



DOCKER

Une introduction de B. Adanlessossi

PRINCIPES ET PHILOSOPHIE



MACHINE VIRTUELLE VS
CONTENEUR



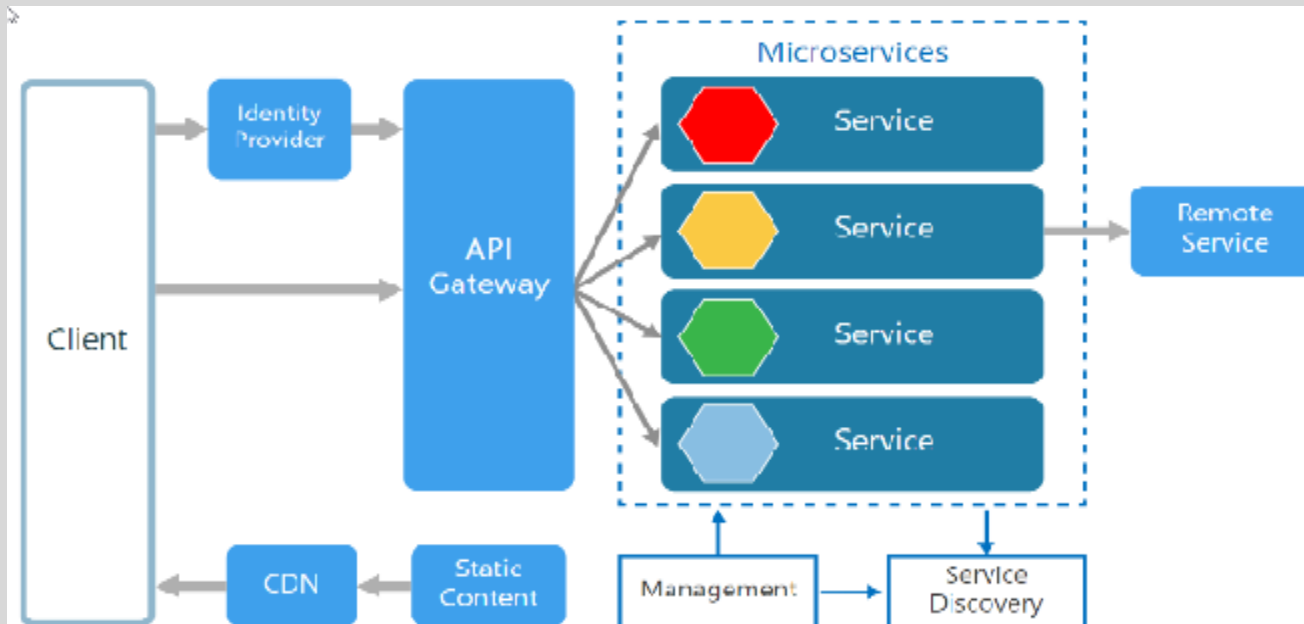
AVANTAGES ET
DESAVANTAGES



COMMENT ET QUAND
UTILIZER LES CONTENEURS.

Les Microservices - rappel

- Dans un précédent workshop, nous avons abondamment parlé des microservices
- Ce sont des services déployés de manière autonome l'un de l'autre mais représentant une plus grande plus value



Conséquences des microservices

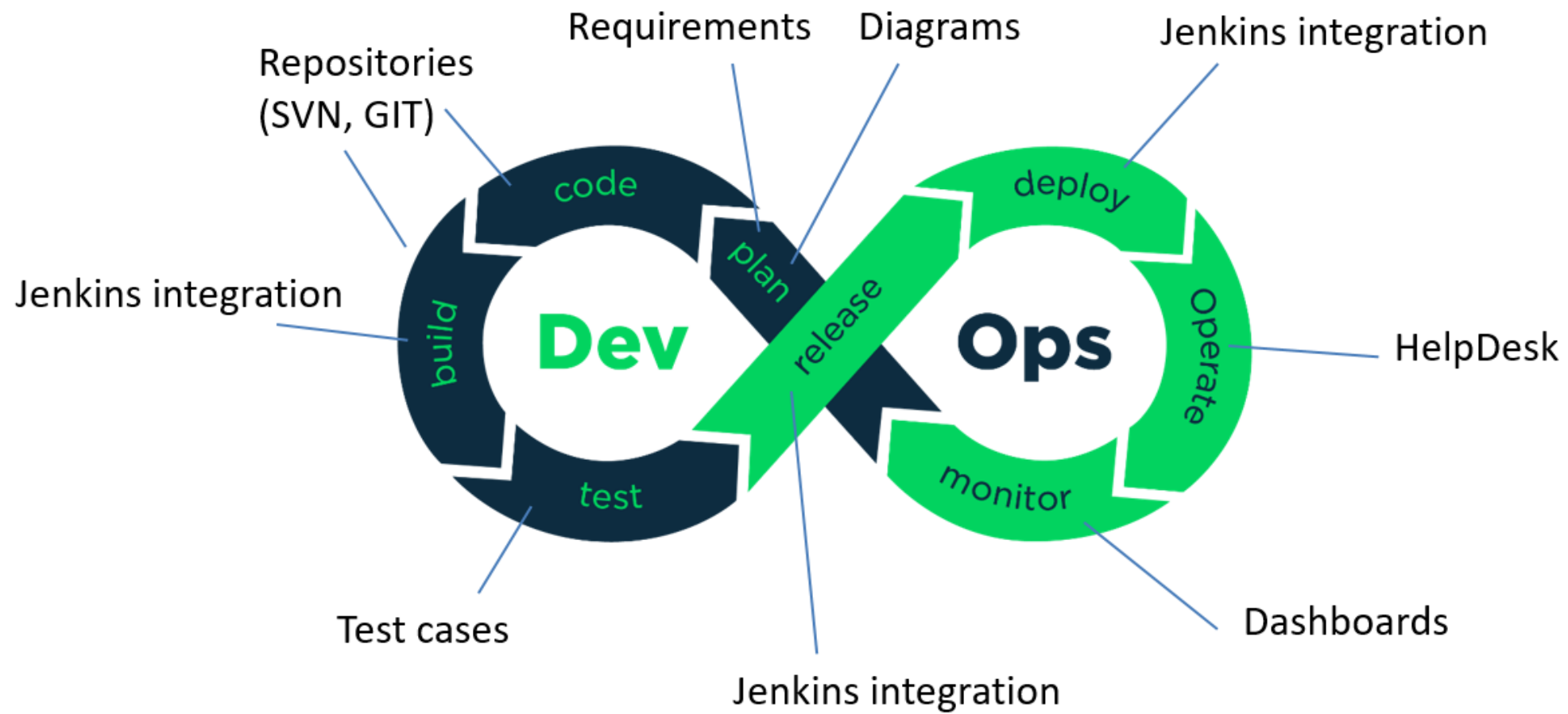
- Devant le succès de l'architecture Microservices, il s'est avéré qu'il fallait trouver une nouvelle manière de développer et de déployer les applications.
- Chaque microservice étant isolé de l'autre, ils peuvent également être déployés séparément, ce qui permet à des équipes différentes de travailler sur des services différents
- Il faudra adopter un nouveau cycle de vie de développement des applications, DevOps
- Il faudra aussi adopter une nouvelle culture de développement des applications, justement basée sur DevOps
- Qu'est ce que DevOps?

Grandes Responsabilités!

- Le développeur de logiciel n'est plus un simple développeur!
- Il est maintenant responsable depuis la spécification jusqu'au déploiement de l'application en production.
- Il doit collaborer avec les utilisateurs métiers, la planification, les tests et l'assurance qualité
- Il doit également être capable de prototyper rapidement, utiliser toute base de données disponible, modéliser la base de données et déployer les données en production.
- Il est également responsable de la bonne marche de la construction de l'application (Jenkins), depuis l'environnement de développement, en passant par l'environnement de test jusqu'à la production!
- Il est devenu à la fois Développeur et Responsable des Opérations!

DevOps - Définition

- C'est une combinaison de pratiques et mentalités de développement de logiciel avec d'autres fonctions au sein de l'organization.
- DevOps met un accent très fort sur la notion de responsabilité partagée entre toutes les équipes, dans le cycle de vie de développement de logiciel.
- Le terme DevOps est souvent associée à Développement (Dev) et IT Operations (Ops) mais peut également être étendues à d'autres fonctions comme la Sécurité (DevSecOps), au QA, Base de données ou reseau.

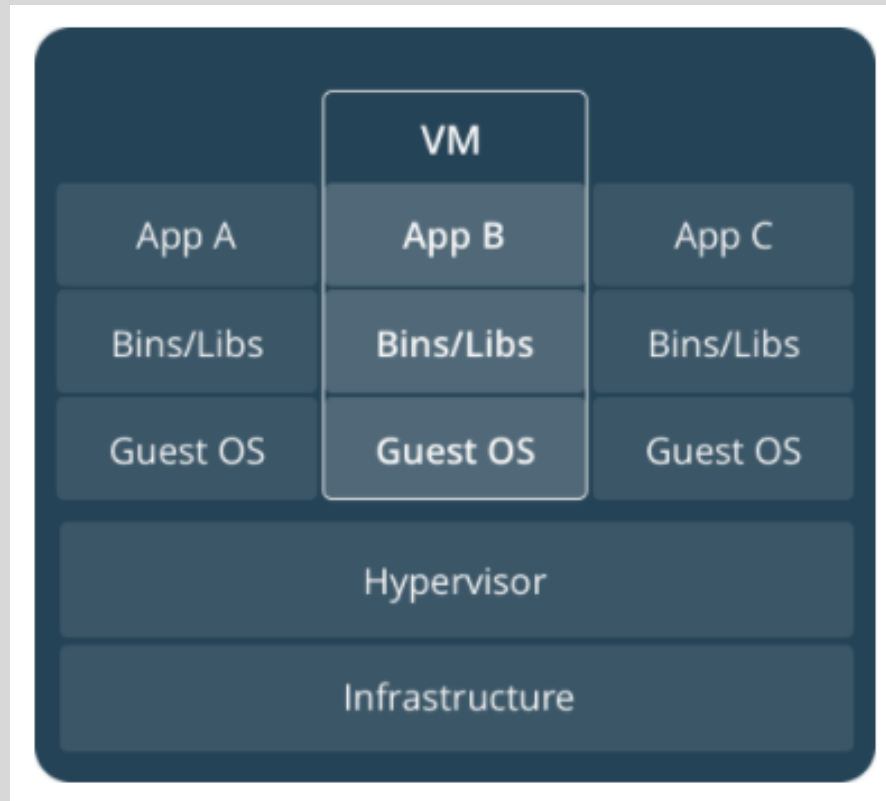


Comment docker peut-il aider?

- Si nous voulons suivre la vélocité dans le développement des logiciels, il va falloir considérer les machines virtuelles actuellement utilisées dans le développement et le déploiement des logiciels.
- Ces machines, en dehors du coût exorbitant, prennent du temps à démarrer ou à migrer.

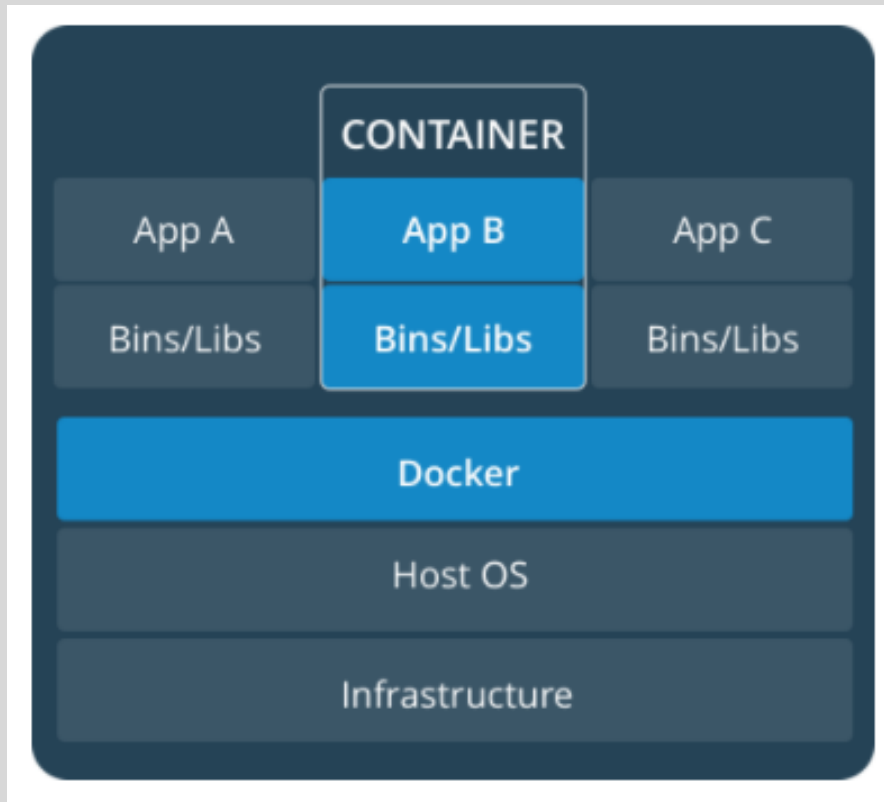
VM contre Docker

- Une Machine Virtuelle (VM), exécute tout un système d'exploitation avec accès virtuel aux ressources de la machine host, à travers un hypervisor. En general, une VM implique plus de ressources qu'effectivement consommées par notre logiciel.



VM contre Docker (2)

- Un conteneur Docker s'exécute nativement sur linux et partage le noyau de la machine avec les autres conteneurs. Il exécute des processus discrets qui ne prennent pas plus de mémoire que les autres executables, ce qui rend docker très léger.

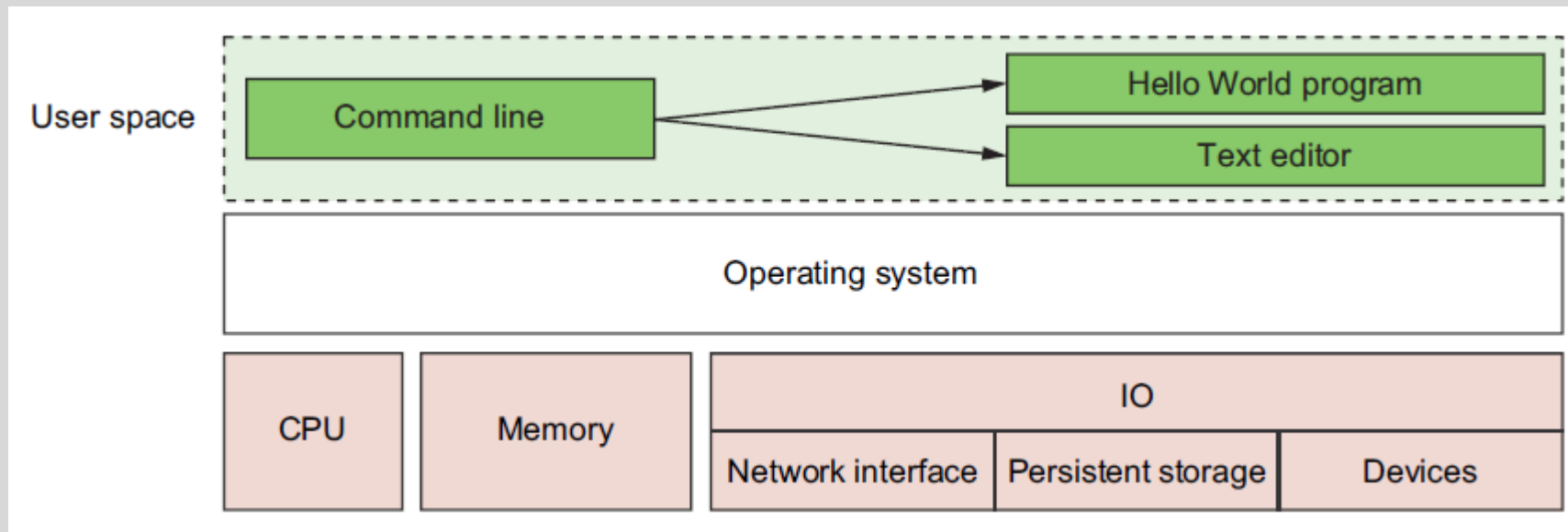


Qu'est ce que docker?

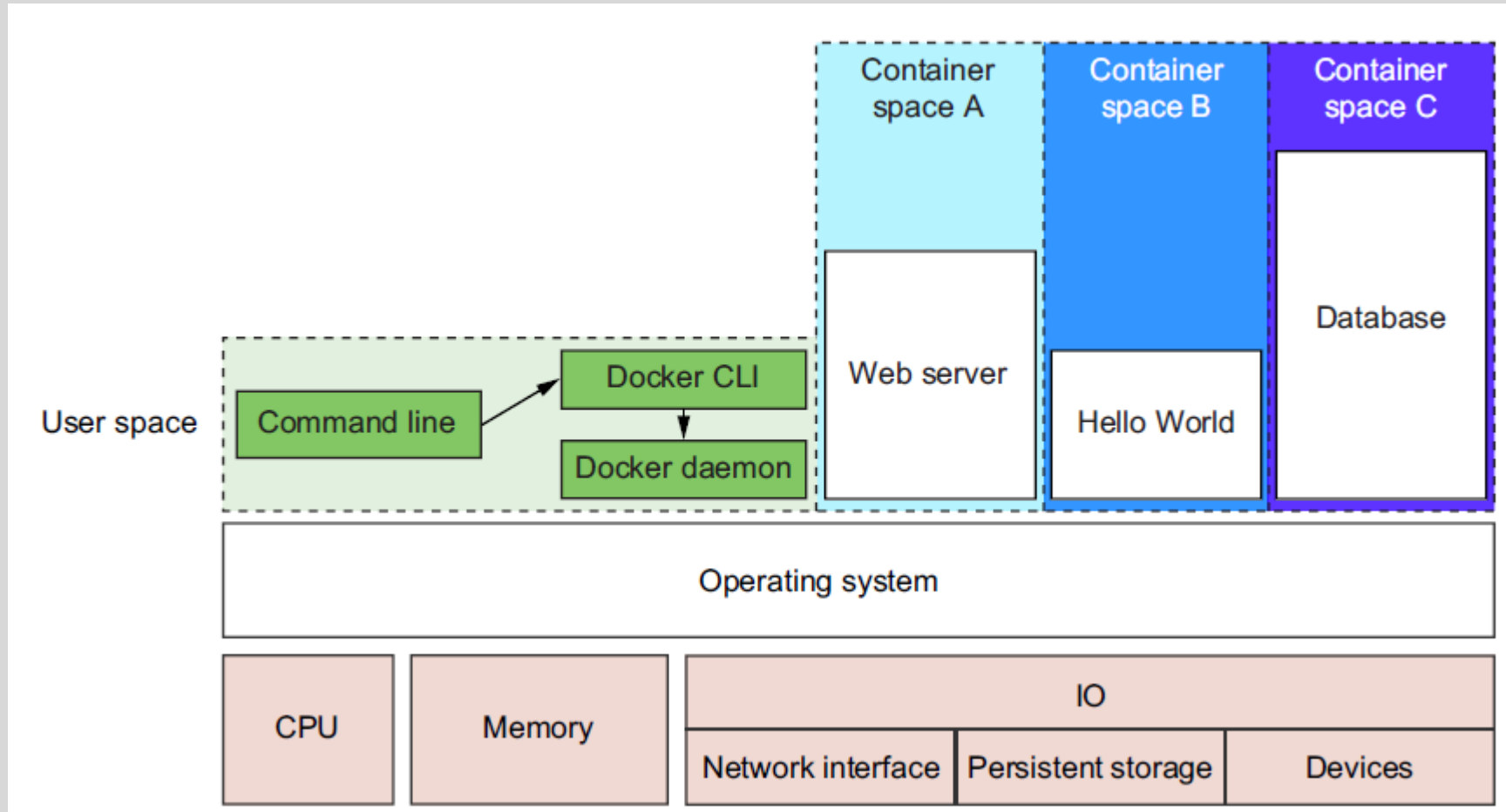
- Docker est un programme de ligne de commande,
- Un service qui tourne en arrière plan, et
- Un nombre de services distants qui prennent une approche logistique pour résoudre des problèmes communs aux logiciels, en simplifiant l'expérience d'installation, d'exécution, de publication et de retraite d'un logiciel.
- Il accomplit tout ceci en se basant sur une technologie linux, appelée "CONTAINER".
- Historiquement, le système d'exploitation linux utilise le terme "jail" pour désigner un programme en exécution isolé, afin que ce programme n'accède pas à d'autres ressources.
- Le terme container s'est répandu pour designer un programme exécutable isolé, accédant à des ressources et processus définis.

Exécution d'un logiciel en isolation

- Sur un ordinateur classique, un programme de ligne de commande s'exécute au-dessus du système d'exploitation



Exécution d'un logiciel en isolation (2)

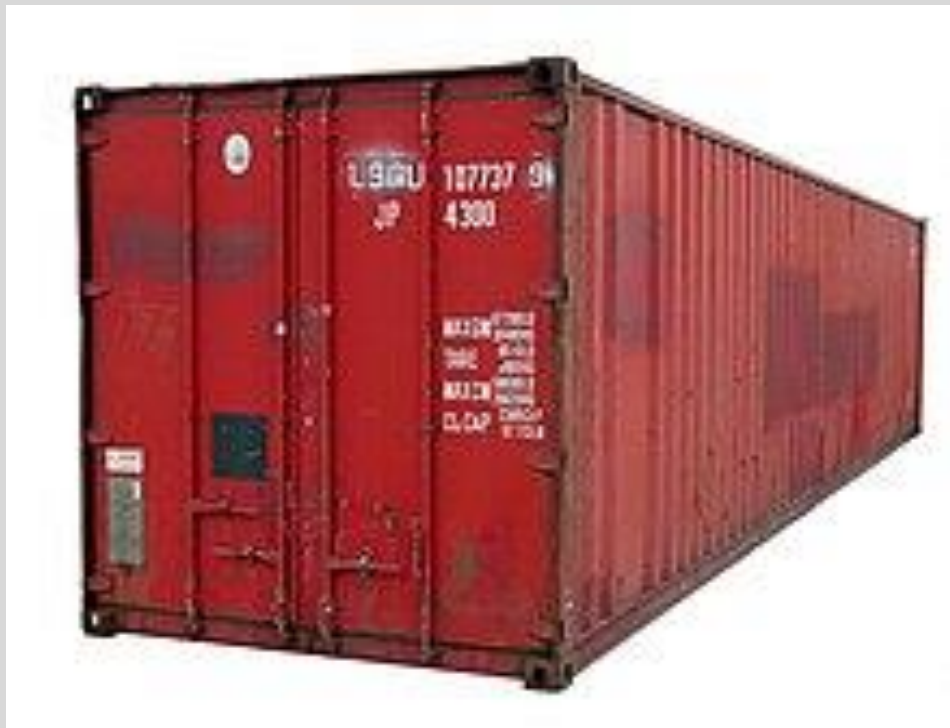


Les conteneurs



Conteneur

- Nous pouvons comparer le conteneur docker à un conteneur physique utilisé sur les bateaux pour transporter les marchandises.
- Dans la boîte, vous entreposez votre logiciel et toutes ses dépendances.



Conteneur

- Exactement comme peuvent les faire des grues, des camions, des trains et des bateaux avec les vrais conteneurs de marchandise, docker peut executer, copier et distribuer les conteneurs avec facilité.



Conteneur

- Et pour compléter l'analogie avec les conteneurs de marchandise, le conteneur docker permet d'empaqueter et de distribuer les logiciels.
- Le composant contenu dans le conteneur est appelée, **image**.

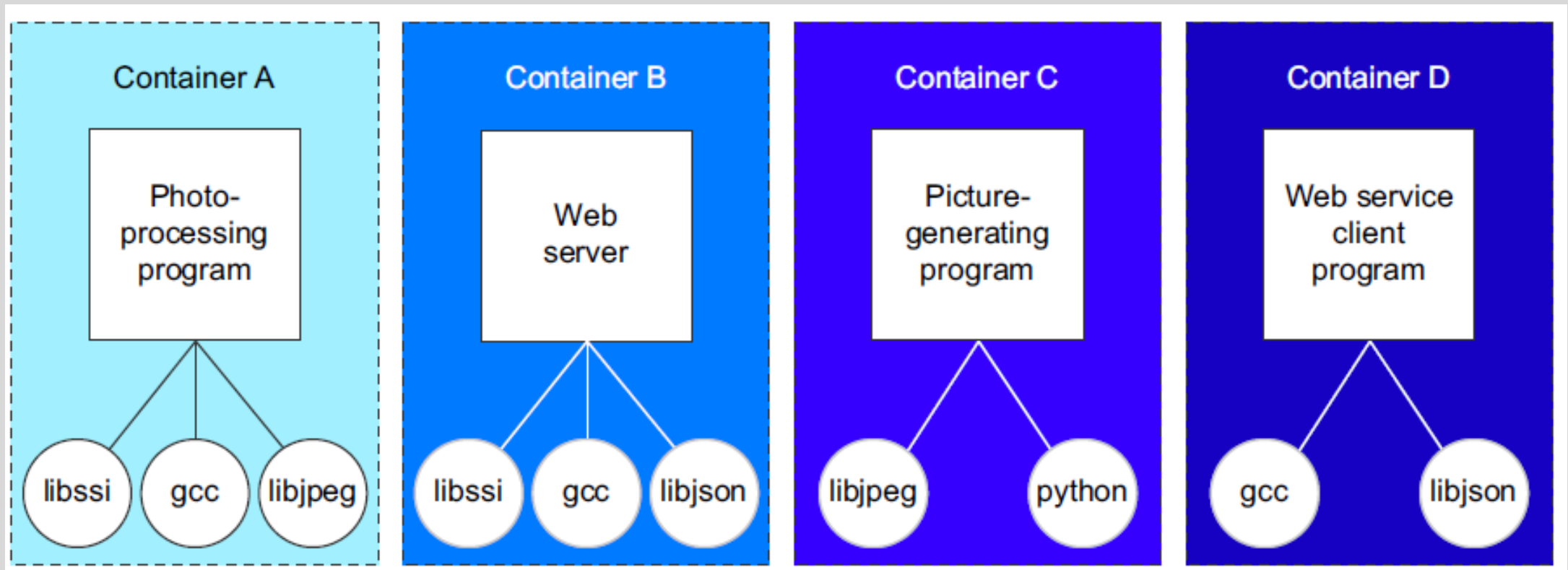


Le monde avant docker

- Utiliser un logiciel de nos jours est compliqué.
- Avant d'installer un logiciel, il faut considerer le système d'exploitation et le lieu de l'installation.
- Certaines installations varient d'un système d'exploitation à l'autre.
- Quelques fois le logiciel n'est pas compatible avec d'autres logiciels déjà installés sur le système d'exploitation.
- Quand il faut faire une mise à jour, de nombreux bugs se fauillent à l'horizon.
- En cas de désinstallation, le logiciel est-il réellement désinstallé?
- Peut-on supprimer les vieilles dépendances?
- La vérité est que plus vous installez de logiciels, plus il devient complexe de les gérer.

Organisation et méthode

- Docker permet de mettre de l'ordre en isolant tout logiciel à l'aide de conteneurs et d'images.



Portabilité des logiciels

- La portabilité entre les différents systems d'exploitation est aussi un problème majeur.
- Bien qu'il soit possible de trouver une compatibilité entre le système d'exploitation linux et MacOS, sous windows les choses s'avèrent quelques fois très compliqués.
- Docker s'exécute nativement en linux et est livré avec une seule et même machine virtuelle pour MacOS et Windows.
- Cette convergence vers linux veut dire qu'un logiciel a besoin d'être développé une seule fois avec les dépendances nécessaires.
- Cette avancée de la portabilité (linux) permet d'exécuter le même logiciel, exactement le même logiciel sur n'importe quel système d'exploitation.

En résumé

- Isolation: les fichiers binaires et les dépendances sont emballés ensemble
 - Il n'est plus question de "ça a marché sur ma machine mais pas en production!"
- Parité entre les environnements de développement, Test, Recette et Production
- Les différentes équipes de développements peuvent délivrer plus vite
- Vous pouvez exécuter une image docker inchangée aussi bien sur votre laptop, sur une machine virtuelle ou dans le cloud.
- Docker utilise une capacité du noyau linux pour isoler le système d'exploitation



UTILISATION DE DOCKER



POURQUOI DOCKER



INSTALLATION ET
PRISE EN MAIN






EXEMPLES

Comment utiliser Docker?

- Docker peut être utilisé pour remplacer les machines virtuelles aux coûts exorbitants.
- Docker sert également à rapidement faire le prototype d'un logiciel sans interrompre le bon fonctionnement du système d'exploitation.
- Parce que l'image docker n'a aucune dépendance extérieure, c'est un outil genial pour délivrer les logiciels.
- La possibilité de lancer des centaines de conteneurs isolés sur une seule machine permet de modéliser le réseau informatique avec facilité.
- Ceci augmente la productivité des développeurs de logiciel et leur permet d'embrasser des technologies diverses et Nouvelles.
- Docker ouvre donc la voie à la "Livraison Continue" de logiciels, Continuous Delivery

Installation

[←](#) [→](#) [↺](#) [🔒](#) <https://docs.docker.com/get-docker/> [☆](#) [☆](#) [📁](#) Not syncing 

 **docker docs** [Home](#) [Guides](#) [Product manuals](#) [Reference](#) [Samples](#) 

Get Docker

[Get started](#) ▾

[Develop with Docker](#) ▾

[Deploy your app to the cloud](#) ▾

[Run your app in production](#) ▾


[Open source at Docker](#) ▾


[Documentation archive](#)



Get Docker

Estimated reading time: 1 minute

Docker is an open platform for developing, shipping, and running applications. Docker enables you to separate your applications from your infrastructure so you can deliver software quickly. With Docker, you can manage your infrastructure in the same ways you manage your applications. By taking advantage of Docker's methodologies for shipping, testing, and deploying code quickly, you can significantly reduce the delay between writing code and

 [Edit this page](#)

 [Request docs changes](#)

 ☒ 

On this page:

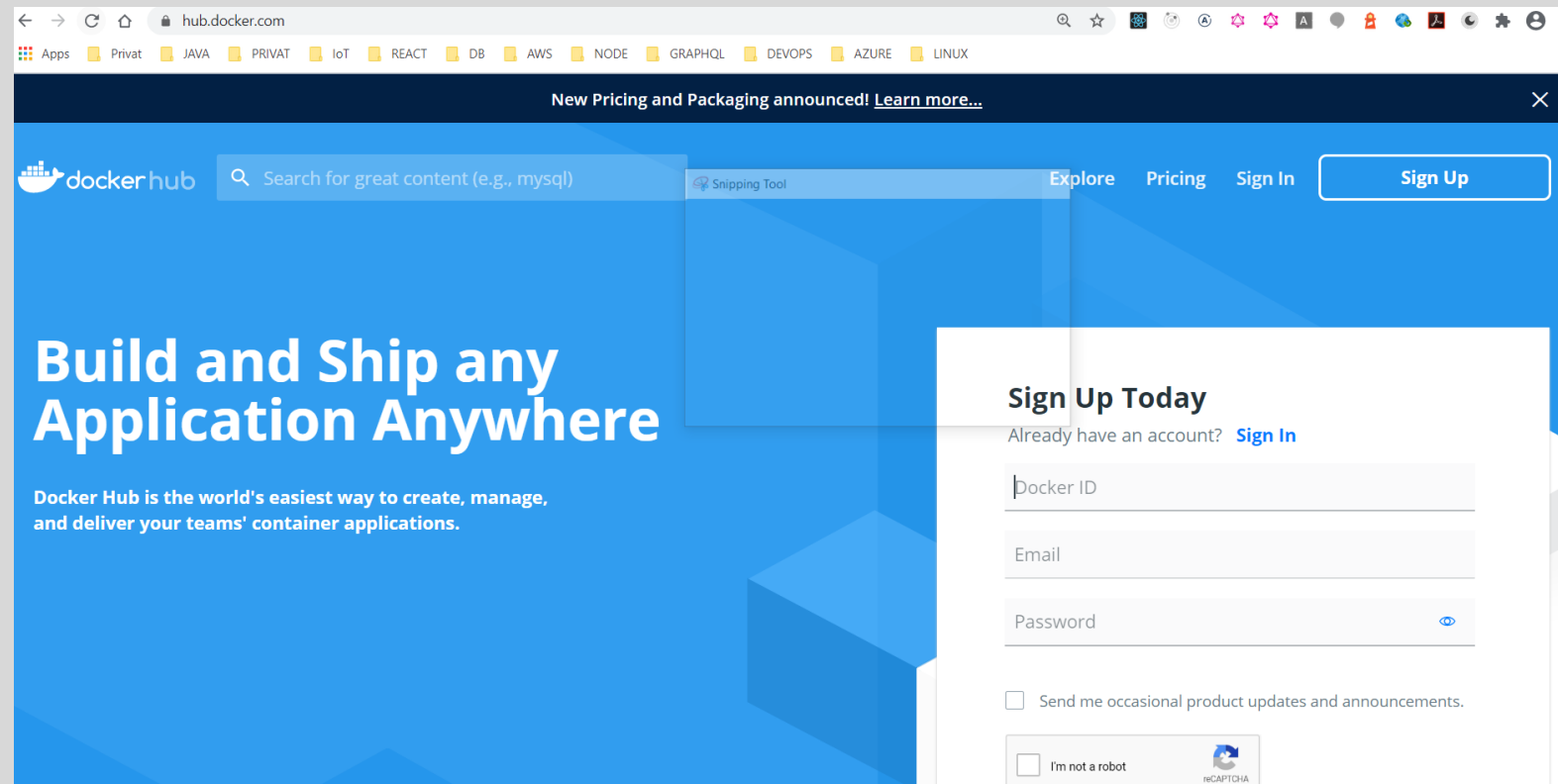
[Docker Desktop for Mac](#)

[Docker Desktop for Windows](#)

[Docker for Linux](#)

Utilisation de Dockerhub

- Lorsque vous utilisez docker, vous avez une conjonction avec Dockerhub.
- Dockerhub est le lieu où sont publiés tous les conteneurs utilisables.



Quelques commandes

- Quelques unes des commandes fréquemment utilisées.

Command	Purpose
<code>docker build</code>	Build a Docker image.
<code>docker run</code>	Run a Docker image as a container.
<code>docker commit</code>	Commit a Docker container as an image.
<code>docker tag</code>	Tag a Docker image.

Test de l'installation Docker

- Nous pouvons tester notre installation en exécutant la commande suivante:

```
Bernard@digitalcloud-1 ~  
$ docker run hello-world  
Unable to find image 'hello-world:latest' locally  
latest: Pulling from library/hello-world  
0e03bdcc26d7: Pull complete  
Digest: sha256:7f0a9f93b4aa3022c3a4c147a449bf11e0941a1fd0bf4a8e6c9408b2600777c5  
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.  
   (amd64)  
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
```

Débuter avec docker

- Pour se mouiller un peu les doigts, tapez la commande `help` pour voir les possibilités.

```
Bernard@digitalcloud-1 /cygdrive/d/ateliiers togojug/workshops
$ docker help

Usage:  docker [OPTIONS] COMMAND

A self-sufficient runtime for containers

Options:
```

--config string	Location of client config files (default "C:\\Users\\adanl\\.docker")
-c, --context string	Name of the context to use to connect to the daemon (overrides DOCKER_HOST env var and default context set with "docker context use")
-D, --debug	Enable debug mode
-H, --host list	Daemon socket(s) to connect to
-l, --log-level string	Set the logging level ("debug" "info" "warn" "error" "fatal") (default "info")
--tls	Use TLS; implied by --tlsverify
--tlscacert string	Trust certs signed only by this CA (default "C:\\Users\\adanl\\.docker\\ca.pem")
--tlscert string	Path to TLS certificate file (default "C:\\Users\\adanl\\.docker\\cert.pem")

Liste des images

- Pour voir la liste des images sur notre machine, exécutez la commande suivante:

```
Bernard@digitalcloud-1 ~  
$ docker image ls
```

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
docker-react_frontend	latest	86e9adcde5e9	5 weeks ago	387MB
docker-react_tests	latest	86e9adcde5e9	5 weeks ago	387MB
mysql	latest	6e447ce4863d	6 weeks ago	544MB
adminer	latest	b1c6e1579623	7 weeks ago	90.4MB
postgres	11.7	028e3a6bd9eb	4 months ago	283MB
docker/desktop-storage-provisioner	v1.1	e704287ce753	5 months ago	41.8MB
docker/desktop-vpnkit-controller	v1.0	79da37e5a3aa	5 months ago	36.6MB
docker/desktop-kubernetes	kubernetes-v1.16.5-cni-v0.7.5-critools-v1.15.0	a86647f0b376	7 months ago	279MB
k8s.gcr.io/kube-controller-manager	v1.16.5	441835dd2301	7 months ago	151MB
k8s.gcr.io/kube-apiserver	v1.16.5	fc838b21afbb	7 months ago	159MB
k8s.gcr.io/kube-scheduler	v1.16.5	b4d073a9efda	7 months ago	83.5MB
k8s.gcr.io/kube-proxy	v1.16.5	0ee1b8a3ebe0	7 months ago	82.7MB
hello-world	latest	bf756fb1ae65	7 months ago	13.3kB
docker/kube-compose-controller	v0.4.25-alpha1	129151cdf35f	10 months ago	35.6MB
docker/kube-compose-api-server	v0.4.25-alpha1	989749268895	10 months ago	50.7MB
docker/kube-compose-installer	v0.4.25-alpha1	2a71ac5a1359	10 months ago	42.3MB

Liste des conteneurs Docker

- Pour lister tous les conteneurs téléchargés sur notre machine, nous pouvons exécuter la commande suivante:

```
Bernard@digitalcloud-1 ~  
$ docker ps --all  
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS              PORTS              NAMES  
c723664ae8bd        hello-world         "/hello"           10 minutes ago     Exited (0) 10 minutes ago              zen_banzai  
ea975b135b2e        postgres:11.7      "docker-entrypoint.s..." 4 weeks ago       Exited (255) 4 weeks ago              event_db_local  
714e63918bd6        adminer             "entrypoint.sh docke..." 5 weeks ago       Up 4 days           0.0.0.0:8080->8080/tcp privat_adminer_1  
58f83cda5d04        mysql              "docker-entrypoint.s..." 5 weeks ago       Up 4 days           3306/tcp, 33060/tcp privat_db_1  
  
Bernard@digitalcloud-1 ~  
$ |
```

Le fichier Dockerfile

- Développement rapide et agile des fonctionnalités métier
- Remplaçabilité
- Isolation et prédictibilité
- Déploiement agile et à échelle
- Alignement avec la structure de l'organisation

```
FROM node:alpine
WORKDIR 'app'
COPY package.json .
RUN npm install
COPY . .
CMD ["npm", "run", "start"]
|
```


Pour avoir un peu d'ordre

- `docker rm -vf $(docker ps -a -q)`
- `docker rmi -f $(docker images -a -q)`
- `docker system prune --all`

Quelques exemples

- Les exemples se trouvent sur github
- <https://github.com/adanlessossi/docker-workshop.git>

CONCLUSION

- Ceci n'est qu'une introduction à docker
- Plus de renseignements:
- <https://docs.docker.com/>
- <https://hub.docker.com/>

