

# DOCKER THAM KHẢO NHANH

Container, image, volume, network, compose

## Cơ Bản

### Chạy 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
```

### Lệnh Thiết Yếu

```
docker ps Liệt kê container đang chạy
docker ps -a Liệt kê tất cả container (kể cả đã dừng)
docker images Liệt kê image cục bộ
docker pull nginx Tải image từ registry
docker info Thông tin toàn hệ thống
```

## Quản Lý Container

### Vòng Đời

```
docker start <id> Khởi động container đã dừng
docker stop <id> Dừng nhẹ nhàng (SIGTERM)
docker kill <id> Dừng cưỡng bức (SIGKILL)
docker restart <id> Khởi động lại container
docker rm <id> Xóa container đã dừng
docker rm -f <id> Xóa bắt buộc (kể cả đang chạy)
```

### Kiểm Tra & Debug

```
docker logs <id> Xem log container
docker logs -f <id> Theo dõi log theo thời gian thực
docker exec -it <id> bash Mở shell vào container đang chạy
docker inspect <id> Metadata chi tiết của container (JSON)
docker top <id> Tiến trình đang chạy trong container
docker stats Sử dụng tài nguyên theo thời gian thực
```

### Sao Chép File

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

## Image

### Build & Tag

```
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
```

### Quản Lý Image

```
docker images Liệt kê tất cả image cục bộ
docker rmi <image> Xóa image
docker image prune Xóa image rỗng (dangling)
docker system prune Xóa toàn bộ dữ liệu không dùng
docker history <image> Xem lịch sử layer của image
```

## Dockerfile

### Chỉ Thị Phổ Biến

```
FROM node:20 Image gốc
WORKDIR /app Đặt thư mục làm việc
COPY . . Sao chép file vào image
RUN npm install Chạy lệnh khi build
CMD ["node", "app.js"] Lệnh mặc định khi chạy
EXPOSE 3000 Khai báo port lắng nghe
ENV NODE_ENV=production Đặt biến môi trường
ARG VERSION=latest Biến tại thời điểm build
ENTRYPOINT ["python"] Executable cố định (CMD = tham số)
```

### Dockerfile Mẫu

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

## Volume

### Lưu Trữ Bền Vững

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

### Lệnh Volume

```
docker volume ls Liệt kê volume
docker volume inspect <v> Chi tiết volume
docker volume rm <v> Xóa volume
docker volume prune Xóa volume không dùng
```

## Network

### Cơ Bản Về Network

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

### Lệnh Network

```
docker network ls Liệt kê network
docker network inspect <n> Chi tiết network
docker network connect <n> <c> Gắn container vào network
docker network rm <n> Xóa network
```

Các container trong cùng network có thể kết nối nhau qua tên

## Docker Compose

### Ví Dụ 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:
```

### Lệnh Compose

```
docker compose up Khởi động tất cả service
docker compose up -d Khởi động ở chế độ nền
docker compose down Dừng và xóa container
docker compose down -v Đồng thời xóa volume
docker compose build Build lại image
docker compose logs -f Theo dõi log tất cả service
docker compose ps Liệt kê service đang chạy
docker compose exec web bash Mở shell vào một service
```

## Mẹo Hữu Ích

### Dọn Dẹp

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

### Công Thức Nhanh

```
Container tạm `docker run --rm -it alpine sh`
Kiểm tra port `docker port <id>`
Biến môi trường `docker run -e KEY=val image`
File env `docker run --env-file .env image`
Chính sách khởi động lại `docker run --restart unless-stopped image`
Giới hạn tài nguyên `docker run --memory 512m --cpus 1 image`
```