

Procédure installation Docker avec Apache, PHP et MYSQL

Qu'est-ce qu'un Docker

Docker est une plateforme logicielle open source qui permet aux développeurs de créer, déployer et exécuter des applications dans des conteneurs logiciels isolés.

Les conteneurs Docker sont similaires aux machines virtuelles, mais ils sont plus légers et plus flexibles.

Les conteneurs Docker contiennent tous les éléments nécessaires pour exécuter une application, y compris le code, les bibliothèques, les dépendances et les fichiers de configuration.

Prérequis, installer VMWare Workstation Pro, avoir une VM Debian fonctionnelle

Pour commencer, passer en root pour avoir les droits administrateurs

```
debian@debian10:~$ su -  
Password:
```

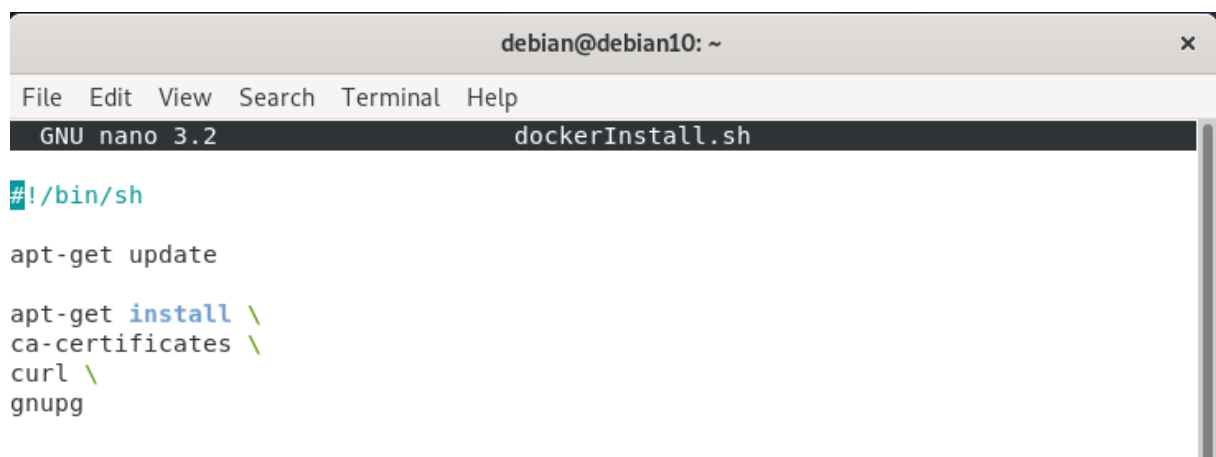
Nous allons installer docker en utilisant un repository

Nous allons donc faire un script Shell « dockerInstall.sh » pour automatiser tout cela

```
root@debian10:~# nano dockerInstall.sh
```

Cette commande ouvre un fichier dans lequel nous pourrons commencer à faire notre script

D'abord faire la mise à jour de l'index des paquets apt et des paquets d'installation pour permettre à apt d'utiliser un dépôt via HTTPS



```
debian@debian10: ~  
File Edit View Search Terminal Help  
GNU nano 3.2 dockerInstall.sh  
#!/bin/sh  
  
apt-get update  
  
apt-get install \  
ca-certificates \  
curl \  
gnupg
```

Ensuite ajout de la clé GPG officielle de Docker

```
sudo mkdir -m 0755 -p /etc/apt/keyrings
curl -fsSL https://download.docker.com/linux/debian/gpg | \
gpg --dearmor -o /etc/apt/keyrings/docker.gpg
```

Ensuite configurer le référentiel

```
echo \
"deb [arch=$(dpkg --print-architecture)] signed-by=/etc/apt/keyrings/docker.gpg$
https://download.docker.com/linux/debian \
"$(. /etc/os-release && echo "$VERSION_CODENAME")" stable" | \
sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
```

Mettre à jour les paquets

(chmod si problème lors de la mise à jour)

Pour commencer l'installation de Docker engine, containerd et Docker Compose

```
sudo apt-get update

sudo chmod a+r /etc/apt/keyrings/docker.gpg
sudo apt-get update

sudo apt-get install docker-ce docker-ce-cli containerd.io docker-buildx-plugin$
```

On vérifie ensuite que l'installation de Docker Engine est réussie en exécutant la commande `hello-world`

Ensuite on fait en sorte que l'utilisateur (user) de la machine doit pouvoir lancer la commande `docker` sans « `sudo` »

```
docker run hello-world

groupadd docker

usermod -aG docker debian

chown "debian":"debian" /home/"debian"/.docker -R
chmod g+rw /home/julien/.docker -R

systemctl enable docker.service
systemctl enable containerd.service
```

Puis activer le lancement automatique de(s) service(s) Docker au démarrage (systemctl)

On installe Docker en utilisant le script « dockerInstall.sh »

`./dockerInstall.sh`

On vérifie ensuite que docker fonctionne correctement avec la commande `docker run hello-world`

```
root@debian10:~# docker run hello-world

Hello from Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
 1. The Docker client contacted the Docker daemon.
 2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
    (amd64)
 3. The Docker daemon created a new container from that image which runs the
    executable that produces the output you are currently reading.
 4. The Docker daemon streamed that output to the Docker client, which sent it
    to your terminal.

To try something more ambitious, you can run an Ubuntu container with:
$ docker run -it ubuntu bash

Share images, automate workflows, and more with a free Docker ID:
https://hub.docker.com/

For more examples and ideas, visit:
https://docs.docker.com/get-started/
```

On télécharge ensuite le conteneur « `ubuntu/apache2:2.4-20.04_beta` »

```
root@debian10:~# docker pull ubuntu/apache2:2.4-20.04_beta
2.4-20.04_beta: Pulling from ubuntu/apache2
0c7ba809c292: Downloading 16.77MB/27.5MB
0c7ba809c292: Download complete
4d8d5052b37c: Downloading 26.56MB/33.95MB
0c7ba809c292: Pull complete
4d8d5052b37c: Pull complete
8a95a8e476a2: Pull complete
Digest: sha256:d9795421a067ad116c4e261f6694828d98f5a6ffb8989284e4dd658183d57686
Status: Downloaded newer image for ubuntu/apache2:2.4-20.04_beta
docker.io/ubuntu/apache2:2.4-20.04_beta
```

On peut ensuite créer le répertoire `/var/www/html_docker`

```
root@debian10:~# mkdir -p /var/www/html_docker
```

Nous pouvons ensuite lancer le conteneur avec la commande suivante :

```
docker run --name serverphp -d -v /var/www/html_docker:/var/www/html -p 8001:80  
ubuntu/apache2:2.4-20.04_beta /usr/sbin/apache2ctl -DFOREGROUND
```

```
root@debian10:~# docker run --name serverphp -d -v /var/www/html_docker:/var/www/html -p 192.168.139.135:8001:80 ubuntu/apache2:2.4-20.04_beta /usr/sbin/apache2ctl -DFOREGROUND  
f2c6daaab04ac2ce0fbbca0d3f2c4f34e1c48fb026daf599a1a4f73b23188262
```

Par la suite, nous allons ajouter ServerName 127.0.0.1 à la fin du fichier /etc/apache2/apache2.conf (de serverphp), pour éviter les bugs.

Pour ce faire, il faut se connecter au conteneur avec la commande : docker exec -it serverphp /bin/bash.

```
root@debian10:~# docker exec -it serverphp /bin/bash  
root@f2c6daaab04a:/# ls  
bin    dev    home  lib32  libx32  mnt    proc  run    srv    tmp    var  
boot   etc    lib   lib64  media   opt    root  sbin   sys    usr
```

Nous pouvons donc maintenant, accéder au fichier et ajouter la ServerName 127.0.0.1 à la fin du fichier grâce aux doubles « >> »

Nous pouvons cat le fichier, pour vérifier que la ligne soit bien ajoutée

```
root@f2c6daaab04a:/etc/apache2# echo "ServerName 127.0.0.1" >>apache2.conf  
root@f2c6daaab04a:/etc/apache2# cat apache2.conf
```

Ensuite, il faut générer une image de serverphp

Puis nous pouvons maintenant lancer le nouveau conteneur

```
root@debian10:~# docker commit serverphp serverphpimg:01  
sha256:1735e0249963b1648a2a9d6e4ef47ce065e4f0e177fbd4116f3a4eac0a7c2168  
root@debian10:~# docker run --name serverphp01 -d -v /var/www/html_docker:/var/www/html -p 192.168.139.135:8001:80 serverphpimg:01 /usr/sbin/apache2ctl -DFOREGROUND
```

Nous sommes maintenant aptes à installer les paquets php8.2 et php8.2-mysql.

Pour ce faire, nous commençons par se connecter au conteneur

Je suis passé du port 8001 au port 8080 mais grosse erreur

```
root@debian10:~# docker container rm a88c6cf29a9bb5f83172f2488bec4da605f15c5ba5f8f4c86fc22ba8ef6a052
a88c6cf29a9bb5f83172f2488bec4da605f15c5ba5ff8f4c86fc22ba8ef6a052
root@debian10:~# docker run --name serverphp01 -d -v /var/www/html_docker:/var/www/html -p 192.168.139.135:8080:80 serverphpimg:01 /usr/sbin/apache2ctl -DFOREGROUND
42746fb7229aa00b3bb8dd8895dd01a75c90a950805e83eb1e7a1502a288f389
root@debian10:~# docker exec -it serverphp01 /bin/bash
root@42746fb7229a:/#
```

Du coup j'ai du tout recommencer

On met ensuite à jour et à niveau les paquets

```
root@42746fb7229a:/# apt update && apt upgrade -y
```

Maintenant, nous procédons à l'installation de PHP avec les commandes suivantes :

```
root@9e10418d6d98:/# apt install software-properties-common
```

```
root@9e10418d6d98:/# add-apt-repository ppa:ondrej/php
```

```
root@9e10418d6d98:/# apt update
```

apt install php8.2 -y

```
root@9e10418d6d98:/# php --version
PHP 8.2.5 (cli) (built: Apr 14 2023 04:26:42) (NTS)
Copyright (c) The PHP Group
Zend Engine v4.2.5, Copyright (c) Zend Technologies
with Zend OPcache v8.2.5, Copyright (c), by Zend Technologies
```

On redémarre ensuite le conteneur :

```
root@9e10418d6d98:/# exit
exit
root@debian10:~# docker stop serverphp01
serverphp01
root@debian10:~# docker start serverphp01
serverphp01
```

Maintenant, nous allons installer le conteneur Mysql avec la commande suivante :

```
root@debian10:~# docker pull mysql
Using default tag: latest
latest: Pulling from library/mysql
328ba678bf27: Pull complete
f3f5ff008d73: Pull complete
dd7054d6d0c7: Pull complete
70b5d4e8750e: Pull complete
cdc4a7b43bdd: Pull complete
a0608f8959e0: Pull complete
5823e721608f: Pull complete
a564ada930a9: Pull complete
539565d00e89: Pull complete
a11a06843fd5: Pull complete
92f6d4aa041d: Pull complete
Digest: sha256:a43f6e7e7f3a5e5b90f857fbed4e3103ece771b19f0f75880f767cf66bbb6577
Status: Downloaded newer image for mysql:latest
docker.io/library/mysql:latest
```

Ensuite créons les répertoires : /u01/mysql/datadir

```
root@debian10:~# mkdir -p /u01/mysql/datadir
```

Nous pourrions donc lancer ce conteneur avec la commande suivante

```
root@debian10:~# docker run --name mysqlDB -e MYSQL_ROOT_PASSWORD=debian -d -v /
u01/mysql/datadir:/var/lib/mysql -p 192.168.139.135:3306:3306 mysql
c878fad6b3c856fd20e9dc901485f527dc096fa7d8c552a9f8725f082524c969
```

Se connecter au conteneur

Puis se connecter à Mysql

```
root@debian10:~# docker exec -it mysqlDB /bin/bash
bash-4.4# mysql -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 8
Server version: 8.0.33 MySQL Community Server - GPL

Copyright (c) 2000, 2023, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql>
```

On peut voir les bases de données avec la commande suivante

```
mysql> SHOW DATABASES;
+-----+
| Database |
+-----+
| information_schema |
| mysql |
| performance_schema |
| sys |
+-----+
4 rows in set (0.18 sec)
```

Sélectionner mysql

```
mysql> use mysql
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A
```

Database changed

Nous pouvons donc maintenant créer la table user01

```
mysql> CREATE TABLE user01 (id int, identifiant varchar(30), password varchar(30));
Query OK, 0 rows affected (0.25 sec)
```

J'ai ajouté un utilisateur à notre table user01

```
mysql> insert into user01 (id, identifiant, password) values (1, 'user', 'prevert');
Query OK, 1 row affected (0.03 sec)
```

Tapper exit puis exit pour sortir du mysql 2 fois

Nous avons maintenant un environnement de développement Docker avec Apache, PHP et MySQL sur une machine Debian

FIN