Riferimento Rapido Docker

Container, immagini, volumi, networking, compose

Basi

Eseguire 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
```

Comandi Essenziali

`docker ps`	Elenca container in esecuzione
`docker ps -a`	Elenca tutti i container (inclusi fermati)
`docker images`	Elenca immagini locali
`docker pull nginx`	Scarica immagine dal registro
`docker info`	Informazioni di sistema

Gestione Container

Ciclo di Vita

`docker start <id>`	Avvia un container fermo
`docker stop <id>`	Arresto corretto (SIGTERM)
`docker kill <id>`	Arresto forzato (SIGKILL)
`docker restart <id>`	Riavvia il container
`docker rm <id>`	Rimuove container fermo
`docker rm -f <id>`	Rimozione forzata (anche in esecuzione)

Ispezione e Debug

`docker logs <id>`	Visualizza log del container
`docker logs -f <id>`	Segui log (in tempo reale)
`docker exec -it <id> bash`	Shell nel container in esecuzione
`docker inspect <id>`	Metadati dettagliati del container (JSON)
`docker top <id>`	Processi in esecuzione nel container
`docker stats`	Utilizzo risorse in tempo reale

Copia File

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

Immagini

Build e Tagging

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

Pubblicazione

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

Gestione Immagini

`docker images`	Elenca tutte le immagini locali
`docker rmi <image>`	Rimuove immagine
`docker image prune`	Rimuove immagini dangling
`docker system prune`	Rimuove tutti i dati inutilizzati
`docker history <image>`	Mostra cronologia dei layer

Dockerfile

Istruzioni Comuni

`FROM node:20`	Immagine base
`WORKDIR /app`	Imposta directory di lavoro
`COPY . .`	Copia file nell'immagine
`RUN npm install`	Esegue comando durante la build
`CMD ["node", "app.js"]`	Comando default a runtime
`EXPOSE 3000`	Documenta la porta in ascolto
`ENV NODE_ENV=production`	Imposta variabile d'ambiente
`ARG VERSION=latest`	Variabile di build
`ENTRYPOINT ["python"]`	Eseguibile fisso (CMD = argomenti)

Esempio Dockerfile

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

Volumi

Storage Persistente

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

Comandi Volume

`docker volume ls`	Elenca volumi
`docker volume inspect <v>`	Dettagli del volume
`docker volume rm <v>`	Rimuove volume
`docker volume prune`	Rimuove volumi inutilizzati

Reti

Basi di Rete

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

Comandi di Rete

`docker network ls`	Elenca reti
`docker network inspect <n>`	Dettagli della rete
`docker network connect <n> <c>`	Collega container alla rete
`docker network rm <n>`	Rimuove rete

I container sulla stessa rete possono comunicare per nome

Docker Compose

Esempio 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:
```

Comandi Compose

`docker compose up`	Avvia tutti i servizi
`docker compose up -d`	Avvia in background
`docker compose down`	Ferma e rimuove i container
`docker compose down -v`	Rimuove anche i volumi
`docker compose build`	Ricostruisce le immagini
`docker compose logs -f`	Segui i log di tutti i servizi
`docker compose ps`	Elenca i servizi in esecuzione
`docker compose exec web bash`	Shell in un servizio

Pattern Utili

Comandi di Pulizia

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

Ricette Rapide

`Container temporaneo`	**`docker run --rm -it alpine sh`**
`Verifica porta`	**`docker port <id>`**
`Variabili d'ambiente`	**`docker run -e KEY=val image`**
`File env`	**`docker run --env-file .env image`**
`Restart policy`	**`docker run --restart unless-stopped image`**
`Limite risorse`	**`docker run --memory 512m --cpus 1 image`**