LYCÉE ARISTIDE BERGÈS
SIO

Découverte

322 - Fonctionnement Docker-compose

Rédigé par

David ROUMANET
Professeur BTS SIO

Changement

Date	Révision

Sommaire

A Introduction

Dans l'activité précédente, nous avons pu découvrir la puissance de Docker et les subtilités pour communiquer avec les conteneurs.

Cependant, les deux inconvénients des commandes Docker sont :

- La longueur et la complexité d'une seule ligne
- L'impossibilité de lancer plusieurs conteneurs en une seule fois

A.1 Généralités

Docker-compose est donc un outil qui va permettre d'exécuter des configurations avec plusieurs conteneurs, avec une commande simple.

Pour cela, Docker-compose s'appuie sur un fichier au format YAML contenant toute la configuration nécessaire pour chaque conteneur.

L'intérêt est alors de ne lancer qu'une seule commande.

B Utilisation

Avant d'utiliser les fonctionnalités de Docker-compose, je recommande de vérifier son fonctionnement avec la commande `docker-compose --version` :

```
Docker Compose version 2.11.2
```

B.1 Syntaxe docker-compose.yml

La commande docker-compose up -d va lire et exécuter le contenu du fichier docker-compose.yml : connaître les éléments qui constituent ce fichier permettra de lire ou créer un tel fichier. Il existe de nombreuses instructions, mais voici celles qui sont les plus importantes :

- Le mot-clé `service:` permet de définir les différents conteneurs qui seront activés.
- L'usage du mot `environment:` permet de définir des variables d'environnement dans le conteneur
- L'instruction `volume:` met en œuvre le mécanisme de persistance des données

Voici un exemple de fichier permettant de démarrer deux conteneurs :

```
version: '3.3'

services:
   db:
     image: mysql:5.7
     volumes:
       - db_data:/var/lib/mysql
     restart: always
     environment:
       MYSQL_ROOT_PASSWORD: somewordpress
       MYSQL_DATABASE: wordpress
       MYSQL_USER: wordpress
       MYSQL_PASSWORD: wordpress

   wordpress:
     depends_on:
       - db
     image: wordpress:latest
     volumes:
       - web_data:/var/www/html
     ports:
       - "8000:80"
     restart: always
     environment:
       WORDPRESS_DB_HOST: db:3306
       WORDPRESS_DB_USER: wordpress
       WORDPRESS_DB_PASSWORD: wordpress
       WORDPRESS_DB_NAME: wordpress

volumes:
    db_data: {}
    web_data: {}
```

Nom choisi pour le conteneur

Image dans docker-Hub

Nom du volume et emplacement dans le conteneur

Variables d'environnement définissant les paramètres

Nom choisi pour l'autre conteneur

Redirection vers le port 8000 du port 80 du conteneur

Il est donc possible de décomposer une commande Docker dans un fichier docker-compose.yml, par exemple :

Docker	Docker-compose.yml
docker run -d --name mysqldb -p 3307:3306 -e MYSQL_ROOT_PASSWORD=password mysql	Version: '3' service: mysqldb: image: mysql port: - 3307:3306 environment: - MYSQL_ROOT_PASSWORD=password

B.2 Exécution des commandes

Le lancement des conteneurs implique que la commande `docker-composer up -d` soit lancé au même emplacement où se trouve le fichier `docker-compose.yml`.

Il suffit ensuite d'envoyer la commande `docker-composer down` pour arrêter l'ensemble des conteneurs.

C Exemple pratique : Wordpress

Nous allons tester la mise en place du CMS Wordpress : cette installation sera utile pour un module ultérieur.

C.1 Fichier de configuration

Créez dans un répertoire choisi, le répertoire Wordpress-Docker et créez le fichier suivant :

docker-compose.yml

```
version: '3'

services:
  # Database
  db:
    image: mysql:5.7
    volumes:
      - db_data:/var/lib/mysql
    restart: always
    environment:
      MYSQL_ROOT_PASSWORD: password
      MYSQL_DATABASE: wordpress
      MYSQL_USER: wordpress
      MYSQL_PASSWORD: wordpress
    networks:
      - wpsite
  # phpmyadmin
  phpmyadmin:
    depends_on:
      - db
    image: phpmyadmin/phpmyadmin
    restart: always
    ports:
      - '8080:80'
    environment:
      PMA_HOST: db
      MYSQL_ROOT_PASSWORD: password
    networks:
      - wpsite
  # Wordpress
  wordpress:
    depends_on:
      - db
    image: wordpress:latest
    ports:
      - '8000:80'
    restart: always
    volumes: ['./:/var/www/html']
    environment:
      WORDPRESS_DB_HOST: db:3306
      WORDPRESS_DB_USER: wordpress
      WORDPRESS_DB_PASSWORD: wordpress
    networks:
      - wpsite
networks:
  wpsite:
volumes:
  db_data:
```

Sauvegardez le fichier.

C.2 Compréhension du fichier

C.2.1 Quiz rapide

Vérifiez que vous comprenez le fichier en vous posant les questions suivantes :

- Quelle sera l'URL pour accéder au service de Wordpress ?
- Quelle sera l'URL pour accéder au service de PhpMyAdmin ?
- Quelle commande Docker me permet de voir les conteneurs actifs ?
- Quelle commande Docker me permet de trouver où sont stockées les données du conteneur pour la base de données ?

C.2.2 Exécution du fichier

Lancer la commande `docker-compose up` dans le répertoire contenant votre fichier.

C.2.3 Réponse au quiz

Le service Wordpress est accessible sur le port 8000, avec le lien http://localhost:8000

Le service PhpMyAdmin est utilisable sur le port 8080, avec le lien http://localhost:8080

Pour vérifier les conteneurs activés par docker-compose, vous pourrez utiliser dans une console séparée `docker ps`. Vous pourrez également vérifier que les noms sont logiques.

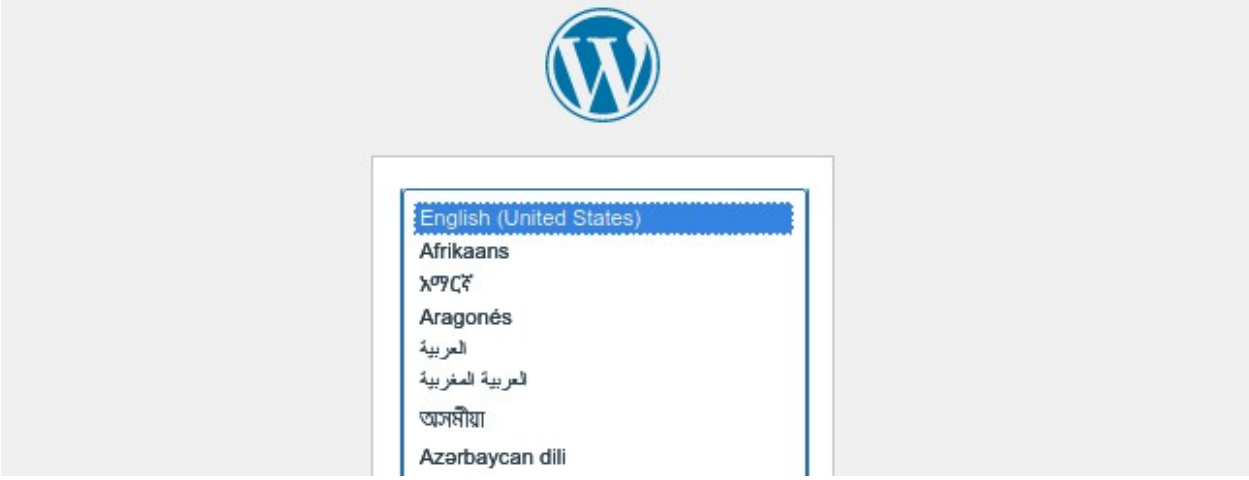
```
CONTAINER ID   IMAGE                 COMMAND                  CREATED          STATUS          PORTS                                      NAMES
3ed43b8b3378   phpmyadmin/phpmyadmin "/docker-entrypoint.…"   30 minutes ago   Up 30 minutes   0.0.0.0:8080->80/tcp, :::8080->80/tcp      wordpress-phpmyadmin-1
3321bf46c7f6   wordpress:latest      "docker-entrypoint.s…"   30 minutes ago   Up 30 minutes   0.0.0.0:8000->80/tcp, :::8000->80/tcp      wordpress-wordpress-1
af5bde1cd732   mysql:5.7             "docker-entrypoint.s…"   30 minutes ago   Up 30 minutes   3306/tcp, 33060/tcp                        wordpress-db-1
```

Pour connaître les volumes utilisés par les conteneurs, ce sera la commande `docker inspect wordpress-db-1`.

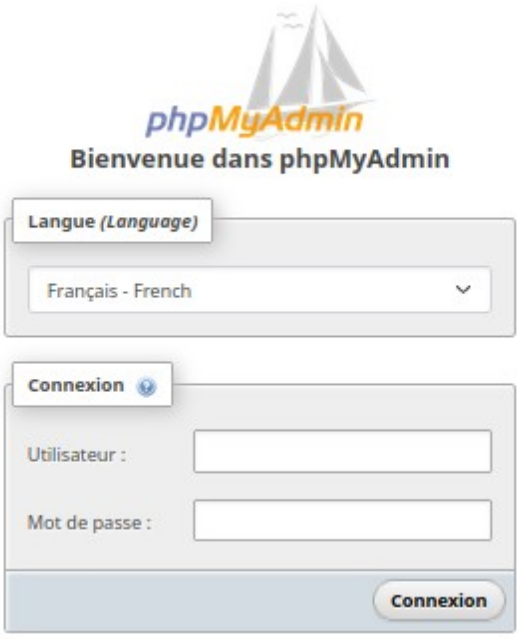
Elle devrait vous indiquer le répertoire /var/lib/mysql de votre machine (peut varier selon l'OS).

C.3 Vérification de fonctionnement

Connectez-vous sur le lien d'accès à WordPress (ne configurez rien), vous devriez obtenir ceci :



De même pour PhpMySql :



Enfin, le répertoire wordpress-docker doit contenir des fichiers :

Lors des différents clics et accès, la console ayant permis le lancement de la console doit également afficher de nouvelles informations, comme cet exemple :

```
...
wordpress-wordpress-1   | 172.18.0.1 - - [02/Nov/2022:09:33:35 +0000] "\x16\x03\x01\x02"
400 484 "-" "-"
wordpress-wordpress-1   | 172.18.0.1 - - [02/Nov/2022:09:33:42 +0000] "GET / HTTP/1.1"
302 405 "-" "Mozilla/5.0 (X11; Linux x86_64; rv:106.0) Gecko/20100101 Firefox/106.0"
wordpress-wordpress-1   | 172.18.0.1 - - [02/Nov/2022:09:33:42 +0000] "GET
/wp-admin/install.php HTTP/1.1" 200 4639 "-" "Mozilla/5.0 (X11; Linux x86_64; rv:106.0)
Gecko/20100101 Firefox/106.0"
wordpress-wordpress-1   | 172.18.0.1 - - [02/Nov/2022:09:33:43 +0000] "GET /wp-
includes/css/dashicons.min.css?ver=6.0.3 HTTP/1.1" 200 36068 "http://localhost:8000/wp-
admin/install.php" "Mozilla/5.0 (X11; Linux x86_64; rv:106.0) Gecko/20100101
Firefox/106.0"
...


...
wordpress-phpmyadmin-1  | 172.18.0.1 - - [02/Nov/2022:10:01:14 +0000] "GET / HTTP/1.1"
200 6540 "-" "Mozilla/5.0 (X11; Linux x86_64; rv:106.0) Gecko/20100101 Firefox/106.0"
wordpress-phpmyadmin-1  | 172.18.0.1 - - [02/Nov/2022:10:01:14 +0000] "GET
/themes/pmahomme/jquery/jquery-ui.css HTTP/1.1" 200 8823 "-" "Mozilla/5.0 (X11; Linux
x86_64; rv:106.0) Gecko/20100101 Firefox/106.0"
wordpress-phpmyadmin-1  | 172.18.0.1 - - [02/Nov/2022:10:01:14 +0000] "GET
/js/vendor/codemirror/lib/codemirror.css?v=5.2.0 HTTP/1.1" 200 2847 "-" "Mozilla/5.0
(X11; Linux x86_64; rv:106.0) Gecko/20100101 Firefox/106.0"
wordpress-phpmyadmin-1  | 172.18.0.1 - - [02/Nov/2022:10:01:14 +0000] "GET
/js/vendor/codemirror/addon/hint/show-hint.css?v=5.2.0 HTTP/1.1" 200 660 "-"
"Mozilla/5.0 (X11; Linux x86_64; rv:106.0) Gecko/20100101 Firefox/106.0"
wordpress-phpmyadmin-1  | 172.18.0.1 - - [02/Nov/2022:10:01:14 +0000] "GET
/js/vendor/codemirror/addon/lint/lint.css?v=5.2.0 HTTP/1.1" 200 1634 "-" "Mozilla/5.0
(X11; Linux x86_64; rv:106.0) Gecko/20100101 Firefox/106.0"
...
```

Notez que les échanges HTTP sont lisibles.

C.4 Arrêt des conteneurs

Après avoir tapé la commande `docker-compose down` vos conteneurs doivent se trouver à l'arrêt (les URL ne sont plus accessibles).

```
[+] Running 4/4
 ⠿ Container wordpress-wordpress-1   Removed
 ⠿ Container wordpress-phpmyadmin-1  Removed
 ⠿ Container wordpress-db-1          Removed
 ⠿ Network wordpress_wpsite          Removed
```

Cette découverte s'arrête ici : il y a beaucoup plus à apprendre mais le programme de BTS SIO est dense. La connaissance de Docker et Docker-compose est actuellement un avantage. Vous pouvez travailler des certifications complémentaires ou des cours en ligne.

Docker a des alternatives dont Redhat Podman, Kubernetes et LXC.

D Annexes

D.1 Commandes docker-compose

```
build              Build or rebuild services
bundle             Generate a Docker bundle from the Compose file
config             Validate and view the Compose file
create             Create services
down               Stop and remove containers, networks, images, and volumes
events             Receive real time events from containers
exec               Execute a command in a running container
help               Get help on a command
images             List images
kill               Kill containers
logs               View output from containers
pause              Pause services
port               Print the public port for a port binding
ps                 List containers
pull               Pull service images
push               Push service images
restart            Restart services
rm                 Remove stopped containers
run                Run a one-off command
scale              Set number of containers for a service
start              Start services
stop               Stop services
top                Display the running processes
unpause            Unpause services
up                 Create and start containers
version            Show the Docker-Compose version information
```