

WEEK 1

So, what is Laravel?

Laravel is an MVC web-development framework written in PHP. It has been designed to improve the quality of your software by reducing both the cost of initial development and ongoing maintenance costs, and to improve the experience of working with your applications by providing clear expressive syntax and a core set of functionality that will save you hours of implementation time. Laravel was designed with the philosophy of using convention over configuration. This means that it makes intelligent assumptions about what you're trying to accomplish so that in most situations you'll be able to accomplish your goals with much less code. Not every application and database that you'll work with will be designed using these conventions. Thankfully, Laravel is flexible enough to work with your system—no matter how unique it is.

INSTALLATION

In five easy steps, you can install Laravel and get it set up on your system.

Step 1 – What do I need?

Before you install Laravel, you will need to check that you have all of the required elements, as follows:

- Laravel requires a web server environment and will run in Apache, IIS, and Nginx easily.
- Laravel should run in any server environment that supports PHP. The easiest way to set up a local webserver for development is to install XAMPP (Windows), MAMP (Mac OSX), or Apache with PHP5 on through a package manager on Linux.
- Laravel is written in the PHP scripting language. Currently, Laravel v3.2.5 requires a minimum of PHP v5.3 to run.
- Laravel requires that you have the File Info and Mcrypt libraries installed. Conveniently, they are almost always installed by default. For our QuickStart application we require a database.
- Out of the box, Laravel supports MySQL, MSSQL, PostgreSQL, and SQLite.

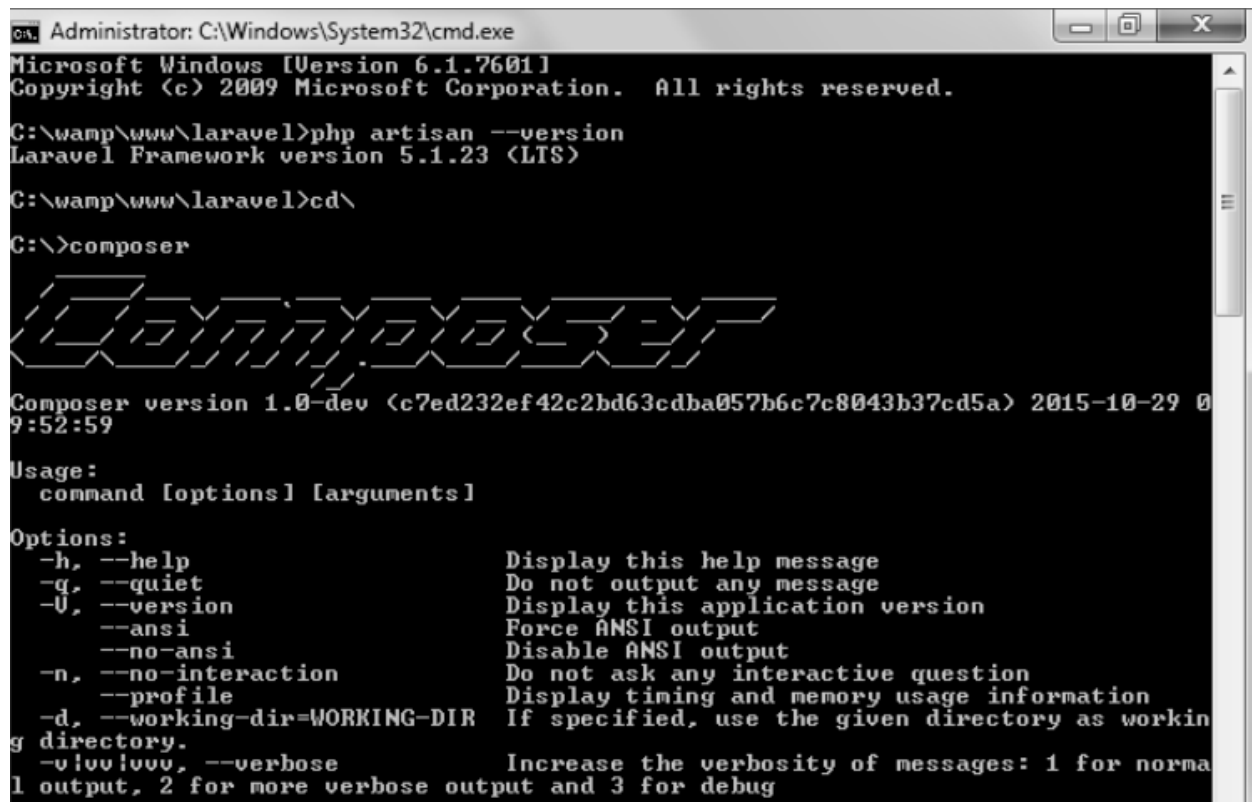
Step 2 – Downloading Laravel

Steps to install Laravel

Step 1: Visit the following URL and download composer to install it on your system.

<https://getcomposer.org/download/>

Step 2: After the Composer is installed, check the installation by typing the Composer command in the command prompt as shown in the following screenshot.



```
Administrator: C:\Windows\System32\cmd.exe
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation. All rights reserved.

C:\wamp\www\laravel>php artisan --version
Laravel Framework version 5.1.23 <LTS>

C:\wamp\www\laravel>cd\

C:\>composer

Composer version 1.0-dev (c7ed232ef42c2bd63cd8a057b6c7c8043b37cd5a) 2015-10-29 09:52:59

Usage:
  command [options] [arguments]

Options:
  -h, --help                Display this help message
  -q, --quiet               Do not output any message
  -V, --version             Display this application version
  --ansi                    Force ANSI output
  --no-ansi                 Disable ANSI output
  -n, --no-interaction      Do not ask any interactive question
  --profile                 Display timing and memory usage information
  -d, --working-dir=WORKING-DIR If specified, use the given directory as working directory.
  -vvvv, --verbose          Increase the verbosity of messages: 1 for normal output, 2 for more verbose output and 3 for debug
```

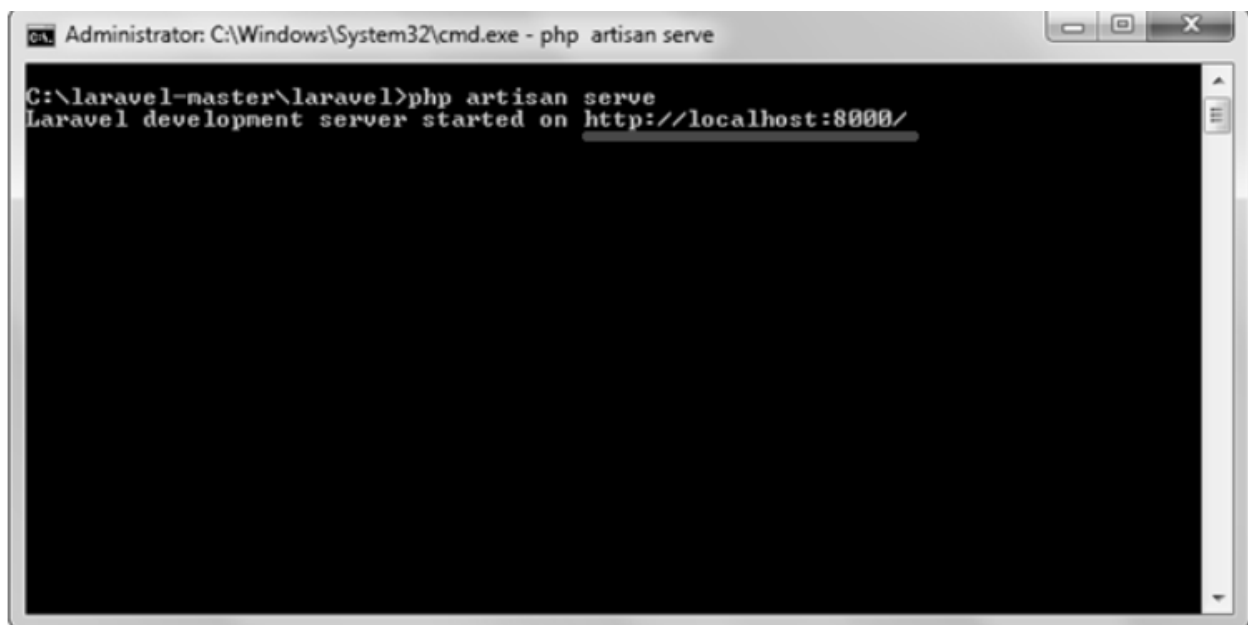
Step 3: Create a new directory anywhere in your system for your new Laravel project. After that, move to path where you have created the new directory and type the following command there to install Laravel.

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

Step 4: The above command will install Laravel in the current directory. Start the Laravel service by executing the following command.

`php artisan serve`

Step 5: After executing the above command, you will see a screen as shown below:

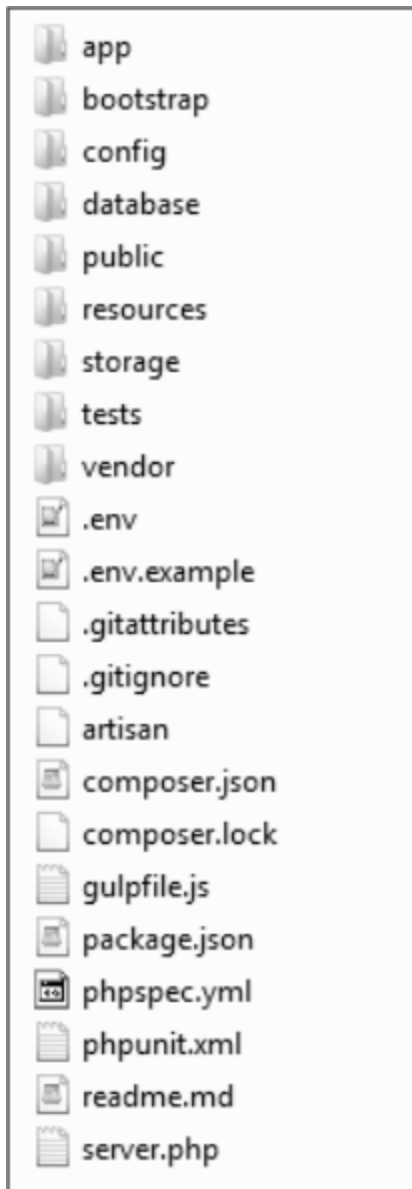


```
Administrator: C:\Windows\System32\cmd.exe - php artisan serve
C:\laravel-master\laravel>php artisan serve
Laravel development server started on http://localhost:8000/
```

Step 6: Copy the URL underlined in gray in the above screenshot and open that URL in the browser. If you see the following screen, it implies Laravel has been installed successfully.

Project Directory

- app: This directory contains the core code of the application.
- bootstrap: This directory contains the application bootstrapping script.
- config: This directory contains configuration files of application.
- database: This folder contains your database migration and seeds.
- public: This is the application's document root. It starts the Laravel application. It also contains the assets of the application like JavaScript, CSS, Images, etc.
- resources: This directory contains raw assets such as the LESS & Sass files, localization and language files, and Templates that are rendered as HTML.
- storage: This directory contains App storage, like file uploads etc. Framework storage (cache), and application-generated logs.
- test: This directory contains various test cases.
- vendor: This directory contains composer dependencies.



Basic Configuration

After installing Laravel, the first thing we need to do is to set the write permission for the directory storage and bootstrap/cache.

Generate Application key to secure session and other encrypted data. If the root directory doesn't contain the .env file then rename the .env.example to .env file and execute the

following command where you have installed Laravel. The newly generated key can be seen in the .env file.

You can also configure the locale, time zone, etc. of the application in the config/app.php file.

Environmental Configuration

Laravel provides facility to run your application in different environment like testing, production etc. You can configure the environment of your application in the .env file of the root directory of your application. If you have installed Laravel using composer, this file will automatically be created.

Naming the Application

The App Directory, by default, is namespaced under App. To rename it, you can execute the following command and rename the namespace.

```
php artisan app:name <name-of-your-application>
```

Replace the <name-of-your-application> with the new name of your application that you want to give.

Maintenance Mode

We need to modify our website on a regular basis. The website needs to be put on maintenance mode for this. Laravel has made this job easier. There are two artisan commands which are used to start and stop the maintenance mode which are described below.

Start Maintenance Mode

To start the maintenance mode, simply execute the following command.

```
php artisan down
```

It will activate the Maintenance mode and all the request to server will be redirected to a single maintenance page as shown in the following screenshot.

Stop Maintenance Mode

After making changes to your website and to start it again, execute the following command.

```
php artisan up
```

Basic Routing

Basic routing is meant to route your request to an appropriate controller. The routes of the application can be defined in `app/Http/routes.php` file. Here is the general route syntax for each of the possible request.

```
Route::get('/', function () {  
    return 'Hello World';  
});
```

```
Route::post('foo/bar', function () {  
    return 'Hello World';  
});
```

```
Route::put('foo/bar', function () {  
    //  
});
```

```
Route::delete('foo/bar', function () {  
    //  
});
```