



Nous traiterons ici de la mise en place d'un serveur **LAMP** (Linux Apache MySQL PHP) au moyen de Docker. Pour cette **stack** (*plusieurs conteneurs empilés et connectés*) nous avons besoin d'une image d'**Apache** avec **PHP**, une image pour **MySQL** et une image pour **phpMyAdmin**.

- Pour **Apache/PHP** nous allons choisir l'[image fournie officiellement par PHP](#)
- Pour **MySQL**, nous pouvons prendre la dernière [image fournie officiellement par MySQL](#)
- Pour **phpMyAdmin**, nous pouvons aussi choisir l'[image officielle](#).

Par la suite nous allons nous servir d'un outil bien pratique pour lancer et connecter plusieurs conteneurs (*la stack*) entre eux avec : **docker-compose**

Cette solution nous permet de configurer la stack en évitant les lignes de commandes à rallonge sur le terminal.

Ouvrir le **terminal Windows** en mode Admin et lancer la commande :

L'aide doit s'afficher, sinon installer « *docker-compose* »

```
docker-compose -h
```

- Création des répertoires pour le contenu que l'on souhaite modifier et conserver, en l'occurrence les fichiers du site (la racine **www**) et les bases de données :

```
mkdir c:\lamp && cd c:\lamp
```

```
md www db-data apache php www\dev
```

Depuis le **CLOUD/RESSOURCES/lamp** : déposer le fichier **php.ini** dans le dossier **php** créée à l'instant.

- Nous allons ensuite créer un fichier vide au format **YAML**, nommé :
« **docker-compose.yml** »

```
type NUL > docker-compose.yml
```

- dans lequel nous allons définir notre stack LAMP :

```
#
# YAML : version:'2'
# WEBDEVOPS PHP APACHE (inclus l'ensemble des extensions PHP compilées), doc :
# https://dockerfile.readthedocs.io/en/latest/content/DockerImages/dockerfiles/php-apache.html
#
services:
  web:
    container_name: lamp-web
    image: webdevops/php-apache:8.1
    ports:
      - "8080:80"
    volumes:
      - .\www:/var/www/html
      - .\php:/usr/local/etc/php
      #Pour les sous-nom d'hôtes :
      # - ./apache/vhost.conf:/opt/docker/etc/httpd/vhost.conf
    environment:
      #Pour les fichiers .HTACCESS :
      - ALLOW_OVERRIDE=true
      - WEB_DOCUMENT_ROOT=/var/www/html
    links:
      - db:db
```

```
db:
  container_name: lamp-db
  image: mysql:8.4
  volumes:
    - ./db-data/mysql:/var/lib/mysql
    - ./db-data/dump:/docker-entrypoint-initdb.d
    - ./db-data/conf:/etc/mysql/conf.d
  ports:
    - "3306:3306"
  environment:
    #Par défaut le nom de l'utilisateur MySQL est : root
    - MYSQL_ROOT_PASSWORD=secret
myadmin:
  container_name: lamp-myadmin
  image: phpmyadmin/phpmyadmin:latest
  ports:
    - "8000:80"
```

ATTENTION A L'INDENTATION : les **tabulations** retourneront des erreurs sur les fichiers YAML ou « *yamèl* » (.yaml). Le format de fichier doit être LF (Unix) / UTF8

Unix (LF)

UTF-8

Sur Notepad++ : Menu->Edition->Convertir les sauts à la ligne->Convertir au format UNIX

EXPLICATION DE LA CONFIGURATION

Les **services** sont les conteneurs Dockerisés, **web**, **db** et **myadmin** sont les noms des services qu'on décide attribuer. Ces noms sont utilisés pour créer des liens entre les différents conteneurs. Par ex. **db** signifie que notre conteneur **db** (du nom de notre conteneur MySQL) correspondra à l'hôte **db** dans notre conteneur **web**. Pour se connecter au serveur MySQL il faudra donc entrer **db** comme nom d'hôte. Chaque conteneur on une « **image** » officielle disponible sur le cloud de Docker (*DockerHub*). Cette image sera déployée (*Pull*) dans le conteneur.

Les **variables d'environnements** « **environment** » sont disponibles dans la documentation officielle du conteneur. Par exemple le mot de passe de l'utilisateur MySQL **root** est **secret**.

De la même manière que l'option **-v** de la ligne de commande, le paramètre « **volumes** » relie les répertoires locaux **c:\lamp\www** et **c:\lamp\db-data\mysql** aux répertoires **/var/www/html** de l'image Apache/PHP et **/var/lib/mysql** de l'image MySQL dans nos conteneurs.

Et le paramètre « **ports** » - de la même manière que l'option **-p**, relie les ports qui nous intéressent de nos conteneurs aux ports de notre machine locale. Ici le port 80 (HTTP) et le port 3306 (MySQL).

- Une fois ce fichier mis en place on peut lancer tous nos conteneurs avec la commande :

```
docker-compose up -d
```

(*up = create, update and start, -d= detach*)

Il faut attendre quelques minutes afin que la stack de LAMP soit téléchargée et déployée (*pull*).

Vérification des conteneurs actifs :

```
docker ps -a
```

Notre serveur LAMP est en route :

```
PS C:\lamp> docker ps -a
CONTAINER ID   IMAGE                                COMMAND
NAMES
76f0cd09f89a   phpmyadmin/phpmyadmin:latest        "/docker-entrypoint..."
lamp-myadmin
97aeace5f41c   php/php-apache:8.1                  "/entrypoint supervi..."
lamp-web
927f4700ee49   mysql:8.4                            "docker-entrypoint.s..."
lamp-db
```

Pour tester le bon fonctionnement du serveur LAMP, on crée une page test dans le dossier « **www** » :

```
type NUL > c:\lamp\www\index.php
```

On ajoute quelques lignes de test :

```
<html><head><title>Hello</title></head>
<body>
<h1>Hello LAMP Container !</h1>
<hr><strong><? = 'PHP Version : ' . phpversion() ?></strong>
</body>
</html>
```

On vérifie sur un navigateur l'IP du serveur, ou le nom d'hôte par défaut :

<http://localhost:8080>



Bienvenue dans phpMyAdmin

Avec **PhpMyAdmin** : [ID : **root** | pass : **secret**] :

<http://localhost:8000>

Langue (Language)

Français - French

Gestion d'un sous-domaine depuis le serveur Apache conteneurisé

Afin de pouvoir gérer les sous-noms d'hôte d'Apache sur VirtualHost, on copie le fichier **vhost.conf** présent sur l'image du serveur « *php:apache* » :

```
lamp> docker ps -a
CONTAINER ID   IMAGE                                COMMAND
09f89a        phpmyadmin/phpmyadmin:latest       "/docker-entrypoint..."
5f41c        php/php-apache:8.1                 "/entrypoint supervi..."
tcp         lamp-web                             "docker-entrypoint.s..."
00ee49        mysql:8.4                           "docker-entrypoint.s..."
lamp-db
```

```
docker cp lamp-web:/opt/docker/etc/httpd/vhost.conf f c:\lamp\apache
```

Modification de la configuration par défaut enregistrée dans le répertoire
« c:\lamp\apache**vhost.conf** » :

```
# (...) CODE / CONFIGURATIONS D'ORIGINE - NE PAS AJOUTER CETTE LIGNE !! #

#####
# Directive du sous-nom d'hôte : dev.localhost :
#####
<VirtualHost *:80>
    ServerName dev.localhost
    DocumentRoot /var/www/html/dev

    <Directory "/var/www/html/dev">
        #Options +FollowSymLinks +Index
        AllowOverride All
        Require all granted
    </Directory>
</VirtualHost>
```

Mettre à jour cette portion à la fin du document : vhost.conf

Modification du document yaml :

- Décommentez la **ligne 16** :

```
(...)  
- ./apache/vhost.conf:/opt/docker/etc/httpd/vhost.conf  
(...)
```

On arrête la stack LAMP :

```
docker-compose down
```

On ajoute une page test dans le répertoire « c:\lamp\www\dev » :

```
type NUL > c:\lamp\www\dev\index.php
```

```
<?php  
/*** PAGE TEST ***/  
phpinfo();
```

On modifie le fichier « hosts » afin de rediriger le sous-domaine *dev.localhost* vers l'IP locale dans le répertoire :

- **C:\WINDOWS\system32\drivers\etc\hosts**
- On ajoute la ligne : **127.0.0.1 dev.localhost**

Depuis le répertoire : RESSOURCES/lamp, ajouter le fichier **php.ini** dans :

- C:\lamp\php\php.ini

Ce fichier peut être modifié afin d'ajouter ou désactiver des extensions, ou d'autres paramètres spécifiques au besoin de nos applications Web PHP.

Après les modifications de la configuration d'Apache ou du php.ini, on relance le serveur :

```
docker-compose up --build --force-recreate -d
```

```
Starting lamp_db  
Recreating lamp_web  
Starting lamp_myadmin
```

A noter qu'à ce stade on peut vérifier le contenu de chaque conteneur par la commande :

- `docker-compose exec nom_du_service bash`


Exemple dans **docker-compose.yml** : "services: web:" **web** est le nom du service !
(ceci appliqué également pour : **db** ou **myadmin**).

- On teste le sous-domaine : <http://dev.localhost:8080>

PHP 8.1.31 - phpinfo()

← → ↻ ⓘ http://dev.localhost:8080

PHP Version 8.1.31



System	Linux 42b3529ae18b 5.15.167.4-microsoft-standard-WSL2 #1 SMP Tue Nov 5 00:21:55 UTC 2024 x86_64
Build Date	Jan 14 2025 02:33:32
Build System	Linux - Docker
Build Provider	https://github.com/docker-library/php
Configure Command	'./configure' '--build=x86_64-linux-gnu' '--with-config-file-path=/usr/local/etc/php' '--with-config-file-scan-dir=/usr/local/etc/php/conf.d' '--enable-option-checking=fatal' '--with-mhash' '--with-pic' '--enable-ftp' '--enable-mbstring' '--enable-mysqlnd' '--with-password-argon2' '--with-sodium=shared' '--with-pdo-sqlite=/usr' '--with-sqlite3=/usr' '--with-curl' '--with-iconv' '--with-openssl' '--with-readline' '--with-zlib' '--disable-phpdbg' '--with-pear' '--with-libdir=lib/x86_64-linux-gnu' '--disable-cgi' '--enable-fpm' '--with-fpm-user=www-data' '--with-fpm-group=www-data' 'build_alias=x86_64-linux-gnu'
Server API	FPM/FastCGI
Virtual Directory Support	disabled
Configuration File (php.ini) Path	/usr/local/etc/php
Loaded Configuration File	/usr/local/etc/php/php.ini
Scan this dir for additional .ini files	/usr/local/etc/php/conf.d
Additional .ini files parsed	(none)
PHP API	20210902
PHP Extension	20210902
Zend Extension	420210902
Zend Extension Build	API420210902,NTS
PHP Extension Build	API20210902,NTS
Debug Build	no
Thread Safety	disabled

- Pour finir on arrête la stack LAMP au besoin :

```
docker-compose down
```

- Et si l'on souhaite la relancer :

```
cd c:\lamp && docker-compose up -d
```

Cours confectionné d'après l'article Ubuntu : Docker&LAMP : https://doc.ubuntu-fr.org/docker_lamp