

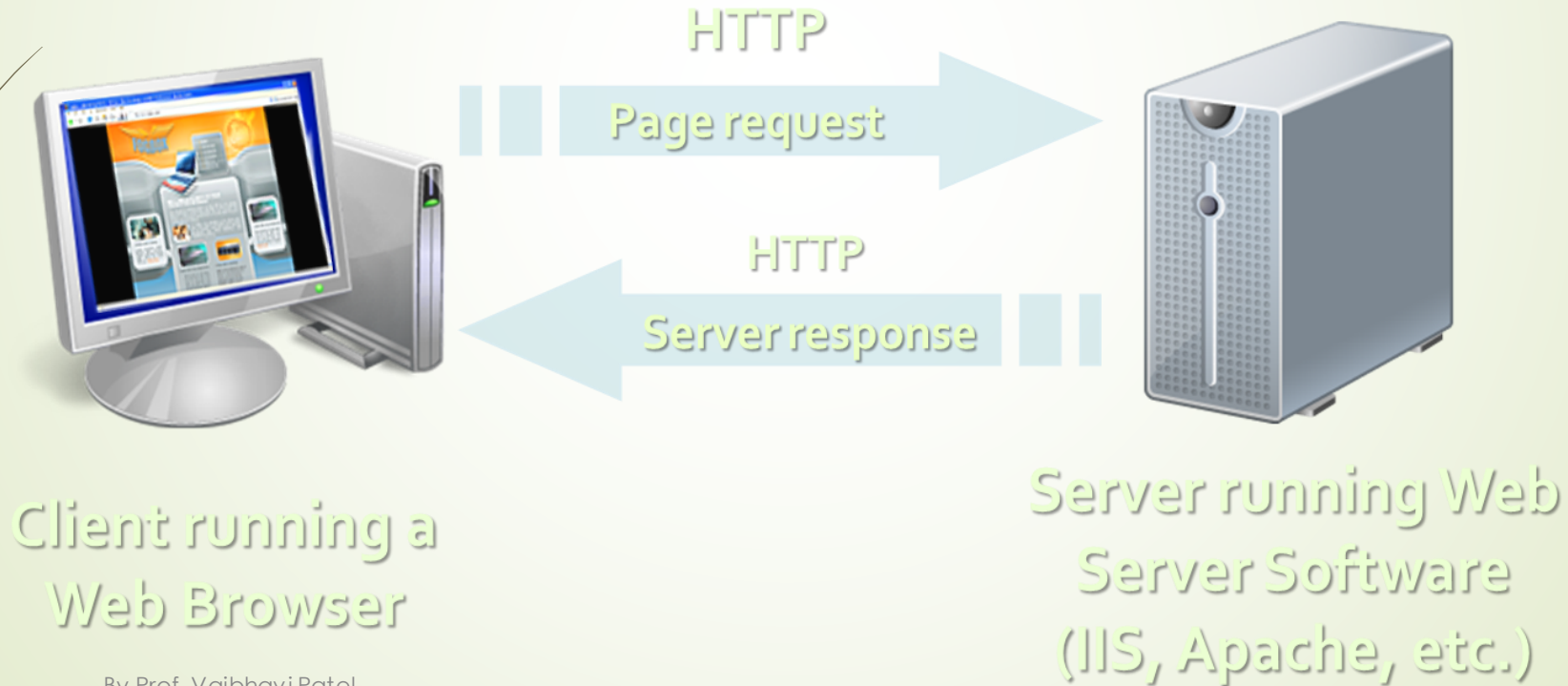


PHP: hypertext Preprocessor

By Prof. Vaibhavi Patel

How Web works?

- WWW use classical client / server architecture
- HTTP is text-based request-response protocol





PHP- Introduction

- PHP stands hypertext Preprocessor
- PHP is open source general-purpose scripting language that is especially suited for web development and can be embedded into HTML
- PHP scripts are executed on the server .
- PHP supports many databases (MySQL, Informix, Oracle, Sybase, Solid, PostgreSQL, Generic ODBC, etc.)
- PHP runs on different platforms (Windows, Linux, Unix, etc.).
- PHP is compatible with almost all servers used today (Apache, IIS, etc.)
- PHP is easy to learn and runs efficiently on the server side

PHP Syntax

- A PHP scripting Block Always Start With `<?php` and End with `?>`.


`<?php`

`.....`

`?>`

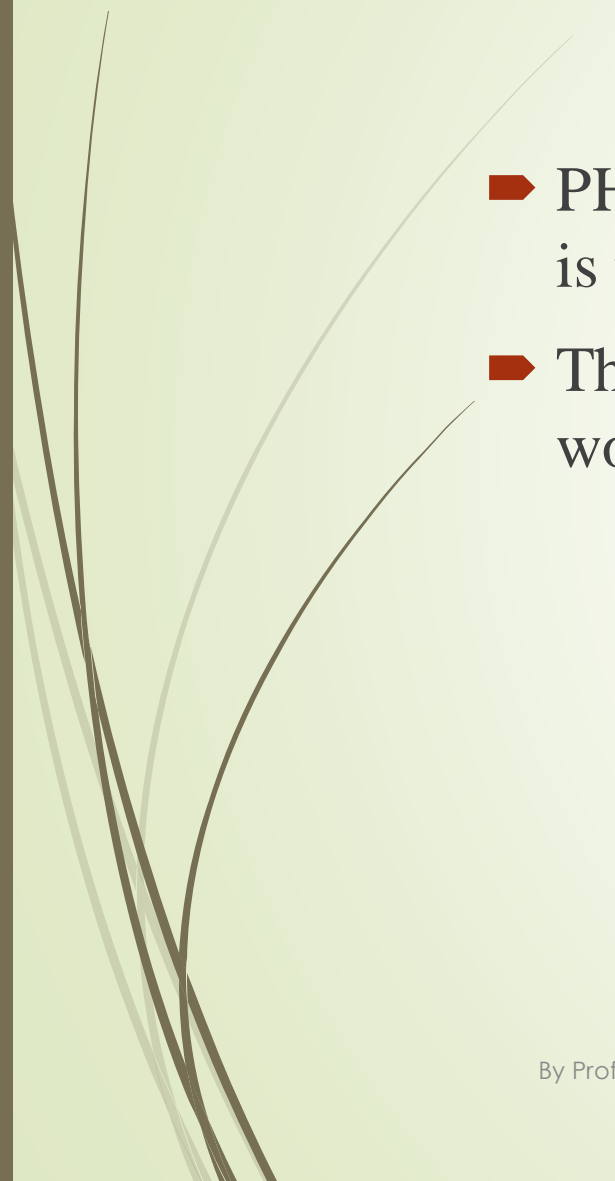
- Php scripting block paced any where in the document.
- Php short-open tags look like this:

`<? ?>`



```
<html>
  <body>
    <?php
      echo "hello";
    ?>
  </body>
</html>
```

- **Echo** used to output data to the screen.
- PHP file must save using **.PHP** extension

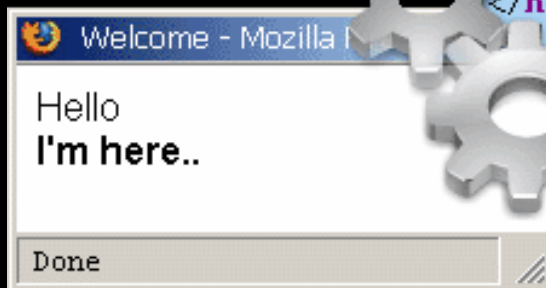
- 
- PHP code is executed on the server, generating HTML which is then sent to the client.
 - The client would receive the results of running that script, but would not know what the underlying code was.

ON SERVER

```
<html>
<head> <title>Welcome</title> </head>
<body>
<?
    echo "Hello";
    print "<br />";
    echo "<b>I'm here..</b>";
?>
</body>
</html>
```



```
<html>
<head> <title>Welcome</title> </head>
<body>
Hello<br /><b>I'm here..</b></body>
</html>
```



PHP Getting Started

On windows, you can download and install WAMP/Xampp.
With one installation and you get an Apache webserver, database server.

<http://www.wampserver.com>

On mac, you can download and install MAMP.

<http://www.mamp.info/en/index.html>

PHP Hello World

```
<html>
  <head>
    <title>PHP Test</title>
  </head>
  <body>
    <?php echo '<p>Hello World</p>'; ?>
  </body>
</html>
```

Above is the PHP source code.

PHP Hello World

It renders as HTML that looks like this:

```
<html>
  <head>
    <title>PHP Test</title>
  </head>
  <body>
    <p>Hello World</p>
  </body>
</html>
```

PHP Comments

In PHP, we use `//` to make a single-line comment or `/*` and `*/` to make a large comment block.

```
<html>
<body>

<?php
//This is a comment

/*
This is
a comment
block
*/
?>

</body>
</html>
```

Variable

- Variables are used for storing values, such as numbers, strings or function results, so that they can be used many times in a script.
- Variables are used for storing a values, like text strings, numbers or arrays.
- All variables in PHP are denoted with a leading dollar sign (\$).
 - Syntax:

\$variable name=value;

- A variable name can only contain alpha-numeric characters, underscores (a-z, A-Z, 0-9, and _)
- PHP is case sensitive \$Mystring and \$mystring are different variable



variable

- Integers :are whole numbers, without a decimal point, like 4195.

`$num=123`

- Doubles: are floating-point numbers, like 3.14159 or 49.1.

`$fl=21.4`

- Booleans: have only two possible values either true or false.

`$b=true`

- NULL: is a special type that only has one value: NULL.

`$n=NULL`

- Strings: are sequences of characters, like 'PHP supports string operations.'

`$s="hello world"`



PHP Operator

➤ Operators are used to operate on values. There are four classifications of operators:

1. Arithmetic
2. Assignment
3. Comparison
4. Logical

Arithmetic Operators

Operator	Description	Example	Result
+	Addition	x=2 x+2	4
-	Subtraction	x=2 5-x	3
*	Multiplication	x=4 x*5	20
/	Division	15/5 5/2	3 2.5
%	Modulus (division remainder)	5%2 10%8 10%2	1 2 0
++	Increment	x=5 x++	x=6
--	Decrement	x=5 x--	x=4



Assignment Operators

Operator	Example	Is The Same As
=	$x=y$	$x=y$
+=	$x+=y$	$x=x+y$
-=	$x-=y$	$x=x-y$
=	$x=y$	$x=x*y$
/=	$x/=y$	$x=x/y$
.=	$x.=y$	$x=x.y$
%=	$x\%=y$	$x=x\%y$



Comparison Operators

Operator	Description	Example
==	is equal to	5==8 returns false
!=	is not equal	5!=8 returns true
<>	is not equal	5<>8 returns true
>	is greater than	5>8 returns false
<	is less than	5<8 returns true
>=	is greater than or equal to	5>=8 returns false
<=	is less than or equal to	5<=8 returns true



Logical Operators

Operator	Description	Example
&&	and	x=6 y=3 (x < 10 && y > 1) returns true
	or	x=6 y=3 (x==5 y==5) returns false
!	not	x=6 y=3 !(x==y) returns true



Decision Making

- Very often when you write code, you want to perform different actions for different decisions. use conditional statements.
- **if statement** - use this statement to execute some code only if a specified condition is true
- **if...else statement** - use this statement to execute some code if a condition is true and another code if the condition is false
- **if...elseif...else statement** - use this statement to select one of several blocks of code to be executed
- **switch statement** - use this statement to select one of many blocks of code to be executed



If()

- The following example will output "Have a nice weekend!" if the current day is Friday:

```
<?php
$day="fri";
if($day=="fri")
{
    echo "have a nice weekend";
}
?>
```

If...else

- Use the if...else statement to execute some code if a condition is true and another code if a condition is false.

```
<?php
    $day="fri";
    if($day=="fri")
    {
        echo "have a nice weekend";
    }
    else
    {
        echo "have a nice day";
    }
```



If....elseif....else

```
<?php
    $day="fri";
    if($day=="fri"){
        echo "have a nice weekend"; }
    else if($day=="sun"){
        echo "have a enjoy day"; }
    else {
        echo "have a nice day"; }
?>
```


Switch..case

- The Switch statement in PHP is used to perform one of several different actions based on one of several different conditions.

```
<?php
```

```
$num=5;

switch($num)
{
case 1:
    echo "The number is one;
    break;
case 2:
    echo "The number is two ;
    break;
default:
    echo "No number found";
break;
}
```



Looping statement

- The same block of code to run over and over again in a row. Instead of adding several almost equal lines in a script we can use loops to perform a task like this.
- **while** - loops through a block of code while a specified condition is true
- **do...while** - loops through a block of code once, and then repeats the loop as long as a specified condition is true
- **for** - loops through a block of code a specified number of times

While loop

- The while loop will execute a block of code if and as long as a condition is true.

```
<?php
    $i=1;
    while($i<=5)
    {
        echo "The number is " . $i . "<br/>";
        $i++;
    }
?>
```

By Prof. Vaibhavi Patel

Do...while

- The do...while loop will execute a block of code at least once-it then will repeat the loop as long as a condition is true.

```
<?php
    $var=0;
    do
    {
        $var++;
        echo "The number is " . $var .
        "<br/>";
    }
    while ($var<5);

?>
```



For loop

- The for loop is used when you know how many times you want to execute a statement or a list of statements

```
<?php  
for ($i=1; $i<=5; $i++)  
{  
    echo "hiiii ! <br/>";  
}  
?>
```



Array

- An array is a special variable, which can store multiple values in one single variable.
- Instead of having many similar variables, you can store the data as elements an array.
- Each element in the array has its own index so that it can be easily accessed
- In PHP, there are three kind of arrays:
- **Numeric array** - An array with a numeric index
- **Associative array** - An array where each ID key is associated with a value
- **Multidimensional array** - An array containing one or more arrays

Numeric array

- A numeric array stores each array element with a numeric index.

```
<?php
```

```
$names = array("PHP","JAVA",".NET");
```

```
or
```

```
$names[0] = "PHP";
```

```
$names[1] = "JAVA";
```

```
$names[2] = ".NET";
```

```
echo $names[0] . " , " . $names[1] . " and " . $names[2] . " are languages";
```

```
?>
```

Associative Array

- With an associative array, each ID key is associated with a value.
- When storing data about specific named values, a numerical array is not always the best way to do it.
- With associative arrays we can use the values as keys and assign values to them.

```
<?php
```

```
$language= array("a"=>"PHP", "b"=>"JAVA",    "c"=>"ASP");  
echo "Language is " . $language['a']
```

```
?>
```


multidimensional array

- In a multidimensional array, each element in the main array can also be an array. And each element in the sub-array can be an array, and so on.

```
<?php  
$language = array("programming"=>array("PHP","ASP","JAVA"), "Designing  
"=>"HTML");
```

```
echo $language['programming']['1'];  
?>
```

Function

- A function is a block of code that can be executed whenever we need it.
- A function will be executed by a call to the function.

Syntax:

```
function fun_name(parameter 1,parameter2.....)
{
    code of block
}
```

Example

```
<?php
    function test()
    {
        echo“abc Technology”;
    }
    test();
?>
```



Adding parameters...

- To add more functionality to a function, we can add parameters. A parameter is just like a variable.
- Parameters are specified after the function name, inside the parentheses.

```
<?php
function test($fname)
{
    echo $fname. "Technology.<br/>";
}
echo "My company is ";
test("abc");
?>
```



isset()

- `isset()` — Determine if a variable is set and is not NULL

```
<?php
// $p = 'tops';
// This will evaluate to TRUE so the text will be printed
if(isset($p))
{
    echo "This var is set so I will print.";
}
?>
```

File inclusion

- You can insert the content of a file into a PHP file before the server executes it, with the `include()` or `require()` function.

```
<html>  
<body>  
<?php
```

```
    include("header.php"); or require("header.php");
```

- ```
?>
<h1>Welcome to my home page</h1>
<p>Some text</p>
</body>
</html>
```



# PHP Global variable

- Several predefined variables in PHP are "superglobals", which means that they are always accessible, regardless of scope - and you can access them from any function, class or file without having to do anything special.

- **Super global variable**

`$GLOBALS`

`$_SERVER`

`$_REQUEST`

`$_POST`

`$_GET`

`$_FILES`

`$_COOKIE`

`$_SESSION`



# GET & POST

- PHP `$_GET` can also be used to collect form data after submitting an HTML form with `method="get"`.
- PHP `$_POST` is widely used to collect form data after submitting an HTML form with `method="post"`.
- The get method is not suitable for large variable values; the value cannot exceed 100 chars.





# THANK YOU

By Prof. Vaibhavi Patel