

# Referensi Cepat Docker

Container, image, volume, jaringan, compose

## Dasar

### Menjalankan Container

```
docker run nginx # run image
docker run -d nginx # detached (background)
docker run -p 8080:80 nginx # map port
docker run --name web nginx # named container
docker run -it ubuntu bash # interactive shell
```

### Perintah Utama

```
docker ps Tampilkan container yang berjalan
docker ps -a Tampilkan semua container (termasuk yang berhenti)
docker images Tampilkan image lokal
docker pull nginx Unduh image dari registry
docker info Informasi sistem secara keseluruhan
```

## Manajemen Container

### Siklus Hidup

```
docker start <id> Mulai container yang berhenti
docker stop <id> Hentikan dengan baik (SIGTERM)
docker kill <id> Paksa berhenti (SIGKILL)
docker restart <id> Restart container
docker rm <id> Hapus container yang berhenti
docker rm -f <id> Paksa hapus (meskipun berjalan)
```

### Inspeksi & Debugging

```
docker logs <id> Lihat log container
docker logs -f <id> Ikuti log (live)
docker exec -it <id> bash Masuk shell container yang berjalan
docker inspect <id> Metadata detail container (JSON)
docker top <id> Proses yang berjalan di container
docker stats Penggunaan resource secara live
```

### Menyalin File

```
docker cp file.txt <id>:/app/ # host → container
docker cp <id>:/app/log.txt ./ # container → host
```

## Image

### Build & Tagging

```
docker build -t myapp . # build from Dockerfile
docker build -t myapp:v2 . # with tag
docker tag myapp user/myapp:v2 # retag image
```

### Publish

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

### Manajemen Image

```
docker images Tampilkan semua image lokal
docker rmi <image> Hapus image
docker image prune Hapus image yang tidak terpakai (dangling)
docker system prune Hapus semua data yang tidak digunakan
docker history <image> Tampilkan riwayat layer image
```

## Dockerfile

### Instruksi Umum

```
FROM node:20 Image dasar
WORKDIR /app Atur direktori kerja
COPY . . Salin file ke image
RUN npm install Jalankan perintah saat build
CMD ["node", "app.js"] Perintah default saat runtime
EXPOSE 3000 Dokumentasikan port yang didengarkan
ENV NODE_ENV=production Atur environment variable
ARG VERSION=latest Variabel saat build
ENTRYPOINT ["python"] Executable tetap (CMD = argumen)
```

### Contoh Dockerfile

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

## Volume

### Penyimpanan Persisten

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

### Perintah Volume

```
docker volume ls Tampilkan volume
docker volume inspect <v> Detail volume
docker volume rm <v> Hapus volume
docker volume prune Hapus volume yang tidak terpakai
```

## Jaringan

### Dasar Jaringan

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

### Perintah Jaringan

```
docker network ls Tampilkan jaringan
docker network inspect <n> Detail jaringan
docker network connect <n> <c> Hubungkan container ke jaringan
docker network rm <n> Hapus jaringan
```

Container dalam jaringan yang sama dapat saling menjangkau berdasarkan nama

## Docker Compose

### Contoh 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:
```

## Perintah Compose

```
docker compose up Mulai semua service
docker compose up -d Mulai di background
docker compose down Hentikan dan hapus container
docker compose down -v Juga hapus volume
docker compose build Rebuild image
docker compose logs -f Ikuti log semua service
docker compose ps Tampilkan service yang berjalan
docker compose exec web bash Masuk shell ke sebuah service
```

## Pola Berguna

### Perintah Pembersihan

```
docker system prune -a # remove all unused
docker container prune # remove stopped
docker image prune -a # remove unused images
```

### Resep Cepat

```
Container sementara docker run --rm -it alpine sh
Cek port docker port <id>
Env var docker run -e KEY=val image
Env file docker run --env-file .env image
Restart policy docker run --restart unless-stopped image
Batas resource docker run --memory 512m --cpus 1 image
```