Learning PHP 7

PHP is a great language for building web applications. It is essentially a server-side scripting language that is also used for general purpose programming. PHP 7 is the latest version with a host of new features, and it provides major backwards-compatibility breaks.

This book begins with the fundamentals of PHP programming by covering the basic concepts, such as variables, functions, class, and objects. You will set up a PHP server on your machine and learn how to read and write procedural PHP code. With an understanding of OOP as a paradigm, you will execute MySQL queries on your database and find out how to use MVC to create applications from scratch and add tests. Then, you will build REST APIs and perform behavioral tests on your applications. By the end of the book, you will have the skills required to read and write files, debug, test, and work with MySQL.

Who this book is written for
If you are a web developer or programmer who wants to create real-life web applications using PHP 7, or a beginner who wants to get started with PHP 7 programming, this book is for you. Prior knowledge of PHP 7 or programming in general is not mandatory.

What you will learn from this book
- Set up a server on your machine with PHP
- Use PHP syntax with the built-in server to create apps
- Apply the OOP paradigm to PHP in order to write richer code
- Use MySQL to manage data in your web applications
- Create a web application from scratch using MVC
- Add tests to your web application and write testable code
- Use an existing PHP framework to build and manage your applications
- Build REST APIs for your PHP applications
- Test the behavior of web applications with Behat

Who this book is written for
If you are a web developer or programmer who wants to create real-life web applications using PHP 7, or a beginner who wants to get started with PHP 7 programming, this book is for you. Prior knowledge of PHP 7 or programming in general is not mandatory.

What you will learn from this book
- Set up a server on your machine with PHP
- Use PHP syntax with the built-in server to create apps
- Apply the OOP paradigm to PHP in order to write richer code
- Use MySQL to manage data in your web applications
- Create a web application from scratch using MVC
- Add tests to your web application and write testable code
- Use an existing PHP framework to build and manage your applications
- Build REST APIs for your PHP applications
- Test the behavior of web applications with Behat

Who this book is written for
If you are a web developer or programmer who wants to create real-life web applications using PHP 7, or a beginner who wants to get started with PHP 7 programming, this book is for you. Prior knowledge of PHP 7 or programming in general is not mandatory.

What you will learn from this book
- Set up a server on your machine with PHP
- Use PHP syntax with the built-in server to create apps
- Apply the OOP paradigm to PHP in order to write richer code
- Use MySQL to manage data in your web applications
- Create a web application from scratch using MVC
- Add tests to your web application and write testable code
- Use an existing PHP framework to build and manage your applications
- Build REST APIs for your PHP applications
- Test the behavior of web applications with Behat

Who this book is written for
If you are a web developer or programmer who wants to create real-life web applications using PHP 7, or a beginner who wants to get started with PHP 7 programming, this book is for you. Prior knowledge of PHP 7 or programming in general is not mandatory.

What you will learn from this book
- Set up a server on your machine with PHP
- Use PHP syntax with the built-in server to create apps
- Apply the OOP paradigm to PHP in order to write richer code
- Use MySQL to manage data in your web applications
- Create a web application from scratch using MVC
- Add tests to your web application and write testable code
- Use an existing PHP framework to build and manage your applications
- Build REST APIs for your PHP applications
- Test the behavior of web applications with Behat

Who this book is written for
If you are a web developer or programmer who wants to create real-life web applications using PHP 7, or a beginner who wants to get started with PHP 7 programming, this book is for you. Prior knowledge of PHP 7 or programming in general is not mandatory.

What you will learn from this book
- Set up a server on your machine with PHP
- Use PHP syntax with the built-in server to create apps
- Apply the OOP paradigm to PHP in order to write richer code
- Use MySQL to manage data in your web applications
- Create a web application from scratch using MVC
- Add tests to your web application and write testable code
- Use an existing PHP framework to build and manage your applications
- Build REST APIs for your PHP applications
- Test the behavior of web applications with Behat

Who this book is written for
If you are a web developer or programmer who wants to create real-life web applications using PHP 7, or a beginner who wants to get started with PHP 7 programming, this book is for you. Prior knowledge of PHP 7 or programming in general is not mandatory.

What you will learn from this book
- Set up a server on your machine with PHP
- Use PHP syntax with the built-in server to create apps
- Apply the OOP paradigm to PHP in order to write richer code
- Use MySQL to manage data in your web applications
- Create a web application from scratch using MVC
- Add tests to your web application and write testable code
- Use an existing PHP framework to build and manage your applications
- Build REST APIs for your PHP applications
- Test the behavior of web applications with Behat

Who this book is written for
If you are a web developer or programmer who wants to create real-life web applications using PHP 7, or a beginner who wants to get started with PHP 7 programming, this book is for you. Prior knowledge of PHP 7 or programming in general is not mandatory.

What you will learn from this book
- Set up a server on your machine with PHP
- Use PHP syntax with the built-in server to create apps
- Apply the OOP paradigm to PHP in order to write richer code
- Use MySQL to manage data in your web applications
- Create a web application from scratch using MVC
- Add tests to your web application and write testable code
- Use an existing PHP framework to build and manage your applications
- Build REST APIs for your PHP applications
- Test the behavior of web applications with Behat

Who this book is written for
If you are a web developer or programmer who wants to create real-life web applications using PHP 7, or a beginner who wants to get started with PHP 7 programming, this book is for you. Prior knowledge of PHP 7 or programming in general is not mandatory.

What you will learn from this book
- Set up a server on your machine with PHP
- Use PHP syntax with the built-in server to create apps
- Apply the OOP paradigm to PHP in order to write richer code
- Use MySQL to manage data in your web applications
- Create a web application from scratch using MVC
- Add tests to your web application and write testable code
- Use an existing PHP framework to build and manage your applications
- Build REST APIs for your PHP applications
- Test the behavior of web applications with Behat

Who this book is written for
If you are a web developer or programmer who wants to create real-life web applications using PHP 7, or a beginner who wants to get started with PHP 7 programming, this book is for you. Prior knowledge of PHP 7 or programming in general is not mandatory.

What you will learn from this book
- Set up a server on your machine with PHP
- Use PHP syntax with the built-in server to create apps
- Apply the OOP paradigm to PHP in order to write richer code
- Use MySQL to manage data in your web applications
- Create a web application from scratch using MVC
- Add tests to your web application and write testable code
- Use an existing PHP framework to build and manage your applications
- Build REST APIs for your PHP applications
- Test the behavior of web applications with Behat

Who this book is written for
If you are a web developer or programmer who wants to create real-life web applications using PHP 7, or a beginner who wants to get started with PHP 7 programming, this book is for you. Prior knowledge of PHP 7 or programming in general is not mandatory.

What you will learn from this book
- Set up a server on your machine with PHP
- Use PHP syntax with the built-in server to create apps
- Apply the OOP paradigm to PHP in order to write richer code
- Use MySQL to manage data in your web applications
- Create a web application from scratch using MVC
- Add tests to your web application and write testable code
- Use an existing PHP framework to build and manage your applications
- Build REST APIs for your PHP applications
- Test the behavior of web applications with Behat

Who this book is written for
If you are a web developer or programmer who wants to create real-life web applications using PHP 7, or a beginner who wants to get started with PHP 7 programming, this book is for you. Prior knowledge of PHP 7 or programming in general is not mandatory.

What you will learn from this book
- Set up a server on your machine with PHP
- Use PHP syntax with the built-in server to create apps
- Apply the OOP paradigm to PHP in order to write richer code
- Use MySQL to manage data in your web applications
- Create a web application from scratch using MVC
- Add tests to your web application and write testable code
- Use an existing PHP framework to build and manage your applications
- Build REST APIs for your PHP applications
- Test the behavior of web applications with Behat

Who this book is written for
If you are a web developer or programmer who wants to create real-life web applications using PHP 7, or a beginner who wants to get started with PHP 7 programming, this book is for you. Prior knowledge of PHP 7 or programming in general is not mandatory.
In this package, you will find:

- The author biography
- A preview chapter from the book, Chapter 1 *Setting Up the Environment*
- A synopsis of the book’s content
- More information on *Learning PHP 7*
About the Author

**Antonio Lopez** is a software engineer with more than 7 years of experience. He has worked with PHP since university, which was 10 years ago, building small personal projects. Later, Antonio started his journey around Europe, working in Barcelona, London, Dublin, and back in Barcelona. He has worked in a number of different areas, from web applications to REST APIs and internal tools. Antonio likes to spend his spare time on personal projects and start-ups and has a strong vocation in education and teaching.
Preface

There is no need to state how much weight web applications have in our lives. We use web applications to know what our friends are doing, to get the latest news about politics, to check the results of our favorite football team in a game, or graduate from an online university. And as you are holding this book, you already know that building these applications is not a job that only a selected group of geniuses can perform, and that it's rather the opposite.

There isn't only one way to build web applications; there are actually quite a lot of languages and technologies with the sole purpose of doing this. However, if there is one language that stands out from the rest, either historically or because it is extremely easy to use, it is PHP and all the tools of its ecosystem.

The Internet is full of resources that detail how to use PHP, so why bother reading this book? That's easy. We will not give you the full documentation of PHP as the official website does. Our goal is not that you get a PHP certification, but rather to teach you what you really need in order to build web applications by yourself. From the very beginning, we will use all the information provided in order to build applications, so you can note why each piece of information is useful.

However, we will not stop here. Not only will we show you what the language offers you, but also we will discuss the best approaches to writing code. You will learn all the techniques that any web developer has to master, from OOP and design patterns such as MVC, to testing. You will even work with the existing PHP frameworks that big and small companies use for their own projects.

In short, you will start a journey in which you will learn how to master web development rather than how to master a programming language. We hope you enjoy it.
What this book covers

Chapter 1, Setting Up the Environment, will guide you through the installation of the different software needed.

Chapter 2, Web Applications with PHP, will be an introduction to what web applications are and how they work internally.

Chapter 3, Understanding PHP Basics, will go through the basic elements of the PHP language—from variables to control structures.

Chapter 4, Creating Clean Code with OOP, will describe how to develop web applications following the object-oriented programming paradigm.

Chapter 5, Using Databases, will explain how you can use MySQL databases in your applications.

Chapter 6, Adapting to MVC, will show how to apply the most famous web design pattern, MVC, to your applications.

Chapter 7, Testing Web Applications, will be an extensive introduction to unit testing with PHPUnit.

Chapter 8, Using Existing PHP Frameworks, will introduce you to existing PHP frameworks used by several companies and developers, such as Laravel and Silex.

Chapter 9, Building REST APIs, will explain what REST APIs are, how to use third-party ones, and how to build your own.

Chapter 10, Behavioral Testing, will introduce the concepts of continuous integration and behavioral testing with PHP and Behat.
Setting Up the Environment

You are about to start a journey—a long one, in which you will learn how to write web applications with PHP. However, first, you need to set up your environment, something that has proven to be tricky at times. This task includes installing PHP 7, the language of choice for this book; MySQL, the database that we will use in some chapters; Nginx, the web server that will allow us to visualize our applications with a browser; and Composer, the favorite PHP dependencies management tool. We will do all of this with Vagrant and also on three different platforms: Windows, OS X, and Ubuntu.

In this chapter, you will learn about:

- Using Vagrant to set up a development environment
- Setting up your environment manually on the main platforms

Setting up the environment with Vagrant

Not so long ago, every time you started working for a new company, you would spend an important part of your first few days setting up your new environment—that is, installing all the necessary tools on your new computer in order to be able to code. This was incredibly frustrating because even though the software to install was the same, there was always something that failed or was missing, and you would spend less time being productive.
Introducing Vagrant

Luckily for us, people tried to fix this big problem. First, we have virtual machines, which are emulations of computers inside your own computer. With this, we can have Linux inside our MacBook, which allows developers to share environments. It was a good step, but it still had some problems; for example, VMs were quite big to move between different environments, and if developers wanted to make a change, they had to apply the same change to all the existing virtual machines in the organization.

After some deliberation, a group of engineers came up with a solution to these issues and we got Vagrant. This amazing software allows you to manage virtual machines with simple configuration files. The idea is simple: a configuration file specifies which base virtual machine we need to use from a set of available ones online and how you would like to customize it—that is, which commands you will want to run the first time you start the machine—this is called "provisioning". You will probably get the Vagrant configuration from a public repository, and if this configuration ever changes, you can get the changes and reprovision your machine. It's easy, right?

Installing Vagrant

If you still do not have Vagrant, installing it is quite easy. You will need to visit the Vagrant download page at https://www.vagrantup.com/downloads.html and select the operating system that you are working with. Execute the installer, which does not require any extra configuration, and you are good to go.

Using Vagrant

Using Vagrant is quite easy. The most important piece is the Vagrantfile file. This file contains the name of the base image we want to use and the rest of the configuration that we want to apply. The following content is the configuration needed in order to get an Ubuntu VM with PHP 7, MySQL, Nginx, and Composer. Save it as Vagrantfile at the root of the directory for the examples of this book.

```ruby
VAGRANTFILE_API_VERSION = "2"

Vagrant.configure(VAGRANTFILE_API_VERSION) do |config|
  config.vm.box = "ubuntu/trusty32"
  config.vm.network "forwarded_port", guest: 80, host: 8080
  config.vm.provision "shell", path: "provisioner.sh"
end
```
As you can see, the file is quite small. The base image's name is `ubuntu/trusty32`, messages to our port 8080 will be redirected to the port 80 of the virtual machine, and the provision will be based on the `provisioner.sh` script. You will need to create this file, which will be the one that contains all the setup of the different components that we need. This is what you need to add to this file:

```bash
#!/bin/bash

sudo apt-get install python-software-properties -y
sudo LC_ALL=en_US.UTF-8 add-apt-repository ppa:ondrej/php -y
sudo apt-get update
sudo apt-get install php7.0 php7.0-fpm php7.0-mysql -y
sudo apt-get --purge autoremove -y
sudo service php7.0-fpm restart

sudo debconf-set-selections <<< 'mysql-server mysql-server/root_password password root'
sudo debconf-set-selections <<< 'mysql-server mysql-server/root_password_again password root'
sudo apt-get -y install mysql-server mysql-client
sudo service mysql start

sudo apt-get install nginx -y
sudo cat > /etc/nginx/sites-available/default <<- EOM
server {
    listen 80 default_server;
    listen [::]:80 default_server ipv6only=on;

    root /vagrant;
    index index.php index.html index.htm;

    server_name server_domain_or_IP;

    location / {
        try_files $uri $uri/ /index.php?$query_string;
    }

    location - \.php\$ {
        try_files $uri /index.php =404;
        fastcgi_split_path_info ^(.+\.php)(/.*)$;
        fastcgi_pass unix:/var/run/php/php7.0-fpm.sock;
        fastcgi_index index.php;
        fastcgi_param SCRIPT_FILENAME $document_root$fastcgi_script_name;
        include fastcgi_params;
    }
}
EOM
sudo service nginx restart
```
Setting Up the Environment

The file looks quite long, but we will do quite a lot of stuff with it. With the first part of the file, we will add the necessary repositories to be able to fetch PHP 7, as it does not come with the official ones, and then install it. Then, we will try to install MySQL, server and client. We will set the root password on this provisioning because we cannot introduce it manually with Vagrant. As this is a development machine, it is not really a problem, but you can always change the password once you are done. Finally, we will install and configure Nginx to listen to the port 8080.

To start the virtual machine, you need to execute the following command in the same directory where Vagrantfile is:

```
$ vagrant up
```

The first time you execute it, it will take some time as it will have to download the image from the repository, and then it will execute the provisioner.sh file. The output should be something similar to this one followed by some more output messages:

```
$ vagrant up
Bringing machine "default" up with 'virtualbox' provider...
  default: Box 'ubuntu/trusty32' could not be found. Attempting to find and install...
  default: Box Provider: virtualbox
  default: Box Version: => 0
  default: Loading metadata for box 'ubuntu/trusty32'
  default: URL: https://vagrant.cachio.com/ubuntu/trusty32
  default: Adding box 'ubuntu/trusty32' (v0.8.0) for provider: virtualbox
  default: Successfully added box 'ubuntu/trusty32' (v0.8.0) for 'virtualbox'
  default: Importing base box 'ubuntu/trusty32'...
  default: Matching MAC address for NAT networking...
  default: Checking if box 'ubuntu/trusty32' is up to date...
  default: Setting the name of the VM vagrant_default_de54b660344f_53416...
  default: Clearing any previously set forwarded ports...
  default: Clearing any previously set network interfaces...
  default: Preparing network interfaces based on configuration...
  default: Adapter 1: not
  default: Forwarding ports...
  default: 44 => 8080 (adapter 1)
  default: 22 => 2222 (adapter 1)
  default: Booting VM...
  default: Waiting for machine to boot. This may take a few minutes...
  default: SSH address: 127.0.0.1:2222
  default: SSH username: vagrant
  default: SSH auth method: private key
```

In order to access your new VM, run the following command on the same directory where you have your Vagrantfile file:

```
$ vagrant ssh
```

Vagrant will start an SSH session to the VM, which means that you are inside the VM. You can do anything you would do with the command line of an Ubuntu system. To exit, just press Ctrl + D.
Sharing files from your laptop to the VM is easy; just move or copy them to the same directory where your Vagrantfile file is, and they will "magically" appear on the /vagrant directory of your VM. They will be synchronized, so any changes that you make while in your VM will be reflected on the files of your laptop.

Once you have a web application and you want to test it through a web browser, remember that we will forward the ports. This means that in order to access the port 80 of your VM, the common one for web applications, you will have to point to the port 8080 on your browsers; here's an example: http://localhost:8080.

Setting up the environment on OS X

If you are not convinced with Vagrant and prefer to use a Mac to develop PHP applications, this is your section. Installing all the necessary tools on a Mac might be a bit tricky, depending on the version of your OS X. At the time of writing this book, Oracle has not released a MySQL client that you can use via the command line that works with El Capitan, so we will describe how to install another tool that can do a similar job.

Installing PHP

If it is the first time you are using a Mac to develop applications of any kind, you will have to start by installing Xcode. You can find this application for free on the App Store:
Another indispensable tool for Mac users is Brew. This is the package manager for OS X and will help us install PHP with almost no pain. To install it, run the following command on your command line:

```bash
$ ruby -e "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/master/install)"
```

If you already have Brew installed, you can make sure that everything works fine by running these two commands:

```bash
$ brew doctor
$ brew update
```

It is time to install PHP 7 using Brew. To do so, you will just need to run one command, as follows:

```bash
$ brew install homebrew/php/php70
```

The result should be as shown in the following screenshot:
Make sure to add the binary to your PATH environment variable by executing this command:

```bash
$ export PATH="$(brew --prefix homebrew/php/php70)/bin:$PATH"
```

You can check whether your installation was successful by asking which version of PHP your system is using with the `php -v` command.

### Installing MySQL

As pointed out at the beginning of this section, MySQL is a tricky one for Mac users. You need to download the MySQL server installer and MySQL Workbench as the client. The MySQL server installer can be found at https://dev.mysql.com/downloads/mysql/. You should find a list of different options, as shown here:
Setting Up the Environment

The easiest way to go is to download **DMG Archive**. You will be asked to log in with your Oracle account; you can create one if you do not have any. After this, the download will start. As with any DMG package, just double-click on it and go through the options—in this case, just click on **Next** all the time. Be careful because at the end of the process, you will be prompted with a message similar to this:

Make a note of it; otherwise, you will have to reset the root password. The next one is MySQL Workbench, which you can find at [http://www.mysql.com/products/workbench/](http://www.mysql.com/products/workbench/). The process is the same; you will be asked to log in, and then you will get a DMG file. Click on **Next** until the end of the installation wizard. Once done, you can launch the application; it should look similar to this:
Installing Nginx

In order to install Nginx, we will use Brew, as we did with PHP. The command is the following:

```bash
$ brew install nginx
```

If you want to make Nginx start every time you start your laptop, run the following command:

```bash
$ ln -sfv /usr/local/opt/nginx/*.plist ~/Library/LaunchAgents
```

If you have to change the configuration of Nginx, you will find the file in `/usr/local/etc/nginx/nginx.conf`. You can change things, such as the port that Nginx is listening to or the root directory where your code is (the default directory is `/usr/local/Cellar/nginx/1.8.1/html`). Remember to restart Nginx to apply the changes with the `sudo nginx` command.

Installing Composer

Installing Composer is as easy as downloading it with the `curl` command; move the binary to `/usr/local/bin/` with the following two commands:

```bash
$ curl -sS https://getcomposer.org/installer | php
$ mv composer.phar /usr/local/bin/composer
```

Setting up the environment on Windows

Even though it is not very professional to pick sides based on personal opinions, it is well known among developers how hard it can be to use Windows as a developer machine. They prove to be extremely tricky when it comes to installing all the software since the installation mode is always very different from OS X and Linux systems, and quite often, there are dependency or configuration problems. In addition, the command line has different interpreters than Unix systems, which makes things a bit more confusing. This is why most developers would recommend you use a virtual machine with Linux if you only have a Windows machine at your disposal.

However, to be fair, PHP 7 is the exception to the rule. It is surprisingly simple to install it, so if you are really comfortable with your Windows and would prefer not to use Vagrant, here you have a short explanation on how to set up your environment.
Installing PHP

In order to install PHP 7, you will first download the installer from the official website. For this, go to http://windows.php.net/download. The options should be similar to the following screenshot:

Choose **x86 Thread Safe** for Windows 32-bit or **x64 Thread Safe** for the 64-bit one. Once downloaded, uncompress it in C:\php7. Yes, that is it!

Installing MySQL

Installing MySQL is a little more complex. Download the installer from http://dev.mysql.com/downloads/installer/ and execute it. After accepting the license agreement, you will get a window similar to the following one:
For the purposes of the book—and actually for any development environment—you should go for the first option: **Developer Default**. Keep going forward, leaving all the default options, until you get a window similar to this:
Setting Up the Environment

Depending on your preferences, you can either just set a password for the root user, which is enough as it is only a development machine, or you can add an extra user by clicking on Add User. Make sure to set the correct name, password, and permissions. A user named test with administration permissions should look similar to the following screenshot:

For the rest of the installation process, you can select all the default options.

Installing Nginx

The installation for Nginx is almost identical to the PHP 7 one. First, download the ZIP file from http://nginx.org/en/download.html. At the time of writing, the versions available are as follows:
You can safely download the mainline version 1.9.10 or a later one if it is stable. Once the file is downloaded, uncompress it in C:\nginx and run the following commands to start the web server:

```
$ cd nginx
$ start nginx
```

**Installing Composer**

To finish with the setup, we need to install Composer. To go for the automatic installation, just download the installer from [https://getcomposer.org/Composer-Setup.exe](https://getcomposer.org/Composer-Setup.exe). Once downloaded, execute it in order to install Composer on your system and to update your `PATH` environment variable.

**Setting up the environment on Ubuntu**

Setting up your environment on Ubuntu is the easiest of the three platforms. In fact, you could take the `provisioner.sh` script from the *Setting up the environment with Vagrant* section and execute it on your laptop. That should do the trick. However, just in case you already have some of the tools installed or you want to have a sense of control on what is going on, we will detail each step.
Setting Up the Environment

Installing PHP

The only thing to consider in this section is to remove any previous PHP versions on your system. To do so, you can run the following command:

$ sudo apt-get -y purge php.*

The next step is to add the necessary repositories in order to fetch the correct PHP version. The commands to add and update them are:

$ sudo apt-get install python-software-properties
$ sudo LC_ALL=en_US.UTF-8 add-apt-repository ppa:ondrej/php -y
$ sudo apt-get update

Finally, we need to install PHP 7 together with the driver for MySQL. For this, just execute the following three commands:

$ sudo apt-get install php7.0 php7.0-fpm php7.0-mysql -y
$ sudo apt-get --purge autoremove -y
$ sudo service php7.0-fpm start

Installing MySQL

Installing MySQL manually can be slightly different than with the Vagrant script. As we can interact with the console, we do not have to specify the root password previously; instead, we can force MySQL to prompt for it. Run the following command and keep in mind that the installer will ask you for the password:

$ sudo apt-get -y install mysql-server mysql-client

Once done, if you need to start the MySQL server, you can do it with the following command:

$ sudo service mysql start

Installing Nginx

The first thing that you need to know is that you can only have one web server listening on the same port. As port 80 is the default one for web applications, if you are running Apache on your Ubuntu machine, you will not be able to start an Nginx web server listening on the same port 80. To fix this, you can either change the ports for Nginx or Apache, stop Apache, or uninstall it. Either way, the installation command for Nginx is as follows:

$ sudo apt-get install nginx -y
Now, you will need to enable a site with Nginx. The sites are files under `/etc/nginx/sites-available`. There is already one file there, `default`, which you can safely replace with the following content:

```nginx
classic {
    listen 80 default_server;
    listen [::]:80 default_server ipv6only=on;

    root /var/www/html;
    index index.php index.html index.htm;

    server_name server_domain_or_IP;

    location / {
        try_files $uri $uri/ /index.php?$query_string;
    }

    location ~ \.(php|py|pyc)$ {
        try_files $uri $uri/ /index.php =404;
        fastcgi_split_path_info ^(.+\.(php|py|pyc))(.+)$;
        fastcgi_pass unix:/var/run/php/php7.0-fpm.sock;
        fastcgi_index index.php;
        fastcgi_param SCRIPT_FILENAME $document_root$fastcgi_script_name;
        include fastcgi_params;
    }
}
```

This configuration basically points the root directory of your web application to the `/var/www/html` directory. You can choose the one that you prefer, but make sure that it has the right permissions. It also listens on the port 80, which you can change with the one you prefer; just keep this in mind that when you try to access your application via a browser. Finally, to apply all the changes, run the following command:

```
$ sudo service nginx restart
```
Setting Up the Environment

Downloading the example code

You can download the example code files for this book from your account at http://www.packtpub.com. If you purchased this book elsewhere, you can visit http://www.packtpub.com/support and register to have the files e-mailed directly to you.

You can download the code files by following these steps:

• Log in or register to our website using your e-mail address and password.
• Hover the mouse pointer on the SUPPORT tab at the top.
• Click on Code Downloads & Errata.
• Enter the name of the book in the Search box.
• Select the book for which you're looking to download the code files.
• Choose from the drop-down menu where you purchased this book from.
• Click on Code Download.

Once the file is downloaded, please make sure that you unzip or extract the folder using the latest version of:

• WinRAR / 7-Zip for Windows
• Zipeg / iZip / UnRarX for Mac
• 7-Zip / PeaZip for Linux

Summary

In this chapter, you learned how easy it is to set up a development environment using Vagrant. If this did not convince you, you still got the chance to set up all the tools manually. Either way, now you are able to work on the next chapters.

In the next chapter, we will take a look at the idea of web applications with PHP, going from the protocols used to how the web server serves requests, thus setting the foundation for the following chapters.
Where to buy this book

You can buy Learning PHP 7 from the Packt Publishing website.
Alternatively, you can buy the book from Amazon, BN.com, Computer Manuals and most internet book retailers.
Click here for ordering and shipping details.