Installation VM

Tout savoir sur la fameuse erreur du gestionnaire de paquets **apt** :

```
Could not get lock /var/lib/dpkg/lock Error
```

Voir dpkg lock error

Démarrage VM

Démarrer votre VM depuis https://servers1.sts-sio-caen.info/

Les informations de connexion au serveur sont données sur le drive.

Démarrer votre VM

Connexion SSH

Démarrer **Putty** pour vous connecter en SSH à votre VM.

Entrer l'adresse publique du serveur : `149.202.77.193` et le numéro de port correspondant à votre machine : 78xx

Identifiant de connexion à la VM :

- login : `user`
- password : **ilovedev**

Apache + PHP 8.4

Connexion en root :

```
su -l
```

Le mot de passe est toujours **ilovedev**

Ajout du ppa Sury :

Ajout de la clé gpg :

```
apt -y install lsb-release apt-transport-https ca-certificates
```

```
wget -O /etc/apt/trusted.gpg.d/php.gpg https://packages.sury.org/php/apt.gpg
```

Ajout du ppa :

```
echo "deb https://packages.sury.org/php/ $(lsb_release -sc) main" | tee
/etc/apt/sources.list.d/php.list
```

Installation

Vérifier les mises à jour :

```
apt update
```

Mettre à jour le système et rebooter :

```
apt upgrade -y && reboot
```

Après reconnexion à la VM avec putty :

```
apt install php8.4 php8.4-cli php8.4-{curl,bz2,mbstring,intl,yaml} libapache2-mod-
php
```

Git

```
apt install git
git version
```

VirtualHost

Clone du repository git

Cloner votre repository dans le root d'apache :

```
cd /var/www/html
git clone https://github.com/repositoryName
```

Création VHost

Créer le fichier **/etc/apache2/sites-available/mysite.conf**

```
<VirtualHost *:80>
        DocumentRoot /var/www/html/repositoryFolder
        ServerName srv1-vm-xxxx.sts-sio-caen.info
        DirectoryIndex /index.php

        <Directory /var/www/html/repositoryFolder>
            Require all granted
        </Directory>

        ErrorLog /var/log/apache2/mysite_error.log
        CustomLog /var/log/apache2/mysite_access.log combined
</VirtualHost>
```

Activer le site et recharger Apache :

```
a2ensite mysite
systemctl reload apache2
```

Résolution des erreurs

Il est probable que votre site vous délivre dans un premier temps une `erreur 500` :

c'est votre code source qui en est à l'origine, il vous faut trouver la ou les erreurs, en examinant les logs apache relatifs à votre site :

Ils sont situés dans le fichier que vous avez spécifié dans la configuration du vhost :

```
/var/log/apache2/mysite_error.log
```

Erreur courantes

Redéclaration

```

```

```
PHP Fatal error:  Cannot redeclare yaml_parse_file()
```

Diagnostic :

L'extension **yaml** est installée avec le php de la VM. La fonction déclarée dans le fichier **yaml** ne doit donc plus l'être (ou de manière conditionnelle).

Résolution :

```
use Symfony\Component\Yaml\Yaml;
use Symfony\Component\Yaml\Exception\ParseException;

if (!function_exists('yaml_parse_file')) {
    function yaml_parse_file($file){
        try {
            return Yaml::parseFile($file);
        } catch (ParseException $exception) {
            printf('Unable to parse the YAML string: <b>%s</b>',
$exception->getMessage());
        }
    }
}
```

Chemins

```
Warning: include(/test/assets/header.php) [function.include]: failed to open
stream: No such file or directory
```

Diagnostic :

Le chemin n'a pas été trouvé pour le fichier à inclure dans le script php...

Résolution :

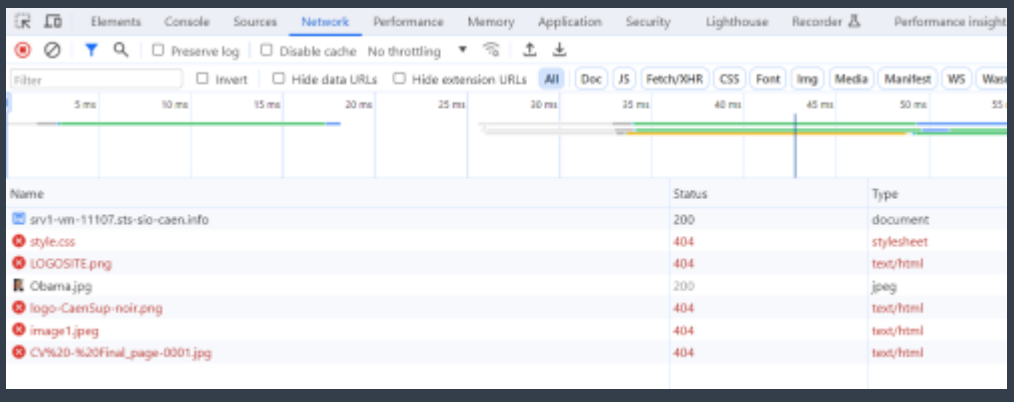
Vérifier le chemin et l'exprimer relativement au fichier php actuellement exécuté, en faisant attention à la **casse** (Linux contrairement à Windows fait la différence entre majuscules et minuscules).

Images/CSS

Diagnostic :

Pas d'erreur côté serveur en php dans les logs, mais les images ou feuilles de style css ne sont pas prises en compte.

La raison est vérifiable en regardant la partie **network/Réseau** de la console du navigateur :



Résolution :

Vérifier les chemins et les exprimer en absolu par rapport à la racine du vhost (**DocumentRoot** déclaré), en faisant attention à la **casse** (Linux contrairement à Windows fait la différence entre majuscules et minuscules).

Bonnes pratiques

- **index.php** doit se trouver à la racine du projet
- Le fichier **composer.json** permettant d'inclure les librairies doit être en un seul exemplaire à la racine du projet, et il faut installer ces librairies sur la VM :

```
composer update
```

- Le dossier **vendor** de composer ne doit pas être publié sur le repository et être ajouté dans le **.gitignore**
- Eviter les chemins à rallonge totalement inutiles (ie. `/portfolio/docs/php/site/…`)

MariaDb

Installation

L'installation de Mysql/mariaDB est facultative, et n'est à faire que si vous en avez besoin dans votre portfolio.

Installation de MariaDb (en root) :

```
apt update
apt install mariadb-server
```

Sécurisation

Sécurisation de l'installation :

```
mysql_secure_installation
```

Mettre le mot de passe **ilovedev** sur le compte root

Tester l'installation en mode client :

```
mariadb
```

PDO

Installation de la librairie PDO pour mariaDb/Mysql :

```
apt install php8.4-{dev,mysql}
```

From:
http://slamwiki2.kobject.net/ - **SlamWiki 2.1**

Permanent link:
http://slamwiki2.kobject.net/web/vm-install

Last update: **2026/03/26 13:52**