OVHcloud ovhai CLI Cheat Sheet

About ovhai CLI

`ovhai` is a CLI that allows you to interact with OVHcloud AI Training (https://www.ovhcloud.com/fr/public-cloud/ai-training/). OVHcloud AI training is a cloud service that allows Data Scientists, ML Engineers and Deep Learning practitioners to create and allocate training ressources on demand.

With the CLI you can: * manage jobs (list, run, stop, execute commands inside…) * manage AI notebooks (start, stop, sync…) * manage object storage (upload data to object storage, attach data to a notebook, delete a bucket/container …) * manage apps (list, run, delete…) * manage containers registries (list, add a private registry…) * manage access tokens * …

Installation

Install through curl

Define the region and then download the binary according to your OS:

```
$ export REGION=gra ; curl -O https://cli.$REGION.trai
$ unzip ovhai-darwin.zip
$ sudo mv ovhai /usr/local/bin/ovhai
$ rm ovhai-darwin.zip
```

Prerequisites

Create an OpenStack user

Create an OpenStack User with the roles "AI Training Operator" and "ObjectStore operator". Follow the guide: https://docs.ovh.com/gb/en/ai-training/create-user/

Usage

Show version

```
$ ovhai --version
```

```
ovhai 2.3.1
```

Login

```
$ ovhai login
```

It's the first command you need to execute. To login into AI Training platform you need the credentials of a user with the roles "AI Training Operator" and ObjectStore operator. Enter the user name and password.

Show current user information

```
$ ovhai me
```

```
Name:      user-NnnmWdWTFjn3
Tenant:    a123a4e56b789c1ba23a456b789fcf01
Trust Id: bea1c2cc345e67a89adf1023c45ccd67
```

Show all the regions and environments

```
$ ovhai config ls
```

```
NAME URL
BHS  https://bhs.training.ai.cloud.ovh.net/v1
GRA  https://gra.training.ai.cloud.ovh.net/v1
```

Jobs

A job is a Docker container running on the OVHcloud AI infrastructure. You can specify a Docker image (from Docker Hub or a private registry), allocate the request hardware requirements (number of GPUs, number of CPUs), link the data in input and output to an Object Storage through volumes, and run it.

A job have a start and an end.

List jobs

```
$ ovhai job ls
```

Run a simple job

```
$ ovhai job run ubuntu—bash -c "while true; do
echo toto; sleep 2; done"
```

Run a job and pass your SSH key

```
$ ovhai job run ubuntu -s $HOME/.ssh/id_rsa.pub
—bash -c "sleep infinity"
```

Run a job, with 1 CPU

```
$ ovhai job run <docker_image> --cpu 1—sleep
infinity
```

-c, --cpu: number of CPUs (ignored if GPUs is specified)

Run a job, with 2 GPUs and plug it to an object storage container (bucket) in read only access mode

```
$ ovhai job run <docker_image> --gpu 2 -v
<container_name>@<REGION>:/data:ro—sleep
infinity
```

-g, –gpu: number of GPUs (default 1)

Get job status

```
$ ovhai job get <job_ID>
```

Diplay job logs

```
$ ovhai job logs <job_ID>
```

Diplay job logs and stream the logs

```
$ ovhai job logs -f <job_ID>
```

Execute commands in a job

```
$ ovhai job exec -it <job_ID>—bash
```

Synchronize - Manually push data from Job to Object Storage (while it is running)

```
$ ovhai job push-data <job_ID>
```

Synchronize - Manually pull data from Object Storage to Job (while it is running)

```
$ ovhai job pull-data <job_ID>
```

Stop a job

```
$ ovhai job stop <job_ID>
```

Data (Object Storage)

Jobs are containers so by default there are ephemeral, it means that if you store data inside jobs, you can lose them. The solution is to link your jobs and notebooks to Object Storage containers (buckets).

List Object Storage containers

```
$ ovhai data ls <REGION>
```

List Object Storage containers that starting with "test%"

```
$ ovhai data ls <REGION> --prefix test
```

Push files (objects) to my-container

```
$ ovhai data upload <REGION> my-container
some/local-file other-file
```

Delete an object on my-container

```
$ ovhai data delete <REGION> my-container my-
object
```

Delete my-container

```
$ ovhai data delete <REGION> my-container
```

Delete all of your containers starting by "test%"

```
$ ovhai data delete <REGION> --prefix test
```

Delete all of your containers

```
$ ovhai data delete <REGION> --all
```

Apps

An app is like a job but for API or daemon process that should never stop. An app runs as a group of load balanced Docker containers within OVHcloud AI infrastructure.

You can specify a Docker image (from Docker Hub or a private registry), allocate the request hardware requirements (number of GPUs, number of CPUs), link the data in input and output to an Object Storage through volumes, and run it.

List apps

```
$ ovhai app ls
```

Run an app and specify we want 3 replicas

```
$ ovhai app run priv-registry.gra.training.ai.cloud.ovh.net/public/infrastr
world:latest -p 8080 --cpu 1 --fixed-scale 3
```

Run an app and mount a volume linked to an Object Storage container

```
$ ovhai app run <docker-image> --gpu 4 --volume my-container@<REGION>:/data
```

Get app status

```
$ ovhai app get <app_ID>
```

Get app’s URL

```
$ ovhai app get <app_ID> -o json | jq ".status.url"
```

Stop an app

```
$ ovhai app stop <app_ID>
```

Notebooks

xxx

xxx

Registries

A registry is a place where you can push and pull your Docker images. By default, you have access to a shared registry scoped to your project, you can only push images and use them for jobs. You can also add private registries.

List registries

```
$ ovhai registry ls
```

Add a private registry

```
$ ovhai registry add <url>
```

Get registry information

```
$ ovhai registry get <registry_ID>
```

Delete a private registry

```
$ ovhai registry delete <registry_ID>
```

Authors :

@aurelievache
DevRel at OVHcloud

v1.0.0