PHP

Module 3

PHP: Introduction, Server side programming, Role of Web Server software,

Including PHP Script in HTML: head, body, external. Comments, Data types, variables and scope, echo and print.

Operators: Arithmetic, Assignment, Relational, Logical.

Conditional Statements, Loops, break and continue.

User Defined Functions.

Web server

- A web server is server software, or hardware dedicated to running this software, that can satisfy client requests on the World Wide Web.

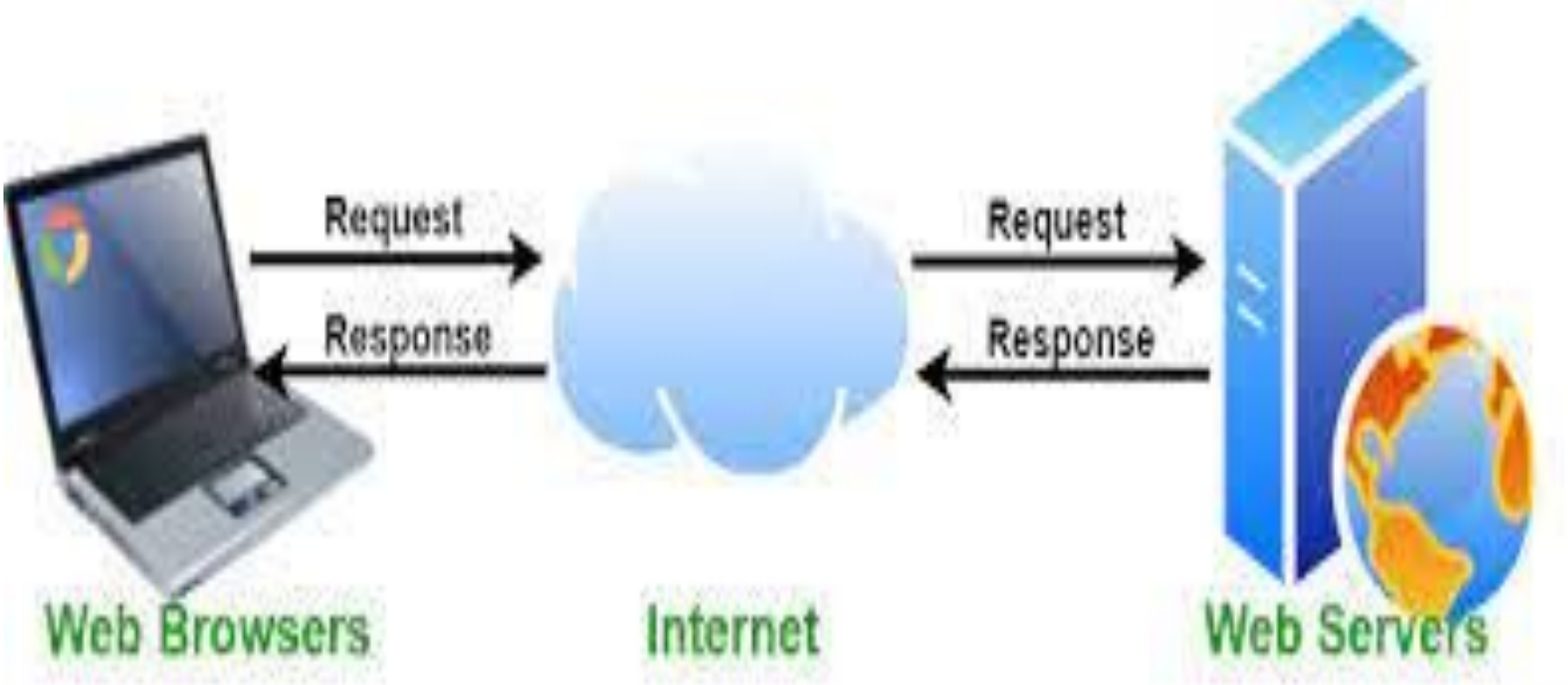
- Basically web server is used to host the web sites. A web server can contain one or more websites.

- The primary function of a web server is to store, process and deliver web pages to clients.

- A web browser initiates communication by making a request for a specific resource using HTTP and the server responds with the content of that resource or an error message if unable to do so.

- The communication between client and server takes place using the Hypertext Transfer Protocol (HTTP).

- Pages delivered are most frequently HTML documents, which may include images, style sheets and scripts in addition to the text content.


Request
Response
Request
Response
Web Browsers
Internet
Web Servers

Server side scripting

- In server side scripting technology, a user's request is fulfilled by running a script directly on the web server to generate dynamic web pages.

- The primary advantage to server side scripting is the ability to customize the responses based on the user's requirements.

- From a security point of view, server side script are never visible to the browser as these scripts are executed on the server.

Server side technologies

Three main server side technologies are :

- CGI
- Web server API programs
- Server side scripting languages.

Common Gateway Interface (CGI)

- CGI is an interface specification for web servers to execute programs like console applications running on a server that generates web pages dynamically. Such programs are known as CGI scripts or simply as CGIs.

- When a user fills out a form on a web page and sends it in, the web server passes the form information to a small CGI application program that processes the data and may send back a confirmation message. This method for passing data back and forth between the server and the application is called CGI.

Web server API programs

- The server side program is written to be a component of a Web server.

- There are many types of server modules and they are typically associated with a particular server.

  Eg: On microsoft - IIS (Internet Information Server)

  Apache server has apache modules

Server side Scripting

- Server Side Includes (SSI)
- Active Server Pages (ASP)
- ASP.NET
- Adobe ColdFusion
- Javaserver Pages (JSP)
- PHP
- Python
- Ruby on Rails

Introduction to PHP

- PHP is a server side scripting language that is especially suited to web development.

- It was originally created by Rasmus Lerdorf in 1994.

- PHP originally stood for Personal Home Page, but it now stands for the recursive acronym, **PHP: Hypertext Preprocessor.**

Basic PHP Syntax

- A PHP script can be placed anywhere in the document.
- A PHP script starts with **<?php** and ends with **?>**

Eg :

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

- The default file extension for PHP files is "**.php**"
- PHP statements end with a semicolon (**;**)

- There are 3 ways to Include PHP Script in HTML :
    - In head section
    - In body section
    - External file

In head section

Eg :

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

In body section

Eg :

```
<html>
<body>
<h1>My first PHP page</h1>
<?php
echo "Hello World";
?>
</body>
</html>
```

External file

- It is possible to insert the content of one PHP file into another PHP file with the include or require statement.

- The include and require statements are identical, except upon failure:
  - require will produce a fatal error and stop the script.
  - include will only produce a warning and the script will continue.

Syntax

```
include  'filename';
        or
require  'filename';
```

Eg:

```
<?php
include 'menu.php';
?>
```

Comments in PHP

A comment in PHP code is the lines that is not executed as a part of the program.

Comments can be used to:

- Let others understand your code
- Remind yourself of what you did.
- To improve readability of your code.

- Two types of Comments :
  - Single-line comments
  - Multi-line comments

Single-line comments

- Single-line comment starts with either two slashes(//) or a hash symbol(#)
- Eg :

```
// This is a single-line comment

# This is also a single-line comment
```

Multi-line comments

- Multi-line comments starts with /* an ends with */

  Eg:

```
/*
This is a multi-line comment block
that spans over multiple lines.
*/
```

Variables

- Variables are containers for storing information.
- In PHP, a variable starts with the $ sign, followed by the name of the variable.
- Eg:

```
<?php

$txt = "Hello world!";

$x = 5;

$y = 10.5;

?>
```

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 alphanumeric characters and underscores (A-z, 0-9, and _)
- Variable names are case-sensitive.

- PHP is a Loosely Typed Language.
- PHP automatically associates a data type to the variable, depending on its value.
- PHP variable names are case-sensitive!

Type Juggling

- PHP automatically converts values from one type to another whenever required. This is called implicit casting or implicit conversion or type juggling.
- Eg :

  $sum = 12+ "20"; will assign the value 32

  $num=2+4.5; will assign the value 6.5

Type Casting

- The process of explicitly converting values from one data type to another is known as type casting or explicit conversion.

  Eg :

```
$x = false;

$y = (int) $x;              // Will output 0
```

Variable scope

- In PHP, variables can be declared anywhere in the script.

- The scope of a variable is the part of the script where the variable can be referenced/used.
- PHP has three different variable scopes:
    - Local variables
    - Global variables
    - Static variables

Local variables

- A variable declared within a function has a local scope and can only be accessed within that function.

Global variables

- A variable declared outside a function has a global scope and can be accessed from any part of the program.

- The global keyword is used to access a global variable from within a function. To do this, use the global keyword before the variables (inside the function).

Static variables

- Normally, when a function is completed/executed, all of its variables are deleted. However, sometimes we want a local variable NOT to be deleted for further job.

- To do this, use the static keyword when declaring the variable for the first time.

PHP Data Types

Variables can store data of different types.

PHP support 8 primitive data types.

- Four scalar types :
  - Integer
  - Float
  - String
  - Boolean

- Two compound types :
  - Array
  - Object

- Two special types :
  - Resource
  - NULL

Integer

Integers are whole numbers, without a decimal point.

Rules for integers :

- An integer must have at least one digit
- An integer must not have a decimal point
- An integer can be either positive or negative
- Integers can be specified in: decimal (base 10), hexadecimal (base 16), octal (base 8), or binary (base 2) notation

Eg: **$x = 10;**

Float or double

- A float (floating point number) is a number with a decimal point or a number in exponential form.
- Eg : **$y = 4.5;**

Boolean

- A Boolean represents two possible states: TRUE or FALSE.
- Eg : **$z = true;**

String

- A string is a sequence of characters..
- A string can hold letters, numbers, and special characters.
- A string is enclosed in single or double quotes.

Eg : **$x = “ Hello World ” ;**

Array

- An array stores multiple values in one single variable.
- Eg : $cars = array("Volvo","BMW","Toyota");

Object

- An object is a data type which stores data and information on how to process that data.
- An object is a specific instance of class.
- Every object has properties and methods.

```
<?php
class greeting {
      $str = “Hello”;
      function show() {
            return $this->str;
  }  }
$msg = new greeting();
echo $msg;
?>
```

NULL

- Null is a special data type which can have only one value: NULL.
- A variable of data type NULL has no value assigned to it.

- If a variable is created without a value, it is automatically assigned a value of NULL.
- Variables can also be emptied by setting the value to NULL.

- Eg : **$x = null;**

Resource

- A resource is a special variable, holding a reference to an external resource.
- A common example of using the resource data type is a database call.

- Eg : **$file = fopen(“abc.txt”, “r”);**

  //var_dump($file);

var_dump() function returns the data type and value.

echo and print Statements

There are two basic ways to get output in PHP :

- echo
- print

Both are used to output data to the screen.

1.

PHP echo Statement

- ‘echo’ can be used to output strings or variables.
- No parenthesis required after the keyword ‘echo’.

- The end of echo statement is identified by the semi-colon.
- It has no return value.

- It can take multiple parameters .
- echo is faster than print.

Eg :

```
echo " Hello ";

echo "Multiple ","argument ";


$num = 10;

echo $num;
```

PHP print Statement

- **print** statement is similar to the echo statement, It is also used to print strings and variables.

- print statement can have only one argument at a time and thus can print a single string.

- print has a return value of 1 so it can be used in expressions.

Eg:

```
print " Hello ";


$num = 10;

print $num;
```

	"echo"	"print"
type	This is a type of output string.	This is a type of output string.
parenthesis	It can or can not be written with parenthesis. ~~Not written with parenthesis~~.	It can or can not be written with parenthesis.
strings	It can output one or more strings.	It can output only one string at a time, and returns 1.
Functionality	echo is faster than print.	print is slower than echo.
argrument	echo($arg1[, $arg2....])	print($arg)

PHP Operators

Operators are used to perform operations on variables and values.

PHP has the following types of operators

- Arithmetic operators
- Increment/Decrement operators
- Assignment operators
- Comparison operators
- Logical operators
- String operators

Arithmetic operators

Operator	Name	Example
+	Addition	$x + $y
-	Subtraction	$x - $y
*	Multiplication	$x * $y
/	Division	$x / $y
%	Modulus	$x % $y
**	Exponentiation	$x ** $y

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

Comparison operators

Operator	Name	Example
>	Greater than	$x > $y
<	Less than	$x < $y
> =	Greater than or equal to	$x >= $y
< =	Less than or equal to	$x <= $y
= =	Equal	$x == $y

! =	Not equal	$x != $y
< >	Not equal	$x <> $y
= = =	Identical	$x === $y
! = =	Not identical	$x !== $y

Logical operators

Operator	Name	Example	Result
&&	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 false

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

String operators

Operator	Name	Example	Result
.	Concatenation	$txt1 . $txt2	Concatenation of $txt1 and $txt2
.=	Concatenation Assignment	$txt1 .= $txt2	Appends $txt2 to $txt1

Conditional Statements

- Conditional statements are used to perform different actions based on different conditions.

- PHP supports the following forms of conditional statements.
  - if statement
  - if...else statement
  - if...else if... statement.
  - switch statement.

The if Statement

- Use the if statement to specify a block of JavaScript code to be executed if a condition is true.

Syntax :

```
if (condition)
    {
        //  block of code to be executed if the condition is true
    }
```

Eg:

```
<?php
$n = 70;
if ( $n < 100)
{
    echo “value is less than 100”;
}
?>
```

The if...else Statement

- It execute one block of code if the specified condition is true and another block of code if it is false.

Syntax:

```
if (condition)
  {
    //  block of code to be executed if the condition is true
  }
else
  {
    //  block of code to be executed if the condition is false
  }
```

Eg:

```
<?php
$n = 450;
if ( $n < 100)
{    echo “value is less than 100”;
}
else
{    echo “value is greater than 100”;
}
?>
```

The else if Statement

- Use the else if statement to specify a new condition if the first condition is false.

Syntax :

```
if (condition1) {
    // block of code to be executed if condition1 is true
} else if (condition2) {
    // block of code to be executed if the condition1 is false and condition2 is true
} else {
  // block of code to be executed if the condition1 is false and condition2 is false }
```

Eg:

```
<?php
$day = 2;
if ( $day == 1)
{    echo " Monday "; }
else if ( $day == 2)
{    echo " Tuesday "; }
else
{    echo " day is not Monday or Tuesday "; }
?>
```

switch Statement

- The switch statement is used to perform different actions based on different conditions.

- This is how it works:
  - The switch expression is evaluated once.
  - The value of the expression is compared with the values of each case.
  - If there is a match, the associated block of code is executed.
  - If there is no match, the default code block is executed.

Syntax :

```
switch(expression)
{
    case value1:
        // code block
        break;
    case value2:
        // code block
        break;
    ...
    default:
        // code block
}
```

Eg :

```
$day = 3;
switch($day)
{
    case 1:
        print "Monday" ;  break;
    case 2:
        print "Tuesday" ;  break;
    case 3:
        print "Wednesday" ;  break;
    default:
        print " Invalid " ;
}
```

Loops

- Loops are used to execute the same block of code again and again, until a certain condition is met.
- Each passage through the loop is known as an iteration.

- PHP supports 4 types of loops :
  - while
  - do..while
  - for
  - foreach

While loop

- The while statement will loops through a block of code until the condition in the while statement evaluate to true.

- Syntax :

```
while (condition)
    {
        // code block to be executed
    }
```

Eg :

```
<?php
$i = 1;
while($i <=10)
{
    echo $i;
    $i++;
}
?>
```

The do-while loop

- This loop will execute the code block once, before checking if the condition is true, then it will repeat the loop as long as the condition is true.

- Syntax

```
do
 {
 // code block to be executed
 }
 while (condition);
```

Eg :

```
<?php
$i = 1;
do
{
    echo $i;
    $i++;
}while($i <=10);
?>
```

The for loop

- The for loop repeats a block of code until a certain condition is met.

- Syntax:

```
for (initialization; condition; updation)
{
  // code block to be executed
}
```

Eg :

```
<?php

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

?>
```

The foreach loop

- The foreach loop is used to iterate over arrays.

- Syntax:

```
foreach ($array as $value)
{
  // code block to be executed
}
```

Eg :

```
<?php
$colors = array(“Red” , ”Green” , ”Blue”);
foreach($colors as $value)
   {
      echo $value;
   }
?>
```

Break Statement

- The break command exits the innermost loop construct that contains it.

Eg:

```
for($x=1; $x<10; $x++)
{
    if($x == 4)
        break;
    print($x);
}
```

Continue Statement

- The continue command skips to the end of the current iteration of the innermost loop that contains it.

Eg:

```
for($x=1; $x<10; $x++)
{
    if($x == 3)
        continue;
    print($x);
}
```

User Defined Functions in PHP

- A function is a self contained block of code that performs a specific task.
- A user-defined function declaration starts with the word function.

Syntax

```
function functionName()
{
  code to be executed;
}
```

Eg:

```php
<?php
function writeMsg()
{
  echo "Hello world";
}

writeMsg(); // call the function
?>
```

Functions with parameters

- Data can be passed to functions through parameters.

- Parameters are specified after the function name, inside the parenthesis.

- Parameters are separated with comma.

Eg:

```php
<?php
function sum($num1, $num2)
{
  $sum1 = $num1 + $num2;
  echo "The result  = $sum1";
}
sum(40,30); // call the function
?>
```

Optional parameters and default values

- Functions can be called without arguments. if so, it will take the default value as argument.

- To provide the default value, insert the parameter name, followed by an equal sign, followed by a default value.

Eg:

```php
<?php
function sum($num1, $num2 = 30)
{
  $sum1 = $num1 + $num2;
  echo "The result  = $sum1";
}
sum(40); // call the function
?>
```

Returning values from a function

- A function can return a value back to the script that called the function using the return statement.

- The return value can be of any types.

- A function cannot return multiple values .

Eg:

```php
<?php
function sum($num1, $num2)
{
  $sum1 = $num1 + $num2;
  return $sum1;
}
echo "The result = " . sum(40,50);
?>
```

Passing Arguments by Value & by Reference

- **Pass by value** means, make a copy of the actual parameters for passing them onto corresponding formal parameters in the function.

- By default, function arguments are passed by value so that if the value of the argument within the function is changed, it does not get affected outside of the function.

- In **pass by reference**, a copy of the address of the actual parameter is passed to formal parameters in the function.

- In PHP, passing an argument by reference is done by prepending an ampersand (&) to the argument name in the function definition.

Eg:

```php
<?php

function display(&$x)          //pass by reference

{

  $x++;

}

$x = 10;

display($x);

echo $x;

?>
```

The End

Thank You

Teacher : Jishna K

College of Applied Science, Thamarassery.