arguments**" describes the command's arguments.

Table 9.2. `fabric:cloud-firewall-edit` Arguments

Argument	Interpretation
`--port`	The target IP port. To specify multiple ports, specify this flag multiple times on the command line—for example, **`--port 1234 --port 5678`**.
`--flush`	Flush all rules.
`--revoke`	Revoke the rule for the specified port. This blocks access to the specified IP port.
`--target-container`	The target container name.
`--source-container`	The source container, which has access granted or revoked.
`--target-node-id`	The target node ID.
`--source-cidr`	The source CIDR, which has access granted or revoked.
`--provider`	The cloud provider name.
`--help`	Displays the online help for this command.

9.3. FABRIC:CLOUD-SERVICE-ADD

Abstract

initialize a cloud provider (which can be used for provisioning containers in the cloud)

Synopsis

fabric:cloud-service-add [--help] [--provider *providerName*] [--name *name*] [--api *APIName*] [--endpoint *URL*] [--identity *accessKeyID*] [--credential *secretAccessKey*] [--owner *owner*] [--option *key=value*] [--async-registration]

Description

This command runs asynchronously. That is, although the command returns immediately, it runs a thread in the background, which completes the initialization of the cloud provider. You can use **fabric:cloud-service-list** to discover when the initialization has completed.

There are two different styles of usage for this command:

- *Commercial cloud provider*—if you are using a commercial cloud provider, JClouds provides prepackaged modules that encapsulate the basic connection details for the provider. The prepackaged modules are available to install as Karaf features (named **jclouds-*ProviderName***) and encapsulate such details as the endpoint URI, cloud API, and so on.

  For example, to install an Amazon Web Services (AWS) EC2 cloud provider, you can perform the following steps (assuming you are working in a standalone container):

  1. Install the basic set of fabric cloud commands:

     ```
     karaf@root> features:install fabric-jclouds
     ```

  2. Install the JClouds module specifically for AWS EC2:

     ```
     karaf@root> features:install jclouds-aws-ec2
     ```

  3. Add the AWS EC2 provider, specifying the login credentials for your EC2 account:

     ```
     karaf@root> fabric:cloud-service-add --provider aws-ec2 --
     identity AccessKeyID
     --credential SecretAccessKey
     ```

  4. You are now ready to start creating compute instances on the **aws-ec2** cloud service, using the **fabric:container-create-cloud** command.

- *Private cloud service*—if you are hosting your compute instances on a private cloud service, you must specify the connection details more explicitly, by supplying the **--api** and **--endpoint** options. In this case, you must also define a name for the cloud service, by supplying the **--name** option.

  For example, to define a connection to a private cloud service that uses the **openstack-nova** API through the endpoint, **http://172.16.0.1:4000/v2.0/**, you can perform the following steps (assuming you are working in a standalone container):

  1. Install the basic set of fabric cloud commands:

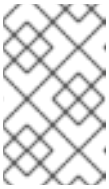
     ```
     karaf@root> features:install fabric-jclouds
     ```

  2. Install the JClouds module for the **openstack-nova** API:

```
karaf@root> features:install jclouds-api-openstack-nova
```

3. Add the private cloud service, specifying the login credentials, API, and endpoint URL:

```
karaf@root> fabric:cloud-service-add --name myOpenStack --api
openstack-nova
--endpoint http://172.16.0.1:4000/v2.0/ --identity AccessKeyID --
credential SecretAccessKey
```



NOTE

You can provide additional customisation of the connection by setting options through the **--option** flag (which can appear multiple times in the command).

4. You are now ready to start creating compute instances on the **myOpenStack** cloud service, using the **fabric:container-create-cloud** command.

Installing the command in a fabric

To access this command from a fabric container, you must have installed the **fabric-jclouds** feature. To install the **fabric-jclouds** feature, deploy the **cloud** profile into the current container, using the **fabric:container-change-profile** command.

For example, if the console is currently logged on to the **root** container of the Fabric, you could add the **cloud** profile as follows:

```
JBossA-MQ:karaf@root> fabric:container-list
[id]                           [version] [alive] [profiles]
[provision status]
root*                          1.0       true    fabric, fabric-
ensemble-0000-1 success
JBossA-MQ:karaf@root> fabric:container-change-profile root fabric fabric-
ensemble-0000-1 cloud
JBossA-MQ:karaf@root> fabric:container-list
[id]                           [version] [alive] [profiles]
[provision status]
root*                          1.0       true    fabric, fabric-
ensemble-0000-1, cloud success
```

Arguments

Table 9.3, "**fabric:cloud-service-add** Arguments" describes the command's arguments.

Table 9.3. fabric:cloud-service-add Arguments

Argument	Interpretation
--help	Displays the online help for this command.

Argument	Interpretation
--provider	The name of a commercial cloud provider (for example, **aws-ec2** or **rackspace**).
--name	The JClouds service context name, which identifies the cloud service uniquely. Defaults to the provider name (as specified by the **--provider** option).
--api	Specifies the cloud API (for example,**ec2**, **openstack-nova**, or **cloudstack**).
--endpoint	Specifies the cloud service's endpoint URL.
--identity	The identity used to access the cloud service.
--credential	The credential used to access the cloud service.
--owner	Specifies the EC2 AMI owner, which enables you to use private images (AWS EC2 only).
--option	Provider-specific properties. For example: **--option jclouds.regions=us-east-1**. If you want to specify more than one option, specify this option multiple times.
--async-registration	Do not wait for the provider registration (that is, complete the registration in a background thread).

9.4. FABRIC:CLOUD-SERVICE-LIST

Abstract

list the configured cloud providers

Synopsis

fabric:cloud-service-list [--help]

Description

For each configured cloud provider, displays the provider name, type (**compute** or **blobstore**), and registration (**local**, for a standalone container, or **fabric**, for a Fabric Container).

To access this command, the current container must belong to a Fabric and you must have installed the **fabric-jclouds** feature. To install the **fabric-jclouds** feature, deploy the **cloud** profile into the current container, using the **fabric:container-change-profile** command.

For example, if the console is currently logged on to the **root** container of the Fabric, you could add the **cloud** profile as follows:

```
JBossA-MQ:karaf@root> fabric:container-list
[id] [version] [alive] [profiles] [provision status] root* 1.0 true
fabric, fabric-ensemble-0000-1 success
JBossA-MQ:karaf@root> fabric:container-change-profile root fabric fabric-
ensemble-0000-1 cloud
JBossA-MQ:karaf@root> fabric:container-list
[id] [version] [alive] [profiles] [provision status] root* 1.0 true
fabric, fabric-ensemble-0000-1, cloud success
```

Arguments

Table 9.4, "**fabric:cloud-service-list** Arguments" describes the command's arguments.

Table 9.4. fabric:cloud-service-list Arguments

Argument	Interpretation
--help	Displays the online help for this command.

9.5. FABRIC:CLOUD-SERVICE-REMOVE

Abstract

removes the specified cloud provider

Synopsis

fabric:cloud-service-remove [--help] { *Name* }

Description

To access this command, the current container must belong to a Fabric and you must have installed the **fabric-jclouds** feature. To install the **fabric-jclouds** feature, deploy the **cloud** profile into the current container, using the **fabric:container-change-profile** command.

For example, if the console is currently logged on to the **root** container of the Fabric, you could add the **cloud** profile as follows:

```
JBossA-MQ:karaf@root> fabric:container-list
[id]                          [version] [alive] [profiles]
[provision status]
root*                         1.0       true    fabric, fabric-
ensemble-0000-1 success
JBossA-MQ:karaf@root> fabric:container-change-profile root fabric fabric-
ensemble-0000-1 cloud
JBossA-MQ:karaf@root> fabric:container-list
[id]                          [version] [alive] [profiles]
[provision status]
```

```
root*                           1.0       true    fabric, fabric-
ensemble-0000-1, cloud success
```

Arguments

Table 9.5, "`fabric:cloud-service-remove` Arguments" describes the command's arguments.

Table 9.5. `fabric:cloud-service-remove` Arguments

Argument	Interpretation
`--help`	Displays the online help for this command.
`Name`	The JClouds service context name, which represents the cloud provider.

9.6. FABRIC:CONTAINER-ADD-PROFILE, CONTAINER-ADD-PROFILE

Abstract

Adds the specified list of profiles to a container

Synopsis

`fabric:container-add-profile` [--help] { *Name* } { *Profiles* }

Arguments

Table 9.6, "`fabric:container-add-profile` Arguments" describes the command's arguments.

Table 9.6. `fabric:container-add-profile` Arguments

Argument	Interpretation
`--help`	Displays the online help for this command
`Name`	Specifies the name of the container.
`Profiles`	Specifies the list of profiles to add to the container.

9.7. FABRIC:CONTAINER-CHANGE-PROFILE, CONTAINER-CHANGE-PROFILE

Abstract

replaces a fuse container's profiles with the specified list of profiles

Synopsis

fabric:container-change-profile [--help] { *Name* } { *Profiles* }

Arguments

Table 9.7, "**fabric:container-change-profile** Arguments" describes the command's arguments.

Table 9.7. fabric:container-change-profile Arguments

Argument	Interpretation
--help	Displays the online help for this command
Name	Specifies the name of the container.
Profiles	Specifies the list of profiles to deploy into the container.

9.8. FABRIC:CONTAINER-CONNECT, CONTAINER-CONNECT

Abstract

connects to a remote Fabric Container and execute the specified command

Synopsis

fabric:container-connect [--help] [[-u] | [--username]***User***] [[-p] | [--password]***Password***] { *ContainerName* } [*Command*]

Description

This command allows you to connect to any container in the current fabric and execute a command. For example, to execute the **osgi:list** command on the **root2** container, you could enter a console command like Example 9.1, "Executing a Command in a Remote Container" .

Example 9.1. Executing a Command in a Remote Container

```
JBossA-MQ:karaf@root> fabric:container-connect -u YourName -p YourPass
root2 osgi:list
```

This command uses fabric JAAS security to log into the container, so the username and password are managed by the container's JAAS realm.

Arguments

Table 9.8, "**fabric:container-connect** Arguments" describes the command's arguments.

Table 9.8. fabric:container-connect Arguments

Argument	Interpretation
--help	Displays the online help for this command
-u, --username	Specifies the username for logging on to the remote container. The default is **admin**.
-p, --password	SPecifies the password for logging on to the remote container. The default is **admin**.
ContainerName	Specifies the name of the remote container.
Command	Specifies the console command to execute on the remote container.

9.9. FABRIC:CONTAINER-CREATE-CHILD

Abstract

create one or more child containers

Synopsis

fabric:container-create-child [--help] [--ensemble-server] [--profile ***profileID***] [--version ***version***] [--jmx-user ***jmxUser***] [--jmx-password ***jmxPass***] [-b, --bind-address ***bindAddr***] [--datastore-type ***storeType***] [--datastore-option ***storeOption***] [--zookeeper-password ***zooPass***] [--jvm-opts ***jvmOpts***] [--resolver ***policy***] [-m, --manual-ip ***IPAddr***] { *parent* } { *name* } [*number*]

Description

Child containers have the following characteristics:

- Each child container has a parent, so that the child containers form a hierarchy, with the root container as the ultimate ancestor.
- The child starts in a new JVM instance (JVM options can be passed to the new JVM through the **--jvm-opts** command option).
- A complete set of data directories are created for the child instance, under the ***ESBInstallDir*/instances/ChildName** directory. The ***ESBInstallDir*/system** directory is shared with the root container.

For example, if you have already created a new fabric (for example, by invoking **fabric:create**), you could add some child containers to the root container by entering the following command:

```
karaf@root> fabric:container-create-child root child 3
```

This command creates three new children under the **root** container. To check that the containers have been successfully created, invoke the **fabric:container-list** command, as follows:

```
karaf@root> fabric:container-list
[id]                           [version] [alive] [profiles]
[provision status]
root                           1.0       true    fabric, fabric-
ensemble-0000-1
  child1                       1.0       true    default
success
  child2                       1.0       true    default
success
  child3                       1.0       true    default
success
```

As you can see, the command creates three new child containers, **child1**, **child2**, and **child3**, with the **default** profile. This containers are ordinary (non-ensemble) containers, running fabric agents (ZooKeeper clients).

If you do not explicitly specify any profile (or profiles) for the new child containers, each of the child containers is created with the OSGi bundles required for a minimal Apache Karaf container and all of the profiles and bundles specified by the **default** profile.

To associate multiple profiles with a new child container, you can specify the **--profile** option multiple times. For example, if you want to deploy your own application profile, **myApp**, together with the **esb** profile, you would use a command like the following:

```
fabric:container-create-child --profile esb --profile myApp root
childMyApp
```

Shutting down child containers

After you create new child containers, the children run as separate processes, independently of the parent. Consequently, when you shut down the parent container, *the child processes continue to run in the background*. If you want to shut down the children, you must explicitly invoke the **fabric:container-stop** command. For example, if a root container has three children— **child1**, **child2**, and **child3**—you can issue the following commands in the root container console to shut down all of the containers:

```
karaf@root> fabric:container-stop child1
karaf@root> fabric:container-stop child2
karaf@root> fabric:container-stop child3
karaf@root> shutdown -f
```

Arguments

Table 9.9, "**fabric:container-create-child** Arguments" describes the command's arguments.

Table 9.9. fabric:container-create-child Arguments

Argument	Interpretation
--help	Displays the online help for this command.
--jmx-user	The JMX username of the parent container.

Argument	Interpretation
--jmx-password	The JMX password of the parent container.
--ensemble-server	Whether the new container should be a Fabric Server.
--profile	A profile ID to associate with the new container. To associate multiple profiles with the container, specify this flag multiple times on the command line —for example, **--profile foo --profile bar**. If no profile is specified, the container is associated with the **default** profile.
--resolver	Specifies how the container will report its address to other containers. Valid values are **localip**, **localhostname**, **publicip**, **publichostname**, **manualip**. For more information see Section 9.20, "fabric:container-resolver-set".
-m, --manual-ip	The IP address to use, when using the **manualip** resolver. Used in combination with the **--resolver** option.
-b, --bind-address	Specifies the default bind address.
--datastore-type	Specifies the datastore type.
--datastore-option	Options to pass to the container's datastore. To specify multiple options, use this flag multiple times.
--zookeeper-password	Used in combination with the **--ensemble-server** option. If creating an ensemble server, specifies the Zookeeper password to use (if not specified, a password is generated automatically).
--version	Specifies the version of the new container (the version must be created in advance using **fabric:version-create**). Defaults to the current default version (use **version-list** to find the current default).
--jvm-opts	Specify options to pass to the container's JVM.
Parent	*(Required)* The parent container ID.

Argument	Interpretation
Name	*(Required)* The name of the container to create. When creating multiple containers, it serves as a prefix
Number	The number of containers that should be created.

Related topics

For more details about resolver policies, see:

fabric:container-resolver-list
fabric:container-resolver-set
fabric:create

9.10. FABRIC:CONTAINER-CREATE-CLOUD

Abstract

creates one or more new containers on the cloud

Synopsis

fabric:container-create-cloud [--help] [--name ***contextName***] [--provider ***cloudProvider***] [--api ***cloudAPI***] [--identity ***cloudIdentity***] [--credential ***loginCredential***] [--imageId ***imageID***] [--os-family ***osFamily***] [--os-version ***osVersion***] [--hardwareId ***hardwareID***] [--instanceType ***instanceType***] [--locationId ***location***] [--user ***userAcc***] [--password ***userPass***] [--public-key-file ***file***] [--owner ***owner***] [--group ***group***] [--proxy-uri ***URI***] [--ensemble-server] [--new-user ***jaasUser***] [--new-user-password ***jaasUserPass***] [--new-user-role ***jaasUserRole***] [-b, --bind-address ***bindAddress***] [--datastore-type ***storeType***] [--datastore-option ***storeOption***] [--zookeeper-password ***zooPass***] [--resolver ***policy***] [-m, --manual-ip ***IPAddr***] [--env ***key=value***] [--min-port ***minPort***] [--max-port ***maxPort***] [--profile ***profileID***] [--version ***version***] [--jvm-opts ***jvmOpts***] [--add-option ***key=value***] [--no-admin-access] [--path ***installPath***] { *Name* } [*Number*]

Description

To access this command, you must have installed the **fabric-jclouds** feature. To install the **fabric-jclouds** feature, enter the following console command:

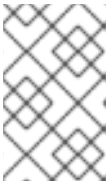
```
features:install fabric-jclouds
```

The **fabric:container-create-cloud** command provisions the container as follows:

1. Creates a new node on the cloud provider. The node is created using a JClouds compute service: either by lookup in the service registry (using the provider ID as a property) or by instantiating a new node, by specifying the identity and credential of the provider.

2. Connects to the created node, using the authentication metadata returned upon the node creation (this is usually a username and private key, where the username can be overridden by the `--user` option). After it connects to the node, it executes a script, which downloads the fabric distribution from the Maven proxy and untars the distribution.

   By default, the script uses the oldest Maven proxy server in the current ensemble (every ensemble server has a Maven proxy server deployed in it). You can optionally override the default Maven proxy by specifying the `--proxy-uri` option. The script would then use the specified Maven proxy server to download the container runtime.



NOTE

The ability to override the Maven proxy is important in certain cases (for example, in a cloud deployment) where the remote host might not be able to access the default Maven proxy server.

3. Starts up the newly installed container (or containers) and installs the specified fabric profile (or profiles).

4. When creating multiple containers using this command (by adding the ***`Number`*** argument), multiple nodes will be created and a root container will be installed on each node.

By default, the newly created cloud containers belong to the current fabric (that is, the same fabric as the container from which you invoked the command). It is possible, however, to create a container on the compute instance that acts as the seed for a completely new fabric, separate from the current one. To create a new fabric on the compute instance, invoke the **`fabric:container-create-cloud`** command with the **`--ensemble-server`** flag, which makes the newly created container (or containers) an ensemble server, with its own fabric registry agent. The newly created ensemble server on the cloud *does not join the current ensemble* it belongs to an independent ensemble (a new fabric).

Arguments

Table 9.10, "**`fabric:container-create-cloud`** Arguments" describes the command's arguments.

Table 9.10. `fabric:container-create-cloud` Arguments

Argument	Interpretation
`--help`	Displays the online help for this command
`--name`	*(Required)* JClouds service context name.
`--provider`	JClouds provider name.
`--api`	The cloud API name.
`--identity`	The identity used to access the cloud service.

Argument	Interpretation
--credential	The credential used to access the cloud service.
--imageId	The image ID to use for the new node(s). Alternatively, the image can be specified indirectly using the **--os-family** and **--os-version** options. Defaults to an instance of the latest version of Ubuntu.
--os-family	Specify the image by requesting a particular kind of operating system—for example, **ubuntu** or **redhat**. To see which O/S families are available, type Tab while entering this option. Defaults to **ubuntu**.
--os-version	Specifies the version of the O/S family. The version number need not be exact (it will be rounded up to the latest available patch version). Defaults to the latest version available.
--hardwareId	Kind of hardware to use.
--instanceType	Type of instance required.
--locationId	The location used to create the new node(s).
--user	Specifies the O/S user account to run on the new nodes. If the user account does not already exist on the new nodes, it will automatically be created. Defaults to the username that matches the current user.
--password	Specifies the password associated with the O/S user account defined by the **--user** option.
--public-key-file	An option to specify a public key file to copy to the created node. Copying a public key file to a node can be used for SSH access using public key authentication. If no key file is specified, Fabric attempts to auto-detect the user's public key and, if found, this key will be used by default.
--owner	Optional owner of images; only really used for EC2, and will be deprecated in future.
--group	Group tag to use on the new node(s). Defaults to **fabric**.

Argument	Interpretation
--proxy-uri	URL of the Maven proxy server used to download the container runtime.
--ensemble-server	Whether the new container should be a Fabric Server (effectively creates a new fabric).
--new-user	Used in combination with the **--ensemble-server** option to ensure that at least one user exists in the JAAS realm of the Zookeeper login module for the new fabric (otherwise it would be impossible to connect to the newly created Fabric Server). When using this option, you *must* also specify a password using the **--new-user-password** option.
--new-user-password	Used in combination with the **--new-user** option and the **--ensemble-server** option to specify the new user's password. No default value.
--new-user-role	Used in combination with the **--new-user** option and the **--ensemble-server** to specify the new user's role. Default is **admin**.
-b, --bind-address	Specifies the default bind address.
--datastore-type	Specifies the datastore type.
--datastore-option	Options to pass to the container's datastore. To specify multiple options, use this flag multiple times.
--zookeeper-password	Used in combination with the **--ensemble-server** option. Specifies the Zookeeper password, which is used to access the Zookeeper nodes under the **/fabric/** path. Defaults to the password of the current session user. If you subsequently try to join the current container to the newly-created Fabric Server (ensemble server) using the **fabric:join** command, you will be prompted to enter the Zookeeper password.
--resolver	Specifies how the container will report its address to other containers. Valid values are **localip**, **localhostname**, **publicip**, **publichostname**, **manualip**. For more information see Section 9.20, "fabric:container-resolver-set".

Argument	Interpretation
-m, --manual-ip	The IP address to use, when using the **manualip** resolver. Used in combination with the **--resolver** option.
--env	Sets an environment variable. To specify multiple environment variables, use this flag multiple times.
--min-port	Specifies the minimum port number of the allowed IP port range. Default is **0**.
--max-port	Specifies the maximum port number of the allowed IP port range. Default is **65535**.
--profile	A list of profile IDs to associate with the new container.
--version	Specifies the version of the new container (the version must be created in advance using **fabric:version-create**). Defaults to the current default version (use **version-list** to find the current default).
--jvm-opts	Specify options to pass to the container's JVM.
--add-option	Specifies generic JCloud properties or provider-specify properties. For example, when using Amazon with Amazon VPC to create a container inside a VPN, you can specify **--option subnetId=*yoursubnetId*** to define the VPC subnet where you want the node to be created. If you want to specify more than one option, specify this option multiple times.
--no-admin-access	Disables admin access, as it might not be feasible on all images.
--path	Path on the remote filesystem where the container is to be installed.
Name	*(Required)* The name of the container to create. When creating multiple containers, it serves as a prefix.
Number	The number of containers that should be created.

Related topics

See the other Fabric cloud commands:

- fabric:cloud-provider-add
- fabric:cloud-provider-list
- fabric:cloud-provider-remove

For more details about resolver policies, see:

- fabric:container-resolver-list
- fabric:container-resolver-set
- fabric:create

9.11. FABRIC:CONTAINER-CREATE-SSH

Abstract

creates one or more new containers through SSH

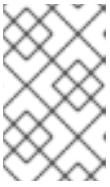
Synopsis

`fabric:container-create-ssh` [--help] [--host ***`host`***] [--port ***`port`***] [--min-port ***`minPort`***] [--max-port ***`maxPort`***] [--path ***`path`***] [--user ***`user`***] [--password ***`password`***] [--new-user ***`newUser`***] [--new-user-password ***`newPassword`***] [--new-user-role ***`newRole`***] [--private-key ***`keyPath`***] [--pass-phrase ***`passPhrase`***] [--ssh-retries ***`retries`***] [--proxy-uri ***`URI`***] [--ensemble-server] [--profile ***`profileID`***] [--version ***`version`***] [-b, --bind-address ***`bindAddress`***] [--datastore-type ***`storeType`***] [--datastore-option ***`storeOption`***] [--zookeeper-password ***`zooPass`***] [--jvm-opts***`jvmOpts`***] [--resolver ***`policy`***] [-m, --manual-ip ***`IPAddr`***] [--env ***`key=value`***] [--with-admin-access] [--disable-distribution-upload] { *Name* } [*Number*]

Description

Specifically, this command provisions the container as follows:

1. Logs into the specified SSH host, using either the provided username and password *or* using the provided username and private key.
2. Runs a script on the remote host that that downloads the container runtime to the remote host. The runtime files are downloaded through a Maven proxy server. By default, the script uses the oldest Maven proxy server in the current ensemble (every Fabric Server has a Maven proxy server deployed in it). You can optionally override the default Maven proxy by specifying the **`--proxy-uri`** option. The script would then use the specified Maven proxy server to download the container runtime.

   NOTE

   The ability to override the Maven proxy is important in certain cases (for example, in a cloud deployment) where the remote host might not be able to access the default Maven proxy server.

3. Starts up the newly installed container (or containers) and installs the specified fabric profile (or profiles).

By default, the newly created containers belong to the current fabric (that is, the same fabric as the container from which you invoked the command). It is possible, however, to create a container on the remote host that acts as the seed for a completely new fabric, separate from the current one. To create a new fabric on the remote host, invoke the **fabric:container-create-ssh** command with the **--ensemble-server** flag, which makes the newly created container (or containers) a Fuse Server. The newly created Fuse Server on the remote host *does not join the current ensemble* it belongs to an independent ensemble (a new fabric).

Arguments

Table 9.11, "**fabric:container-create-ssh** Arguments" describes the command's arguments.

Table 9.11. fabric:container-create-ssh Arguments

Argument	Interpretation
--help	Displays the online help for this command
--host	*(Required)* Host name to SSH into.
--port	The IP port number for the SSH connection. Default is **22**.
--min-port	The minimum port number of the allowed IP port range. Default is **0**.
--max-port	The maximum port number of the aollowed IP port range. Default is **65535**.
--path	Path on the remote filesystem where the container is to be installed.
--user	*(Required)* User name for login.
--password	Password for login. If the password is omitted, private key authentication is used instead.
--new-user	Used in combination with the **--ensemble-server** option to ensure that at least one user exists in the JAAS realm of the Zookeeper login module for the new fabric (otherwise it would be impossible to connect to the newly created Fabric Server). When using this option, you *must* also specify a password using the **--new-user-password** option.
--new-user-password	Used in combination with the **--new-user** option and the **--ensemble-server** option to specify the new user's password. No default value.

Argument	Interpretation
--new-user-role	Used in combination with the **--new-user** option and the **--ensemble-server** to specify the new user's role. Default is **admin**.
--private-key	Specifies the path to the private key on the local file system. The default is **~/.ssh/id_rsa** on *NIX platforms or **C:\Documents and Settings*UserName*\.ssh\id_rsa** on Windows.
--pass-phrase	The pass phrase of the key, if private key authentication is used and the private key is encrypted.
--ssh-retries	Maximum number or times to retry SSH connection.
--proxy-uri	URL of the Maven proxy server used to download the container runtime.
--ensemble-server	Creates a container on the remote host that acts as the seed for a completely new fabric, separate from the container that invokes the command. The newly created container is a Fuse server on the remote host and does not join the current ensemble. The new container belongs to an independent ensemble (a new fabric).
--profile	A list of profile IDs to associate with the new container.
--version	Specifies the version of the new container (the version must be created in advance using **fabric:version-create**). Defaults to the current default version (use **version-list** to find the current default).
-b, --bind-address	Specifies the default bind address.
--datastore-type	Specifies the datastore type.
--datastore-option	Options to pass to the container's datastore. To specify multiple options, use this flag multiple times.
--zookeeper-password	Used in combination with the **--ensemble-server** option. If creating an ensemble server, specifies the Zookeeper password to use (if not specified, a password is generated automatically).

Argument	Interpretation
--jvm-opts	Specify options to pass to the container's JVM.
--resolver	Specifies how the container will report its address to other containers. Valid values are **localip**, **localhostname**, **publicip**, **publichostname**, **manualip**. For more information see Section 9.20, "fabric:container-resolver-set".
-m, --manual-ip	The IP address to use, when using the **manualip** resolver. Used in combination with the **--resolver** option.
--env	Sets an environment variable. To specify multiple environment variables, use this flag multiple times.
--with-admin-access	Indicates that the target user has administrative access (password-less **sudo**). When this option is specified, Fabric will attempt to install any missing dependencies on the target host.
--disable-distribution-upload	Flag to disable uploading the JBoss Fuse distribution. When used, the target host downloads the distribution through Maven instead.
Name	*(Required)* The name of the container to create. When creating multiple containers, it serves as a prefix.
Number	The number of containers that should be created.

Related topics

For more details about resolver policies, see:

fabric:container-resolver-list
fabric:container-resolver-set
fabric:create

9.12. FABRIC:CONTAINER-DEFAULT-JVM-OPTIONS, CONTAINER-DEFAULT-JVM-OPTIONS

Abstract

get or set the default JVM options to use when creating a new container

Synopsis

`fabric:container-default-jvm-options` [--help] [*JVMOptions*]

Arguments

Table 9.12, "**`fabric:container-default-jvm-options`** Arguments" describes the command's arguments.

Table 9.12. `fabric:container-default-jvm-options` Arguments

Argument	Interpretation
`--help`	Displays the online help for this command
`JVMOptions`	Sets the default JVM options for this container. If this argument is omitted, get the current default JVM options.

9.13. FABRIC:CONTAINER-DELETE

Abstract

stops and deletes a Fuse Container

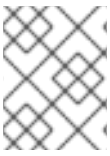
Synopsis

`fabric:container-delete` [--help] [--user ***`user`***] [--password ***`password`***] [-f, --force] [[-r] | [--recursive]] { *GlobList* }

Description

Deleting a Fuse Container deletes all of the files associated with the container from the host.

If the container has children, the default behavior of the command is to leave the children in place. You can force the deletion of the children using the **`-r`** option.



NOTE

If the container to be deleted is a Fabric Server, you must first remove it from the ensemble using **`fabric:ensemble-remove`**.

Arguments

Table 9.13, "**`fabric:container-delete`** Arguments" describes the command's arguments.

Table 9.13. `fabric:container-delete` Arguments

Argument	Interpretation
--help	Displays the online help for this command.
--user	Specifies the username of the user performing this action.
--password	Specifies the password of the user performing this action.
-f, --force	Forces execution of the command, regardless of the known state of the container.
-r, --recursive	Recursively stops and deletes all child containers.
GlobList	*(Required)* Specifies the list of containers to delete, separated by spaces. Globbing is supported as follows: **?**, matches zero or one characters; *****, matches zero or more characters.

Using alternative credentials

Each time a container is created Fabric stores the credentials used to create the container and will reuse them for all life cycle operations (start, stop, delete). If the credentials have changed, the updated credentials need to be specified as options. Each container type (child, ssh, jclouds) uses different kind of credentials, as follows:

- Child containers: Use the JMX credentials of the parent container.
- SSH containers: Use the SSH credentials of the target host.
- JClouds containers: Use the ssh credentials of the instance.

For example:

```
fabric:container-delete --user NewJmxUserOfParent --password
NewJmxPasswordOfParent child1
fabric:container-delete --password NewSshPasswordOfTargetHost ssh1
fabric:container-delete --password NewSshPasswordOfTargetHost cloud1
```

9.14. FABRIC:CONTAINER-EDIT-JVM-OPTIONS

Abstract

Replace the JVM options used to launch a container

Synopsis

fabric:container-edit-jvm-options [--help] [-u, --username ***user***] [-p, --password ***password***] { *Container* } [*JVMOptions*]

Description

Enables you to set the Java Virtual Machine (JVM) options that are passed to the Java runtime when launching the container. Note that this command *replaces all of the existing JVM options for the container*. Hence, when you invoke this command, you must specify *all* of the container's JVM options, not just the ones you want to add.

For example, suppose you have a container named **ssh2** and you want to add the following JVM options: **-Xms2048M -Xmx8192M**. Proceed as follows:

1. Invoke the **container-edit-jvm-options** command to discover the current JVM options used by the **ssh2** container, as follows:

```
JBossFuse:karaf@root> container-edit-jvm-options ssh2
-XX:+UnlockDiagnosticVMOptions -XX:+UnsyncloadClass -
Djava.endorsed.dirs=/usr/lib/jvm/jre/jre/lib/endorsed:/usr/lib/jvm/j
re/lib/endorsed:/home/fuse/containers/ssh2/fabric8-karaf-
1.2.0.redhat-133/lib/endorsed -
Djava.ext.dirs=/usr/lib/jvm/jre/jre/lib/ext:/usr/lib/jvm/jre/lib/ext
:/home/fuse/containers/ssh2/fabric8-karaf-1.2.0.redhat-133/lib/ext -
Dkaraf.instances=/home/fuse/containers/ssh2/fabric8-karaf-
1.2.0.redhat-133/instances -
Dkaraf.home=/home/fuse/containers/ssh2/fabric8-karaf-1.2.0.redhat-
133 -Dkaraf.base=/home/fuse/containers/ssh2/fabric8-karaf-
1.2.0.redhat-133 -Dkaraf.data=/home/fuse/containers/ssh2/fabric8-
karaf-1.2.0.redhat-133/data -
Dkaraf.etc=/home/fuse/containers/ssh2/fabric8-karaf-1.2.0.redhat-
133/etc -Djava.io.tmpdir=/home/fuse/containers/ssh2/fabric8-karaf-
1.2.0.redhat-133/data/tmp -
Djava.util.logging.config.file=/home/fuse/containers/ssh2/fabric8-
karaf-1.2.0.redhat-133/etc/java.util.logging.properties -
Djavax.management.builder.initial=org.apache.karaf.management.boot.K
arafMBeanServerBuilder -Dkaraf.startLocalConsole=false -
Dkaraf.startRemoteShell=true
```

2. Invoke the **container-edit-jvm-options** command, passing all of new JVM options as the second argument to the command (enclosed in single or double quotes). That is, to construct the second argument, copy the JVM options output from the previous step, append your additional JVM arguments, and enclose all of these JVM options in quotes. For example:

```
JBossFuse:karaf@root> container-edit-jvm-options ssh2 '-
XX:+UnlockDiagnosticVMOptions -XX:+UnsyncloadClass -
Djava.endorsed.dirs=/usr/lib/jvm/jre/jre/lib/endorsed:/usr/lib/jvm/j
re/lib/endorsed:/home/fuse/containers/ssh2/fabric8-karaf-
1.2.0.redhat-133/lib/endorsed -
Djava.ext.dirs=/usr/lib/jvm/jre/jre/lib/ext:/usr/lib/jvm/jre/lib/ext
:/home/fuse/containers/ssh2/fabric8-karaf-1.2.0.redhat-133/lib/ext -
Dkaraf.instances=/home/fuse/containers/ssh2/fabric8-karaf-
1.2.0.redhat-133/instances -
Dkaraf.home=/home/fuse/containers/ssh2/fabric8-karaf-1.2.0.redhat-
133 -Dkaraf.base=/home/fuse/containers/ssh2/fabric8-karaf-
1.2.0.redhat-133 -Dkaraf.data=/home/fuse/containers/ssh2/fabric8-
karaf-1.2.0.redhat-133/data -
Dkaraf.etc=/home/fuse/containers/ssh2/fabric8-karaf-1.2.0.redhat-
133/etc -Djava.io.tmpdir=/home/fuse/containers/ssh2/fabric8-karaf-
```

```
1.2.0.redhat-133/data/tmp -
Djava.util.logging.config.file=/home/fuse/containers/ssh2/fabric8-
karaf-1.2.0.redhat-133/etc/java.util.logging.properties -
Djavax.management.builder.initial=org.apache.karaf.management.boot.K
arafMBeanServerBuilder -Dkaraf.startLocalConsole=false -
Dkaraf.startRemoteShell=true -Xms2048M -Xmx8192M'
```

3. In order for these new JVM options to take effect, you need to restart the **ssh2** container, as follows:

```
JBossFuse:karaf@root> container-stop ssh2
JBossFuse:karaf@root> container-start ssh2
```

4. Verify that the current JVM settings have the new values, as follows:

```
JBossFuse:karaf@root> container-edit-jvm-options ssh2
...
```

Arguments

Table 9.14, "**fabric:container-edit-jvm-options** Arguments" describes the command's arguments.

Table 9.14. fabric:container-edit-jvm-options Arguments

Argument	Interpretation
--help	Displays the online help for this command.
-u, --username	Specifies the username to log on to a remote container.
-p, --password	Specifies the password to log on to a remote container.
Container	The name of the container.
JVMOptions	Specifies the container's new JVM options, which completely replace the old JVM options. If this argument is omitted, the command shows the existing JVM options for the specified container.

9.15. FABRIC:CONTAINER-DOMAINS, CONTAINER-DOMAINS

Abstract

lists a container's JMX domains

Synopsis

fabric:container-domains [--help] { *Name* }

Arguments

Table 9.15, "**fabric:container-domains** Arguments" describes the command's arguments.

Table 9.15. fabric:container-domains Arguments

Argument	Interpretation
--help	Displays the online help for this command.
Name	Specifies the name of the container.

9.16. FABRIC:CONTAINER-INFO, CONTAINER-INFO

Abstract

displays information about the specified container

Synopsis

fabric:container-info [--help] [*ContainerName*]

Arguments

Table 9.16, "**fabric:container-info** Arguments" describes the command's arguments.

Table 9.16. fabric:container-info Arguments

Argument	Interpretation
--help	Displays the online help for this command
ContainerName	Specifies the name of the container for which information is provided. If not specified, the container name defaults to **root**.

9.17. FABRIC:CONTAINER-LIST, CONTAINER-LIST

Abstract

lists the containers in a fabric

Synopsis

fabric:container-list [--help] [--version *Version*] [[-v] | [--verbose]] [[*ID*] | [*profile*]]

Arguments

Table 9.17, "`fabric:container-list` Arguments" describes the command's arguments.

Table 9.17. `fabric:container-list` Arguments

Argument	Interpretation
`--help`	Displays the online help for this command
`--version`	Specifies a profile version to use as filter.
`-v, --verbose`	Display verbose output.
ID	Specifies a container ID to use in filtering the output.
profile	Specifies a profile to use in filtering the output. When a profile is specified only the containers with the profile are listed.

9.18. FABRIC:CONTAINER-REMOVE-PROFILE, CONTAINER-REMOVE-PROFILE

Abstract

removes the specified list of profiles from the container

Synopsis

`fabric:container-remove-profile` [--help] { *Name* } { *Profiles* }

Arguments

Section 9.18, "fabric:container-remove-profile, container-remove-profile" describes the command's arguments.

Table 9.18. `fabric:container-remove-profile` Arguments

Argument	Interpretation
`--help`	Displays the online help for this command
Name	Specifies the name of the container.
Profiles	Specifies the list of profiles to remove from the container.

9.19. FABRIC:CONTAINER-RESOLVER-LIST

Abstract

show the resolver policies for the specified containers

Synopsis

`fabric:container-resolver-list` [--help] [*containers*]

Description

For all containers in the fabric, list the resolver policy and the following variants of the host address: local IP address, local hostname, public IP address, public hostname, and manually specified IP address.

The host addresses are found by looking them up in the Fabric Registry for each container. This information is stored in the Fabric Registry at the time when the container is created. In most cases, only the local IP address and the local hostname are known. The public IP address and public hostname are generally available only for cloud containers.

Arguments

Table 9.19, "**`fabric:container-resolver-list`** Arguments" describes the command's arguments.

Table 9.19. `fabric:container-resolver-list` Arguments

Argument	Interpretation
`--help`	Displays the online help for this command
`containers`	List of containers for which the resolver policy is displayed. Shows all containers by default.

Related topics

fabric:container-resolver-set

9.20. FABRIC:CONTAINER-RESOLVER-SET

Abstract

specifies how the container reports its address to other containers

Synopsis

`fabric:container-resolver-set` [--help] [--container ***name***] [--all] { *Resolver* }

Description

Apply the specified resolver policy to the specified container or containers, where the resolver policy can take one of the following values:

```
localip
localhostname
publicip
publichostname
manualip
```

The **localip** and **localhostname** resolver policies are suitable for accessing a container in a LAN. The **publicip** and **publichostname** resolver policies are suitable for accessing a container in a WAN (Internet), but they are typically only available for cloud containers. In the case of a the cloud, **localip** and **localhostname** can be used for container-to-container connections within the cloud, but for container-to-container connections from outside the cloud, you must use **publicip** or **publichostname**.

Fabric manages host addresses as follows:

- When you create a new container, fabric tries to discover as much as it can about the container's host address and stores this information in the following fields in the fabric registry: **localip** (local IP address); **localhostname** (local hostname); **publicip** (public IP address); **publichostname** (public hostname).

  For example, if you create a new container using the **fabric:container-create-ssh** command and specify the local IP address to the **--host** option, fabric attempts to perform a reverse lookup to obtain the corresponding local hostname and then stores both the local IP address *and* the local hostname in the Fabric Registry.

  If you create a new container in the cloud, the metadata sent by the cloud provider typically includes a complete set of host addresses: **localip**, **localhostname**, **publicip**, and **publichostname**.

- Every container in the fabric has its own *resolver policy*, which determines what kind of host address is returned to another container that wants to connect to it. The container's resolver policy is set in one of the following ways:

  - *(Default)* By inheriting the resolver policy from the global resolver policy (specified at the time the fabric is created)
  - By specifying the resolver policy explicitly at the time the container is created (through the **--resolver** option).
  - By invoking the **fabric:container-resolver-set** command.

- The container's resolver policy is applied whenever fabric looks up the container's host address, irrespective of what protocol is involved. In particular, the resolver policy determines the form of the host address used in the following URLs:

  - Fabric Ensemble URL,
  - SSH URL (console client port),
  - Maven proxy URL,
  - JMX URL.

For example, if your fabric includes a container called **SSH1** (originally created using the **fabric:container-create-ssh** command) and the **SSH1** container is configured with the **localip** resolver policy, any container that tries to connect to **SSH1** will automatically receive the

local IP address of **SSH1** when it looks up the Fabric Registry.

NOTE

A container's resolver policy only affects the host address returned when *other* containers want to connect to it. The container's own policy has no effect on how the container resolves the host addresses of the other containers. In other words, if containers **X**, **Y**, and **Z** want to connect to container **SSH1**, the form of host address they get is determined by **SSH1**'s resolver policy. But if **SSH1** wants to connect to container **X**, it is container **X**'s resolver policy that is used.

Manual IP resolver policy

The **manualip** resolver policy is a special case. If none of the standard resolver policies are suitable for your network set-up, you can manually specify a container's host address by setting the following key in the Fabric Registry:

```
/fabric/registry/containers/config/ContainerName/manualip
```

Arguments

Table 9.20, "**fabric:container-resolver-set** Arguments" describes the command's arguments.

Table 9.20. fabric:container-resolver-set Arguments

Argument	Interpretation
--help	Displays the online help for this command
--container	Apply the resolver policy to the specified container. To specify multiple containers, specify this flag multiple times on the command line—for example, **--container foo --container bar**.
Resolver	*(Required)* The resolver policy to set on the specified container(s). Possible values are: **localip**, **localhostname**, **publicip**, **publichostname**, **manualip**.

9.21. FABRIC:CONTAINER-ROLLBACK

Abstract

roll back the specified containers to an older version

Synopsis

fabric:container-rollback [--help] [--all] { *Version* } [*ContainerList*]

Description

For an example of how this command is used, see fabric:container-upgrade.

Arguments

Table 9.21, "**fabric:container-rollback** Arguments" describes the command's arguments.

Table 9.21. fabric:container-rollback Arguments

Argument	Interpretation
--help	Displays the online help for this command
--all	Roll back all containers.
Version	*(Required)* The version to roll back to.
ContainerList	The list of containers to roll back. An empty list implies the current container.

9.22. FABRIC:CONTAINER-START

Abstract

Start the specified container

Synopsis

fabric:container-start [--help] [--user ***user***] [--password ***password***] [-f, --force] { *GlobList* }

Arguments

Table 9.22, "**fabric:container-start** Arguments" describes the command's arguments.

Table 9.22. fabric:container-start Arguments

Argument	Interpretation
--help	Displays the online help for this command.
--user	Specifies the username of the user performing this action.

Argument	Interpretation
--password	Specifies the password of the user performing this action.
-f, --force	Forces execution of the command, regardless of the known state of the container.
GlobList	*(Required)* Specifies the list of containers to start, separated by spaces. Globbing is supported as follows: **?**, matches zero or one characters; *****, matches zero or more characters.

Using alternative credentials

Each time a container is created Fabric stores the credentials used to create the container and will reuse them for all life cycle operations (start, stop, delete). If the credentials have changed, the updated credentials need to be specified as options. Each container type (child, ssh, jclouds) uses different kind of credentials, as follows:

- Child containers: Use the JMX credentials of the parent container.
- SSH containers: Use the SSH credentials of the target host.
- JClouds containers: Use the ssh credentials of the instance.

For example:

```
fabric:container-start --user NewJmxUserOfParent --password
NewJmxPasswordOfParent child1
fabric:container-start --password NewSshPasswordOfTargetHost ssh1
fabric:container-start --password NewSshPasswordOfTargetHost cloud1
```

9.23. FABRIC:CONTAINER-STOP

Abstract

Shuts down the specified container

Synopsis

fabric:container-stop [--help] [--user ***user***] [--password ***password***] [-f, --force] { *GlobList* }

Arguments

Table 9.23, "**fabric:container-stop** Arguments" describes the command's arguments.

Table 9.23. fabric:container-stop Arguments

Argument	Interpretation
--help	Displays the online help for this command
--user	Specifies the username of the user performing this action.
--password	Specifies the password of the user performing this action.
-f, --force	Forces execution of the command, regardless of the known state of the container.
GlobList	*(Required)* Specifies the list of containers to stop, separated by spaces. Globbing is supported as follows: **?**, matches zero or one characters; *****, matches zero or more characters.

Using alternative credentials

Each time a container is created Fabric stores the credentials used to create the container and will reuse them for all life cycle operations (start, stop, delete). If the credentials have changed, the updated credentials need to be specified as options. Each container type (child, ssh, jclouds) uses different kind of credentials, as follows:

- Child containers: Use the JMX credentials of the parent container.
- SSH containers: Use the SSH credentials of the target host.
- JClouds containers: Use the ssh credentials of the instance.

For example:

```
fabric:container-stop --user NewJmxUserOfParent --password
NewJmxPasswordOfParent child1
fabric:container-stop --password NewSshPasswordOfTargetHost ssh1
fabric:container-stop --password NewSshPasswordOfTargetHost cloud1
```

9.24. FABRIC:CONTAINER-UPDATE-SSH

Abstract

Update the SSH credentials for a specified container.

Synopsis

fabric:container-update-ssh-credentials [--help] { --user } { --password } { *Container* }

Description

This command is used to update the credentials of a remote SSH container. The credentials are held in ZooKeeper in binary format, so they cannot be edited manually.

Arguments

The table below contains the arguments for the **fabric:container-update-ssh** command.

Table 9.24. fabric:container-update-ssh-credentials Arguments

Argument	Interpretation
--help	Displays the online help for this command
--user	*(Required)* The user ID of the user performing the operation.
--password	*(Required)* The password of the user performing the operation.
Container	*(Required)* The name of the container on which to perform the operation.

9.25. FABRIC:CONTAINER-UPGRADE

Abstract

upgrade the specified containers to a new version

Synopsis

fabric:container-upgrade [--help] [--all] { *Version* } [*ContainerList*]

Description

This command is typically used in combination with the **fabric:profile-edit** command to guarantee atomicity of profile modifications. That is, if multiple edits need to be made to a profile, you can use **fabric:container-upgrade** to roll out all of the changes in one step.

For example, consider the container, **child1**, which is currently assigned to version 1.0 and has the **sample** profile deployed inside it. If you need to make multiple changes to the **sample** profile, you can roll out these changes atomically, as follows:

1. Create a new version, 1.1, to hold the pending changes, as follows:

```
karaf@root> fabric:version-create
Created version: 1.1 as copy of: 1.0
```

2. Now start editing the new version of the sample profile, remembering to specify **1.1**, so that the modifications are applied to version 1.1 of **sample**. For example, to add the **camel-quartz** feature to the sample profile, enter the following command:

```
fabric:profile-edit --feature camel-quartz sample 1.1
```

NOTE

Instead of adding the option **1.1** to every edit command, you could change the default version to 1.1 by entering the command, **fabric:version-set-default 1.1**.

3. When you have finished editing the **sample** profile and you are ready to let the changes take effect on the container, **child1**, you can roll out the changes by upgrading the **child1** container to version 1.1, as follows:

```
fabric:container-upgrade 1.1 child1
```

4. If you are not happy with the changes you made, you can easily roll back to the old version of the **sample** profile, using the **fabric:container-rollback** command, as follows:

```
fabric:container-rollback 1.0 child1
```

Arguments

Table 9.25, "**fabric:container-upgrade** Arguments" describes the command's arguments.

Table 9.25. fabric:container-upgrade Arguments

Argument	Interpretation
--help	Displays the online help for this command
--all	Upgrade all containers.
Version	*(Required)* The version to upgrade to.
ContainerList	The list of containers to upgrade. An empty list implies the current container.

9.26. FABRIC:CREATE

Abstract

creates a new fabric and imports fabric profiles

Synopsis

fabric:create [--help] [--clean] [--no-import] [--import-dir ***dir***] [[-v] | [--verbose]] [[-t] | [--time]***millis***] [[-n] | [--non-managed]] [[-p] | [--profile]***profile***] [[-b] | [--bind-address]***bindAddr***] [--new-user ***username***] [--new-user-password ***password***] [--new-user-role ***role***] [--zookeeper-password ***zooPassword***] [--generate-zookeeper-password] [--zookeeper-data-dir

dataDir] [--zookeeper-init-limit ***ticks***] [--zookeeper-sync-limit ***ticks***] [--zookeeper-ticktime ***millis***] [--zookeeper-server-port ***zkport***] [--wait-for-provisioning] [[-t] | [--time]***millis***] [[-g] | [--global-resolver]***policy***] [[-r] | [--resolver]***policy***] [[-m] | [--manual-ip]***ipAddress***] [--min-port ***port***] [--max-port ***port***] [--external-git-url ***gitURL***] [--external-git-user ***gitUser***] [--external-git-password ***gitPass***] [--bootstrap-timeout] [*ContainerList*] [--zookeeper-purge-interval ***hours***] [--zookeeper-snap-retain-count ***number***]

Description

This command is used to create a new fabric. It can also be used to change the Fabric Servers in an existing fabric. Converting the current container into a fabric has two important side effects:

- The contents of a container should now be managed using *fabric profiles*. Do not try to deploy bundles and features directly in a fabric container.
- The default JAAS realm is superseded by the Zookeeper login module, which stores user data in the Zookeeper registry. As the fabric is created it initializes the user data by importing all of the user data that it finds in the **`etc/users.properties`** file. If the **`users.properties`** file is empty, you can specify a new user explicitly using the **`--new-user`** and **`--new-user-password`** options (at least one user *must* be defined).

Arguments

Table 9.26, “**`fabric:create` Arguments**” describes the command's arguments.

Table 9.26. `fabric:create` Arguments

Argument	Interpretation
`--help`	Displays the online help for this command
`--clean`	Clean local zookeeper cluster and configurations.
`--no-import`	Disable the import of the sample registry data.
`--import-dir`	Directory of files to import into the newly created ensemble.
`-v, --verbose`	Flag to enable verbose output of files being imported.
`-t, --time`	How long to wait (milliseconds) for the ensemble to start up, before trying to import the default data.
`-n`, **`--non-managed`**	Specifies that the container remains unmanaged.
`-p, --profile`	Specifies the profile (or profiles) to use for the ensemble containers in the new fabric. To specify multiple profiles, specify this flag multiple times on the command line—for example, **`--profile foo --profile bar`**.

Argument	Interpretation
-b, --bind-address	Specifies the IP address of the embedded Zookeeper server. For example, this can be a useful option to specify if the container is deployed on a multi-homed host.
--new-user	Create a new user in the new fabric's JAAS realm. Because the **fabric:create** command automatically imports user data from the **etc/users.properties** file, you would only need to specify this option, if the **etc/users.properties** file contains no valid user entries. When using this option, you *must* also specify a password using the **--new-user-password** option.
--new-user-password	Used in combination with the **--new-user** option to specify the new user's password. No default value.
--new-user-role	Used in combination with the **--new-user** option to specify the new user's role. Default is **admin**.
--zookeeper-password	Specifies the Zookeeper password, which is used to access the Zookeeper nodes under the **/fabric/** path. Defaults to the password of the current session user. Subsequently, because the Zookeeper password is cached in the current session, you normally do not need to provide it when executing fabric commands. You can display the Zookeeper password at any time using the **fabric:ensemble-password** command.
--generate-zookeeper-password	Directs Fabric to generate a random Zookeeper password. Subsequently, you can display the Zookeeper password using the **fabric:ensemble-password** command.
--zookeeper-data-dir	The location where ZooKeeper stores the in-memory database snapshots and, unless specified otherwise, the transaction log of updates to the database. Defaults to **data/zookeeper**.
--zookeeper-init-limit	The amount of time, in ticks, to allow followers to connect and sync to a leader. Defaults to 10.
--zookeeper-sync-limit	The amount of time, in ticks, to allow followers to sync with ZooKeeper. Defaults to 5.

Argument	Interpretation
--zookeeper-ticktime	The length of a single tick, which is the basic time unit used by ZooKeeper, as measured in milliseconds. It is used to regulate heartbeats and timeouts. For example, the minimum session timeout is two ticks. Defaults to **2000**.
--zookeeper-server-port	Specifies the IP port number of the ZooKeeper server, on which ZooKeeper listens for incoming connections. Default is 2181.
--zookeeper-purge-interval	Set the interval in hours between triggers of the autopurge task. Must be set to a positive integer (1 and above). The default is 0.
--zookeeper-snap-retain-count	Define the number of recent Snapshots and corresponding transaction logs in the **dataDir** and **dataLogDir** to retain on autopurge. The minimum number is 3.
--wait-for-provisioning	Flag to wait for the initial container provisioning.
-g, --global-resolver	Specifies the global resolver policy, which becomes the default resolver policy applied to all new containers created in this fabric. Possible values are: **localip**, **localhostname**, **publicip**, **publichostname**, **manualip**. The default is **localhostname**.
-r, --resolver	Specifies the local resolver policy. Possible values are: **localip**, **localhostname**, **publicip**, **publichostname**, **manualip**. The default is **localhostname**.
-m, --manual-ip	If you select the **manualip** resolver policy (using either the **--resolver** or **--global-resolver** options), specifies the IP address to use for the resolver.
--min-port	Specifies the minimum port number of the allowed IP port range. Default is **0**.
--max-port	Specifies the maximum port number of the allowed IP port range. Default is **65535**.
--external-git-url	Specifies an external git url.
--external-git-user	Specifies an external git user.

Argument	Interpretation
--external-git-password	Specifies an external git password.
--bootstrap-timeout	How long to wait (in milliseconds) for the initial fabric bootstrap. Default is 120000.
ContainerList	The list of containers to include in the ensemble. An empty list implies the current container.

Examples

Create a fabric and import sample profiles from the ***ESBInstallDir*/fabric/import** directory, as follows:

```
fabric:create --clean
```

Create a fabric *without* imported profiles, as follows:

```
fabric:create --clean --no-import
```

Create a fabric and import profiles from the custom import directory, *CustomImportDir*, as follows:

```
fabric:create --clean --import-dir CustomImportDir
```

Re-create a fabric such that the containers, **reg1**, **reg2**, and **reg3**, are now included in the registry ensemble (an ensemble must consist of an odd number of containers):

```
fabric:create reg1 reg2 reg3
```

In this case, the contents of the Zookeeper registry are preserved and the ensemble is expanded to include the specified containers.

Related topics

For more details about resolver policies, see:

- Section 9.19, “fabric:container-resolver-list”.
- Section 9.20, “fabric:container-resolver-set”.

9.27. FABRIC:ENSEMBLE-ADD

Abstract

extend the current Fabric Ensemble by converting the specified containers into Fuse Servers

Synopsis

fabric:ensemble-add [--help] { *ContainerList* }

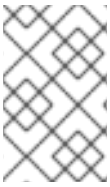
Description

Because the total number of containers in an ensemble must always be odd, you should add an even number of containers.

For example, consider a fabric consisting of three containers—**root1**, **root2**, and **root3**—where **root1** is an Fuse Server and **root2** and **root3** are ordinary Fabric Containers. You can now add **root2** and **root3** to the current ensemble by entering the following console command:

```
fabric:ensemble-add root2 root3
```

Normally, it makes sense to have at most one Fabric Server running on each host, so that the specified containers are actually running on remote hosts (hence, it usually does not make sense to add child containers to an ensemble). You do not need to provide any information about where the containers are running, however, because fabric already knows the location of the containers in the fabric.



NOTE

Because the Fabric Ensemble is the key component of Fuse Fabric, changing the ensemble is a critical operation. All data will be preserved and copied to the new Fuse Servers before switching.

Arguments

Table 9.27, "**fabric:ensemble-add** Arguments" describes the command's arguments.

Table 9.27. fabric:ensemble-add Arguments

Argument	Interpretation
--help	Displays the online help for this command
ContainerList	The list of containers to add.

9.28. FABRIC:ENSEMBLE-LIST

Abstract

lists the Fuse Servers in the current Fabric Ensemble

Synopsis

fabric:ensemble-list [--help]

Description

For a complete listing of *all* the containers in the fabric, use **fabric:container-list** instead.

Arguments

Table 9.28, "`fabric:ensemble-list` Arguments" describes the command's arguments.

Table 9.28. `fabric:ensemble-list` Arguments

Argument	Interpretation
`--help`	Displays the online help for this command

9.29. FABRIC:ENSEMBLE-PASSWORD

Abstract

Displays or modifies the ensemble password

Synopsis

`fabric:ensemble-password` [--help] [--commit] [*NewPassword*]

Description

The ensemble password protects access to the Zookeeper nodes under the **`/fabric/`** path, which contains critical configuration data for the fabric. This command can be used either to display the current ensemble password or to change the ensemble password to a new value.

To display the current ensemble password, invoke the command without any arguments:

```
JBossFuse:karaf@root> fabric:ensemble-password
```

To change the ensemble password requires a two-step process. First, specify the new password as follows:

```
JBossFuse:karaf@root> fabric:ensemble-password mysecret
Updating the password...

Password updated. Please wait a little while for the new password to
get delivered as a config update to all the fabric nodes. Once, the
nodes all updated (nodes must be online), please run:

  fabric:ensemble-password --commit

JBossFuse:karaf@root> fabric:ensemble-password --commit
Only the current password is allowed access to fabric now.

JBossFuse:karaf@root>
```

After the new password has propagated to all of the ensemble nodes, commit the change as follows:

```
JBossFuse:karaf@root> fabric:ensemble-password --commit
```

Arguments

Table 9.29, "`fabric:ensemble-password` Arguments" describes the command's arguments.

Table 9.29. `fabric:ensemble-password` Arguments

Argument	Interpretation
`NewPassword`	Specifies the new ZooKeeper password for the ensemble.
`--help`	Displays the online help for this command
`--commit`	Displays the online help for this command

9.30. FABRIC:ENSEMBLE-REMOVE

Abstract

remove the specified containers from the current ensemble

Synopsis

`fabric:ensemble-remove` [--help] { *ContainerList* }

Description

Re-create the current ensemble, excluding the specified containers from the ensemble. All containers are switched to this new ensemble.

NOTE

Because the Fabric Ensemble is the key component of Fuse Fabric, changing the ensemble is a critical operation. All data will be preserved and copied to the new ensemble before switching.

Arguments

Table 9.30, "`fabric:ensemble-remove` Arguments" describes the command's arguments.

Table 9.30. `fabric:ensemble-remove` Arguments

Argument	Interpretation
`--help`	Displays the online help for this command
`ContainerList`	The list of containers to remove. Must be an even number of containers.

9.31. FABRIC:JOIN

Abstract

join a container to an existing fabric

Synopsis

`fabric:join` [--help] [[-f] | [--force]] [[-p] | [--profile]***`Profile`***] [[-n] | [--non-managed]] [--zookeeper-password *zooPassword*] [[-r] | [--resolver]***`policy`***] [[-m] | [--manual-ip]***`ipAddress`***] [--min-port *port*] [--max-port *port*] [[-b] | [--bind-address]***`BindAddress`***] [[-v] | [--version]***`Version`***] *URL* [*ContainerName*]

Description

The **`fabric:join`** command can be used to join a standalone container to fabric.

Arguments

Table 9.31, "**`fabric:join`** Arguments" describes the command's arguments.

Table 9.31. `fabric:join` Arguments

Argument	Interpretation
`--help`	Displays the online help for this command.
`-f`, **`--force`**	Forces the provided container name to be used.
`-p`, **`--profile`**	Specifies the profile to associate with the container after it joins the fabric. The **`fabric`** profile, which installs the Fabric Agent, is automatically assigned to all managed containers.
`-n`, **`--non-managed`**	Registers the container with the fabric's ensemble, but does not install a Fabric Agent into the container. The container's configuration is not managed by the fabric and continues to behave like a standalone container except that it can be discovered through the fabric's ensemble.
`--zookeeper-password`	The ensemble password for the fabric that you are trying to join. If you do not specify this option, you will be prompted to enter the password.
`-r, --resolver`	Specifies the local resolver policy. Possible values are: **`localip`**, **`localhostname`**, **`publicip`**, **`publichostname`**, **`manualip`**. The default is **`localhostname`**.

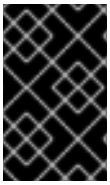
Argument	Interpretation
-m, --manual-ip	If you select the **manualip** resolver policy (using the **--resolver** option), specifies the IP address to use for the resolver.
--min-port	Specifies the minimum port number of the allowed IP port range. Default is **0**.
--max-port	Specifies the maximum port number of the allowed IP port range. Default is **65535**.
-b, **--bind-address**	Specifies the IP address of the embedded Zookeeper server (if there is one). For example, this can be a useful option to specify if the container is deployed on a multi-homed host. Note that it only makes sense to specify this option if the current container deploys a Zookeeper server (for example, if the current container is an ensemble container).
-v, **--version**	Specifies the version of the container after joining the fabric. Default is **1.0**.
URL	Specifies the URL of one of the Fabric Servers, specified in the format ***Host*[:*Port*]**. The ***Port*** value defaults to **2181**.
ContainerName	Specifies a unique name for the container to use when joining the fabric. By default, the value of the **karaf.name** property from the **etc/system.properties** file is used.

Examples

The following command will add a standalone container to a fabric as a managed container:

```
fabric:join myhostA ishmael
```

Where **myhostA** is the hostname of a Fabric Server (you must connect to a Fabric Server, not an ordinary fabric container) and the container is assigned the name **ishmael**. You will be prompted to enter the fabric's Zookeeper password.

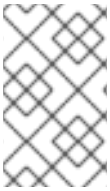


IMPORTANT

If the container being added to a fabric is assigned the same name as a container that is already a part of the fabric, the original container will be reset to have the same settings as the new container.

WARNING

If no container name is specified as part of the command, the command will use the value of the **karaf.name** property from the **etc/system.properties** file. The default setting for this property is **root**. To avoid conflicts, you should either specify a container name or change the value of the **karaf.name** property.



NOTE

The container where you run the **fabric:join** command *must* be a standalone container. It is an error to invoke **fabric:join** in a container that is already part of a fabric.

To make sure that the container starts up with a specific profile, you use the **-p** argument as follows:

```
fabric:join -p whaler myhostA ishmael
```

The container **ishmael** is assigned the profile, **whaler**, when it joins the fabric.

If you want to be able to configure the container manually, but take advantage of the fabric's discovery features, you can add the container as a non-managed container using the following command:

```
fabric:join -n myhostA ishmael
```

9.32. FABRIC:MQ-CREATE

Abstract

create a new broker profile

Synopsis

fabric:mq-create [--help] [--group *groupName*] [--network *brokerGroup*] [--networks-username *user*] [--networks-password *password*] [--create-container *containerID*, ...] [--assign-container *containerID*, ...] [--config *configFile*] [--data *dataDir*] [--kind *brokerKind*] [--replicas *num*] [--port *port*] [--profile *profile*] [--parent-profile *parentProfile*] [--client-profile *clientProfile*] [--client-parent-profile *clientParentProfile*] [--minimum-instances *num*] [[--property] | [-D]*prop*] [--jmx-user *jmxUser*] [--jmx-password *jmxPassword*] [--jvm-opts *jvmOpts*] [--version *version*] { *name* }

Arguments

Table 9.32, "**fabric:mq-create** Arguments" describes the command's arguments.

Table 9.32. fabric:mq-create Arguments

Argument	Description
--help	Displays the online help for this command.
--group *groupName*	Specifies the name of the group to which brokers using this profile are assigned. By default brokers are assigned to the **default** group.
--network *brokerGroup*	Specifies a broker group to which brokers using this profile will establish network connections to form a network of brokers. To specify multiple broker groups, specify this flag multiple times on the command line—for example, **--network GroupA --network GroupB**.
--networks *brokerGroup*	**Deprecated.**
--networks-username	Specifies the username part of the credentials that are used to connect to the broker networks specified by the **--network** option.
--networks-password	Specifies the password part of the credentials that are used to connect to the broker networks specified by the **--network** option.
--create-container *containerID*,...	Specifies a comma separated list of child containers to create using the new profile. The new containers will be children of the container from which the command is executed.
--assign-container *containerID*,...	Specifies a comma separated list of containers to which the new profile will be deployed.
--config *configFile*	Specifies the ensemble path of the XML configuration template used by the profile. The path will have the syntax **/fabric/configs/versions/*version*/profiles/*profile*/*config*.xml**.
--data *dataDir*	Specifies the path, relative to the container, for storing the persistence data for a broker using the profile.
--kind *brokerKind*	The kind of broker to create.
--replicas *num*	Number of replicas required for replicated brokers (which typically use a parent-profile of mq-replicated profile).

Argument	Description
--port *port*	Port number for a transport connector, specified using the syntax **--port *transportConnectorName=port***, where ***transportConnectorName*** is the value of the **name** attribute from a **transportConnector** element in the configuration template. To specify multiple ports, specify this flag multiple times on the command line—for example, **--port openwire=11111 --port mqtt-ssl=4321**.
--ports *port*	**Deprecated.**
--no-ssl	Disables support for SSL/TLS protocol.
--profile *profile*	The profile name to create or update, if defining **N+1** broker groups. Defaults to **mq-broker-$GROUP.$NAME**.
--parent-profile *parentProfile*	The parent profile to extend.
--client-profile *clientProfile*	The profile name for clients to use to connect to the broker group. Defaults to **mq-client-$GROUP**.
--client-parent-profile *clientParentProfile*	The parent profile used for the client-profile for clients connecting to the broker group. Defaults to **default**.
--minimum-instances *num*	Minimum number of containers required of this broker's profile.
--property, -D	Additional properties to define in the profile. To specify multiple properties, specify this flag multiple times on the command line—for example, **--property keyA=valA --property keyB=valB**.
--jmx-user	The JMX username for logging on to the parent's JMX port.
--jmx-password	The JMX password for logging on to the parent's JMX port.
--jvm-opts	Specify options to pass to the container's JVM.
--version *version*	Specifies the version into which the profile is stored. Defaults to the current default version.
name	Specifies the name of the new broker profile.

Examples

To create a new broker profile with the name **myBrokerProfile** that uses the XML template file **myConfigTemplate.xml** use the command:

```
fabric:mq-create --config /fabric/configs/versions/1.0/profiles/mq-
base/myConfigTemplate.xml myBrokerProfile
```

To create a new broker profile and create a new container using the new profile use the command:

```
fabric:mq-create --config /fabric/configs/versions/1.0/profiles/mq-
base/myConfigTemplate.xml --create-container broker1 myBrokerProfile
```

To create a new broker profile and associate it with an existing container use the command:

```
fabric:mq-create --config /fabric/configs/versions/1.0/profiles/mq-
base/myConfigTemplate.xml --assign-container container1 myBrokerProfile
```

SSL/TLS support

For SSL/TLS support, the Java command-line utility, **keytool**, *must* be available on your PATH. To disable SSL/TLS support, specify the **--no-ssl** option.

9.33. FABRIC:PATCH-APPLY

Abstract

Apply a patch to the specified version or versions.

Synopsis

fabric:patch-apply [--help] [[-u] | [--username]***User***] [[-p] | [--password]***Password***] [--version *Version*] [--all-versions] { *URL* }

Arguments

Table 9.33, “fabric:patch-apply Arguments” describes the command's arguments.

Table 9.33. fabric:patch-apply Arguments

Argument	Interpretation
--help	Displays the online help for this command.
-u, --username	Remote user name.
-p, --password	Remote user password.

Argument	Interpretation
--version	Applies the patch to the specified version. Defaults to the current default version.
--all-versions	Applies the patch to all versions.
URL	Specifies the URL from which the patch is downloaded.

9.34. FABRIC:PROFILE-CHANGE-PARENTS

Abstract

replace the profile's parents with the specified list of parents (where the parents are specified as a space-separated list)

Synopsis

fabric:profile-change-parents [--help] [--version ***version***] { *Name* } { *ParentList* }

Arguments

Table 9.34, "**fabric:profile-change-parents** Arguments" describes the command's arguments.

Table 9.34. fabric:profile-change-parents Arguments

Argument	Interpretation
--help	Displays the online help for this command
--version	The profile version. Defaults to the current default version (use **version-list** to find the current default).
Name	*(Required)* Name of the profile.
ParentList	*(Required)* The list of new parent profiles.

9.35. FABRIC:PROFILE-COPY, PROFILE-COPY

Abstract

copies the specified version of the source profile (where the version defaults to the current default version)

Synopsis

fabric:profile-copy [--help] [--version *Version*] [-f] [--force] { *SourceProfile* } { *TargetProfile* }

Arguments

Table 9.35, "**fabric:profile-copy** Arguments" describes the command's arguments.

Table 9.35. fabric:profile-copy Arguments

Argument	Interpretation
--help	Displays the online help for this command.
--version	The profile version to copy. Defaults to the current default version.
-f, --force	Flag to allow overwriting the target profile (if it already exists).
SourceProfile	The name of the profile to copy.
TargetProfile	The name of the newly created copy.

9.36. FABRIC:PROFILE-CREATE

Abstract

create a new profile with the specified name and version

Synopsis

fabric:profile-create [--help] [--version ***version***] [--parent ***parent***] { *Name* }

Description

The new profile is created *only for the version you specify*(or the current default version). If you want to create a profile for every version, you must invoke **fabric:profile-create** separately for each version (use **fabric:version-list** to list all versions).

The newly created profile is initially empty, apart from the settings inherited from the parent profiles. To add settings to the new profile, use the **fabric:profile-edit** command.

For example, to add the new profile, **test**, which has the current default version and inherits from the parent profiles, **mq** and **camel**, enter the following console command:

```
fabric:profile-create --parent mq --parent camel test
```

Arguments

Table 9.36, "`fabric:profile-create` Arguments" describes the command's arguments.

Table 9.36. `fabric:profile-create` Arguments

Argument	Interpretation
`--help`	Displays the online help for this command
`--version`	The profile version. Defaults to the current default version (use `version-list` to find the current default).
`--parent`	Optionally specifies one or multiple parent profiles. To specify multiple parent profiles, specify this flag multiple times on the command line—for example, `--parent foo --parent bar`.
`--parents`	**Deprecated.**
Name	*(Required)* Name of the new profile.

9.37. FABRIC:PROFILE-DELETE

Abstract

delete the specified version of the specified profile (where the version defaults to the current default version)

Synopsis

`fabric:profile-delete` [--help] [--version ***`version`***] { *Profile* }

Arguments

Table 9.37, "`fabric:profile-delete` Arguments" describes the command's arguments.

Table 9.37. `fabric:profile-delete` Arguments

Argument	Interpretation
`--help`	Displays the online help for this command
`--version`	The profile version to delete. Defaults to the current default version (use `version-list` to find the current default).
Profile	*(Required)* Name of the profile to delete.

9.38. FABRIC:PROFILE-DISPLAY

Abstract

displays information about the specified version of the specified profile (where the version defaults to the current default version)

Synopsis

`fabric:profile-display` [--help] [--version ***`version`***] [[-o] | [--overlay]] { *Profile* }

Arguments

Table 9.38, "**`fabric:profile-display`** Arguments" describes the command's arguments.

Table 9.38. `fabric:profile-display` Arguments

Argument	Interpretation
`--help`	Displays the online help for this command
`--version`	Select a specific profile version. Defaults to the current default version (use **`version-list`** to find the current default).
`-o, --overlay`	Enable overlay. Shows the effective profile settings, taking into account the settings inherited from parent profiles.
`Profile`	*(Required)* The name of the profile.

9.39. FABRIC:PROFILE-EDIT

Abstract

edits the specified version of the specified profile (where the version defaults to the current default version)

Synopsis

`fabric:profile-edit` [--help] [[-p] | [--pid]***`PID`***] [[-r] | [--repository] | [-f] | [--feature] | [-b] | [--bundle] | [-c] | [--config] | [-s] | [--system] | [-o] | [--overrides] | [-l] | [--lib] | [-n] | [--endorsed] | [-x] | [--extension]] [[--set] | [--delete]] [[--append] | [--remove]] [--import-pid] [[--delimiter]***`delim`***] [[--resource]***`ResourceName`***] { *Profile* } [*Version*]

Description

In the specified profile, you can edit different kinds of settings, as follows:

- *Feature repository locations*–to add a feature repository to the profile, enter a command in the following format:

```
fabric:profile-edit --repository RepoURL Profile [Version]
```

For example, to add the **fuse-fabric** feature repository to the profile, enter a command like the following:

```
fabric:profile-edit --repository mvn:io.fabric8/fuse-
fabric/6.3.0.redhat-187/xml/features Profile [Version]
```

To delete repositories, enter a command of the following form:

```
fabric:profile-edit --delete --repository RepoURL Profile [Version]
```

To edit repository locations directly, using a visual text editor, enter the following command:

```
fabric:profile-edit Profile [Version]
```

The visual editor opens, showing the contents of the specified profile's agent properties. To edit the repository settings, add, modify, or delete lines of the following form:

```
repository.ID=URL
```

Where ***ID*** is an arbitrary unique identifier and ***URL*** gives the location of a single feature repository. Only one repository URL can be specified on each line.

- *Features to install*–to add features to the profile, enter a command in the following format:

```
fabric:profile-edit --feature FeatureName Profile [Version]
```

To add multiple features, you can specify the **--feature** flag multiple times in this command. For example, to add the **camel-jetty** and the **camel-quartz** features to the default version of the **sample** profile, enter a command like the following:

```
fabric:profile-edit --feature camel-jetty --feature camel-quartz
sample
```

To delete features, enter a command of the following form:

```
fabric:profile-edit --delete --feature FeatureName Profile [Version]
```

To edit features directly, using the visual text editor, enter the following command:

```
fabric:profile-edit Profile [Version]
```

The visual editor opens, showing the contents of the specified profile's agent properties. To edit the features, add, modify, or delete lines of the following form:

```
feature.ID=FeatureName
```

Where ***ID*** is an arbitrary unique identifier and ***FeatureName*** is the name of a feature.

- *Bundles to install*—to add bundles to the profile, enter a command in the following format:

NOTE

The **fabric:profile-edit** command supports two flag variations for specifying bundles—**--bundle** and **--bundles**. The **--bundles** flag is included for backwards compatibility. Regardless of which variation you use, to specify multiple bundles on the same command line, you must include the flag with each bundle specification.

```
fabric:profile-edit --bundle BundleURL Profile [Version]
```

For example, to add **camel-quartz** bundle to the **sample** profile, enter a command like the following:

```
fabric:profile-edit --bundle mvn:org.apache.camel/camel-
quartz/2.17.0.redhat-630187 sample
```

To delete bundles, enter a command of the following form:

```
fabric:profile-edit --delete --bundle BundleURL Profile [Version]
```

To edit bundles directly, using the visual text editor, enter the following command:

```
fabric:profile-edit Profile [Version]
```

The visual editor opens, showing the contents of the specified profile's agent properties. To edit the bundles, add, modify, or delete lines of the following form:

```
bundle.ID=URL
```

Where ***ID*** is an arbitrary unique identifier and ***URL*** specifies the bundle's location.

- *Configuration settings for the OSGi Config Admin service*—to modify or create a configuration setting from the OSGi Config Admin service, enter a command in the following format:

```
fabric:profile-edit --pid PID/Property=Value Profile [Version]
```

Where *PID* is a persistent ID, which is used in the context of the OSGi Config Admin service to identify a collection of related properties. For example, to change the value of the secure HTTPS port used by the Jetty server in the **sample** profile, you could edit the **org.osgi.service.http.port.secure** property from the **org.ops4j.pax.web** PID using a command like the following:

```
fabric:profile-edit --pid
org.ops4j.pax.web/org.osgi.service.http.port.secure=8553 sample
```

To delete a property, enter a command of the following form:

```
fabric:profile-edit --delete --pid PID/Property Profile [Version]
```

If the value of the PID property has the form of a comma-separated list, you can use the **--**

append option and the **--remove** option to manipulate the list value. For example:

```
fabric:profile-edit --pid org.example.foo/my.prop=a Profile
[Version]
fabric:profile-edit --append --pid org.example.foo/my.prop=b Profile
[Version]
fabric:profile-edit --append --pid org.example.foo/my.prop=c Profile
[Version]
fabric:display Profile
...
PID: org.example.foo
  my.prop a,b,c
```

To edit OSGi Config Admin settings directly, using the visual text editor, enter the following command:

```
fabric:profile-edit --pid PID Profile [Version]
```

The visual editor opens, showing the contents of the specified profile's ***PID*.properties** file (which is actually stored in the ZooKeeper registry). To edit the properties, add, modify, or delete lines of the following form:

```
Property=Value
```

- *Property settings from etc/config.properties*—to modify or create a Java system property in the container's **etc/config.properties** file (which affects the container), enter a command in the following format:

```
fabric:profile-edit --config Property=Value Profile [Version]
```

For example, to change the value of the **karaf.startlevel.bundle** Java system property in **config.properties**, you would enter a command like the following:

```
fabric:profile-edit --config karaf.startlevel.bundle=80 Profile
[Version]
```

To delete a Java system property from **config.properties**, enter a command of the following form:

```
fabric:profile-edit --delete --config Property Profile [Version]
```

If the value of the configuration property has the form of a comma-separated list, you can use the **--append** option and the **--remove** option to manipulate the list value. For example:

```
fabric:profile-edit --config my.prop=a Profile [Version]
fabric:profile-edit --append --config my.prop=b Profile [Version]
fabric:profile-edit --append --config my.prop=c Profile [Version]
fabric:display Profile
...
Config Properties :
          my.prop =     a,
                b,
                c
```

-

To edit the Java system properties directly, using the visual text editor, enter the following command:

```
fabric:profile-edit Profile [Version]
```

The visual editor opens, showing the contents of the specified profile's agent properties. To edit the Java system properties (analogous to **etc/config.properties**), add, modify, or delete lines of the following form:

```
config.Property=Value
```

- *Property settings from etc/system.properties*—to modify or create a Java system property in the container's **etc/system.properties** file (which affects bundles deployed in the container), enter a command in the following format:

```
fabric:profile-edit --system Property=Value Profile [Version]
```

For example, to change the default port for the OSGi HTTP service, you would enter a command like the following:

```
fabric:profile-edit --system org.osgi.service.http.port=8181 Profile
[Version]
```

If the system property, ***Property***, is already set at the JVM level (for example, through the **--jvm-opts** option to the **fabric:container-create** command), the preceding **fabric:profile-edit** command *will not override the JVM level setting*. If you want to override the JVM level setting, you must indicate this explicitly by adding the **karaf.override** prefix to the property name, ***Property***—for example:

```
fabric:profile-edit --system karaf.override.Property=Value Profile
[Version]
```

To delete a Java system property from **system.properties**, enter a command of the following form:

```
fabric:profile-edit --delete --system Property Profile [Version]
```

If the value of the system property has the form of a comma-separated list, you can use the **--append** option and the **--remove** option to manipulate the list value. For example:

```
fabric:profile-edit --system my.prop=a Profile [Version]
fabric:profile-edit --append --system my.prop=b Profile [Version]
fabric:profile-edit --append --system my.prop=c Profile [Version]
fabric:display Profile
...
System Properties :
          my.prop =     a,
                b,
                c
```

To edit the Java system properties directly, using the visual text editor, enter the following command:

```
fabric:profile-edit Profile [Version]
```

The visual editor opens, showing the contents of the specified profile's agent properties. To edit the Java system properties (analogous to **etc/system.properties**), add, modify, or delete lines of the following form:

```
system.Property=Value
```

If you want to ensure that this setting overrides any JVM level setting, set the system property as follows:

```
system.karaf.override.Property=Value
```

IMPORTANT

Any modifications you make to a profile using **fabric:profile-edit** are *immediately* propagated to the containers that use that profile. This is not the recommended way to edit profiles, however: if you change multiple settings in the profile, you could potentially put the affected containers into an inconsistent state. To guarantee atomicity, it is better to use the **fabric:profile-edit** command in combination with the **fabric:container-upgrade** command—see fabric:container-upgrade.

Enclosing an Option Value in Quotes

The Karaf shell strips double quotes from an option by default. Hence, to enclose an option value in double quotes, it is necessary to enclose the whole setting in double quotes and to escape the quotes around the option value. For example, to define the system property setting, **http.nonProxyHosts="myserver1|myserver2"**, on the **default** profile, you would use the following command:

```
fabric:profile-edit --system
"karaf.override.http.nonProxyHosts=\"myserver1|myserver2\"" default
```

Where the **karaf.override** prefix is prepended to the property name, because **http.nonProxyHosts** is already set at the JVM level and needs to be overridden.

Arguments

Table 9.39, "**fabric:profile-edit** Arguments" describes the command's arguments.

Table 9.39. fabric:profile-edit Arguments

Argument	Interpretation
--help	Displays the online help for this command
-p, --pid	Edit an OSGi configuration property, specified in the format *PID/Property*. To specify multiple values, specify this flag multiple times on the command line —for example, **--pid *PID/Property=Value*** **--pid *PID/Property=Value***.

Argument	Interpretation
-r, --repository	Add (or delete) a feature repository. To specify multiple values, specify this flag multiple times on the command line—for example, **--repository foo --repository bar**.
--repositories	**Deprecated.**
-f, --feature	Add (or delete) a feature. To specify multiple values, specify this flag multiple times on the command line—for example, **--feature foo --feature bar**.
--features	**Deprecated.**
-b, --bundle	Add (or delete) a bundle. To specify multiple values, specify this flag multiple times on the command line—for example, **--bundle foo --bundle bar**.
--bundles	**Deprecated.**
-c, --config	Edit the Java system properties that affect the container (analogous to editing **etc/config.properties** in a root container). To edit multiple configuration properties, specify this flag multiple times on the command line—for example, **--config *Property=Value* --config *Property=Value***.
-s, --system	Edit the Java system properties that affect installed bundles (analogous to editing **etc/system.properties** in a root container). To edit multiple system properties, specify this flag multiple times on the command line—for example, **--system *Property=Value* --system *Property=Value***.
-o, --overrides	Add (or delete) a bundle override. A bundle override can be used to override the bundle version installed by a feature. For example, if a feature installs version 1.0.0 of a particular bundle, you could use a bundle override to install version 1.0.1 of the bundle instead. To specify multiple values, specify this flag multiple times on the command line—for example, **--overrides *BundleURL* --overrides *BundleURL***.
-l, --lib	Add (or delete) a library. To specify multiple values, specify this flag multiple times on the command line—for example, **--lib *LibURL* --lib *LibURL***.

Argument	Interpretation
--libs	**Deprecated.**
-n, --endorsed	Add (or delete) an endorsed library. To specify multiple values, specify this flag multiple times on the command line—for example, **--endorsed *LibURL* --endorsed *LibURL***.
-x, --extension	Add (or delete) an extension library. To specify multiple values, specify this flag multiple times on the command line—for example, **--extension *LibURL* --extension *LibURL***.
--set	Set or create values (selected by default).
--delete	Delete values.
--append	When editing list values, append the specified value to the list. Can only be used in combination with the **--config**, **--system**, and **--pid** options.
--remove	When editing list values, remove the specified value from the list. Can only be used in combination with the **--config**, **--system**, and **--pid** options.
--delimiter	Specifies the delimiter to use in combination with the **--append** and **--remove** options. Default is **,** (comma).
--resource	When editing with the visual text editor, specifies the name of the resource to edit.
-i, --import-pid	Imports the PIDs that are edited, from local OSGi Config Admin.
Profile	*(Required)* Name of the profile to edit.
Version	Version of the profile to edit. Defaults to the current default version (use **version-list** to find the current default).

9.40. FABRIC:PROFILE-EXPORT

Abstract

exports a profile to the specified location

Synopsis

`fabric:profile-export` { *ProfileDestination* }

Description

The **`fabric:profile-export`** command allows you to export profiles to files.

The default export location is the **`fabric/export`** folder under the karaf home directory. To change the default location, specify the path as an argument:

```
fabric:profile-export /path/to/profile/location
```

Arguments

Table 9.40, "**`fabric:profile-export`** Arguments" describes the command's arguments.

Table 9.40. `fabric:profile-export` Arguments

Argument	Interpretation
`ProfileDestination`	The location to which the profile will be exported.

9.41. FABRIC:PROFILE-IMPORT

Abstract

imports a profile from the specified location

Synopsis

`fabric:profile-import` { *ProfileSource* }

Description

The **`fabric:profile-import`** command allows you to import profiles stored in the form of ZIP files.

The ZIP files can be accessed using a URL:

```
fabric:profile-import /path/to/profile/location/profiles.zip
```

Or you can use maven coordinates instead of a URL:

```
fabric:profile-import mvn:com.foo/location/1.0/profile/profiles.zip
```

Arguments

Table 9.41, "**`fabric:profile-import`** Arguments" describes the command's arguments.

Table 9.41. `fabric:profile-import` Arguments

Argument	Interpretation
ProfileSource	The location of the profile to be imported.

9.42. FABRIC:PROFILE-LIST

Abstract

lists all profiles that belong to the specified version (where the version defaults to the current default version)

Synopsis

fabric:profile-list [--help] [--version ***version***] [--hidden]

Description

Arguments

Table 9.42, "**fabric:profile-list** Arguments" describes the command's arguments.

Table 9.42. fabric:profile-list Arguments

Argument	Interpretation
--help	Displays the online help for this command
--version	Specifies the version of the profiles to list. Defaults to the current default version (use **version-list** to find the current default).
--hidden	Shows hidden profiles.

9.43. FABRIC:PROFILE-REFRESH, PROFILE-REFRESH

Abstract

performs a change to the profile, that triggers the deployment agent. It's intended to be used for scanning for snapshot changes

Synopsis

fabric:profile-refresh [--help] { *Profile* } [*Version*]

Arguments

Table 9.43, "`fabric:profile-refresh` Arguments" describes the command's arguments.

Table 9.43. `fabric:profile-refresh` Arguments

Argument	Interpretation
`--help`	Displays the online help for this command.
`Profile`	The profile to refresh.
`Version`	The profile version to refresh. Defaults to the current default version.

9.44. FABRIC:PROFILE-RENAME, PROFILE-RENAME

Abstract

rename the specified version of the source profile (where the version defaults to the current default version)

Synopsis

`fabric:profile-rename` [--help] [--version *Version*] [-f] [--force] { *OldName* } { *NewName* }

Arguments

Table 9.44, "`fabric:profile-rename` Arguments" describes the command's arguments.

Table 9.44. `fabric:profile-rename` Arguments

Argument	Interpretation
`--help`	Displays the online help for this command.
`--version`	The profile version to rename. Defaults to the current default version.
`-f, --force`	Flag to allow replacing the target profile (if it already exists).
`OldName`	The current name of the profile.
`NewName`	The new name of the profile.

9.45. FABRIC:REQUIRE-PROFILE-DELETE

Abstract

deletes requirements on the specified profile

Synopsis

fabric:require-profile-delete [--help] { *Profile* }

Arguments

Table 9.45, "**fabric:require-profile-delete** Arguments" describes the command's arguments.

Table 9.45. fabric:require-profile-delete Arguments

Argument	Interpretation
--help	Displays the online help for this command
Profile	A profile ID.

9.46. FABRIC:REQUIRE-PROFILE-LIST

Abstract

lists all profile requirements in the current fabric

Synopsis

fabric:require-profile-list [--help]

Description

For example, if both the **example-camel** profile and the **example-cxf** profile have requirements set, you could see output like the following:

```
karaf@root> fabric:require-profile-list
[profile]                                [# minimum]    [# maximum]
[depends on]
example-camel                            2              4
example-cxf                              2              4
```

Arguments

Table 9.46, "**fabric:require-profile-list** Arguments" describes the command's arguments.

Table 9.46. fabric:require-profile-list Arguments

Argument	Interpretation
--help	Displays the online help for this command

9.47. FABRIC:REQUIRE-PROFILE-SET

Abstract

associates requirements with the specified profile

Synopsis

`fabric:require-profile-set` [--help] [--minimum ***`MinInstance`***] [--maximum ***`MaxInstance`***] [--dependsOn ***`Dependency`***] { *Profile* }

Description

Requirements associated with a profile are used to assess the health of the current fabric. Profile requirements are entirely passive. For example, if the number of running instances of a profile is less than the minimum or greater than the maximum, monitoring tools can be configured to indicate a problem or to trigger an alert. Otherwise, the requirements have no effect on the fabric.

In Fuse IDE a green/red bar indicates what proportion of the required profile instances are currently running in the fabric.

For example, to require a range of 2 to 4 running instances of the **`example-camel`** profile, you would enter the following command:

```
karaf@root> require-profile-set --minimum 2 --maximum 4 example-camel
```

Arguments

Table 9.47, "**`fabric:require-profile-set`** Arguments" describes the command's arguments.

Table 9.47. `fabric:require-profile-set` Arguments

Argument	Interpretation
`--help`	Displays the online help for this command
`--minimum`	The minimum number of instances of this profile expected to be running in the fabric.
`--maximum`	The maximum number of instances of this profile expected to be running in the fabric.
`--dependsOn`	The profile IDs that must be provisioned before this profile. To specify multiple profile IDs, specify this flag multiple times on the command line—for example, **`--dependsOn foo --dependsOn bar`**.
`Profile`	A profile ID.

9.48. FABRIC:STATUS

Abstract

displays the current status of the fabric, based on the configured profile requirements

Synopsis

fabric:status [--help]

Description

This command summarizes the health of the fabric, based on requirements previously configured by the **fabric:require-profile-set** command. For example, if you configured the **example-camel** profile to require a minimum of two instances and a maximum of four instances, and there is currently only one instance running, the **example-camel** profile would get a health rating of 50%.

The **fabric:status** command produces output like the following:

```
karaf@root> fabric:status
[profile]                                [instances]    [health]
cloud                                    1              100%
example-camel                            0              0%
example-cxf                              0              0%
fabric                                   1              100%
fabric-ensemble-0000-1                   1              100%
```

Arguments

Table 9.48, "**fabric:status** Arguments" describes the command's arguments.

Table 9.48. fabric:status Arguments

Argument	Interpretation
--help	Displays the online help for this command

Related topics

For more details, see:

- Section 9.47, "fabric:require-profile-set"
- Section 9.46, "fabric:require-profile-list"
- Section 9.45, "fabric:require-profile-delete"

9.49. FABRIC:VERSION-CREATE

Abstract

create a new version

Synopsis

fabric:version-create [--help] [--parent ***parentVersion***] { *Version* }

Description

Create a new version, by default copying all of the profiles *from the current latest version*into the new version. You can specify which version to copy the profiles from using the **--parent** option. Using the **--description** option, you can add a note to describe the new version. If no version is specified, the command creates a new minor version by default. For example:

```
karaf@root> fabric:version-list
[version]       [default] [# containers]
1.0             true      1
karaf@root> fabric:version-create --description "Test version"
Created version: 1.1 as copy of: 1.0
karaf@root> fabric:version-list
[version]       [default] [# containers]  [description]
1.0             true      1
1.1             false     0               Test version
```

NOTE

Be aware that Fabric8 Karaf does not handle leading and trailing zeros in version numbers. Numbers such as 1.10, 01.1, 1.1 will all be interpreted as 1.1. When specifying version numbers, any numbers with leading or trailing zeros must be enclosed in double quotes. For example, "1.10". This will force the number to be interpreted as presented.

Arguments

Table 9.49, "**fabric:version-create** Arguments" describes the command's arguments.

Table 9.49. fabric:version-create Arguments

Argument	Interpretation
--help	Displays the online help for this command
--default	Set the created version to be the new default version.
--description	Add a description of the newly created version. Enclose the text within double quotes.
--parent	The parent version. By default, uses the latest version as the parent.

Argument	Interpretation
Version	The new version to create. If not specified, defaults to the next minor version.

9.50. FABRIC:VERSION-DELETE

Abstract

delete the specified version

Synopsis

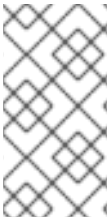
`fabric:version-delete` [--help] { *Version* }

Description

Delete the specified version.

WARNING

This command also deletes all of the profile data associated with the deleted version.



NOTE

Be aware that Karaf does not handle leading and trailing zeros in version numbers. Numbers such as 1.10, 01.1, 1.1 will all be interpreted as 1.1. When specifying version numbers, any numbers with leading or trailing zeros must be enclosed in double quotes. For example, "1.10". This will force the number to be interpreted as presented.

Arguments

Table 9.50, "**`fabric:version-delete`** Arguments" describes the command's arguments.

Table 9.50. `fabric:version-delete` Arguments

Argument	Interpretation
`--help`	Displays the online help for this command
Version	*(Required)* The version to delete.

9.51. FABRIC:VERSION-LIST

Abstract

lists the existing versions

Synopsis

`fabric:version-list` [--help]

Description

For example:

```
karaf@root> fabric:version-list
[version]       [default] [# containers]
1.0             true      1
1.1             false     0
```

Arguments

Table 9.51, "**`fabric:version-list`** Arguments" describes the command's arguments.

Table 9.51. `fabric:version-list` Arguments

Argument	Interpretation
`--help`	Displays the online help for this command

9.52. FABRIC:VERSION-SET-DEFAULT

Abstract

set the new default version (must be one of the existing versions)

Synopsis

`fabric:version-set-default` [--help] { *Version* }

Description

Many of the fabric console commands work with a default version. For example, when you create a new profile with **`fabric:profile-create`**, the new profile is created in the default version by default. The **`fabric:version-set-default`** changes the default version that is used by these commands.

Arguments

Table 9.52, "**`fabric:version-set-default`** Arguments" describes the command's arguments.

Table 9.52. `fabric:version-set-default` Arguments

Argument	Interpretation
--help	Displays the online help for this command
Version	*(Required)* Version number to use as the new default version.

9.53. FABRIC:WATCH

Abstract

Watches and automatically updates bundles

Synopsis

fabric:watch [--help] [--no-upload] [[--start] | [--stop]] [-i *interval*] [--list] [--remove] { *bundles* ...}

Arguments

Table 9.53, "**fabric:watch** Arguments" describes the commands arguments.

Table 9.53. fabric:watch Arguments

Argument	Interpretation
--help	Displays the online help for this command
--no-upload	If specified, updated bundles are not uploaded to the fabric's Maven proxy repository.
--stop	Stop watching the specified bundles
--start	Start watching the specified bundles
-i	Specifies the interval, in milliseconds, to check the bundles.
--list	List the bundles being watched.
--remove	Remove the specified bundles from the watch list.
bundles...	Specifies a whitespace delimited list of bundle URLs or bundle IDs.

IMPORTANT

Only Maven URLs and Maven snapshots are updated automatically. So, if you enter the command, **`fabric:watch *`**, Fabric monitors all bundles whose location matches mvn:* and that have -SNAPSHOT in their URL.

CHAPTER 10. FEATURES CONSOLE COMMANDS

The **features** commands allow you to provision entire applications using the Apache Karaf features facility. Features allow you to provision a collection of bundles using a single name.

Type **features:** then press **Tab** at the **karaf>** prompt to view the available commands.

10.1. FEATURES:ADDURL, ADDURL

Abstract

registers one or more URLs to feature repositories with the container

Synopsis

features:addurl [--help] [[-i] | [--install-all]] { *urls* }

Description

Each feature repository defines one or more features, and each feature is made up of a collection of bundles that work together to provide some functionality. When a feature is loaded, the container loads any required bundles that are not already present into the container and activates them.

Arguments

Table 10.1, "**features:addurl** Arguments" describes the command's arguments.

Table 10.1. features:addurl Arguments

Argument	Interpretation
--help	Displays the online help for this command
-i, --install-all	Install all of the features in the specified feature repository URLs.
urls	One or more repository URLs separated by whitespaces.

10.2. FEATURES:CHOOSEURL, CHOOSEURL

Abstract

registers the feature repository URL for a well known project

Synopsis

features:chooseurl [--help] { *project* } { *version* }

Description

Red Hat JBoss Fuse uses a number of features implemented by well-known projects. To simplify the process of adding their feature repositories, the **chooseurl** command allows you to add a feature repository without knowing its Maven URL. The list of projects supported by **chooseurl** is configured by the org.apache.karaf.features.repos PID.

Arguments

Table 10.2, "`features:chooseurl` Arguments" describes the command's arguments.

Table 10.2. `features:chooseurl` Arguments

Argument	Interpretation
`--help`	Displays the online help for this command
`feature`	Specifies the project name for the feature repository to add.
`version`	Specifies the version of the project's feature repository to add.

10.3. FEATURES:INFO

Abstract

show information about the specified feature with the optionally specified version

Synopsis

`features:info` [--help] [[-c] | [--configuration]] [[-b] | [--bundle]] [[-t] | [--tree]] [[-d] | [--dependency]] { *featureName* } { *version* }

Arguments

Table 10.3, "`features:info` Arguments" describes the command's arguments.

Table 10.3. `features:info` Arguments

Argument	Interpretation
`--help`	Displays the online help for this command
`-c, --configuration`	Display configuration information.
`-b, --bundle`	Display bundle information.
`-t, --tree`	Display feature tree.

Argument	Interpretation
-d, --dependency	Display dependency information.
command	

10.4. FEATURES:INSTALL

Abstract

installs a feature

Synopsis

features:install [--help] { *name* } [*version*]

Arguments

Table 10.4, "**features:install** Arguments" describes the command's arguments.

Table 10.4. features:install Arguments

Argument	Interpretation
--help	Displays the online help for this command
name	The name of the feature to install
version	The version of the feature

10.5. FEATURES:LIST

Abstract

Lists all existing features available from the defined repositories

Synopsis

features:list [--help] [[-i] | [--installed]]

Arguments

Table 10.5, "**features:list** Arguments" describes the command's arguments.

Table 10.5. features:list Arguments

Argument	Interpretation
`--help`	Displays the online help for this command
`-i, --installed`	Displays the list of all installed features

10.6. FEATURES:LISTURL

Abstract

lists the features repository URLs

Synopsis

`features:listurl` [--help] [[-v] | [--validate]] [[-vo] | [--verbose]]

Arguments

Table 10.6, "`features:listurl` Arguments" describes the command's arguments.

Table 10.6. `features:listurl` Arguments

Argument	Interpretation
`--help`	Displays the online help for this command
`-v,--validate`	Validate current version of descriptors.
`-vo,--verbose`	Shows validation output.

10.7. FEATURES:LISTVERSIONS, LISTVERSIONS

Abstract

lists all versions of a feature available from the current feature repositories

Synopsis

`features:listVersions` [--help] { *feature* }

Arguments

Table 10.7, "`features:listVersions` Arguments" describes the command's arguments.

Table 10.7. `features:listVersions` Arguments

Argument	Interpretation
`--help`	Displays the online help for this command
`feature`	Name of a feature.

10.8. FEATURES:REFRESHURL

Abstract

reloads the list of available features from the repositories

Synopsis

`features:refreshUrl` [--help] { *urls* }

Arguments

Table 10.8, "`features:refreshUrl` Arguments" describes the command's arguments.

Table 10.8. `features:refreshUrl` Arguments

Argument	Interpretation
`--help`	Displays the online help for this command
`urls`	The repository URLs to reload (leave empty for all).

10.9. FEATURES:REMOVEURL

Abstract

removes the specified list of repository URLs from the features service

Synopsis

`features:removeUrl` [--help] { *urls* }

Arguments

Table 10.9, "`features:removeUrl` Arguments" describes the command's arguments.

Table 10.9. `features:removeUrl` Arguments

Argument	Interpretation
--help	Displays the online help for this command
-n,--interval	
urls	One or more repository URLs separated by whitespace.

10.10. FEATURES:REMOVEREPOSITORY

Abstract

removes the specified repository from the features service

Synopsis

features:removeRepository [--help] { *repository* }

Arguments

Table 10.10, "**features:removeRepository** Arguments" describes the command's arguments.

Table 10.10. features:removeRepository Arguments

Argument	Interpretation
--help	Displays the online help for this command
repository	The name of a features repository.

10.11. FEATURES:UNINSTALL

Abstract

uninstalls a feature with the specified name and version

Synopsis

features:uninstall [--help] { *features* }

Arguments

Table 10.11, "**features:uninstall** Arguments" describes the command's arguments.

Table 10.11. features:uninstall Arguments

Argument	Interpretation
--help	Displays the online help for this command
features	A space-separated list of features to uninstall, where each feature is specified in the format ***feature*[*/version*]** (that is, the version is optional).

CHAPTER 11. JAAS CONSOLE COMMANDS

The jaas commands are used for editing JAAS realm and user data. Editing a JAAS realm is done in two stages. The changes are placed in a queue until they are applied by executing the **`jaas:update`**.

When editing JAAS settings the commands are used as follows:

1. Start the editing session.

   `jaas:manage`

2. Edit the realm's user data.

   - **`jaas:users`**

     Lists all of the users.

   - **`jaas:useradd`**

     Add a new user.

   - **`jaas:userdel`**

     Delete a user.

   - **`jaas:roleadd`**

     Add a new role to a user.

   - **`jaas:roledel`**

     Delete a role from a user.

   - **`jaas:pending`**

     Lists all of the pending changes that have been made to the realms, but have not been applied to the container.

3. Apply the changes to the JAAS realm and ends the editing session.

   `jaas:update`

You can abandon an editing session using **`jaas:cancel`** before the changes applied to the JAAS settings.

Type **`jaas:`** then press **`Tab`** at the prompt to view the available commands.

11.1. JAAS:CANCEL, CANCEL

Abstract

cancels a JAAS editing session without applying the pending changes

Synopsis

jaas:cancel [--help]

Details

When editing a JAAS realm, the changes are buffered until the editing session is closed. The **jaas:cancel** command clears the buffer without saving the changes and closes the editing session.

You can see a list of the buffered changes using the **jaas:pending** command.

Arguments

Table 11.1, "**jaas:cancel** Arguments" describes the command's arguments.

Table 11.1. jaas:cancel Arguments

Argument	Interpretation
--help	Displays the online help for this command

11.2. JAAS:GROUPADD

Abstract

Add a user to a group.

Synopsis

jaas:groupadd [--help] { *username* } { *group* }

Details

When editing a JAAS realm, add the ***username*** user to the ***group*** group in the current realm.

Arguments

Table 11.2, "**jaas:groupadd** Arguments" describes the command's arguments.

Table 11.2. jaas:groupadd Arguments

Argument	Interpretation
--help	Displays the online help for this command
username	The user to add to the group.
group	The name of the group to which the user is added.

11.3. JAAS:GROUPCREATE

Abstract

Create a new group in the current realm.

Synopsis

`jaas:groupcreate` [--help] { *group* }

Details

When editing a JAAS realm, creates a new user group with the specified name, ***`group`***.

Arguments

Table 11.3, "**`jaas:groupcreate`** Arguments" describes the command's arguments.

Table 11.3. `jaas:groupcreate` Arguments

Argument	Interpretation
`--help`	Displays the online help for this command
`group`	Name of the group to create.

11.4. JAAS:GROUPDEL

Abstract

Remove a user from a group.

Synopsis

`jaas:groupdel` [--help] { *username* } { *group* }

Details

When editing a JAAS realm, remove the ***`username`*** user from the ***`group`*** group in the current realm.

Arguments

Table 11.4, "**`jaas:groupdel`** Arguments" describes the command's arguments.

Table 11.4. `jaas:groupdel` Arguments

Argument	Interpretation
`--help`	Displays the online help for this command

Argument	Interpretation
username	The user to remove from the group.
group	The name of the group from which the user is removed.

11.5. JAAS:GROUPROLEADD

Abstract

Add a role to a group.

Synopsis

jaas:grouproleadd [--help] { *groupname* } { *role* }

Details

When editing a JAAS realm, add the ***role*** role to the ***groupname*** group in the current realm.

Arguments

Table 11.5, "**jaas:grouproleadd** Arguments" describes the command's arguments.

Table 11.5. jaas:grouproleadd Arguments

Argument	Interpretation
--help	Displays the online help for this command
groupname	The name of the group to which you are adding a role.
role	The name of the role to add to the group.

11.6. JAAS:GROUPROLEDEL

Abstract

Remove a role from a group.

Synopsis

jaas:grouproledel [--help] { *groupname* } { *role* }

Details

When editing a JAAS realm, remove the ***role*** role from the ***groupname*** group.

Arguments

Table 11.6, "**jaas:grouproledel** Arguments" describes the command's arguments.

Table 11.6. jaas:grouproledel Arguments

Argument	Interpretation
--help	Displays the online help for this command
groupname	The name of the group from which you are removing a role.
role	The name of the role to remove from the group.

11.7. JAAS:GROUPS

Abstract

List groups in a realm.

Synopsis

jaas:groups [--help]

Details

When editing a JAAS realm, lists the available groups in the current realm.

Arguments

Table 11.7, "**jaas:groups** Arguments" describes the command's arguments.

Table 11.7. jaas:groups Arguments

Argument	Interpretation
--help	Displays the online help for this command

11.8. JAAS:MANAGE, MANAGE

Abstract

opens a JAAS realm for editing

Synopsis

jaas:manage [--help] {[--realm *realm*] | [--index *index*]} [--module *module*] [--force]

Details

The **jaas:manage** command is the first step in editing a JAAS realm. It opens the realm so that calls to the **jaas:*** editing commands will update the selected realm. The edits made by the **jaas:*** editing commands are placed in a buffer associated with the selected realm and not written to the realm until the editing session is ended by the **jaas:update** command.

If you use the **jaas:manage** command before saving the changes to a realm that is open for editing, the changes to the previously open realm are abandoned. The pending edits for the previous realm are cleared without being saved.

While editing a realm you can get a list of the pending changes using the **jaas:pending** command.

Arguments

Table 11.8, "**jaas:manage** Arguments" describes the command's arguments.

Table 11.8. jaas:manage Arguments

Argument	Interpretation
--help	Displays the online help for this command.
--realm	Select the realm to edit by specifying its realm name.
--index	Select the realm to edit by specifying its index.
--module	Specify which of the realm's login modules are to be edited.
--force	Force the switch to the specified realm. If a different realm was already opened for editing its changes are abandoned without being applied.

Examples

You can select the realm to manage *either* by specifying its realm name *or* by specifying its index.

If the installed realm names are all distinct (which you can check using **jaas:realms**), you can identify the realm to manage by specifying the **--realm** option. For example, if the container is a standalone instance (no fabric installed), you can start to edit the **karaf** realm as follows:

```
jaas:manage --realm karaf
```

If the container belongs to a fabric, however, the **fabric-jaas** feature automatically installs another realm named **karaf** at a higher priority, so that it overrides the default **karaf** realm. For example, in a fabric, the **jaas:realms** command returns a list similar to the following:

```
Index Realm Module Class 1 karaf
org.apache.karaf.jaas.modules.properties.PropertiesLoginModule 2 karaf
io.fabric8.jaas.ZookeeperLoginModule
```

In this case, you must identify the realm to manage using the **--index** option, specifying one of the index values from the list. The current active **karaf** realm is the **ZookeeperLoginModule**, which is selected by the index value, **2**, as follows:

```
jaas:manage --index 2
```

11.9. JAAS:PENDING, PENDING

Abstract

lists the changes waiting to be applied to the realm being edited

Synopsis

jaas:pending [--help]

Details

When editing a JAAS realm, the changes are stored in a buffer until the editing session is closed. The **jaas:pending** command shows a list of the changes buffered during the currently open editing session.

The **jaas:update** command saves the changes and closes the editing session.

The **jaas:cancel** command clears the buffer without saving the changes and closes the editing session.

Arguments

Table 11.9, "**jaas:pending** Arguments" describes the command's arguments.

Table 11.9. jaas:pending Arguments

Argument	Interpretation
--help	Displays the online help for this command.

11.10. JAAS:REALMS, REALMS

Abstract

lists the JAAS realms know to the container

Synopsis

jaas:realms [--help]

Arguments

Table 11.10, "**jaas:realms** Arguments" describes the command's arguments.

Table 11.10. jaas:realms Arguments

Argument	Interpretation
--help	Displays the online help for this command.

11.11. JAAS:ROLEADD, ROLEADD

Abstract

adds a role to a user

Synopsis

jaas:roleadd [--help] { *username* } { *role* }

Details

When editing a JAAS realm, the changes are buffered until the editing session is closed. When you add a new role using the **jaas:roleadd** command, the change is stored in the buffer and does not take effect until the editing session is closed.

The **jaas:update** command saves the changes and closes the editing session.

The **jaas:cancel** command clears the buffer without saving the changes and closes the editing session.

Arguments

Table 11.11, "**jaas:roleadd** Arguments" describes the command's arguments.

Table 11.11. jaas:roleadd Arguments

Argument	Interpretation
--help	Displays the online help for this command.
username	Specifies the name of the user to modify.
role	Specifies the role which is appended to the user data.

11.12. JAAS:ROLEDEL, ROLEDEL

Abstract

deletes a role from a user

Synopsis

jaas:roledel [--help] { *username* } { *role* }

Details

When editing a JAAS realm, the changes are buffered until the editing session is closed. When you delete a role using the **jaas:roledel** command, the change is stored in the buffer and does not take effect until the editing session is closed.

The **jaas:update** command saves the changes and closes the editing session.

The **jaas:cancel** command clears the buffer without saving the changes and closes the editing session.

Arguments

Table 11.12, "**jaas:roledel** Arguments" describes the command's arguments.

Table 11.12. jaas:roledel Arguments

Argument	Interpretation
--help	Displays the online help for this command.
username	Specifies the name of the user to modify.
role	Specifies the role which is removed from the user data.

11.13. JAAS:UPDATE

Abstract

applies all pending changes to the JAAS realm and closes the editing session

Synopsis

jaas:update [--help]

Details

When editing a JAAS realm, the changes are buffered until the editing session is closed. The **jaas:update** command saves the buffered changes to the realm and closes the editing session.

You can see a list of the buffered changes using the **jaas:pending** command.

Arguments

Table 11.13, "**jaas:update** Arguments" describes the command's arguments.

Table 11.13. jaas:update Arguments

Argument	Interpretation
--help	Displays the online help for this command.

11.14. JAAS:USERADD, USERADD

Abstract

adds a user to the JAAS realm being edited

Synopsis

jaas:useradd [--help] { *username* } { *password* }

Details

When editing a JAAS realm, the changes are buffered until the editing session is closed. When you add a new user using the **jaas:useradd** command, the change is stored in the buffer and does not take effect until the editing session is closed.

The **jaas:update** command saves the changes and closes the editing session.

The **jaas:cancel** command clears the buffer without saving the changes and closes the editing session.

Arguments

Table 11.14, "**jaas:useradd** Arguments" describes the command's arguments.

Table 11.14. jaas:useradd Arguments

Argument	Interpretation
--help	Displays the online help for this command.
username	Specifies the name of the user to add.

Argument	Interpretation
password	Specifies the password used to authenticate the user.

11.15. JAAS:USERDEL, USERDEL

Abstract

deletes a user from the JAAS realm being edited

Synopsis

jaas:userdel [--help] { *username* }

Details

When editing a JAAS realm, the changes are buffered until the editing session is closed. When you delete a user using the **jaas:useradd** command, the change is stored in the buffer and does not take effect until the editing session is closed.

The **jaas:update** command saves the changes and closes the editing session.

The **jaas:cancel** command clears the buffer without saving the changes and closes the editing session.

Arguments

Table 11.15, "**jaas:userdel** Arguments" describes the command's arguments.

Table 11.15. jaas:userdel Arguments

Argument	Interpretation
--help	Displays the online help for this command.
--without-group-deletion or -w	If the user is the last one of this group, the group will not be deleted.
username	Specifies the name of the user to add.

11.16. JAAS:USERS, USERS

Abstract

lists the users in the JAAS realm being edited

Synopsis

jaas:users [--help]

Arguments

Table 11.16, "**jaas:users** Arguments" describes the command's arguments.

Table 11.16. jaas:users Arguments

Argument	Interpretation
--help	Displays the online help for this command.

CHAPTER 12. JASYPT ENCRYPTION

You can use a command to use Jasypt to encrypt passwords in the console.

12.1. JASYPT:ENCRYPT

Abstract

Encrypts a password

Synopsis

jasypt:encrypt [*password*] [--help]

Details

Use this command to encrypt a plain text password

```
JBossA-MQ:karaf@root> jasypt:encrypt passwordsample
JBossA-MQ:karaf@root>
(ENC)J0qF8Sk5P4KJRdyLs8LzQYRHfyXs7yWT/CyIDPqHqK2wwc6FbgwsTPK+WCsBgMK9
```

Arguments

The command's arguments are as follows:

Table 12.1. jasypt:encrypt Arguments

Argument	Interpretation
password	Enter the password to be encrypted in plain text
--help	Displays the online help for this command

CHAPTER 13. LOG CONSOLE COMMANDS

The log commands allow you to display and change log levels.

Type **log:** then press **Tab** at the prompt to view the available commands.

13.1. LOG:CLEAR

Abstract

clears the log

Synopsis

log:clear [--help]

Arguments

Table 13.1, "**log:clear** Arguments" describes the command's arguments.

Table 13.1. log:clear Arguments

Argument	Interpretation
--help	Displays the online help for this command

13.2. LOG:DISPLAY, DISPLAY, LD

Abstract

displays log entries

Synopsis

log:display [--help] [-p *pattern*] [-n *numLines*] [--no-color]

Arguments

Table 13.2, "**log:display** Arguments" describes the command's arguments.

Table 13.2. log:display Arguments

Argument	Interpretation
--help	Displays the online help for this command
-p	The pattern for formatting the output

Argument	Interpretation
`-n`	The number of entries to display
`--no-color`	Do not use syntax highlighting when displaying the log.

13.3. LOG:DISPLAY-EXCEPTION, DISPLAY-EXCEPTION, LDE

Abstract

displays the last thrown exception from the log

Synopsis

`log:display-exception` [--help]

Arguments

Table 13.3, "`log:display-exception` Arguments" describes the command's arguments.

Table 13.3. `log:display-exception` Arguments

Argument	Interpretation
`--help`	Displays the online help for this command

13.4. LOG:GET, GET

Abstract

shows the log level

Synopsis

`log:get` [--help] { *logger* }

Arguments

Table 13.4, "`log:get` Arguments" describes the command's arguments.

Table 13.4. `log:get` Arguments

Argument	Interpretation
`--help`	Displays the online help for this command

Argument	Interpretation
logger	Specifies the logger name, **ALL**, or **ROOT**. The default is **ROOT**.

13.5. LOG:SET, SET

Abstract

sets the log level

Synopsis

log:set [--help] {[DEFAULT] | [TRACE] | [DEBUG] | [INFO] | [WARN] | [ERROR]} { *logger* }

Arguments

Table 13.5, "**log:set** Arguments" describes the command's arguments.

Table 13.5. log:set Arguments

Argument	Interpretation
--help	Displays the online help for this command
level	Specifies the logging level.
logger	Specifies the logger name. The default is **ROOT**.

13.6. LOG:TAIL

Abstract

continually displays log entries

Synopsis

log:tail [--help] [-p *pattern*] [-n *numLines*] [--no-color]

Arguments

Table 13.6, "**log:tail** Arguments" describes the command's arguments.

Table 13.6. log:tail Arguments

Argument	Interpretation
`--help`	Displays the online help for this command
`-p`	The pattern for formatting the output
`-n`	The number of entries to display
`--no-color`	Do not use syntax highlighting when displaying the log.

CHAPTER 14. OBR CONSOLE COMMANDS

The **obr** commands allow you to access the OSGi Bundle Repository (OBR) Service API.

NOTE

This feature is not installed by default. To install the **obr** shell, run the following command:

```
JBossFuse:karaf@root:> features:install obr
```

Type **obr:** then press **Tab** at the **JBossFuse:karaf@root>** prompt to view the available commands.

14.1. OBR:ADDURL

Abstract

adds a list of repository URLs to the OBR service

Synopsis

obr:addUrl [--help] { *urls* }

Arguments

This command takes the following arguments.

Table 14.1. obr:addUrl Arguments

Argument	Interpretation
--help	Displays the online help for this command
urls	The repository URLs to add to the OBR service, separated by whitespaces

14.2. OBR:DEPLOY

Abstract

deploys a list of bundles using the OBR service

Synopsis

obr:deploy [--help] { *bundles* }

Arguments

This command takes the following arguments.

Table 14.2. `obr:deploy` Arguments

Argument	Interpretation
`--help`	Displays the online help for this command
`bundles`	A list of bundle names to deploy, separated by whitespaces

14.3. OBR:INFO

Abstract

prints information about OBR bundles

Synopsis

`obr:info` [--help] { *bundles* }

Arguments

This command takes the following arguments.

Table 14.3. `obr:info` Arguments

Argument	Interpretation
`--help`	Displays the online help for this command
`bundles`	Specifies the bundles to query for information, separated by whitespaces

14.4. OBR:LIST

Abstract

lists OBR bundles

Synopsis

`obr:list` [--help] { *args* }

Arguments

This command takes the following arguments.

Table 14.4. `obr:list` Arguments

Argument	Interpretation
`--help`	Displays the online help for this command
`args`	The arguments

14.5. OBR:LISTURL

Abstract

displays the repository URLs currently associated with the OBR service

Synopsis

`obr:listUrl` [--help]

Arguments

This command takes the following arguments.

Table 14.5. `obr:listUrl` Arguments

Argument	Interpretation
`--help`	Displays the online help for this command

14.6. OBR:REFRESHURL

Abstract

reloads the repositories to obtain a fresh list of bundles

Synopsis

`obr:refreshUrl` [--help] { *urls* }

Arguments

This command takes the following arguments.

Table 14.6. `obr:refreshUrl` Arguments

Argument	Interpretation
`--help`	Displays the online help for this command

Argument	Interpretation
urls	The repository URLs to refresh (leave empty for all)

14.7. OBR:REMOVEURL

Abstract

removes a list of repository URLs from the OBR service

Synopsis

obr:removeUrl [--help] { *urls* }

Arguments

This command takes the following arguments.

Table 14.7. obr:removeUrl Arguments

Argument	Interpretation
--help	Displays the online help for this command
urls	The repository URLs to remove from the OBR service, separated by whitespace

14.8. OBR:SOURCE

Abstract

downloads the sources for an OBR bundle

Synopsis

obr:source [--help] [-x] { *folder* } { *bundles* }

Arguments

This command takes the following arguments.

Table 14.8. obr:source Arguments

Argument	Interpretation
--help	Displays the online help for this command

Argument	Interpretation
`-x`	Extracts the archive
`folder`	The local directory or folder for storing sources
`bundles`	A list of bundles to download the sources for

14.9. OBR:START

Abstract

deploys and starts a list of bundles using OBR

Synopsis

`obr:start` [--help] { *bundles* }

Arguments

This command takes the following arguments.

Table 14.9. `obr:start` Arguments

Argument	Interpretation
`--help`	Displays the online help for this command
`bundles`	List of bundle names to deploy, separated by whitespaces

CHAPTER 15. OSGI CONSOLE COMMANDS

The osgi commands provide for managing the OSGi runtime. It includes commands for listing OSGi bundles and services and managing bundle lifecycles.

Type **osgi:** then press **Tab** at the prompt to view the available commands.

15.1. OSGI:BUNDLE-LEVEL, BUNDLE-LEVEL

Abstract

gets or sets the start level of a given bundle

Synopsis

osgi:bundle-level [--help] [--force] { *id* } [*startLevel*]

Arguments

Table 15.1, "**osgi:bundle-level** Arguments" describes the command's arguments.

Table 15.1. osgi:bundle-level Arguments

Argument	Interpretation
--help	Displays the online help for this command.
--force	Forces the command to execute.
id	Specifies the id for the bundle.
startLevel	Specifies the new start level for the bundle.

15.2. OSGI:BUNDLE-SERVICES, BUNDLE-SERVICES

Abstract

lists the OSGi services provided by a bundle

Synopsis

osgi:bundle-services [-u] [-p] [-a] [--help] [--force] { *id* }

Arguments

Table 15.2, "**osgi:bundle-services** Arguments" describes the command's arguments.

Table 15.2. osgi:bundle-services Arguments

Argument	Interpretation
-u	Displays the services used by the bundle.
-p	Displays the properties for each service.
-a	Displays all of the services provided by the bundle including the Apache Karaf commands which are hidden by default.
--help	Displays the online help for this command.
--force	Forces the command to execute.
id	Specifies the id for the bundle.

15.3. OSGI:CLASSES, CLASSES

Abstract

lists all of the classes in the specified bundle or bundles

Synopsis

osgi:classes [--help] [--force] [[-a] | [--display-all-files]] { *ids* }

Arguments

Table 15.3, "**osgi:classes** Arguments" describes the command's arguments.

Table 15.3. osgi:classes Arguments

Argument	Interpretation
--help	Displays the online help for this command.
--force	Forces the command to execute.
-a, --display-all-files	Also lists the files contained in the bundles.
ids	Space-separated list of bundle IDs.

15.4. OSGI:FIND-CLASS, FIND-CLASS

Abstract

locates a specified class in any deployed bundle

Synopsis

osgi:find-class [--help] { *className* }

Arguments

Table 15.4, "**osgi:find-class** Arguments" describes the command's arguments.

Table 15.4. osgi:find-class Arguments

Argument	Interpretation
--help	Displays the online help for this command.
className	Class name or partial class name to find.

15.5. OSGI:HEADERS, HEADERS

Abstract

displays the headers of a specified OSGi bundle

Synopsis

osgi:headers [--help] { *id* ...}

Arguments

Table 15.5, "**osgi:headers** Arguments" describes the command's arguments.

Table 15.5. osgi:headers Arguments

Argument	Interpretation
--help	Displays the online help for this command
id	Specifies a space delimited list of bundle IDs.

15.6. OSGI:INFO

Abstract

displays detailed information about OSGi bundles

Synopsis

osgi:info [--help] { *id* ...}

Arguments

Table 15.6, "`osgi:info` Arguments" describes the command's arguments.

Table 15.6. `osgi:info` Arguments

Argument	Interpretation
`--help`	Displays the online help for this command
`id`	Specifies a space delimited list of bundle IDs.

15.7. OSGI:INSTALL, INSTALL

Abstract

installs one or more OSGi bundles

Synopsis

`osgi:install` [--help] [[-s] | [--start]] { *url* ...}

Arguments

Table 15.7, "`osgi:install` Arguments" describes the command's arguments.

Table 15.7. `osgi:install` Arguments

Argument	Interpretation
`--help`	Displays the online help for this command
`-s`, `--start`	Starts the bundles after installation
`url`	Specifies a space delimited list of bundle URLs.

15.8. OSGI:LIST, LIST

Abstract

lists the installed bundles whose start level equals or exceeds the specified threshold

Synopsis

`osgi:list` [--help] [-u] [-t *threshold*] [-i] [-s]

Arguments

Table 15.8, "`osgi:list` Arguments" describes the command's arguments.

Table 15.8. `osgi:list` Arguments

Argument	Interpretation
`--help`	Displays the online help for this command
`-u`	Shows the update locations
`-t`	Specifies the start level threshold. The default is the value of the `karaf.systemBundlesStartLevel` property whose default value is 50.
`-l`	Shows the locations of the bundles
`-s`	Shows the symbolic names of the bundles

15.9. OSGI:LS, LS

Abstract

lists OSGi services

Synopsis

`osgi:ls` [--help] [-a] [-u] [--force] [*id* ...]

Arguments

Table 15.9, "`osgi:ls` Arguments" describes the command's arguments.

Table 15.9. `osgi:ls` Arguments

Argument	Interpretation
`--help`	Displays the online help for this command
`-a`	Lists all services
`-u`	Lists the services in use
`--force`	Forces the command to execute
`id`	Specifies a space separated list of bundle IDs.

15.10. OSGI:REFRESH, REFRESH

Abstract

refreshes an OSGi bundle

Synopsis

osgi:refresh [--help] [--force] { *id* ...}

Arguments

Table 15.10, "**osgi:refresh** Arguments" describes the command's arguments.

Table 15.10. osgi:refresh Arguments

Argument	Interpretation
--help	Displays the online help for this command.
--force	Forces the command to execute.
id	Specifies a space delimited list of bundle IDs.

15.11. OSGI:RESOLVE, RESOLVE

Abstract

resolves an OSGi bundle's dependencies

Synopsis

osgi:resolve [--help] [--force] { *id* ...}

Arguments

Table 15.11, "**osgi:resolve** Arguments" describes the command's arguments.

Table 15.11. osgi:resolve Arguments

Argument	Interpretation
--help	Displays the online help for this command.
--force	Forces the command to execute.
id	Specifies a space delimited list of bundle IDs.

15.12. OSGI:RESTART, RESTART

Abstract

stops and restarts an OSGi bundle

Synopsis

osgi:restart [--help] [--force] { *id* ...}

Arguments

Table 15.12, "**osgi:restart** Arguments" describes the command's arguments.

Table 15.12. osgi:restart Arguments

Argument	Interpretation
--help	Displays the online help for this command.
--force	Forces the command to execute.
id	Specifies a space delimited list of bundle IDs.

15.13. OSGI:SHUTDOWN, SHUTDOWN

Abstract

stops the OSGi framework

Synopsis

osgi:shutdown [--help] [[-f] | [--force]] [[*hh*:*mm*] | [+*m*]]

Arguments

Table 15.13, "**osgi:shutdown** Arguments" describes the command's arguments.

Table 15.13. osgi:shutdown Arguments

Argument	Interpretation
--help	Displays the online help for this command.
-f, **--force**	Forces the command to execute.

Argument	Interpretation
hh:*mm*	Specifies the time to shut down the broker in hours and minutes. The time is specified in 24 hour time. For example, **13:30** specifies that the container will shutdown at 1:30pm.
+*m*	Specifies the time, in minutes, to pause before shutting down the OSGi framework. For example, **+30** specifies that the container will wait thirty minutes before shutting down the OSGi framework.

15.14. OSGI:START, START

Abstract

starts an OSGi bundle

Synopsis

osgi:start [--help] [--force] { *id* ...}

Arguments

Table 15.14, "**osgi:start** Arguments" describes the command's arguments.

Table 15.14. osgi:start Arguments

Argument	Interpretation
--help	Displays the online help for this command.
--force	Forces the command to execute.
id	Specifies a space delimited list of bundle IDs.

15.15. OSGI:START-LEVEL, START-LEVEL

Abstract

gets or sets the OSGi framework's active start level

Synopsis

osgi:start [--help] [*level*]

Arguments

Table 15.15, "`osgi:start-level` Arguments" describes the command's arguments.

Table 15.15. `osgi:start-level` Arguments

Argument	Interpretation
`--help`	Displays the online help for this command.
`level`	Specifies the new start level to set.

15.16. OSGI:STOP, STOP

Abstract

stops an OSGi bundle

Synopsis

`osgi:stop` [--help] [--force] { *id* ...}

Arguments

Table 15.16, "`osgi:stop` Arguments" describes the command's arguments.

Table 15.16. `osgi:stop` Arguments

Argument	Interpretation
`--help`	Displays the online help for this command.
`--force`	Forces the command to execute.
`id`	Specifies a space delimited list of bundle IDs.

15.17. OSGI:UNINSTALL, UNINSTALL

Abstract

uninstalls an OSGi bundle

Synopsis

`osgi:uninstall` [--help] [--force] { *id* ...}

Arguments

Table 15.17, "`osgi:uninstall` Arguments" describes the command's arguments.

Table 15.17. `osgi:uninstall` Arguments

Argument	Interpretation
`--help`	Displays the online help for this command.
`--force`	Forces the command to execute.
`id`	Specifies a space delimited list of bundle IDs.

15.18. OSGI:UPDATE, UPDATE

Abstract

updates an OSGi bundle

Synopsis

`osgi:update` [--help] [--force] { *id* } [*location*]

Arguments

Table 15.18, "`osgi:update` Arguments" describes the command's arguments.

Table 15.18. `osgi:update` Arguments

Argument	Interpretation
`--help`	Displays the online help for this command.
`--force`	Forces the command to execute.
`id`	Specifies ID of the bundle.
`location`	Specifies the location from which the update is loaded. If no location is specified the container will use either the bundle's Bundle-UpdateLocation property or the bundle's original location.

CHAPTER 16. PACKAGES CONSOLE COMMANDS

The **packages** commands are used for showing all packages imported and exported by the OSGi bundles currently installed.

Type **packages:** then press **Tab** at the prompt to view the available commands.

16.1. PACKAGES:EXPORTS, EXPORTS

Abstract

displays the packages exported OSGi bundles

Synopsis

packages:export [--help] [[-d] | [--details]] [-s] [[-i] | [--imports]] [*id* ...]

Arguments

Table 16.1, "**package:exports** Arguments" describes the commands arguments.

Table 16.1. package:exports Arguments

Argument	Interpretation
--help	Displays the online help for this command
-d, --details	Reformat the output in master/detail layout, which makes it easier to see how related details are grouped together.
-s	Show the **Symbolic name** column, which shows the symbolic name of the bundle to which the exported package belongs.
-i, --imports	Show the **Imported by** column, which lists all of the bundles that import the exported package.
id	Specifies a whitespace separated list of bundle IDs to check.

16.2. PACKAGES:IMPORTS, IMPORTS

Abstract

displays the packages imported by OSGi bundles

Synopsis

packages:imports [--help] [[-i] | [--show-importer]] [*id* ...]

Arguments

Table 16.2, "**package:imports** Arguments" describes the commands arguments.

Table 16.2. package:imports Arguments

Argument	Interpretation
--help	Displays the online help for this command
-i, --show-importer	Show the bundle(s) that import a package.
id	Specifies a whitespace separated list of bundle IDs to check.

CHAPTER 17. PATCH CONSOLE COMMANDS

The patch commands allow you to download, install, and manage patches.

Patches contain a discreet set of bundles intended to update a standalone container. Each patch includes the following metadata:

- the patch name
- a description of the patch
- the list of bundles included in the patch

The basic procedure applying a patch is:

1. You receive a notice from customer support that a patch is available.
2. Using the URL provided by customer support, you download the patch using the **`patch:add`** command.

   This command downloads an archive file, unzips the archive, and puts the relevant JAR files under the container's **`system/`** directory. The patch does *not* overwrite any of the existing JAR files and the patch is not actually installed until you run the **`patch:install`** command.
3. You install the patch using the **`patch:install`** command.
4. If you notice that the patch is causing issues, you can remove it using the **`patch:rollback`** command.

IMPORTANT

These commands are *not* suitable for use with containers that are part of a fabric. They are *only* for use in applying patches to standalone containers.

Type **`patch:`** then press **`Tab`** at the prompt to view the available commands.

17.1. PATCH:ADD, DOWNLOAD

Abstract

download a patch file from a remote location and places the relevant JAR files in the container's **`system`** directory

Synopsis

`patch:add` [--help] [--bundles] { *URL* }

Arguments

Table 17.1, "**`patch:add`** Arguments" describes the command's arguments.

Table 17.1. `patch:add` Arguments

Argument	Interpretation
--help	Displays the online help for this command.
--bundles	List the bundles included in the patch.
URL	Specifies the URL from which the patch is downloaded.

17.2. PATCH:FABRIC-INSTALL

Abstract

Installs a rollup patch in a Fabric system.

Synopsis

patch:fabric-install [--help] { *Patch* }

Arguments

Table 17.2, "**patch:fabric-install** Arguments" describes the command's arguments.

Table 17.2. patch:fabric-install Arguments

Argument	Interpretation
--help	Displays the online help for this command.
--upload	Upload the patch artifacts to a Maven repository.
-s, --simulation	Simulates patch installation.
--merge-prefer-new	When patching a resource file in a profile, if there is a merge conflict in a property setting, prefer the value from the patch (new).
--merge-keep-old	When patching a resource file in a profile, if there is a merge conflict in a property setting, prefer the existing value (old).
--merge-overwrite	When patching a resource file in a profile, replace the entire resource file with the patched one.
-u, --username	Remote user name.
-p, --password	Remote user password.

Argument	Interpretation
`--version`	Profile version to which the patch is applied.
Patch	The name of the patch to install.

17.3. PATCH:FABRIC-SYNCHRONIZE

Abstract

Synchronize patch information from the current container to the Fabric ensemble's git server, to make sure that patched profile data are made accessible to all containers in the fabric.

Synopsis

`patch:fabric-synchronize` [--help]

Arguments

Table 17.3, "**`patch:fabric-synchronize` Arguments**" describes the command's arguments.

Table 17.3. `patch:fabric-synchronize` Arguments

Argument	Interpretation
`--help`	Displays the online help for this command.

17.4. PATCH:INSTALL

Abstract

installs a patch that was previously downloaded

Synopsis

`patch:install` [--help] { *patch* }

Arguments

Table 17.4, "**`patch:install` Arguments**" describes the command's arguments.

Table 17.4. `patch:install` Arguments

Argument	Interpretation
`--help`	Displays the online help for this command.

Argument	Interpretation
patch	Specifies the name of the patch to install.

17.5. PATCH:LIST

Abstract

lists all known patches, showing the patch name and status (installed or not)

Synopsis

patch:list [--help] [--bundles]

Arguments

Table 17.5, "**patch:list** Arguments" describes the command's arguments.

Table 17.5. patch:list Arguments

Argument	Interpretation
--help	Displays the online help for this command.
--bundles	List the bundles for each patch.

17.6. PATCH:ROLLBACK

Abstract

reverses a patch installation

Synopsis

patch:rollback [--help] { *patch* }

Arguments

Table 17.6, "**patch:rollback** Arguments" describes the command's arguments.

Table 17.6. patch:rollback Arguments

Argument	Interpretation
--help	Displays the online help for this command.

Argument	Interpretation
patch	Specifies the name of the patch to roll back.

17.7. PATCH:SIMULATE, SIMULATE

Abstract

logs all of the actions that would be performed during a patch install, without actually performing the install

Synopsis

`patch:simulate` [--help] { *patch* }

Arguments

Table 17.7, "**`patch:simulate`** Arguments" describes the command's arguments.

Table 17.7. `patch:simulate` Arguments

Argument	Interpretation
`--help`	Displays the online help for this command.
patch	Specifies the name of the patch to simulate installing.

CHAPTER 18. SERVICE COMPONENT RUNTIME (SCR) CONSOLE COMMANDS

The **scr** commands are used for managing components declared using the OSGi Declarative Services specification.

18.1. SCR:ACTIVATE

Abstract

activate the specified SCR component

Synopsis

scr:activate [--help] { *ComponentName* }

Arguments

Table 18.1, "**scr:activate** Arguments" describes the command's arguments.

Table 18.1. scr:activate Arguments

Argument	Interpretation
--help	Displays the online help for this command.
ComponentName	The SCR component name (which can be found from the listing produced by the **scr:list** command).

18.2. SCR:DEACTIVATE

Abstract

deactivate the specified SCR component

Synopsis

scr:deactivate [--help] { *ComponentName* }

Arguments

Table 18.2, "**scr:deactivate** Arguments" describes the command's arguments.

Table 18.2. scr:deactivate Arguments

Argument	Interpretation
`--help`	Displays the online help for this command.
ComponentName	The SCR component name (which can be found from the listing produced by the `scr:list` command).

18.3. SCR:DETAILS

Abstract

show details for the specified SCR component

Synopsis

`scr:details` [--help] [[-s] | [--show-hidden]] { *ComponentName* }

Arguments

Table 18.3, "`scr:details` Arguments" describes the command's arguments.

Table 18.3. `scr:details` Arguments

Argument	Interpretation
`--help`	Displays the online help for this command.
`-s, --show-hidden`	Show all installed components, including the system components (which are hidden by default).
ComponentName	The SCR component name (which can be found from the listing produced by the `scr:list` command).

18.4. SCR:LIST

Abstract

list all of the components defined using the OSGi Declarative Services framework

Synopsis

`scr:list` [--help] [[-s] | [--show-hidden]]

Arguments

Table 18.4, "`scr:list` Arguments" describes the command's arguments.

Table 18.4. `scr:list` Arguments

Argument	Interpretation
`--help`	Displays the online help for this command.
`-s, --show-hidden`	Show all installed components, including the system components (which are hidden by default).

CHAPTER 19. SSH CONSOLE COMMANDS

The ssh commands allow you to connect to or create a secure shell (SSH) server.

Type **ssh:** then press **Tab** at the prompt to view the available commands.

19.1. SSH:SSH, SSH

Abstract

connects to a remote SSH server

Synopsis

ssh:ssh [--help] [[-l *username*] | [--username *username*]] [[-P *password*] | [--password *password*]] [[-p *port*] | [--port *port*]] { *hostname* } [*command*]

Arguments

Table 19.1, "**ssh:ssh** Arguments" describes the commands arguments.

Table 19.1. ssh:ssh Arguments

Argument	Interpretation
--help	Displays the online help for this command
-l, --username	The username for remote login
-P, --password	The password for remote login
-p, --port	The port to use for the SSH connection
hostname	The hostname to connect to via SSH
command	Specifies a command to execute upon connecting.

19.2. SSH:SSHD, SSHD

Abstract

creates an SSH server

Synopsis

ssh:sshd [--help] [[-b] | [--background]] [[-p *port*] | [--port *port*]]

Arguments

Table 19.2, "**ssh : sshd** Arguments" describes the commands arguments.

Table 19.2. ssh : sshd Arguments

Argument	Interpretation
--help	Displays the online help for this command
-b, --background	Specifies that the service will run in the background.
-p, --port	Specifies the port to setup for the SSH server. The default is **8101**.

CHAPTER 20. WEB CONSOLE COMMANDS

The web command group is used to get information about WARs deployed in the container.

Type **web:** then press **Tab** at the prompt to view the commands in this group.

20.1. WEB:LIST

Abstract

lists the WARs deployed in the container

Synopsis

web:list [--help]

Arguments

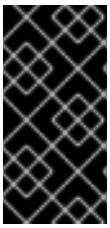
Table 20.1, "**web:list** Arguments" describes the command's arguments.

Table 20.1. web:list Arguments

Argument	Interpretation
--help	Displays the online help for this command

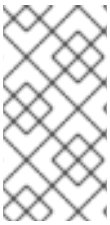
CHAPTER 21. THE WRAPPER:INSTALL COMMAND

The **wrapper:install** command installs Red Hat JBoss Fuse as a system service.



IMPORTANT

The **wrapper:install** Karaf console command—for installing JBoss Fuse as a service—is deprecated since JBoss Fuse 6.3 and will be removed in a future release of JBoss Fuse. To install the Apache Karaf container as a service, it is recommended that you use the new **karaf-service-*.sh** scripts from the **bin/contrib** directory instead.



NOTE

This feature is not installed by default. To install the **wrapper** shell, run the following command:

```
JBossFuse:karaf@root:> features:install wrapper
```

21.1. WRAPPER:INSTALL

Abstract

installs the container as a system service in the operating system

Synopsis

wrapper:install [--help] [[-s] | [--start-type] ***mode***] [[-n] | [--name] ***serviceName***] [[-d] | [--display] ***displayName***] [[-D] | [--description] ***description***]

Arguments

This command takes the following arguments.

Table 21.1. wrapper:install Arguments

Argument	Interpretation
--help	Displays the online help for this command
-s, --start-type	The mode in which the service is installed, either **AUTO_START** or **DEMAND_START**; the default is **AUTO_START**
-n, --name	The service name used when installing the service; the default is **karaf**
-d, --display	The display name of the service
-D, --description	The description of the service

CHAPTER 22. ZOOKEEPER CONSOLE COMMANDS

By default, the ZooKeeper commands are not installed in a Fabric Container. To make the ZooKeeper commands available, install the **`fabric-zookeeper-commands`** feature, as follows:

```
features:install fabric-zookeeper-commands
```

22.1. ZK:CREATE

Abstract

create a znode

Synopsis

`zk:create` [--help] [-r|--recursive] [-i|--import] [-e|--ephemeral] [-s|--sequential] [-a|--acl *ListOfACLs*] [-o|--overwrite] { *path* } { *data* }

Description

Using this command, you can create the following different types of znode:

Persistent

The new znode is permanently stored in the ZooKeeper registry. This is the default.

Persistent sequential

The new znode is permanently stored in the ZooKeeper registry and a 10-digit sequence number is appended to the specified znode name. Selected by the **`--sequential`** option.

Ephemeral

The new znode exists only for the duration of the current client session. When the session is over, the znode is removed. Selected by the **`--ephemeral`** option.

Ephemeral sequential

The new znode exists only for the duration of the current client session and a 10-digit sequence number is appended to the specified znode name. When the session is over, the znode is removed. Selected by combining the **`--ephemeral`** option with the **`--sequential`** option.

You can optionally specify a list of ACLs to apply to the newly created znode. The ACL is specified as a comma-separated list, where each entry in the list has the following format:

```
Scheme:ID:Permissions
```

ZooKeeper supports the following built-in schemes:

`world:anyone`

The permissions apply to all users.

auth:

The permissions apply to all authenticated users, irrespective of their identity (the ***ID*** field is left empty).

digest:*MD5Hash*

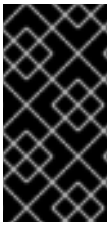
The permissions apply to the user whose username and password generate the specified MD5 hash value, ***MD5Hash***.

ip:*IPAddress*

The permissions apply to the ZooKeeper client with the specified IP address.

The ***Permissions*** string consists of one or more of the following characters: **r** (read), **w** (write), **c** (create), **d** (delete), and **a** (admin). For example, to create a new znode that explicitly grants all permissions to all users (which is, in fact, the default), you could use a command like the following:

```
karaf@root> zk:create --acl world:anyone:rwcda /path/to/the/new/znode
```



IMPORTANT

To avoid corruption of the fabric registry, you should *not* create any znodes under the **/fabric/** path using the **zk:create** command. These registry nodes should only be created through the **fabric** console commands—see Chapter 9, *Fabric Console Commands*.

NOTE

Fuse Fabric does *not* use the ACL security features of ZooKeeper. Currently, all znodes in the fabric registry are created without any ACL restrictions (equivalent to the **world:anyone:rwcda** ACL setting).

Arguments

Table 22.1, "**zk:create** Arguments" describes the commands arguments.

Table 22.1. zk:create Arguments

Argument	Interpretation
--help	Displays the online help for this command
-r,--recursive	Automatically create any missing parent nodes in the specified path.
-i,--import	Interpret the data argument as a URL that locates a resource containing the initial data for the new znode.

Argument	Interpretation
-e,--ephemeral	Make the new znode epehemeral, so that it is automatically deleted after the current ZooKeeper client session closes.
-s,--sequential	Make the new znode sequential, which implies that a unique 10-digit suffix is appended to the znode name.
-a,--acl	Specifies the znode's ACL as a comma-separated list, where each entry in the list has the format, ***Scheme*:*ID*:*Permissions***. The ***Permissions*** string consists of the following characters, concatenated in any order: **r** (read), **w** (write), **c** (create), **d** (delete), and **a** (admin).
-o,--overwrite	Overwrite the existing znode at this location, if there is one.
path	*(Required)* Path of the znode to create.
data	Initial data for the node or, if **--import** is specified, a URL pointing at a location that contains the initial data.

22.2. ZK:DELETE

Abstract

delete the specified znode

Synopsis

zk:delete [--help] [-v|--version *version*] [-r|--recursive] { *path* }

Arguments

Table 22.2, "**zk:delete** Arguments" describes the commands arguments.

Table 22.2. zk:delete Arguments

Argument	Interpretation
--help	Displays the online help for this command
-v,--version	The ZooKeeper znode version to delete. Defaults to -1 (all versions).

Argument	Interpretation
`-r,--recursive`	Recursively delete children. Defaults to`false`.
path	Path of the znode to delete.

22.3. ZK:GET

Abstract

get a znode's data

Synopsis

zk:get [--help] { *path* }

Arguments

Table 22.3, "**zk:get** Arguments" describes the commands arguments.

Table 22.3. zk:get Arguments

Argument	Interpretation
`--help`	Displays the online help for this command
path	*(Required)* Path of the znode to get.

22.4. ZK:LIST

Abstract

list a znode's children

Synopsis

zk:list [--help] [-r|--recursive] [-d|--display] { *path* }

Arguments

Table 22.4, "**zk:list** Arguments" describes the commands arguments.

Table 22.4. zk:list Arguments

Argument	Interpretation
`--help`	Displays the online help for this command
`-r, --recursive`	List children recursively.
`-d, --display`	Display a znode's value, if set.
path	Path of the znode to list. Defaults to `/`.

22.5. ZK:SET

Abstract

set a znode's data

Synopsis

`zk:set` [--help] [-i|--import] { *path* } { *data* }

Description

The data stored in a znode should not be too large. ZooKeeper imposes an absolute limit of 1 MB, but in practice a data item should normally be much smaller than that.

IMPORTANT

To avoid corruption of the Fabric Registry, you should *not* modify any znodes under the `/fabric/` path using the `zk:set` command. These registry values should only be changed through the `fabric` console commands—see Chapter 9, *Fabric Console Commands*.

Arguments

Table 22.5, "`zk:set` Arguments" describes the commands arguments.

Table 22.5. `zk:set` Arguments

Argument	Interpretation
`--help`	Displays the online help for this command
`-i,--import`	Import data from a URL.
path	*(Required)* Path of the znode to set.
data	*(Required)* The new data or URL to import.