

# Présentation d'un environnement de développement Docker

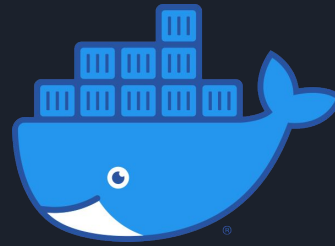


# Dépendances et outils de développement

- Virtualisation : Virtualbox 7
- OS : Ubuntu 22.04 Desktop
- BDD : MySQL
- Docker
- CMS : Wordpress
- IDE : Visual Studio Code



Virtualbox : outil de virtualisation.  
Permet d'**exécuter** un nombre illimité de machines virtuelles, avec pour seules limites l'espace disque et la mémoire de l'ordinateur.



**Docker est un système d'exploitation (ou environnement d'exécution) pour conteneurs**



Ubuntu : système d'exploitation GNU/Linux basé sur la distribution Debian, libre de droits et gratuit.



WORDPRESS

WordPress est un système de gestion de contenu gratuit, libre et open-source. Ce logiciel écrit en PHP repose sur une base de données MySQL et est distribué par la fondation WordPress.org



MySQL est un système qui **permet de créer et administrer une base de données et sur lequel on peut effectuer des requêtes SQL.**



Visual Studio Code est un éditeur de code extensible développé par Microsoft pour Windows, Linux et macOS



# I. Mise en place de la VM, dépendances et outils

Téléchargement de l'image de Ubuntu 22.04, interprétation de l'image par Virtualbox, 4go de Ram, 2 processeurs


OS vierge : installation des outils requis

Docker : toujours commencer par un `sudo apt update` pour MAJ la liste des paquets.

Ce qu'il se passe avec `apt update` : nous demandons à un repository de nous fournir le dernier paquet connu. Sans cette commande, risque de repo obsolète

Une fois Docker installé, on peut vérifier la version avec `docker -v`, qui valide aussi la procédure d'installation de l'outil

```
clem@clem:~/Documents/Mise-en-situation$ docker -v
Docker version 23.0.1, build a5ee5b1
```



Installation de docker compose avec apt install docker-compose, sinon pas d'interprétation des fichiers YAML par Docker

Docker Compose **est un outil destiné à définir et exécuter des applications Docker à plusieurs conteneurs**

Container déjà actif : docker ps

```
clem@clem:~/Documents/Mise-en-situation$ sudo docker ps
[sudo] Mot de passe de clem :
CONTAINER ID   IMAGE                COMMAND                  CREATED        STATUS        PORTS
505990084d42   wordpress:latest     "docker-entrypoint.s..." 29 minutes ago Up 29 minutes 0.0.0.0:8000->80/tcp, :::8000->80/tcp
ituation_wordpress_1
bf04ef7e1de8   mysql:5.7            "docker-entrypoint.s..." 29 minutes ago Up 29 minutes 3306/tcp, 33060/tcp
ituation_db_1
```


**wordpress:latest** la dernière image du CMS Wordpress qui a été pull sur le site officiel de Docker est active sur le port 8000 en local

Nous allons voir ensemble la création du fichier YAML ainsi que son interprétation par la commande docker-compose et nous allons pour cela supprimer les containers actifs et reprendre depuis le début

Je stop les container avec docker stop, je liste tous les containers même inactifs avec docker ps -a puis je les supprime avec docker rm <ID du container>, ou docker compose down : vérifier BDD


## II. Fichier YAML

```
docker-compose.yml X
docker-compose.yml
1  version: '3'
2
3  services:
4    #Database
5    db:
6      image: mysql:5.7
7      volumes:
8        - db_data:/var/lib/mysql
9      restart: always
10     environment:
11       MYSQL_ROOT_PASSWORD: admin
12       MYSQL_DATABASE: wordpress
13       MYSQL_USER: wordpress
14       MYSQL_PASSWORD: admin
15     networks:
```




Nous allons ensemble analyser et comprendre ce fichier YAML

- version: '3': Version de docker compose.
- services: Section décrivant les différents services qui seront exécutés pour l'application :
  - db: Premier service défini dans le fichier : BDD
  - image: mysql:5.7: Spécifie l'image Docker utilisée pour exécuter la base de données, MySQL 5.7.
  - volumes: - db\_data:/var/lib/mysql: Monte un volume (persistance des données) nommé "db\_data" pour stocker les données de la base de données dans le répertoire "/var/lib/mysql" à l'intérieur du conteneur.
  - restart: always: Conteneur toujours redémarré en cas de plantage.
- environment: Section qui définit les variables d'environnement nécessaires pour la base de données : mot de passe root, nom de la base de données, nom d'utilisateur et mot de passe.
- networks: Cette section définit les réseaux sur lesquels les services sont connectés; le service "db" est connecté au réseau "wpsite".

- 
- `wordpress`: Décrit le deuxième service qui est Wordpress
    - `depends_on: ...`: Indique que le service "wordpress" dépend du service "db"
    - `image: wordpress:latest`: Spécifie l'image Docker utilisée ainsi que sa version
    - `ports: - '8000:80'`: Relie le port 8000 de l'hôte à l'intérieur du conteneur au port 80 utilisé par WordPress.
    - `volumes: [ './:/var/www/html' ]`: Monte le répertoire courant du fichier YAML sur le répertoire `"/var/www/html"` à l'intérieur du conteneur.
  - `environment: ...`: Définit les variables d'environnement pour le service Wordpress (nom d'utilisateur, mot de passe...)
  - `networks: ...`: Cette section définit les réseaux sur lesquels les services sont connectés. Dans ce cas, le service "wordpress" est connecté au réseau "wpsite".
  - `networks: wpsite: ...`: Cette section définit le réseau "wpsite" qui est utilisé par les deux services.
  - `volumes: db_data: ...`: Cette section définit le volume nommé "db\_data" qui stocke les données de la base de données.





Maintenant que nous avons revu en détails le fonctionnement du fichier YAML, nous pouvons lancer la commande `docker-compose up -d`, qui start les containers en tâche de fond et les laisse tourner.

Le fichier va donc nous créer deux containers pour les deux services, Wordpress et MySQL.

```
clem@clem:~/Documents/Mise-en-situation$ sudo docker-compose up -d
[sudo] Mot de passe de clem :
Creating mise-en-situation_db_1 ... done
Creating mise-en-situation_wordpress_1 ... done
```

Comme vu précédemment, le port 8000 de l'hôte à l'intérieur du conteneur est relié au port 80 utilisé par WordPress. Nous pouvons donc accéder au container Wordpress sur localhost:8000

La base de données est pour l'instant vide : `show tables from wordpress;`

Faisons dorénavant un tour du côté du container de la base de données : l'installation de Wordpress a provoqué des écritures dans la base de données. Nous allons la parcourir avec MySQL.



### III. Base de données MySQL

Pour parcourir la base de données, nous utilisons la commande “sudo docker exec -it <nom cont bdd> bash”

Avec cette commande, nous entrons dans le container, nous ouvrons une session interactive ce qui va permettre d’entrer des commandes manuelles. Nous exécutons bash ce qui va nous permettre d’exécuter des commandes Linux.

mysql -p : nous exécutons mysql en entrant le mot de passe défini plus tôt

show databases : montre les BDD

show tables from wordpress : montre toutes les tables de la BDD wordpress



## IV. Fichier .env

```
1  MYSQL_PASSWORD=admin
2  MYSQL_DATABASE=wordpress
3  MYSQL_USER=wordpress
4  MYSQL_PASSWORD=admin
5
6  WORDPRESS_DB_HOST=db:3306
7  WORDPRESS_DB_USER=wordpress
8  WORDPRESS_DB_PASSWORD=admin|
```

# V. Git repo

main ▾

1 branch

0 tags

Go to file

Add file ▾

<> Code ▾



clementdebatisse modified docker compose

bbec3ba 7 hours ago

2 commits



docker-compose.yaml

modified docker compose

7 hours ago

Help people interested in this repository understand your project by adding a README.

Add a README



# Commandes

`sudo docker exec -it <nom cont bdd> bash`

`mysql -p`

`show grants for wordpress`

données en dure -> dans fichier env