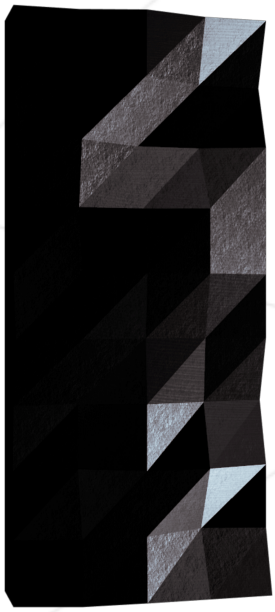

Training PHP Symfony - 1

Composer

Summary: Following 42 formation course, you will learn what Composer is and how you can use it in your applications.

Version: 1

Contents

Chapter I

Foreword

While writing applications in PHP from scratch, you will probably find that it feels like you have to keep re-inventing the wheel anytime you want to do a common task such as User Authentication, Database Management, etc. Here comes the Composer which is a dependency manager for PHP that will pull in all the required libraries, dependencies and manage them all in one place. This kind of management for dependencies in a project is not a new concept, and in fact, much of Composer is actually inspired from Node.js's NPM and Ruby's Bundler. In comparison with other former known package manager like PEAR, the Composer will allow you to install packages on a project-by-project basis rather than system wide.

Chapter II

General rules

- Your project must be realized in a virtual machine.
- Your virtual machine must have all the necessary software to complete your project. These softwares must be configured and installed.
- You can choose the operating system to use for your virtual machine.
- You must be able to use your virtual machine from a cluster computer.
- You must use a shared folder between your virtual machine and your host machine.
- During your evaluations you will use this folder to share with your repository.
- Your functions should not quit unexpectedly (segmentation fault, bus error, double free, etc) apart from undefined behaviors. If this happens, your project will be considered non functional and will receive a `0` during the evaluation.
- We encourage you to create test programs for your project even though this work **won't have to be submitted and won't be graded**. It will give you a chance to easily test your work and your peers' work. You will find those tests especially useful during your defence. Indeed, during defence, you are free to use your tests and/or the tests of the peer you are evaluating.
- Submit your work to your assigned git repository. Only the work in the git repository will be graded. If Deepthought is assigned to grade your work, it will be done after your peer-evaluations. If an error happens in any section of your work during Deepthought's grading, the evaluation will stop.

Chapter III

Day-specific rules

If no other explicit information is displayed, you must assume the following versions of languages :

- `PHP - Symfony LTS`
- `HTML 5`
- `CSS 3`

Chapter IV

Exercise 00

	Exercise 00
Exercise 00: Install Composer globally	
Turn-in directory : $ex00/$	
Files to turn in : `Files and folder from your application`	
Allowed functions : `All methods are allowed`	

This exercise is mandatory. To resolve all the others, you need to complete this one. The requirement for this exercise is to install ***Composer*** globally, from your terminal.

Chapter V

Exercise 01

	Exercise 01
Exercise 01: Specifying versions	
Turn-in directory : *ex*01/	
Files to turn in : `Files and folder from your application`	
Allowed functions : `All methods are allowed`	

This exercise will have many subtasks. For each one of them you need to create a new composer file.
For this exercise you need to install many versions of **Monolog** package, using **Composer**.
The requirements are:

1. version 2.3.0
2. version greater than 2.2.0 and less or equal with 2.3.5
3. version between 2.1.0 and 2.2.0
4. version greater or equal with 2.0.0 and less than 2.0.2
5. version greater than 2.0.0 and less than 2.3.5

Hint: *To resolve these exercises you need to use* ***composer install*** *command.*

Chapter VI

Exercise 02

	Exercise 02
Exercise 02: Development requirement	
Turn-in directory : $ex02/$	
Files to turn in : `Files and folder from your application`	
Allowed functions : `All methods are allowed`	

Install the version LTS of PHPUnit package as a development requirement.

Chapter VII

Exercise 03

	Exercise 03
Exercise 03: Composer install vs composer update	
Turn-in directory : *ex*03/	
Files to turn in : `composer.json, composer.lock, files and folder from your application`	
Allowed functions : `All methods are allowed`	

This exercise will have two subtasks. For each of them you need to create a subdirectory with the composer files.
Having the supplied **composer.json** and **composer.lock** files in your current directory, you need to run **composer install** and **composer update** commands. Observe which files are created/updated and what is the difference between *install* and *update*.
Steps to follow:

1. Copy the ***composer.json*** and ***composer.lock*** files in your current directory
2. Run **composer install** command
3. Run **composer update** command

Chapter VIII

Submission and peer-evaluation

Turn in your assignment in your `Git` repository as usual. Only the work inside your repository will be evaluated during the defense. Don't hesitate to double check the names of your folders and files to ensure they are correct.

`The evaluation process will happen on the computer of the evaluated group.`