Chapter 1

Dr. Matt Smith

January 21, 2018

Introduction

What is Symfony?

It’s a PHP ‘framework’ that does loads for you, if you’re writing a secure, database-drive web application.

What to I need on my computer to get started?

I recommend you install the following:

- PHP 7.1.3 (on windows Laragon works pretty well)[1]
- a database driver enabled for your PHP setup
  – for a quick-start just enable the SQLite driver in `php.ini`
  – when you have time, get youreIf a MySQL database server (e.g. MySQLWorkbench or MariaDB etc.)
- a good text editor (I like PHPStorm, but then it's free for educational users…)
- Composer (PHP package manager - on windows Laragon works pretty well)
  – ensure your version of Composer is up-to-date, so run `composer self-update` at the CLI

or … you could use something like Cloud9, web-based IDE. You can get started on the free version and work from there …

[1] See the The Symfony requirements web page for current PHP version neeed.

How to I get started?

Either:

- install the Symfony command line, then create a project like this (to create a new project in a directory named `project01`):

```
$ symfony new project01
```

or

- use Composer to create a new blank project for you, like this (to create a new project in a directory named `project01`):

```
$  composer create-project symfony/framework-standa
```

Learn about both these methods at:

- the Symfony download-installer page
- the Symfony setup page

or

- download one of the projects accompanying this book

Where are the projects accompanying this book?

There are on Github:

- `https://github.com/dr-matt-smith/php-symfony3-book-codes`

Download a project (e.g. `git clone URL`), then type `composer update` to download 3rd-party packages into a `/vendor` folder.

How to I run a Symfony webapp?

From the CLI

If you're not using a database engine like MySQL, then you can use the Symfony console command to 'serve up' your Symfony project from the command line
At the CLI (command line terminal) ensure you are at the base level of your project (i.e. the same directory that has your `composer.json` file), and type the following[2]:

```
$ php bin/console server:run
```

Webserver

If you are running a webserver (or combined web and database server like XAMPP or Laragon), then point your web server root to the `/web` folder - this is where public files go in Symfony projects.

[2]Since you'll be typing this a lot when testing, you could add a `script` shortcut in your `composer.json` file. I have one named `run`, so I can run

It isn't working! (Problem Solving)

If you have trouble with running Symfony, take a look at Appendix **??**, which lists some common issues and how to solve them.

Opaque 500 Server error message

If Symfony thinks you are in **production** (live public website) then when an error occurs it will throw a 500 server error (which a real production site would catch and display some nicely sanitised message for website visitors).
Since we are in **development** we want to see the **details** of any errors. So after a 500 Server error, refresh the browser page, but prepend `/app_dev.php/` to your URL - you'll then get a much more detailed description of the error (including the class / line / template causing the problem etc.).
Also, if you know where your error logs are stored, you can see the errors written to the log file…