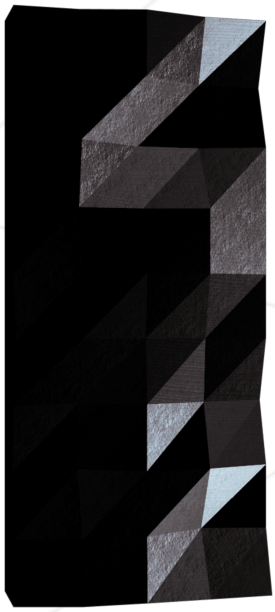

Formation PHP - Symfony

D08 - Deployment

Summary: Every application needs to be properly developed but deployment is an inevitable step when the application needs to get to a production environment.

Contents

Chapter I

Foreword

The void type, in several programming languages derived from C and Algol68, is the type for the result of a function that returns normally, but does not provide a result value to its caller. Usually such functions are called for their side effects, such as performing some task or writing to their output parameters. The usage of the void type in such context is comparable to procedures in Pascal and syntactic constructs which define subroutines in Visual Basic. It is also similar to the unit type used in functional programming languages and type theory.

Chapter II

General Rules

- This subject is the one and only trustable source. Don't trust any rumor.
- This subject can be updated up to one hour before the turn-in deadline.
- The assignments in a subject must be done in the given order. Later assignments won't be rated unless all the previous ones are perfectly executed.
- Be careful about the access rights of your files and folders.
- You must follow the `turn-in process` for each assignment. The url of your `GIT` repository for this day is available on your intranet.
- Your assignments will be evaluated by your Piscine peers.
- In addition to your peers evaluation, a program called the "Moulinette" should also evaluate your assignments. Fully automated, The Moulinette is tough and unforgiving in its evaluations. As a consequence, it is impossible to bargain your grade with it. Uphold the highest level of rigor to avoid unpleasant surprises.
- All shell assignments must run using `/bin/sh`.
- You must not leave in your turn-in repository any file other than the ones explicitly requested By the assignments.
- You have a question? Ask your left neighbor. Otherwise, try your luck with your right neighbor.
- Every technical answer you might need is available in the `mans` or on the Internet.
- Remember to use the Piscine forum of your intranet and also Slack!
- You must read the examples thoroughly. They can reveal requirements that are not obvious in the assignment's description.
- By Thor, by Odin! Use your brain!!!

Chapter III

Day-specific rules

- Every exercise should be resolved in a different directory.
- For solving each exercise you need to respect the requirements and naming the files properly. If one of these conditions are not met, no points will be given.

Chapter IV

Exercise 00

	Exercise 00
Exercise 00: Create the virtual machine	
Turn-in directory : *ex*00/	
Files to turn in : `Files and folder from your application`	
Allowed functions : `All methods are allowed`	

For this exercise you need to create a virtual machine (with vagrant) to simulate the production server. The vm needs to have installed ***"ubuntu/trusty64"*** box. The memory allocated for it should be set to ***1024***.

To have a functional server, you need to do the following on your vagrant machine:

- install *curl*
- install *php5*
- install *apache2*
- install *mysql*
- install *git*
- install *composer*

All these commands will be added in a provision file, that will be passed to vagrant.

Note:
The provision's file name should be **create_server.sh**.
To prevent *"stdin: is not a tty"* warning, you should add these lines in your provision file:

```
config.vm.provision "fix-no-tty", type: "shell" do |s|
   s.privileged = false
   s.inline = "sudo sed -i '/tty/!s/mesg n/tty -s \\&\\& mesg n/' /root/.profile"
```

Chapter V

Exercise 01

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

This exercise is dedicated to ***deployment*** part. For this action we chose **deployer.php**, a library for automatic deployment on the production server. Having the production server configured on the previous exercise, we need to add our project on it.
The requirement for this exercise is to add the **deploy.php** file on the current project. This file will be configured to deploy the **Symfony-Repo** (https://github.com/symfony/symfony-standard.git) on your vagrant machine. The server name will be **production**. The path for your project on server will be **/var/www/production** and the **master** branch will be deployed.

To access your production server, you need to configure a virtual host for your vagrant machine. This action could be added in the provision file for your vagrant.

Note:
The first step of this exercise is to copy the VagrantFile from the previous exercise in order to create your production server.
To obtain the vagrant configs, please use **"vagrant ssh-config"** command.

Chapter VI

Exercise 02

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

Create a new **task** on **deploy.php** file which will update a **VERSION.txt** file with the latest hash commit. The task's name will be **"update:hash:version"**.