

# CheatSheet de Comandos Docker – Con Explicaciones

## Comandos Docker CLI Básicos

### **docker run [opciones] imagen**

Crea y ejecuta un contenedor a partir de una imagen.

### **docker ps**

Lista los contenedores en ejecución.

### **docker ps -a**

Lista todos los contenedores, incluyendo los detenidos.

### **docker stop <id>**

Detiene un contenedor enviando SIGTERM.

### **docker kill <id>**

Fuerza la detención de un contenedor usando SIGKILL.

### **docker rm <id>**

Elimina un contenedor detenido.

### **docker create imagen**

Crea un contenedor sin iniciarlo.

### **docker start <id>**

Inicia un contenedor ya creado.

### **docker logs [-f] <id>**

Muestra los logs del contenedor (-f sigue en tiempo real).

### **docker inspect <id>**

Muestra todos los detalles internos en formato JSON.

## Acceso y Ejecución

### **docker exec -it <id> /bin/bash**

Abre una shell dentro de un contenedor en ejecución.

### **docker run -p HOST:CONT imagen**

Mapea un puerto del host a un puerto dentro del contenedor.

### **docker run -e VAR=VALOR imagen**

Define variables de entorno para el contenedor.

## Imágenes

### **docker pull imagen**

Descarga una imagen del registro (Docker Hub).

### **docker search termino**

Busca imágenes públicas en Docker Hub.

### **docker build -t nombre .**

Construye una imagen usando un Dockerfile.

### **docker tag origen destino**

Crea una etiqueta nueva para una imagen existente.

### **docker push destino**

Sube una imagen a un registro.

### **docker commit -m 'msg' <id> nueva\_img**

Crea una imagen desde un contenedor modificado (no recomendado para producción).

## Instrucciones Dockerfile

### **FROM**

Imagen base de la cual parte el contenedor.

### **RUN**

Ejecuta comandos durante la construcción de la imagen.

### **COPY**

Copia archivos del host al contenedor.

### **ADD**

Igual que COPY, pero permite URLs y extraer archivos .tar.

### **CMD**

Comando por defecto al iniciar el contenedor.

### **ENTRYPOINT**

Comando principal que siempre se ejecuta.

### **WORKDIR**

Establece el directorio de trabajo.

### **ENV**

Declara variables de entorno.

**USER**

Usuario con el que se ejecutarán los procesos.

**EXPOSE**

Documenta puertos usados (no los abre).

**VOLUME**

Crea un punto de montaje persistente.

**HEALTHCHECK**

Define un test de salud automático.

**LABEL**

Añade metadatos a la imagen.

## Volúmenes y Persistencia

**docker volume create nombre**

Crea un volumen persistente.

**docker volume ls**

Lista los volúmenes existentes.

**docker volume rm nombre**

Elimina un volumen (si no está en uso).

**docker run --mount type=volume,source=...,target=...**

Monta un volumen de forma explícita.

**docker run -v host:cont imagen**

Bind mount: enlaza una carpeta del host a una del contenedor.

## Redes

**docker network create nombre**

Crea una red para contenedores.

**docker network ls**

Lista redes existentes.

**docker network inspect nombre**

Muestra detalles de una red.

**docker run --network nombre imagen**

Inicia un contenedor dentro de una red.

## Docker Compose

### **docker-compose up [-d]**

Levanta todos los servicios definidos en docker-compose.yml.

### **docker-compose down [-v]**

Detiene y elimina contenedores, redes y volúmenes.

### **docker-compose stop**

Detiene servicios sin borrarlos.

### **docker-compose start servicio**

Inicia servicios detenidos.

### **docker-compose logs [-f]**

Muestra los logs de los servicios.

### **docker-compose exec servicio /bin/bash**

Abre shell dentro de un servicio.

### **docker-compose build**

Reconstruye las imágenes definidas en build.

## Docker Swarm

### **docker swarm init**

Inicializa un clúster Swarm.

### **docker swarm join --token ...**

Une un nodo al clúster.

### **docker node ls**

Lista los nodos que forman el clúster.

### **docker service ls**

Lista servicios en ejecución en el clúster.

### **docker service scale servicio=n**

Escala un servicio replicado.

### **docker stack deploy -c archivo.yml nombre**

Despliega un stack basado en Compose.