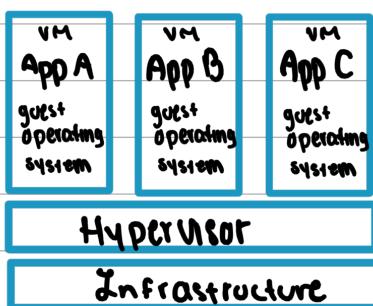


DOCKER

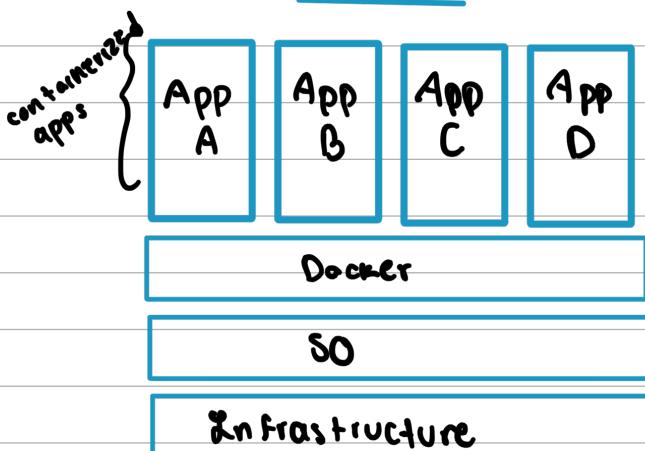
1. Correr programa en un mismo entorno
2. fácil de mover

Virtual Machine



vs

Docker



* Docker corre un contenedor a partir de una imagen.

- La imagen se genera usando Dockerfile

Docker file

```
FROM ubuntu
RUN apt-get install
    apache2
COPY code /var/www
CMD ["apache2"]
```

build ⇒ **Imagen** ⇒ **contenedor**

↳ docker hub para imágenes

\$ docker run [Imagen] : [tag] → versión
(si no se pone descarga la última)

distribuciones:
alpine, Ubuntu,
stretch, etc

• Comandos Comunes

- docker ... run → correr
pull → saco descarga
images → muestra
ps → muestra contenedores
ps -a → log de ↑
start [C.id] → iniciar
docker logs [C.id] → ver logs
Ejecuta comandos ← exec -it [C.id] sh" → ,
sesión interactiva
"emular una shell
stop → detener contenedor
run -d [Imagen] → correr cont. en background

* NO es bueno guardar datos en un contenedor

• ¿Cómo se usa? (en el directorio):

1. Crear Dockerfile → / \$ vim Dockerfile
 - FROM [Imagen] : [tag]
 - WORKDIR [directorío]
 - COPY . . → copia todos los archivos del dockerfile en el directorio
 - compilar ← RUN yarn install -- production
 - (CMD [comando, argumentos]) ↓
 - wando inicia → se corre este binario
ex: "node"
 - ex: "app | curl index.js"

2. construir la imagen → \$ docker build -t [nombre]

3. correr imagen → \$ docker run [nombre]

↳ 3.1 \$ docker run -p -d 3000:3000 [nombre]
↳ poder acceder al puerto

• Volumenes mantener datos

\$ docker run -d -v [directorio] -p 3000:3000 [nombre]
↓
datos example

* permiten modificar el código "al vuelo":

\$ vim [dir script]

\$ docker run -d -v [directorio]

-p 3000:3000

-v [dir script] [nombre] img

sobreescribe
el código
que ya
tenía



después de modificar se debe reconstruir la img

\$ docker build -t [nombre]:[tag].

↓
para
diferenciar
ex: v2

• Compartir Imágenes Propias:

* se deben subir a un registro

1. log in

2. \$ docker tag [@id] [autor/nombre:tag]

3. \$ docker push [Imagen nombre]

* En la vida real se utilizan muchos contenedores.
↳ separados para que no haya problemas entre dependencias

• ¿Cómo? (esto es manual, se puede automatizar con Docker Compose así que salten esta parte)

1. Crear nueva red → \$ docker network create [nombre]

2. Correr → \$ docker run -d
network
comparar archivos ↪ > --network [nombre] --network-alias [alias]
> -v [nombre directorio]:[directorío] ↪ montar en ...

> -e [variable de entorno]

-e
... ↪ variables de entorno

ex: MYSQL-DATABASE = [nombre]

> [Imagen] : [tag]

↳ las variables de entorno se ven en la documentación de la imagen

3. \$ docker run -it --network [nombre] nicolaka/netshoot
\$ dig mysql

↙ abrir contenedor que me permita conectarme a la network

4. \$ docker run -d 3000:3000 |

> -e ... } var de entorno
... }
... ex: MYSQL-HOST
... MYSQL-USER
... MYSQL-PASSWORD
... MYSQL-DB

> [Nombre imagen]

↙ abriendo contenedor conectado a la network anteriormente creada

• Docker Compose

- facilita escribir los comandos ex: ahorrarnos la creación de networks.

\$ vim docker-compose.yaml

[nombre-container]

secciones → version

services → declarar servicios que vamos a correr (ver documentación)

- se llenan los campos

\$ docker-compose up -d → correr todo en el archivo

\$ docker-compose down → bajar todo