MiCADO command line

different command:

start
: this command use the ClouldBroker API to launch instance to be added to the infrastructure.

stop
: this command stop the desire instance running in the infrastructure.

launch
: this command launch a container on the infrastructure.

quit
: this command allow to stop a running container.

list
: this command list either the container or the instances running on the infrastructure.

build
: this command allow to build a new docker image.

info
: this command gives info on the running containers (special for the developers)

logs
: this command give back the logs of the wanted container to be able to debug them.

remove
: this command allow to remove the wanted container (the container should not be on running mode).

execute
: this command allow to execute any command inside the container.

different parameter for each commands:

start

-c / --cloud: allow to choose on which Cloud to start a new instance.
-a / --application: allow to choose which instance to run.
-f / --flavour: allow to choose which flavour for the instance.
-t / --tag: assign a Tag to the instance to launch.
-n / --name: choose on either application or a database server.
-I: parameter to use for launching the first instance.

example

./main.py start -c CloudSigma -a “Node DSCR” -f Small -t application01 -n application

stop

-t / --tag: the name of the tag given for the instance we want to stop.

example

./main.py stop -t application01

launch

-i / --image: the name of the image container wanted.
-n / --name: the name of the client to the container is launched for.
-l / --location: where the container should run.
-s / --script: inside path of the script to be run at the execution of the container.
-d / --domain: domain name to access the application run in the container run.
-a / --arguments: extra argument to pass to the container when it will be run.
-p / --port: choose to run the container on a specific port of the instance.
-e / --environment: allow to set an Environment variable inside the container.

example

./main.py launch -i container-default -n client -l application -s /usr/src/app/app.sh -d domain -a user host_ip application address port

quit

-n / - -name: name of the container to stop
-rm: option to remove the container after it has been stopped.

example

./main.py quit -n container-abc -rm

list

-i / --instance: list all the instance up in the infrastructure.
-c / --containers: list all the containers running in the infrastructure.
-a / --all: optional argument for the listing all the container stop or not in the infrastructure.

example

./main.py list -c -a

build

-n / - -name: name to give the image of the docker image. This name should have the pattern of “container-name”
-f / - -filename: tarball containing the Dockerfile and all other files needed. those file should be inside a directory having the name you want to give to the container.

example

./main.py build -n container-application -f tarball.tar

info

-s / - -service: name of the services to get the info from
-a / - -all: list all the service running on the infrastructure

example

./main.py info -a

logs

-n / - -name: container name to get the logs from.

example

./main.py logs -n container-x

remove

-n / - -name: name of the container to remove, it should not be running.

example

./main.py remove -n container-x

execute

-n / - -name: name of the container to execute the command in.
-c / - -command: command to be executed inside the container.

example

./main.py execute -n container-x -c bash