

REFERENCIA RÁPIDA DE DOCKER

Contenedores, imágenes, volúmenes, redes, compose

Fundamentos

Ejecutar Contenedores

```
docker run nginx # ejecutar imagen
docker run -d nginx # desacoplado (segundo plano)
docker run -p 8080:80 nginx # mapear puerto
docker run --name web nginx # contenedor con nombre
docker run -it ubuntu bash # shell interactivo
```

Comandos Esenciales

```
docker ps Listar contenedores en ejecución
docker ps -a Listar todos los contenedores (incluidos
detenidos)
docker images Listar imágenes locales
docker pull nginx Descargar imagen del registro
docker info Información del sistema
```

Gestión de Contenedores

Ciclo de Vida

```
docker start <id> Iniciar contenedor detenido
docker stop <id> Detener con gracia (SIGTERM)
docker kill <id> Detener forzado (SIGKILL)
docker restart <id> Reiniciar contenedor
docker rm <id> Eliminar contenedor detenido
docker rm -f <id> Eliminar forzado (incluso si está en
ejecución)
```

Inspección y Depuración

```
docker logs <id> Ver logs del contenedor
docker logs -f <id> Seguir logs (en vivo)
docker exec -it <id> bash Shell dentro de contenedor en
ejecución
docker inspect <id> Metadatos detallados del
contenedor (JSON)
docker top <id> Procesos en ejecución en el
contenedor
docker stats Uso de recursos en tiempo real
```

Copiar Archivos

```
docker cp file.txt <id>:/app/ # host - contenedor
docker cp <id>:/app/log.txt ./ # contenedor - host
```

Imágenes

Construir y Etiquetar

```
docker build -t myapp . # construir desde Dockerfile
docker build -t myapp:v2 . # con etiqueta
docker tag myapp user/myapp:v2 # re-etiquetar imagen
```

Publicar

```
docker login
docker push user/myapp:v2
docker pull user/myapp:v2
```

Gestión de Imágenes

```
docker images Listar todas las imágenes locales
docker rmi <image> Eliminar imagen
docker image prune Eliminar imágenes huérfanas
docker system prune Eliminar todos los datos no utilizados
docker history <image> Mostrar historial de capas de la imagen
```

Dockerfile

Instrucciones Comunes

```
FROM node:20 Imagen base
WORKDIR /app Establecer directorio de trabajo
COPY . . Copiar archivos a la imagen
RUN npm install Ejecutar comando durante la
construcción
CMD ["node", "app.js"] Comando por defecto en tiempo de
ejecución
EXPOSE 3000 Documentar puerto de escucha
ENV NODE_ENV=production Establecer variable de entorno
ARG VERSION=latest Variable de tiempo de construcción
ENTRYPOINT ["python"] Ejecutable fijo (CMD = argumentos)
```

Ejemplo de Dockerfile

```
FROM node:20-alpine
WORKDIR /app
COPY package*.json ./
RUN npm ci --production
EXPOSE 3000
CMD ["node", "server.js"]
```

Volúmenes

Almacenamiento Persistente

```
docker volume create mydata
docker run -v mydata:/app/data nginx
docker run -v $(pwd):/app nginx # bind mount
```

Comandos de Volumen

```
docker volume ls Listar volúmenes
docker volume inspect <v> Detalles del volumen
docker volume rm <v> Eliminar volumen
docker volume prune Eliminar volúmenes no utilizados
```

Redes

Fundamentos de Red

```
docker network create mynet
docker run --network mynet --name api nginx
docker run --network mynet --name db postgres
```

Comandos de Red

```
docker network ls Listar redes
docker network inspect <n> Detalles de la red
docker network connect <n> <c> Conectar contenedor a la
red
docker network rm <n> Eliminar red
```

Los contenedores en la misma red pueden alcanzarse por nombre

Docker Compose

Ejemplo de compose.yaml

```
services:
  web:
    build: .
    ports: ["3000:3000"]
    depends_on: [db]
  db:
    image: postgres:16
    environment:
      POSTGRES_PASSWORD: secret
    volumes: [pgdata:/var/lib/postgresql/data]
volumes:
  pgdata:
```

Comandos de Compose

```
docker compose up Iniciar todos los servicios
docker compose up -d Iniciar en segundo plano
docker compose down Detener y eliminar
contenedores
docker compose down -v También eliminar volúmenes
docker compose build Reconstruir imágenes
docker compose logs -f Seguir logs de todos los
servicios
docker compose ps Listar servicios en ejecución
docker compose exec web bash Shell en un servicio
```

Patrones Útiles

Comandos de Limpieza

```
docker system prune -a # eliminar todo lo no usado
docker container prune # eliminar contenedores detenidos
docker image prune -a # eliminar imágenes no usadas
```

Recetas Rápidas

```
Contenedor temporal `docker run --rm -it alpine sh`
Verificar puertos `docker port <id>`
Variables de entorno `docker run -e KEY=val image`
Archivo de entorno `docker run --env-file .env image`
Política de reinicio `docker run --restart unless-stopped
image`
Límite de recursos `docker run --memory 512m --cpus 1
image`
```