Complete notes on

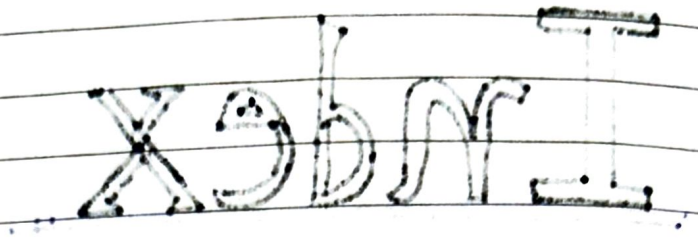
PHP

Written By:-
Dipti Gaydhane
(Software Engineer)

PAGE NO.:
DATE : / /

Index

PHP Tutorial :



13) PHP Switch

14) PHP Loops

15) PHP Functions

16) PHP Arrays

17) PHP Superglobals

18) PHP RegEx

PHP Forms :

1) PHP Form Handling

2) PHP Form validation

3) PHP Form Required

4) PHP Form URL/E-mail

5) PHP Form Complete

PAGE NO.:
DATE: / /

PHP Advanced :

1) PHP Include
2) PHP File Handling
3) PHP File Open / Read
4) PHP File Upload
5) PHP Cookies
6) PHP Sessions
7) PHP Callback Functions
8) PHP JSON
9) PHP Exceptions

PHP OOP :

1) PHP What is OOP
2) PHP Classes / Objects
3) PHP Constructor
4) PHP Destructor

5) PHP Access Modifiers

6) PHP Inheritance

7) PHP Constants

8) PHP Abstract Classes

9) PHP Interfaces

10) PHP Static Methods

MySQL Database :

1) MySQL Database

2) MySQL Connect

3) MySQL Create DB

4) MySQL Create Table

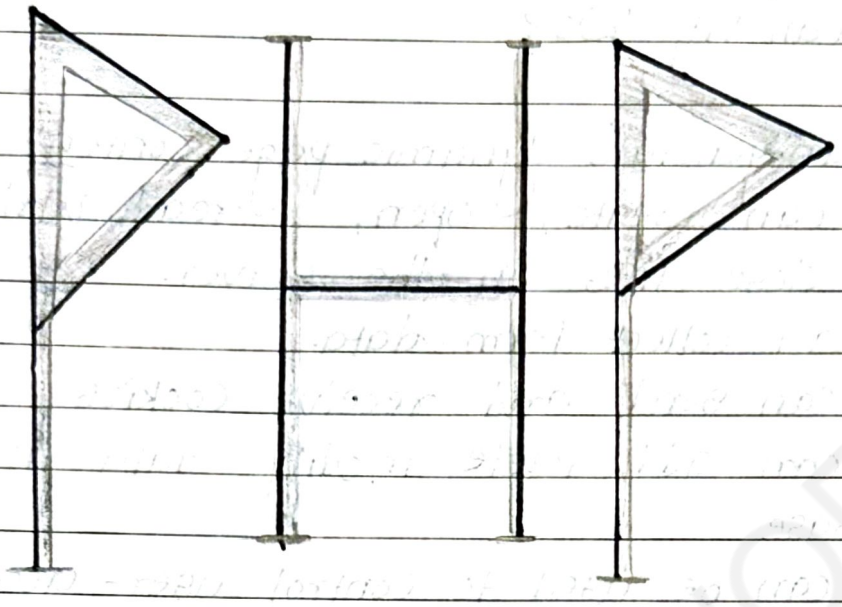
5) MySQL Insert Data

6) MySQL Select Data

7) MySQL Delete Data

8) MySQL Update Data

PAGE NO.:
DATE : / /



PHP Introduction :

What is PHP ?

- PHP is an acronym for "PHP : Hypertext Preprocessor"
- PHP is a widely used, open source scripting language
- PHP scripts are executed on the server
- PHP is free to download and use.

What is PHP File ?

- PHP files can contain text, HTML, CSS, Javascript, and php code
- PHP code is executed on the server, and the result is returned to the browser as plain HTML.
- PHP files have extension ".php"

What Can PHP Do?

- PHP can generate dynamic page content
- PHP can create, open, read, write, delete, and close files on the server.
- PHP can collect form data.
- PHP can send and receive cookies
- PHP can add, delete, modify data in your database
- PHP can be used to control user-access
- PHP can encrypt data.

Why PHP?

- PHP runs on various platforms (Windows, Linux, Unix, Mac OSX, etc.)
- PHP is compatible with almost all Servers used today (Apache, IIS, etc.)
- PHP supports a wide range of databases
- PHP is free. Download it from the official PHP resource: www.php.net
- PHP is easy to learn and runs efficiently on the server side

What's new in PHP 7

- PHP 7 is much faster than the previous popular stable release (PHP 5.6)
- PHP 7 has improved Error Handling.
- PHP 7 supports stricter Type Declarations

for function arguments.

- PHP 7 supports new operators (like the spaceship operator: <=>)

PHP Syntax :

A PHP script is executed on the server, and the plain result is sent back to the browser.

Basic PHP Syntax

A PHP script can be placed anywhere in the document.

A PHP script starts with <?php and ends with ?> :

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

The default file extension for PHP files is ".php."

A PHP file normally contains HTML tags, and some PHP scripting code.

Function "echo" to output the text "Hello World!" on a web page.

DATE : / /

Example 1

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

<h1>My first PHP page</h1>

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

</body>
</html>
```

PHP Comments :

Comments in PHP

A comment in PHP code is a line that is not executed as a part of the program. Its only purpose is to be ready by someone who is looking at the code.

Comments can be used to:

- Let others understand your code
- Remind yourself of what you did - Most programmers have experienced coming

back to their own work a year or two year later and having to re-figure out what they did. Comments can remind you wrote the code.
- Leave out some parts of your code.

Example ↓

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

# This is also a single-line comment

/* This is a multi-line comment*/
```

PHP Multiline Comments :

Multi-line Comments

Multi-line comments start with /* and end with */.

Any text between /* and */ will be ignored.

Example ↓

```
/*
The next statement will
print a welcome message
*/
echo "Welcome Home!";
```

PAGE NO.:
DATE : / /

PHP Variables :

Creating PHP Variables

In PHP, a variable starts with the $ sign, followed by the name of the variable.

Example],

```
$ = 5;
$y = "John"
```

PHP variables Scope :

PHP Variables Scope

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

The scope of variable is the part of the script where the variable can be referenced/ used.

PHP has three different variable scopes:

- local
- global
- static

Global and local scope

A variable declared outside a function has a GLOBAL SCOPE and can only be accessed outside a function.

Example ↓

```
$x = 5; // global scope

Function myTest () {
    // using x inside this function will generate an
    error echo "<p> variable x inside function is:
    $x </p>";
}

myTest ();

echo "<p> variable x outside function is: $x </p>";
```

PHP echo and print Statements :

With PHP, there are two basic ways to get output: echo and print.

In this tutorial we use echo or print in almost every example. So, this chapter contains a little more info about those two output statements.

PAGE NO.:
DATE : / /

PHP echo and print Statements

echo and print are more or less the same. They are both used to output data to the screen.

The differences are small: echo has no return value while print has a return value of 1 so can be used in expressions. echo can take multiple parameters while print can take one argument. echo is marginally faster than print.

The PHP echo Statement

The echo statement can be used with or with the echo command.

Example 1

```
echo "<h2>PHP is fun!</h2>";
echo "Hello World!<br>";
echo "I'm about to learn PHP!<br>";
echo "This ", "string ", "was ", "made ", "with ",
     "Multiple parameters.";
```

PAGE NO.:
DATE : / /

PHP Data Types :

Variables can store data of different types, and different data types can do different things.

- String
- Integer
- Float (Floating point numbers - also called double)
- Boolean
- Array
- Object
- NULL
- Resource

Getting the Data Types

Get the data type of any object by using the var_dump() Function.

Example↓

The var_dump() function returns the data type and the value.

```
$x = 5;
var_dump($x);
```

PAGE NO.:
DATE : / /

PHP Strings :

A string is a sequence of characters, like "Hello World!".

Strings

Strings in PHP are surrounded by either double quotation marks, or single quotation marks.

Example ↓

```
echo "Hello";
echo 'Hello';
```

PHP Modify Strings :

php has a set of built-in functions that you can use to modify strings.

Upper case

Example ↓

The strtoupper() function returns the string in upper case.

```
$x = "Hello World!";
echo strtoupper($x);
```

PAGE NO.:
DATE : / /

Lower case

Example↓

The Strtolower() function returns the string in lower case:

```
$x = "Hello World";
echo Strtolower ($x);
```

PHP - Concatenate Strings :

String Concatenation

To concatenate, or combine, two Strings you can use the . operator.

Example↓

```
$x = "Hello";
$y = "World";
$z = $x . $y;
echo $z
```

The result of the example above is HelloWorld, Without a space between the two words.

PAGE NO.:
DATE : / /

PHP - Slicing Strings :

Slicing

You can return a range of characters by using the substr() function.

Specify the start index and the number of characters you want to return.

Example ↓

Start the slice at index 6 and end the slice 5 positions later.

```
$x = "Hello World!";
echo substr($x, 6, 5);
```

PHP - Escape characters :

Escape characters

To insert characters that are illegal in a string, use an escape character.

An escape character is a backslash \ followed by the character you want to insert.

An example of an illegal character is a double

quote inside a string that is surrounded by double quotes.

Example↓

```
$x = "We are the so-called "vikings" from the north.";
```

PHP Numbers :

There are three main numeric types in PHP.

- Integer
- Float
- Number Strings

In addition PHP has two more data types used for Numbers.

- Infinity
- NaN

Example↓

```
$a = 5;
$b = 5.34;
$c = "25";
```

PAGE NO.:
DATE : / /

PHP Casting :

Change Data Types

Casting in PHP is done With these statements.

* (String) - Converts to data type string
* (int) - Converts to data type integer
* (Float) - Converts to data type float
* (bool) - Converts to data type boolean
* (array) - Converts to data type Array
* (object) - Converts to data type object
* (unset) - Converts to data type NULL

Cast to String

Example ↓

```
$a = 5;          // Integer
$b = 5.34;       // Float
$c = "hello";    // String
$d = true;       // Boolean
$e = Null;       // NULL

$a = (String) $a;
$b = (String) $b;
$c = (String) $c;
$d = (String) $d;
$e = (String) $e;
```

To verify the type of any object in PHP, use the var_dump() function:

```
var_dump ($a);
var_dump ($b);
var_dump ($c);
var_dump ($d);
var_dump ($e);
```

PHP Math :

PHP pi() Function

The pi() function returns the value of PI.

Example → echo(pi());

PHP min() and max() Functions

The min() and max() functions can be used to find the lowest or highest value in a list of arguments.

example ↓

```
echo (min(0, 150, 30, 20, -8, -200)
echo (max(0, 150, 30, 20, -8, -200)
```

PAGE NO.:
DATE : / /

PHP Constants :

A constant is an identifier (name) for a simple value (cannot be changed during the script.

A valid constant name starts with a letter or underscore (no $ before the constant name)

Create a PHP Constant

To create a constant, use the define() function.

Syntax

```
define(name, value, case-insensitive);
```

Example↓

```
define("GREETING", "Welcome to Coders.World.Com!");
```

Const vs. define()

- Const are always case-sensitive
- define() has a case-insentive option
- Const cannot be created inside another block scope, like inside a function or inside an if statement.
- define can be created inside another block scope.

PAGE NO.:
DATE : / /

PHP Operators :

PHP operators

- Arithmetic operators
- Asighment operators
- Comparison operators
- Increment / Decrement operators
- Logical operators
- String operators
- Array Operators
- Conditional assignment operators.

PHP Arithmetic Operators

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

PAGE NO.:
DATE : / /

PHP Assignment Operators :

The PHP assignment operators are used with numeric values to write a value to a variable.

The basic assignment operator in PHP is "=". It means that the left operand gets set to the value of the assignment expression on the right.

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

PAGE NO.:
DATE : / /

x/=y	x = x /y	Division
x%=y	x = x %y	Modulus

PHP Comparison Operators

Operator	Name	Example
==	Equal	$x == $y
===	Identical	$x === $y
!=	Not equal	$x != $y
<>	Not equal	$x <> $y
!==	Not identical	$x !== $y
>	Greater than	$x > $y
<	Less than	$x < $y
>=	Greater than or equal to	$x >= $y

PAGE NO.:
DATE: / /

<=	Less than or equal to	$x <= $y
<=>	Spaceship	$x <=> $y

PHP Increment / Decrement Operators

In PHP increment operators are used to increment a variable's value.

Operator	Same as	Description
++$x	pre-increment	Increments $x by one, then returns $x
$x++	post-increment	Returns $x, then increments $x by one
--$x	pre-decrement	Decrements $x by one, then returns $x
$x--	post-decrement	Returns $x, then decrements $x by one

PAGE NO.:
DATE : / /

PHP Logical Operators

Operator	Name	Example
and	And	$x and $y
or	Or	$x or $y
xor	Xor	$x xor $y
&&	And	$x && $y
||	Or	$x || $y
!	Not	!$x

PHP string Operators

Operator	Name	Example
.	Concatenation	$txt1 . $txt2
.=	Concatenation assignment	$txt1 .= $txt2

PAGE NO.:
DATE : / /

PHP Array Operators

PHP array operators are used to compare arrays.

Operator	Name	Example
+	Union	$x + $y
==	Equality	$x == $y
===	Identity	$x === $y
!=	Inequality	$x != $y
<>	Inequality	$x <> $y
!==	Not identity	$x !== $y

PHP Conditional Assignment Operators

Operator	Name	Example
?:	Ternary	$x = expr1 ? expr2 : expr3

PAGE NO.:
DATE : / /

?? Null Coalescing

$x = expr1 ?? expr2

PHP if Statements :

PHP Conditional Statements

- if statement - executes some code if one condition is true

- if....else statement - executes some code if a condition is true and another code if that condition is false.

- if...elseif....else statement - executes different codes for more than two conditions.

- Switch statement - selects one of many blocks of code to be executed.

PHP - The if Statement

if statements executes some code if one condition is true.

Syntax

```
if (condition) {
    // Code to be executed if condition is true;
}
```

Example ↓

```
if (5>3){
   echo "Have a good day!";
}
```

PHP if Operators :

Comparision Operators

If statements usually contain conditions that compare two values.

Example ↓

check if $t is equal to 14.

```
$t =14;

if ( $t = = 14 ) {
   echo "Have a good day!";
}
```

PHP if ... else Statements :

PHP - The if ... else Statement

The if .... else statement executes some code if a condition is true and another code if that condition is false.

PAGE NO.:
DATE : / /

Syntax

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

Example ↓

```
$t = date ("H");

if ($t < "20") {
  echo "Have a good day!";
} else {
  echo "Have a good night!";
}
```

PHP - The if ... elseif ... else Statement

The if...elseif...else Statement executes different codes for more than two conditions.

Syntax

```
if (condition) {
  code to be executed if this condition is true;
} elseif (condition) {
  // code to be executed if first condition is false
     and this condition is true;
} else {
```

```
    // code to be executed if all conditions
       are false;
}
```

Example:

```
$t = date ("H");

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

PHP Shorthand if Statements:

Short Hand If

Example:

```
$a = 5;
if ($a < 10) $b = "Hello";
echo $b
```

Short Hand If ... Else

if...else statements can also be written in one line, but the syntax is a bit different.

Example ↓

```
$a = 13;

$b = $a < 10 ? "Hello" : "Good Bye";

echo $b
```

PHP Nested if Statement :

Nested If

if statements inside if statements, this is called nested If Statements.

Example ↓

```
$a = 13;
if ($a > 10) {
  echo "above 10";
 if ($a > 20) {
    echo "and also above 20";
  } else {
    echo "but not above 20";
    }
}
```

PHP Switch Statements :

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

The PHP Switch Statement

Use the switch statement to select one of many blocks of code to be executed.

Syntax

```
switch (expression){
  case label1:
    // code block
  break;
case label2:
  // code block:
    break;
case label3:
  // code block
    break;
default:
    // code block
}
```

Example

```
$favcolor = "red";
```

```
switch ($favcolor ){
  case "red":
    echo "your favorite color is red!";
    break;
 case "blue":
    "your favourite color is blue!";
    break;
  case "green":
    echo "your favorite color is green!";
    break;
  default:
    echo "your favourite color is neither red, blue,
          nor green!";
}
```

PHp Loops :

- While - loops through a block of code as long as the specified condition is true.
- do...While - loops through a block of code once, and then repeats the loop as long as the specified condition is true.
- for - loops through a block of code a specified number of times.
- foreach - loops through a block of code for each element in an array.

PHP While Loop :

The While loop - Loops through a block of code as long as the specified condition is true.

The PHP While Loop

The While loop - Loops through a block of code as long as the specified condition is true.

Example ↓

```
$i = 1;
While ($i < 6) {
 echo $i;
 $i++;
}
```

The Break Statement

With the break statement we can stop the loop even if condition is still true.

Example ↓

```
$i = 1;
While ( $i < 6 ) {
   if  $i == 3 break;
 echo $i;
 $i++;
}
```

PAGE NO.:
DATE : / /

The Continue Statement :

With the continue statement we can stop the current iteration, and continue with the next.

Example ↓

```
$i = 0;
while ($i < 6) {
  $i++;
 if ($i == 3) continue;
  echo $i;
}
```

Alternative Syntax

The while loop syntax can also be written with the endwhile statement like

Example ↓

print $i as long as $i is less than 6.

```
$i = 1;
while ($i < 6):
   echo $i;
  $i++;
endwhile
```

PHP do While Loop :

The do...While loop - Loops through a block of code once, and then repeats the loop as long as the specified condition is true.

THE PHP do.... While Loop

The do...While loop will always execute the block of code at least once, it will then check the condition, and repeat the the loop While the specified condition is true.

Example ↓

print $i as long as $i is less than 6.

```
$i = 1;

do {
  echo $i;
  $i++;
} While ($i < 6);
```

The break Statement

```
$i = 1;

do {
  if ($i == 3) break;
```

```
    echo $i;
    $i++;
} while ($i<6);
```

The Continue Statement

With the continue Statement we can stop the current iteration, and continue with the next.

Example ↓

```
$i = 0;

do{
  $i++;
  if ($i == 3) continue;
  echo $i;
} while ($i < 6);
```

PHP for Loop :

The for loop - Loops through a block of code a specified number of times.

The PHP for Loop

The for loop is used when you know how

many times the script should run.

Syntax

```
for (expression1, expression2, expression3) {
    // code block
}
```

Example ↓

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

The break statement

Break statement we can stop the loop even if the condition is still true.

Example ↓

```
for ($x=0; $x<=10; $x++) {
  if ($i==3) break;
  echo "The number is: <br>";
}
```

The Continue Statement

With the continue statement we can stop the current iteration, and continue with next:

Example ↓

```
for ($x=0; $x<=10; $x++) {
  if ($x==3) continue;
echo "The number is: $x <br>";
}
```

PHP Foreach Loop:

Foreach loop - Loops through a block of code for each element in an array or each property in an object.

The Foreach Loop on Arrays

The most common use of the foreach loop, is to loop through the items of an array.

Example ↓

```
$colors = array("red", "green", "blue", "yellow",);

foreach ($colors as $x ) {
  echo "  <br>";
}
```

Alternative Syntax

The foreach loop syntax can also be written

With the Endforeach Statement

Example:

```
$colors = array ("red", "green", "blue", "yellow",);

    foreach ( $colors as $x) :
        echo "$x <br>";
    endforeach;
```

PHP Break :

The break statement can be used to jump out of different kind of loops.

Break in for loop

The break statement can be used to jump out of a for loop

Example:

```
for ($x = 0; $x < 10; $x++) {
    if ( $x == 4 ) {
        break;
    }
    echo " The number is: $x <br>";
}
```

PHP Continue :

- The continue Statement can be used to jump out of the current iteration of a loop, and continue with the next

Continue in For Loops

The continue statement stops the current iteration in the for loop and continue with next.

Example ↓

```
for ($x = 0; $x < 10; $x++) {
  if ($x == 4) {
    continue;
  }
  echo "The number is: $x <br>";
}
```

PHP Functions :

PHP User Defined Functions

- A function is a block of statement that can be used repeatedly in a program.
- A function will not execute automatically when a page loads.

- A function will be executed by a call to the function.

Create a Function

A user-defined function declaration starts with the function, followed by the name of the function.

Example ↓

```
function mymessage() {
  echo "Hello World!";
}
```

PHp Arrays :

An array stores multiple values in one single variable.

Example ↓

```
$cars = array ("Volvo", "BMW", "Toyota",);
```

What is an array ?

An array is a special variable that can hold many values under a single name, and you can access the values by referring to an index number or name.

PHP Array Types

In PHP, there are three types of arrays.

- Indexed arrays - Arrays with a numeric index
- Associative arrays - Arrays with named keys
- Multidimensional Arrays - Arrays containing one or more arrays.

Working With arrays

In this tutorial you will learn how to work with arrays, including:

- Create Arrays
- Access Arrays
- Update Arrays
- Remove Array Items
- Sort Arrays

Array Items

Array items can be of any data type.

The most common are Strings and numbers (int, float), but array items can also be objects, functions or even arrays.

PAGE NO.:
DATE : / /

Example ↓

```
$myArr = array ("volvo", 15, ["apples", "Mango",]
                                MyFunction);
```

PHP Indexed Arrays:

~~PHP Indexed Arrays~~

In indexed arrays each items has an index number.

By default, the first item has index 0, the second item has item 1, etc.

Example ↓

```
$cars = array ("volvo", "BMW", "Toyota",);
var_dump ($cars);
```

PHP Associative Arrays :

Associative arrays are arrays that use named keys that you assign to them.

Example ↓

```
$cars = array ("brand =>" "Ford", "Mustang", "year"=>
var_dump ($car);                                1964);
```

PAGE NO.:
DATE : / /

PHP Create Arrays :

Create Array

Create arrays by using the array() function.

Example ↓

```
$cars = array("Volvo", "BMW", "Toyota");
```

Multiple Lines

Line breaks are not important, so an array declaration can span multiple lines.

Example ↓

```
$cars = [
    "Volvo",
  "BMW",
      "Toyota"
];
```

Trailling Comma

Example ↓

```
$cars = [ "Volvo", "BMW", "Toyota",
        ];
```

PAGE NO.:
DATE : / /

PHP Access Arrays :

Access Array Item

To access an array item, you can refer to the index number for indexed arrays, and the key name for associative arrays.

Example↓

```
$cars = array("Volvo", "BMW", "Toyota");
echo $cars[2];
```

PHP Update Array Items :

Update Array item

To update an existing array item, you can refer to the index number for indexed arrays, and the key name for associative arrays.

Example↓

Change the second array item from "BMW" to "Ford".

```
$cars = array("Volvo", "BMW", "Toyota");
$cars[1] = "Ford";
```

PHP Add Array Items:

Add Array Item

To add items to an existing array, you can use the bracket [ ] syntax.

Example ↓

Add one or more item to the fruits arrays.

```
$fruits = array("Apple", "Banana", "Cherry");
$fruits = ["Orange"];
```

PHP Delete Array Items:

Remove Array Item

To remove an existing item from an array, you can use the array_splice() function.

With the array_splice() function you specify the index and how many items you want to delete

Example ↓
Remove the second Item:

```
$cars = array("Volvo", "BMW", "Toyota");
array_splice($cars, 1, 1);
```

PAGE NO.:
DATE : / /

PHP Sorting Arrays :

The elements in an array can be sorted in alphabetical or numerical order, descending or ascending.

PHP - Sort Functions For Arrays

- Sort() → sort arrays in ascending order.
- rsort() → sort arrays in descending order
- asort() → Sort associative arrays in ascending order, according to the value.
- ksort() → Sort associative arrays in ascending order, according to the key.
- arsort() → Sort associative arrays in descending order, according to the value.
- krsort() → Sort associative arrays in descending order, according to the key.

Sort Array in Ascending Order - sort()

Example →
```
$cars = array("Volvo", "BMW", "Toyota");
sort($cars);
```

Sort Array in descending Order - rsort()

Example↓

```
$cars = array("Volvo", "BMW", "Toyota");
rsort($cars);
```

PHP Multidimensional Arrays :

A Multidimensional arrays that containing one or more arrays.

PHP supports multidimensional arrays that are two, three, four, five, or more levels deep. However, arrays more than three levels deep are hard to manage for most people.

PHP - Two Dimensional Arrays

A multidimensional array is an array containing one or more arrays.

PHP supports multidimensional arrays that are two, three, four, five, or more levels deep. However, arrays more than three levels deep are hard to manage for most people.

PHP Array Functions :

PHP has a set of built-in functions that you can use on arrays.

PHP $GLOBALS :

$GLOBALS is an array that contains all global variables.

Global Variables

Global variables are variables that can be accessed from any scope.

Variables of the outer most scope are automatically global variables, and can be used by any scope. eg. inside a function.

To use a global variable inside a function you have to either define them as global with the global keyword. or refer to them by using the $GLOBALS syntax.

Example 1,

Refer to the global variable $x inside a function

```
$x = 75;

function myfunction () {
    echo $GLOBALS['x'];
}

myfunction()
```

PHP - $_SERVER :

$_SERVER is a PHP Super global variable which holds information about headers, paths,

and Script locations.

Example 1,

```
echo $_SERVER['PHP_SELF'];
echo $_SERVER['SERVER_NAME'];
echo $_SERVER['HTTP_HOST'];
echo $_SERVER['HTTP_REFERER'];
echo $_SERVER['HTTP_USER_AGENT'];
echo $_SERVER['SCRIPT_NAME'];
```

PHP $_REQUEST :

$_Request

$_REQUEST is a PHP super global variable which contains submitted from data, and all Cookie data.

In other words, $_REQUEST is an array containing data from $_GET, $_POST, and $_COOKIE.

Access the data with the $_REQUEST keyword followed by the name of the form field, or cookie, like this.

```
$_REQUEST['firstname']
```

PHP file

```
$name = $_REQUEST['fname'];
  echo $name;
```

PAGE NO.:
DATE : / /

Example ↓

```
<html>
<body>

<form method = "post" action = "<?php echo $_Server
                ['PHP_SELF']; ?>">
  Name: <input type = "text" Name = "fname">
  <input type = "submit">
</form>

<?php
if ($_SERVER["REQUEST_METHOD"] == "post"){
  $name = htmlspecialchars ($_REQUEST['fname']);
  if (empty ($name)){
    echo "Name is empty";
  } else {
    echo $name;
  }
}
?>
</body>
</html>
```

PHP - $_POST :

$_POST contains an array of variables recieved via the HTTP POST method.

- HTML forms
- Javascript HTTP requests

$_POST in HTML Forms

A HTML form submits information via the HTTP POST Method if the form's Method attribute is set to "POST"

Example ↓

```
<html>
<body>

<form method = "POST" action = "<?php echo $_SERVER
                ['PHP_SELF']; ?>">
 Name: <input type = "text" name = "fname">
 <input type = "submit">
 </form>

 <?php
 if ($_SERVER["REQUEST_METHOD"] == "POST") {
   $name = htmlspecialchars ($_POST['fname']);
  if (empty ($name)) {
    echo "Name is empty";
  } else {
    echo $name;
  }
}
?>

</body>
</html>
```

PAGE NO.:
DATE : / /

$_POST in JavaScript HTTP Requests

When sending a HTTP request in Javascript, you can specify that the HTTP method is POST.

Example↓

How to post and recieve data from a HTTP request.

```
<html>
<script>
function myfunction(){
  const xhttp = new XMLHttpRequest();
  xhttp.open("POST", "demo_ajax.php");
  xhttp.setRequestHeader("Content-type", "application/
        x-www-form-urlencoded");
  xhttp.onload = function(){
    document.getElementById("demo").innerHTML=
    this.responseText;
  }xhttp.send("fname=Mary");
}
}
</script>
<body>

<button onclick="myfunction()">Click me!</button>

<h1 id="demo"></h1>
```

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

PHP Superglobal - $_GET :

PHP $_GET

$_GET contains an array of variables recieved via the HTTP GET Method.

Query String in the URL

A query string a data added at the end of a URL.

The query string above contains two key/value pairs.

```
Subject = PHP
Web = Coders.World.
```

In the PHP file use the $_GET variable to collect the value of the query string.

Example :

The php file demo_phpfile.php :

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

PAGE NO.:
DATE : / /

```
<?php
echo "Study" . $_GET['Subject']. "at": $_GET
        ['Web'];
?>

</body>
</html>
```

PHP Regular Expression :

What is Regular Expression ?
A Regular expression is a sequence of characters that forms a search pattern. When you search for data in a text. you can use this search pattern to describe what you are searching for.

A Regular expression can be a single characters or a more complicated pattern.

Regular expressions can be used to perform all types of text search and text replace operations.

Syntax

```
$exp = "/coders.World/i";
```

Using preg_match()

The preg_match() function will whether a

String contains matches of a pattern.

Example ↓

```
$str = "Visit Coders.World";
$pattern = "/Coders.World/i";
  echo preg_match ($pattern , $str);
```

PHP FORMS :

PHP Form Handling :

PHP - A Simple HTML Form

Example ↓

```
<html>
<body>

<form action = "Welcome.php"
method = "post">
Name: <input type = "text"
name = "email"><br>
Email: <input type = "text".
name = "email"><  >
<input type = "submit">
</form>
</body>
</html>
```

PAGE NO.:
DATE : / /

PHP Form Validation :

This and the next chapters show how to use PHP to validate form data.

PHP Form Validation

The HTML form we will we be working at in these chapters, contains various input fields: required and optional text fields, radio buttons, and a Submit button.

PHP Form Validation Example

* required field

Name : *

E-mail : *

Website :

Comment :

Gender : O Female O Male O Other *

Submit

Your Input :

PAGE NO.:
DATE : / /

Field	Validation Rules
Name	Required. + Must only contain letters and whitespace
E-mail	Required. + Must contain a valid email address (with @ and .)
Website	Optional. If present, it must contain a valid URL
Comment	Optional. Multi-line input field (textarea)
Gender	Required. Must select one

Example ↓

```
<?php
// define variables and set a
empty values
$name = $email = $gender =
$comment = $website = "";

if ($_SERVER["REQUEST_METHOD"] ==
        ) {
  $name =
test_input ($_post["name"]);
  $email =
test_input ($post["email"]);
  $website =
```

PAGE NO.:
DATE : / /

```
test_input($_POST["website"]);
    $comment =
test_input($_POST["comment"]);
    $gender =
test_input($_POST["gender"]);
}

function test_input($data){
    $data = trim($data);
    $data = stripslashes($data);
    $data =
htmlspecialchars($data);
        return $data;
    }
?>
```

PHP Forms - Required Fields :

PHP - Required Fields

From the validation rules table on the previous page, we see that "Name", "Email", and "Gender" fields are required.

PHP - Display The Error Messages

Example ↓

```
Name: <input type="text"
name="name">
```

```
<span class = "error"> * <?php echo
$nameErr; ?> </span>
<br> <br>
```

PHP Forms - Validate E-mail and URL :

PHP - Validate Name

The code shows a simple way to check if the name field only contains letters, dashes, apostrophes and whitespaces. If the value of the name field is not valid, then store an error message.

PHP Advanced :-

PHP Include files :

The include (or require) statement takes all the text/code/markup that exists in the specified file and copies it into in the file that uses the include statement.

Including files is very useful when you want to include the same PHP, HTML, or text on multiple pages of a website.

PAGE NO.:
DATE : / /

PHP Include and require Statements

The include and require statements are identical, except upon failure.

- require will produce a fatal error and stop the script.
- include will only produce a warning and the script will continue.

Syntax

```
include 'filename';
```

or

```
require 'filename';
```

Example

```
<html>
<body>

<h1> Welcome to my page! </h1>
<p> Some text. </p>
<p> Some more text. </p>
<?php include 'footer.php'; ?>

</body>
</html>
```

PAGE NO.:
DATE : / /

PHP File Handling :

PHP readfile() function

The readfile() function reads a file and writes it to be output buffer.

Assume We have a text file called "Webdictionary.txt", stored on the server, that looks like

AJAX = Asynchronous Javascript and XML
CSS = Cascading Style Sheets
HTML = Hype Text Markup Language
PHP = PHP Hypertext preprocessor
SQL = Structured Querry Language
SVG = Scalable vector Graphics
XML = Extensible Markup Language.

Example↓

```
<?php
echo
readfile("Webdictionary.txt");
?>
```

PAGE NO.:
DATE : / /

PHP File Open / Read / Close :

PHP Open File - Fopen()

A better Method to open files is with the Fopen() function. This function give you more options than readfile() function.

The first parameter of Fopen() contains the name of the file to be opened and the second parameter specifies in which mode the file should the opened.

Example:

```
<?php
$myfile =
Fopen("Webdictionary.txt", "r")
or die ("Unable to open file!");
echo
Fread ($myfile, filesize ("Webdictionary.
      txt" ));
fclose ($myfile);

?>
```

PHP File Create / Write :

PHP Create File - Fopen()

The fopen() function is use to create a file. Maybe a little confusing, but in PHP, a file is created using the same function used to open files.

If you use fopen() on a file that does not exists, it will create it, given that the file is opened for writing (w) or appending (a)

The example creates a new file called "testfile.txt".

Example↓

```
$myfile = fopen("testfile.txt", "w")
```

PHP File Upload :

With PHP, it is easy to upload files to the server.

However, with ease comes danges, so always be careful When allowing File uploads!

Create the HTML form

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

<form action = "upload.php"
```

```
method = "post"
enctype="mutipart/form-data">
    Select image to upload:
<input type="file"
name = "fileToUpload"
id = "fileToUpload"
<input type = "submit"
Value = "Upload Image"
name = "submit">
</form>

</body>
</html>
```

PHP Cookies:

What is a Cookie?

A Cookie is often used to identify a user. A Cookie is a small file that the server embeds on the user's computer. Each time the same computer requests a page with a browser, it will send the cookie too. With php, you can both create and retrive Cookie values.

Create Cookies With PHP

A Cookie is created with the setcookie() function

PAGE NO.:
DATE : / /

Syntax

```
setcookie ( name, value, expire,
path, domain, secure, httponly );
```

PHP Create / Retrive a Cookie

Example creates a cookie named "user" with the value "John Doe". The cookie will expire after 30 days (86400 * 30). The "/" means that the cookie is available in entire website. also use the isset() function to find out if the cookie is set.

Example ↓

```
<?php
$cookie_name = "user";;
$cookie_value = "John Doe";
setcookie ($cookie_name,
$cookie_value, time() + (86400 *
30), "/" );
?>
<html>
<body>

<?php
if ( !isset ( $_COOKIE[$cookie_name]
) ) {
  echo "Cookie named '"
```

```
$cookie_name . "' is not set!";
} else {
   echo "Cookie '" . $cookie_name
 . "' is set! <br>";
    echo value is: " .
 $:- COOKIE [$cookie_name];
}
?>

</body>
</html>
```

PHP Sessions :

What is a php Session ?
When you work with an application, you open it, do some changes, and then you close it. This is much like a Session. The Computer knows who you are. It knows when you start the application and when you end. But on the internet there is one problem. the web server does not know who you are or what you do, because the HTTP address doesn't maintain state.

Session variables solve this problem by storing user information to be used across multiple pages (e.g. username, favourite color, etc.) By default, session variables last until

PAGE NO.:
DATE : / /

the user closes the browser.

Start a PHP Session

Session is started with the Session_start() function.

Session variables are set with the PHP global variable: $_SESSION.

Example ↓

```
<?php
// Start the session
Session_start();
?>
<!DOCTYLE html>
<html>
<body>

<?php
// set session variables
$_SESSION["favcolor"] = "green";
$_SESSION["favanimal"] = "cat";
 echo "Session variables are
set."
?>

</body>
</html>
```

PAGE NO.:
DATE : / /

PHP Callback Functions :

Callback Functions

A Callback function is a function which is passed as an argument into another function.

Any existing function can be used as a callback function. To use a function as a callback function, pass a string containing the name of the function as the argument of another function.

Example ↓

pass a callback to PHP's array_map() function to calculate the lenght of every string in an array.

```
<?php
function my_callback ($item) {
    return strlen ($item);
}

$strings = ["apple", "orange",
"banana", "coconut"];
$lenghts =
array_map ("my_callback",
$strings);
print_r ($lenghts);
?>
```

PAGE NO.:
DATE : / /

PHP and JSON :

What is JSON?

JSON stands for Javascript Object Notation, and is a syntax for storing and exchanging data.

Since the JSON format is a text-based format, it can easily be sent to and form a server, and used as a data format by any programming language.

PHP and JSON

- json_encode()
- json_decode()

PHP - json_encode()

The json_encode() function is used to encode a value to JSON format

Example,

```
<?php
$age = array("peter" => 35,
"Ben" => 37, "Joe" => 43);

    json_encode($age);
?>
```

php - json_decode()

The json_decode() function is uses to decode a JSON object into a PHP object or an associative array.

Example 1,

```
<?php
$jsonobj =
'{"Peter":35,"Ben":37,"Joe":43}';

var_dump(json_decode($jsonobj));
?>
```

PHP Exceptions :

What is an exception?
An exception is an object that describes an error or unexpected behaviour of a PHP script.

Exceptions are thrown by many PHP functions and classes.

Exceptions are a good way to stop a function when it comes across data that it cannot use.

PAGE NO.:
DATE : / /

Throwing An Exception

The throw statement allows a user defined function or Method to throw an exception. When an exception is thrown, the code following it not be executed.

"Uncaught Exception"

Example ↓

```
<?php
function devide ($dividend,
$divisor) {
   if ($divisor == 0) {
      throw new Exception ("Division
by zero");
   }
   return $dividend / $divisor;
}

echo divide (5, 0);
?>
```

PHP OOP :

PHP - What is OOP ?

OOP stands for Object - Oriented programming.

PAGE NO.:
DATE : / /

procedural programming is about writing procedures or functions that perform operations on the data, while object-oriented programming is about creating that contain both data and functions.

object-oriented programming has several advantages over procedural programming.

- OOP is faster and easier to execute
- OOP provides a clear structure for the programs.
- OOP helps to keep the PHP code DRY "Don't Repeat yourself", and makes the code easier to maintain, modify and debug.
- OOP makes it possible to create full reusable applications with less code and shorter development time.

PHP - What are classes and Objects?

Classes and objects are the two main aspects of object-oriented programming.

Example ↴

Class	Objects
Fruit	Apple Banana Mango

PAGE NO.:
DATE: / /

PHP OOP - Classes and Objects :

OOP Case :

A Fruit can have properties like name, color, Weight, etc. We can define variables like $name, $color, and $weight to hold the values of these properties.

When the individual objects (apple, banana, etc.) are created, they inherit all the properties and behaviors from the class, but each object will have different values for the properties.

Define a class

A class is defined by using the class keyword, followed by the name of the class and a pair of curly braces ({}). All its properties and methods go inside the braces.

Syntax

```
<?php
class Fruit {
  // Code goes here....
}
?>
```

PAGE NO.:
DATE : / /

Define Objects

Objects of a class are created using the new keyword

Example:

```
<?php
class Fruit {
  // properties
 public $name;
 public $color;

  // Methods
 function set_name($name) {
   $this->name = $name;
 }
 function get_name() {
   return $this->name;
 }
}

$apple = new Fruit();
$banana = new Fruit();
$apple->set_name('Apple');
$banana->set_name('Banana');

 echo $apple->get_name();
 echo "<br>";
 echo $banana->get_name();
?>
```

PAGE NO.:
DATE : / /

PHP OOP - Constructor :

PHP - The __construct Function

A constructor allows you to initialize an object's properties upon creation of the object.

If you create a __construct() function, PHP will automatically call this function when you create an object from a class.

Example :

```
<?php
class Fruit {
    public $name;
    public $color;

  function __construct ($name) {
     $this -> name = $name;
  }
   function get_name () {
     return $this -> name;
   }
}

$apple =      Fruit ("Apple");
    $apple -> get_name ();
?>
```

PAGE NO.:
DATE : / /

PHP OOP - Destructor :

PHP - The __destruct Function

A destructor is called when the object is destructed or the script is stopped or exited.

If you create a __destruct() function, PHP will automatically call this function at the end of the script.

Notice that the destruct function starts with two underscores. (__)!

Example,

```
<?php
class Fruit {
    public $name;
    public $color;

  function __construct ($name) {
    $this -> name =$name;
}

  function __destruct () {
   echo "The fruit is {$this->name}.";
    }
}

$apple = new Fruit ("Apple");
?>
```

PAGE NO.:
DATE : / /

PHP OOP - Access Modifiers :

- public - the property or Method can be accessed from everywhere. This is default
- protected - the property or method can be accessed within the class and by classes derived from that class.
- private - the property or method can only be access within the class:

Example ↓

```
<?php
class Fruit {
    public $name;
    protected $color;
    private $weight;
}

$mango = new Fruit();
$mango->name = 'Mango';  // OK
$mango->color = 'yellow';  // ERROR
$mango->weight = '300';  // ERROR
?>
```

PHP OOP - Inheritance :

What is Inheritance ?
Inheritance in oop = When a class derives from another class.

An inherited class is defined by using the extends keyword.

Example:

```
<?php
class A {
    public int $prop;
}
class B extends A {
    // illegal: Read-Write -> readonly
    public readonly int $prop;
}
?>
```

PHP OOP - Interfaces:

What are Interfaces?

Interfaces allow you to specify what methods a class should implement.

Interfaces make it easy to use a variety of different classes in the same way. When one or more classes use the same interface, it is referred to as "polymorphism".

Interfaces are declared with the interface keyword.

PAGE NO.:
DATE : / /

Syntax :

```
<?php
interface InterfaceName {
    public function someMethod1 ();
    public function
 someMethod2 ($name, $color);
    public function someMethod ();
String ;
}
?>
```

PHP OOP - Static Methods :

Static methods can be called directly - without creating an instance of the class first.

Static Methods are declared with the static keyword.

Syntax

```
<?php
class ClassName {
  public static function
 staticMethod () {
    echo "Hello World!";
  }
}
?>
```

PAGE NO.:
DATE : / /

Example 7.

```
<?php
class greeting {
    public static function
welcome () {
        echo "Hello World!";
    }
}

// call static method
greeting::welcome();
?>
```

MySQL Database:

What is MySQL?

- MySQL is a database system used on the web.
- MySQL is database system that runs on a server.
- MySQL is ideal for both small and large applications.
- MySQL is very fast, reliable, and easy to use
- MySQL uses standard SQL
- MySQL compiles on a number of platforms.
- MySQL is free to download and use
- MySQL is developed, distributed, and supported by oracle corporation

PAGE NO.:
DATE : / /

Databases are useful for storing information categorically. A company may have a database with the tables.

- Employees
- Products
- Customers
- Orders

Database Queries

A query is a question or a request.

```
SELECT LastName FROM Employees
```

PHP Connect to MySQL :

- MySQLi extension (the "i" stands for improved)
- PDO (PHP Data Objects)

Example ↓

```
<?php
$servername = "localhost";
$username = "username";
$password = "password";

// Create Connection
```

PAGE NO.:
DATE: / /

```
$conn = new mysqli ($servername,
$username, $password);

  // Check connection
if ($conn -> connect_error) {
 die ("Connection failed: " .
$conn -> connect_error);
}

echo "Connected successfully";
?>
```

PHP Create a MySQL Database :

Database consists of one or more tables.

Create a MySQL Database Using MySQL cmd PDO

The CREATE DATABASE statement is used to create a database in mysql.

Example:

```
<?php
$con = mysql_connect ("localhost", "peterjohny",
       "abc124");
if (!$con)
    {
      die ('Could not connect:' . mysql_error());
}
```

```
// Some PHP code
mysql_close($con);

?>
```

PHP MySQL Create Table :

The CREATE TABLE statement is used to create a table in mysql.

Create a table named "MyGuests", with five columns: "id", "firstname", "lastname", "email" and "reg_date."

Example 1,

```
<?php
$servername = "localhost";
$username = "username";
$password = "password";
$dbname = "myDB";

// Create connection
$conn = mysqli($servername,
$username, $password, $dbname);
// Check connection
if ($conn->connect_error) {
  die("connection failed:".
$conn->connect_error);
}
?>
```

PAGE NO.:
DATE : / /

PHP MySQL Insert Data :

- The SQL query must be quoted in PHP
- String values inside the SQL query must be quoted.
- Numeric values must not be quoted
- The word NULL must not be quoted.

```
INSERT INTO table_name (column1,
column2, column3, ....)
VALUES (value1, value2,
value3, ...)
```

PHP MySQL Select Data :

Select Data from a MySQL Database

The SELECT statement is used to select data from one or more tables.

```
SELECT column_name(s) FROM
table_name
```

```
SELECT * FROM table_name
```

We can use the * character to select ALL columns from a table.

PHP MySQL Delete Data :

The DELETE statement is used to delete records from a table.

```
DELETE FROM table_name
WHERE Some_column = Some_value
```

PHP MySQL Update Data :

The UPDATE statement is used to upload existing records in table:

```
UPDATE table_name
SET Column1 = value,
Column2 = value2, ....
WHERE Some_column = some_value
```