BIT 2nd Year
Semester 3
IT 3505

Web Application Development II

Server Side Web Development (PHP & MySQL) – Part 1

Instructional Objectives

- Install PHP in a windows environment
- Install PHP in Linux environment
- Explain basic features of PHP
- Articulate MVC architecture
- Differentiate available PHP frameworks
- Explain MVC
- Use web services with PHP
- Develop a web application with PHP

Sub Topics

1.1 Introduction to PHP (Ref 01 Pg:271-278)
1.2. Configuring the environment (Ref 01 Pg : 76 - 85)
1.3. PHP Syntax and Semantics
    1.3.1. Variables (Ref 01 Pg:281-287)
    1.3.2. Constants (Ref 01 pg:287 - 296)
    1.3.3. Conditional Statements (Ref 01 pg:320-335)
    1.3.4. Loops (Ref 01 Pg:335-346)
    1.3.5. Functions (Ref 01 Pg: 346-357)
1.4. Arrays and data processing with arrays (Ref 01 Pg: 296-307)
1.5. Form processing with PHP (Ref 02)
1.6. Session control and Cookies (Ref 01 Pg:437-446)
1.7. File system management (Ref 01 Pg: 366-389)
1.8. Email sending using PHP (Ref 03)
1.9. Object Orientation with PHP (Ref 01 pg :397-423)
1.10. Working with MySQL database (Ref 01 PG:515-528)
1.11. Introduction to PHP frameworks (Ref 5)
1.12. Fundamentals of MVC (Ref 6)
1.13. How to call web service using PHP (Ref 01 pg:541-553)

Introduction to PHP

- PHP is an acronym for “PHP Hypertext Preprocessor”.
  - Other Names : Personal Home Page, Professional Home Page
- Is a widely-used open source general-purpose scripting language
  - *PHP scripts are executed on the server*
  - *Predominantly used for generating HTML pages*
- *In this course the main emphasis is to explore how PHP could be used for Web application development.*

Why PHP ?

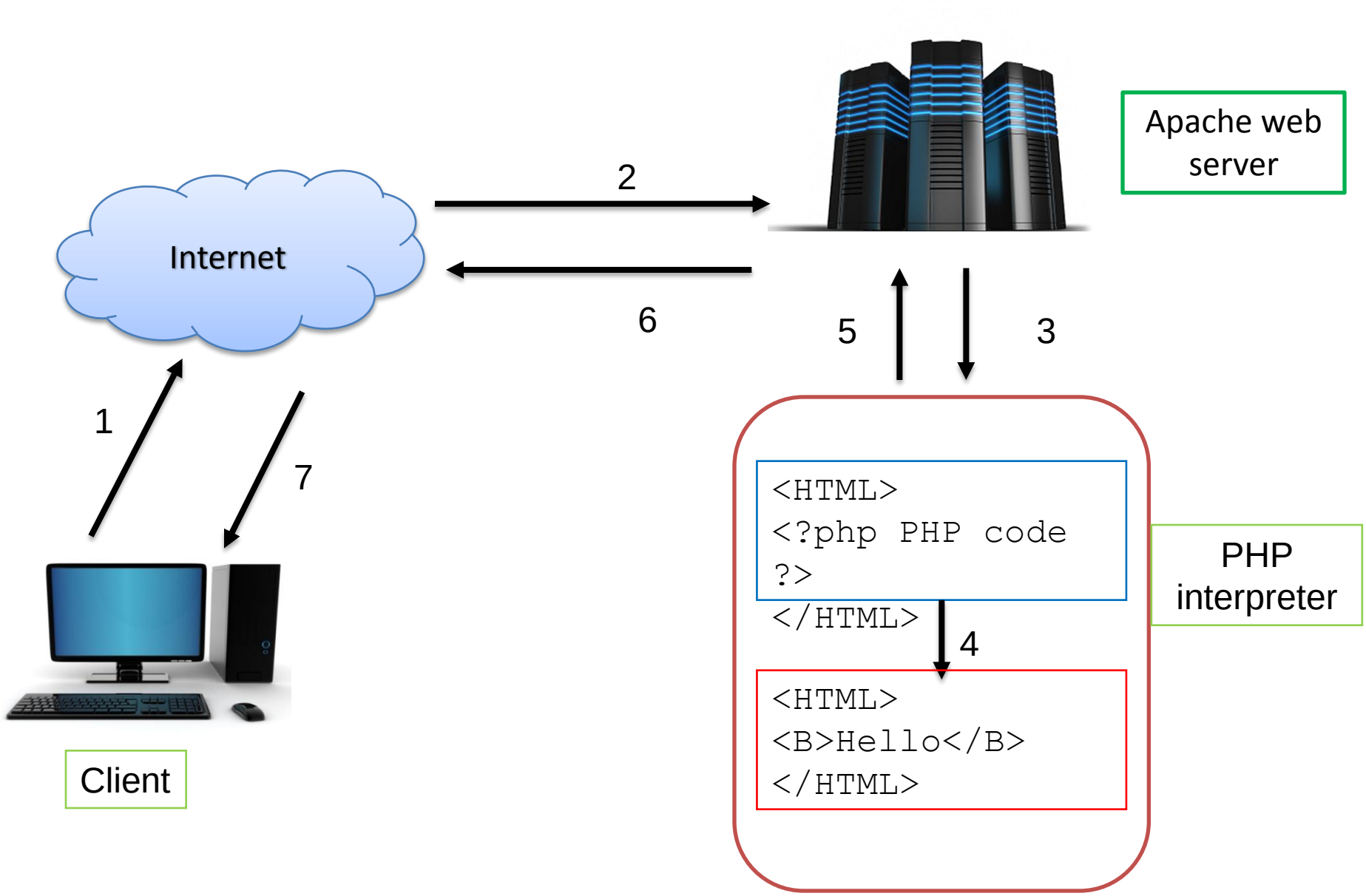
- Can easily be embedded into HTML pages
- Interoperable
  - Runs on various platforms.
- Supports a wide range of databases
- PHP is free.
- Compatible with almost all servers (Apache, IIS, etc.)
- Features
  - Similar to almost all compiled languages such as C#, Java , C++

Embedding PHP in HTML Pages - Example

```
<html>
    <head>
        <title>PHP Example</title>
    </head>
    <body>

        <?php
            echo “Hellow World!";
        ?>

    </body>
</html>
```


Internet
2
6
Apache web server
5
3
1
7
<HTML>
<?php PHP code
?>
</HTML>
4
<HTML>
<B>Hello</B>
</HTML>
PHP interpreter
Client

INSTALLING THE REQUIRED SOFTWARE

What you need

- Need to install
  - A Webserver (Apache, IIS, etc.)
  - PHP http://php.net/manual/en/install.php
  - MySQL

  - all required software can be easily installed by installing WAMP server.
  - WAMP is available at http://sourceforge.net/projects/wampserver/

FUNDAMENTALS

Coding PHP Scripts

- You can use any text editor to create PHP scripts.
- PHP scripts should be saved in files with the .php extension.
- PHP scripts should be coded within <?php and ?> tags.
- In PHP scripting statement terminator is ";" symbol. Thus each statement in a script should be terminated from the symbol ;
- You may have multiple statements in a single physical line, but having multiple statements in a single physical line is not a good practice.
- PHP is a free format language. Thus you can have spaces in between statement components as you wish.

Printing text on the screen

The **echo** command can be used to print text on the screen. The syntax of the echo command is given below

echo “some text”;

Executing a script

Type the following script in a file by using a text editor and save it with the name “example.php”

```
<?php
echo “My first php script”;
?>
```

Execute the script at the DOS command line by typing the command

php example.php

Embedding a PHP Script in a HTML Page

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

<h1>My first PHP page</h1>

<?php
      echo "Hello World!";
?>

</body>
</html>
```

Printing text on the screen

- When the server encounters the PHP tags it switches from the HTML to PHP mode.

- There are four different ways to embed the PHP code
  1. <?php echo("Some PHP code"); ?>
  2. <? echo("Some PHP code"); ?>
  3. <SCRIPT Language='php'> echo("Some PHP code"); </SCRIPT>
  4. <% echo("Some PHP code"); %>

Data Types

- A Data type can be described as a collection of values and a set of operations over these values.
- The different primitive data types provided by PHP can be classified as
  - Scalar Types
  - Compound Types
  - Special Types

PHP Scalar Data Types

- PHP supports the following primitive scalar data types.
  - Integer
  - Floating point (or Double)
  - String
  - Boolean

PHP Scalar Data Types

- PHP supports the following primitive scalar data types.
  - Integer
  - Floating point (or Double)
  - String
  - Boolean

Integer Data Type

- Any number in the set {….,-2,-1,0,1,2,……} is considered as an integer. The maximum and the minimum integer value that can be used in a program are platform dependent.
- The number of bytes allocated to store an integer and the maximum integer value that can be used in your platform can be determined by using the constants PHP_INIT_SIZE and PHP_INT_MAX
- If PHP encounters a number larger than PHP_INT_MAX, the number will be interpreted as a floating point number.

Example :

```
<?php
 echo PHP_INT_SIZE,"\n",PHP_INT_MAX;
?>
```

Integer Data Type .....

- An integer literal can be specified in decimal, octal, hexadecimal or binary.

BNF Definition of integers in PHP

<Integer> : [+-]?<decimal> | [+-]?<hexadecimal> | [+-]?<octal> | [+-]?<binary>

<decimal> : [1-9][0-9]* | 0

<hexadecimal> : 0[xX][0-9a-fA-F]+

<octal> : 0[0-7]+

<Binary> : 0b[01]+

Integer Data Type .....

Examples:

- 1234 // a positive integer in decimal form
- -123 // a negative integer in decimal form
- 0123 // integer 83 in octal form
- 0x2b1 // integer 689 in hexadecimal form
- 0b01101 // integer 13 in binary form

Floating Point Data Type

- The numbers with decimal points are considered as floating point (double or real) numbers.
- The floating point numbers are represented internally by using IEEE floating point representation. Thus the representations are not exact. Therefor floating point numbers should not compare for equality.
- Floating point literals can be coded in a number of different ways.

Example :

-1.23

1.2e3

34.45E-12

Floating Point Data Type …

BNF Definition of floating point numbers in PHP

<Floating point numbers> : <LNUM> | <DNUM>| <EXPONENT_DNUM>

<LNUM> : [0-9]+

<DNUM> : [0-9]*. <LNUM> | <LNUM> . [0-9]*

<EXPONENT_DNUM> : [+-]?(<LNUM>|<DNUM>) [eE][+-]? <LNUM>

Arithmetic Operators

- The following operators can be applied on both integers and floating point numbers.

Operator	Result
Unary -	Negation of the number
Binary -	Subtraction
+	Addition
*	Multiplication
/	Division
%	Remainder

Arithmetic Operators

Example :

```
<?php
 echo -3,"\t",5 - 3,"\t",5.2*3.4,
        "\t",10/2,"\t",
        10/4,"\t",10%3,"\n";
?>
```

Automatic type conversion in addition

- The following type conversion happens automatically when addition(+) operator is applied on its operands.
- If either operand is a float, then both operands are evaluated as floats, and the result will be a float.
- Otherwise, the operands will be interpreted as integers, and the result will also be an integer.

String Data Type

- A string comprises of a series of characters.
- The series of characters in a string literal are normally represented either in single quotes or in double quotes.

Example :

```
<?php
 echo "This is a string literal","\n";
 echo 'Another string literal';
?>
```

String Data Type

- A string comprises of a series of characters.
- The series of characters in a string literal are normally represented either in single quotes or in double quotes.

Example :

```
<?php
 echo "This is a string literal \n";
 echo 'Another string literal';
?>
```

Special character sequences

- Certain character sequences are given special meanings in PHP.

Character sequence	Special Meaning
\n	Linefeed
\t	Horizontal tab
\$	Dollar sign
\”	Double quote

Special character sequences

- The special character sequences retain their special meanings only when used inside double quotes.

Example :

```
<?php
 echo 'How the character sequence \n works';
 echo "\n As a line feed";
?>
```

Boolean Literals

- Boolean values are represented by using the two Literals – **TRUE** or **FALSE**. Both are case-insensitive.
- Values are automatically converted as appropriate if an operator, function or control structure requires a Boolean argument.

Conversion to Boolean

- When converting to Boolean the following values are considered as **FALSE**.
  - The Boolean literal FALSE.
  - The integer literal 0
  - The floating point literal 0.0.
  - The empty string “” and the string “0”.
  - An array with zero elements
  - The special type NULL.
- A Boolean TRUE value is converted to the string “1” and FALSE is converted to “”(empty string).

Activity

What is the output of the following script?

```
<?php
  echo "True -",true,"\n";
  echo "False -",false,"\n";
  echo "1 > 0 - ", 1 > 0, "\n";
  echo "2 > 5 - ", 2 > 5, "\n";
  echo "0 && true -",0 && true,"\n";
  echo "'' && true -','' && true,"\n";
  echo "TRUE > FALSE - ", TRUE > FALSE, "\n";
  echo "TRUE < FALSE - ", TRUE < FALSE, "\n";
?>
```

Activity

What is the output of the following script?

```
<?php
  echo true,"\n";                                    1
  echo false,"\n";              empty string
  echo 1 > 0, "\n";             1
  echo  2 > 5, "\n";            empty string
  echo 0 && true,"\n";          empty string
  echo "" && true,"\n";         empty string
  echo TRUE > FALSE, "\n";              1
  echo TRUE < FALSE, "\n";              empty string
?>
```

String conversion to numbers

- String are converted to numbers when strings are used with mathematical operations. How this conversion happens is given below.
- The value is given by the initial portion of the string.
- The leading white spaces are ignored.
- If the string starts with valid numeric data, this will be the value used. Otherwise, the value will be 0 (zero).
- Valid numeric data is an optional sign, followed by one or more digits (optionally containing a decimal point), followed by an optional exponent. The exponent is an 'e' or 'E' followed by one or more digits.

When a comparison operation involves a string or a numerical string, then each string is converted to a number and a numerical comparison is performed.

Activity

What is the output of the following script?

```
<?php
  echo "10" > 1 ,"\n";
  echo "10 items" > 2 ,"\n";
  echo  "items" > 2 , "\n";
  echo True == 1, "\n";
  echo false == 0, "\n";
  echo TRUE == FALSE, "\n";
?>
```

Activity

What is the output of the following script?

```
<?php
  echo "10" > 1 ,"\n"; 1
  echo "10 items" > 2 ,"\n";   1
  echo "items" > 2 , "\n";      empty string
  echo  True == 1, "\n";        1
  echo  false == 0, "\n";       1
  echo TRUE == FALSE, "\n";  empty string
?>
```

PHP echo and print Statements

- There are some differences between echo and print:
  - echo - can output one or more strings
  - print - can only output one string, and returns always 1
  - **Tip:** echo is marginally faster compared to print as echo does not return any value.

PHP Constants

- .Values that never changes
- Constants are defined in PHP by using the define() function.
  - For e.g.

  define("NCST", "National Centre for Software Technology")
- defined() function says whether the constant exists or not.

PHP echo and print Statements

- There are some differences between echo and print:
  - **echo** - can output one or more strings
  - **print** - can only output one string, and returns always 1
  - **Tip:** echo is marginally faster compared to print as echo does not return any value.

PHP Constants

- Values that never changes.

- Constants are defined in PHP by using the define() function.
  - For e.g.
    define("NCST", "National Centre for Software Technology")

- defined() function says whether the constant exists or not.

PHP Variables

- Rules for PHP variables:
  - A variable starts with the $ sign, followed by the name of the variable
  - A variable name must start with a letter or the underscore character
  - A variable name cannot start with a number
  - A variable name can only contain alpha-numeric characters and underscores (A-z, 0-9, and _)
  - Variable names are case sensitive ($y and $Y are two different variables)

PHP Variables

- The variables in PHP are declared by appending the $ sign to the variable name.
  - For e.g
    $company = “NCST”;
    $sum = 10.0;
- Variable data type is changed by the value that is assigned to the variable.
- Type casting allows to change the data type explicitly.
- Rich set of functions for working with variable.
  - For e.g.
    gettype, settype, isset, unset, is_int, intval etc etc

PHP Variables Scope

- PHP has three different variable scopes:
  - Local
    - A variable declared **within** a function
  - Global
    - A variable declared **outside** a function
  - Static
    - When a function is executed, all of its variables are deleted.
    - If you want a local variable NOT to be deleted, use the **static** keyword

PHP Operators

- All the operators such as arithmetic, assignment, Comparison, and logical operators are similar to the operators in Java or C++.

- In PHP the string concatenation operator is denoted by ‘’.
  - For e.g.
    $name = “My name is”.$myname;

Conditional Statements

- Need to take different actions for different decisions.
- In PHP we have the following conditional statements:
  - **if statement** - executes some code only if a specified condition is true
  - **if...else statement** - executes some code if a condition is true and another code if the condition is false
  - **if...elseif....else statement** - selects one of several blocks of code to be executed
  - **switch statement** - selects one of many blocks of code to be executed

Conditional Statements Syntax

- IF statement

```
if (<condition>) {
        //php code goes
          here
        }
```

IF Else statement

```
if (<condition>) {
        //php code goes here
        }
else {
        //php code goes here
        }
```

```
<?php

        $color="Red";

        if ($color =="Red") {
                echo "Please STOP";
        } else {
                echo "You can GO";
        }

?>
```

Conditional Statements Syntax

- **if...elseif....else statement**

```
if (condition) {
  //php code goes here
} elseif (condition) {
 //php code goes here
} else {
  //php code goes here
}
```

```
<?php

      $color="Red";

      if ($color =="Red") {
          echo "Please Stop" ;
      } elseif ($color =="Yellow") {
          echo "Get ready" ;
      } else {
          echo "You can GO" ;
      }

?>
```

Conditional Statements Syntax

```
•<?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!";
 }
 ?>
```

Switch statement

- **select one of many blocks of code to be executed.**

Loops

- When you need the same block of code to be executed over and over again .
- In PHP, we have the following looping statements:
  - **while** - loops through as long as the given condition is true
  - **do...while** - loops through the code once, and then repeats the loop as long as the given condition is true
  - **for** - loops through a the code a given number of times
  - **foreach** - loops through the code for each element in a collection

Loops- While Loop

```
while (condition is true) {
  //Do this;
}
```

```
<?php
$i=1;

while($i<=5) {
  echo “Number: $i <br>";
  $i++;
}
?>
```

Number: 1
Number : 2
Number : 3
Number : 4
Number : 5

Loops-Do while Loop

```
do {
  //Php code
} while (condition is
true);
```

```
<?php
$i=1;

do {
  echo “Number: $i <br>";
  $i++;
} while ($i<=5);
?>
```

```
Number: 1
Number : 2
Number : 3
Number : 4
Number : 5
```

Loops-For Loop

```
for (initialize counter; check; increment counter) {
  //Do this;
}
```

```
<?php
for ($i=0; $i<=10; $i++) {
  echo "The number is: $i
<br>";
}
?>
```

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

Loops- Foreach Loop

This works only on collections such as arrays ,lists

```
foreach ($array as $value)
{
 //Do this
}
```

```
<?php
$person =
array("Nimal","Kamal","Sunil"
,"Amal");

foreach $person as $value) {
  echo "$value <br>";
}
?>
```

Nimal
Kamal
Sunil
Amal

FUNCTIONS

Functions

- Will make your code easy to read and reuse.
- Large projects would be unmanageable without functions because the problem of repetitive code would bog down the development process
- A function accepts values, processes them, and then performs an action (printing to the browser, for example), returns a new value, or both.
- PHP has 2 types of functions
  - Language defined functions
  - User defined functions

Language defined functions

- PHP has hundreds of built-in functions . For example strlen() returns the length of a string, in characters .

```
<?php
echo strlen("Hello
world!");
?>
```

→ **12**

- You can find a compete set of language defined functions here

User defined functions

- You can define your own functions in PHP using the function statement

```
function functionName() {
  //Php code
}
```

- A function name can start with a letter or underscore but not a number

variables names are case sensitive in PHP, function names are not

User defined functions-Function with no parameters

```
Function() {
    statements;
}
```

```
function WriteThis() {
        echo "I'm an undergraduate";
}
```

Now when you call this function
WirteThis() ➔ will print `I'm an undergraduate`

User defined functions

- Function with parameters
  - Parameter types and return types are not written .
  - A function with no return statements implicitly returns NULL

```
function name(parameterName, ...,
parameterName) {
    statements;
}
```

```
function multiply($a, $b, $c) {
        return $a*$b*$c;
}
```

Default Parameter Values

```
<?php
function setMarks($minMark=50) {
  echo "The Mark is : $minMark=50 <br>";
}

setMarks(95);
setMarks(); // will use the default value of 50
setMarks(78);
setMarks(80);
?>
```

if no value is passed, the default will be used