La Virtualisation

Ou comment reproduire des environnements de développement divers sur la même machine

Introduction

L'avantage de travailler dans l'informatique, c'est qu'il existe presque toujours une multitude de façon différentes de faire la même chose, mais le cœur de notre métier consiste à trouver celle qui sera la plus adaptée à notre situation. Toutes les bonnes solutions répondent à un besoin, et une bonne solution dans une situation donnée peut-être inadaptée dans d'autres situations.

Prenons un exemple : Lorsque nous avons démarré l'année avec la mise en place d'un site, nous n'avions au début pas besoin d'un serveur web, puisque nous travaillons sur des fichiers HTML / CSS directement côté client. La solution était donc de travailler sur ces fichiers directement depuis VSCode et de les ouvrir dans le navigateur.

Puis, nous avons implémenté un peu de PHP et de base de données. Nous avons donc eu besoin d'un serveur Web capable d'exécuter ce code PHP et d'héberger une base de données **persistante**. La solution la plus adaptée ici était WAMP puisque nous n'avions qu'un seul site à gérer, et c'était aussi le plus facile à installer puisque tout tenait en un seul logiciel.

Cette solution est malheureusement aujourd'hui inadaptée, puisque nous allons avoir besoin de gérer plusieurs sites potentiellement accessibles au même moment, et que WAMP ne permet pas de répondre efficacement à cette problématique.

C'est ici que l'on arrive à la virtualisation.

Qu'est-ce que c'est ?

La virtualisation consiste à allouer une partie des ressources physiques de notre machine à la création d'une **autre machine**, virtuelle, qui sera joignable comme si elle était réelle. On laisse déjà WAMP le faire lorsque l'on démarre WAMP. Il crée une machine virtuelle qui va servir de **serveur web**.

Nous allons donc vouloir reproduire ce comportement, à savoir émuler un serveur web, mais avec une configuration beaucoup plus poussée et même une gestion des **noms de domaines**.

C'est ici que rentrent en jeu trois composants logiciels qui vont fonctionner ensemble :

Virtualbox

VirtualBox est un logiciel d'Oracle qui permet de gérer l'allocation des ressources de notre ordinateur aux différentes VMs que l'on a sur notre machine. Il permet de consulter leur état, de les démarrer, de les éteindre, et surtout pour celles qui l'autorisent, de communiquer avec elles via un **terminal**. Nous allons simplement l'installer et laisser Vagrant gérer la communication avec.

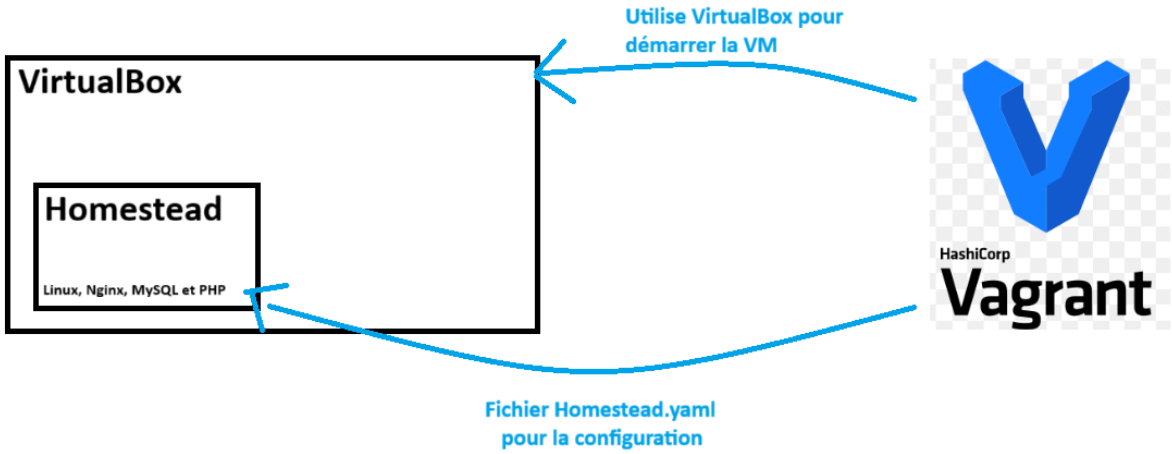
Vagrant

Vagrant va être notre interface entre VirtualBox et notre terminal. Vagrant va nous permettre d'installer, de télécharger, de configurer, démarrer et éteindre les VMs dont nous avons besoin. Il s'occupe de la transposabilité entre nos besoins et VirtualBox : Vagrant se sert de VirtualBox pour démarrer nos VMs, mais c'est bien avec Vagrant que nous allons intéragir.

Homestead

Homestead est le nom d'une machine virtuelle, à la base prévue pour fonctionner avec le framework Laravel.

Elle est toutefois particulièrement adaptée à notre cas d'usage puisqu'elle embarque PHP, MariaDb, un serveur web capable de gérer les noms de domaines et surtout : Elle est d'une simplicité déconcertante à paramétrer.



INSTALLATION

Installer Vagrant et VirtualBox 6.1 (reprendre les fichiers .exe)

Installer Git for Windows (Reprendre le .exe)

Se rendre dans le dossier /dev et cloner le dépôt homestead :

```
git clone https://github.com/laravel/homestead.git Homestead
```

Puis se rendre dans le dossier Homestead avec cd Homestead et checkout sur la release :

```
git checkout release
```

Double cliquer sur le fichier init.bat

Récupérer le fichier Homestead.yaml du prof

Ouvrir Bloc-notes en tant qu'admin et modifier le fichier C :/Windows/System32/drivers/etc/hosts, faire pointer l'IP dans le YAML vers le domaine souhaité dans la partie « sites »

```
ssh-keygen -t rsa -b 4096 -C "n.savoca@osengo.fr"
```

```
vagrant up
```

```
vagrant ssh
```

```
sudo apt update
```

Reprendre la config de xdebug dans

```
sudo nano /etc/php/8.3/fpm/conf.d/20-xdebug.ini
```

Et passer la valeur de xdebug.mode a « develop,trace »

Ctrl + X pour sortir, Y pour valider l’enregistrement, et entrée pour confirmer le nom.

```
vagrant halt
```

```
vagrant up --provision
```

MODE D'EMPLOI DE VAGRANT

!!! AVANT CHAQUE ÉTEIGNAGE DE L'ORDINATEUR, PENSER À ÉTEINDRE LA VM !!!

Pour cela on va simplement lui demander :

```
vagrant halt
```

Lorsque l'on démarre une nouvelle journée, après le démarrage du PC, il faut se rendre dans le dossier Homestead. Pour cela, on utilise la commande `cd` suivie du chemin du dossier. Par exemple :

```
cd .\dev\Homestead\
```

Chez certains, ce chemin sera \Desktop\dev\Homestead, ou encore \Documents\dev\homestead… à vous de savoir où vous êtes en regardant le chemin dans le Powershell :

```
PS C:\Users\pc>
```

Par exemple dans ce cas, mon dossier Homestead est dans C:\Users\pc\dev\Homestead. Donc depuis cet endroit, je devrais utiliser `cd .\dev\Homestead\`

Ensuite, on va simplement lui dire de démarrer la VM avec `vagrant up`

Si on a besoin de mettre à jour le fichier Homestead.yaml, alors on va devoir réapprovisionner ce fichier à Vagrant. On utilise pour cela soit vagrant provision, soit on peut lui dire de démarrer **ET** de reprendre le fichier yaml, avec : `vagrant up --provision`