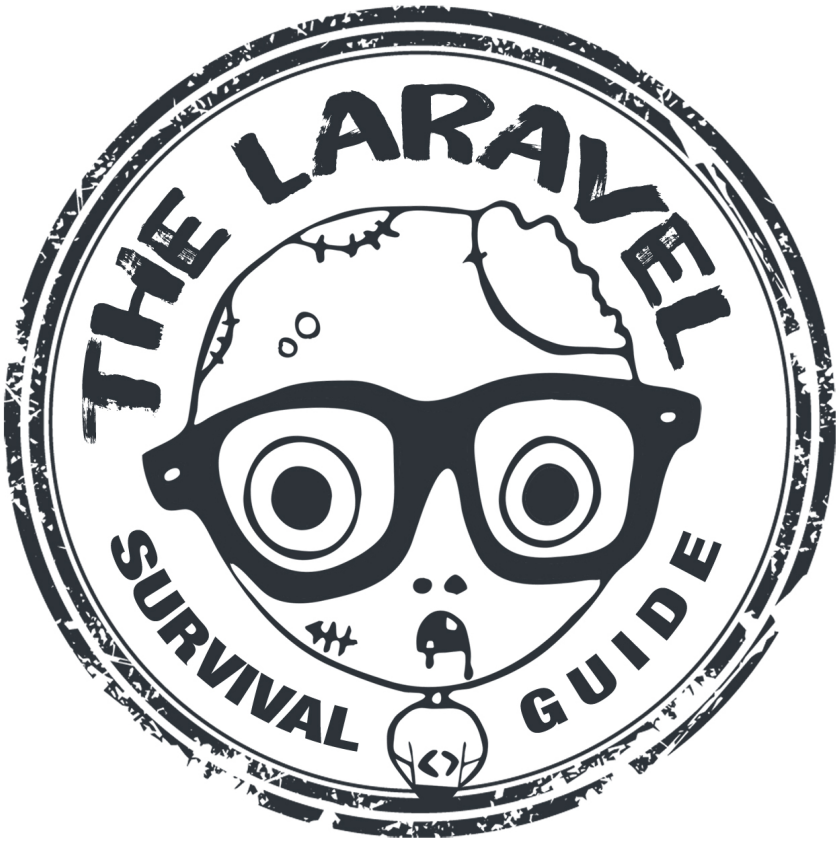


BY TONY LEA & THE DEVDOJO

EDITION 1

THE LARAVEL SURVIVAL GUIDE

A FUN WAY TO LEARN THE BASICS OF LARAVEL
& SAVE YOURSELF FROM BECOMING A ZOMBIE DEVELOPER!

The Laravel Survival Guide

Tony Lea

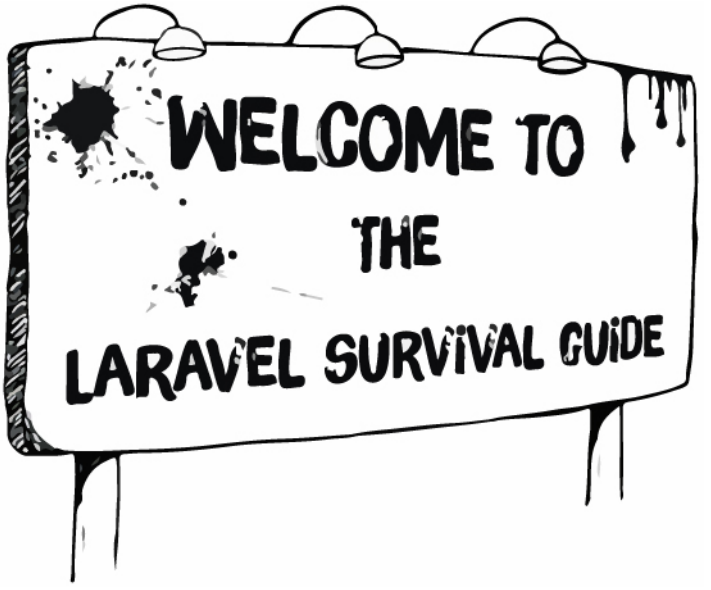
This book is for sale at http://leanpub.com/laravelsurvivalguide

This version was published on 2016-01-24

This is a Leanpub book. Leanpub empowers authors and publishers with the Lean Publishing process. Lean Publishing is the act of publishing an in-progress ebook using lightweight tools and many iterations to get reader feedback, pivot until you have the right book and build traction once you do.

Contents

Introduction



Why The Book? Well, it's not really a book… It's more of a guide (hence the title). A guide to save yourself and others from becoming a zombie developer!

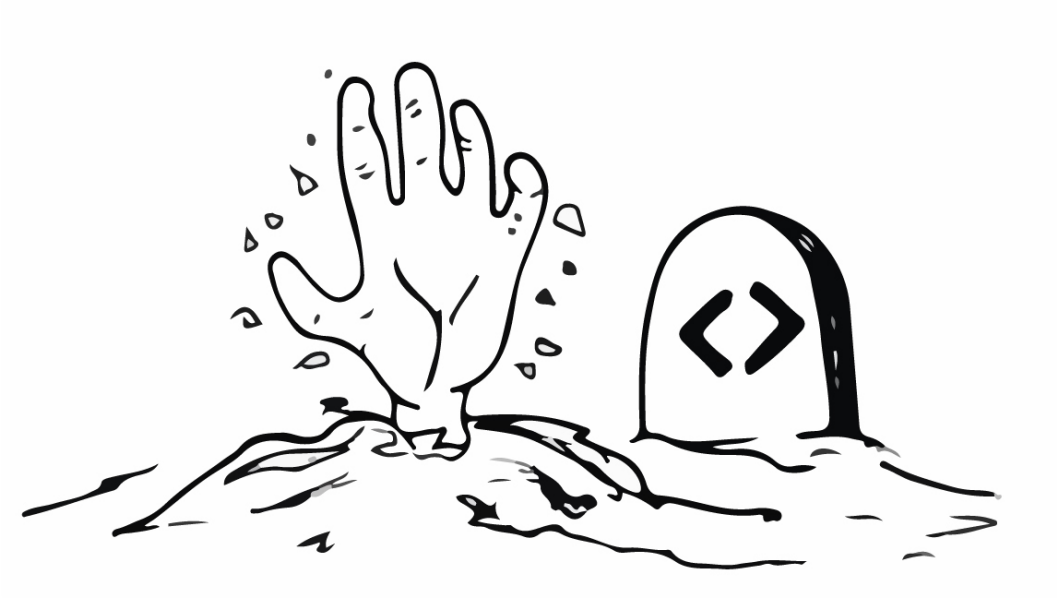
What exactly is a zombie developer? Well, a zombie developer is a developer like you or I, yet they mindlessly hack on PHP apps and do the same thing over and over. These repetitive tasks are incredibly time consuming, and make the developer brain dead. When this happens it gives them a thirst for blood and an urge to kill.

So, instead of letting this happen the developer could have used the amazing Laravel framework for rapid application development. This will help them keep their sanity and it will make coding enjoyable again. Oh, yeah… And it'll save lives.

By learning the basics of Laravel you can save yourself, and possibly others, from becoming a deteriorating zombie developer.

Don't let that inner zombie revive, be sure to keep in hand this Laravel survival guide.

Chapter 1 - Getting Started



A zombie developer is very slow at setting up a new project, whereas a Laravel developer can run a few commands to set up a new project in a matter of seconds!

Being aware of one's environment is crucial to surviving the zombie developer apocalypse. In this chapter, we will briefly go over setting up your local environment.

> Quick note: We will not go into full detail of installing system requirements such as PHP, MySQL, and Apache. Instead, we are just going to give a quick overview on setting up your local development environment so that way we can start getting into the code as soon as possible.

Your Local Development Environment

A local environment, also referred to as the "development environment", is when you work on your web app from your computer. When you are ready to make your application available for the world to see you will move your code to another server referred to as the "production environment".

So, when we set up your local development environment we are referring to setting up the required applications to run a Laravel app on your computer.

A basic local development environment for Laravel typically requires 3 applications, which are:

1. Apache or Nginx (the **web server** for your application)
2. MySQL (the **database** for your application)
3. PHP (the **server-side scripting** language for your application)

There are many programs for each operating system that will install all these applications for you. Just to name a few there is MAMP, WampServer, and XAMPP. You can feel free to check out the links below for these applications:

- https://www.mamp.info/en/ (Mac and Windows)
- http://www.wampserver.com/en/ (Windows)
- https://www.apachefriends.org/ (Mac, Windows, and Linux)

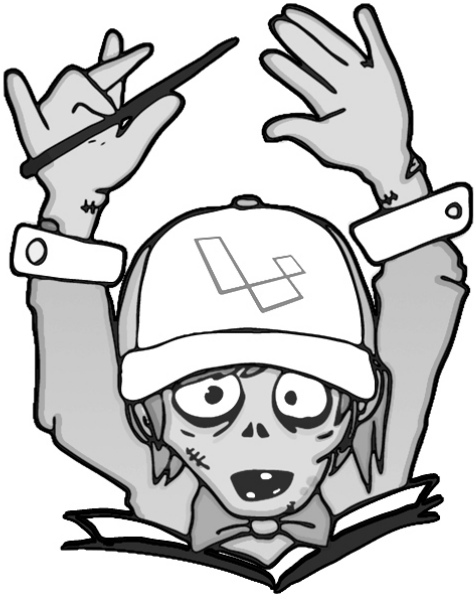
Be sure to read the docs for all the system requirements at http://laravel.com/docs.

There is also one last way of getting your local development environment setup for Laravel that is well worth mentioning. It is called a virtual machine that has all the system requirements already installed. You can learn more about setting up a Laravel virtual machine at http://laravel.com/docs/homestead.

There is no right or wrong way to setup your local development environment as long as you meet the minimum system requirements for Laravel. Find a way that works for you and start building the next latest and greatest web app!

Okay, Let's move on to Composer and the Laravel Installer.

Chapter 2 - Composer & The Laravel Installer



A zombie developer manually moves files into their project, whereas a Laravel developer leverages composer to install tools and libraries.

Taking on the zombie developer apocalypse on your own would be almost impossible. Getting help from others is essential, and that's exactly what composer allows us to do. Composer is used to include libraries from other developers into our application.

Let's continue.

Composer & The Laravel Installer

How easy would it be if you could open up a command prompt and type in:

```
$ laravel new blog
```

And a Laravel app is created inside a folder named “blog”. Well, that’s what the laravel installer does. So, let’s learn how we can use this on our computer.

The Laravel installer, as well as many PHP packages, make use of a dependency manager called Composer (http://getcomposer.org) to add this functionality.

So, What exactly is a dependency manager?

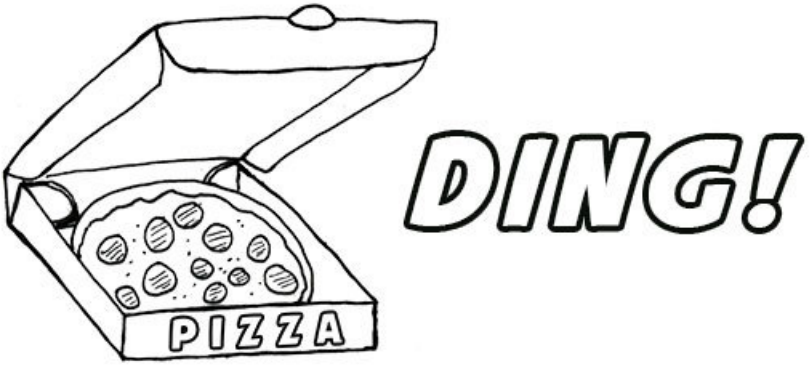
Well, a dependency manager is nothing more than a tool to manage your dependencies.

“WHAT!!!?”, yeah the definition still sounds pretty abstract, right?

Let me put this another way to help you understand how Composer works.

I know you are probably a fan of eating pizza instead of eating brains, so let’s pretend we could use a command to make us a pizza:

```
$ composer make pizza
```



By default, we are given a pepperoni pizza. But let's say we wanted this command to make us a pizza with different toppings. Perhaps we wanted a meat-lovers pizza. We would probably need the following toppings:

```
{
    "toppings" : [
        "peperoni", "ham", "bacon", "beef", "sausage"
    ]
}
```

Now, if we save this file in our current directory and name it 'composer.json' and run the command again:

```
$ composer make pizza
```

DING! We now get our meat-lovers pizza instead of our pepperoni pizza.

Hazzzaa!

As you can see Composer is a way of managing the things we need to build our app (or pizza).

Composer is also a command line tool we can use it to install other command line tools.

One of those tools is the Laravel installer.

To add the Laravel installer to our computer, we must first install composer.

Visit https://getcomposer.org/, click on the 'Getting Started' button, navigate to 'Installing Globally', and walk through how to globally install composer on your machine.

After downloading and installing composer you can run the following command to add the Laravel installer.

```
$ composer global require "laravel/installer=~1.1"
```

Now we have successfully added the laravel installer to your machine, and you can easily create a new laravel app by typing:

```
$ laravel new app_name
```

Then navigate to your new `app_name` folder in a command prompt. And run:

```
$ php artisan serve
```

Finally, navigate to http://localhost:8000/ in your browser and you will see a 'Welcome to Laravel' screen.

Now you're ready to start building your amazing app!

> **Warning**: If the laravel installer does not globally work you may need to specify where the composer bin directory is located on your machine. Visit http://devdojo.com/post/composer-bin-directory-path to learn how to do this.

How awesome is this?

With a single command line, you can be up and running with a new Laravel app in a few seconds!

You can learn more about the Laravel installer at http://laravel.com/docs.