Unit-4
PHP
Hypertext Pre-processor

Introduction of PHP

- PHP is a server scripting language, and a powerful tool for making dynamic and interactive Web pages.

- Syntax:

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

Introduction of PHP

- **WAMP**

Windows Apache MySQL and PHP

XAMPP

X(cross Platform) Apache Mysql PHP Perl

- **LAMP**

Linux Apache MySQL and PHP

What is PHP?

- PHP is an acronym for

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

What is PHP?

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

What Can PHP Do?

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

Why PHP?

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

Beyond Syllabus

List of PHP framework	
LaraVel	Yii
Symfony	Fatfree
CodeIgniter	Kohana
CakePHP	FuelPHP
Zend	Flight
Phalcon	PHP-mini
Slim	Tyler
	Zikula

Why Framework?

- Framework abstracts you from low level details, makes you more productive, and protects you from low level error (such as preventing SQL injection attacks).

- A framework also **adds structure to the code**, prompting the developer to write **better, more readable, and more maintainable code**. Ultimately, **a framework makes programming easier**, since it packages complex operations into simple statements.

Definition of Framework

- In computer systems, a framework is often a layered structure indicating what kind of programs can or should be built and how they would interrelate.

Laravel

- Laravel. Although Laravel is a relatively new PHP framework (it was released in 2011), according to Sitepoint's recent online survey it is the most popular framework among developers. ...
- Latest Version: Laravel 5.4
- Laravel 4.2 requires PHP 5.4 or greater.

Basic Example of PHP

```
<html>
<head></head>
<body>
<?php
        echo “Hello I am hiren mer”;

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

PHP Comments

```
<?php
  // This is a single-line comment

  # This is also a single-line comment

  /*
  This is a multiple-lines comment block
  that lines
  */
?>
```

PHP Variables

- In PHP variable name is case sensitive.
- Variable name should be start with $ symbol.
- Variable name datatype declaration is as same as Javascript.

- $a=10; // Number
- $b=10.5; // Number
- $c="hiren"; // String

Global Variable

- The global keyword is used to access a global variable from within a function.

```
<?php
        $x = 5;
        $y = 10;
        function myTest()
         {
                global $x, $y;
                $y = $x + $y;
                echo $y;
        }
?>
```

Static Variable

- Normally, when a function is completed/executed, all of its variables are deleted. However, sometimes we want a local variable NOT to be deleted. We need it for a further job.
-
```
<?php
function myTest() {
    static $x = 0;
    echo $x;
    $x++;
}
myTest();
myTest();
?>
```

PHP echo and Print

- In PHP there are two basic ways to get output: echo and print.
- echo has no return value while print has a return value of 1 so it can be used in expressions.
- echo can take multiple parameters (although such usage is rare) while print can take one argument.
- echo is marginally faster than print.

The PHP echo Statement

- The echo statement can be used with or without parentheses: echo or echo().

```
<?php
$a=2;
echo "This ", "string ", "was ", "made ", "with multiple parameters.";
echo $a;
?>
```

The PHP print Statement

- The print statement can be used with or without parentheses: print or print().

```
<?php
print "<h1>Hello</h1>";
print "Hello world!<br>";
print "I'm about to learn PHP!";
?>
```

The PHP print_r() Statement

```
<?php   $var = array(1,2,3,4);
                print_r($var);                    ?>
```

Output:

```
Array ( [0] => 1
        [1] => 2
        [2] => 3
        [3] => 4 )
```

```
<?php
    $var1='abc';
    $result = print_r($var1);
    echo $result.'<br>';
    $abc = array(‘a'=>‘a',‘b'=>‘b',‘c'=>‘c',‘d'=>array(5,6,7,8));
    $result = print_r($abc);
    echo $result.'<br>';
?>
```

Output

```
abc
Array ( [a] => a [b] => b [c] =>c [] => Array ( [0] => 5 [1]
=> 6 [2] => 7 [3] => 8 ) )
```

PHP Data Types

- PHP supports the following data types:

1. String
2. Integer
3. Float (floating point numbers - also called double)
4. Boolean
5. Array
6. Object
7. NULL

PHP String

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

```
<?php
$x = "Hello world!";
echo $x;

?>
```

PHP Integer

- An integer data type is a non-decimal number between -2,147,483,648 and 2,147,483,647.
- Rules for integers:
  1. An integer must have at least one digit
  2. An integer must not have a decimal point
  3. An integer can be either positive or negative
  4. Integers can be specified in three formats: decimal (10-based), hexadecimal (16-based - prefixed with 0x) or octal (8-based - prefixed with 0)

PHP Integer

- ```
  <?php
  $x = 5985;
  var_dump($x);
  ?>
  ```
  ```

PHP Float

- A float (floating point number) is a number with a decimal point or a number in exponential form.

```
<?php
$x = 10.365;
var_dump($x);
?>
```

PHP Boolean

- A Boolean represents two possible states: TRUE or FALSE.

```
<?php
$x = TRUE;
var_dump($x);
?>
```

PHP Array

- An array stores multiple values in one single variable.

```
<?php
$c = array("Volvo","BMW","Toyota");
var_dump($c);
?>
```

PHP Object

- An object is a data type which stores data and information on how to process that data.
- In PHP, an object must be explicitly declared.
- First we must declare a class of object. For this, we use the class keyword. A class is a structure that can contain properties and methods:

PHP Object

```
<?php
class f
{
    function do_funny()
    {
        echo "Doing funny.";
    }
}

$b = new f;
$b->do_funny();
?>
```

PHP NULL Value

- Null is a special data type which can have only one value: NULL.

```
<?php
$x = null;
var_dump($x);
?>
```

PHP if...else...elseif Statements

- In PHP we have the following conditional statements:

1. **if statement** - executes some code if one condition is true
2. **if...else statement** - executes some code if a condition is true and another code if that condition is false
3. **if...elseif....else statement** - executes different codes for more than two conditions
4. **switch statement** - selects one of many blocks of code to be executed

PHP - The if Statement

- Syntax

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

Example

```
<?php
$t = 20;

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

PHP - The if...else Statement

- Syntax

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

Example

```
<?php
$t = 25;

if ($t < 20) {
    echo “it’s less than 20";
} else {
    echo " it’s greater than 20 ";
}
?>
```

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

- Syntax

```
if (condition)
{
   statement-1;
}
elseif (condition) {
    statement-2;
}
else {
   statement-3;
}
```

Example

```
<?php
$t = 30;
if ($t < 10) {
    echo "Have a good morning!";
} elseif ($t < 20) {
    echo "Have a good day!";
} else {
    echo "Have a good night!";
}
?>
```

PHP switch Statement

Syntax:

```
switch (n) {
    case label1:
        code to be executed if n=label1;
        break;
    case label2:
        code to be executed if n=label2;
        break;
    case label3:
        code to be executed if n=label3;
        break;
    ...
    default:
        code to be executed if n is different from all labels;
}
```

Example

```
<?php
$favcolor = "red";

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

PHP Loops

- In PHP, we have the following looping statements:

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

The PHP while Loop

Syntax

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

Example

```
<?php
   $x = 1;

   while($x <= 5) {
      echo "The number is: $x <br>";
      $x++;
   }
   ?>
```

The PHP do...while Loop

- Syntax

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

Example

```
<?php
$x = 1;

do {
    echo "The number is: $x <br>";
    $x++;
} while ($x <= 5);

?>
```

The PHP for Loop

- Syntax

for (*initialization; test condition; incr/decr*)

{

code to be executed;

}

Example

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

The PHP foreach Loop

- The foreach loop works only on arrays, and is used to loop through each key/value pair in an array.
- Syntax

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

Example

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

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

?>
```

PHP Functions

- Here, we can create our own functions.
- A function is a block of statements that can be used repeatedly in a program.
- A function will not execute immediately when a page loads.
- A function will be executed by a call to the function.

PHP Functions

- Syntax

  function *functionName*()

  {

  code to be executed;

  }

Example

Example:

```
<?php
   function writeMsg() {
       echo "Hello world!";
   }

   writeMsg(); // call the function
   ?>
```

PHP Function Arguments

```
<?php
function name($fname)
{
   echo “Hello $fname<br>";
}
name(“hiren");
name(“hemansir");
?>
```

PHP Functions - Returning values

```
<?php
   function sum($x, $y)
   {
      $z = $x + $y;
      return $z;
   }
   echo sum(5, 10) ;

   ?>
```

PHP Arrays

- In PHP, there are three types of arrays:

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

PHP Indexed Arrays

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

  $name = array(“hiren", “rahul", “heman");

- And the index can be assigned manually:

  $name[0] = “2nd sem";
  $name[1] = “4th sem ";
  $name[2] = " 6th sem ";

Loop with an Indexed Array

- ```
  <?php
  $x = array(“a", “b", “c");
  $len = count($x);

  for($i = 0; $i < $len; $i++) {
      echo $x[$i];
      echo "<br>";
  }
  ?>
  ```
  ```

PHP Associative Arrays

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

```
$age = array(“a"=>"30", “b"=>“40", “c"=>“50");
```

or:

```
$age[‘a'] = "30";
$age[‘b'] = “40";
$age[‘c'] = “50";
```

Example

- <?php
  $age
  = array(“a"=>"35", “b"=>"37", “c"=>"43");

  echo “a is " . $age[‘a'] . " years old.";
  ?>

Loop with an Associative Array

- <?php
  $age
  = array(“a"=>"35", “b"=>"37", “c"=>"43");

  foreach($age as $x => $value)
  {
  echo "Key=" . $x . ", Value=" . $value;
  echo "
";
  }
  ?>

Multidimentional Array

- $c = array (

  array(“a",22),

  array(“b",24)

  );

- <?php
  echo ”name”.$c[0][0].“age".$c[0][1]."
";
  echo ”name”.$c[1][0].“age".$c[1][1]."
";
  ?>

Browser Control

- PHP can control various features of browser.This is important as need to reload and redirect the page.

  header(“Location:pagename”)

Browser Control

- It is also possible to reload current page.

```
$x=$_SERVER['PHP_SELF'];
<form action="<?php echo $x?>"
   method="get">

</form>
```

PHP STRING Function

1. strlen($var):

this function returns length of string.

Example:

```
<?php
echo strlen(“Hello”);
?>
```

Output:

5

2. str_word_count()

This function returns the word count of the string.

Example:

```
<?php
echo str_word_count(“I am hiren”);
?>
```

3.strrev($var)

This function reverse given string.

Example:

```
echo strrev(“hirenmer”);
```

4.strpos($var,$search)

The PHP strpos() function searches for a specific text within a string. If a match is found, the function returns the character position of the first match. If no match is found, it will return FALSE.

Example:

strpos(“hiren mer”,”mer”);

Output is 6. because string start with 0.

5.str_replace($s1,$s2,$s3)

This function replace the string in given string.$s1 to be replace in given string and $s2 replace in that string $s3.

Example:

str_replace(“hi”, ”good morning all”, ”hello hi”);

In this example “hello hi” is given string and hi to be replaced with good. So output of this function is “hello good morning all” .

6.str_pad(*string,length,pad_string,pad_type*)

The str_pad() function pads a string to a new length.

Example:

```
<?php
   $str = "Hello World";
   echo str_pad($str,20,".");
?>
```

In above example hello world length is 12 and remaining 8 dots are padding by default right.

Output: Hello World……..

- Possible values:
- STR_PAD_BOTH - Pad to both sides of the string. If not an even number, the right side gets the extra padding
- STR_PAD_LEFT - Pad to the left side of the string
- STR_PAD_RIGHT - Pad to the right side of the string. This is default

Example

```
<?php
   $str = "Hello";
   echo str_pad($str,10,“$“, STR_PAD_LEFT);
?>
```

7. str_shuffle($var)

This function shuffle the position of the each and every character.

Example: str_shuffle(“hirenmer”);

Output will be : rimherne

8. str_split($str,length)

- The str_split() function splits a string into an array.
- Example : str_split(“hello”,2)

Output

Array ([0] => He [1] => ll [2] => o)

9. strchr($str,$match)

This function find the first occurrence of string inside given string and return the rest of the string.

Example:

```
echo strchr("hiren mer!","re");
```

Output:

ren mer!

10. strcmp(*string1,string2*)

This function match string 1 and string2.

- Example

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

Output : -6

- Return Value:This function returns:

0 - if the two strings are equal

<0 - if string1 is less than string2

>0 - if string1 is greater than string2

11. strtolower($str)

This function convert uppercase to lower case.

Example: echo strtolower(“HIREN");

Output: hiren

12. strtoupper($str)

The strtoupper() function converts a string to uppercase.

Example : Example: echo strtoupper(“hiren");

Output: HIREN

13. substr($var,value)

This function returns the string from value index.

Example:

echo substr("hirenmer",3);

Output:

Enmer

note:index start from 0

Substr($var,start,end)

When we want substring start from and to then we can use above function.

Example:

echo substr(“hirenmer",3,3);

Output:

enm

14. stripslashes(*string*)

This function strip the slashes \ in the string which we have used to escape symbol.

- Example

```
echo stripslashes("Who\'s katappa?");
```

Output:

Who's katappa?

15. strstr($s1,$s2)

This function search $s2 in $s1 and return the portion of $s1.this function is same as **strchr($str,$match).**

Example:

strstr(“hirenmer123”,”nm”)

Output: nmer123

16. strtok($s1,$d)

This functions plits a string (str) into smaller strings (tokens), with each token being delimited by any character from token.

```
<?php
$string = "Hello world. Beautiful day today.";
 $token = strtok($string, ".");
while ($token !== false)
  {
  echo "$token<br>";
  $token = strtok(".");
  }
?>
```

17. trim($s1,$s2)

This function Remove characters/space from both sides of a string .

```
<?php
$str = “6th BYBX”;
echo trim($str,“6");
?>
```

18.ltrim($str,$s1)

This function Remove characters/space from left side of a string .

19.rtrim($str,$s1)

This function Remove characters/space from right side of a string .

20.ucwords($str)

This function retruns value with first letter Capital in each word.

ucwords(“salman khan”);

Output: Salman Khan

21. strip_tags($str,”allowable tag”)

Example:

$str=“hiren mer”;

strip_tags($str,””);

Output: *hiren mer*

PHP FILES

- Read a file using PHP:

readfile(filename)

This function show the file in browser with number of characters in file.

- Example

```
echo readfile("myfile.txt");
```

- Open read and close a file using PHP:

$var=fopen(filename,mode)

This function open the file name in different mode.

fread($var,filesize(filename))

This function read the file content as per the second parameter.

fclose($var)

This function close the file which already opened by fopen.

- Example:

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

Mode	Use
r	Read mode
w	Write mode create new file
a	Append mode and also create new file
x	**Creates a new file for write only**. Returns FALSE and an error if file already exists

fgets($ptr)

- This function read the first line of the file.
- Example

```
<?php
$myfile = fopen(“abc.txt", "r") or die("Unable to
open file!");
echo fgets($myfile);
fclose($myfile);
?>
```

feof()

- The feof() function checks if the "end-of-file" (EOF) has been reached.
- The feof() function is useful for looping through data of unknown length.

- Example

```
<?php
$myfile = fopen(“abc.txt", "r") or die("Unable
to open file!");
while(!feof($myfile))
{
  echo fgets($myfile) . "<br>";
}
fclose($myfile);
?>
```

fgetc()

- The fgetc() function is used to read a single character from a file.
- Example

```
<?php
$myfile = fopen("myfile.txt", "r") or die("Unable to
    open file!");
while(!feof($myfile))
{
  echo fgetc($myfile);
}
fclose($myfile);
?>
```

fwrite()

- The fwrite() function is used to write to a file.

Example

```
<?php
   $myfile = fopen(“myfile.txt", "w") or die("Unable to open
   file!");
   $txt = “hello 6th semester\n";
   fwrite($myfile, $txt);
   fclose($myfile);
   ?>
```

File Upload

- We can upload the file using php page with html input type file.
- First we have to configure php.ini file for the file upload.
- Open php.ini file and make file upload ON.

Rule for File upload

- Some rules to follow for the HTML form above:

1. Make sure that the form uses method="post"
2. The form also needs the following attribute: enctype="multipart/form-data". It specifies which content-type to use when submitting the form.
3. Without the requirements above, the file upload will not work.

File Contents

- $_FILES[‘file’][‘name’] this returns the name of the uploaded file.
- $_FILES[‘file’][‘type’] this returns the MIME type of the uploaded file.
- $_FILES[‘file’][‘size’] this returns the size of the uploaded file in Byte.
- $_FILES[‘file’][‘tmp_name’] returns the temporary name for the uploaded file.
- $_FILES['userfile']['error'] The error code associated with this file upload.

To upload file on server

- First we have to make one folder upload into our directory.
- Then store the path into variable.
- Then apply the function below to move the file.

move_uploaded_file($tmp_name,path)

To upload file on server

- $_FILES['file']['tmp_name']; will contain the temporary file name of the file on the server. This is just a placeholder on your server until you process the file.

PHP Cookies

- A cookie is used to identify the 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.

Create Cookie

- Syntax

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

Only the *name* parameter is required. All other parameters are optional.

Example

Setcookie(“user”,”hiren”,time()+3600);

Type	Description
Name	Name is used to set the name of user
Value	Value for the cookie value
Expire	Set expire time like time()+3600 is for 1hour
Path	Set the path on server for the cookie storage. If set to '/', the cookie will be available within the entire domain.
Domain	The (sub)domain that the cookie is available to. Setting this to a subdomain (such as *'www.example.com'*) will make the cookie available to that subdomain and all other sub-domains of it (i.e. w2.www.example.com). To make the cookie available to the whole domain (including all subdomains of it), simply set the value to the domain name (*'example.com'*, in this case).
Secure	Whether cookie use HTTPS connection or not. Return true(1) or false(0).

Delete cookie

```
setcookie("user", "", time() - 3600);
```

Session

- A session is a way to store information by Variable to be used across multiple pages.
- Unlike a cookie, the information is not stored on the users computer.
- Session variables solve this problem by storing user information to be used across multiple pages (e.g. username, favorite color, etc). By default, session variables last until the user closes the browser.

Start PHP Session

Session_start()

Above statement is used to start the session.after starting session we can store the value into session variable.

$_SESSION[‘a’]=10;

Access Session Variable value at another page

- When we want to access the session variable value at another page,we have to start the session first then we can access the session variable values.

```
Session_start();
Echo $_SESSION[‘a’];
```

Output:

10

Destroy a PHP Session

- If user want to destroy the session then he can use below function,

session_unset();

Or

Session_destroy();

- Generaly we can use this method in the login and logout pages.

Sending Email

- PHP must be configured correctly in the **php.ini** file with the details of how your system sends email.

- Windows users should ensure that two directives are supplied. The first is called SMTP that defines your email server address. The second is called sendmail_from which defines your own email address.

How to set php.ini for Email

[mail function]

;For Win32 only.

SMTP = smtp.atmiya.net

;For win32 only

sendmail_from =hiren@aits.edu.in

Syntax

mail(to, subject, message, headers, parameters);

To: Required. Specifies the receiver / receivers of the email.

Subject : Required. Specifies the subject of the email. This parameter can not contain any newline characters

Message: Required. Defines the message to be sent. Each line should be separated with a LF (\n). Lines should not exceed 70 characters

Conti.

Headers: Optional. Specifies additional headers, like From, Cc, and Bcc. The additional headers should be separated with a CRLF (\r\n)

Parameters: Optional. Specifies an additional parameter to the send mail program

```php
<?php
$to = "somebody@example.com";
$subject = "My subject";
$txt = "Hello world!";
$headers = "From:
webmaster@example.com" . "\r\n" .
"CC: somebodyelse@example.com";

mail($to,$subject,$txt,$headers);
?>
```

PHP Timestamp

- We can store the php timestamp using time() function.

Example:

```
$time=time();
echo $time;
```

Output:

1490549125

This output in seconds from 1st jan 1970.

PHP Timestamp

- We can print date using this function.

```
$t=time();
echo date(‘h:i:s ’.$t); // 10:25:55
echo date('D M Y',$t); //Sun Mar 2017
echo date(‘d m y',$t); //  26 03 17
```

Advanced PHP
(OOPS concepts)

Definitions

- **Class** − This is a programmer-defined data type, which includes local functions as well as local data. You can think of a class as a template for making many instances of the same kind (or class) of object.
- **Object** − An individual instance of the data structure defined by a class. You define a class once and then make many objects that belong to it. Objects are also known as instance.
- **Member Variable** − These are the variables defined inside a class. This data will be invisible to the outside of the class and can be accessed via member functions. These variables are called attribute of the object once an object is created.

- **Member function** − These are the function defined inside a class and are used to access object data.
- **Inheritance** − When a class is defined by inheriting existing function of a parent class then it is called inheritance. Here child class will inherit all or few member functions and variables of a parent class.
- **Parent class** − A class that is inherited from by another class. This is also called a base class or super class.
- **Child Class** − A class that inherits from another class. This is also called a subclass or derived class.
- **Polymorphism** − This is an object oriented concept where same function can be used for different purposes. For example function name will remain same but it make take different number of arguments and can do different task.

- **Overloading** − a type of polymorphism in which some or all of operators have different implementations depending on the types of their arguments. Similarly functions can also be overloaded with different implementation.
- **Data Abstraction** − Any representation of data in which the implementation details are hidden (abstracted).
- **Encapsulation** − refers to a concept where we encapsulate all the data and member functions together to form an object.
- **Constructor** − refers to a special type of function which will be called automatically whenever there is an object formation from a class.
- **Destructor** − refers to a special type of function which will be called automatically whenever an object is deleted or goes out of scope.

OPPS in PHP

- A class implements the concept of ADT(abstract data type).
- It is a compilation of methods and objects.A class definition is always begin with keyword “class” followed by identifier which represents class name.
- And then it followed by { } curly braces which contains the members of class.

OPPS in PHP

Example:

```
<?php
    Class abc
    {
        public  $a;
            function getval()
            {
                        $this->a=10;
            }
            function showval()
            {
                        echo “you have entered”.$this->a;
            }
    }
    $myobj=new abc();
    $myobj->getval();
    $myobj->showval()
?>
```

Inheritance

```
<?php
        Class aaa
        {
            public $x;
            public $y;
            public function const()
            {
                $this->y="hello";
                $this->x=0;
            }
        }
        class bbb extends aaa
        {
            public $a;
            public function search()
            {
            $this->mode="search";
            }
        }
        $p=new aaa();
        print $p->const();
        print $p->search();

?>
```

Inheritance

- We can use different types of inheritance as same as C++.
- Like
- Multiple
- Multilevel

PHP & MySQL

Introduction of MYsql

- MySQL is a database system used on the web
- MySQL is a database system that runs on a server
- MySQL is ideal for both small and large applications
- MySQL compiles on a number of platforms
- MySQL is free to download and use.
- MySQL is developed, distributed, and supported by Oracle Corporation

Introduction of Mysql

- PHP combined with MySQL are cross-platform (you can develop in Windows and serve on a Unix platform)

Connection To Server

- We can connect to server using mysql command.

mysql_connect(“servername”,”username”,”password”);

MySQLi extension (the "i" stands for improved)

mysqli_connect(“servername”,”username”,”password”, "my_db");

Creating Database

- We can create database after connecting server.
- Example

```
mysql_connect(“localhost”,”root”,””);
$x=“CREATE DATABASE abc”;
mysql_query($x);
```

Selecting Database

After connecting server and after Creating database using PHP we can select the database using following code:

mysql_select_db(“dbname”)

Listing Database

- We can list the database names using mysql query.

```
<?php
$x=mysql_connect("localhost","root","");
$result=mysql_list_dbs($x);
while ($row = mysql_fetch_array($result)) {
   echo $row[0]."<br>";
}
?>
```

Listing Database Example 2

```
<?php
$x=mysql_connect("localhost","root","");
$result = mysql_query("SHOW DATABASES");
while ($row = mysql_fetch_array($result)) {
    echo $row[0]."<br>";
}
?>
```

Creating Mysql Table

- The CREATE TABLE statement is used to create a table in MySQL.syntax is same as Oracle DB.

```
CREATE TABLE tablename(
colname1 datatype(size),
colname2 datatype(size),
.
.
);
```

Example

```
<?php
mysql_connect(“localhost”,”root”,””);
mysql_Select_db(“dbname”);
$q1=“create table abc(eno INT(20),
Class varchar(20));”;
mysql_query($q);
?>
```

List tables

```
<?php
$x=mysql_connect("localhost","root","");
$db=mysql_select_db("test");
$result=mysql_query('SHOW TABLES');
 while ($row = mysql_fetch_array($result))
 {
    echo $row[0]."<br>";
}
?>
```

Inserting Data into Mysql Table

syntax rules :

1. The SQL query must be quoted in PHP
2. String values inside the SQL query must be quoted
3. Numeric values must not be quoted
4. The word NULL must not be quoted

Syntax

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

- Example:

  $sql = "INSERT INTO MyGuests (firstname, lastname, email) VALUES (‘MS', ‘dhoni', ‘abc@example.com')";

  Mysql_query($sql)

Altering Data into Table

- Syntax

  UPDATE table_name SET column1=value, column2=value2,…

  WHERE some_column=some_value

Example:

$sql = "UPDATE MyGuests SET lastname=‘raina'
  WHERE id=2";

mysql_query($sql);

Delete Data From a MySQL Table

- Syntax

  DELETE FROM table_name
  WHERE some_column = some_value;

Example:

```
$sql = "DELETE FROM MyGuests WHERE id=3";
mysql_query($sql);
```

PHP Select Data From MySQL

- Syntax

  SELECT column_name(s) FROM table_name;

Example :

```
$q= “SELECT * from abc”;
$res=mysql_query($q);
While($x=mysql_fetch_array($res))
{
        echo $x[0].”and ”.$x[1].”<br>”;
}
```

Datatypes in PHP

- Properly defining the fields in a table is important to the overall optimization of your database. You should use only the type and size of field you really need to use; don't define a field as 10 characters wide if you know you're only going to use 2 characters. These types of fields (or columns) are also referred to as data types, after the **type of data** you will be storing in those fields.
- MySQL uses many different data types broken into three categories: numeric, date and time, and string types.

Numeric Data Types:

- **INT** - A normal-sized integer that can be signed or unsigned. If signed, the allowable range is from -2147483648 to 2147483647. If unsigned, the allowable range is from 0 to 4294967295. **You can specify a width of up to 11 digits.**
- **TINYINT** - A very small integer that can be signed or unsigned. If signed, the allowable range is from -128 to 127. If unsigned, the allowable range is from 0 to 255. **You can specify a width of up to 4 digits.**
- **SMALLINT** - A small integer that can be signed or unsigned. If signed, the allowable range is from -32768 to 32767. If unsigned, the allowable range is from 0 to 65535. **You can specify a width of up to 5 digits.**
- **MEDIUMINT** - A medium-sized integer that can be signed or unsigned. If signed, the allowable range is from -8388608 to 8388607. If unsigned, the allowable range is from 0 to 16777215. **You can specify a width of up to 9 digits.**

- **BIGINT** - A large integer that can be signed or unsigned. If signed, the allowable range is from -9223372036854775808 to 9223372036854775807. If unsigned, the allowable range is from 0 to 18446744073709551615. **You can specify a width of up to 20 digits.**
- **FLOAT(M,D)** - A floating-point number that cannot be unsigned. You can define the display length (M) and the number of decimals (D). This is not required and will default to 10,2, where 2 is the number of decimals and 10 is the total number of digits (including decimals). **Decimal precision can go to 24 places for a FLOAT.**
- **DOUBLE(M,D)** - A double precision floating-point number that cannot be unsigned. You can define the display length (M) and the number of decimals (D). This is not required and will default to 16,4, where 4 is the number of decimals. **Decimal precision can go to 53 places for a DOUBLE. REAL is a synonym for DOUBLE.**
- **DECIMAL(M,D)** - An unpacked floating-point number that cannot be unsigned. In unpacked decimals, each decimal corresponds to one byte. Defining the display length (M) and the number of decimals (D) is required. **NUMERIC is a synonym for DECIMAL.**

Date and Time Types:

- **DATE** - A date in YYYY-MM-DD format, between 1000-01-01 and 9999-12-31. For example, December 30th, 1973 would be stored as 1973-12-30.
- **DATETIME** - A date and time combination in YYYY-MM-DD HH:MM:SS format, between 1000-01-01 00:00:00 and 9999-12-31 23:59:59. For example, 3:30 in the afternoon on December 30th, 1973 would be stored as 1973-12-30 15:30:00.
- **TIMESTAMP** - A timestamp between midnight, January 1, 1970 and sometime in 2037. This looks like the previous DATETIME format, only without the hyphens between numbers; 3:30 in the afternoon on December 30th, 1973 would be stored as 19731230153000 (YYYYMMDDHHMMSS).
- **TIME** - Stores the time in HH:MM:SS format.
- **YEAR(M)** - Stores a year in 2-digit or 4-digit format. If the length is specified as 2 (for example YEAR(2)), YEAR can be 1970 to 2069 (70 to 69). If the length is specified as 4, YEAR can be 1901 to 2155. The default length is 4.

String Types:

- **CHAR(M)** - A fixed-length string between 1 and 255 characters in length (for example CHAR(5)), right-padded with spaces to the specified length when stored. Defining a length is not required, but the default is 1.
- **VARCHAR(M)** - A variable-length string between 1 and 255 characters in length; for example VARCHAR(25). You must define a length when creating a VARCHAR field.
- **BLOB or TEXT** - A field with a maximum length of 65535 characters. BLOBs are "Binary Large Objects" and are used to store large amounts of binary data, such as images or other types of files. Fields defined as TEXT also hold large amounts of data; the difference between the two is that sorts and comparisons on stored data are case sensitive on BLOBs and are not case sensitive in TEXT fields. You do not specify a length with BLOB or TEXT.

- **TINYBLOB or TINYTEXT** - A BLOB or TEXT column with a maximum length of 255 characters. You do not specify a length with TINYBLOB or TINYTEXT.
- **MEDIUMBLOB or MEDIUMTEXT** - A BLOB or TEXT column with a maximum length of 16777215 characters. You do not specify a length with MEDIUMBLOB or MEDIUMTEXT.
- **LONGBLOB or LONGTEXT** - A BLOB or TEXT column with a maximum length of 4294967295 characters. You do not specify a length with LONGBLOB or LONGTEXT.
- **ENUM** - An enumeration, which is a fancy term for list. When defining an ENUM, you are creating a list of items from which the value must be selected (or it can be NULL). For example, if you wanted your field to contain "A" or "B" or "C", you would define your ENUM as ENUM ('A', 'B', 'C') and only those values (or NULL) could ever populate that field.

Thank you

- I wish you all the best for the Exam.