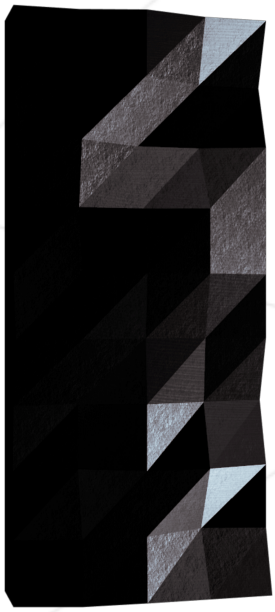

Formation PHP - Symfony

D01 - PHP Language Basics

Summary: Today, you will learn about the basics of the PHP Programming Language.

Contents

Chapter I

Foreword

Some wise quotes from the past:

640K ought to be enough for anybody.

Computers in the future may weigh no more than 1.5 tons.

We will never make a 32-bit operating system.

Spam will be a thing of the past in two years' time.

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

Exercise 00

	Exercise 00
Exercise 00: Var	
Turn-in directory : $ex00/$	
Files to turn in : `var.php`	
Allowed functions :	

Create a file named `var.php`, containing four variables a, b, c and d. At runtime, your progam initializes those variables and produces the following output:

```
# > php var.php
My first variables:
a contains : 10 and has type : integer
b contains : 10 and has type : string
c contains : ten and has type : string
d contains : 10 and has type : double
# >
```

You are not allowed to hard-code your variables' types in this program.

`Changing the values (and only the values) of a, b, c or d must change the output in a significant way.`

Chapter IV

Exercise 01

	Exercise 01
Exercise 01: CSV	
Turn-in directory : $ex01/$	
Files to turn in : `csv.php`	
Allowed functions :	

Create a `csv.php` file that will read a `ex01.txt` file, available in the same directory (and also available in the resources of this project).

The text file contains values separated by commas. Your program will read the content of this file and display each value on a new line.

```
# > cat ex01.txt
first,second,third,fourth
# > php csv.php
first
second
third
fourth
# >
```

Chapter V

Exercise 02

	Exercise 02
Exercise 02: Old times	
Turn-in directory : $ex02/$	
Files to turn in : `array2hash.php`	
Allowed functions :	

Create an `array2hash` function that takes an array, containing one or more arrays, as argument. These arrays contain a *name* string and an *age* integer each.

`array2hash` converts this array of arrays in a `hash` where keys are ages and values are names.

Example:

```
#> cat test02.php
<?php
  include('./array2hash.php');
  $array = array(array("Pierre","30"), array("Mary","28"));
  print_r ( array2hash($array) );
#> php test02.php
Array
(
    [30] => Pierre
    [28] => Mary
)
```

Chapter VI

Exercise 03

	Exercise 03
Exercise 03: Sorted times	
Turn-in directory : $ex03/$	
Files to turn in : `array2hash_sorted.php`	
Allowed functions :	

Create an `array2hash_sorted` function that takes an array, containing one or more arrays, as argument. These arrays contain a *name* string and an *age* integer each.

`array2hash_sorted` converts this array of arrays into a `hash` where keys are now names and values are now ages. This hash must be sorted by reverse alphabetical order.

Exemple:

```
#> cat test03.php
<?php
  include('./array2hash_sorted.php');
  $array = array(array("Pierre","30"), array("Mary","28"), array("Nelly", "22"));
  print_r ( array2hash_sorted($array) );
#> php test03.php
Array
(
    [Nelly] => 22
    [Mary] => 28
    [Pierre] => 30
)
```

Chapter VII

Exercise 04

	Exercise 04
Exercise 04: States & Capitals	
Turn-in directory : $ex04/$	
Files to turn in : `capital_city_from.php`	
Allowed functions :	

Using the following arrays, write a `capital_city_from` function that takes as argument the name of a state and returns its capital city. If the capital city doesn't exist, it returns "Unknown".

```
#> cat capital_city_from.php
<?php
[...]
$states = [
  'Oregon' => 'OR',
  'Alabama' => 'AL',
  'New Jersey' => 'NJ',
  'Colorado' => 'CO',
];
$capitals = [
  'OR' => 'Salem',
  'AL' =>  'Montgomery',
  'NJ' => 'trenton',
  'KS' => 'Topeka',
];
[...]
#> cat test04.php
<?php
  include('./capital_city_from.php');
  echo capital_city_from('Oregon');
  echo capital_city_from('Origan');
#> php test04.php
Salem
Unknown
```

Chapter VIII

Exercise 05

	Exercise 05
Exercise 05: Searching states or capitals	
Turn-in directory : $ex05/$	
Files to turn in : `search_by_states.php`	
Allowed functions :	

Using the same arrays as in the previous exercise, create a `search_by_states` function that takes a string, composed of one of more states, as argument.

This function will return a set of strings formated like this:

```
# > cat test05.php
<?php
  include('./search_by_states.php');
  search_by_states("Oregon, trenton, Topeka, NewJersey");
# > php test05.php
Salem is the capital of Oregon.
trenton is the capital of New Jersey.
Topeka is neither a capital nor a state.
NewJersey is neither a capital nor a state.
# >
```

Chapter IX

Exercise 06

	Exercise 06
Exercise 06: Mendeleiev table	
Turn-in directory : $ex06/$	
Files to turn in : `mendeleiev.php`	
Allowed functions :	

Create a program that opens the `ex06.txt` file that contains the Mendeleiev table, formated in a specific way. This program will create the file `mendeleiev.html` that will contain the HTML code of the Mendeleiev table with the data included in `ex06.txt`, following these rules:

- Each element should be a cell of an HTML **table**
- The title should be inside an **h4** tag
- The attributes should be inside an **HTML list**
- You should respect the general format of the Mendeleiev table, eg empty cells, new rows etc.

You can find some Mendeleiev tables easily on the Internet.

Sample output:

```
[...]
  <table>
    <tr>
      <td style="border: 1px solid black; padding:10px">
        <h4>Hydrogen</h4>
            <ul>
                <li>No 42</li>
                <li>H</li>
                <Li> 1.00794 </ li>
                <li>1 electron</li>
            <ul>
      </td>
[...]
```

Feel free to customize it however you see fit to make it look fancier ;)