الجمهورية العربية السورية
جامعة دمشق
كلية الهندسة المعلوماتية

Programming Languages

Lecture Two

PHP Laravel – Installation

&

PHP Basics

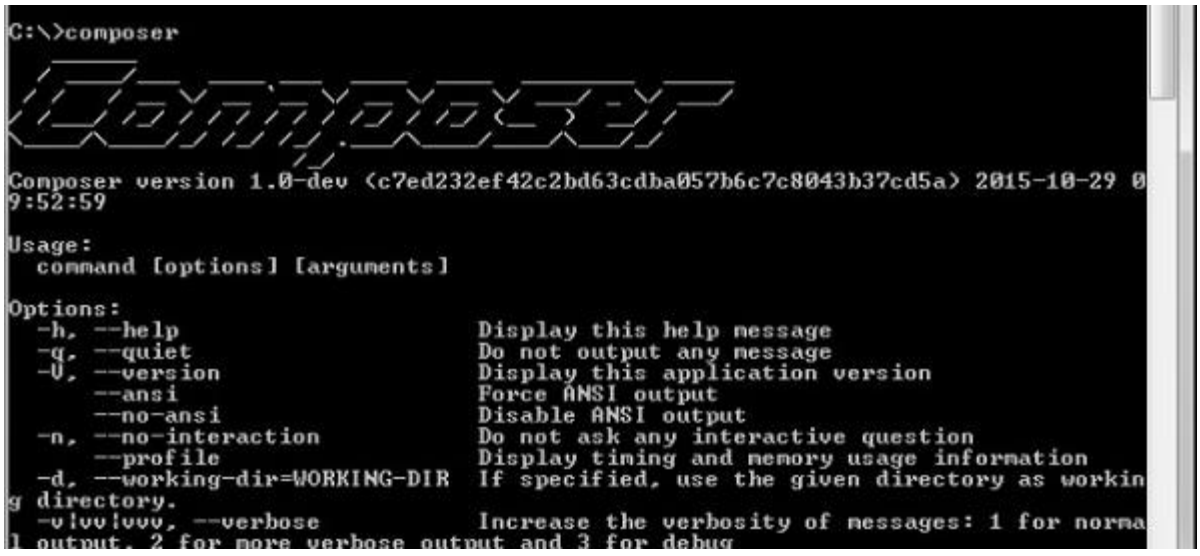
Prepared By

Eng. Rawan Koroni

Eng. Ahmed Al-Ahmed

1- PHP Laravel Installation (Use VPN)

- Install XAMPP as it is the most popular PHP development environment:
  1. Go to https://www.apachefriends.org/index.html
  2. Download the version according to your operating system.
  3. Install it on you device.
- Install composer as it is a tool for dependency management in PHP. It allows you to declare the libraries your project depends on and it will manage (install/update) them for you.
  1. Go to https://getcomposer.org/download/
  2. Download the file, run, and install on your device.
  3. After the Composer is installed, check the installation by typing the Composer command in the command prompt as shown in the following screenshot:



4. Create a new directory anywhere in your system for your new Laravel project. After that, move to path where you have created the new directory and type the following command there to install Laravel:

```
composer create-project laravel/laravel --prefer-dist
```

5. Now, we will focus on installation of version 5.7. In Laravel version 5.7, you can install the complete framework by typing the following command

```
composer create-project laravel/laravel test dev-develop
```

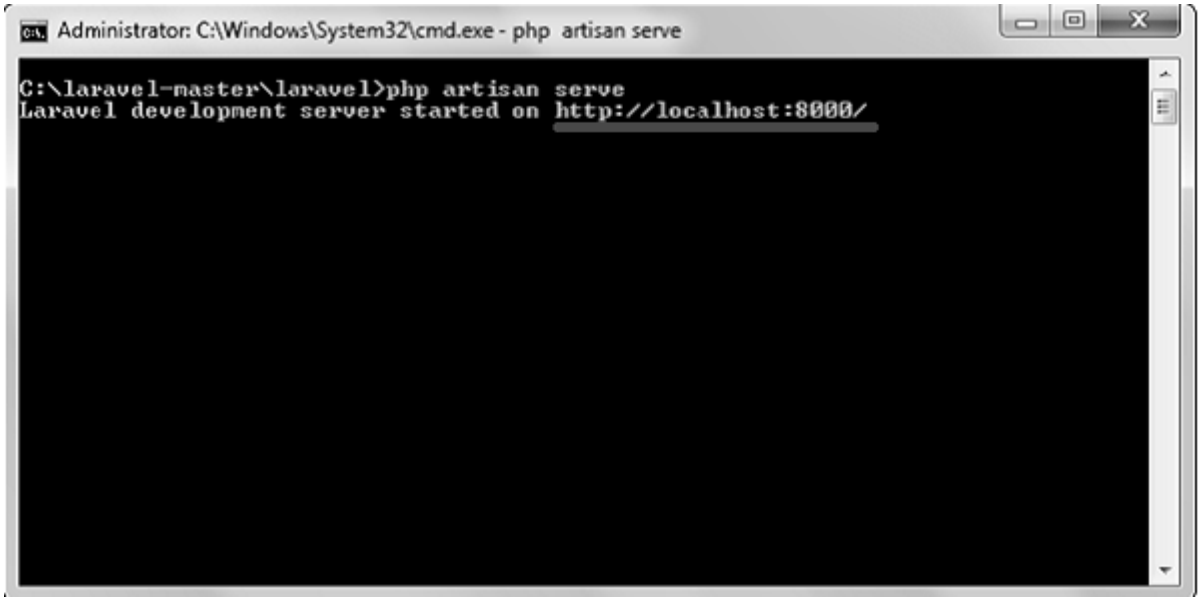
The output of the command is as shown below:

```
➜ code composer create-project laravel/laravel test dev-develop
Installing laravel/laravel (dev-develop d6acad21cb2288713d9c09a31f9b4ab86f116039)
  - Installing laravel/laravel (dev-develop develop): Cloning develop from cache
Created project in test
> @php -r "file_exists('.env') || copy('.env.example', '.env');"
Loading composer repositories with package information
Updating dependencies (including require-dev)
Package operations: 71 installs, 0 updates, 0 removals
  - Installing vlucas/phpdotenv (v2.5.1): Loading from cache
  - Installing symfony/css-selector (v4.1.3): Loading from cache
  - Installing tijsverkoyen/css-to-inline-styles (2.2.1): Loading from cache
  - Installing symfony/polyfill-php72 (v1.9.0): Loading from cache
  - Installing symfony/polyfill-mbstring (v1.9.0): Loading from cache
  - Installing symfony/var-dumper (v4.1.3): Loading from cache
  - Installing symfony/routing (v4.1.3): Loading from cache
  - Installing symfony/process (v4.1.3): Loading from cache
  - Installing symfony/polyfill-ctype (v1.9.0): Loading from cache
  - Installing symfony/http-foundation (v4.1.3): Loading from cache
  - Installing symfony/event-dispatcher (v4.1.3): Loading from cache
  - Installing psr/log (1.0.2): Loading from cache
  - Installing symfony/debug (v4.1.3): Loading from cache
  - Installing symfony/http-kernel (v4.1.3): Loading from cache
  - Installing paragonie/random_compat (v9.99.99): Loading from cache
```

6. The above command will install Laravel in the current directory. Start the Laravel service by executing the following command:

```
php artisan serve
```

7. After executing the above command, you will see a screen as shown below:



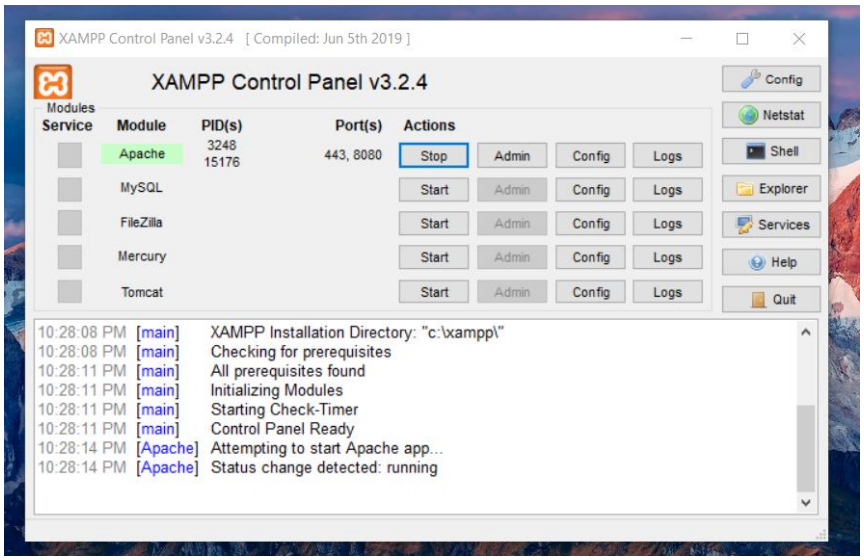
8. Copy the URL underlined in gray in the above screenshot and open that URL in the browser. If you see the following screen, it implies Laravel has been installed successfully.

2- What is PHP?

The PHP Hypertext Preprocessor (PHP) is a programming language that allows web developers to develop web based software applications.

3- Hello world example

- Run XAMPP:



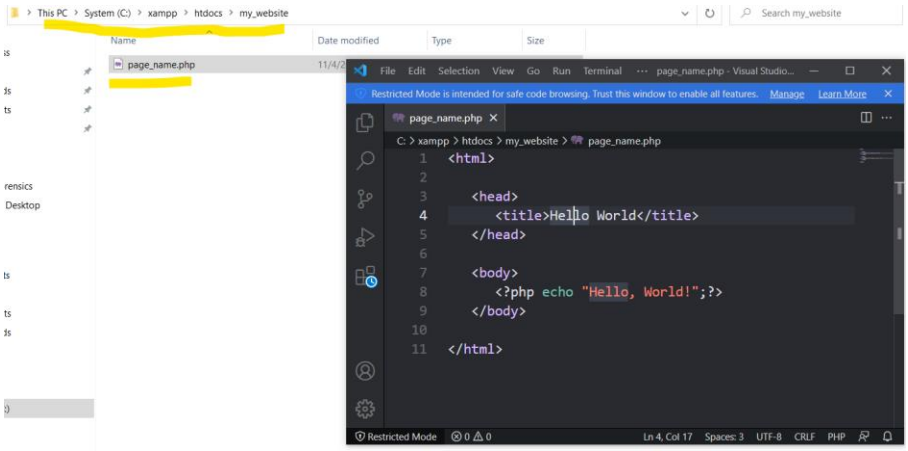
- Start Apache:

  Just click on the button `start` next to Apache services (by default the port is 80 but in my case I changed it to 8080).

- Create test.php (or any name you want) file with the following code:

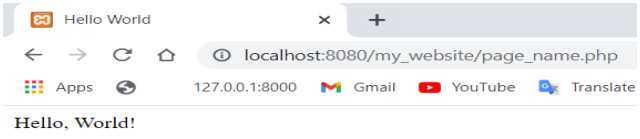
```
<html>
   <head>
      <title>Hello World</title>
   </head>

   <body>
      <?php echo "Hello, World!";?>
   </body>
</html>
```

- Copy your file to the following path:



- Open your browser and open the following url (will be changed depending on your Apache port, and your file path and name): http://localhost:8080/my_website/page_name.php

- And you should see the following result



4- Escaping to PHP

The PHP parsing engine needs a way to differentiate PHP code from other elements in the page. The mechanism for doing so is known as 'escaping to PHP', and The most universally effective PHP tag style is:

```
<?php...?>
```

5- Commenting PHP Code

There are two commenting formats in PHP

- Single-line comments

```
<?
   # This is a comment, and
   # This is the second line of the comment

   // This is a comment too. Each style comments only
   print "An example with single line comments";
?>
```

- Multi-lines comments

```
<?
   /* This is a comment with multiline
      Author : Mohammad Mohtashim
      Purpose: Multiline Comments Demo
      Subject: PHP
   */

   print "An example with multi line comments";
?>
```

6- PHP is case sensitive

PHP is a case sensitive language. Try out following example:

PHP Code

```
<html>
   <body>

      <?php
         $capital = 67;
         print("Variable capital is $capital<br>");
         print("Variable CaPiTaL is $CaPiTaL<br>");
      ?>

   </body>
</html>
```

Result

```
Variable capital is 67
Variable CaPiTaL is
```

7- Statements

A statement in PHP is any expression that is followed by a semicolon (;).Any sequence of valid PHP statements that is enclosed by the PHP tags is a valid PHP program. Here is a typical statement in PHP, which in this case assigns a string of characters to a variable called $greeting:

```
$greeting = "Welcome to PHP!";
```

8- Braces make blocks

Although statements cannot be combined like expressions, you can always put a sequence of statements anywhere a statement can go by enclosing them in a set of curly braces.

Here both statements are equivalent:

```
if (3 == 2 + 1)
   print("Good - I haven't totally lost my mind.<br>");

if (3 == 2 + 1) {
   print("Good - I haven't totally");
   print("lost my mind.<br>");
}
```

9- PHP - Variable Types

PHP has a total of eight data types which we use to construct our variables:

- **Integers** − are whole numbers, without a decimal point, like 4195.
- **Doubles** − are floating-point numbers, like 3.14159 or 49.1.
- **Booleans** − have only two possible values either true or false.
- **NULL** − is a special type that only has one value: NULL.
- **Strings** − are sequences of characters, like 'PHP supports string operations.'
- **Arrays** − are named and indexed collections of other values.
- **Objects** − are instances of programmer-defined classes, which can package up both other kinds of values and functions that are specific to the class.
- **Resources** − are special variables that hold references to resources external to PHP (such as database connections).

The first five are simple types, and the next two (arrays and objects) are compound - the compound types can package up other arbitrary values of arbitrary type, whereas the simple types cannot.

Examples:

Integers:

```php
<?php
   $int_var = 12345;
   $another_int = -12345 + 12345;
?>
```

Doubles:

```php
<?php
   $many = 2.2888800;
   $many_2 = 2.2111200;
   $few = $many + $many_2;

   print("$many + $many_2 = $few <br>");
?>
```

Strings:

PHP Code	
	<?php $variable = "name"; $literally = 'My $variable will not print!'; print($literally); print " "; $literally = "My $variable will print!"; print($literally); ?>
Result	My $variable will not print! My name will print

10- PHP - Constants Types

A constant is a name or an identifier for a simple value. A constant value cannot change during the execution of the script.

Example:

```
<?php
   define("MINSIZE", 50);

   echo MINSIZE;
   echo constant("MINSIZE"); // same thing as the previous line
?>
```

11- PHP - Operator Types

PHP language supports following type of operators:

- Arithmetic Operators (+, -, *, /, %, ++, --)
- Comparison Operators (==, !=, >, <. >=, <=)
- Logical (or Relational) Operators (and, or, &&, ||, !)
- Assignment Operators (=, +=, -=, *=, /=, %=)
- Conditional (or ternary) Operators (? :), example:

  (If Condition is true ? Then value X : Otherwise value Y)

12- PHP - Decision Making

If Else Example:

PHP Code

```
<html>
   <body>

      <?php
         $d = date("D");

         if ($d == "Fri")
            echo "Have a nice weekend!";

         else
            echo "Have a nice day!";
      ?>

   </body>
</html>
```

Result

```
Have a nice weekend!
```

ElseIf Example:

PHP Code

```
<html>
   <body>
      <?php
         $d = date("D");

         if ($d == "Fri")
            echo "Have a nice weekend!";

         elseif ($d == "Sun")
            echo "Have a nice Sunday!";

         else
            echo "Have a nice day!";
      ?>
   </body>
</html>
```

Result

```
Have a nice weekend!
```

Switch Example:

PHP Code

```
<html>
   <body>
      <?php
         $d = date("D");

         switch ($d){
            case "Mon":
               echo "Today is Monday";
               break;

            case "Tue":
               echo "Today is Tuesday";
               break;

            case "Wed":
               echo "Today is Wednesday";
               break;

            case "Thu":
               echo "Today is Thursday";
               break;

            case "Fri":
               echo "Today is Friday";
               break;

            case "Sat":
               echo "Today is Saturday";
               break;

            case "Sun":
               echo "Today is Sunday";
               break;

            default:
               echo "Wonder which day is this ?";
         }
      ?>
   </body>
</html>
```

Result

```
Today is Monday
```

13- PHP - Loop Types

Loops in PHP are used to execute the same block of code a specified number of times. PHP supports following four loop types:

- **for** – loops through a block of code a specified number of times.
- **while** – loops through a block of code if and as long as a specified condition is true.
- **do...while** – loops through a block of code once, and then repeats the loop as long as a special condition is true.
- **foreach** – loops through a block of code for each element in an array.

For Loop Example:

PHP Code

```
<html>
   <body>

      <?php
         $a = 0;
         $b = 0;

         for( $i = 0; $i<5; $i++ ) {
            $a += 10;
            $b += 5;
         }

         echo ("At the end of the loop a = $a and b = $b" );
      ?>

   </body>
</html>
```

Result

```
At the end of the loop a = 50 and b = 25
```

- While Loop Example:

PHP Code

```
<html>
   <body>

      <?php
         $i = 0;
         $num = 50;

         while( $i < 10) {
            $num--;
            $i++;
         }

         echo ("Loop stopped at i = $i and num = $num" );
      ?>

   </body>
</html>
```

Result

```
Loop stopped at i = 10 and num = 40
```

- Do..While Example:

PHP Code

```
<html>
   <body>

      <?php
         $i = 0;
         $num = 0;

         do {
            $i++;
         }

         while( $i < 10 );
         echo ("Loop stopped at i = $i" );
      ?>

   </body>
</html>
```

Result

```
Loop stopped at i = 10
```

- Foreach Example:

PHP Code	<?php
	$array = array(1, 2, 3, 4, 5);
	
	foreach($array as $value) {
	echo "Value is $value
	";
	}
	?>

Result	Value is 1 Value is 2 Value is 3 Value is 4 Value is 5

14- Arrays

An array is a data structure that stores one or more similar type of values in a single value. For example if you want to store 100 numbers then instead of defining 100 variables its easy to define an array of 100 length.

There are three different kind of arrays:

- **Numeric array** – An array with a numeric index. Values are stored and accessed in linear fashion.
- **Associative array** – An array with strings as index. This stores element values in association with key values rather than in a strict linear index order.
- **Multidimensional array** – An array containing one or more arrays and values are accessed using multiple indices

- Numeric Array Example:

PHP Code

```
<html>
   <body>

      <?php
         /* First method to create array. */
         $numbers = array( 1, 2, 3, 4, 5);

         foreach( $numbers as $value ) {
            echo "Value is $value <br />";
         }

         /* Second method to create array. */
         $numbers[0] = "one";
         $numbers[1] = "two";
         $numbers[2] = "three";
         $numbers[3] = "four";
         $numbers[4] = "five";

         foreach( $numbers as $value ) {
            echo "Value is $value <br />";
         }
      ?>

   </body>
</html>
```

Result

```
Value is 1
Value is 2
Value is 3
Value is 4
Value is 5
Value is one
Value is two
Value is three
Value is four
Value is five
```

- Associative Array Example:

PHP Code

```
<html>
   <body>

      <?php
         /* First method to associate create array. */
         $salaries = array("mohammad" => 2000, "qadir" => 1000,
"zara" => 500);

         echo "Salary of mohammad is ". $salaries['mohammad'] .
"<br />";
         echo "Salary of qadir is ".  $salaries['qadir']. "<br
/>";
         echo "Salary of zara is ".  $salaries['zara']. "<br />";

         /* Second method to create array. */
         $salaries['mohammad'] = "high";
         $salaries['qadir'] = "medium";
         $salaries['zara'] = "low";

         echo "Salary of mohammad is ". $salaries['mohammad'] .
"<br />";
         echo "Salary of qadir is ".  $salaries['qadir']. "<br
/>";
         echo "Salary of zara is ".  $salaries['zara']. "<br />";
      ?>

   </body>
</html>
```

Result

```
Salary of mohammad is 2000
Salary of qadir is 1000
Salary of zara is 500
Salary of mohammad is high
Salary of qadir is medium
Salary of zara is low
```

- Multidimensional Array Example:

PHP Code

```
<html>
   <body>

      <?php
         $marks = array(
            "mohammad" => array (
               "physics" => 35,
               "maths" => 30,
               "chemistry" => 39
            ),

            "qadir" => array (
               "physics" => 30,
               "maths" => 32,
               "chemistry" => 29
            ),

            "zara" => array (
               "physics" => 31,
               "maths" => 22,
               "chemistry" => 39
            )
         );

         /* Accessing multi-dimensional array values */
         echo "Marks for mohammad in physics : " ;
         echo $marks['mohammad']['physics'] . "<br />";

         echo "Marks for qadir in maths : ";
         echo $marks['qadir']['maths'] . "<br />";

         echo "Marks for zara in chemistry : " ;
         echo $marks['zara']['chemistry'] . "<br />";
      ?>

   </body>
</html>
```

Result

```
Marks for mohammad in physics : 35
Marks for qadir in maths : 32
Marks for zara in chemistry : 39
```

15-String functions

There are many string function that we can use in PHP, for example:

PHP Code	<?php echo strlen("Hello world!"); ?>
Result	12

PHP Code	<?php echo strpos("Hello world!","world"); ?>
Result	6