# Docker

Le but de ce petit projet est de démontrer l'utilisation de Docker en quelques étapes. Il rend disponile le jeu 2048, adapté en HTML5 par Gabriele Cirulli dans un container qui utilise NGINX comme serveur web.

## Table of Contents



## Mise en route

Après l'installation de Docker, exécuter la commande `docker run hello-world` pour démarrer le container de démonstration. Lire les différentes étapes suivies par Docker pour afficher le message. On remarque aussi que le container hello-world a été téléchargé de Docker Hub.

Si on relance la commande `docker run hello-world`, le résultat est plus rappide car le container a maintenant son image déjà téléchargée en local.

La commande `docker images` permet de visualiser les images déjà téléchargées.

## Construire une image

La recette de construction d'une image Docker s'écrit dans un fichier nommé **Dockerfile**. Pour commencer, créer le fichier `Dockerfile` et y écrire :

```
FROM ubuntu:20.04
```

> Cette recette dit que notre image sera basée sur l'image ubuntu:20:04. Sans autres indications, notre image sera en tout point identique à celle d'ubuntu:20.04

Pour construire l'image (en d'autres termes, rendre possible son exécution sur l'ordinateur de la personne, ou son partage sur Docker Hub), utiliser la commande :

```
docker build -t organisation/image:tag .
```

> Les personnes qui n'ont pas de compte payant sur Docker Hub doivent suivre la syntaxe `organisation/image`, par exemple `epfldojo/2048`. Si vide, le `tag` sera par défaut `latest`.

Dans notre cas, nous utilisons `docker build -t epfldojo/2048 .`.

## Lancer le container

Pour lancer et entrer dans le container et voir son contenu, utiliser :

```
docker run -it epfldojo/2048 bash
```

Le paramètre `-it` permet d'entrer dans le container en mode interactif et en allouant un tty. `bash` à la fin permet de redéfinir l'entrypoint du container.

> On remarque que le système de fichier, bien que possiblement similaire à celui du hôte, n'est pas le même. Il n'y a pas de partage des fichiers entre le container et l'hôte.

Si nécessaire, l'option `-d` permet de lancer le container en "deamon".

Aussi, l'option `--name` permet de nommer le container (pour le retrouver plus facilement avec la commande `docker ps`).

Pour exposer des ports, l'argument `-p` (hôte:container) permet de les exposer. par exemple :

```
docker run --rm --name 2048 -p 1337:80 -d epfldojo/2048
```

### Entrer dans le container

Lorsqu'un container est en service, utiliser le `exec` de docker pour y entrer, par exemple :

```
docker exec -it epfldojo/2048 bash
```

### Publier le container

Afin que d'autres personnes puissent utiliser le même container, la commande :

```
docker push epfldojo/2048
```

permet d'uploader l'image sur Docker Hub. Les autres utilisateurs peuvent lancer

```
docker run --rm --name 2048 -p 1337:80 -d epfldojo/2048
```

et se rendre sur https://localhost:1337 pour accéder au jeu.

### Pour le développement (volumes)

L'utilisation de volumes est très pratique pour le développmement. Par exemple, les commandes suivantes permettent de modifier le code de l'application tout en utilisant Docker pour le serveur web :

```
git clone https://github.com/gabrielecirulli/2048.git .
docker run -p 1337:80 -v $(pwd):/usr/share/nginx/html nginx
firefox https://localhost:1337
```

> En 3 commandes, l'utilisateur a alors un serveur web est le code source de l'application sur sa machine. La modification du code de 2048, par exemple changer le style CSS, est alors très simple et nécessite uniquement un rafraîchissement de la page.

### En résumé

L'option de rendre disponible l'image d'un container sur Docker Hub permet de facilement partager une application et de pouvoir la «shipper» sur d'autre serveurs. Bien sûr, l'application en question est une candidate facile, car elle est «stateless», c'est à dire qu'elle n'a pas besoin de stocker des données ou d'en échanger avec un système tier pour fonctionner.

L'utilisation de volumes permet d'avoir un environnement de développement rapide et évite de devoir installer des logiciels sur son ordinateur. De plus, cette méthode est facilement partageable entre développeur et assure l'homogénéité entre les environnements de développement.

## Pour aller plus loin

Pour en savoir plus sur Docker et les containers, commencer par lire la page Wikipedia. Le site officiel, https://docker.com et sa documentation sont assurément un passage incontournable. Ensuite, le site Kata Coda propose des formations interactives très intéressantes. Il est également nécessaire de mentionner `docker-compose`, l'outils qui permet de lancer un ensemble de container qui communiqueront, afin d'utiliser Docker dans des systèmes plus complexes, comme par exemple ceux qui doivent utiliser une base de donnée.

Finalement, l'expérience vient avec la pratique, s'imposer l'utilisation de Docker pour des projets même simples semble tout indiqué.