TP-14

PHP, Laravel

PHP – Short Introduction

PHP is a general-purpose server-side scripting language that is used to create dynamic and interactive web pages quickly

PHP Example

```
<!DOCTYPE html>
<html>
<head>
   <title>PHP Script Example</title>
</head>
<body>

<?php
   echo "Hello world, I'm a PHP script!"
?>

</body>
</html>
```

Program Output:

```
Hello world, I'm a PHP script!
```

PHP – Short Introduction

Some Facts About PHP

- PHP was developed by **Rasmus Lerdorf** in **1995** and is later being developed as an open-source.
- PHP has many syntaxes similar to C, Java, and Perl, and has many unique features and specific functions.
- PHP page is a file with a **.php** extension can contain a combination of HTML Tags and PHP scripts.
- **PHP recursive acronym for PHP(Hypertext Preprocessor)**: **HyperText** means, text containing all sorts of web markups, **PreProcessor** means all of the HyperText is processed first and then the result is sent as pure HTML to the web browser. A client cannot see the PHP source code because it is preprocessed and interpreted.
- **PHP is Server-side scripting language**: Server-side scripting means that the PHP code is processed on the web server rather than the client machine.
- PHP supports many databases (MySQL and PHP combination is widely used).
- PHP is an open-source scripting language.
- PHP is free to download and use.

PHP – Short Introduction

To run PHP code, you need the following three software on your local machine:

1. Web Server (e.g., Apache)
2. PHP (Interpreter)
3. MySQL Databases (optional)

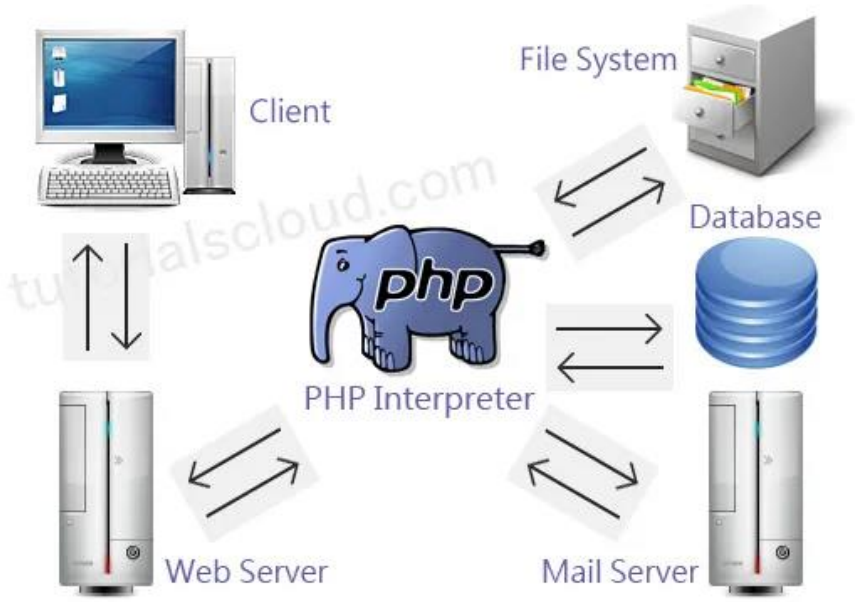
- WAMP (Windows, Apache, MySQL, PHP)
- LAMP (Linux, Apache, MySQL, PHP)
- MAMP (MAC, Apache, MySQL, PHP)
- XAMPP (Windows/Linux/MAC, Apache, MySQL, PHP)

The php.ini File

php.ini is a plain text file that configures PHP settings. PHP interpreter reads the php.ini file to determine what settings to use.

PHP – Short Introduction

How PHP works?



PHP Tags

Syntax:

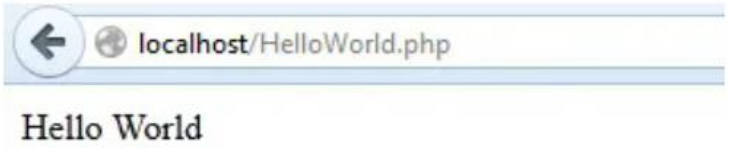
```
<?php
    //your php code goes here
?>
```

PHP with HTML

Example:

```
<html>
<title>Hello World program in PHP</title>
<body>
<?php echo "Hello World"; ?>
</body>
</html>
```

Program Output:



PHP – Short Introduction

Numeric Data Type

- Integer

Example:

```php
<?php
$intValue = 100;
?>
```

- Double

Example:

```php
<?php
$doubValue = 55.5;
?>
```

- String

Example:

```php
<?php
    $strName = "Neel";
    $strId = "Neel456";
?>
```

- Converting Between Data Types

Example:

```php
<?php
    $a = 9.88;
    echo (int) $a; // Outputs 9
?>
```

PHP – Short Introduction

Variable Definition and Initialization

Example:

```php
<?php
   $me = "I am David";
   echo $me;
   $num = 24562;
   echo $num;
   $name = "David";  //Valid variable name
   $_name = "Alex";  //Valid variable name
   $1name = "Jhon";  //Invalid variable name, starts with a number
?>
```

Constant Definition in PHP

Example:

```php
<?php
    define("EMAIL", "me@example.com"); // Valid constant name
    echo EMAIL; // Displays "me@example.com"

    define("myCon", true);
    if (myCon) { } // Evaluates to true

    define("ONECONSTANT", "some value"); // Invalid constant name

    define("CONSTANT", "Hello world.");
    echo CONSTANT; // outputs "Hello world."
    echo Constant; // outputs "Constant" and issues a notice.

    define("GREETING", "Hello world.", true);
    echo GREETING; // outputs "Hello world."
    echo Greeting; // outputs "Hello world."
?>
```

PHP – Short Introduction

String Concatenation in PHP

```php
<?php
    $like = "I like php";
    $num = 7;
    echo $like . $num;

    echo "<p>";
    echo $like . " " . $num;
    echo "</p>";

    echo "My favorite php version is $num";
?>
```

PHP – Short Introduction

Operators in PHP

Arithmetic

Operator	Description
+	Addition
-	Subtraction
*	Multiplication
/	Division
%	Modulus
++	Increment
--	Decrement

Example:

```
<?php
    $var = 1;       // Assign the integer 1 to $var
    echo $var++;    // Print 1, $var is now equal to 2
    echo ++$var;    // Print 3, $var is now equal to 3
    echo --$var;    // Print 2, $var is now equal to 2
    echo $var--;    // Print 2, $var is now equal to 1
?>
```

Assignment

Operator	Description
=	Assign
+=	Increments then assigns
-=	Decrements then assigns
*=	Multiplies then assigns
/=	Divides then assigns
%=	Modulus then assigns

Example:

```
<?php
  $var = "value";  // $var now contains the string "value"
  $var = 1;    // $var now contains the integer value 1
  $var += 3;    //$var now contains the integer 4
?>
```

Comparison

Operator	Description
==	Is equal to
===	Identical
!=	Is not equal to
<>	Is not equal to
!==	Is Identical
>	Greater than
<	Less than
>=	Greater than or equal to
<=	Less than or equal to

https://www.w3schools.in/

PHP – Short Introduction

Operators in PHP

Logical

Operator	Description
&&	And operator performs logical conjunction on two expressions (if both expressions evaluate to True, result is True. If either expression evaluates to False, the result is False)
||	Or operator performs a logical disjunction on two expressions (if either or both expressions evaluate to True, the result is True).
!	Not operator performs logical negation on an expression.

Concatenation

Operator	Description
.	The PHP concatenation operator (.) is used to combine two string values to create one string.
.=	Concatenation assignment.

Example:

```php
<?php
    $name="John";
    $lastName="Travolta";
    echo $name." ".$lastName; // Outputs John Travolta

    $a="Hello";
    $a .= " John!";
    echo $a; // Outputs Hello John!
?>
```

PHP – Short Introduction

PHP Decision Making

If else statement

Example:

```php
<?php
$date=date("m-d");

if ($date=="01-10") {
  echo "Wishing you a very Happy Birthday";
}
else{
  //nothing
}
?>
```

Elseif statement

Example:

```php
<?php
$date=date("m-d");

if ($date=="01-10") {
  echo "Wishing you a very Happy Birthday";
}
elseif($date=="08-15"){
  echo "Happy Independence Day";
}
else{
  //nothing
}
?>
```

Switch statement

Example:

```php
<?php
$myFavColor='red';
switch ($myFavColor)
{
case 'pink':
  echo 'My favorite car color is pink!';
  break;
case 'red':
  echo 'My favorite car color is red!';
  break;
case 'orange':
  echo 'My favorite car color is orange!';
  break;
default:
  echo 'My favorite car color is not pink, red, or orange!';
}
?>
```

PHP – Short Introduction

PHP Loops

While loop

Example:

```
<?php
$i = 1;

while ($i <= 5){
  echo "Hello while $i times"."<br>";
  $i++;
}
?>
```

Do-while loop

Example:

```
<?php
$i=0;
do{
  $i++;
  echo "php do...while loop $i times"."<br>";
}
  while ($i<=5);
?>
```

For loop

Example:

```
<?php
for ($i=1; $i <= 5; $i++ ){
  echo "PHP for loop print $i times"."<br>";
}
?>
```

Foreach loop

Example:

```
<?php
$salary[]=2000;
$salary[]=3000;
$salary[]=5000;

foreach($salary as $value){
  echo "Salary: $value<br>";
}
?>
```

PHP – Short Introduction

PHP Array

- Numeric Array: Is an indexed array

Example:

```php
<?php
    $friends[0] = 'Jhon';
    $friends[1] = 'Ramson';
    $friends[2] = 'Nikita';
?>
```

or Example:

```php
<?php
    $friends = array('Jhon','Ramson','Nikita');
?>
```

```php
<?php
$age = array("Peter"=>"35", "Ben"=>"37", "Joe"=>"43");

foreach($age as $x => $val) {
  echo "$x = $val<br>";
}
?>
```

- Associative: used named keys

Example:

```php
<?php
    $salary['Jhon'] = 15000;
    $salary['Ramson'] = 25000;
    $salary['Nikita'] = 27000;
?>
```

or

```php
<?php
    $salary = array('John'=>15000, 'Ramson'=>25000, 'Nikita'=>27000);
?>
```

- Multi-dimensional: contains other array as value

Example:

```php
<?php
    $multiDArray = array(
    "A" => array(0 => "red", 2 => "blue", 3 => "green"),
    "B" => array(1 => "orange", 2 => "black"),
    "C" => array(0 => "white", 4 => "purple", 8 => "grey")
    );

    echo $multiDArray["A"][3]; // Outputs green
    echo "<br>";
    echo $multiDArray["C"][8]; // Outputs grey
?>
```

PHP – Short Introduction

PHP Functions

- include() Function

Syntax:

```
include ('fileName');
```

Example:

```
<?php include("header.php"); ?>
```

- require() Function

Syntax:

```
require ('fileName');
```

Example:

```
<?php require("menu.php"); ?>
```

Difference Between include and require Statements:

```
1  <?php require "my_variables.php"; ?>
2  <?php require "my_functions.php"; ?>
3  <!DOCTYPE html>
4  <html lang="en">
5  <head>
6      <title><?php displayTitle($home_page); ?></title>
7  </head>
8  <body>
9  <?php include "header.php"; ?>
10 <?php include "menu.php"; ?>
11     <h1>Welcome to Our Website!</h1>
12     <p>Here you will find lots of useful information.</p>
13 <?php include "footer.php"; ?>
14 </body>
15 </html>
```

PHP – Short Introduction

PHP Headers

PHP headers can perform certain things, some of them are listed below:

- Tell browser not to cache the pages.
- Content-Type declaration
- Page Redirection

- Redirecting Browser

```php
<?php
  header("Location: http://www.example.com/");
?>
```

- Do not cache pages

```php
<?php
  //Date in the past, tells the browser that the cache has expired
  header("Expires: Mon, 20 Feb 2005 20:12:03 GMT");

  /* The following tell the browser that the last modification is right not so it must load th
e page again */
  header("Last-Modified: ". gmdate("D, d M Y H:i:s"). "GMT");

  //HTTP/1.0
  header("Pragma: no-cache");
?>
```

- Content Types

```php
<?php
  //Browser will deal page as PDF
  header ( "Content-type: application/pdf" );

  //myPDF.pdf will called
  header ( "Content-Disposition: attachment; filename=myPDF.pdf' " );
?>
```

PHP – Short Introduction

PHP Cookie

- Create cookie

Syntax:

```
setcookie(name, value, expiration);
```

Example:

```
<?php setcookie("username", "Jhon", time()+3600); ?>
```

- Retrieve a Cookie Value

```
<?php echo $_COOKIE["username"]; ?>
```

- Delete a Cookie

```
<?php setcookie("username", "Jhon", time()-3600); ?>
```

PHP Session

- Starting/storing/retrieving a Session

```
<?php
session_start();

// store session data
$_SESSION["username"] = "nikita";
$_SESSION["email"] = "nikita@example.com";

//retrieve session data
echo $_SESSION["username"];
echo "<br>";
echo $_SESSION["email"];
?>
```

- Destroy a Session

Example:

```
<?php
session_start();

session_destroy();
?>
```

PHP – Short Introduction

PHP Forms

- PHP POST Form Handling

Example:

```
<html>
<body>

<form action="registration.php" method="post">
Name: <input type="text" name="name">
Email: <input type="text" name="email">
<input type="submit">
</form>

</body>
</html>
```

Submit to

registration.php

```
<html>
<body>

Welcome <?php echo $_POST["name"]; ?>!
Your email address is <?php echo $_POST["email"]; ?>

</body>
</html>
```

PHP – Short Introduction

PHP Forms

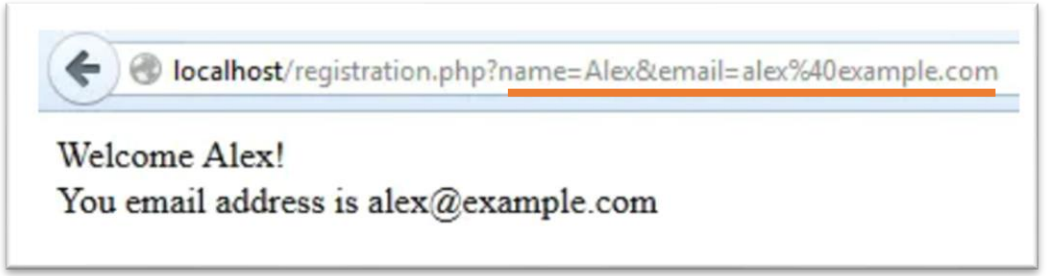
- PHP GET Form Handling

Example:

```
<html>
<body>

<form action="registration.php" method="get">
Name: <input type="text" name="name">
Email: <input type="text" name="email">
<input type="submit">
</form>

</body>
</html>
```



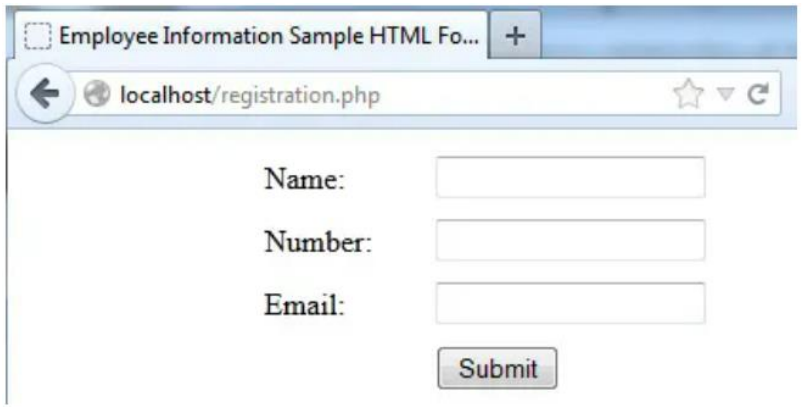
```
<html>
<body>

Welcome <?php echo $_GET["name"]; ?>!
Your email address is <?php echo $_GET["email"]; ?>

</body>
</html>
```

PHP – Short Introduction

Sample HTML Forms



Submit to

```php
<?php
if (isset($_POST['Submit'])) {

  $emp_name = trim($_POST["emp_name"]);
  $emp_number = trim($_POST["emp_number"]);
  $emp_email = trim($_POST["emp_email"]);

  if ($emp_name == "") {
    $errorMsg =  "error : You did not enter a name.";
    $code = "1";
  } elseif ($emp_number == "") {
    $errorMsg =  "error : Please enter number.";
    $code = "2";
  }
  //check if the number field is numeric
  elseif (is_numeric(trim($emp_number)) == false) {
    $errorMsg =  "error : Please enter numeric value.";
    $code = "2";
  } elseif (strlen($emp_number) < 10) {
    $errorMsg =  "error : Number should be ten digits.";
    $code = "2";
  }
  //check if email field is empty
  elseif ($emp_email == "") {
    $errorMsg =  "error : You did not enter a email.";
    $code = "3";
  } //check for valid email
  elseif (!preg_match("/^[_\.0-9a-zA-Z-]+@([0-9a-zA-Z][0-9a-zA-Z-]+\.)+[a-zA-Z]{2,6}$/i", $emp_email)) {
    $errorMsg = 'error : You did not enter a valid email.';
    $code = "3";
  } else {
    echo "Success";
    //final code will execute here.
  }
}
```

PHP – Short Introduction

PHP File Upload

- Create an HTML Upload-File Form

```
<form action="" method="post" enctype="multipart/form-data" name="form1">
<input type="file" name="resume" id="resume">
<input type="submit" name="SubmitBtn" id="SubmitBtn" value="Upload Resume">
</form>
```

Submit to

- $_FILES["file"]["name"] - uploaded file name
- $_FILES["file"]["type"] - uploaded file type
- $_FILES["file"]["size"] - uploaded file size in bytes
- $_FILES["file"]["tmp_name"] - uploaded file temporary file name
- $_FILES["file"]["error"] - the error code resulting from the file upload

```php
<?php
if (isset($_POST["SubmitBtn"])) {

  $fileName = $_FILES["resume"]["name"];
  $fileSize = $_FILES["resume"]["size"] / 1024;
  $fileType = $_FILES["resume"]["type"];
  $fileTmpName = $_FILES["resume"]["tmp_name"];

  if ($fileType == "application/msword") {
    if ($fileSize <= 200) {

      //New file name
      $random = rand(1111, 9999);
      $newFileName = $random . $fileName;

      //File upload path
      $uploadPath = "testUpload/" . $newFileName;

      //function for upload file
      if (move_uploaded_file($fileTmpName, $uploadPath)) {
        echo "Successful";
        echo "File Name :" . $newFileName;
        echo "File Size :" . $fileSize . " kb";
        echo "File Type :" . $fileType;
      }
    } else {
      echo "Maximum upload file size limit is 200 kb";
    }
  } else {
    echo "You can only upload a Word doc file.";
  }
}
```

PHP – Short Introduction

PHP Composer

- What Is Dependency Manager?

  A **software tool to manage (install, upgrade, configure, and remove)** the **various types of libraries required by a project** in a logical and meaningful way.

- What is PHP Composer? (is inspired by the node's npm)

  A **dependency manager or dependency management tool** specifically built for PHP.

- Downloading and Installing PHP Composer 

  - **Windows user** getcomposer.org
  - Linux/Unix/MacOS

    → **go to your new project folder** and enter the following command in the terminal

    ```
    $ curl -s https://getcomposer.org/installer | PHP
    ```

    → To ensure that the composer is successfully installed, execute the below command

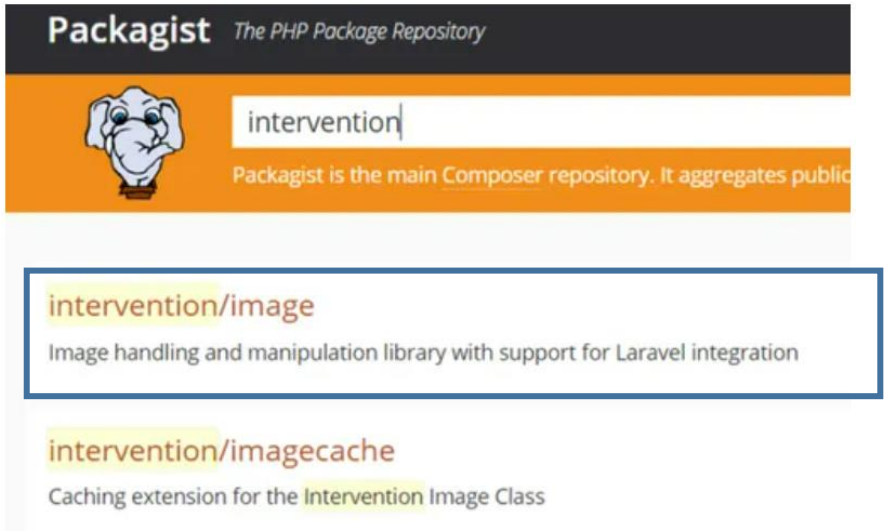
    ```
    $ composer
    ```

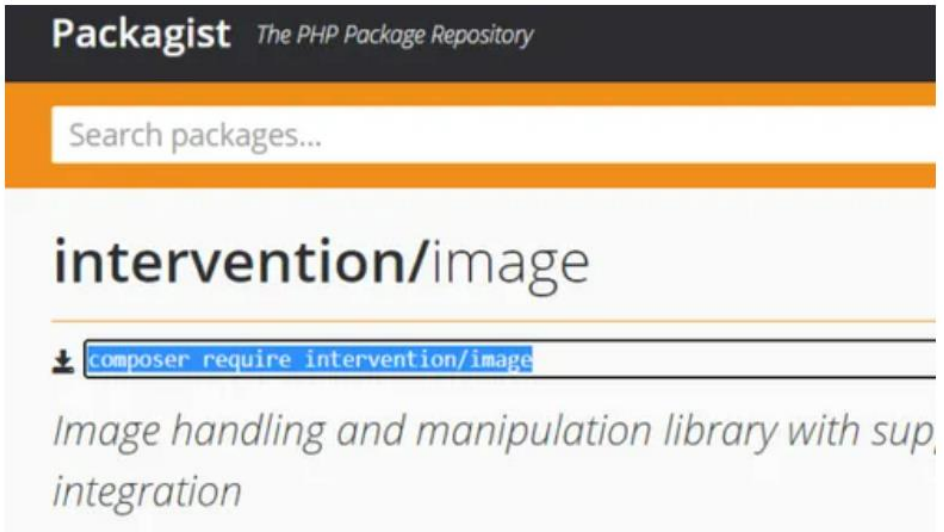
PHP – Short Introduction

PHP Composer

- Package Installation Using Composer

https://packagist.org/





→ To install a package inside your working project, execute a given command

Example:

```
$ composer require intervention/image
```

PHP – Short Introduction

PHP Composer

- Import the installed package to a file

Example:

```php
<?php

// include composer autoload
require "vendor/autoload.php";

// import the Intervention image manager class file
use Intervention\Image\ImageManagerStatic as Image;

$Image = Image::make("upload/image.jpg")->resize(200, 200)->save("img/thumbnail.jpg",100);

?>
```

Also, you can see that it will create a **composer.json** file inside the project folder

composer.json file:

```
{
    "require": {
        "intervention/image": "^2.5"
    }
}
```

→ To install packages from composer.json

```
$ composer install
```

Use all practical exercises as TP.

(Please practice only those exercises marked with red star ☆)

Submission :: Screenshot both source code and the result

Good luck 👌