Preparar el entorno de desarrollo

Puedes lanzar una aplicación desarrollada con Laravel en cualquier máquina o servidor con Apache/Nginx, PHP y Composer. Para facilitar la creación del entorno de desarrollo existen dos opciones principales:

- Laravel Homestead. Homestead es una Vagrant box que provee de un entorno de desarrollo con todo el software necesario: PHP, servidor web, base de datos, gestor de dependencias, etc.
- Laravel Sail. Laravel Sail es una interfaz de línea de comandos (CLI) para interactuar con la configuración de Docker predeterminada de Laravel. Proporciona una interfaz fácil de usar para ejecutar tareas comunes como levantar entornos de desarrollo local, ejecutar pruebas o desplegar aplicaciones.

No obstante, si te sientes más cómodo utilizando tu propio servidor o entorno de desarrollo, puedes hacerlo perfectamente siempre que cumplas con los requerimientos citados.

Laravel Sail

Con Laravel Sail puedes configurar fácilmente un entorno de desarrollo local para tu aplicación Laravel ejecutando un solo comando. Este entorno incluye un servidor de desarrollo PHP, una base de datos MySQL y todas las demás herramientas que se necesitan para empezar a trabajar en una aplicación. Laravel Sail está disponible para macOS, Linux y Windows (vía WSL2).

Pasos

Para crear una aplicación de Laravel con Laravel Sail, necesitarás tener instalado Docker Desktop. En el caso de Windows, asegúrate también que Windows Subsystem for Linux 2 (WSL2) está instalado y habilitado.

1. Crea un nuevo proyecto de Laravel utilizando el servicio laravel.build

Abre una terminal (en el caso de Windows, ábrela iniciando una sesión en WSL2) y ejecuta el siguiente comando dentro del directorio en el que deseas crear tu aplicación:

```
curl -s "https://laravel.build/example-app" | bash
```

Sustituye "example-app" por el nombre que quieres dar a tu aplicación. Otro aspecto importante es que el comando anterior instalará por defecto mysql, redis, meilisearch, mailhog, y selenium.

Puedes evitar esto y elegir los servicios que quieres instalar indicándolos como parámetro al final de la URL:

```
curl -s "https://laravel.build/example-app?with=mysql,redis" | bash
```

2. Inicia el entorno de desarrollo

Espera a que Laravel Sail instale todas las dependencias necesarias para tu aplicación. Este proceso puede llevar varios minutos ya que debe preparar todos los contenedores para el desarrollo desde tu máquina local.

Una vez que la instalación haya finalizado, entra en la carpeta creada para tu aplicación `cd example-app` y ejecuta el comando `./vendor/bin/sail up` para levantar el entorno de desarrollo local (utiliza `./vendor/bin/sail up -d` para arrancarlo en el background).

Abre tu navegador y visita la dirección http://localhost para ver tu aplicación de Laravel en funcionamiento.

Ya puedes empezar a trabajar en tu aplicación de Laravel.

Puedes utilizar el comando `sail stop` para parar el entorno.

(Opcional) 3. Crea un alias para sail

Para evitar tener que escribir `./vendor/bin/sail` continuamente, puedes crear un alias y así ejecutar simplemente `sail` directamente.

Edita tu archivo añadiendo esto al final del fichero `~/.zshrc` o `~/.bashrc`:

```
alias sail='[ -f sail ] && sh sail || sh vendor/bin/sail'
```

Una forma sencilla de editar/crear el fichero puede ser mediante el editor de texto nano:

```
nano ~/.zshrc
```

Vuelve a abrir el terminal y a partir de ahora ya podrás ejecutar comandos como `sail up` o `sail stop` directamente.

4. Ejecutar comandos en Sail

En Laravel es muy común ejecutar comandos de utilizades como Composer, Artisan o Node. Para ejecutar esos comandos sin tener que conectarte al contenedor `laravel.test` puedes hacerlo de la siguiente forma:

```
# Ejemplos de comandos desde dentro del contenedor:
php --version
php script.php
php artisan queue:work
composer require laravel/sanctum
node --version

# Ejemplos de comandos utilizando Laravel Sail:
sail php --version
sail php script.php
sail artisan queue:work
sail composer require laravel/sanctum
sail node --version
```

De todas formas, siempre tienes la opción de conectarte al contenedor con el siguiente comando:

```
docker exec -it <id_contenedor> bash
```

Además del anterior comando, Laravel Sail también ofrece la posibilidad de acceder al contenedor mediante el siguiente comando:

```
sail shell
```

5. Configuración inicial

Todos los archivos de configuración sobre la aplicación se almacenan en el directorio de configuración `config`. Laravel trae una configuración establecida por defecto, por lo que no es necesario configurar nada adicional para comenzar a desarrollar. No obstante, puedes entrar a ver el archivo `config/app.php` para hacerte a la idea de distintos parámetros a configurar.

La configuración relativa al entorno se almacena en el fichero `.env`. El tipo de configuración almacenada en este entorno es especialmente aquella que variará en función de si se trata de un entorno de desarrollo local, producción, etc. Nunca deberías subir este fichero a tu repositorio de código (por ejemplo Github).

6. Base de datos

Por defecto Laravel Sail crea dos bases de datos automáticamente. La primera con el mismo nombre de la aplicación. Puedes encontrar los datos relativos a la base de datos en el fichero `.env`.

```
DB_CONNECTION=mysql
DB_HOST=mysql
DB_PORT=3306
DB_DATABASE=example_app
DB_USERNAME=sail
DB_PASSWORD=password
```

Si deseas conectarte a la base de datos (mediante TablePlus o MySQL Workbench por ejemplo) deberás utilizar los datos de conexión de este fichero.

La segunda es una base de datos llamada `test` que se utilizará para testing (en caso de necesitarlo).

7. Configuración adicional

Si quieres saber más detalles sobre la configuración de Sail, como por ejemplo cambiar la versión de PHP por defecto o configurar opciones de testing, puedes visitar el siguiente enlace: https://laravel.com/docs/9.x/sail

8. Posibles errores

¿Errores durante la configuración del entorno? En este gist encontrarás algunos errores comunes y su posible solución.

Laravel Homestead

Pasos

1. Instalar VirtualBox 6.x

Descarga VirtualBox desde la página web oficial www.virtualbox.org e instálalo en tu ordenador. También puedes utilizar VMWare, Parallels o Hyper-V en lugar de Virtual Box.

2. Instalar Vagrant.

Descarga e instala Vagrant tal y como lo indica la documentación de la página web oficial https://www.vagrantup.com.

3. Añadir laravel/homestead box a tu instalación de Vagrant

Una Vagrant box es una imagen base utilizada para clonar de forma rápida y sencilla una máquina virtual. Ejecuta el siguiente comando para descargar e instalar la Vagrant box the Larabel y así poder utilizarla para crear tantos entornos como quieras.

```
vagrant box add laravel/homestead
```

Puedes comprobar que se ha añadido correctamente utilizando el comando `vagrant box list`.

4. Instalar Homestead

Clona el repositorio oficial de Homestead en un directorio. Como recomendación, puedes clonarlo en una carpeta llamada Homestead en la raíz de tu sistema, y utilizar esta máquina virtual para todos tus proyectos de Laravel.

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

5. Crea el archivo de configuración Homestead.yaml

Ejecuta el comando init.sh en un terminal dentro del directorio donde clonaste Homestead. Este comando creará el archivo de configuración Homestead.yaml.

```
// Mac / Linux...
bash init.sh

// Windows...
init.bat
```

El archivo Homestead.yaml creado tendrá el siguiente aspecto:

```
ip: "192.168.10.10"
memory: 2048
cpus: 2
provider: virtualbox

authorize: ~/.ssh/id_rsa.pub

keys:
    - ~/.ssh/id_rsa

folders:
    - map: ~/code
      to: /home/vagrant/code

sites:
    - map: homestead.test
      to: /home/vagrant/code/public

databases:
    - homestead
```

6. Genera las claves ssh

Si no dispones de unas clasves ssh en el sistema necesitarás generarlas desde un terminal. Para saber si ya dispones de ellas, intanta encontrar el directorio `.ssh` en tu sistema operativo (normalmente suele encontrarse en C:\Users\USER_NAME en Windows y el el directorio raíz ~ en Linux/Mac) y busca dos archivos llamados `id_rsa` y `id_rsa.pub`. Si los has encontrado, puedes saltarte este paso. En caso contrario, lanza el siguiente comando desde la consola:

```
ssh-keygen -t rsa -C "your_email@example.com"
```

El script te realizará algunas preguntas, simplemente pulsa ENTER y te creará los archivos `id_rsa` y `id_rsa.pub` en un directorio con nombre `.ssh`.

7. Configurar Homestead

En esta instalación no tendrás que tocar el archivo `Vagrantfile` (probablemente estés familiarizado con él ya que es típicamente el utilizado para configurar el entorno). Homestead delega la configuración en el archivo `Homestead.yaml`, por lo que a continuación te mostraremos las opciones más importantes que tendrás que configurar:

- **ssh keys**: el fichero de configuración Homestead.yaml tendrá la configuración realizada correctamente tanto para Mac como Linux. Si estás utilizando Windows, modifica los siguientes valores:

```
authorize: c:/Users/USER_NAME/.ssh/id_rsa.pub

keys:
    - c:/Users/USER_NAME/.ssh/id_rsa
```

- **Shared folders**: en este apartado se indican los directorios de tu sistema operativo que se mantendrán sincronizados con la máquina virtual creada. Modifica el valor de 'map' y escribe la carpeta donde almacenas tus proyectos (p.ej. `/Users/USER_NAME/dev/proyectos` en Linux/Mac o `c:/dev/proyectos` en Windows). En el siguiente ejemplo puedes ver cómo se sincronizarían los ficheros de nuestra carpeta `~/code/projects` en la carpeta `/home/vagrant/projects` de nuestra máquina virtual:

```
folders:
    - map: ~/code/projects
      to: /home/vagrant/projects
```

- **Sites**: La propiedad sites permite mapear fácilmente un dominio con un directorio de nuestro entorno virtual. De este forma podremos utilizar el dominio indicado (p.ej. aplicacion1.test) para acceder a nuestra aplicación desde el navegador. Ten en cuenta que el punto de entrada a una aplicación de Laravel es el archivo `index.php` ubicado en la carpeta `/public` de nuestro proyecto Laravel, por lo que el dominio tendrá que apuntar siempre a esta carpeta. Por ejemplo:

```
sites:
    - map: aplicacion1.test
      to: /home/vagrant/projects/aplicacion1/public
    - map: aplicacion2.test
      to: /home/vagrant/projects/aplicacion2/public
```

8. Configurar el archivo local hosts

Es probable que vayas a utilizar el entorno virtual creado para múltiples proyectos o aplicaciones, por lo que necesitarás añadir los dominios indicados en el anterior apartado de 'sites' al archivo `hosts` de tu ordenador. De esta forma puedes redirigir las peticiones a dominios concretos a aplicaciones de tu entorno virtual Homestead.

Modifica el archivo `hosts` (lo encontrarás en `/etc/hosts` en Mac/Linux y en `C:\Windows\System32\drivers\etc\hosts` en Windows):

```
192.168.10.10  aplicacion1.test
192.168.10.10  aplicacion2.test
```

Para evitar otro tipo de problemas, la recomendación general es utilizar dominios de tipo ".localhost", ".invalid", ".test", or ".example".

9. Arrancar y prueba el entorno creado

Una vez ya tenemos Homestead configurado y el archivo `hosts` modificado, ya podemos ir al directorio `Homestead` y ejecutar el siguiente comando para arranchar la máquina virtual (la primera vez tardará más tiempo al tener que realizar la preparación del entorno):

```
vagrant up
```

Puedes acceder mediante SSH a tu máquina virtual:

```
vagrant ssh
```

¡Enhorabuena! Ya estás preparado para comenzar a crear tu primera aplicación web con Laravel.

Puedes detener la máquina virtual con el comando `vagrant halt`.

Error 'input file not specified'

Es posible que al intentar cargar una aplicación el navegador te muestre el mensaje 'input file not specified'. Esto es algo común cuando por ejemplo cambiamos el archivo `host` y añadimos alguna URL más. En caso de que te aparezca este error, para la máquina virtual y vuélvela arrancar con el siguiente comando:

```
vagrant up --provision
```

Error de VBoxManage

Es posible que a la hora de hacer `vagrant up` no finalice correctamente y muestre el siguiente error:

```
There was an error while executing `VBoxManage`, a CLI used by Vagrant
for controlling VirtualBox. The command and stderr is shown below.

Command: ["startvm", "b3d4e98d-8a88-4335-8430-28a731e02f07", "--type",
"headless"]

Stderr: VBoxManage.exe: error: Failed to open/create the internal network
'HostInterfaceNetworking-VirtualBox Host-Only Ethernet Adapter #2'
(VERR_INTNET_FLT_IF_NOT_FOUND).
VBoxManage.exe: error: Failed to attach the network LUN
(VERR_INTNET_FLT_IF_NOT_FOUND)
VBoxManage.exe: error: Details: code E_FAIL (0x80004005), component
ConsoleWrap, interface IConsole.
```

El error puede ser debido a las versiones de Vagrant y VirtualBox utilizadas. Para solucionarlo debería ser suficiente con actualizar VirtualBox. También puede ser necesario desinstalar el Extension Pack anterior e instalar la última versión.

!!! tip "Bonus: Extensiones de VS Code para desarrollar en Laravel" Puedes instalar algunas extensiones de VS Code para facilitar el desarrollo con Laravel. En el siguiente enlace tienes una recopilación de las más recomendadas.