Introduction to Docker

RES, Lecture 3

Olivier Liechti

heig-vd

HAUTE ÉCOLE
D'INGÉNIERIE ET DE GESTION
DU CANTON DE VAUD

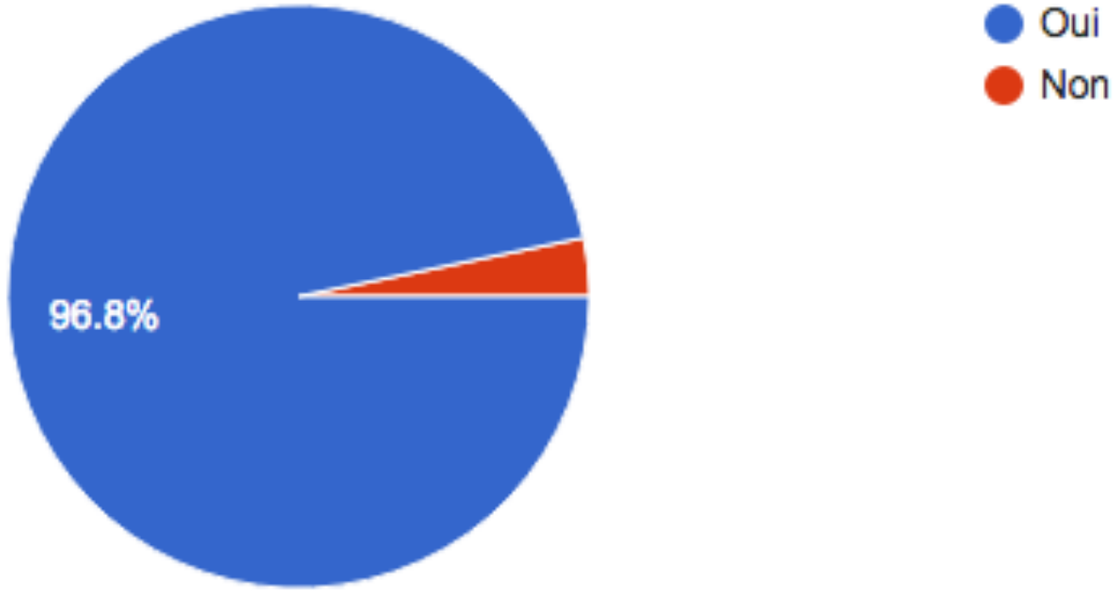
www.heig-vd.ch

heig-vd
HAUTE ÉCOLE
D'INGÉNIERIE ET DE GESTION
DU CANTON DE VAUD
www.heig-vd.ch

Preparation work for lab 02

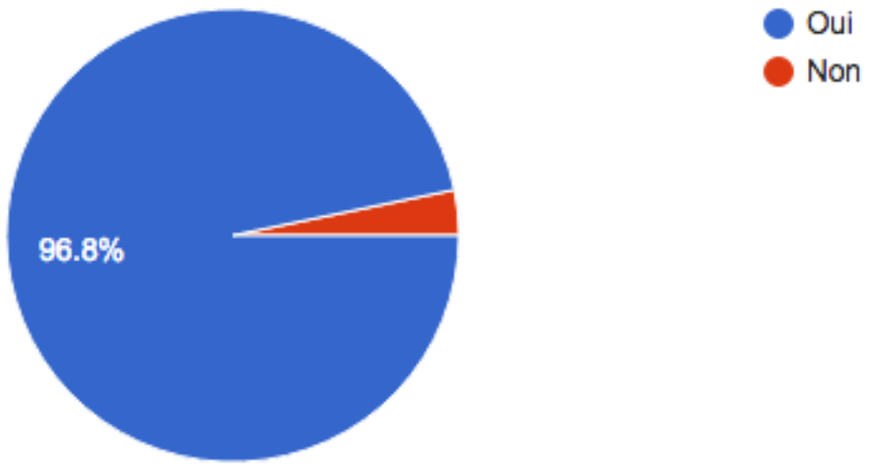
Je suis capable d'écrire un CLIENT TCP en Java. Je suis capable d'ouvrir une connexion vers un serveur, d'envoyer des bytes et de lire des bytes.

62 responses



Je suis capable d'écrire un SERVEUR TCP en Java, qui traite un client après l'autre (single threaded). Quand un client arrive, le serveur envoie l'heure courante (UTC) et ferme la connexion.

62 responses

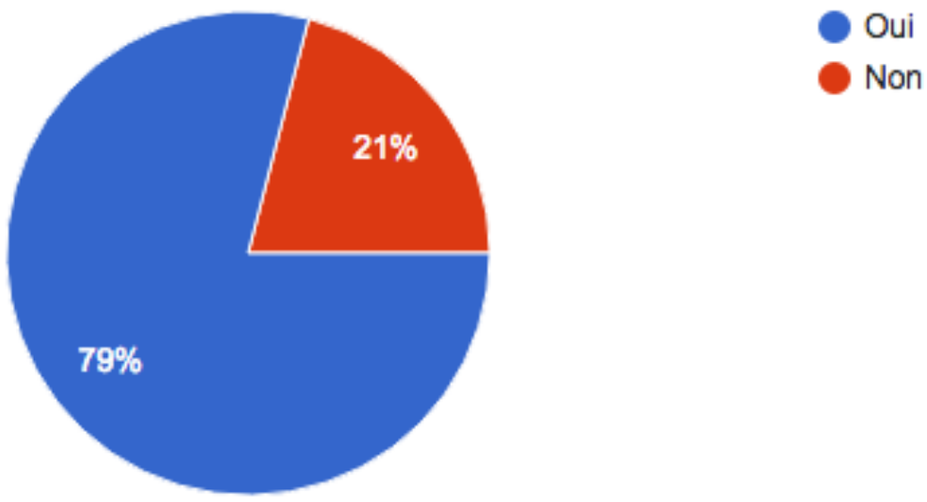


- (2)
Je ne sais pas le faire du tout
Aucun, j'ai suivi les exemples du cours.

Mauvaise gestion de mon temps et supposition erronée que je pourrais rattraper le retard pris un autre jour.

Je suis capable d'écrire un SERVEUR TCP en Java qui traite plusieurs clients en même temps (multi-threaded). Quand un client arrive, le serveur lui envoie l'heure courante une fois par seconde pendant 20 secondes. Après les 20 secondes, il ferme la connexion.

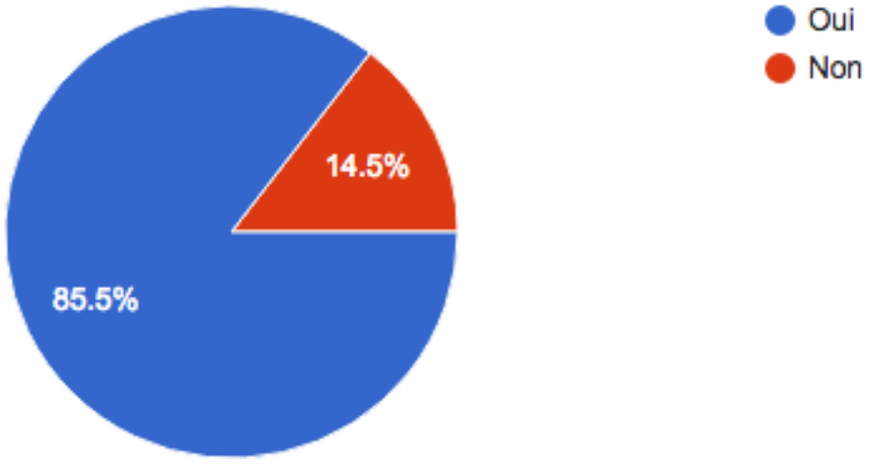
62 responses



Je ne sais pas le faire du tout
Pas eu le temps de faire cela
j'ai eu un petit soucis avec le flush de writer normalement il doit envoyé ligne par ligne mais il fait totalement l'inverse il stock dans le writer aprés quand le buffer est plein il envoie eu client
Le serveur n'envoie le temps qu'une seule fois par connection
Idem
la méthodologie pour la programmation multithreadée mais j'y travaille
j'ai rencontré un problème au niveau du lancement d'un thread à l'arrivée de chaque client
problème lié au lancement des threads en java notion pas claires
Je n'arrive pas à faire du multithreading en java, à bien décomposer le code du serveur pour executer les bonnes tâches. (ce qu'il faut déléguer dans la fonction run)
Je ne suis pas encore à l'aise avec des notions précédemment étudiés et je n'ai pas encore implémentée cette partie
Je n'ai pas encore terminé, mais il fallait soumette ce formulaire à 23h.

Je suis capable d'implémenter un SERVEUR TCP qui fournit l'heure dans 3 time zones: New York, Paris et Tokyo. Quand un client se connecte, il indique la time zone de son choix. Le serveur lui répond l'heure dans cette time zone et ferme la connexion.

62 responses



Je ne sais pas le faire du tout
Non par contre, je suis pas sûr si c'est la bonne méthode. Je pars du principe qu'il faut lire les données du client, enregistrer sa zone, vérifier la zone et lui donner la bonne heure via un switch.
-
Si l'utilisateur utilise la bonne syntaxe pour la fonction TimeZone.getTimeZone() il peut demander la TimeZone de son choix, en cas d'erreur, retourne l'heure en GMT 0
Je ne sais pas comment le vérifier, je n'arrive pas a créer un client qui fonctionne avec.
Petits problèmes au debut avec les TimeZone mais resolu.
Idem
la méthodologie pour la programmation multithreadée mais j'y travaille
je n'ai pas encore fini l'implémentation du serveur qui est selon moi la partie la plus compliqué
J'arrive à envoyer un message différent en fonction de ce que le client dit, mais je n'ai pas réussi à récupérer l'heure en fonction de la timezone
Même chose que en haut.

Schedule

heig-vd
HAUTE ÉCOLE
D'INGÉNIERIE ET DE GESTION
DU CANTON DE VAUD
www.heig-vd.ch

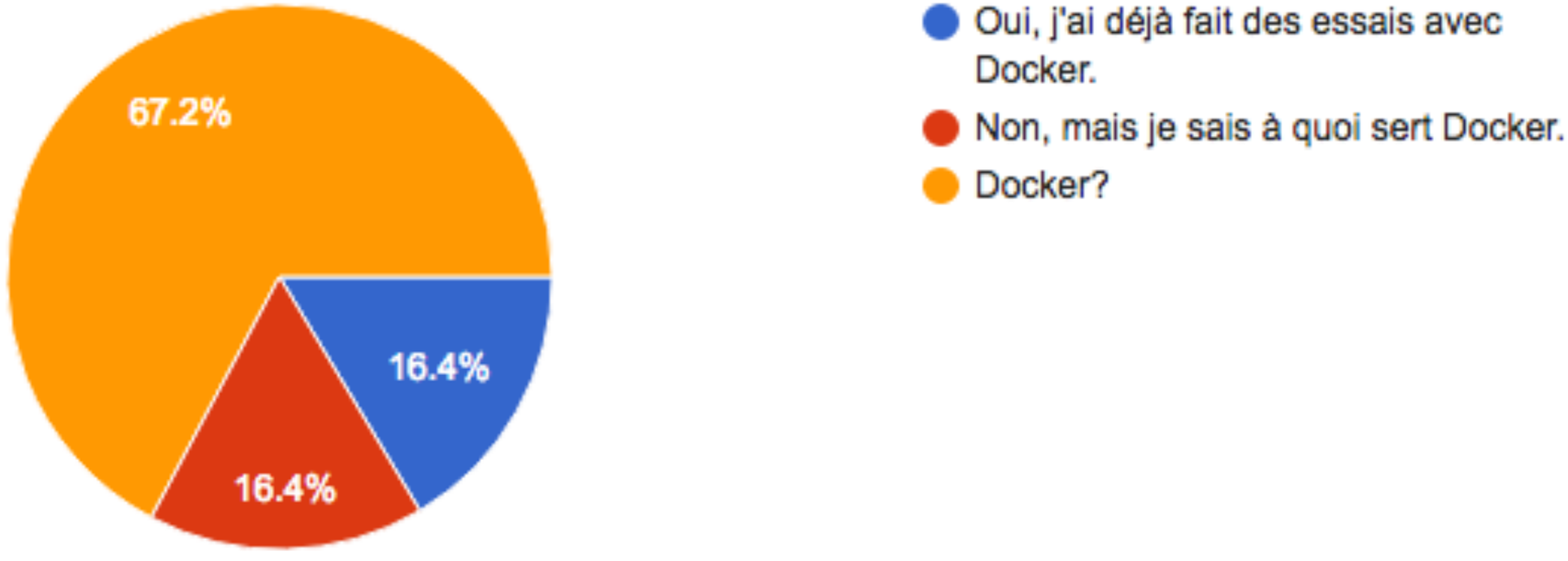
- **Mardi 20 mars**: introduction à Docker, revue TCP et préparation labo 02
- **Mercredi 21 mars** et **vendredi 23 mars**: validation de votre setup Docker et démarrage labo 02
- **Mardi 27 mars**: cours consacré au labo 02
- **Mercredi 28 mars** et **vendredi 30 mars**: vous devez avoir atteint les objectifs suivants:
  - être capable de développer un serveur TCP en Java et de le packager dans une image Docker
  - être capable de développer un client TCP en Java et de le packager dans une image Docker
  - être capable de lancer un container Docker depuis vos images
  - être capable d'afficher la liste des containers Docker qui tournent sur votre machine
  - être capable de vous connecter à un container Docker en exécution avec la commande docker exec -it
  - être capable de lancer un container en mode interactive avec la commande docker run -it
  - maîtriser les classes d'entrées/sorties vues dans les premiers labos
  - maîtriser le workflow de développement GitHub
- **Mercredi 4 avril 23h**: avoir achevé les étapes 1 à 4 du labo 02 et **avoir soumis vos tests automatisés** dans une PR
- **Dimanche 8 avril 23h**: rendu final du labo 02 (formulaire à suivre)

HAUTE ÉCOLE
D'INGÉNIERIE ET DE GESTION
DU CANTON DE VAUD
www.heig-vd.ch

Introduction to Docker

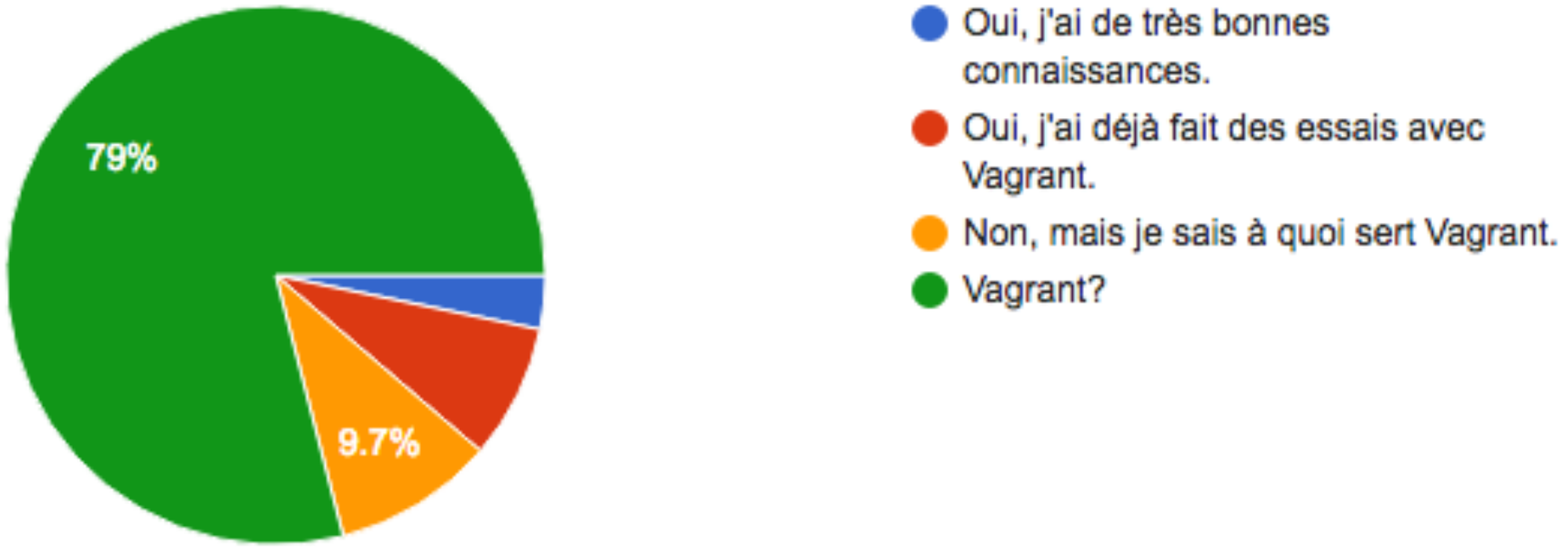
Est-ce que vous connaissez et avez utilisé la technologie Docker?

61 responses



Est-ce que vous connaissez et avez utilisé la technologie Vagrant?

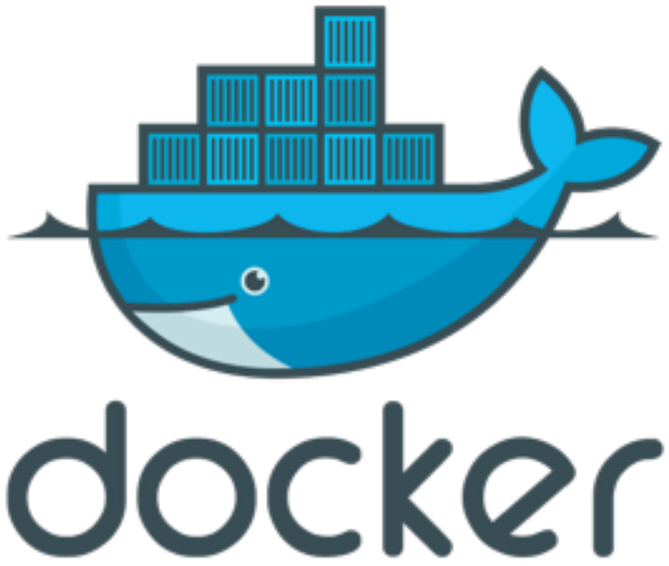
62 responses

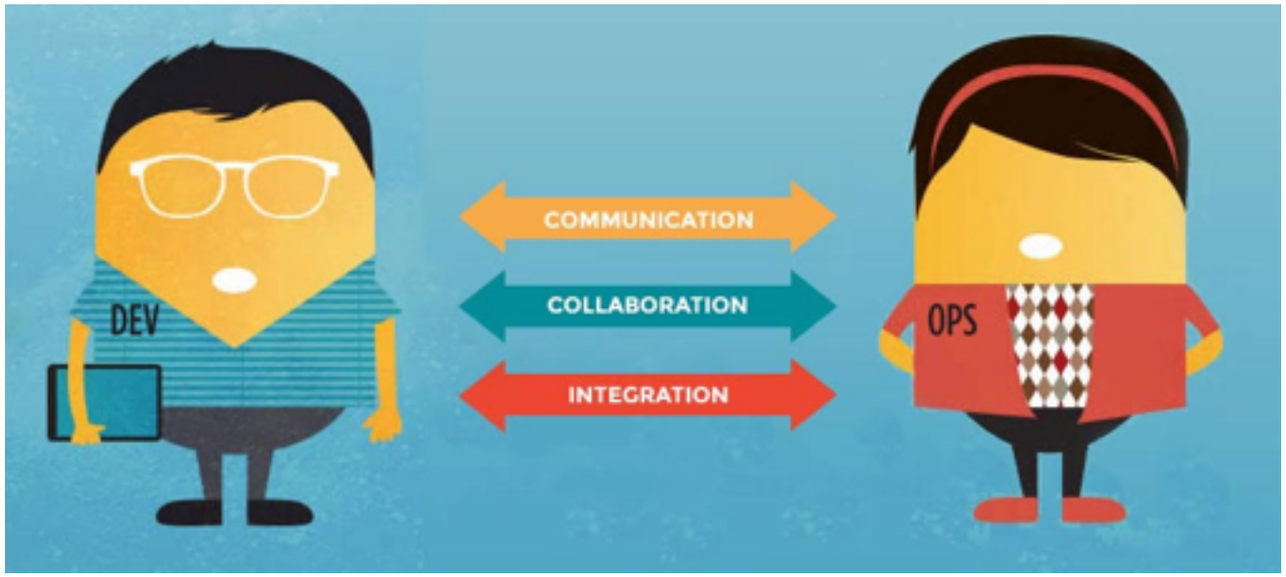


Vagrant permet d'**automatiser** la création et l'utilisation de **machines virtuelles**. Tout ce qui est fait via le GUI de VirtualBox peut être fait dans des scripts. La communauté peut également partager des "boxes" avec des piles logicielles.
Dans le cours, nous n'utilisons plus Vagrant. Vous allez encore en entendre parler dans certains webcasts, mais nous travaillons maintenant uniquement avec Docker.

heig-vd

Haute Ecole d'Ingénierie et de Gestion
du Canton de Vaud

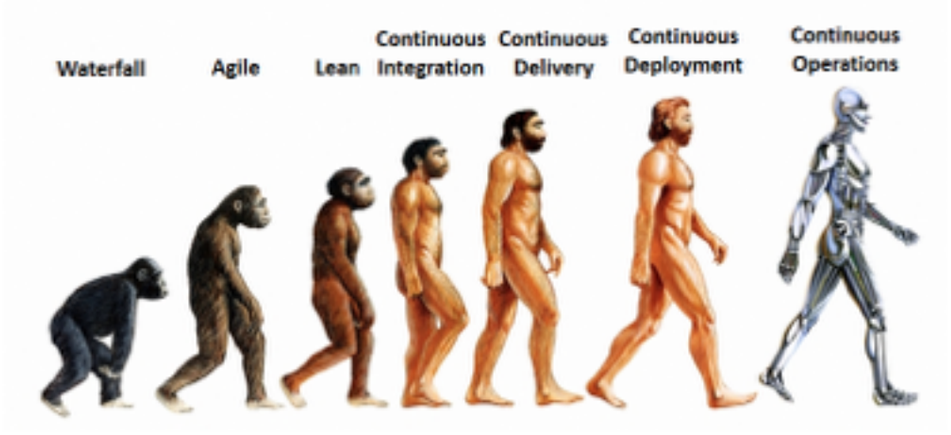


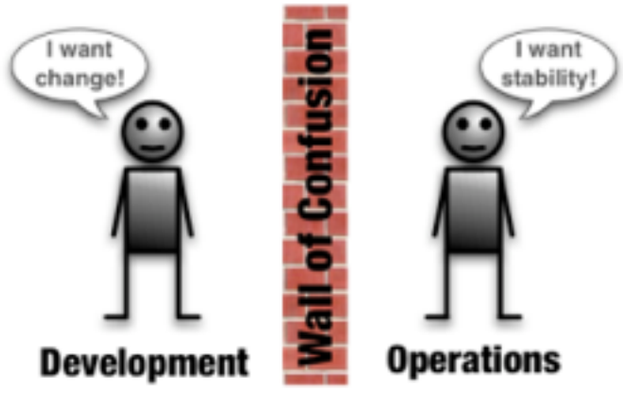


heig-vd
Haute Ecole d'Ingénierie et de Gestion
du Canton de Vaud

DevOps

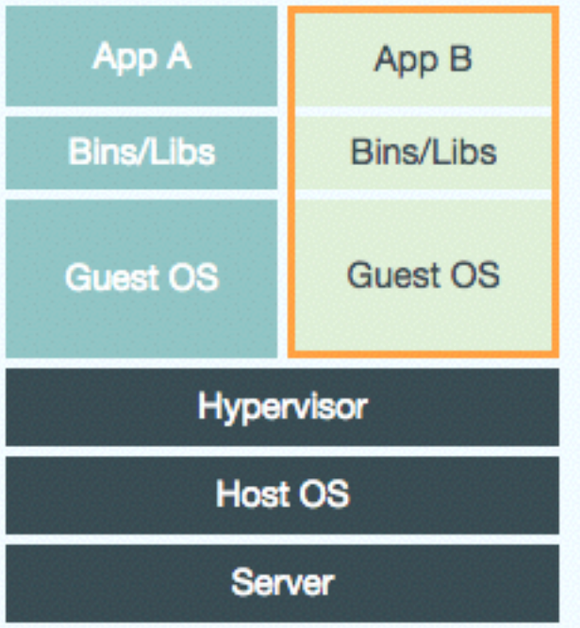
DevOps Movement





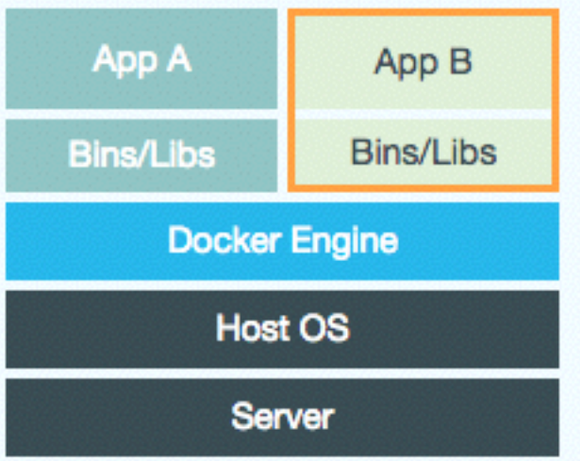
heig-vd

d'Ingénierie et de Gestion
e Vaud



Virtual Machines

Each virtualized application includes not only the application - which may be only 10s of MB - and the necessary binaries and libraries, but also an entire guest operating system - which may weigh 10s of GB.



Docker

The Docker Engine container comprises just the application and its dependencies. It runs as an isolated process in userspace on the host operating system, sharing the kernel with other containers. Thus, it enjoys the resource isolation and allocation benefits of VMs but is much more portable and efficient.

https://www.docker.com/whatisdocker/

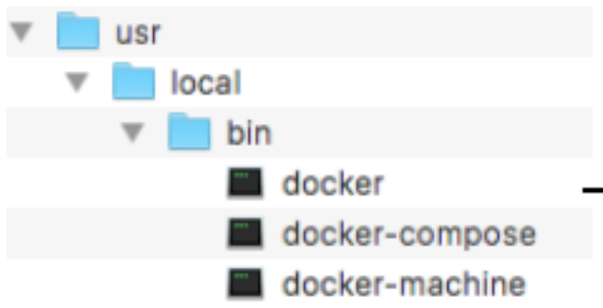
Installing Docker

HAUTE ÉCOLE
D'INGÉNIERIE ET DE GESTION
DU CANTON DE VAUD
www.heig-vd.ch

- Docker is based on the **Linux Containers** (LXC) technology. If you run a linux distribution on your laptop, you will use Docker natively. Just be careful with the version you are using.
- If you run **Windows** or **Mac OS X**, you have different options. Behind the scenes, you will need a **Linux VM**. The tools that you will install will hide this VM (more or less), but it will still be here.
- On the Docker website, you will be encouraged to install **Docker for Windows** or **Docker for Mac**. This might not be the best choice, and you might want to go for the (legacy) **Docker Toolbox** and **Docker Machine** toolset.
- If you run Windows, be aware that you cannot run **Docker for Windows** and **VirtualBox** at the same time. You will need to enable/disable **Hyper V** and reboot to switch between environments. Docker Toolbox does not have this issue.
- Personally, I still use **Docker Machine** and not Docker for Mac. Reasons: performance (file system), ability to have different environments, the Linux VM is less hidden (has its own IP address). In my demos / webcasts, you will see 192.168.99.100 and not localhost.

HAUTE ÉCOLE
D'INGÉNIERIE ET DE GESTION
DU CANTON DE VAUD
www.heig-vd.ch

Docker Toolbox installation




```
DOCKER_HOST=tcp://xxx.xxx.xxx.xxx:2376
DOCKER_MACHINE_NAME=default
DOCKER_TLS_VERIFY=1
DOCKER_CERT_PATH=$HOME/.docker/machine/machines/default
```

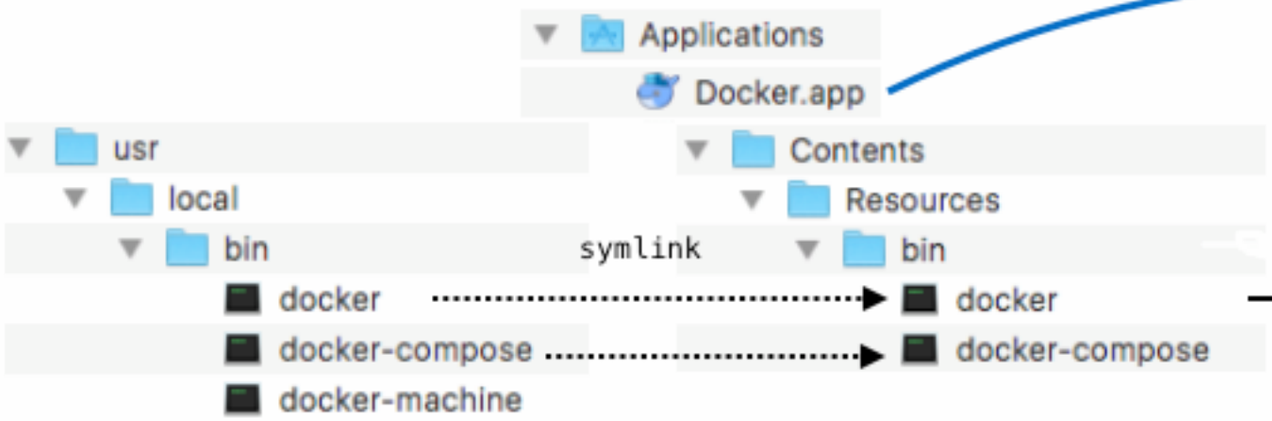
manages

default

IP xxx.xxx.xxx.xxx

Docker for Mac installation

manages




```
unset ${!DOCKER_*}
```

```
/var/run/docker.sock
```

docker.local

heig-vd
HAUTE ÉCOLE
D'INGÉNIERIE ET DE GESTION
DU CANTON DE VAUD
www.heig-vd.ch

Docker image

Docker container

Docker containers (1)

heig-vd
Haute Ecole d'Ingénierie et de Gestion
du Canton de Vaud

- You can think of a container as a **lightweight** virtual machine. Each container is isolated from the others and has its own IP address.
- Each container is created from a docker image (one could say that a container is a **running instance of an image**).
- There are **commands** to start, list, stop and delete containers.

```
# Start a container (more on this later)
$ docker run

# List running containers
$ docker ps

# List all containers
$ docker ps -a

# Delete a container
$ docker rm

# Display logs produced by a container
$ docker logs
```

Docker containers (2)

heig-vd
Haute Ecole d'Ingénierie et de Gestion
du Canton de Vaud

- With Docker, the philosophy is to have **one application-level service per container** (it is not a strict rule).
- With Docker, the philosophy is also to **run several (many) containers on the same machine**.
- If you think of a typical **Web Application infrastructure**, you would have one or more containers for the apache web server, one container for the database, one container for the reverse proxy, etc.
- With Docker, containers tend also to be **short-lived**. Each container has an **entry point**, i.e. a command that is executed at startup. When this command returns, the container is stopped (and will typically be removed).
- **If a container dies, it should not be a big deal**. Instead of trying to fix it, one will create a new one (from the same image).

Docker images (1)

heig-vd
Haute Ecole d'Ingénierie et de Gestion
du Canton de Vaud

- **A Docker image is a template**, which is used to create containers.
- Every image is **built from a base image** and adds its own configuration and content.
- With Vagrant, we use a file named Vagrantfile to configure and provision a Vagrant box. With Docker, we use a file name **Dockerfile** to create an image. The file contains statements (FROM, RUN, COPY, ADD, EXPOSE, CMD, VOLUME, etc.)
- Just like the community is sharing Vagrant boxes, **the community is sharing Docker images**. This happens on the Docker Hub registry (https://registry.hub.docker.com/).

Docker images (2)

heig-vd
Haute Ecole d'Ingénierie et de Gestion
du Canton de Vaud

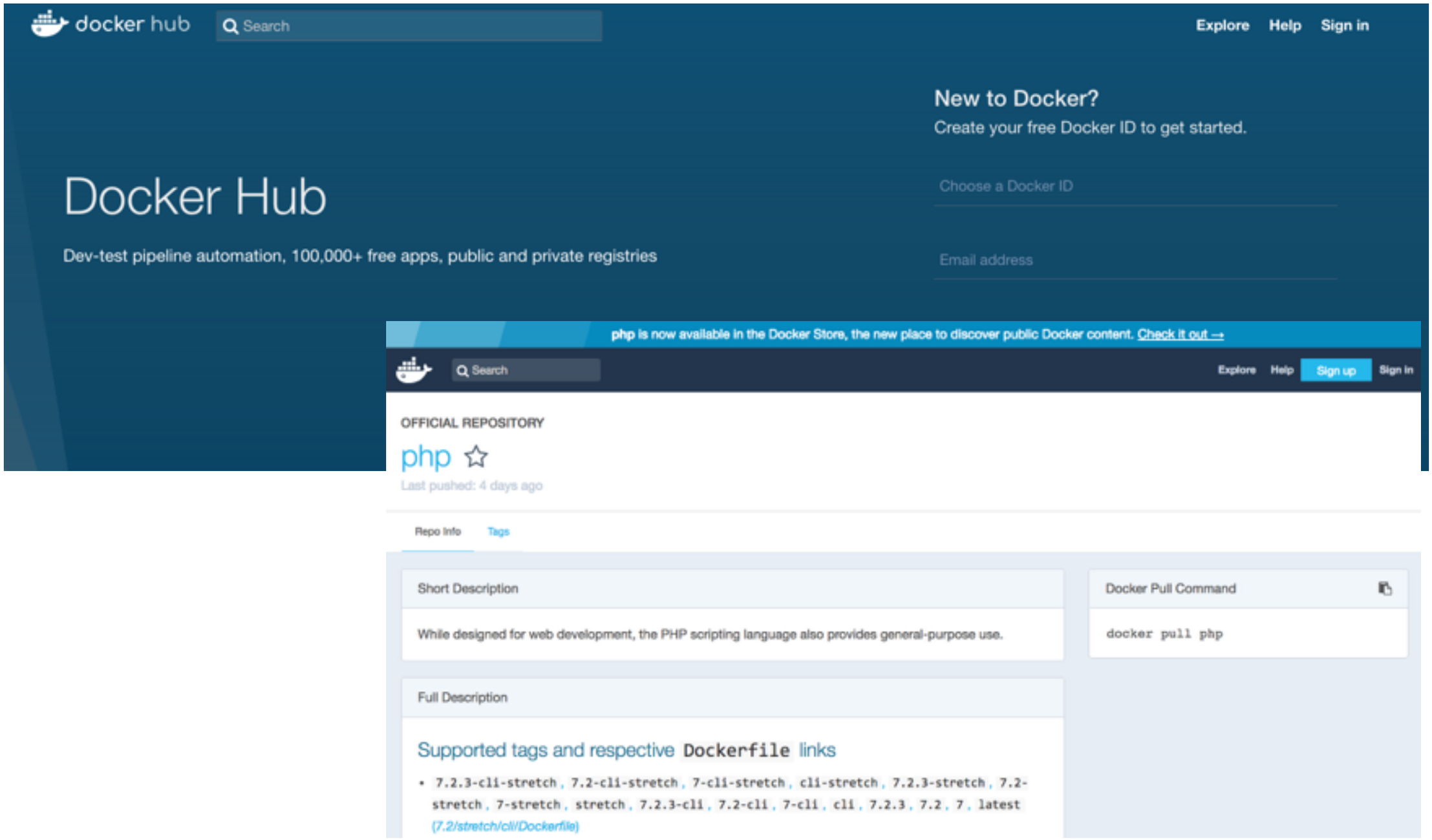
- Here is an example for a Dockerfile (used for first experiments, does not

```
# This image is based on another image

FROM dockerfile/nodejs:latest

# For information: who maintains this Dockerfile?

MAINTAINER Olivier Liechti

# When we create the image, we copy files from the host into
# the image file system. This is NOT a shared folder!

COPY file_system /opt/res/

# With RUN, we can execute commands when we create the image. Here,
# we install the PM2 process manager

RUN npm install -g pm2@0.12.9
```

```
# Create an image from this Dockerfile
$ docker build -t heigvd/res-demo .

# Execute /bin/bash in a new container, created from the image
$ docker run -i -t heigvd/res-demo /bin/bash
```

Docker Hub

HAUTE ÉCOLE
D'INGÉNIERIE ET DE GESTION
DU CANTON DE VAUD
www.heig-vd.ch



heig-vd
Haute Ecole d'Ingénierie et de Gestion
du Canton de Vaud

Host (your laptop running Windows or Mac OS X) 10.192.116.213

Linux VM (e.g. 192.168.99.100

docker0
bridge
172.17.0.1

172.17.0.6
/opt/res
Docker
container

172.17.0.2
/opt/res
Docker
container

172.17.0.4
/opt/res
Docker
container

172.17.0.3
/opt/res
Docker
container

Sniffing UDP traffic

heig-vd
Haute Ecole d'Ingénierie et de Gestion
du Canton de Vaud

Using tcpdump on the Vagrant box

```
$ sudo tcpdump -i docker0 udp -A
```

```
E..J..@...E.........q&..6.e{"timestamp":1427800010160,"temperature":null}
11:06:50.362733 IP 172.17.0.8.58225 > 239.255.22.5.9907: UDP, length 46
E..J..@...E.........q&..6.e{"timestamp":1427800010362,"temperature":null}
11:06:50.565816 IP 172.17.0.8.58225 > 239.255.22.5.9907: UDP, length 46
E..J..@...E.........q&..6.e{"timestamp":1427800010565,"temperature":null}
11:06:50.768966 IP 172.17.0.8.58225 > 239.255.22.5.9907: UDP, length 46
E..J..@...E.........q&..6.e{"timestamp":1427800010768,"temperature":null}
11:06:50.970691 IP 172.17.0.8.58225 > 239.255.22.5.9907: UDP, length 46
E..J..@...E.........q&..6.e{"timestamp":1427800010970,"temperature":null}
11:06:51.172537 IP 172.17.0.8.58225 > 239.255.22.5.9907: UDP, length 46
E..J..@...E.........q&..6.e{"timestamp":1427800011172,"temperature":null}
11:06:51.374546 IP 172.17.0.8.58225 > 239.255.22.5.9907: UDP, length 46
E..J..@...E.........q&..6.e{"timestamp":1427800011374,"temperature":null}
11:06:51.578663 IP 172.17.0.8.58225 > 239.255.22.5.9907: UDP, length 46
E..J..@...E.........q&..6.e{"timestamp":1427800011578,"temperature":null}
```

Demo 1

HAUTE ÉCOLE
D'INGÉNIERIE ET DE GESTION
DU CANTON DE VAUD
www.heig-vd.ch

https://github.com/SoftEng-HEIGVD/Teaching-Docker-SimpleJavaServer

Demo 2

HAUTE ÉCOLE
D'INGÉNIERIE ET DE GESTION
DU CANTON DE VAUD
www.heig-vd.ch

https://github.com/SoftEng-HEIGVD/Teaching-HEIGVD-RES-2018-Labo-02

Demo 3

HAUTE ÉCOLE
D'INGÉNIERIE ET DE GESTION
DU CANTON DE VAUD
www.heig-vd.ch

https://github.com/SoftEng-HEIGVD/Teaching-Docker-UDP-sensors

Example: **06-PresenceApplication**