



# Training PHP Symfony - 0

## Starting

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

*Version: 1.2*

# Contents

<b>I</b>	<b>Foreword</b>	<b>2</b>
<b>II</b>	<b>General rules</b>	<b>3</b>
<b>III</b>	<b>Day-specific rules</b>	<b>4</b>
<b>IV</b>	<b>Exercise 00</b>	<b>5</b>
<b>V</b>	<b>Exercise 01</b>	<b>6</b>
<b>VI</b>	<b>Exercise 02</b>	<b>7</b>
<b>VII</b>	<b>Exercise 03</b>	<b>8</b>
<b>VIII</b>	<b>Exercise 04</b>	<b>9</b>
<b>IX</b>	<b>Exercise 05</b>	<b>10</b>
<b>X</b>	<b>Exercise 06</b>	<b>11</b>
<b>XI</b>	<b>Submission and peer-evaluation</b>	<b>13</b>

# 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

- 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: Var	
Turn-in directory : <i>ex00/</i>	
Files to turn in : <b>var.php</b>	
Allowed functions :	

Create a file named `var.php`, containing four variables `a`, `b`, `c` and `d`. At runtime, your program 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 V

## Exercise 01

	Exercise 01
Exercise 01: CSV	
Turn-in directory : <i>ex01/</i>	
Files to turn in : <b>csv.php</b>	
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 VI

## Exercise 02

	Exercise 02
Exercise 02: Old times	
Turn-in directory : <i>ex02/</i>	
Files to turn in : <b>array2hash.php</b>	
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 VII

## Exercise 03

	Exercise 03
Exercise 03: Sorted times	
Turn-in directory : <i>ex03/</i>	
Files to turn in : <b>array2hash_sorted.php</b>	
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.


The `array2hash_sorted` function should convert this array of arrays into an associative array (or hash) where the keys are the names and the values are the corresponding ages. The resulting hash must be sorted by the keys (names) in 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
(
    [Pierre] => 30
    [Nelly] => 22
    [Mary] => 28
)
```

# Chapter VIII

## Exercise 04


	Exercise 04
Exercise 04: States & Capitals	
Turn-in directory : <i>ex04/</i>	
Files to turn in : <code>capital_city_from.php</code>	
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 IX

## Exercise 05

	Exercise 05
Exercise 05: Searching states or capitals	
Turn-in directory : <i>ex05/</i>	
Files to turn in : <b>search_by_states.php</b>	
Allowed functions :	

Using the same arrays as in the previous exercise, create a **search\_by\_states** function that takes a string, composed of one or more state names or capital names, as an argument.


The string should use a comma (,) as the separator, and there may be spaces around the separators, at the beginning, or at the end of the string.

This function will return an array of strings formatted like this:

```
# > cat test05.php
<?php
include('./search_by_states.php');
$results = search_by_states("Oregon, trenton, Topeka, NewJersey");
foreach ($results as $result)
{
    echo $result . "\n";
}
# > 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 X

## Exercise 06

	Exercise 06
Exercise 06: Mendeleiev table	
Turn-in directory : <i>ex06/</i>	
Files to turn in : <b>mendeleiev.php</b>	
Allowed functions :	

Create a program that opens the **ex06.txt** file that contains the Mendeleiev table, formatted 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>
  </tr>
</table>
[...]
```

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

# Chapter XI

## 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.