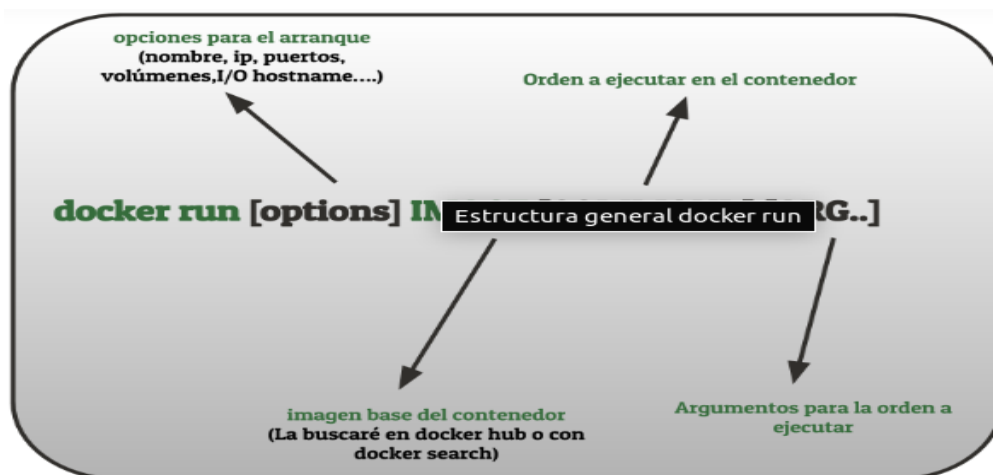


docker pull nombre_imagen:version	Nos descargará desde el repositorio una imagen con la versión indicada o la última versión (latest) si no indicamos versión.
docker run	<p>Su función principal es poner en ejecución contenedores en base a una imagen de referencia que le indicaremos.</p> <p>Para buscar las imágenes utilizaremos Docker hub.</p>



- d o **--detach** para ejecutar un contenedor (normalmente porque tenga un servicio) en background.
- e o **--env** para establecer variables de entorno en la ejecución del contenedor.
- h o **--hostname** para establecer el nombre de red para 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.
- workdir** o -w para establecer el directorio de trabajo en un contenedor.

<p>flag -it</p>	<p>Nos permite interactuar con el contenedor en el caso de que sea un contenedor que no tenga servicios.</p> <p>-it es la unión del flag -i (--interactive) y el flag -t (--tty) que lo que hacen es abrir la entrada estándar del contenedor que estamos ejecutando y permitir la posibilidad de abrir un terminal en el contenedor.</p> <p>No debemos añadir al final otra orden que no sea /bin/bash</p>
<p>flag -d</p>	<p>Hace que el servicio se ejecute en modo background o Dettach.</p> <p>Si no lo hacemos se bloqueará el terminal mostrando el log del servicio (en ciertas ocasiones puede interesarnos) y tendremos que salir del mismo con Ctrl+C.</p>
<p>flag -p</p>	<p>Si queremos que el servicio que vamos a lanzar sea accesible desde el exterior</p> <p>con el (-p) PUERTO_EN_HOST:PUERTO_EN_CONTENEDOR que normalmente sería el puerto por defecto del servicio. Esto es una REDIRECCIÓN DE PUERTOS.</p>
<p>flag -e NOMBRE_VARIABLE=VALOR.</p>	<p>Nos servirá para poder comprobar y definir las variables de entorno.</p>

Ejecuto un servidor Apache sin el flag -d ni redirección de puertos. Se bloquea el terminal mostrando los logs y tendré que salir con Ctrl+C

```
> docker run httpd
```

```
AH00558: httpd: Could not reliably determine the server's fully qualified domain name, using 172.17.0.22. Set the 'ServerName' directive globally to suppress this message
```

```
AH00558: httpd: Could not reliably determine the server's fully qualified domain name, using 172.17.0.22. Set the 'ServerName' directive globally to suppress this message
```

```
[Mon Dec 07 18:27:28.561909 2020] [mpm_event:notice] [pid 1:tid 140253864719488] AH00489: Apache/2.4.46 (Unix) configured -- resuming normal operations
```

```
[Mon Dec 07 18:27:28.562072 2020] [core:notice] [pid 1:tid 140253864719488] AH00094: Command line: 'httpd -D FOREGROUND'
```

Ejecuto un servidor Apache en background y accediendo desde el exterior a través del puerto 8888 de mi máquina.

```
> docker run -d -p 8888:80 httpd
```

Creación de un servidor de base de datos mariadb accediendo desde el exterior a través del puerto 3306 y estableciendo una contraseña de root mediante una variable de

entorno

```
> docker run -it -d -p 3306:3306 -e MYSQL_ROOT_PASSWORD=root mariadb
```

flag --name

Sirve para **asignar nombre elegidos por nosotros**, que serán más fáciles de recordar que los asignados por defecto. Además podemos elegir nombres que tenga relación con la función que va a desempeñar dicho contenedor.

Damos el nombre de servidorBD a un contenedor de la imagen mysql:8.0.22

```
> docker run -d --name servidorBD -p 3306:3306 mysql:8.0.22
```

Damos el nombre de servidorWeb a un contenedor de la imagen httpd:latest (Apache)

```
> docker run -d --name servidorWeb -p 80:80 httpd
```

<p>docker exec</p> <p>(ES NECESARIO QUE EL CONTENEDOR ESTÉ EN EJECUCIÓN.)</p> <p>Nos permite ejecutar acciones como:</p> <ul style="list-style-type: none"> ● Instalar paquetes. ● Modificar o ver el contenido de ciertos ficheros. ● Habilitar ciertos módulos de servicios ● etc... 	<p>docker exec [opciones] nombre_contenedor orden [argumentos]</p> <ul style="list-style-type: none"> ● -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.
<pre># Obtener un terminal en un contenedor que ejecutar un servidor Apache (httpd) y que se llama web > docker exec -it web /bin/bash root@5d96ce1f7374:/usr/local/apache2# # Mostrar el contenido de la carpeta /usr/local/apache2/htdocs del contenedor web. Como no hace falta interactividad no es necesario -it > docker exec web ls /usr/local/apache2/htdocs # Crear directamente un fichero "HOLA MUNDO" en el directorio raíz del servidor apache. Utilizo sh -c para ordenes compuestas o complejas > docker exec -it web sh -c "echo 'HOLA MUNDO' > /usr/local/apache2/htdocs/index.html"</pre>	
<p>docker cp</p>	<p>Nos permite mover ficheros desde mi sistema al contenedor y desde el contenedor a mi sistema.</p>

```
# Copiar mi fichero prueba.html al fichero /usr/local/apache2/htdocs/index.html de mi contenedor llamado web que es un servidor Apache (httpd)
```

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

```
#Copiar el fichero index.html que se encuentra en /usr/local/apache2/htdocs/index.html de m contenedor llamado web un fichero llamado test.html en mi directorio HOME
```

```
> docker cp web:/usr/local/apache2/htdocs/index.html $HOME/test.html
```

NOTA: LOS CONTENEDORES VIENEN CON SOLO LO IMPRESCINDIBLE INSTALADO. SI QUIERO INSTALAR ALGO DEBO NORMALMENTE HACER ANTES UN APT UPDATE (ya que la mayoría son basados en Debian).

docker ps

Obtener información de los contenedores ya arrancados:
El **estado** del contenedor, la **imagen** de la que deriva, el **tamaño** actual del contenedor, la **orden** que ejecuta el contenedor al arrancar, lo que se llama el **ENTRYPOINT**, el **nombre** del contenedor, cuando fue **creado** el contenedor y las redirecciones de **puertos**

```

# Mostrar los contenedores que están en ejecución

> docker ps

# 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

# Filtrar 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 nombreContenedor

ó IDcontenedor

Información detallada del contenedor que seleccione.

- El **id** del contenedor.
- Los **puertos** abiertos y sus redirecciones
- Los bind mounts y **volúmenes** usados.
- El **tamaño** del contenedor
- La **configuración de red** del contenedor.
- El **ENTRYPOINT** que es lo que se ejecuta al hacer docker run.
- El valor de las **variables** de entorno.
- ...

```
# Mostrar la ip del contenedor

> docker inspect --format 'La ip es {{.NetworkSettings.Networks.bridge.IPAddress}}' jenkins

La ip es 172.17.0.2

# Mostrar las redirecciones de puertos del contenedor

> docker inspect --format 'Las redirecciones de puertos son {{.NetworkSettings.Ports}}' jenkins

Las redirecciones de puertos son map[50000/tcp:[{0.0.0.0 50000}] 8080/tcp:[{0.0.0.0 9090}]]
```

docker logs

Nos dan **información** de lo **que está pasando** en el contenedor. Que me va a servir tanto para contenedores que estén parados como para contenedores en ejecución.

```
# 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

# Opción -f o --follow . Sigue escuchando la salida que pueden dar los logs del contenedor

> docker logs -f jenkins

# Opción --tail 5. Muestra las 5 últimas líneas de los logs del contenedor en cuestión

> docker logs --tail 5 jenkins
```

docker stop

Para **detener** el contenedor, ya sea por **nombre** o por **ID**.

docker rm	<p>Para borrar el contenedor, ya sea por nombre o por ID.</p> <p>SI UN CONTENEDOR ESTÁ EN EJECUCIÓN NO PODEMOS BORRARLO salvo que usemos la opción -f.</p>
docker start	<p>Iniciar un contenedor que estaba parado previamente, ya sea por nombre o por ID.</p>
docker restart	<p>Para reiniciar un contenedor que previamente ya estaba en ejecución.</p> <p>Si hago docker restart y el contenedor ya está parado, es lo mismo que si ejecutar un docker start.</p>
<pre> # 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 </pre>	

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 segundos (-t o --time)

> docker restart -t 10 ea9b922190d8