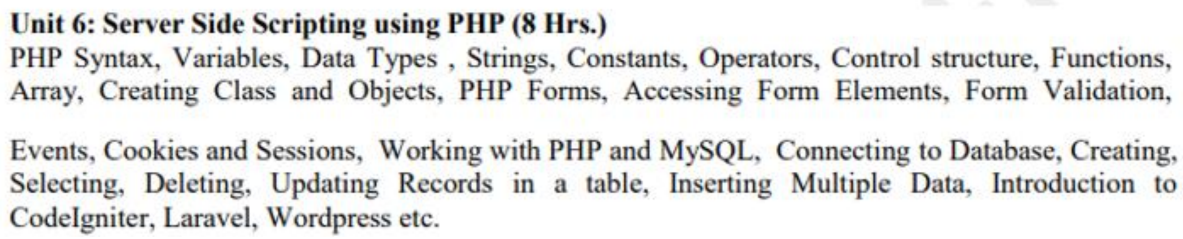
BSc CsIt SEM - V
Web technology

Unit 6: Server-Side Scripting using PHP

Syllabus:



Unit 6: Server Side Scripting using PHP (8 Hrs.)
PHP Syntax, Variables, Data Types , Strings, Constants, Operators, Control structure, Functions, Array, Creating Class and Objects, PHP Forms, Accessing Form Elements, Form Validation,

Events, Cookies and Sessions, Working with PHP and MySQL, Connecting to Database, Creating, Selecting, Deleting, Updating Records in a table, Inserting Multiple Data, Introduction to CodeIgniter, Laravel, Wordpress etc.

Introduction to PHP

PHP stands for Hypertext Pre-processor which is a server-side scripting language used to develop static or dynamic websites or web applications. PHP uses extension ".php" and is widely used, free and efficient. PHP code are executed on the server and the result is returned to the browser as plain HTML. When a browser requests a document that includes PHP script, the web server that provides the documents calls it PHP processor. The server determines that a document includes PHP script by the file name extension ".php" or ".php3" or ".phtml". Besides giving out on plain HTML, PHP can output images, PDF files and even flash movies. When a PHP processor finds only markup code (client-side scripting) in the input file then it simply copies to the output file. When the PHP processor encounter PHP script in the input file, it interprets it and sends any output of the script to the output file. Such output file is sent to the requesting browser and shows output as plain HTML.

Syntax for Using PHP

PHP scripts are either embedded in markup documents or are in separate file referenced on current document. PHP code can be embedded in any document by enclosing it between <?php ?>;.

Syntax:

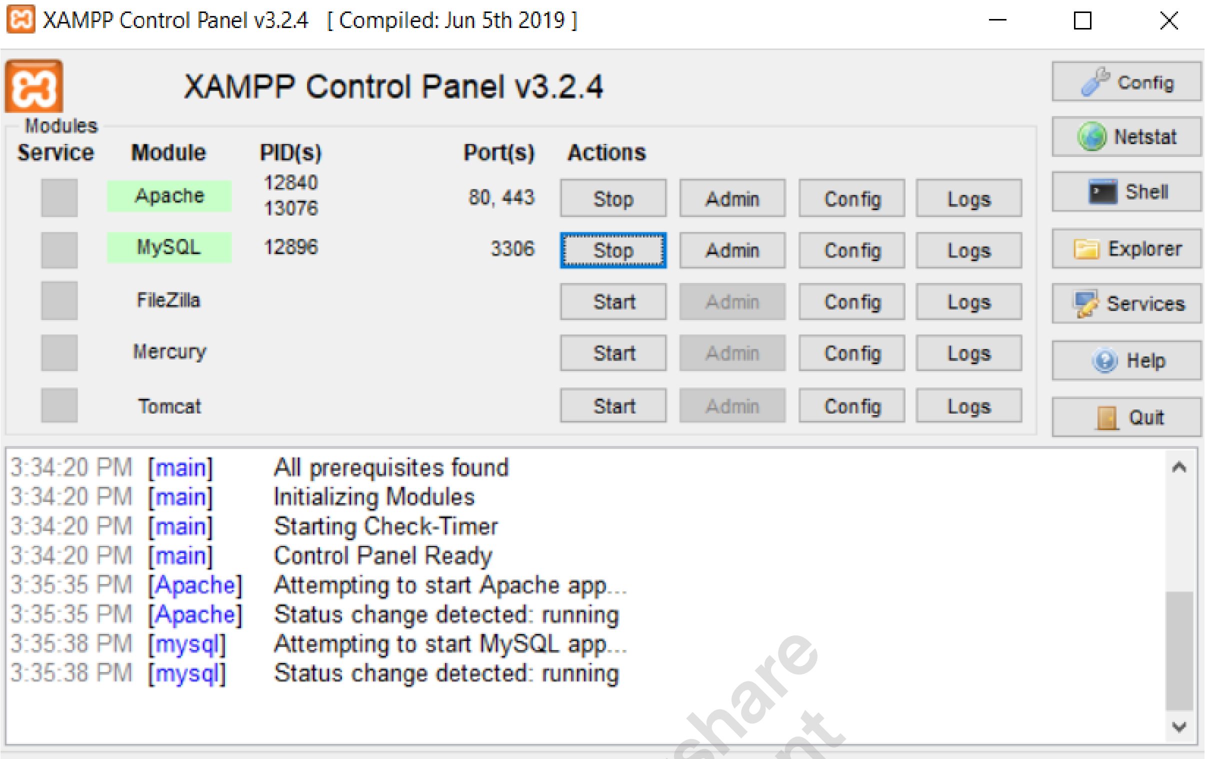
```
<?php
   // necessary php code
 ?>
```

This php tag can be used anywhere in a document and php file should be saved using ".php" extension.

Steps for Running PHP File:

Step 1: local server and editor is required. We will use xampp as local server and sublime or visual studio code as editor.
Download xampp: https://www.apachefriends.org/download.html
Download visual studio code: https://code.visualstudio.com/download
Download sublime text: https://www.sublimetext.com/3

Step 2: After Xampp is downloaded start apache and mysql from xampp console.



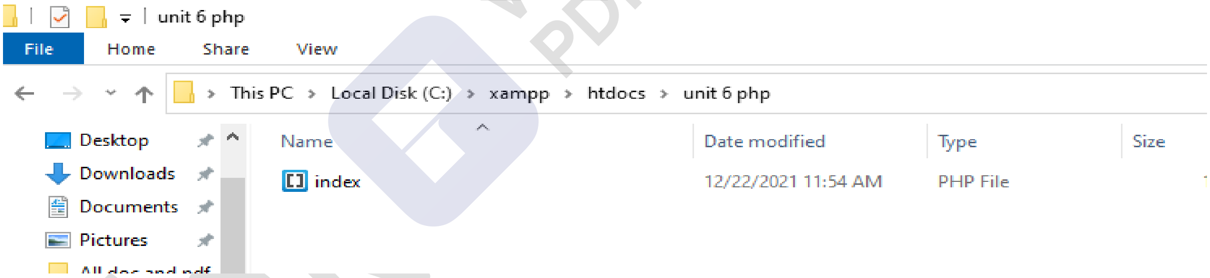
XAMPP Control Panel v3.2.4 [Compiled: Jun 5th 2019]

XAMPP Control Panel v3.2.4

Service	Module	PID(s)	Port(s)	Actions			
	Apache	12840 13076	80, 443	Stop	Admin	Config	Logs
	MySQL	12896	3306	Stop	Admin	Config	Logs
	FileZilla			Start	Admin	Config	Logs
	Mercury			Start	Admin	Config	Logs
	Tomcat			Start	Admin	Config	Logs

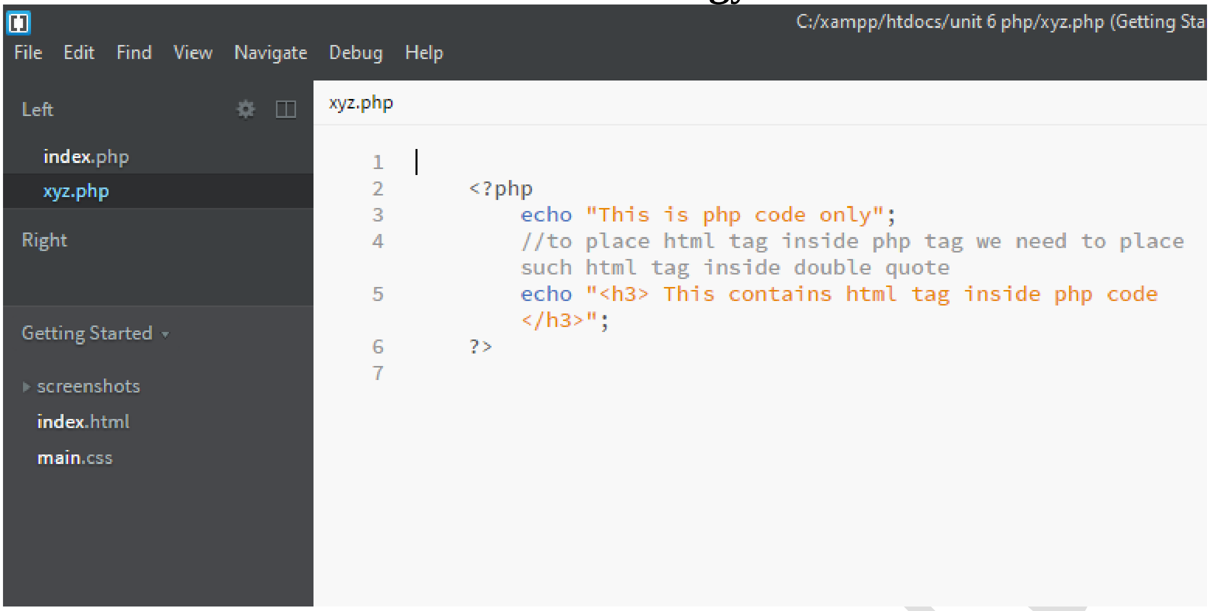
```
3:34:20 PM  [main]    All prerequisites found
3:34:20 PM  [main]    Initializing Modules
3:34:20 PM  [main]    Starting Check-Timer
3:34:20 PM  [main]    Control Panel Ready
3:35:35 PM  [Apache]  Attempting to start Apache app...
3:35:35 PM  [Apache]  Status change detected: running
3:35:38 PM  [mysql]   Attempting to start MySQL app...
3:35:38 PM  [mysql]   Status change detected: running
```

Step 3: create a new folder on htdocs folder of xampp where your php file will resides. In my case: I have xampp folder on C drive and I have created a new folder named unit 6 php on htdocs folder.



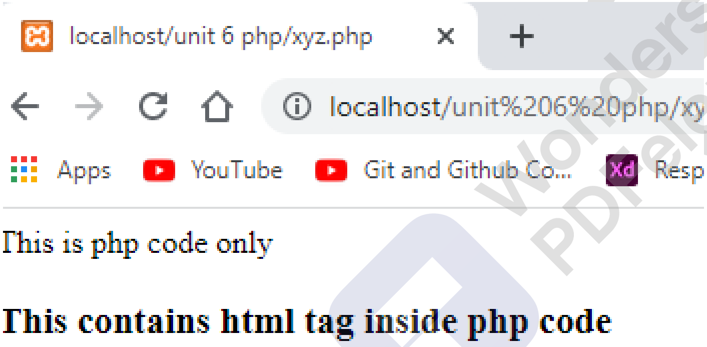
Name	Date modified	Type	Size
index	12/22/2021 11:54 AM	PHP File	

Step 4: Open editor (sublime or visual studio code or brackets) and from such editor open a folder you have just created. After opening folder, create a new file with extension “.php”. In my case I have use visual studio code as editor and created index.php file on practice folder.


C:/xampp/htdocs/unit 6 php/xyz.php (Getting Sta
File Edit Find View Navigate Debug Help
Left
index.php
xyz.php
Right
Getting Started
screenshots
index.html
main.css
xyz.php

```
<?php
    echo "This is php code only";
    //to place html tag inside php tag we need to place
    such html tag inside double quote
    echo "<h3> This contains html tag inside php code
    </h3>";
?>
```

Step 5: to see output, go on web browser and type localhost/folder name/file name. In my case: localhost/unit 6 php/xyz.php. if you have put index.php as file name then you don’t have to mention file name. you can put localhost/folder name.



This is php code only

This contains html tag inside php code

Embedding PHP within HTML:

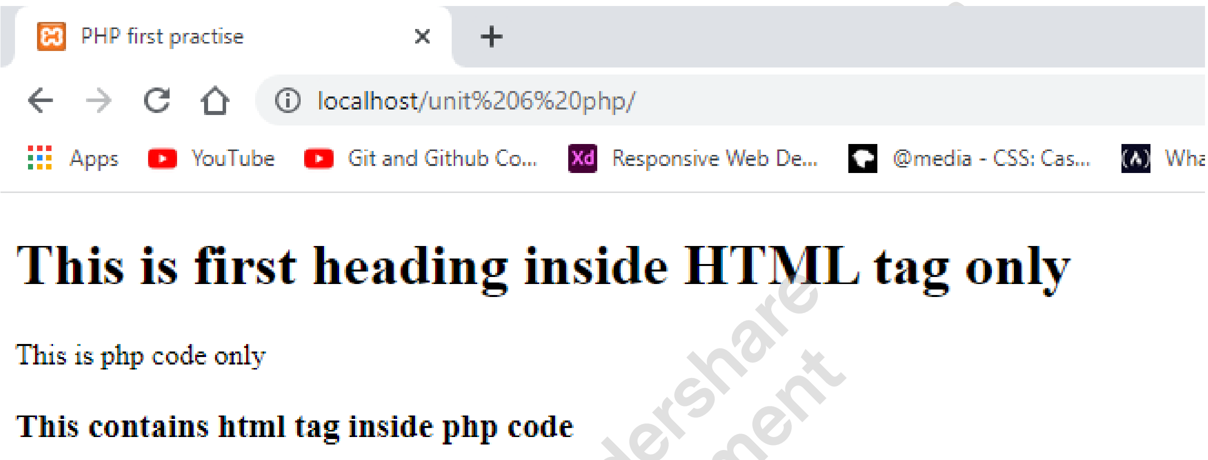
PHP file can contain markup or client-side script like HTML, CSS and JavaScript and when it encounters such file it gives output in plain HTML form. HTML script can run in PHP file and we can put PHP code anywhere in the document or anywhere in HTML script. Html tag can be used on php code only inside double quote. Semicolon should be used for indicating the end of statement otherwise parse error message will be produced. PHP will treat multiple statement like one statement if semicolon is not used at end of statement.

Example:

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <title>PHP first practise</title>
</head>
<body>
```

```
    <h1> This is first heading inside HTML tag only </h1>
    <?php
       echo "This is php code only";
       //to place html tag inside php tag we need to place such html tag inside double quote
       echo "<h3> This contains html tag inside php code </h3>";
    ?>
</body>
</html>
```

Output:

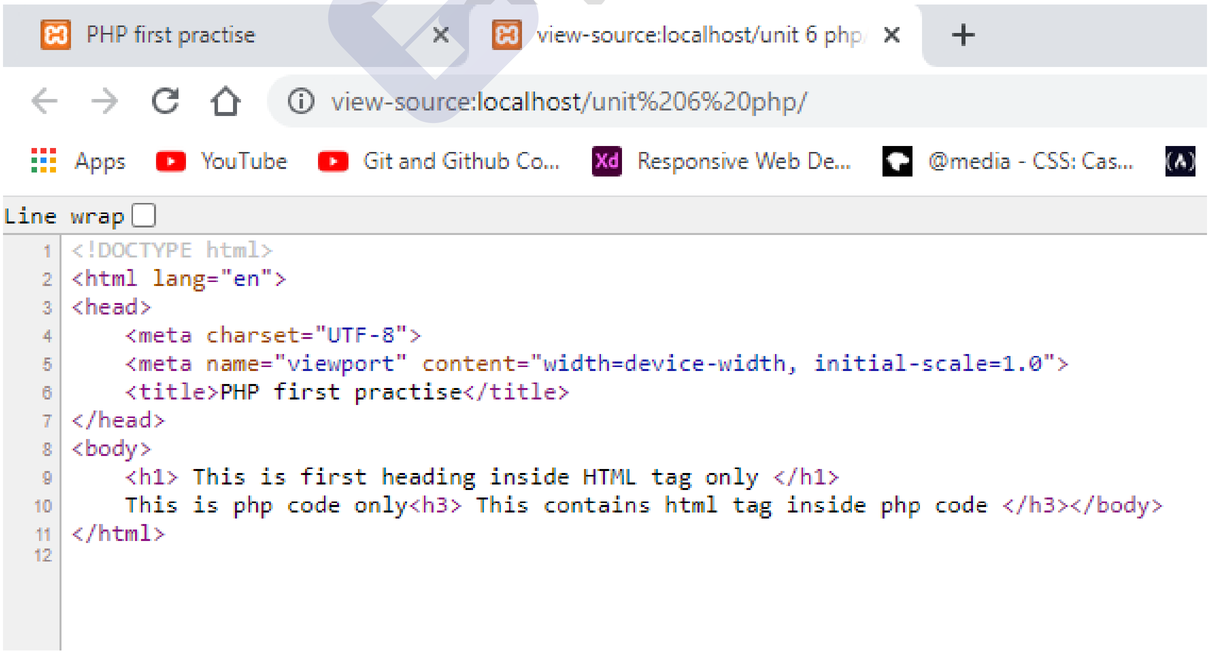


This is first heading inside HTML tag only

This is php code only

This contains html tag inside php code

While inspecting a document, only HTML script are visible but PHP code are hide by the browser which provides security from external intrusion.
For example: for above output press ctrl+u or right click on browser and press view page source. It will show:



```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>PHP first practise</title>
</head>
<body>
    <h1> This is first heading inside HTML tag only </h1>
    This is php code only<h3> This contains html tag inside php code </h3></body>
</html>
```

In above figure, only HTML script are shown but php code is hidden.

Comments in PHP:

Comments are used to add some description of code i.e. it provides information about code to make it more readable. Comments line are not executable statement so, any things inside comment line are ignored by browser. It is mostly used for documenting the code. There are two ways to add comment in php. The **first way** is using a pair of forward slashes i.e. "//" which turns a whole single line into comment. The **second way** is using /* */ symbol which will turns multiple line into comment. Anything between such symbol are ignore by browser. We can use comment any where inside code.

Example:

```
<?php
        //this is single line comment
        /* this is multiple line comment
                All content is ignored until "/*" symbol is placed
        */
?>
```

PHP Echo and Print Command:

Both echo and print are used to produce output to the screen and both the statement can be used with or without parenthesis.

Syntax:

```
        echo "hello"; or echo("hello");
        Print "hello"; or print("hello");
```

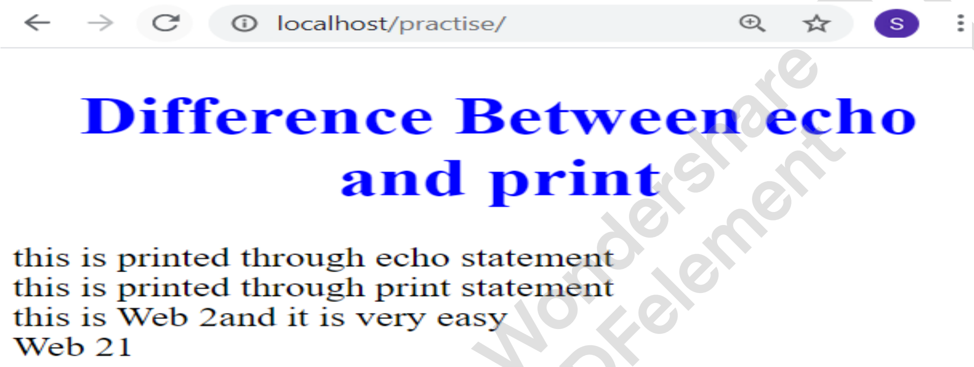
The two commands are quite similar but there is small difference on print and echo. **Print** is a function like construct that takes a single parameter and has a return value which is always 1 whereas **echo** is purely php language construct and can take multiple parameters. **Echo** is faster than print because it does not have return value**. Print** can be used in more complex expression as it is implemented like a function but **echo** cannot be used in more complex expression.

Example on Difference Between Echo and Print:

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <title>integer data type</title>
</head>
<body>
<h1 style="color: blue;text-align: center;"> Difference between echo and print </h1>
   <?php
```

```
Echo “ this is printed through echo statement  </br>”;
Print “this is printed through print statement</br>”
$subject= “ web 2”;
$comment= “it is very easy”;
$res=print $subject;
Echo $res;

    ?>
</body>
</html>
```

Output:



Difference Between echo and print

```
this is printed through echo statement
this is printed through print statement
this is Web 2and it is very easy
Web 21
```

Variable:

Variables are used to store data temporarily i.e. they are like container for storing data. In PHP, variables are dynamically typed that means a variable can be used for anything or their types are not determined. The type of the variable is set every time the variable is assigned a value i.e. PHP has no command for declaring a variable.

In PHP, variables are declared using dollar “$” symbol.

Syntax:

```
$ (name of variable) = value;
$ course = “Web II”;
```

A variable can contain text, numbers, array element, Boolean value and empty value known as null. A variable with unassigned value has the value NULL which is a type of NULL and such NULL value is changed to another value depends on the context of use. If a context specifies a number, Null is changed to 0 and if a context specifies a string, Null is changed to empty string. IsSet function can be used to test whether a variable contains value or not. This function takes variable name as input and returns a Boolean value i.e. true if a variable contains non-null value otherwise false.

Rules for Naming Variables:

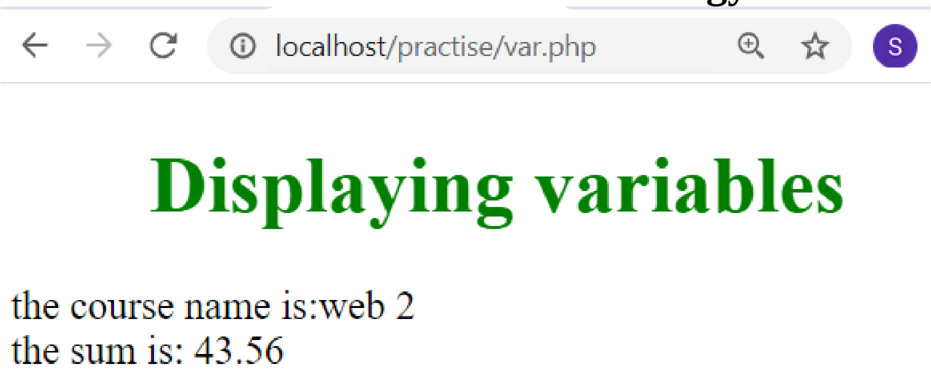
- Variable name after the dollar sign must starts with letter of alphabet or the _(underscore) character. Variable name should not start with numbers or other character. For example
  $name, $className, $subject1 ⟶ right way
  $1name, $-name, $32subject ⟶ wrong way

- A variable name can contains only alphanumeric character and underscores (a-z, A-Z, 0-9 and _).
- Variable may not contain space. If more than one words should be used then a good way is to separate words with the underscore (_) character. For example: $course_name;
- Variables name are case sensitive. For example, $age and $Age are not same.

Following example provides creating variables, assigning values and displaying.

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <title>integer data type</title>
</head>
<body>
<h1 style="color: blue;text-align: center;"> Displaying variables. </h1>
   <?php
         $cname= “web 2”;
         $firstnum=10.56;
         $secondnum=33;
         $sum= $firstnum+$secondnum;
         Echo “the course name is :”, $course, “<br/>”;
         Echo “the sum is :”,$sum;

   ?>
</body>
</html>
```

Output:

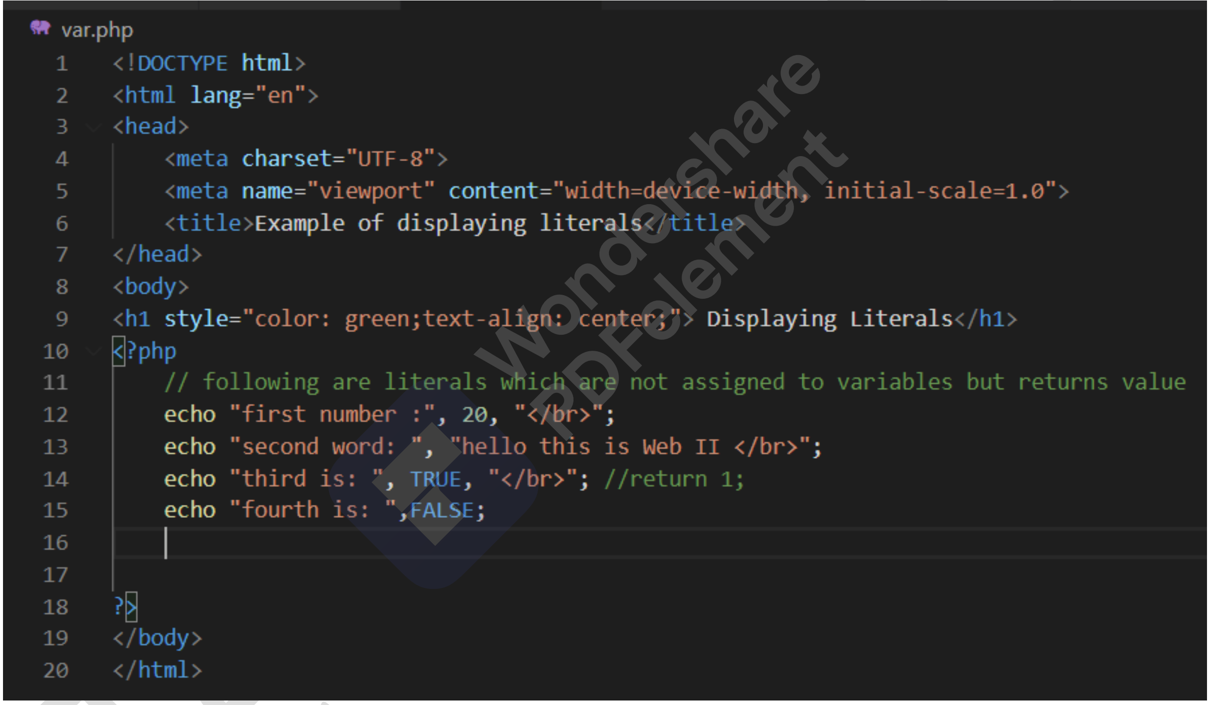


Displaying variables

the course name is:web 2
the sum is: 43.56

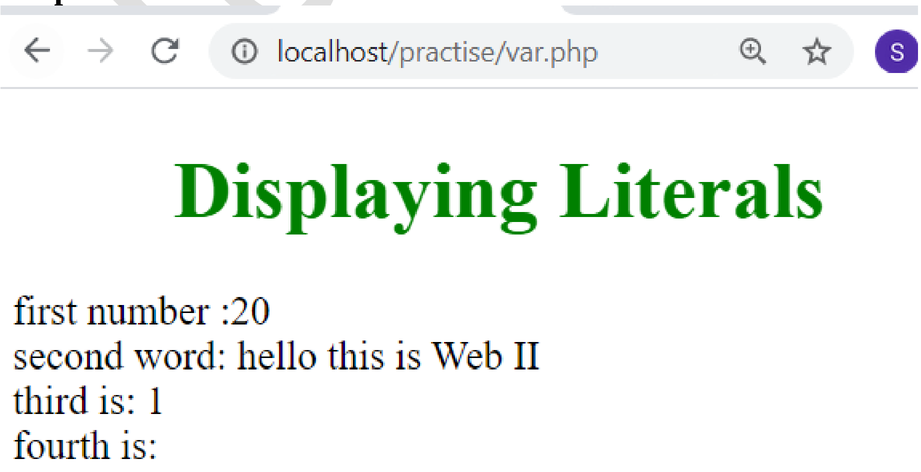
Literals:

Literals simply means something that evaluates to itself such as number (44, 55 etc) or string ("hello") i.e. literals are not assigned to variables. The simplest expression is just a single literal or variable because both return a value. For example:



```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Example of displaying literals</title>
</head>
<body>
<h1 style="color: green;text-align: center;"> Displaying Literals</h1>
<?php
    // following are literals which are not assigned to variables but returns value
    echo "first number :", 20, "</br>";
    echo "second word: ", "hello this is Web II </br>";
    echo "third is: ", TRUE, "</br>"; //return 1;
    echo "fourth is: ",FALSE;


?>
</body>
</html>
```

Output:



Displaying Literals

first number :20
second word: hello this is Web II
third is: 1
fourth is:

BSc CsIt SEM - V
Web technology

PHP Data Types:

PHP supports 8 primitive data types that can be categorized further in 3 types:

- Scalar Types (predefined): PHP has four scalar types – Boolean, integer, double and string.
- Compound Types (user-defined): two compound type- array and object
- Special Types: two special types- resource and NULL

Integer type:

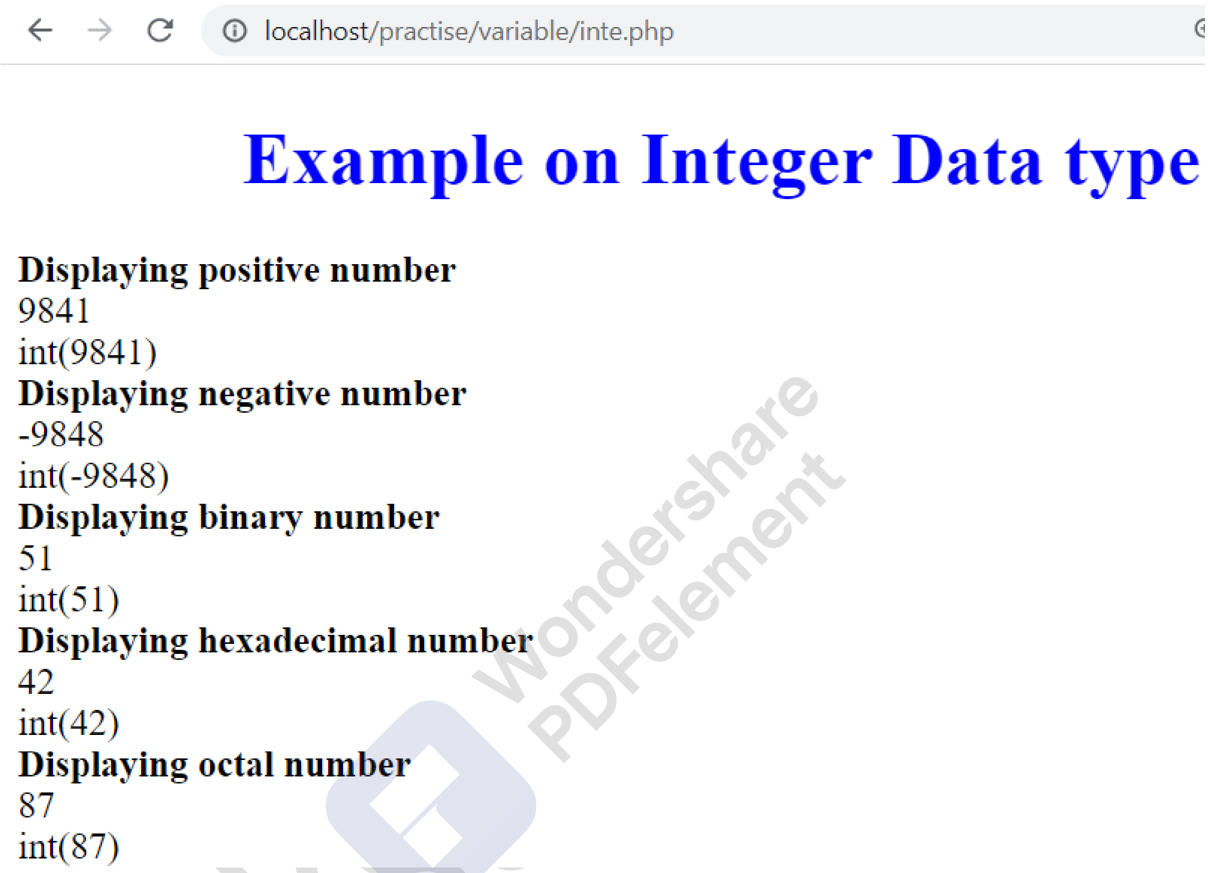
Integer are whole number without a decimal point like (1, 2, 3 ,4, -2, -3 etc.). integer has range from -2,147,483,648 to 2,147,483,648 and can be either positive or negative and can be specified in: decimal (base 10), hexadecimal (base 16) written using 0x and octal (base 8) prefixed with 0.

Note: var_dump(); function can be using to know type and value of variable.

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <title>integer data type</title>
</head>
<body>
<h1 style="color: blue;text-align: center;"> Example on Integer Data type </h1>
   <?php
      $positiveNum = 9841; //positive number
      $negNum = -9848; //negative number
      $hexaNum = 0x2A; //hexadecimal number
      $octNum = 0127; //octal number
      $binNum = 0b110011; //binary number
      echo "<b> Displaying positive number </b> </br>";
      echo $positiveNum,"</br>";
      var_dump($positiveNum); //shows type and value of variable
      echo "</br>";

      echo "<b> Displaying negative number </b> </br>";
      echo $negNum, "</br>";
      var_dump($negNum);
      echo "</br>";
      echo "<b> Displaying binary number </b> </br>";
      echo $binNum, "</br>"; //will convert to decimal
      var_dump($binNum);
      echo "</br>";
      echo "<b> Displaying hexadecimal number </b> </br>";
      echo $hexaNum, "</br>";
      var_dump($hexaNum);
      echo "</br>";
```

```
        echo "<b> Displaying octal number </b> </br>";
        echo $octNum, "</br>";
        var_dump($octNum);
        echo "</br>";
    ?>
</body>
</html>
```

Output:



localhost/practise/variable/inte.php

```
Example on Integer Data type

Displaying positive number
9841
int(9841)
Displaying negative number
-9848
int(-9848)
Displaying binary number
51
int(51)
Displaying hexadecimal number
42
int(42)
Displaying octal number
87
int(87)
```

Floating Point or Doubles

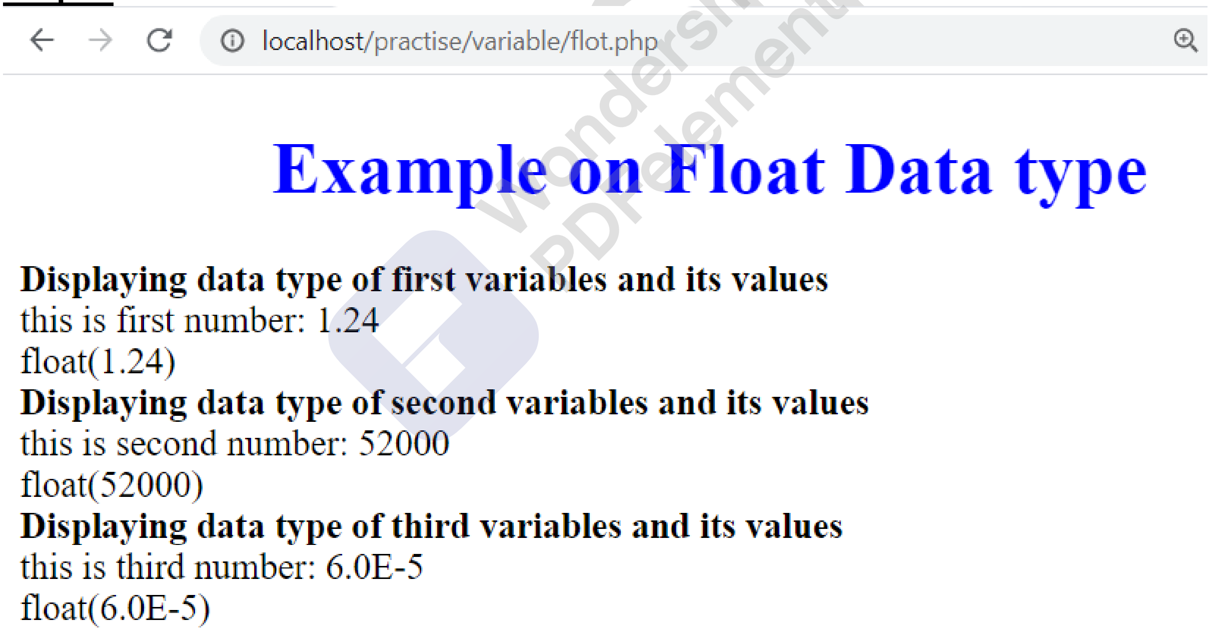
A floating-point numbers (real)are decimal or fractional number with a decimal point or a number in an exponential form. For example: 1.24, 10.5e2, 4E-5 etc.
Following examples provide different ways to create floating type variable and displaying result:

Example:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Floating Type</title>
</head>
<body>
<h1 style="color: blue;text-align: center;"> Example on Float Data type </h1>
<?php
```

```
    $firstNum = 1.24;
    $secondNum = 5.2e4; //exponential number
    $thirdNum = 6E-5;

    echo "<b>Displaying data type of first variables and its values</b></br>";
    echo "this is first number: ", $firstNum, "</br>";
    var_dump($firstNum);
    echo "</br>";
    echo "<b>Displaying data type of second variables and its values</b></br>";
    echo "this is second number: ", $secondNum, "</br>";
    var_dump($secondNum);
    echo "</br>";
    echo "<b>Displaying data type of third variables and its values</b></br>";
    echo "this is third number: ", $thirdNum, "</br>";
    var_dump($thirdNum);
    echo "</br>";
?>
</body>
</html>
```

Output:



localhost/practise/variable/flot.php

Example on Float Data type

Displaying data type of first variables and its values
this is first number: 1.24
float(1.24)
Displaying data type of second variables and its values
this is second number: 52000
float(52000)
Displaying data type of third variables and its values
this is third number: 6.0E-5
float(6.0E-5)

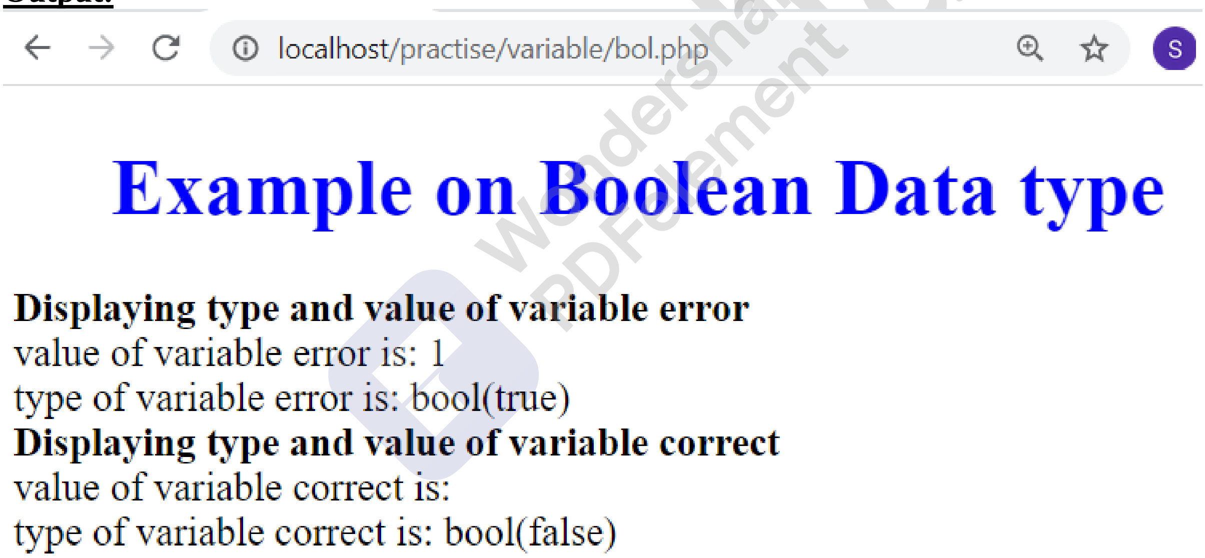
Boolean:

Boolean represent two possible states either true (1) or false (NULL) and used in conditional testing. Boolean values are usually used for evaluating relation or Boolean expression. Following examples shows on how Boolean variable are created and executed:

Example:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Boolean data type</title>
</head>
<body>
<h1 style="color: blue;text-align: center;"> Example on Boolean Data type </h1>
<?php
    $error = true;
    $correct = FALSE;
    echo "<b> Displaying type and value of variable error </b> </br>";
    echo "value of variable error is: ",$error, "</br>"; //shows value of variable
    echo "type of variable error is: ", var_dump($error); // shows type of variable
    echo "</br>";
    echo "<b> Displaying type and value of variable correct </b> </br>";
    echo "value of variable correct is: ",$correct, "</br>"; //will not display any value
    echo "type of variable correct is: ", var_dump($correct);
    echo "</br>";
?>
</body>
</html>
```

Output:



Example on Boolean Data type

Displaying type and value of variable error
value of variable error is: 1
type of variable error is: bool(true)
Displaying type and value of variable correct
value of variable correct is:
type of variable correct is: bool(false)

String Type:

A string is a sequence of character which are written inside single or double quote in PHP. For example, “hello web II” or ‘hello world’. If single quote is used then it will preserve the exact content. Anything inside single quote will be consider as sequence of character whether it is variable or anything else. If double quote is used PHP will evaluate such variable as variable not string i.e. the value of variable will be shown.

If actual single or double quote is needed in any places while writing string literals using single or double quote then backslash should be used. For example:

$content = ‘ I’ve never seen this type of question’. //produce error
Correct way to write this statement is:
$content = ‘I\’ve never seen this type of question’.

Sometimes a string needs to contain special characters which convey special meaning such as \t for tab, \n for new line, \r for carriage return etc. This is known as escape character. The combination of escape character (\ and a letter) are used to signify that the character after escape character should be treated specially. Some of the escape character and their function are described in table below:

Escape Character	Function
\”	to print next character as double quote, not as a string closer
\’	To print next character as single quote, not as a string closer
\n	Print new line character like print statement
\t	To tab character (put space)
\$	To print dollar symbol
\\	Print next character as a backslash not a escape character
\r	Print a carriage return (not used much)

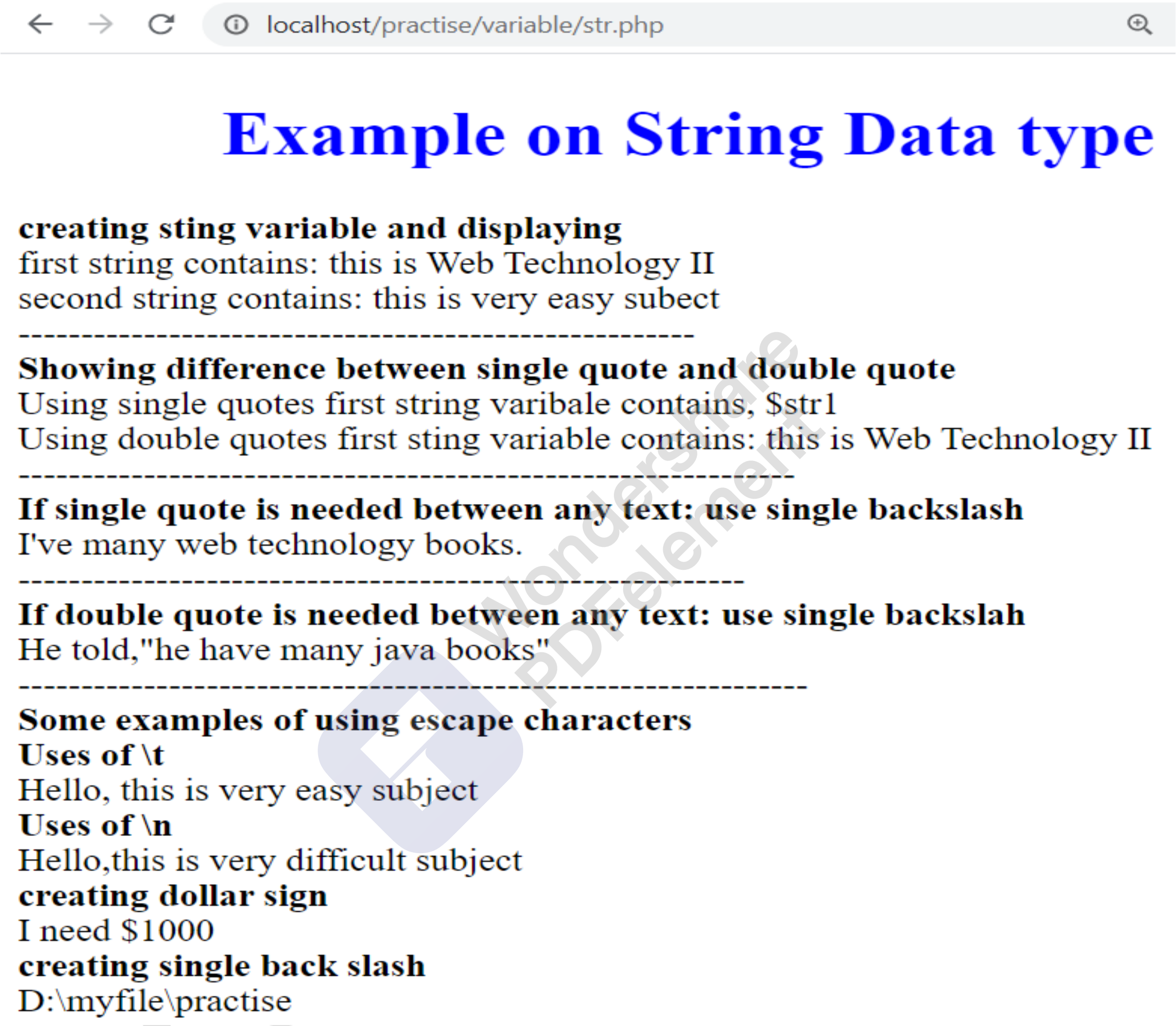
Following example will shows how string variable are created, how escape character is written, difference between single and double quote.

Example:

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>String Example</title>
</head>
<body>
  <h1 style="color: blue;text-align: center;"> Example on String Data type </h1>
  <?php
    echo "<b> creating sting variable and displaying </b></br>";
    $str1 = "this is Web Technology II";
    $str2 = "this is very easy subect";
    echo "first string contains: ", $str1, "</br>";
    echo "second string contains: ", $str2, "</br>";
    echo "------------------------------------------------------------- </br>";
    echo "<b>Showing difference between single quote and double quote </b></br>";
   echo 'Using single quotes first string varibale contains, $str1 </br>'; //treats variable as string
 echo "Using double quotes first sting variable contains: $str1 </br>"; //Shows value of variable
    echo "---------------------------------------------------------------------- </br>";
    echo '<b>If single quote is needed between any text: use single backslash </b></br>';
    echo 'I\'ve many web technology books.</br>';
    echo '----------------------------------------------------------------- </br>';
    echo "<b>If double quote is needed between any text: use single backslah </b> </br>";
    echo "He told,\"he have many java books\"</br>";
    echo "------------------------------------------------------------------------</br>";
    echo "<b>Some examples of using escape characters</b></br>";
    echo '<b>Uses of \t </b> </br>';
    echo "Hello, this\tis very easy subject </br>"; // \t will insert space
    echo '<b> Uses of \n </b></br>';
    echo "Hello,this is very difficult subject\n </br>"; // for new line character
    echo '<b>creating dollar sign </b> </br>';
    echo "I need \$1000 </br>"; // \$ will put dollar sign
    echo '<b> creating single back slash</b></br>';
```

```
        echo "D:\\myfile\\practise"; // to put single backslash
    ?>
</body>
</html>
```

Output:



localhost/practise/variable/str.php

```
Example on String Data type

creating sting variable and displaying
first string contains: this is Web Technology II
second string contains: this is very easy subect
----------------------------------------------------
Showing difference between single quote and double quote
Using single quotes first string varibale contains, $str1
Using double quotes first sting variable contains: this is Web Technology II
------------------------------------------------------------
If single quote is needed between any text: use single backslash
I've many web technology books.
--------------------------------------------------------
If double quote is needed between any text: use single backslah
He told,"he have many java books"
-------------------------------------------------------------
Some examples of using escape characters
Uses of \t
Hello, this is very easy subject
Uses of \n
Hello,this is very difficult subject
creating dollar sign
I need $1000
creating single back slash
D:\myfile\practise
```

String Concatenation:

Concatenation means joining or putting something after another thing. To join another character or string or to append one string of characters to another, dot or period (.) symbol is used. For example: echo “this is” . $course;

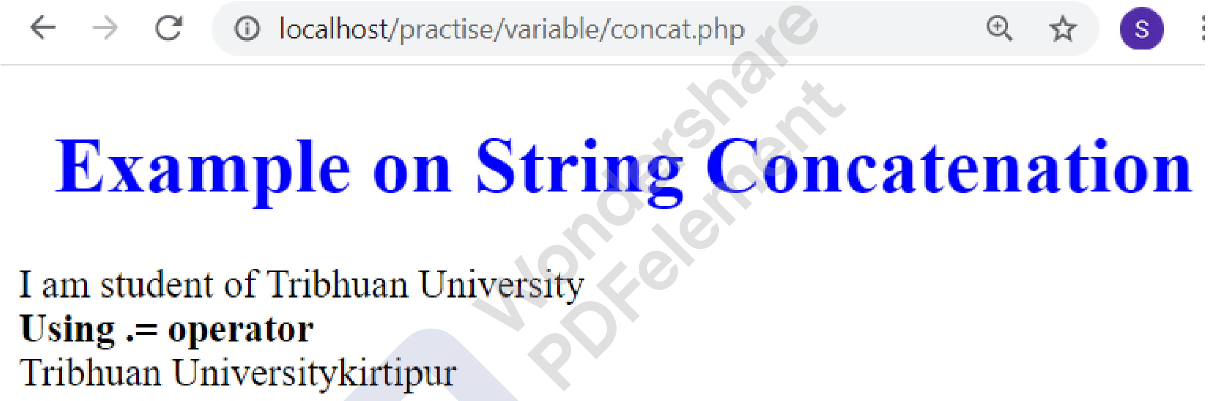
“.=” operator can also be used to append one string to another string.

For example:

```
<!DOCTYPE html>
<html lang="en">
```

```
<head>
   <meta charset="UTF-8">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <title>Document</title>
</head>
<body>
   <h1 style="color: blue;text-align: center;"> Example on String Concatenation </h1>
   <?php
      $college = "Tribhuan University";
      $location = "kirtipur";
      echo "I am student of ".$college."</br>"; // using dot operator
      echo "<b>Using .= operator </b></br>";
      $college .=$location;
      echo $college;
   ?>
</body>
</html>
```

Output:



Example on String Concatenation

```
I am student of Tribhuan University
Using .= operator
Tribhuan Universitykirtipur
```

PHP Constant:

Constants are identifiers same like variable except that once they are defined their value cannot be changed or undefined. Once constant is defined its value is set to the remainder of the program and cannot be altered. Common example of declaring constant is for configuration setting like database username and password, server location, company name, website's base URL etc. Constant are automatically global across the entire script.
In PHP, constant are defined using define() function which takes three argument: first is name of the constant, second is value of constant. And third is case insensitive which specifies whether constant name should be case insensitive or not. Default is false After defining constant, its value can be accessed by referring its constant name.

Syntax:

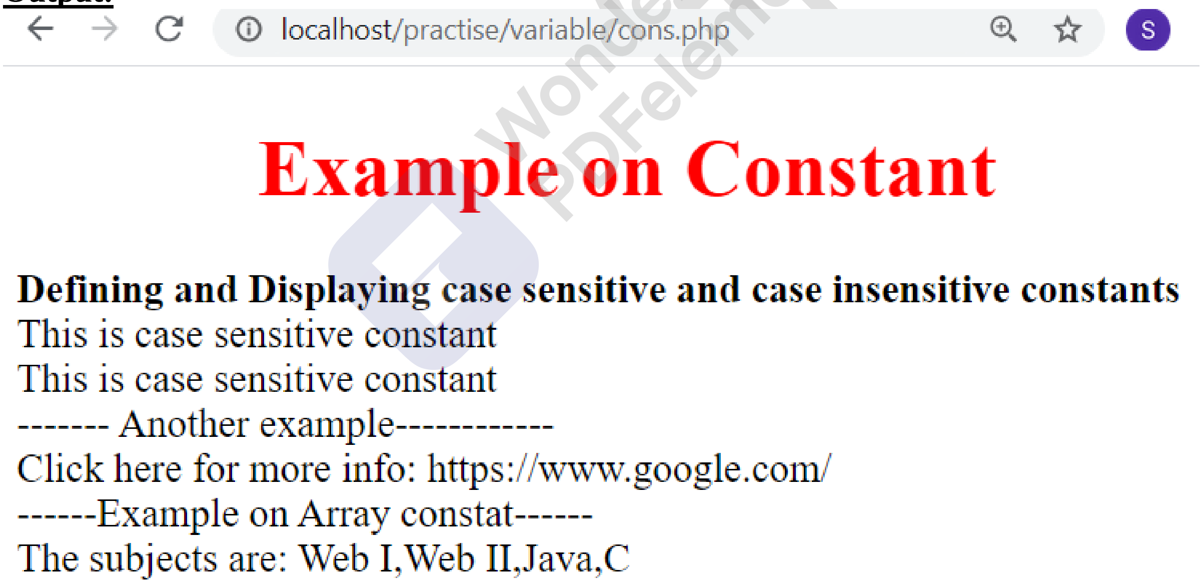
define(name of constant, value of constant, case-insensitive);

For example: define("coreSubject", "Web Technology II", true);

```
<!DOCTYPE html>
<html lang="en">
<head>
```

```
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Constant example</title>
</head>
<body>
<h1 style="color: red;text-align: center;"> Example on Constant  </h1>
<?php
echo "<b>Defining and Displaying case sensitive and case insensitive constants</b></br>";
define("result","This is case sensitive constant",false); //case sensitive constant
define("res","This is case sensitive constant",true); //case insensitive constant

echo result,"</br>";
echo res,"</br>";
echo "------- Another example------------</br>";
define ("URL","https://www.google.com/");
echo "Click here for more info: ",URL,"</br>";
echo "------Example on Array constat------</br>";
define("subject",["Web I","Web II","Java","C"]);
echo "The subjects are: ".subject[0].",".subject[1].",".subject[2].",".subject[3];
?>
</body>
</html>
```

Output:



Example on Constant

Defining and Displaying case sensitive and case insensitive constants
This is case sensitive constant
This is case sensitive constant
------- Another example------------
Click here for more info: https://www.google.com/
------Example on Array constat------
The subjects are: Web I,Web II,Java,C

Operators:

An operator is symbol that performs some mathematical or logical manipulation. Operator are used in programs to manipulate data and variables and they usually form a part of the mathematical or logical expressions. Each operator takes on different number of operand like: **unary operator** which is used for performing operation on single operand like increment (++), decrement (- -) and negation (!$a) Following are the types of operator used in PHP, **binary operator** which is used for performing operation on two operand like addition, subtraction etc, and final one is **ternary operator** which takes the form expr ? x : y which requires three operands

Arithmetic Operators:

Arithmetic operators are used for performing some arithmetic operation like addition, subtraction, multiplication etc. with numeric values. Following are the different types of arithmetic operators:

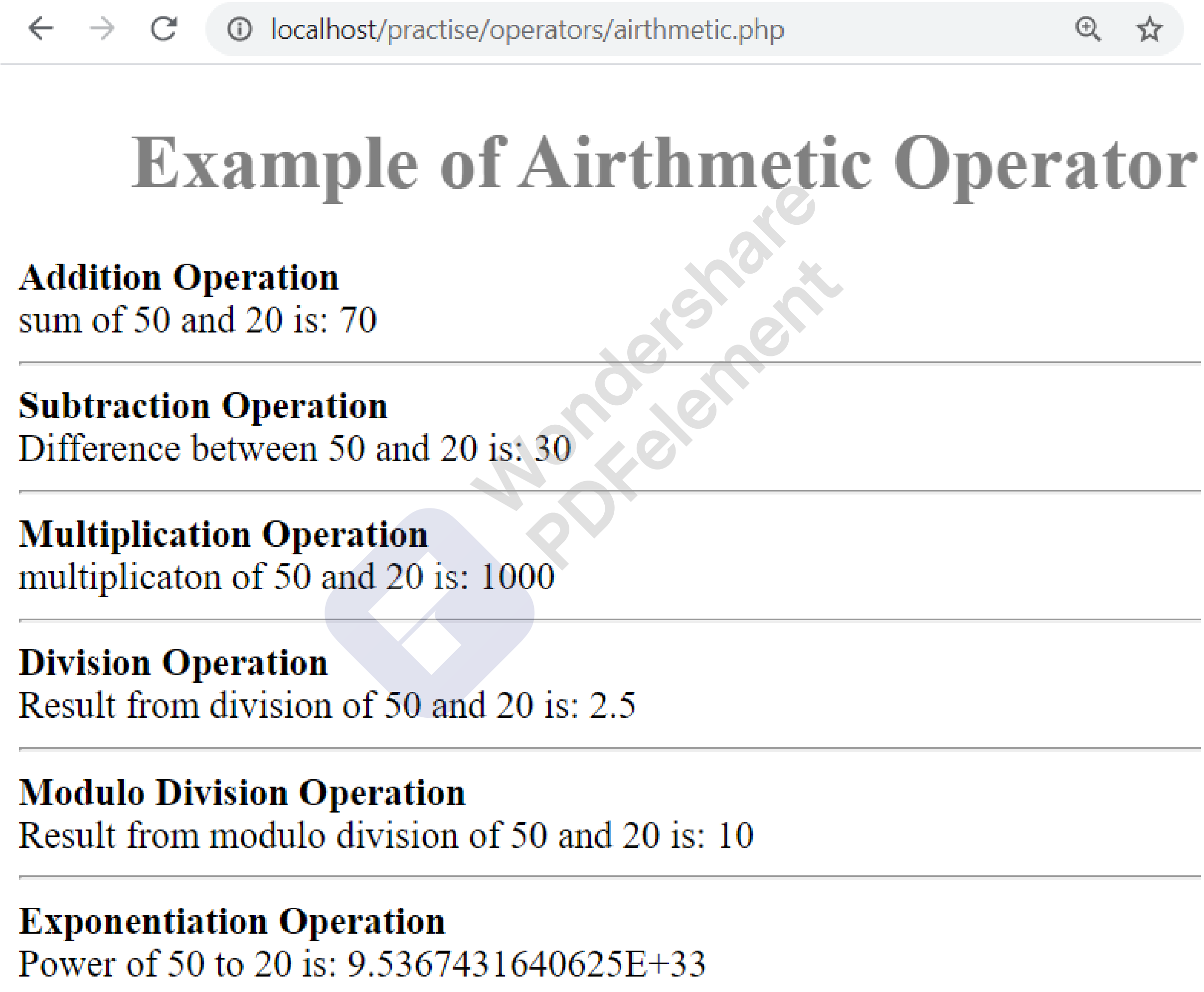
Operator	Description	Example
+	Use for addition	$x + $y
-	Use for subtraction	$x - $y
*	Use for multiplication	$x * $y
/	Use for division. Gives quotient	$x/$y
%	Use for modulo division. Gives remainder	$x %y
**	use for raising $x to the $yth power	$x ** $y

For Example:

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <title>Example of assignment operator</title>
</head>
<body>
   <h1 style="color: grey;text-align: center;"> Example of Airthmetic Operator </h1>
   <?php
      $x = 50;
      $y = 20;
      echo "<b>Addition Operation</b><br/>";
      echo "sum of $x and $y is: ",$x+$y, "<br/>";
      echo "<hr/>";
      echo "<b>Subtraction Operation</b><br/>";
      echo "Difference between $x and $y is: ", $x-$y, "<br/>";
      echo "<hr/>";
      echo "<b> Multiplication Operation</b><br/>";
      echo "multiplicaton of $x and $y is: ",$x*$y,"<br/>";
      echo "<hr/>";
```

```
        echo "<b>Division Operation</b><br/>";
        echo "Result from division of $x and $y is: ",$x/$y,"<br/>";
        echo "<hr/>";
        echo "<b>Modulo Division Operation</b><br/>";
        echo "Result from modulo division of $x and $y is: ",$x%$y,"<br/>";
        echo "<hr/>";
        echo "<b>Exponentiation Operation</b><br/>";
        echo "Power of $x to $y is: ", $x**$y;

    ?>
</body>
</html>
```

Output:



localhost/practise/operators/airthmetic.php

Example of Airthmetic Operator

Addition Operation
sum of 50 and 20 is: 70

Subtraction Operation
Difference between 50 and 20 is: 30

Multiplication Operation
multiplicaton of 50 and 20 is: 1000

Division Operation
Result from division of 50 and 20 is: 2.5

Modulo Division Operation
Result from modulo division of 50 and 20 is: 10

Exponentiation Operation
Power of 50 to 20 is: 9.5367431640625E+33

Assignment Operator:

Assignment Operator are used to assign the result of an expression to a variable. The basic assignment operator is “=” which will assign value to the left side operand. Some of the assignment operator are:

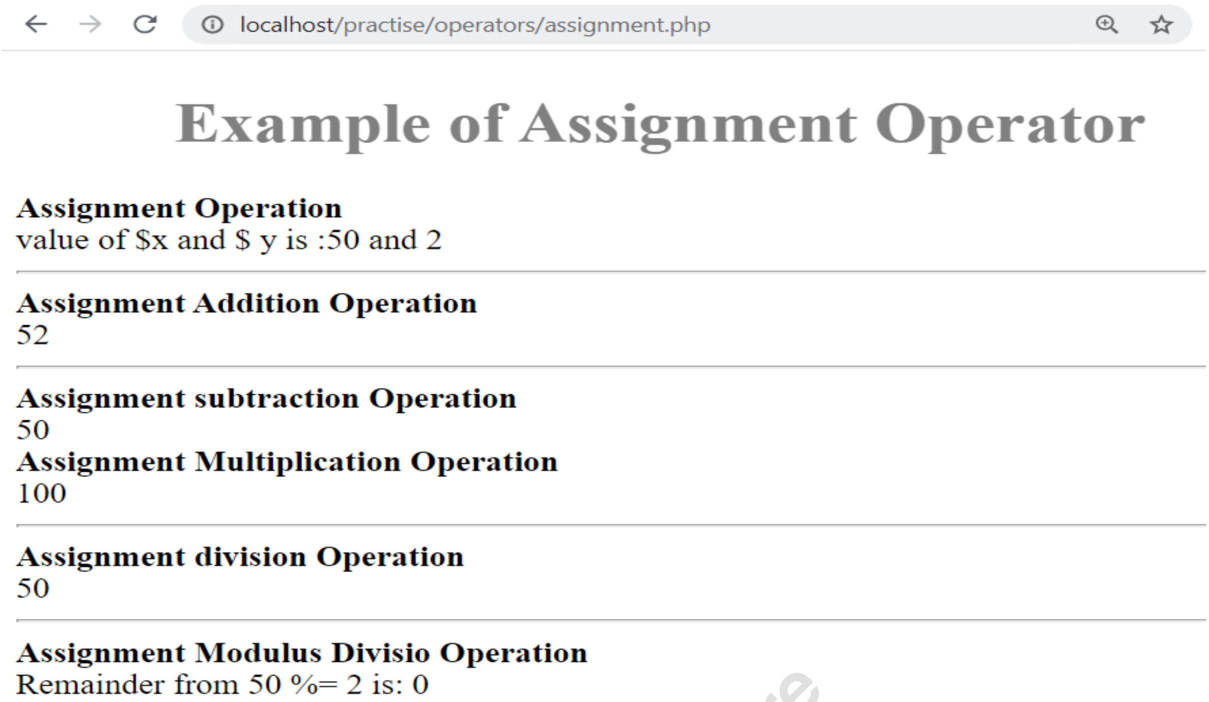
Operator	Example	Description

=	x=y	The x will get the value of y or x will be assigned a value contains by y.
+=	x = x+y	here x will be assigned a value generated from the addition of x and y
-=	x = x - y	here x will be assigned a value generated from the subtraction of x and y
*=	x = x * y	here x will be assigned a value generated from the multiplication of x and y
/=	x = x / y	here x will be assigned a value generated from the result of x/y
%=	x = x % y	here x will be assigned a value generated from the modulo division of x and y

For Example:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title> Example of Assignment Operator
    </title>
</head>
<body>
    <h1 style="color: grey;text-align: center;"> Example of Assignment Operator </h1>
    <?php
        echo "<b>Assignment Operation</b><br/>";
        $x = 50;
        $y = 2;
        echo 'value of $x and $ y is :',$x," ","and"," ",$y, "<br/>";
        echo "<hr/>";
        echo "<b>Assignment Addition Operation</b><br/>";
        echo $x += $y, "<br/>";
        echo "<hr/>";
        echo "<b>Assignment subtraction Operation</b><br/>";
        echo $x -=$y,"<br/>";
        echo "<b>Assignment Multiplication Operation</b><br/>";
        echo $x *= $y, "<br/>";
        echo "<hr/>";
        echo "<b>Assignment division Operation</b><br/>";
        echo $x /=$y, "<br/>";
        echo "<hr/>";
        echo "<b>Assignment Modulus Divisio Operation </b> <br/>";
        echo "Remainder from $x %= $y is: ",$x %= $y, "<br/>";
    ?>
</body>
</html>
```

Output:



Example of Assignment Operator

Assignment Operation
value of $x and $ y is :50 and 2

Assignment Addition Operation
52

Assignment subtraction Operation
50
Assignment Multiplication Operation
100

Assignment division Operation
50

Assignment Modulus Divisio Operation
Remainder from 50 %= 2 is: 0

Comparison or Relational Operator:

Relational operator is used for comparing two values either number or string. The result or output from relational operator are either 1 or 0. If a specified relation is true its value is 1 otherwise 0. For example, 10>5 is true but 5>10 is false. Following are the different relational operators:

Operator	Example	Description
==	$x = = y	This equal operator will return true if value of $x is equal to y
===	$x = = = y	This identical operator will return true if both the type and value of $x is equal to type and value of $y
>	$x > $y	This greater than operator will return true if value of $x is greater than value of $y. Otherwise, false
<	$x < $y	This less than operator will return true if value of $x is less than value of $y.
>=	$x >= $y	This greater or equal to operator will return true if and only if value of $x is greater or equal to value of y otherwise false
<=	$x <= $y	This greater or equal to operator will return true if and only if value of $x is less or equal to value of y otherwise false
< = >	$x < = > $y	This spaceship operator returns zero if value of $x is equal to value of $y, 1 if value of x is greater than y and -1 if value of x is smaller than value of y.
!= or < >	$x != $y or $x < > $y	This not equal operator returns true if $x is not equal to y otherwise false

!==	$x !== y	This not identical operator returns true if both value and type of $x is not equal to value and type of $y

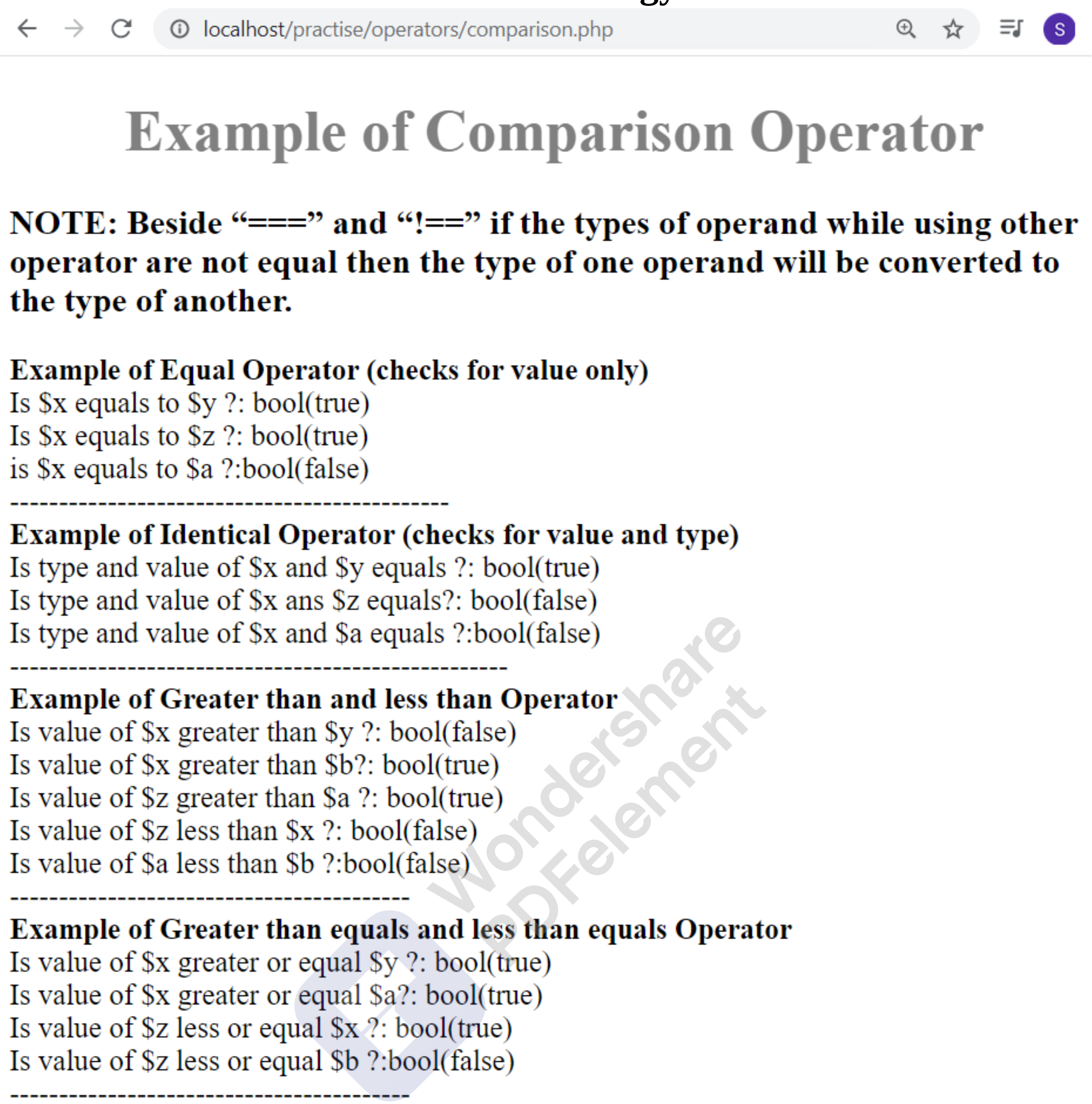
Beside "===" and "!==" if the types of operand while using other operator are not equal then the type of one operand will be converted to the type of another. This means if we have numeric string ("12") in one variable and number in another variable (for eg. 12) then first numeric string is converted to number and numeric comparison is done. If the string cannot be converted to number then number will be converted to string and string comparison is done. If both operands are string that can be converted to number then numeric comparison is done.

So, to avoid such problem (string to number conversion) i.e. either both or any operand are string that could be converted to number then strcmp function should be used for comparison.

For Example:

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <title>Document</title>
</head>
<body>
<h1 style="color: grey;text-align: center;"> Example of Comparison Operator </h1>
<?php
echo "<h3> NOTE:
Beside "===" and "!==" if the types of operand while using other operator are not equal
then the type of one operand will be converted to the type of another. </h3>";
$x = 500;
$y = 500;
$z = "500";
$b = 200;
$a = "200";
$c = 600;
echo "<b> Example of Equal Operator (checks for value only) </b><br/>";
echo 'Is $x equals to $y ?: ', var_dump($x == $y),"<br/>";
echo 'Is $x equals to $z ?: ',var_dump($x==$z),"<br/>"; //string is converted to int
echo 'is $x equals to $a ?:', var_dump($x == $a),"<br/>";
echo "-------------------------------------------- <br/>";
echo "<b> Example of Identical Operator (checks for value and type) </b><br/>";
echo 'Is type and value of $x and $y equals ?: ', var_dump($x === $y),"<br/>";
echo 'Is type and value of $x ans $z equals?: ',var_dump($x === $z),"<br/>"; //value is same
but type different
echo 'Is type and value of $x and $a equals ?:', var_dump($x === $a),"<br/>";
echo "-------------------------------------------------- <br/>";
echo "<b> Example of Greater than and less than Operator </b><br/>";
echo 'Is value of $x greater than $y ?: ', var_dump($x > $y),"<br/>";
echo 'Is value of $x greater than $b?: ',var_dump($x > $b),"<br/>";
echo 'Is value of $z greater than $a ?: ', var_dump($x > $a),"<br/>";
echo 'Is value of $z less than $x ?: ',var_dump($z < $x),"<br/>";
echo 'Is value of $a less than $b ?:', var_dump($a < $b),"<br/>";
echo '----------------------------------------- <br/>';
echo "<b> Example of Greater than equals and less than equals Operator </b><br/>";
echo 'Is value of $x greater or equal $y ?: ', var_dump($x >= $y),"<br/>";
echo 'Is value of $x greater or equal $a?: ',var_dump($x >= $a),"<br/>";
echo 'Is value of $z less or equal $x ?: ',var_dump($z <= $x),"<br/>";
echo 'Is value of $z less or equal $b ?:', var_dump($z <= $b),"<br/>";
echo '----------------------------------------- <br/>';
echo "<b> Example of spaceship Operator (<=>) </b><br/>";
echo 'Value of $x euqals $y so result is : ', $x <=> $y,"<br/>";
echo 'Value of $x is greater than $a so the result is : ',$x <=> $a,"<br/>";
echo 'Value of $x is less than $c so the result is : ',$x <=> $c,"<br/>";
echo '---------------------------------------------<br/>';
echo "<b> Example of not equal Operator (!= or <>) </b><br/>";
echo 'Is value of $x not equal $y ?: ', var_dump($x != $y),"<br/>";
echo 'Is value of $x not equal $a ?: ', var_dump($x != $a),"<br/>";
echo "<b> Example of not identical Operator (!==) </b><br/>";
echo 'Is type and value of $x not equal $y ?: ', var_dump($x !== $y),"<br/>";
echo 'Is type and value of $x not equal $b ?: ', var_dump($x !== $b),"<br/>";
echo 'Is type and value of $z not equal $a ?: ', var_dump($z !== $a),"<br/>";
?>
</body>
</html>
```

Output:



localhost/practise/operators/comparison.php

Example of Comparison Operator

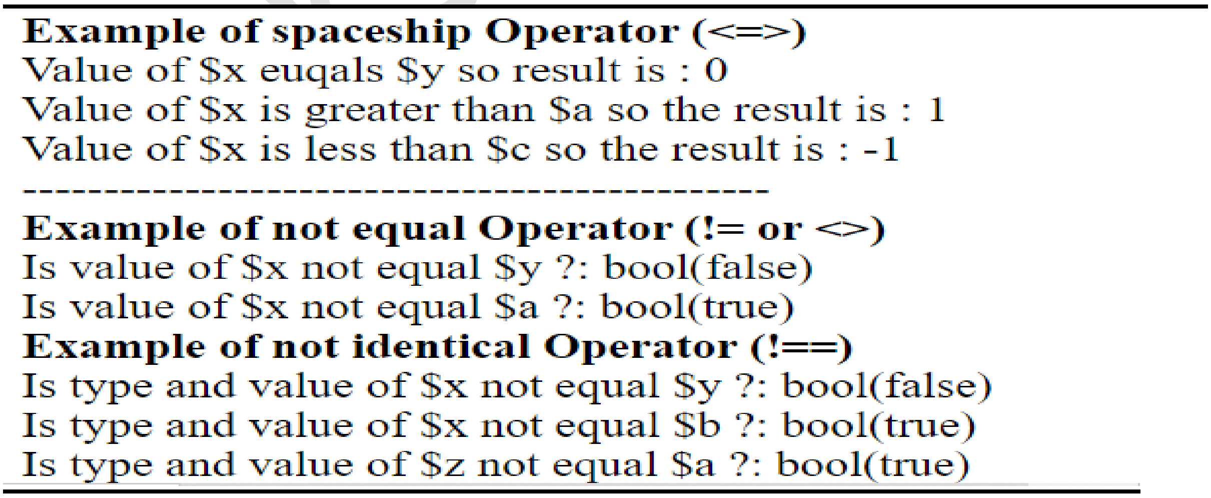
NOTE: Beside "===" and "!==" if the types of operand while using other operator are not equal then the type of one operand will be converted to the type of another.

Example of Equal Operator (checks for value only)
Is $x equals to $y ?: bool(true)
Is $x equals to $z ?: bool(true)
is $x equals to $a ?:bool(false)

Example of Identical Operator (checks for value and type)
Is type and value of $x and $y equals ?: bool(true)
Is type and value of $x ans $z equals?: bool(false)
Is type and value of $x and $a equals ?:bool(false)

Example of Greater than and less than Operator
Is value of $x greater than $y ?: bool(false)
Is value of $x greater than $b?: bool(true)
Is value of $z greater than $a ?: bool(true)
Is value of $z less than $x ?: bool(false)
Is value of $a less than $b ?:bool(false)

Example of Greater than equals and less than equals Operator
Is value of $x greater or equal $y ?: bool(true)
Is value of $x greater or equal $a?: bool(true)
Is value of $z less or equal $x ?: bool(true)
Is value of $z less or equal $b ?:bool(false)



Example of spaceship Operator (<=>)
Value of $x euqals $y so result is : 0
Value of $x is greater than $a so the result is : 1
Value of $x is less than $c so the result is : -1

Example of not equal Operator (!= or <>)
Is value of $x not equal $y ?: bool(false)
Is value of $x not equal $a ?: bool(true)

Example of not identical Operator (!==)
Is type and value of $x not equal $y ?: bool(false)
Is type and value of $x not equal $b ?: bool(true)
Is type and value of $z not equal $a ?: bool(true)

PHP Increment and Decrement Operator:

Increment operator is used for incrementing a value of variable by 1 i.e. adds 1 to operand whereas decrement operator is used for decrementing a value of variable by 1 i.e. subtracts 1 from operand. Increment and decrement operator are unary operators and requires single variable as their operand. Following are the types of increment and decrement operator:

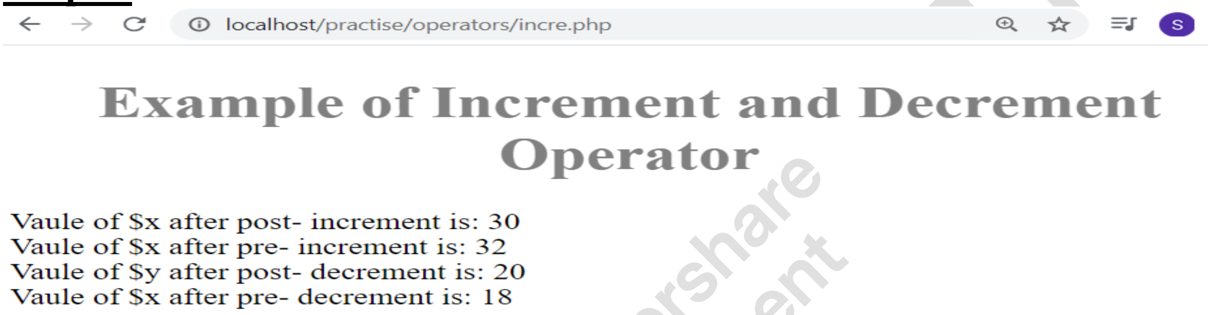
Operator	Description
++$x	This operator is known as pre-increment operator which will first increment the value of $x and then expression is evaluated using new value of variable (return new value of x.
$x++	This operator is known as post-increment operator which will use original value of $x and then value of variable $x is incremented by one.
--$x	This operator is known as pre-decrement operator which will first decrement the value of $x and then expression is evaluated using new value of variable (return new value of x).
$x--	This operator is known as post-decrement operator which will use original value of $x and then value of variable $x is decremented by one.

For Example:

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <title>Example of Increment and Decrement Operator</title>
</head>
<body>
<h1 style="color: grey;text-align: center;"> Example of Increment and Decrement Operator
</h1>
<?php
   $x = 30;
   $y= 20;
   /*
      following code will show value of x to 30 but value has beeen
      increase from 30 to 31.
   */
   echo 'Vaule of $x after post- increment is: ', $x++, "<br/>";
   /*
      following code will show value of x to 32 because value of
      $x becames 31 from first line of code. Now, pre increment will
      increase value of $x from 31 to 32 and assign to $x.
   */
   echo 'Vaule of $x after pre- increment is: ', ++$x, "<br/>";
   /*
```

```
        follwoing code will show value of y to 20 but value has been
        decrease from 20 to 19;
        */
    echo 'Vaule of $y after post- decrement is: ', $y--, "<br/>";
    /*
        following code will show value of y to 18 because value of
        $x becames 19 from first line of code. Now, pre decrement will
        decrease value of $y from 19 to 18 and assign to $y.
    */
    echo 'Vaule of $x after pre- decrement is: ', --$y, "<br/>";
?>
</body>
</html>
```

Output:



localhost/practise/operators/incre.php

Example of Increment and Decrement Operator

```
Vaule of $x after post- increment is: 30
Vaule of $x after pre- increment is: 32
Vaule of $y after post- decrement is: 20
Vaule of $x after pre- decrement is: 18
```

Logical or Boolean Operator:

Logical operator is used for conditional statement which produce true or false based on the condition. Some of the logical operators are:

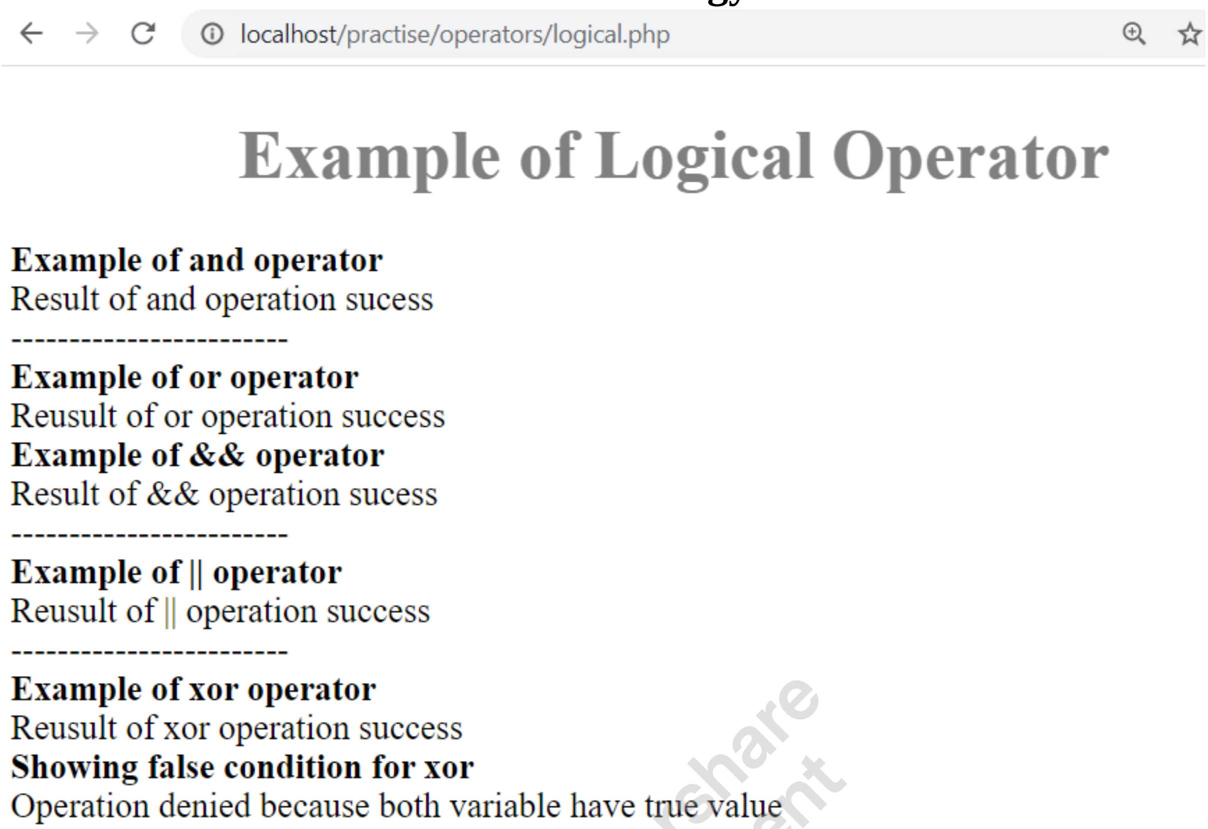
Operator	Example	Description
And	$a and $b	Return true if both $a and $b are true
Or	$a or $b	Return true if either $a or $b is true.
&&	$a $$ $b	Same as "and" operator but this operator has higher precedence (priority) than "and" operator.
||	$a || $b	Same as "or" operator but this operator has higher precedence (priority) than "or" operator
Xor	$a xor $y	Return true if either $x or $y is true but not both
!	!$a	Return true if $a is not true

For Example:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Example of Logical Operator</title>
</head>
<body>
<h1 style="color: grey;text-align: center;"> Example of Logical Operator </h1>
```

```
<?php
   $x=10;
   $y=20;
   echo "<b>Example of and operator</b><br/>";
   if($x==10 and $y==20){
      echo "Result of and operation sucess";
   }
   echo "<br/>";
   echo "------------------------<br/>";
   echo "<b>Example of or operator</b><br/>";
   if ($x==50 or $y==20){
      echo "Reusult of or operation success";
   }
   echo "<br/>";
   echo "<b>Example of && operator</b><br/>";
   if($x==10 && $y==20){
      echo "Result of && operation sucess";
   }
   echo "<br/>";
   echo "------------------------<br/>";
   echo "<b>Example of || operator</b><br/>";
   if ($x==50 || $y==20){
      echo "Reusult of || operation success";
   }
   echo "<br/>------------------------<br/>";
   echo "<b>Example of xor operator</b><br/>";
   //Return true if either $x or $y is true but not both
   if ($x==50 xor $y==20){
      echo "Reusult of xor operation success";
   }
   echo "<br/><b>Showing false condition for xor </b> <br/>";
   if($x==10 xor $y==20){
      echo "Sucess";
   }else{
      echo "Operation denied because both variable have true value";
   }
?>
</body>
</html>
```

Output:



localhost/practise/operators/logical.php

Example of Logical Operator

Example of and operator
Result of and operation sucess

Example of or operator
Reusult of or operation success
Example of && operator
Result of && operation sucess

Example of || operator
Reusult of || operation success

Example of xor operator
Reusult of xor operation success
Showing false condition for xor
Operation denied because both variable have true value

PHP Ternary Operator

Ternary operator is used to set value of variable based on condition and it shorten if/else condition structure. It has form **expression 1 ? expression 2 : expression 3. For example:**

$a = exp1? exp2: exp3

First exp1 is evaluated and if it is non zero (true) then the exp2 will is evaluated and value of exp2 will be assigned to $a. if exp1 is false then exp3 will be evaluated and value of exp3 will be assigned to $a.

PHP Null Coalescing Operator:

This operator is also used to set the value of variable based on condition but it will test for not null value i.e. return the not null value. It takes the form expr1??expr2.

For Example:

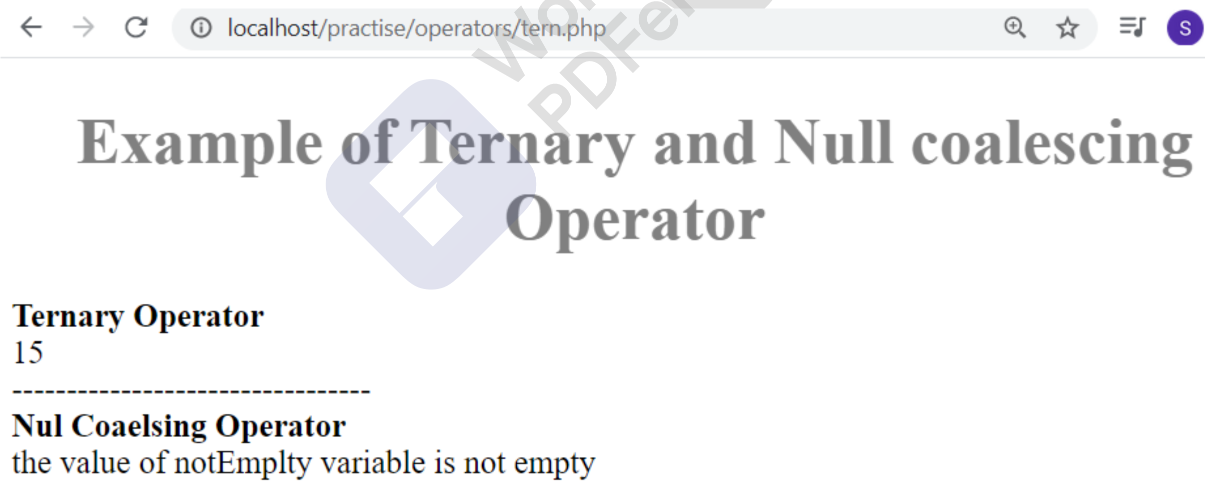
$a = exp1 ?? exp2

First exp 1 is evaluated and if its value is not null then value of exp1 is assigned to $a and if exp1 has null value then value of exp2 is assigned. It decides which value of expression to use based on the value of the left-hand operand.

Example on Ternary and Null Coalescing Operator

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Ternary and Nul Coaelsing</title>
</head>
<body>
<h1 style="color: grey;text-align: center;"> Example of Ternary and  Null coalescing Operator</h1>
<?php
echo "<b>Ternary Operator</b><br/>";
$x = 12;
$y = 15;
//first $x>$y evaluated if true $x will return otherwise $y;
$result = $x>$y ? $x:$y;
echo $result;
echo "<br/>";
echo "--------------------------------- <br/>";
echo "<b>Nul Coaelsing Operator</b><br/>";
$nonEmpty = "the value of notEmplty variable is not empty";
$empty;
//here first $nonEmpty is checked if it is not null then value is return
$resultNext = $nonEmpty ?? $empty;
echo $resultNext;
?>
</body>
</html>
```

Output:



Example of Ternary and Null coalescing Operator

```
Ternary Operator
15
---------------------------------
Nul Coaelsing Operator
the value of notEmplty variable is not empty
```

PHP String Operators:

This operator is used for string operation. Following are the types of string operator.

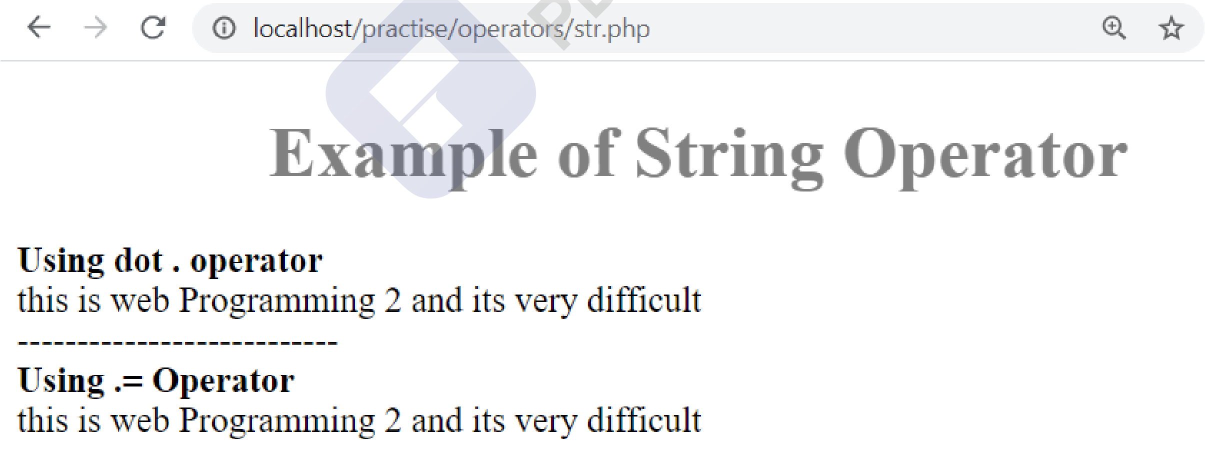
Operator	Example	Description
•	$str1.$str2	Use for concatenation (joining) two string values

.=	$str1 .= $str2	This is concatenation assignment operator which will appends $str2 to $str 1. Same like $str1 = $str1 . $str2

For Example:

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <title>Document</title>
</head>
<body>
<h1 style="color: grey;text-align: center;"> Example of String Operator</h1>
<?php
   $str1 = "this is web Programming 2";
   $str2 = " and its very difficult";
   echo "<b>Using dot . operator</b> <br/>";
   echo $str1.$str2."<br/>";
   echo "---------------------------<br/>";
   echo "<b>Using .= Operator</b><br/>";
   echo $str1 .=$str2;
?>
</body>
</html>
```

Output:



Example of String Operator

Using dot . operator
this is web Programming 2 and its very difficult

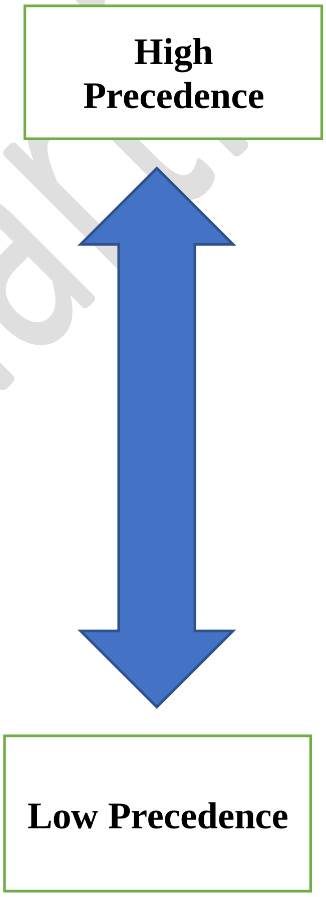
Using .= Operator
this is web Programming 2 and its very difficult

Operator Precedence and Associativity:

Precedence of operator specify which expression or which operator to execute first while evaluating any expression i.e. which operator to give highest priority than other. Associativity determines in which direction the specific operator need to be evaluated if the operator have same precedence. It decides which operator to evaluated first when they have same precedence. Operator with equal precedence and are non-associative cannot be used next to each other. For example 1<3<4.

PHP have following operator precedence and associativity:

Operator	Associativity
()	Left to Right
Clone new	Non associativity
**	Right to Left
++, -- , unary - datatype	Right to Left
Instance of	Non associative
!	Right to Left
*/ %	Left to Right
+ - .	Left to Right
<<>>	Left to Right
< <= > >=	Non associative
==, !=, ===, !==, <>, < =, >	Non associative
&	Left to Right
^	Left to Right
|	Left to Right
&&	Left to Right
||	Left to Right
?:	Left to Right
= += -= *= **= /= %= .= &&= ||= <<= >>=	Right to Left
And	Left to Right
Xor	Left to Right
Or	Left to Right



Example on Operator Precedence:

Consider following expression:

$cal = 5+4*6/3+8-3;

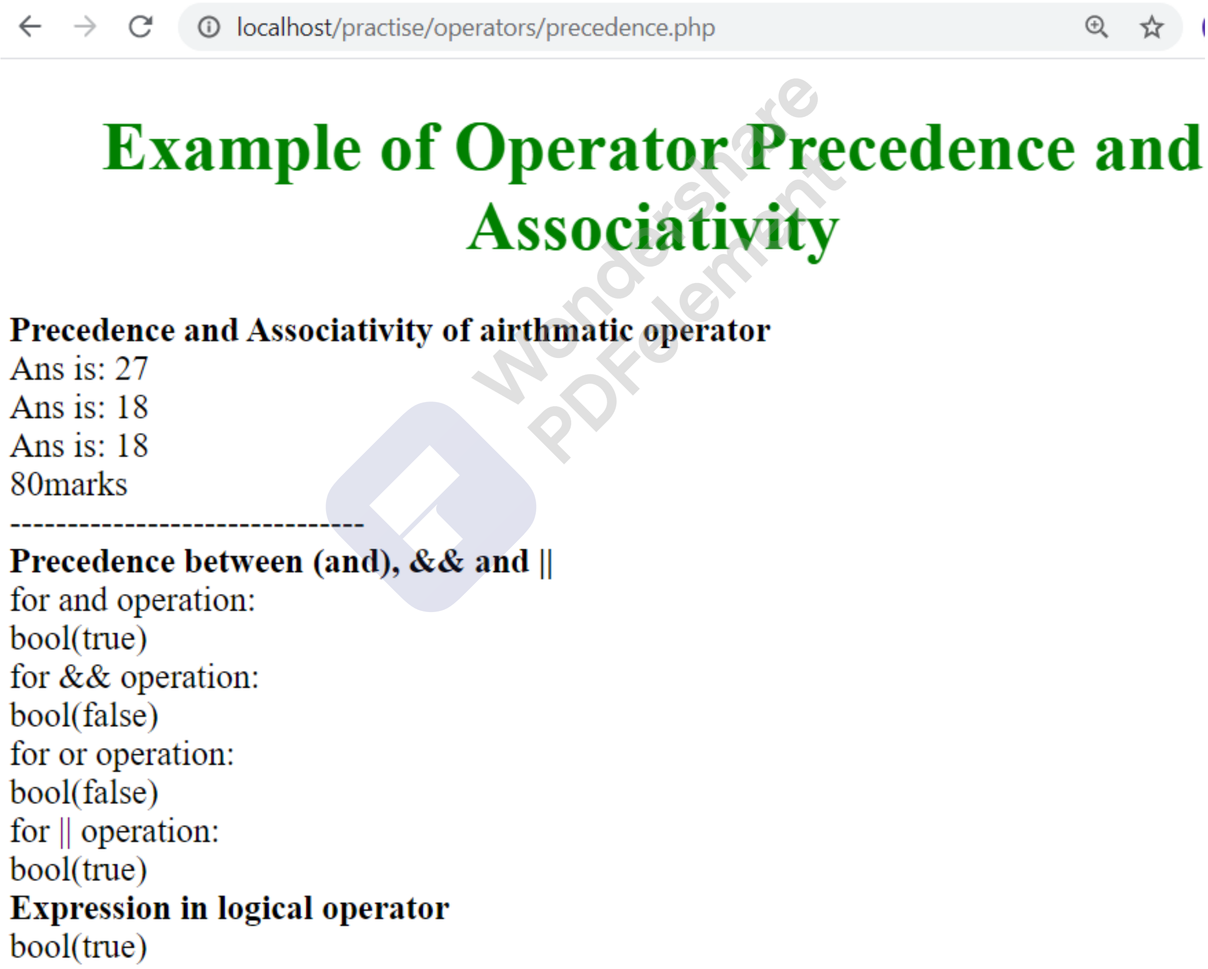
First, +, *, / and – have more precedence over = so first expression will be evaluated. On expression, * and / has more precedence over + and – so either * or / will be evaluated first than + and -. As the precedence of / and * are same its associativity should be considered. Associativity of / and * is from left so in left side there is * in our expression so first 4*6 will be evaluated then the result from it will be divide from 3 then added with 5 and 8 and finally the result will be subtracted from 3 and assigned to variable cal.

$cal = 5+ 24/3+8-3; //multiply is evaluated first as associative is left to right
$cal = 5+8+8-3 // divide is evaluated as it has more precedence than + and –
$cal = 13+8-3 // 5+8 is evaluated as associative is from left to right
$cal = 21-3; //13 + 8 is evaluates as associative is from left to right
$cal = 18 // subtract is evaluated at last.

Consider following example for precedence and associativity of arithmetic, string and logical operation:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Operator precedence and associativity example</title>
</head>
<body>
<h1 style="color: green;text-align: center;"> Example of Operator Precedence and Associativity</h1>
<?php
    echo "<b>Precedence and Associativity of airthmatic operator</b><br/>";
    //here * has more precedence than + so * will evaluated first
    $first = 2 + 5*5;
    echo "Ans is: ",$first,"<br/>";
    // for folowing second statement, first * is evaluted and then /
    //because associativity is left to right
    $second = 5+4*6/3+8-3;
    echo "Ans is: ", $second, "<br/>";
    //for following: statement inside ( ) is evaluated
    $third = 5+4*(6/3)+8-3;
    echo "Ans is: ", $second, "<br/>";
    //$fourth = "marks: ".30 + 20;
    $fifth = 40 + 40 ."marks";
    echo $fifth;
    echo "<br/>--------------------------------<br/>";
    echo "<b>Precedence between (and), && and ||</b><br/>";
    echo "for and operation: <br/>";
    $checkAnd = true and false;
    //this will produce true because = have high precedence over and
    // this conditon is like (($checkAnd=true) and false);
    var_dump($checkAnd);
    echo "<br/> for && operation: <br/>";
    $checkAnd2 = true && false;
    //this will produce false as && have higher precedence over =
    var_dump($checkAnd2);
    echo "<br/>for or operation: <br/>";
    $checkOr = false or true;
    //this will produce false because = have high precedence over or
    var_dump($checkOr);
    echo "<br/> for || operation: <br/>";
    $checkOr2 = false || true;
    //this will produce false as && have higher precedence over =
    var_dump($checkOr2);
    echo "<br/><b>Expression in logical operator</b><br/>";
    $expr = true and false && true || false;
    //here first && operator will evalueated then || then = will be evaluated
    var_dump ($expr) ;
?>
</body>
</html>
```

Output:



localhost/practise/operators/precedence.php

Example of Operator Precedence and Associativity

Precedence and Associativity of airthmatic operator
Ans is: 27
Ans is: 18
Ans is: 18
80marks

Precedence between (and), && and ||
for and operation:
bool(true)
for && operation:
bool(false)
for or operation:
bool(false)
for || operation:
bool(true)
Expression in logical operator
bool(true)

Conditional Statement:

Conditional statement is used to perform certain action based on certain condition and controls the flow of execution. The conditional statement can be either a single statement or a compound statement. Following are the types of conditional statement

The if Statement:

It is basically a two-way decision-making statement and used with conjunction with an expression in which some action is performed if one condition is true. It takes the form: **if (test expression) {code to execute if condition is true};**

First the expression inside small bracket () is evaluated and depending on whether the value of expression (relation or codition) is 'true' or 'false', it transfer the control to a particular statement which is placed inside middle bracket { }. If the condition is false then control is transfer outside of curly bracket. If there is only single statement to execute then middle braces can be ignore.

For example:

```
<?php

        $marks = 50;
        If ($marks <= 60){
        Echo "your passed in second division";
        };
?>
```

The if … else statement:

The if … else statement is an extension of the simple if statement in which code inside middle braces of if part will be executed if condition is true and if false then code inside else will be executed. It takes the form:

```
If (test condition){
        True block statement or code to be executed if condition is true;
} else{
        False block statement or code to be executed if condition is false;
}
```

In either case, either true block statement or false block statement will be executed but not both at a same time.

```
<?php
        $marks = 30;
        if ($marks >= 40){
        echo "your passed";
        }else {
        echo "  you are fail";
        }
?>
```

The if .. elseif.. else statement:

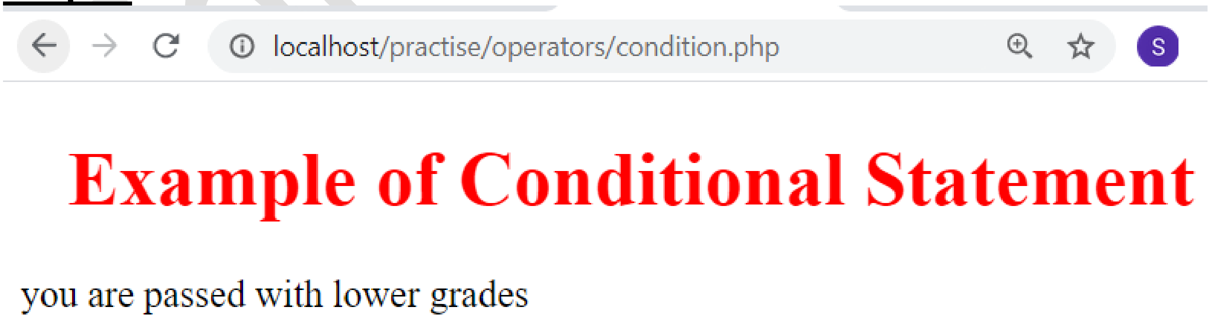
It is a chain of if in which the statement associated with each else is an if and executes different codes based on the condition. If the condition of if statement is false then the control is passed to else if and if the condition of if and all else if is false then control will be passed to else statement and code of else is executed. It takes the form:

```
if (test condition) {
Code to be executed if condition is true;
}
elseif (test condition){
Code to be executed if the condition of if statement is false and condition of elseif is true;
}
else{
Code to be executed if the condition of both if and else if statement is false;
}
```

For example:

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <title>Example of condtional statement</title>
</head>
<body>
<h1 style="color: red;text-align: center;"> Example of Conditional Statement</h1>
<?php
   $marks = 50;
   if ($marks >60){
      echo "you are passed with higher grades";
   }elseif ($marks >30 && $marks <=60){
      echo "you are passed with lower grades";
   }else{
      echo "you are failed";
   }
?>
</body>
</html>
```

Output:



Example of Conditional Statement

you are passed with lower grades

The Switch Statement:

The switch statement is used when one variable or the result of expression can have multiple value and each of which should be trigger a different activity. The switch statement tests the value of a given variable or expression against the list of case values and when a match is found a block of statements associated with that case is executed. It takes the form:

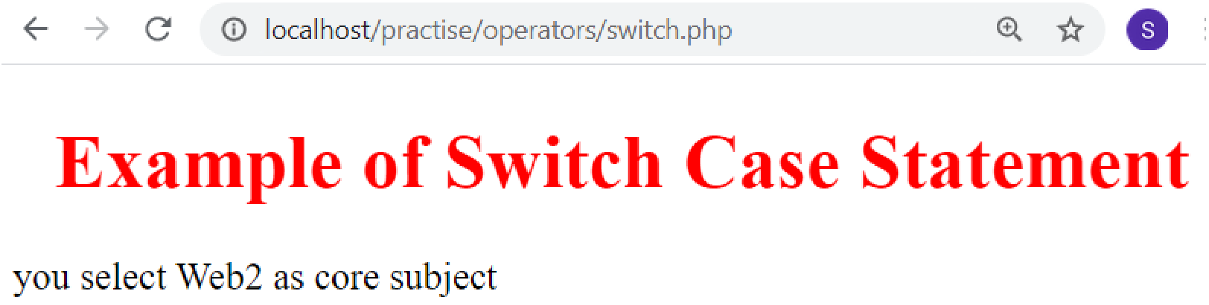
```
switch (expression)
{
        case value-1:
                Block-1 i.e. code to be executed if expression = value-1
                Break;
        case value-2:
                Block-2 i.e. code to be executed if expression  = value-2
                Break;
……..
………
        default:
Default block i.e. code to be executed if expression is different from all cases
        break;
}
```

Here, when switch is executed, the value of the expression is compared against the case value-1, value-2 and so on. If a value of expression matched with value of any case then the block of statement that follows the matched case are executed. The break statement signals the end of particular case and transfer the control out of the switch block. The default statement is executed if the value of expression does not match with any of the case values.

For example:

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Switch case example</title>
</head>
<body>
<h1 style="color: red;text-align: center;"> Example of Switch Case Statement</h1>
<?php

  $selectedCourse = "Web2";

  switch($selectedCourse){

    case "java":
      echo "you select java as core subject";
    break;
    case "C":
      echo "you select C program as core subject";
    break;
    case "Web2":
      echo "you select Web2 as core subject";
    break;
    default:
    echo "You have select other option rather than java, C and Web2";
    }
?>
</body>
</html>
```

Output:



localhost/practise/operators/switch.php

Example of Switch Case Statement

you select Web2 as core subject

Loops:

Loops are used if we need to execute same block of code for specified number of times. A sequence of statement is executed until some condition for termination of loop are satisfied. Therefore, loop consist of two segments, one is body of the loop and other one is control statement which test the condition and directs the repeated execution of statement contained in the body of loop. In looping process, following steps are include:

- Setting and initialization of a condition variable
- Execution of the statement in the loop
- Test for a specified value of the condition variable for execution of the loop.
- Incrementing or updating the condition variable.

PHP consist of following loop:

while loop:

While loop is used to execute a block of code as long as test condition is true. It takes the form:

```
While (test-condition){
        Body of loop
}
```

First test-condition is evaluated and if the condition is true then body of the loop is executed. After execution, once again test condition is evaluated and if it is true again the body of loop is executed. The same process is repeated until the test-condition becomes false and control is transfer out of the loop.

do while loop:

Do while loop will execute a block of code or body of loop first then only test condition is evaluated i.e. it will execute a block of code at least one time. It takes the form:

```
do {
Body of loop
}while (test-condition);
```

Here, on reaching the do statement, first the body of loop is evaluated. Then, the test condition is evaluated. If the condition is true, the body of loop will be evaluated once again. Same process will be continuing until the condition becomes false. After condition is false, the control goes out of the loop.

for loop:

for loop combines the abilities to set up variables as loop is enter, test for condition while iterating loops and modify variables after each iteration. For loop is used if we know how many times the script should run. It takes the form:

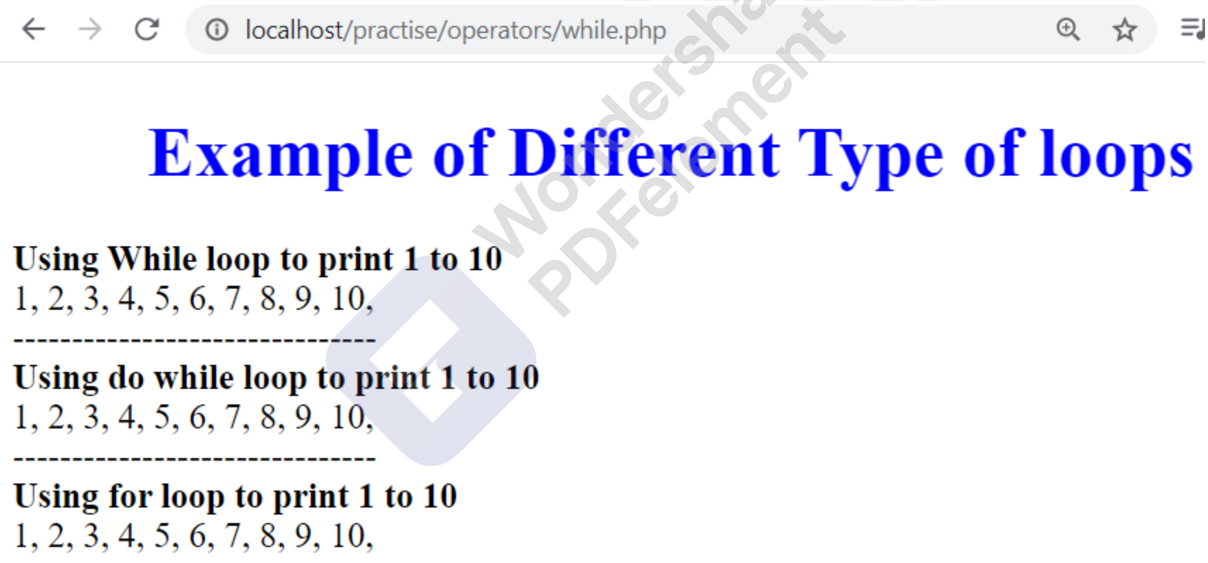
```
for (initialization; test-condition; increment counter) {
body of the loop;
}
```

First initialization of the control variable is done using assignment operator like i = 1. After this, value of control variable is checked with test condition. If the condition is true, the body of the loop will be executed otherwise control will be transfer out of the loop. After evaluating last statement of the body of loop the control is transfer back to for statement where the value of control variable will be increment or decrement by using assignment or increment, decrement operator. For example, i=i+1 or i++.

Example of while, do .. while and for loop:

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <title>Example of loops</title>
</head>
<body>
<h1 style="color: blue;text-align: center;"> Example of Different Type of loops</h1>
<?php
   echo "<b>Using While loop to print 1 to 10</b><br/>";
   $x = 1;
   while ($x<=10){
       echo $x,",","  ";
       $x++;
   }
   echo "<br/> ------------------------------- <br/>";
   echo "<b>Using do while loop to print 1 to 10</b><br/>";
   $y=1;
   do{
       echo $y,",","  ";
       $y++;
   }while($y<=10);
   echo "<br/> ------------------------------- <br/>";
   echo "<b>Using for loop to print 1 to 10</b><br/>";

   for ($i = 1; $i<=10; $i++){
       echo $i,",","  ";
   }
?>
</body>
</html>
```

Output:



```
Example of Different Type of loops

Using While loop to print 1 to 10
1, 2, 3, 4, 5, 6, 7, 8, 9, 10,
-------------------------------
Using do while loop to print 1 to 10
1, 2, 3, 4, 5, 6, 7, 8, 9, 10,
-------------------------------
Using for loop to print 1 to 10
1, 2, 3, 4, 5, 6, 7, 8, 9, 10,
```

for each loop:

For each loop is used for iterating over arrays through each key value pair. For each loop can only be used with array and object. It takes the form:

```
foreach ($array as $value){
Body of loop;
}
Or
foreach ($array as $key=>$value) {
Body of loop;
}
```

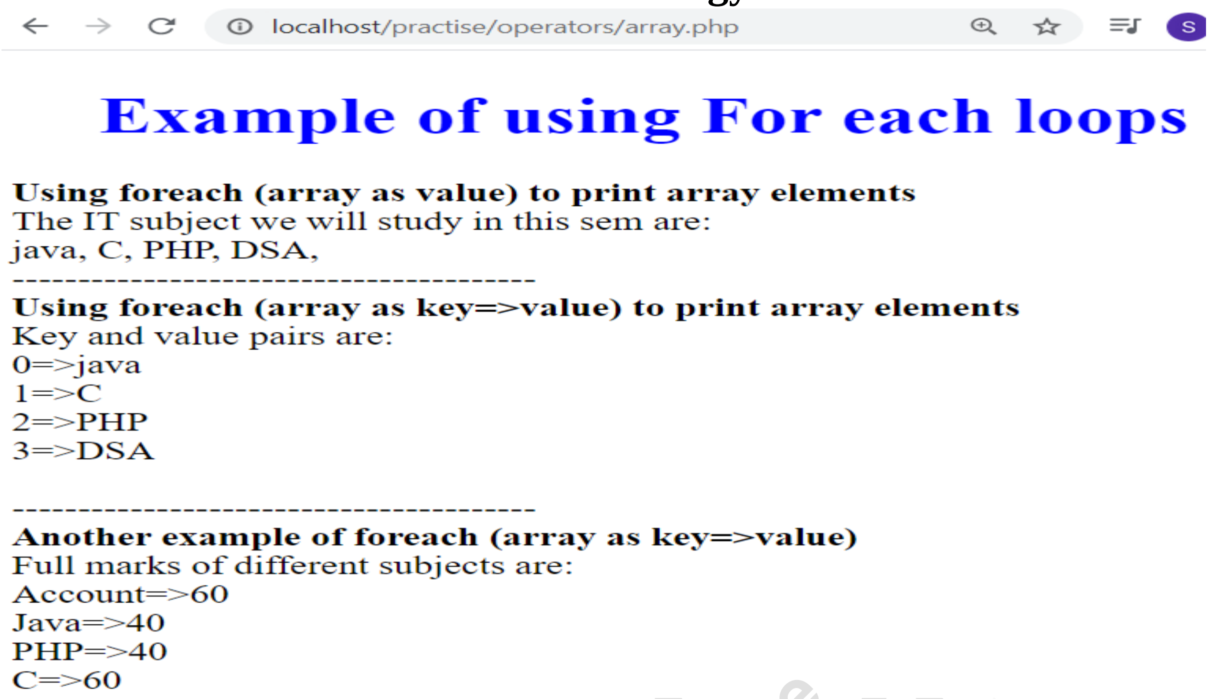
The first form loops over the array given by array_expression in which the value of current array element is assigned to $value and the array pointer is increased by one until it reaches the last array element.

The second form will additionally assign current key of element to $key on each iteration. This is used to output both keys and values of the array.

For example:

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <title>Example of for each loop</title>
</head>
<body>
<h1 style="color: blue;text-align: center;"> Example of using For each loops</h1>
<?php
   $course = array("java","C","PHP","DSA");
   echo '<b>Using foreach (array as value) to print array elements</b> <br/>';
   echo "The IT subject we will study in this sem are: <br/>";
   foreach($course as $value){
      echo $value,","," ";
   }
   echo "<br/>----------------------------------------- <br/>";
   echo "<b> Using foreach (array as key=>value) to print array elements </b> <br/>";
   echo "Key and value pairs are: <br/>";
   foreach($course as $key=>$value){
      echo "{$key}","=>","$value","<br/>";
   }
   echo "<br/>----------------------------------------- <br/>";
   echo "<b> Another example of foreach (array as key=>value)</b> <br/>";
   $marks = array("Account"=>"60","Java"=>"40","PHP"=>"40","C"=>"60");
   echo "Full marks of different subjects are: <br/>";
   foreach ($marks as $key=>$value){
      echo "{$key}","=>","$value","<br/>";
   }
?>
</body>
</html>
```

Output:



localhost/practise/operators/array.php

Example of using For each loops

```
Using foreach (array as value) to print array elements
The IT subject we will study in this sem are:
java, C, PHP, DSA,
----------------------------------------
Using foreach (array as key=>value) to print array elements
Key and value pairs are:
0=>java
1=>C
2=>PHP
3=>DSA

----------------------------------------
Another example of foreach (array as key=>value)
Full marks of different subjects are:
Account=>60
Java=>40
PHP=>40
C=>60
```

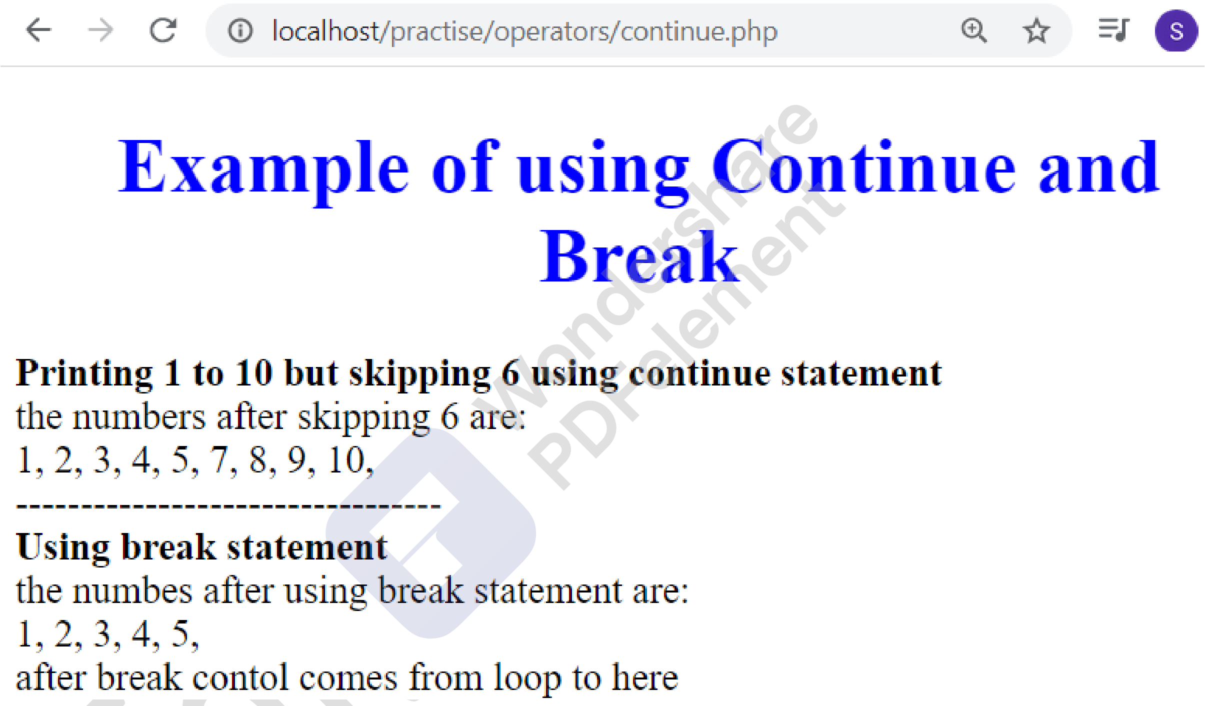
Break and continue:

During the loop operation, it may be necessary to skip a part of the body of the loop under certain condition. **Break** statement is used to jump out of loops i.e. it terminates whole iteration of a loop. **Continue** statement causes loop to be continued with next iteration after skipping any statements in between i.e. it skips rest of the body of loop and control goes to loop statement. The continue statement break only one iteration if condition occurs whereas break skips whole iteration.

Following example shows uses of break and continue statement:

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <title>Example on continue and break</title>
</head>
<body>
<h1 style="color: blue;text-align: center;"> Example of using Continue and Break</h1>
<?php
   echo "<b>Printing 1 to 10 but skipping 6 using continue statement</b><br/>";
   echo"the numbers after skipping 6 are:<br/>";
   for($i = 1; $i<=10; $i++){
      if ($i==6){
         continue;
      }
      echo $i,",","," ";
   }
   echo "<br/>--------------------------------<br/>";
   echo "<b>Using break statement</b><br/>";
   //the loop will be terminate if i==6 occurs
   echo "the numbes after using break statement are: <br/>";
   for($i = 1; $i<=10; $i++){
       if ($i==6){
           break;
       }
       echo $i,","," ";
   }
echo "<br/>";
   echo "after break contol comes from loop to here";
?>
</body>
</html>
```

Output:



localhost/practise/operators/continue.php

Example of using Continue and Break

Printing 1 to 10 but skipping 6 using continue statement

```
the numbers after skipping 6 are:
1, 2, 3, 4, 5, 7, 8, 9, 10,
---------------------------------
```

Using break statement

```
the numbes after using break statement are:
1, 2, 3, 4, 5,
after break contol comes from loop to here
```

Functions

We have already used some built in function of PHP like string handling function for eg. Strtolower(), strtoupper() etc., array manipulation function like push(), pop(), count() etc. Beside these inbuilt functions we can create our own function known as user defined function. The main distinction between user defined and in-built function is that the inbuilt function is not required to be written by us whereas user defined function has to be developed by the user at the time of writing a program.

A **function** is a block of statements that can be used repeatedly in a program or it is a self-contained block of code that performs a particular task. It is useful when we have to write similar code each time. For example: we might have to perform algebraic operation for same variable multiple times or we might have to perform factorial of some number multiple times thought the program. In such situation we can create a function instead of writing code for multiple times. For a function to execute it needs to be called.

To use a user defined function, we need to established following things:

- Function definition which is an independent program that is specially written to implement the requirements of the function.
- Function call: to invoke the function
- The calling function should declare any function that is to be used later in the program known as **function declaration**

Syntax:

In PHP function is created by using function keyword followed by function name, parameter (optional) and block of code inside curly braces:

```
function function_name (Parameter (optional))
      {
         //code to execute
      }
```

For example:

```
Function displayMsg ()
    {
       Echo "Hello, csit 5th Semester";
    }
```

//calling function to execute: function should always be called by function name
displayMsg();

Rules:

- The naming convention for function name is same as variable naming convention rule i.e., name of function must start with letters or underscore followed by any number of letters, numbers or underscores.
- Parameter can be one or more which should be separated by comma and is optional. We declare the variable in parameter which will receive some data from calling function or program. Parameter serve as an input data to function to carry out the specified task.
- Function names are case insensitive.
- For a function to works it needs to be called by its name.

Functions have following advantages over contiguous code:

- Involve less typing and reduce syntax
- Decrease loading time of program files
- Decrease execution time as each function is compiled only once

Returning A Value:

A function might send a value back to calling function. If it does, it sends value through **return** statement. It is possible to pass multiple value to a function but the called function can only return one value at a time. To capture a return value, a variable should be assigned to calling function.

Syntax:

return (expression);

when a return is encounter, the control is passed back to the calling function.

For example:

```
function add()
  {
        $sum = 5+10
        Return $sum;
  }
```

//calling a function. Here a variable result is assigned to calling function add() which will capture return value.

```
$result = add();
echo “the sum is: $result”;
```

A function may have more than one return statement. This situation arises when the value returned is based on certain condition.

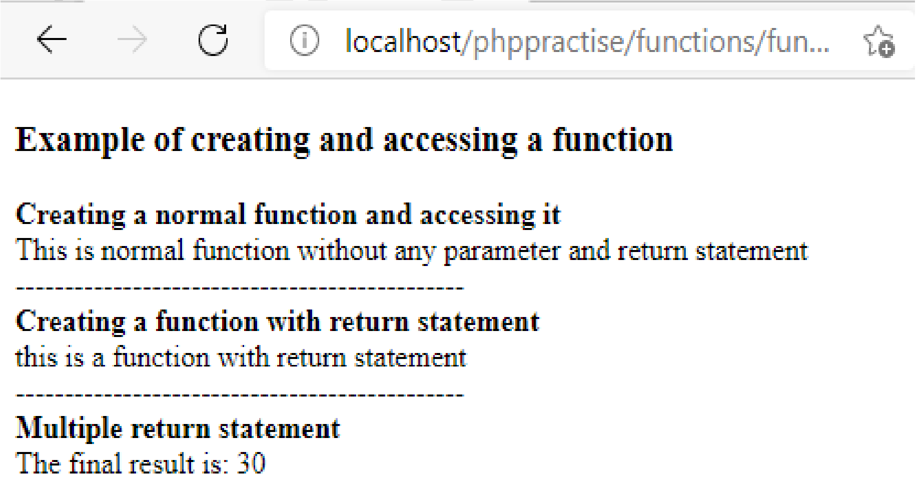
For example:

```
If($X>$y) {
return “$x is greater than y”;
}
else {
return “$y is greater than $x”;
}
```

Example on creating a function and return statement:

```
<!DOCTYPE html>
<html lang=”en”>
<head>
   <meta charset=”UTF-8”>
   <meta http-equiv=”X-UA-Compatible” content=”IE=edge”>
   <meta name=”viewport” content=”width=device-width, initial-scale=1.0”>
   <title>Function Example</title>
</head>
<body>
   <h3>Example of creating and accessing a function </h3>
   <?php
   echo “<b>Creating a normal function and accessing it</b><br/>”;
     function simpleFunction(){
       echo “This is normal function without any parameter and return statement <br/>”;

     }
     //calling a function
     simpleFunction();
     echo “----------------------------------------------<br/>”;
     echo “<b>Creating a function with return statement </b><br/>”;
     function returnMsg(){
       return “this is a function with return statement”;
       //only single value can be returned by return statement
     }
     $result = returnMsg();
     echo $result,”<br/>”;
     echo “----------------------------------------------<br/>”;
     echo “<b>Multiple return statement </b><br/>”;
     //we can use multiple return statement by using conditional statement
     function multipleReturn(){
       $a = 10;
       $b = 20;
       if($a<=$b){
         $sum = $a+$b;
         return $sum;
       } else {
         $diff = $a - $b;
         return $diff;
       }
     }
     $results = multipleReturn();
     echo “The final result is: $results”;
     ?>
</body>
</html>
```

Output:



```
Example of creating and accessing a function

Creating a normal function and accessing it
This is normal function without any parameter and return statement
----------------------------------------------
Creating a function with return statement
this is a function with return statement
----------------------------------------------
Multiple return statement
The final result is: 30
```

Passing Argument to Functions:

Argument or parameter is an input values supplied to a function. Argument makes the function more flexible. The input value is supplied from calling function to called function. Parameter can be single or multiple. Comma should be used to separate multiple parameters.

For example:

```
//single parameter
function dispalyMsg($msg)
     {
        echo $msg;
     }

//calling a function
displayMsg (“Hello csit 5th Sem”);
```

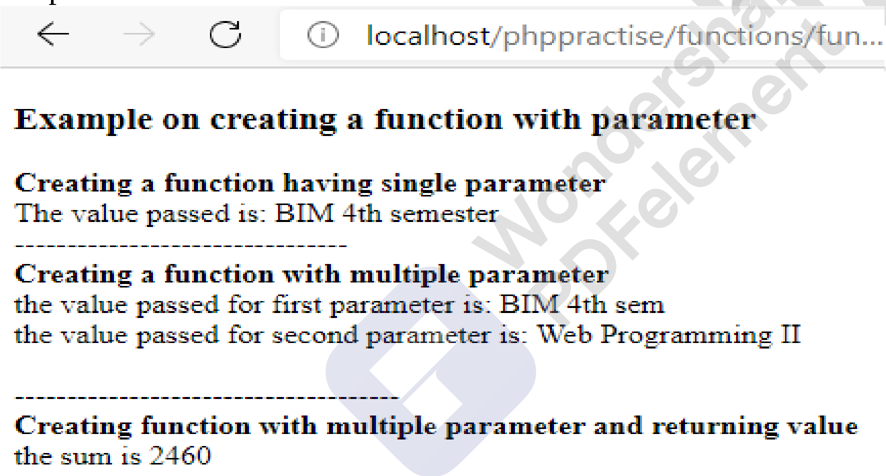
If a function contains a parameter then appropriate value for such parameter should be provided from a calling function. In above example there is only one parameter so only one value is supplied that is “Hello BIM 4th Sem”. If a function contains multiple parameters then same number of multiple values should be provided i.e., if function contains two parameters then values should be provided for both parameters.

Example to create function with one and multiple parameter:

```
<!DOCTYPE html>
<html lang=”en”>
<head>
   <meta charset=”UTF-8”>
   <meta http-equiv=”X-UA-Compatible” content=”IE=edge”>
   <meta name=”viewport” content=”width=device-width, initial-scale=1.0”>
   <title>Example of 45arameterized function</title>
</head>
<body>
   <h3>Example on creating a function with parameter</h3>
   <?php
      echo “<b>Creating a function having single parameter</b><br/>”;
      function displayMsg($message) {
         echo “The value passed is: $message”;
      }
      //calling function
      //displayMsg(); (produce error)
      displayMsg(“CSit 5th semester”);
      echo “<br/>--------------------------------<br/>”;
      echo “<b>Creating a function with multiple parameter</b><br/>”;
      function displayNewMsg($msg1,$msg2){
         echo “the value passed for first parameter is: $msg1 <br/>”;
         echo “the value passed for second parameter is: $msg2 <br/>”;
      }
      //displayNewMsg(); (produce error)
      //displayNewMsg(“hello”); (produce error)
      displayNewMsg(“BIM 4th sem”,”Web Programming II”);
      echo “<br/>-------------------------------------<br/>”;
      echo “<b>Creating function with multiple parameters and returning value</b><br/>”;
```

```
        //finding sum of even number between two parameters
        function calculateSum($input1, $input2){
           $sum = 0;
         if($input1<=$input2) {
            for ($i=$input1; $i<=$input2; $i++) {
               if($i%2==0) {
                  $sum+=$i;
               }
            }
            return “the sum is $sum”;
         } else{
            return “first parameter should be less than second parameter”;
         }
        }
        $res = calculateSum(20,100);
        echo $res;
        ?>
</body>
</html>
```

Output:



```
Example on creating a function with parameter

Creating a function having single parameter
The value passed is: BIM 4th semester
--------------------------------
Creating a function with multiple parameter
the value passed for first parameter is: BIM 4th sem
the value passed for second parameter is: Web Programming II

-------------------------------------
Creating function with multiple parameter and returning value
the sum is 2460
```

Array:

An array is a variable that can hold more than one value at a time. It is useful to aggregate a series of related items together. Each data value in array are referenced to unique index or key. There are two ways to create an array in PHP. The first way is to assign a value in a sub scripted variable i.e. to assigned a value into indexed variable. For example:

$subject [0] = “php”;

If empty brackets are used in assignment to an array then a numeric key or index is implicitly converted to one time greater than the largest numeric key used. For example:

$subject [1] = “java”;
$subject[ ] = “javascript”; //the index or key will be 2

The second way to create an array is to use array contructor. The parameter in a array specify the value to be placed in a new array and sometimes keys.

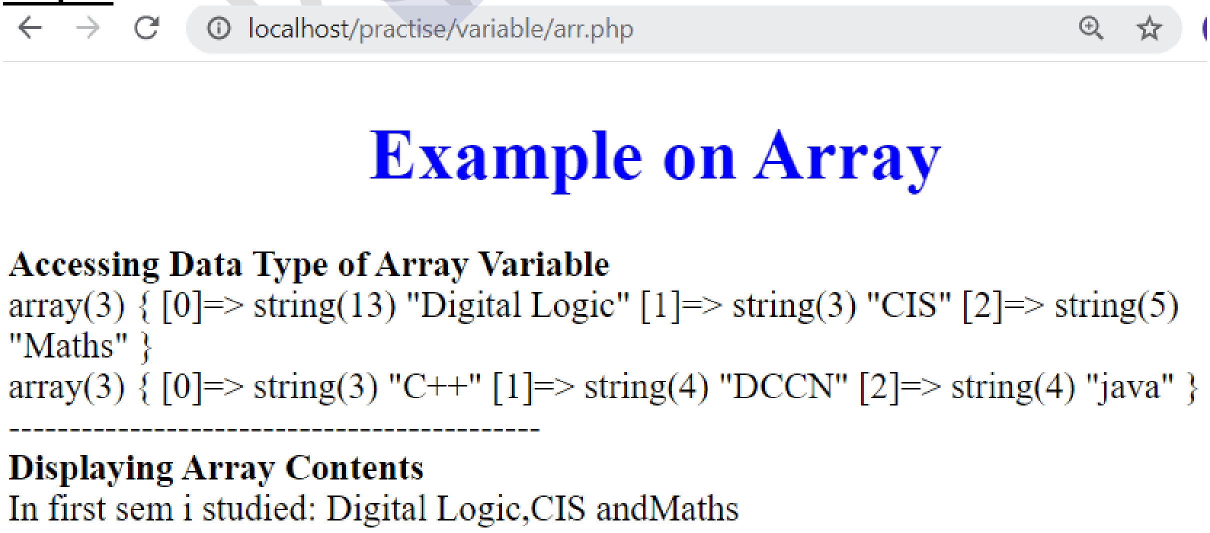
Syntax:

$subject = array (“java”, “php”, “javascript”);

Following examples shows how array are created and displayed:

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <title>Array </title>
</head>
<body>
<h1 style="color: blue;text-align: center;"> Example on Array  </h1>
   <?php
   //Creating array using array consturctor;
   $firsSem = array("Digital Logic","CIS","Maths");
   //Creating array using assignment operator";
   $secSem[0] = "C++";
   $secSem[1] = "DCCN";
   $secSem[] = "java";
   echo"<b>Accessing Data Type of Array Variable</b></br>";
   var_dump($firsSem);
   echo "</br>";
   var_dump($secSem);
   echo"</br>";
   echo "--------------------------------------------</br>";
   echo "<b>Displaying Array Contents</b></br>";
   echo "In first sem i studied: ".$firsSem[0].",".$firsSem[1]." "."and".$firsSem[2];
   ?>
</body>
</html>
```

Output:



Example on Array

Accessing Data Type of Array Variable

```
array(3) { [0]=> string(13) "Digital Logic" [1]=> string(3) "CIS" [2]=> string(5)
"Maths" }
array(3) { [0]=> string(3) "C++" [1]=> string(4) "DCCN" [2]=> string(4) "java" }
--------------------------------------------
```

Displaying Array Contents

In first sem i studied: Digital Logic,CIS andMaths

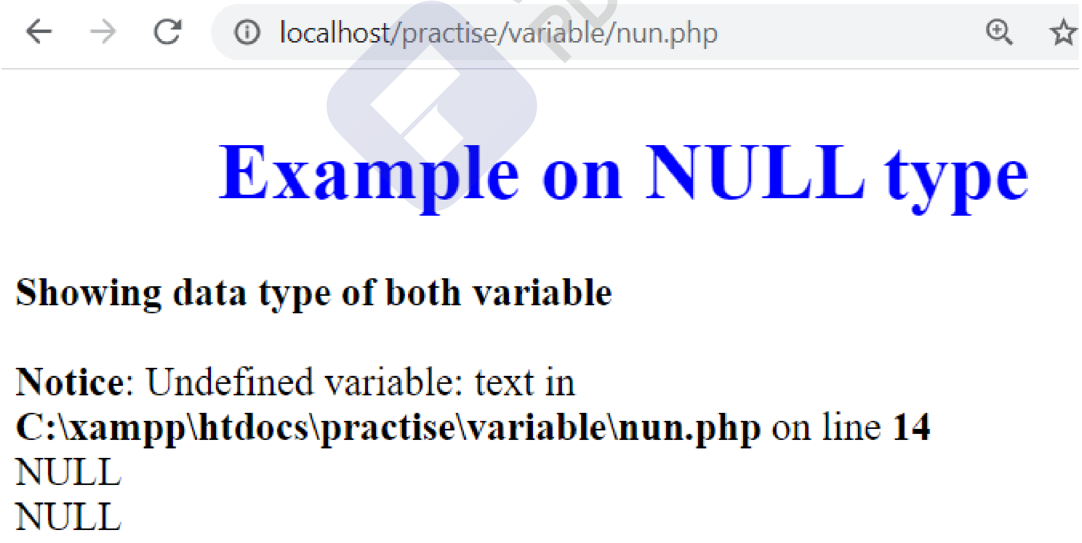
Null Type:

The NULL value is used to represent empty variable i.e. variable with unassigned value. NULL is the only possible value of type null. If a variable is declared without any value then it is automatically assigned a value of null or if variable is assigned NULL as value then it's type will be Null. For example: $text; or $text =NULL;

For example:

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <title>Null example</title>
</head>
<body>
<h1 style="color: blue;text-align: center;"> Example on NULL type  </h1>
<?php
   $text;
   $course = NULL;
   echo "<b>Showing data type of both variable</b></br>";
   var_dump($text);
   echo "</br>";
   var_dump($course);
?>
</body>
</html>
```

Output:



```
Example on NULL type

Showing data type of both variable

Notice: Undefined variable: text in
C:\xampp\htdocs\practise\variable\nun.php on line 14
NULL
NULL
```

Resource Type:

A resource is a special variable but not an actual data type used for storing a reference to function and external resources. For example, resource data type is used in a database call, open a file etc.

PHP String Function:

Calculating length of string:

The strlen() function is used to find the number of character in string. This function return the length of a string:

Syntax: strlen(String literals or String variable);
Strlen("Hello, this is Web Technology");

Calculating number of words in a String:

The str_word_count () function is used to count the number of words in a string.

Syntax: str_word_count(String literals or String Variable);
Str_word_count ("Hello, this is Web Technology");

Replacing text within string:

The str_replace() function is used to replace some character with some other character in string. It takes three argument, first argument contains string which is to be replaced by other character or word, second argument contains exact string that will put on the place of replaceable word or character, third argument contains original string from which certain character will be replaced. Sometime fourth argument can be passed to know the number of times the string replacement was performed.

Syntax:
Str_replace("word to be replaced", "word to be placed", "original string");
Str_replace("web1", "web2", "this is web1");

Reversing a String:

The strrev() function is used to reverse a string. It takes variable or string literal as argument and revers the content of them.

Syntax:
Strrev(variable or string literal)
Strrev("Hello Web2");

Search for a text within a string:

The strpos() function is used for searching text within a string which will return a position of the first occurrence of a text within string. If no match is found it will return FALSE.

strpos("string", "word or character to be search")
strpos("this is web2", "is");

The strrpos() is used for searching text within a string which will return a position of the last occurrence of a text within string. If no match is found it will return FALSE. This function is case-sensitive. Similarly, strripos() function can be also used for searching a text but it is case insensitive.

These each function takes two argument; first arguments contain exact string and second arguments contains word to be search.

Syntax: strrpos("string", "word or character to be search")
strrpos("this is web2", "is");

Converting String to lower or upper case:

The strtolower() function is used to convert whole string into lower case letter and strtoupper() function is used to convert whole string into upper case letter. The lcfirst() function is used to convert the first character of a string to lowercase, ucfirst() function is used to convert the first character of a string to uppercase and ucwords() is used to convert the first character of each word in a string to uppercase.

Syntax: strtolower (variable or string literals); //similar for all

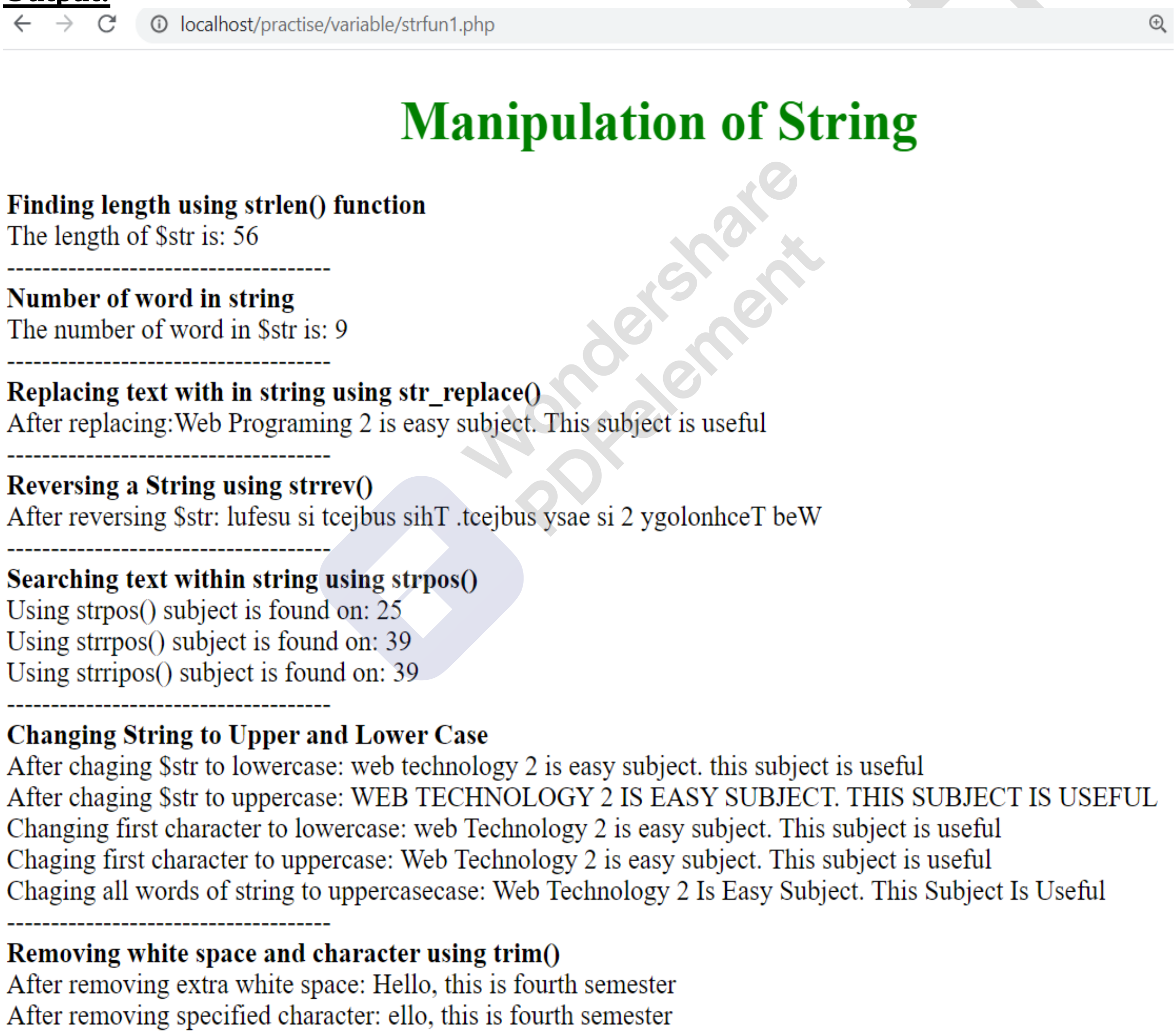
PHP trim() function:

The trim() function is used to removes whitespace or other character from both side of string. The ltrim() can be used to remove white space or other character from left side of string and from right side of string rtrim() can be used.
Syntax: trim(string variable or literals, charlist);
Charlist specifies character to be removed;

Complete Example On String Function From Strlen to trim () Function:

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <title>String function 1</title>
</head>
<body>
<h1 style="color: green;text-align: center;"> Manipulation of String </h1>
<?php
   $str = "Web Technology 2 is easy subject. This subject is useful";
   echo "<b>Finding length using strlen() function</b></br>";
   $len = strlen($str);
   echo 'The length of $str is: ',$len,"</br>";
   echo "------------------------------------ </br>";
   echo "<b>Number of word in string</b></br>";
   $numWord = str_word_count($str);
   echo 'The number of word in $str is: ',$numWord, "</br>";
   echo "------------------------------------ </br>";
   echo "<b>Replacing text with in string using str_replace()</b></br>";
   $replace = str_replace("Technology","Programing",$str);
   echo "After replacing:",$replace,"</br>";
   echo "------------------------------------ </br>";
   echo "<b>Reversing a String using strrev()</b></br>";
   $revese = strrev($str);
   echo 'After reversing $str: ',$revese,"</br>";
   echo "------------------------------------ </br>";
   echo "<b>Searching text within string using strpos()</b></br>";
   $search = strpos($str,"subject"); //return first postion of occurence
   echo "Using strpos() subject is found on: ",$search,"</br>";
   $search2 = strrpos($str,"subject"); //case sensitive & return last postion
   $search3 = strripos($str,"Subject"); //case insesitive
   echo "Using strrpos() subject is found on: ",$search2,"</br>";
   echo "Using strripos() subject is found on: ",$search3, "</br>";
   echo "------------------------------------ </br>";
   echo "<b> Changing String to Upper and Lower Case</b></br>";
   $lower = strtolower($str); //change whole string to lower case
   $upper = strtoupper($str); //change whole string to upper case
   $firstlower = lcfirst($str);
   $firstupper = ucfirst($str);
   $wordupper = ucwords($str);
   echo 'After chaging $str to lowercase: ',$lower,"</br>";
   echo 'After chaging $str to uppercase: ',$upper,"</br>";
   echo 'Changing first character to lowercase: ',$firstlower, '</br>';
   echo 'Chaging first character to uppercase: ',$firstupper,"</br>";
   echo 'Chaging all words of string to uppercasecase: ',$wordupper,"</br>";
   echo "------------------------------------ </br>";
   echo "<b>Removing white space and character using trim()</b></br>";
   $str2 = "Hello,              this is fourth semester"; //whitespace
   $trimWhite = trim($str2," "); //remove white space
   $trmChar = trim($str2,"His"); //remove specified character
   echo "After removing extra white space: ",$trimWhite,"</br>";
   echo "After removing specified character: ",$trmChar,"</br>";
?>
</body>
</html>
```

Output:



```
Manipulation of String

Finding length using strlen() function
The length of $str is: 56
------------------------------------
Number of word in string
The number of word in $str is: 9
------------------------------------
Replacing text with in string using str_replace()
After replacing:Web Programing 2 is easy subject. This subject is useful
------------------------------------
Reversing a String using strrev()
After reversing $str: lufesu si tcejbus sihT .tcejbus ysae si 2 ygolonhceT beW
------------------------------------
Searching text within string using strpos()
Using strpos() subject is found on: 25
Using strrpos() subject is found on: 39
Using strripos() subject is found on: 39
------------------------------------
Changing String to Upper and Lower Case
After chaging $str to lowercase: web technology 2 is easy subject. this subject is useful
After chaging $str to uppercase: WEB TECHNOLOGY 2 IS EASY SUBJECT. THIS SUBJECT IS USEFUL
Changing first character to lowercase: web Technology 2 is easy subject. This subject is useful
Chaging first character to uppercase: Web Technology 2 is easy subject. This subject is useful
Chaging all words of string to uppercasecase: Web Technology 2 Is Easy Subject. This Subject Is Useful
------------------------------------
Removing white space and character using trim()
After removing extra white space: Hello, this is fourth semester
After removing specified character: ello, this is fourth semester
```

Some Other String Manipulation Function:

String Function	Description
count_chars()	Returns information on number of different characters used on string or which character have not been used or have been used. **Syntax:** count_chars(string, mode)

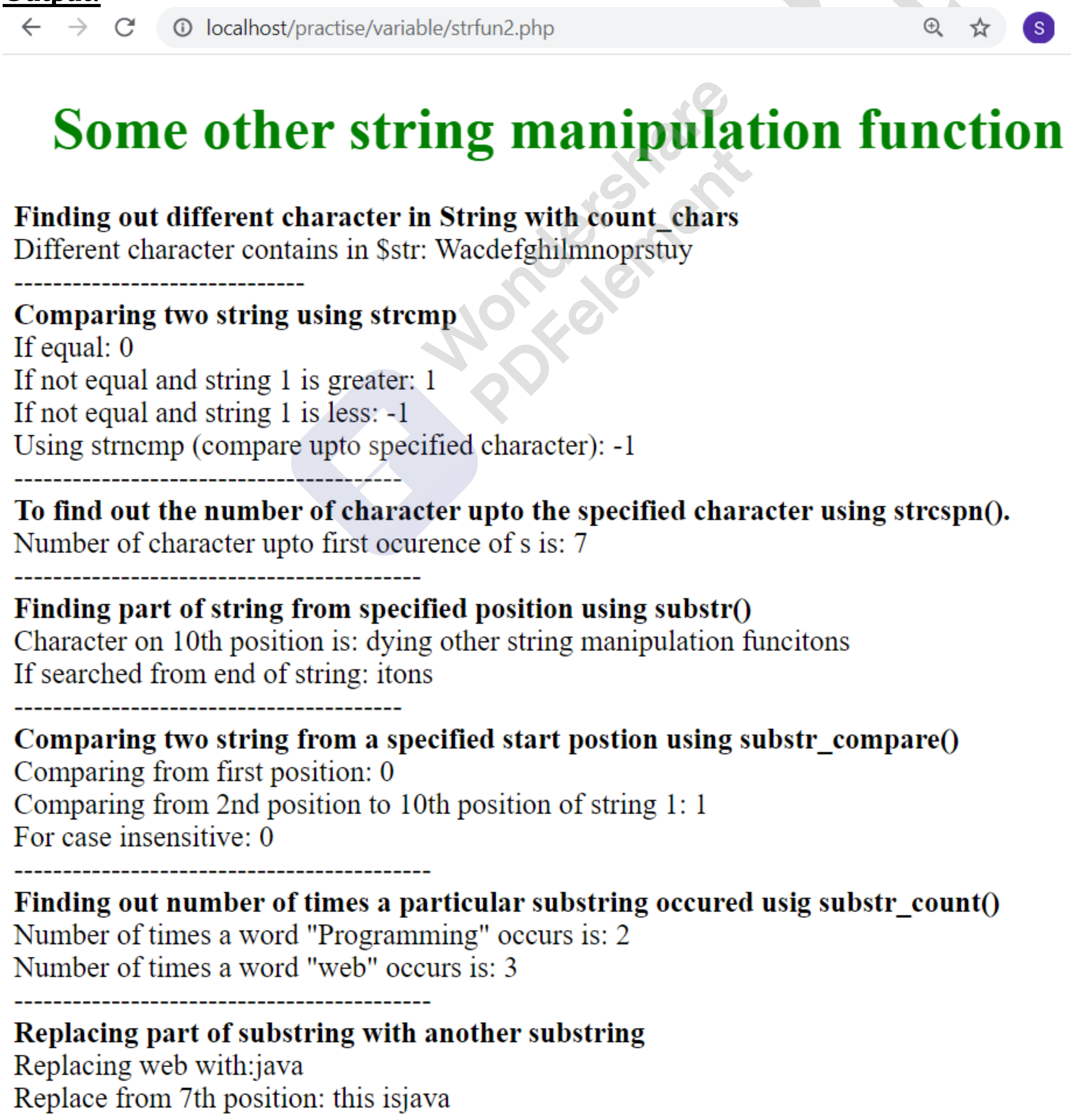
	In mode: 0,1 and 2 is used for array, 3 is used to find a string with different character used and 4 is used to find a string with all the unused characters.
strcmp()	Used for string comparison i.e. it compares two strings and are case sensitive. **Syntax:** strcmp(string1, string2) It return 0 if both string matches, less than 0 if string 1 is less than string 2 and greater than 0 is string 1 is greater than string 2.
strncmp()	Same as strcmp() but difference is that here, a number of character to be used for comparison from each string can be determined. **Syntax:** strncmp(String1, String2, length); The numeric value in length determine number of characters to be used for comparison from each string.
strcspn()	Return the number of character found upto the specified character. **Syntax:** strcspn(string, char, start, length). Strcspn("Hello, Web", "W", 0,8); Here, string specifies string to search, char refer to specified character to be search, start is optional which specifies from which position to start and length (optional) specifies how much of string to be searched.
substr()	It returns the part of string from specified position. **Syntax:** substr(string, start) Substr("this is web 2", 5) Here, string specifies original string, start specifies a position from which a string to be returned. Start has three value: positive number starts at a specified position from start of string, negative number starts at a specified position from the end of the string and 0 (zero) start at the first character in string.
substr_compare	Compares two string from a specified position and return 0 is strings are equal, negative if first string is less than second string, positive if first string is greater than second. **Syntax:** substr_compare (string1, string3, startpos, length, case) Here, string 1 and 2 are string to be compare with, startpos can be positive (start comparing from starting position) and negative (start comparing from end of string), length is optional which specifies how much of string1 to compare and case include true (which determines case insensitive) or false .
substr_count()	Return number of times a substring occurs in string. **Syntax:** substr_count(string, substring ,start, length) Here, string is original string, substring specifies a string to be search for, start is optional which contains positive value (specifies a position to start searching) and negative value which will start counting from end of string. Length is optional which specifies ending point of search.
Substr_replace ()	Used for replacing a part of string with other string. **Syntax:** substr_replace(string, replacement,start) Here, string is original string, replacement specifies a string to be replaced, start which contains positive value (search for replacement

	will start from starting postion), negative value (search for replacement will start from end of string).

Example of Other String Manipulation Function:

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <title>some other string function</title>
</head>
<body>
<h1 style="color: green;text-align: center;"> Some other string manipulation function </h1>
<?php
   $str = "We are studying other string manipulation funcitons";
   echo "<b>Finding out different character in String with count_chars</b></br>";
   $diffchar= count_chars($str,3); //finds different character
   echo 'Different character contains in $str: ',$diffchar,"</br>";
   echo "-----------------------------</br>";
   echo "<b>Comparing two string using strcmp</b></br>";
   echo "If equal: ",strcmp("this is php","this is php"),'</br>';
   echo "If not equal and string 1 is  greater: ", strcmp("this is php","this is C"),'</br>';
   echo "If not equal and string 1 is less: ",strcmp("this is php","this is web program"),'</br>';
   //folowing statement compare upto 3 character
   echo "Using strncmp (compare upto specified character): ",strncmp("the is php","this is
php",3),"</br>";
   echo "---------------------------------------</br>";
   echo "<b>To find out the number of character upto the specified character using strcspn().
</b></br>";
   $find = strcspn($str,"s");
   echo  "Number of character upto first ocurence of s is: ",$find,"</br>";
   echo "----------------------------------------</br>";
   echo "<b>Finding part of string from specified position using substr()</b></br>";
   $search = substr($str,10); //search from first position of string
   $searchEnd = substr($str,-5); //search from end of string
   echo "Character on 10th position is: ",$search,"</br>";
   echo "If searched from end of string: ",$searchEnd,"</br>";
   echo "---------------------------------------</br>";
   echo "<b>Comparing two string from a specified start postion using
substr_compare()</b></br>";
   echo "Comparing from first position: ", substr_compare("Hello Programming","Hello
Programming",0),"</br>";
   //comparing from and to specified position
   echo "Comparing from 2nd position to 10th position of string 1: ",substr_compare("Hello
Programming","Hello Programming",2,10),"</br>";
   //comparing from and to specified position and case insensitive
   echo "For case insensitive: ",substr_compare("Hello Programming","hello
progRamming",0,10,True),"</br>";
   echo "-----------------------------------------</br>";
   echo "<b>Finding out number of times a particular substring occured usig
substr_count()</b></br>";
   $str2 = "This is web programming and web programming is fun. web is easy as well";
   echo "Number of times a word \"Programming\" occurs is: ",
substr_count($str2,"programming"),"</br>";
   echo "Number of times a word \"web\" occurs is: ", substr_count($str2,"web"),"</br>";
   echo "-------------------------------------------</br>";
   echo "<b>Replacing part of substring with another substring</b></br>";
   echo "Replacing web with:", substr_replace("web","java",0),"</br>";
   echo "Replace from 7th position: ",substr_replace("this is web","java",7);

?>
</body>
</html>
```

Output:



```
Some other string manipulation function

Finding out different character in String with count_chars
Different character contains in $str: Wacdefghilmnoprstuy
------------------------------
Comparing two string using strcmp
If equal: 0
If not equal and string 1 is greater: 1
If not equal and string 1 is less: -1
Using strncmp (compare upto specified character): -1
----------------------------------------
To find out the number of character upto the specified character using strcspn().
Number of character upto first ocurence of s is: 7
------------------------------------------
Finding part of string from specified position using substr()
Character on 10th position is: dying other string manipulation funcitons
If searched from end of string: itons
----------------------------------------
Comparing two string from a specified start postion using substr_compare()
Comparing from first position: 0
Comparing from 2nd position to 10th position of string 1: 1
For case insensitive: 0
-------------------------------------------
Finding out number of times a particular substring occured usig substr_count()
Number of times a word "Programming" occurs is: 2
Number of times a word "web" occurs is: 3
-------------------------------------------
Replacing part of substring with another substring
Replacing web with:java
Replace from 7th position: this isjava
```

Creating Class and Objects

Classes and objects are the two main aspects of object-oriented programming. A class is a self-contained, independent collection of variables and functions which work together to perform one or more specific tasks, while objects are individual instances of a class.

A class acts as a template or blueprint of object from which lots of individual objects can be created. When individual objects are created, they inherit the same generic properties and behaviours, although each object may have different values for certain properties.

For example, think of a class as a blueprint for a house. The blueprint itself is not a house, but is a detailed plan of the house. While, an object is like an actual house built according to that blueprint. We can build several identical houses from the same blueprint, but each house may have different paints, interiors and families inside.

Creating Class

To define a Cat class, we use the class keyword followed by the class name (typically title cased in PHP) and curly brackets:

```
class Cat
 {
 }
```

Within the curly brackets, we can add properties, which define the data each object of the class will contain. The syntax is similar to how we define variables:

```
class Cat
   {
     public $name, $color;
   }
```

Example:

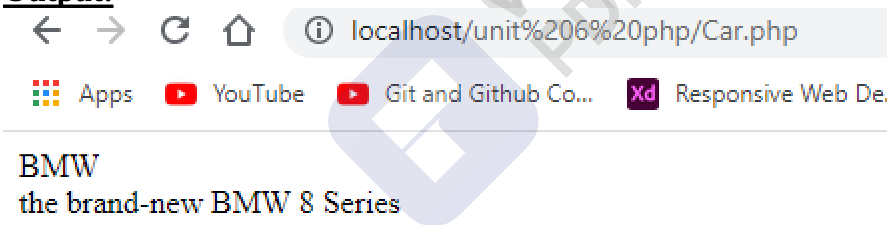
```
<?php
        class Car
           {
             // to store name of person
               var $url = "BMW";

             // simple class method
             function Show()
                 {
                    echo " the brand new BMW 8 Series";
                 }
            }
             // creating class object
          $obj = new car();
?>
```

As you can see, we used the new keyword in the last line to create an object of the class Car and assigned it to the variable $obj.

Once we have an object of any class, we can use it to access the class methods and variable using -> operator.

```
<?php
        class Car
            {
               // to store name of person
                 var $kind = "BMW";

         // simple class method
               function Show()
                   {
                      echo " the brand-new BMW 8 Series";
                   }
              }

           // creating class object
                  $obj = new car();

           // accessing class variable
                 echo $obj->kind . "<br/>";
           // calling class method
                 $obj-> Show();
?>
```

Output:



```
BMW
the brand-new BMW 8 Series
```

PHP Forms

A form is an HTML tag that contains graphical user interface items such as input box, check boxes radio buttons etc. Forms are used to get input from the user and submit it to the web server for processing. Example like when you login into a website or into your mail box, you are interacting with a form.

To create a form, you use the <form> element as follows:

```
<form action="form.php" method="post">

</form>
```

The <form> element has two important attributes:

- **action:** specifies the URL that processes the form submission. In this example, the form.php will process the form.

- **method:** specifies the HTTP method for submitting the form. The most commonly used form methods are POST and GET.

The form method is case-insensitive. It means that you can use either post or POST. If you don’t specify the method attribute, the form element will use the get method by default.

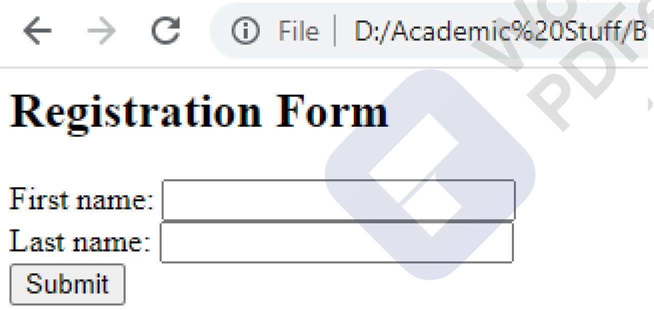
Create a simple registration form

Example

Form2.html

```
<html>
<head>
   <title>form</title>
   </head>
   <body>

   <form action="form2.php" method="POST">
      <h2>Regustration form</h2>
            First name:<input type="text" name="firstname"><br>
            Last name:<input type="text" name="lastname"><br>
            <input type="submit" name="submit" value="Submit">
         </form>
   </body>
</html>
```

Output



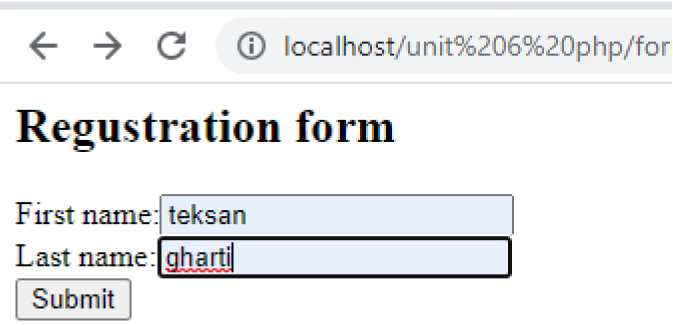
File | D:/Academic%20Stuff/B

Registration Form

First name:

Last name:

Submit



localhost/unit%206%20php/for

Regustration form

First name: teksan

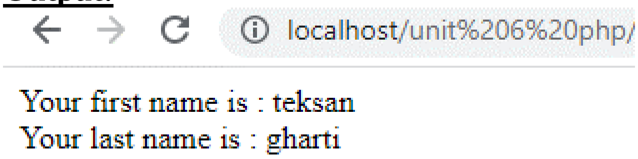
Last name: gharti

Submit

Form2.php

```
<?php
if(isset($_POST['firstname'],$_POST['lastname']))
   {
      $username = $_POST['firstname'];
      $lastname = $_POST['lastname'];
      echo "Your first name is : $username  <br>";
      echo "Your last name is : $lastname";
   }
   ?>
```

Output:



```
Your first name is : teksan
Your last name is : gharti
```

Form Validation

Formvalidation1.html

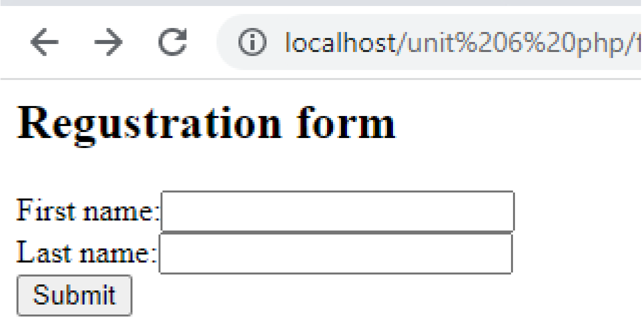
```
<html>
<head>
   <title>form</title>
   </head>
   <body>

   <form action="formvalidation1.php" method="POST">
      <h2>Regustration form</h2>
            First name:<input type="text" name="firstname"><br>
            Last name:<input type="text" name="lastname"><br>
            <input type="submit" name="submit" value="Submit">
         </form>

   </body>
</html>
```

Formvalidation1.php

```
<?php
if(isset($_POST['submit'])){
   $username = $_POST['firstname'];
   $lastname = $_POST['lastname'];

   if(empty($username)){
      echo "username field is empty";
   }
   else if(empty($lastname)){
      echo "lastname is emptly";
```

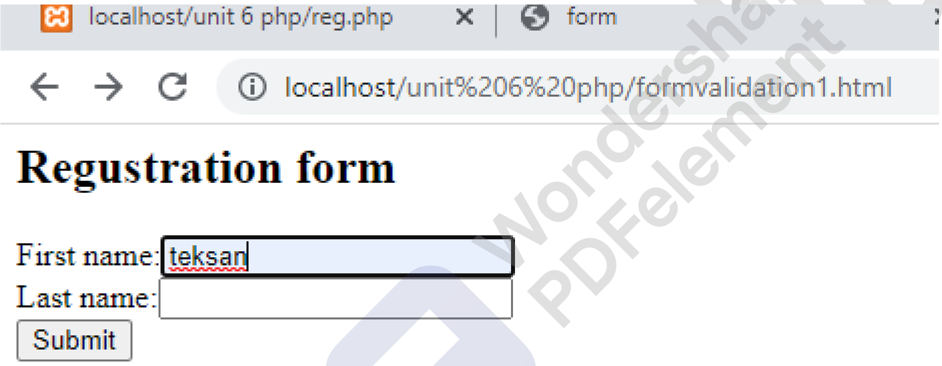
```
    }
    else{
        echo "form submitted";
    }
}

?>
```

Output



Regustration form

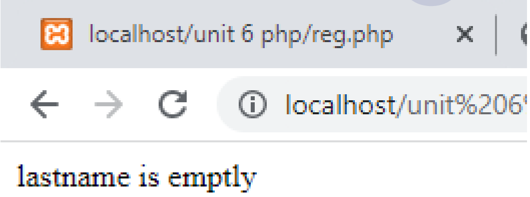
First name:
Last name:
Submit



localhost/unit 6 php/reg.php
form
localhost/unit%206%20php/formvalidation1.html

Regustration form

First name: teksan
Last name:
Submit



lastname is emptly

PHP State Management (Session and Cookies):

The basic foundation of a data communication system for the internet is HTTP (Hypertext Transfer Protocol) which is an application layer protocol, distributive and collaborative, and hypermedia information system. As we know that HTTP is a generic and stateless protocol to manage state in applications like E-commerce, Social Media, Blog sites, many commercial sites sessions and cookies are used.

Type of state management system

Server-side state management system: In server-side state management system where we used to store user specific information to identify user on server and information is available in every web pages.
Example: session variables
Client side State management System: In a client-side state management system, the user information is stored by the browser.
Example: cookies

PHP Session:

An alternative way to make data accessible across the various pages of an entire website is to use a PHP Session. PHP session is used to store and pass information from one page to another temporarily until user close the website or destroy the session.

A session creates a file in a temporary directory on the server where registered session variables and their values are stored. This data will be available to all pages on the site during that visit.

PHP session technique is widely used in shopping websites where we need to store and pass cart information e.g. username, product code, product name, product price etc from one page to another.

PHP session creates unique user id for each browser to recognize the user and avoid conflict between multiple browsers. Session variables hold information about one single user, and are available to all pages in one application.

Start a PHP Session

A session is started with the ***session_start()*** function. Session variables are set with the PHP global variable: $_SESSION.
Now, let's create a new page called "sessionStart.php". In this page, we start a new PHP session and set some session variables:
sessionStart.php

```
<?php
   session_start();
   echo "Session Started <br>";
   //set session variable
   $_SESSION['user'] = "teksan";
   echo "Session variable is set";
?>
```

Get PHP Session Data

Next, we create another page called "sessionVariableDisplay.php". From this page, we will access the session information we set on the first page ("sessionStart.php").
Notice that session variables are not passed individually to each new page, instead they are retrieved from the session we open at the beginning of each page by using session_start() function.
Also notice that all session variable values are stored in the global $_SESSION variable:

```
<?php
   session_start();
   echo"User Name is: ".$_SESSION['user'];
?>
```

Change session data

Session variables are mutable, which means they can be changed during runtime. To change the value of a session variable, we simply assign a new value to it.

```
<?php
   session_start();
   $_SESSION['user']="vijay";
   echo "Session data is changed";
?>
```

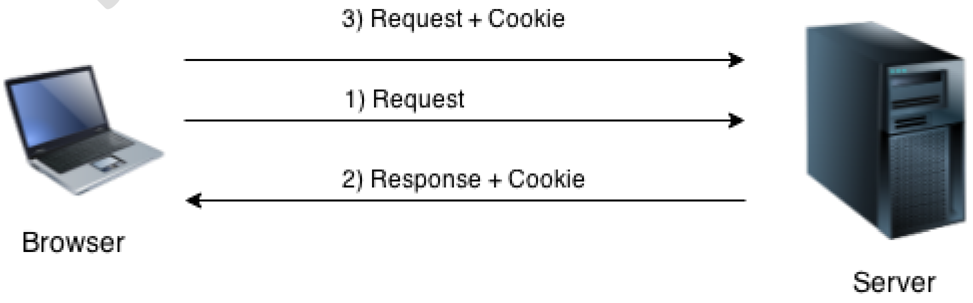
Destroy a PHP Session

To remove all global session variables and destroy the session,
use session_unset() and session_destroy() functions.

```
<?php
   session_start();
   session_unset();
   session_destroy();
   echo "Session is Destroyed";
?>
```

PHP Cookie

PHP cookie is a small piece of information which is stored at client browser. It is used to recognize the user.
Cookie is created at server side and saved to client browser. Each time when client sends request to the server, cookie is embedded with request. Such way, cookie can be received at the server side.



In short, cookie can be created, sent and received at server end.

Cookies are states that are saved to the user's system, instead of the server. Unlike a session, a cookie has a 1024 byte size limit. Cookies are sent to the web server as header information in every HTTP request.
NOTE: *Cookies are not stored on the server, they can be modified and deleted. Cookies are less reliable and secure than sessions.*

As an example, let's consider an application with a member area. Once a user enters their log in details, a cookie is created on that user's system that saves those details.
If the user comes back to the application, the login details can be automatically filled into the form so that the user doesn't have to repeat the process.
Cookies are also commonly used to serve advertisements based on products that the user views around the web.
For example, if a user views a product on Amazon, they will find advertisements of similar products when using Facebook or Google services.

Set a cookie

PHP provides us with the ***setcookie()*** function to create, or set, a cookie.
Syntax:
setcookie(name, value, expire, path, domain, security);
The first argument, ***name***, is mandatory for every cookie. The rest of the arguments are optional, but recommended.

Argument	**Usage**
Name	Required. The name the cookie will be referred to. Must be a string.
Value	Optional. The value of the cookie.
Expire	Optional. If an expiration time is not set, the cookie will expire when the browser is closed.
Path	Optional. The path on the server the cookie will be available on. The cookie can be set to '/' to be available to the entire domain.
Domain	Optional. The (sub) domain that the cookie is available to. Sub-domains of the specified domain are automatically included.
Security	Optional. If set to true (1), the cookie will only be set for a HTTPS secure connection.

Example:

```
<?php
   $cookiename = "user";
   $cookievalue = "Gopal";
   // Expires when the browser closes
   setcookie("$cookiename","$cookievalue");
   echo "cookie is set successfully";

   // Expires in 1 minute (60 seconds)
   setcookie("$cookiename", "$cookievalue", time()+60);
   echo "Cookie set";

   // Only available to a specific subdomain over HTTPS parameter 1 means true
   setcookie("$cookiename", "$cookievalue", time()+60, "/mypath/", "mydomain.com", 1);
    echo "cookie is set";
?>
```

Accessing Cookie Values:

For accessing a cookie value, the php **$_COOKIE** superglobal variable is used. It is an associative array that contains a record of all the cookies values sent by the browser in the current request. The records are stored as a list where the cookie name is used as the key. To access a cookie named “user”, the following code can be executed.

Example:

```
<?php
  echo $_COOKIE['user'];
?>
```

Modify Cookie values:

To modify a cookie, just set (again) the cookie using the setcookie() function:

Example:

```
<?php
   $cookiename = "user";
   $cookievalue = "Rohan";
   setcookie("user","Rohan",time()+60);
   echo "cookie is Modify successfully";
?>
```

Checking Whether a Cookie Is Set Or Not:

It is always advisable to check whether a cookie is set or not before accessing its value. Therefore to check whether a cookie is set or not, the PHP **isset()** function is used.

Example: To check whether a cookie “user” is set or not, the **isset()** function is executed as follows:

```
<?php
  if(isset($_COOKIE['user']))
  {
     echo $_COOKIE['user'];
  }
  else
  {
    echo "Cookie is not set";
  }
?>
```

Deleting Cookies:

The **setcookie()** function can be used to delete a cookie. For deleting a cookie, the **setcookie()** function is called by passing the cookie name and other arguments or empty strings but however this time, the expiration date is required to be set in the past or time in negative value.

Example: To delete a cookie named “user”, the following code can be executed.

```
<?php
   setcookie("user","",time()-60);
   echo "cookie is deleted successfully";
?>
```

This (-60) means it will delete all the cookies named “user” set 1minute before. If we set (-3600) it means all cookies set before 1 hour will be deleted.

Important Points:

- If the expiration time of the cookie is set to 0 or omitted, the cookie will expire at the end of the session i.e. when the browser closes.
- The same path, domain, and other arguments should be passed that were used to create the cookie in order to ensure that the correct cookie is deleted.

Difference between Storing Data in Cookies and Session Variable:

- Cookies are returned and stored in the user’s browser, session data is stored in web server.
- The life span of cookie can be set to almost any duration of our choice. Php session have a predefined short life (when the browser is closed).
- Depending on how web server is configured session data is often stored in public temporary directory on the server. As such it is possible that other users on the server may be able to peek at the data stored in server.

When to Use Session Rather than Cookies:

When we need the data stored on the server and not in user’s browser:

When a cookie is set, it is return to the user and stored in browser. Sometimes this is not a good idea.

For example: some website have a CAPTCHA test on their web comment forms where an image showing a few random letters and numbers is displayed and users are supposed to type in those characters to prove that they are human and not some spam. In order for this to work the script generating the image needs to be stored in secret word somewhere so that the program doing the checking can verify the user’s answer.

In such case returning a cookie to the user is not good since a span on receiving that cookie can find out the secret word. We can encrypt it but why bother when PHP session is exactly what we need for this purpose.

When the data is transient, and only relevant for the current browsing session:

Since we don’t know how long session data will be stored it stands to reason that we should only use session when we don’t really need data for long periods of time and when the data is not so important.

When the data does not contain any information that needs to be securely kept:

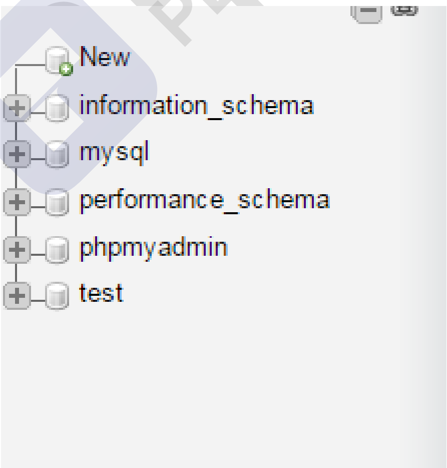
As session data is kept in temporary directory on web server, this is usually a public accessible folder that anyone with an account on the computer can read. So we should be careful while storing information in session variable. For e.g. do not store credit card number, person particulars, passwords, username etc. in session variable.

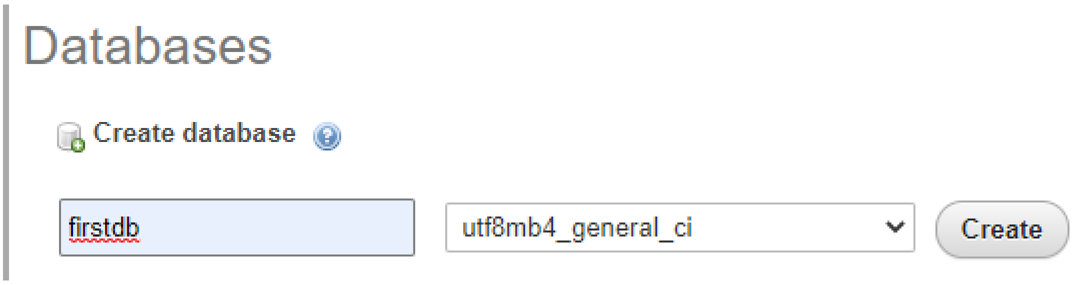
Database connection

1. Create MySQL Database at the Localhost
    a. Create Database
    b. Create a Folder in htdocs
    c. Create Database Connection File In PHP
    d. Create new php file to check your database connection
    e. Run it

Create Database

Now return to the homepage of PHPMyAdmin. Click the New button to create a new database.




Databases

Create database

firstdb

utf8mb4_general_ci

Create a Folder in htdocs

Now, locate the folder where you installed XAMPP and open the htdocs folder. Create a new folder inside c:/xampp/htdocs/ and name it “firstdb” we will place web files in this folder. Why we have created a folder in htdocs? XAMPP uses folders in htdocs to execute and run your PHP sites.

Create Database Connection File In PHP

Create a new PHP file and name it conn.php and save it.

Conn.php

```
<?php
$servername="localhost";
$hostusername="root";
$hostpassword="";
$db="db_final";

$conn = mysqli_connect($servername,$hostusername,$hostpassword,$db);
if(!$conn){
        die("error in connection->".mysqli_connect_error());
}else{
   echo "Connected to db";
}
?>
```

Making form

Form3.php

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <meta http-equiv="X-UA-Compatible" content="IE=edge">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <title>Document</title>
</head>
<body>
   <h2> making form for Insert delete, display and update</h2>
   <form action="val1.php" method="POST">
   <label>Username</label><input type="text" name="uname"/><br/>
   <label>Gender: </label><input type="radio" name="gender" value="male"/>Male
&nbsp;&nbsp;
   <input type="radio" name="gender" value="female"/>Fe-Male <br/>
   <label>Age</label><input type="number" name="age"/><br/>
   <input type="submit" name="submit" value="submit"/>
   </form>
```

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

For Form Validation

Val1.php

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <meta http-equiv="X-UA-Compatible" content="IE=edge">
   <meta name="viewport" content="width=device-width, initial-scale=1.0">
   <title>Document</title>
</head>
<body>
   <h2>validating form field and inserting in database</h2>
   <?php
      if(isset($_POST['submit'])){
         if(empty($_POST['uname'])||!isset($_POST['gender'])||empty($_POST['age'])){
            echo "please select all the field";
         }else{
            include "conn.php";
            $uname=$_POST['uname'];
            $gen = $_POST['gender'];
            $age = $_POST['age'];

            //inserting into db
            $sql_insert="INSERT INTO tbl_final(username,gender,age) VALUES
('$uname','$gen','$age')";
            $sql_insert_query = mysqli_query($conn,$sql_insert);
            if($sql_insert_query){
               echo "inserted";
              header("Location: dis.php");
            }else{
               echo"not inserted";
            }
         }
      }else{
         echo "error in submit";
      }
?>
</body>
</html>
```

Displaying Database information in PHP

Dis.php

```
<?php
include "conn.php";
$sql_display= "SELECT * From tbl_final"; //to fetch all info
$sql_display_single = "SELECT * FROM std_tbl where id=1";//display single info
```

```
$sql_display_name = "SELECT * FROM std_tbl where username='Arju'";//display by name
$sql_display_gender="SELECT * From std_tbl where gender='female'";//display by gender
$sql_display_age = "SELECT * From std_tbl where age='20'";//display by age
$sql_display_execute=mysqli_query($conn,$sql_display_age);

if($sql_display_execute){
   $numberofrecord = mysqli_num_rows($sql_display_execute); //fetch no. of record form
table
   echo "<br/>$numberofrecord";
   echo "<table border=1px;>";
   echo "<tr>";
   echo "<th>id</th>";
   echo "<th>username</th>";
   echo "<th>gender</th>";
   echo "<th>age</th>";
   echo "<th>Delete</th>";
   echo "</tr>";

   while($row=mysqli_fetch_assoc($sql_display_execute)){
      $id=$row['id'];
      $username = $row['username'];
      $gender = $row['gender'];
      $age = $row['age'];
      //echo "your id is $id";
      echo "<tr>";
      echo "<td>$id</td>";
      echo "<td>$username</td>";
      echo "<td>$gender</td>";
      echo "<td>$age</td>";
      echo "<td><a href='del2.php'>Delete</a></td>";
      echo "</tr>";
   }
   echo "</table>";
}
?>
```

Making Form, Validating For, Connecting to Database and performing CRUDE Operation

Step 1: Connecting to database

Conn.php

```
<?php
$servername="localhost";
$hostusername="root";
$hostpassword="";
$db="db_final";
$conn = mysqli_connect($servername,$hostusername,$hostpassword,$db);
if(!$conn){
```

```
            die("error in connection->".mysqli_connect_error());
}else{
    echo "Connected to db";
}
?>
```

Step 2: Making Form

Form.php

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
<body>
    <h2> making form for Insert delete, display and update</h2>
    <form action="val1.php" method="POST">
    <label>Username</label><input type="text" name="uname"/><br/>
    <label>Gender: </label><input type="radio" name="gender" value="male"/>Male
&nbsp;&nbsp;
    <input type="radio" name="gender" value="female"/>Fe-Male <br/>
    <label>Age</label><input type="number" name="age"/><br/>
<label>Hobby:</label><input type="checkbox" name="chk[]"
value="swimming"/>Swimming &nbsp;
    <input type="checkbox" name="chk[]" value="dancing"/>Dancing &nbsp;
    <input type="checkbox" name="chk[]" value="singing"/>Singing &nbsp;
    <br/>
    <input type="submit" name="submit" value="submit"/>
    </form>
</body>
</html>
```

Step 3: Validating The Form

Val1.php

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
<body>
    <h2>validating form field and inserting in database</h2>
    <?php
        if(isset($_POST['submit'])){
            if(empty($_POST['uname'])||!isset($_POST['gender'])||empty($_POST['age'])||
!isset($_POST['chk'])){
                echo "please select all the field";
```

```
      }else{
        include "conn.php";
        $uname=$_POST['uname'];
        $gen = $_POST['gender'];
        $age = $_POST['age'];
      $hobby = $_POST['chk'];
      $hobby_str = implode(",",$hobby);
        //inserting into db
        $sql_insert="INSERT INTO tbl_final(username,gender,age,hobby) VALUES
('$uname','$gen','$age',’hobby_str’)";
        $sql_insert_query = mysqli_query($conn,$sql_insert);
        if($sql_insert_query){
          echo "inserted";
          header("Location: dis.php");

        }else{
          echo"not inserted";
        }
      }
    }else{
      echo "error in submit";
    }
?>
</body>
</html>
```

Step 4: Displaying data of database in PHP

Dis.php

```
<?php
include "conn.php";
$sql_display= "SELECT * From tbl_final"; //to fetch all info
$sql_display_single = "SELECT * FROM tbl_final where id=1";//display single info
$sql_display_name = "SELECT * FROM tbl_final where username='Arju'";//display by
name
$sql_display_gender="SELECT * From tbl_final where gender='female'";//display by gender
$sql_display_age = "SELECT * From tbl_final where age='20'";//display by age
$sql_display_execute=mysqli_query($conn,$sql_display);

if($sql_display_execute){
  $numberofrecord = mysqli_num_rows($sql_display_execute); //fetch no. of record form
table
  echo "<br/>$numberofrecord";
  echo "<table border=1px;>";
  echo "<tr>";
  echo "<th>id</th>";
  echo "<th>username</th>";
  echo "<th>gender</th>";
  echo "<th>age</th>";
```

```
    echo “<th>hobby</th>”;
    echo "<th>Delete</th>";
    echo "</tr>";

    while($row=mysqli_fetch_assoc($sql_display_execute)){
        $id=$row['id'];
        $username = $row['username'];
        $gender = $row['gender'];
        $age = $row['age'];
        $hobby = $row[‘hobby’];
        //echo "your id is $id";
        echo "<tr>";
        echo "<td>$id</td>";
        echo "<td>$username</td>";
        echo "<td>$gender</td>";
        echo "<td>$age</td>";
        echo "<td>$hobby</td>";
        echo "<td><a href='del2.php'>Delete</a></td>";
        echo "</tr>";
    }
    echo "</table>";
}
?>
```

Step 5: Updating Record of Database

Update.php

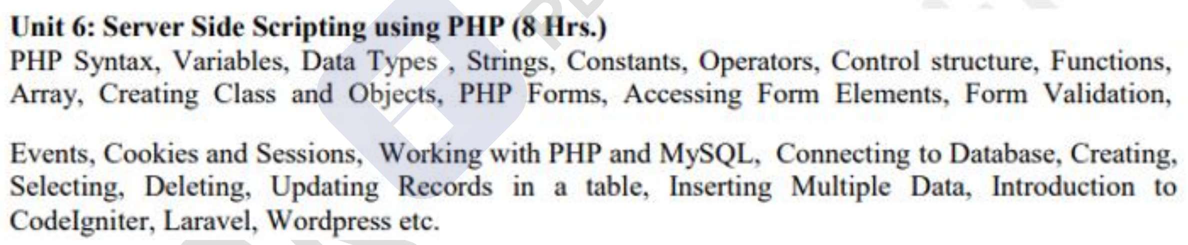
```
<?php
include "conn.php";
if(isset($_POST['submit'])){

if(empty($_POST['id'])||empty($_POST['uname'])||!isset($_POST['gender'])||empty($_POST[
'age'])){
        echo "please select all the field";
    }else{
        $id=$_POST['id'];
        $update_username=$_POST['uname'];
        $update_gender=$_POST['gender'];
        $update_age=$_POST['age'];
             $update_hobby=$_POST['chk'];
            $hobby_str = implode(",",$update_hobby); //converting into string
            $sql_update="UPDATE tbl_final SET
username='$update_username',gender='$update_gender',age='$update_age',hobby=’$hobby_
str’ Where id=$id";
        $sql_update_query=mysqli_query($conn,$sql_update);
        if($sql_update_query){
            echo "record updated";
            //header("Location: dis.php");
        }else{
            echo "record not updated";
        }
    }
}

?>

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
<body>
    <h2>insert id and info of a person that you want to updata</h2>
    <form action="" method="POST">
    <label>Id:</label><input type="number" name="id"/><br/>
    <label>Username</label><input type="text" name="uname"/><br/>
    <label>Gender: </label><input type="radio" name="gender" value="male"/>Male
&nbsp;&nbsp;
    <input type="radio" name="gender" value="female"/>Fe-Male <br/>
    <label>Age</label><input type="number" name="age"/><br/>
<label>Hobby:</label><input type="checkbox" name="chk[]"
value="swimming"/>Swimming &nbsp;
    <input type="checkbox" name="chk[]" value="dancing"/>Dancing &nbsp;
    <input type="checkbox" name="chk[]" value="singing"/>Singing &nbsp;
    <br/>
    <input type="submit" name="submit" value="submit"/>
    </form>
</body>
</html>
```

Step 6: Deleting a Record

Del.php

```
<?php
include "conn.php";
    if(isset($_POST['submit'])){
        if(empty($_POST['id'])){
            echo"select id you want to delete";
        }else{$del_id = $_POST['id'];
            $sql_delete = "DELETE FROM tbl_final WHERE id=$del_id";
            $sql_del_execute = mysqli_query($conn,$sql_delete);
            if($sql_del_execute){
                echo "<br/>record deleted";
                header ("location: dis.php");
            }else{
                echo "record not deleted";
            }

            }
        }
    else{
        echo "errpr";
    }
    ?>
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
<body>
    <h2>insert id of person of which u want to delete record</h2>
    <form action="" method="POST">
    <label>id<label><input type="number" name="id"/><br/>
    <input type = "submit" name="submit" value="delete"/>
    </form>
</body>
</html>
```

===================== Unit- 6 End=======================



Unit 6: Server Side Scripting using PHP (8 Hrs.)
PHP Syntax, Variables, Data Types , Strings, Constants, Operators, Control structure, Functions, Array, Creating Class and Objects, PHP Forms, Accessing Form Elements, Form Validation,

Events, Cookies and Sessions, Working with PHP and MySQL, Connecting to Database, Creating, Selecting, Deleting, Updating Records in a table, Inserting Multiple Data, Introduction to CodeIgniter, Laravel, Wordpress etc.

Unit 6: Server-Side Scripting using PHP (8 Hrs.)

PHP Syntax, Variables, Data Types , Strings, Constants, Operators, Control structure, Functions, Array,

Creating Class and Objects, PHP Forms, Accessing Form Elements, Form Validation, Events, Cookies and Sessions, Working with PHP and MySQL, Connecting to Database, Creating, Selecting, Deleting, Updating Records in a table, Inserting Multiple Data, Introduction to CodeIgniter, Laravel, Wordpress etc.