# PHP INTRODUCTION

PHP is a server side scripting language designed for web development but also used aux general purpose programming language PHP is now installed on more than 244 million websites and 2.1 million web servers Originally created by Rasmus Lerdorf in 1995 the reference implementation of PHP now produced by the PHP Group. While PHP originally stood for Personal Home Page it now stands for PHP Hypertext Pre-processor, a recursive acronym. PHP code is interpreted by a web server with a PHP processor module, which generates the resulting web page: PHP commands can be embedded directly into an HTML, source document rather than calling an external file to process data. It has also evolved to include a command line interface capability and can be used in standalone graphical applications PHP is free software released under the PHP License, which is incompatible with the GNU General Public License

(GPL) due to restrictions on the usage of the term PHP can be deployed on must web servers and also as a standalone shelf on almost every operating system and platform five of charge PHP is an acronym for "PHP Hypertext Pre-processor PHP is a widely used open source scripting language PHP scripts are executed on the server PHP costs nothing, it is free to download and use.

PHP is a scripting language it gets interpreted instead of being compiled like C++ and Java Unlike JavaScript which is executed by the web browser, all PHP code is executed on the web server. The syntax is very similar to Perl and C .Variables are case sensitive, function names are not, and statements must be terminated with a semicolon. PHP code should be placed between< ?Code?>or<? Php code? > tags. The second method is preferred so your scripts are XML compatible. There is no limitation as to where PHP code can be inserted. To see information about how PHP is configured, version information, and the settings of all environment variables (eg HTTP\_USER\_AGENT and QUERY\_STRING), call the phpInfo () function in any script. The php.ini file is the main configuration file for PHP. It can be edited by the system administrator to change any of the configuration settings. A change to this file requires the web server be restarted since the file is only read once when the web server starts up.

**DAY 1**

**HTML**

* **Introduction to HTML**

**What is HTML?**

* HTML stands for Hyper Text Markup Language
* HTML is the standard markup language for Web pages
* HTML **elements** are the building blocks of HTML pages

**Structure of HTML Document**

* Start with <!DOCTYPE html>
* Begin with <html> and end with </html>
* Visible part between <body> and </body>
* The header part between <head> and </head> which includes Styles and Scripts

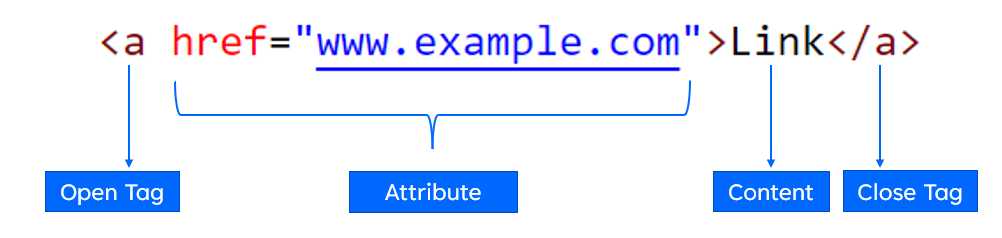
**DAY 2**

* **Tags, Elements and Attributes**

**Tags and Elements**

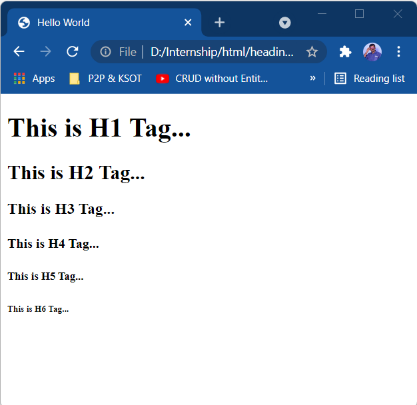
* Tags are like keywords which define that how the browser will format and display the contents
* The HTML Elements contains Three parts
  + - Open tag
    - Contents
    - Close Tags

**Attributes**

****

**DAY 3**

* **HTML Heading Element**





**HTML Text Formatting**

**Example Code**

<!DOCTYPE html><html>

<head>

<title>HTML Formatting Tags</title>

</head>

<body>

<h2>HTML Formatting Tags</h2>

This is <b>Bold text</b>...<br>

This is <strong>Important text</strong>...<br>

This is <i>Italic text</i>...<br>

This is <em>Emphasized text</em>...<br>

This is <mark>Marked text</mark>...<br>

This is <small>Smaller text</small>...<br>

This is <del>Deleted text</del>...<br>

This is <ins>Inserted text</ins>...<br>

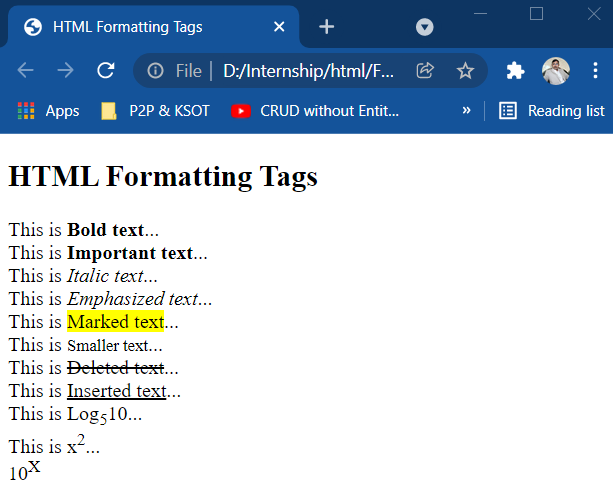
This is Log<sub>5</sub>10...<br>

This is x<sup>2</sup>...<br>

10<sup>X</sup>

</body>

</html>

****

**DAY 4**

* **HTML List**
* HTML lists allow web developers to group a set of related items in lists.

**Unordered HTML List**

* An unordered list starts with the <ul> tag. Each list item starts with the <li> tag.
* The list items will be marked with bullets (small black circles) by default:

**Ordered HTML List**

* An ordered list starts with the <ol> tag. Each list item starts with the <li> tag.
* The list items will be marked with numbers by default:

**HTML Image Size - Width and Height**

**Example Code**

<!DOCTYPE html>

<html>

<head>

<title>Hello World</title>

</head>

<body>

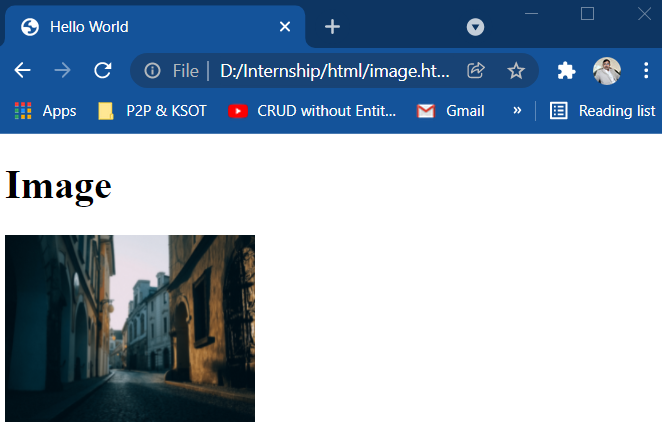
<h1>Image</h1>

<img src="image/photo1.png" alt="Photo 1" width="200px" height="150px">

</body>

</html>

**Output:**

****

**DAY 5**

* **HTML Tables**
* HTML tables allow web developers to arrange data into rows and columns.
* A table in HTML consists of table cells inside rows and columns

**HTML Links**

* Links are found in nearly all web pages. Links allow users to click their way from page to page.
* HTML links are hyperlinks.
  + You can click on a link and jump to another document.
  + When you move the mouse over a link, the mouse arrow will turn into a little hand.

**DAY 6**

* **HTML Forms**
* An HTML form is used to collect user input. The user input is most often sent to a server for processing.

The <form> Element

* The HTML <form> element is used to create an HTML form for user input:

**The <input> Element**

* The HTML <input> element is the most used form element.
* An <input> element can be displayed in many ways, depending on the type attribute.

**DAY 7**

* **BOOTSTRAP**

**Introduction**

Get started with Bootstrap, the world’s most popular framework for building responsive, mobile-first sites, with Bootstrap CDN and a template starter page.

Bootstrap is developed mobile first, a strategy in which we optimize code for mobile devices first and then scale up components as necessary using CSS media queries. To ensure proper rendering and touch zooming for all devices, add the responsive viewport meta tag to your <head>.

**Layout**

* **Containers**

Containers are the most basic layout element in Bootstrap and are required when using our default grid system. Choose from a responsive, fixed-width container (meaning its max-width changes at each breakpoint) or fluid-width (meaning it’s 100% wide all the time).

**DAY 8**

* **Grid** **system**

Use our powerful mobile-first flexbox grid to build layouts of all shapes and sizes thanks to a twelve column system, five default responsive tiers, Sass variables and mixins, and dozens of predefined classes.

Bootstrap’s grid system uses a series of containers, rows, and columns to layout and align content. It’s built with flexbox and is fully responsive.

* **Grid options**

While Bootstrap uses ems or rems for defining most sizes, pxs are used for grid breakpoints and container widths. This is because the viewport width is in pixels and does not change with the font size.

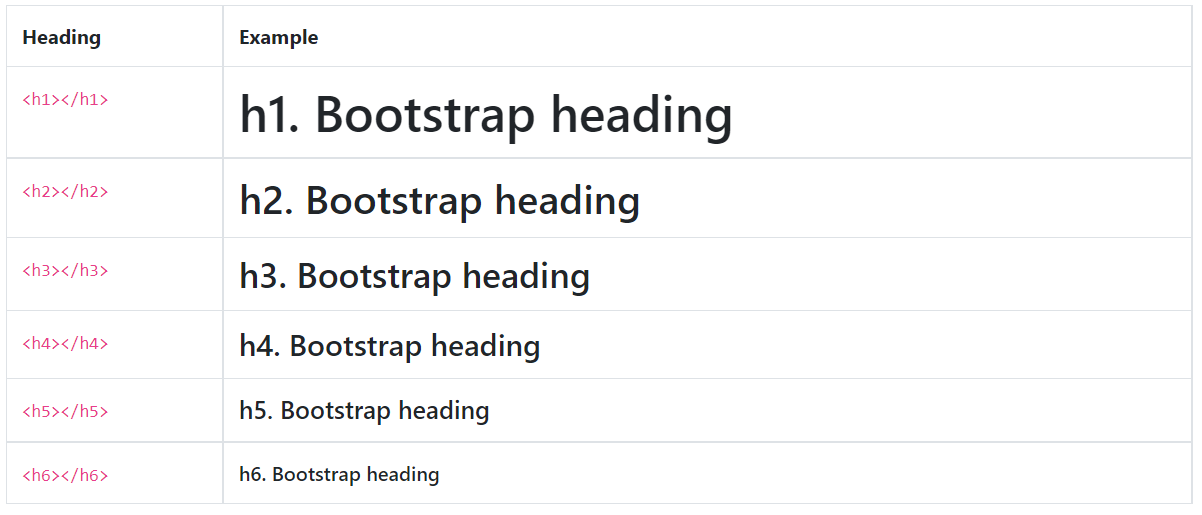
**DAY 9**

**Typography**

* **Headings**

All HTML headings, <h1> through <h6>, are available.

.h1 through .h6 classes are also available, for when you want to match the font styling of a heading but cannot use the associated HTML element.



* **Images**

Documentation and examples for opting images into responsive behavior (so they never become larger than their parent elements) and add lightweight styles to them—all via classes.

* **Table**

Using the most basic table markup, here’s how .table-based tables look in Bootstrap. All table styles are inherited in Bootstrap 4, meaning any nested tables will be styled in the same manner as the parent.

* **Alerts**

Provide contextual feedback messages for typical user actions with the handful of available and flexible alert messages.

<div class="alert alert-primary" role="alert">

A simple primary alert—check it out!

</div>

<div class="alert alert-secondary" role="alert">

A simple secondary alert—check it out!

</div>

<div class="alert alert-success" role="alert">

A simple success alert—check it out!

</div>

<div class="alert alert-danger" role="alert">

A simple danger alert—check it out!

</div>

<div class="alert alert-warning" role="alert">

A simple warning alert—check it out!

</div>

<div class="alert alert-info" role="alert">

A simple info alert—check it out!

</div>

<div class="alert alert-light" role="alert">

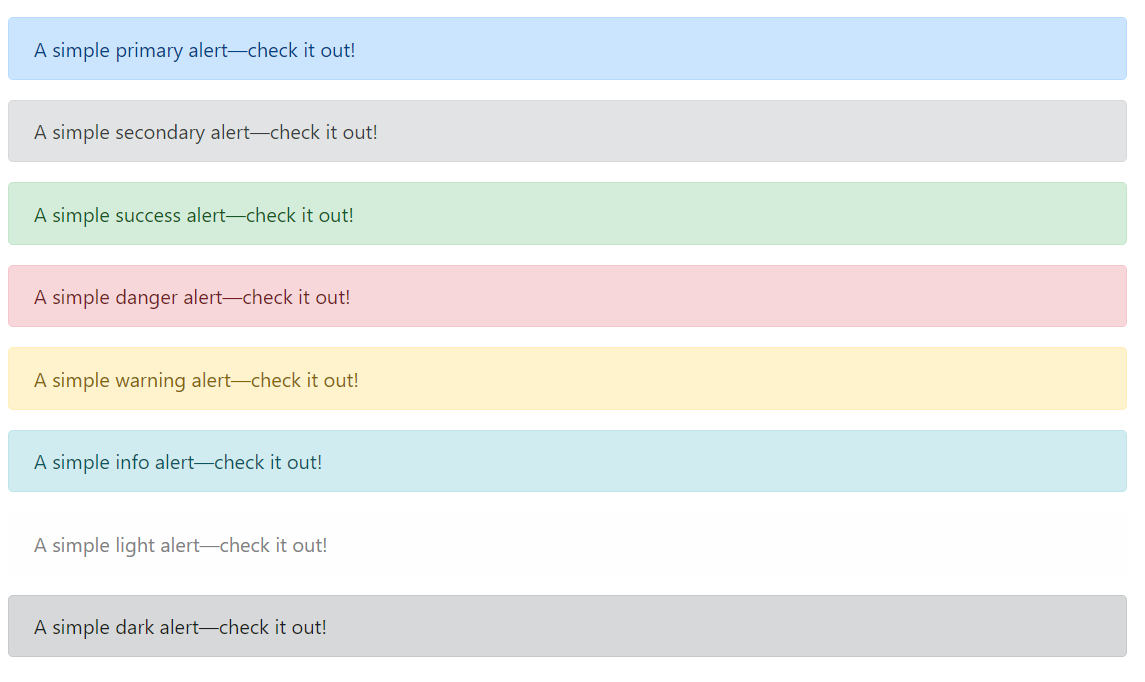
A simple light alert—check it out!

</div>

<div class="alert alert-dark" role="alert">

A simple dark alert—check it out!

</div>



**DAY 10**

* **Badges and Buttons**

**Badges**

Documentation and examples for badges, our small count and labeling component. Badges scale to match the size of the immediate parent element by using relative font sizing and em units**Buttons**

Use Bootstrap’s custom button styles for actions in forms, dialogs, and more with support for multiple sizes, states, and more.

Bootstrap includes several predefined button styles, each serving its own semantic purpose, with a few extras thrown in for more control.

**Cards**

Bootstrap’s cards provide a flexible and extensible content container with multiple variants and options.

<div class="card" style="width: 18rem;">

<img src="..." class="card-img-top" alt="...">

<div class="card-body">

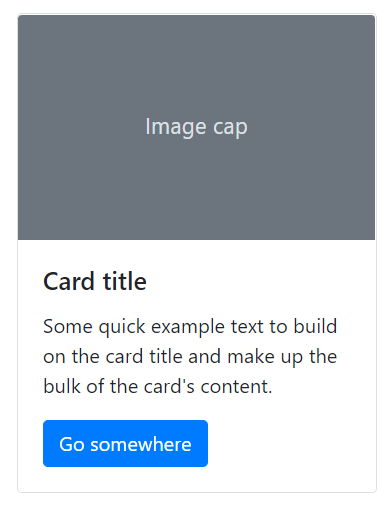
<h5 class="card-title">Card title</h5>

<p class="card-text">Some quick example text to build on the card title and make up the bulk of the card's content.</p>

<a href="#" class="btn btn-primary">Go somewhere</a>

</div>

</div>



**DAY 11**

* **Carousels**

Carousels don’t automatically normalize slide dimensions. As such, you may need to use additional utilities or custom styles to appropriately size content. While carousels support previous/next controls and indicators, they’re not explicitly required. Add and customize as you see fit.

The .active class needs to be added to one of the slides otherwise the carousel will not be visible. Also be sure to set a unique id on the .carousel for optional controls, especially if you’re using multiple carousels on a single page. Control and indicator elements must have a data-target attribute (or href for links) that matches the id of the .carousel element.

**DAY 12**

* **Forms**

Examples and usage guidelines for form control styles, layout options, and custom components for creating a wide variety of forms. Bootstrap’s form controls expand on our Rebooted form styles with classes. Use these classes to opt into their customized displays for a more consistent rendering across browsers and devices.

Be sure to use an appropriate type attribute on all inputs (e.g., email for email address or number for numerical information) to take advantage of newer input controls like email verification, number selection, and more.

Here’s a quick example to demonstrate Bootstrap’s form styles. Keep reading for documentation on required classes, form layout, and more.

<form>

<div class="form-group">

<label for="exampleInputEmail1">Email address</label>

<input type="email" class="form-control" id="exampleInputEmail1" aria-describedby="emailHelp">

<small id="emailHelp" class="form-text text-muted">We'll never share your email with anyone else.</small>

</div>

<div class="form-group">

<label for="exampleInputPassword1">Password</label>

<input type="password" class="form-control" id="exampleInputPassword1">

</div>

<div class="form-group form-check">

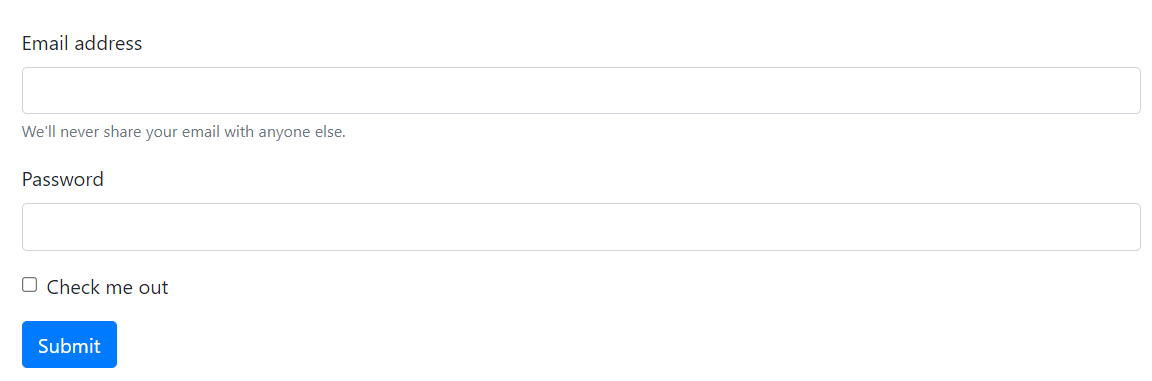
<input type="checkbox" class="form-check-input" id="exampleCheck1">

<label class="form-check-label" for="exampleCheck1">Check me out</label>

</div>

<button type="submit" class="btn btn-primary">Submit</button>

</form>



**DAY 13**

* **Navbar**

Documentation and examples for Bootstrap’s powerful, responsive navigation header, the navbar. Includes support for branding, navigation, collapse plugin, and more.

* **How it works**

Here’s what you need to know before getting started with the navbar:

* Navbars require a wrapping .navbar with .navbar-expand{-sm|-md|-lg|-xl} for responsive collapsing and color scheme classes.
* Navbars and their contents are fluid by default. Use optional containers to limit their horizontal width.
* Use our spacing and flex utility classes for controlling spacing and alignment within navbars.
* Navbars are responsive by default, but you can easily modify them to change that. Responsive behavior depends on our Collapse JavaScript plugin.
* Navbars are hidden by default when printing. Force them to be printed by adding .d-print to the .navbar. See the display utility class.
* Ensure accessibility by using a <nav> element or, if using a more generic element such as a <div>, add a role="navigation" to every navbar to explicitly identify it as a landmark region for users of assistive technologies.

**DAY 14**

* **PHP**

**Introduction to PHP**

* PHP is a server side scripting language, and a powerful tool for making dynamic and interactive Web pages.
* PHP code is executed on the server.
* PHP case sensitive.

**What is PHP**

* PHP stands 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 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

**PHP Syntax**

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



**DAY 15**

* **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 read by someone who is looking at the code.
* PHP Variables

A variable can have a short name (like x and y) or a more descriptive name (age, carname, total\_volume).

Rules for PHP variables:

* A variable starts with the $ sign, followed by the name of the variable
* A variable name must start with a letter or the underscore character
* A variable name cannot start with a number
* A variable name can only contain alpha-numeric characters and underscores (A-z, 0-9, and \_ )
* Variable names are case-sensitive ($age and $AGE are two different variables)

**DAY 16**

* **PHP Datatypes**
* 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
* **PHP String**
  + **strlen() - Return the Length of a String**
  + **str\_word\_count() - Count Words in a String**
  + **strrev() - Reverse a String**
  + **strpos() - Search For a Text Within a String**
  + **str\_replace() - Replace Text Within a String**
  + **str\_replace() - Replace Text Within a String**
  + **PHP explode() Function**
  + **PHP str\_ireplace() Function**
  + **PHP md5() Function**
  + **PHP strcmp() and strcasecmp() Function**
  + **PHP trim() Function**

**DAY 17**

* **PHP Conditional Statements**

In PHP we have the following 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

**DAY 18**

* **PHP Loops**

In PHP, we have the following loop types:

* 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

**DAY 19**

* **PHP Operators**

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

**DAY 20**

* **PHP Arrays**

An array stores multiple values in one single variable:

* **Create an Array in PHP**

In PHP, the array() function is used to create an array:

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

<?php  
$cars = array("Volvo", "BMW", "Toyota");  
echo "I like " . $cars[0] . ", " . $cars[1] . " and " . $cars[2] . ".";  
?>

* **PHP Indexed Arrays**

There are two ways to create indexed arrays:

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

The index can be assigned automatically (index always starts at 0), like this:

$cars[0] = "Volvo";

$cars[1] = "BMW";

$cars[2] = "Toyota";

* **PHP Associative Arrays**

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

<?php  
$age = array("Peter"=>"35", "Ben"=>"37", "Joe"=>"43");  
echo "Peter is " . $age['Peter'] . " years old.";  
?>

* **PHP - Multidimensional 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.

$cars = array (

array("Volvo",22,18),

array("BMW",15,13),

array("Saab",5,2),

array("Land Rover",17,15)

);

**DAY 21**

* **PHP - A Simple HTML Form**

The example below displays a simple HTML form with two input fields and a submit button:

<html>  
<body>  
<form action="welcome.php" method="post">  
Name: <input type="text" name="name"><br>  
E-mail: <input type="text" name="email"><br>  
<input type="submit">  
</form>  
</body>  
</html>

When the user fills out the form above and clicks the submit button, the form data is sent for processing to a PHP file named "welcome.php". The form data is sent with the HTTP POST method.

To display the submitted data you could simply echo all the variables. The "welcome.php" looks like this:

<html>  
<body>  
Welcome <?php echo $\_POST["name"]; ?><br>  
Your email address is: <?php echo $\_POST["email"]; ?>  
</body>  
</html>

The same result could also be achieved using the HTTP GET method:

<html>  
<body>  
<form action="welcome\_get.php" method="get">  
Name: <input type="text" name="name"><br>  
E-mail: <input type="text" name="email"><br>  
<input type="submit">  
</form>  
</body>  
</html>

and "welcome\_get.php" looks like this:

<html>

<body>

Welcome <?php echo $\_GET["name"]; ?><br>

Your email address is: <?php echo $\_GET["email"]; ?>

</body>

</html>

The code above is quite simple. However, the most important thing is missing. You need to validate form data to protect your script from malicious code.

**GET vs. POST**

$\_GET is an array of variables passed to the current script via the URL parameters.

$\_POST is an array of variables passed to the current script via the HTTP POST method.

**DAY 22**

* **PHP MySQL Database**

With PHP, you can connect to and manipulate databases. MySQL is the most popular database system used with PHP

* **What is MySQL?**
* 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 Database**

**Create Database**

The CREATE DATABASE statement is used to create a new SQL database.

**Syntax**

***CREATE DATABASE databasename;***

**Example**

CREATE DATABASE testDB;

**Create Table**

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

**Syntax**

***CREATE TABLE table\_name (***

***column1 datatype,***

***column2 datatype,***

***column3 datatype,***

***....***

***);***

**DAY 23**

* **MySQL ALTER TABLE Statement**

The ALTER TABLE statement is used to add, delete, or modify columns in an existing table.

The ALTER TABLE statement is also used to add and drop various constraints on an existing table.

* **ALTER TABLE - ADD Column**

To add a column in a table, use the following syntax:

***ALTER TABLE table\_name***

***ADD column\_name datatype;***

The following SQL adds an "Email" column to the "Customers" table:

**Example**

ALTER TABLE Customers

ADD Email varchar(255);

* **ALTER TABLE - DROP COLUMN**

To delete a column in a table, use the following syntax (notice that some database systems don't allow deleting a column):

***ALTER TABLE table\_name***

***DROP COLUMN column\_name;***

The following SQL deletes the "Email" column from the "Customers" table:

**Example**

ALTER TABLE Customers

DROP COLUMN Email;

* **ALTER TABLE - MODIFY COLUMN**

To change the data type of a column in a table, use the following syntax:

***ALTER TABLE table\_name***

***MODIFY COLUMN column\_name datatype;***

**Example**

ALTER TABLE Persons

MODIFY COLUMN DateOfBirth year;

**DAY 24**

* **MySQL PRIMARY KEY**
* The PRIMARY KEY constraint uniquely identifies each record in a table.
* Primary keys must contain UNIQUE values, and cannot contain NULL values.
* A table can have only ONE primary key;
* **MySQL FOREIGN KEY**
* The FOREIGN KEY constraint is used to prevent actions that would destroy links between tables.
* A FOREIGN KEY is a field (or collection of fields) in one table, that refers to the PRIMARY KEY in another table.
* The table with the foreign key is called the child table, and the table with the primary key is called the referenced or parent table.

**DAY 25**

* **The SELECT Statement**

The SELECT statement is used to select data from a database.

The data returned is stored in a result table, called the result-set.

**Syntax**

*SELECT column1, column2, ...*

*FROM table\_name;*

**Example**

SELECT

`id`,

`name`,

`status`

FROM `department`;

**The WHERE Clause**

The WHERE clause is used to filter records.

**Syntax**

*SELECT column1, column2, ...*

*FROM table\_name*

*WHERE condition;*

**Example**

SELECT

`id`,

`name`,

`status`

FROM `department`

WHERE `status` = 1;

* **The UPDATE Statement**

The UPDATE statement is used to modify the existing records in a table.

**Syntax**

*UPDATE table\_name*

*SET column1 = value1, column2 = value2, ...*

*WHERE condition;*

**Example**

UPDATE `department` SET

`name` = 'B.Sc'

where id=1;

**DAY 26**

* **The DELETE Statement**

The DELETE statement is used to delete existing records in a table.

**Syntax**

DELETE FROM table\_name WHERE condition;

**Example**

DELETE FROM `department` where id=1

* **Join Statement**

**INNER JOIN**

MySQL INNER JOIN Keyword

The INNER JOIN keyword selects records that have matching values in both tables.



***INNER JOIN Syntax***

*SELECT column\_name(s)*

*FROM table1*

*INNER JOIN table2*

*ON table1.column\_name = table2.column\_name;*

**DAY 27**

* **LEFT JOIN Keyword**

The LEFT JOIN keyword returns all records from the left table (table1), and the matching records (if any) from the right table (table2).



***LEFT JOIN Syntax***

*SELECT column\_name(s)*

*FROM table1*

*LEFT JOIN table2*

*ON table1.column\_name = table2.column\_name;*

* **RIGHT JOIN Keyword**

The RIGHT JOIN keyword returns all records from the right table (table2), and the matching records (if any) from the left table (table1).



***RIGHT JOIN Syntax***

*SELECT column\_name(s)*

*FROM table1*

*RIGHT JOIN table2*

*ON table1.column\_name = table2.column\_name;*

**DAY 28**

* **PHP Connect to MySQL**

PHP 5 and later can work with a MySQL database using:

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

**Open a Connection to MySQL**

<?php

$servername = "localhost";

$username = "username";

$password = "password";

$dbname = "myDB";

// Create connection

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

// Check connection

if ($conn->connect\_error) {

die("Connection failed: " . $conn->connect\_error);

}

?>

**DAY 29**

* **Insert Data Into MySQL Using MySQLi and PDO**
* 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

The INSERT INTO statement is used to add new records to a MySQL table:

***INSERT INTO table\_name (column1, column2, column3,...)***

***VALUES (value1, value2, value3,...)***

* **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***

or we can use the \* character to select ALL columns from a table:

***SELECT \* FROM table\_name***

**DAY 30**

* **Update Data In a MySQL Table Using MySQLi and PDO**

The UPDATE statement is used to update existing records in a table:

***UPDATE table\_name***

***SET column1=value, column2=value2,...***

***WHERE some\_column=some\_value***

* **Delete Data From a MySQL Table Using MySQLi and PDO**

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

***DELETE FROM table\_name***

***WHERE some\_column = some\_value***

**CONCLUSION**

**PHP is a great tool for writing dynamic web pages. Non-technical users can easily. learn a few handy tricks to make their web pages easier to manage, and more useful. Because its syntax resembles most C-like languages, any Computer Science student is able to learn it very quickly. When creating a PHP enhanced pages, there are a few things we must remember PHP is a server side technology, and does not work in a browser. The filename must have php extension PHP enhanced pages can contain a mixture of HTML and PHP code PHP code must be enclosed in <?php?> tag. For more PHP information and tips, please visit php.net, or do a web search.**

**REFERENCES**

**www.php net**

**Wikipedia**

**http://www.w3schools.com/**