



Introduction to docker



Plan

- C'est quoi Docker
- Containers vs Virtual Machines
- Aperçu de la plateforme et terminologie
 - Docker engine
 - Images
 - Containers
 - Registry
- Repositories
- Docker Hub
- Outils Docker
- Introduction aux images
- Commencer avec les containers



C'est quoi Docker

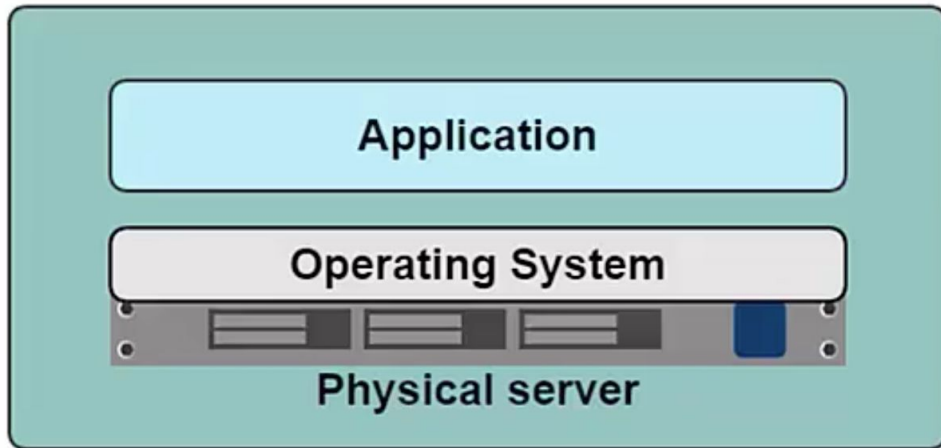
Docker est une plateforme pour développer, livrer et exécuter des applications en utilisant la virtualisation.

Il y a plusieurs composants ou outils disponibles :

Docker Engine	Docker Swarm
Docker Hub	Docker Compose
Docker Machine	Kitematic

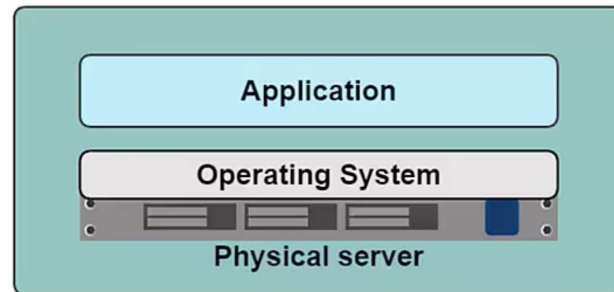
Une leçon d'histoire

Il y a longtemps, une seule application tournait sur un seul serveur physique.



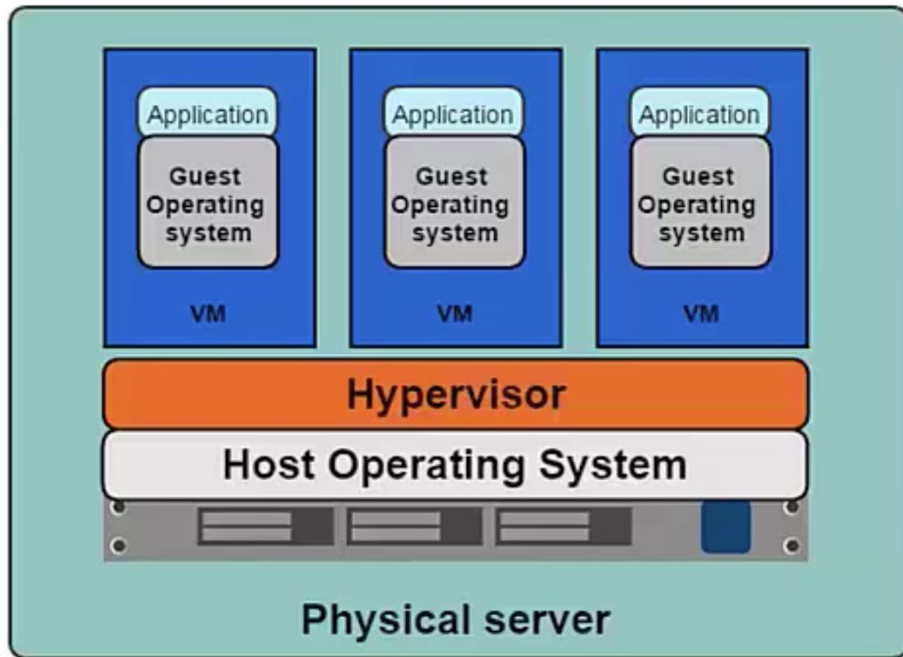
Problème rencontrés

- Déploiement lent
- Coût élevé
- Ressources perdues
- Difficulté à faire évoluer
- Difficulté à migrer
- Lien fort avec les fournisseurs



La virtualisation

- Un serveur physique contient plusieurs applications
- Chaque application tourne dans une machine virtuelle



Avantages des VM

- Meilleurs gestion des ressources
 - Un serveur physique est divisé en plusieurs machines virtuelles
- Évolution facilité
- VM dans le cloud
 - Souplesse et rapidité
 - Ne payer que ce que l'on consomme



Limites des VM

- Chaque VM requiert pour ses besoins :
 - Allocation CPU
 - Espace disque
 - RAM
 - Un OS hôte complet
- Plus il y a de VM, plus il y a besoin de ressources
- L'OS hôte consomme des ressources
- Le portage des applications n'est pas garanti

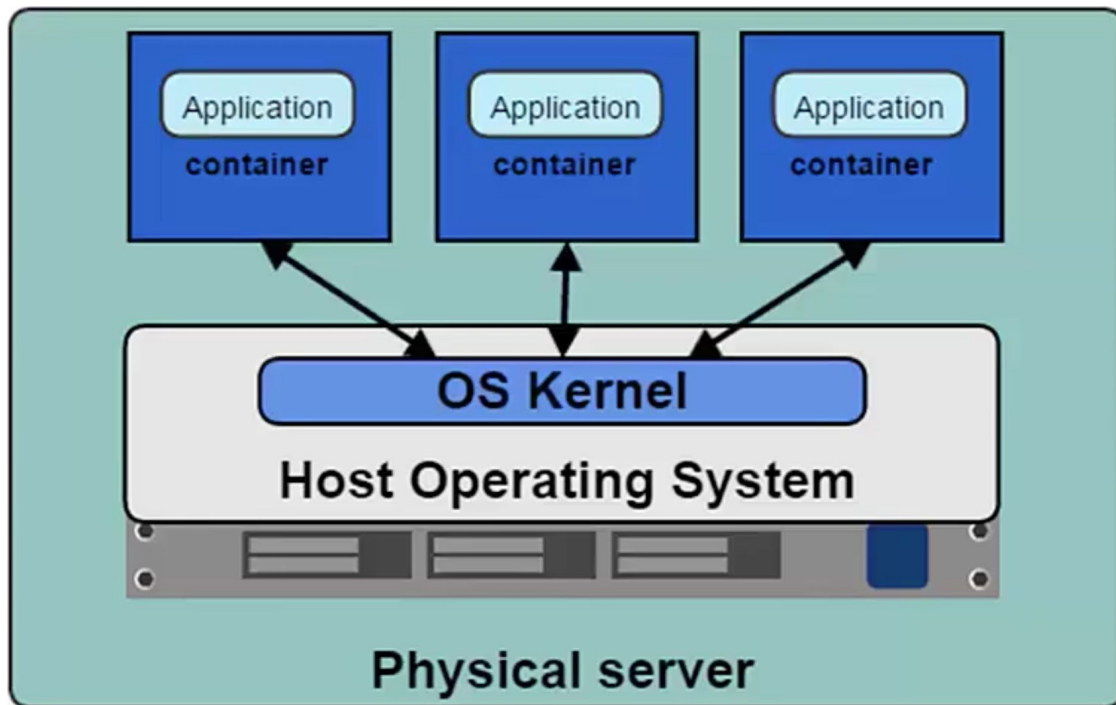


Les Containers

Les Containers utilisent le noyau de l'OS sur lequel ils sont, pour faire tourner de multiple instances hôtes

- Chaque instance est appelée un container
- Chaque container a son/ses propre(s) :
 - Système de fichiers
 - Processus
 - Appels mémoire
 - Matériels
 - Ports réseau

Les Containers



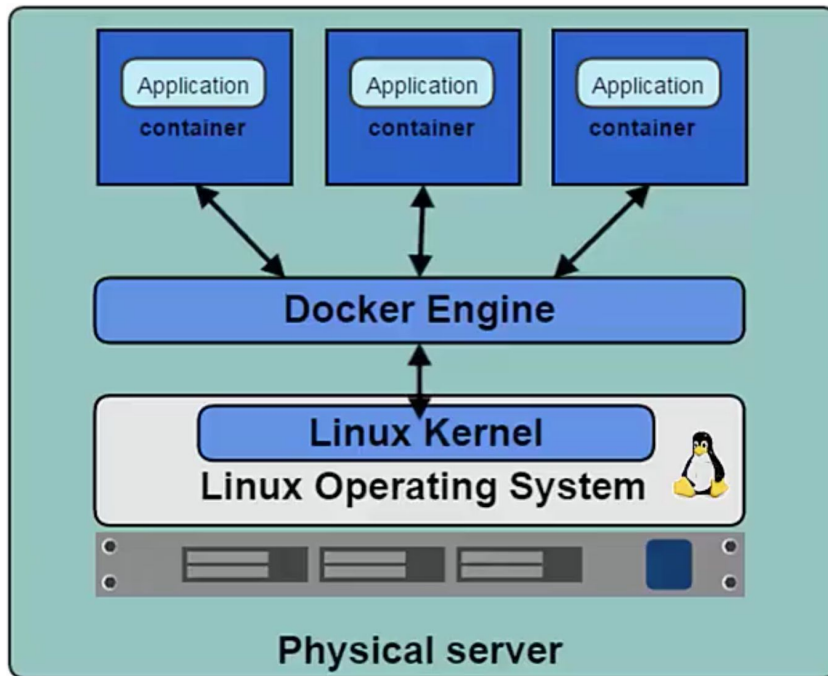
Containers vs VMs

- Les containers sont beaucoup plus légers
- Pas besoin d'installer d'OS
- Moins de CPU, RAM et disque utilisés
- Il est possible de mettre beaucoup plus de containers que de VMs par serveur
- Extrême portabilité

Docker, concepts et terminologie

Docker et le noyau Linux

- Docker Engine est le programme qui permet la construction, le déploiement et l'exécution du container
- Docker Engine utilise les namespaces du noyau Linux ainsi que les groupes de contrôle
- Les namespaces permet l'isolation des espaces de travail





Installation de docker

- Suivez les instructions sur <https://docs.docker.com/installation> pour installer la dernière version de docker sur votre machine
- Exécutez hello-world container pour tester votre installation :
 - ***sudo docker run hello-world***
- Si besoin, sortez de votre terminal pour que cela soit pris en compte
- Maintenant vous pouvez entrer
 - ***docker run hello-world***
 - notez les différences par rapport à la première fois que vous avez lancé la commande



Ma première commande

```
dm-fbn:~ fbaillon$ docker run hello-world
Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world

c04b14da8d14: Pull complete
Digest: sha256:0256e8a36e2070f7bf2d0b0763dbabdd67798512411de4cdcf9431a1feb60fd9
Status: Downloaded newer image for hello-world:latest

Hello from Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
 1. The Docker client contacted the Docker daemon.
 2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
 3. The Docker daemon created a new container from that image which runs the
    executable that produces the output you are currently reading.
 4. The Docker daemon streamed that output to the Docker client, which sent it
    to your terminal.

To try something more ambitious, you can run an Ubuntu container with:
$ docker run -it ubuntu bash

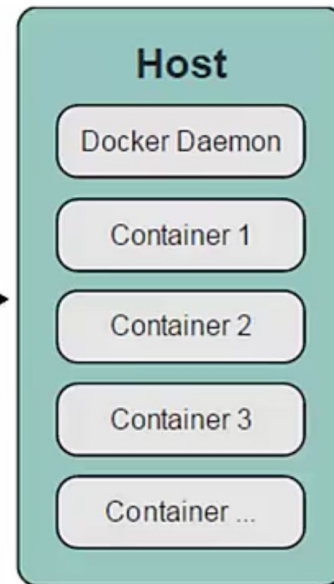
Share images, automate workflows, and more with a free Docker Hub account:
https://hub.docker.com

For more examples and ideas, visit:
https://docs.docker.com/engine/userguide/
```

Client et Daemon Docker

- Architecture client/serveur
- Le client prend les données en entrée et les transmet au Daemon
- Celui-ci les construit, les exécute et les distribue aux containers
- Client et Daemon peuvent tourner ou non sur le même hôte
- Kitematic est un GUI pour docker

Client



Vérifier Client et Daemon

- Run
 - ***docker version***

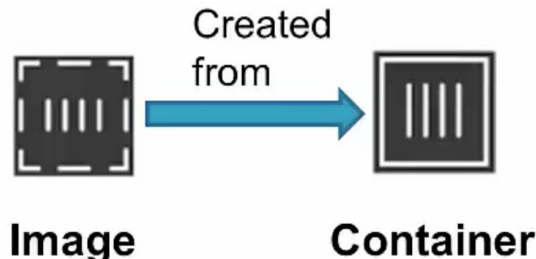
```
[dm-fbn:~ fbaillon$ docker version
Client:
 Version:           1.12.0
 API version:       1.24
 Go version:        go1.6.3
 Git commit:        8eab29e
 Built:             Thu Jul 28 23:54:00 2016
 OS/Arch:           darwin/amd64

Server:
 Version:           1.12.2-rc1
 API version:       1.24
 Go version:        go1.6.3
 Git commit:        45bed2c
 Built:             Tue Sep 27 23:38:15 2016
 OS/Arch:           linux/amd64
 Experimental:      true
```

Container et images

- Images

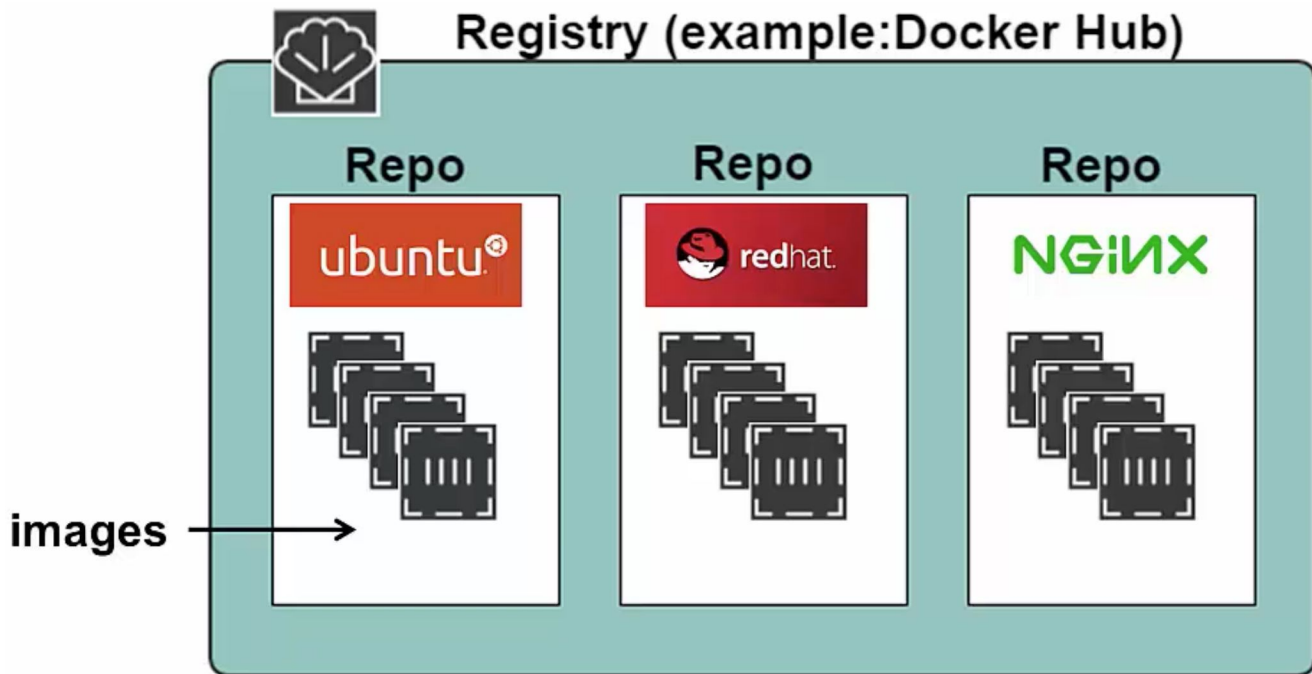
- Modèle en lecture seule qui permet de créer des containers
- Créés par vous ou par d'autres utilisateurs de docker
- Stockées dans docker hub ou localement sur votre machine



- Containers

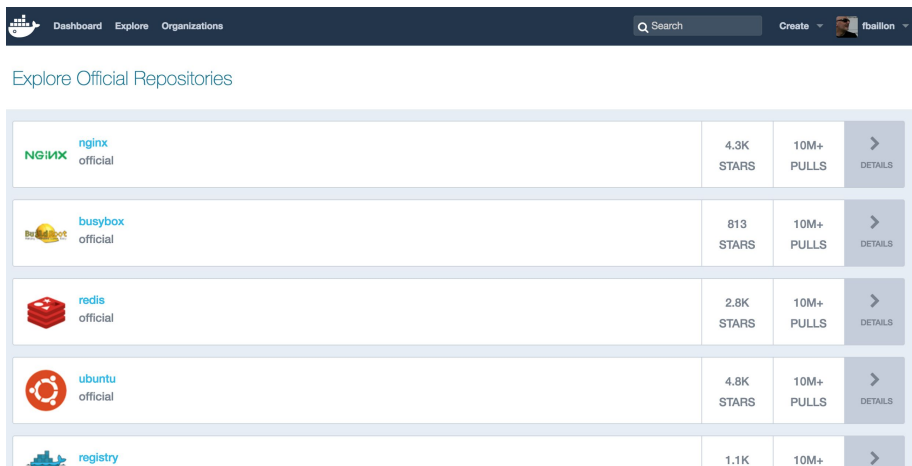
- “Entité” d’une application qui se suffit à elle-même
- Contient tout ce qui est nécessaire pour faire tourner l’application
- Basés sur une ou plusieurs images






Enregistrements et dépôts



Docker hub

- Docker hub est un dépôt public qui contient un très grand nombre d'images pour tous les usages

A screenshot of the Docker Hub website. The top navigation bar includes 'Dashboard', 'Explore', and 'Organizations'. A search bar is on the right, followed by a 'Create' button and a user profile for 'fbailion'. Below the navigation bar, the text 'Explore Official Repositories' is displayed. The main content area shows a list of official Docker images in a table format.

Repository	Stars	Pulls	Details
 nginx official	4.3K STARS	10M+ PULLS	DETAILS
 busybox official	813 STARS	10M+ PULLS	DETAILS
 redis official	2.8K STARS	10M+ PULLS	DETAILS
 ubuntu official	4.8K STARS	10M+ PULLS	DETAILS
 registry	1.1K	10M+	DETAILS



Docker orchestration

- 3 outils pour orchestrer la distribution d'applications avec docker
 - Docker Machine
 - Docker Swarm
 - Docker Compose


Avantages de docker

- Séparation des préoccupations
 - Les développeurs travaillent sur leurs applications
 - Les sys admins sur le déploiement
- Cycle de développement rapide
- Portabilité des applications
 - Développées dans un environnement, distribuées dans un autre
- Monté en charge
 - Il est très facile de gérer la montée en charge
- Plus d'applications peuvent tourner sur un serveur

Introduction aux images docker

Images sur docker hub

- Beaucoup d'images sont disponibles
- Les images sont dans différents dépôts




Dashboard

Explore

Organizations


Q tomcat

Create

 fballion

Repositories (2556)

All



tomcat

official

951


STARS

5M+

PULLS

>

DETAILS



cloudesire/tomcat

public | automated build

9


STARS

7.1K

PULLS

>

DETAILS



andreptb/tomcat

public | automated build

6


STARS

2.3K

PULLS

>

DETAILS



fbrx/tomcat

public | automated build

2

STARS

1.6K

PULLS

>

DETAILS



Créez votre compte docker hub

- Allez sur <https://hub.docker.com/account/signup/> pour créer votre compte si vous n'en avez pas encore un
 - Il n'y a pas besoin de carte de crédit ;)
- Activez votre compte avec l'email reçu
- Naviguez sur le site dans les dépôts
- Cherchez des images de vos outils, langages, serveurs favoris...

Afficher les images locales

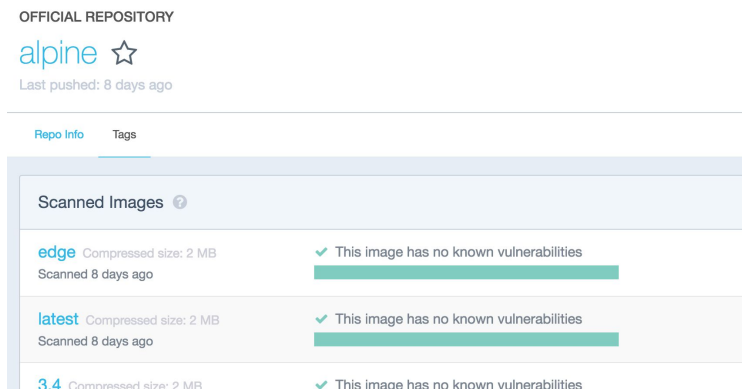
- run ***docker images***
- C'est localement que docker engine va essayer de trouver une image pour construire le container
- S'il ne trouve pas, il essaiera de trouver l'image sur docker hub

```
dm-fbn:~ fbaillon$ docker images
```

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
phpmyadmin/phpmyadmin	latest	995d529beadc	4 days ago	116.2 MB
wordpress	latest	88e010fa60ea	7 days ago	419.4 MB
mariadb	latest	bae1077764d3	7 days ago	389.8 MB
hello-world	latest	c54a2cc56cbb	3 months ago	1.848 kB
docker/whalesay	latest	6b362a9f73eb	16 months ago	247 MB

Le tag sur les images

- Les images sont identifiées par des tags
- Une même image peut avoir plusieurs tags
- Le tag par défaut est **latest**
- Dans docker hub, il y a le détail de chaque tag



Commencer avec les containers

Créer un container

- Utiliser la commande **run command**
- Syntax
 - ***docker run [options] image [command] [args]***
- image est définie par ***repository:tag***

Examples

```
docker run ubuntu:14.04 echo "Hello World"
```

```
docker run ubuntu ps ax
```



Exécuter un container simple

- Dans votre terminal, tapez la commande suivante :
 - ***docker run ubuntu echo "Hello world"***
- Observez ce que vous obtenez
- Puis tapez :
 - ***docker run ubuntu ps ax***
- Observez ce que vous obtenez
- Puis tapez :
 - ***docker run ubuntu***
- Observez ce que vous obtenez
- Vous pouvez voir que la seconde commande est presque instantanée car maintenant l'image ubuntu:16.04 est locale

Container avec un terminal

- Utilisez les options `-i` et `-t` lors de l'exécution
 - L'option `-i` demande la connexion à STDIN dans le container
 - et l'option `-t` demande un pseudo terminal
- Vous devez spécifier un processus de terminal (par exemple `/bin/bash`)

Example

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

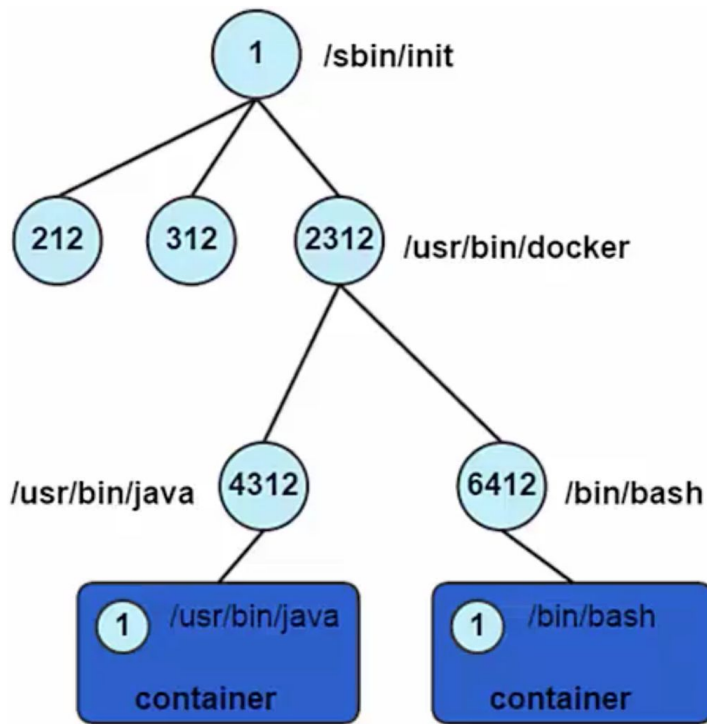


Terminal dans un container

- Créez une image ubuntu:14.04, connectez STDIN et lancez un terminal:
 - ***docker run -it ubuntu:14.04 /bin/bash***
- Dans le container, créez un nouvel utilisateur
 - ***adduser new***
- Puis ajoutez le au groupe “sudo”
 - ***adduser new sudo***
- Faites des manipulations avec lui
- Sortez du container, puis relancez la même commande pour créer un container, l'utilisateur n'est plus là

Container processus

- Le container s'exécute tant que le processus que vous avez lancé avec **run** est actif
- Le numéro du PID de la commande lancée dans votre container porte toujours le numéro 1



ID d'un container

- Il est possible d'accéder à un container par son nom ou par son ID
- L'ID est long, la version courte est plus utile
- Le nom et l'ID peuvent être obtenus en lançant la commande ***docker ps*** qui donne la liste des containers
- L'ID long est obtenu en inspectant le container

Trouver un container

- Utiliser ***docker ps*** pour lister les containers actifs
- L'option ***-a*** permet de les lister tous, y compris ceux qui sont arrêtés

~ — root@ec4158666929: / — -bash

~ — root@c150908c41ca: / — docker ru

dm-fbn:~ fbaillon\$ docker ps -a

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
c150908c41ca	ubuntu:14.04	"/bin/bash"	34 seconds ago	Up 33 seconds		elated_goldwasser
ec4158666929	ubuntu:14.04	"/bin/bash"	19 minutes ago	Exited (0) 18 minutes ago		elated_babbage
a147ccb69b41	ubuntu:14.04	"/bin/bash"	26 minutes ago	Exited (0) 23 minutes ago		hungry_lovelace
f2862c842a7d	ubuntu:14.04	"/bin/bash"	18 hours ago	Exited (0) 18 hours ago		zen_kilby
ee5971a08bac	ubuntu:14.04	"ps ax"	19 hours ago	Exited (0) 19 hours ago		desperate_darwin
4ca0a4d439fb	ubuntu:14.04	"echo 'Hello world'"	19 hours ago	Exited (0) 19 hours ago		evil_gates
237468a5a232	ubuntu:14.04	"echo 'Hello world'"	19 hours ago	Exited (0) 19 hours ago		cranky_leakey
2d7f9b9b005a	hello-world	"/hello"	23 hours ago	Exited (0) 23 hours ago		gloomy_borg
d82e8ca3fb30	phpmyadmin/phpmyadmin	"/run.sh phpmyadmin"	3 days ago	Exited (0) 3 days ago		mypma
a37cf678401a	mariadb	"docker-entrypoint.sh"	3 days ago	Exited (0) 3 days ago		mymariadb
c49a8ea40ceb	mariadb	"docker-entrypoint.sh"	3 days ago	Exited (0) 3 days ago		angry_noether
59aec58368e5	docker/whalesay	"cowsay boo"	3 days ago	Exited (0) 3 days ago		nostalgic_payne



Exécution en mode détaché

- Aussi appelé en tâche de fond ou daemon
- Il faut utiliser l'option **-d**
- Pour observer la sortie, utilisez ***docker logs [container ID]***



Lister vos containers

- Exécutez :
 - ***docker run -d ubuntu ping -c 100 127.0.0.1***
- Afficher la liste de vos containers en utilisant
 - ***docker ps***
- Vérifier que le container créé est actif
- Afficher la liste de tous vos containers en utilisant
 - ***docker ps -a***
- Vérifier que l'ensemble des containers créés depuis le début est présente

Documentation docker



Documentation commandes

- <https://docs.docker.com/engine/reference/commandline/>
- ***docker run***
 - <https://docs.docker.com/engine/reference/commandline/run/>
- ***docker images***
 - <https://docs.docker.com/engine/reference/commandline/images/>
- ***docker ps***
 - <https://docs.docker.com/engine/reference/commandline/ps/>