Tutoriel OpenShift CLI (oc) : Premiers pas

1. Installation de l'outil oc

OpenShift propose une interface en ligne de commande très puissante appelée oc.
Pour l'installer :

- Ouvre ton tableau de bord OpenShift dans ton navigateur.
- Clique sur **l'icône Aide** (le point d'interrogation en haut à droite).
- Choisis **Command Line Tools**.
- Télécharge l'outil pour ton système (Windows, Linux ou Mac).
- Installe-le selon ton système :

Linux :

tar xvf oc.tar

sudo mv oc /usr/bin

macOS :

unzip oc.zip

sudo mv oc /usr/local/bin

Windows (en PowerShell Administrateur) :

Expand-Archive oc.zip

cd .\oc

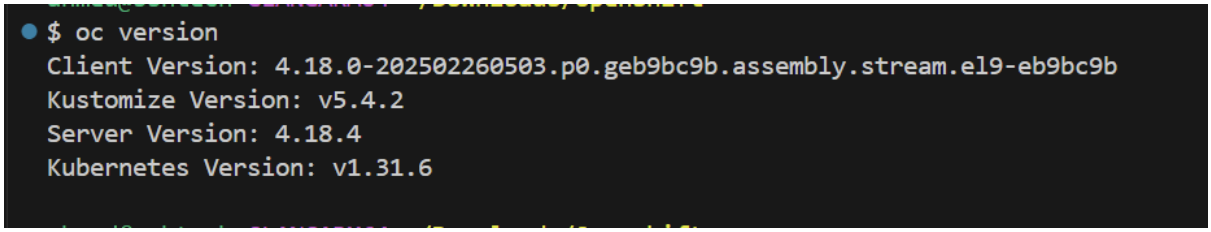
mkdir $ENV:ProgramFiles\OpenShift-Client

Move-Item oc.exe $ENV:ProgramFiles\OpenShift-Client

Vérifie l'installation :

oc version

Résultat attendu : la version du client oc s'affiche. Tu peux ignorer l'erreur "Unauthorized" pour l'instant.



- **Client Version** : La version de ton binaire oc installé localement (4.18.0).
- **Kustomize Version** : La version intégrée de Kustomize utilisée pour manipuler les manifests YAML.
- **Server Version** : La version d'OpenShift sur ton cluster distant (4.18.4).
- **Kubernetes Version** : Le moteur Kubernetes sur lequel OpenShift s'appuie (v1.31.6).

---

2. Connexion au cluster OpenShift

Pour te connecter :

- Dans ton tableau de bord OpenShift, clique sur ton projet.
- Clique sur **Copy login command**.
- Copie la commande affichée et exécute-la dans ton terminal :

  oc login --token=TON_TOKEN --server=TON_SERVEUR

Résultat attendu : un message confirmant que tu es connecté et indiquant ton projet actuel.

---

3. Créer et exposer une application depuis une image

Pour créer une application à partir d'une image existante :

oc new-app nginx --name=nginx-test --labels=app=nginx-test

Explication :

- **oc new-app** : crée une nouvelle application à partir d'une image existante ou d'un code source.
- **--image=nginx** : précise l'image à utiliser (stockée sur quay.io).
- **--name=nginx-test** : donne le nom **nginx-test** à ton application.
- **--labels=app=nginx-test** : ajoute un label personnalisé (app=nginx-test) pour faciliter la gestion de l'app.

Résultat :

- OpenShift a trouvé l'image nginx.
- Il a créé **trois objets** :
    - **ImageStream** : pour suivre la version de l'image.
    - **Deployment** : pour déployer ton application.
    - **Service** : pour exposer ton application à l'intérieur du cluster.

```
$ oc new-app nginx --name=nginx-test --labels=app=nginx-test
 --> Found image 9333c1e (4 months old) in image stream "openshift/nginx" under tag "1
 .24-ubi9" for "nginx"

     Nginx 1.24
     ----------
     Nginx is a web server and a reverse proxy server for HTTP, SMTP, POP3 and IMAP pr
 otocols, with a strong focus on high concurrency, performance and low memory usage. T
 he container image provides a containerized packaging of the nginx 1.24 daemon. The i
 mage can be used as a base image for other applications based on nginx 1.24 web serve
 r. Nginx server image can be extended using source-to-image tool.

     Tags: builder, nginx, nginx-124


 --> Creating resources with label app=nginx-test ...
     deployment.apps "nginx-test" created
     service "nginx-test" created
 --> Success
     Application is not exposed. You can expose services to the outside world by execu
 ting one or more of the commands below:
      'oc expose service/nginx-test'
     Run 'oc status' to view your app.
```

Expose ensuite ton application au public :

oc expose service/nginx-test

Résultat : un objet Route est créé, permettant l'accès via une URL publique.

```
$ oc expose service/nginx-test
route.route.openshift.io/nginx-test exposed
```

Maintenant, récupère l'URL pour accéder à ton app avec la commande suivante :

oc get route nginx-test -o jsonpath='{.spec.host}'

---

4. Déployer une application depuis un dépôt Git

Pour créer une application directement depuis un code source :

oc new-app https://github.com/redhat-developer-demos/qotd.git --name=qotd --labels=sandbox=qotd

```
$ oc new-app https://github.com/redhat-developer-demos/qotd.git --name=qotd --labels=sandbox=qot
d
--> Found container image ec829c4 (3 weeks old) from docker.io for "docker.io/golang:latest"

    * An image stream tag will be created as "golang:latest" that will track the source image
    * A Docker build using source code from https://github.com/redhat-developer-demos/qotd.git w
ill be created
      * The resulting image will be pushed to image stream tag "qotd:latest"
      * Every time "golang:latest" changes a new build will be triggered

--> Creating resources with label sandbox=qotd ...
    imagestream.image.openshift.io "golang" created
    imagestream.image.openshift.io "qotd" created
    buildconfig.build.openshift.io "qotd" created
    deployment.apps "qotd" created
    service "qotd" created
--> Success
    Build scheduled, use 'oc logs -f buildconfig/qotd' to track its progress.
    Application is not exposed. You can expose services to the outside world by executing one or
 more of the commands below:
     'oc expose service/qotd'
    Run 'oc status' to view your app.
```

Résultat : OpenShift :

- a trouvé l'image docker.io/golang:latest
- a **planifié un Build** (construction de l'application à partir du code source Go).
- a créé tous les objets nécessaires :
    - **ImageStream** (golang, qotd)
    - **BuildConfig** (qotd)
    - **Deployment** (qotd)
    - **Service** (qotd)

Puis expose-la :

oc expose service/qotd

Résultat attendu : ton application sera accessible via une URL publique générée.

---

5. Lister les pods :

Pour lister les pods liés à un déploiement spécifique :

oc get pods

```
$ oc get pods
NAME                    READY   STATUS      RESTARTS   AGE
qotd-1-build            0/1     Completed   0          22m
qotd-84f95489cd-w7bph   1/1     Running     0          20m
```

Résultat attendu : la liste des pods associés.

---

6. Lister les déploiements :

Pour lister les déploiements :

oc get deploy

```
$ oc get deploy
NAME   READY   UP-TO-DATE   AVAILABLE   AGE
qotd   1/1     1            1           24m
```

---

7. Modifier un déploiement en direct

Pour éditer un déploiement directement :

oc edit deploy/qotd

Dans l'éditeur qui s'ouvre, ajoute par exemple un nouveau label :

labels:

editedby: ahmed

Sauvegarde et ferme.

Vérifie ensuite :

oc get deploy/qotd -o jsonpath='{.metadata.labels}'

Résultat attendu : tu verras ton label ajouté dans la liste.

---

8. Déboguer un pod

Pour ouvrir une session de debug dans un pod :

oc debug pod/qotd

Résultat attendu : un terminal à l'intérieur du pod s'ouvre.
Tape exit pour revenir à ton terminal local.

---

9. Filtrer les ressources avec un label

Pour voir toutes les ressources associées à un label spécifique :

oc get all -l editedby=ahmed

Résultat attendu : OpenShift liste les objets correspondant au label donné.

---

10. Supprimer des ressources en fonction d'un label

Pour supprimer toutes les ressources créées avec un label spécifique :

oc delete all -l editedby=ahmed

Résultat attendu : confirmation de suppression des ressources.
