# Orsys - docker

Nicolas Rousset

Tuesday, 6th of december

## 1 Exercices courts

On rappelle que pour tester un Dockerfile, il faut se mettre dans le répertoire et tapez :

```
docker build -t nom_de_l_image .
```

Et on peut la lancer avec :

```
docker run nom_de_l_image
```

Tous les Dockerfile commencent par :

```
FROM <image de base>
```

Souvent FROM ubuntu dans les premiers exercices

## 2 Exercice 1 - ubuntu avec répertoire courant

En utilisant ADD, créez un Dockerfile qui ajoute le répertoire courant dans un répertoire /app

## 3 Exercice 2 - ubuntu avec postgresqlclient

En utilisant RUN, créez un Dockerfile qui contient ubuntu et le client postgresql (la commande pour l'installer est "apt-get update && apt-get install -y postgresql-client")

## 4 Exercice 3 - ubuntu plus

Créez une image partant de ubuntu, ubuntu_plus, avec les deux commandes suivantes

```
VOLUME /app
EXPOSE 5000
```

Maintenant comparez les résultats de docker image inspect ubuntu et docker image inspect ubuntu_plus

## 5 Exercice 4 - image hello world

Créez une image from ubuntu, avec la commande suivante :

```
CMD ["echo", "Hello world !"]
```

Exécutez la (docker run) sans commande pour le container. Que se passe-t-il ?

## 6 Exercice 5 - image python hello world

Créez une image depuis ubuntu, où vous installerez le package python3 (on a vu comment installer un package) et ajoutez les deux lignes suivantes :

```
CMD ["print('Hello world !')"]
ENTRYPOINT ["python3", "-c"]
```

Que se passe-t-il ? Comment se fait qu'une commande qui ne soit pas shell soit reconnue ?

## 7 Exercice 6 - env variable

Créez un Dockerfile de base ubuntu Définissez une variable VERSION_PYTHON dans une premier temps égale à 3, puis ajoutez trois lignes pour :

```
installer le package python$VERSION_PYTHON
exécutez la commande ln -s /bin/python$VERSION_PYTHON /bin/python
```

Puis faites en sorte que la commande par défaut python –version

Buildez et lancez deux fois cette image, une fois avec VERSION_PYTHON à 3, une fois à 2. Quelle est la différence ?

## 8 Exercice 6 - bonus

Même exercice que le précédent, mais utilisez ARG et faites donc en sorte que la version de python soit un argument de la commande.

## 9 Exercice 7 - workdir

Créez une image de ubuntu, avec les caractéristiques suivantes :

Dans le répertoire home, éxécutez les commandes :

```
touch home1
touch home2
```

Dans le répertoire root, éxécutez la commande :

```
touch root1
```

Enfin faites en sorte que lorsque l'on lance l'image en mode interactif, on arrive directement dans le répertoire /bin. Faites cela en utilisant 3 fois workdir.

## 9.1 Exercice 8 & 9 - pourquoi tant de and (&&) ?

Ecrivez deux Dockerfile, Dockerfile_01 et Dockerfile_02 (on rappelera l'otion -f de docker build pour choisir un docker)

Dans chacun, mettez les 3 lignes de commande suivantes :

```
apt-get update
apt-get install -y postgresql-client
rm -rf /var/lib/apt/lists/*
```

Mais dans le premier utilisez 3 commandes docker séparées, alors que dans le second, mettez les commandes sur la même ligne en utilisant && (mieux sur 3 lignes et une commande en utilisant , mais seulement si vous maitrisez)

Buildez les deux Dockerfile et comparez les tailles des images obtenus. Quelle différence ?

## 9.2 Exercice 10 - maven repository

Et maintenant nous allons créez une image depuis 0 ! Un petit projet spring boot, le but étant qu'en se connectant sur localhost:8080 (ou le port que vous aurez choisi d'allouer) un message "Greeting from Spring boot !" apparaisse.

Pour ceci, vous aurez besoin :

- de partir de l'image eclipse-temurin:11 (une des images java officiel de docker)
- d'installer les packages git et maven
- de cloner le projet sample : git clone https://github.com/Aenori/Spring_boot_sample.git