A

Mini PROJECT REPORT

ON

WEB SITE DESIGINING & DEVELOPMENT

Submitted in partial fulfillment for the award of degree of
Bachelor of Technology
2019-2020

UNDERTAKEN AT

LOGIC TECH POINT
GEETA VATIKA, GORAKHPUR

SUBMITTED BY

PRAVESH MAURYA
ENROLLMENT NO. ITM/18/CS/53

UNDER SUPERVISION OF
MR. TANVEER SHEIKH (TRAINER)

SUBMITTED TO:
MR.VIJENDRA SIR

CANDIDATE DECLARATION

I, **Pravesh Maurya**, hereby declare that this project work entitled **WEB SITE DESIGINING & DEVELOPMENT** is my own work, carried out in **Logic Tech Point** from 10 Jun 2019 to 3 July 2019, under the external guidance of Mr. Tanveer Sheikh(Trainer).

(**PRAVESH**)

- Enroll No-ITM/18/CS/53

WEB SITE DESIGINING & DEVELOPMENT

OVERVIEW

1.HTML

- ✓ Introduction to HTML
- ✓ HTML Tags
- ✓ Creating Forms
- ✓ Managing home page

2.CSS

- ✓ Introduction to CSS
- ✓ Three ways to use CSS
- ✓ CSS Properties
- ✓ Designing Website
- ✓ Working with templates

3.Introduction To PHP

- ✓ Evaluation of PHP
- ✓ Basic syntax
- ✓ Defining variable and constant
- ✓ PHP Data types
- ✓ Operator and Expression

4.Handling HTML Form with PHP

- ✓ Capturing Form Data
- ✓ Dealing with multi-value field
- ✓ Generating File uploaded Form

- ✓ Reading a form after submission

5.Decisions and Loop

- ✓ Making Decisions
- ✓ Doing Repetitive task with looping
- ✓ Mixing Decisions and looping with HTML

6.Function

- ✓ What is Function
- ✓ Define a Function
- ✓ Call by value and call by reference
- ✓ Recursive Function

7.Strings

- ✓ Creating and accessing string
- ✓ Searching & Replacing string
- ✓ Formating string
- ✓ String Related Library Function

8.Array

- ✓ Anatomy of an Array
- ✓ Creating index based and associative array
- ✓ Accessing array Element
- ✓ Looping with Index based array
- ✓ Looping with associative array using each()and foreach()

9.Database Connectivity with MySQL

- ✓ Introduction to RDBMS
- ✓ Connection with Mysql Database
- ✓ Performing basic database operation(DML)(insert,Delete,Update,S.)
- ✓ Setting query parameter
- ✓ Executive query

10.Wordpress Introduction

- ✓ Understanding and using domain names
- ✓ Wordpress Hosting Options
- ✓ Installing Wordpress on a Dedicated Server
- ✓ Understanding Directory Permissions

11.Basic of the Wordpress User Interface

- ✓ Understanding the Wordpress Dashboard
- ✓ Pages,Tags,Media and content Administration
- ✓ Core Wordpress Settings

12.MINI PROJECT(Website with database Using HTML ,CSS,PHP &Website Through Wordpress)

HTML

1.Intoduction to HTML:

HTML is the standard markup language for creating Web pages.

- HTML stands for Hyper Text Markup Language
- HTML describes the structure of a Web page
- HTML consists of a series of elements
- HTML elements tell the browser how to display the content
- HTML elements are represented by tags
- HTML tags label pieces of content such as "heading", "paragraph", "table", and so on
- Browsers do not display the HTML tags, but use them to render the content of the page

2. CREATING TABLES:

An HTML table is defined with the `<table>` tag.

Each table row is defined with the `<tr>` tag. A table header is defined with the `<th>` tag. By default, table headings are bold and centered. A table data/cell is defined with the `<td>` tag.

Note: The `<td>` elements are the data containers of the table.
They can contain all sorts of HTML elements; text, images, lists, other tables, etc.

A border is set using the CSS `border` property:

- Use the HTML `<table>` element to define a table
- Use the HTML `<tr>` element to define a table row
- Use the HTML `<td>` element to define a table data
- Use the HTML `<th>` element to define a table heading
- Use the HTML `<caption>` element to define a table caption
- Use the CSS `border` property to define a border
- Use the CSS `border-collapse` property to collapse cell borders
- Use the CSS `padding` property to add padding to cells
- Use the CSS `text-align` property to align cell text
- Use the CSS `border-spacing` property to set the spacing between cells
- Use the `colspan` attribute to make a cell span many columns
- Use the `rowspan` attribute to make a cell span many rows
- Use the `id` attribute to uniquely define one table

3.CREATING FORM:

The HTML `<form>` element defines a form that is used to collect user input:

```
<form>
.
form elements
.
</form>
```

An HTML form contains **form elements**.

Form elements are different types of input elements, like text fields, checkboxes, radio buttons, submit buttons, and more.

The <input> Element

The `<input>` element is the most important form element.

The `<input>` element can be displayed in several ways, depending on the **type** attribute.

Here are some examples:

Type	Description
<input type="text">	Defines a one-line text input field
<input type="radio">	Defines a radio button (for selecting one of many choices)
<input type="submit">	Defines a submit button (for submitting the form)

`<input type="text">` defines a one-line input field for **text input**:

Attribute	Description
accept-charset	Specifies the charset used in the submitted form (default: the page charset).
action	Specifies an address (url) where to submit the form (default: the submitting p
autocomplete	Specifies if the browser should autocomplete the form (default: on).
enctype	Specifies the encoding of the submitted data (default: is url-encoded).
method	Specifies the HTTP method used when submitting the form (default: GET).
name	Specifies a name used to identify the form (for DOM usage: document.forms.r
novalidate	Specifies that the browser should not validate the form.
target	Specifies the target of the address in the action attribute (default: _self).

4.HTML TAGS

- All HTML elements can have **attributes**
- Attributes provide **additional information** about an element
- Attributes are always specified in **the start tag**
- Attributes usually come in name/value pairs like: **name="value"**

The href Attribute

HTML links are defined with the `<a>` tag. The link address is specified in the `href` attribute:

The src Attribute

HTML images are defined with the `<img>` tag.

- All HTML elements can have **attributes**
- The `title` attribute provides additional "tool-tip" information
- The `href` attribute provides address information for links
- The `width` and `height` attributes provide size information for images
- The `alt` attribute provides text for screen readers
- At W3Schools we always use **lowercase** attribute names
- At W3Schools we always **quote** attribute values

5.Managing homepage:

1. Open a text editor. ...
2. Set up your document type for HTML. ...
3. Add a tab title for your web page. ...
4. Indicate the beginning of your page's body text. ...
5. Create a page heading. ...
6. Add additional headings as you go. ...
7. Create a paragraph. ...
8. Change text color.

```
<!DOCTYPE html>
<html>
<head>
<title>My Web Page</title>
</head>
<h1>Welcome to My Page!</h1>
<h2>My name is Bob.</h2>
<h3>I hope you like it here.</h3>
<b>Bold text</b>
<i>Italic text</i>
<u>Underlined text</u>
<sub>Subscript text</sub>
<sup>Superscript text</sup>
```

```
<a href="https://www.facebook.com">Facebook</a>.
<img src="http://www.mypicture.com/lake.jpg">
```

CSS

1.Introduction to CSS:

- **CSS** stands for **C**ascading **S**tyle **S**heets
- CSS describes **how HTML elements are to be displayed on screen, paper, or in other media**
- CSS **saves a lot of work**. It can control the layout of multiple web pages all at once
- External stylesheets are stored in **CSS files**

CSS Example

```
body {
  background-color: lightblue;
}

h1 {
  color: white;
  text-align: center;
}

p {
  font-family: verdana;
  font-size: 20px;
}
```

The selector points to the HTML element you want to style.

The declaration block contains one or more declarations separated by semicolons.

Each declaration includes a CSS property name and a value, separated by a colon.

A CSS declaration always ends with a semicolon, and declaration blocks are surrounded by curly braces.

2.Three ways using CSS:

There are three ways of inserting a style sheet:

- External style sheet
- Internal style sheet
- Inline style

External Style Sheet

With an external style sheet, you can change the look of an entire website by changing just one file!

Each page must include a reference to the external style sheet file inside the <link> element.

Example

External styles are defined within the <link> element, inside the <head> section of an HTML page:

```
<head>
<link rel="stylesheet" type="text/css" href="mystyle.css">
</head>
```

Internal Style Sheet

An internal style sheet may be used if one single page has a unique style.

Example

Internal styles are defined within the <style> element, inside the <head> section of an HTML page:

```
<head>
<style>
body {
  background-color: linen;
}

h1 {
  color: maroon;
  margin-left: 40px;
}
</style>
</head>
```

Inline Styles

An inline style may be used to apply a unique style for a single element.

To use inline styles, add the style attribute to the relevant element. The style attribute can contain any CSS property.

Example

Inline styles are defined within the "style" attribute of the relevant element:<h1 style="color:blue;margin-left:30px;">This is a heading</h1>

3.CSS properties:

align-content	Specifies the alignment between the lines inside a flexible conta items do not use all available space
align-items	Specifies the alignment for items inside a flexible container
align-self	Specifies the alignment for selected items inside a flexible cont
all	Resets all properties (except unicode-bidi and direction)
animation	A shorthand property for all the *animation*-* properties
animation-delay	Specifies a delay for the start of an animation
animation-direction	Specifies whether an animation should be played forwards, bac alternate cycles
animation-duration	Specifies how long an animation should take to complete one c
animation-fill-mode	Specifies a style for the element when the animation is not play starts, after it ends, or both)
animation-iteration-count	Specifies the number of times an animation should be played
animation-name	Specifies a name for the @keyframes animation

animation-play-state	Specifies whether the animation is running or paused
animation-timing-function	Specifies the speed curve of an animation

B

backface-visibility	Defines whether or not the back face of an element should be v facing the user
background	A shorthand property for all the *background-** properties
background-attachment	Sets whether a background image scrolls with the rest of the p
background-blend-mode	Specifies the blending mode of each background layer (color/in
background-clip	Defines how far the background (color or image) should extend element
background-color	Specifies the background color of an element
background-image	Specifies one or more background images for an element
background-origin	Specifies the origin position of a background image
background-position	Specifies the position of a background image

background-repeat	Sets if/how a background image will be repeated
background-size	Specifies the size of the background images
border	A shorthand property for *border-width, border-style* and *borde*
border-bottom	A shorthand property for *border-bottom-width, border-bottom-style* and *border-bottom-color*
border-bottom-color	Sets the color of the bottom border
border-bottom-left-radius	Defines the radius of the border of the bottom-left corner
border-bottom-right-radius	Defines the radius of the border of the bottom-right corner
border-bottom-style	Sets the style of the bottom border
border-bottom-width	Sets the width of the bottom border
border-collapse	Sets whether table borders should collapse into a single border
border-color	Sets the color of the four borders
border-image	A shorthand property for all the *border-image-** properties

border-image-outset	Specifies the amount by which the border image area extends b border box
border-image-repeat	Specifies whether the border image should be repeated, rounde
border-image-slice	Specifies how to slice the border image
border-image-source	Specifies the path to the image to be used as a border
border-image-width	Specifies the width of the border image
border-left	A shorthand property for all the *border-left-** properties
border-left-color	Sets the color of the left border
border-left-style	Sets the style of the left border
border-left-width	Sets the width of the left border
border-radius	A shorthand property for the four *border-*-radius* properties
border-right	A shorthand property for all the *border-right-** properties
border-right-color	Sets the color of the right border

border-right-style	Sets the style of the right border
border-right-width	Sets the width of the right border
border-spacing	Sets the distance between the borders of adjacent cells
border-style	Sets the style of the four borders
border-top	A shorthand property for *border-top-width, border-top-style* an *color*
border-top-color	Sets the color of the top border
border-top-left-radius	Defines the radius of the border of the top-left corner
border-top-right-radius	Defines the radius of the border of the top-right corner
border-top-style	Sets the style of the top border
border-top-width	Sets the width of the top border
border-width	Sets the width of the four borders
bottom	Sets the elements position, from the bottom of its parent elem

box-decoration-break	Sets the behavior of the background and border of an element or, for in-line elements, at line-break.
box-shadow	Attaches one or more shadows to an element
box-sizing	Defines how the width and height of an element are calculated include padding and borders, or not
break-after	Specifies the page-, column-, or region-break behavior after th
break-before	Specifies the page-, column-, or region-break behavior before t box
break-inside	Specifies the page-, column-, or region-break behavior inside t box

C

caption-side	Specifies the placement of a table caption
caret-color	Specifies the color of the cursor (caret) in inputs, textareas, or that is editable
@charset	Specifies the character encoding used in the style sheet
clear	Specifies on which sides of an element floating elements are no float

clip	Clips an absolutely positioned element
color	Sets the color of text
column-count	Specifies the number of columns an element should be divided
column-fill	Specifies how to fill columns, balanced or not
column-gap	Specifies the gap between the columns
column-rule	A shorthand property for all the *column-rule-** properties
column-rule-color	Specifies the color of the rule between columns
column-rule-style	Specifies the style of the rule between columns
column-rule-width	Specifies the width of the rule between columns
column-span	Specifies how many columns an element should span across
column-width	Specifies the column width
columns	A shorthand property for *column-width* and *column-count*

content	Used with the :before and :after pseudo-elements, to insert ge
counter-increment	Increases or decreases the value of one or more CSS counters
counter-reset	Creates or resets one or more CSS counters
cursor	Specifies the mouse cursor to be displayed when pointing over

D

direction	Specifies the text direction/writing direction
display	Specifies how a certain HTML element should be displayed

E

empty-cells	Specifies whether or not to display borders and background on empty cells

F

filter	Defines effects (e.g. blurring or color shifting) on an element before the el displayed
flex	A shorthand property for the *flex-grow, flex-shrink*, and the *flex-basis* prop

flex-basis	Specifies the initial length of a flexible item
flex-direction	Specifies the direction of the flexible items
flex-flow	A shorthand property for the *flex-direction* and the *flex-wrap* properties
flex-grow	Specifies how much the item will grow relative to the rest
flex-shrink	Specifies how the item will shrink relative to the rest
flex-wrap	Specifies whether the flexible items should wrap or not
float	Specifies whether or not a box should float
font	A shorthand property for the *font-style, font-variant, font-weight, font-size* the *font-family* properties
@font-face	A rule that allows websites to download and use fonts other than the "web
font-family	Specifies the font family for text
font-feature-settings	Allows control over advanced typographic features in OpenType fonts

@font-feature-values	Allows authors to use a common name in font-variant-alternate for feature differently in OpenType
font-kerning	Controls the usage of the kerning information (how letters are spaced)
font-language-override	Controls the usage of language-specific glyphs in a typeface
font-size	Specifies the font size of text
font-size-adjust	Preserves the readability of text when font fallback occurs
font-stretch	Selects a normal, condensed, or expanded face from a font family
font-style	Specifies the font style for text
font-synthesis	Controls which missing typefaces (bold or italic) may be synthesized by the
font-variant	Specifies whether or not a text should be displayed in a small-caps font
font-variant-alternates	Controls the usage of alternate glyphs associated to alternative names defi feature-values
font-variant-caps	Controls the usage of alternate glyphs for capital letters

font-variant-east-asian	Controls the usage of alternate glyphs for East Asian scripts (e.g Japanese
font-variant-ligatures	Controls which ligatures and contextual forms are used in textual content c applies to
font-variant-numeric	Controls the usage of alternate glyphs for numbers, fractions, and ordinal m
font-variant-position	Controls the usage of alternate glyphs of smaller size positioned as superso regarding the baseline of the font
font-weight	Specifies the weight of a font

G

grid	A shorthand property for the *grid-template-rows, grid-template-columns, g areas, grid-auto-rows, grid-auto-columns*, and the *grid-auto-flow*properties
grid-area	Either specifies a name for the grid item, or this property is a shorthand pr the *grid-row-start*, *grid-column-start*, *grid-row-end*, and *grid-column-end*p
grid-auto-columns	Specifies a default column size
grid-auto-flow	Specifies how auto-placed items are inserted in the grid
grid-auto-rows	Specifies a default row size

grid-column	A shorthand property for the *grid-column-start* and the *grid-column-end*pr
grid-column-end	Specifies where to end the grid item
grid-column-gap	Specifies the size of the gap between columns
grid-column-start	Specifies where to start the grid item
grid-gap	A shorthand property for the *grid-row-gap* and *grid-column-gap* properties
grid-row	A shorthand property for the *grid-row-start* and the *grid-row-end* propertie
grid-row-end	Specifies where to end the grid item
grid-row-gap	Specifies the size of the gap between rows
grid-row-start	Specifies where to start the grid item
grid-template	A shorthand property for the *grid-template-rows*, *grid-template-columns* a *areas* properties
grid-template-areas	Specifies how to display columns and rows, using named grid items
grid-template-columns	Specifies the size of the columns, and how many columns in a grid layout

grid-template-rows	Specifies the size of the rows in a grid layout

H

hanging-punctuation	Specifies whether a punctuation character may be placed outside the line b
height	Sets the height of an element
hyphens	Sets how to split words to improve the layout of paragraphs

I

image-rendering	Gives a hint to the browser about what aspects of an image are to preserve when the image is scaled
@import	Allows you to import a style sheet into another style sheet
isolation	Defines whether an element must create a new stacking conter

J

justify-content	Specifies the alignment between the items inside a flexible container when not use all available space

K

@keyframes	Specifies the animation code

L

left	Specifies the left position of a positioned element
letter-spacing	Increases or decreases the space between characters in a text
line-break	Specifies how/if to break lines
line-height	Sets the line height
list-style	Sets all the properties for a list in one declaration
list-style-image	Specifies an image as the list-item marker
list-style-position	Specifies the position of the list-item markers (bullet points)
list-style-type	Specifies the type of list-item marker

M

margin	Sets all the margin properties in one declaration

margin-bottom	Sets the bottom margin of an element
margin-left	Sets the left margin of an element
margin-right	Sets the right margin of an element
margin-top	Sets the top margin of an element
max-height	Sets the maximum height of an element
max-width	Sets the maximum width of an element
@media	Sets the style rules for different media types/devices/sizes
min-height	Sets the minimum height of an element
min-width	Sets the minimum width of an element
mix-blend-mode	Specifies how an element's content should blend with its direct parent back

O

object-fit	Specifies how the contents of a replaced element should be fitted to the bo its used height and width

object-position	Specifies the alignment of the replaced element inside its box
opacity	Sets the opacity level for an element
order	Sets the order of the flexible item, relative to the rest
orphans	Sets the minimum number of lines that must be left at the bottom of a pag break occurs inside an element
outline	A shorthand property for the *outline-width, outline-style*, and the *outline-c*
outline-color	Sets the color of an outline
outline-offset	Offsets an outline, and draws it beyond the border edge
outline-style	Sets the style of an outline
outline-width	Sets the width of an outline
overflow	Specifies what happens if content overflows an element's box
overflow-wrap	Specifies whether or not the browser may break lines within words in order overflow (when a string is too long to fit its containing box)

overflow-x	Specifies whether or not to clip the left/right edges of the content, if it ove element's content area
overflow-y	Specifies whether or not to clip the top/bottom edges of the content, if it o element's content area

P

padding	A shorthand property for all the *padding*-* properties
padding-bottom	Sets the bottom padding of an element
padding-left	Sets the left padding of an element
padding-right	Sets the right padding of an element
padding-top	Sets the top padding of an element
page-break-after	Sets the page-break behavior after an element
page-break-before	Sets the page-break behavior before an element
page-break-inside	Sets the page-break behavior inside an element
perspective	Gives a 3D-positioned element some perspective

perspective-origin	Defines at which position the user is looking at the 3D-positioned element
pointer-events	Defines whether or not an element reacts to pointer events
position	Specifies the type of positioning method used for an element (static, relativ fixed)

Q

quotes	Sets the type of quotation marks for embedded quotations

R

resize	Defines if (and how) an element is resizable by the user
right	Specifies the right position of a positioned element

S

scroll-behavior	Specifies whether to smoothly animate the scroll position in a s instead of a straight jump

T

tab-size	Specifies the width of a tab character

table-layout	Defines the algorithm used to lay out table cells, rows, and colu
text-align	Specifies the horizontal alignment of text
text-align-last	Describes how the last line of a block or a line right before a fo aligned when text-align is "justify"
text-combine-upright	Specifies the combination of multiple characters into the space character
text-decoration	Specifies the decoration added to text
text-decoration-color	Specifies the color of the text-decoration
text-decoration-line	Specifies the type of line in a text-decoration
text-decoration-style	Specifies the style of the line in a text decoration
text-indent	Specifies the indentation of the first line in a text-block
text-justify	Specifies the justification method used when text-align is "justi
text-orientation	Defines the orientation of the text in a line
text-overflow	Specifies what should happen when text overflows the containi

text-shadow	Adds shadow to text
text-transform	Controls the capitalization of text
text-underline-position	Specifies the position of the underline which is set using the tex property
top	Specifies the top position of a positioned element
transform	Applies a 2D or 3D transformation to an element
transform-origin	Allows you to change the position on transformed elements
transform-style	Specifies how nested elements are rendered in 3D space
transition	A shorthand property for all the *transition*-* properties
transition-delay	Specifies when the transition effect will start
transition-duration	Specifies how many seconds or milliseconds a transition effect complete
transition-property	Specifies the name of the CSS property the transition effect is f
transition-timing-function	Specifies the speed curve of the transition effect

U

unicode-bidi	Used together with the direction property to set or return whet should be overridden to support multiple languages in the sam
user-select	Specifies whether the text of an element can be selected

V

vertical-align	Sets the vertical alignment of an element
visibility	Specifies whether or not an element is visible

W

white-space	Specifies how white-space inside an element is handled
widows	Sets the minimum number of lines that must be left at the top a page break occurs inside an element
width	Sets the width of an element
word-break	Specifies how words should break when reaching the end of a l
word-spacing	Increases or decreases the space between words in a text

word-wrap	Allows long, unbreakable words to be broken and wrap to the n
writing-mode	Specifies whether lines of text are laid out horizontally or vertic

Z

z-index	Sets the stack order of a positioned element

4.Designing Website:

Website Layout

A website is often divided into headers, menus, content and a footer:

There are tons of different layout designs to choose from. However, the structure above, is one of the most common, and we will take a closer look at it in this tutorial.

Header
Navigation Menu
Content Main Content Content
Footer

Header

A header is usually located at the top of the website (or right below a top navigation menu). It often contains a logo or the website name:

Example

```
.header {
  background-color: #F1F1F1;
  text-align: center;
  padding: 20px;

      }
```

5.Working with templates:

W3.CSS Website Templates

We have created some responsive W3.CSS website templates for you to use.

You are free to modify, save, share, and use them in all your projects.

Introductin to php

1.Evaluation of 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
- 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
- PHP runs on various platforms (Windows, Linux, Unix, Mac OS X, 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

2.Basic 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.

Below, we have an example of a simple PHP file, with a PHP script that uses a built-in PHP function "echo" to output the text "Hello World!" on a web page:

Example

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

<h1>My first PHP page</h1>

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

</body>
</html>
```

Note: PHP statements end with a semicolon (;).

3.Defining variable and consonant:

What is Variable in PHP

Variables are used to store data, like string of text, numbers, etc. Variable values can change over the course of a script. Here're some important things to know about variables:

- In PHP, a variable does not need to be declared before adding a value to it. PHP automatically converts the variable to the correct data type, depending on its value.
- After declaring a variable it can be reused throughout the code.
- The assignment operator (=) used to assign value to a variable.

  In PHP variable can be declared as: `$var_name = value;`

4.php data types:

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

PHP supports the following data types:

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

5.operator and Expressions:

Operators are used to perform operations on variables and values.

PHP divides the operators in the following groups:

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

PHP Arithmetic Operators

The PHP arithmetic operators are used with numeric values to perform common arithmetical operations, such as addition, subtraction, multiplication etc.

Operator	Name	Example	Result
+	Addition	$x + $y	Sum of $x and $y
-	Subtraction	$x - $y	Difference of $x and $y
*	Multiplication	$x * $y	Product of $x and $y
/	Division	$x / $y	Quotient of $x and $y
%	Modulus	$x % $y	Remainder of $x divided by $y
**	Exponentiation	$x ** $y	Result of raising $x to the $y'th in PHP 5.6)

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 th

x += y	x = x + y	Addition	
x -= y	x = x - y	Subtraction	
x *= y	x = x * y	Multiplication	
x /= y	x = x / y	Division	
x %= y	x = x % y	Modulus	

Operator	**Name**	**Example**	**Result**
==	Equal	$x == $y	Returns true if $x is equal
===	Identical	$x === $y	Returns true if $x is equal of the same type
!=	Not equal	$x != $y	Returns true if $x is not e
<>	Not equal	$x <> $y	Returns true if $x is not e

!==	Not identical	$x !== $y	Returns true if $x is not e are not of the same type
>	Greater than	$x > $y	Returns true if $x is great
<	Less than	$x < $y	Returns true if $x is less t
>=	Greater than or equal to	$x >= $y	Returns true if $x is great
<=	Less than or equal to	$x <= $y	Returns true if $x is less t
++$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	

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

PHP has two operators that are specially designed for strings.

Operator	Name	Example	Result
.	Concatenation	$txt1 . $txt2	Concatenatio $txt2
.=	Concatenation assignment	$txt1 .= $txt2	Appends $txt

PHP Array Operators

The PHP array operators are used to compare arrays.

Operator	Name	Example	Result
+	Union	$x + $y	Union of $x and $y
==	Equality	$x == $y	Returns true if $x and $y have the same ke
===	Identity	$x === $y	Returns true if $x and $y have the same ke same order and of the same types
!=	Inequality	$x != $y	Returns true if $x is not equal to $y
<>	Inequality	$x <> $y	Returns true if $x is not equal to $y

!==	Non-identity	$x !== $y	Returns true if $x is not identical to $y

Handling HTML form with PHP

1.Capturing form Data:

Capturing Form Data with PHP

- A superglobal is a built-in PHP variable that is available in any scope:
  - at the top level of your script,
  - within a function, or
  - within a class method.
- There are three superglobal arrays for using form data:

  $_GET

  Contains a list of all the field **names** and **values** sent by a form using the get method

  $_POST

  Contains a list of all the field **names** and **values** sent by a form using the post method

  $_REQUEST

  Contains the values of both the $_GET and $_POST arrays combined, along with the values of the $_COOKIE superglobal array

Example

First name Last name

Are you male... ...or female? Submit

```
<form action="" method="get">
    <label for="firstName">First
name</label>
    <input type="text" name="firstName"
id="firstName" value="" />

    <label for="lastName">Last name</label>
    <input type="text" name="lastName"
id="lastName" value="" />

    <label for="genderMale">Are you
male...</label>
    <input type="radio" name="gender"
id="genderMale" value="M" />
    <label for="genderFemale">...or
female?</label>
    <input type="radio" name="gender"
id="genderFemale" value="F" />

    <input type="submit" value="Submit"
id="moveRight">
</form>
```

Show one case of data as below:
Valerie Chu is a female.
	<?php
	
	$first=$_GET["firstName"];
	$last=$_GET["lastName"];
	$sex=$_GET["gender"];
	
	if($sex == "F")
	   echo "$first $last is a female.
	";
	else
	   echo "$first $last is a male.
	";
	?>

2.Dealing with multi valued field:

Dealing with Multi-Value Fields

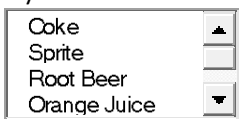
Treat the field as a nested array in the superglobal arrays

What are your favorite soft drink?

Coke Sprite Root Beer Orange Juice

Apple Juice Water

What are your favorite soft drink?



Note that hold a **ctrl** key to choose more than one item.

Submit Reset

```
<form
action="http://itech.loc.edu/~chu/itec415
/multiValue.php" method="post">
<h2>What are your favorite soft
drink?</h2>
<label>Coke</label>
<input type="checkbox" name="drink[]"
value="coke" />
<label>Sprite</label>
<input type="checkbox" name="drink[]"
value="sprite" />
<label>Root Beer</label>
<input type="checkbox" name="drink[]"
value="root beer" />
<label>Orange Juice</label>
<input type="checkbox" name="drink[]"
value="orange juice" />
<label>Apple Juice</label>
<input type="checkbox" name="drink[]"
value="apple juice" />
<label>Water</label>
<input type="checkbox" name="drink[]"
value="water" />
<h2>What are your favorite soft
drink?</h2>
<select name="favor[]" size = "4"
multiple = "multiple">
  <option value="coke">Coke</option>
  <option value="sprite">Sprite</option>
  <option value="root beer">Root
Beer</option>
  <option value="orange juice">Orange
Juice</option>
  <option value="apple juice">Apple
Juice</option>
  <option value="water">Water</option>
</select>
<p>Note that hold a <b>ctrl</b> key to
choose more than one item.</p>
<input type="submit" name="submit"
id="moveRight" value="Submit" />
<input type="reset" name="reset"
value="Reset" style="margin-left:
20px;display:inline;" />
</form>
```

One case of output is followed:

Your favorite drink from checkboxes are
coke
root beer
apple juice
water

Your favorite drink from pull-down menu are
sprite
orange juice
water

```
<?php
$drinklist=$_POST["drink"]; //assign an
array to a local array
$favorlist=$_POST["favor"];

echo "Your favorite drink from checkboxes
are <br />";
foreach($drinklist as $drink)
     echo $drink . "<br /> ";
echo "<br />";

echo "Your favorite drink from pull-down
menu are <br /> ";
foreach($favorlist as $favor)
     echo $favor. "<br />";
```

	?>

Another format of output by regular expression to take off a comma and space at the end.
Your favorite drink from checkboxes are sprite, root beer,
 	
Your favorite drink from pull-down menu are sprite, oran
	<?php
	$drinklist=$_POST["drink"];
	$favorlist=$_POST["favor"];
	$drinkOutput=""; //Initialize an empty string for output
	$favorOutput="";
	
	foreach($drinklist as $drink)
	$drinkOutput .= $drink . ", ";
	//accumulate output
	$drinkOutput = preg_replace("/, $/", ".", $drinkOutput);//Take off last comma
	
	echo "Your favorite drink from checkboxes are ".$drinkOutput."
	
	";
	
	foreach($favorlist as $favor)
	$favorOutput .= $favor. ", ";
	$favorOutput = preg_replace("/, $/", ".", $favorOutput);
	
	echo "Your favorite drink from pull-down menu are ".$favorOutput."
	";
	?>

4.Generating a fille uploaded form:

n your "php.ini" file, search for the `file_uploads` directive, and set it to On:

```
file_uploads = On
```

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

<form action="upload.php" method="post" enctype="multipart/form-data">
    Select image to upload:
    <input type="file" name="fileToUpload" id="fileToUpload">
    <input type="submit" value="Upload Image" name="submit">
</form>

</body>
</html>
```

5.Redirecting after a form submission:

```
 <?php

if(isset($_POST['submit'])){
// Fetching variables of the form which travels in URL
$name = $_POST['name'];
$email = $_POST['email'];
$contact = $_POST['contact'];
$address = $_POST['address'];
if($name !=''&& $email !=''&& $contact !=''&& $address !='')
{
//  To redirect form on a particular page
header("Location:https://www.formget.com/app/");
}
else{
?><span><?php echo "Please fill all fields.....!!!!!!!!!!!!";?></span> <?php
}
}
?>
```

Decisions and loop

1.Making Decisions:

- **if...else statement** – use this statement if you want to execute a set of code when a condition is true and another if the condition is not true
- **elseif statement** – is used with the if...else statement to execute a set of code if **one** of the several condition is true
- **switch statement** – is used if you want to select one of many blocks of code to be executed, use the Switch statement. The switch statement is used to avoid long blocks of if..elseif..else code.

```
if (condition)
   code to be executed if condition is true;
else
   code to be executed if condition is false;
Else..if
if (condition)
   code to be executed if condition is true;
elseif (condition)
   code to be executed if condition is true;
else
   code to be executed if condition is false;
```

swtch

```
switch (expression){
   case label1:
      code to be executed if expression = label1;
      break;

```

```
    case label2:
       code to be executed if expression = label2;
       break;
       default:
```

Doing Repetative task with looping:

PHP supports four different types of loops.
While — loops through a block of code until the condition is evaluate to true.
Do…While — the block of code executed once and then condition is evaluated. If the condition is true the statement is repeated as long as the specified condition is true.
For — loops through a block of code until the counter reaches a specified number.
Foreach — loops through a block of code for each element in an array.
You will also learn how to loop through the values of array using foreach() loop at the end of this chapter. The foreach() loop work specifically with arrays.
while(condition){
// Code to be executed
}

do{
// Code to be executed
}
while(condition);

for(initialization; condition; increment){
// Code to be executed
}

PHP foreach Loop
The foreach loop is used to iterate over arrays.
foreach($array as $value){
// Code to be executed
}

3.Mixing decisions with html and php:

```
<?php if(conditions) { ?>
... HTML CODE ...
<?php } ?>
<?php if(conditions): ?>
... HTML CODE ...
<?php endif; ?>

<?php while(conditions) : ?>
... HTML CODE ...
<?php endwhile; ?>
```

FUNCTION

1.What is a function:

PHP functions are similar to other programming languages. A function is a piece of code which takes one more input in the form of parameter and does some processing and returns a value.

You already have seen many functions like **fopen()** and **fread()** etc. They are built-in functions but PHP gives you option to create your own functions as well.

There are two parts which should be clear to you –

- Creating a PHP Function
- Calling a PHP Function

2.Define a function:

Creating and Invoking Functions

The basic syntax of creating a custom function can be give with:

```
function functionName(){
    // Code to be executed
}
```

Example

Run this code »

```
<?php
// Defining function
function whatIsToday(){
    echo "Today is " . date('l', mktime());
}
// Calling function
whatIsToday();
?>
```

3.call by value and call by reference:

Call by value means passing the value directly to a function. The called function uses the value in a local variable; any changes to it **do not** affect the source variable.

```php
<?php
//Call by value program
function abc($x)
{
$x=$x+10;
return($x);
}
$a=20;
echo abc($a)."<br>";
echo ($a);
?>
```

Note: Call by value: in the call by value method, the called function creates a new set of variables and copies the values of arguments into them.

Output:

Call by reference means passing the address of a variable where the actual value is stored. The called function uses the value stored in the passed address; any changes to it **do** affect the source variable.

```php
<?php
//call by reference program in php
function abc($x)
{
$x=$x-10;
return($x);
}
$a=50;
echo abc($a)."<br>";
echo ($a);
?>
```

Note: Call by reference: in the call by reference method, instead of passing a value to the function being called a reference/pointer to the original variable is passed.

Output:

4.recursive function:

HP also supports recursive function call like C/C++. In such case, we call current function within function. It is also known as recursion.

It is recommended to avoid recursive function call over 200 recursion level because it may smash the stack and may cause the termination of script.

Example 1: Printing number

```php
<?php
function display($number) {
    if($number<=5){
     echo "$number <br/>";
     display($number+1);
    }
```

```
7. }
8.
9. display(1);
10. ?>
```

STRINGS

1.Creating and accessing string:

As you learned in previous section, creating a string variable is as simple as assigning a literal string value to a new variable name.
Both single and double quote are used to create string.

Syntax

$var=”shubhneet”;
$my_variable = ” hello ” ;

Illustrate With Example

```
<?php
$myvariable = " Welcome to php string " ;
echo "$myvariable "."<br>";
echo 'Example of php';
?>
\
```

To access a character at a particular position.
Syntax
$char = $str [position] ;
String Example

```
<?php
$myStr = "Welcome to the php string";
echo $myStr[0] ."<br>"; // print "W"
echo $myStr[6] ."<br>"; // print "e"
$myStr[25] = '?';//Welcome to the php string?
echo $myStr ."<br>";
?>
```

2.Searching and repacing:

PHP str_replace() Function

‹ PHP String Reference

Example

Replace the characters "world" in the string "Hello world!" with "Peter":

```
<?php
echo str_replace("world","Peter","Hello world!");
?>
```

Syntax

`str_replace(find,replace,string,count)`

Parameter	Description
Find	Required. Specifies the value to find
Replace	Required. Specifies the value to replace the value in *find*
String	Required. Specifies the string to be searched
Count	Optional. A variable that counts the number of replacements

3.Formating string:

Formatting Strings

There's a pair of string functions that are particularly useful when you want to format data for display (such as when you're formatting numbers in string form): `printf` and `sprintf`. The `printf` function echoes text directly, and you assign the return value of `sprintf` to a string. Here's how you use these functions (items in square brackets, `[` and `]`, in function specifications like this one are optional):

```
printf (format [, args])

sprintf (format [, args])
```

4.string related library function:

PHP string functions are used to manipulate string values.

We are now going to look at some of the commonly used string functions in PHP

Function	Description	Example	Output
strtolower	Used to convert all string characters to lower case letters	echo strtolower('Benjamin');	outputs benjamin
strtoupper	Used to convert all string characters to upper case letters	echo strtoupper('george w bush');	outputs GEORGE W BUSH
strlen	The string length function is used to count the number of character in a string. Spaces in between characters are also counted	echo strlen('united states of america');	24
explode	Used to convert strings into an array variable	$settings = explode(';', "host=localhost; db=sales; uid=root; pwd=demo"); print_r($settings);	Array ([0] => host=localhost [1] => db=sales [2] => uid=root [3] => pwd=demo)
substr	Used to return part of the string. It accepts three (3) basic parameters. The first one is the string to be shortened, the second parameter is the position of the starting point, and the third parameter is the number of characters to be returned.	$my_var = 'This is a really long sentence that I wish to cut short';echo substr($my_var,0, 12).'...';	This is a re...
str_replace	Used to locate and replace specified string values in a given string. The function accepts three arguments. The first argument is the text to be replaced, the second argument is the replacement text and the third argument is the text that is analyzed.	echo str_replace ('the', 'that', 'the laptop is very expensive');	that laptop is very expensive
strpos	Used to locate the and return the position of a	echo strpos('PHP Programing','Pro');	4

Function	Description	Example	Output
	character(s) within a string. This function accepts two arguments		
sha1	Used to calculate the SHA-1 hash of a string value	echo sha1('password');	5baa61e4c 9b93f3f0 682250b6cf8331b 7ee68fd8
md5	Used to calculate the md5 hash of a string value	echo md5('password');	9f961034ee 4de758 baf4de09ceeb1a75
str_word_count	Used to count the number of words in a string.	echo str_word_count ('This is a really long sentence that I wish to cut short');	12
ucfirst	Make the first character of a string value upper case	echo ucfirst('respect');	Outputs Respect
lcfirst	Make the first character of a string value lower case	echo lcfirst('RESPECT');	Outputs rESPECT

Array

1.Anotomy of an array:

An array is a data structure that stores one or more similar type of values in a single value. For example if you want to store 100 numbers then instead of defining 100 variables its easy to define an array of 100 length.
There are three different kind of arrays and each array value is accessed using an ID c which is called array index.
Numeric array − An array with a numeric index. Values are stored and accessed in linear fashion.
Associative array − An array with strings as index. This stores element values in association with key values rather than in a strict linear index order.

2.Creating index based and associative array:

There are basically three types of arrays in PHP:

- **Indexed or Numeric Arrays:** An array with a numeric index where values are stored linearly.
- **Associative Arrays:** An array with a string index where instead of linear storage, each value can be assigned a specific key.
- **Multidimensional Arrays:** An array which contains single or multiple array within it and can be accessed via multiple indices.

Indexed or Numeric Arrays

These type of arrays can be used to store any type of elements, but an index is always a number. By default, the index starts at zero. These arrays can be created in two different ways as shown in the following example:

```
<?php

// One way to create an indexed array
$name_one = array("Zack", "Anthony", "Ram", "Salim", "Raghav");

// Accessing the elements directly
echo "Accessing the 1st array elements directly:\n";
echo $name_one[2], "\n";
echo $name_one[0], "\n";
echo $name_one[4], "\n";

// Second way to create an indexed array
$name_two[0] = "ZACK";
$name_two[1] = "ANTHONY";
$name_two[2] = "RAM";
$name_two[3] = "SALIM";
$name_two[4] = "RAGHAV";

// Accessing the elements directly
echo "Accessing the 2nd array elements directly:\n";
echo $name_two[2], "\n";
echo $name_two[0], "\n";
```

```
echo $name_two[4], "\n";
```

Associative Arrays

These type of arrays are similar to the indexed arrays but instead of linear storage, every value can be assigned with a user-defined key of string type.

Example:

filter_none

edit

play_arrow

brightness_4

```
<?php

// One way to create an associative array
$name_one = array("Zack"=>"Zara", "Anthony"=>"Any",
                  "Ram"=>"Rani", "Salim"=>"Sara",
                  "Raghav"=>"Ravina");

// Second way to create an associative array
$name_two["zack"] = "zara";
$name_two["anthony"] = "any";
$name_two["ram"] = "rani";
$name_two["salim"] = "sara";
$name_two["raghav"] = "ravina";

// Accessing the elements directly
echo "Accessing the elements directly:\n";
echo $name_two["zack"], "\n";
echo $name_two["salim"], "\n";
echo $name_two["anthony"], "\n";
echo $name_one["Ram"], "\n";
echo $name_one["Raghav"], "\n";

?>

?>
```

3.Accessing array elements:

```
<?php
$pantry = array(
  1 => "apples",
  2 => "oranges",
  3 => "bananas"
  );
$findit = $pantry[2];
echo "The value of findit is $findit.";
?>
<?php
$pantry = array(
  1 => "apples",
  2 => "oranges",
  3 => "bananas"
```

```
  );
$findit = $pantry[2];
echo "The value of findit is $findit.";
?>
```

4.Looping with index based *array:*

You already know you can loop over an array using a `for` loop and the `count` function, which determines how many elements an array contains:

```
<?php

    $fruits[0] = "pineapple";

    $fruits[1] = "pomegranate";

    $fruits[2] = "tangerine";

    for ($index = 0; $index < count($fruits); $index++){



   echo $fruits[$index], "\n";



   }

?>
```

5.Each()and foreach()function:

The foreach loop is mainly used for looping through the values of an array. It loops over the array, and each value for the current array element is assigned to $value, and the array pointer is advanced by one to go the next element in the array.

Syntax:

```
<?php
```

```
foreach (array as $value){
```

```
  //code to be executed;

}

?>
```

Database connectivity withMySQL

1.Introduction To Rdbms:

- MySQL is a database system used on the web
- MySQL is a 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
- MySQL is named after co-founder Monty Widenius's daughter: My
- The normal forms are used to ensure that various types of anomalies and inconsistencies are not **introduced** into the database. **RDBMS** stands for **Relational**Database Management System. **RDBMS** data is structured in database tables, fields and records. Each **RDBMS** table consists of database table rows.

2.Connection with mysql database:

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

// Create connection
$conn = new mysqli($servername, $username, $password);

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

echo "Connected successfully";
?>
```

Create MySQL Database at the Localhost

1. Create **Database**. Now return to the homepage of phpmyadmin. ...
2. Create a Folder in htdocs. ...
3. Create **Database Connection** File In **PHP**. ...
4. Create new **php** file to check your **database connection**. ...
5. Run it! ...
6. Create **Database Connection**. ...
7. MySQLi Procedural Query. ...
8. **Connect MySQL Database** with **PHP** Using PDO.

3.Database operation:

SQL | DDL, DML, DCL and TCL Commands

Structured Query Language(SQL) as we all know is the database language by the use of which we can perform certain operations on the existing database and also we can use this language to create a database. SQL uses certain commands like Create, Drop, Insert etc. to carry out the required tasks.

These SQL commands are mainly categorized into four categories as discussed below:

1. **DDL(Data Definition Language) :** DDL or Data Definition Language actually consists of the SQL commands that can be used to define the database schema. It simply deals with descriptions of the database schema and is used to create and modify the structure of database objects in database.
   Examples of DDL commands:
   - **CREATE** – is used to create the database or its objects (like table, index, function, views, store procedure and triggers).
   - **DROP** – is used to delete objects from the database.
   - **ALTER**-is used to alter the structure of the database.
   - **TRUNCATE**–is used to remove all records from a table, including all spaces allocated for the records are removed.
   - **COMMENT** –is used to add comments to the data dictionary.
   - **RENAME** –is used to rename an object existing in the database.
2. **DML(Data Manipulation Language) :** The SQL commands that deals with the manipulation of data present in database belong to DML or Data Manipulation Language and this includes most of the SQL statements.
   Examples of DML:
   - **SELECT** – is used to retrieve data from the a database.
   - **INSERT** – is used to insert data into a table.
   - **UPDATE** – is used to update existing data within a table.
   - **DELETE** – is used to delete records from a database table.
3. **DCL(Data Control Language) :** DCL includes commands such as GRANT and REVOKE which mainly deals with the rights, permissions and other controls of the database system.
   Examples of DCL commands:
   - **GRANT**-gives user's access privileges to database.
   - **REVOKE**-withdraw user's access privileges given by using the GRANT command.
4. **TCL(transaction Control Language) :** TCL commands deals with the transaction within the database.
   Examples of TCL commands:
   - **COMMIT**– commits a Transaction.
   - **ROLLBACK**– rollbacks a transaction in case of any error occurs.
   - **SAVEPOINT**–sets a savepoint within a transaction.
   - **SET TRANSACTION**–specify characteristics for the transaction.

4.Setting query operations:

You can execute a MySQL query towards a given database by opening the database with phpMyAdmin and then clicking on the **SQL**tab. A new page will load, where you can provide the desired query. When ready click the **Go** button to perform the execution.
The page will refresh and you will see the results from the query you provided.

5.Executing query:

Execute MySQL queries with the SQL tab

You can **execute** a **MySQL query** towards a given database by opening the database with phpMyAdmin and then clicking on the SQL tab. A new page will load, where you can provide the desired **query**. When ready click the Go button to perform the **execution**.

1. Navigate to the area your SQL query will apply to. The phpMyAdmin home page if you want the query to apply to the whole server or hosting account. The database you want to run queries against. ...
2. Click on the SQL tab.
3. Type in your SQL query.
4. Click on Go to execute the query.

In relational database management systems, a **query** is any command used to retrieve data from a table. In Structured **Query** Language (SQL), **queries** are almost always made using the SELECT **statement**.
... **MySQL** is an open-source relational database management system.

Wordpress Introduction

1.Understanding and using domain name:

In wordpress we create a database name in phpmyadmin ,there create username and

Your domain name is a key part of your online address, and is what your visitors will use to find you easily. Your domain name is unique to you; once you have registered it, nobody else can register the same one for as long as you continue to renew it.

Domain names are used to identify one or more IP addresses. For example, the domain name wordpress.org represents about a dozen IP addresses. Domain names are used in URLs to identify particular Web pages. For example, in the
URL `https://d5creation.com` or `www.d5creation.com` or `https://www.d5creation.com`, the domain name is d5creation.com.
WordPress is the world's most popular tool for creating websites. WordPress is capable of creating any kind of website, from a simple blog to a full-featured business website. You can even use WordPress to create an online store (using the popular WooCommerce plugin).

You say you've never built a website? That's not a problem! With WordPress you don't need any coding or design skills t

2.Wordpress hosting options:

Without further ado, here are the 10 best web hosting services for WordPressin 2019:

1. Bluehost (www.Bluehost.com) ...
2. HostGator Cloud (www.HostGator.com) ...
3. SiteGround (www.SiteGround.com) ...
4. A2 **Hosting** (www.A2Hosting.com) ...
5. Site5 **Hosting** (www.Site5.com) ...
6. iPage (www.iPage.com) ...
7. Dreamhost (www.Dreamhost)

XAMPP server is a best web hoster. Wordpress can be esily hosted on xampp server.

3.Installing Wordpress on a dedicated server:

- ✓ First we download wordpress from wordpress.org
- ✓ Zip file extract
- ✓ Copy the file on xampp-htdocs-folder
- ✓ Go to chrome browser
- ✓ Localhost-foldername and run the wordpress

4.Understanding directory permission:

wp-config.php

It is one of the cores **WordPress** files which contains information about the database, including the name, host (typically localhost), username, and password. There are many other folders and files, but these are the most important folders and files in the**WordPress directory** structure.

The WordPress file structure is honestly pretty simple at the higher levels. You have your *public_html* folder, where its three key folders are located, as well as a lot of important files such as *wp-config.php* and *.htaccess*.

The *wp-admin* Folder

The *wp-content* Folder

This is the section of the back end where you're likely to spend most of your time during the course of your relationship with WordPress. Its two most popular functions are located inside – of course, we're talking about themes and plugins:

Each plugin you upload to WordPress will have its own subfolder within the plugins folder, as seen in the example above. The contents of each of these vary from plugin to plugin. Here, for example, is a quick look inside the Akismet plugin's folder:

As with plugins, each theme you install on your WordPress site gets its own corresponding folder on the back end, which you've probably already seen unless you've been uploading every theme through the dashboard rather than using FTP.

The *wp-includes* Folder

1. Get acquainted with the WordPress directory structure, especially the *wp-admin*, *wp-content*, and *wp-includes* folders.
2. Familiarize yourself with the WordPress core files, including *wp-config.php*, *functions.php*, and *.htaccess*.

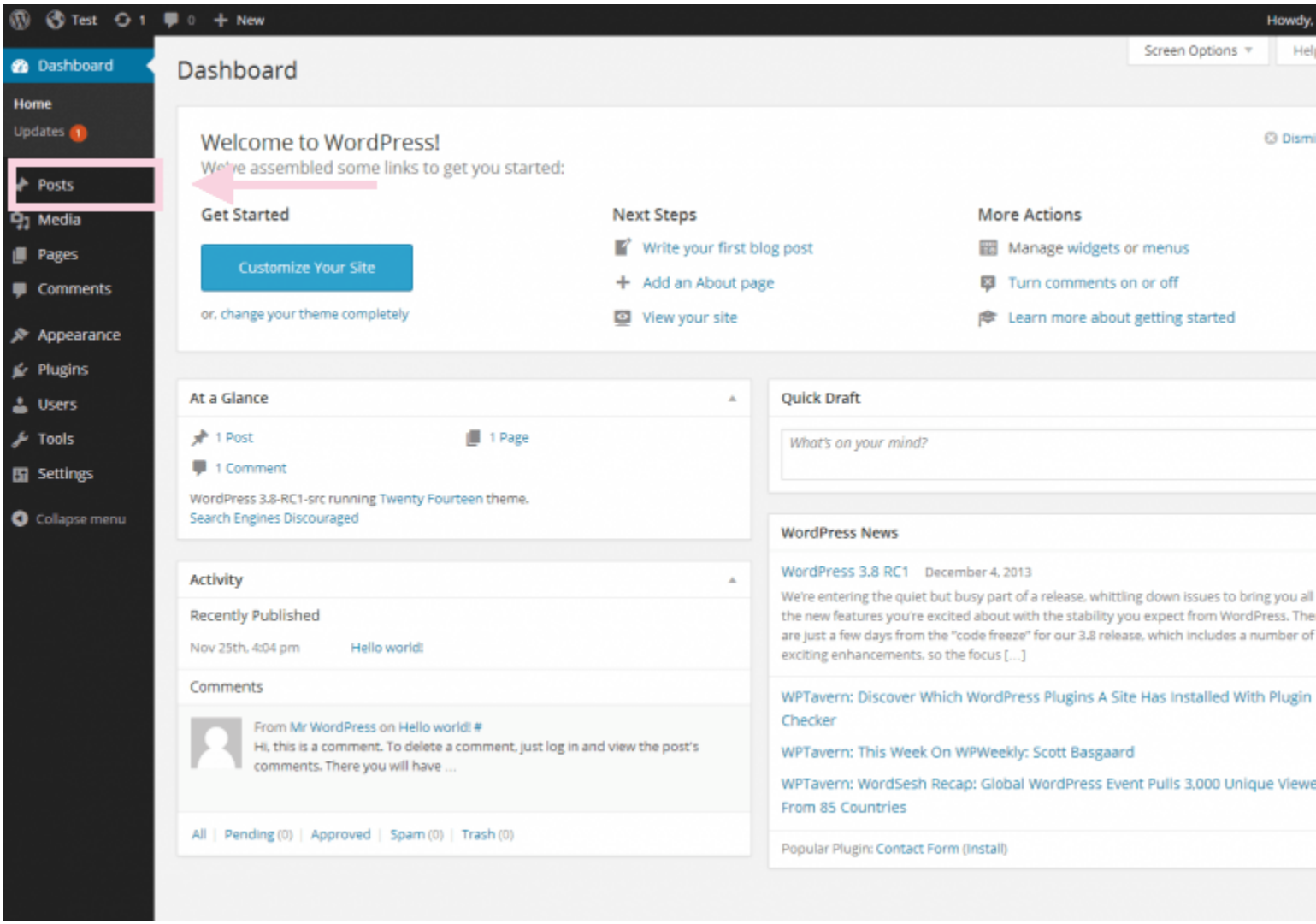
Basics of the Wordpress User Interface

1.Understanding the wordpress dashboard:

When you first log into **WordPress**, your **dashboard**will look similar to the screenshot above. This is your**WordPress dashboard**. You can edit what is on your**dashboard** by clicking on screen options at the top. Then, a dropdown will appear where you can select what you want to see on your **dashboard**.

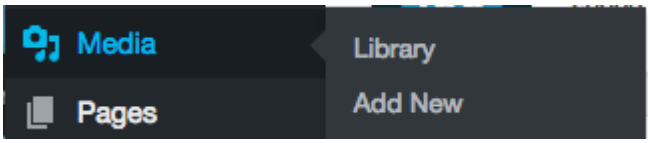
Logging in

Posts



Posts are where your blog posts will be located. You can select to view all posts, add a new post, view categories or tags.

Media



Next up is media. This is where any of your photos, videos, PDFs, and more are located. If you are just starting out, your library will be empty.

Pages are next after media. Pages are the main anchors of your website. Here are a few of our pages:

Then you will see comments. These include all of the comments that have been left on your blog. If you are brand new, you won't have any comments yet.

The appearance option is where people can start to get confused. This is where any of your themes and anything related to the appearance of your website is hosted.

Plugins are ways to extend and add to the functionality that already exists in WordPress. You can learn more about plugins here.

If you need to have multiple users with various login permissions, users is where you can add new or edit your own profile.

Tools contain options to perform some non-routine management tasks. As a beginner, you most likely won't need to use this option.

General settings is where you can edit your site title, tagline, URL, contact email address, timezone and more.

Writing is where you can edit your default categories and blog formats.

Reading is where you can edit what the front page displays (either your latest posts or a static page), how many blog posts are shown, and how much of each blog article you want to be shown.

Media is where you can determine the maximum dimensions in pixels to use when adding an image to the Media Library.

2.Pages,tags media and content administration:

To add **tags** to a new post, go to your blog's admin area > Posts > Add New. When you write your new post, you can add a **tag** to it by typing the **tag** word in the **Tags**field on the right and clicking the Add button. You can add as many **tags** as you want.
Following are the simple steps to **Edit Tags in WordPress**. Step (1) − Click on Posts → **Tags in WordPress**. **Edit** − Click on **Edit** option in **Tags** section as shown in the following screen. You can **edit** any of the required field, and then click on Update button as shown in the following screen.

3.Core wordpress setting:

- The .htaccess and wp-config.php files. The .htaccess and wp-config.php files are part of every WordPress installation's top (or main) directory. ...
- The wp-admin folder. The wp-admin folder houses most of the files that power your WordPress dashboard. ...
- The wp-includes folder.

WordPress stores uploaded images and **media** in the file system, but pages and posts are **stored** in the MYSQL database. A **WordPress** installation creates several folders where it stores system **files**. Plugins, Themes and uploaded **media** are all**stored** under the wp_contents folder.

1. From within your WordPress Dashboard, go to Media, then Add New.
2. Click the Select Files button.
3. Locate the image on your computer, and double click it.
4. Once the image is finished uploading, click on the Edit link.
5. The Image URL is in the File URL box on the right side of the screen.

First, open a post or a **page** to which you would like to **add** your **media**. Click your cursor on the area of the post or **page** where you would like the **media** to show. Next, click the **Add Media** button in the top left corner of your toolbar: And click **Add** New to upload new **media** to your **website.**

MINI PROJECT

- ✓ A College website using HTML ,CSS & PHP MYSQL,
- ✓ A e-commerce website using wordpress