# PHP and MySql

Tiji Thomas

HOD
Department of Computer Applications
MACFAST
tiji@macfast.org

## What is PHP?

- ❖PHP stands for **P**HP: **H**ypertext **P**reprocessor
- ❖PHP is a server-side scripting language, like ASP
- ❖PHP scripts are executed on the server
- ❖PHP supports many databases (MySQL, Oracle, Sybase, PostgreSQL etc.)
- ❖PHP is an open source software

## What is a PHP File?

- PHP files can contain text, HTML tags and scripts
- PHP files have a file extension of ".php"

## Basic PHP Syntax

A PHP scripting block always starts with
**<?php**
and ends with
**?>**.
A PHP scripting block can be placed anywhere in the document.

```
<?php

?>
```

Example

```
<html>
<body>
<?php
    echo "Hello World";
?>
</body>
</html>
```

## Comments in PHP

In PHP, we use // to make a single-line comment or /* and */ to make a large comment block.

```
<html>
<body>
<?php
//This is a comment
/*
This is
a comment
block
*/
?>
</body>
</html>
```

## Variables in PHP

All variables in PHP start with a $ sign symbol.

The correct way of setting a variable in PHP:

```
$var_name = value;
```

❖**PHP is a Loosely Typed Language**

In PHP a variable does not need to be declared before being set.

# Variable Naming Rules

- ❖A variable name must start with a letter or an underscore "_"

- ❖A variable name can only contain alpha-numeric characters and underscores (a-Z, 0-9, and _ )

- ❖A variable name should not contain spaces.

# PHP Operators

**Arithmetic Operators**

1. +
2. -
3. *
4. /
5. %
6. ++
7. --

**Assignment Operators**

1. =
2. +=
3. -=
4. *=
5. /=
6. .=
7. %=

## PHP Operators

### Comparison Operators

1. ==
2. !=
3. >
4. <
5. >=
6. <=

### Logical Operators

1. &&
2. ||
3. !

## The Ternary Operator

It is called the ternary operator because it takes three operands - a condition, a result for true, and a result for false

```
<?php
   $agestr = ($age < 16) ? 'child' : 'adult';
?>
```

## Strings in PHP

Example

```
<?php
$txt="Hello World";
echo $txt;
?>
```

## The Concatenation Operator

The concatenation operator (.)  is used to put two string values together.

Example:

```
<?php
$txt1="Department  :";
$txt2="of Computer Applications , UC College";
echo $txt1 . " " . $txt2;
?>
```

## 1. strlen()

**The strlen() function is used to find the length of a string**

```
<?php
echo strlen("Hello world!");
?>
```

## 2. strpos()

The strpos() function is used to search for a string or character within a string

```
<?php

echo strpos("Hello world!","world");

?>
```

## PHP Date()

Synatax:

date(format,timestamp)

| **Parameter** | **Description** |
|---|---|
| format | Required. Specifies the format of the timestamp |
| timestamp | Optional. Specifies a timestamp. Default is the current date and time (as a timestamp) |

## PHP Date - Format the Date

The first parameter in the date() function specifies how to format the date/time. It uses letters to represent date and time formats. Here are some of the letters that can be used:

- d - The day of the month (01-31)
- m - The current month, as a number (01-12)
- Y - The current year in four digits
- l -A full textual representation of the day of the week (Sunday through Saturday )
- F - A full textual representation of a month, such as January or March (January through December
  S-English ordinal suffix for the day of the month, 2 characters
  (st, nd, rd or th)

## Example - 23

```
<?php
echo date("Y/m/d");
echo "<br />";
echo date("Y.m.d");
echo "<br />";
echo date("Y-m-d");
echo "<br />";

echo date("D-M-Y");
echo "<br />";

echo date('l dS \of F Y');


?>
```

## Conditional Statements

**The If...Else Statement**

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

Example -3

```
<html>
<body>
<?php
$d=date("D");
if ($d=="Fri")
  {
  echo "Hello!<br />";
  echo "Have a nice weekend!";
  echo "See you on Monday!";
  }
?>
</body>
</html>
```

## The If...Else Statement

```
<html>
<body>
<?php
$d=date("D");
if ($d=="Fri")
  echo "Have a nice weekend!";
else
  echo "Have a nice day!";
?>
</body>
</html>
```

## The If...Else Statement

```
html>
<body>
<?php
$d=date("D");
if ($d=="Fri")
  echo "Have a nice weekend!";
elseif ($d=="Sun")
  echo "Have a nice Sunday!";
else
  echo "Have a nice day!";
?>
</body>
</html>
```

## The Switch Statement

```
 switch (expression)
{
case label1:
  code to be executed if expression = label1;
  break;
case label2:
  code to be executed if expression = label2;
  break;
default:
  code to be executed
  if expression is different
  from both label1 and label2;
}
```

## Example - Switch

```
<html>
<body>
<?php
$x = 1;
switch ($x)
{
case 1:
  echo "Number 1";
  break;
case 2:
  echo "Number 2";
  break;
case 3:
  echo "Number 3";
  break;
default:
  echo "No number between 1 and 3";
}
?>
</body>
</html>
```

## PHP Arrays

1. ***Numeric array*** *- An array with a numeric ID key*
2. ***Associative array*** *- An array where each ID key is associated with a value*
3. ***Multidimensional array*** *- An array containing one or more arrays*

## Numeric Arrays

A numeric array stores each element with a numeric ID key.

There are different ways to create a numeric array

```
$names = array(“Raju",“Rajan",“Ramu");
```

```
$names[0] = “Raju";
$names[1] = “Rajan";
$names[2] = “Ramu";
```

## Example

- <?php
- $names[0] = “Raju";
- $names[1] = “Meera";
- $names[2] = “Ramu";
- echo $names[1] . " and " . $names[2] .
- " are ". $names[0] . "'s neighbors";
- ?>

## Associative Arrays

An associative array, each ID key is associated with a value.

```
$ages = array(“Tony"=>32, “Rony"=>30,
“Sony"=>34)
```

```
$ages[‘Tony'] = "32";
$ages[‘Rony'] = "30";
$ages[‘Sony'] = "34";
```

Example -8

```
<?php
$ages["Rony"] = "32";
$ages['Tony'] = "30";
$ages['Sony'] = "34";
echo “Tony is " . $ages["Tony"] . " years old.";
?>
```

## Multidimensional Arrays

In a multidimensional array, each element in the main array can also be an array. And each element in the sub-array can be an array, and so on.

```
$families = array
(
  “Thomas"=>array
  (
  “Roni",
  “Toni",
  “Soni"
  ),
  “Joseph"=>array
  (
  “Jomy"
  ),
  “Mathew"=>array
  (
  “Rani",
  “Ramu",
  “Reena"
  )
);
print_r($families["Thomas"][2]);
```

## Looping

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

## The while Statement

```
while (condition)
{
code to be executed;
}
```

```
<html>
<body>
<?php
$i=1;
while($i<=5)
  {
  echo "The number is " . $i . "<br />";
  $i++;
  }
?>
</body>
</html>
```

## The do...while Statement

The do...while statement will execute a block of code **at least once** - it then will repeat the loop **as long as** a condition is true.

```
do
{
code to be executed;
}
while (condition);
```

Example-12

```
<html>
<body>
<?php
$i=0;
do
  {
  $i++;
  echo "The number is " . $i . "<br />";
  }
while ($i<5);
?>
</body>
</html>
```

## For Statement

```
for (init; cond; incr)
{
    code to be executed;
}
```

```
<html>
<body>
<?php
for ($i=1; $i<=5; $i++)
{
  echo "Hello World!<br />";
}
?>
</body>
</html>
```

## Foreach Statement

```
foreach (array as value)
{
   code to be executed;
}
```

Example-14

```
 <html>
<body>
<?php
$arr=array("one", "two", "three");
foreach ($arr as $value)
{
  echo "Value: " . $value . "<br />";
}
?>
</body>
</html>
```

## Sorting an array

```
<?php

$fruits = array("lemon", "orange", "banana",
"apple");
sort($fruits);
foreach ($fruits as $key => $val) {
   echo "fruits[" . $key . "] = " . $val . "\n";
}

?>
```

# PHP Functions

## Creating PHP functions

- All functions start with the word "function()"
- The name can start with a letter or underscore (not a number)
- Add a "{" - The function code starts after the opening curly brace
- Insert the function code
- Add a "}" - The function is finished by a closing curly brace

## Example -15

```
<html>
<body>
<?php
function writeMyName()
  {
  echo "Raju Thomas";
  }
writeMyName();
 ?>
</body>
</html>
```

## PHP Functions - Adding parameters

▪

```
<html>
<body>
<?php
function writeMyName($fname)
  {
  echo $fname ."<br />";
  }
echo "My name is ";
writeMyName(“Thomas");
echo "My name is ";
writeMyName(“Varghese");
echo "My name is ";
writeMyName("Tom");
?>
</body>
</html>
```

## Function return values

Example-17

```
<html>
<body>
<?php
function add($x,$y)
  {
        $total = $x + $y;
        return $total;
  }
echo "1 + 16 = " . add(1,16);
?>
</body>
</html>
```

## Form Handling

•A form is an area that can contain form elements.

•Form elements are elements that allow the user to enter information (like text fields ,drop-down menus, radio buttons, checkboxes, etc.) in a form.

•A form is defined with the <form> tag.

# Form Handling In HTML

```
<form>
  <input>
  <input>
</form>
```

**Input**
**The most used form tag is the <input> tag. The type of input is specified with the type attribute. The most commonly used input types are explained below.**

1. **Text Fields**

**Text fields are used when you want the user to type letters, numbers, etc. in a form.**

```
<form>
First name:
<input type="text" name="firstname">
<br>
Last name:
<input type="text" name="lastname">
</form>
```

## Radio Buttons

Radio Buttons are used when you want the user to select one of a limited number of choices.

```
<form>
<input type="radio" name="sex" value="male"> Male
<br>
<input type="radio" name="sex" value="female"> Female
</form>
```

## Checkboxes

Checkboxes are used when you want the user to select one or more options of a limited number of choices.

Example-20

```
<form>
I have a bike:
<input type="checkbox" name="vehicle" value="Bike">
<br>
I have a car:
<input type="checkbox" name="vehicle" value="Car">
<br>
I have an airplane:
<input type="checkbox" name="vehicle" value="Airplane">
</form>
```

## The Form's Action Attribute and the Submit Button

```
<form name="input" action="html_form_action.asp"
method="get">
Username:
<input type="text" name="user">
<input type="submit" value="Submit">
</form>
```

## Example -22

```
<body>

<form >
<select name="cars">
<option value="800">  Maruthi 800</option>
<option value="Alto">Maruthi ALTO</option>
<option value="Wagonor">Maruthi Wagonor</option>
<option value="Swift">Maruthi Swift</option>
</select>
</form>

</body>
</html>
```

# PHP Form handling

The most important thing to notice when dealing with HTML forms and PHP is that any form element in an HTML page will **automatically** be available to your PHP scripts.

```
<html>
<body>
<form action="welcome.php" method="post">
Name: <input type="text" name="name" />
Age: <input type="text" name="age" />
<input type="submit" />
</form>
</body>
</html>
```

Welcome.php

```
<html>
<body>
Welcome <?php echo $_POST["name"]; ?>.<br />
You are <?php echo $_POST["age"]; ?> years old.
</body>
</html
```

## The $_GET Variable

The $_GET variable is an array of variable names and values sent by the HTTP GET method.

The $_GET variable is used to collect values from a form with method="get". Information sent from a form with the GET method is visible to everyone (it will be displayed in the browser's address bar) and it has limits on the amount of information to send (max. 100 characters).

## The $_POST Variable

The $_POST variable is an array of variable names and values sent by the HTTP POST method.

The $_POST variable is used to collect values from a form with method="post". Information sent from a form with the POST method is invisible to others and has no limits on the amount of information to send.

## The $_REQUEST Variable

**The PHP $_REQUEST variable contains the contents of both $_GET, $_POST.**
**The PHP $_REQUEST variable can be used to get the result from form data sent with both the GET and POST methods.**

```
Welcome <?php echo $_REQUEST["name"]; ?>.<br />
You are <?php echo $_REQUEST["age"]; ?> years old!
```

## Server Side Includes

You can insert the content of a file into a PHP file before the server executes it, with the include() or require() function

The two functions are identical in every way, except how they handle errors.

The include() function generates a warning (but the script will continue execution) while the require() function generates a fatal error (and the script execution will stop after the error).

These two functions are used to create functions, headers, footers, or elements that can be reused on multiple pages

## The include() Function

- The include() function takes all the text in a specified file and copies it into the file that uses the include function.

Example

```
<html>
<body>
<?php include("header.php"); ?>
<h1>Welcome to my home page</h1>
<p>Some text</p>
</body>
</html>
```

Menu.php

```
<html>
<body>
<a href="default.php">Home</a> |
<a href="about.php">About Us</a> |
<a href="contact.php">Contact Us</a>
</body>
</html>
```

```
<?php include("menu.php"); ?>
<h1>Welcome to my home page</h1>
<p>Some text</p>
</body>
</html>
```

## The require() Function

- The require() function is identical to include(), except that it handles errors differently.
- The include() function generates a warning (but the script will continue execution) while the require() function generates a fatal error (and the script execution will stop after the error).
-

## Opening a File

- The fopen() function is used to open files in PHP.
- The first parameter of this function contains the name of the file to be opened and the second parameter specifies in which mode the file should be opened:

| **Modes** | **Description** |
|---|---|
| r | Read only. Starts at the beginning of the file |
| r+ | Read/Write. Starts at the beginning of the file |
| w | Write only. Opens and clears the contents of file; or creates a new file if it doesn't exist |
| w+ | Read/Write. Opens and clears the contents of file; or creates a new file if it doesn't exist |
| a | Append. Opens and writes to the end of the file or creates a new file if it doesn't exist |
| a+ | Read/Append. Preserves file content by writing to the end of the file |
| x | Write only. Creates a new file. Returns FALSE and an error if file already exists |
| x+ | Read/Write. Creates a new file. Returns FALSE and an error if file already exists |

## Reading a File Line by Line

Example-43

```
<?php
$file = fopen("Cinderella .txt", "r") or exit("Unable to open file!");
//Output a line of the file until the end is reached
while(!feof($file))
  {
  echo fgets($file). "<br />";
  }
fclose($file);
?>
```

## Create an Upload-File Form

To allow users to upload files from a form can be very useful.

Example 40

```
<html>
<body>
<form action="upload_file.php" method="post"
enctype="multipart/form-data">
 Filename:
<input type="file" name="file" id="file" />
<br />
<input type="submit" name="submit" value="Submit" />
</form>
</body>
</html>
```

- • The enctype attribute of the <form> tag specifies which content-type to use when submitting the form. "multipart/form-data" is used when a form requires binary data, like the contents of a file, to be uploaded
- • The type="file" attribute of the <input> tag specifies that the input should be processed as a file. For example, when viewed in a browser, there will be a browse-button next to the input field

## upload_file.php

```
<?php
if ($_FILES["file"]["error"] > 0)
  {
  echo "Error: " . $_FILES["file"]["error"] . "<br />";
  }
else
  {
  echo "Upload: " . $_FILES["file"]["name"] . "<br />";
  echo "Type: " . $_FILES["file"]["type"] . "<br />";
  echo "Size: " . ($_FILES["file"]["size"] / 1024) . " Kb<br />";
  echo "Stored in: " . $_FILES["file"]["tmp_name"];
  }
?>
```

By using the global PHP $_FILES array you can upload files from a client computer to the remote server.

The first parameter is the form's input name and the second index can be either "name", "type", "size", "tmp_name" or "error". Like this:

- $_FILES["file"]["name"] - the name of the uploaded file
- $_FILES["file"]["type"] - the type of the uploaded file
- $_FILES["file"]["size"] - the size in bytes of the uploaded file
- $_FILES["file"]["tmp_name"] - the name of the temporary copy of the file stored on the server
- $_FILES["file"]["error"] - the error code resulting from the file upload

## Saving the Uploaded File

The temporary copied files disappears when the script ends. To store the uploaded file we need to copy it to a different location:

Example Program

- **move_uploaded_file**
- This function checks to ensure that the file designated by *filename* is a valid upload file (meaning that it was uploaded via PHP's HTTP POST upload mechanism). If the file is valid, it will be moved to the filename given by *destination*.
-
- *filename* The filename of the uploaded file.
- *destination* The destination of the moved file.

## What is a Cookie?

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

## How to Create a Cookie?

The setcookie() function is used to set a cookie.

Note: The setcookie() function must appear BEFORE the <html> tag.

Syntax

setcookie(name, value, expire, path, domain);

```
<?php
setcookie("user", “UC  College  ", time()+3600);
?>
```

## How to Retrieve a Cookie Value?

- The PHP $_COOKIE variable is used to retrieve a cookie value.

- In the example below, we retrieve the value of the cookie named "user" and display it on a page:

```
<?php
// Print a cookie
echo $_COOKIE["user"];
// A way to view all cookies
print_r($_COOKIE);
?>
```

## isset() function

- In the following example we use the isset() function to find out if a cookie has been set:

```
<html>
<body>
<?php
if (isset($_COOKIE["user"]))
  echo "Welcome " . $_COOKIE["user"] . "!<br />";
else
  echo "Welcome guest!<br />";
?>
</body>
</html>
```

## PHP Error Handling

- When creating scripts and web applications, error handling is an important part. If your code lacks error checking code, your program may look very unprofessional and you may be open to security risks.

## Basic Error Handling: Using the die() function

This language construct is equivalent to **exit()**.

```
<?php
$file=fopen("welcome.txt","r");
?>
```

If the file does not exist you might get an error :

```
<?php
if(!file_exists("welcome.txt"))
 {
 die("File not found");
 }
else
 {
 $file=fopen("welcome.txt","r");
 }
?>
```

# PHP Filter?

- A PHP filter is used to validate and filter data coming from insecure sources.

**Why use a Filter?**
**Almost all web applications depend on external input. Usually this comes from a user or another application (like a web service). By using filters you can be sure your application gets the correct input type.**

**What is external data?**

- **Input data from a form**
- **Cookies**
- **Web services data**
- **Server variables**
- **Database query results**

## Functions and Filters

- To filter a variable, use one of the following filter functions:
- • filter_var() - Filters a single variable with a specified filter
- • filter_var_array() - Filter several variables with the same or different filters
- • filter_input - Get one input variable and filter it
- • filter_input_array - Get several input variables and filter them with the same or different filters

## validate an integer using the filter_var() function

```
<?php
$int = 123;
if(!filter_var($int, FILTER_VALIDATE_INT))
 {
 echo("Integer is not valid");
 }
else
 {
 echo("Integer is valid");
 }
?>
```

## Example - Validate an Email Id

```
<?php
$a = 'tiji@macfast.org';
if (  filter_var('$a', FILTER_VALIDATE_EMAIL))
{
              echo "Email is valid";
}
else
{
  echo "Email is not valid";
 }
 var_dump($a);

?>
```

Var_dump - This function displays structured information about one or more expressions that includes its type and value.

## Validating and Sanitizing

There are two kinds of filters:

Validating filters:

- Are used to validate user input
- Strict format rules (like URL or E-Mail validating)
- Returns the expected type on success or FALSE on failure

Sanitizing filters:

- Are used to allow or disallow specified characters in a string
- No data format rules
- Always return the string

```
<?php
  $no = 12.5;

  var_dump(filter_var($no, FILTER_SANITIZE_NUMBER_INT));
 ?>
```

## Validate Input

filter_input() function is using for validate input

```
<form action="email.php" method="get">
  Enter Email
    <input name="email" type="text" />
    <input type="submit" name="Submit" value="Submit">
</form>
```

## Email.php

```
<?php

 if (!filter_input(INPUT_GET, "email", FILTER_VALIDATE_EMAIL))
  {
  echo "E-Mail is not valid";
  }
 else
  {
  echo "E-Mail is valid";
  }
?>
```

## What is MySQL?

- MySQL is a database server
- MySQL is ideal for both small and large applications
- MySQL supports standard SQL
- MySQL compiles on a number of platforms
- MySQL is free to download and use
- PHP combined with MySQL are cross-platform (you can develop in Windows and serve on a Unix platform)

## Basic commands on mysql

1. CREATE DATABASE;
2. SHOW DATABASES;
3. USE   database name;
4. CREATE TABLE
5. SHOW TABLES;
6. DESCRIBE table name;
7. DROP TABLE  table name;
8. DROP DATABASE database name;
9. ALTER TABLE table name

11. SELECT

12 . DELETE FROM table name;

13 .TRUNCATE [TABLE] *tbl_name*

## Connection to a MySQL Database

- Before you can access data in a database, you must create a connection to the database.
- In PHP, this is done with the mysql_connect() function.

**Syntax**
**mysql_connect(servername,username,password);**

| **Parameter** | **Description** |
|---|---|
| servername | Specifies the server to connect to. Default value is "localhost" |
| username | Specifies the username to log in with. Default value is the name of the user that owns the server process |
| password | Specifies the password to log in with. Default is "" |

## Example- MySQL Connection

```
<?php
$con = mysql_connect("localhost","root","");
if (!$con)
  {
  die('Could not connect: ' . mysql_error());
  }
// some code
mysql_close($con);
?>
```

## Example – Create Database

```
<?php
$con = mysql_connect("localhost","root","");
if (!$con)
  {
  die('Could not connect: ' . mysql_error());
  }
if (mysql_query("CREATE DATABASE pmg",$con))
  {
  echo "Database created";
  }
else
  {
  echo "Error creating database: " . mysql_error();
  }
mysql_close($con);
?>
```

mysql_query() sends an unique query to the currently active database on the server

## Example - \macfast\php\phps3\example

**mysql_select_db**

- **Select a MySQL database**
- Sets the current active database on the server
- Every subsequent call to **mysql_query()** will be made on the active database.

## Insert Data Into a Database Table

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

mysql_query("INSERT INTO student (Rollno, Name, Mark)
VALUES (1, 'Raju', 35 )", $con);
mysql_close($con);
?>
```

## Insert Data From a Form Into a Database

```
<html>
<body>
<form action="insert.php" method="post">
Roll No: <input type="text" name="rollNo" />
Name : <input type="text" name="name" />
Mark : <input type="text" name="mark" />
<input type="submit" />
</form>
</body>
</html>
```

Insert.php

```
<?php
$con = mysql_connect("localhost","root","");
if (!$con)
  {
  die('Could not connect: ' . mysql_error());
  }
mysql_select_db(“pmg", $con);
$sql="INSERT INTO student (rollno, name, mark)
VALUES
('$_POST[rollno]','$_POST[name]','$_POST[mark]')";
if (!mysql_query($sql,$con))
  {
  die('Error: ' . mysql_error());
  }
  print_r($_POST);
echo "1 record added";

mysql_close($con)
?>
```

## Select Data From a Database Table

```
<?php
$con = mysql_connect("localhost","root","");
if (!$con)
  {
  die('Could not connect: ' . mysql_error());
  }
mysql_select_db("mca2007", $con);
$result = mysql_query("SELECT * FROM Student");

while($row = mysql_fetch_array($result))
  {
  echo $row['Rollno'] . " " . $row['Name']. " " . $row['Mark'];
  echo "<br />";
  }
mysql_close($con);
?>
```

# Display the Result in an HTML Table

- Example Program

**Where clause** Example Program

**Update** Example Program

**Delete**

*THANK YOU*