

TP DevOps - Docker (Partie 1)

Amine M'zali

May 16, 2025

Réponses aux questions

1-1. Pourquoi vaut-il mieux utiliser `-e` pour les variables d'environnement ?

Car cela évite de stocker des informations sensibles (comme les mots de passe) directement dans l'image Docker, ce qui est une mauvaise pratique de sécurité. Avec `-e`, les variables sont injectées au moment de l'exécution.

1-2. Pourquoi a-t-on besoin d'un volume pour Postgres ?

Pour persister les données même lorsque le conteneur est supprimé. Le volume permet de conserver les données sur la machine hôte.

1-3. Dockerfile et commandes du conteneur Postgres

Dockerfile:

```
FROM postgres:17.2-alpine
ENV POSTGRES_DB=db \
    POSTGRES_USER=usr \
    POSTGRES_PASSWORD=pwd
```

Commandes:

```
docker build -t tp-database ./database
docker run --name postgres-db --network app-network \
    -v tp-db-data:/var/lib/postgresql/data -d tp-database
```

1-4. Pourquoi un multistage build ?

Pour séparer les étapes de compilation et d'exécution. Cela permet d'alléger l'image finale, en n'y gardant que le minimum (JRE).

Étapes du Dockerfile :

- Étape 1 (build): compile le code avec Maven et JDK
- Étape 2 (run): copie uniquement le `.jar` compilé et exécute avec le JRE

1-5. Pourquoi utiliser un reverse proxy ?

Pour centraliser les requêtes HTTP, exposer un seul point d'accès, masquer les services internes, et éventuellement gérer un futur front-end ou HTTPS.

1-6. Pourquoi Docker Compose est important ?

Il facilite le lancement, l'arrêt, la configuration et la mise à jour de plusieurs conteneurs en un seul fichier. Cela simule une vraie architecture de production.

1-7. Commandes Docker Compose importantes

```
docker compose build
docker compose up -d
docker compose down -v
docker compose ps
docker compose logs -f
```

1-8. Fichier docker-compose utilisé

```
services:
  database:
    build: ./database
    container_name: postgres-db
    environment:
      POSTGRES_DB: db
      POSTGRES_USER: usr
      POSTGRES_PASSWORD: pwd
    networks:
      - app-network
    volumes:
      - db-data:/var/lib/postgresql/data

  backend:
    build: ./backend-simple-api
    container_name: springboot-api
    depends_on:
      - database
    networks:
      - app-network
    ports:
      - "8080:8080"

  httpd:
    build: ./http
    container_name: apache-reverse-proxy
    ports:
      - "8088:80"
    depends_on:
      - backend
```

```
    networks:
      - app-network

networks:
  app-network:

volumes:
  db-data:
```

1-9. Commandes de publication DockerHub

```
docker login

docker tag tp-database simiamine/tp-database:1.0
docker push simiamine/tp-database:1.0


docker tag tp-backend simiamine/tp-backend:1.0
docker push simiamine/tp-backend:1.0


docker tag tp-httpd simiamine/tp-httpd:1.0
docker push simiamine/tp-httpd:1.0
```

1-10. Pourquoi publier ses images ?

Pour les partager facilement avec une équipe ou une autre machine, les utiliser dans des CI/CD, ou les réutiliser plus tard sans tout reconstruire localement.


Capture d'écran DockerHub




Amine
 Community User

Repositories Starred


Displaying 1 to 3 of 3 repositories



simiamine/tp-httpd
By [simiamine](#) · Updated 2 hours ago



simiamine/tp-backend
By [simiamine](#) · Updated 2 hours ago



simiamine/tp-database
By [simiamine](#) · Updated 2 hours ago