Gérer les conteneurs

Dans cette partie, nous verrons comment créer un conteneur, mais aussi comment le supprimer, comment les gérer, les relancer et pleins d'autres choses indispensables.

Etape 1 : Lancer, arrêter et lister des conteneurs

Un conteneur ne peut se lancer que s'il a une commande à exécuter.

La première commande que nous utiliserons, sera **docker run** qui s'utilise comme ceci :

```
docker run alpine echo "Hello World"
```

Il s'est passé quoi ?

- Nous avons créé et exécuté notre conteneur, mais puisqu'il n'a pas trouvé l'image **alpine** en local, il l'a téléchargé de lui même (sans avoir à utiliser `docker image pull`).
- Ensuite il a exécuté la commande qu'on lui a passé, à savoir écrire "Hello World".
- Et c'est tous, puisque l'echo est terminé, il a éteint le conteneur.

Nous allons maintenant vérifier mes dires, nous allons vérifier si ce conteneur est démarré ou pas, pour ce faire nous utiliserons **docker container ls** ou **docker ps** :

```
docker ps
```

Nous n'avons aucun conteneur en cours.

Pour le voir, il suffit d'ajouter l'option -a, qui permet de voir tous les conteneurs :

```
docker ps -a
```

- CONTAINER ID : ID du conteneur, généré de manière à ce qu'il soit unique
- IMAGE : L'image utilisée pour ce conteneur
- COMMAND : La commande exécutée
- CREATED : Temps depuis création du conteneur
- STATUS : Le statut actuel du conteneur, ici exited avec un code retour 0 (sans erreur) depuis 5 minutes
- PORTS : Liste des ports écoutés (nous verrons ceci plus tard)
- NAMES : Nom du conteneur, ici c'est un nom aléatoire car nous n'en avons pas défini à notre conteneur

Etape 2 : Exécuter un conteneur en mode attaché

Il est possible de se connecter en lançant une session interactive -i dans un terminal -t et en y exécutant **bash** :

```
docker run -t -i ubuntu:latest /bin/bash
```


```
root@c7d659d18356:/# ps
root@c7d659d18356:/# hostname
root@c7d659d18356:/# exit
```

Nous n'avons plus aucun conteneur en cours.

```
docker ps
```

Pour voir les conteneurs qui sont arrêtés

```
docker ps -a
```

Mais comment le relancer ?

```
vagrant@mydocker:~$ docker start c7d659d18356
c7d659d18356
```

Vérifier que le conteneur est démarré

```
docker ps
```

```
vagrant@mydocker:~$ docker ps
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS              PORTS               NAMES
c7d659d18356        ubuntu:latest       "/bin/bash"         15 hours ago        Up 5 minutes                            adoring_proskuriakova
```

Lorsque vous démarrez un conteneur arrêté, vous ne pouvez pas voir la sortie. Mais vous pouvez vous y attacher.

```
docker attach c7
```

```
vagrant@mydocker:~$ docker attach c7
root@c7d659d18356:/#
```

La commande (ctrl + p) + q, permet de quitter le terminal sans arrêter le conteneur

```
(ctrl + p ) + q
```

Vérifier que le conteneur est toujours démarré

```
docker ps
```

```
vagrant@mydocker:~$ docker ps
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS              PORTS               NAMES
c7d659d18356        ubuntu:latest       "/bin/bash"         15 hours ago        Up 8 minutes                            adoring_proskuriakova
```

Pour arrêter le process

```
docker attach c7
```

Puis

```
root@c7d659d18356:/# exit
```

Vérifier que le conteneur est arrêté

```
docker ps
```

Etape 3 : Exécuter un conteneur en mode détaché

```
docker run -d  centos ping 127.0.0.1
```

```
vagrant@mydocker:~$ docker run -d -it centos ping 127.0.0.1
Unable to find image 'centos:latest' locally
latest: Pulling from library/centos
d8d02d457314: Pull complete
Digest: sha256:307835c385f656ec2e2fec602cf093224173c51119bbebd602c53c3653a3d6eb
Status: Downloaded newer image for centos:latest
f4a0b3f1ebfc2187ee516ba79dbce313cbfc998f3f64007733ad73dec736d223
```

Vérifier que le conteneur est démarré

```
docker ps
```

```
vagrant@mydocker:~$ docker ps
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS              PORTS               NAMES
f4a0b3f1ebfc        centos              "ping 127.0.0.1"    47 seconds ago      Up 43 seconds                           boring_curie
```

Voir les logs des conteneurs

```
vagrant@mydocker:~$ docker logs f4
```

```
vagrant@mydocker:~$ docker logs f4
PING 127.0.0.1 (127.0.0.1) 56(84) bytes of data.
64 bytes from 127.0.0.1: icmp_seq=1 ttl=64 time=0.018 ms
64 bytes from 127.0.0.1: icmp_seq=2 ttl=64 time=0.075 ms
```

Attacher un terminal

```
vagrant@mydocker:~$ docker attach f4
```

```
vagrant@mydocker:~$ docker attach f4
64 bytes from 127.0.0.1: icmp_seq=294 ttl=64 time=0.022 ms
64 bytes from 127.0.0.1: icmp_seq=295 ttl=64 time=0.048 ms
64 bytes from 127.0.0.1: icmp_seq=296 ttl=64 time=0.051 ms
64 bytes from 127.0.0.1: icmp_seq=297 ttl=64 time=0.050 ms
```

Pour arrêter le processus

```
Ctrl + c
```

Vérifier que le conteneur est arrêté

```
docker ps
```

Conclusion : Attach ne sert pas à exécuter une tâche supplémentaire dans un conteneur, mais à attacher au processus en cours.

Exécuter un conteneur en mode détaché avec un nom

```
docker run -d -it --name=mycentos centos ping 127.0.0.1
```

Vérifier que le conteneur est démarré

```
docker ps
```

```
vagrant@mydocker:~$ docker ps
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS              PORTS               NAMES
c6875dfcfb87        centos              "ping 127.0.0.1"    11 minutes ago      Up 11 minutes                           mycentos
```

Nous allons utiliser « **docker exec** » qui est spécifiquement destiné à l'exécution de nouvelles choses dans un conteneur déjà démarré, que ce soit un shell ou un autre processus.

```
docker exec -it c6 bash
```

```
vagrant@mydocker:~$ docker exec -it c6 bash
[root@c6875dfcfb87 /]#
```

Pour voir les processus

```
[root@c6875dfcfb87 /]# ps all
```

```
[root@c6875dfcfb87 /]# ps all
F   UID   PID  PPID PRI  NI    VSZ   RSS WCHAN  STAT TTY        TIME COMMAND
4     0     1     0  20   0  24864  1964 skb_wa Ss+  pts/0      0:00 ping 127.0.0.1
4     0     6     0  20   0  11832  3036 do_wai Ss   pts/1      0:00 bash
0     0    23     6  20   0  49628  3044 -      R+   pts/1      0:00 ps all
[root@c6875dfcfb87 /]#
```

```
[root@c6875dfcfb87 /]# exit
```

Vérifier que le conteneur est toujours démarré

```
docker ps
```

```
vagrant@mydocker:~$ docker ps
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS              PORTS               NAMES
c6875dfcfb87        centos              "ping 127.0.0.1"    About an hour ago   Up About an hour                        mycentos
```

Etape 4 : Arrêt de conteneurs

```
docker ps
```

```
vagrant@mydocker:~$ docker ps
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS              PORTS               NAMES
c6875dfcfb87        centos              "ping 127.0.0.1"    About an hour ago   Up About an hour                        mycentos
```

```
docker stop mycentos
```

```
vagrant@mydocker:~$ docker stop mycentos
mycentos
```

```
docker ps
```

```
vagrant@mydocker:~$ docker ps
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS              PORTS               NAMES
```

Etape 5 : Retrait des conteneurs

Comme vous le rappelez, Docker garde les conteneurs arrêtés (sauf si vous les avez courus avec `-rm`)

L'ajout de `-a` montre tous les conteneurs:

```
docker ps -a
```

```
vagrant@mydocker:~$ docker ps -a
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS                      PORTS               NAMES
c6875dfcfb87        centos              "ping 127.0.0.1"    3 hours ago         Exited (137) 2 hours ago                        mycentos
f4a0b3f1ebfc        centos              "ping 127.0.0.1"    4 hours ago         Exited (0) 3 hours ago                          boring_curie
c7d659d18356        ubuntu:latest       "/bin/bash"         23 hours ago        Exited (130) 8 hours ago                        adoring_proskuriakova
```

```
docker rm mycentos
```

```
vagrant@mydocker:~$ docker rm mycentos
mycentos
```

```
docker ps -a
```

```
vagrant@mydocker:~$ docker ps -a
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS                      PORTS               NAMES
f4a0b3f1ebfc        centos              "ping 127.0.0.1"    4 hours ago         Exited (0) 3 hours ago                          boring_curie
c7d659d18356        ubuntu:latest       "/bin/bash"         23 hours ago        Exited (130) 8 hours ago                        adoring_proskuriakova
```

```
docker rm f4
```

```
vagrant@mydocker:~$ docker rm f4
f4
```

Si vous souhaitez tous les supprimer

```
vagrant@mydocker:~$ docker rm $(docker ps -a -q)
c7d659d18356
```

```
docker ps -a
```

```
vagrant@mydocker:~$ docker ps -a
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS              PORTS               NAMES
```