cloudify Documentation

Release 3.3

Gigaspaces

December 14, 2015

Contents

The documention here includes reference for the various `cfy` subcommands. For general usage, please refer to the documentation located at getcloudify.org.

Contents:

CHAPTER 1

Commands

There are two flags that can be used for all operations:

- `--verbose` prints the traceback and prints the events in verbose mode (a full event json)
- `--debug` sets all loggers declared in the config file to debug mode.

  In particular, sets the rest client logger to debug mode, this means that the output will include http communication with the rest server (response, requests and headers).

Inputs and Parameters

All commands that accept inputs or paramaters (e.g. "cfy execute" or "cfy deployments create") expect the value to represent a dictionary. Valid formats are:

- A path to the YAML file
- A string formatted as YAML
- A string formatted as "key1=value1;key2=value2"

1.1 cfy

Manages Cloudify in different Cloud Environments

```
usage: cfy [-h] [--version]
           {status,blueprints,bootstrap,teardown,workflows,recover,node-instances,snapshots
           ...
```

Options:

--version show version information and exit

Sub-commands:

status Show a management server's status

The command prints out the currently active manager's IP address and a status of the active manager's running services.

```
usage: cfy status [-h] [-v] [--debug]
```

Options:

-v=False, --verbose=False Set verbose output

--debug=False Set debug output

blueprints Manages Cloudify's Blueprints

```
usage: cfy blueprints [-h]
                      {inputs,publish-archive,download,validate,get,list,upload,del
                      ...
```

Sub-commands:

inputs command for listing all available blueprint inputs

```
usage: cfy blueprints inputs [-h] -b BLUEPRINT_ID [-v] [--debug]
```

Options:

-b, --blueprint-id The id of the blueprint

-v=False, --verbose=False Set verbose output

--debug=False Set debug output

publish-archive command for publishing a blueprint archive from a path or URL to the management server

```
usage: cfy blueprints publish-archive [-h] -l ARCHIVE_LOCATION -b BLUEPRINT_I
                                      [-n BLUEPRINT_FILENAME] [-v] [--debug]
```

Options:

-l, --archive-location Path or URL to the application's blueprint archive file

-b, --blueprint-id The id of the blueprint

-n, --blueprint-filename Name of the archive's main blueprint file

-v=False, --verbose=False Set verbose output

--debug=False Set debug output

download command for downloading a blueprint from the management server

```
usage: cfy blueprints download [-h] [-o OUTPUT] -b BLUEPRINT_ID [-v] [--debug
```

Options:

-o, --output The output file path of the blueprint to be downloaded

-b, --blueprint-id The id of the blueprint

-v=False, --verbose=False Set verbose output

--debug=False Set debug output

validate command for validating a blueprint

```
usage: cfy blueprints validate [-h] -p BLUEPRINT_FILE [-v] [--debug]
```

Options:

-p, --blueprint-path Path to the application's blueprint file

-v=False, --verbose=False Set verbose output

--debug=False Set debug output

get command for getting a blueprint by its id

```
usage: cfy blueprints get [-h] -b BLUEPRINT_ID [-v] [--debug]
```

Options:

-b, --blueprint-id The id of the blueprint

-v=False, --verbose=False Set verbose output

--debug=False Set debug output

list command for listing all blueprints on the Manager

```
usage: cfy blueprints list [-h] [-v] [--debug]
```

Options:

-v=False, --verbose=False Set verbose output

--debug=False Set debug output

upload command for uploading a blueprint to the management server

```
usage: cfy blueprints upload [-h] -p BLUEPRINT_FILE -b BLUEPRINT_ID [-v]
                             [--debug]
```

Options:

-p, --blueprint-path Path to the application's blueprint file

-b, --blueprint-id The id of the blueprint

-v=False, --verbose=False Set verbose output

--debug=False Set debug output

delete command for deleting a blueprint

```
usage: cfy blueprints delete [-h] -b BLUEPRINT_ID [-v] [--debug]
```

Options:

-b, --blueprint-id The id of the blueprint

-v=False, --verbose=False Set verbose output

--debug=False Set debug output

bootstrap Bootstrap a Cloudify management environment

The command takes care of creating the topology and installing the Cloudify Manager to function.

Note: The command also allows you to run validations without actually bootstrapping to verify that the resources required are available for the bootstrap process.

```
usage: cfy bootstrap [-h] [--task-thread-pool-size TASK_THREAD_POOL_SIZE]
                     [--install-plugins] [--keep-up-on-failure]
                     [--validate-only] [--skip-validations] -p BLUEPRINT_PATH
                     [-i INPUTS] [--task-retries TASK_RETRIES]
                     [--task-retry-interval TASK_RETRY_INTERVAL] [-v]
                     [--debug]
```

Options:

--task-thread-pool-size=1 The size of the thread pool size to execute tasks in

--install-plugins=False Install necessary plugins of the given blueprint.

--keep-up-on-failure=False If the bootstrap fails, the management server will remain running

--validate-only=False Run validations without actually performing the bootstrap process.

--skip-validations=False Run bootstrap without, validating resources prior to bootstrapping the manager

-p, --blueprint-path Path to a manager blueprint

-i, --inputs Inputs file/string for a manager blueprint (formatted as YAML or as "key1=value1;key2=value2")

--task-retries=5 How many times should a task be retried in case it fails

--task-retry-interval=30 How many seconds to wait before each task is retried

-v=False, --verbose=False Set verbose output

--debug=False Set debug output

teardown Teardown Cloudify

```
usage: cfy teardown [-h] [-f] [--ignore-deployments] [-v] [--debug]
```

Options:

-f=False, --force=False Confirmation for the teardown request

--ignore-deployments=False Perform teardown even if deploymentsexist on the manager

-v=False, --verbose=False Set verbose output

--debug=False Set debug output

workflows Manages Deployment Workflows

```
usage: cfy workflows [-h] {list,get} ...
```

Sub-commands:

list command for listing workflows for a deployment

```
usage: cfy workflows list [-h] [-d DEPLOYMENT_ID] [-v] [--debug]
```

Options:

-d, --deployment-id The id of the deployment whose workflows to list

-v=False, --verbose=False Set verbose output

--debug=False Set debug output

get command for getting a workflow by its name and deployment

```
usage: cfy workflows get [-h] [-d DEPLOYMENT_ID] -w WORKFLOW [-v] [--debug]
```

Options:

-d, --deployment-id The id of the deployment for which the workflow belongs

-w, --workflow The id of the workflow to get

-v=False, --verbose=False Set verbose output

--debug=False Set debug output

recover Performs recovery of the management machine and all its contained nodes.

```
usage: cfy recover [-h] [-s SNAPSHOT_PATH] [--task-retries TASK_RETRIES] [-f]
                   [--task-thread-pool-size TASK_THREAD_POOL_SIZE]
                   [--task-retry-interval TASK_RETRY_INTERVAL] [-v] [--debug]
```

Options:

-s, --snapshot-path Path to the snapshot that will be restored

--task-retries=5 How many times should a task be retried in case it fails.

-f=False, --force=False Confirmation for the recovery request

--task-thread-pool-size=1 The size of the thread pool size to execute tasks in

--task-retry-interval=30 How many seconds to wait before each task is retried.

-v=False, --verbose=False Set verbose output

--debug=False Set debug output

node-instances Manage node instances

```
usage: cfy node-instances [-h] {list,get} ...
```

Sub-commands:

list Command for getting node instances

```
usage: cfy node-instances list [-h] [--node-name NODE_NAME] [-d DEPLOYMENT_ID
                               [-v] [--debug]
```

Options:

--node-name Filter node instances according to the node name

-d, --deployment-id Filter node instances for a given deployment according to the deployment ID

-v=False, --verbose=False Set verbose output

--debug=False Set debug output

get Command for getting a node instance according to it's ID

```
usage: cfy node-instances get [-h] --node-instance-id NODE_INSTANCE_ID [-v]
                              [--debug]
```

Options:

--node-instance-id The ID of the node instance to get

-v=False, --verbose=False Set verbose output

--debug=False Set debug output

snapshots Manages Cloudify's Snapshots

```
usage: cfy snapshots [-h] {restore,create,list,upload,download,delete} ...
```

Sub-commands:

restore Restore manager state to a specific snapshot

```
usage: cfy snapshots restore [-h] -s SNAPSHOT_ID [-f]
                             [--without-deployments-envs] [-v] [--debug]
```

Options:

-s, --snapshot-id The id of the snapshot

-f=False, --force=False Force restoring the snapshot on a dirty manager

--without-deployments-envs=False Restore snapshot without deployment environments

-v=False, --verbose=False Set verbose output

--debug=False Set debug output

create Create a new snapshot

```
usage: cfy snapshots create [-h] -s SNAPSHOT_ID [--exclude-credentials]
                            [--include-metrics] [-v] [--debug]
```

Options:

-s, --snapshot-id A unique id that will be assigned to the created snapshot

--exclude-credentials=False Do not store credentials in snapshot

--include-metrics=False Include metrics datain the snapshot

-v=False, --verbose=False Set verbose output

--debug=False Set debug output

list List all snapshots on the manager

```
usage: cfy snapshots list [-h] [-v] [--debug]
```

Options:

-v=False, --verbose=False Set verbose output

--debug=False Set debug output

upload Upload a snapshot to the management server

```
usage: cfy snapshots upload [-h] -s SNAPSHOT_ID -p SNAPSHOT_FILE [-v]
                            [--debug]
```

Options:

-s, --snapshot-id The id of the snapshot

-p, --snapshot-path Path to the manager's snapshot file

-v=False, --verbose=False Set verbose output

--debug=False Set debug output

download Download a snapshot from the management server

```
usage: cfy snapshots download [-h] -s SNAPSHOT_ID [-o OUTPUT] [-v] [--debug]
```

Options:

-s, --snapshot-id The id of the snapshot

-o, --output The output file path of the snapshot to be downloaded

-v=False, --verbose=False Set verbose output

--debug=False Set debug output

delete Delete a snapshot from the manager

```
usage: cfy snapshots delete [-h] -s SNAPSHOT_ID [-v] [--debug]
```

Options:

-s, --snapshot-id The id of the snapshot

-v=False, --verbose=False Set verbose output

--debug=False Set debug output

deployments Manages and Executes Cloudify's Deployments

```
usage: cfy deployments [-h] {outputs,create,list,delete} ...
```

Sub-commands:

outputs command for getting a specific deployment outputs

```
usage: cfy deployments outputs [-h] [-d DEPLOYMENT_ID] [-v] [--debug]
```

Options:

-d, --deployment-id The id of the deployment to get outputs for

-v=False, --verbose=False Set verbose output

--debug=False Set debug output

create command for creating a deployment of a blueprint

```
usage: cfy deployments create [-h] [-d DEPLOYMENT_ID] [-i INPUTS] -b
                              BLUEPRINT_ID [-v] [--debug]
```

Options:

-d, --deployment-id A unique id that will be assigned to the created deployment

-i, --inputs Inputs file/string for the deployment creation (formatted as YAML or as "key1=value1;key2=value2")

-b, --blueprint-id The id of the blueprint

-v=False, --verbose=False Set verbose output

--debug=False Set debug output

list command for listing all deployments or all deployments of a blueprint

```
usage: cfy deployments list [-h] [-b BLUEPRINT_ID] [-v] [--debug]
```

Options:

-b, --blueprint-id The id of the blueprint

-v=False, --verbose=False Set verbose output

--debug=False Set debug output

delete command for deleting a deployment

```
usage: cfy deployments delete [-h] [-f] [-d DEPLOYMENT_ID] [-v] [--debug]
```

Options:

-f=False, --ignore-live-nodes=False Delete the deployment even if there are existing live nodes for it

-d, --deployment-id the id of the deployment to delete

-v=False, --verbose=False Set verbose output

--debug=False Set debug output

init Initialize cfy work environment

Initializes CLI configuration files and work environment in the current working directory.

```
usage: cfy init [-h] [-r] [-v] [--debug]
```

Options:

-r=False, --reset-config=False Overwriting existing configuration is allowed

-v=False, --verbose=False Set verbose output

--debug=False Set debug output

agents Manages Cloudify's Agents

```
usage: cfy agents [-h] {install} ...
```

Sub-commands:

install command for installing agents on deployments

```
usage: cfy agents install [-h] [-d DEPLOYMENT_ID] [-l] [-v] [--debug]
```

Options:

-d, --deployment-id The id of the deployment to install agents for. If ommited, this will install agents for all deployments

-l=False, --include-logs=False Include logs in returned events

-v=False, --verbose=False Set verbose output

--debug=False Set debug output

dev Executes fabric tasks on the management machine

Cloudify's CLI provides an interface to running premade fabric tasks on the management server.

This supplies an easy way to run personalized, complex ssh scripts on the manager without having to manually connect to it.

Note: The tasks don't have to be decorated with the `@task` decorator as they're directly called from the cli's code just like any other python function. Also, as fabric is one of the cli's dependencies, you don't have to install it separately unless you're using the cli as a binary in which case you'll have to install fabric yourself.

Example:

```
cfy dev --tasks-file my_tasks.py -v -t my_task -a --arg1=something --arg2=otherthing ...
cfy dev -v -t my_task -a arg1_value arg2_value ...
```

`--tasks-file my_tasks.py` can be omitted if a `tasks.py` file exists in your current working directory.

So for instance, if you want to echo `something` in your currently running manager, all you have to do is supply a `tasks.py` file with the following:

```
from fabric.api import run

def echo(text):
    run('echo {0}'.format(text))
```

and then run:

```
cfy dev -t echo -a something
```

Cloudify provides a tasks repo from which users can obtain tasks and to which developers should contribute for the benefit of all.

```
usage: cfy dev [-h] [-t TASK] [-p TASKS_FILE] [-a ...] [-v] [--debug]
```

Options:

-t, --task	name of fabric task to run
-p, --tasks-file	Path to a tasks file
-a, --args	arguments for the fabric task
-v=False, --verbose=False	Set verbose output
--debug=False	Set debug output

use Use/switch to the specified management server

```
usage: cfy use [-h] [--port REST_PORT] -t MANAGEMENT_IP [-v] [--debug]
```

Options:

--port=80	Specify the rest server port
-t, --management-ip	The cloudify management server ip address
-v=False, --verbose=False	Set verbose output
--debug=False	Set debug output

plugins Manages Cloudify's plugins

```
usage: cfy plugins [-h] {download,delete,list,upload,get} ...
```

Sub-commands:

download Command for downloading a plugin from the management server

```
usage: cfy plugins download [-h] [-o OUTPUT] -p PLUGIN_ID [-v] [--debug]
```

Options:

-o, --output	The output file path of the plugin to be downloaded
-p, --plugin-id	The plugin id
-v=False, --verbose=False	Set verbose output
--debug=False	Set debug output

delete Command for deleting a plugin

```
usage: cfy plugins delete [-h] -p PLUGIN_ID [-v] [--debug]
```

Options:

-p, --plugin-id	The plugin id
-v=False, --verbose=False	Set verbose output
--debug=False	Set debug output

list Command for listing all plugins on the Manager

```
usage: cfy plugins list [-h] [-v] [--debug]
```

Options:

-v=False, --verbose=False Set verbose output

--debug=False Set debug output

upload command for uploading a plugin to the management server

```
usage: cfy plugins upload [-h] -p PLUGIN_FILE [-v] [--debug]
```

Options:

-p, --plugin-path Path to the plugin file

-v=False, --verbose=False Set verbose output

--debug=False Set debug output

get Command for listing all modules according to their plugin id

```
usage: cfy plugins get [-h] -p PLUGIN_ID [-v] [--debug]
```

Options:

-p, --plugin-id The plugin id

-v=False, --verbose=False Set verbose output

--debug=False Set debug output

nodes Manage nodes

```
usage: cfy nodes [-h] {list,get} ...
```

Sub-commands:

list Command for getting all nodes

```
usage: cfy nodes list [-h] [-d DEPLOYMENT_ID] [-v] [--debug]
```

Options:

-d, --deployment-id Filter nodes for a given deployment according to the deployment ID

-v=False, --verbose=False Set verbose output

--debug=False Set debug output

get command for getting a node by its ID

```
usage: cfy nodes get [-h] --node-id NODE_ID -d DEPLOYMENT_ID [-v] [--debug]
```

Options:

--node-id The ID of the node to get

-d, --deployment-id Filter nodes for a given deployment according to the deployment ID

-v=False, --verbose=False Set verbose output

--debug=False Set debug output

executions Manages Cloudify's Executions

```
usage: cfy executions [-h] {cancel,start,list,get} ...
```

Sub-commands:

cancel Cancel an execution by its id

```
usage: cfy executions cancel [-h] -e EXECUTION_ID [-f] [-v] [--debug]
```

Options:

-e, --execution-id The id of the execution to cancel

-f=False, --force=False Terminate the execution abruptly, rather than request an orderly termination

-v=False, --verbose=False Set verbose output

--debug=False Set debug output

start Command for starting a workflow execution on a deployment

```
usage: cfy executions start [-h] [--allow-custom-parameters]
                            [-d DEPLOYMENT_ID] [-p PARAMETERS] [-f]
                            [--timeout TIMEOUT] -w WORKFLOW [-l] [-v]
                            [--debug]
```

Options:

--allow-custom-parameters=False Allow the passing of custom parameters (parameters which were not defined in the workflow's schema in the blueprint) to the execution

-d, --deployment-id The deployment id

-p={}, --parameters={} Parameters for the workflow execution (formatted as YAML or as "key1=value1;key2=value2")

-f=False, --force=False Whether the workflow should execute even if there is an ongoing execution for the provided deployment

--timeout=900 Operation timeout in seconds (The execution itself will keep going, it is the CLI that will stop waiting for it to terminate)

-w, --workflow The workflow to start

-l=False, --include-logs=False Include logs in returned events

-v=False, --verbose=False Set verbose output

--debug=False Set debug output

list command for listing all executions of a deployment

```
usage: cfy executions list [-h] [-d DEPLOYMENT_ID] [--system-workflows] [-v]
                           [--debug]
```

Options:

-d, --deployment-id filter executions for a given deployment by the deployment's id

--system-workflows=False Include executions of system workflows.

-v=False, --verbose=False Set verbose output

--debug=False Set debug output

get command for getting an execution by its id

```
usage: cfy executions get [-h] -e EXECUTION_ID [-v] [--debug]
```

Options:

-e, --execution-id The id of the execution to get

-v=False, --verbose=False Set verbose output

--debug=False Set debug output

local Execute workflows locally

```
usage: cfy local [-h]
                 {execute,install-plugins,instances,init,create-requirements,output
                 ...
```

Sub-commands:

execute Execute a workflow locally

```
usage: cfy local execute [-h] [--allow-custom-parameters] [-p PARAMETERS]
                         [--task-retries TASK_RETRIES] -w WORKFLOW
                         [--task-thread-pool-size TASK_THREAD_POOL_SIZE]
                         [--task-retry-interval TASK_RETRY_INTERVAL] [-v]
                         [--debug]
```

Options:

--allow-custom-parameters=False Allow the passing of custom parameters (parameters which were not defined in the workflow's schema in the blueprint) to the execution

-p={}, --parameters={} Parameters for the workflow execution (formatted as YAML or as "key1=value1;key2=value2")

--task-retries=0 How many times should a task be retried in case it fails

-w, --workflow The workflow to execute locally

--task-thread-pool-size=1 The size of the thread pool size to execute tasks in

--task-retry-interval=1 How many seconds to wait before each task is retried

-v=False, --verbose=False Set verbose output

--debug=False Set debug output

install-plugins Installs the necessary plugins for a given blueprint

```
usage: cfy local install-plugins [-h] -p BLUEPRINT_PATH [-v] [--debug]
```

Options:

-p, --blueprint-path Path to a blueprint

-v=False, --verbose=False Set verbose output

--debug=False Set debug output

instances Display node instances

```
usage: cfy local instances [-h] [--node-id NODE_ID] [-v] [--debug]
```

Options:

--node-id Only display node instances of this node id

-v=False, --verbose=False Set verbose output

--debug=False Set debug output

init Init a local workflow execution environment in in the current working directory

Initializes CLI configuration files and work environment in the current working directory.

```
usage: cfy local init [-h] [--install-plugins] -p BLUEPRINT_PATH [-i INPUTS]
                      [-v] [--debug]
```

Options:

--install-plugins=False Install necessary plugins of the given blueprint.

-p, --blueprint-path Path to a blueprint

-i, --inputs Inputs file/string for the local workflow creation (formatted as YAML or as "key1=value1;key2=value2")

-v=False, --verbose=False Set verbose output

--debug=False Set debug output

create-requirements Creates a PIP compliant requirements file for the given blueprint

```
usage: cfy local create-requirements [-h] [-o REQUIREMENTS_OUTPUT] -p
                                     BLUEPRINT_PATH [-v] [--debug]
```

Options:

-o, --output Path to a file that will hold the requirements of the blueprint

-p, --blueprint-path Path to a blueprint

-v=False, --verbose=False Set verbose output

--debug=False Set debug output

outputs Display outputs

```
usage: cfy local outputs [-h] [-v] [--debug]
```

Options:

-v=False, --verbose=False Set verbose output

--debug=False Set debug output

events Manages Cloudify's events

```
usage: cfy events [-h] {list} ...
```

Sub-commands:

list Displays Events for different executions

```
usage: cfy events list [-h] -e EXECUTION_ID [-l] [--tail] [-v] [--debug]
```

Options:

-e, --execution-id The id of the execution to list events for

-l=False, --include-logs=False Includes logs in the returned events

--tail=False tail the events of the specified execution until it ends

-v=False, --verbose=False Set verbose output

--debug=False Set debug output

ssh SSH to management server

```
usage: cfy ssh [-h] [-c COMMAND] [-p] [-v] [--debug]
```

Options:

-c, --command Execute command over SSH

-p=False, --plain=False Leave authentication to user

-v=False, --verbose=False Set verbose output

--debug=False Set debug output

1.2 blueprints

```
usage: cfy blueprints [-h]
                      {inputs,publish-archive,download,validate,get,list,upload,delete}
                      ...
```

Sub-commands:

inputs command for listing all available blueprint inputs

```
usage: cfy blueprints inputs [-h] -b BLUEPRINT_ID [-v] [--debug]
```

Options:

-b, --blueprint-id The id of the blueprint

-v=False, --verbose=False Set verbose output

--debug=False Set debug output

publish-archive command for publishing a blueprint archive from a path or URL to the management server

```
usage: cfy blueprints publish-archive [-h] -l ARCHIVE_LOCATION -b BLUEPRINT_ID
                                      [-n BLUEPRINT_FILENAME] [-v] [--debug]
```

Options:

-l, --archive-location Path or URL to the application's blueprint archive file

-b, --blueprint-id The id of the blueprint

-n, --blueprint-filename Name of the archive's main blueprint file

-v=False, --verbose=False Set verbose output

--debug=False Set debug output

download command for downloading a blueprint from the management server

```
usage: cfy blueprints download [-h] [-o OUTPUT] -b BLUEPRINT_ID [-v] [--debug]
```

Options:

-o, --output The output file path of the blueprint to be downloaded

-b, --blueprint-id The id of the blueprint

-v=False, --verbose=False Set verbose output

--debug=False Set debug output

validate command for validating a blueprint

```
usage: cfy blueprints validate [-h] -p BLUEPRINT_FILE [-v] [--debug]
```

Options:

-p, --blueprint-path Path to the application's blueprint file

-v=False, --verbose=False Set verbose output

--debug=False Set debug output

get command for getting a blueprint by its id

```
usage: cfy blueprints get [-h] -b BLUEPRINT_ID [-v] [--debug]
```

Options:

-b, --blueprint-id The id of the blueprint

-v=False, --verbose=False Set verbose output

--debug=False Set debug output

list command for listing all blueprints on the Manager

```
usage: cfy blueprints list [-h] [-v] [--debug]
```

Options:

-v=False, --verbose=False Set verbose output

--debug=False Set debug output

upload command for uploading a blueprint to the management server

```
usage: cfy blueprints upload [-h] -p BLUEPRINT_FILE -b BLUEPRINT_ID [-v]
                             [--debug]
```

Options:

-p, --blueprint-path Path to the application's blueprint file

-b, --blueprint-id The id of the blueprint

-v=False, --verbose=False Set verbose output

--debug=False Set debug output

delete command for deleting a blueprint

```
usage: cfy blueprints delete [-h] -b BLUEPRINT_ID [-v] [--debug]
```

Options:

-b, --blueprint-id The id of the blueprint

-v=False, --verbose=False Set verbose output

--debug=False Set debug output

1.3 deployments

```
usage: cfy deployments [-h] {outputs,create,list,delete} ...
```

Sub-commands:

outputs command for getting a specific deployment outputs

```
usage: cfy deployments outputs [-h] [-d DEPLOYMENT_ID] [-v] [--debug]
```

Options:

-d, --deployment-id The id of the deployment to get outputs for

-v=False, --verbose=False Set verbose output

--debug=False Set debug output

create command for creating a deployment of a blueprint

```
usage: cfy deployments create [-h] [-d DEPLOYMENT_ID] [-i INPUTS] -b
                              BLUEPRINT_ID [-v] [--debug]
```

Options:

-d, --deployment-id A unique id that will be assigned to the created deployment

-i, --inputs Inputs file/string for the deployment creation (formatted as YAML or as "key1=value1;key2=value2")

-b, --blueprint-id The id of the blueprint

-v=False, --verbose=False Set verbose output

--debug=False Set debug output

list command for listing all deployments or all deployments of a blueprint

```
usage: cfy deployments list [-h] [-b BLUEPRINT_ID] [-v] [--debug]
```

Options:

-b, --blueprint-id The id of the blueprint

-v=False, --verbose=False Set verbose output

--debug=False Set debug output

delete command for deleting a deployment

```
usage: cfy deployments delete [-h] [-f] [-d DEPLOYMENT_ID] [-v] [--debug]
```

Options:

-f=False, --ignore-live-nodes=False Delete the deployment even if there are existing live nodes for it

-d, --deployment-id the id of the deployment to delete

-v=False, --verbose=False Set verbose output

--debug=False Set debug output

1.4 executions

```
usage: cfy executions [-h] {cancel,start,list,get} ...
```

Sub-commands:

cancel Cancel an execution by its id

```
usage: cfy executions cancel [-h] -e EXECUTION_ID [-f] [-v] [--debug]
```

Options:

-e, --execution-id The id of the execution to cancel

-f=False, --force=False Terminate the execution abruptly, rather than request an orderly termination

-v=False, --verbose=False Set verbose output

--debug=False Set debug output

start Command for starting a workflow execution on a deployment

```
usage: cfy executions start [-h] [--allow-custom-parameters]
                            [-d DEPLOYMENT_ID] [-p PARAMETERS] [-f]
                            [--timeout TIMEOUT] -w WORKFLOW [-l] [-v]
                            [--debug]
```

Options:

--allow-custom-parameters=False Allow the passing of custom parameters (parameters which were not defined in the workflow's schema in the blueprint) to the execution

-d, --deployment-id The deployment id

-p={}, --parameters={} Parameters for the workflow execution (formatted as YAML or as "key1=value1;key2=value2")

-f=False, --force=False Whether the workflow should execute even if there is an ongoing execution for the provided deployment

--timeout=900 Operation timeout in seconds (The execution itself will keep going, it is the CLI that will stop waiting for it to terminate)

-w, --workflow The workflow to start

-l=False, --include-logs=False Include logs in returned events

-v=False, --verbose=False Set verbose output

--debug=False Set debug output

list command for listing all executions of a deployment

```
usage: cfy executions list [-h] [-d DEPLOYMENT_ID] [--system-workflows] [-v]
                           [--debug]
```

Options:

-d, --deployment-id filter executions for a given deployment by the deployment's id

--system-workflows=False Include executions of system workflows.

-v=False, --verbose=False Set verbose output

--debug=False Set debug output

get command for getting an execution by its id

```
usage: cfy executions get [-h] -e EXECUTION_ID [-v] [--debug]
```

Options:

-e, --execution-id The id of the execution to get

-v=False, --verbose=False Set verbose output

--debug=False Set debug output

1.5 local

```
usage: cfy local [-h]
                 {execute,install-plugins,instances,init,create-requirements,outputs}
                 ...
```

Sub-commands:

execute Execute a workflow locally

```
usage: cfy local execute [-h] [--allow-custom-parameters] [-p PARAMETERS]
                         [--task-retries TASK_RETRIES] -w WORKFLOW
                         [--task-thread-pool-size TASK_THREAD_POOL_SIZE]
                         [--task-retry-interval TASK_RETRY_INTERVAL] [-v]
                         [--debug]
```

Options:

--allow-custom-parameters=False Allow the passing of custom parameters (parameters which were not defined in the workflow's schema in the blueprint) to the execution

-p={}, --parameters={} Parameters for the workflow execution (formatted as YAML or as "key1=value1;key2=value2")

--task-retries=0 How many times should a task be retried in case it fails

-w, --workflow The workflow to execute locally

--task-thread-pool-size=1 The size of the thread pool size to execute tasks in

--task-retry-interval=1 How many seconds to wait before each task is retried

-v=False, --verbose=False Set verbose output

--debug=False Set debug output

install-plugins Installs the necessary plugins for a given blueprint

```
usage: cfy local install-plugins [-h] -p BLUEPRINT_PATH [-v] [--debug]
```

Options:

-p, --blueprint-path Path to a blueprint

-v=False, --verbose=False Set verbose output

--debug=False Set debug output

instances Display node instances

```
usage: cfy local instances [-h] [--node-id NODE_ID] [-v] [--debug]
```

Options:

--node-id Only display node instances of this node id

-v=False, --verbose=False Set verbose output

--debug=False Set debug output

init Init a local workflow execution environment in in the current working directory

```
usage: cfy local init [-h] [--install-plugins] -p BLUEPRINT_PATH [-i INPUTS]
                      [-v] [--debug]
```

Options:

--install-plugins=False Install necessary plugins of the given blueprint.

-p, --blueprint-path Path to a blueprint

-i, --inputs Inputs file/string for the local workflow creation (formatted as YAML or as "key1=value1;key2=value2")

-v=False, --verbose=False Set verbose output

--debug=False Set debug output

create-requirements Creates a PIP compliant requirements file for the given blueprint

```
usage: cfy local create-requirements [-h] [-o REQUIREMENTS_OUTPUT] -p
                                     BLUEPRINT_PATH [-v] [--debug]
```

Options:

-o, --output Path to a file that will hold the requirements of the blueprint

-p, --blueprint-path Path to a blueprint

-v=False, --verbose=False Set verbose output

--debug=False Set debug output

outputs Display outputs

```
usage: cfy local outputs [-h] [-v] [--debug]
```

Options:

-v=False, --verbose=False Set verbose output

--debug=False Set debug output

1.6 events

```
usage: cfy events [-h] {list} ...
```

Sub-commands:

list Displays Events for different executions

```
usage: cfy events list [-h] -e EXECUTION_ID [-l] [--tail] [-v] [--debug]
```

Options:

-e, --execution-id The id of the execution to list events for

-l=False, --include-logs=False Includes logs in the returned events

--tail=False tail the events of the specified execution until it ends

-v=False, --verbose=False Set verbose output

--debug=False Set debug output

1.7 workflows

```
usage: cfy workflows [-h] {list,get} ...
```

Sub-commands:

list command for listing workflows for a deployment

```
usage: cfy workflows list [-h] [-d DEPLOYMENT_ID] [-v] [--debug]
```

Options:

-d, --deployment-id The id of the deployment whose workflows to list

-v=False, --verbose=False Set verbose output

--debug=False Set debug output

get command for getting a workflow by its name and deployment

```
usage: cfy workflows get [-h] [-d DEPLOYMENT_ID] -w WORKFLOW [-v] [--debug]
```

Options:

-d, --deployment-id The id of the deployment for which the workflow belongs

-w, --workflow The id of the workflow to get

-v=False, --verbose=False Set verbose output

--debug=False Set debug output

CHAPTER 2

Indices and tables

- genindex
- modindex
- search