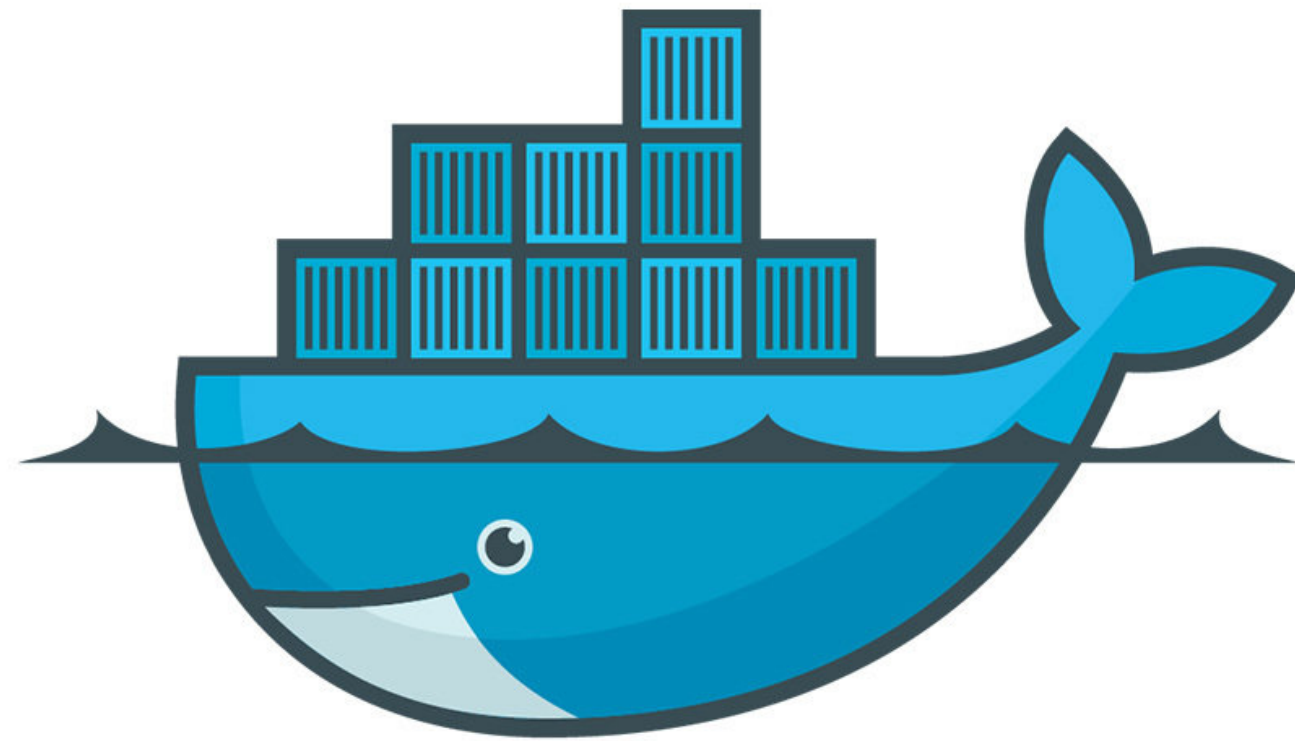


# Docker 101

(offline version)



# Définition – Wikipedia

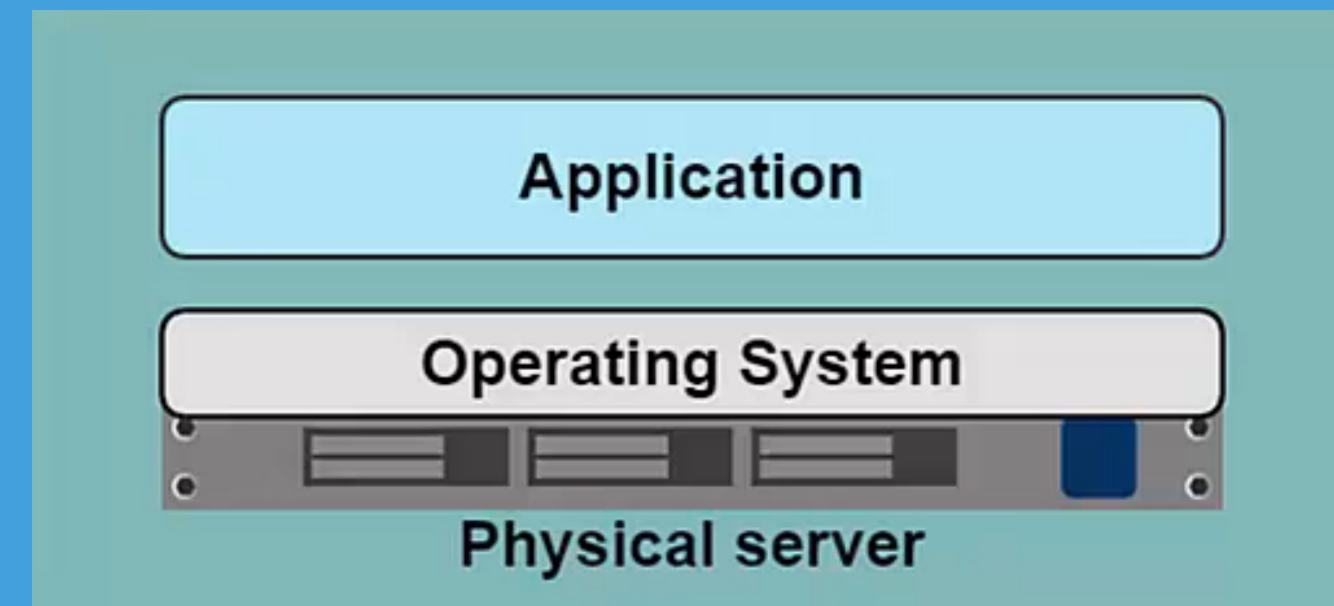
« Docker est un logiciel libre qui automatise le déploiement d'applications dans des conteneurs logiciels. Docker est un outil qui peut empaqueter une application et ses dépendances dans un conteneur isolé, qui pourra être exécuté sur n'importe quel serveur Linux »

# Faire tourner une application

~3 façons

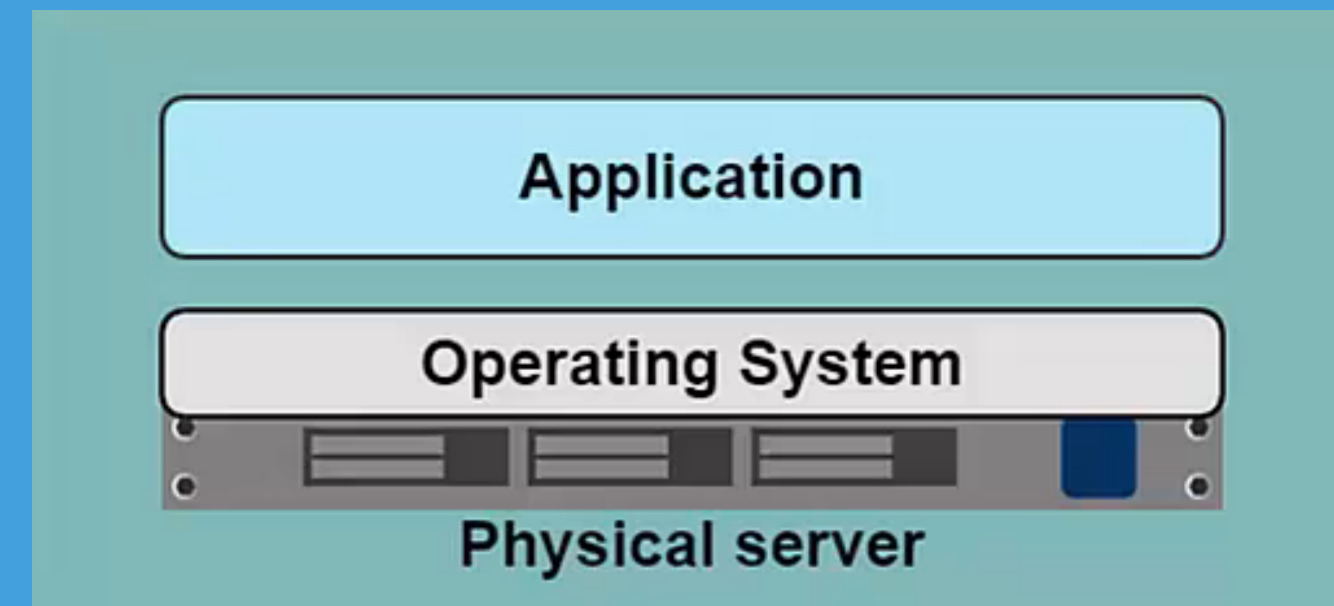
# 1 application par serveur

- fonctionnement historique



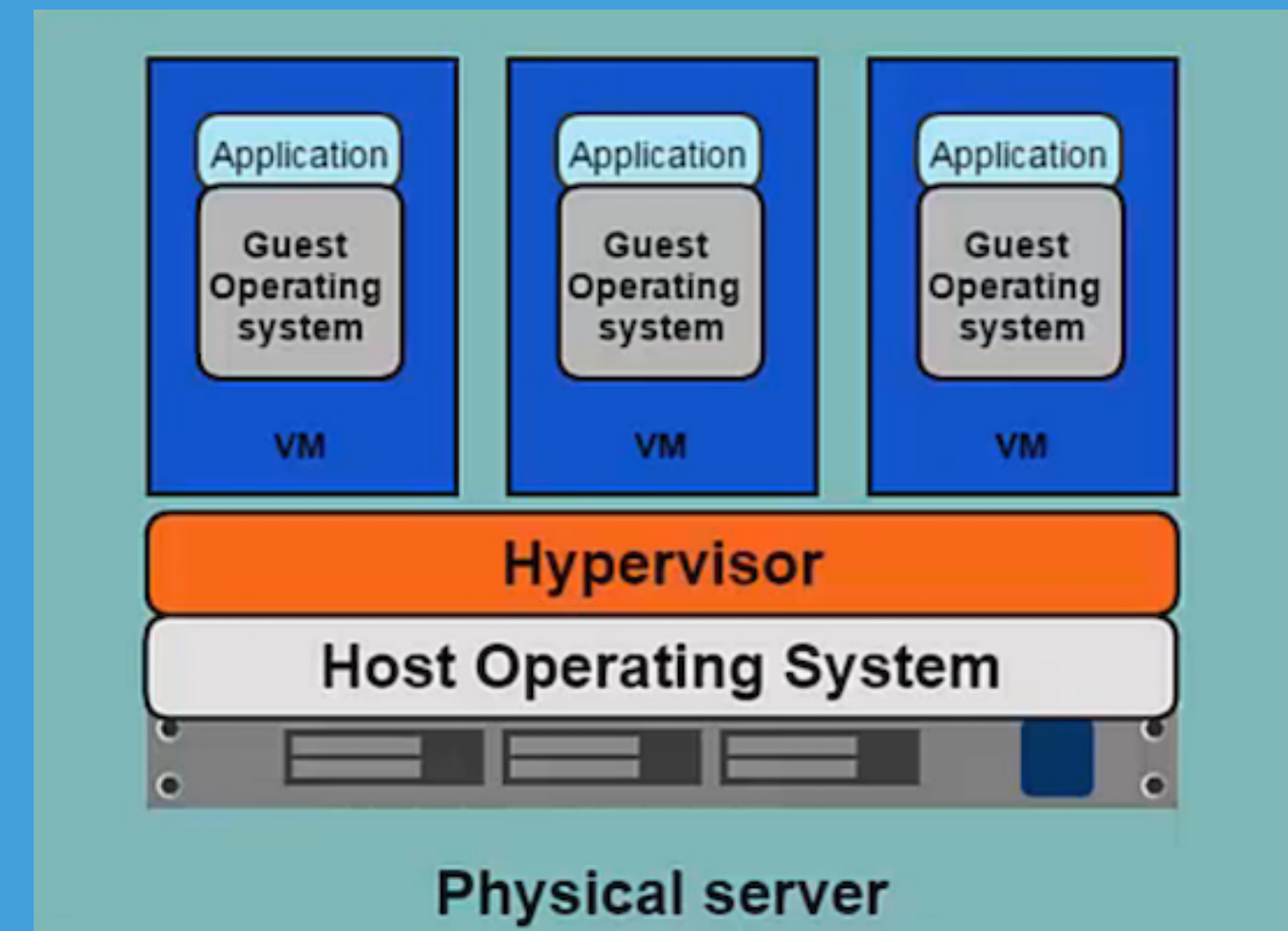
# Problèmes

- déploiement lent
- coûte cher
- ressources gaspillées
- migration et scaling difficile



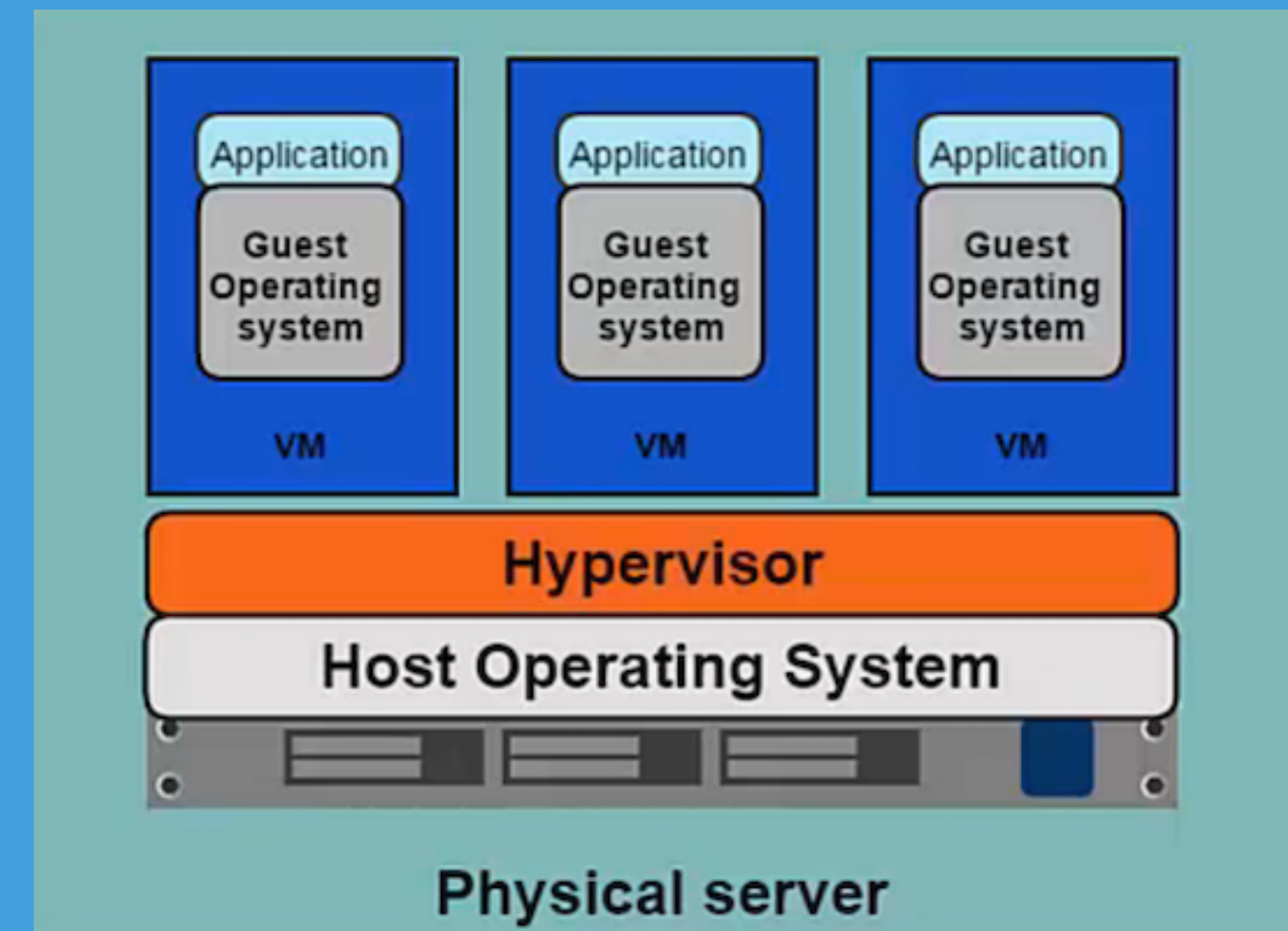
# 1 application par machine virtuelle

- 1 serveur => plusieurs applications
- 1 application <=> 1 machine virtuelle



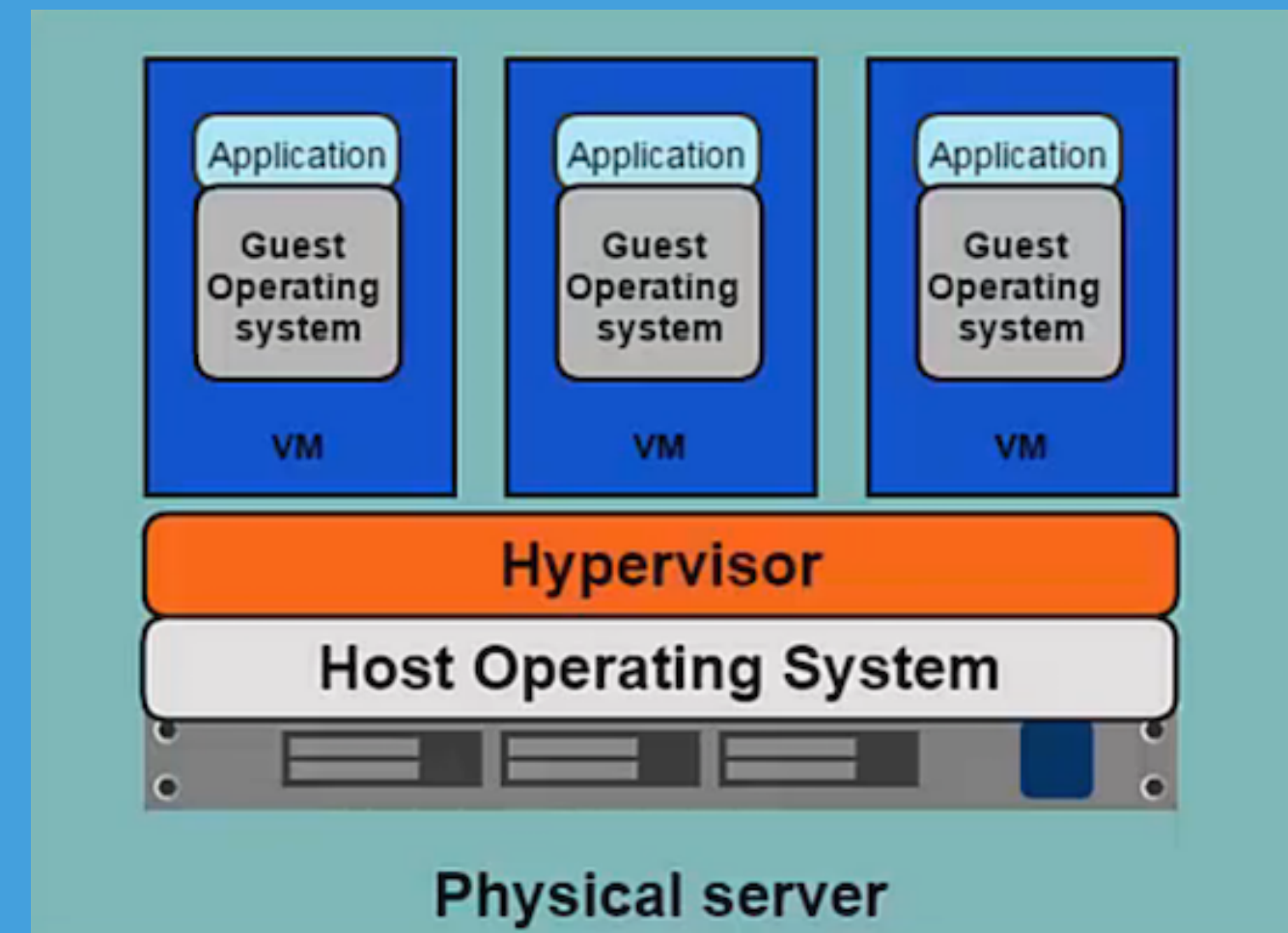
# Avantages

- meilleur gestion des ressources
- scaling plus facile
- portabilité d'une machine virtuelle -> possibilité de faire du Cloud



# Inconvénients

- chaque VM coûte : CPU/Stockage/RAM
- 1 VM = 1 OS
- vite limité en nombre



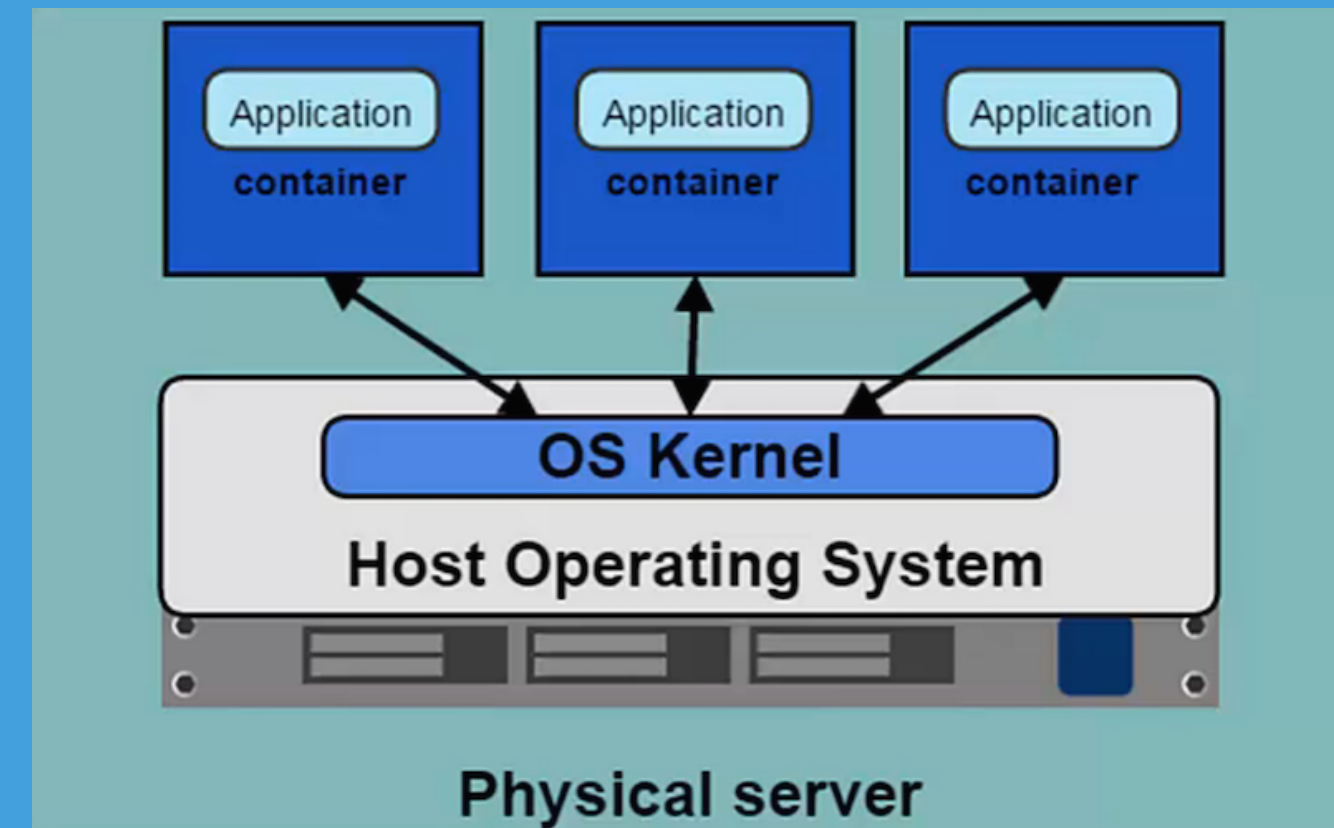


# 1 application par conteneur

Chaque instance logicielle est un conteneur

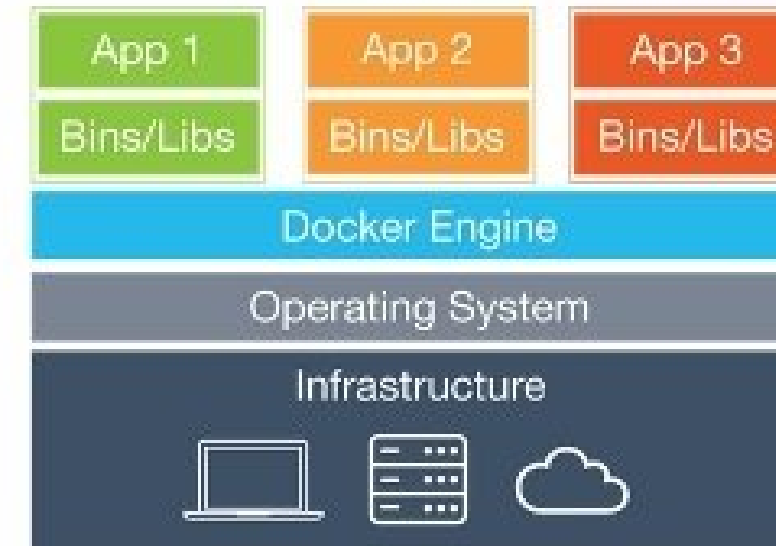
Chaque conteneur à :

- un système de fichier
- ses processus
- sa mémoire
- ses ports réseau





Virtual Machines



Containers

# VM versus Conteneur

Conteneur :

- Pas besoin d'OS invité
- Moins consommateur de CPU/RAM/Stockage
- Plus portable
- Conteneur plus léger

Un Raspberry Pi 2  
> 2500 serveurs web

(DockerCon 2015)



# Et Docker dans tout ça ?



# Images et conteneurs

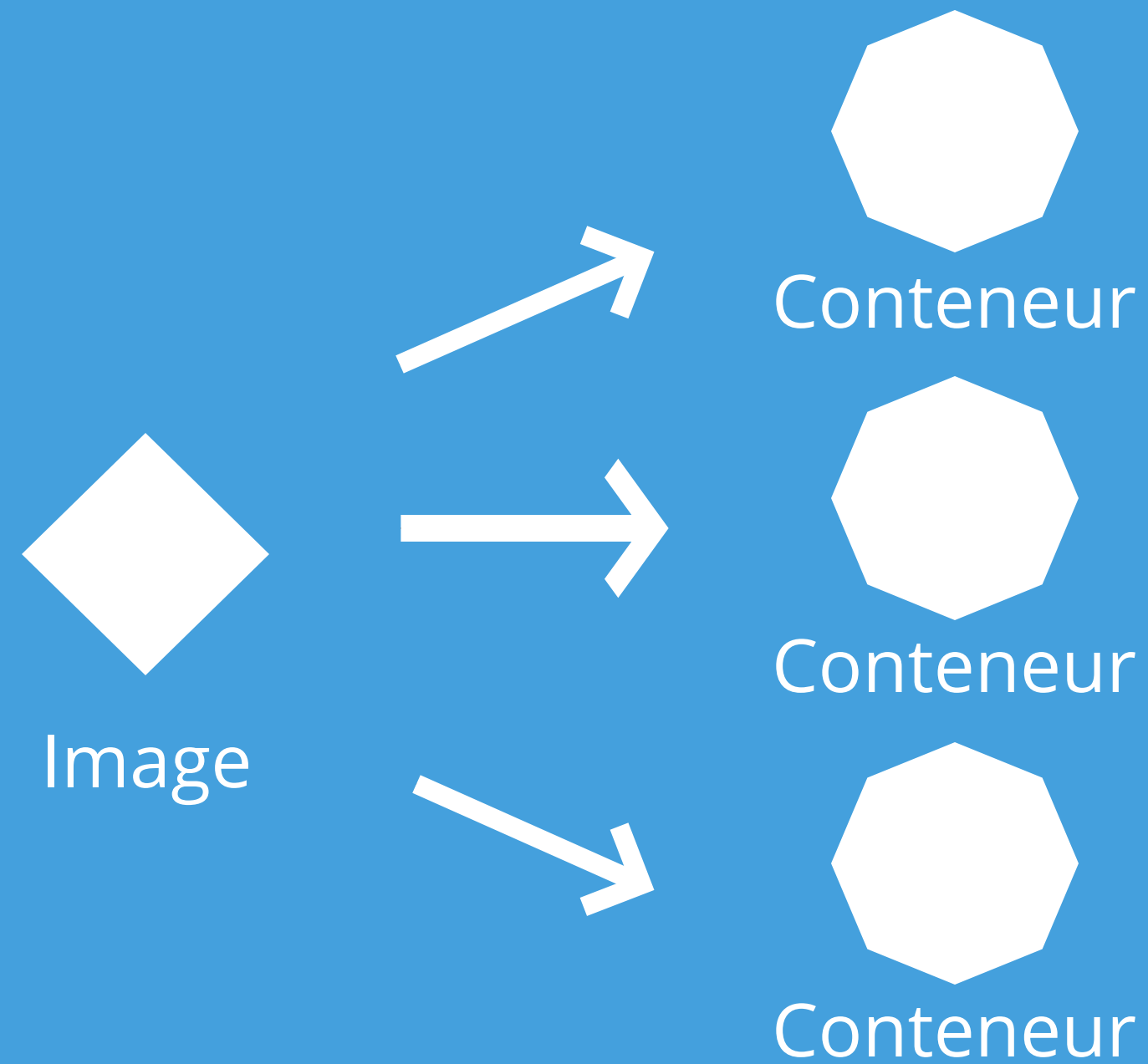
Images :

- fichiers utilisés pour créer un conteneur
- construites par vous ou d'autres utilisateurs
- stockées dans un Registry

Conteneurs :

- instance application isolée
- contient tout le nécessaire pour faire tourner l'application
- basé sur les images

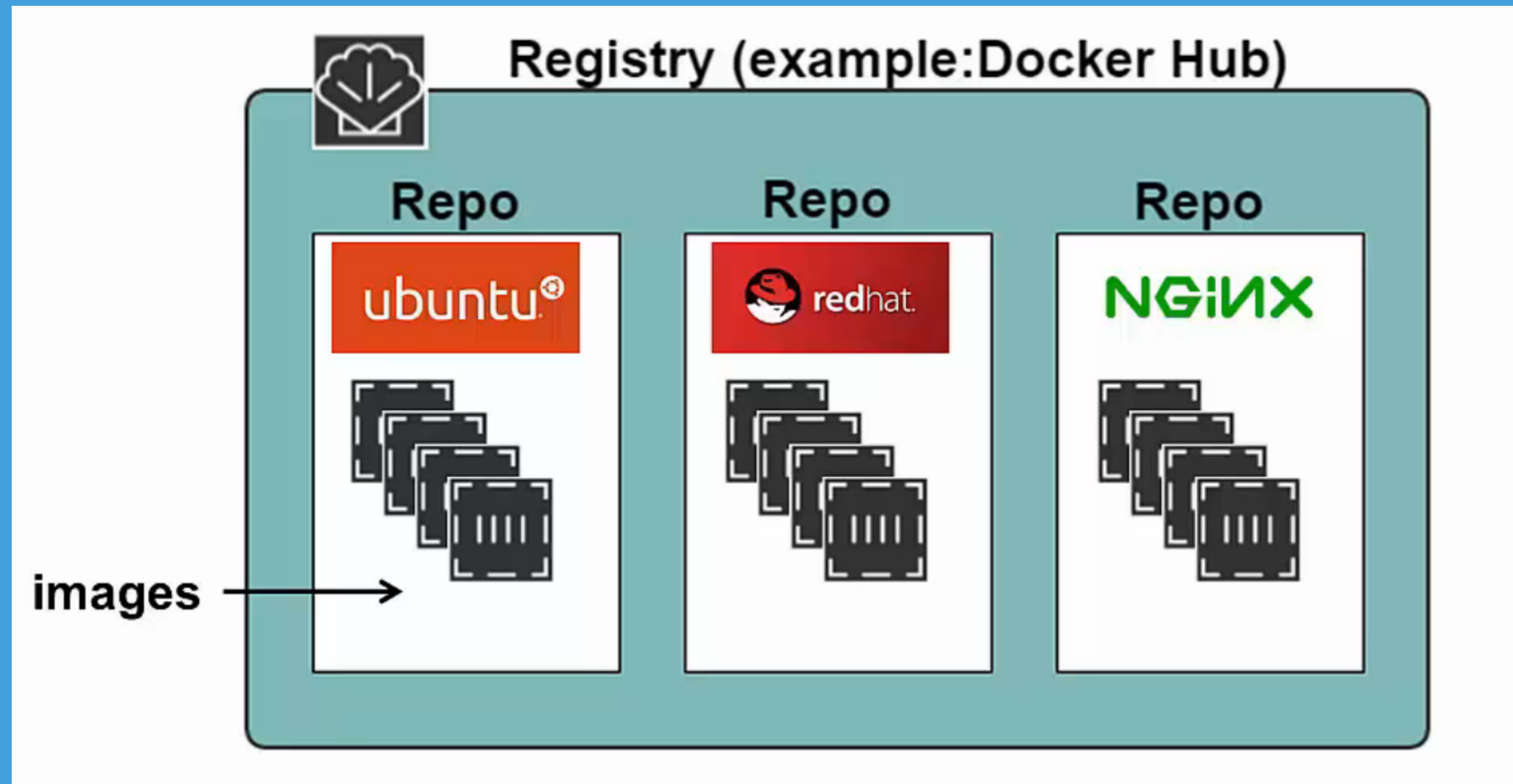
# Principe - image/conteneur














# Images locales

```
# docker images
REPOSITORY          TAG                 IMAGE ID            CREATED             SIZE
drupal               latest             4e47ab49efee       17 hours ago       445MB
ubuntu               latest             6a2f32de169d       6 days ago         117MB
centos               7                  a8493f5f50ff       12 days ago        192MB
centos               latest             a8493f5f50ff       12 days ago        192MB
mongo                latest             b7721338ba63       2 weeks ago        360MB
wordpress            latest             4ad41adc2794       3 weeks ago        401MB
debian               latest             8cedef9d7368       4 weeks ago        123MB
ubuntu               17.04              9c2402b05ac4       7 weeks ago        102MB
hello-world          latest             48b5124b2768       3 months ago       1.84kB
fedora                23                 60ba3309bebb       7 months ago       214MB
```

# Registry distant





<div>  <a href="#">Explore</a> <a href="#">Help</a> </div> <div> <input type="text" value="Search"/> <a href="#">Sign up</a> <a href="#">Log In</a> </div>			
Explore Official Repositories			
 <div>centos official</div>	1295 STARS	1932791 PULLS	<a href="#">   DETAILS         </a>
 <div>busybox official</div>	262 STARS	33397875 PULLS	<a href="#">   DETAILS         </a>
 <div>ubuntu official</div>	2206 STARS	16851379 PULLS	<a href="#">   DETAILS         </a>
 <div>scratch official</div>	89 STARS	209905 PULLS	<a href="#">   DETAILS         </a>
 <div>fedora official</div>	200 STARS	177316 PULLS	<a href="#">   DETAILS         </a>

Official repository  
(**Docker Hub**)

# Premiers pas



# Gérer ses images

```
docker pull hello-world
```

```
docker images
```

```
docker rmi hello-world
```

```
docker images
```

# Mon premier conteneur

```
docker pull ubuntu:17.10  
docker run ubuntu:17.10 bash  
docker run -ti ubuntu:17.10
```

```
docker run [options] <image>[:version] [commande à exécuter]
```

La dernière commande :

- options : partage de l'entrée standard (clavier)
- image : ubuntu (version 17.10)
- commande : par défaut /bin/bash (défini par l'image)

# Gérer ses conteneurs

```
docker pull ubuntu:17.10  
docker run ubuntu:17.10 bash -c "while sleep 1;do date;done"
```

```
docker ps  
docker stop -t 0 <container_name>  
docker ps -a
```

```
docker start <container_name>  
docker attach <container_name>
```

# Lancer un programme Go !

```
docker pull golang:1.8.1  
docker run -v "$PWD":/app golang:1.8.1 go run /app/hello.go
```

Partage d'un volume avec le conteneur

Idem avec du Python ?

```
docker run -v "$PWD":/app python:2.7.13-alpine python /app/date.py
```

# Lancer un serveur web

Partage d'un port avec le conteneur -> <http://localhost>

```
docker run -p 80:80 nginx:1.13.0-alpine
```

Une préférence pour apache ? <http://localhost:1234>

```
docker run -p 1234:80 eboraas/apache
```

*Astuce :*

*Ajouter "-d" pour que le serveur tourne en background*

# Lancer une application de bureau

Partage du réseau et de l'affichage :

```
docker run -ti --rm --privileged \  
  --net=host \  
  -e DISPLAY=$DISPLAY \  
  -v /tmp/.X11-unix:/tmp/.X11-unix \  
  manell/wireshark
```

Possible avec n'importe quelle application : firefox, eclipse, etc...



# Résumé des actions :

## Images

pull  
rmi

## Conteneurs

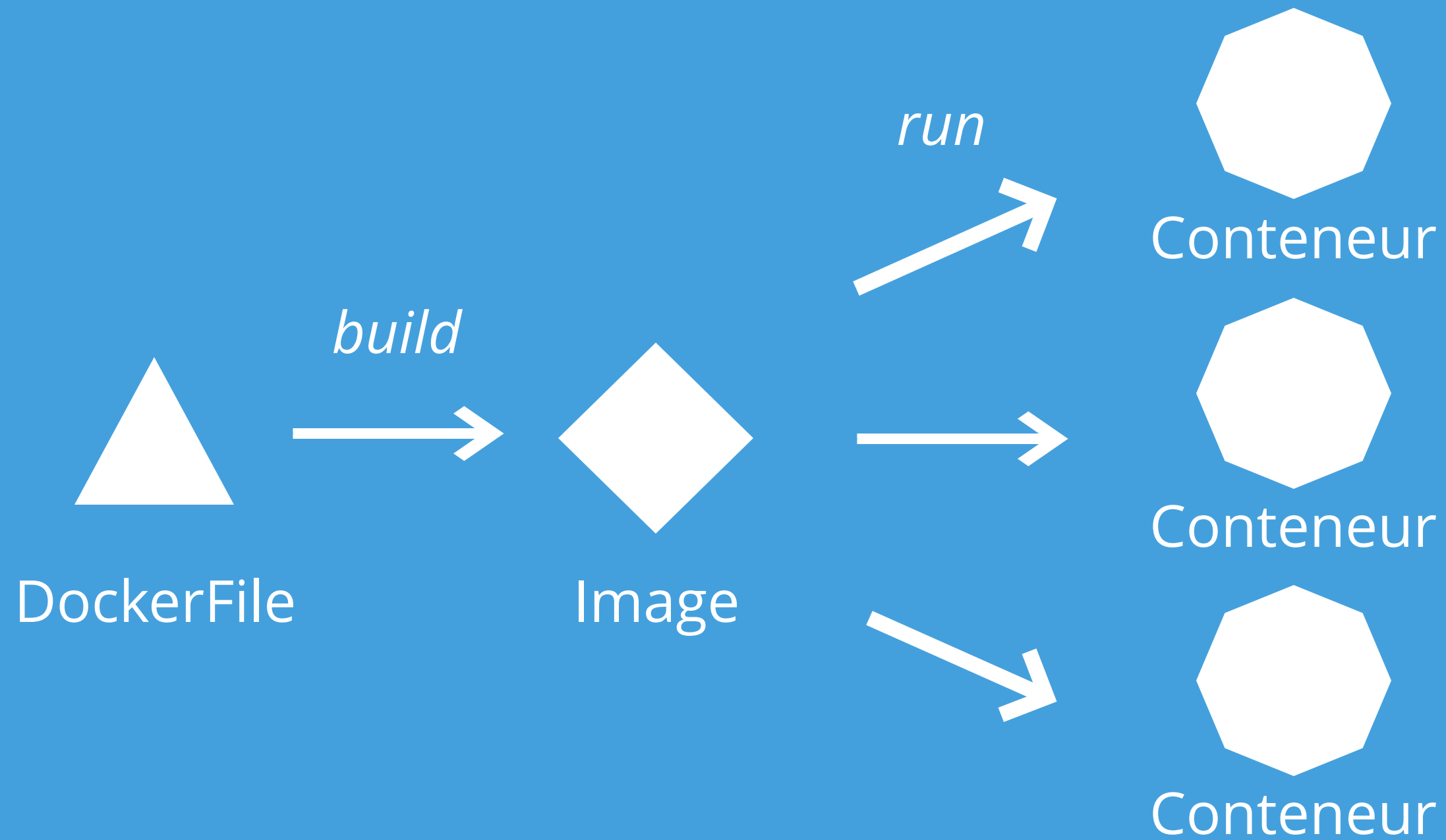
create  
start } run  
stop  
rm

# DockerFiles

La recette pour  
créer ses images



# Principe - image/conteneur



# Besoins particuliers

## Besoin de la commande ping

```
docker run ubuntu:17.10 ping 127.75.101.23
```

```
[...].  
starting container process caused  
"exec: \"ping\": executable file not found in $PATH"  
[..]
```

# Deuxièmes pas



# Créer ses propres images

## Dockerfile

```
FROM ubuntu:17.10  
  
RUN apt-get update  
RUN apt-get install -y iputils-ping  
  
CMD ping 127.0.0.1
```

## Construction et lancement

```
docker build . -t ubuntu-ping  
docker run ubuntu-ping ping 127.75.101.23
```

# Résumé

