Formation Laravel

IMIE Angers – Mai 2019

Déroulé

- Présentation et mécanique de base de Laravel (MVC, injection de dépendances …)
- Installation / Configuration de Laravel
- Routage d’une requête / Contrôleurs
- Vues avec Blade
- Connexion à la base de données / ORM Eloquent
- Gestion de formulaires
- Authentification
- Laravel avancé

Pré-requis

- Maitriser la programmation orientée objet en PHP
- Connaitre le patron de conception MVC est un plus
- Environnement WAMP / MAMP / LAMP fonctionnel
- Utilisation de PHPStorm recommandée

Chapitre 1 : Présentation

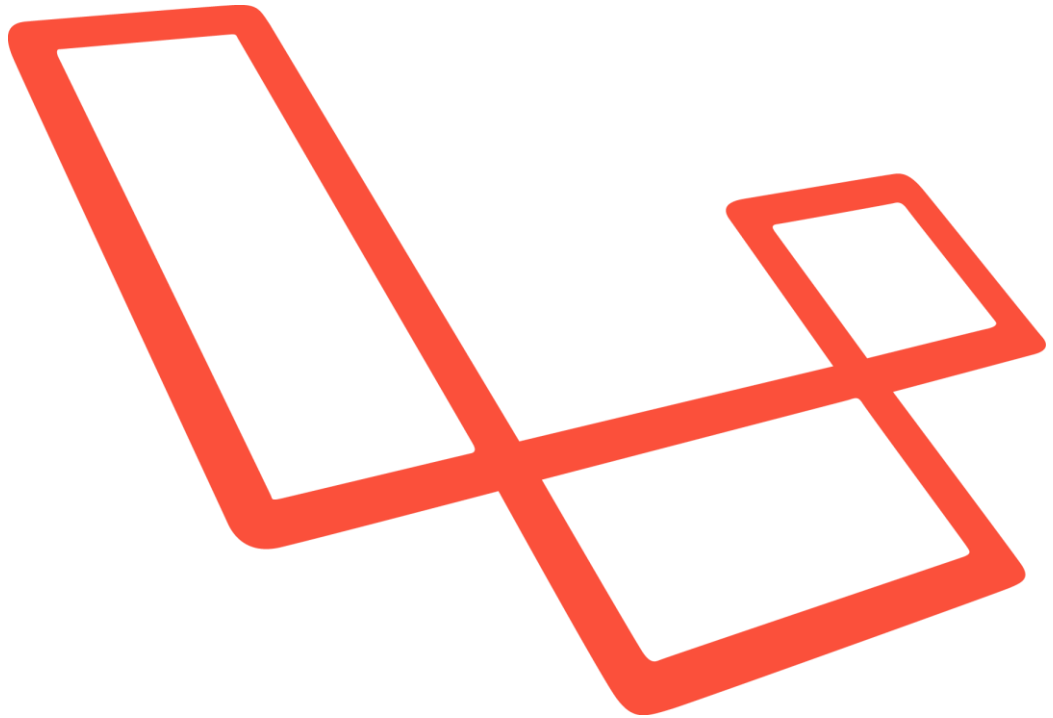
Présentation succincte de Laravel, fonctionnement de base

Objectifs

- Qu’est-ce qu’un framework ? Intérêts ?
- Les alternatives à Laravel
- Pourquoi Laravel ?
- Rappels sur le patron de conception MVC

Laravel

- Framework PHP open-source
- Créé en 2011 par Taylor Otwell
- Principe MVC
- Dernière version majeure : 5.8
- S’appuye sur plusieurs composant Symfony



Pourquoi un framework ?

- Offre une structure de code permettant un travail en équipe
- Bibliothèque de composants
- Évite de réinventer la roue
- Productivité accrue
- Produit du code de qualité

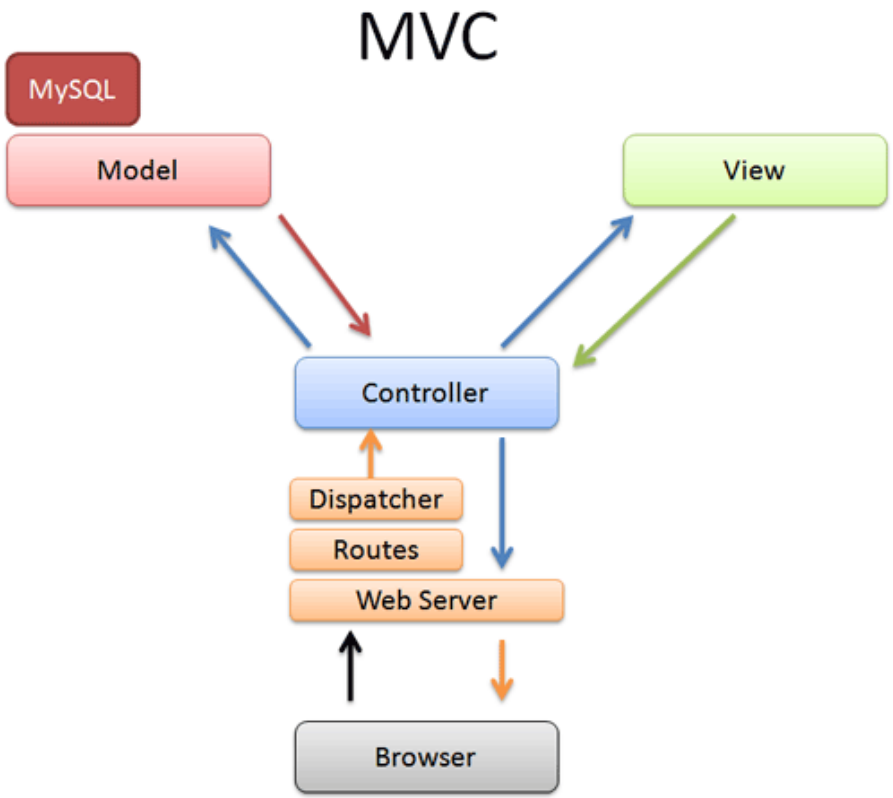
Alternatives à Laravel

- Frameworks
  - Symfony
  - Cake PHP
  - Phalcon PHP
  - Code Igniter
  - …
- CMS
  - WordPress
  - Joomla
  - Drupal
  - Prestashop
  - Magento

Pourquoi utiliser Laravel ?

- Framework N°1 sur Github (+50K stars)
- Très grosse communauté de développeurs
- Nombreux packages connexes permettant d’ajouter des fonctionnalités
- Plus facile à appréhender que Symfony

Patron de conception MVC



Patron de conception MVC

Sans MVC : une ressource par requête

http://www.monsite.com/	/home/monsite.com/htdocs/index.php
http://www.monsite.com/page1.php	/home/monsite.com/htdocs/page1.php
http://www.monsite.com/page2.php	/home/monsite.com/htdocs/page2.php
http://www.monsite.com/dossier/test.php	/home/monsite.com/htdocs/dossier/test.php

Patron de conception MVC

Avec MVC : un seul point d'entrée, la requête sera routée en fonction des paramètres

Les URLs peuvent être « plus propres » grâce à la réécriture d'URL

http://www.monsite.com/	/home/monsite.com/htdocs/index.php
http://www.monsite.com/nous-contacter/	/home/monsite.com/htdocs/index.php?controller=page&action=contact
http://www.monsite.com/news/	/home/monsite.com/htdocs/index.php?controller=news&action=index
http://www.monsite.com/news/5-titre-article.html	/home/monsite.com/htdocs/index.php?controller=news&action=view&id=5

Injection de dépendances

- Permet de séparer les « tâches » de la logique applicative
- La dépendance ne sera injectée qu'au cas par cas
- Economie de ressources, temps de chargement accéléré
- Système de facades (appel statique, binding de dépendances)

Chapitre 2 : Installation / Configuration

Configuration de l’environnement, installation de Laravel, configuration d’un projet Laravel

Pré-requis

- Version PHP >= 7.1.3
- Extensions PHP à activer : OpenSSL, PDO, MbString, XML (les autres sont normalement actives par défaut, sauf installation personnalisée)
- Environnement Apache ou NGINX

Solution 1 : Laravel Homestead

Avantages

- Stack complète et compatible
- Solution « Clé en main »
- Performances

Inconvénients

- Première installation longue et fastidieuse
- Plugin vagrant payant pour VMWare (Virtualbox peut être utilisé)
- Ne pas utiliser en production

[Windows] Solution 2 : WSL (W10)

Avantages

- Stack Linux
- Développement sous Windows, exécution sous Linux

Inconvénients

- Première installation pouvant être fastidieuse si non-habitué aux stacks Web Linux
- Performances OK mais non optimales

[Windows] Solution 3 : WAMP

Avantages

- Simplicité et rapidité d'installation
- Prêt à l'emploi

Inconvénients

- Performances médiocres
- Problèmes récurrents avec la version 64 bits de WAMP (nécessité d'installer des packages VC)

Solution 4 : PHPStorm Built-in server

Avantages

- Fourni avec PHPStorm (nécessite l'installation d'une DB si besoin)
- Performances OK (NGINX)

Inconvénients

- Obligation d'utiliser PHPStorm
- Le projet doit être correctement configuré

PHPStorm

- Environnement de développement complet et optimisé pour le web
- Nombreux plugins (dont Laravel)
- Built-in server
- Console intégrée

Composer

- Gestionnaire de paquet PHP
- Gestion des dépendances
- Gestionnaire d'auto-chargement des dépendances
- Liste des paquets disponibles : https://packagist.org/
- Accès à internet requis
- Installateur disponible sous Windows

Ajouter composer dans le PATH lors de l'installation !

Installation de Laravel

Via l'installateur Laravel

- composer global require laravel/installer
- laravel new mysite

Via Composer

- composer create-project --prefer-dist laravel/laravel mysite

Configuration d'un Virtual Host (Apache)

1. Modification du fichier HOSTS Windows

```
127.0.0.1       localhost
127.0.0.1       www.laravel.local
```

2. Création d'un virtual host Apache

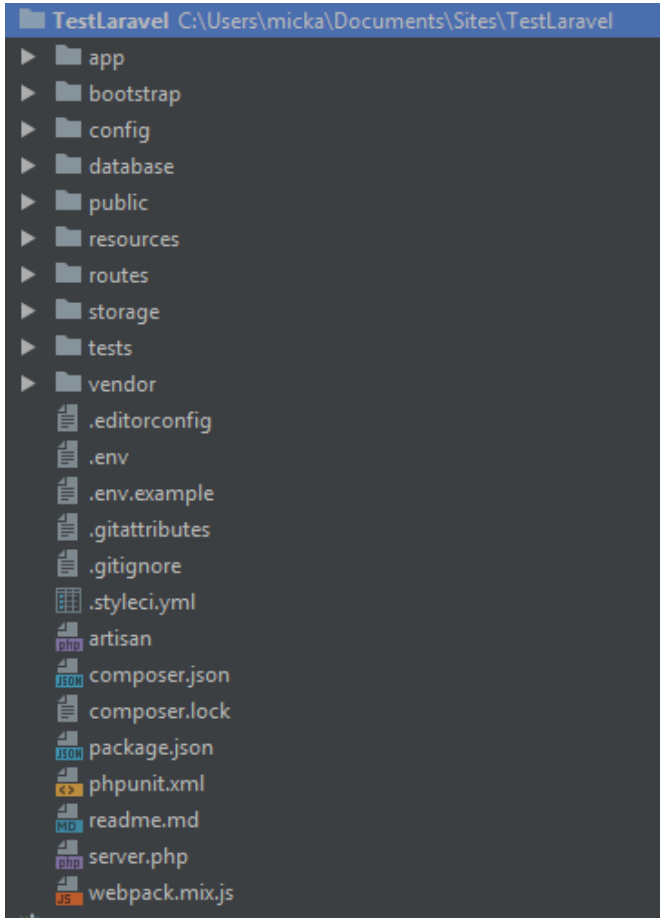
```
<VirtualHost *:80>
        ServerAdmin webmaster@localhost
        ServerName www.laravel.local
        DocumentRoot /home/mickael/Sites/laravel/public

        <Directory /home/mickael/Sites/laravel/public/>
                Options Indexes FollowSymLinks
                AllowOverride All
                Require all granted
        </Directory>
</VirtualHost>
```

Architecture de Laravel

- app : dossier de l'application web
- database : emplacement pour mettre le « schema » de la DB ainsi que les données à insérer automatiquement
- public : Document root du site, assets compilés
- resources : vues, assets non compilés, traductions
- storage : cache, sessions, logs …
- vendor : dépendances composer

Interdiction formelle de toucher au dossier « vendor » !



Configuration Laravel

- /config
- Fichier configuration général : app.php (à éditer pour les Facades)
- Il n’est plus nécessaire de déclarer les providers de packages externes (Laravel auto discover)
- Possibilité de créer un fichier par package / fonctionnalité
- Récupération d’une variable :

```
config(`[filename].[keyname]', $defaultValue);
```

```
$siteURL = config('app.url', 'http://www.monsite.com');
```

DOTENV Laravel

- Fichier .env
- https://github.com/vlucas/phpdotenv
- Très similaire au DOTENV Symfony

```php
$var = env('APP_URL', 'http://www.monsite.com');
```

Ligne de commande « artisan »

- Permet d'exécuter des commandes relative au projet associé
- Commande : « *php artisan* »
- Lister les commandes disponibles : « *php artisan list* »
- Aide sur une commande

```
php artisan help [COMMAND]
```

Ligne de commande « artisan »

- Générer clé de sécurité : *php artisan key:generate*
  - A quoi sert cette clé ?
- Générer du code : *php artisan make:XXX*
  - Contrôleurs
  - Modèles
  - …
- Nettoyer le cache : *php artisan cache:clear*
  - Possibilité de « catégoriser » les caches via des « stores » (DB, vues …)
- Base de données : *php artisan db:XXX / php artisan migrate*

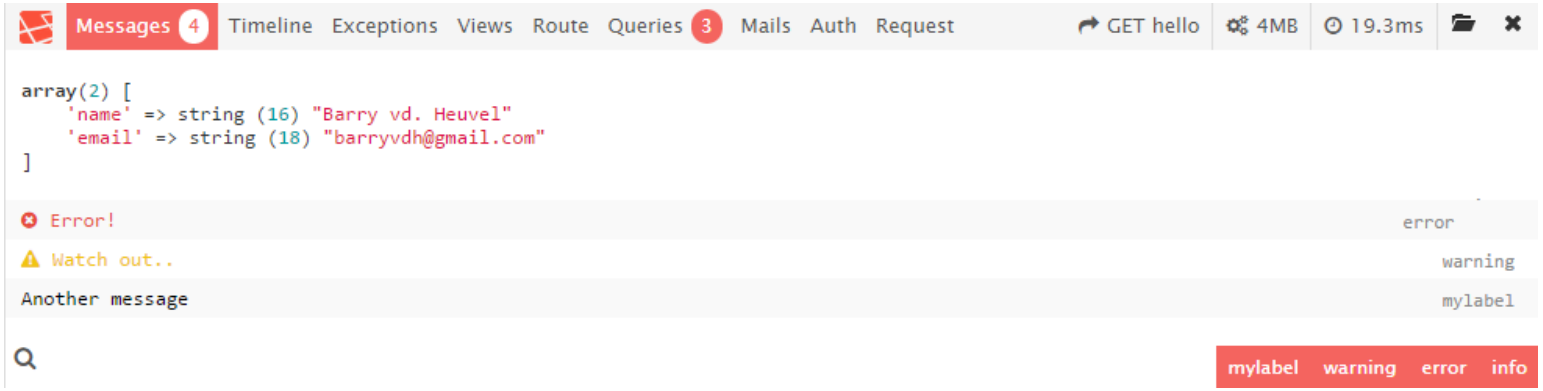
Troubleshoot

- Droits d’écriture dossier storage
- Mod Rewrite activé
- VirtualHost doit pointer dans le sous-dossier « public »
- Serveur rechargé ?
- Clé générée ?

Bonus : Bar de debug en DEV

- Package barryvdh/laravel-debugbar
- A ne mettre qu'en développement (même si une sécurité est présente)

composer require barryvdh/laravel-debugbar --dev



TP 1

Installation de la vidéothèque « Video Turfu »