SLIIT
Discover Your Future

Lecture 06

PHP - Introduction

IT1100 Internet and Web technologies

SLIIT
FACULTY OF COMPUTING

Content

- Introduction
- Variables and Constants
- Operators
- Control structures

Introduction

- PHP is a scripting language for developing server-side components

- The components developed with PHP should be hosted in a compatible web server
  - Apache, IIS

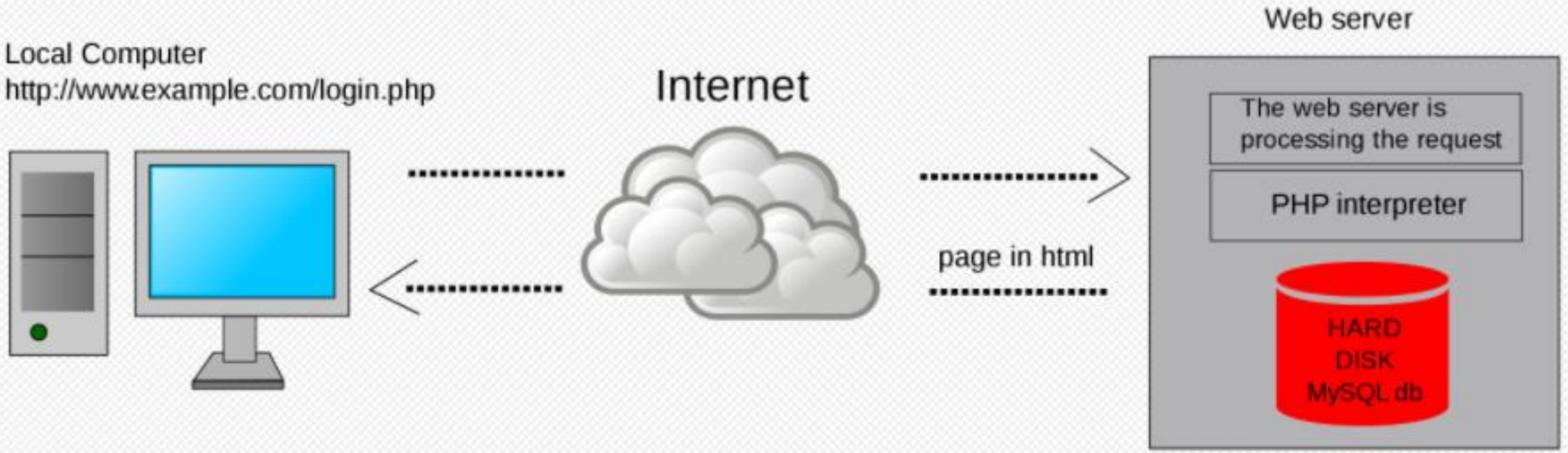
NOTE: You will learn to host PHP application and access it, in the practical class.

What Can PHP Do?

- Generate dynamic page content
- Create, open, read, write, delete, and close files on the server
- Collect form data
- Send and receive cookies
- Insert, delete, update or search data in your database
- Restrict users to access some pages on your website
- Encrypt data
- You can output
  - images, PDF files, Flash movies, text, XHTML and XML.

How to run your first .php file

1. Write and save php code as a .php file.
2. Copy .php file into the web server.
   - Ex.
     - C:\xampp\htdocs\ita_demo
3. Open a web browser
4. Type the URL and call your .php file
   - Ex
     - http://localhost/ita_demo/lec_1/Test.php



PHP Execution flow

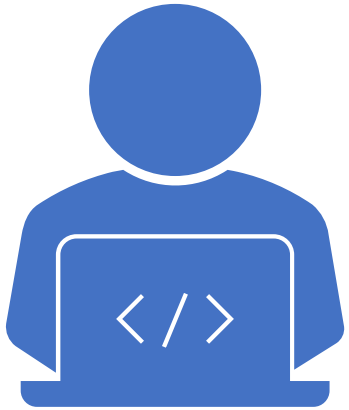
PHP Execution flow

1. The client requests a page that contains PHP tags.
2. The Web server will pass any page requests containing a PHP file extension to the PHP processor.
3. PHP processor scans the page and processes all PHP tags. Might also retrieve information from a database.
4. PHP processor returns only HTML and other client-side technologies to the Web server.
5. The Web server passes the page back to the browser

PHP Code

- The file/page with PHP script is saved to .php extension
- The script/code is written between the PHP tag

```
<?php
    echo “<h1>Hello world</h1>”;
    // The output may contain HTML
?>
```



PHP code can be written

1. Inline with HTML

   //Not recommended

2. On the top of the file

   //Like internal CSS sheets, JavaScripts

3. As an external file

   //Like external CSS sheets, JavaScripts

Inline with HTML

```
<html>
 <head></head>
 <body>
        <h1>PHP example</h1>
        <?php
            echo “<h1>Hello world</h1>”;
        ?>
 </body>
</html>
```

Output

PHP example

Hello world

PHP code on top of the page

```
<?php
        echo "<h1>Hello world</h1>";
?>
<html>
    <head></head>
    <body>
        <h1>PHP example</h1>
    </body>
 </html>
```

Output

Hello world

PHP example

PHP code in external file

index.php

```
<?php
  include("Logic.php"); //Link the external file
?>
<html>
  <head></head>
  <body>
          <h1>PHP example</h1>
</body>
</html>
```

Output

Hello world

PHP example

Logic.php

```
<?php
   echo "<h1>Hello world</h1>";
?>
```

Variables

- Variable names must begin with a “$”
- Must not start with a number or special characters ' < & , > ^
- Must not contain spaces.
- Must be less than 32 characters
  - **$3_name** – incorrect
  - **$name_** – correct
  - **$name it** – incorrect
  - **$name** – correct

Variables

- All user-defined functions, classes, and keywords are NOT case-sensitive.
- All variables are case-sensitive.
- Double quotes (") will replace a variable's name with its value.
  - $a=5;
  - echo "$a"; //Will display **5**
- Single quotes (') will treat them literally (display exactly what you type).
  - $a=5;
  - echo '$a'; //Will display **$a**

Data types

- PHP is weakly types language

- string $myString = "Hello world";
- integers $myNumber = 21;
- floating-point /double $myNumber = 21.4;
- boolean $gameOver = false;

- How PHP determine the datatype?

PHP concatenation

- PHP concatenation uses a dot " ."

```
<?php
        $a = "Hello";
        $b = "World";
        $c = $a . " " . $b;
        echo $c;
?>
```

Output

Hello World

Double-quoted and Single-quoted strings

- Double quotes can be used within single-quoted strings and vice versa. Both valid:
    - $phrase = "It's time to go";
    - $phrase = 'She said "OK“ ';
- The following are not valid (error due to mismatch of quotes):
    - $phrase = 'It's time to go';
    - $phrase = "She said "OK“ ";
- If you want to use the same quote within a quoted string you must escape it by using a backslash.
    - $phrase = 'It\'s time to go';
    - $phrase = "He said \"OK\“ ";

Question 1

```php
<?php

$variable = 'Saman';
echo 'My name is $variable';

?>
```

Output

My name is $variable

```php
<?php

$variable = 'Saman';
echo "My name is $variable";

?>
```

Output

My name is Saman

Constants

- Syntax:

  define(name, value, case-insensitive)

  - Parameters:
  - *name*: Specifies the name of the constant
  - *value*: Specifies the value of the constant
  - *case-insensitive*: Specifies whether the constant name should be case-insensitive. Default is false

Examples:

```
define("UNI", "University of Westminster");
define("PRICE", 79.99);
echo UNI;
print UNI;
```

Operators - Arithmetic Operators

PHP Arithmetic Operators

Operator	Name	Example	Result
+	Addition	$x + $y	Sum of $x and $y
-	Subtraction	$x - $y	Difference of $x and $y
*	Multiplication	$x * $y	Product of $x and $y
/	Division	$x / $y	Quotient of $x and $y
%	Modulus	$x % $y	Remainder of $x divided by $y

PHP Increment / Decrement Operators

Operator	Name	Description
++$x	Pre-increment	Increments $x by one, then returns $x
$x++	Post-increment	Returns $x, then increments $x by one
--$x	Pre-decrement	Decrements $x by one, then returns $x
$x--	Post-decrement	Returns $x, then decrements $x by one

Assignment Operators

Assignment	Same as...	Description
x = y	x = y	The left operand gets set to the value of the expression on the right
x += y	x = x + y	Addition
x -= y	x = x - y	Subtraction
x *= y	x = x * y	Multiplication
x /= y	x = x / y	Division
x %= y	x = x % y	Modulus

PHP String Operators

Operator	Name	Example	Result
.	Concatenation	$txt1 = "Hello" $txt2 = $txt1 . " world!"	Now $txt2 contains "Hello world!"
.=	Concatenation assignment	$txt1 = "Hello" $txt1 .= " world!"	Now $txt1 contains "Hello world!"

http://www.w3schools.com/php/php_operators.asp

Comparison Operators

PHP Comparison Operators

The PHP comparison operators are used to compare two values (number or string):

Operator	Name	Example	Result
==	Equal	$x == $y	True if $x is equal to $y
===	Identical	$x === $y	True if $x is equal to $y, and they are of the same type
!=	Not equal	$x != $y	True if $x is not equal to $y
<>	Not equal	$x <> $y	True if $x is not equal to $y
!==	Not identical	$x !== $y	True if $x is not equal to $y, or they are not of the same type
>	Greater than	$x > $y	True if $x is greater than $y
<	Less than	$x < $y	True if $x is less than $y
>=	Greater than or equal to	$x >= $y	True if $x is greater than or equal to $y
<=	Less than or equal to	$x <= $y	True if $x is less than or equal to $y

Logical Operators

PHP Logical Operators

Operator	Name	Example	Result
and	And	$x and $y	True if both $x and $y are true
or	Or	$x or $y	True if either $x or $y is true
xor	Xor	$x xor $y	True if either $x or $y is true, but not both
&&	And	$x && $y	True if both $x and $y are true
||	Or	$x || $y	True if either $x or $y is true
!	Not	!$x	True if $x is not true

http://www.w3schools.com/php/php_operators.asp

Array Operators

PHP Array Operators

The PHP array operators are used to compare arrays:

Operator	Name	Example	Result
+	Union	$x + $y	Union of $x and $y (but duplicate keys are not overwritten)
==	Equality	$x == $y	True if $x and $y have the same key/value pairs
===	Identity	$x === $y	True if $x and $y have the same key/value pairs in the same order and of the same types
!=	Inequality	$x != $y	True if $x is not equal to $y
<>	Inequality	$x <> $y	True if $x is not equal to $y
!==	Non-identity	$x !== $y	True if $x is not identical to $y

http://www.w3schools.com/php/php_operators.asp

Control structures

Selection - simple if-else

```
if ($number < 10)
{
        // code to be executed when the condition is true
        echo "$number is less than ten";
}
else
{
        // code to be executed when the condition is true
        echo "$number is not less than ten";
}
```

Selection - if-else ladder

```
<!DOCTYPE html>
<html>
<body>

<?php
$t=date("H");

if ($t<"10") {
   echo "Have a good morning!";
} elseif ($t<"20") {
   echo "Have a good day!";
} else {
   echo "Have a good night!";
}
?>

</body>
</html>
```

Output ?

Selection - switch

```
<!DOCTYPE html>
<html>
<body>
<?php
$favcolor="red";
switch ($favcolor) {
   case "red":
     echo "Your favorite color is red!";
     break;
   case "blue":
     echo "Your favorite color is blue!";
     break;
   case "green":
     echo "Your favorite color is green!";
     break;
   default:
     echo "Your favorite color is neither red, blue,
or green!";
}
?>
  </body>
</html>
```

Output ?

'our favorite color is red!

Question 2

- What is the output ?

```
<html>
<head></head>
<body>

<?php
$x = rand(1,5);  // random integer
echo "x = $x <br/><br/>";
switch ($x)
{
case 1:
  echo "Number 1";
  break;
case 2:
  echo "Number 2";
  break;
case 3:
  echo "Number 3";
  break;
default:
  echo "No number between 1 and 3";
  break;
}
?>

</body>
</html>
```

Iteration- while

```
while ($i <= 10)
{
        echo $i++;

        $i++;
}
```

Iteration- while

- Write a php code to get the following output
  - Use a while loop

```
<!DOCTYPE html>
<html>
<body>
<?php
$x=1;
while($x<=5) {
  echo "The number is: $x <br>";
$x++;
}
?>
</body>
</html>
```

Output ?

The number is: 1
The number is: 2
The number is: 3
The number is: 4
The number is: 5

Iteration- do while

- Write a php code to get the following output
  - Use a do-while loop

```
<!DOCTYPE html>
<html>
<body>
<?php
$x=4;

do {
  echo "The number is: $x <br>";
  $x++;
} while ($x<=5);
?>
</body>
</html>
```

Output ?

The number is: 4
The number is: 5

Iteration - for

```
for ($i = 1; $i <= 10; $i++)
{
        echo $i;
}
```

Iteration - for

- Write a php code to get the following output
  - Use a for loop

```
<!DOCTYPE html>
<html>
<body>
<?php
for ($x=0; $x<=10; $x++) {
   echo "The number is: $x <br>";
}
?>
</body>
</html>
```

Output ?

The number is: 0
The number is: 1
The number is: 2
The number is: 3
The number is: 4
The number is: 5
The number is: 6
The number is: 7
The number is: 8
The number is: 9
The number is: 10

Iteration - foreach

```
<!DOCTYPE html>
<html>
<body>
<?php
$colors = array("red","green","blue","yellow");
foreach ($colors as $value) {
   echo "$value <br>";
}
?>
</body>
</html>
```

Output ?

red
green
blue
yellow

Continue and Break

- **Break** ends execution of the current for, foreach, while, do-while or switch structure.

- **Continue** is used within looping structures to skip the rest of the current loop iteration and continue execution at the condition evaluation and then the beginning of the next iteration.

```
<!DOCTYPE html>
<html>
<body>
<?php
$i = 0;
for ($i = 0;$i <= 5;$i++)
{
if ($i==2)
{
break;
} echo $i;
echo "<br />";
}
echo "End of for loop" ;
?>
</body>
</html>
```

Output ?

```
0
1
End of for loop
```

```
<!DOCTYPE html>
<html>
<body>
<?php
$i = 0;
for ($i = 0;$i <= 5;$i++)
{
if ($i==2)
{
continue;
}
echo $i;
echo "<br />";
}
echo "End of for loop" ;
?>
</body>
</html>
```

Output ?

```
0
1
3
4
5
End of for loop
```

End of for loop

Summary

- Introduction
- Variables and Constants
- Operators
- Control structures