



Orchestration avec docker-compose

Sommaire

- Orchestration avec docker-compose
 - Objectif
 - Qu'est-ce qu'un service ?
 - Variables d'environnement
 - Exemple : API + base de données PostgreSQL
 - Commandes utiles

Objectif

Gérer plusieurs **services** (API, base de données, etc.) dans un **fichier unique** : `docker-compose.yml`.

Qu'est-ce qu'un service ?

Un **service** représente un container Docker avec sa configuration :

- image ou build
- ports à exposer
- variables d'environnement
- volumes
- dépendances éventuelles (`depends_on`)

Variables d'environnement

Deux types :

- **Variables prédéfinies** (ex: `POSTGRES_USER`, `POSTGRES_DB`, etc.)
- **Variables personnalisées** (ex: `DATABASE_URL`, `APP_MODE`, etc.)

Elles permettent de configurer le comportement des services **sans modifier le code**.

Exemple : API + base de données PostgreSQL

```
version: "3.8"

services:
  api:
    build: .
    ports:
      - "8000:8000"
    volumes:
      - ./app
    depends_on:
      - db
    environment:
      DATABASE_URL: postgresql://user:password@db:5432/dbname

  db:
    image: postgres:15
    environment:
      POSTGRES_USER: user
      POSTGRES_PASSWORD: password
      POSTGRES_DB: dbname
    volumes:
      - pgdata:/var/lib/postgresql/data

volumes:
  pgdata:
```

Commandes utiles

```
docker-compose up --build    # Lancer les services
docker-compose down -v      # Arrêter et supprimer les containers + volumes
```