

Fuse ESB Enterprise **Console Reference**

7.1
December 2012

Integration Everywhere

Console Reference

7.1

Updated: 08 Jan 2014

Copyright © 2012 Red Hat, Inc. and/or its affiliates.

Trademark Disclaimer

These materials and all Progress® software products are copyrighted and all rights are reserved by Progress Software Corporation. The information in these materials is subject to change without notice, and Progress Software Corporation assumes no responsibility for any errors that may appear therein. The references in these materials to specific platforms supported are subject to change.

Fuse, FuseSource, Fuse ESB, Fuse ESB Enterprise, Fuse MQ Enterprise, Fuse Mediation Router, Fuse Message Broker, Fuse Services Framework, Fuse IDE, Fuse HQ, Fuse Management Console, and Integration Everywhere are trademarks or registered trademarks of FuseSource Corp. or its parent corporation, Progress Software Corporation, or one of their subsidiaries or affiliates in the United States. Apache, ServiceMix, Camel, CXF, and ActiveMQ are trademarks of Apache Software Foundation. Any other names contained herein may be trademarks of their respective owners.

Third Party Acknowledgements

One or more products in the Fuse ESB Enterprise release includes third party components covered by licenses that require that the following documentation notices be provided:

- JLine (<http://jline.sourceforge.net>) jline:jline:jar:1.0

License: BSD (LICENSE.txt) - Copyright (c) 2002-2006, Marc Prud'hommeaux <mwpl@cornell.edu>

All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
- Neither the name of JLine nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

- Stax2 API (<http://woodstox.codehaus.org/StAX2>) org.codehaus.woodstox:stax2-api:jar:3.1.1

License: The BSD License (<http://www.opensource.org/licenses/bsd-license.php>)

Copyright (c) <YEAR>, <OWNER> All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

- jibx-run - JiBX runtime (<http://www.jibx.org/main-reactor/jibx-run>) org.jibx:jibx-run:bundle:1.2.3

License: BSD (<http://jibx.sourceforge.net/jibx-license.html>) Copyright (c) 2003-2010, Dennis M. Sosnoski.

All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
- Neither the name of JiBX nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

- JavaAssist (<http://www.jboss.org/javassist>) org.jboss.javassist:com.springsource.javassist:jar:3.9.0.GA:compile

License: MPL (<http://www.mozilla.org/MPL/MPL-1.1.html>)

- HAPI-OSGI-Base Module (<http://hl7api.sourceforge.net/hapi-osgi-base/>) ca.uhn.hapi:hapi-osgi-base:bundle:1.2
License: Mozilla Public License 1.1 (<http://www.mozilla.org/MPL/MPL-1.1.txt>)

Table of Contents

1. Using the Command Console	19
2. Shell Console Commands	25
shell:cat	26
shell:clear	27
shell:each	28
shell:echo	29
shell:exec	30
shell:grep	31
shell:head	33
shell:history	34
shell:if	35
shell:info	36
shell:java	37
shell:logout	38
shell:more	39
shell:new	40
shell:printf	41
shell:sleep	42
shell:sort	43
shell:source	44
shell:tac	45
shell:tail	46
shell:watch	47
3. ActiveMQ Console Commands	49
activemq:browse	50
activemq:bstat	53
activemq:list	54
activemq:purge	55
activemq:query	57
4. Admin Console Commands	59
admin:change-opts	60
admin:change-rmi-registry-port	61
admin:change-rmi-server-port	62
admin:change-ssh-port	63
admin:clone	64
admin:connect	65
admin:create	66
admin:destroy	67
admin:list	68
admin:rename	69
admin:start	70

admin:stop	71
5. Camel Console Commands	73
camel:context-info	74
camel:context-list	75
camel:context-start	76
camel:context-stop	77
camel:endpoint-list	78
camel:route-info	79
camel:route-list	80
camel:route-resume	81
camel:route-show	82
camel:route-start	83
camel:route-stop	84
camel:route-suspend	85
6. Config Console Commands	87
config:cancel	89
config:delete	90
config:edit	91
config:list	92
config:propappend	93
config:propdel	94
config:proplist	95
config:propset	96
config:update	97
7. CXF Console Commands	99
cxf:list-busses	100
cxf:list-endpoints	101
cxf:start-endpoint	102
cxf:stop-endpoint	103
8. Dev Console Commands	105
dev:classloaders	106
dev:create-dump	107
dev:dynamic-import	108
dev:framework	109
dev:print-stack-traces	110
dev:restart	111
dev:show-tree	112
dev:threads	113
dev:wait-for-service	114
dev:watch	116
9. Fuse Application Bundle(FAB) Console Commands	117
fab:headers	118
fab:info	120
fab:start	121

fab:stop	122
fab:tree	123
fab:uninstall	124
10. Fabric Console Commands	125
fabric:cluster-list	127
fabric:cloud-firewall-edit	128
fabric:cloud-service-add	129
fabric:cloud-service-list	133
fabric:cloud-service-remove	134
fabric:container-add-profile	135
fabric:container-connect	136
fabric:container-create	138
fabric:container-create-child	141
fabric:container-create-cloud	144
fabric:container-create-ssh	149
fabric:container-delete	152
fabric:container-domains	153
fabric:container-list	154
fabric:container-remove-profile	155
fabric:container-resolver-list	156
fabric:container-resolver-set	157
fabric:container-rollback	160
fabric:container-start	161
fabric:container-stop	162
fabric:container-upgrade	163
fabric:create	165
fabric:ensemble-add	169
fabric:ensemble-list	171
fabric:ensemble-password	172
fabric:ensemble-remove	173
fabric:export	174
fabric:import	176
fabric:join	178
fabric:mq-create	181
fabric:profile-change-parents	183
fabric:profile-create	184
fabric:profile-delete	186
fabric:profile-display	187
fabric:profile-edit	188
fabric:profile-list	192
fabric:require-profile-delete	193
fabric:require-profile-list	194
fabric:require-profile-set	195
fabric:status	196

fabric:version-create	198
fabric:version-delete	199
fabric:version-list	200
fabric:version-set-default	201
11. Features Console Commands	203
features:addurl	204
features:chooseurl	205
features:info	206
features:install	207
features:list	208
features:listurl	209
features:listVersions	210
features:refreshUrl	211
features:removeUrl	212
features:removeRepository	213
features:uninstall	214
12. JAAS Console Commands	215
jass:cancel	217
jaas:manage	218
jaas:pending	220
jaas:realms	221
jaas:roleadd	222
jaas:roledel	223
jaas:update	224
jaas:useradd	225
jaas:userdel	226
jaas:users	227
13. JBI Console Commands	229
jbi:list	230
jbi:shutdown	231
jbi:start	232
jbi:stop	233
14. Log Console Commands	235
log:clear	236
log:display	237
log:display-exception	238
log:get	239
log:set	240
log:tail	241
15. The nmr:list Command	243
nmr:list	244
16. OBR Console Commands	245
obr:addUrl	246
obr:deploy	247

obr:info	248
obr:list	249
obr:listUrl	250
obr.refreshUrl	251
obr.removeUrl	252
obr.source	253
obr:start	254
17. OSGi Console Commands	255
osgi:bundle-level	256
osgi:bundle-services	257
osgi:classes	258
osgi.find-class	259
osgi:headers	260
osgi:info	261
osgi:install	262
osgi:list	263
osgi:ls	264
osgi:refresh	265
osgi:resolve	266
osgi:restart	267
osgi.shutdown	268
osgi:start	269
osgi:start-level	270
osgi:stop	271
osgi:uninstall	272
osgi:update	273
18. Packages Console Commands	275
packages:exports	276
packages:imports	277
19. Patch Console Commands	279
patch:add	280
patch:install	281
patch:list	282
patch.rollback	283
patch:simulate	284
20. SSH Console Commands	285
ssh:ssh	286
ssh:sshd	287
21. Web Console Commands	289
web:list	290
22. The wrapper:install Command	291
wrapper:install	292
23. ZooKeeper Console Commands	293
zk:create	294

zk:delete	297
zk:get	298
zk:list	299
zk:set	300
A. Command Aliases	301

List of Tables

1.1. Apache ActiveMQ Command Groups	22
2.1. shell:cat Arguments	26
2.2. shell:clear Arguments	27
2.3. shell:each Arguments	28
2.4. shell:echo Arguments	29
2.5. shell:exec Arguments	30
2.6. shell:grep Arguments	31
2.7. shell:head Arguments	33
2.8. shell:history Arguments	34
2.9. shell:if Arguments	35
2.10. shell:info Arguments	36
2.11. shell:java Arguments	37
2.12. shell:logout Arguments	38
2.13. shell:more Arguments	39
2.14. shell:new Arguments	40
2.15. shell:printf Arguments	41
2.16. shell:sleep Arguments	42
2.17. shell:sort Arguments	43
2.18. shell:source Arguments	44
2.19. shell:tac Arguments	45
2.20. shell:tail Arguments	46
2.21. shell:watch Arguments	47
3.1. activemq:browse Arguments	50
3.2. Message Headers for Filtering	51
3.3. activemq:bstat Arguments	53
3.4. activemq:list Arguments	54
3.5. activemq:purge Arguments	55
3.6. activemq:query Arguments	57
4.1. admin:change-opts Arguments	60
4.2. admin:change-rmi-registry-port Arguments	61
4.3. admin:change-rmi-server-port Arguments	62
4.4. admin:change-ssh-port Arguments	63
4.5. admin:clone Arguments	64
4.6. admin:connect Arguments	65
4.7. admin:create Arguments	66
4.8. admin:destroy Arguments	67
4.9. admin:list Arguments	68
4.10. admin:rename Arguments	69
4.11. admin:start Arguments	70
4.12. admin:stop Arguments	71
5.1. camel:context-info Arguments	74

5.2. camel:context-list Arguments	75
5.3. camel:context-start Arguments	76
5.4. camel:context-stop Arguments	77
5.5. camel:endpoint-list Arguments	78
5.6. camel:route-info Arguments	79
5.7. camel:route-list Arguments	80
5.8. camel:route-resume Arguments	81
5.9. camel:route-show Arguments	82
5.10. camel:route-start Arguments	83
5.11. camel:route-stop Arguments	84
5.12. camel:route-suspend Arguments	85
6.1. config:cancel Arguments	89
6.2. config:delete Arguments	90
6.3. config:edit Arguments	91
6.4. config:list Arguments	92
6.5. config:propappend Arguments	93
6.6. config:propdel Arguments	94
6.7. config:proplist Arguments	95
6.8. config:propset Arguments	96
6.9. config:update Arguments	97
7.1. cxflist-busses Arguments	100
7.2. cxflist-endpoints Arguments	101
7.3. cxflist-start-endpoint Arguments	102
7.4. cxflist-stop-endpoint Arguments	103
8.1. dev:classloader Arguments	106
8.2. dev:create-dump Arguments	107
8.3. dev:dynamic-import Arguments	108
8.4. dev:framework Arguments	109
8.5. dev:print-stack-traces Arguments	110
8.6. dev:restart Arguments	111
8.7. dev:show-tree Arguments	112
8.8. dev:threads Arguments	113
8.9. dev:wait-for-service Arguments	114
8.10. dev:watch Arguments	116
9.1. fab:headers Arguments	119
9.2. fab:info Arguments	120
9.3. fab:start Arguments	121
9.4. fab:stop Arguments	122
9.5. fab:tree Arguments	123
9.6. fab:uninstall Arguments	124
10.1. fabric:cluster-list Arguments	127
10.2. fabric:cloud-firewall-edit Arguments	128
10.3. fabric:cloud-service-add Arguments	131
10.4. fabric:cloud-service-list Arguments	133

10.5. fabric:cloud-service-remove Arguments	134
10.6. fabric:container-add-profile Arguments	135
10.7. fabric:container-connect Arguments	136
10.8. fabric:container-create Arguments	139
10.9. fabric:container-create-child Arguments	143
10.10. fabric:container-create-cloud Arguments	145
10.11. fabric:container-create-ssh Arguments	150
10.12. fabric:container-delete Arguments	152
10.13. fabric:container-domains Arguments	153
10.14. fabric:container-list Arguments	154
10.15. fabric:container-remove-profile Arguments	155
10.16. fabric:container-resolver-list Arguments	156
10.17. fabric:container-resolver-set Arguments	159
10.18. fabric:container-rollback Arguments	160
10.19. fabric:container-start Arguments	161
10.20. fabric:container-stop Arguments	162
10.21. fabric:container-upgrade Arguments	164
10.22. fabric:create Arguments	166
10.23. fabric:ensemble-add Arguments	169
10.24. fabric:ensemble-list Arguments	171
10.25. fabric:ensemble-password Arguments	172
10.26. fabric:ensemble-remove Arguments	173
10.27. fabric:export Arguments	174
10.28. fab:start Arguments	176
10.29. fabric:join Arguments	178
10.30. fabric:mq-create Arguments	181
10.31. fabric:profile-change-parents Arguments	183
10.32. fabric:profile-create Arguments	184
10.33. fabric:profile-delete Arguments	186
10.34. fabric:profile-display Arguments	187
10.35. fabric:profile-edit Arguments	191
10.36. fabric:profile-list Arguments	192
10.37. fabric:require-profile-delete Arguments	193
10.38. fabric:require-profile-list Arguments	194
10.39. fabric:require-profile-set Arguments	195
10.40. fabric:status Arguments	196
10.41. fabric:version-create Arguments	198
10.42. fabric:version-delete Arguments	199
10.43. fabric:version-list Arguments	200
10.44. fabric:version-set-default Arguments	201
11.1. features:addurl Arguments	204
11.2. features:chooseurl Arguments	205
11.3. features:info Arguments	206
11.4. features:install Arguments	207

11.5. features:list Arguments	208
11.6. features:listurl Arguments	209
11.7. features:listVersions Arguments	210
11.8. features:refreshUrl Arguments	211
11.9. features:removeUrl Arguments	212
11.10. features:removeRepository Arguments	213
11.11. features:uninstall Arguments	214
12.1. jaas:cancel Arguments	217
12.2. jaas:manage Arguments	218
12.3. jaas:pending Arguments	220
12.4. jaas:realms Arguments	221
12.5. jaas:roleadd Arguments	222
12.6. jaas:roledel Arguments	223
12.7. jaas:update Arguments	224
12.8. jaas:useradd Arguments	225
12.9. jaas:userdel Arguments	226
12.10. jaas:users Arguments	227
13.1. jbi:list Arguments	230
13.2. jbi:shutdown Arguments	231
13.3. jbi:start Arguments	232
13.4. jbi:stop Arguments	233
14.1. log:clear Arguments	236
14.2. log:display Arguments	237
14.3. log:display-exception Arguments	238
14.4. log:get Arguments	239
14.5. log:set Arguments	240
14.6. log:tail Arguments	241
15.1. nmr:list Arguments	244
16.1. obr:addUrl Arguments	246
16.2. obr:deploy Arguments	247
16.3. obr:info Arguments	248
16.4. obr:list Arguments	249
16.5. obr:listUrl Arguments	250
16.6. obr:refreshUrl Arguments	251
16.7. obr:removeUrl Arguments	252
16.8. obr:source Arguments	253
16.9. obr:start Arguments	254
17.1. osgi:bundle-level Arguments	256
17.2. osgi:bundle-services Arguments	257
17.3. osgi:classes Arguments	258
17.4. osgi:find-class Arguments	259
17.5. osgi:headers Arguments	260
17.6. osgi:info Arguments	261
17.7. osgi:install Arguments	262

17.8. osgi:list Arguments	263
17.9. osgi:ls Arguments	264
17.10. osgi:refresh Arguments	265
17.11. osgi:resolve Arguments	266
17.12. osgi:restart Arguments	267
17.13. osgi:shutdown Arguments	268
17.14. osgi:start Arguments	269
17.15. osgi:start-level Arguments	270
17.16. osgi:stop Arguments	271
17.17. osgi:uninstall Arguments	272
17.18. osgi:update Arguments	273
18.1. package:exports Arguments	276
18.2. package:imports Arguments	277
19.1. patch:add Arguments	280
19.2. patch:install Arguments	281
19.3. patch:list Arguments	282
19.4. patch:rollback Arguments	283
19.5. patch:simulate Arguments	284
20.1. ssh:ssh Arguments	286
20.2. ssh:sshd Arguments	287
21.1. web:list Arguments	290
22.1. wrapper:install Arguments	292
23.1. zk:create Arguments	295
23.2. zk:delete Arguments	297
23.3. zk:get Arguments	298
23.4. zk:list Arguments	299
23.5. zk:set Arguments	300
A.1. Console Command Aliases	301

List of Examples

1.1. The Fuse ESB Enterprise Console	19
1.2. Console Help	20
1.3. Help for a Command	21
1.4. Console Commands	21
10.1. Executing a Command in a Remote Container	136

Chapter 1. Using the Command Console

Overview

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

The console provides commands that you can use to perform basic management of your Fuse ESB Enterprise 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 Fuse ESB Enterprise, open a command prompt in the installation directory and enter:

Windows	bin\fuseesb.bat
*NIX	bin/fuseesb.sh

Fuse ESB Enterprise starts and the console is ready. You should see the prompt shown in [Example 1.1 on page 19](#).

Example 1.1. The Fuse ESB Enterprise Console

```
Fuse ESB (7.0.0.fuse-beta-042)
http://fusesource.com/products/fuse-esb-enterprise/

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 Fuse ESB.
```

```
FuseESB:karaf@root>
```

Getting help

The console provides two levels of help:

- console help—list all of the commands along with a brief summary of the commands function
- command help—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 1.2 on page 20](#).

Example 1.2. Console Help

```
FuseESB: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.
    ...
FuseESB: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 1.3 on page 21](#), entering **admin:start --help** displays the help for that command.

Example 1.3. Help for a Command

```
FuseESB: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

FuseESB: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 1.4 on page 21](#) is shown.

Example 1.4. Console Commands

```
activemq:browse
activemq:bstat          activemq:create-broker
activemq:destroy-broker  activemq:list
activemq:purge           activemq:query
FuseMQ: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 on page 22](#) 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.
camel	Manages Apache Camel contexts and routes
config	Manages configuration.
cdf	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.

Command Group	Description
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.

Chapter 2. Shell Console Commands

shell:cat	26
shell:clear	27
shell:each	28
shell:echo	29
shell:exec	30
shell:grep	31
shell:head	33
shell:history	34
shell:if	35
shell:info	36
shell:java	37
shell:logout	38
shell:more	39
shell:new	40
shell:printf	41
shell:sleep	42
shell:sort	43
shell:source	44
shell:tac	45
shell:tail	46
shell:watch	47

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.

Name

shell:cat, cat — displays the contents of a file or URL

Synopsis

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

Arguments

[Table 2.1 on page 26](#) 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
<i>path</i>	The path(s) of the file to display, separated by whitespace (separated by - for STDIN)
<i>URL</i>	The URL(s) to display, separated by whitespace (separated by - for STDIN)

Name

shell:clear, clear — clears the console buffer

Synopsis

```
shell:clear [--help]
```

Arguments

[Table 2.2 on page 27](#) describes the command's arguments.

Table 2.2. shell:clear Arguments

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

Name

shell:each, each — execute a closure on a list of arguments

Synopsis

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

Arguments

[Table 2.3 on page 28](#) 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.

Name

shell:echo, echo — prints arguments to the standard output

Synopsis

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

Arguments

[Table 2.4 on page 29](#) 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.
<i>argument</i>	Specifies a space delimited list of arguments to print.

Name

shell:exec, exec — executes system processes

Synopsis

shell:exec [--help] {command}

Arguments

[Table 2.5 on page 30](#) 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.

Name

shell:grep, grep — 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 on page 31](#) 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 <i>regex</i> 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: <ul style="list-style-type: none">• 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.

Argument	Interpretation
<i>regex</i>	Specifies the regular expression to match.

Name

shell:head, head — displays the first lines of a file

Synopsis

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

Arguments

[Table 2.7 on page 33](#) 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.
<i>path</i>	The path(s) of the file to display, separated by whitespace (separated by - for STDIN)
<i>URL</i>	Specifies the URL(s) to display, separated by whitespace (separated by - for STDIN)

Name

shell:history, history — prints the command history

Synopsis

```
shell:history [--help]
```

Arguments

[Table 2.8 on page 34](#) describes the arguments for this command.

Table 2.8. shell:history Arguments

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

Name

shell:if, if — executes an if/then/else block

Synopsis

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

Arguments

[Table 2.9 on page 35](#) describes the command's arguments.

Table 2.9. *shell:if Arguments*

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

Name

shell:info, info — displays system information and statistics about the container

Synopsis

```
shell:info [--help]
```

Arguments

[Table 2.10 on page 36](#) describes the command's arguments.

Table 2.10. shell:info Arguments

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

Name

shell:java, java — execute a Java application

Synopsis

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

Arguments

[Table 2.11 on page 37](#) 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 <code>main()</code> .
<i>className</i>	Specifies the name of the class to invoke.
<i>arguments</i>	Specifies the arguments to pass to the method of the given <i>className</i> .

Name

shell:logout, logout — disconnects the shell from the current session

Synopsis

```
shell:logout [--help]
```

Arguments

[Table 2.12 on page 38](#) describes the command's arguments.

Table 2.12. shell:logout Arguments

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

Name

shell:more, more — displays output as pages of a specified length

Synopsis

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

Arguments

[Table 2.13 on page 39](#) 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.

Name

shell:new, new — creates a new Java object of the specified class

Synopsis

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

Arguments

[Table 2.14 on page 40](#) describes the command's arguments.

Table 2.14. shell:new Arguments

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

Name

shell:printf, printf — formats and prints the specified output

Synopsis

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

Arguments

[Table 2.15 on page 41](#) describes the command's arguments.

Table 2.15. shell:printf Arguments

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

Name

shell:sleep, sleep — sleeps for a specified time, then wakes up

Synopsis

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

Arguments

[Table 2.16 on page 42](#) 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).
<i>duration</i>	The time to sleep in milliseconds (default) or in seconds (with the <code>-s</code> option).

Name

shell:sort, sort — 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 on page 43](#) 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, --ignore-leading-blanks	Ignores 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.
<i>files</i>	Specifies a space delimited list of files to sort.

Name

shell:source, source — run a shell script

Synopsis

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

Arguments

[Table 2.18 on page 44](#) describes the command's arguments.

Table 2.18. *shell:source Arguments*

Argument	Interpretation
--help	Displays the online help for this command
<i>script</i>	A URI pointing to the script
<i>arguments</i>	Arguments to pass to the script

Name

shell:tac, tac — 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 on page 45](#) 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.

Name

shell:tail, tail — displays the last lines of a file

Synopsis

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

Arguments

[Table 2.20 on page 46](#) 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.
<i>path</i>	A space delimited list of file paths to display.
<i>URL</i>	A space delimited list of file URLs to display.

Name

shell:watch, watch — watches and refreshes the output of a command

Synopsis

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

Arguments

[Table 2.21 on page 47](#) describes the command's arguments.

Table 2.21. shell:watch Arguments

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

Chapter 3. ActiveMQ Console Commands

activemq:browse	50
activemq:bstat	53
activemq:list	54
activemq:purge	55
activemq:query	57

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.

Name

activemq:browse, browse — 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 on page 50](#) describes the command's arguments.

Table 3.1. activemq:browse Arguments

Argument	Interpretation
--amqurl <i>brokerURL</i>	Specifies the URL of the broker to which you are connecting.
--msgsel <i>msgsel1,msgsel2,...</i>	Displays messages matched by the message selector.
--factory <i>className</i>	Load <i>className</i> as the javax.jms.ConnectionFactory to use for creating connections.
--passwordFactory <i>className</i>	Load <i>className</i> as the org.apache.activemq.console.command.PasswordFactory for retrieving the password from a keystore.
--user <i>username</i>	Username to use for JMS connections.
--password <i>password</i>	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.
-Vbody	Shows the body of the message.
--view <i>attr1,attr1,...</i>	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 on page 51](#) 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	<code>javax.jms.Destination</code>
JMSExpiration	long
JMSMessageID	String
JMSPriority	int
JMSRedelivered	boolean
JMSReplyTo	<code>javax.jms.Destination</code>
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:

```
FuseMQ: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:

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

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

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

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

```
FuseMQ: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:

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

Name

activemq:bstat, bstat — 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 on page 53](#) describes the command's arguments.

Table 3.3. *activemq:bstat* Arguments

Argument	Description
--jmxurl <i>URL</i>	Sets the JMX URL used to locate brokers.
--pid <i>PID</i>	Set the pid to connect to (only on Sun JVM).
--jmxuser <i>user</i>	Sets the JMX user, used for authentication.
--jmxpassword <i>password</i>	Sets the JMX password, used for authentication.
--jmxlocal	Use the local JMX server instead of a remote server.
--version	Displays the version information.
-h, -?, --help	Displays the online help for this command.
<i>brokerName</i>	The name of the broker

Name

activemq:list — 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 on page 54](#) describes the command's arguments.

Table 3.4. *activemq:list* Arguments

Argument	Interpretation
--jmxurl <i>URL</i>	Sets the JMX URL to connect to
--pid <i>PID</i>	Set the pid to connect to (only on Sun JVM).
--jmxuser <i>user</i>	Sets the JMX user, used for authentication
--jmxpassword <i>password</i>	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

Name

activemq:purge, purge — 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 on page 55](#) describes the command's arguments.

Table 3.5. activemq:purge Arguments

Option	Interpretation
--msgsel <i>msgsel1,msgsel2,...</i>	Purges messages matched by the message selector. See Message filters on page 51 .
--jmxurl <i>URL</i>	Sets the JMX URL used to locate the broker.
--pid <i>PID</i>	Set the pid to connect to (only on Sun JVM).
--jmxuser <i>user</i>	Sets the JMX user, used for authentication.
--jmxpassword <i>password</i>	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
<i>destName</i>	The specified message destination(s)

Examples

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

```
FuseMQ:karaf@root>activemq:purge TEST.FOO
```

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

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

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

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

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

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


Name

activemq:query, query — queries the for broker information on specific objects

Synopsis

```
activemq:query [-QMQBeanType=name] [-xQMBeanType=name] [--objname
query] [--xobjname query] [--view {attr...}] [--jmxurl JMXURL] [--pid PID]
[-jmxuser userName] [-jmxpassword password] [-jmxlocal] [--version] [--help]
| [-h] | [-?]]
```

Arguments

[Table 3.6 on page 57](#) 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
--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:

```
FuseMQ:karaf@root>activemq:query
```

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

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

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

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

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

```
FuseMQ: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.:`

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

Chapter 4. Admin Console Commands

admin:change-opts	60
admin:change-rmi-registry-port	61
admin:change-rmi-server-port	62
admin:change-ssh-port	63
admin:clone	64
admin:connect	65
admin:create	66
admin:destroy	67
admin:list	68
admin:rename	69
admin:start	70
admin:stop	71

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.

Name

admin:change-opts, change-opts — changes the Java options of an existing container

Synopsis

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

Arguments

[Table 4.1 on page 60](#) describes the command's arguments.

Table 4.1. admin:change-opts Arguments

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

Name

admin:change-rmi-registry-port, change-rmi-registry-port — 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 on page 61](#) describes the command's arguments.

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

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

Name

admin:change-rmi-server-port, change-rmi-server-port — 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 on page 62](#) describes the command's arguments.

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

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

Name

admin:change-ssh-port, changessh-port — changes the secure shell port of a container

Synopsis

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

Arguments

[Table 4.4 on page 63](#) 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

Name

admin:clone, clone — 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 on page 64](#) 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.
-v, --verbose	Display actions performed by the command (disabled by default).
<i>name</i>	Name of the original container instance.
<i>cloneName</i>	Name of the cloned container instance.

Name

admin:connect, connect — connects to an existing container

Synopsis

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

Arguments

[Table 4.6 on page 65](#) 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 <code>karaf</code>
-p, --password	The remote user password; the default is <code>karaf</code>
<i>container</i>	The container to connect to
<i>command</i>	Command to execute on connecting

Name

`admin:create`, `create` — 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]
RMIServPort] [--o] | [--java-opts] javaOpts] {name}
```

Arguments

[Table 4.7 on page 66](#) describes the command's arguments.

Table 4.7. `admin:create` Arguments

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

Name

admin:destroy, destroy — destroys a child container

Synopsis

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

Arguments

[Table 4.8 on page 67](#) describes the command's arguments.

Table 4.8. *admin:destroy Arguments*

Argument	Interpretation
<code>--help</code>	Displays the online help for this command
<i>name</i>	The name of the container to destroy

Name

admin:list — 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 on page 68](#) 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

Name

admin:rename, rename — renames a child container

Synopsis

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

Arguments

[Table 4.10 on page 69](#) describes the command's arguments.

Table 4.10. admin:rename Arguments

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

Name

`admin:start` — starts a child container

Synopsis

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

Arguments

[Table 4.11 on page 70](#) describes the command's arguments.

Table 4.11. *admin:start* Arguments

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

Name

admin:stop — stops a child container

Synopsis

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

Arguments

[Table 4.12 on page 71](#) describes the command's arguments.

Table 4.12. admin:stop Arguments

Argument	Interpretation
--help	Displays the online help for this command
<i>name</i>	The name of the container to start

Chapter 5. Camel Console Commands

camel:context-info	74
camel:context-list	75
camel:context-start	76
camel:context-stop	77
camel:endpoint-list	78
camel:route-info	79
camel:route-list	80
camel:route-resume	81
camel:route-show	82
camel:route-start	83
camel:route-stop	84
camel:route-suspend	85

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

Name

camel:context-info — 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
<i>contextName</i>	The name of the Camel context.

Name

camel:context-list — 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

Name

camel:context-start — 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.

Name

camel:context-stop — 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
<i>contextName</i>	The name of the Camel context.

Name

camel:endpoint-list — 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

Name

camel:route-info — 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.
contextName	(Optional) The Camel context name.

Name

camel:route-list — 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.

Name

camel:route-resume — 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.

Name

camel:route-show — 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.

Name

camel:route-start — 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.

Name

camel:route-stop — 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.

Name

camel:route-suspend — 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.

Chapter 6. Config Console Commands

config:cancel	89
config:delete	90
config:edit	91
config:list	92
config:propappend	93
config:propdel	94
config:proplist	95
config:propset	96
config:update	97

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.

Name

`config:cancel` — 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 on page 89](#) describes the command's arguments.

Table 6.1. *config:cancel Arguments*

Option	Interpretation
<code>--help</code>	Displays the online help for this command

Name

config:delete, delete — 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 on page 90](#) describes the command's arguments.

Table 6.2. config:delete Arguments

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

Name

config:edit, edit — 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 on page 91](#) 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

Name

`config:list` — lists the existing configurations for the container

Synopsis

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

Arguments

[Table 6.4 on page 92](#) describes the command's arguments.

Table 6.4. *config:list* Arguments

Argument	Interpretation
<code>--help</code>	Displays the online help for this command
<code>query</code>	An LDAP query

Name

`config:propappend`, `propappend` — 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 on page 93](#) describes the command's arguments.

Table 6.5. `config:propappend` Arguments

Argument	Interpretation
<code>--help</code>	Displays the online help for this command.
<code>-b</code> , <code>--bypass-storage</code>	Do not write the change to the local file.
<code>-p</code> , <code>--pid</code>	Specifies the PID of the configuration in which to make the change. The default is to change the configuration currently open for editing.
<i>name</i>	Specifies the name of the property to change.
<i>value</i>	Specifies the value to append to the property.

Name

config:propdel, propdel — 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 on page 94](#) 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.
<i>name</i>	Specifies the name of the property to delete.

Name

config:proplist, proplist — lists the properties in the configuration being edited

Synopsis

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

Arguments

[Table 6.7 on page 95](#) 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

Name

config:propset, propset — 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 on page 96](#) describes the command's arguments.

Table 6.8. *config:propset Arguments*

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

Name

`config:update` — saves the changes made to the configuration being edited and propagates them to the container

Synopsis

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

Arguments

[Table 6.9 on page 97](#) describes the command's arguments.

Table 6.9. `config:update` Arguments

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

Chapter 7. CXF Console Commands

cxflist-busses	100
cxflist-endpoints	101
cxfstart-endpoint	102
cxfstop-endpoint	103

The **cxfl** 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:

```
karaf@root> features:install cxf-osgi
```

Name

`cxflist-busses` — lists all Apache CXF buses

Synopsis

`cxflist-busses [--help]`

Arguments

This command takes the following arguments.

Table 7.1. *cxflist-busses* Arguments

Argument	Interpretation
<code>--help</code>	Displays the online help for this command

Name

`cxflistendpoints` — list all active endpoints belonging to the specified Apache CXF bus

Synopsis

```
cxflistendpoints [--help] {busID}
```

Arguments

This command takes the following arguments.

Table 7.2. *cxflistendpoints* Arguments

Argument	Interpretation
<code>--help</code>	Displays the online help for this command
<code>busID</code>	The Apache CXF bus ID.

Name

`cxf:start-endpoint` — 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
<code>--help</code>	Displays the online help for this command
<code>busID</code>	The Apache CXF bus ID.
<code>endpointName</code>	The name of the Apache CXF endpoint.

Name

`cxf:stop-endpoint` — 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
<code>--help</code>	Displays the online help for this command
<code>busID</code>	The Apache CXF bus ID.
<code>endpointName</code>	The name of the Apache CXF endpoint.

Chapter 8. Dev Console Commands

dev:classloaders	106
dev:create-dump	107
dev:dynamic-import	108
dev:framework	109
dev:print-stack-traces	110
dev:restart	111
dev:show-tree	112
dev:threads	113
dev:wait-for-service	114
dev:watch	116

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.

Name

dev:classloaders, classloaders — displays a list of leaking bundle classloaders

Synopsis

dev:classloaders [--help]

Arguments

[Table 8.1 on page 106](#) describes the commands arguments.

Table 8.1. dev:classloader Arguments

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

Name

dev:create-dump, create-dump — creates a ZIP file containing diagnostic information

Synopsis

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

Arguments

[Table 8.2 on page 107](#) 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
<i>dumpName</i>	Specifies the name for the dump file

Name

dev:dynamic-import, dynamic-import — enables/disables dynamic imports for a bundle

Synopsis

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

Arguments

[Table 8.3 on page 108](#) describes the commands arguments.

Table 8.3. dev:dynamic-import Arguments

Argument	Interpretation
--help	Displays the online help for this command
<i>bundleID</i>	A bundle ID.

Name

dev:framework, framework — enables/disables debugging for an OSGi framework

Synopsis

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

Arguments

[Table 8.4 on page 109](#) describes the commands arguments.

Table 8.4. dev:framework Arguments

Argument	Interpretation
--help	Displays the online help for this command
-nodebug, --disable-debug	Disable debugging for the OSGi framework.
-debug, --enable-debug	Enable debugging for the OSGi framework.
<i>framework</i>	Name of the OSGi framework

Name

dev:print-stack-traces, print-stack-traces — 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 on page 110](#) 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

Name

dev:restart — restart the container

Synopsis

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

Arguments

[Table 8.6 on page 111](#) 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.

Name

`dev:show-tree`, `show-tree` — shows the tree of bundles based on the wiring information

Synopsis

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

Arguments

[Table 8.7 on page 112](#) describes the commands arguments.

Table 8.7. `dev:show-tree` Arguments

Argument	Interpretation
<code>--help</code>	Displays the online help for this command
<i>bundleID</i>	A bundle ID.

Name

dev:threads, threads — shows the threads in the JVM

Synopsis

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

Arguments

[Table 8.8 on page 113](#) 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

Name

dev:wait-for-service, wait-for-service — 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 &(object
Class=org.apache.karaf.shell.console.SubShell) (name=osgi)
true
```

Arguments

[Table 8.9 on page 114](#) describes the commands arguments.

Table 8.9. dev:wait-for-service Arguments

Argument	Interpretation
<code>--help</code>	Displays the online help for this command

Argument	Interpretation
<code>-t, --timeout</code>	Timeout (specified in milliseconds: negative to not wait at all, zero to wait forever). Default is forever.
<code>-e, --exception</code>	Throw an exception if the wait command times out (the service is not found). Default is false.
<code>serviceClassOrFilter</code>	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).

Name

`dev:watch`, `watch` — watches and automatically updates bundles

Synopsis

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

Arguments

[Table 8.10 on page 116](#) describes the commands arguments.

Table 8.10. *dev:watch Arguments*

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



Caution

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

```
FuseMQ:karaf@root> dev:watch *
```

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

Chapter 9. Fuse Application Bundle(FAB) Console Commands

fab:headers	118
fab:info	120
fab:start	121
fab:stop	122
fab:tree	123
fab:uninstall	124

The **fab** commands are used for managing Fuse Application Bundle(FAB)s.

There is no dedicated install command for FABs. To install a FAB, use the `osgi:install` command combined with the `fab:` URL prefix. For example, to install the FAB `mvn:org.fusesource.example/camel-example/1.0` use the following console command:

```
FuseMQ:karaf@root> osgi:install fab:mvn:org.fusesource.example/camel-example/1.0
```

Name

fab:headers — displays the headers of a FAB

Synopsis

```
fab:headers [--help] [--indent style] {URL}
```

Description

Displays the header entries from the `META-INF/MANIFEST.MF` file embedded in the FAB JAR file. This is *not* the same thing as the bundle headers returned by the `osgi:headers` command, because the `osgi:headers` command shows the effective headers *after* the FAB is converted into an OSGi bundle.

For example, a typical FAB might have headers like the following:

```
FuseESB:karaf@root> fab:headers mvn:org.fusesource.ex
amples/cbr/7.0.0.fuse-beta-042

Manifest-Version = 1.0
Archiver-Version = Plexus Archiver
Built-By = username
Build-Jdk = 1.6.0_29
Created-By = Apache Maven
```

After the FAB is deployed, the corresponding OSGi bundle could have headers like the following (given that the bundle ID of the deployed FAB is 228):

```
FuseESB:karaf@root> osgi:headers 228

org.fusesource.examples.cbr (228)
-----
Manifest-Version = 1
Bnd-LastModified = 1334306872960
Archiver-Version = Plexus Archiver
Tool = Bnd-1.43.0
Originally-Created-By = Apache Maven
FAB-URL = mvn:org.fusesource.examples/cbr/7.0.0.fuse-beta-042
Generated-By-FAB-From = mvn:org.fusesource.ex
amples/cbr/7.0.0.fuse-beta-042
Built-By = username
FAB-Id = org.fusesource.examples:cbr:7.0.0.fuse-beta-042:jar
Build-Jdk = 1.6.0_29
Created-By = 1.6.0_29 (Apple Inc.)
```

```
Bundle-Name = org.fusesource.examples.cbr
Bundle-SymbolicName = org.fusesource.examples.cbr
Bundle-Version = 7.0.0.fuse-beta-042
Bundle-ManifestVersion = 2

Export-Package =
    OSGI-INF.blueprint,
    OSGI-INF
```

Arguments

[Table 9.1 on page 119](#) describes the commands arguments.

Table 9.1. *fab:headers Arguments*

Argument	Interpretation
--help	Displays the online help for this command
--indent	Specify the indent style. Valid values are 1, 2, or 3. The default is -1.
URL	The URL of the FAB

Name

`fab:info` — display information about a FAB, including the list of shared and unshared dependencies, and the list of features installed as part of the FAB resolution process

Synopsis

```
fab:info [--help] {bundleID}
```

Arguments

[Table 9.2 on page 120](#) describes the commands arguments.

Table 9.2. *fab:info* Arguments

Argument	Interpretation
<code>--help</code>	Displays the online help for this command
<i>bundleID</i>	The bundle ID of the FAB (for deployed FABs) or the URL location of the FAB (for FABs that are not yet deployed).

Name

fab:start — starts the specified FAB

Synopsis

fab:start [--help] [--timeout *millis*] {*bundleID*}

Description

Depending on how a FAB is configured, it can be associated with multiple dependent bundles.

When a FAB is initially installed in the container, the transitive dependencies of the FAB are determined by scanning the embedded POM file, `META-INF/maven/GroupID/ArtifactID/pom.xml`. Any transitive dependencies that are shared (for example, by being marked as provided or because the dependency is already packaged as an OSGi bundle), are deployed as separate OSGi bundles in the container.

When you start the FAB using `fab:start`, the runtime attempts to start *all* of the corresponding bundles, starting with the leaves of the dependency tree and working its way up the tree to the FAB's bundle. In particular, this implies that any OSGi services, blueprint XML files, and Spring XML files in the dependent OSGi bundles are activated in the appropriate order.

Arguments

[Table 9.3 on page 121](#) describes the commands arguments.

Table 9.3. fab:start Arguments

Argument	Interpretation
--help	Displays the online help for this command
--timeout	Specifies, in milliseconds, how long to wait for the FAB bundle to start up. Default is 30000.
<i>bundleID</i>	The bundle ID of the FAB.

Name

fab:stop — stops the specified FAB bundle together with its shared transitive dependencies, except for those dependencies that are being used by other bundles.

Synopsis

fab:stop [--help] {bundleID}

Arguments

[Table 9.4 on page 122](#) describes the commands arguments.

Table 9.4. fab:stop Arguments

Argument	Interpretation
--help	Displays the online help for this command
bundleID	The bundle ID of the FAB.

Name

fab:tree, tree — displays the dependency tree of a FAB

Synopsis

```
fab:tree [--help] {bundleID}
```

Arguments

[Table 9.5 on page 123](#) describes the commands arguments.

Table 9.5. fab:tree Arguments

Argument	Interpretation
--help	Displays the online help for this command
<i>bundleID</i>	Specifies the bundle ID, URL, or filename of the FAB.

Name

`fab:uninstall` — uninstall the specified FAB and all of its transitive dependencies, except for those dependencies that are being used by other bundles

Synopsis

```
fab:uninstall [--help] {bundleID}
```

Description

Depending on how a FAB is configured, it can be associated with multiple dependent bundles.

When a FAB is initially installed in the container, the transitive dependencies of the FAB are determined by scanning the embedded POM file, `META-INF/maven/GroupID/ArtifactID/pom.xml`. Any transitive dependencies that are shared (for example, by being marked as provided or because the dependency is already packaged as an OSGi bundle), are deployed as separate OSGi bundles in the container.

When you uninstall the FAB using `fab:uninstall`, the runtime attempts to uninstall all of the corresponding OSGi bundles, *except* for any bundles that are still being used by other applications in the container.

Arguments

[Table 9.6 on page 124](#) describes the commands arguments.

Table 9.6. *fab:uninstall* Arguments

Argument	Interpretation
<code>--help</code>	Displays the online help for this command
<code>bundleID</code>	The bundle ID of the FAB.

Chapter 10. Fabric Console Commands

fabric:cluster-list	127
fabric:cloud-firewall-edit	128
fabric:cloud-service-add	129
fabric:cloud-service-list	133
fabric:cloud-service-remove	134
fabric:container-add-profile	135
fabric:container-connect	136
fabric:container-create	138
fabric:container-create-child	141
fabric:container-create-cloud	144
fabric:container-create-ssh	149
fabric:container-delete	152
fabric:container-domains	153
fabric:container-list	154
fabric:container-remove-profile	155
fabric:container-resolver-list	156
fabric:container-resolver-set	157
fabric:container-rollback	160
fabric:container-start	161
fabric:container-stop	162
fabric:container-upgrade	163
fabric:create	165
fabric:ensemble-add	169
fabric:ensemble-list	171
fabric:ensemble-password	172
fabric:ensemble-remove	173
fabric:export	174
fabric:import	176
fabric:join	178
fabric:mq-create	181
fabric:profile-change-parents	183
fabric:profile-create	184
fabric:profile-delete	186
fabric:profile-display	187
fabric:profile-edit	188
fabric:profile-list	192
fabric:require-profile-delete	193
fabric:require-profile-list	194
fabric:require-profile-set	195
fabric:status	196

fabric:version-create 198
 fabric:version-delete 199
 fabric:version-list 200
 fabric:version-set-default 201

This chapter describes **fabric** console commands.

Name

fabric:cluster-list, cluster-list — lists the members of a Fuse MQ Enterprise cluster

Synopsis

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

Description

This command lists all message brokers in the fabric. It allows you to see which brokers are grouped into clusters. The resulting list will enable you to see which brokers are participating in a particular cluster.

Arguments

[Table 10.1 on page 127](#) describes the command's arguments.

Table 10.1. fabric:cluster-list Arguments

Argument	Interpretation
--help	Displays the online help for this command
<i>Path</i>	Specifies the path of the registry node to list. If not specified, all clusters will be listed.

Related topics

[fabric:mq-create on page 181](#)

Name

fabric:cloud-firewall-edit — manage a cloud container's firewall

Synopsis

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

Arguments

[Table 10.2 on page 128](#) describes the command's arguments.

Table 10.2. fabric:cloud-firewall-edit Arguments

Argument	Interpretation
--port	The target IP port.
--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.

Name

`fabric:cloud-service-add` — 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 Access
KeyID --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:

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

Arguments

[Table 10.3 on page 131](#) describes the command's arguments.

Table 10.3. *fabric:cloud-service-add Arguments*

Argument	Interpretation
<code>--help</code>	Displays the online help for this command.
<code>--provider</code>	The name of a commercial cloud provider (for example, <code>aws-ec2</code> or <code>rackspace</code>).
<code>--name</code>	The JClouds service context name, which identifies the cloud service uniquely. Defaults to the provider name (as specified by the <code>--provider</code> option).

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

Name

fabric:cloud-service-list — 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:

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

Arguments

[Table 10.4 on page 133](#) describes the command's arguments.

Table 10.4. *fabric:cloud-service-list Arguments*

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

Name

fabric:cloud-service-remove — 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:

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

Arguments

[Table 10.5 on page 134](#) describes the command's arguments.

Table 10.5. fabric:cloud-service-remove Arguments

Argument	Interpretation
<code>--help</code>	Displays the online help for this command.
<i>Name</i>	The JClouds service context name, which represents the cloud provider.

Name

fabric:container-add-profile, container-add-profile — Adds the specified list of profiles to a container

Synopsis

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

Arguments

[Table 10.6 on page 135](#) describes the command's arguments.

Table 10.6. fabric:container-add-profile Arguments

Argument	Interpretation
<code>--help</code>	Displays the online help for this command
<i>Name</i>	Specifies the name of the container.
<i>Profiles</i>	Specifies the list of profiles to add to the container.

Name

`fabric:container-connect`, `container-connect` — 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 10.1 on page 136](#).

Example 10.1. Executing a Command in a Remote Container

```
FuseMQ: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 10.7 on page 136](#) describes the command's arguments.

Table 10.7. *fabric:container-connect* Arguments

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

Related topics

["JAAS Console Commands" on page 215](#)

Name

`fabric:container-create`, `container-create` — creates one or more Fabric Containers

Synopsis

```
fabric:container-create [--help] [--parent ParentID] [--url URL]
[--proxy-uri ProxyURL] [--ensemble-server] [--profile ProfileID] [--resolver
policy] [--version Version] [--jvm-opts JvmOpts] {Name} [Number]
```

Examples

This command is a generic container create command. It combines the functionality of the `fabric:container-create-child`, `fabric:container-create-cloud`, and `fabric:container-create-ssh` commands. The type of container that is created, depends on the specified URL.

Child container

To create a child container, specify a URL in the following format:

```
child://ParentName
```

Where *ParentName* is the name of the child's parent container.

Cloud container

To create a cloud container, specify a URL in the following format:

```
jclouds://ProviderId?imageId=ImageID&locationId=LocationID&group=Group&user=User
```

For a detailed explanation of the options appearing in this URL, see [fabric:container-create-cloud on page 144](#).

SSH container

To create an SSH container with username and password credentials, specify a URL in the following format:

```
ssh://User:Password@Host:Port
```

Where *User* and *Password* are the credentials for logging in to the machine at *Host:Port*, through the SSH protocol.

To create an SSH container with username and private key credentials, specify a URL in the following format:

```
ssh://User@Host:Port?privateKeyFile=KeyPath
```

Where *KeyPath* is the pathname of the private key file on the local filesystem.

Arguments

[Table 10.8 on page 139](#) describes the command's arguments.

Table 10.8. *fabric:container-create* Arguments

Argument	Interpretation
--help	Displays the online help for this command
--parent	Specifies the parent container's ID.
--url	Specifies the URL of the new container.
--proxy-uri	Specifies the Maven proxy URI to use.
--ensemble-server	Specifies if the new container should be a Fabric Server.
--profile	Specifies a list of profiles to deploy into the new container.
--resolver	Specifies how the container will report its address to other containers. Valid values are <code>localip</code> , <code>localhostname</code> , <code>publicip</code> , <code>publichostname</code> , <code>manualip</code> . For more information see fabric:container-resolver-set on page 157 .
--version	Specifies the version of the profiles used by the new container. Defaults to the current default version.
--jvm-opts	Specifies options to pass to the container's JVM.
<i>Name</i>	Specifies the name of the new container. When creating multiple containers, the name serves as a prefix.
<i>Number</i>	Specifies the number of containers that should be created.

Related topics

[fabric:container-create-child on page 141](#)
[fabric:container-create-cloud on page 144](#)
[fabric:container-create-ssh on page 149](#)
[fabric:container-resolver-list on page 156](#)
[fabric:container-resolver-set on page 157](#)

Name

fabric:container-create-child — create one or more child containers

Synopsis

```
fabric:container-create-child [--help] [--ensemble-server] [--profile
profileID] [--version version] [--jvm-opts jvmOpts] [--resolver policy]
{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] status]	[version]	[alive]	[profiles]	[provision]
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. These 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. In particular, the newly created containers do *not* contain all of the features and bundles associated with a full Fuse ESB Enterprise container. If you want a child container to deploy all of the bundles associated with a full Fuse ESB Enterprise container, you can explicitly associate the child with the `esb` profile, as follows:

```
fabric:container-create-child --profile esb root childESB
```

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 10.9 on page 143](#) describes the command's arguments.

Table 10.9. *fabric:container-create-child* Arguments

Argument	Interpretation
<code>--help</code>	Displays the online help for this command
<code>--ensemble-server</code>	Whether the new container should be a Fabric Server.
<code>--profile</code>	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, <code>--profile foo --profile bar</code> . If no profile is specified, the container is associated with the <code>default</code> profile.
<code>--version</code>	Specifies the version of the new container (the version must be created in advance using <code>fabric:version-create</code>). Defaults to the current default version (use <code>version-list</code> to find the current default).
<code>--jvm-opts</code>	Specify options to pass to the container's JVM.
<code>--resolver</code>	Specifies how the container will report its address to other containers. Valid values are <code>localip</code> , <code>localhostname</code> , <code>publicip</code> , <code>publichostname</code> , <code>manualip</code> . For more information see fabric:container-resolver-set on page 157 .
<i>Parent</i>	(Required) The parent container ID.
<i>Name</i>	(Required) The name of the container to create. When creating multiple containers, it serves as a prefix
<i>Number</i>	The number of containers that should be created.

Related topics

For more details about resolver policies, see:

[fabric:container-resolver-list on page 156](#)

[fabric:container-resolver-set on page 157](#)

[fabric:create on page 165](#)

Name

`fabric:container-create-cloud` — 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] [--zookeeper-password zooPass]
[--resolver policy] [--min-port minPort] [--max-port maxPort] [--profile
profileID] [--version version] [--jvm-opts jvmOpts] [--add-option key=value]
[--no-admin-access] {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 10.10 on page 145](#) describes the command's arguments.

Table 10.10. `fabric:container-create-cloud` Arguments

Argument	Interpretation
<code>--help</code>	Displays the online help for this command
<code>--name</code>	(Required) JClouds service context name.
<code>--provider</code>	JClouds provider name.
<code>--api</code>	The cloud API name.

Argument	Interpretation
<code>--identity</code>	The identity used to access the cloud service.
<code>--credential</code>	The credential used to access the cloud service.
<code>--imageId</code>	The image ID to use for the new node(s). Alternatively, the image can be specified indirectly using the <code>--os-family</code> and <code>--os-version</code> options. Defaults to an instance of the latest version of Ubuntu.
<code>--os-family</code>	Specify the image by requesting a particular kind of operating system—for example, <code>ubuntu</code> or <code>redhat</code> . To see which O/S families are available, type <code>Tab</code> while entering this option. Defaults to <code>ubuntu</code> .
<code>--os-version</code>	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.
<code>--hardwareId</code>	Kind of hardware to use.
<code>--instanceType</code>	Type of instance required.
<code>--locationId</code>	The location used to create the new node(s).
<code>--user</code>	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.
<code>--password</code>	Specifies the password associated with the O/S user account defined by the <code>--user</code> option.
<code>--public-key-file</code>	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.
<code>--owner</code>	Optional owner of images; only really used for EC2, and will be deprecated in future.
<code>--group</code>	Group tag to use on the new node(s). Defaults to <code>fabric</code> .

Argument	Interpretation
<code>--proxy-uri</code>	URL of the Maven proxy server used to download the container runtime.
<code>--ensemble-server</code>	Whether the new container should be a Fabric Server (effectively creates a new fabric).
<code>--new-user</code>	<p>Used in combination with the <code>--ensemble-server</code> 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).</p> <p>When using this option, you <i>must</i> also specify a password using the <code>--new-user-password</code> option.</p>
<code>--new-user-password</code>	Used in combination with the <code>--new-user</code> option and the <code>--ensemble-server</code> option to specify the new user's password. No default value.
<code>--new-user-role</code>	Used in combination with the <code>--new-user</code> option and the <code>--ensemble-server</code> to specify the new user's role. Default is <code>admin</code> .
<code>--zookeeper-password</code>	<p>Used in combination with the <code>--ensemble-server</code> option. Specifies the Zookeeper password, which is used to access the Zookeeper nodes under the <code>/fabric/</code> path. Defaults to the password of the current session user.</p> <p>If you subsequently try to join the current container to the newly-created Fabric Server (ensemble server) using the <code>fabric:join</code> command, you will be prompted to enter the Zookeeper password.</p>
<code>--resolver</code>	Specifies how the container will report its address to other containers. Valid values are <code>localip</code> , <code>localhostname</code> , <code>publicip</code> , <code>publichostname</code> , <code>manualip</code> . For more information see fabric:container-resolver-set on page 157 .
<code>--min-port</code>	Specifies the minimum port number of the allowed IP port range. Default is <code>0</code> .
<code>--max-port</code>	Specifies the maximum port number of the allowed IP port range. Default is <code>65535</code> .

Argument	Interpretation
<code>--profile</code>	A list of profile IDs to associate with the new container.
<code>--version</code>	Specifies the version of the new container (the version must be created in advance using <code>fabric:version-create</code>). Defaults to the current default version (use <code>version-list</code> to find the current default).
<code>--jvm-opts</code>	Specify options to pass to the container's JVM.
<code>--add-option</code>	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 <code>--option subnetId=yoursubnetId</code> 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.
<code>--no-admin-access</code>	Disables admin access, as it might not be feasible on all images.
<i>Name</i>	<i>(Required)</i> The name of the container to create. When creating multiple containers, it serves as a prefix.
<i>Number</i>	The number of containers that should be created.

Related topics

See the other Fabric cloud commands:

- [fabric:cloud-provider-add on page 129](#)
- [fabric:cloud-provider-list on page 133](#)
- [fabric:cloud-provider-remove on page 134](#)

For more details about resolver policies, see:

- [fabric:container-resolver-list on page 156](#)
- [fabric:container-resolver-set on page 157](#)
- [fabric:create on page 165](#)

Name

`fabric:container-create-ssh` — creates one or more new containers through SSH

Synopsis

```
fabric:container-create-ssh [--help] [--host host] [--path path] [--user user] [--password password] [--private-key keyPath] [--port port] [--ssh-retries retries] [--proxy-uri URI] [--ensemble-server] [--profile profileID] [--version version] [--jvm-opts jvmOpts] [--resolver policy] {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 10.11 on page 150](#) describes the command's arguments.

Table 10.11. *fabric:container-create-ssh* Arguments

Argument	Interpretation
<code>--help</code>	Displays the online help for this command
<code>--host</code>	<i>(Required)</i> Host name to SSH into.
<code>--path</code>	Path on the remote filesystem where the container is to be installed.
<code>--user</code>	<i>(Required)</i> User name for login.
<code>--password</code>	Password for login. If the password is omitted, private key authentication is used instead.
<code>--private-key</code>	Specifies the path to the private key on the local file system. The default is <code>~/.ssh/id_rsa</code> on *NIX platforms or <code>C:\Documents and Settings\UserName\.ssh\id_rsa</code> on Windows.
<code>--port</code>	The IP port number for the SSH connection.
<code>--ssh-retries</code>	Maximum number of times to retry SSH connection.
<code>--proxy-uri</code>	URL of the Maven proxy server used to download the container runtime.
<code>--ensemble-server</code>	Whether the new container should be a Fabric Server.
<code>--profile</code>	A list of profile IDs to associate with the new container.
<code>--version</code>	Specifies the version of the new container (the version must be created in advance using <code>fabric:version-create</code>). Defaults to the current default version (use <code>version-list</code> to find the current default).
<code>--jvm-opts</code>	Specify options to pass to the container's JVM.

Argument	Interpretation
<code>--resolver</code>	Specifies how the container will report its address to other containers. Valid values are <code>localip</code> , <code>localhostname</code> , <code>publicip</code> , <code>publichostname</code> , <code>manualip</code> . For more information see fabric:container-resolver-set on page 157 .
<i>Name</i>	<i>(Required)</i> The name of the container to create. When creating multiple containers, it serves as a prefix.
<i>Number</i>	The number of containers that should be created.

Related topics

For more details about resolver policies, see:

[fabric:container-resolver-list on page 156](#)

[fabric:container-resolver-set on page 157](#)

[fabric:create on page 165](#)

Name

`fabric:container-delete`, `container-delete` — stops and deletes a Fuse Container

Synopsis

`fabric:container-delete` [--help] [--r] | [--recursive]] {*Name*}

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 10.12 on page 152](#) describes the command's arguments.

Table 10.12. *fabric:container-delete Arguments*

Argument	Interpretation
<code>--help</code>	Displays the online help for this command.
<code>-r</code> , <code>--recursive</code>	Recursively stops and deletes all child containers.
<i>Name</i>	Specifies the name of the container to delete.

Name

`fabric:container-domains`, `container-domains` — lists a container's JMX domains

Synopsis

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

Arguments

[Table 10.13 on page 153](#) describes the command's arguments.

Table 10.13. *fabric:container-domains* Arguments

Argument	Interpretation
<code>--help</code>	Displays the online help for this command.
<i>Name</i>	Specifies the name of the container.

Name

`fabric:container-list`, `container-list` — lists the containers in a fabric

Synopsis

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

Arguments

[Table 10.14 on page 154](#) describes the command's arguments.

Table 10.14. *fabric:container-list* Arguments

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

Name

`fabric:container-remove-profile`, `container-remove-profile` — removes the specified list of profiles from the container

Synopsis

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

Arguments

[fabric:container-remove-profile](#), on page 155 describes the command's arguments.

Table 10.15. *fabric:container-remove-profile Arguments*

Argument	Interpretation
<code>--help</code>	Displays the online help for this command
<i>Name</i>	Specifies the name of the container.
<i>Profiles</i>	Specifies the list of profiles to remove from the container.

Name

`fabric:container-resolver-list` — 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 10.16 on page 156](#) describes the command's arguments.

Table 10.16. `fabric:container-resolver-list` Arguments

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

Related topics

[fabric:container-resolver-set on page 157](#)

Name

`fabric:container-resolver-set` — 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 10.17 on page 159](#) describes the command's arguments.

Table 10.17. *fabric:container-resolver-set Arguments*

Argument	Interpretation
<code>--help</code>	Displays the online help for this command
<code>--container</code>	Apply the resolver policy to the specified container.
<code>--all</code>	Apply the resolver policy to all containers in the fabric.
<i>Resolver</i>	<i>(Required)</i> The resolver policy to set on the specified container(s). Possible values are: <code>localip</code> , <code>localhostname</code> , <code>publicip</code> , <code>publichostname</code> , <code>manualip</code> .

Name

fabric:container-rollback — 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 on page 163](#).

Arguments

[Table 10.18 on page 160](#) describes the command's arguments.

Table 10.18. 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.

Name

`fabric:container-start`, `container-start` — start the specified container

Synopsis

`fabric:container-start` [--help] {*name*}

Arguments

[Table 10.19 on page 161](#) describes the command's arguments.

Table 10.19. *fabric:container-start* Arguments

Argument	Interpretation
<code>--help</code>	Displays the online help for this command
<i>Name</i>	Specifies the name of the container.

Name

fabric:container-stop, container-stop — shuts down the specified container

Synopsis

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

Arguments

[Table 10.20 on page 162](#) describes the command's arguments.

Table 10.20. fabric:container-stop Arguments

Argument	Interpretation
--help	Displays the online help for this command
<i>Name</i>	(<i>Required</i>) The name of the container.

Name

`fabric:container-upgrade` — 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 --features 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 10.21 on page 164](#) describes the command's arguments.

Table 10.21. `fabric:container-upgrade` Arguments

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

Name

`fabric:create` — 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] [--new-user username] [--new-user-password password]  
[--new-user-role role] [--zookeeper-password zooPassword]  
[--generate-zookeeper-password] [--g | [--global-resolver]policy] [--r |  
--resolver]policy] [--m | [--manual-ip]ipAddress] [--min-port port]  
[--max-port port] [ContainerList]
```

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).

If you want to create your own import directory with custom profile data, it is recommended that you proceed as follows:

1. Create a fabric that imports the sample profiles (for example, using `fabric:create`).
2. Modify the sample profiles using the `fabric:profile-create`, `fabric:profile-delete`, and `fabric:edit` commands.
3. Export the modified profiles using the `fabric:export` command.

Arguments

Table 10.22 on page 166 describes the command's arguments.

Table 10.22. *fabric:create* Arguments

Argument	Interpretation
<code>--help</code>	Displays the online help for this command
<code>--clean</code>	Clean local zookeeper cluster and configurations.
<code>--no-import</code>	Disable the import of the sample registry data.
<code>--import-dir</code>	Directory of files to import into the newly created ensemble.
<code>-v, --verbose</code>	Flag to enable verbose output of files being imported.
<code>-t, --time</code>	How long to wait (milliseconds) for the ensemble to start up, before trying to import the default data.
<code>-n, --non-managed</code>	Specifies that the container remains unmanaged.
<code>-p, --profile</code>	Specifies the profile to use for the ensemble containers in the new fabric.
<code>--new-user</code>	<p>Create a new user in the new fabric's JAAS realm. Because the <code>fabric:create</code> command automatically imports user data from the <code>etc/users.properties</code> file, you would only need to specify this option, if the <code>etc/users.properties</code> file contains no valid user entries.</p> <p>When using this option, you <i>must</i> also specify a password using the <code>--new-user-password</code> option.</p>
<code>--new-user-password</code>	Used in combination with the <code>--new-user</code> option to specify the new user's password. No default value.
<code>--new-user-role</code>	Used in combination with the <code>--new-user</code> option to specify the new user's role. Default is <code>admin</code> .
<code>--zookeeper-password</code>	Specifies the Zookeeper password, which is used to access the Zookeeper nodes under the <code>/fabric/</code> path. Defaults to the password of the current session user.

Argument	Interpretation
	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 <code>fabric:ensemble-password</code> command.
<code>--generate-zookeeper-password</code>	Directs Fabric to generate a random Zookeeper password. Subsequently, you can display the Zookeeper password using the <code>fabric:ensemble-password</code> command.
<code>-g, --global-resolver</code>	Specifies the global resolver policy, which becomes the default resolver policy applied to all new containers created in this fabric. Possible values are: <code>localip</code> , <code>localhostname</code> , <code>publicip</code> , <code>publichostname</code> , <code>manualip</code> . The default is <code>localhostname</code> .
<code>-r, --resolver</code>	Specifies the local resolver policy. Possible values are: <code>localip</code> , <code>localhostname</code> , <code>publicip</code> , <code>publichostname</code> , <code>manualip</code> . The default is <code>localhostname</code> .
<code>-m, --manual-ip</code>	If you select the <code>manualip</code> resolver policy (using either the <code>--resolver</code> or <code>--global-resolver</code> options), specifies the IP address to use for the resolver.
<code>--min-port</code>	Specifies the minimum port number of the allowed IP port range. Default is 0.
<code>--max-port</code>	Specifies the maximum port number of the allowed IP port range. Default is 65535.
<code>ContainerList</code>	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:

- [fabric:container-resolver-list on page 156](#).
- [fabric:container-resolver-set on page 157](#).

Name

fabric:ensemble-add — 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 10.23 on page 169](#) describes the command's arguments.

Table 10.23. *fabric:ensemble-add* Arguments

Argument	Interpretation
<code>--help</code>	Displays the online help for this command

Argument	Interpretation
<i>ContainerList</i>	The list of containers to add.

Name

`fabric:ensemble-list` — 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 10.24 on page 171](#) describes the command's arguments.

Table 10.24. *fabric:ensemble-list* Arguments

Argument	Interpretation
<code>--help</code>	Displays the online help for this command

Name

`fabric:ensemble-password` — display the ensemble password

Synopsis

`fabric:ensemble-password [--help]`

Description

The ensemble password protects access to the Zookeeper nodes under the `/fabric/` path, which contains critical configuration data for the fabric. To ensure integrity of the fabric configuration data, you should modify the fabric configuration exclusively using the `fabric:*` console commands.

Arguments

[Table 10.25 on page 172](#) describes the command's arguments.

Table 10.25. `fabric:ensemble-password` Arguments

Argument	Interpretation
<code>--help</code>	Displays the online help for this command

Name

fabric:ensemble-remove — 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 10.26 on page 173](#) describes the command's arguments.

Table 10.26. 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.

Name

`fabric:export` — export the contents of the Fabric Registry to the specified directory in the filesystem

Synopsis

```
fabric:export [--help] [-d|--delete] [-p|--path path] [-f|--regex regex]  
[-rf|--reverse-regex regex] [-t|--trim] [--dry-run] {target}
```

Description

The output of this command is compatible with the import options of the other `fabric` commands.

The regular expression options, `-f` and `-rf`, provide you with considerable flexibility at specifying which parts of the Fabric Registry to export. For example, to export every version of the `default` profile's data, you could use a command like the following:

```
fabric:export -f /fabric/configs/versions/[0-9\\.]*profiles/default/.*
```

Where a double-backslash, `\\`, is required to escape the period, `.`, so that the period gets interpreted as a character literal.

Arguments

[Table 10.27 on page 174](#) describes the commands arguments.

Table 10.27. *fabric:export Arguments*

Argument	Interpretation
<code>--help</code>	Displays the online help for this command
<code>-d, --delete</code>	Delete the existing contents of the target directory before exporting. Caution: Performs a recursive delete!
<code>-p, --path</code>	Top-level znode to export. Default is <code>/</code> .
<code>-f, --regex</code>	Specifies a regular expression that matches the znode paths you want to <i>include</i> in the export. For multiple include expressions, specify this option multiple times.

Argument	Interpretation
	The regular expression syntax is defined by the <code>java.util.regex</code> package.
<code>-rf, --reverse-regex</code>	Specifies a regular expression that matches the znode paths you want to <i>exclude</i> from the export. For multiple exclude expressions, specify this option multiple times. The regular expression syntax is defined by the <code>java.util.regex</code> package.
<code>-t, --trim</code>	Trims the first timestamp comment line in properties files starting with the <code>#</code> character.
<code>--dry-run</code>	Log the actions that would be performed during an export, but do not actually perform the export.
<code>target</code>	Path of the directory to export to. Default is <code>./export</code> .

Name

`fabric:import` — import data either from a filesystem or from a properties file into the Fabric Registry

Synopsis

```
fabric:import [--help] [-fs|--filesystem] [-props|--properties URL] [-t|--target path] [-d|--delete] [-f|--regex regex] [-rf|--reverse-regex regex] [-v|--verbose] [--dry-run] {source}
```

Arguments

[Table 10.28 on page 176](#) describes the commands arguments.

Table 10.28. *fab:start Arguments*

Argument	Interpretation
<code>--help</code>	Displays the online help for this command
<code>-fs,--filesystem</code>	Indicates that the <i>source</i> argument is a directory on the filesystem. Defaults to <code>true</code> .
<code>-props,--properties</code>	Indicates that the <i>source</i> argument is a properties file. Defaults to <code>false</code> .
<code>-t,--target</code>	Path of the znode that the data is imported into. Default is <code>/</code> .
<code>-d,--delete</code>	Delete any paths that are not in the tree being imported. Ignored when importing a properties file. Caution: Using this option could permanently delete all or part of the Fabric Registry.
<code>-f,--regex</code>	Specifies a regular expression that matches the znode paths you want to <i>include</i> in the import. For multiple include expressions, specify this option multiple times. The regular expression syntax is defined by the <code>java.util.regex</code> package.
<code>-rf,--reverse-regex</code>	Specifies a regular expression that matches the znode paths you want to <i>exclude</i> from the import. For multiple exclude expressions, specify this option multiple times.

Argument	Interpretation
	The regular expression syntax is defined by the <code>java.util.regex</code> package.
<code>-v, --verbose</code>	Verbose log of files being imported.
<code>--dry-run</code>	Log the actions that would be performed during an import, but do not actually perform the import.
<code>source</code>	Location of a filesystem (if <code>--filesystem</code> is specified) or a properties file (if <code>--properties</code> is specified). Defaults to <code>./import</code> .

Name

fabric:join — 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] URL [ContainerName]
```

Description

The `fabric:join` command can be used in either of the following scenarios:

- You have an existing fabric, A, and you want to join a standalone container to fabric A.
- You have two separate fabrics, A and B, and you want to transfer a container from fabric B to fabric A.

Arguments

[Table 10.29 on page 178](#) describes the command's arguments.

Table 10.29. *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 <code>fabric</code> 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.

Argument	Interpretation
<code>--zookeeper-password</code>	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.
<code>-r, --resolver</code>	Specifies the local resolver policy. Possible values are: <code>localip</code> , <code>localhostname</code> , <code>publicip</code> , <code>publichostname</code> , <code>manualip</code> . The default is <code>localhostname</code> .
<code>-m, --manual-ip</code>	If you select the <code>manualip</code> resolver policy (using the <code>--resolver</code> option), specifies the IP address to use for the resolver.
<code>--min-port</code>	Specifies the minimum port number of the allowed IP port range. Default is 0.
<code>--max-port</code>	Specifies the maximum port number of the allowed IP port range. Default is 65535.
<i>URL</i>	Specifies the URL of one of the Fabric Servers, specified in the format <code>Host[:Port]</code> . The <code>Port</code> value defaults to 2181.
<i>ContainerName</i>	Specifies a unique name for the container to use when joining the fabric. By default, the value of the <code>karaf.name</code> property from the <code>etc/system.properties</code> 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.

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
```

Name

fabric:mq-create — create a new broker profile

Synopsis

```
fabric:mq-create [--help] [--group groupName] [--networks  
brokerGroup,... ] [--create-container containerID,... ] [--assign-container  
containerID,... ] [--config configFile] [--data dataDir] [--version  
version] {name}
```

Arguments

[Table 10.30 on page 181](#) describes the command's arguments.

Table 10.30. fabric:mq-create Arguments

Argument	Description
--help	Displays the online help for this command.
--group <i>groupName</i>	Specifies the name of the group to which brokers using this profile are assigned. By default brokers are assigned to the <code>default</code> group.
--networks <i>brokerGroup</i> ,...	Specifies a comma separated list of broker groups to which brokers using this profile will establish network connections to form a network of brokers. See Using Networks of Brokers for more information.
--create-container <i>containerID</i> ,...	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 <i>containerID</i> ,...	Specifies a comma separated list of containers to which the new profile will be deployed.
--config <i>configFile</i>	Specifies the ensemble path of the XML configuration template used by the profile. The path will have the syntax <code>/fabric/configs/versions/<i>version</i>/profiles/<i>profile</i>/config.xml</code> .
--data <i>dataDir</i>	Specifies the path, relative to the container, for storing the persistence data for a broker using the profile.

Argument	Description
<code>--version <i>version</i></code>	Specifies the version into which the profile is stored. Defaults to the current default version.
<code><i>name</i></code>	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
```

Name

`fabric:profile-change-parents` — 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 10.31 on page 183](#) describes the command's arguments.

Table 10.31. *fabric:profile-change-parents* Arguments

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

Name

`fabric:profile-create` — create a new profile with the specified name and version

Synopsis

```
fabric:profile-create [--help] [--version version] [--parents parentList]
{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 --parents mq --parents camel test
```

Arguments

[Table 10.32 on page 184](#) describes the command's arguments.

Table 10.32. `fabric:profile-create` Arguments

Argument	Interpretation
<code>--help</code>	Displays the online help for this command
<code>--version</code>	The profile version. Defaults to the current default version (use <code>version-list</code> to find the current default).
<code>--parents</code>	Optionally specifies one or multiple parent profiles. To specify multiple parent profiles, specify this flag multiple times on the command line—for example, <code>--parents foo --parents bar</code> .

Argument	Interpretation
<i>Name</i>	<i>(Required)</i> Name of the new profile.

Name

`fabric:profile-delete` — 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 10.33 on page 186](#) describes the command's arguments.

Table 10.33. *fabric:profile-delete* Arguments

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

Name

fabric:profile-display — 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 10.34 on page 187](#) describes the command's arguments.

Table 10.34. *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 <code>version-list</code> to find the current default).
-o, --overlay	Enable overlay. Shows the effective profile settings, taking into account the settings inherited from parent profiles.
<i>Profile</i>	(Required) The name of the profile.

Name

`fabric:profile-edit` — 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] | [--repositories] |
[-f] | [--features] | [-b] | [--bundles] | [-c] | [--config] | [-s] | [--system]]
[--set] | [--delete]] [--import-pid] {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 --repositories RepoList Profile [Version]
```

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

```
fabric:profile-edit --repositories mvn:org.fusesource.fabric/fuse-fabric/7.1.0.fuse-047/xml/features Profile [Version]
```

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

```
fabric:profile-edit --delete --repositories RepoList Profile
[Version]
```

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

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

Where *FeatureList* is a comma-separated list of features. 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 --features camel-jetty,camel-quartz
sample
```

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

```
fabric:profile-edit --delete --features FeatureList Profile [Version]
```

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

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

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

```
fabric:profile-edit --bundles mvn:org.apache.camel/camel-quartz/ sample
```

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

```
fabric:profile-edit --delete --bundles BundleList Profile [Version]
```

- *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]
```

- *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 Apache Karaf 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]
```

- *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]
```

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

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



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 on page 163](#).

Arguments

[Table 10.35 on page 191](#) describes the command's arguments.

Table 10.35. *fabric:profile-edit* Arguments

Argument	Interpretation
<code>--help</code>	Displays the online help for this command
<code>-p, --pid</code>	Edit an OSGi configuration property, specified in the format <i>PID/Property</i> .
<code>-r, --repositories</code>	Edit the list of feature repositories.
<code>-f, --features</code>	Edit features, specifying a comma-separated list of features to add (or delete).
<code>-b, --bundles</code>	Edit bundles, specifying a comma-separated list of bundles to add (or delete).
<code>-c, --config</code>	Edit the Java system properties that affect the Apache Karaf container (analogous to editing <code>etc/config.properties</code> in a root container).
<code>-s, --system</code>	Edit the Java system properties that affect installed bundles (analogous to editing <code>etc/system.properties</code> in a root container).
<code>--set</code>	Set or create values (selected by default).
<code>--delete</code>	Delete values.
<code>--import-pid</code>	Imports the PIDs that are edited, from local OSGi Config Admin.
<i>Profile</i>	<i>(Required)</i> Name of the profile to edit.
<i>Version</i>	Version of the profile to edit. Defaults to the current default version (use <code>version-list</code> to find the current default).

Name

fabric:profile-list — 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]
```

Description

Arguments

[Table 10.36 on page 192](#) describes the command's arguments.

Table 10.36. 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 <code>version-list</code> to find the current default).

Name

`fabric:require-profile-delete` — deletes requirements on the specified profile

Synopsis

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

Arguments

[Table 10.37 on page 193](#) describes the command's arguments.

Table 10.37. *fabric:require-profile-delete* Arguments

Argument	Interpretation
<code>--help</code>	Displays the online help for this command
<i>Profile</i>	A profile ID.

Name

fabric:require-profile-list — 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 10.38 on page 194](#) describes the command's arguments.

Table 10.38. fabric:require-profile-list Arguments

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

Name

fabric:require-profile-set — 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 ex  
ample-camel
```

Arguments

[Table 10.39 on page 195](#) describes the command's arguments.

Table 10.39. *fabric:require-profile-set* Arguments

Argument	Interpretation
<code>--help</code>	Displays the online help for this command
<code>--minimum</code>	The minimum number of instances of this profile expected to be running in the fabric.
<code>--maximum</code>	The maximum number of instances of this profile expected to be running in the fabric.
<code>--dependsOn</code>	The profile IDs that must be provisioned before this profile.
<i>Profile</i>	A profile ID.

Name

`fabric:status` — 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 10.40 on page 196](#) describes the command's arguments.

Table 10.40. *fabric:status Arguments*

Argument	Interpretation
<code>--help</code>	Displays the online help for this command

Related topics

For more details, see:

- [fabric:require-profile-set on page 195](#)

- [fabric:require-profile-list on page 194](#)
- [fabric:require-profile-delete on page 193](#)

Name

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

Arguments

[Table 10.41 on page 198](#) describes the command's arguments.

Table 10.41. fabric:version-create Arguments

Argument	Interpretation
<code>--help</code>	Displays the online help for this command
<code>--default</code>	Set the created version to be the new default version.
<code>--parent</code>	The parent version. By default, use the latest version as the parent.
<i>Version</i>	The new version to create. If not specified, defaults to the next minor version.

Name

fabric:version-delete — 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.

Arguments

[Table 10.42 on page 199](#) describes the command's arguments.

Table 10.42. *fabric:version-delete Arguments*

Argument	Interpretation
--help	Displays the online help for this command
<i>Version</i>	(<i>Required</i>) The version to delete.

Name

`fabric:version-list` — 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 10.43 on page 200](#) describes the command's arguments.

Table 10.43. `fabric:version-list` Arguments

Argument	Interpretation
<code>--help</code>	Displays the online help for this command

Name

`fabric:version-set-default` — 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 10.44 on page 201](#) describes the command's arguments.

Table 10.44. *fabric:version-set-default* Arguments

Argument	Interpretation
<code>--help</code>	Displays the online help for this command
<i>Version</i>	<i>(Required)</i> Version number to use as the new default version.

Chapter 11. Features Console Commands

features:addurl	204
features:chooseurl	205
features:info	206
features:install	207
features:list	208
features:listurl	209
features:listVersions	210
features:refreshUrl	211
features:removeUrl	212
features:removeRepository	213
features:uninstall	214

The **features** commands allow you to provision entire applications using the Fuse ESB Enterprise 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.

Name

features:addurl, addurl — 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 11.1 on page 204](#) describes the command's arguments.

Table 11.1. 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.

Name

features:chooseurl, chooseurl — registers the feature repository URL for a well known project

Synopsis

```
features:chooseurl [--help] {project} {version}
```

Description

Fuse ESB Enterprise 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 11.2 on page 205](#) describes the command's arguments.

Table 11.2. 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.

Name

features:info — 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 11.3 on page 206](#) describes the command's arguments.

Table 11.3. 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.
-d, --dependency	Display dependency information.
command	

Name

features:install — installs a feature

Synopsis

```
features:install [--help] {name} [version]
```

Arguments

[Table 11.4 on page 207](#) describes the command's arguments.

Table 11.4. features:install Arguments

Argument	Interpretation
--help	Displays the online help for this command
<i>name</i>	The name of the feature to install
<i>version</i>	The version of the feature

Name

features:list — Lists all existing features available from the defined repositories

Synopsis

features:list [--help] [--i] | [--installed]

Arguments

[Table 11.5 on page 208](#) describes the command's arguments.

Table 11.5. features:list Arguments

Argument	Interpretation
--help	Displays the online help for this command
-i, --installed	Displays the list of all installed features

Name

features:listurl — lists the features repository URLs

Synopsis

```
features:listurl [--help] [[-v] | [--validate]] [[-vo] | [--verbose]]
```

Arguments

[Table 11.6 on page 209](#) describes the command's arguments.

Table 11.6. 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.

Name

`features:listVersions`, `listVersions` — lists all versions of a feature available from the current feature repositories

Synopsis

```
features:listVersions [--help] {feature}
```

Arguments

[Table 11.7 on page 210](#) describes the command's arguments.

Table 11.7. *features:listVersions* Arguments

Argument	Interpretation
<code>--help</code>	Displays the online help for this command
<i>feature</i>	Name of a feature.

Name

`features:refreshUrl` — reloads the list of available features from the repositories

Synopsis

```
features:refreshUrl [--help] {urls}
```

Arguments

[Table 11.8 on page 211](#) describes the command's arguments.

Table 11.8. `features:refreshUrl` Arguments

Argument	Interpretation
<code>--help</code>	Displays the online help for this command
<code>urls</code>	The repository URLs to reload (leave empty for all).

Name

`features:removeUrl` — removes the specified list of repository URLs from the features service

Synopsis

```
features:removeUrl [--help] {urls}
```

Arguments

[Table 11.9 on page 212](#) describes the command's arguments.

Table 11.9. `features:removeUrl` Arguments

Argument	Interpretation
<code>--help</code>	Displays the online help for this command
<code>-n,--interval</code>	
<code>urls</code>	One or more repository URLs separated by whitespace.

Name

`features:removeRepository` — removes the specified repository from the features service

Synopsis

```
features:removeRepository [--help] {repository}
```

Arguments

[Table 11.10 on page 213](#) describes the command's arguments.

Table 11.10. *features:removeRepository* Arguments

Argument	Interpretation
<code>--help</code>	Displays the online help for this command
<code>repository</code>	The name of a features repository.

Name

`features:uninstall` — uninstalls a feature with the specified name and version

Synopsis

```
features:uninstall [--help] {features}
```

Arguments

[Table 11.11 on page 214](#) describes the command's arguments.

Table 11.11. `features:uninstall` Arguments

Argument	Interpretation
<code>--help</code>	Displays the online help for this command
<i>features</i>	A space-separated list of features to uninstall, where each feature is specified in the format <i>feature[/version]</i> (that is, the version is optional).

Chapter 12. JAAS Console Commands

jass:cancel	217
jaas:manage	218
jaas:pending	220
jaas:realms	221
jaas:roleadd	222
jaas:roledel	223
jaas:update	224
jaas:useradd	225
jaas:userdel	226
jaas:users	227

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.

- **jass:users**

Lists all of the users.

- **jass:useradd**

Add a new user.

- **jass:userdel**

Delete a user.

- **jass:roleadd**

Add a new role to a user.

- **jass: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.

Name

jaas:cancel, cancel — 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 12.1 on page 217](#) describes the command's arguments.

Table 12.1. jaas:cancel Arguments

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

Name

jaas:manage, manage — 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 12.2 on page 218](#) describes the command's arguments.

Table 12.2. 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	org.fusesource.fabric.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
```

Name

jaas:pending, pending — 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 12.3 on page 220](#) describes the command's arguments.

Table 12.3. jaas:pending Arguments

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

Name

`jaas:realms`, `realms` — lists the JAAS realms known to the container

Synopsis

`jaas:realms [--help]`

Arguments

[Table 12.4 on page 221](#) describes the command's arguments.

Table 12.4. *jaas:realms* Arguments

Argument	Interpretation
<code>--help</code>	Displays the online help for this command.

Name

`jaas:roleadd`, `roleadd` — 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 12.5 on page 222](#) describes the command's arguments.

Table 12.5. *jaas:roleadd Arguments*

Argument	Interpretation
<code>--help</code>	Displays the online help for this command.
<code>username</code>	Specifies the name of the user to modify.
<code>role</code>	Specifies the role which is appended to the user data.

Name

`jaas:roledel`, `roledel` — 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 12.6 on page 223](#) describes the command's arguments.

Table 12.6. *jaas:roledel* Arguments

Argument	Interpretation
<code>--help</code>	Displays the online help for this command.
<code>username</code>	Specifies the name of the user to modify.
<code>role</code>	Specifies the role which is removed from the user data.

Name

`jaas:update` — 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 12.7 on page 224](#) describes the command's arguments.

Table 12.7. *jaas:update* Arguments

Argument	Interpretation
<code>--help</code>	Displays the online help for this command.

Name

`jaas:useradd`, `useradd` — 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 12.8 on page 225](#) describes the command's arguments.

Table 12.8. `jaas:useradd` Arguments

Argument	Interpretation
<code>--help</code>	Displays the online help for this command.
<code>username</code>	Specifies the name of the user to add.
<code>password</code>	Specifies the password used to authenticate the user.

Name

jaas:userdel, userdel — 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 12.9 on page 226](#) describes the command's arguments.

Table 12.9. jaas:userdel Arguments

Argument	Interpretation
--help	Displays the online help for this command.
<i>username</i>	Specifies the name of the user to add.

Name

jaas:users, users — lists the users in the JAAS realm being edited

Synopsis

jaas:users [--help]

Arguments

[Table 12.10 on page 227](#) describes the command's arguments.

Table 12.10. jaas:users Arguments

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

Chapter 13. JBI Console Commands

jbi:list	230
jbi:shutdown	231
jbi:start	232
jbi:stop	233

The **jbi** commands allow you to manage JBI artifacts that are deployed in the Fuse ESB Enterprise runtime. For information about working with JBI artifacts in Fuse ESB Enterprise, see [Using Java Business Integration](#).

Type **jbi:** then press **Tab** at the `karaf>` prompt to view the available commands.

Name

`jbi:list` — lists JBI endpoints

Synopsis

`jbi:list [--help]`

Arguments

This command takes the following arguments.

Table 13.1. *jbi:list* Arguments

Argument	Interpretation
<code>--help</code>	Displays the online help for this command

Name

jbi:shutdown — shuts down a JBI artifact

Synopsis

```
jbi:shutdown [--help] [[-a] | [--service-assembly] serviceAssembly] [[-c] | [--component] component] [--force] {VAL}
```

Arguments

This command takes the following arguments.

Table 13.2. jbi:shutdown Arguments

Argument	Interpretation
--help	Displays the online help for this command
-a, --service-assembly	Specifies a service assembly
-c, --component	Specifies a component
--force	Forces a shutdown of the specified JBI artifact
<i>VAL</i>	The name of the JBI artifact

Name

jbi:start — starts a JBI artifact

Synopsis

```
jbi:start [--help] [[-a] | [--service-assembly] serviceAssembly] [--c] |  
[--component] component] {VAL}
```

Arguments

This command takes the following arguments.

Table 13.3. *jbi:start Arguments*

Argument	Interpretation
--help	Displays the online help for this command
-a, --service-assembly	Specifies a service assembly
-c, --component	Specifies a component
<i>VAL</i>	The name of the JBI artifact

Name

`jbi:stop` — stops a JBI artifact

Synopsis

```
jbi:stop [--help] [[-a] | [--service-assembly] serviceAssembly] [[-c] |  
[--component] component] {VAL}
```

Arguments

This command takes the following arguments.

Table 13.4. *jbi:stop Arguments*

Argument	Interpretation
<code>--help</code>	Displays the online help for this command
<code>-a, --service-assembly</code>	A service assembly
<code>-c, --component</code>	A component
<i>VAL</i>	The name of the JBI artifact

Chapter 14. Log Console Commands

log:clear	236
log:display	237
log:display-exception	238
log:get	239
log:set	240
log:tail	241

The log commands allow you to display and change log levels. For information about logging, see ["Using Logging"](#) in *Managing and Monitoring a Broker*.

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

Name

`log:clear` — clears the log

Synopsis

`log:clear` [--help]

Arguments

[Table 14.1 on page 236](#) describes the command's arguments.

Table 14.1. `log:clear` Arguments

Argument	Interpretation
<code>--help</code>	Displays the online help for this command

Name

log:display, display, ld — displays log entries

Synopsis

log:display [--help] [-p *pattern*] [-n *numLines*] [--no-color]

Arguments

[Table 14.2 on page 237](#) describes the command's arguments.

Table 14.2. *log:display* 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.

Name

`log:display-exception`, `display-exception`, `lde` — displays the last thrown exception from the log

Synopsis

```
log:display-exception [--help]
```

Arguments

[Table 14.3 on page 238](#) describes the command's arguments.

Table 14.3. *log:display-exception* Arguments

Argument	Interpretation
<code>--help</code>	Displays the online help for this command

Name

log:get, get — shows the log level

Synopsis

log:get [--help] {*logger*}

Arguments

[Table 14.4 on page 239](#) describes the command's arguments.

Table 14.4. *log:get Arguments*

Argument	Interpretation
--help	Displays the online help for this command
<i>logger</i>	Specifies the logger name, ALL, or ROOT. The default is ROOT.

Name

`log:set, set` — sets the log level

Synopsis

```
log:set [--help] {[DEFAULT] | [TRACE] | [DEBUG] | [INFO] | [WARN] |  
[ERROR]} {logger}
```

Arguments

[Table 14.5 on page 240](#) describes the command's arguments.

Table 14.5. `log:set` Arguments

Argument	Interpretation
<code>--help</code>	Displays the online help for this command
<code>level</code>	Specifies the logging level.
<code>logger</code>	Specifies the logger name. The default is <code>ROOT</code> .

Name

log:tail — continually displays log entries

Synopsis

log:tail [--help] [-p *pattern*] [-n *numLines*] [--no-color]

Arguments

[Table 14.6 on page 241](#) describes the command's arguments.

Table 14.6. 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 15. The nmr:list Command

nmr:list	244
----------------	-----

The **nmr:list** command allows you to list normalized message router (NMR) endpoints. The NMR enables the interaction between different components in the OSGi container. The main purpose of the NMR is to mediate between service consumers and service providers.

Name

nmr:list — lists NMR endpoints

Synopsis

```
nmr:list [--help]
```

Arguments

This command takes the following arguments.

Table 15.1. *nmr:list Arguments*

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

Chapter 16. OBR Console Commands

obr:addUrl	246
obr:deploy	247
obr:info	248
obr:list	249
obr:listUrl	250
obr:refreshUrl	251
obr:removeUrl	252
obr:source	253
obr:start	254

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:

```
karaf@root:> features:install obr
```

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

Name

`obr:addUrl` — adds a list of repository URLs to the OBR service

Synopsis

```
obr:addUrl [--help] {urls}
```

Arguments

This command takes the following arguments.

Table 16.1. *obr:addUrl* Arguments

Argument	Interpretation
<code>--help</code>	Displays the online help for this command
<code>urls</code>	The repository URLs to add to the OBR service, separated by whitespaces

Name

obr:deploy — deploys a list of bundles using the OBR service

Synopsis

```
obr:deploy [--help] {bundles}
```

Arguments

This command takes the following arguments.

Table 16.2. *obr:deploy Arguments*

Argument	Interpretation
--help	Displays the online help for this command
<i>bundles</i>	A list of bundle names to deploy, separated by whitespaces

Name

`obr:info` — prints information about OBR bundles

Synopsis

```
obr:info [--help] {bundles}
```

Arguments

This command takes the following arguments.

Table 16.3. *obr:info* Arguments

Argument	Interpretation
<code>--help</code>	Displays the online help for this command
<i>bundles</i>	Specifies the bundles to query for information, separated by whitespaces

Name

`obr:list` — lists OBR bundles

Synopsis

`obr:list [--help] {args}`

Arguments

This command takes the following arguments.

Table 16.4. *obr:list* Arguments

Argument	Interpretation
<code>--help</code>	Displays the online help for this command
<code>args</code>	The arguments

Name

obr:listUrl — displays the repository URLs currently associated with the OBR service

Synopsis

```
obr:listUrl [--help]
```

Arguments

This command takes the following arguments.

Table 16.5. *obr:listUrl Arguments*

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

Name

`obr:refreshUrl` — reloads the repositories to obtain a fresh list of bundles

Synopsis

```
obr:refreshUrl [--help] {urls}
```

Arguments

This command takes the following arguments.

Table 16.6. *obr:refreshUrl Arguments*

Argument	Interpretation
<code>--help</code>	Displays the online help for this command
<code>urls</code>	The repository URLs to refresh (leave empty for all)

Name

`obr:removeUrl` — removes a list of repository URLs from the OBR service

Synopsis

```
obr:removeUrl [--help] {urls}
```

Arguments

This command takes the following arguments.

Table 16.7. *obr:removeUrl Arguments*

Argument	Interpretation
<code>--help</code>	Displays the online help for this command
<code>urls</code>	The repository URLs to remove from the OBR service, separated by whitespace

Name

`obr:source` — downloads the sources for an OBR bundle

Synopsis

```
obr:source [--help] [-x] {folder} {bundles}
```

Arguments

This command takes the following arguments.

Table 16.8. *obr:source Arguments*

Argument	Interpretation
<code>--help</code>	Displays the online help for this command
<code>-x</code>	Extracts the archive
<i>folder</i>	The local directory or folder for storing sources
<i>bundles</i>	A list of bundles to download the sources for

Name

`obr:start` — deploys and starts a list of bundles using OBR

Synopsis

```
obr:start [--help] {bundles}
```

Arguments

This command takes the following arguments.

Table 16.9. *obr:start* Arguments

Argument	Interpretation
<code>--help</code>	Displays the online help for this command
<i>bundles</i>	List of bundle names to deploy, separated by whitespaces

Chapter 17. OSGi Console Commands

osgi:bundle-level	256
osgi:bundle-services	257
osgi:classes	258
osgi:find-class	259
osgi:headers	260
osgi:info	261
osgi:install	262
osgi:list	263
osgi:ls	264
osgi:refresh	265
osgi:resolve	266
osgi:restart	267
osgi:shutdown	268
osgi:start	269
osgi:start-level	270
osgi:stop	271
osgi:uninstall	272
osgi:update	273

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.

Name

`osgi:bundle-level`, `bundle-level` — gets or sets the start level of a given bundle

Synopsis

```
osgi:bundle-level [--help] [--force] {id} [startLevel]
```

Arguments

[Table 17.1 on page 256](#) describes the command's arguments.

Table 17.1. *osgi:bundle-level Arguments*

Argument	Interpretation
<code>--help</code>	Displays the online help for this command.
<code>--force</code>	Forces the command to execute.
<i>id</i>	Specifies the id for the bundle.
<i>startLevel</i>	Specifies the new start level for the bundle.

Name

osgi:bundle-services, bundle-services — lists the OSGi services provided by a bundle

Synopsis

osgi:bundle-level [-u] [-p] [-a] [--help] [--force] {id}

Arguments

[Table 17.2 on page 257](#) describes the command's arguments.

Table 17.2. 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.

Name

`osgi:classes`, `classes` — lists all of the classes in the specified bundle or bundles

Synopsis

```
osgi:classes [--help] [--force] [[-a] | [--display-all-files]] {ids}
```

Arguments

[Table 17.3 on page 258](#) describes the command's arguments.

Table 17.3. *osgi:classes* Arguments

Argument	Interpretation
<code>--help</code>	Displays the online help for this command.
<code>--force</code>	Forces the command to execute.
<code>-a</code> , <code>--display-all-files</code>	Also lists the files contained in the bundles.
<code>ids</code>	Space-separated list of bundle IDs.

Name

`osgi:find-class`, `find-class` — locates a specified class in any deployed bundle

Synopsis

```
osgi:find-class [--help] {className}
```

Arguments

[Table 17.4 on page 259](#) describes the command's arguments.

Table 17.4. *osgi:find-class* Arguments

Argument	Interpretation
<code>--help</code>	Displays the online help for this command.
<i>className</i>	Class name or partial class name to find.

Name

`osgi:headers`, `headers` — displays the headers of a specified OSGi bundle

Synopsis

`osgi:headers [--help] {id...}`

Arguments

[Table 17.5 on page 260](#) describes the command's arguments.

Table 17.5. *osgi:headers* Arguments

Argument	Interpretation
<code>--help</code>	Displays the online help for this command
<code>id</code>	Specifies a space delimited list of bundle IDs.

Name

`osgi:info` — displays detailed information about OSGi bundles

Synopsis

```
osgi:info [--help] {id...}
```

Arguments

[Table 17.6 on page 261](#) describes the command's arguments.

Table 17.6. *osgi:info* Arguments

Argument	Interpretation
<code>--help</code>	Displays the online help for this command
<code>id</code>	Specifies a space delimited list of bundle IDs.

Name

`osgi:install`, `install` — installs one or more OSGi bundles

Synopsis

```
osgi:install [--help] [--s] | [--start]] {url...}
```

Arguments

[Table 17.7 on page 262](#) describes the command's arguments.

Table 17.7. *osgi:install Arguments*

Argument	Interpretation
<code>--help</code>	Displays the online help for this command
<code>-s</code> , <code>--start</code>	Starts the bundles after installation
<code>url</code>	Specifies a space delimited list of bundle URLs.

Name

`osgi:list`, `list` — lists the installed bundles whose start level equals or exceeds the specified threshold

Synopsis

```
osgi:list [--help] [-u] [-t threshold] [-l] [-s]
```

Arguments

[Table 17.8 on page 263](#) describes the command's arguments.

Table 17.8. *osgi:list* Arguments

Argument	Interpretation
<code>--help</code>	Displays the online help for this command
<code>-u</code>	Shows the update locations
<code>-t</code>	Specifies the start level threshold. The default is the value of the <code>karaf.systemBundlesStartLevel</code> property whose default value is 50.
<code>-l</code>	Shows the locations of the bundles
<code>-s</code>	Shows the symbolic names of the bundles

Name

`osgi:ls, ls` — lists OSGi services

Synopsis

`osgi:ls [--help] [-a] [-u] [--force] [id...]`

Arguments

[Table 17.9 on page 264](#) describes the command's arguments.

Table 17.9. *osgi:ls* Arguments

Argument	Interpretation
<code>--help</code>	Displays the online help for this command
<code>-a</code>	Lists all services
<code>-u</code>	Lists the services in use
<code>--force</code>	Forces the command to execute
<i>id</i>	Specifies a space separated list of bundle IDs.

Name

osgi:refresh, refresh — refreshes an OSGi bundle

Synopsis

```
osgi:refresh [--help] [--force] {id...}
```

Arguments

[Table 17.10 on page 265](#) describes the command's arguments.

Table 17.10. *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.

Name

`osgi:resolve`, `resolve` — resolves an OSGi bundle's dependencies

Synopsis

```
osgi:resolve [--help] [--force] {id...}
```

Arguments

[Table 17.11 on page 266](#) describes the command's arguments.

Table 17.11. *osgi:resolve Arguments*

Argument	Interpretation
<code>--help</code>	Displays the online help for this command.
<code>--force</code>	Forces the command to execute.
<code>id</code>	Specifies a space delimited list of bundle IDs.

Name

`osgi:restart`, `restart` — stops and restarts an OSGi bundle

Synopsis

```
osgi:restart [--help] [--force] {id...}
```

Arguments

[Table 17.12 on page 267](#) describes the command's arguments.

Table 17.12. *osgi:restart* Arguments

Argument	Interpretation
<code>--help</code>	Displays the online help for this command.
<code>--force</code>	Forces the command to execute.
<code>id</code>	Specifies a space delimited list of bundle IDs.

Name

osgi:shutdown, shutdown — stops the OSGi framework

Synopsis

osgi:shutdown [--help] [[-f] | [--force]] [[hh:mm] | [+m]]

Arguments

[Table 17.13 on page 268](#) describes the command's arguments.

Table 17.13. *osgi:shutdown Arguments*

Argument	Interpretation
--help	Displays the online help for this command.
-f, --force	Forces the command to execute.
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.

Name

`osgi:start`, `start` — starts an OSGi bundle

Synopsis

```
osgi:start [--help] [--force] {id...}
```

Arguments

[Table 17.14 on page 269](#) describes the command's arguments.

Table 17.14. *osgi:start* Arguments

Argument	Interpretation
<code>--help</code>	Displays the online help for this command.
<code>--force</code>	Forces the command to execute.
<code>id</code>	Specifies a space delimited list of bundle IDs.

Name

`osgi:start-level`, `start-level` — gets or sets the OSGi framework's active start level

Synopsis

```
osgi:start [--help] [level]
```

Arguments

[Table 17.15 on page 270](#) describes the command's arguments.

Table 17.15. *osgi:start-level* Arguments

Argument	Interpretation
<code>--help</code>	Displays the online help for this command.
<code><i>level</i></code>	Specifies the new start level to set.

Name

`osgi:stop`, `stop` — stops an OSGi bundle

Synopsis

`osgi:stop [--help] [--force] {id...}`

Arguments

[Table 17.16 on page 271](#) describes the command's arguments.

Table 17.16. *osgi:stop Arguments*

Argument	Interpretation
<code>--help</code>	Displays the online help for this command.
<code>--force</code>	Forces the command to execute.
<code>id</code>	Specifies a space delimited list of bundle IDs.

Name

`osgi:uninstall`, `uninstall` — uninstalls an OSGi bundle

Synopsis

```
osgi:uninstall [--help] [--force] {id...}
```

Arguments

[Table 17.17 on page 272](#) describes the command's arguments.

Table 17.17. *osgi:uninstall* Arguments

Argument	Interpretation
<code>--help</code>	Displays the online help for this command.
<code>--force</code>	Forces the command to execute.
<code>id</code>	Specifies a space delimited list of bundle IDs.

Name

osgi:update, update — updates an OSGi bundle

Synopsis

```
osgi:update [--help] [--force] {id} [location]
```

Arguments

[Table 17.18 on page 273](#) describes the command's arguments.

Table 17.18. *osgi:update* Arguments

Argument	Interpretation
<code>--help</code>	Displays the online help for this command.
<code>--force</code>	Forces the command to execute.
<i>id</i>	Specifies ID of the bundle.
<i>location</i>	Specifies the location from which the update is loaded. If no location is specified the container will use either the bundle's <code>Bundle-UpdateLocation</code> property or the bundle's original location.

Chapter 18. Packages Console Commands

packages:exports	276
packages:imports	277

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.

Name

packages:exports, exports — displays the packages exported OSGi bundles

Synopsis

```
packages:export [--help] [--d] | [--details] [-s] [--i] | [--imports] [id...]
```

Arguments

[Table 18.1 on page 276](#) describes the commands arguments.

Table 18.1. *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 <code>Symbolic name</code> column, which shows the symbolic name of the bundle to which the exported package belongs.
-i, --imports	Show the <code>Imported by</code> column, which lists all of the bundles that import the exported package.
id	Specifies a whitespace separated list of bundle IDs to check.

Name

packages:imports, imports — displays the packages imported by OSGi bundles

Synopsis

```
packages:imports [--help] [[-i] | [--show-importer]] [id...]
```

Arguments

[Table 18.2 on page 277](#) describes the commands arguments.

Table 18.2. package:imports Arguments

Argument	Interpretation
--help	Displays the online help for this command
-i, --show-importer	Show the bundle(s) that import a package.
<i>id</i>	Specifies a whitespace separated list of bundle IDs to check.

Chapter 19. Patch Console Commands

patch:add	280
patch:install	281
patch:list	282
patch:rollback	283
patch:simulate	284

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.

Name

patch:add, download — 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 19.1 on page 280](#) describes the command's arguments.

Table 19.1. *patch:add Arguments*

Argument	Interpretation
<code>--help</code>	Displays the online help for this command.
<code>--bundles</code>	List the bundles included in the patch.
<code>URL</code>	Specifies the URL from which the patch is downloaded.

Name

patch:install — installs a patch that was previously downloaded

Synopsis

```
patch:install [--help] {patch}
```

Arguments

[Table 19.2 on page 281](#) describes the command's arguments.

Table 19.2. *patch:install* Arguments

Argument	Interpretation
--help	Displays the online help for this command.
<i>patch</i>	Specifies the name of the patch to install.

Name

patch:list — lists all known patches, showing the patch name and status (installed or not)

Synopsis

patch:list [--help] [--bundles]

Arguments

[Table 19.3 on page 282](#) describes the command's arguments.

Table 19.3. patch:list Arguments

Argument	Interpretation
--help	Displays the online help for this command.
--bundles	List the bundles for each patch.

Name

patch:rollback — reverses a patch installation

Synopsis

patch:rollback [--help] {*patch*}

Arguments

[Table 19.4 on page 283](#) describes the command's arguments.

Table 19.4. patch:rollback Arguments

Argument	Interpretation
--help	Displays the online help for this command.
<i>patch</i>	Specifies the name of the patch to roll back.

Name

patch:simulate, simulate — 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 19.5 on page 284](#) describes the command's arguments.

Table 19.5. patch:simulate Arguments

Argument	Interpretation
--help	Displays the online help for this command.
<i>patch</i>	Specifies the name of the patch to simulate installing.

Chapter 20. SSH Console Commands

ssh:ssh	286
ssh:sshd	287

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.

Name

ssh:ssh, ssh — 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 20.1 on page 286](#) describes the commands arguments.

Table 20.1. *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
<i>hostname</i>	The hostname to connect to via SSH
<i>command</i>	Specifies a command to execute upon connecting.

Name

ssh:sshd, sshd — creates an SSH server

Synopsis

ssh:sshd [--help] [[-b] | [--background]] [[-p *port*] | [--port *port*]]

Arguments

[Table 20.2 on page 287](#) describes the commands arguments.

Table 20.2. *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 21. Web Console Commands

web:list	290
----------------	-----

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.

Name

`web:list` — lists the WARs deployed in the container

Synopsis

`web:list [--help]`

Arguments

[Table 21.1 on page 290](#) describes the command's arguments.

Table 21.1. `web:list` Arguments

Argument	Interpretation
<code>--help</code>	Displays the online help for this command

Chapter 22. The `wrapper:install` Command

`wrapper:install` 292

The **`wrapper:install`** command installs Fuse ESB Enterprise as a system service.



Note

This feature is not installed by default. To install the `wrapper` shell, run the following command:

```
karaf@root:> features:install wrapper
```

Name

`wrapper:install` — 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 22.1. *wrapper:install Arguments*

Argument	Interpretation
<code>--help</code>	Displays the online help for this command
<code>-s, --start-type</code>	The mode in which the service is installed, either <code>AUTO_START</code> or <code>DEMAND_START</code> ; the default is <code>AUTO_START</code>
<code>-n, --name</code>	The service name used when installing the service; the default is <code>karaf</code>
<code>-d, --display</code>	The display name of the service
<code>-D, --description</code>	The description of the service

Chapter 23. ZooKeeper Console Commands

zk:create	294
zk:delete	297
zk:get	298
zk:list	299
zk:set	300

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
```

Name

zk:create — 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:rwcd  
/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 ["Fabric Console Commands" on page 125](#).



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:rwcd` ACL setting).

Arguments

[Table 23.1 on page 295](#) describes the commands arguments.

Table 23.1. `zk:create` Arguments

Argument	Interpretation
<code>--help</code>	Displays the online help for this command

Argument	Interpretation
<code>-r, --recursive</code>	Automatically create any missing parent nodes in the specified path.
<code>-i, --import</code>	Interpret the data argument as a URL that locates a resource containing the initial data for the new znode.
<code>-e, --ephemeral</code>	Make the new znode ephemeral, so that it is automatically deleted after the current ZooKeeper client session closes.
<code>-s, --sequential</code>	Make the new znode sequential, which implies that a unique 10-digit suffix is appended to the znode name.
<code>-a, --acl</code>	Specifies the znode's ACL as a comma-separated list, where each entry in the list has the format, <i>Scheme:ID:Permissions</i> . The <i>Permissions</i> string consists of the following characters, concatenated in any order: <i>r</i> (read), <i>w</i> (write), <i>c</i> (create), <i>d</i> (delete), and <i>a</i> (admin).
<code>-o, --overwrite</code>	Overwrite the existing znode at this location, if there is one.
<i>path</i>	<i>(Required)</i> Path of the znode to create.
<i>data</i>	Initial data for the node or, if <code>--import</code> is specified, a URL pointing at a location that contains the initial data.

Name

zk:delete — delete the specified znode

Synopsis

zk:delete [--help] [-v|--version *version*] [-r|--recursive] {*path*}

Arguments

[Table 23.2 on page 297](#) describes the commands arguments.

Table 23.2. 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).
-r, --recursive	Recursively delete children. Defaults to <i>false</i> .
<i>path</i>	Path of the znode to delete.

Name

zk:get — get a znode's data

Synopsis

zk:get [--help] {*path*}

Arguments

[Table 23.3 on page 298](#) describes the commands arguments.

Table 23.3. zk:get Arguments

Argument	Interpretation
--help	Displays the online help for this command
<i>path</i>	(Required) Path of the znode to get.

Name

zk:list — list a znode's children

Synopsis

zk:list [--help] [-r|--recursive] [-d|--display] {path}

Arguments

[Table 23.4 on page 299](#) describes the commands arguments.

Table 23.4. 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 /.

Name

zk:set — 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 ["Fabric Console Commands" on page 125](#).

Arguments

[Table 23.5 on page 300](#) describes the commands arguments.

Table 23.5. 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.

Appendix A. Command Aliases

The command console shell uses a number of short cuts, or aliases for common commands. [Table A.1 on page 301](#) lists the command aliases available in the command console.

Table A.1. Console Command Aliases

Alias	Command
ld	log:display on page 237
lde	log:display-exception on page 238
la	osgi:list on page 263 -t 0
cl	config:list on page 92 "(service.pid=\$args) "
man	help

