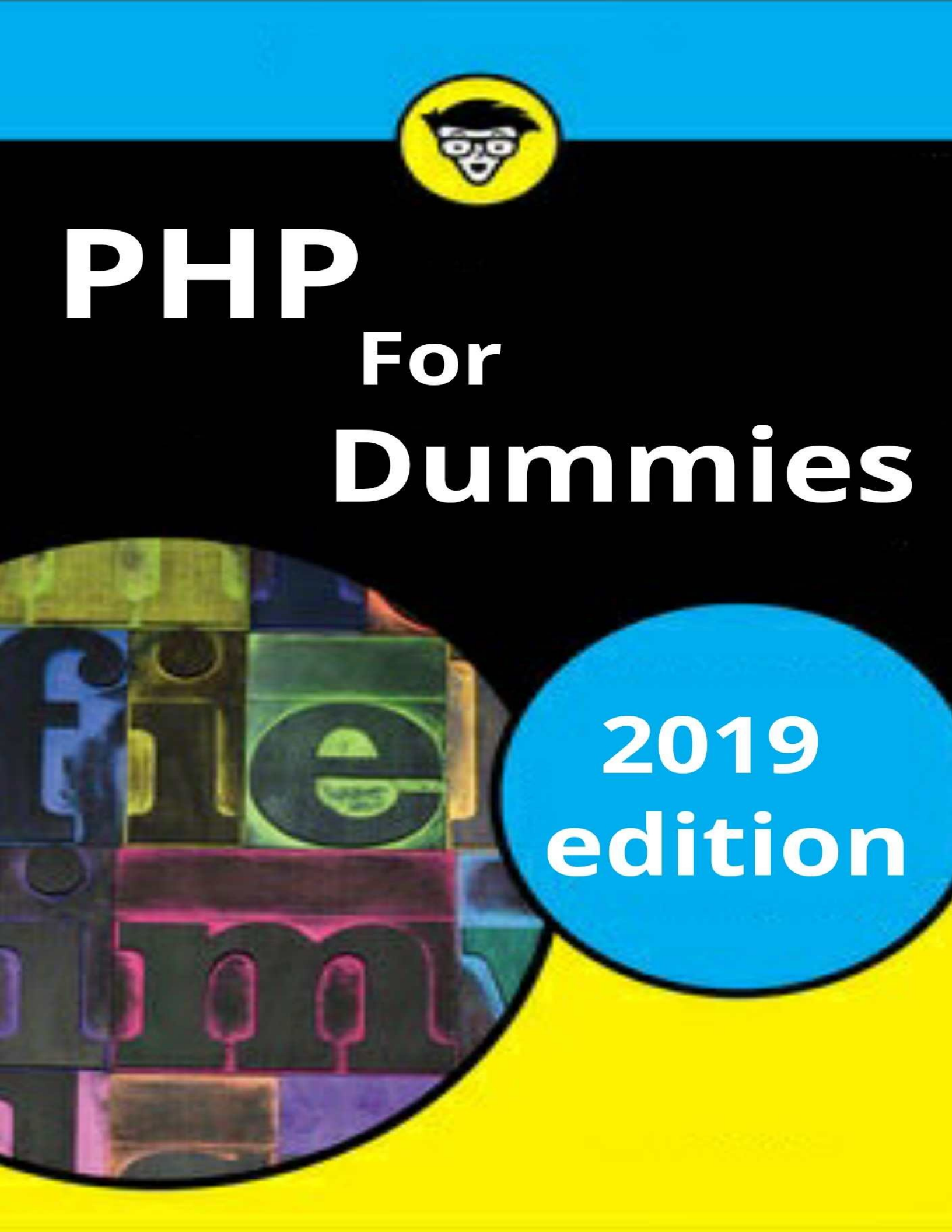

PHP
For
Dummies
2019
edition

PHP For Dummies 2019

Content

What is PHP? Write your first PHP Program

What is PHP?

PHP is a server side scripting language. that is used to develop Static websites or Dynamic websites or Web applications. PHP stands for Hypertext Pre-processor, that earlier stood for Personal Home Pages.

PHP scripts can only be interpreted on a server that has PHP installed.

The client computers accessing the PHP scripts require a web browser only.

A PHP file contains PHP tags and ends with the extension ".php".

In this tutorial, you will learn-

What is a Scripting Language?

A script is a set of programming instructions that is interpreted at runtime.

A scripting language is a language that interprets scripts at runtime. Scripts are usually embedded into other software environments.

The purpose of the scripts is usually to enhance the performance or perform routine tasks for an application.

Server side scripts are interpreted on the server while client side scripts are interpreted by the client application.

PHP is a server side script that is interpreted on the server while JavaScript is an example of a client side script that is interpreted by the client browser. Both PHP and JavaScript can be embedded into HTML pages.

Programming Language Vs Scripting Language

Programming language	**Scripting language**
Has all the features needed to develop complete applications.	Mostly used for routine tasks
The code has to be compiled before it can be executed	The code is usually executed without compiling
Does not need to be embedded into other languages	Is usually embedded into other software environments.

What does PHP stand for?

PHP means - **Personal Home Page**, but it now stands for the recursive backronym PHP: Hypertext Preprocessor.

PHP code may be embedded into HTML code, or it can be used in combination with various web template systems, web content management system and web frameworks.

Php Syntax

```
<?php
    echo 'Hello World';
?>
```

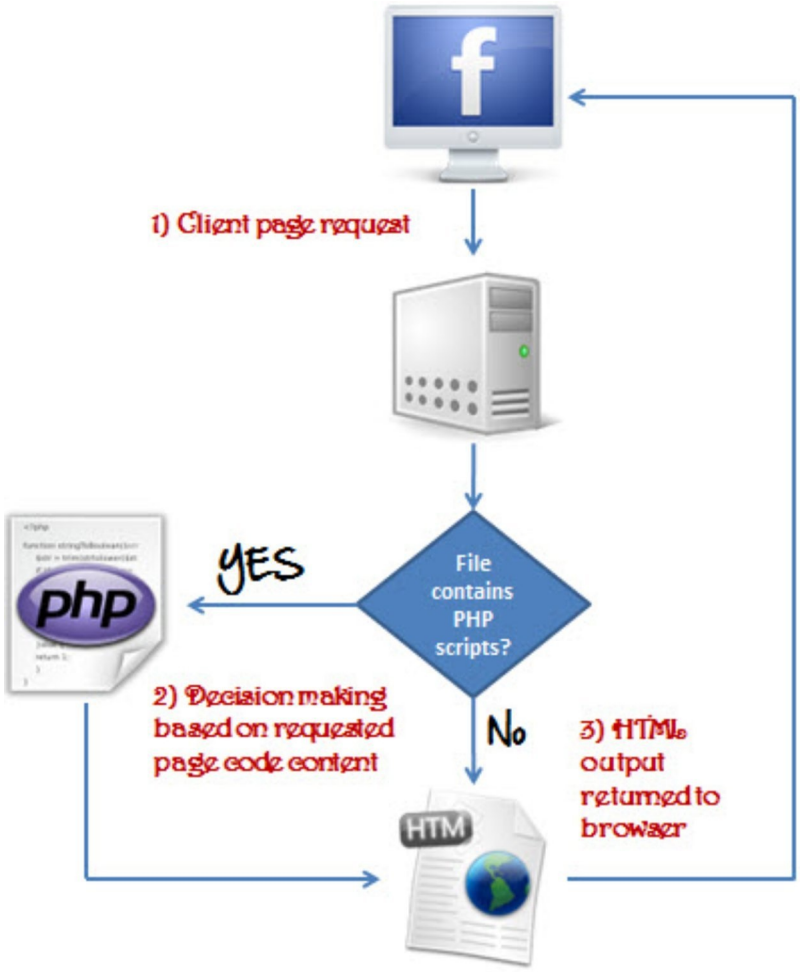
A PHP file can also contain tags such as HTML and client side scripts such as JavaScript.

- **HTML is an added advantage** when learning PHP Language. You can even learn PHP without knowing HTML but it’s recommended you at least

know the basics of HTML.

- **Database management systems** DBMS for database powered applications.
- For more advanced topics such as interactive applications and web services, you will need **JavaScript and XML**.

The flowchart diagram shown below illustrates the basic architecture of a PHP web application and how the server handles the requests.



Why use PHP?

You have obviously head of a number of programming languages out there; you may be wondering why we would want to use PHP as our poison for the web programming. Below are some of the compelling reasons.

- PHP is **open source and free.**
- Short learning curve compared to other languages such as JSP, ASP etc.
- Large community document

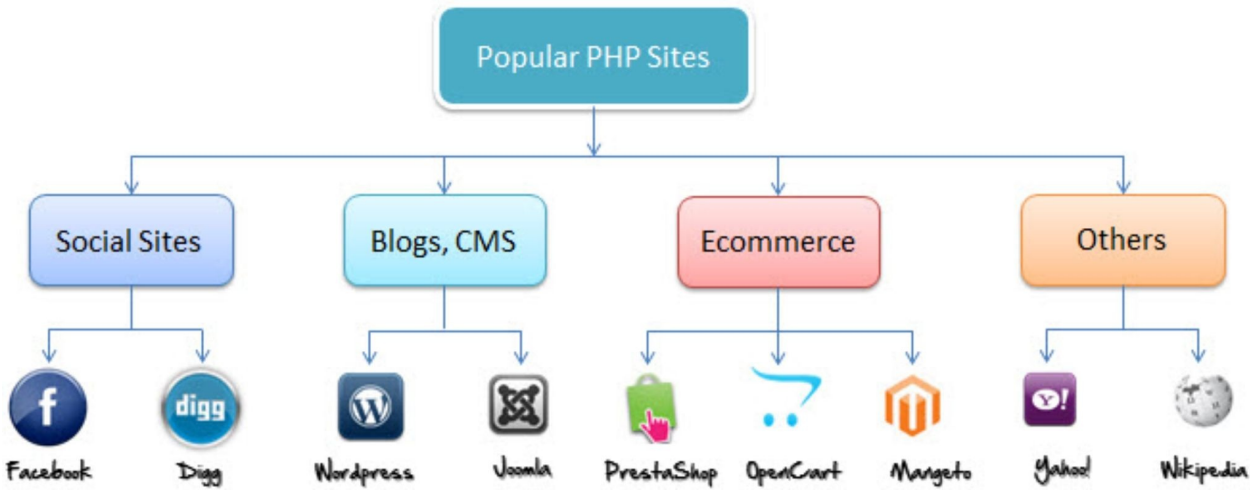
- Most web hosting servers support PHP by default unlike other languages such as ASP that need IIS. This makes PHP a cost effective choice.
- PHP is regular updated to keep abreast with the latest technology trends.
- Other benefit that you get with PHP is that it's a **server side scripting language**; this means you only need to install it on the server and client computers requesting for resources from the server do not need to have PHP installed; only a web browser would be enough.
- PHP has **in built support for working hand in hand with MySQL**; this doesn't mean you can't use PHP with other database management systems. You can still use PHP with
    - Postgres
    - Oracle
    - MS SQL Server
    - ODBC etc.
- PHP is **cross platform;** this means you can deploy your application on a number of different operating systems such as windows, Linux, Mac OS etc.

What is PHP used for & Market share

In terms of market share, there are over 20 million websites and application on the internet developed using PHP scripting language.

This may be attributed to the points raised above;

The diagram below shows some of the popular sites that use PHP



PHP vs Asp.Net VS JSP VS CFML

ASP – Active Server Pages, JSP – Java Server Pages, CFML – Cold Fusion Markup language The table below compares the various server side scripting languages with PHP

FEATURE	PHP	ASP	JSP	CFML
Learning curve	short	Longer than PHP	Longer than PHP	Longer than PHP
Web hosting	Supported by almost all hosting servers	Needs dedicated server	Fairly supported	Needs dedicated server
Open source	Yes	No	Yes	Both commercial and open source
Web services support	Built in	Uses the .NET framework	Uses add on libraries	Built in
Integration with HTML	Easy	Fairly complex	Fairly complex	Easy
MySQL support	Native	Needs third party drivers	Needs third party drivers	Current version has native support. Older versions use ODBC
Easily extended by other languages	Yes	No	Extended using Java classes and libraries.	Yes

PHP File Extensions

File extension and Tags In order for the **server** to **identify** our **PHP files** and **scripts**, we must **save** the **file** with the **".php" extension**. Older PHP file extensions include

- .phtml
- .php3
- .php4
- .php5
- .phps

PHP was designed to work with HTML, and as such, it can be embedded into the HTML code.

You can create PHP files without any html tags and that is called Pure PHP file .

The server interprets the PHP code and outputs the results as HTML code to the web browsers.

In order for the server to identify the PHP code from the HTML code, we must always enclose the PHP code in PHP tags.

A PHP tag starts with the less than symbol followed by the question mark and then the words “php”.

PHP is a case sensitive language, “VAR” is not the same as “var”.

The PHP tags themselves are not case-sensitive, but it is strongly recommended that we use lower case letter. The code below illustrates the above point.

```
<?php … ?>
```

We will be referring to the PHP lines of code as statements. PHP statements end with a semi colon (;). If you only have one statement, you can omit the semi colon. If you have more than one statement, then you must end each line with a semi colon. For the sake of consistency, it is recommended that you always end your statement(s) with a semi colon. PHP scripts are executed on the server. The output is returned in form of HTML.

PHP Hello world

The program shown below is a basic PHP application that outputs the words “Hello World!” When viewed in a web browser.

```
<?php
echo "Hello world";
?>
```

Output:

```
Hello world
```

Summary

- PHP stands for Hypertext pre-processor
- PHP is a server side scripting language. This means that it is executed on the server. The client applications do not need to have PHP installed.
- PHP files are saved with the ".php" file extension, and the PHP development code is enclosed in tags.
- PHP is open source and cross platform

How to Download & Install XAMPP & NetBeans: PHP Tutorial

What is XAMPP?

XAMPP is an open source cross platform web server, MySQL database engine, and PHP and Perl package. It is compiled and maintained by apache. The acronym XAMPP stands for;

- X – [cross platform operating systems] meaning it can run on any OS Mac OX , Windows , Linux etc.
- A – Apache - this is the web server software.
- M – MySQL - Database.
- P – PHP
- P – Perl – scripting language

Why use XAMPP?

- In order to use PHP, you will need to install PHP, Apache and may be even MySQL. It's not easy to install Apache and configure it. If you install Apache on its own, you will still have to set it up and integrate it with PHP and Perl among other things. XAMPP deals with all the complexity in setting up and integrating with PHP and Perl. Unlike Java that runs with the Java SDK only, PHP requires a web server to work
- XAMPP provides an easy to use control panel to manage Apache, MySQL and other programs such as Tomcat, filezilla etc. You don't have to memorize commands for starting apache, MySQL etc.

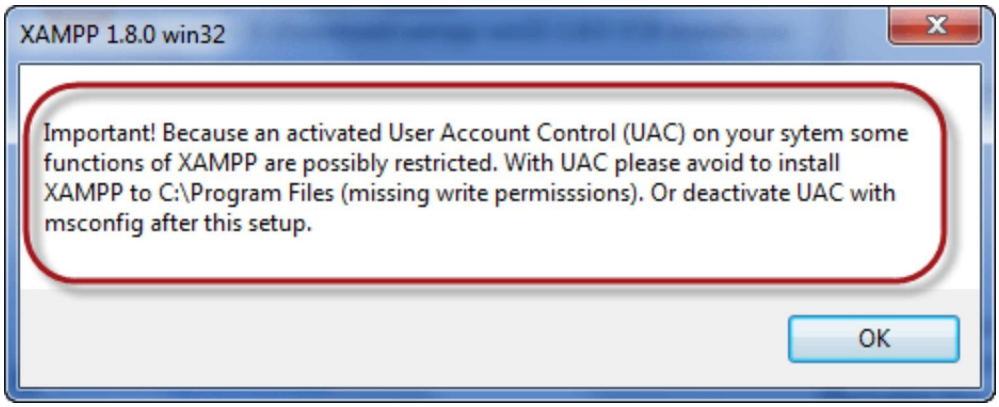
In this tutorial, you will learn-

- Basic Web server configuration
- XAMPP Control Panel
- Configure XAMPP
- What is the best PHP IDE?
- Introduction to Netbeans IDE
- Creating a new PHP project using the Netbeans IDE
- Running your first PHP Example

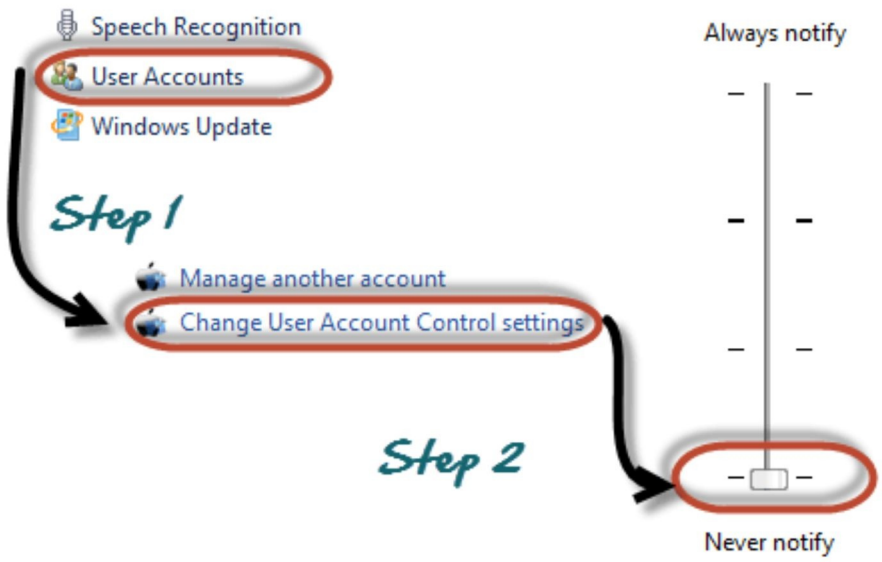
How to Download and Install XAMPP

We look into step by step process to install XAMPP for Windows. For Other Operating Systems, installation steps are similar.

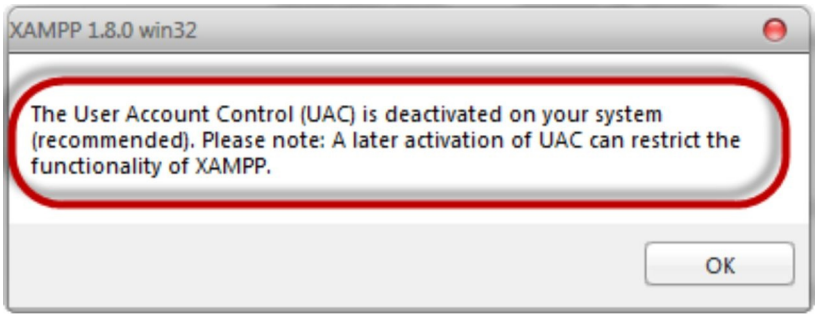
- Download the XAMPP installer at http://www.apachefriends.org/en/xampp-windows.html
- Installation XAMPP is just like installing any other windows program. There are however, a few things that we must note.
- After you have downloaded XAMPP, run the setup. The warning message dialog window shown below appears.



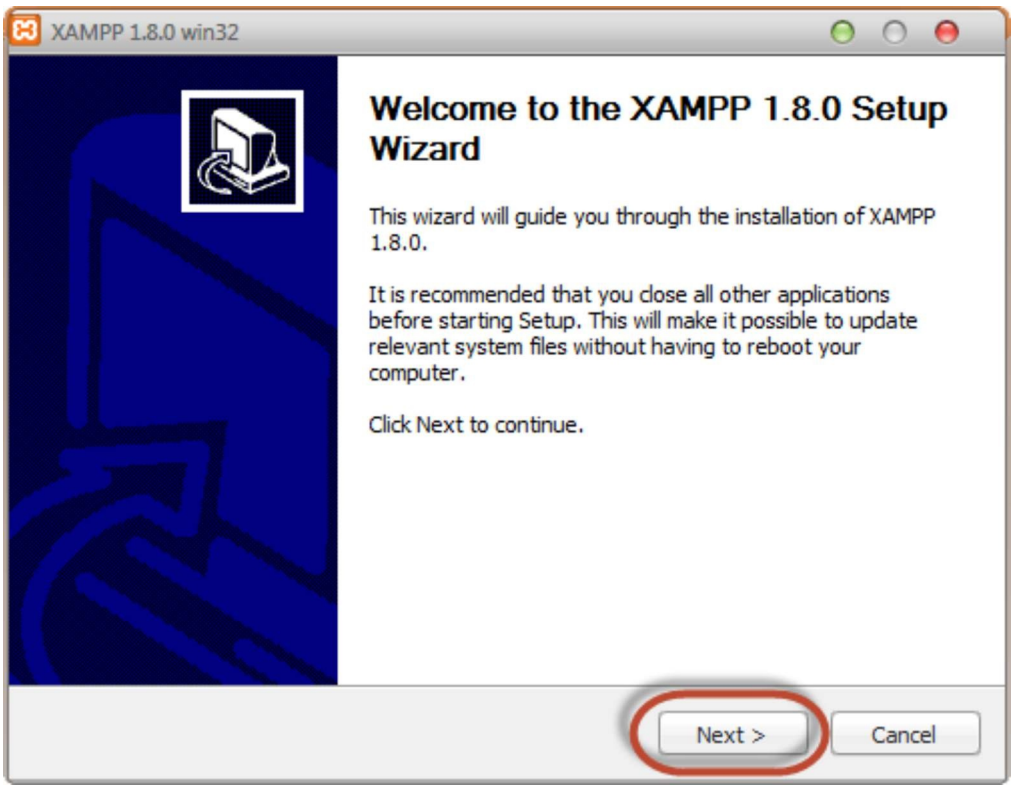
- If you are using windows Vista or Windows 7, make sure that you deactivate the User Account Control feature. To do this, Select Control Panel > User Accounts > Change User Access Control settings. The diagram below illustrates the main steps.



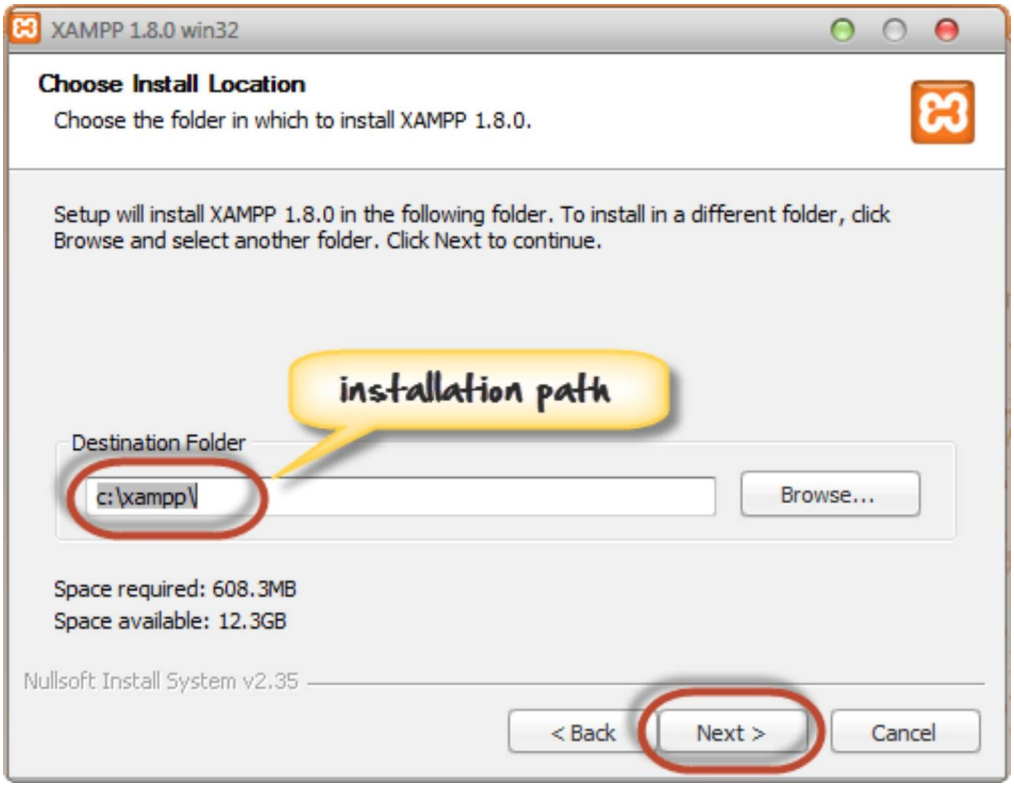
- After you have deactivated the User Account Control, click on OK button on the warning message box.
- This time you get following message



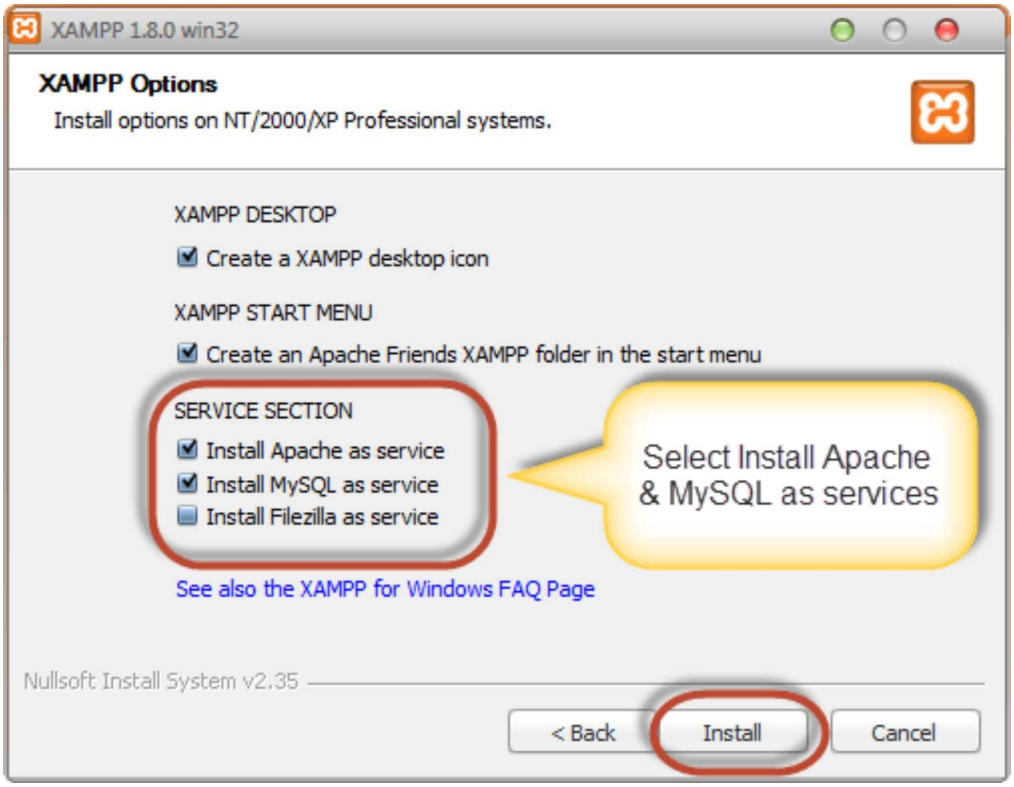
- In the succeeding screen, click next



- In the next screen, Change the installation path if required. Click Next



- In the next screen select Apache and MySQL. You may optionally select FileZilla (FTP Client) if needed. Click Install



Note *a service is a long-running program in windows that does not require user intervention. Services can be set to run automatically whenever the windows operating system is started.* ***For you to use Apache and MySQL, they are supposed to be running in the background****.* ***Installing them as services runs both Apache and MySQL automatically in the background whenever you power up your computer****. If you have not installed Apache and MySQL as services, then you have to manually start them every time that you want to use them. You will have to do this from the XAMPP control panel.PHP and*

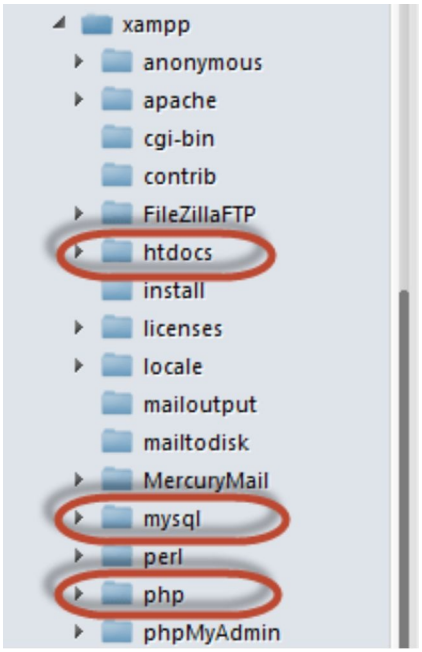
- On successful completion of installation, you will see following window

- Click on Finish button

Before we test our XAMPP installation, let's first look at the basic directories that we will be working with.

Basic Web server configuration

This tutorial assumes that you have **installed XAMPP on drive C in Windows using the steps mentioned above**. The following is a list of the basic directories that you are supposed to be aware of.



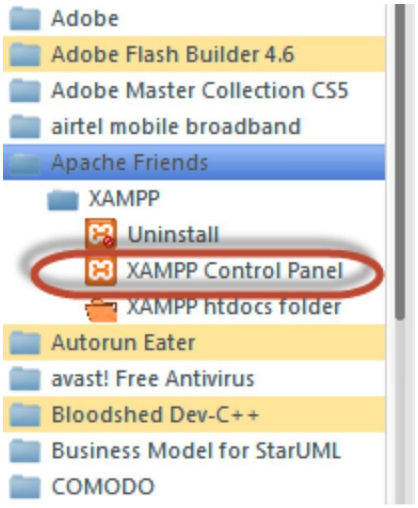
- **htdocs**; this is the web root directory. All of our PHP codes will be placed in this directory.
- **mysql** – this directory contains all the information related to MySQL database engine, by default it runs on port 3306.
- **php** – this directory contains PHP installation files. It contains an important file named php.ini. This directory is used to configure how PHP behaves on your server.

By default, the Apache web server runs on **port 80**. If port 80 is taken by another web server, you can use a different port number. For this tutorial we will assume we are using port 80. Note,If you use SKYPE , it uses the same port. Close Skype if you want to use Apache on port 80

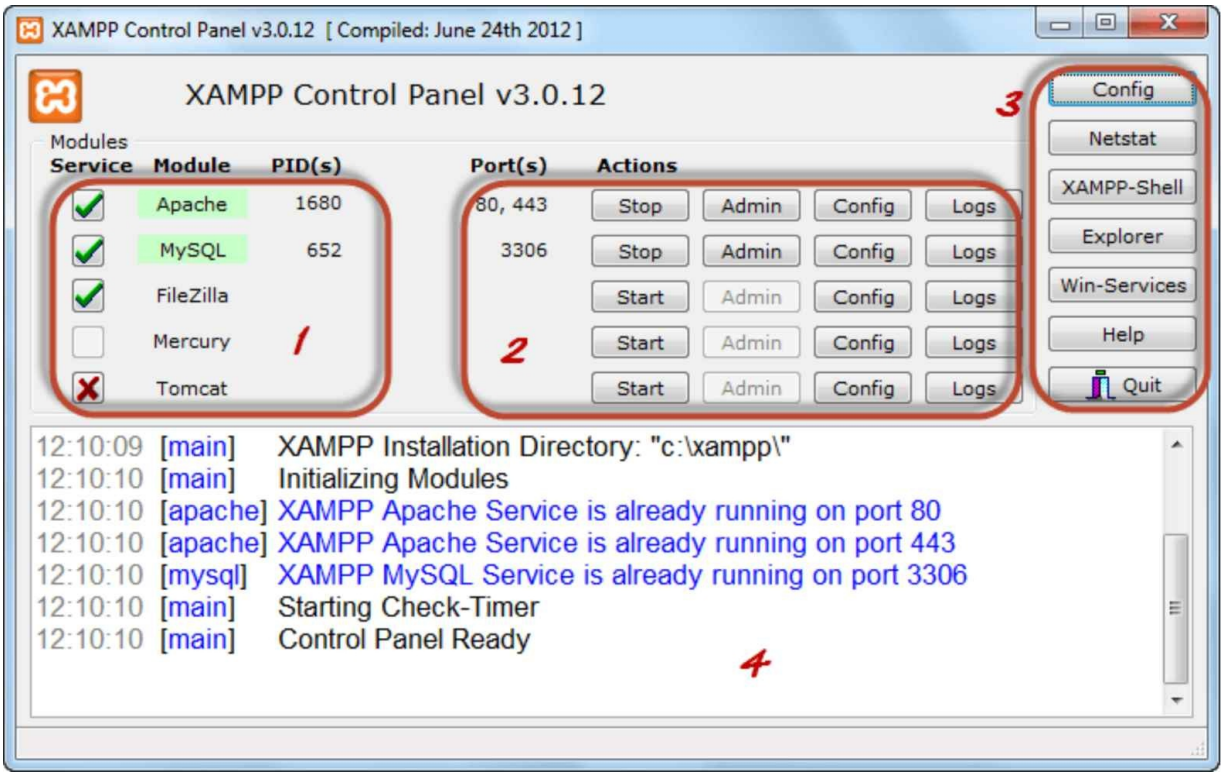
XAMPP Control Panel

The control panel is used to manage programs installed via XAMPP. To open the XAMPP control panel,

- Click on start menu
- Explore the programs directory and locate Apace Friends then XAMPP as shown in the diagram below



- The diagram below shows the control panel.



1) This section lists the installed services, modules and the process IDs PID(s). A green tick means the module has been installed as a service. The red mark means it has not been installed as a service. To install a service, click on the red mark. If the button shows a green tick and you click on it, the control panel will ask you if you want to uninstall the system.

2) This section shows Port(s) associated with the modules. The actions section is for;

1. starting and stopping modules

2. Open the administrative windows for Apache and MySQL
3. Open configuration files for Apache, MySQL etc. to make changes
4. View log files for the modules

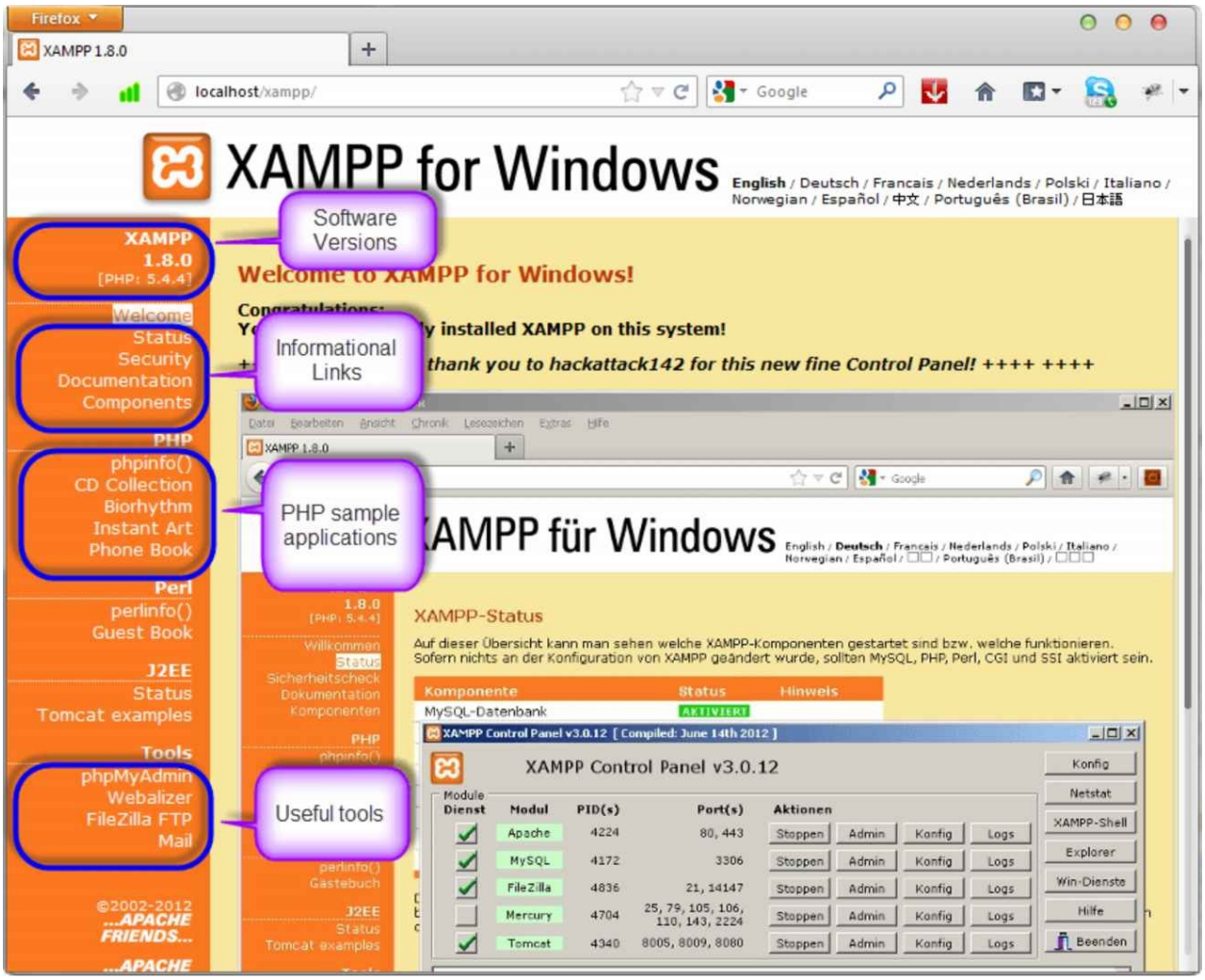
3) This section contains useful utilities such as Netsat, windows services short cuts etc.

4) This section displays status information on the modules. The control panel can be used to;

- Install and uninstall services such as Apache, MySQL etc. that are installed via XAMPP
- Start and stop services.
- Open configure files etc.

Configure XAMPP

Let's now look at the basic configurations required before we start using our XAMPP installation for developing PHP powered web sites. Type the URL **http://localhost/xampp/** in your favorite browser. For this tutorial, we will be using Firefox as our web browser.



If you are able to see the above screen then you have installed XAMPP successfully. The panel on the left hand side contains links to useful information such as;

- The version of PHP installed
- Security settings of XAMPP
- Access to utilities such as phpMyAdmin etc.

The PHP version shipped with XAMPP 1.8.0 is PHP 5.4.4

What is the best PHP IDE?

A PHP IDE is a program that allows you to easily write PHP codes. PHP IDEs are often equipped with syntax highlighting features and in some cases autocomplete features too. This means that if you write a PHP keyword that is known by the PHP interpreter, then the keyword will be highlighted a different color from the one used for regular statements. The autocomplete features automatically pops up known PHP keywords as you type them. Notepad can also

be used to write and editor PHP codes. The disadvantage of using an editor such as Notepad is that debugging the scripts becomes difficult because it is not easy to spot errors such as misspelt keywords, unclosed braces etc. an IDE will highlight the statements with errors so it's easy for you to spot them. The table shown below shows 5 popular PHP editors

Editor	**License**	**Cross Platform**	**Brief description**
Netbeans IDE	Open Source	Yes	• Dedicated PHP coding environment with syntax highlighting and code completion for keywords and other known information. • Supports integration with PHP MVC frameworks i.e. Zend, • Code History that shows the changes made to a file • SFTP,FTP and SVN via plugins.
Dreamweaver	Commercial	Yes	• Supports HTML and PHP. • Syntax highlighting, code folding and completion for keywords and other known information. • Supports SFTP and FTP.
Zend studio	Commercial	Yes	• Integrated with Zend Server and Zend PHP MVC framework, PHPUnit, phpDocumentor etc. • Has syntax highlighting, code folding, • Support for Web services etc.
PHP Eclipse	Open Source	Yes	• Code formatter • Supports SVN, SHH/FTP
Notepad ++	Freeware	Windows only	• Syntax highlighting • Supports SFTP and FTP via plugins.

Netbeans IDE PHP editor As briefly highlighted in the above table, Netbeans IDE has powerful features that enhance the productive of PHP coders. The IDE can be freely downloaded from the http://netbeans.org/downloads/index.html

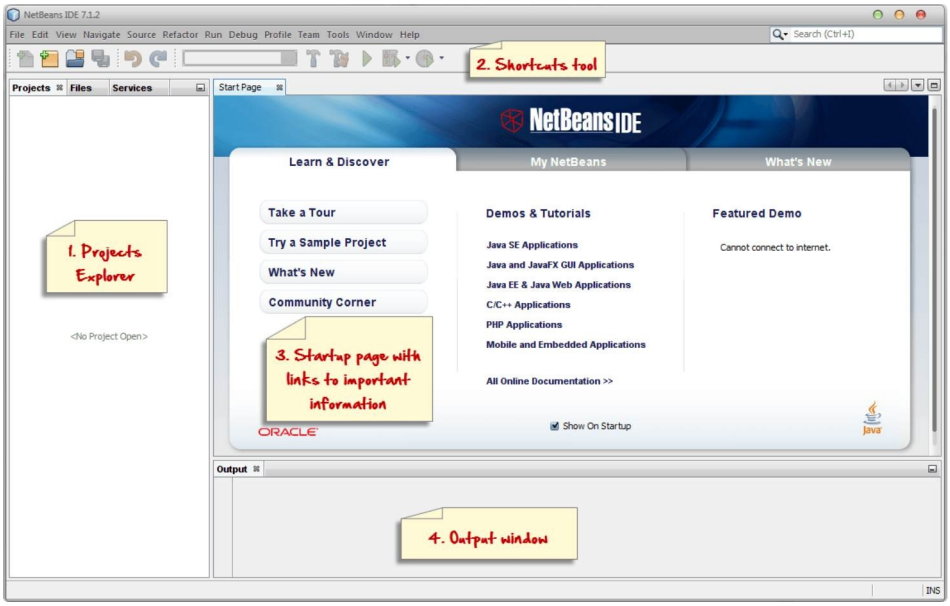
- Syntax highlighting and auto-complete features enhances your **productivity**

- It has native support for database systems like MySQL. **You don’t need to use two programs to code and develop your database**.
- The IDE can be used in a **collaborative environment**. This comes in handy when you have to work with other developers as a team.
- The IDE has **support for other languages** such as;
    - Java SE
    - Java EE
    - C
    - C++

The current version of the Netbeans IDE as of this writing is version 7.3

Introduction to Netbeans IDE

After you have successfully installed the Netbeans IDE PHP editor, run the program just like any other windows program. The window shown below appears



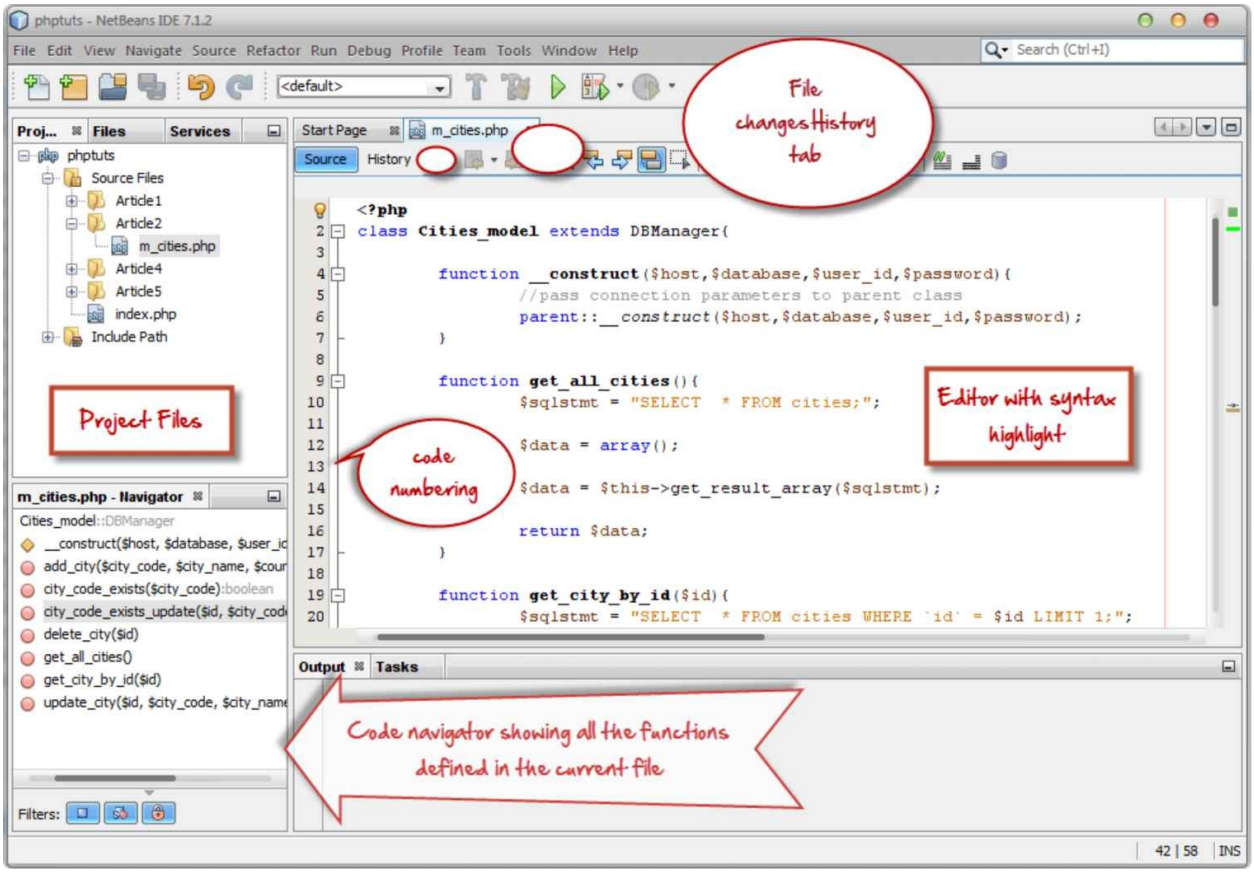
1. Project explorer – this panel is used to display all the opened projects. The projects are listed in a tree view.

2. Shortcuts tool bar – this toolbar contains shortcuts to frequently performed tasks such as creating a new project, opening an existing project, undo and redo actions etc.

3. Startup page – this page contains 3 tabs namely- Learn & Discover, My Netbeans and What’s New.

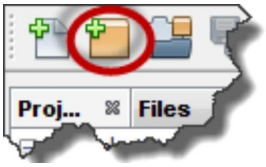
- The first tab [Learn and Discover] introduces you to the features of the Netbeans IDE, showcases some demos and tutorials that can be developed in the Netbeans IDE.
- The second tab [My Netbeans] lists the recently opened projects, allows you to install plugins and activate features of the IDE.

4. Output window – it is used to display output from programs such as Java console applications. It is also used to display log and debug information. The screenshot below shows the IDE with a project open.



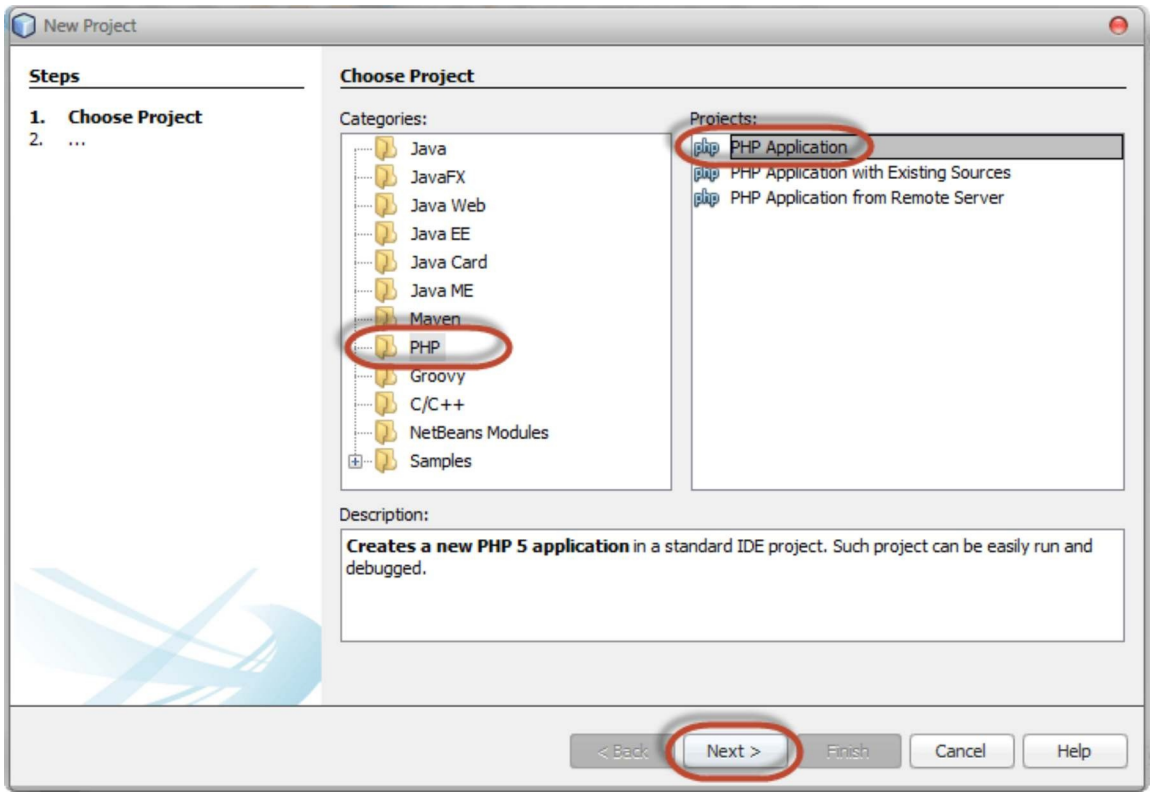
Creating a new PHP project using the Netbeans IDE

- Click on the create new project button on the tool bar as shown below

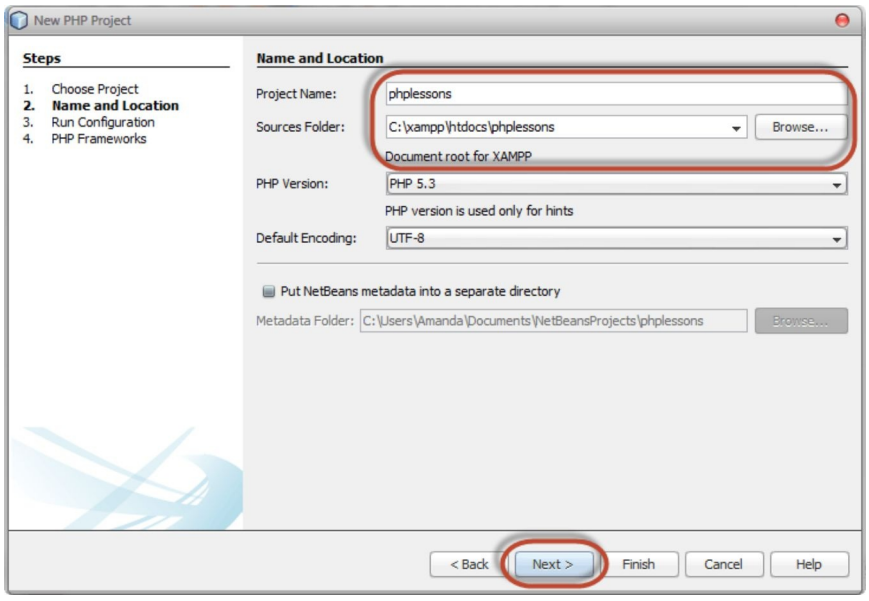


- If you downloaded all the bundles available in the Netbeans IDE, make sure you choose PHP under project category, PHP Application under

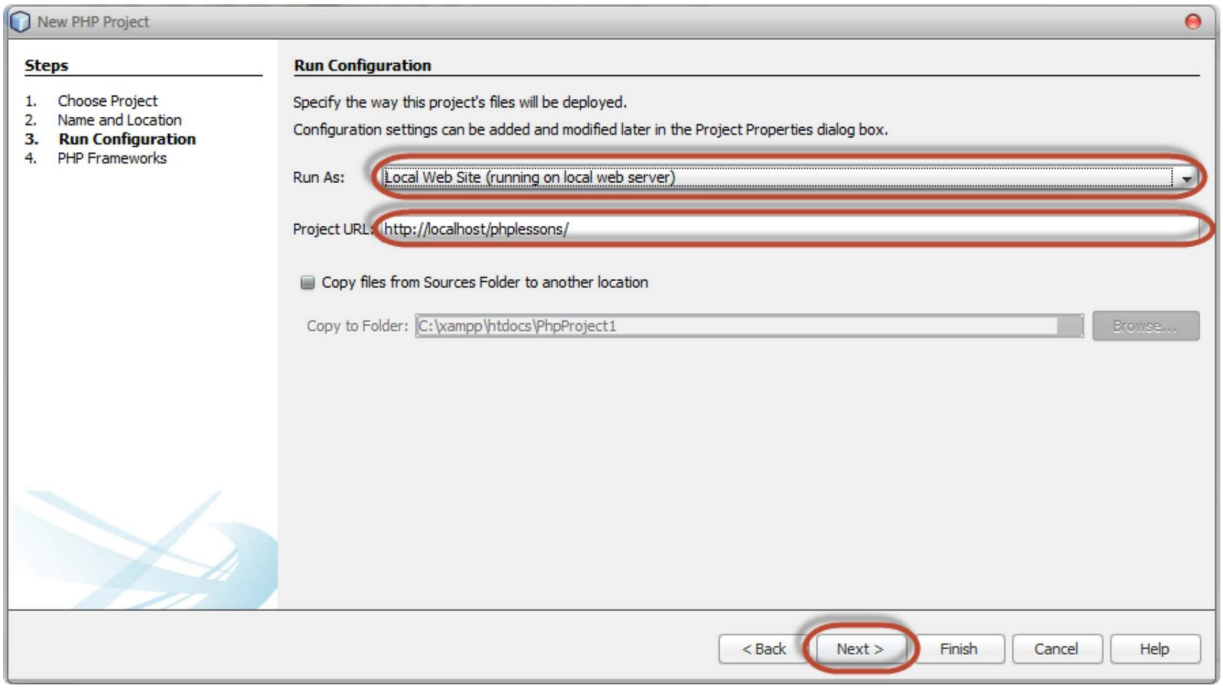
Projects then click on Next button.



- Enter the project name as shown below.



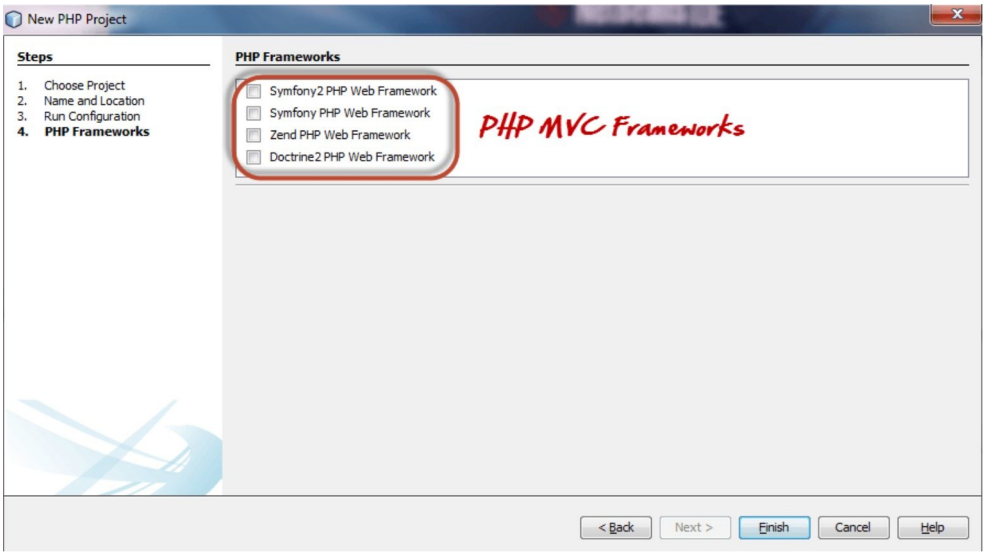
- Make sure the folder is saved in the XAMPP installation directory as shown above.
- Click on next button when done.



- **Make sure Run as: is set to Local Web Site(running on local web server)**
    - The Project URL: is set to **http://localhost/phplessons/**

Note *the above settings will be set for you by default. You don't have to change anything unless you are an expert*

- Click on Next button



- The Netbeans PHP editor allows for integration with PHP MVC frameworks such as Symfony and Zend. For now we will not select any MVC framework. Click on Finish button.

- Your newly created project will be displayed in the project browser and an index.php page created for you.
- The newly create page contains some html code. Replace it with the following code shown below.

Running your first PHP Example

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

- Click on the run button on the toolbar as shown below

- Your default browser will be opened with the URL **http://localhost/phplessons/index.php** . The output “Hello World!” will be displayed in your browser.

Summary

- XAMPP is the acronym for X-cross platform, Apache, MySQL, PHP and Perl
- A PHP editor is a program that allows you to write PHP code within the shortest possible time and allows you to debug your syntax errors at design time.
- Netbeans PHP editor is a cross platform open source editor that enhances the productivity of PHP developers.

PHP Data Types, Variables, Constant, Operators Tutorial

PHP Data Types

A Data type is the classification of data into a category according to its attributes;

- Alphanumeric characters are classified as strings
- Whole numbers are classified integers
- Numbers with decimal points are classified as floating points.
- True or false values are classified as Boolean.

PHP is a loosely typed language; it does not have explicit defined data types. PHP determines the data types by analyzing the attributes of data supplied. PHP implicitly supports the following data types

- Integer – whole numbers e.g. -3, 0, 69. The maximum value of an integer is platform-dependent. On a 32 bit machine, it's usually around 2 billion. 64 bit machines usually have larger values. The constant PHP_INT_MAX is used to determine the maximum value.

```
<?php
echo PHP_INT_MAX;
?>
```

Output:

```
9223372036854775807
```

- Floating point number – decimal numbers e.g. 3.14. they are also known as double or real numbers. The maximum value of a float is platform-dependent. Floating point numbers are larger than integers.
- Character string – e.g. Hello World
- Boolean – e.g. True or false.

Before we go into more details discussing PHP data types, let's first discuss variables.

PHP Variable

A variable is a name given to a memory location that stores data at runtime.

The scope of a variable determines its visibility.

A Php global variable is accessible to all the scripts in an application.

A local variable is only accessible to the script that it was defined in.

Think of a variable as a glass containing water. You can add water into the glass, drink all of it, refill it again etc.

The same applies for variables. Variables are used to store data and provide stored data when needed. Just like in other programming languages, PHP supports variables too. Let's now look at the rules followed when creating variables in PHP.

- All variable names must start with the dollar sign e.g.
- $my_var
- Variable names are case sensitive; this means $my_var is different from $MY_VAR
- $my_var ≠ $MY_VAR
- All variables names must start with a letter follow other characters e.g. $my_var1. $1my_var is not a legal variable name.
- ✔ $my_var1; ✖ ~~$1my_var;~~
- Variable names must not contain any spaces, "$first name" is not a legal variable name. You can instead use an underscore in place of the space e.g. $first_name. You cant use characters such as the dollar or minus sign to separate variable names.
- ✔ $my_var; ✖ $my var

Let's now look at how PHP determines the data type depending on the attributes of the supplied data.

```
<?php
$my_var = 1;
echo $my_var;
?>
```

Output:

```
1
```

Floating point numbers

```
<?php
$my_var = 3.14;
echo $my_var;
?>
```

Output:

```
3.14
```

Character strings

```
<?php
$my_var ="Hypertext Pre Processor";
echo $my_var;
?>
```

Output:

```
Hypertext Pre Processor
```

Use of Variables

Variables help separate data from the program algorithms.

The same algorithm can be used for different input data values.

For example, suppose that you are developing a calculator program that adds up two numbers, you can create two variables that accept the numbers then you use the variables names in the expression that does the addition.

Variable Type Casting

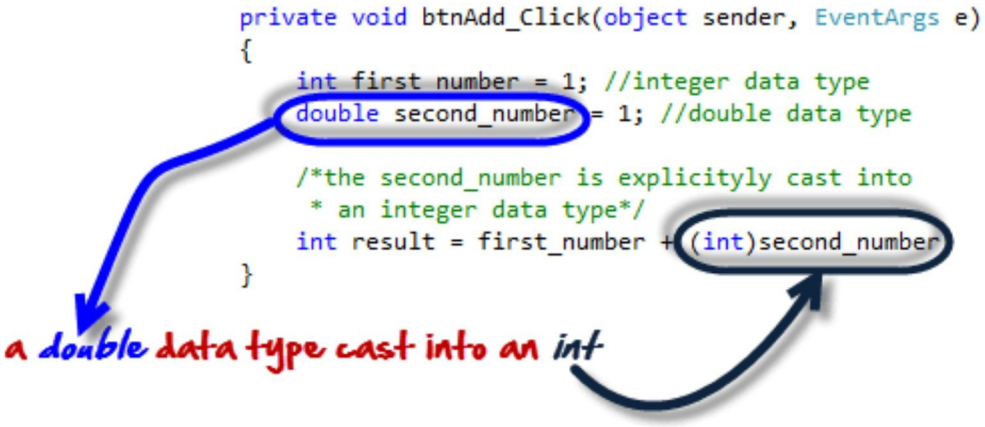
Performing arithmetic computations using variables in a language such as C# requires the variables to be of the same data type.

Type casting is converting a variable or value into a desired data type.

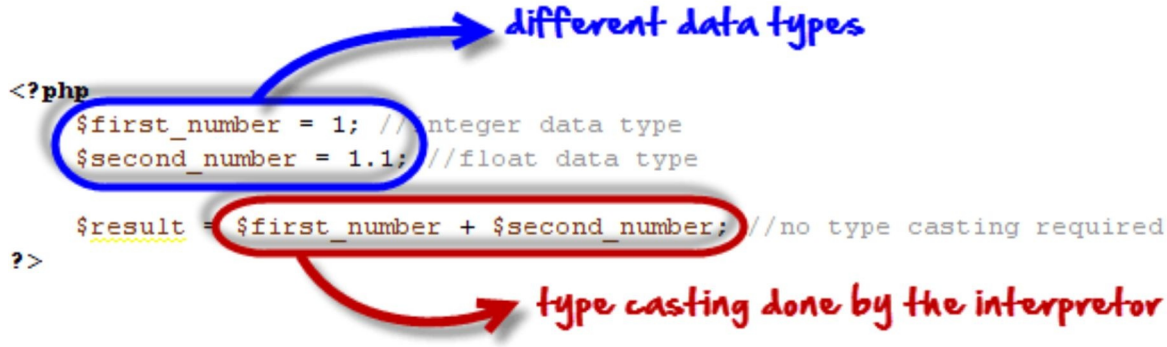
This is very useful when performing arithmetic computations that require variables to be of the same data type.

Type casting in PHP is done by the interpreter.

In other languages such as C#, you have to cast the variables. The code below shows type casting in C#.



The diagram below shows PHP implementing the above example.



PHP also allows you to cast the data type. This is known as explicit casting. The code below demonstrates explicit type casting.

```
<?php
$a = 1;
$b = 1.5;
$c = $a + $b;
$c = $a + (int) $b;
echo $c;
?>
```

Output:

```
2
```

Above Code Output 2 The var_dump function is used to determine the data type. The code below demonstrates how to use the var_dump function.

```
<?php
$a = 1;
var_dump($a);
$b = 1.5;
var_dump($b);
$c = "I Love PHP";
var_dump($c);
$d = true;
var_dump($d);
?>
```

Output:

```
int(1) float(1.5) string(10) "I Love PHP" bool(true)
```

PHP Constant

Define constant- A constant is a variable whose value cannot be changed at runtime.

Suppose we are developing a program that uses the value of PI 3.14, we can use a constant to store its value.

Let’s now look at an example that defines a constant. define('PI',3.14); //creates a constant with a value of 3.14 Once you define PI as 3.14 , writing a code like below will generate an error PI = 4; //PI has been defined as a constant therefore assigning a value is not permissible.

PHP Operators

Arithmetic operators

Arithmetic operators are used to perform arithmetic operations on numeric data. The concatenate operator works on strings values too. PHP supports the following operators.

Operator	Name	Description	Example	Output
+	Addition	Summation of x and y	1 + 1;	2
-	Subtraction	Difference between x and y	1 – 1;	0
*	Multiplication	Multiplies x and y	3 * 7;	21
/	Division	Quotient of x and y	45 / 5;	9
%	Php Modulus	Gives reminder of diving x and y	10 % 3;	1
-n	Negation	Turns n into a negative number	-(-5);	5
x . y	Concatenation	Puts together x and y	"PHP" . " ROCKS";10 . 3;	PHP ROCKS103

Assignment Operators

Assignment operators are used to assign values to variables. They can also be used together with arithmetic operators.

Operator	Name	Description	Example	Output
x = ?	assignment	Assigns the value of x to ?	$x = 5;	5
x += ?	addition	Increments the value of x by ?	$x = 2;$x += 1;	3
X -= ?	subtraction	Subtracts ? from the value of x	$x = 3;$x -= 2;	1
X *=?	multiplication	Multiplies the value of x ? times	$x = 0;$x *=9;	0
X /=?	division	Quotient of x and ?	$x = 6;$x /=3;	2
X %=?	modulus	The reminder of dividing x by?	$x = 3;$x %= 2;	1
X .=?	concatenate	Puts together items	" $x = ‘Pretty’;$x .= ‘ Cool!’;"	Pretty Cool!

Comparison operators

Comparison operators are used to compare values and data types.

Operator	Name	Description	Example	Output
X == y	Equal	Compares x and y then returns true if they are equal	1 == "1";	True or 1
X === y	identical	Compares both values and data types.	1 === "1";	False or 0. Since 1 is integer and “1” is string
X != y, x <> y	PHP Not equal	Compares values of x and y. returns true if the values are not equal	2 != 1;	True or 1

X > y	Greater than	Compares values of x and y. returns true if x is greater than y	3 > 1;	True or 1
X < y	Less than	Compares values of x and y. returns true if x is less than y	2 < 1;	False or 0
X >= y	Greater than or equal	Compares values of x and y. returns true if x is greater than or equal to y	1 >=1	True or 1
X <= y	Less than or equal	Compares values of x and y. returns true if x is greater than or equal to y	8 <= 6	False or 0

Logical operators

When working with logical operators, any number greater than or less than zero (0) evaluates to true. Zero (0) evaluates to false.

Operator	**Name**	**Description**	**Example**	**Output**
X and y, x && y	And	Returns true if both x and y are equal	1 and 4;True&& False;	True or 1False or 0
X or y, x || y	Or	Returns true if either x or y is true	6 or 9;0 || 0;	True or 1False or 0
X xor y	Exclusive or, xor	Returns true if only x is true or only y is true	1 xor 1;1 xor 0;	False or 0True or 1
!x	Not	Returns true if x is false and false if x is true	!0;	True or 1

Summary

- PHP is a loosely typed language.
- Variables are memory locations used to store data
- The value of constants cannot be changed at runtime
- Type casting is used to convert a value or variable into a desired data type
- Arithmetic operators are used to manipulate numeric data
- Assignment operators are used to assign data to variables
- Comparison operators are used to compare variables or values
- Logical operators are used to compare conditions or values

PHP Comments, Include/Include_once, Require/Require_once

Why use Comments?

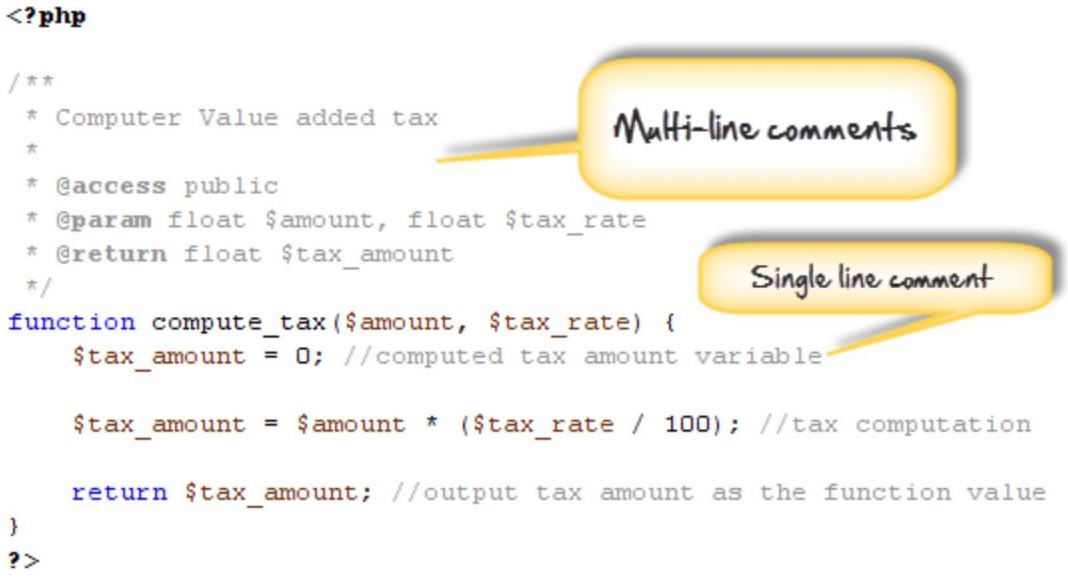
- If you don’t work on the source code for some time, it’s easy to forget what the code does. Commenting the source code helps remember what the code does.
- Commenting source code is also very important when multiple developers have to work on the same project. The changes made by one developer can be easily understood by other developers by simply reading the comments.
- As the best practice, you must have 3 lines of comments for every 10 lines of code

PHP Comments

- Comments help us to understand the code
- Comments are explanations that we include in our source code. These comments are for human understanding.
- Single line comments start with double forward slashes // and they end in the same line.
- //
- Multiple line comments start with a forward slash followed by the asterisk /* and end with the asterisk followed by the forward slash */.

```
/*this is a multiple-line
*comment
/*example
```

The diagram below shows a PHP file with both multiple line and single line comments PHP Example




```php
<?php

/**
 * Computer Value added tax
 *
 * @access public
 * @param float $amount, float $tax_rate
 * @return float $tax_amount
 */
function compute_tax($amount, $tax_rate) {
    $tax_amount = 0; //computed tax amount variable

    $tax_amount = $amount * ($tax_rate / 100); //tax computation

    return $tax_amount; //output tax amount as the function value
}
?>
```

PHP Include & PHP Include_once

The “include” php statement is used to include other files into a PHP file.

It has two variations, include and include_once. Include_once is ignored by the PHP interpreter if the file to be included.

The include statement has the following syntax

```
<?php
include 'file_name';
?>
```

The include_once statement has the following syntax

```
<?php
include_once 'file_name';
?>
```

HERE,

- “Include/include_once” is the statement that includes file
- “'file_name'” is the name of the file to be included.

Example : Include / Include_once

Suppose you are developing a website that contains the same navigation menu across all the pages.

You can create a common header then include it in every page using the include statement Let’s see how this can be done.

- We will create 2 files names
- header.php, index.php

Below are the codes for; *header.php*

```
<a href="/index.php">Home</a>

<a href="/aboutus.php">About us</a>

<a href="/services.php">Services</a>

<a href="/contactus.php">Contact Us</a>
```

index.php

```
<?php

include 'header.php';

?>
```

The header page above will output

PHP Require & PHP require_once

The require statement has two variations, require and require_once.

The require/require_once statement is used to include file.

Require_once is ignored if the required file has already been added by any of the four include statements.

It has the following syntax

```
<?php
require 'file_name';
?>
<?php
require_once 'file_name';
?>
```

HERE,

- “require/require_once” is the statement that includes file
- “'file_name'” is the name of the file to be included.

Example : Require

Suppose we are developing a database powered application.

We can create a configuration file that we can include in all pages that connect to the database using the require statement. config.php

```
<?php

$config['host'] = 'localhost';

$config['db'] = 'my_database';

$config['uid'] = 'root';

$config['password'] = '';

?>
```

Let’s now look at the sample code that requires the config file. *Pages_model.php*

```
<?php

require 'config.php'; //require the config file

//other code for connecting to the database

?>
```

Php include vs require

The difference between include / require

Include	**Require**
Issues a warning when an error occurs	Does not issue a warning
Execution of the script continues	Execution of the script stops when

when an error occurs	an error occurs.

Generally, it’s recommended using the include statement so that when an error occurs, execution of the script continues to display the webmaster email address or the contact us page.

The require statement should be used if the entire script cannot run without the requested file.

The “include” and “require” statements can be used at any line in the source codes where you want the code to appear.

PHP Array: Associative, Multidimensional

What is a PHP Array?

A PHP array is a variable that stores more than one piece of related data in a single variable.

Think of an array as a box of chocolates with slots inside.

The box represents the array itself while the spaces containing chocolates represent the values stored in the arrays.

The diagram below illustrates the above syntax.

Numeric Arrays

Numeric arrays use number as access keys.

An access key is a reference to a memory slot in an array variable.

The access key is used whenever we want to read or assign a new value an array element.

Below is the syntax for creating numeric array in php. Array Example

```
<?php
$variable_name[n] = value;
?>
```

Or

```
<?php
$variable_name = array(n => value, …);
?>
```

HERE,

- "$variable_name…" is the name of the variable
- "[n]" is the access index number of the element
- "value" is the value assigned to the array element.

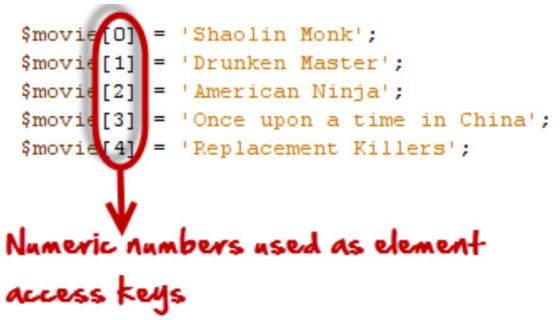
Let's now look at an example of a numeric array.

Suppose we have 5 movies that we want to store in array variables.

We can use the example shown below to do that.

```
<?php

$movie[0] = 'Shaolin Monk';
$movie[1] = 'Drunken Master';
$movie[2] = 'American Ninja';
$movie[3] = 'Once upon a time in China';
$movie[4] = 'Replacement Killers';

?>
```

Here,



Each movie is given an index number that is used to retrieve or modify its value.

Observe the following code-

```
<?php
$movie[0]="Shaolin Monk";
$movie[1]="Drunken Master";
$movie[2]="American Ninja";
$movie[3]="Once upon a time in China";
$movie[4]="Replacement Killers";
echo $movie[3];
$movie[3] = " Eastern Condors";
echo $movie[3];
?>
```

Output:

```
Once upon a time in China Eastern Condors
```

As you can see from the above examples, working with arrays in PHP when dealing with multiple values of the same nature is very easy and flexible.

Alternatively, the above array variables can also be created using the following code.

```
<?php
$movie = array(0 => "Shaolin Monk",
            1 => "Drunken Master",
            2 => "American Ninja",
            3 => "Once upon a time in China",
            4 =>"Replacement Killers" );
echo $movie[4];
?>
```

Output:

```
Replacement Killers
```

PHP Associative Array

Associative array differ from numeric array in the sense that associative arrays use descriptive names for id keys.

Below is the syntax for creating associative array in php.

```
<?php
$variable_name['key_name'] = value;

$variable_name = array('keyname' => value);
?>
```

HERE,

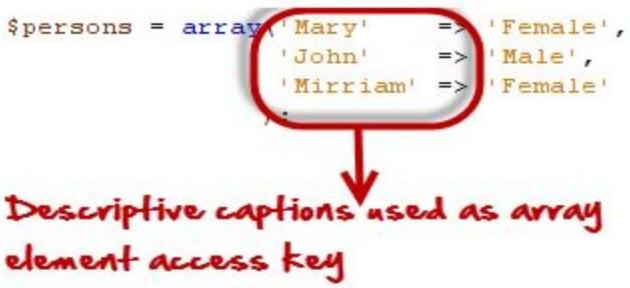
- "$variable_name…" is the name of the variable
- "['key_name']" is the access index number of the element
- "value" is the value assigned to the array element.

Let's suppose that we have a group of persons, and we want to assign the gender of each person against their names.

We can use an associative array to do that.The code below helps us to do that.

```
<?php
$persons = array("Mary" => "Female", "John" => "Male", "Mirriam" => "Female");
print_r($persons);
echo "";
echo "Mary is a " . $persons["Mary"];
?>
```

HERE,



Output:

```
Array ( [Mary] => Female [John] => Male [Mirriam] => Female ) Mary is a Female
```

Associative array are also very useful when retrieving data from the database. The field names are used as id keys.

PHP Multi-dimensional arrays

These are arrays that contain other nested arrays.

The advantage of multidimensional arrays is that they allow us to group related data together.

Let’s now look at a practical example that implements a php multidimensional array.

The table below shows a list of movies by category.

Movie title	Category
Pink Panther	Comedy
John English	Comedy
Die Hard	Action
Expendables	Action
The Lord of the rings	Epic
Romeo and Juliet	Romance
See no evil hear no evil	Comedy

The above information can be represented as a multidimensional array. The code below shows the implementation.

```
<?php
$movies =array(
"comedy" => array("Pink Panther", "John English", "See no evil hear no evil"),
"action" => array("Die Hard", "Expendables"),
"epic" => array("The Lord of the rings"),
"Romance" => array("Romeo and Juliet")
);
print_r($movies);
?>
```

HERE,

Output:

```
Array ( [comedy] => Array ( [0] => Pink Panther [1] => John English [2] => See no evil hear no evil ) [action] => Array ( [0] => Die Hard [1] => Expendables ) [epic] => Array ( [0] => The Lord of the rings ) [Romance] => Array ( [0] => Romeo and Juliet ) )
```

Another way to define the same array is as follows

```
<?php
$film=array(

        "comedy" => array(

                0 => "Pink Panther",

                1 => "john English",

                2 => "See no evil hear no evil"

                ),

        "action" => array (

                0 => "Die Hard",

                1 => "Expendables"

                ),

        "epic" => array (

                0 => "The Lord of the rings"

                ),

        "Romance" => array

                (

                0 => "Romeo and Juliet"

                )

);
echo $film["comedy"][0];
?>
```

Output:

```
Pink Panther
```

Note: the movies numeric array has been nested inside the categories associative array

PHP Arrays: Operators

Operator	Name	Description	How to do it	Output
x + y	Union	Combines elements from both arrays	<?php $x = array('id' => 1); $y = array('value' => 10); $z = $x + $y; ?>	Array([id] => 1 [value] => 10)

X == y	Equal	Compares two arrays if they are equal and returns true if yes.	<?php
			$x = array("id" => 1);
			
			$y = array("id" => "1");
			
			if($x == $y)
			{
			echo "true";
			}
			else
			{
			echo "false";
			
			}
			?>
				True or 1

X === y	Identical	Compares both the values and data types	<?php
			$x = array("id" => 1);
			
			$y = array("id" => "1");
			
			if($x === $y)
			{
			echo "true";
			}
			else
			{
			echo "false";
			}
			?>
				False or 0

			<?php $x = array("id" => 1); $y = array("id" => "1");	

X != y, x <> y	Not equal		if($x != $y)
			{
			echo "true";
			}
			else
			{
			echo "false";
			}
			?>
				False or 0

X !== y	Non identical		<?php
			$x = array("id" => 1);
			
			$y = array("id" => "1");
			
			if($x !== $y)
			{
			echo "true";
			}
			else
			{
			echo "false";
			}
			?>
				True or 1

PHP Array Functions

Count function

The count function is used to count the number of elements that an php array contains. The code below shows the implementation.

```
<?php
$lecturers = array("Mr. Jones", "Mr. Banda", "Mrs. Smith");
echo count($lecturers);
?>
```

Output:

```
3
```

is_array function

The is_array function is used to determine if a variable is an array or not. Let’s now look at an example that implements the is_array functions.

```
<?php
$lecturers = array("Mr. Jones", "Mr. Banda", "Mrs. Smith");
echo is_array($lecturers);
?>
```

Output:

```
1
```

Sort

This function is used to sort arrays by the values.

If the values are alphanumeric, it sorts them in alphabetical order.

If the values are numeric, it sorts them in ascending order.

It removes the existing access keys and add new numeric keys.

The output of this function is a numeric array

```
<?php
$persons = array("Mary" => "Female", "John" => "Male", "Mirriam" => "Female");

sort($persons);

print_r($persons);
?>
```

Output:

```
Array ( [0] => Female [1] => Female [2] => Male )
```

ksort

This function is used to sort the array using the key. The following example illustrates its usage.

```
<?php
$persons = array("Mary" => "Female", "John" => "Male", "Mirriam" => "Female");

ksort($persons);

print_r($persons);
?>
```

Output:

```
Array ( [John] => Male [Mary] => Female [Mirriam] => Female )
```

asort

This function is used to sort the array using the values. The following example illustrates its usage.

```
<?php

$persons = array("Mary" => "Female", "John" => "Male", "Mirriam" => "Female");

asort($persons);

print_r($persons);

?>
```

Output:

```
Array ( [Mary] => Female [Mirriam] => Female [John] => Male )
```

Why use arrays?

- Contents of Arrays can be stretched,
- Arrays easily help group related information such as server login details together
- Arrays help write cleaner code.

PHP Control Structures: If else, Switch Case

What is a control structure?

Code execution can be grouped into categories as shown below

- **Sequential** – this one involves executing all the codes in the order in which they have been written.

- **Decision** – this one involves making a choice given a number of options. The code executed depends on the value of the condition.

A control structure is a block of code that decides the execution path of a program depending on the value of the set condition.

Let's now look at some of the control structures that PHP supports.

PHP IF Else

If… then... else is the **simplest control structure**. It evaluates the conditions using Boolean logic

When to use if… then… else

- You have a block of code that should be executed only if a certain condition is true
- You have two options, and you have to select one.
- If… then… else if… is used when you have to select more than two options and you have to select one or more

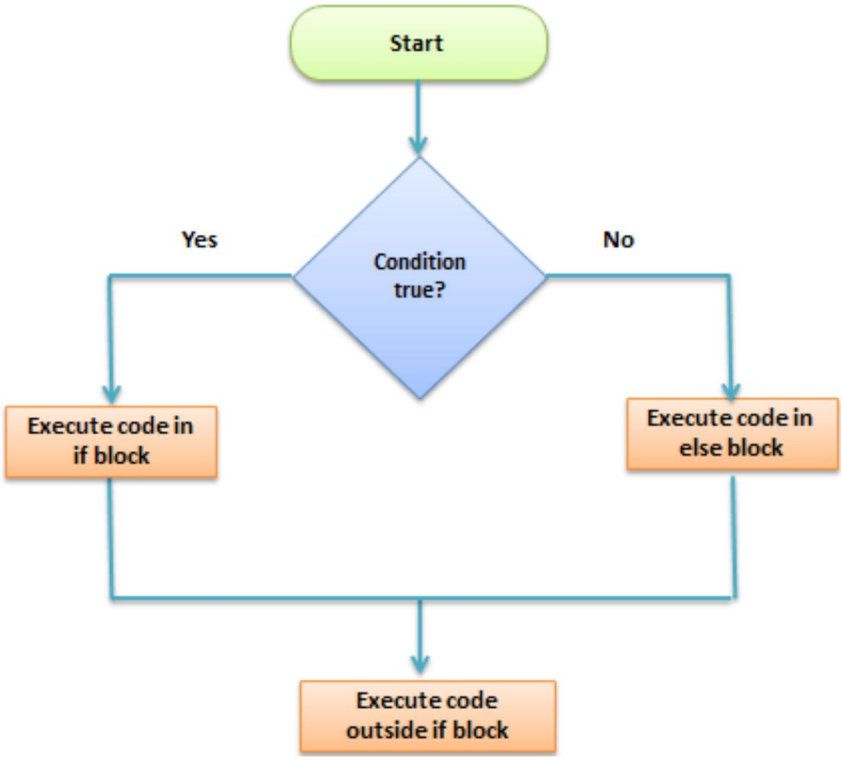
Syntax The syntax for if… then… else is;

```
<?php
if (condition is true) {

block one

else

block two

}
?>
```

HERE,

- "**if (condition is true)"** is the control structure
- "**block one**" is the code to be executed if the condition is true
- **{…else…}** is the fallback if the condition is false
- "**block two**" is the block of code executed if the condition is false

How it works The flow chart shown below illustrates how the if then… else control structure works



Let's see this in action The code below uses "if… then… else" to determine the larger value between two numbers.

```
<?php

$first_number = 7;

$second_number = 21;

if ($first_number > $second_number){

echo "$first_number is greater than $second_number";

}else{

echo "$second_number is greater than $first_number";

}

?>
```

Output:

```
21 is greater than 7
```

PHP Switch Case

Switch… case is similar to the **if then… else** control structure.

It only **executes** a single block of code depending on the **value** of the condition.

If no condition has been met then the default block of code is executed.
It has the following basic syntax.

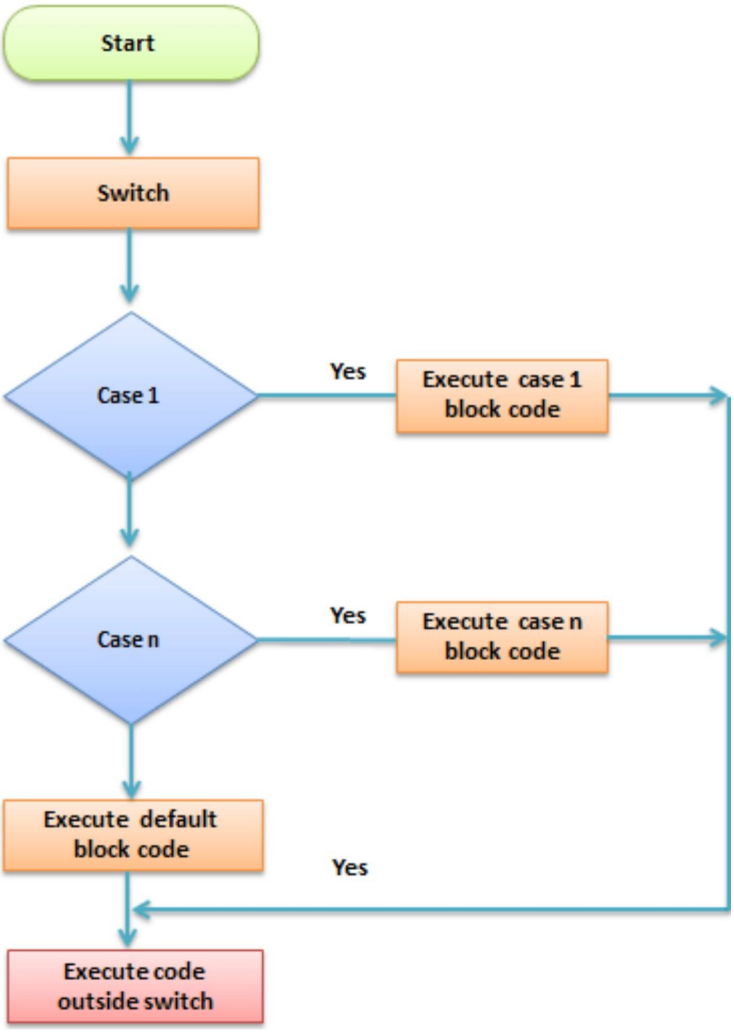
```
<?php
switch(condition){

case value:

//block of code to be executed

break;

case value2:

//block of code to be executed

break;

default:

//default block code

break;

}
?>
```

HERE,

- **“switch(…){…}”** is the control structure block code
- **“case value: case…”** are the blocks of code to be executed depending on the value of the condition
- **“default:”** is the block of code to be executed when no value matches with the condition

How it works

The flow chart shown below illustrates how the switch control structure works



Practical example

The code below uses the switch control structure to display a message depending on the day of the week.

```
<?php

$today = "wednesday";

switch($today){

case "sunday":

echo "pray for us sinners.";

break;

case "wednesday":

echo "ladies night, take her out for dinner";
```

```
break;

case "saturday":

echo "take care as you go out tonight.";

break;

default:

echo "have a nice day at work";

break;

}

?>
```

Output:

```
ladies night, take her out for dinner
```

PHP Loop: For, ForEach, While, Do While [Example]

A Loop is an Iterative Control Structure that involves executing the same number of code a number of times until a certain condition is met.

PHP For Loop

The above code outputs “21 is greater than 7” For loops For... loops execute the block of code a specifiednumber of times. There are basically two types of for loops;

- for
- for… each.

Let’s now look at them separately. **For loop** It has the following basic **syntax**

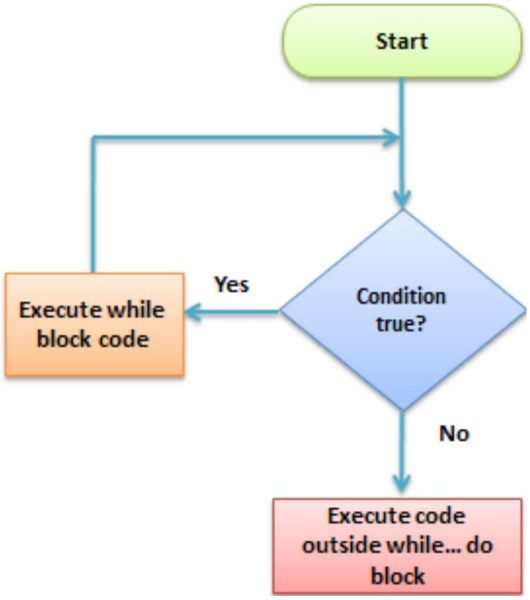
```
<?php
for (initialize; condition; increment){

//code to be executed

}
?>
```

HERE,

- **“for…{…}”** is the loop block
- “**initialize**” usually an integer; it is used to set the counter’s initial value.
- **“condition”** the condition that is evaluated for each php execution. If it evaluates to true then execution of the for... loop is terminated. If it evaluates to false, the execution of the for... loop continues.
- **“increment”** is used to increment the initial value of counter integer.

How it works

The flowchart shown below illustrates how for loop in php works



How to code

The code below uses the “for… loop” to print values of multiplying 10 by 0 through to 10

```
<?php

for ($i = 0; $i < 10; $i++){

$product = 10 * $i;

echo "The product of 10 * $i is $product <br/>";
}

?>
```

Output:

```
The product of 10 x 0 is 0
The product of 10 x 1 is 10
The product of 10 x 2 is 20
The product of 10 x 3 is 30
The product of 10 x 4 is 40
The product of 10 x 5 is 50
The product of 10 x 6 is 60
The product of 10 x 7 is 70
The product of 10 x 8 is 80
The product of 10 x 9 is 90
```

PHP For Each loop

The php foreach loop is used to iterate through array values. It has the following basic syntax

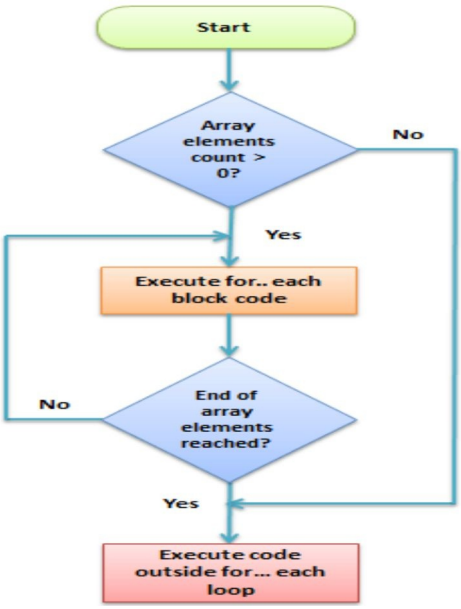
```
<?php
```

```
foreach($array_variable  as $array_values){

block of code to be executed

}
?>
```

HERE,

- **“foreach(…){…}”** is the foreach php loop block code
- **“$array_data”** is the array variable to be looped through
- **“$array_value “** is the temporary variable that holds the current array item values.
- “block of code…” is the piece of code that operates on the array values

How it works The flowchart shown below illustrates how the for… each… loop works



Practical examples

The code below uses for… each loop to read and print the elements of an array.

```
<?php

$animals_list = array("Lion","Wolf","Dog","Leopard","Tiger");

foreach($animals_list as $array_values){

echo $array_values . "<br>";

}

?>
```

Output:

```
Lion
Wolf
Dog
Leopard
Tiger
```

Let’s look at another example that loops through an **associative array**.

An associative array uses alphanumeric words for access keys.

```
<?php

$persons = array("Mary" => "Female", "John" => "Male", "Mirriam" => "Female");

foreach($persons as $key => $value){

echo "$key is $value"."<br>";

}

?>
```

The names have been used as array keys and gender as the values.

Output:

```
Mary is Female
John is Male
Mirriam is Female
```

While Loop

PHP While loop

They are used to execute a block of code a repeatedly until the set condition gets satisfied

When to use while loops

- While loops are used to execute a block of code until a certain condition becomes true.
- You can use a while loop to read records returned from a database query.

Types of while loops

- **Do… while** - executes the block of code at least once before evaluating the condition
- **While…** - checks the condition first. If it evaluates to true, the block of code is executed as long as the condition is true. If it evaluates to false, the execution of the while loop is terminated.

While loop

It has the following syntax

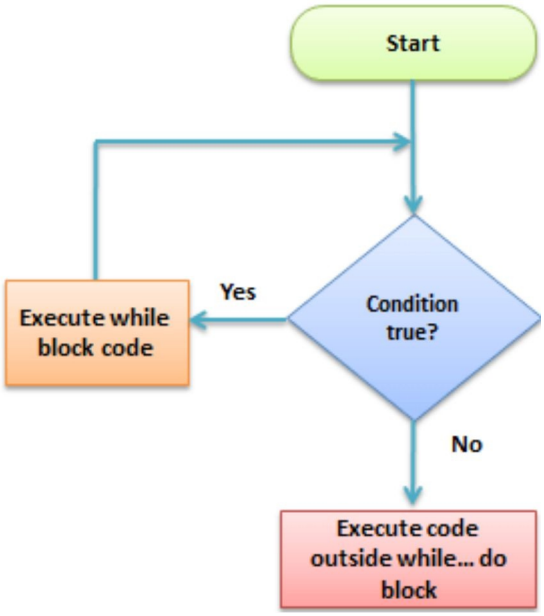
```
<?php
while (condition){

block of code to be executed;

}
?>
```

HERE,

- **“while(…){…}”** is the while loop block code
- **“condition”** is the condition to be evaluated by the while loop
- **“block of code…”** is the code to be executed if the condition gets satisfied

How it works

The flow chart shown below illustrates how the while… loop works



Practical example

The code below uses the while… loop to print numbers 1 to 5.

```
<?php

$i = 0;

while ($i < 5){

echo $i + 1 . "<br>";

$i++;

}

?>
```

Output:

```
1
2
3
4
5
```

PHP Do While

The difference between While… loop and Do… while loop is do… while is executed at-least once before the condition is evaluated.

Let’s now look at the basic syntax of a do… while loop

```
<?php
do{
```

```
block of code to be executed

}
?>
```

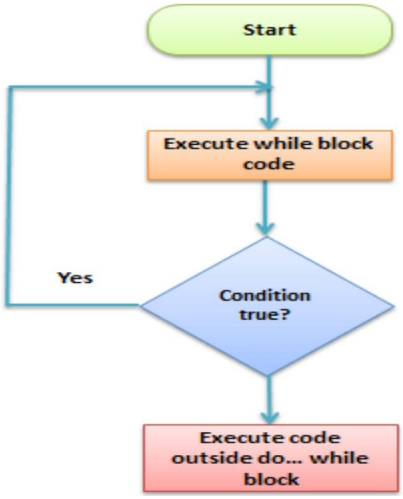
```
while(condition);
```

HERE,

- **“do{…} while(…)”** is the do… while loop block code
- **“condition”** is the condition to be evaluated by the while loop
- **“block of code…”** is the code that is executed at least once by the do… while loop

How it works

The flow chart shown below illustrates how the while… loop works



Practical example

We are now going to modify the while… loop example and implement it using the do… while loop and set the counter initial value to 9.

The code below implements the above modified example

```
<?php

$i = 9;

do{

   echo "$i is"." <br>";

}

while($i < 9);
```

```
?>
```

The above code outputs:

```
9
```

***Note** the above example outputs 9 only.*

This is because the do… while loop is executed at least once even if the set condition evaluates to false.

PHP String Functions: substr, strlen, strtolower, explode, strpos, str_replace

What is a string?

A string is a collection of characters. String is one of the data types supported by PHP.

The string variables can contain alphanumeric characters. Strings are created when;

- You declare variable and assign string characters to it
- You can directly use them with echo statement.
- String are language construct, it helps capture words.
- Learning how strings work in PHP and how to manipulate them will make you a very effective and productive developer.

PHP Create strings

Let’s now look at the four different ways of creating strings.

Creating Strings Using Single quotes: The simplest way to create a string is to use single quotes.

Let’s look at an example that creates a simple string in PHP.

```
<?php
   var_dump("You need to be logged in to view this page");
?>
```

Output:

```
string(42) "You need to be logged in to view this page"
```

If the single quote is part of the string value, it can be escaped using the backslash.

The code below illustrates how to escape a single quote.

```
<?php
```

```
echo "I \'ll be back after 20 minutes";
?>
```

Output:

```
I \'ll be back after 20 minutes
```

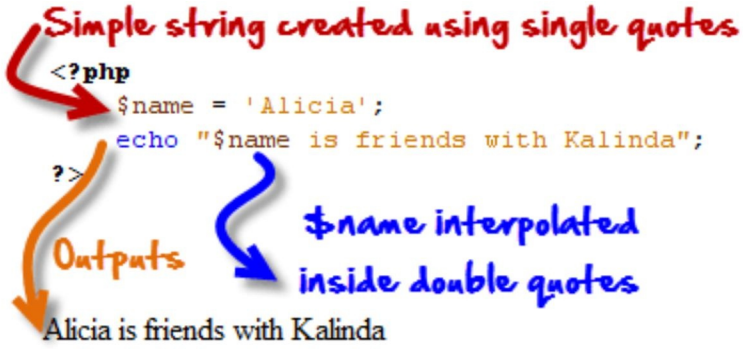
PHP Create Strings Using Double quotes

The double quotes are used to create relatively complex strings compared to single quotes.

Variable names can be used inside double quotes and their values will be displayed.

Let’s look at an example.

```
<?php
$name='Alicia';
echo "$name is friends with kalinda";
?>
```



HERE,

- The above example creates a simple string with the value of Alicia.
- The variable name is then used in the string created using double quotes and its value is interpolated at run time.

Output:

```
Alicia is friends with kalinda
```

In addition to variable interpolations, the double quote string can also escape more special characters such as “\n for a linefeed, \$ dollar for the dollar sign” etc.

More examples Let’s suppose that we have the following code

```
<?php $pwd = "pas$word"; echo $pwd; ?>
```

Output:

```
NOTICE : Undefined variable
pas
```

executing the above codes issues a notice “Notice: Undefined variable”.

This is because $word is treated as a variable.

If we want the dollar sign to be treated as a literal value, we have to escape it.

```
<?php
$word="word";
$pwd = "pas\$word";
echo $pwd; ?>
```

Output:

```
pas$word
```

PHP Heredoc

This heredoc methodology is used to create fairly complex strings as compared to double quotes.

The heredoc supports all the features of double quotes and allows creating string values with more than one line without php string concatenation.

Using double quotes to create strings that have multiple lines generates an error.

You can also use double quotes inside without escaping them.

The example below illustrates how the Heredoc method is used to create string values.

```
<?php

$baby_name = "Shalon";

echo <<<EOT

   When $baby_name was a baby,

   She used to look like a "boy".

EOT;

?>
```

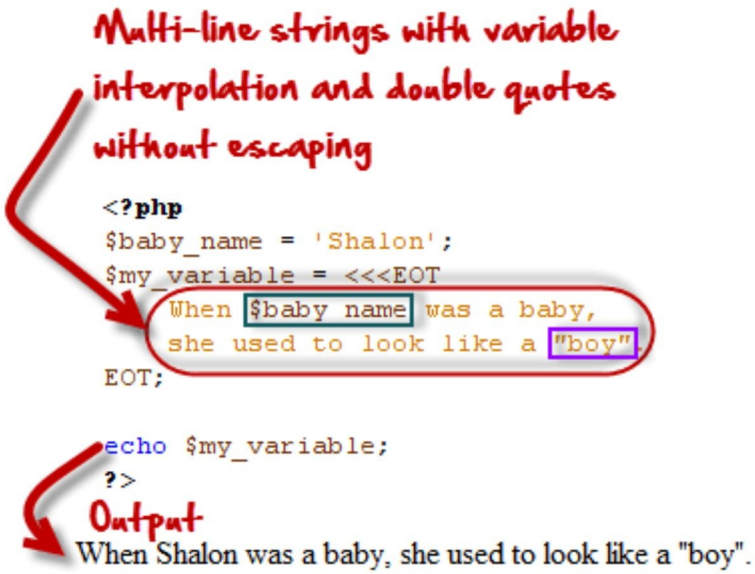
HERE,

<<<EOT is the string delimiter.

EOT is the acronym for end of text.

It should be defined in its on line at the beginning of the string and at the end.

Note: you can use anything you like in place of EOT



Output:

```
When Shalon was a baby, She used to look like a "boy".
```

PHP Nowdoc

The Nowdoc string creation method is similar to the heredoc method but works like the way single quotes work.

No parsing takes place inside the Nowdoc.

Nowdoc is ideal when working with raw data that do not need to be parsed.

The code below shows the Nowdoc implementation

```
<?php

$baby_name = "Shalon";

$my_variable = <<<'EOT'

   When $baby_name was a baby,

   She used to look like a "boy".

EOT;

echo $my_variable;

?>
```

Output:

```
When $baby_name was a baby, She used to look like a "boy".
```

PHP string functions

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	echo str_replace ('the', 'that', 'the	that laptop is very expensive

	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.	laptop is very expensive');	
strpos	Used to locate the and return the position of a character(s) within a string. This function accepts two arguments	echo strpos('PHP Programing','Pro');	4
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

For a complete list of PHP strings, check http://php.net/manual/en/ref.strings.php

PHP Function: Numeric, Built in, String,

Date, User Defined

What is a Function?

A function is a reusable piece or block of code that performs a specific action.

Functions can either return values when called or can simply perform an operation without returning any value.

PHP has over 700 functions built in that perform different tasks.

Why use Functions?

- Better code organization – functions allow us to group blocks of related code that perform a specific task together.
- Reusability – once defined, a function can be called by a number of scripts in our PHP files. This saves us time of reinventing the wheel when we want to perform some routine tasks such as connecting to the database
- Easy maintenance- updates to the system only need to be made in one place.

Built in Functions

Built in functions are functions that exist in PHP installation package.

These built in functions are what make PHP a very efficient and productive scripting language.

The built in functions can be classified into many categories. Below is the list of the categories.

String Functions

These are functions that manipulate string data, refer to the article on strings for implementation examples of string functions

Numeric Functions

Numeric functions are function that return numeric results.

Numeric php function can be used to format numbers, return constants, perform mathematical computations etc.

The table below shows the common PHP numeric functions

Function	Description	Example	Output
is_number	Accepts an argument	<?php if(is_numeric("guru"))	false

	and returns true if its numeric and false if it’s not	{
		  echo "true";
		}
		else
		{
		  echo "false";
		}
		?>	

		<?php
		if(is_numeric (123))
		{
		  echo "true";
		}
		else
		{
		  echo "false";
		}
		?>
			true

number_format	Used to formats a numeric value using digit separators and decimal points	<?php echo number_format(2509663); ?>	2,509,663
rand	Used to generate a random number.	<?php echo rand(); ?>	Random number
round	Round off a number with decimal points to the nearest whole number.	<?php echo round(3.49); ?>	3
sqrt	Returns the square root of a number	<?php echo sqrt(100); ?>	10
cos	Returns the cosine	<?php echo cos(45); ?>	0.52532198881773
sin	Returns the sine	<?php echo sin(45); ?>	0.85090352453412
tan	Returns the tangent	<?php echo tan(45); ?>	1.6197751905439
pi	Constant that returns the value of PI	<?php echo pi(); ?>	3.1415926535898

Date Function

The date function is used to format Unix date and time to human readable format.

Check the article on PHP date functions for more details.

Other functions

These include;

- Arrays – see the article on arrays for examples
- Files – see the article on files for examples
- Database functions – see the article on MySQL PHP and other database access methods v2

Why use User Defined Functions?

User defined functions come in handy when;

- you have routine tasks in your application such as adding data to the database
- performing validation checks on the data
- Authenticating users in the system etc.

These activities will be spread across a number of pages.

Creating a function that all these pages can be calling is one of the features that make PHP a powerful scripting language.

Before we create our first user defined function, let's look at the rules that we must follow when creating our own functions.

- Function names must start with a letter or an underscore but not a number
- The function name must be unique
- The function name must not contain spaces
- It is considered a good practice to use descriptive function names.
- Functions can optionally accept parameters and return values too.

Let's now create our first function. We will create a very basic function that illustrates the major components of a function in PHP.

```
<?php

//define a function that displays hello function

function add_numbers(){
```

```
        echo 1 + 2;
}
add_numbers ();
?>
```

Output:

```
3
```

HERE,

- “function…(){…}” is the function block that tells PHP that you are defining a custom function
- “add_numbers” is the function name that will be called when using the function.
- “()” can be used to pass parameters to the function.
- “echo 'Hello function!';” is the function block of code that is executed. It could be any code other than the one used in the above example.

Let’s now look at a fairly complex example that accepts a parameter and display a message just like the above function.

Suppose we want to write a function that prints the user name on the screen, we can write a custom function that accepts the user name and displays it on the screen.

The code below shows the implementation.

```
<?php
function display_name($name)
{
echo "Hello " . $name;
}
display_name("Martin Luther King");
?>
```

Output:

```
Hello Martin Luther King
```

HERE,

- “…($name){…” is the function parameter called name and is initialized to nameless. If no parameter is passed to the function, nameless will be displayed as the name. This comes in handy if not supplying any parameter to the function can result in unexpected errors.

Let’s now look at a function that accepts a parameter and then returns a value. We will create a function that converts kilometers to miles. The kilometers will be passed as a parameter. The function will return the miles equivalent to the passed kilometers. The code below shows the implementation.

```
<?php
function kilometers_to_miles($kilometers = 0)
{
$miles_scale = 0.62;
return $kilometers * $miles_scale;
}
echo kilometers_to_miles(100);
?>
```

Output:

```
62
```

Summary

- Functions are blocks of code that perform specific tasks
- Built in functions are functions that are shipped with PHP
- PHP has over 700 built in functions
- String functions manipulate string data
- Numeric functions manipulate numeric data
- Date functions manipulate date data
- Other functions such as is_array, fopen etc. are used to manipulate arrays and files respectively
- User defined functions are functions that you can create yourself to enhance PHP

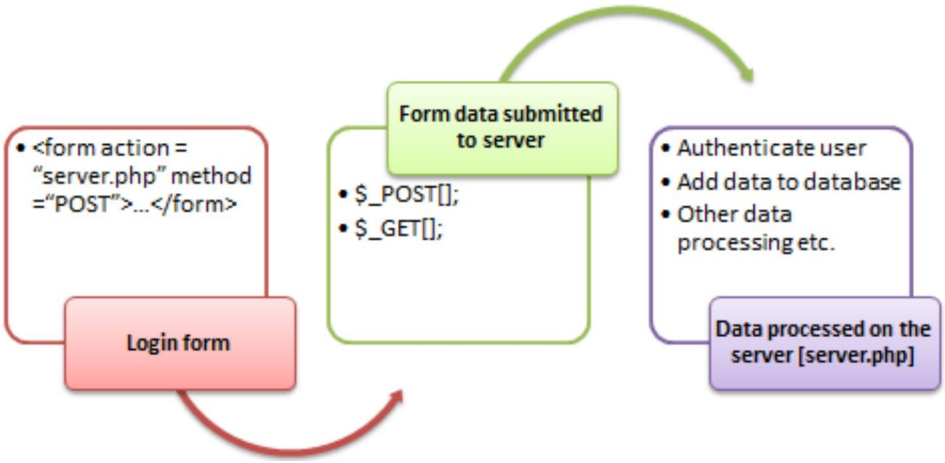
PHP Registration Form using GET, POST Methods with Example

What is Form?

When you login into a website or into your mail box, you are interacting with a form.

Forms are used to get input from the user and submit it to the web server for processing.

The diagram below illustrates the form handling process.



A form is an HTML tag that contains graphical user interface items such as input box, check boxes radio buttons etc.

The form is defined using the <form>...</form> tags and GUI items are defined using form elements such as input.

When and why we are using forms?

- Forms come in handy when developing flexible and dynamic applications that accept user input.
- Forms can be used to edit already existing data from the database

Create a form

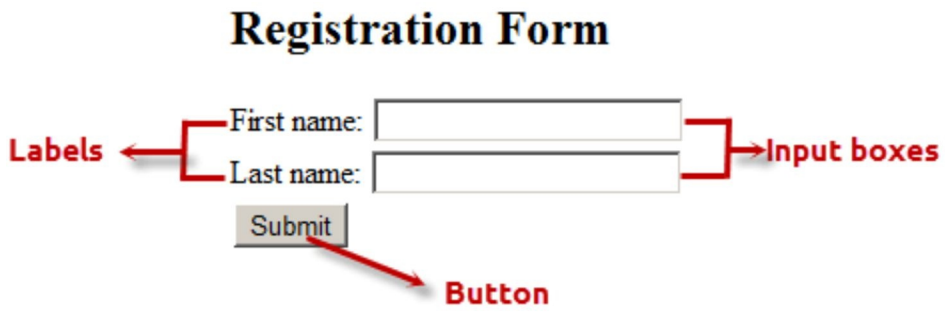
We will use HTML tags to create a form. Below is the minimal list of things you need to create a form.

- Opening and closing form tags <form>…</form>
- Form submission type POST or GET
- Submission URL that will process the submitted data
- Input fields such as input boxes, text areas, buttons,checkboxes etc.

The code below creates a simple registration form

```
<html>
<head>
        <title>Registration Form</title>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
</head>
<body>

  <h2>Registration Form</h2>

  <form action="registration_form.php" method="POST"> First name:

    <input type="text" name="firstname"> <br> Last name:

    <input type="text" name="lastname">

    <input type="hidden" name="form_submitted" value="1" />

    <input type="submit" value="Submit">

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

Viewing the above code in a web browser displays the following form.



HERE,

- <form…>…</form> are the opening and closing form tags

- action="registration_form.php" method="POST"> specifies the destination URL and the submission type.
- First/Last name: are labels for the input boxes
- <input type=”text”…> are input box tags
-
 is the new line tag
- <input type="hidden" name="form_submitted" value="1"/> is a hidden value that is used to check whether the form has been submitted or not
- <input type="submit" value="Submit"> is the button that when clicked submits the form to the server for processing

Submitting the form data to the server

The action attribute of the form specifies the submission URL that processes the data. The method attribute specifies the submission type.

PHP POST method

- This is the built in PHP super global array variable that is used to get values submitted via HTTP POST method.
- The array variable can be accessed from any script in the program; it has a global scope.
- This method is ideal when you do not want to display the form post values in the URL.
- A good example of using post method is when submitting login details to the server.

It has the following syntax.

```
<?php
$_POST['variable_name'];
?>
```

HERE,

- "$_POST[…]" is the PHP array
- "'variable_name'" is the URL variable name.

PHP GET method

- This is the built in PHP super global array variable that is used to get values submitted via HTTP GET method.
- The array variable can be accessed from any script in the program; it has a global scope.
- This method displays the form values in the URL.
- It's ideal for search engine forms as it allows the users to book mark the results.

It has the following syntax.

```
<?php
$_GET['variable_name'];
?>
```

HERE,

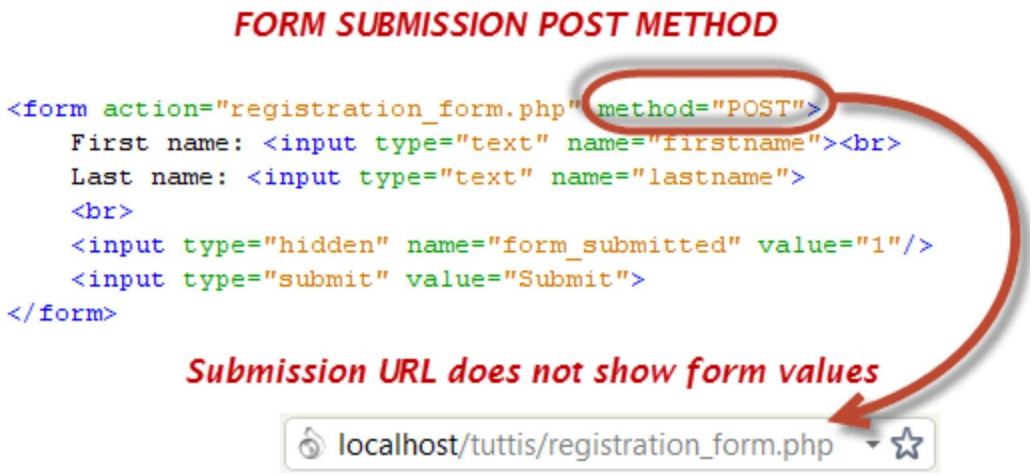
- "$_GET[…]" is the PHP array

- “'variable_name'” is the URL variable name.

GET vs POST Methods

POST	GET
Values not visible in the URL	Values visible in the URL
Has not limitation of the length of the values since they are submitted via the body of HTTP	Has limitation on the length of the values usually 255 characters. This is because the values are displayed in the URL. Note the upper limit of the characters is dependent on the browser.
Has lower performance compared to Php_GET method due to time spent encapsulation the Php_POST values in the HTTP body	Has high performance compared to POST method dues to the simple nature of appending the values in the URL.
Supports many different data types such as string, numeric, binary etc.	Supports only string data types because the values are displayed in the URL
Results cannot be book marked	Results can be book marked due to the visibility of the values in the URL

The below diagram shows the difference between get and post



FORM SUBMISSION GET METHOD

```
<form action="registration_form.php" method="GET">
    First name: <input type="text" name="firstname"><br>
    Last name: <input type="text" name="lastname">
    <br>
    <input type="hidden" name="form_submitted" value="1"/>
    <input type="submit" value="Submit">
</form>
```

SUBMISSION URL SHOWS FORM VALUES

Processing the registration form data

The registration form submits data to itself as specified in the action attribute of the form.

When a form has been submitted, the values are populated in the $_POST super global array.

We will use the PHP isset function to check if the form values have been filled in the $_POST array and process the data.

We will modify the registration form to include the PHP code that processes the data. Below is the modified code

```
<html>
<head>
        <title>Registration Form</title>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">

</head>
<body>

  <?php if (isset($_POST['form_submitted'])): ?> //this code is executed when the form is submitted

    <h2>Thank You <?php echo $_POST['firstname']; ?> </h2>

    <p>You have been registered as
      <?php echo $_POST['firstname'] . ' ' . $_POST['lastname']; ?>
    </p>

    <p>Go <a href="/registration_form.php">back</a> to the form</p>

    <?php else: ?>

      <h2>Registration Form</h2>

      <form action="registration_form.php" method="POST">

        First name:
        <input type="text" name="firstname">

        <br> Last name:
        <input type="text" name="lastname">

                        <input type="hidden" name="form_submitted" value="1" />

        <input type="submit" value="Submit">

      </form>

  <?php endif; ? >
```

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

HERE,

- <?php if (isset($_POST['form_submitted'])): ?> checks if the form_submitted hidden field has been filled in the $_POST[] array and display a thank you and first name message.

  If the form_fobmitted field hasn’t been filled in the $_POST[] array, the form is displayed.

More examples

Simple search engine

We will design a simple search engine that uses the PHP_GET method as the form submission type.

For simplicity’s sake, we will use a PHP If statement to determine the output.

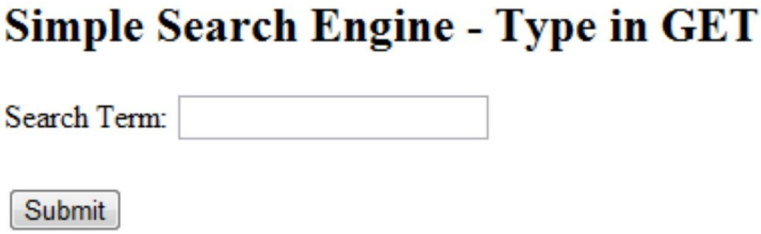
We will use the same HTML code for the registration form above and make minimal modifications to it.

```
<html>
<head>
            <title>Simple Search Engine</title>
            <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
</head>
<body>

  <?php if (isset($_GET['form_submitted'])): ?>

     <h2>Search Results For <?php echo $_GET['search_term']; ?> </h2>
     <?php if ($_GET['search_term'] == "GET"): ?>

        <p>The GET method displays its values in the URL</p>

        <?php else: ?>
                <p>Sorry, no matches found for your search term</p>

        <?php endif; ?>

           <p>Go <a href="/search_engine.php">back</a> to the form</p>

           <?php else: ?>

             <h2>Simple Search Engine - Type in GET </h2>

             <form action="search_engine.php" method="GET">

                 Search Term:
                 <input type="text" name="search_term">
                 <br>

                                  <input type="hidden" name="form_submitted" value="1" />

               <input type="submit" value="Submit">

            </form>
          <?php endif; ?>
</body>
```

```
</html>
```

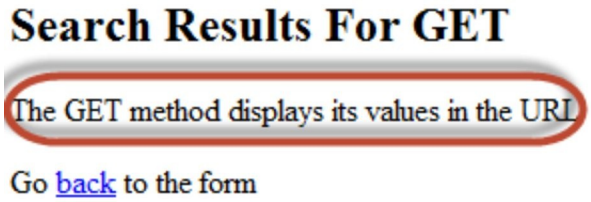
View the above page in a web browser

The following form will be shown

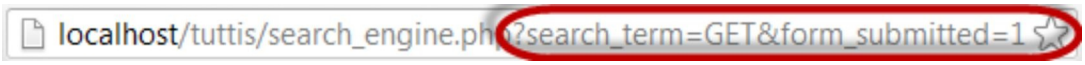


Type GET in upper case letter then click on submit button.

The following will be shown



The diagram below shows the URL for the above results



Note the URL has displayed the value of search_term and form_submitted. Try to enter anything different from GET then click on submit button and see what results you will get.

Working with check boxes, radio buttons

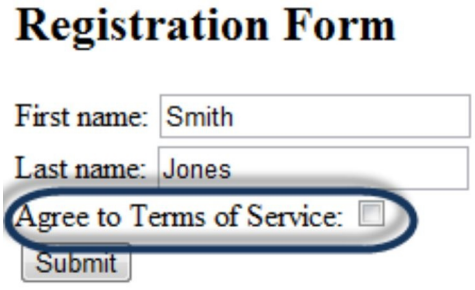
If the user does not select a check box or radio button, no value is submitted, if the user selects a check box or radio button, the value one (1) or true is submitted.

We will modify the registration form code and include a check button that allows the user to agree to the terms of service.

```
<html>
<head>
        <title>Registration Form</title>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
</head>
<body>

  <?php if (isset($_POST['form_submitted'])): ?>

    <?php if (!isset($_POST['agree'])): ?>

      <p>You have not accepted our terms of service</p>
```

```
        <?php else: ?>

            <h2>Thank You <?php echo $_POST['firstname']; ?></h2>

            <p>You have been registered as
                <?php echo $_POST['firstname'] . ' ' . $_POST['lastname']; ?>
            </p>

            <p> Go <a href="/registration_form2.php">back</a> to the form</p>

        <?php endif; ?>

        <?php else: ?>

                <h2>Registration Form</h2>

                <form action="registration_form2.php" method="POST">

                    First name:
                    <input type="text" name="firstname">

                    <br> Last name:
                    <input type="text" name="lastname">

                    <br> Agree to Terms of Service:
                    <input type="checkbox" name="agree">
                    <br>

                    <input type="hidden" name="form_submitted" value="1" />

                    <input type="submit" value="Submit">

                </form>
            <?php endif; ?>
</body>
</html>
```

View the above form in a browser

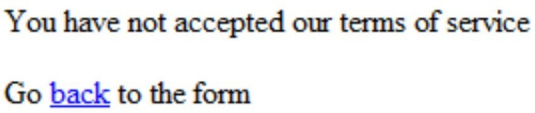


Fill in the first and last names

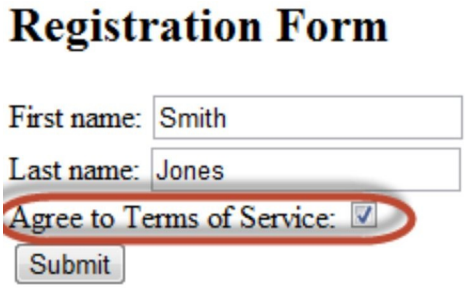
Note the Agree to Terms of Service checkbox has not been selected.

Click on submit button

You will get the following results

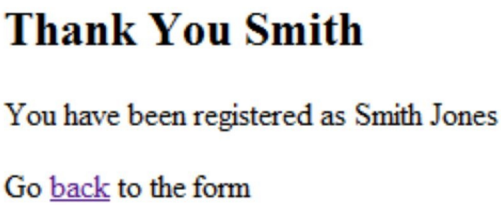


Click on back to the form link and then select the checkbox



Click on submit button

You will get the following results



PHP Session & PHP Cookies with Example

What is Cookie?

A cookie is a small file with the maximum size of 4KB that the web server stores on the client computer.

Once a cookie has been set, all page requests that follow return the cookie name and value.

A cookie can only be read from the domain that it has been issued from. For example, a cookie set using the domain www.guru99.com can not be read from the domain career.guru99.com.

Most of the websites on the internet display elements from other domains such as advertising. The domains serving these elements can also set their own cookies. These are known as third party cookies.

A cookie created by a user can only be visible to them. Other users cannot see its

value.

Most web browsers have options for disabling cookies, third party cookies or both.

If this is the case then PHP responds by passing the cookie token in the URL.

The diagram shown below illustrates how cookies work.

Here,

1) A user requests for a page that stores cookies

2) The server sets the cookie on the user's computer

3) Other page requests from the user will return the cookie name and value

Why and when to use Cookies?

- Http is a stateless protocol; cookies allow us to track the state of the application using small files stored on the user's computer.

  The path were the cookies are stored depends on the browser.

  Internet Explorer usually stores them in Temporal Internet Files folder.
- Personalizing the user experience – this is achieved by allowing users to select their preferences.

  The page requested that follow are personalized based on the set preferences in the cookies.
- Tracking the pages visited by a user

Creating Cookies

Let's now look at the basic syntax used to create a cookie.

```
<?php

setcookie(cookie_name, cookie_value, [expiry_time], [cookie_path], [domain], [secure], [httponly]);
```

```
?>
```

HERE,

- Php“setcookie” is the PHP function used to create the cookie.
- “cookie_name” is the name of the cookie that the server will use when retrieving its value from the $_COOKIE array variable. It’s mandatory.
- “cookie_value” is the value of the cookie and its mandatory
- “[expiry_time]” is optional; it can be used to set the expiry time for the cookie such as 1 hour. The time is set using the PHP time() functions plus or minus a number of seconds greater than 0 i.e. time() + 3600 for 1 hour.
- “[cookie_path]” is optional; it can be used to set the cookie path on the server. The forward slash “/” means that the cookie will be made available on the entire domain. Sub directories limit the cookie access to the subdomain.
- “[domain]” is optional, it can be used to define the cookie access hierarchy i.e. www.cookiedomain.com means entire domain while www.sub.cookiedomain.comlimits the cookie access to www.sub.cookiedomain.com and its sub domains. *Note it’s possible to have a subdomain of a subdomain as long as the total characters do not exceed 253 characters.*
- “[secure]” is optional, the default is false. It is used to determine whether the cookie is sent via https if it is set to true or http if it is set to false.
- “[Httponly]” is optional. If it is set to true, then only client side scripting languages i.e. JavaScript cannot access them.

Note: the php set cookie function must be executed before the HTML opening tag.

Let’s now look at an example that uses cookies.

We will create a basic program that allows us to store the user name in a cookie that expires after ten seconds.

The code below shows the implementation of the above example “cookies.php”.

```
<?php
    setcookie("user_name", "Guru99", time()+ 60,'/'); // expires after 60 seconds
    echo 'the cookie has been set for 60 seconds';
?>
```

Output:

```
the cookie has been set for 60 seconds
```

Retrieving the Cookie value

Create another file named “cookies_read.php” with the following code.

```
<?php
    print_r($_COOKIE);    //output the contents of the cookie array variable
?>
```

Output:

```
Array ( [PHPSESSID] => h5onbf7pctbr0t68adugdp2611 [user_name] => Guru99 )
```

Note: $_COOKIE is a PHP built in super global variable.

It contains the names and values of all the set cookies.

The number of values that the

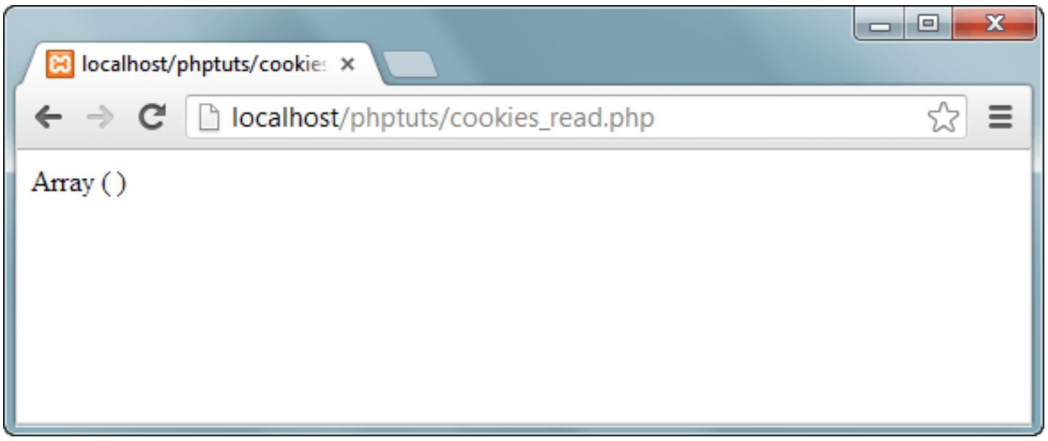
$_COOKIE array can contain depends on the memory size set in php.ini.

The default value is 1GB.

Testing our application.

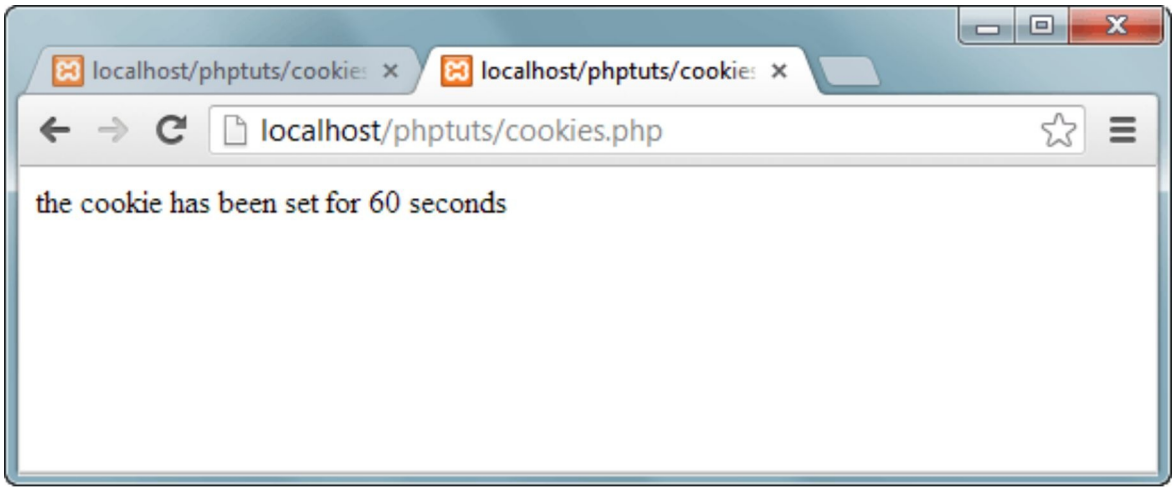
Let’s assume you have saved your PHP files in phptus folder.

- Step 1 – open your web browser and enter the URL **http://localhost/phptuts/cookies.php**



Note: Only an empty array has been displayed

- Step 2 – Browser to the URL **http://localhost/phptuts/cookies.php**



- Step 3 – Switch back to the first tab then click on refresh button

Wait for a minute then click on refresh button again. What results did you get?

Delete Cookies

- If you want to destroy a cookie before its expiry time, then you set the expiry time to a time that has already passed.
- Create a new filed named cookie_destroy.php with the following code

```
<?php

setcookie("user_name", "Guru99", time() - 360,'/');

?>
```

- Repeat steps 1 through to 3 from the above section on retrieving cookie values.

- Open the URL **http://localhost/phptuts/cookie_destroy.php**
- Switch to the URL **http://localhost/phptuts/cookies_read.php** what results does it display?

What is a Session?

- A session is a global variable stored on the server.
- Each session is assigned a unique id which is used to retrieve stored values.
- Whenever a session is created, a cookie containing the unique session id is stored on the user's computer and returned with every request to the server. If the client browser does not support cookies, the unique php session id is displayed in the URL
- Sessions have the capacity to store relatively large data compared to cookies.
- The session values are automatically deleted when the browser is closed. If you want to store the values permanently, then you should store them in the database.
- Just like the $_COOKIE array variable, session variables are stored in the $_SESSION array variable. Just like cookies, the session must be started before any HTML tags.
- You want to store important information such as the user id more securely on the server where malicious users cannot temper with them.
- You want to pass values from one page to another.
- You want the alternative to cookies on browsers that do not support cookies.
- You want to store global variables in an efficient and more secure way compared to passing them in the URL
- You are developing an application such as a shopping cart that has to temporary store information with a capacity larger than 4KB.

Why and when to use Sessions?

- You want to store important information such as the user id more securely on the server where malicious users cannot temper with them.
- You want to pass values from one page to another.
- You want the alternative to cookies on browsers that do not support cookies.

- You want to store global variables in an efficient and more secure way compared to passing them in the URL
- You are developing an application such as a shopping cart that has to temporary store information with a capacity larger than 4KB.

Creating a Session

In order to create a session, you must first call the PHP session_start function and then store your values in the $_SESSION array variable.

Let's suppose we want to know the number of times that a page has been loaded, we can use a session to do that.

The code below shows how to create and retrieve values from sessions

```
<?php

session_start(); //start the PHP_session function

if(isset($_SESSION['page_count']))
{
    $_SESSION['page_count'] += 1;
}
else
{
    $_SESSION['page_count'] = 1;
}
echo 'You are visitor number ' . $_SESSION['page_count'];

?>
```

Output:

```
You are visitor number 1
```

Destroying Session Variables

The session_destroy() function is used to destroy the whole Php session variables.

If you want to destroy only a session single item, you use the unset() function.

The code below illustrates how to use both methods.

```
<?php

session_destroy(); //destroy entire session

?>
<?php

unset($_SESSION['product']); //destroy product session item
```

```
?>
```

Session_destroy removes all the session data including cookies associated with the session.

Unset only frees the individual session variables.

Other data remains intact.

PHP File() Function: File_exists, Fopen, Fwrite, Fclose, Fgets, copy, unlink

What is a File?

A file is simply a resource for storing information on a computer.

Files are usually used to store information such as;

- Configuration settings of a program
- Simple data such as contact names against the phone numbers.
- Images, Pictures, Photos, etc.

PHP File Formats Support

PHP file functions support a wide range of file formats that include;

- File.txt
- File.log
- File.custom_extension i.e. file.xyz
- File.csv
- File.gif, file.jpg etc
- Files provide a permanent cost effective data storage solution for simple data compared to databases that require other software and skills to manage DBMS systems.
- You want to store simple data such as server logs for later retrieval and analysis
- You want to store program settings i.e. program.ini

PHP files Functions

PHP provides a convenient way of working with files via its rich collection of built in functions.

Operating systems such as Windows and MAC OS are not case sensitive

whileLinux or Unix operating systems are case sensitive.

Adopting a naming conversion such as lower case letters only for file naming is a good practice that ensures maximum cross platform compatibility.

Let's now look at some of the most commonly used PHP file functions.

PHP File_exists Function

This function is used to determine whether a file exists or not.

- It comes in handy when we want to know if a file exists or not before processing it.
- You can also use this function when creating a new file and you want to ensure that the file does not already exist on the server.

The file_exist function has the following syntax.

```
<?php
file_exists($filename);
?>
```

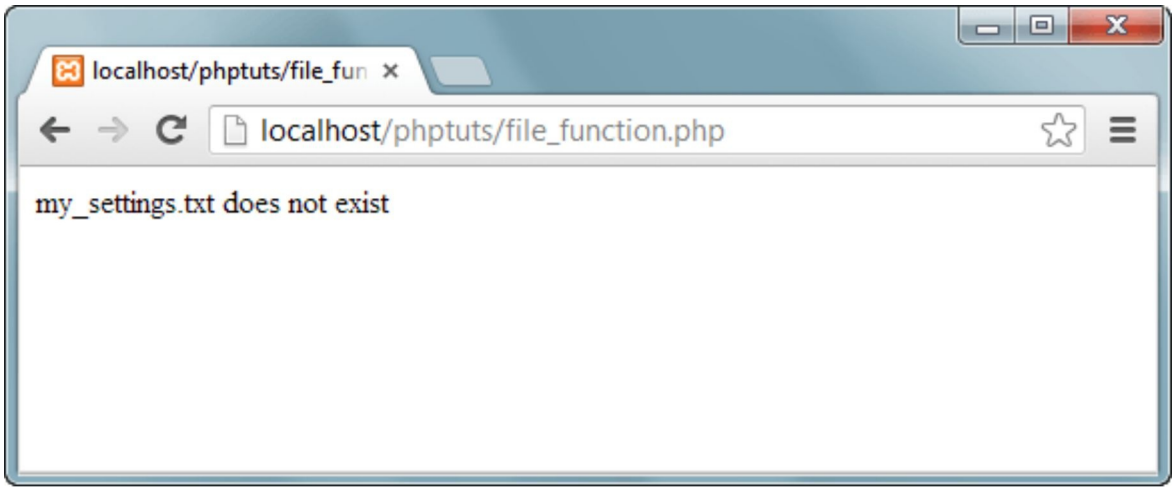
HERE,

- "file_exists()" is the PHP function that returns true if the file exists and false if it does not exist.
- "$file_name" is the path and name of the file to be checked

The code below uses file_exists function to determine if the file my_settings.txt exists.

```
<?php
if (file_exists('my_settings.txt'))
{
	echo 'file found!';
}
else
{
	echo 'my_settings.txt does not exist';
  }
?>
```

Save the above code in a file named file_function.php Assuming you saved the file in phptuts folder in htdocs, open the URL **http://localhost/phptuts/file_function.php** in your browser You will get the following results.



PHP Fopen Function

The fopen function is used to open files. It has the following syntax

```
<?php
fopen($file_name,$mode,$use_include_path,$context);
?>
```

HERE,

- “fopen” is the PHP open file function
- “$file_name” is the name of the file to be opened
- “$mode” is the mode in which the file should be opened, the table below shows the modes

Mode	Description
r	• Read file from beginning. • Returns false if the file doesn’t exist. • Read only
r+	• Read file from beginning • Returns false if the file doesn’t exist. • Read and write
w	• Write to file at beginning • truncate file to zero length • If the file doesn’t exist attempt to create it.

	• Write only
w+	• Write to file at beginning, truncate file to zero length • If the file doesn't exist attempt to create it. • Read and Write
a	• Append to file at end • If the file doesn't exist attempt to create it. • Write only
a+	• Php append to file at end • If the file doesn't exist attempt to create it • Read and write

- "$use_include_path" is optional, default is false, if set to true, the function searches in the include path too.
- "$context" is optional, can be used to specify the context support.

PHP Fwrite Function

The fwrite function is used to write files.

It has the following syntax

```
<?php
fwrite($handle, $string, $length);
?>
```

HERE,

- "fwrite" is the PHP function for writing to files
- "$handle" is the file pointer resource
- "$string" is the data to be written in the file.
- "$length" is optional, can be used to specify the maximum file length.

PHP Fclose Function

Is is used to close a file in php which is already open

It has the following syntax.

```
<?php
fclose($handle);
?>
```

HERE,

- “fclose” is the PHP function for closing an open file
- “$handle” is the file pointer resource.

Let’s now look at an example that creates my_settings.txt.

We will use the following functions.

- Fopen
- Fwrite
- fclose

The code below “create_my_settings_file.php” implements the above example.

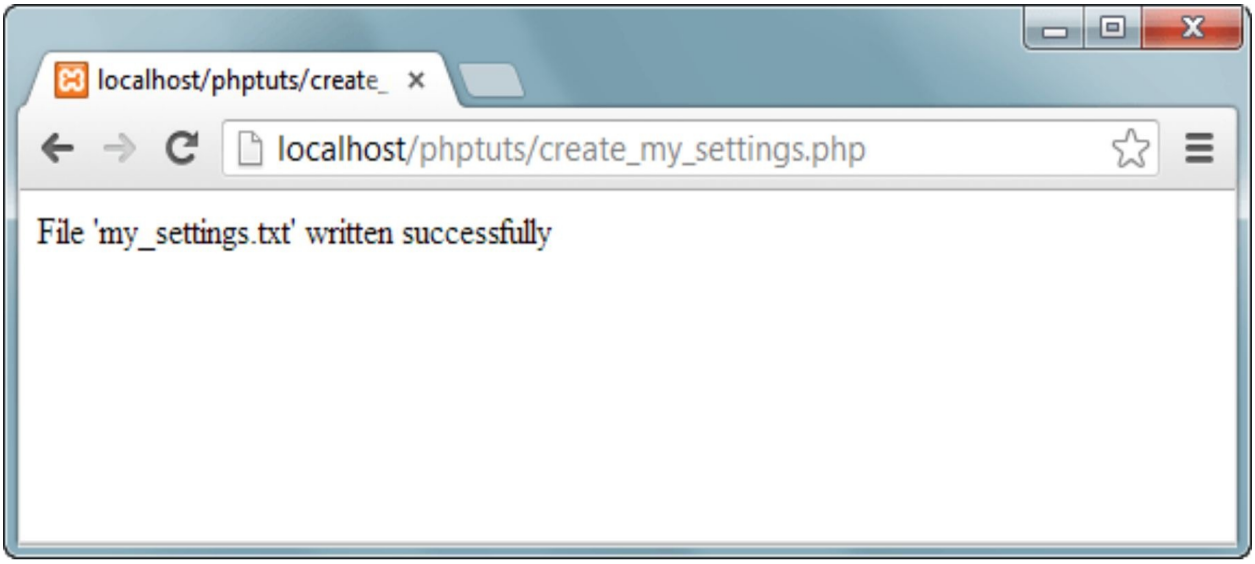
Open a file	<?php $fh = fopen("my_settings.txt", 'w') or die("Failed to create file"); ?>
Closing a file	<?php fclose($fh); ?>
Create File	<?php $fh = fopen("my_settings.txt", 'w') or die("Failed to create file"); $text = <<<_END localhost;root;pwd1234;my_database _END; fwrite($fh, $text) or die("Could not write to file"); fclose($fh); echo "File 'my_settings.txt' written successfully"; ?>

Testing the code

Open the URL **http://localhost/phptuts/create_my_settings.php** in your browser.

You will get the following page

.



Note: if your disk is full or you do not have permission to write files, you will get an error message.

Switch back to the URL **http://localhost/phptuts/file_function.php** .

What results do you get?

PHP Fgets Function

The fgets function is used to read php files line by line. It has the following basic syntax. fgets($handle); HERE,

- "$fgets" is the PHP function for reading file lines
- "$handle" is the file pointer resource.

Let's now look at an example that reads my_settings.txt file using the fopen and fgets functions.

The code below read_my_settings.php implements the above example.

```
<?php
$fh = fopen("my_settings.txt", 'r') or die("File does not exist or you lack permission to open it");
$line = fgets($fh);
echo $line; fclose($fh);
?>
```

HERE,

- "fopen" function returns the pointer to the file specified in the file path
- "die()" function is called if an error occurs. It displays a message and exists execution of the script

PHP Copy Function

The PHP copy function is used to copy files. It has the following basic syntax. copy($file,$copied_file); HERE,

- “$file” specifies the file path and name of the file to be copied.
- “copied_file” specified the path and name of the copied file

The code below illustrates the implementation

```
<?php
copy('my_settings.txt', 'my_settings_backup.txt') or die("Could not copy file");
echo "File successfully copied to 'my_settings_backup.txt'";
?>
```

Deleting a file

The unlink function is used to delete the file. The code below illustrates the implementation.

```
<?php
if (!unlink('my_settings_backup.txt'))
{
    echo "Could not delete file";
}
else
{
    echo "File 'my_settings_backup.txt' successfully deleted";
}
?>
```

PHP File_get_contents Function

The file_get_contents function is used to read the entire file contents.

The code below illustrates the implementation.

The difference between file_get_contents and fgets is that file_get_contents returns the file data as a string while fgets reads the file line by line.

```
<?php
echo "<pre>"; // Enables display of line feeds
echo file_get_contents("my_settings.txt");
echo "</pre>"; // Terminates pre tag
?>
```

Function	**Description**
File_exists	Used to determine if a file exists or not
fopen	Used to open a file. Returns a pointer to the opened file

fwrite	Used to write to files
fclose	Used to open closed files
fgets	Used to read a file line by line
copy	Used to copy an existing file
unlink	Used to delete an existing file
file_get_contents	Used to return the contents of a file as a string

PHP Try Catch Example: Exception & Error Handling Tutorial

What is an Exception?

An error is an unexpected program result that cannot be handled by the program itself.

Errors are resolved by fixing the program. An example of an error would be an infinite loop that never stops executing.

An exception is unexpected program result that can be handled by the program itself.

Examples of exception include trying to open a file that does not exist.

This exception can be handled by either creating the file or presenting the user with an option of searching for the file.

-

Why handle exception?

- Avoid unexpected results on our pages which can be very annoying or irritating to our end users
- Improve the security of our applications by not exposing information which malicious users may use to attack our applications
- Php Exceptions are used to change the normal flow of a program if any predictable error occurs.

PHP Error handling

When an error occurs, depending on your configuration settings, PHP displays

the error message in the web browser with information relating to the error that occurred.

PHP offers a number of ways to handle errors.

We are going to look at three (3) commonly used methods;

1. **Die statements**– the die function combines the echo and exit function in one. It is very useful when we want to output a message and stop the script execution when an error occurs.

2. **Custom error handlers** – these are user defined functions that are called whenever an error occurs.

3. **PHP error reporting** – the error message depending on your PHP error reporting settings. This method is very useful in development environment when you have no idea what caused the error. The information displayed can help you debug your application.

Error handling examples

Let's now look at some simple examples with error handling routines.

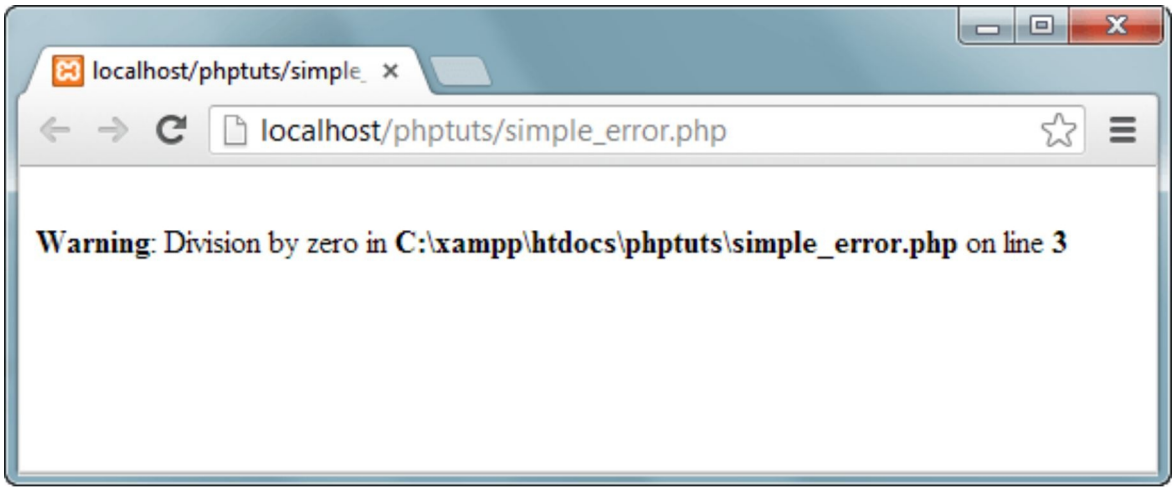
Let's suppose that we have developed an application that uses text files to store data. We might want to check for the file's existence before we attempt to read data from it.

The code below implements the above example.

```
<?php
$denominator = 0;
echo 2 / $denominator;
?>
```

Assuming you saved the file simple_error.php in phptuts folder, open the URL **http://localhost/phptuts/simple_error.php**

You will get the following results

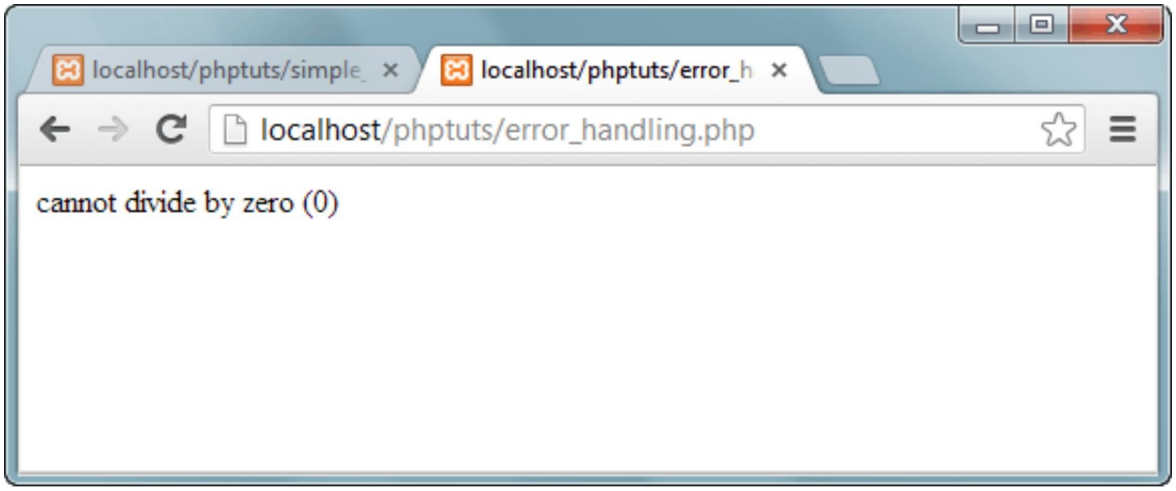


As you can see from the above results, it makes our application look unprofessional and can be annoying to the user.

We will modify the above code and write an error handler for the application

```
<?php
$denominator = 0;
if ($denominator != 0) {
   echo 2 / $denominator;
} else {
   echo "cannot divide by zero (0)";
}
?>
```

Assuming you saved the above code as error_handling.php, open the URL **http://localhost/phptuts/error_handling.php**



Note: it's a good security practice to display a message as the one shown above instead of showing the message like "File not found".

Let's look at another example that uses a custom error handler.

The custom error handler will be set as the default PHP error handling function

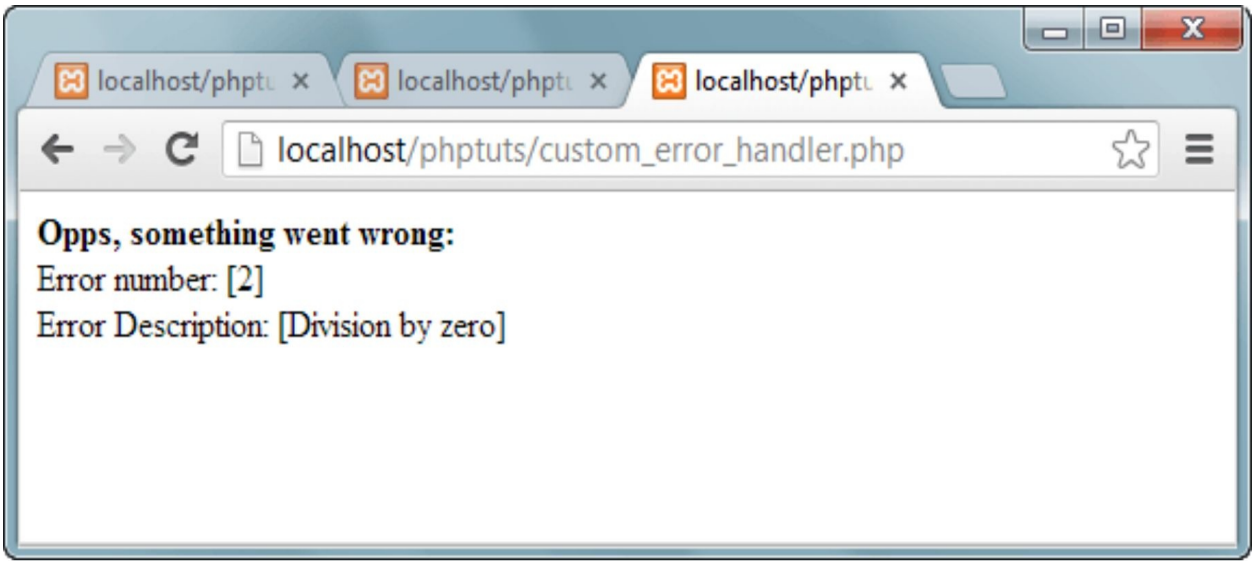
and will basically display an error number and message.

The code below illustrates the implementation of the above example

```
<?php
function my_error_handler($error_no, $error_msg)
{
   echo "Opps, something went wrong:";
   echo "Error number: [$error_no]";
   echo "Error Description: [$error_msg]";
}
set_error_handler("my_error_handler");
echo (5 / 0);
?>
```

Open the URL **http://localhost/phptuts/custom_error_handler.php** you will get the following results

.



As you can see from the above example, custom error handlers are powerful in the sense that

- They allow us to customize the error messages.
- The custom error handler can also include error logging in a file/database, emailing the developer etc.

Let's now look at the third type of error handling. We will be using the PHP built in function error_reporting function. It has the following basic syntax

```
<?php
error_reporting($reporting_level);
?>
```

HERE,

- "error_reporting" is the PHP error reporting function

- “$reporting_level" is optional, can be used to set the reporting level. If no reporting level has been specified, PHP will use the default error reporting level as specified in the php.ini file.

Reporting Level	**Description**	**Example**
E_WARNING	Displays warning messages only. Does not halt the execution of the script	error_reporting(E_WARNING);
E_NOTICE	Displays notices that can occur during normal execution of a program or could be an error.	error_reporting(E_ NOTICE);
E_USER_ERROR	Displays user generated errors i.e. custom error handler	error_reporting(E_ USER_ERROR);
E_USER_WARNING	Displays user generated warning messages	error_reporting(E_USER_WARNIN(
E_USER_NOTICE	Displays user generated notices	error_reporting(E_USER_NOTICE);
E_RECOVERABLE_ERROR	Displays error that	error_reporting(E_RECOVERABLE_

	are not fatal and can be handled using custom error handlers	
E_ALL	Displays all errors and warnings	error_reporting(E_ ALL);

Difference between Errors and Exception

- Exceptions are thrown and intended to be caught while errors are generally irrecoverable.
- Exceptions are handled in an object oriented way.

  This means when an exception is thrown; an exception object is created that contains the exception details.

The table below shows the exception object methods

Method	**Description**	**Example**
getMessage()	Displays the exception's message	<?php echo $e->getMessage(); ?>
getCode()	Displays the numeric code that represents the exception	<?php echo $e->getCode(); ?>
getFile()	Displays the file name and path where the exception occurred	<?php echo $e->getFile(); ?>
getLine()	Displays the line number where the exception occurred	<?php echo $e->getLine(); ?>
getTrace()	Displays an array of the backtrace before the exception	<?php print_r($e->getTrace()); ?>
getPrevious()	Displays the previous exception before the current one	<?php echo $e->getPrevious(); ?>
getTraceAsString()	Displays the backtrace of the exception as a string instead of an array	<?php echo $e->getTraceAsString(); ?>

__toString()	Displays the entire exception as a string	<?php echo $e- >__toString(); ?>

Below is the basic syntax for throwing an exception.

```
<?php
throw new Exception("This is an exception example");
?>
```

HERE,

- “throw” is the keyword used to throw the exception
- “new Exception(…)” creates an exception object and passes “This is an exception example “ string as the message parameter.

The above code outputs the following message.

We are now going to look at an example that implements the throw and catch exceptions.

We will modify the above example and include the try, throw and catch.

It has the following basic syntax.

```
<?php
try {
   //code goes here that could potentially throw an exception
}
catch (Exception $e) {
   //exception handling code goes here
}
?>
```

HERE,

- “try{…}” is the block of code to be executed that could potentially raise an exception
- “catch(Exception $e){…}” is the block of code that catches the thrown

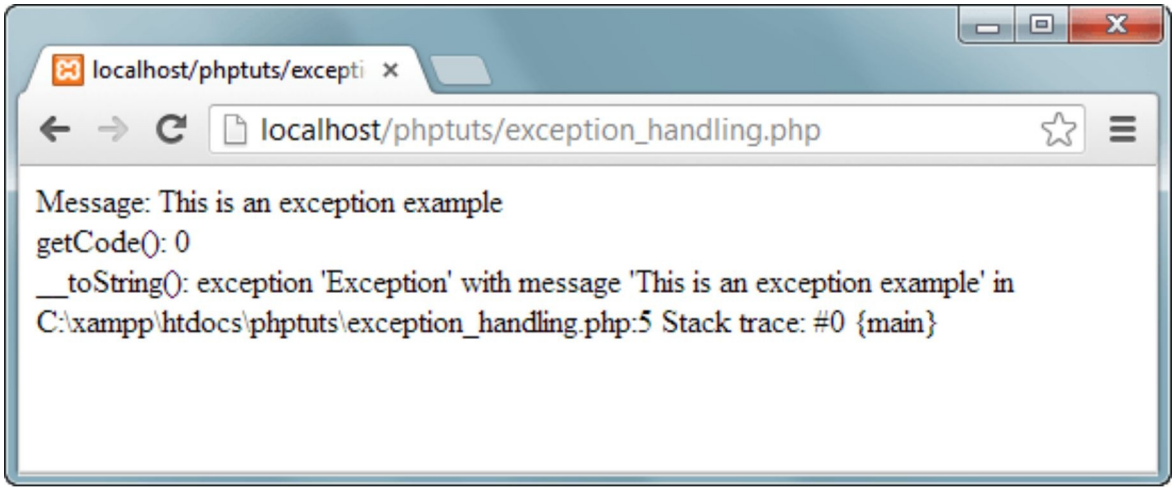
exception and assigns the exception object to the variable $e.

The code below shows the basic exception example with the try, throw and catch exception implemented.

The program deliberately throws an exception which it then catches.

```
<?php
try {
   $var_msg = "This is an exception example";
   throw new Exception($var_msg);
}
catch (Exception $e) {
   echo "Message: " . $e->getMessage();
   echo "";
   echo "getCode(): " . $e->getCode();
   echo "";
   echo "__toString(): " . $e->__toString();
}
?>
```

Open the URL **http://localhost/phptuts/exception_handling.php** You will get the following results.



It’s also possible to create multiple exceptions for one php try statement depending on the type of exception thrown.

See the article on MySQL, PHP data access… for implementation examples of multiple exceptions

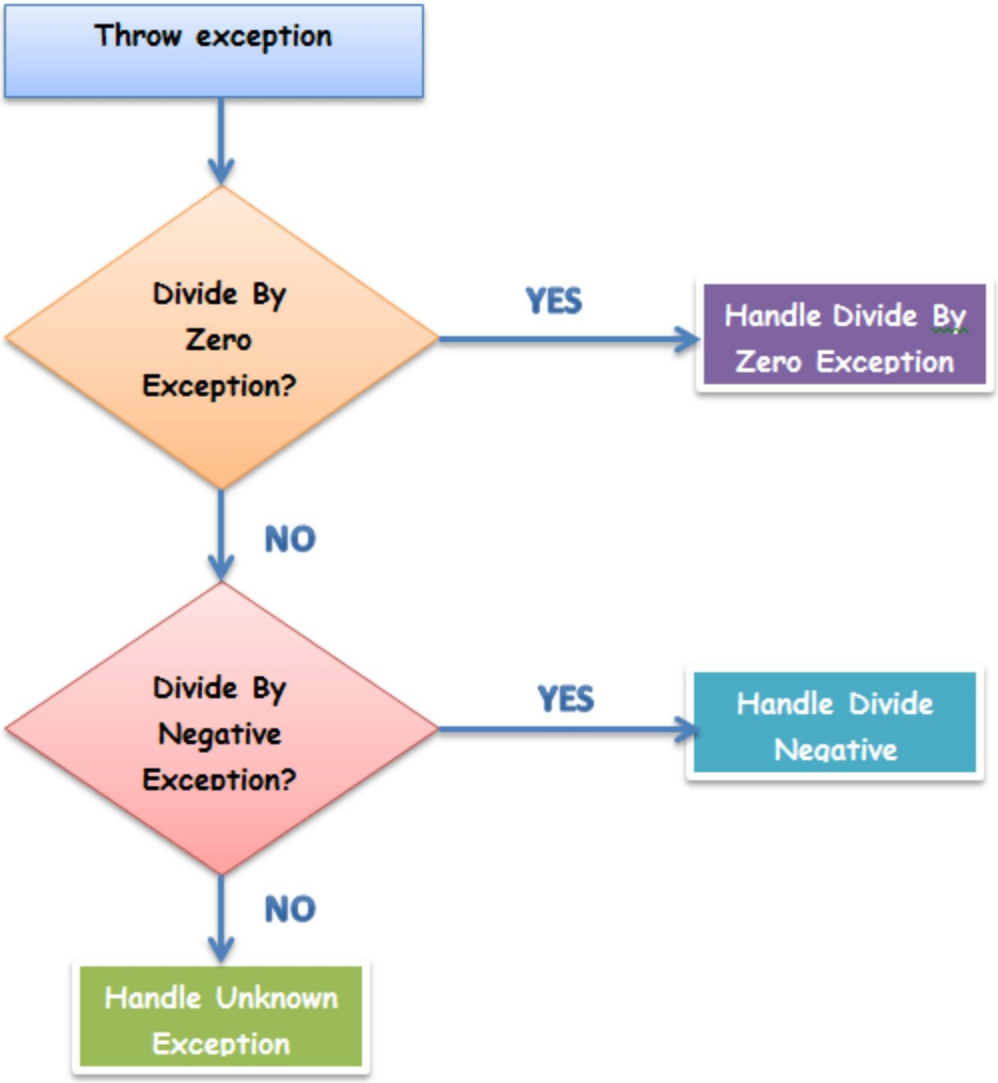
Multiple Exceptions

Multiple exception use multiple try catch blocks to handle the thrown exceptions. Multiple exceptions are useful when;

- You want to display a customized message depending on the exception thrown

- You want to perform a unique operation depending on the exception thrown

The flowchart below illustrates the how multiple exceptions work



Let's look at an example that uses multiple exceptions.

We will modify the code that divides a number by the passed in denominator.

We expect two types of exceptions to occur;

- Division by zero
- Division by a negative number

For simplicity's sake, we will only display the exception type in our catch blocks.

The PHP built in Exception class is used to throw exceptions.

We will create two classes that extend the exception class and use them to throw exceptions.

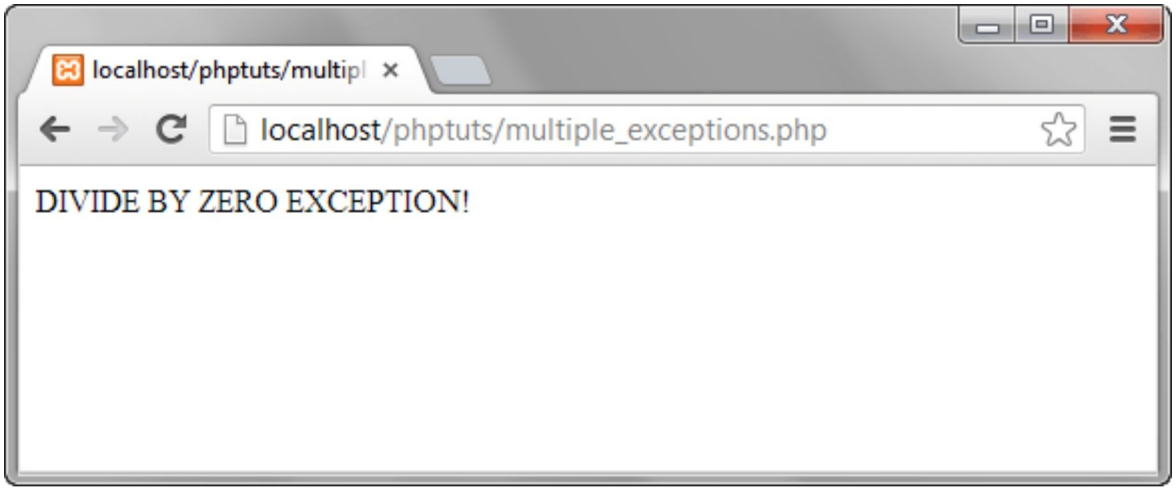
The code below shows the implementation.

```
<?php
class DivideByZeroException extends Exception {};
class DivideByNegativeException extends Exception {};
function process($denominator)
{
	try
	{
		if ($denominator == 0)
		{
			throw new DivideByZeroException();
		}
		else if ($denominator < 0)
		{
			throw new DivideByNegativeException();
		}
		else
		{
			echo 25 / $denominator;
		}
	}
	catch (DivideByZeroException $ex)
	{
		echo "DIVIDE BY ZERO EXCEPTION!";
	}
	catch (DivideByNegativeException $ex)
	{
		echo "DIVIDE BY NEGATIVE NUMBER EXCEPTION!";
	}
	catch (Exception $x)
	{
	echo "UNKNOWN EXCEPTION!";
	}
}
process(0);
?>
```

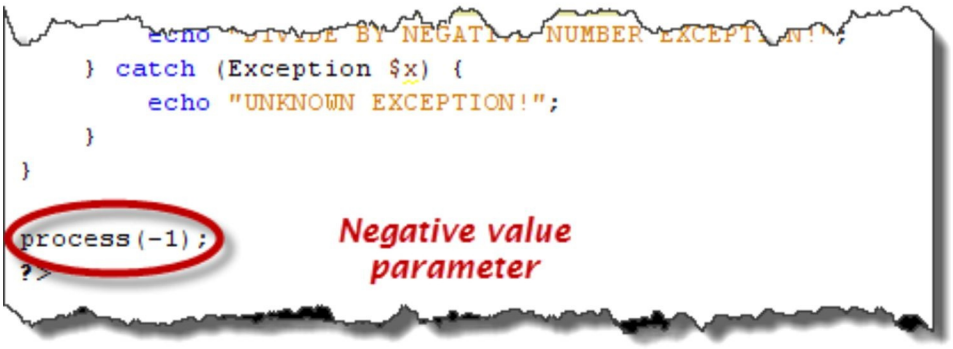
Testing the code

We will assume you saved multiple_exceptions.php in phptuts folder.

Browse to the URL **http://localhost/phptuts/multiple_exceptions.php**



Switch back to the PHP file and pass -1 as the parameter as shown in the following diagram.



Browse to the URL **http://localhost/phptuts/multiple_exceptions.php**.

What results do you get? Pass 3 as the parameter.

What results do you get?

PHP Regular Expressions Tutorial: Preg_match, Preg_split, Preg_replace

What is a Regular Expressions?

Regular expressions are powerful pattern matching algorithm that can be performed in a single expression.

Regular expressions use arithmetic operators such as (+,-,^) to create complex expressions.

Regular expressions help you accomplish tasks such as validating email addresses, IP address etc.

Why to use regular expressions

- Regular expressions simplify identifying patterns in string data by calling a single function. This saves us coding time.
- When validating user input such as email address, domain names, telephone numbers, IP addresses,
- Highlighting keywords in search results
- When creating a custom HTML template. Regular expressions can be used to identify the template tags and replace them with actual data.

Regular expressions in PHP

PHP has built in functions that allow us to work with regular functions. Let’s now look at the commonly used regular expression functions in PHP.

- preg_match – this function is used to perform a pattern match on a string. It returns true if a match is found and false if a match is not found.
- preg_split – this function is used to perform a pattern match on a string and then split the results into a numeric array
- preg_replace – this function is used to perform a pattern match on a string and then replace the match with the specified text.

Below is the syntax for a regular expression function such as preg_match,preg_split or preg_replace.

```
<?php
function_name('/pattern/',subject);
?>
```

HERE,

- "function_name(...)" is either preg_match, preg_split or preg_replace.
- "/.../" The forward slashes denote the beginning and end of our regular expression
- "'/pattern/'" is the pattern that we need to matched
- "subject" is the text string to be matched against

Let’s now look at practical examples that implement the above regular expression functions in PHP.

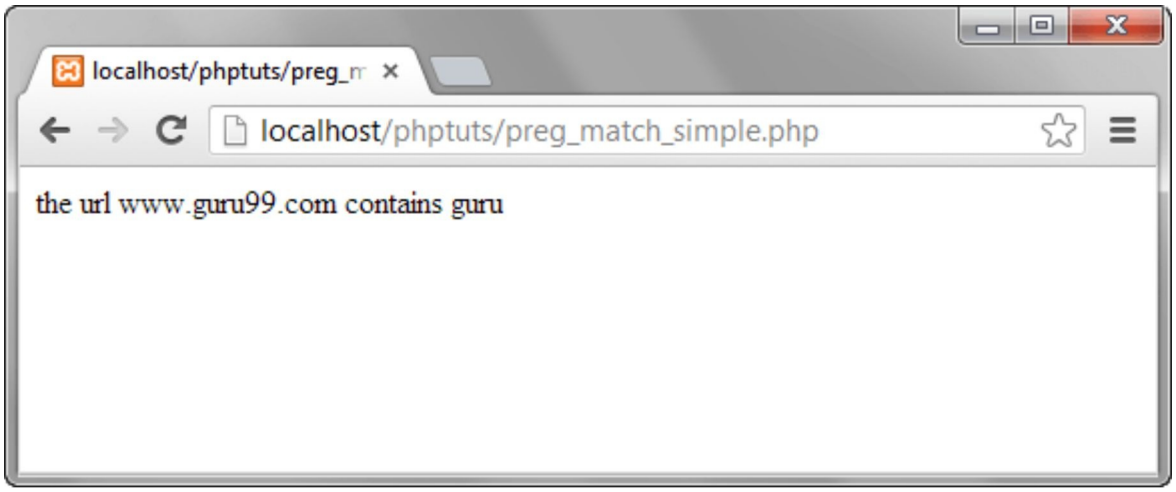
PHP Preg_match

The first example uses the preg_match function to perform a simple pattern match for the word guru in a given URL.

The code below shows the implementation for the above example.

```
<?php
$my_url = "www.guru99.com";
if (preg_match("/guru/", $my_url))
{
        echo "the url $my_url contains guru";
}
else
{
        echo "the url $my_url does not contain guru";
}
?>
```

Browse to the URL **http://localhost/phptuts/preg_match_simple.php**



Let's examine the part of the code responsible for our output "*preg_match('/guru/', $my_url)"* HERE,

- "preg_match(...)" is the PHP regular expression function
- "'/guru/'" is the regular expression pattern to be matched
- "$my_url" is the variable containing the text to be matched against.

The diagram below summarizes the above points

PHP Preg_split

Let's now look at another example that uses the preg_split function.

We will take a string phrase and explode it into an array; the pattern to be matched is a single space.

The text string to be used in this example is "I Love Regular Expressions".

The code below illustrates the implementation of the above example.

```
<?php

$my_text="I Love Regular Expressions";

$my_array  = preg_split("/ /", $my_text);

print_r($my_array );

?>
```

Browse to the URL **http://localhost/phptuts/preg_split.php**

PHP Preg_replace

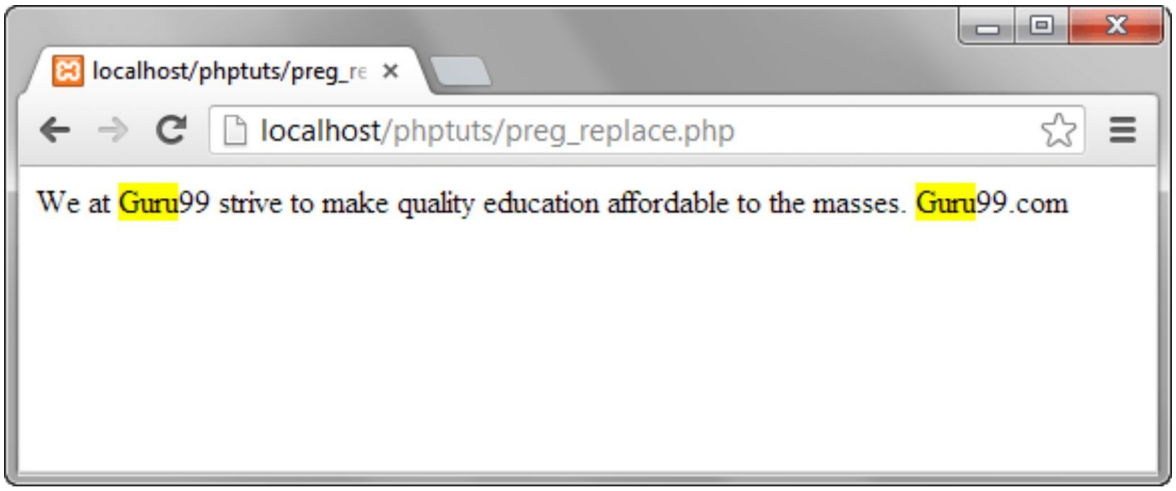
Let’s now look at the preg_replace function that performs a pattern match and then replaces the pattern with something else.

The code below searches for the word guru in a string.

It replaces the word guru with the word guru surrounded by css code that highlights the background colour.

```
<?php

$text = "We at Guru99 strive to make quality education affordable to the masses. Guru99.com";

$text = preg_replace("/Guru/", '<span style="background:yellow">Guru</span>', $text);

echo $text;

?>
```

Assuming you have saved the file preg_replace.php, browser to the URL **http://localhost/phptuts/preg_replace.php**



Meta characters

The above examples used very basic patterns; metacharacters simply allow us to perform more complex pattern matches such as test the validity of an email address. Let's now look at the commonly used metacharacters.

Metacharacter	Description	Example
.	Matches any single character except a new line	/./ matches anything that has a single character
^	Matches the beginning of or string / excludes characters	/^PH/ matches any string that starts with PH
$	Matches pattern at the end of the string	/com$/ matches guru99.com,yahoo.com Etc.
*	Matches any zero (0) or more characters	/com*/ matches computer, communication etc.
+	Requires preceding character(s) appear at least once	/yah+oo/ matches yahoo
\	Used to escape meta characters	/yahoo+\.com/ treats the dot as a literal value
[...]	Character class	/[abc]/ matches abc
a-z	Matches lower case letters	/a-z/ matches cool, happy etc.
A-Z	Matches upper case letters	/A-Z/ matches WHAT, HOW, WHY etc.
0-9	Matches any number between 0 and 9	/0-4/ matches 0,1,2,3,4

The above list only gives the most commonly used metacharacters in regular

expressions.

Let’s now look at a fairly complex example that checks the validity of an email address.

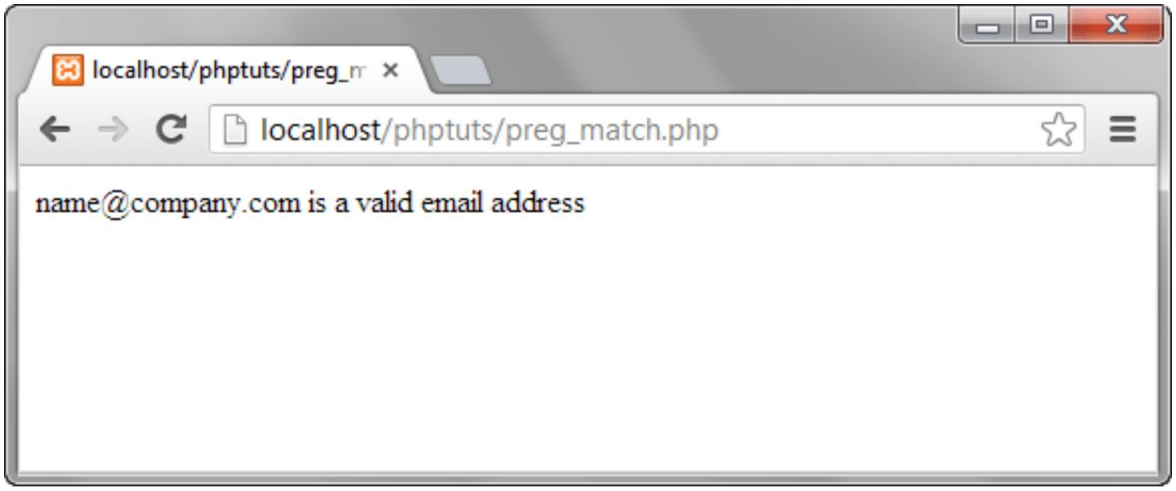
```
<?php
$my_email = "name@company.com";
if (preg_match("/^[a-zA-Z0-9._-]+@[a-zA-Z0-9-]+\.[a-zA-Z.]{2,5}$/", $my_email)) {
echo "$my_email is a valid email address";
}
else
{
  echo "$my_email is NOT a valid email address";
}
?>
```

Explaining the pattern "[/^[a-zA-Z0-9._-]+@[a-zA-Z0-9-]+\.[a-zA-Z.]{2,5}$/]"

HERE,

- "'/.../'" starts and ends the regular expression
- "^[a-zA-Z0-9._-]" matches any lower or upper case letters, numbers between 0 and 9 and dots, underscores or dashes.
- "+@[a-zA-Z0-9-]" matches the @ symbol followed by lower or upper case letters, numbers between 0 and 9 or dashes.
- "+\.[a-zA-Z.]{2,5}$/" escapes the dot using the backslash then matches any lower or upper case letters with a character length between 2 and 5 at the end of the string.

Browse to the URL **http://localhost/phptuts/preg_match.php**



As you can see from the above example breakdown, metacharacters are very powerful when it comes to matching patterns.

How to Send Email using PHP mail() Function

What is PHP mail?

PHP mail is the built in PHP function that is used to send emails from PHP scripts.

The mail function accepts the following parameters;

- Email address
- Subject
- Message
- CC or BC email addresses
    - It’s a cost effective way of notifying users on important events.
    - Let users contact you via email by providing a contact us form on the website that emails the provided content.
    - Developers can use it to receive system errors by email
    - You can use it to email your newsletter subscribers.
    - You can use it to send password reset links to users who forget their passwords
    - You can use it to email activation/confirmation links. This is useful when registering users and verifying their email addresses

Why/When to use the mail PHP

Sending mail using PHP

The PHP mail function has the following basic syntax

```
<?php
mail($to_email_address,$subject,$message,[$headers],[$parameters]);
?>
```

HERE,

- “$to_email_address” is the email address of the mail recipient
- “$subject” is the email subject
- “$message” is the message to be sent.
- “[$headers]” is optional, it can be used to include information such as CC, BCC
    - CC is the acronym for carbon copy. It’s used when you want to send a copy to an interested person i.e. a complaint email sent to a company can also be sent as CC to the complaints board.
    - BCC is the acronym for blind carbon copy. It is similar to CC. The email addresses included in the BCC section will not be shown to the other recipients.

Simple Transmission Protocol (SMTP)

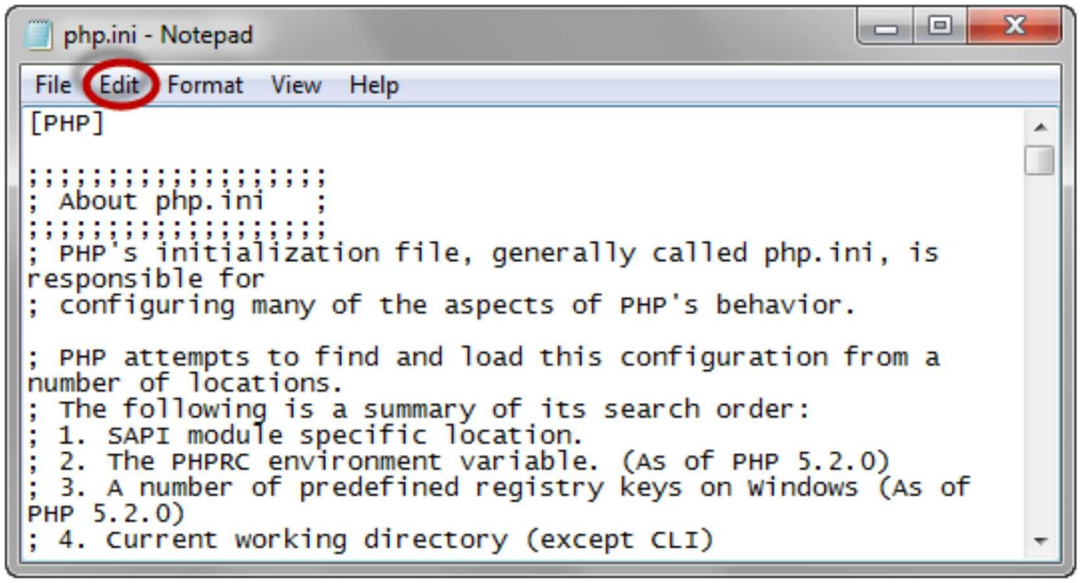
PHP mailer uses Simple Mail Transmission Protocol (SMTP) to send mail.

On a hosted server, the SMTP settings would have already been set.

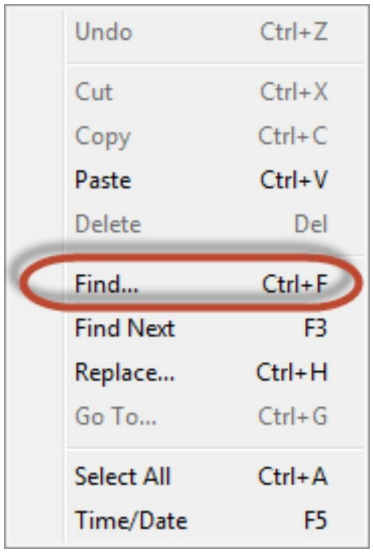
The SMTP mail settings can be configured from “php.ini” file in the PHP installation folder.

Configuring SMTP settings on your localhost Assuming you are using xampp on windows, locate the “php.ini” in the directory “C:\xampp\php”.

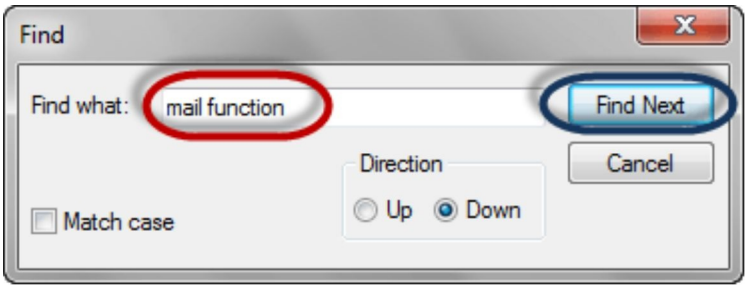
- Open it using notepad or any text editor. We will use notepad in this example. Click on the edit menu



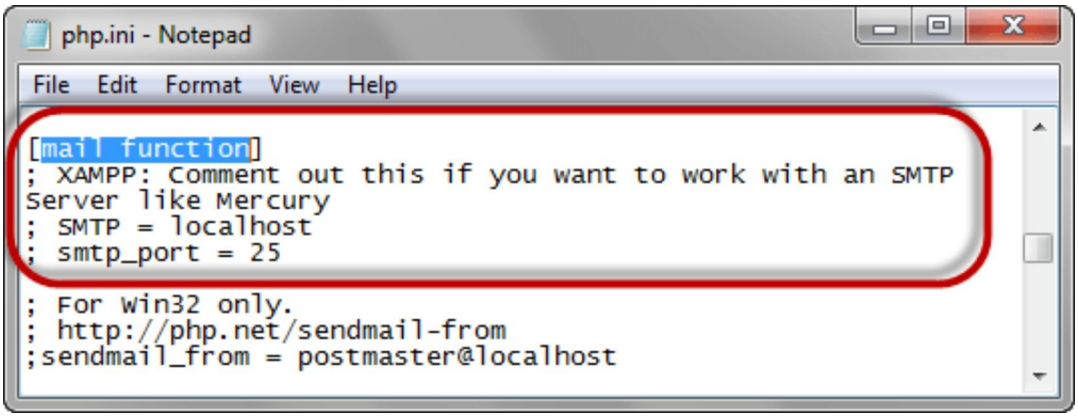
- Click on Find… menu



- The find dialog menu will appear



- Click on Find Next button



- Locate the entries
  - *[mail function]*
  - *; XAMPP:* Don't remove the semi column if you want to work with an SMTP Server like Mercury
  - ; SMTP = localhost
  - ; smtp_port = 25
  - Remove the semi colons before SMTP and smtp_port and set the SMTP to your smtp server and the port to your smtp port. Your settings should look as follows
    - SMTP = smtp.example.com
    - smtp_port = 25
    - *Note the SMTP settings can be gotten from your web hosting providers.*
    - If the server requires authentication, then add the following lines.
      - auth_username = example_username@example.com
      - auth_password = example_password
      - Save the new changes.

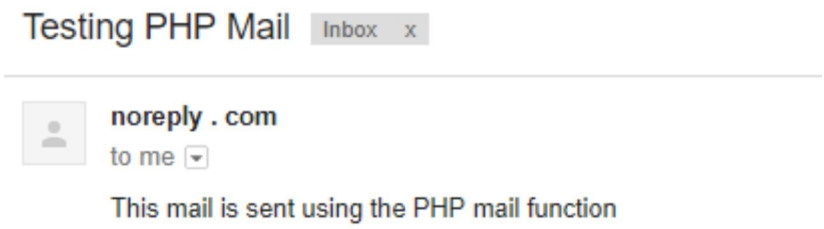
- Restart Apache server.

Php Mail Example

Let’s now look at an example that sends a simple mail.

```
<?php
$to_email = 'name @ company . com';
$subject = 'Testing PHP Mail';
$message = 'This mail is sent using the PHP mail function';
$headers = 'From: noreply @ company . com';
mail($to_email,$subject,$message,$headers);
?>
```

Output:



Note: the above example only takes the 4 mandatory parameters.

You should replace the above fictitious email address with a real email address.

Sanitizing email user inputs

The above example uses hard coded values in the source code for the email address and other details for simplicity.

Let's assume you have to create a contact us form for users fill in the details and then submit.

- Users can accidently or intentional inject code in the headers which can result in sending spam mail
- To protect your system from such attacks, you can create a custom function that sanitizes and validates the values before the mail is sent.

Let's create a custom function that validates and sanitizes the email address using the filter_var built in function.

Filter_var function The filter_var function is used to sanitize and validate the user input data.

It has the following basic syntax.

```
<?php
filter_var($field, SANITIZATION TYPE);
?>
```

HERE,

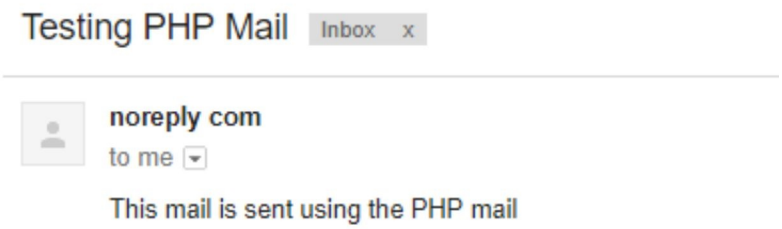
- "filter_var(…)" is the validation and sanitization function
- "$field" is the value of the field to be filtered.
- "SANITIZATION TYPE" is the type of sanitization to be performed on the field such as;
    - **FILTER_VALIDATE_EMAIL** – it returns true for valid email addresses and false for invalid email addresses.
    - **FILTER_SANITIZE_EMAIL** – it removes illegal characters from email addresses. info\@domain.(com) returns info@domain.com.
    - **FILTER_SANITIZE_URL** – it removes illegal characters from URLs. http://www.example@.comé returns >http://www.example@.com
    - **FILTER_SANITIZE_STRING** - it removes tags from string values. am bold becomes am bold.

The code below implements uses a custom function to send secure mail.

```
<?php
function sanitize_my_email($field) {
   $field = filter_var($field, FILTER_SANITIZE_EMAIL);
   if (filter_var($field, FILTER_VALIDATE_EMAIL)) {
```

```
      return true;
   } else {
      return false;
   }
}
$to_email = 'name @ company . com';
$subject = 'Testing PHP Mail';
$message = 'This mail is sent using the PHP mail ';
$headers = 'From: noreply @ company. com';
//check if the email address is invalid $secure_check
$secure_check = sanitize_my_email($to_email);
if ($secure_check == false) {
   echo "Invalid input";
} else { //send email
   mail($to_email, $subject, $message, $headers);
   echo "This email is sent using PHP Mail";
}
?>
```

Output:



Secure Mail

Emails can be intercepted during transmission by unintended recipients.

This can exposure the contents of the email to unintended recipients.

Secure mail solves this problem by transmitting emails via Hypertext Transfer Protocol Secure (HTTPS).

HTTPS encrypts messages before sending them.

PHP MySQLi Functions: mysqli_query, mysqli_connect, mysqli_fetch_array

PHP has a rich collection of built in functions for manipulating MySQL databases.

PHP mysqli_connect function

The PHP mysql connect function is used to connect to a MySQL database server.

It has the following syntax.

```
<?php;
$db_handle = mysqli_connect($db_server_name, $db_user_name, $db_password);
?>
```

HERE,

- “$db_handle” is the database connection resource variable.
- “mysqli_connect(…)” is the function for php database connection
- “$server_name” is the name or IP address of the server hosting MySQL server.
- “$user_name” is a valid user name in MySQL server.
- “$password” is a valid password associated with a user name in MySQL server.

PHP mysqli_select_db function

The mysqli_select_db function is used to select a database.

It has the following syntax.

```
<?php
```

```
mysqli_select_db($db_handle,$database_name);
?>
```

HERE,

- “mysqli_select_db(…)” is the database selection function that returns either true or false
- “$database_name” is the name of the database
- “$link_identifier” is optional, it is used to pass in the server connection link

PHP mysqli_query function

The mysqli_query function is used to execute SQL queries.

The function can be used to execute the following query types;

- Insert
- Select
- Update
- delete

It has the following syntax.

```
<?php
mysqli_query($db_handle,$query) ;
?>
```

HERE,

- “mysqli_query(…)” is the function that executes the SQL queries.
- “$query” is the SQL query to be executed
- “$link_identifier” is optional, it can be used to pass in the server connection link

PHP mysqli_num_rows function

The mysqli_num_rows function is used to get the number of rows returned from a select query.

It has the following syntax.

```
<?php
mysqli_num_rows($result);
?>
```

HERE,

- “mysqli_num_rows(…)” is the row count function
- “$result” is the mysqli_query result set

PHP mysqli_fetch_array function

The mysqli_fetch_array function is used fetch row arrays from a query result set.

It has the following syntax.

```
<?php
mysqli_fetch_array($result);
?>
```

HERE,

- “mysqli_fetch_array(…)” is the function for fetching row arrays
- “$result” is the result returned by the mysqli_query function.

PHP mysqli_close function

The mysqli_close function is used to close an open database connection.

It has the following syntax.

```
<?php
mysqli_close($db_handle);
?>
```

HERE,

- “mysqli_close(…)” is the PHP function
- “$link_identifier” is optional, it is used to pass in the server connection resource

Let’s look at practical examples that take advantage of these functions.

Creating the MySQL database This tutorial assumes knowledge of MySQL and SQL, if these terms are unfamiliar to you, refer to our MySQL and SQL tutorials.

We will create a simple database called my_personal_contacts with one table only.

Below are the steps to create the database and table.

- Connect to MySQL using your favorite access tool such as MySQL workbench, phpMyAdmin etc.
- Create a database named my_person_contacts
- Execute the script shown below to create the table and insert some dummy data

```
<?php
CREATE TABLE IF NOT EXISTS `my_contacts` (

 `id` int(11) NOT NULL AUTO_INCREMENT,
```

```
 `full_names` varchar(255) NOT NULL,

 `gender` varchar(6) NOT NULL,

 `contact_no` varchar(75) NOT NULL,

 `email` varchar(255) NOT NULL,

 `city` varchar(255) NOT NULL,

 `country` varchar(255) NOT NULL,

 PRIMARY KEY (`id`)

) ENGINE=InnoDB  DEFAULT CHARSET=latin1 AUTO_INCREMENT=5 ;

INSERT INTO `my_contacts` (`id`, `full_names`, `gender`, `contact_no`, `email`, `city`, `country`)
VALUES

(1, 'Zeus', 'Male', '111', 'zeus @ olympus . mt . co', 'Agos', 'Greece'),

(2, 'Anthena', 'Female', '123', 'anthena @ olympus . mt . co', 'Athens', 'Greece'),

(3, 'Jupiter', 'Male', '783', 'jupiter @ planet . pt . co', 'Rome', 'Italy'),

(4, 'Venus', 'Female', '987', 'venus @ planet . pt . co', 'Mars', 'Italy');
?>
```

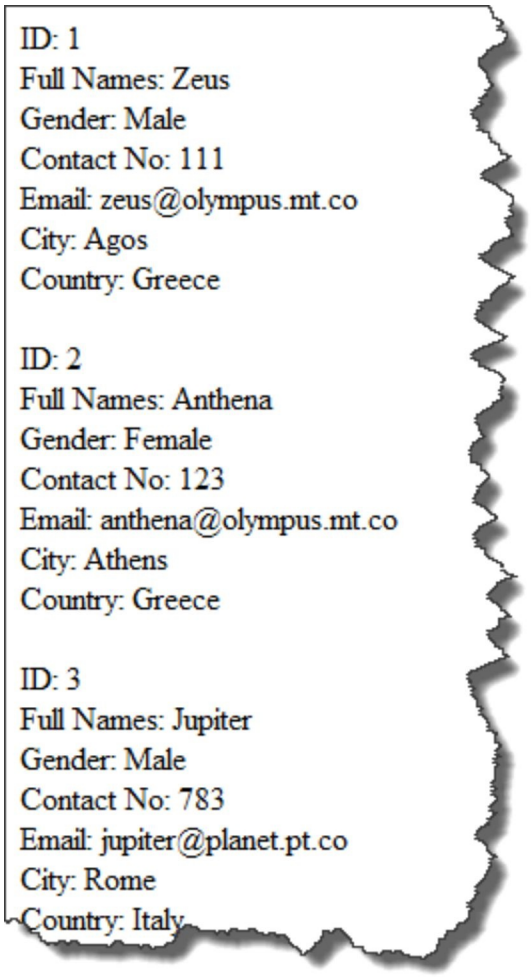
We now have a database set up that we will manipulate from PHP.

Reading records from the database We will now create a program that prints the records from the database.

```
<?php
        $dbh = mysqli_connect('localhost', 'root', 'melody');
        //connect to MySQL server if (!$dbh)

        die("Unable to connect to MySQL: " . mysqli_error());

        //if connection failed output error message
  if (!mysqli_select_db($dbh,'my_personal_contacts'))

        die("Unable to select database: " . mysqli_error());
  //if selection fails output error message

        $sql_stmt = "SELECT * FROM my_contacts";
  //SQL select query

   $result = mysqli_query($dbh,$sql_stmt);
   //execute SQL statement
```

```
        if (!$result)
                die("Database access failed: " . mysqli_error());
      //output error message if query execution failed

                $rows = mysqli_num_rows($result);
              // get number of rows returned

        if ($rows) {

        while ($row = mysqli_fetch_array($result)) {
                echo 'ID: ' . $row['id'] . '<br>';
                echo 'Full Names: ' . $row['full_names'] . '<br>';
                echo 'Gender: ' . $row['gender'] . '<br>';
                echo 'Contact No: ' . $row['contact_no'] . '<br>';
                echo 'Email: ' . $row['email'] . '<br>';
                echo 'City: ' . $row['city'] . '<br>';
                echo 'Country: ' . $row['country'] . '<br><br>';
        }
}
mysqli_close($dbh); //close the database connection
?>
```

Executing the above code returns the results shown in the diagram shown below



Inserting new records

Let's now look at an example that adds a new record into our table. the code below shows the implementation.

```
<?php
        $dbh = mysqli_connect('localhost', 'root', 'melody');
        //connect to MySQL server if (!$dbh)

        die("Unable to connect to MySQL: " . mysqli_error());
        //if connection failed output error message

        if (!mysqli_select_db($dbh,'my_personal_contacts'))
        die("Unable to select database: " . mysql_error());
        //if selection fails output error message

        $sql_stmt = "INSERT INTO `my_contacts`
(`full_names`,`gender`,`contact_no`,`email`,`city`,`country`)";
```

```
	$sql_stmt .= " VALUES('Poseidon','Mail','541',' poseidon @ sea . oc ','Troy','Ithaca')";

	$result = mysqli_query($dbh,$sql_stmt); //execute SQL statement

	if (!$result)
		die("Adding record failed: " . mysqli_error());
		//output error message if query execution failed echo "Poseidon has been successfully
added to your contacts list";

mysqli_close($dbh); //close the database connection
?>
```

Executing the above code outputs “Poseidon has been successfully added to your contacts list” go back to the select query example and retrieval your contacts again.

See if Poseidon has been added to your list.

Updating records Let’s now look at an example that updates a record in the database.

Let’s suppose that Poseidon has changed his contact number and email address.

```
<?php
	$dbh = mysqli_connect('localhost', 'root', 'melody');
	//connect to MySQL server

	if (!$dbh)
	die("Unable to connect to MySQL: " . mysqli_error());
  //if connection failed output error message

	if (!mysqli_select_db($dbh,'my_personal_contacts'))
	die("Unable to select database: " . mysql_error());
	//if selection fails output error message

	$sql_stmt = "UPDATE `my_contacts` SET `contact_no` = '785',`email` = ' poseidon @ ocean . oc
';
	//SQL select query $sql_stmt .= " WHERE `id` = 5";

	$result = mysqli_query($dbh,$sql_stmt);
	//execute SQL statement if (!$result)

	die("Deleting record failed: " . mysqli_error());
	//output error message if query execution failed

	echo "ID number 5 has been successfully updated";
mysqli_close($dbh); //close the database connection
```

```
?>
```

Deleting records

Let’s now look at an example that removes records from the database. Let’s suppose that Venus has a restraining order against us, and we must remove her contacts info from our database.

```
<?php
        $dbh = mysqli_connect('localhost', 'root', 'melody');
        //connect to MySQL server

        if (!$dbh)
        die("Unable to connect to MySQL: " . mysqli_error());
        //if connection failed output error message

        if (!mysqli_select_db($dbh,'my_personal_contacts'))
        die("Unable to select database: " . mysqli_error());
        //if selection failes output error message $id = 4;
        //Venus's ID in the database

        $sql_stmt = "DELETE FROM `my_contacts` WHERE `id` = $id";
        //SQL Delete query

        $result = mysqli_query($dbh,$sql_stmt);
        //execute SQL statement

        if (!$result)
        die("Deleting record failed: " . mysqli_error());
        //output error message if query execution failed

        echo "ID number $id has been successfully deleted";
mysqli_close($dbh); //close the database connection
?>
```

PHP Data Access Object PDO

The PDO is a class that allows us to manipulate different database engines such as MySQL, PostGres, MS SQL Server etc.

The code below shows the database access method using the PDO object.

Note: the code below assumes knowledge of SQL language, arrays, exception handling and foreach loop.

```
<?php
```

```
            try {
            $pdo = new PDO("mysql:host=localhost;dbname=my_personal_contacts", 'root', 'melody');

            $pdo->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
            $pdo->exec('SET NAMES "utf8"');

            $sql_stmt = "SELECT * FROM `my_contacts`";
            $result = $pdo->query($sql_stmt);
            $result->setFetchMode(PDO::FETCH_ASSOC);
            $data = array();
            foreach ($result as $row) {
            $data[] = $row;
  }
print_r($data);
}
catch (PDOException $e) {
            echo $e->getMessage();
}
?>
```

HERE,

- “try{…catch…}” is the exception handling block
- “$pdo = new PDO("mysql…” creates an instance of the PDO object and passes the database drivers, server and database names, user id and password.
- “$pdo->setAtt…” sets the PDO error mode and exception mode attributes
- “$pdo->exec('SET NA…” sets the encoding format

ODBC ODBC is the acronym for Open Database Connectivity. It has the following basic syntax.

```
<?php $conn = odbc_connect($dsn, $user_name, $password); ?>
```

HERE,

- “odbc_connect” is the PHP built in function
- “$dsn” is the ODBC data source name.
- “$user_name” is optional, it is used for the ODBC user name
- “$password” is optional, it is used for the ODBC password

The example used assumes you;

- Are Using Windows OS
- You have created an ODBC link to the northwind Microsoft Access database named northwind

Below is the implementation code for ODBC data access

```
<?php
	$dbh = odbc_connect('northwind', '', '');
	if (!$dbh) {
	exit("Connection Failed: " . $dbh);
}
	$sql_stmt = "SELECT * FROM customers";
	$result = odbc_exec($dbh, $sql_stmt);
	if (!$result) {
	exit("Error access records");
}
	while (odbc_fetch_row($result)) {
	$company_name = odbc_result($result, "CompanyName");
	$contact_name = odbc_result($result, "ContactName");
	echo "<b>Company Name (Contact Person):</b> $company_name ($contact_name) <br>";
}
odbc_close($dbh);
?>
```

PHP Object Oriented Programming (OOPs) concept Tutorial with Example

What is OOPs?

Object Oriented is an approach to software development that models application around real world objects such as employees, cars, bank accounts, etc. A class defines the properties and methods of a real world object. An object is an occurrence of a class.

The three basic components of object orientation are;

- Object oriented analysis – functionality of the system
- Object oriented designing – architecture of the system
- Object oriented programming – implementation of the application

Object Oriented Programming Principles

The three major principles of OOP are;

- **Encapsulation** – this is concerned with hiding the implementation details and only exposing the methods. The main purpose of encapsulation is to;
    - Reduce software development complexity – by hiding the

implementation details and only exposing the operations, using a class becomes easy.
    - Protect the internal state of an object – access to the class variables is via methods such as get and set, this makes the class flexible and easy to maintain.
    - The internal implementation of the class can be changed without worrying about breaking the code that uses the class.
- **Inheritance** – this is concerned with the relationship between classes. The relationship takes the form of a parent and child. The child uses the methods defined in the parent class. The main purpose of inheritance is;
    - Re-usability– a number of children, can inherit from the same parent. This is very useful when we have to provide common functionality such as adding, updating and deleting data from the database.
- **Polymorphism** – this is concerned with having a single form but many different implementation ways. The main purpose of polymorphism is;
    - Simplify maintaining applications and making them more extendable.

OOPs Concepts in PHP

PHP is an object oriented scripting language; it supports all of the above principles. The above principles are achieved via;

- **Encapsulation** - via the use of “get” and “set” methods etc.
- **Inheritance** - via the use of extends keyword
- **Polymorphism** - via the use of implements keyword

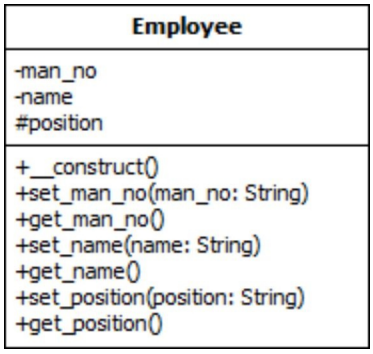
Now that we have the basic knowledge of OOP and how it is supported in PHP, let us look at examples that implement the above principles

What is UML?

Unified Modeling Language UML is a technique used to design and document object oriented systems.

UML produces a number of documents, but we will look at the class diagram which is very important to object oriented php programming.

Class Diagram Example



Class Diagram Key

- The **Upper box** contains the class name
- The **middle box** contains the class variables
- The **lower box** contains the class methods
- The **minus (-)** sign means private scope
- The **plus (+)** sign means public scope
- The **hash (#)** sign means protected scope

How to Create a class in PHP

The class keyword is used to define a class in PHP. Below are the rules for creating a class in PHP.

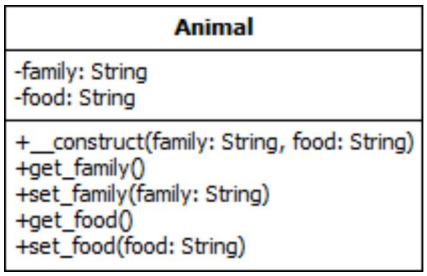
- The class name should start with a letter
- The class name cannot be a PHP reserved word
- The class name cannot contain spaces

Let’s say we want to create a class for representing animals.

We will start with identifying the features that are common to all animals.

- All animals belong to a family such as a herbivore, carnival, etc.
- All animals eat food

The diagram below shows the diagram for the animal



Let’s now code our animal class

```
<?php
```

```
class Animal
{
   private $family;
   private $food;
   public function __construct($family, $food)
   {
      $this->family = $family;
      $this->food   = $food;
   }
   public function get_family()
   {
      return $this->family;
   }
   public function set_family($family)
   {
      $this->family = $family;
   }
   public function get_food()
   {
      return $this->food;
   }
   public function set_food($food)
   {
      $this->food = $food;
   }
}
?>
```

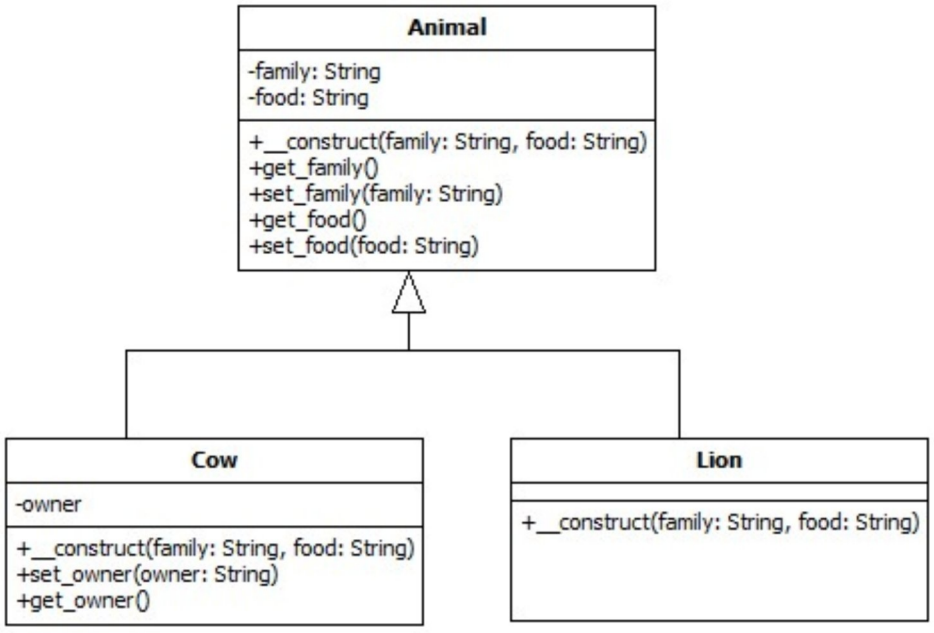
HERE,

- “private $family, $food” means the variables cannot be accessed directly outside the class (Encapsulation).
- “public function __construct($family…)” is the php constructor method. This function is called whenever an instance of the class has been created. In this case, we are setting the family and food.
- “public function get…()” is the method used to access the family or food value (Encapsulation)
- “public function set…()” is the method used to set the family or food value (Encapsulation)

How implement Inheritance in PHP

We will work with a cow and a lion. Both the cow and lion inherit from the Animal class.

The class diagram below shows the relationships.



Note the cow inherits from the animal class and defines its own variable and methods too.

Let’s now code the Cow class

```
<?php
class Cow extends Animal
{
   private $owner;
   public function __construct($family, $food)
   {
      parent::__construct($family, $food);
   }
   public function set_owner($owner)
   {
      $this->owner = $owner;
   }
   public function get_owner()
   {
      return $this->owner;
   }
}
?>
```

Let’s now code the Lion class

```
<?php
class Lion extends Animal
{
   public function __construct($family, $food)
   {
      parent::__construct($family, $food);
```

```
  }
}
?>
```

HERE,

- “class … extends Animal” makes the cow and lion use methods from the Animal class (Inheritance).

How to Create object of the class

The Animal, Cow, and Lion classes should all be in the same directory for simplicity’s sake.

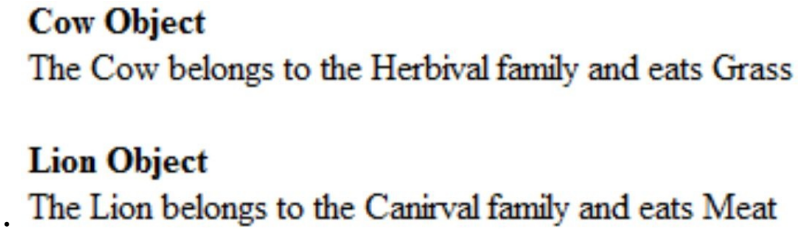
Let’s now create the application that uses our classes.

PHP Class Example

```
<?php
require 'Animal.php';
require 'Cow.php';
require 'Lion.php';
$cow  = new Cow('Herbivore', 'Grass');
$lion = new Lion('Canirval', 'Meat');
echo '<b>Cow Object</b> <br>';
echo 'The Cow belongs to the ' . $cow->get_family() . ' family and eats ' . $cow->get_food() . '<br><br>';
echo '<b>Lion Object</b> <br>';
echo 'The Lion belongs to the ' . $lion->get_family() . ' family and eats ' . $lion->get_food();
?>
```

Testing our application

Let’s now view our application in a web browser



Fantastic right! Let’s now look at the third principle of OOP, polymorphism.

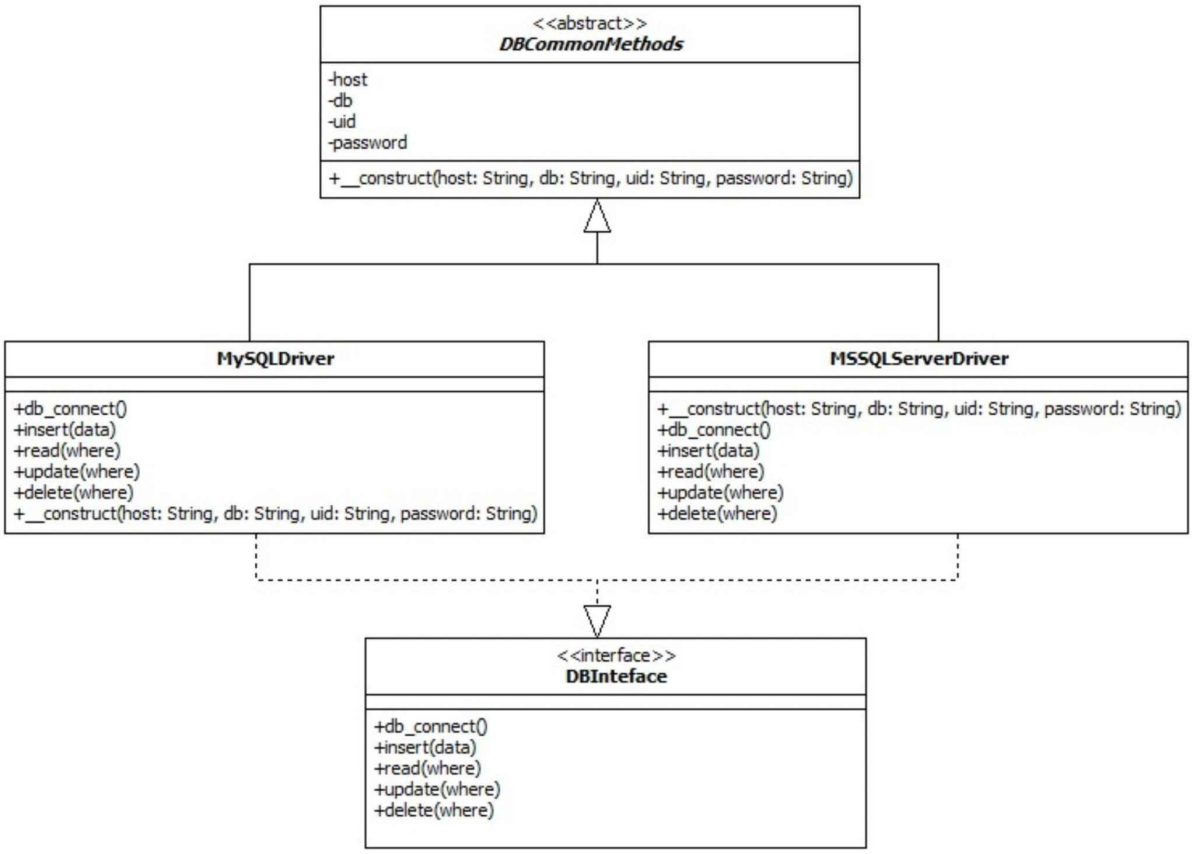
Let’s say we want to develop an application that connects to different database engines such as MySQL and SQL Server but use the same uniform interface.

We can create an interface that defines the standard methods and an abstract class that implements the common methods.

- **Interface** – it is similar to a class. It only defines the methods and parameters.

- **Abstract class** – it is a class that cannot be used to create an object directly. Its purpose is to provide partial or whole implementations of common methods.

The class diagram below illustrates the relationship among our abstract class, interface, and implementation classes.



Let's now create our abstract class

```
<?php
abstract class DBCommonMethods
{
   private $host;
   private $db;
   private $uid;
   private $password;
   public function __construct($host, $db, $uid, $password)
   {
      $this->host     = $host;
      $this->db       = $db;
      $this->uid      = $uid;
      $this->password = $password;
   }
}
```

```
?>
```

HERE,

- “abstract class” means the class cannot be used directly to php create object
- “$host,$db…” are class variables common to all implementations
- “function __construct(…)” is the php class constructor method that sets the common variables values at initialization

Let’s now create the interface that contains the standard methods which will be implemented differently depending on the database engine.

```
<?php
interface DBInterface
{
   public function db_connect();
   public function insert($data);
   public function read($where);
   public function update($where);
   public function delete($where);
}
?>
```

HERE,

- “interface” is the keyword for creating interfaces
- “public function…(…)” are the standard methods that should be implemented

Let’s now create the concrete classes that will extend the DBCommonMethods class and extend the DBInterface interface. MySQLDriver.php

```
<?php class MySQLDriver extends
DBCommonMethods implements DBInterface { public function __construct($host, $db, $uid, $password)
{
parent::__construct($host, $db, $uid, $password); }
public function db_connect() { //connect code goes here }
public function delete($where) { //delete code goes here }
public function insert($data) { //insert code goes here }
public function read($where) { //read code goes here }
public function update($where) { //update code goes here }
} ?>
```

MSSQLServerDriver.php

```
<?php
class MSSQLServerDriver extends
DBCommonMethods implements DBInterface { public function __construct($host, $db, $uid, $password)
{
parent::__construct($host, $db, $uid, $password); }
public function db_connect() { //connect code goes here }
public function delete($where) { //delete code goes here }
```

```
public function insert($data) { //insert code goes here }
public function read($where) { //read code goes here }
public function update($where) { //update code goes here }
} ?>
```

HERE,

- “class … extends DBCommonMethods” use the methods in the DBCommonMethods
- “… implements DBInterface” ensures that the class provides standard methods regardless of the database driver used.

Usage of the above code The code using the above class would look like this

```
<?php $db = new MySQLDriver($host,$db,$uid,$password); ?>
```

Or

```
<?php $db = new MSSQLServerDriver ($host,$db,$uid,$password); ?>
```

The rest of the code would be the same for both drivers such as;

```
<?php
$db->db_connect();
$db->insert($data);
?>
```

PHP Date & Time Function with Example

What is PHP Date Function?

PHP date function is an in-built function that simplify working with date data types. The PHP date function is used to format a date or time into a human readable format. It can be used to display the date of article was published. record the last updated a data in a database.

PHP Date Syntax & Example

PHP Date the following basic syntax

```
<?php
date(format,[timestamp]);
?>
```

HERE,

- “date(…)” is the function that returns the current time on the server.
- “format” is the general format which we want our output to be i.e.;
    - “Y-m-d” for PHP date format YYYY-MM-DD
    - “Y” to display the current year
    - “[timestamp]” is optional. If no timestamp has been provided, PHP will get the use the php current date time on the server.

Let’s look at a basic example that displays the current year.

```
<?php

echo date("Y");

?>
```

Output:

```
2018
```

What is a TimeStamp?

A timestamp is a numeric value in seconds between the current time and value as at 1st January, 1970 00:00:00 Greenwich Mean Time (GMT).

The value returned by the time function depends on the default time zone.

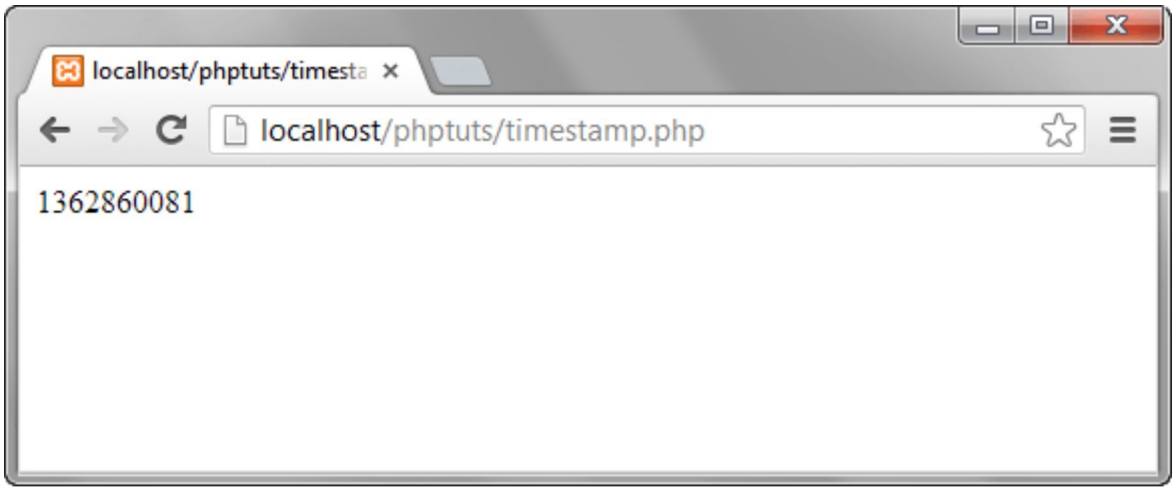
The default time zone is set in the php.ini file.

It can also be set programmatically using date_default_timezone_set function.

The code below displays the current time stamp

```
<?php

echo time();

?>
```

Assuming you saved the file timestamp.php in phptuts folder, browse to the URL **http://localhost/phptuts/timestamp.php**



Note: the value of the timestamp is not a constant. It changes every second.

Getting a list of available time zone identifiers

Before we look at how to set the default time zone programmatically, let's look at how to get a list of supported time zones.

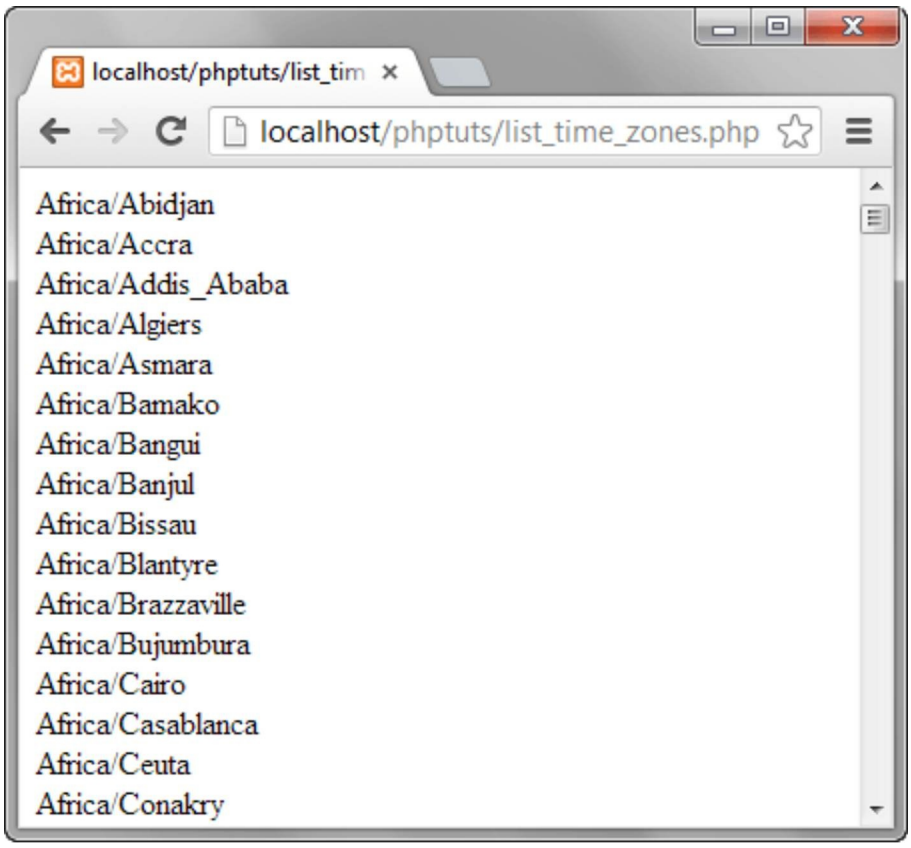
```
<?php

$timezone_identifiers = DateTimeZone::listIdentifiers();

foreach($timezone_identifiers as $key => $list){

echo $list . "<br/>";

}
?>
```

HERE,

- "$timezone_identifiers = DateTimeZone::listIdentifiers();" calls the listIdentifiers static method of the DateandTime Zone built in class.

  The listIdentifiers method returns a list of constants that are assigned to the variable $timezone_identifiers.
- "foreach{…}" iterates through the numeric array and prints the values.

Assuming you saved the file list_time_zones.php in phptuts folder, browse to the URL **http://localhost/phptuts/list_time_zones.php**



PHP set Timezone Programmatically

The date_default_timezone_set function allows you to set the default time zone from a PHP script.

The set time zone will then be used by all date php function scripts. It has the following syntax.

```
<?php
date_default_timezone_set ( string $timezone_identifier );
?>
```

HERE,

- “date_default_timezone_set()” is the function that sets the default time zone
- “string $timezone_identifier” is the time zone identifier

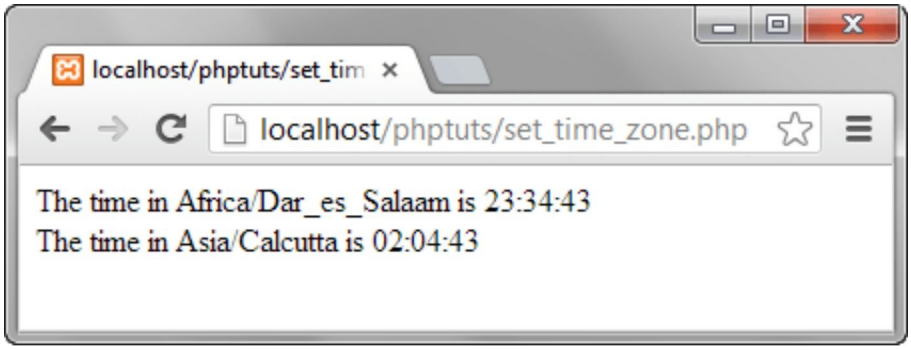
The script below displays the time according to the default time zone set in php.ini.

It then changes the default time zone to Asia/Calcutta and displays the time again.

```
<?php
echo "The time in " . date_default_timezone_get() . " is " . date("H:i:s");
```

```
date_default_timezone_set("Asia/Calcutta");
echo "The time in " . date_default_timezone_get() . " is " . date("H:i:s");
?>
```

Assuming you have saved the file set_time_zone.php in the phptuts folder, browse to the URL **http://localhost/phptuts/set_time_zone.php**



PHP Mktime Function

The mktime function returns the timestamp in a Unix format.

It has the following syntax.

```
<?php
mktime(hour, minute, second, month, day, year, is_dst);
?>
```

HERE,

- “mktime(…)” is the make php timestamp function
- “hour” is optional, it is the number of hour
- “minute” is optional, it is the number of minutes
- “second” is optional, it is the number of seconds
- “month” is optional, it is the number of the month
- “day” is optional, it is the number of the day
- “year” is optional, it is the number of the year
- “is_dst” is optional, it is used to determine the day saving time (DST). 1 is for DST, 0 if it is not and -1 if it is unknown.

Let’s now look at an example that creates a timestamp for the date 13/10/2025 using the mktime function.

```
<?php

echo mktime(0,0,0,10,13,2025);

?>
```

HERE,

- “0,0,0” is the hour, minute and seconds respectively.
- “13” is the day of the month
- “10” is the month of the year
- “2025” is the year

Output:

```
1760328000
```

PHP Date function reference

The table below shows the common parameters used when working with the date php function.

PHP Time parameters

Parameter	Description	Example
“r”	Returns the full date and time	<?php echo date("r"); ?>
“a”,”A”	Returns whether the current time is am or pm, AM or PM respectively	<?php echo date("a"); echo date("A"); ?>
“g”,”G”	Returns the hour without leading zeroes [1 to 12], [0 to 23] respectively	<?php echo date("g"); echo date("G"); ?>
“h”,”H”	Returns the hour with leading zeros [01 to 12],[00 to 23] respectively	<?php echo date("h"); echo date("H"); ?>
“i”,”s”	Returns the minutes/seconds with leading zeroes [00 to 59]	<?php echo date("i"); echo date("s"); ?>

Day parameters

Parameter	Description	Example
“d”	Returns the day of the month with leading zeroes [01 to 31]	<?php echo date("d"); ?>
“j”	Returns the day of the month without leading zeroes [1 to 31]	<?php echo date("j"); ?>
“D”	Returns the first 3 letters of the day name [Sub to Sat]	<?php echo date("D"); ?>
“l”	Returns day name of the week [Sunday to Saturday]	<?php echo date("l"); ?>
“w”	Returns day of the week without leading zeroes [0 to 6] Sunday is represent by zero (0) through to Saturday represented by six (6)	<?php echo date("w"); ?>
“z”	Returns the day of the year without leading spaces [0 through to 365]	<?php echo date("z"); ?>

Month Parameters

Parameter	Description	Example
“m”	Returns the month number with leading zeroes [01 to 12]	<?php echo date("m"); ?>
“n”	Returns the month number without leading zeroes [01 to 12]	<?php echo date("n"); ?>
“M”	Returns the first 3 letters of the month name [Jan to Dec]	<?php echo date("M"); ?>
“F”	Returns the month name [January to December]	<?php echo date("F"); ?>
“t”	Returns the number of days in a month [28 to 31]	<?php echo date("t"); ?>

Year Parameters

Parameter	Description	Example
“L”	Returns 1 if it’s a leap year and 0 if it is not a leap year	<?php echo date("L"); ?>
“Y”	Returns four digit year format	<?php echo date("Y"); ?>
“y”	Returns two (2) digits year format (00 to 99)	<?php echo date("y"); ?>

PHP Security Function: strip_tags, filter_var, Md5 and sha1

Potential security threats

They are basically two groups of people that can attack your system

- Hackers – with the intent to gain access to unauthorized data or disrupt the application
- Users – they may innocently enter wrong parameters in forms which can have negative effects on a website or web application.

The following are the kinds of attacks that we need to look out for.

SQL Injection – This type of attack appends harmful code to SQL statements.

This is done using either user input forms or URLs that use variables.

The appended code comments the condition in the WHERE clause of an SQL statement. The appended code can also;

- insert a condition that will always be true
- delete data from a table
- update data in a table
- This type of attack is usually used to gain unauthorized access to an application.

Cross-site scripting – this type of attack inserts harmful code usually JavaScript. This is done using user input forms such as contact us and comments forms. This is done to;

- Retrieve sensitive information such as cookies data
- Redirect the user to a different URL.
- Other threats can include – PHP code injection, Shell Injection, Email Injection, Script Source Code Disclosure etc.

PHP Application Security Best Practices

Let's now look at some of the PHP Security best practices that we must consider when developing our applications.

PHP strip_tags

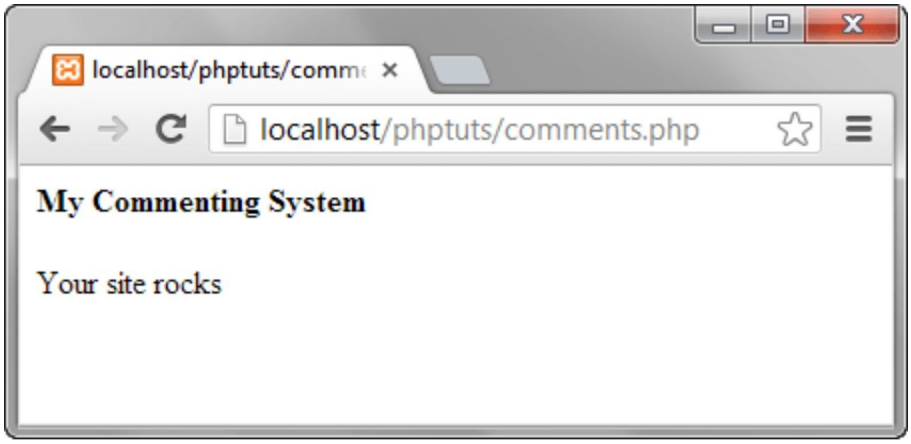
The strip_tags functions removes HTML, JavaScript or PHP tags from a string.

This function is useful when we have to protect our application against attacks such as cross site scripting.

Let's consider an application that accepts comments from users.

```
<?php

$user_input = "Your site rocks";

echo "<h4>My Commenting System</h4>";

echo $user_input;

?>
```

Assuming you have saved comments.php in the phptuts folder, browse to the URL**http://localhost/phptuts/comments.php**

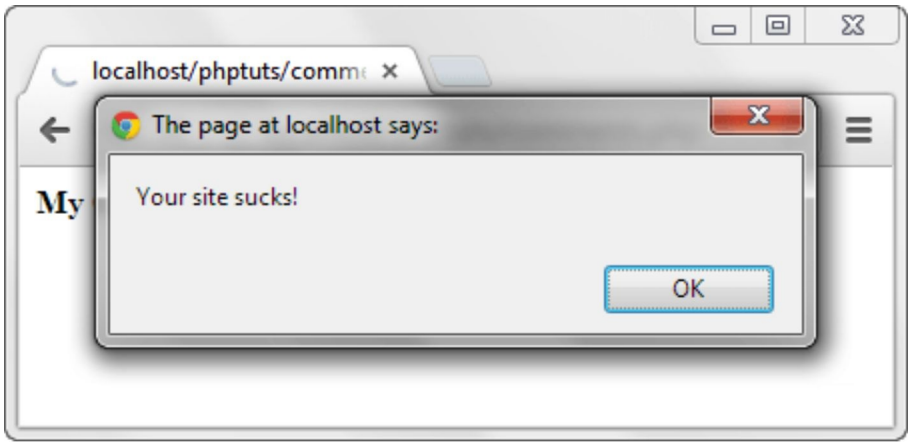


Let's assume you receive the following as the user input <script>alert('Your site

sucks!');</script>

```
<?php

$user_input = "<script>alert('Your site sucks!');</script>";

echo "<h4>My Commenting System</h4>";

echo $user_input;

?>
```

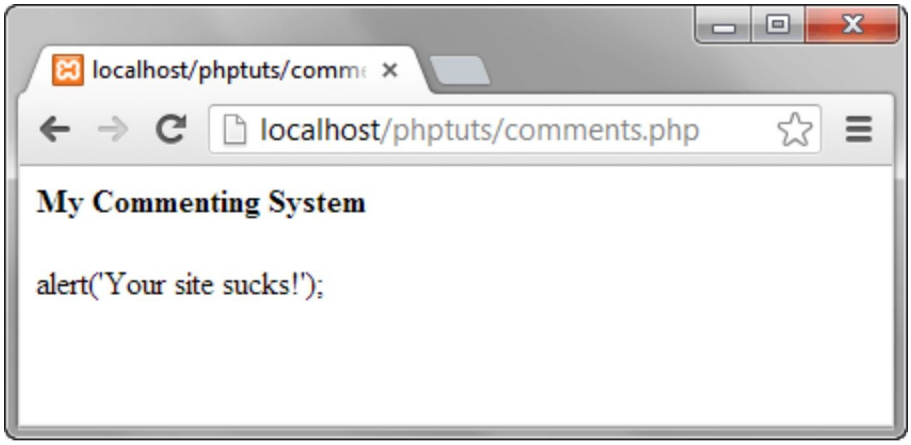
Browse to the URL **http://localhost/phptuts/comments.php**



Let’s now secure our application from such attacks using strip_tags function.

```
<?php

$user_input = "<script>alert('Your site sucks!');</script>";

echo strip_tags($user_input);

?>
```

Browse to the URL **http://localhost/phptuts/comments.php**



PHP filter_var function

The filter_var function is used to validate and sanitize data.

Validation checks if the data is of the right type. A numeric validation check on a string returns a false result.

Sanitization is removing illegal characters from a string.

Check this link for the complete reference filter_var

The code is for the commenting system.

It uses the filter_var function and FILTER_SANITIZE_STRIPPED constant to strip tags.

```
<?php

$user_input = "<script>alert('Your site sucks!');</script>";

echo filter_var($user_input, FILTER_SANITIZE_STRIPPED);

?>
```

Output:

```
alert('Your site sucks!');
```

Mysql_real_escape_string function This function is used to protect an application against SQL injection.

Let’s suppose that we have the following SQL statement for validating the user id and password.

```
<?php
SELECT uid,pwd,role FROM users WHERE uid = 'admin' AND password = 'pass';
?>
```

A malicious user can enter the following code in the user id text box. ' OR 1 = 1 -- And 1234 in the password text box Let’s code the authentication module

```
<?php

$uid = "' OR 1 = 1 -- ";

$pwd = "1234";

$sql = "SELECT uid,pwd,role FROM users WHERE uid = '$uid' AND password = '$pwd';";

echo $sql;

?>
```

The end result will be

```
SELECT uid,pwd,role FROM users WHERE uid = '' OR 1 = 1 -- ' AND password = '1234';
```

HERE,

- “SELECT * FROM users WHERE user_id = ‘”” tests for an empty user id

- “' OR 1 = 1 “ is a condition that will always be true
- “--" comments that part that tests for the password.

The above query will return all the users Let’s now use mysql_real_escape_string function to secure our login module.

```
<?php

$uid = mysql_real_escape_string("' OR 1 = 1 -- ");

$pwd = mysql_real_escape_string("1234");

$sql = "SELECT uid,pwd,role FROM users WHERE uid = '$uid' AND password = '$pwd';";

echo $sql;

?>
```

The above code will output

```
SELECT uid,pwd,role FROM users WHERE uid = '\' OR 1 = 1 -- ' AND password = '1234';
```

Note the second single quote has been escaped for us, it will be treated as part of the user id and the password won’t be commented.

PHP Md5 and PHP sha1

Md5 is the acronym for Message Digest 5 and sha1 is the acronym for Secure Hash Algorithm 1.

They are both used to encrypt strings.

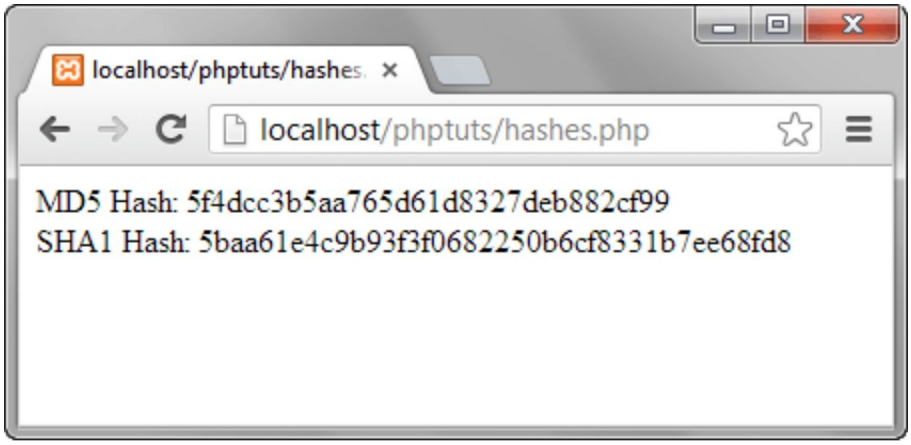
Once a string has been encrypted, it is tedious to decrypt it.

Md5 and sha1 are very useful when storing passwords in the database.

The code below shows the implementation of md5 and sha1

```
<?php
echo "MD5 Hash: " . md5("password");
echo "SHA1 Hash: " . sha1("password");
?>
```

Assuming you have saved the file hashes.php in phptuts folder, browse to the URL



As you can see from the above hashes, if an attacker gained access to your database, they still wouldn't know the passwords for them to login.

PHP XML Tutorial: Create, Parse, Read with Example

What is XML?

XML is the acronym for Extensible Markup Language.

XML is used to structure, store and transport data from one system to another.

XML is similar to HTML.

It uses opening and closing tags.

Unlike HTML, XML allows users to define their own tags.

What is DOM?

DOM is the acronym for Document Object Model.

It's a cross platform and language neutral standard that defines how to access and manipulate data in;

- HTML
- XHTML
- XML

DOM XML is used to access and manipulate XML documents. It views the XML document as a tree-structure.

XML Parsers

An XML parser is a program that translates the XML document into an XML

Document Object Model (DOM) Object.

The XML DOM Object can then be manipulated using JavaScript, Python, and PHP etc.

The keyword CDATA which is the acronym for (Unparsed) Character Data is used to ignore special characters such as “<,>” when parsing an XML document.

Why use XML?

- Web services such as SOAP and REST use XML format to exchange information. Learning what XML is and how it works will get you competitive advantage as a developer since modern applications make heavy use of web services.
- XML documents can be used to store configuration settings of an application
- It allows you to create your own custom tags which make it more flexible.

XML Document example

Let’s suppose that you are developing an application that gets data from a web service in XML format.

Below is the sample of how the XML document looks like.

```
<?xml version="1.0" encoding="utf-8"?>

<employees status = "ok">

      <record man_no = "101">

         <name>Joe Paul</name>

         <position>CEO</position>

      </record>

      <record man_no = "102">

         <name>Tasha Smith</name>

         <position>Finance Manager</position>

      </record>

</employees>
```

HERE,

- “<?xml version="1.0" encoding="utf-8"?>” specifies the xml version to be

used and encoding

- “<employees status = "ok">” is the root element.
- “<record…>…</record>” are the child elements of administration and sales respectively.

How to Read XML using PHP

Let’s now write the code that will read the employees XML document and display the results in a web browser. *Index.php*

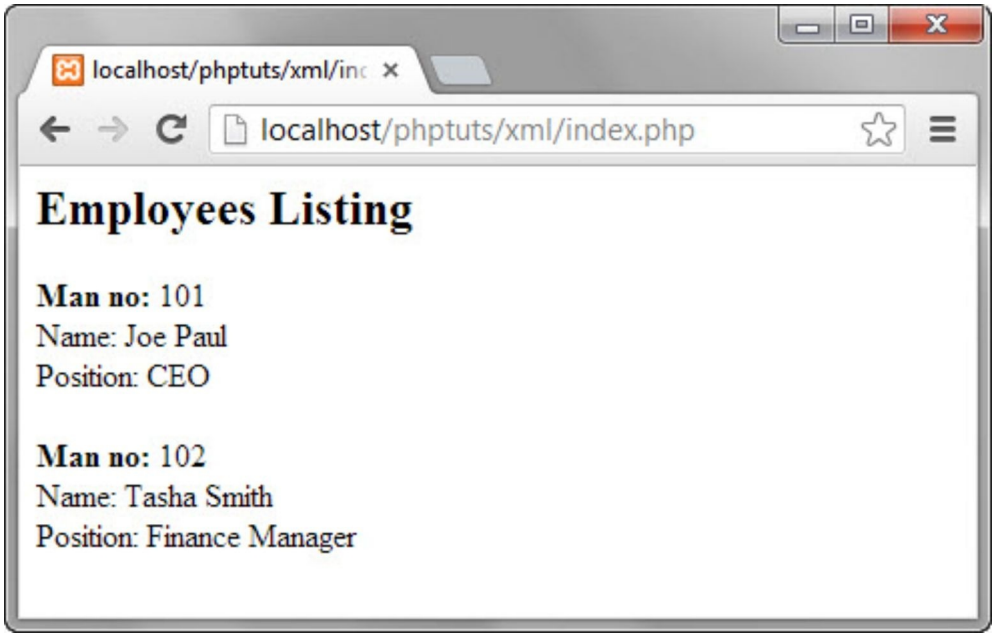
```
<?php

$xml = simplexml_load_file('employees.xml');

echo '<h2>Employees Listing</h2>';

$list = $xml->record;

for ($i = 0; $i < count($list); $i++) {

   echo '<b>Man no:</b> ' . $list[$i]->attributes()->man_no . '<br>';

   echo 'Name: ' . $list[$i]->name . '<br>';

   echo 'Position: ' . $list[$i]->position . '<br><br>';

}
?>
```

HERE,

- “$xml = simplexml_load_file('employees.xml');” uses the simplexml_load_file function to load the file name employees.xml and assign the contents to the array variable $xml.
- “$list = $xml->record;” gets the contents of the record node.
- “for ($i = 0; $i < count(…)…” is the for loop that reads the numeric array and outputs the results
- “$list[$i]->attributes()->man_no;” reads the man_no attribute of the element
- “$list[$i]->name;” reads the value of the name child element
- “$list[$i]->position;” reads the value of the position child element

Testing our application

Assuming you saved the file index.php in phptus/xml folder, browse to the URL**http://localhost/phptuts/xml/index.php**



How to Create an XML document using PHP

We will now look at how to create an XML document using PHP.

We will use the example above in the DOM tree diagram.

The following code uses the PHP built in class DOMDocument to create an XML document.

```
<?php

	$dom = new DOMDocument();

		$dom->encoding = 'utf-8';

		$dom->xmlVersion = '1.0';

		$dom->formatOutput = true;

	$xml_file_name = 'movies_list.xml';

		$root = $dom->createElement('Movies');

		$movie_node = $dom->createElement('movie');

		$attr_movie_id = new DOMAttr('movie_id', '5467');

		$movie_node->setAttributeNode($attr_movie_id);

	$child_node_title = $dom->createElement('Title', 'The Campaign');
```

```
                $movie_node->appendChild($child_node_title);

                $child_node_year = $dom->createElement('Year', 2012);

                $movie_node->appendChild($child_node_year);

        $child_node_genre = $dom->createElement('Genre', 'The Campaign');

                $movie_node->appendChild($child_node_genre);

                $child_node_ratings = $dom->createElement('Ratings', 6.2);

                $movie_node->appendChild($child_node_ratings);

                $root->appendChild($movie_node);

                $dom->appendChild($root);

        $dom->save($xml_file_name);

        echo "$xml_file_name has been successfully created";
?>
```

HERE,

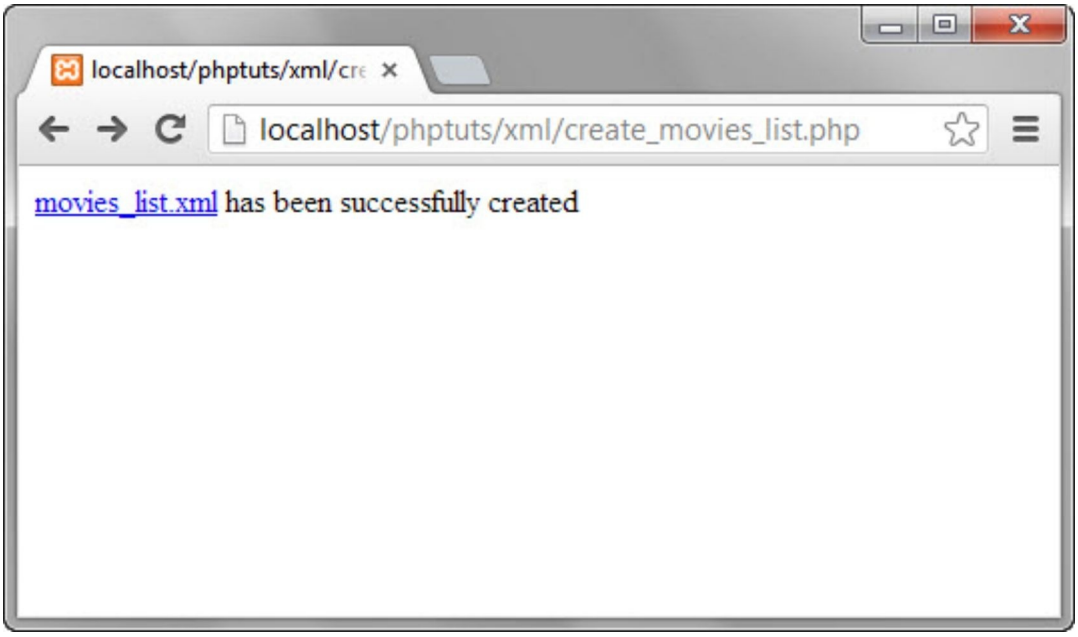
- “$dom = new DOMDocument();” creates an instance of DOMDocument class.
- “$dom->encoding = 'utf-8';” sets the document encoding to utf-8
- “$dom->xmlVersion = '1.0';” specifies the version number 1.0
- “$dom->formatOutput = true;” ensures that the output is well formatted
- “$root = $dom->createElement('Movies');” creates the root node named Movies
- “$attr_movie_id = new DOMAttr('movie_id', '5467');” defines the movie id attribute of Movies node
- “$child_node_element_name = $dom->createElement('ElementName', 'ElementValue')” creates the child node of Movies node. ElementName specifies the name of the element e.g. Title. ElementValue sets the child node value e.g. The Campaign.
- “$root->appendChild($movie_node);” appends the movie_node elements to the root node Movies
- “$dom->appendChild($root);” appends the root node to the XML

document.

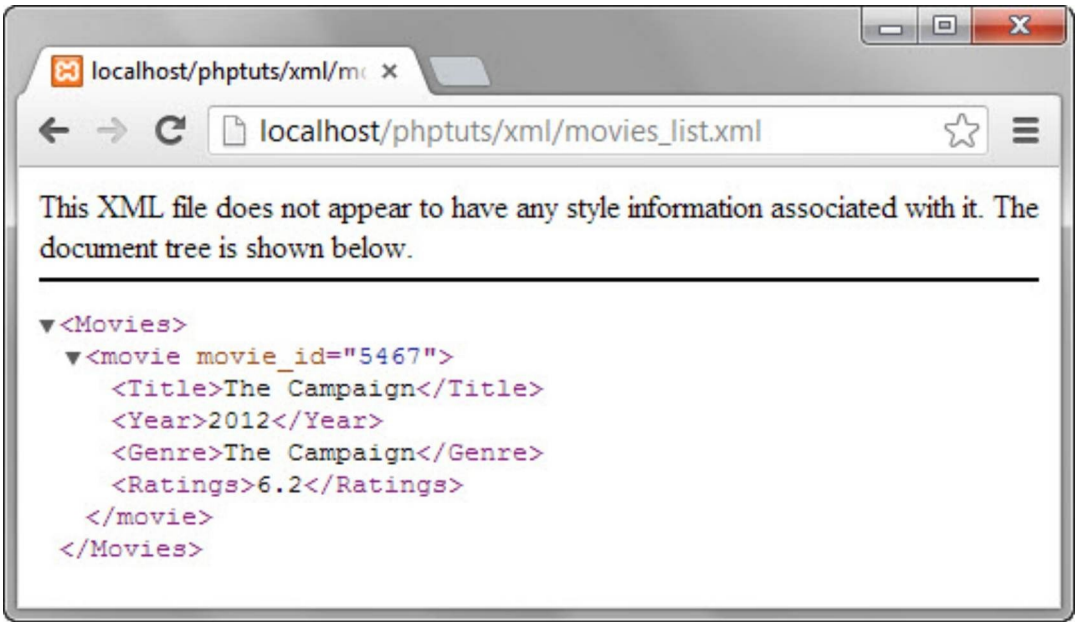
- “$dom->save($xml_file_name);” saves the XML file in the root directory of the web server.
- “echo '' . $xml_file_name . ' has been successfully created';” creates the link to the XML file.

Testing our application

Assuming you saved the file create_movies_list in phptuts/xml folder, browse to the URL **http://localhost/phptuts/xml/create_movies_list.php**



Click on movies_list_xml link



PHP Projects: Create an Opinion Poll Application

In this PHP project**,** we are going to create an opinion poll application.

The opinion poll will consist of 3 major components;

Front controller – this is the index page that will determine the HTML code to be loaded. This will ensure that our application has a single entry point. This will give us more control over the application.

Business Logic – this will contain the PHP code for interacting with the database. This will allow us to separate the business logic from the presentation making our application easy to maintain

Views – this will contain the HTML code. We will have two pages namely;

- opinion.html.php – this will contain the HTML code with the question and options
- results.html.php – this will contain the HTML code that displays the opinion poll results

Assumptions made

The opinion poll will ask the question –

What is your favourite JavaScript Library?

Answers wold be

- JQuery

- MooTools
- YUI Library
- Glow

Here are the steps to create the application –

Step 1) Database Connectivity

This section assumes knowledge of MySQL and how to administer it, if you are not familiar with these MySQL,.

Our application will have one table only with 3 fields namely;

- id – auto generate number as the primary key
- choice – the number representing a presidential candidate
- ts – the timestamp for the vote

The script below creates our js_libraries table.

```
<?php
CREATE TABLE `js_libraries` (

 `id` int(11) NOT NULL AUTO_INCREMENT,

 `choice` tinyint(4) NOT NULL DEFAULT '0',

 `ts` timestamp NULL DEFAULT NULL,

 PRIMARY KEY (`id`)

);
?>
```

Step 2) Coding our application

Let’s now create our business logic layer that will handle the database connectivity. *'opinion_poll_model.php'*

```
<?php

class Opinion_poll_model {

 private $db_handle; private $host = 'localhost'; private $db = 'opinion_poll';private $uid = 'root'; private $pwd = 'melody';

   public function __construct() {

      $this->db_handle = mysqli_connect($this->host, $this->uid, $this->pwd); //connect to MySQL server

      if (!$this->db_handle) die("Unable to connect to MySQL: " . mysqli_error());
```

```
    if (!mysqli_select_db($this->db_handle,$this->db)) die("Unable to select database: " . mysqli_error());
  }
  private function execute_query($sql_stmt) {
    $result = mysqli_query($db_handle,$sql_stmt); //execute SQL statement
    return !$result ? FALSE : TRUE;
  }
  public function select($sql_stmt) {
    $result = mysqli_query($db_handle,$sql_stmt);
    if (!$result) die("Database access failed: " . mysqli_error());
    $rows = mysqli_num_rows($result);
    $data = array();
    if ($rows) {
      while ($row = mysqli_fetch_array($result)) {
        $data = $row;
      }
    }
    return $data;
  }
  public function insert($sql_stmt) {
    return $this->execute_query($sql_stmt);
  }
  public function __destruct(){
    mysqli_close($this->db_handle);
  }
}
?>
```

HERE,

- “public function __construct()” is the class constructor method that is used to establish the database connection
- “public function execute_query(…)” is the method for executing queries such as insert, update and delete
- “public function select” is the method for retrieving data from the database and returning a numeric array.
- “public function insert(…)” is the insert method that calls the execute_query method.
- “public function __destruct()” is the class destructor that closes the database connection.

Let’s now create the front controller *index.php*

```
<?php

require 'opinion_poll_model.php';

$model = new Opinion_poll_model();

if (count($_POST) == 1) {

    echo "<script>alert('You did not vote!');</script>";

}

if (count($_POST) > 1) {

    $ts = date("Y-m-d H:i:s");

    $option = $_POST['vote'][0];

    $sql_stmt = "INSERT INTO js_libraries (`choice`,`ts`) VALUES ($option,'$ts')";

    $model->insert($sql_stmt);

    $sql_stmt = "SELECT COUNT(choice) choices_count FROM js_libraries;";

    $choices_count = $model->select($sql_stmt);

    $libraries = array("", "JQuery", "MooTools", "YUI Library", "Glow");

    $table_rows = '';

    for ($i = 1; $i < 5; $i++) {

        $sql_stmt = "SELECT COUNT(choice) choices_count FROM js_libraries WHERE choice = $i;";
```

```
        $result = $model->select($sql_stmt);

        $table_rows .= "<tr><td>" . $ libraries [$i] . " Got:</td><td><b>" . $result[0] . "</b> votes</td>
</tr>";

    }

    require 'results.html.php';

    exit;

}

require 'opinion.html.php';

?>
```

HERE,

- “require 'opinion_poll_model.php';” loads the business logic class
- “$model = new Opinion_poll_model();” creates an instance of the business logic class
- “if (count($_POST) == 1)…” performs the data validation and uses JavaScript to display a message box if not candidate has been voted for.
- “if (count($_POST) > 1)…” checks if a vote has been selected by counting the number of items in the $_POST array. If no item has been select, the $_POST will only contain the submit item. If a candidate has been chosen, the $_POST array will two elements, the submit and vote item. This code is also used to insert a new vote record and then display the results page
- “exit;” is used to terminate the script execution after the results have been displayed so that the opinion poll form is not displayed.
- “require 'opinion.html.php';” displays the opinion poll form if nothing has been selected.

Let’s now create the views. *opinion.html.php*

```
<html>

    <head>

        <title>JavaScript Libraries - Opinion Poll</title>

    </head>

    <body>

        <h2>JavaScript Libraries - Opinion Poll</h2>
```

```
    <p><b>What is your favorite JavaScript?</b></p>
    <form method="POST" action="index.php">
      <p> <input type="radio" name="vote" value="1" />JQuery
        <br /><input type="radio" name="vote" value="2" />MooToolsl
        <br /><input type="radio" name="vote" value="3" />YUI Library
        <br /><input type="radio" name="vote" value="4" />Glow
      </p>
      <p><input type="submit" name="submitbutton" value="OK" /></p>
    </form>
  </body>
</html>
```

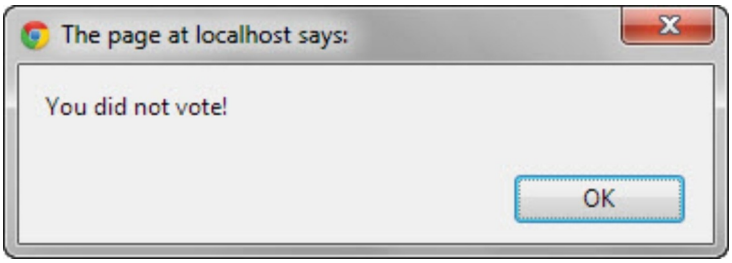
results.html.php

```
<html>
  <head>
    <title>JavaScript Libraries Poll Results</title>
  </head>
  <body>
    <h2>Opinion Poll Results</h2>
    <p><b>What is your favorite JavaScript Library?</b></p>
    <p><b><?php echo $choices_count[0]; ?></b> people have thus far taken part in this poll:</p>
    <p>
<table>
      <?php echo($table_rows); ?>
    </table>
</body>
</html>
```

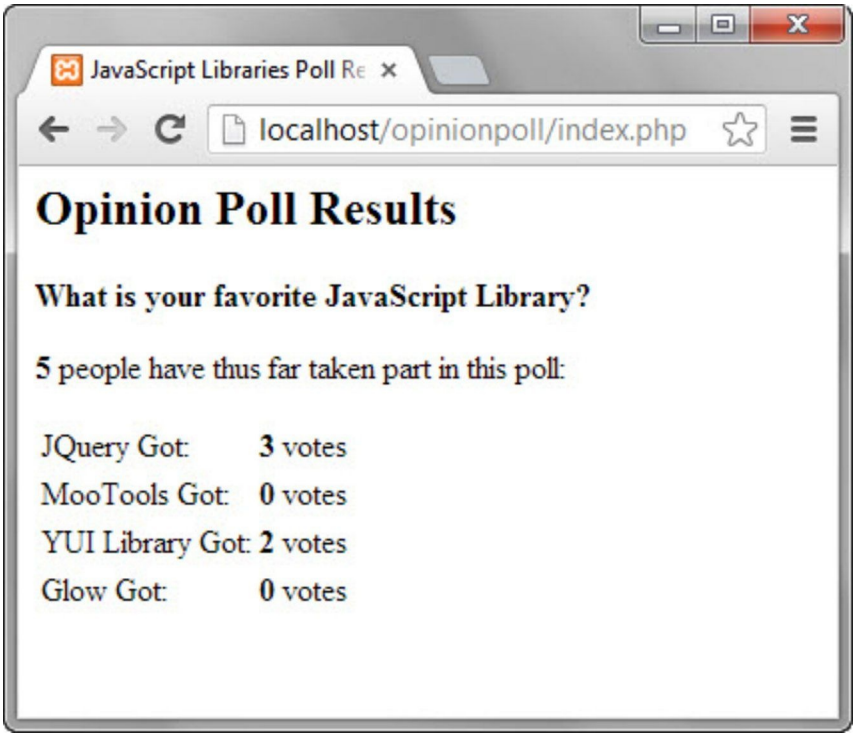
Step 3) Testing our application

Assuming you have saved the files in opinionpoll folder, browse to the URL http://localhost/opinionpoll/

If you click on Ok button without selecting a JS library, you will get the following message box.



Select a JS library then click on OK button. You will get the results page similar to the one shown below.



PHP Ajax Tutorial with Example

What is Ajax?

AJAX is the acronym for AsynchronousJavaScript & XML.

It is a technology that reduces the interactions between the server and client.

It does this by updating only part of a web page rather than the whole page.

The asynchronous interactions are initiated by JavaScript.

JavaScript is a client side scripting language. It is executed on the client side by the web browsers that support JavaScript.JavaScript code only works in browsers that have JavaScript enabled.

XML is the acronym for Extensible Markup Language. It is used to encode messages in both human and machine readable formats. It's like HTML but allows you to create your custom tags. For more details on XML, see the article on XML

Why use AJAX?

- It allows developing rich interactive web applications just like desktop applications.

- Validation can be performed done as the user fills in a form without submitting it. This can be achieved using auto completion. The words that the user types in are submitted to the server for processing. The server responds with keywords that match what the user entered.
- It can be used to populate a dropdown box depending on the value of another dropdown box
- Data can be retrieved from the server and only a certain part of a page updated without loading the whole page. This is very useful for web page parts that load things like
    - Tweets
    - Commens
    - Users visiting the site etc.

How to Create an PHP Ajax application

We will create a simple application that allows users to search for popular PHP MVC frameworks.

Our application will have a text box that users will type in the names of the framework.

We will then use mvc AJAX to search for a match then display the framework's complete name just below the search form.

Step 1) Creating the index page

Index.php

```
<html>

  <head>

    <title>PHP MVC Frameworks - Search Engine</title>

    <script type="text/javascript" src="/auto_complete.js"></script>

  </head>

  <body>

    <h2>PHP MVC Frameworks - Search Engine</h2>

    <p><b>Type the first letter of the PHP MVC Framework</b></p>

    <form method="POST" action="index.php">

      <p><input type="text" size="40" id="txtHint"  onkeyup="showName(this.value)"></p>
```

```
    </form>

    <p>Matches: <span id="txtName"></span></p>

  </body>

</html>
```

HERE,

- “onkeyup="showName(this.value)"” executes the JavaScript function showName everytime a key is typed in the textbox.

  This feature is called auto complete

Step 2) Creating the frameworks page

frameworks.php

```
<?php

$frameworks = array("CodeIgniter","Zend Framework","Cake PHP","Kohana") ;

$name = $_GET["name"];

if (strlen($name) > 0) {

  $match = "";

  for ($i = 0; $i < count($frameworks); $i++) {

    if (strtolower($name) == strtolower(substr($frameworks[$i], 0, strlen($name)))) {

      if ($match == "") {

        $match = $frameworks[$i];

      } else {

        $match = $match . " , " . $frameworks[$i];

      }

    }

  }

}

echo ($match == "") ? 'no match found' : $match;

?>
```

Step 3) Creating the JS script

auto_complete.js

```
<script>
function showName(str){

   if (str.length == 0){ //exit function if nothing has been typed in the textbox

      document.getElementById("txtName").innerHTML=""; //clear previous results

      return;

   }

   if (window.XMLHttpRequest) {// code for IE7+, Firefox, Chrome, Opera, Safari

      xmlhttp=new XMLHttpRequest();

   } else {// code for IE6, IE5

      xmlhttp=new ActiveXObject("Microsoft.XMLHTTP");

   }

   xmlhttp.onreadystatechange=function() {

      if (xmlhttp.readyState == 4 && xmlhttp.status == 200){

         document.getElementById("txtName").innerHTML=xmlhttp.responseText;

      }

   }

   xmlhttp.open("GET","frameworks.php?name="+str,true);

   xmlhttp.send();

}
</script>
```

HERE,

- “if (str.length == 0)” check the length of the string. If it is 0, then the rest of the script is not executed.

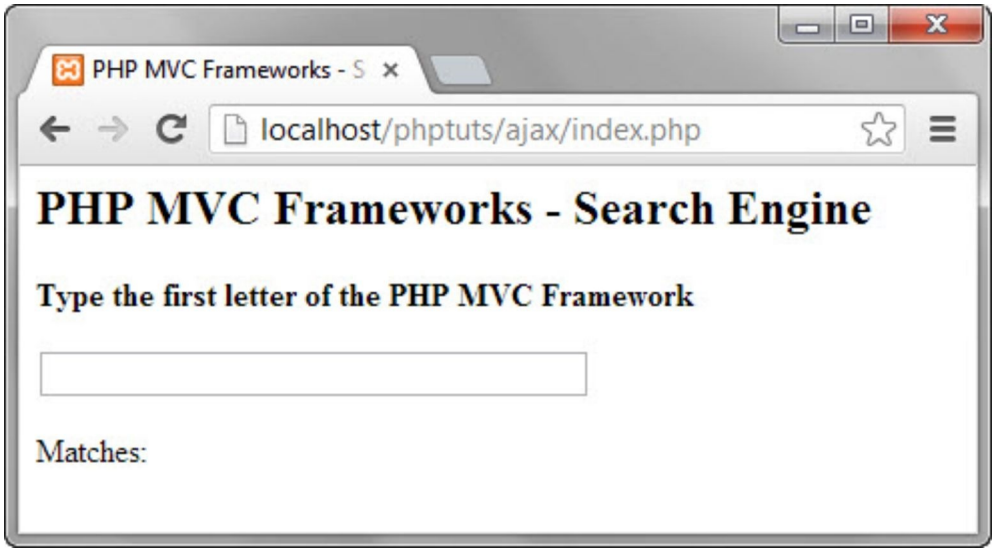
- “if (window.XMLHttpRequest)…” Internet Explorer versions 5 and 6 use

ActiveXObject for AJAX implementation. Other versions and browsers such as Chrome, FireFox use XMLHttpRequest. This code will ensure that our application works in both IE 5 & 6 and other high versions of IE and browsers.

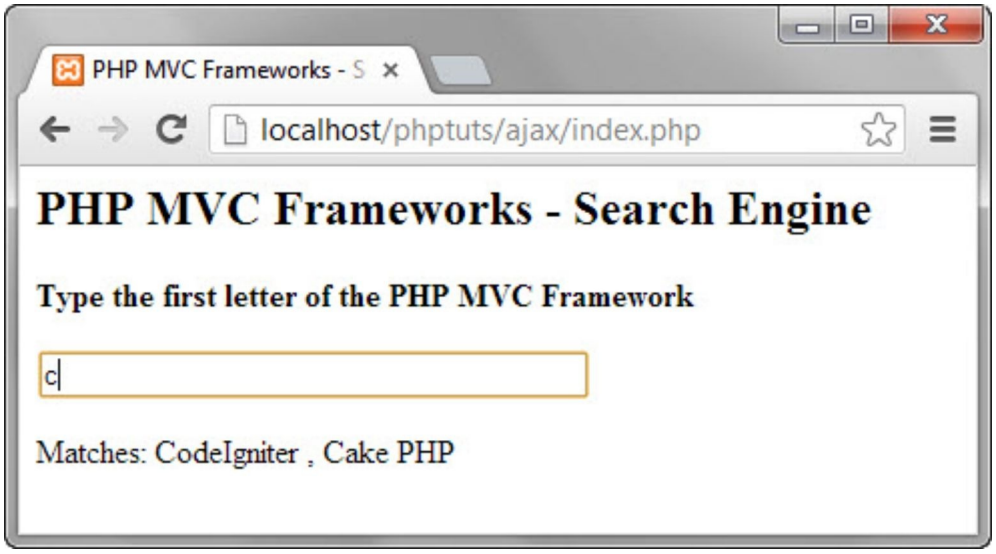
- "xmlhttp.onreadystatechange=function…" checks if the AJAX interaction is complete and the status is 200 then updates the txtName span with the returned results.

Step 4) Testing our PHP Ajax application

Assuming you have saved the file index.php In phututs/ajax, browse to the URL http://localhost/phptuts/ajax/index.php



Type the letter C in the text box You will get the following results.



The above example demonstrates the concept of AJAX and how it can help us create rich interaction applications.

PHP MVC Framework Tutorial: CodeIgniter Example

What is PHP MVC framework?

PHP MVC is an application design pattern that separates the application data and business logic (model) from the presentation (view). MVC stands for Model, View & Controller.

The controller mediates between the models and views.

Think of the MVC design pattern as a car and the driver.

The car has the windscreens (view) which the driver (controller) uses to monitor traffic ahead then speed or brake (model) depending on what he sees ahead.

Why use PHP MVC Framework?

- PHP MVC Frameworks simplify working with complex technologies by;
  - Hiding all the complex implementation details
  - Providing standard methods that we can use to build our applications.
  - Increased developer productivity, this is because the base implementation of activities such as connecting to the database, sanitizing user input etc. are already partially implemented.
  - Adherence to professional coding standards

PHP MVC Design Pattern

Let's now briefly discuss each component of the MVC design pattern.

Model – this part is concerned with the business logic and the application data. It can be used to perform data validations, process data and store it. The data can come from;

- flat file
- database
- XML document
- Other valid data sources.

Controller – this is the part deals with the users' requests for resources from the server.

As an example, when the users requests for the URL *…/index.php?products=list*, the controller will load the products model to retrieve the products data then output the results in the list view.

In a nutshell, the controller links the models and views together depending on the requested resources.

Views – this part deals with presenting the data to the user. This is usually in form of HTML pages.

Types of PHP MVC framework

Selecting the best PHP framework is a challenge.

You don't have to write your own framework to benefit from the advantages of MVC.

You should only attempt to create your own MVC related application design for understanding how MVC frameworks work.

Once you are comfortable with the way MVC frameworks work, you should move on to the mature and already tested frameworks.

The table below briefly describes some of the popular php frameworks and the features that each framework offers.

Framework	Description
CodeIgniter	It is one of the most popular PHP MVC frameworks. It's lightweight and has a short learning curve. It has a rich set of libraries that help build websites and applications rapidly.

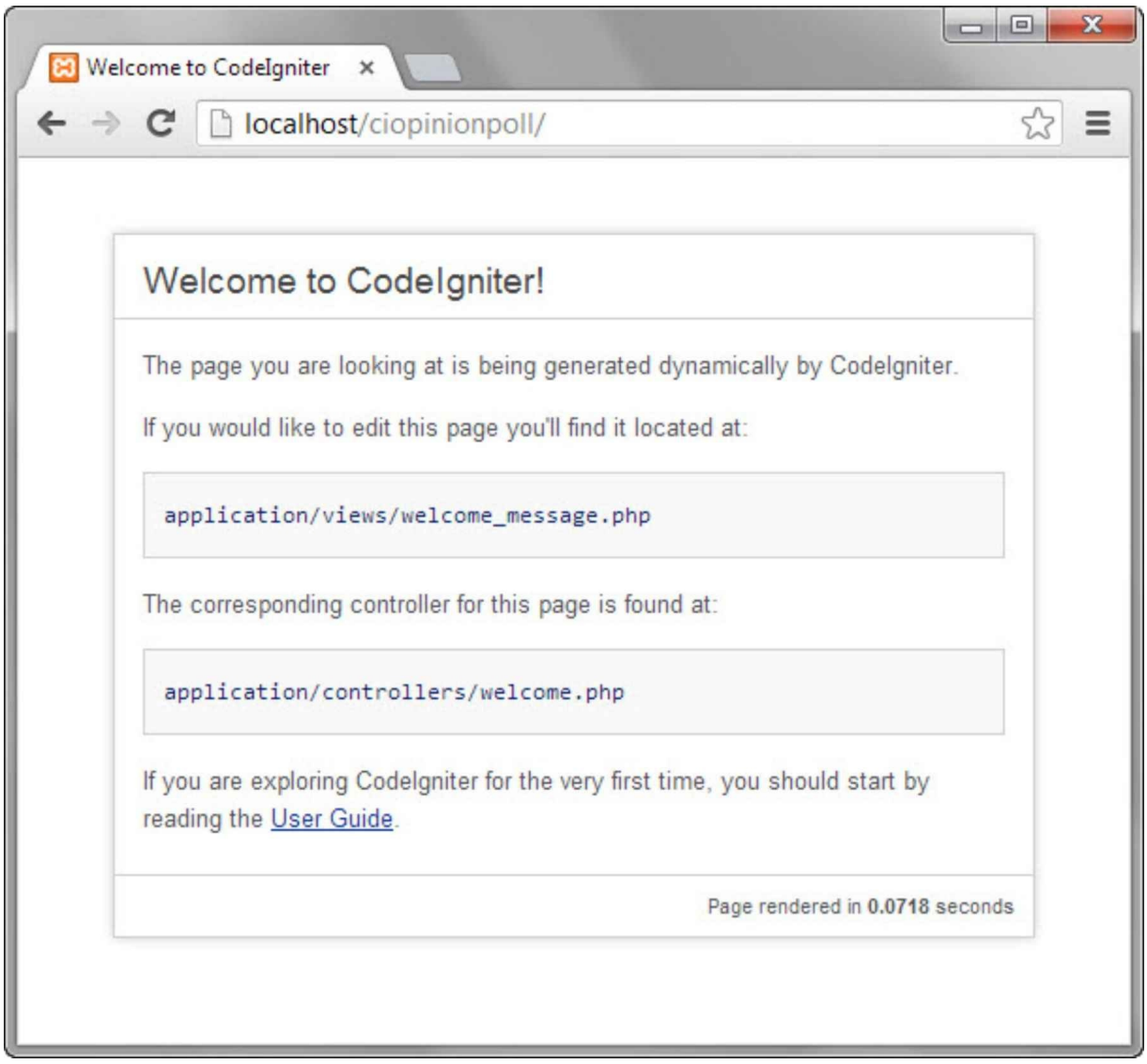
http://ellislab.com/codeigniter	Users with limited knowledge of OOP programming can also use it. CodeIgniter powered applications include; • https://www.pyrocms.com/ • http://www.shopigniter.com/
 Kohana http://kohanaframework.org	It's a Hierarchical Model View Controller HMVC secure and lightweight framework. It has a rich set of components for developing applications rapidly. Companies that use Kohana include; • www.wepay.com • http://kids.nationalgeographic.com/kids/ • http://www.sittercity.com/
 CakePHP www.cakephp.org	It is modeled after Ruby on rails. It's known for concepts such as software design patterns, convention over configuration, ActiveRecord etc. CakePHP powered applications include; • http://invoicemachine.com/ • http://www.fmylife.com/
 www.framework.zend.com Zend	It is a powerful framework that is; • Secure, reliable, fast, and scalable • Supports Web 2.0 and creation of web services. It features APIs from vendors like Amazon, Google, Flickr, Yahoo etc. It's ideal for developing business applications. Zend powered applications include; • Pimcore CMS, • DotKernel. Companies using the Zend framework include; • BBC • Cisco

	• Webex • Offers.com

Porting the opinion poll application to CodeIgniter

In this tutorial, we created a PHP poll application. Here, we will port that code to CodeIgniter

- Download the latest version of CodeIgniter from their website.
- Extract the contents of the zipped file to your development directory in your web server directory. We will use ciopinionpoll as the folder name in this lesson.
- Browse to the URL **http://localhost/ciopinionpoll/**



We are now going to port our opinion poll application to CodeIgniter. Recall that our application was divided into three major components namely the;

- Front controller – this is the part that responds to URL requests and returns the requested page. This code will go into the controller
- Model – this is the code that responds to data requested and returns the requested data. This code will go into the model
- Views – this is the code responsible for formatting and displaying the data. This code will go into the view
    - Browse to ciopinionpoll folder
    - Open the *database.php* file located in application/config directory.
    - Locate the following lines of code

Database configuration settings

```
$db['default']['hostname'] = 'localhost';
$db['default']['username'] = '';
$db['default']['password'] = '';
$db['default']['database'] = '';
```

- Set the username to root
- Set the password to your localhost root password
- Database name to opinion_poll. Note we will be using the database created in the previous lesson.
- Save the changes and close the file.

Creating Our Model

Next we are going to create our model that will extend the CI_Model. The CI_Model is part of the CodeIgniter libraries. The model will be located in application/models *opinion_poll_model.php*

```
<?php
class Opinion_poll_model extends CI_Model
{
   public function __construct()
   {
         $this->load->database();
   }

   public function total_votes()
   {
         $query = $this->db->select('COUNT(choice) as choices_count')->get('js_libraries');
      return $query->row()->choices_count;
   }

   public function get_results()
   {
         $libraries = array("", "JQuery", "MooTools", "YUI Library", "Glow");
      $table_rows = '';

      for ($i = 1; $i < 5; $i++)
      {
         $sql_stmt = "SELECT COUNT(choice) choices_count FROM js_libraries WHERE choice = $i;";
         $result = $model->

         select($sql_stmt); $table_rows .= "<tr><td>" . $ libraries [$i] . " Got:</td><td><b>" . $result[0] . "
</b> votes</td></tr>";
      }
      public function add_vote($choice)
      {
```

```
        $ts = date("Y-m-d H:i:s"); $data = array('choice' => $choice, 'ts' => '$ts'); $this->db->insert('js_libraries', $data);
      }
  }
?>
```

HERE,

- “class Opinion_poll_model extends CI_Model…” is our model that extends the CI_Model
- “…parent:: __construct();” calls the CI_Model constructor
- “$this->load->database();” loads the database library so that our application can interact with the database
- “$this->db->” is CodeIgniter’s active record. Check this link for more information on the active record.

Creating Our Controller Let’s now create the controller. We will use the default CodeIgniter controller located in application/controllers/welcome.php. Replace its source codes with the following code.

```
<?php

if (!defined('BASEPATH'))  exit('No direct script access allowed');

class Welcome extends CI_Controller {

   public function __construct() {

      parent::__construct();

      $this->load->model('opinion_poll_model');

   }

   public function index() {

      if ($this->input->post('submitbutton') && !$this->input->post('vote')) {

         echo "<script>alert('You did not vote!');</script>";

      }

      if ($this->input->post('vote')) {

         $this->opinion_poll_model->add_vote($this->input->post('vote'));

         $data['total_votes'] = $this->opinion_poll_model->total_votes();

         $data['rows'] = $this->opinion_poll_model->get_results();
```

```
            $this->load->view('results', $data);

        } else {

            $this->load->view('opinion_poll_form');

        }

    }

}

/* End of file welcome.php */

/* Location: ./application/controllers/welcome.php */
?>
```

HERE,

- “if (!defined('BASEPATH')) exit('No direct script access allowed');” ensures that users do not directly access the controller class
- “class Welcome extends CI_Controller…” our controller extends the CI_Controller class
- “public function __construct()” calls CI_Controller’s class contructor method and loads our Opinion_poll_model model
- “public function index()…” is the function that maps to index.php. it uses CodeIgniter’s input class to check if a vote has been submitted, add it to the database then display the results. If the post array of the input class is empty, it loads the voting page.
- “$this->input->post('…')” is the CodeIgniter input class that grabs the contents of the $_POST global variable.
- “$this->opinion_poll_model->add_vote($this->input->post('vote'))” calls the model’s add_vote method to add the vote into the database.

Creating Our Views

Recall from the previous example that we had two HTML pages, one for voting and the other for results. We will use the same HTML code with minimal modifications to create our views. Create the following files in application/views directory

```
opinion_poll_form.php
<html>
<head>
   <title>
      JavaScript Libraries - Opinion Poll
   </title>
```

```
</head>

<body>
   <h2>JavaScript Libraries - Opinion Poll</h2>
   <p><b>What is your favorite JavaScript Library?</b></p>
   <form method="POST" action="index.php">
      <p>
         <input type="radio" name="vote" value="1" /> JQuery
         <br />
         <input type="radio" name="vote" value="2" /> MooTools
         <br />
         <input type="radio" name="vote" value="3" /> YUI Library
         <br />
         <input type="radio" name="vote" value="4" /> Glow </p>
      <p>
         <input type="submit" name="submitbutton" value="OK" />
      </p>
   </form>
</body>
</html>
```

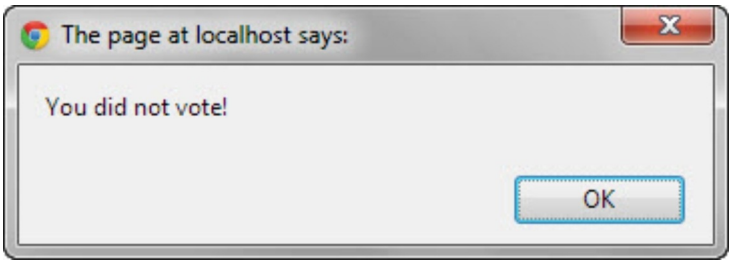
Let’s now create the results page results.php

```
<html>
   <head>
      <title>JavaScript Libraries - Opinion Poll Results</title>
   </head>
   <body>

      <h2>JavaScript Libraries - Opinion Poll Results</h2>

      <p><b>What is your favorite JavaScript Library?</b></p>

      <p><b><?php echo $total_votes; ?></b> people have thus far taken part in this poll:</p>

      <p><table><tr><td>

         <?php print($rows); ?>

      </tr></td></table></p>

      <p><a href="">Return to voting page</a></p>
</body>
</html>
```

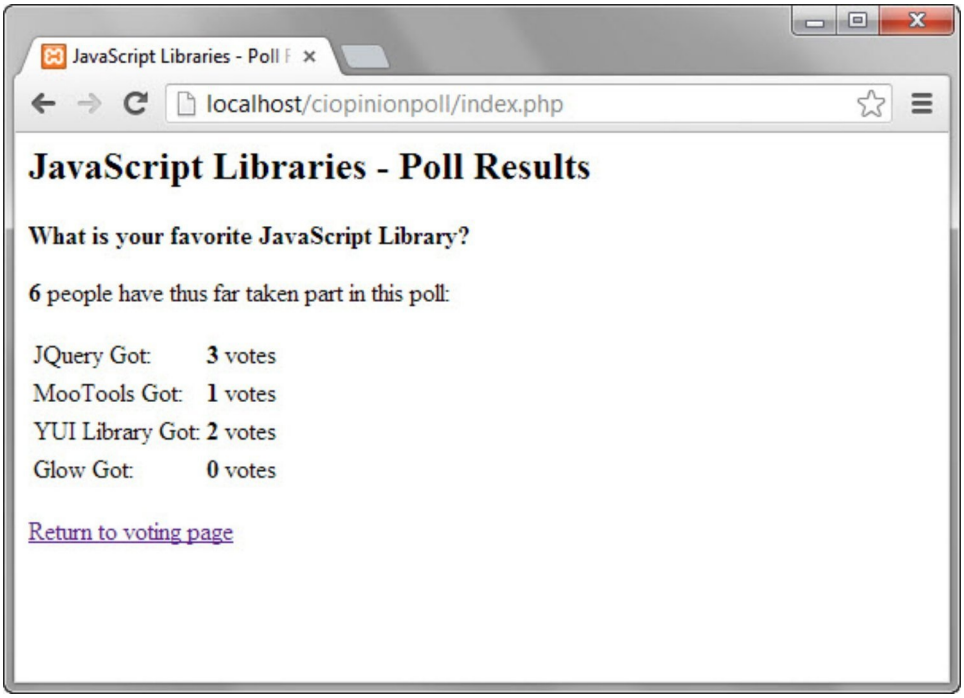
Testing our application

Assuming the root directory of your application is ciopinion, browse to http://localhost/ciopionpoll/

Click on OK button, you will see the following alert message



Vote for your favorite candidate then click on OK You will see the following results page



Conclusion

CodeIgniter is an easy to learn and use PHP MVC framework that can greatly

reduce the time spent developing applications.

PHP vs JavaScript: Must Know Differences

PHP

Is not fair to compare PHP vs JavaScript, as they both have different purposes for web-site development. PHP is a server-side scripting language while JavaScript is a client-side scripting language. In fact, the most dynamic website is created when we use functions of both these languages together. If PHP is like a paint-brush to paint picture, then JavaScript is a paint-color.

PHP stands for "Hypertext Preprocessor", is a programming language embedded in HTML that does all sort of things like build custom web content, send and receive cookies, evaluate form data sent from a browser, etc. It is integrated with number of popular databases like Postgre SQL, Oracle, Sybase, SQL, and MySQL. PHP also supports major protocols like IMAP, POP3 and LDAP.

PHP can handle forms, save data to a file, return data to the user, gather data from files, etc.

Example: Let say a website that takes user to view the order status after logging in. By PHP coding, you would send a query to the database that would then output the specific user information based on what information is in the database

JavaScript

While, JavaScript is designed for creating network-centric applications. With JavaScript, web pages will no longer be static HTML and allows the program that interacts with the user, control the browser, and dynamically create the HTML content. The advantage of JavaScript is that it has less server interaction, allowing you to validate user input before sending the page off which means less load on your server and less server traffic. JavaScript allows immediate feedback to the visitors.

Example: When you hover over the menu tab on the web-page, the drop down effect is done through JavaScript.

Note: JavaScript now supports server side execution via NodeJS

PHP vs Java-Script

Features	Java-Script	PHP
Developed by	Brendan Eich (1995)	Rasmus Lerdorf (1994)
Object-oriented	Yes	Yes
Easy to use existing code	Yes	Yes
Server side scripting language	No	Yes
Client side scripting language	Yes	No
Accepts both upper case and lower case boolean variable	No	Yes
Case sensitive to variables	Yes	Yes
Case sensitive in function	Yes	No
Objects & Arrays interchangeable	Yes	No
Requires HTTP to execute	Yes	Yes
Updates files on server	No	Yes
Execute with browser window	Yes	No
Supports framework	Yes	Yes
Platform Independent	Yes	Yes
Open Source	Yes	Yes
Support database	No	Yes
Memory Management (garbage collection)	Yes	Yes

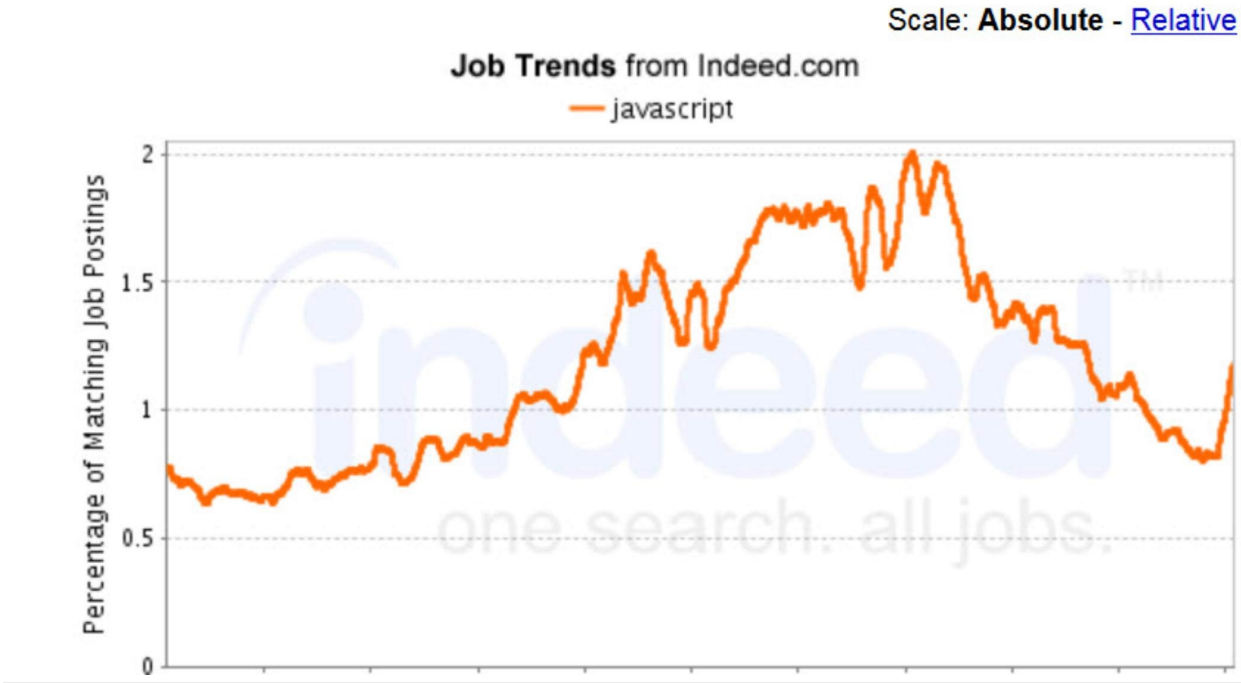
Library	Yes	Yes
Exceptional Handling	Yes	Yes
Performance	Fast	Slow
Support of features	Less	More

Repository

Repository	**JavaScript**	**PHP**
Github	404077	387773
Stack-Overflow	1639397	1207635
Source-Forge	10814	25090

Trend of JavaScript vs PHP

Job trends from indeed.com shows millions of jobs search for Java-script and PHP from thousands of job sites. It relatively shows the growing and falling trend of the job for both languages in consecutive years.



Likewise, regional interest of people for these two languages are also highlighted over-here. In graph, though we can see the fall of interest of PHP and JavaScript languages over the period of time due to introduction of new languages,

JavaScript still remains on top of PHP.

Top 100 PHP Interview Questions and Answers

1) What is PHP?

PHP is a web language based on scripts that allow developers to dynamically create generated web pages.

2) What do the initials of PHP stand for?

PHP means PHP: Hypertext Preprocessor.

3) Which programming language does PHP resemble?

PHP syntax resembles Perl and C

4) What does PEAR stand for?

PEAR means "PHP Extension and Application Repository". It extends PHP and provides a higher level of programming for web developers.

5) What is the actually used PHP version?

Version 7.1 or 7.2 is the recommended version of PHP.

6) How do you execute a PHP script from the command line?

Just use the PHP command line interface (CLI) and specify the file name of the script to be executed as follows:

```
php script.php
```

7) How to run the interactive PHP shell from the command line interface?

Just use the PHP CLI program with the option -a as follows:

```
php -a
```

8) What is the correct and the most two common way to start and finish a PHP block of code?

The two most common ways to start and finish a PHP script are:

```
<?php [   ---  PHP code---- ] ?> and <? [---  PHP code  ---] ?>
```

9) How can we display the output directly to the browser?

To be able to display the output directly to the browser, we have to use the special tags <?= and ?>.

10) What is the main difference between PHP 4 and PHP 5?

PHP 5 presents many additional OOP (Object Oriented Programming) features.

11) Is multiple inheritance supported in PHP?

PHP supports only single inheritance; it means that a class can be extended from only one single class using the keyword 'extended'.

12) What is the meaning of a final class and a final method?

'final' is introduced in PHP5. Final class means that this class cannot be extended and a final method cannot be overridden.

13) How is the comparison of objects done in PHP?

We use the operator '==' to test is two objects are instanced from the same class and have same attributes and equal values. We can test if two objects are referring to the same instance of the same class by the use of the identity operator '==='.

14) How can PHP and HTML interact?

It is possible to generate HTML through PHP scripts, and it is possible to pass pieces of information from HTML to PHP.

15) What type of operation is needed when passing values through a form or an URL?

If we would like to pass values through a form or an URL, then we need to encode and to decode them using htmlspecialchars() and urlencode().

16) How can PHP and Javascript interact?

PHP and Javascript cannot directly interact since PHP is a server side language and Javascript is a client-side language. However, we can exchange variables since PHP can generate Javascript code to be executed by the browser and it is possible to pass specific variables back to PHP via the URL.

17) What is needed to be able to use image function?

GD library is needed to execute image functions.

18) What is the use of the function 'imagetypes()'?

imagetypes() gives the image format and types supported by the current version of GD-PHP.

19) What are the functions to be used to get the image's properties (size, width, and height)?

The functions are getimagesize() for size, imagesx() for width and imagesy() for height.

20) How failures in execution are handled with include() and require() functions?

If the function require() cannot access the file then it ends with a fatal error.

However, the include() function gives a warning, and the PHP script continues to execute.

21) What is the main difference between require() and require_once()?

require(), and require_once() perform the same task except that the second function checks if the PHP script is already included or not before executing it.

(same for include_once() and include())

22) How can I display text with a PHP script?

Two methods are possible:

```
<!--?php echo "Method 1"; print "Method 2"; ?-->
```

23) How can we display information of a variable and readable by a human with PHP?

To be able to display a human-readable result we use print_r().

24) How is it possible to set an infinite execution time for PHP script?

The set_time_limit(0) added at the beginning of a script sets to infinite the time of execution to not have the PHP error 'maximum execution time exceeded.' It is also possible to specify this in the php.ini file.

25) What does the PHP error 'Parse error in PHP - unexpected T_variable at line x' means?

This is a PHP syntax error expressing that a mistake at the line x stops parsing and executing the program.

26) What should we do to be able to export data into an Excel file?

The most common and used way is to get data into a format supported by Excel. For example, it is possible to write a .csv file, to choose for example comma as a separator between fields and then to open the file with Excel.

27) What is the function file_get_contents() useful for?

file_get_contents() lets reading a file and storing it in a string variable.

28) How can we connect to a MySQL database from a PHP script?

To be able to connect to a MySQL database, we must use mysqli_connect() function as follows:

```
<!--?php $database = mysqli_connect("HOST", "USER_NAME", "PASSWORD");
mysqli_select_db($database,"DATABASE_NAME"); ?-->
```

29) What is the function mysql_pconnect() useful for?

mysql_pconnect() ensure a persistent connection to the database, it means that the connection does not close when the PHP script ends.

This function is not supported in PHP 7.0 and above

30) How be the result set of Mysql handled in PHP?

The result set can be handled using mysqli_fetch_array, mysqli_fetch_assoc, mysqli_fetch_object or mysqli_fetch_row.

31) How is it possible to know the number of rows returned in the result set?

The function mysqli_num_rows() returns the number of rows in a result set.

32) Which function gives us the number of affected entries by a query?

mysqli_affected_rows() return the number of entries affected by an SQL query.

33) What is the difference between mysqli_fetch_object() and mysqli_fetch_array()?

The mysqli_fetch_object() function collects the first single matching record where mysqli_fetch_array() collects all matching records from the table in an array.

34) How can we access the data sent through the URL with the GET method?

To access the data sent via the GET method, we use $_GET array like this:

```
www.url.com?var=value
$variable = $_GET["var"]; this will now contain 'value'
```

35) How can we access the data sent through the URL with the POST method?

To access the data sent this way, you use the $_POST array.

Imagine you have a form field called 'var' on the form when the user clicks submit to the post form, you can then access the value like this:

```
$_POST["var"];
```

36) How can we check the value of a given variable is a number?

It is possible to use the dedicated function, is_numeric() to check whether it is a number or not.

37) How can we check the value of a given variable is alphanumeric?

It is possible to use the dedicated function, ctype_alnum to check whether it is an alphanumeric value or not.

38) How do I check if a given variable is empty?

If we want to check whether a variable has a value or not, it is possible to use the empty() function.

39) What does the unlink() function mean?

The unlink() function is dedicated for file system handling. It simply deletes the file given as entry.

40) What does the unset() function mean?

The unset() function is dedicated for variable management. It will make a variable undefined.

41) How do I escape data before storing it in the database?

The addslashes function enables us to escape data before storage into the database.

42) How is it possible to remove escape characters from a string?

The stripslashes function enables us to remove the escape characters before apostrophes in a string.

43) How can we automatically escape incoming data?

We have to enable the Magic quotes entry in the configuration file of PHP.

44) What does the function get_magic_quotes_gpc() means?

The function get_magic_quotes_gpc() tells us whether the magic quotes is switched on or no.

45) Is it possible to remove the HTML tags from data?

The strip_tags() function enables us to clean a string from the HTML tags.

46) what is the static variable in function useful for?

A static variable is defined within a function only the first time, and its value can be modified during function calls as follows:

```
<!--?php function testFunction() { static $testVariable = 1; echo $testVariable; $testVariable++; } testFunction();  //1 testFunction();  //2 testFunction();  //3 ?-->
```

47) How can we define a variable accessible in functions of a PHP script?

This feature is possible using the global keyword.

48) How is it possible to return a value from a function?

A function returns a value using the instruction 'return $value;'.

49) What is the most convenient hashing method to be used to hash passwords?

It is preferable to use crypt() which natively supports several hashing algorithms or the function hash() which supports more variants than crypt() rather than using the common hashing algorithms such as md5, sha1 or sha256 because they are conceived to be fast. Hence, hashing passwords with these algorithms can create vulnerability.

50) Which cryptographic extension provide generation and verification of digital signatures?

The PHP-OpenSSL extension provides several cryptographic operations including generation and verification of digital signatures.

51) How is a constant defined in a PHP script?

The define() directive lets us defining a constant as follows:

```
define ("ACONSTANT", 123);
```

52) How can you pass a variable by reference?

To be able to pass a variable by reference, we use an ampersand in front of it, as follows $var1 = &$var2

53) Will a comparison of an integer 12 and a string "13" work in PHP?

"13" and 12 can be compared in PHP since it casts everything to the integer type.

54) How is it possible to cast types in PHP?

The name of the output type has to be specified in parentheses before the variable which is to be cast as follows:

* (int), (integer) - cast to integer
* (bool), (boolean) - cast to boolean
* (float), (double), (real) - cast to float
* (string) - cast to string
* (array) - cast to array
* (object) - cast to object

55) When is a conditional statement ended with endif?

When the original if was followed by: and then the code block without braces.

56) How is the ternary conditional operator used in PHP?

It is composed of three expressions: a condition, and two operands describing what instruction should be performed when the specified condition is true or false as follows:

```
Expression_1?Expression_2 : Expression_3;
```

57) What is the function func_num_args() used for?

The function func_num_args() is used to give the number of parameters passed into a function.

58) If the variable $var1 is set to 10 and the $var2 is set to the character var1, what's the value of $$var2?

$$var2 contains the value 10.

59) What does accessing a class via :: means?

:: is used to access static methods that do not require object initialization.

60) In PHP, objects are they passed by value or by reference?

In PHP, objects passed by value.

61) Are Parent constructors called implicitly inside a class constructor?

No, a parent constructor have to be called explicitly as follows:

```
parent::constructor($value)
```

62) What's the difference between __sleep and __wakeup?

__sleep returns the array of all the variables that need to be saved, while __wakeup retrieves them.

63) What is faster?

1- Combining two variables as follows:

```
$variable1 = 'Hello ';

$variable2 = 'World';

$variable3 = $variable1.$variable2;
```

Or

```
2- $variable3 = "$variable1$variable2";
```

$variable3 will contain "Hello World". The first code is faster than the second code especially for large large sets of data.

64) what is the definition of a session?

A session is a logical object enabling us to preserve temporary data across multiple PHP pages.

65) How to initiate a session in PHP?

The use of the function session_start() lets us activating a session.

66) How can you propagate a session id?

You can propagate a session id via cookies or URL parameters.

67) What is the meaning of a Persistent Cookie?

A persistent cookie is permanently stored in a cookie file on the browser's computer. By default, cookies are temporary and are erased if we close the browser.

68) When do sessions end?

Sessions automatically end when the PHP script finishes executing but can be manually ended using the session_write_close().

69) What is the difference between session_unregister() and session_unset()?

The session_unregister() function unregister a global variable from the current

session and the session_unset() function frees all session variables.

70) What does $GLOBALS mean?

$GLOBALS is associative array including references to all variables which are currently defined in the global scope of the script.

71) What does $_SERVER mean?

$_SERVER is an array including information created by the web server such as paths, headers, and script locations.

72) What does $_FILES means?

$_FILES is an associative array composed of items sent to the current script via the HTTP POST method.

73) What is the difference between $_FILES['userfile']['name'] and $_FILES['userfile']['tmp_name']?

$_FILES['userfile']['name'] represents the original name of the file on the client machine,

$_FILES['userfile']['tmp_name'] represents the temporary filename of the file stored on the server.

74) How can we get the error when there is a problem to upload a file?

$_FILES['userfile']['error'] contains the error code associated with the uploaded file.

75) How can we change the maximum size of the files to be uploaded?

We can change the maximum size of files to be uploaded by changing upload_max_filesize in php.ini.

76) What does $_ENV mean?

$_ENV is an associative array of variables sent to the current PHP script via the environment method.

77) What does $_COOKIE mean?

$_COOKIE is an associative array of variables sent to the current PHP script using the HTTP Cookies.

78) What does the scope of variables mean?

The scope of a variable is the context within which it is defined. For the most part, all PHP variables only have a single scope. This single scope spans included and required files as well.

79) what the difference between the 'BITWISE AND' operator and the 'LOGICAL AND' operator?

$a and $b: TRUE if both $a and $b are TRUE.

$a & $b: Bits that are set in both $a and $b are set.

80) What are the two main string operators?

The first is the concatenation operator ('.'), which returns the concatenation of its right and left arguments. The second is ('.='), which appends the argument on the right to the argument on the left.

81) What does the array operator '===' means?

$a === $b TRUE if $a and $b have the same key/value pairs in the same order and of the same types.

82) What is the differences between $a != $b and $a !== $b?

!= means inequality (TRUE if $a is not equal to $b) and !== means non-identity (TRUE if $a is not identical to $b).

83) How can we determine whether a PHP variable is an instantiated object of a certain class?

To be able to verify whether a PHP variable is an instantiated object of a certain class we use instanceof.

84) What is the goto statement useful for?

The goto statement can be placed to enable jumping inside the PHP program. The target is pointed by a label followed by a colon, and the instruction is specified as a goto statement followed by the desired target label.

85) what is the difference between Exception::getMessage and Exception:: getLine?

Exception::getMessage lets us getting the Exception message and Exception::getLine lets us getting the line in which the exception occurred.

86) What does the expression Exception::__toString means?

Exception::__toString gives the String representation of the exception.

87) How is it possible to parse a configuration file?

The function parse_ini_file() enables us to load in the ini file specified in filename and returns the settings in it in an associative array.

88) How can we determine whether a variable is set?

The boolean function isset determines if a variable is set and is not NULL.

89) What is the difference between the functions strstr() and stristr()?

The string function strstr(string allString, string occ) returns part of allString from the first occurrence of occ to the end of allString. This function is case-

sensitive. stristr() is identical to strstr() except that it is case insensitive.

90) what is the difference between for and foreach?

for is expressed as follows:

for (expr1; expr2; expr3)

statement

The first expression is executed once at the beginning. In each iteration, expr2 is evaluated. If it is TRUE, the loop continues, and the statements inside for are executed. If it evaluates to FALSE, the execution of the loop ends. expr3 is tested at the end of each iteration.

However, foreach provides an easy way to iterate over arrays, and it is only used with arrays and objects.

91) Is it possible to submit a form with a dedicated button?

It is possible to use the document.form.submit() function to submit the form. For example: <input type=button value="SUBMIT" onClick="document.form.submit()">

92) What is the difference between ereg_replace() and eregi_replace()?

The function eregi_replace() is identical to the function ereg_replace() except that it ignores case distinction when matching alphabetic characters.

93) Is it possible to protect special characters in a query string?

Yes, we use the urlencode() function to be able to protect special characters.

94) What are the three classes of errors that can occur in PHP?

The three basic classes of errors are notices (non-critical), warnings (serious errors) and fatal errors (critical errors).

95) What is the difference between characters \034 and \x34?

\034 is octal 34 and \x34 is hex 34.

96) How can we pass the variable through the navigation between the pages?

It is possible to pass the variables between the PHP pages using sessions, cookies or hidden form fields.

97) Is it possible to extend the execution time of a PHP script?

The use of the set_time_limit(int seconds) enables us to extend the execution time of a PHP script. The default limit is 30 seconds.

98) Is it possible to destroy a cookie?

Yes, it is possible by setting the cookie with a past expiration time.

99) What is the default session time in PHP?

The default session time in php is until the closing of the browser

100) Is it possible to use COM component in PHP?

Yes, it's possible to integrate (Distributed) Component Object Model components ((D)COM) in PHP scripts which is provided as a framework.

101) Explain whether it is possible to share a single instance of a Memcache between multiple PHP projects?

Yes, it is possible to share a single instance of Memcache between multiple projects. Memcache is a memory store space, and you can run memcache on one or more servers. You can also configure your client to speak to a particular set of instances. So, you can run two different Memcache processes on the same host and yet they are completely independent. Unless, if you have partitioned your data, then it becomes necessary to know from which instance to get the data from or to put into.

102) Explain how you can update Memcached when you make changes to PHP?

When PHP changes you can update Memcached by

- **Clearing the Cache proactively:** Clearing the cache when an insert or update is made
- **Resetting the Cache:** It is similar to the first method but rather than just deleting the keys and waiting for the next request for the data to refresh the cache, reset the values after the insert or update.