Contenedores e imágenes

Los contenedores son instancias de imágenes que se ejecutan en Docker. Un contenedor es una instancia de una imagen que se ejecuta en un entorno aislado. Los contenedores son ligeros y portables, y se pueden ejecutar en cualquier lugar donde Docker esté instalado.

Comandos generales

Crear un contenedor

Para crear un contenedor se usa docker run especificando la imagen a usar y las opciones necesarias.

Por ejemplo:

docker run -d -p 7001:80 httpd:latest

RECUERDA: la imagen debe ser el último parámetro.

RECUERDA: si no se especifica un tag, se usa latest.

Al ejecutarlo, primero comprueba si la imagen httpd está en local. En caso contrario, la descarga de Docker Hub. El parametro -d indica que se ejecuta en segundo plano. El parametro -p indica el mapeo de puertos. En este caso, el puerto 80 del contenedor se mapea al puerto 7001 del host.

Si accedemos a [http://localhost:7001](http://localhost:7001/) veremos la página de inicio de Apache.

Ver contenedores

Para ver los contenedores en ejecución:

docker ps

Para ver todos los contenedores:

docker ps -a

Parar un contenedor

Para parar un contenedor:

docker stop <container\_id>

Eliminar un contenedor

Para eliminar un contenedor:

docker rm <container\_id>

Ver imágenes

Para ver las imagenes locales:

docker imagenes

Eliminar una imagen

Para eliminar una imagen:

docker rmi <image\_id>

Volumenes

Los datos que hay en el contenedor son volátiles. Si se elimina el contenedor, se pierden los datos. Para persistir los datos, se pueden usar volúmenes.

Comandos generales

Crear un volumen

Para crear un volumen, se usa el comando docker volume create. Por ejemplo:

docker volume create auth\_volume

Ver volúmenes

Para ver los volúmenes creados:

docker volume ls

Inspeccionar un volumen

Para ver los detalles de un volumen:

docker volume inspect auth\_volume

Eliminar un volumen

Para eliminar un volumen:

docker volume rm auth\_volume

Montar un volumen en un contenedor

Para montar un volumen en un contenedor, se usa la opción -v. Por ejemplo:

docker run -d -p 7004:8080 --env-file ./Docker/env/api-aut.env -v auth\_volume:/app/trazas registry2.spyrosoftware.com/rd/autenticacion/api/api-aut:master

En este caso, el volumen auth\_volume se monta en el directorio /app/trazas del contenedor. Si se elimina el contenedor, los datos del volumen se mantienen.

También se puede crear un volumen ligado a un directorio local. Por ejemplo:

docker run -d -p 7004:8080 --env-file ./Docker/env/api-aut.env -v <d

Redes por defecto

Cuando instalamos Docker, se crean tres redes por defecto:

* bridge: Es la red por defecto que se utiliza para conectar los contenedores en el mismo host. Los contenedores en la red bridge pueden comunicarse entre sí utilizando sus direcciones IP.
* host: Es una red especial que permite a los contenedores compartir la red del host. Los contenedores en la red host pueden acceder a los servicios del host utilizando la dirección IP localhost.
* none: Es una red nula que no tiene acceso a la red. Los contenedores en la red none no pueden comunicarse con otros contenedores ni con el host.

Crear una red

Para crear una red en Docker, podemos utilizar el comando docker network create. Por ejemplo, para crear una red llamada mi-red:

docker network create mi-red

Ver redes

Podemos ver las redes en Docker utilizando el comando docker network ls:

docker network ls

Conectar un contenedor a una red

Para conectar un contenedor a una red en Docker, podemos utilizar la opción --network o -network al crear el contenedor. Por ejemplo, para conectar un contenedor a la red mi-red:

docker run --name mi-contenedor --network mi-red -d nginx

También podemos conectar un contenedor a más redes existente utilizando el comando docker network connect. Por ejemplo, para conectar el contenedor mi-contenedor a la red mi-red:

docker network connect mi-red mi-contenedor

NOTA: Un contenedor puede estar conectado a múltiples redes.

Desconectar un contenedor de una red

Para desconectar un contenedor de una red en Docker, podemos utilizar el comando docker network disconnect. Por ejemplo, para desconectar el contenedor mi-contenedor de la red mi-red:

docker network disconnect mi-red mi-contenedor

Eliminar una red

Para eliminar una red en Docker, podemos utilizar el comando docker network rm. Por ejemplo, para eliminar la red mi-red:

docker network rm mi-red

Docker-compose

docker-compose up