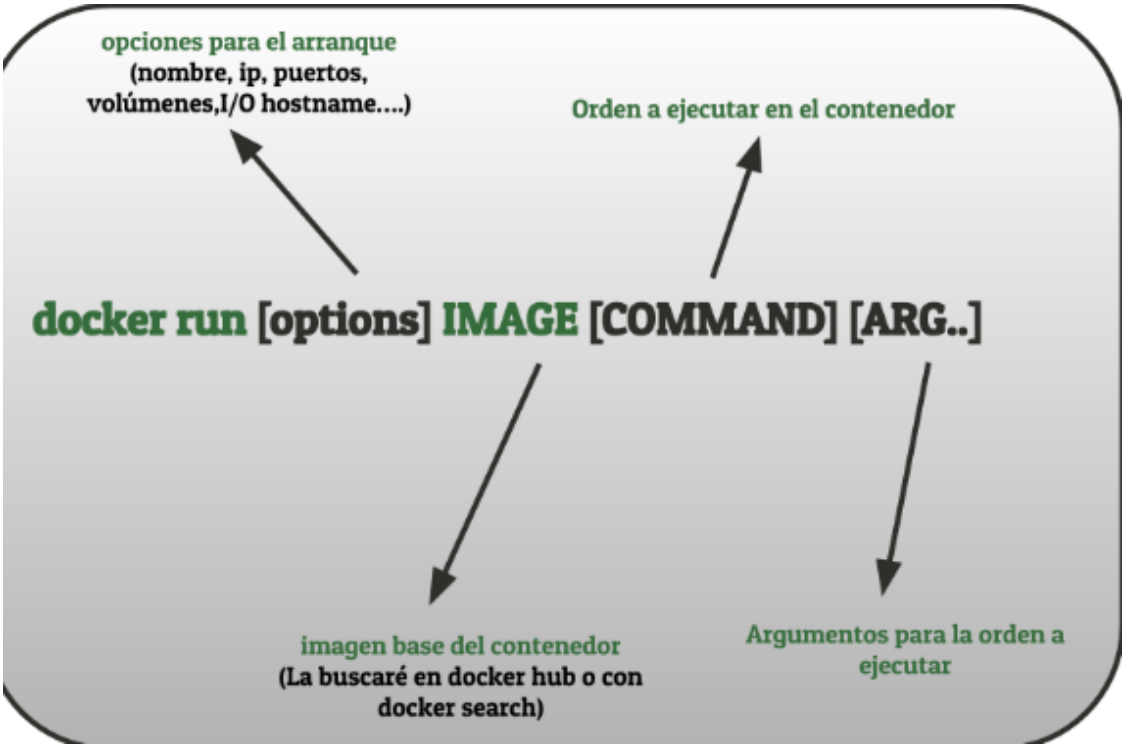
Chuletario comandos Docker

Román Millán Díaz

Comando	Descripción
docker pull nombreImagen:version	Para descargar la imagen
docker [options] run [COMMAND][ARG..]	Para poner en ejecución los contenedores en base a la imagen descargada. Si no tenemos la imagen la descarga automáticamente y después la ejecuta.



- **-e o --env** para establecer variables de entorno en la ejecución del contenedor.
- **-h o --hostname** para establecer el nombre de red parar el contenedor.
- **--help** para obtener ayuda de las opciones de docker.
- **--interactive o -i** para mantener la STDIN abierta en el contenedor.
- **--ip** si quiero darle una ip concreta al contenedor.
- **--name** para darle nombre al contenedor.
- **--net o --network** para conectar el contenedor a una red determinada.
- **-p o --publish** para conectar puertos del contenedor con los de nuestro host.
- **--restart** que permite reiniciar un contenedor si este se "cae" por cualquier motivo.
- **--rm** que destruye el contenedor al pararlo.
- **--tty o -t** para que el contenedor que vamos a ejecutar nos permita un acceso a un terminal para poder ejecutar órdenes en él.
- **--user o -u** para establecer el usuario con el que vamos a ejecutar el contenedor.
- **--volume o -v** para montar un bind mount o un volumen en nuestro contenedor.
- **--wordirk o -w** para establecer el directorio de trabajo en un contenedor.

docker start [opcional] nombreContenedor	Para arrancar el contenedor
-d	para ejecutarlo en modo background o dettach
-p PUERTO_EN_HOST:PUERTO_EN_CONTENEDOR	para redireccionar el puerto.
-e NOMBRE_VARIABLE=VALOR	Comprobar y definir si es necesario las variables de entorno que puede tener el contenedor.
--name nombreAponer	dar nombre por defecto
docker exec [opciones] nombre_contenedor orden [argumentos]	Para ejecutar órdenes en contenedores en ejecución.

- **-it** (-i y -t juntos) si vamos a querer tener interactividad con el contenedor ejecutando un shell (/bin/bash normamente). Una vez tenemos el terminal ya podremos trabajar desde dentro del propio sistema.
- **-u o --user** si quiero ejecutar la orden como si fuera un usuario distinto del de root.
- **-w o --workdir** si quiero ejecutar la orden desde un directorio concreto.

docker cp	permite mover ficheros desde mi sistema al contenedor y desde el contenedor a mi sistema

Ejemplo
docker cp prueba.html web:/usr/local/apache2/htdocs/index.html

En ese contexto hay varios comandos docker que me van a ayudar a obtener información de un contenedor. En este curso vamos a usar los dos siguientes:

- La orden **docker ps**.
- La orden **docker inspect.**
- La orden **docker logs.**

docker ps	para obtener información de los contenedores ya arrancados

```
# Mostrar todos los contenedores, estén parados o en ejecución (-a o --all)

> docker ps -a

# Añadir la información del tamaño del contenedor a la información por defecto (-s o --size)

> docker ps -a -s

# Mostrar información del último contenedor que se ha creado (-l o --latest). Da igual el estado

> docker ps -l

# Filtar los contenedores de acuerdo a algún criterio usando la opción (-f o --filter)

# Filtrado por nombre

> docker ps --filter name=servidor_web

# Filtrado por puerto. Contenedores que hacen público el puerto 8080

> docker ps --filter publish=8080
```

docker inspect	os va a dar una **información detallada** del contenedor que seleccione

Ejemplo:

```
# Por nombre. Por ejemplo: Mostrar información detallada del contenedor cuyo nombre es jenkins

> docker inspect jenkins

# Por id. Por ejemplo: Mostrar información detallada del contenedor cuyo id es
```

```
5e5adf6815bc

> docker inspect 5e5adf6815bc
```

docker logs	dan información de lo que está pasando en el contenedor. Sirven tanto en contenedores que estén parados como para contenedores en ejecución.

Ejemplo:

Por nombre. Por ejemplo: Mostrar los logs del contenedor cuyo nombre es jenkins

> docker logs jenkins

Por id. Por ejemplo: Mostrar los logs cuyo id es 5e5adf6815bc

> docker logs 5e5adf6815bc

- **docker stop** para detener el contenedor, ya sea por nombre o por ID.
- **docker rm** para borrar el contenedor, ya sea por nombre o por ID.
- **docker start** iniciar un contenedor que estaba parado previamente, ya sea por nombre o por ID.
- **docker restart** para reiniciar un contenedor que previamente ya estaba en ejecución.

Ejemplos:

```
# Para un contenedor en ejecución que se llame servidorWeb

> docker stop servidorWeb

# Para un contenedor en ejecución cuyo ID es ea9b922190d8 pero esperando 10 segundo (-t o --time)

> docker stop -t 10 ea9b922190d8

# Borrar un contenedor que se llama servidorBD

> docker rm servidorBD

# Borrado un contenedor que se llame jenkins aunque esté en ejecución (--force o -f)

> docker rm -f jenkins
```

```
# Inicio de un contenedor con nombre jenkins

> docker start jenkins

# Inicio de un contenedor con nombre jenkins pero haciendo el attach de la entrada estándar para poder interactuar con él (-i o --interactive)

> docker start -i jenkins

# Reinicio de un contenedor con ID ea9b922190d8

> docker restart ea9b922190d8

# Reinicio de un contenedor con ID ea9b922190d8 pero esperando 10 segundo (-t o --time)

> docker restart -t 10 ea9b922190d8
```

Es importante destacar que SI UN CONTENEDOR ESTÁ EN EJECUCIÓN NO PODEMOS BORRARLO salvo que usemos la opción -f.