PHP

PHP

PHP is an HTML-embedded scripting language. Much of its syntax is borrowed from C, Java and Perl with a couple of unique PHP-specific features thrown in. The goal of the language is to allow web developers to write dynamically generated pages quickly.

PHP is a powerful, behind the scenes scripting language. When someone visits your PHP webpage, your web server processes the PHP code. It then sees which parts it needs to show to visitors (content and pictures) and hides the other stuff (file operations, math calculations, etc.) then translates your PHP into HTML. After the translation into HTML, it sends the webpage to your visitor's web browser.

Advantages of PHP

- Reduce the time to create large websites.
- Create a customized user experience for visitors based on information that you have gathered from them.
- Allow creation of shopping carts for e-commerce websites.

PHP Syntax

The rules that must be followed to write properly structured code.PHP's syntax and semantics are similar to most other programming languages (C, Java, Perl) with the addition that all PHP code is contained with a tag, of sorts. All PHP code must be contained within the following.

Standard Form

```
<?php
?>
```

Short Hand form

```
<?
?>
```

If PHP scripts are to be distributed them, use the standard form, rather than the shorthand form. This will ensure that your scripts will work, even when running on other servers with different settings.

Simple PHP program- Embedding PHP script in HTML

```
<html>
<head>
<title>My First PHP Page</title>
</head>
<body>
<?php
    echo "First PHP Script!";
?>
</body>
</html>
```

This file is saved with php extension and placed on a PHP enabled server and loaded in web browser, then **First PHP Script!** will be displayed. The PHP function echo is used to write on the browser window.
As with HTML, whitespace is ignored between PHP statements. This means it is OK to have one line of PHP code, then 20 lines of blank space before the next line of PHP code. The PHP interpreter will ignore white spaces as well.

PHP - Variables
A variable is a means of storing a value, such as text string "Hello World!" or the integer value 4. A variable can then be reused throughout php code, instead of having to type out the actual value over and over again. In PHP a variable with the following form:
$variable_name = Value;
In PHP a variable should begin with $.If dollar sign is missing then, it will not work. This is a common mistake for new PHP programmers.

PHP Variable Naming Conventions
- There are a few rules that you need to follow when choosing a name for your PHP variables.
- PHP variables must start with a letter or underscore "_".
- PHP variables may only be comprised of alpha-numeric characters and underscores. a-z, A-Z, 0-9, or _ .
- Variables with more than one word should be separated with underscores. $my_variable
- Variables with more than one word can also be distinguished with capitalization. $myVariable .

PHP Code using variables for storing values

```
<?php
$a = 5;
$b = "5";
echo "The value of variables are :";//quoted string
echo $a; // variables
echo $b  // variables
?>
```

PHP - echo
The PHP function *echo* is a used for outputting text to the web browser.PHP echo function can display variable value and string value that is quoted. [see above code]
Since echo function is used to display contents on browser, php code can be intermixed with HTML code.

```
<?php
$a = 5;
$b = "5";
echo "The value of variables are :<br/>";//quoted
string
```

```
echo $a; // variables
echo $b  // variables
?>
```

Quoted string with quotes

Care should be taken when HTML code or any other string that includes quotes is echoed. The echo function uses quotes to define the beginning and end of the string, so do the following in case string to be displayed has quotes.

1. Don't use quotes inside your string
2. Escape quotes that are within the string with a slash. To escape a quote a slash is placed before the quotation mark, i.e. \".
3. Use single quotes (apostrophes) for quotes inside your string.

```
<?php
echo "Amirtha's Day out";    //3
echo " \" Sunny Days\" ";   //2
echo "My Day out";          //1
?>
```

Displaying Special Characters

```
<?php
$newline = "A newline is \n";
echo $newline
$ret = "A carriage return is \r";
echo $ret
$tab = "A tab is \t";
echo $tab
$dollar = "A dollar sign is \$";
echo $dollar
$doublequote = "A double-quote is \"";
echo $doublequote
?>
```

**New line using
 of HTML in PHP**

```
<?php
$newline = "A newline is <br/>";
echo $newline
?>
```

PHP - String Creation Heredoc

PHP introduces a more robust string creation tool called heredoc that is used to create multi-line strings without using quotations. However, creating a string using heredoc is more difficult and can lead to problems if not properly coded.

```
<?php
$my_string = <<<TEST
School Of Information Technology
Vellore Institute Of Technology
Vellore
TEST;
echo $my_string;
?>
```

Few **very** important things to remember when using heredoc:

- Use <<< and some identifier that you choose to begin the heredoc. In the php code TEST is identifier.
- Repeat the identifier followed by a semicolon to end the heredoc string creation.
- The closing sequence TEST must occur on a line by itself and **cannot** be indented.

The output is

School Of Information Technology Vellore Institute Of Technology Vellore

***The string will not be printed in multiple lines as there is no \n or
**

PHP - Operators

In all programming languages, operators are used to manipulate or perform operations on variables and values. There are many operators used in PHP.

- Assignment Operators
- Arithmetic Operators
- Comparison Operators
- String Operators
- Combination Arithmetic & Assignment Operators

Assignment Operators

Assignment operators are used to set a variable equal to a value or set a variable to another variable's value. Such an assignment of value is done with the "=", or equal character. Example:

$my_var = 4;

$another_var = $my_var

Now both $my_var and $another_var contain the value 4. Assignments can also be used in conjunction with arithmetic operators.

Concatenation Operator- .

- It is used to concatenate strings.
- Equivalent to + operator of Java.

```
<?php
echo "PHP – Server"."Side Script"."\n";
$val_1 = 5;
$val_2 = 6;
echo " The value of value 1 is :".$val_1."\n";
echo $val_2;
```

```
?>
```

Arithmetic Operators

Operator	Example
+	2 + 4 or $a + $b or $a + 5
-	6 - 2 or $a - $b or $a - 5
*	5 * 3 or $a * $b or $a * 5
/	15 / 3 or $a / $b or $a / 5
%	43 % 10 or $a % $b or $a % 5

Comparison Operators

Comparisons are used to check the relationship between variables and/or values. Comparison operators are used inside conditional statements and evaluate to either true or false.**Assume: $x = 4 and $y = 5;**

Operator	Example	Result
==	$x == $y	false
!=	$x != $y	true
<	$x < $y	true
>	$x > $y	false
<=	$x <= $y	true
>=	$x >= $y	false

Combination Arithmetic & Assignment Operators

Arithmetic Operators can be combined with assignment operator

$counter = $counter + 1;

However, there is shorthand for doing this.

$counter += 1;

This combination assignment/arithmetic operator would accomplish the same task. The downside to this combination operator is that it reduces code readability to those programmers who are not used to such an operator. Here are some examples of other common shorthand operators. In general, "+=" and "-=" are the most widely used combination operators.

Operator	Example	Equivalent Operation
+=	$x += 2;	$x = $x + 2;
-=	$x -= 4;	$x = $x - 4;
*=	$x *= 3;	$x = $x * 3;
/=	$x /= 2;	$x = $x / 2;
%=	$x %= 5;	$x = $x % 5;
.=	$my_str.="hello";	$my_str = $my_str . "hello";

Pre/Post-Increment & Pre/Post-Decrement

To add one to a variable or "increment" use the "++" operator:
$x++; Which is equivalent to $x += 1; or $x = $x + 1;
To subtract 1 from a variable, or "decrement" use the "--" operator:
$x--; Which is equivalent to $x -= 1; or $x = $x - 1;

PHP Code:

```
<?php
$x = 4;
echo "The value of x with post-incr = " . $x++;
echo "<br /> The value of x after the post incr is " . $x;
$x = 4;
echo "<br />The value of x with with pre-incr = " . ++$x;
echo "<br /> The value of x after the pre-incr is " . $x;
?>
```

The value of x with post-incr = 4 The value of x after the post-incr is = 5 The value of x with with pre-incr = 5 The value of x after the pre-incr is = 5 .The value of $x++ is not reflected in the echoed text because the variable is not incremented until after the line of code is executed. However, with the pre-increment "++$x" the variable does reflect the addition immediately.

Comments in PHP
Comments in PHP are similar to comments that are used in HTML. The PHP comment syntax always begins with a special character sequence and all text that appears between the start of the comment and the end will be ignored by the browser.

HTML Code:
<!--- This is an HTML Comment -->

PHP Comment Syntax: Single Line Comment
PHP has two types. The first type we will discuss is the single line comment. The single line comment tells the interpreter to ignore everything that occurs on that line to the right of the comment. To do a single line comment type "//" and all text to the right will be ignored by PHP interpreter.

PHP Comment Syntax: Multi Line Comment
The multi-line PHP comment can be used to comment out large blocks of code or writing multiple line comments. The multiple line PHP comment begins with " /* " and ends with " */ ".

PHP Code:

```
<?php
/* Program used to display variable values & Strings */
$a = 5;
$b = "5";
echo "The value of variables are :";//quoted string
echo $a; // variables
echo $b  // variables
```

```
?>
```

The Include Function

The *include* function takes a file name and simply inserts that file's contents into the script that calls used the *include* function. First of all, this means that you can type up a common header or menu file that you want all your web pages to include. When you add a new page to your site, instead of having to update the links on several web pages, you can simply change the Menu file.

An Include Example

menu.php Code:

```
<html>
<body>
<a href="http://www.example.com/index.php">Home</a> -
<a href="http://www.example.com/about.php">About Us</a> -
<a href="http://www.example.com/links.php">Links</a> -
<a  href="http://www.example.com/contact.php">Contact  Us</a>
<br />
```

Save the above file as "menu.php". Now create a new file, "index.php" in the same directory as "menu.php". The include function to add a common menu.

index.php

```
<?php include("menu.php"); ?>
<p>This is my home page that uses a common menu to save me
time when I add new pages to my website!</p>
</body>
</html>
```

The follow code will display the following.

Home - About Us - Links - Contact Us

What does Include do?

The include command simply takes all the text that exists in the specified file and copies it into the file that uses the include function. Include is quite useful when you want to include the same PHP, HTML, or text segment on multiple pages of a website. The include function is used widely by PHP web developers.

PHP Require Function

The require function is used to include a file into your PHP code. However there is one huge difference between the two functions, though it might not seem that big of a deal.

Require vs Include

```
<?php
include(file.php");
echo "Hello World!";
```

```
?>
```

When you include a file with the *include* function and PHP cannot find it you will see an error message like

Warning: main(file.php): failed to open stream.no such File or directory.
Warning failed opening the file.
Hello World!

echo statement is still executed.

```
<?php
require(file.php");
echo "Hello World!";
?>
```

Warning: main(file.php): failed to open stream.File doesn't exist.
Fatal error-failed opening the required file.
echo statement is not executed.

Selective Control Structures

The PHP if statement, if else statement and if elseif statement, nested if statement and switch case are very similar to other programming languages control structures. Block of statements is enclosed within { }

Usage of if

```
<html>
<head>
<title>My First PHP Page</title>
</head>
<body>
<?php
$a= "VIT";
if ($a == "VIT")
   echo "Hello" . $a. "ians";
```

Usage of if else

```
<html>
<head>
<title>My First PHP Page</title>
</head>
<body>
<?php
$a=5;
$b="5";
if ($a === $b)
   echo "both are equal";
else
   echo "both are not equal";
?>
```

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

Usage of if elseif

```
<?php
$mark = 90;
if($mark > 74 && $mark < 101)
   echo "distinction";
elseif($mark > 59 && $mark < 75)
   echo "first class";
elseif($mark >49 && $mark < 60 )
   echo "second class";
else
  echo "Fail";
?>
```

Usage of switch case – switch variable can be checked against numeric value of case, char value of case and string value of case.

```
<?php
$dest = "Coimbatore";
echo "Traveling to $destination<br />";
switch ($destination){
case "Ooty":
echo "The Queen Of Mountains".$dest;
break;
case "Coimbatore":
echo "The Manchester of Tamil Nadu".$dest;
break;
case "Jaipur":
echo "The Pink City".$dest;
break;
case "Kodaikannal":
echo "Kashmir of South".$dest;
break;
case "Vellore":
echo "City of VIT".$dest;
break;
default:
echo "OOPS,I don't have information".$dest;
}
?>
```

Repetitive Control Structures

The repetitive control structures while, do while, for are same as other programming languages.

while

```
<?php
$a = Array(2,1,-89,0,5);
$ser= 2;
$f = 0;
$i = 0;
while ( $i < 5 )
{
   if ($ser == $a[ $i ])
   {
      $f = 1;
      break;
   }

   $i++;
}
if ($f == 1)
 echo "element found";
else
 echo "element not found";

?>
```

do while

```
<?php
$a = Array(2,1,-89,0,5);
$ser= -2;
$f = 0;
$i = 0;
do
{
   if ($ser == $a[ $i ])
   {
      $f = 1;
      break;
   }
      $i++;
}while ( $i < 5 );
if ($f == 1)
 echo "element found";
else
 echo "element not found";
 ?>
```

for loop

```
<?php
$a = Array(2,1,-89,0,5);
$ser= 0;
$f = 0;
for( $i =0; $i < 5 ;$i++ )
{
   if ($ser == $a[ $i ])
   {
      $f = 1;
      break;
   }
}
if ($f == 1)
 echo "element found";
else
 echo "element not found";
?>
```

foreach

```
<?php
$a = Array(2,1,-89,0,5);
$ser= -1;
$f = 0;
foreach( $a as $k => $v )
{
   if ($ser == $v)
   {
      $f = 1;
      break;
   }
}
if ($f == 1)
 echo "element found";
else
 echo "element not found";

?>
```

PHP - Arrays

An array is a data structure that stores one or more values in a single variable. The PHP's arrays are actually maps (each key is mapped to a value). Key is index and value is the data stored at that particular index.

PHP - A Numerically Indexed Array

In PHP the index can be numeric like other programming languages c , c++ and java.The numeric indexing starts from 0 to length – 1;

```
<?php
$a = Array(2,1,-89,0,5); // I^st Way-initialization

/* or element by element initialization 2^and way
$a[0]=2;
$a[1]=1;
$a[2]=-89;
$a[3]=0;
$a[4]=5;
*/
for( $i= 0; $i < 5 ; $i++)
   echo "element at" . $i . "is :" . $a[ $i ];
 // here $i is key and $a[ $i ] is value
?>
```

PHP - Associative Arrays

In an associative array a key is associated with a value. If the index is not numeric, like if array is required to store marks stored by students, where student name is the key/index and the value stored at key is value at index/key then a numerically indexed array would not be the best choice. Instead, second method of non numeric indexing is a best choice where student name is key/index.The advantage is that it is very self explanatory , descriptive or expressive.

```
<?php
/* I^st Initialization
$a=Array(Akshaya=>98,Amirtha=>98,Thanvi=>100,Rewa=>99,Maitr
iya=>99);
*/
$a["Akshaya"]=98;
$a["Amirtha"]=98;
$a["Thanvi"]=100;
$a["Rewa"]==99;
$a["Maitriya"]==99;
foreach( $a as $k => $v)
   echo "Student" . $k . "mark is :" . $v;
 // here $k is key and $v is value
?>
```

PHP Functions

A function is just a name we give to a block of code that can be executed whenever we need it.

Syntax

```
function function_name(){ stmt block;}
```

The key word function give's information PHP interpreter that you are defining function and it is mandatory that functions in PHP should be preceded with function.

Function can

i. Takes no arguments and doesn't return any value.
ii. Takes arguments and doesn't return any value.
iii. Takes no arguments and does return a value.
iv. Takes arguments and does return a value.

```
<?pup
// function returning value, no params passed
function selectChoice()
{
   return 2;
}
// function returning no value, no params passed
function display()
{
   echo "<center><h1>Functions Display</h1></center>
<br/>";
}
// function returning value and params passed
function factorial( $num )
{
    echo "<h3>Factorial</h3>";
    $fact = 1;
    for( $i = 1 ; $i <= $num ; $i++)
        $fact *= i;
   return $fact;
}
// function returning no value and params passed
function fibanocci( $num )
{
   echo "<h3>Fibanocci Series</h3>";
   $a = 0;
   $b = 1;
   $i = 0;
   echo $b . "<br/>";
   do
   {
      $c = $a + $b;
      echo $c . "<br/>";
      $a = $b;
      $b = $c;
      $i++;
   }while($i < $num);
```

```
}
display();
$ch = selectChoice();
switch($ch)
{
    case 1:
         $f = factorial( 5  );
         echo "Factorial    :" . $f;
   break;
   case 2:
         $n = 10;
         fibanocci( $n );
   break;
   default:
      echo "Haven't Chosen Any Thing";
   break;

}
?>
```

Using PHP With HTML Forms

A very common application of PHP is to have an HTML form gather information from a user request and then use PHP to do process that information.

Creating the HTML Form

HTML form can be created using form tag and components within created using input tag.

PHP Processing:

Like Servlet , PHP also follows Request-Response Model.The client/user information entered in the form is given to the server side php script using 2 methods called GET and POST .The information is in associative arrays $_GET (if get method is used) $_POST (if post method is used). $_REQUEST (for both get and post).The information is stored in respective arrays, if the client submits the information by clicking submit button.

HTML Form for getting client information

```
<html>
<head><title>Form Registration</title>
</head>
<body>
<form method="post" action="regisp.php">
<pre>
Enter Ur Regno    :<input type="text" name="Regno"/>
Enter Ur Name     :<input type="text" name="Name"/>
Enter Ur Hobbies  :<input type="text" name="hobby"/>
Choose Ur Membership:
```

```
<input type="radio" name="member" value="CSI"/>  :CSI
<input type="radio" name="member" value="ISTE"/> :ISTE
<input type="radio" name="member" value="ACM"/> :ACM
<select name="choice">
<option>Web Technologies</option>
<option>Open Source Programming</option>
<option>Others</option>
</select>
<input type="submit" name="submit"/>
</pre>
</form>
</body>
</html>
```

regis.php for displaying the form information

```
<html>
<head><title>Form Registration Details</title>
</head>
<body>
<?php

   foreach( $_POST as $k => $v)
   {
      if ( $k != "submit" )
         echo "<b>". $k.": "."</b>".$v."<br>";
   }
?>
</body>
</html>
```

PHP - Files

Manipulating files is a basic necessity for serious programmers and PHP has a great deal of tools for creating, uploading, and editing files.

PHP - Files: Be Careful

When manipulating files care should be taken as a lot of damages can be done.

Common errors include

i. Editing the wrong file
ii. Filling a hard-drive with garbage data
iii. Deleting a file's contents.

PHP – Files

PHP - File Create

In PHP, a file is created using a command that is also used to open files. The fopen function is used to open files. However, it can also **create** a file if it does not **find** the file

specified in the function call. So fopen on a file that does not exist, it will create it, given that the file is opened /created for writing or appending.

Open/create a File

The fopen function needs two important pieces of information to operate correctly. First, the name of the file that has to be created or opened. Secondly, what type of operation is done on the file.

PHP Code

```
<?php
$f_name = "tf.txt";
$file_ref = fopen($f_name, 'w') or die("can't open file");
fclose($file_ref);
?>
```

The file "tf.txt" should be created in the same directory where this PHP code resides. If the file exists,it will open the file or else it will create the file named tf.txt.

$f_name = "tf.txt";

Here we create the name of our file, "tf.txt" and store it into a PHP String variable $f_name

file_ref = fopen($f_name, 'w') or die("can't open file");

The fopen function has two parts. First is the file(file name) and second is mode of opening the file.

The fopen function returns reference to a file(memory) that is created.The return reference is stored in $file_ref variable.

fclose($file_ref);

fclose takes the file reference that is to be closed.

Different Ways to Open a File

There are the three basic ways to open a file.

Read: 'r'

Open a file for read only use. The file pointer begins at the front of the file.

Write: 'w'

Open a file for write only use. In addition, the data in the file is erased if the file already exist and writing begins at the **beginning** of the file. This is also called truncating a file.

Append: 'a'

Open a file for write only use. However, the data in the file is preserved and writing data at the **end** of the file. The file pointer begins at the end of the file. If the file doesn't exist, the file is created and file pointer is in the beginning of the file.

These three basic ways to open a file have distinct purposes. If you want to get information out of a file, like search an e-book for the occurrences of a particular word, then the file is opened in read only.

If contents are to be written in a new file, or overwrite an existing file, then file is opened in "w" option. This would wipe clean all existing data within the file.

If you wanted to add the contents to existing file, then open the file in append mode.

PHP - File Open: Advanced

There are additional ways to open a file. Above we stated the standard ways to open a file. However, a file can be opened many other ways. This can be done by placing a plus sign "+" after the file mode character.

Read/Write: 'r+'

Opens a file so that it can be read from and written to. The file pointer is at the beginning of the file.

Write/Read: 'w+'

This is exactly the same as r+, **except** that it deletes all information in the file when the file is opened.

Append: 'a+'

This is exactly the same as r+, **except** that the file pointer is at the end of the file.

Closing File

The function fclose function is used the close the file after all the operation on the file is over. The fclose function takes file_ref as argument.

PHP - File Write

Apart from open, close operation is manipulations is the most useful part of file operation.The function that does that manipulation is fwrite.It takes two arguments,one is the file handle/reference and the second one is data to be written

.

PHP - File Read

File should be first opened using fopen.If the file doesn't exist, die function is executed.If the file exist, and then file pointer is positioned to the beginning of file.

Contents of file can be read using fread/fgets

fread is used to read the number of characters given as second argument or the whole file can be read by getting the size of file.The fgets function allows line by line reading of file.

Syntax (Take PHP code given below)

```
fread ( $f_ref, 5) // reading 5 bytes/chars
fread ($f_ref, filesize($f_name)) // reading whole file.
filesize($f_name) // gives the size of file.
```

PHP Code: Writing and Reading

```
<?php
$f_name = "a.txt";
$f_ref = fopen($f_name,'w') or die("cannot open");
```

```
$user = "cra";
$pwd = "inar\n";
$both = $user. "  ". $pwd;
fwrite($f_ref , $both);
$s = "pbl nuop\n";
fwrite($f_ref ,$s);
fclose($f_ref);
$f_ref = fopen($f_name, 'r') or die("cannot open");
$c = 0;
while(!feof($f_ref))
{
    $s = fgets($f_ref); //getting line by line
    echo ++ $c;
    echo $s."<br/>";
}
?>
```

***feof ($f_ref) returns true if end of file is reached.**

PHP - File Unlink

The files can be deleted using unlink function in PHP.Before a file is deleted, the file should be closed using fclose function.

```
<?php
$f_name = "t.txt";
unlink($f_name);
?>
```

PHP - File Append

If the previous content has to restore then open the file in append mode to a file that is, add on to the existing data.

```
<?php
$myFile = "dt.txt";
$fh = fopen($myFile, 'a') or die("can't open file");
$stringData = "M.Sujay\n";
fwrite($fh, $stringData);
$stringData = "S/o Madhan.G\n";
fwrite($fh, $stringData);
fclose($fh);
?>
```

The contents of file
M.Sujay
S/o Madhan.G

```
<?php
$myFile = "dt.txt";
$fh = fopen($myFile, 'a') or die("can't open file");
$stringData = "Washermanpet\n";
fwrite($fh, $stringData);
$stringData = "Chennai\n";
fwrite($fh, $stringData);
fclose($fh);
?>
```

The contents of file
M.Sujay
S/o Madhan.G
Washermanpet
Chennai

PHP – Append – Major Use
The above example may not seem very useful, but appending data onto a file is actually used everyday. Almost all web servers have a log of some sort. These various logs keep track of all kinds of information, such as: errors, visitors, and even files that are installed on the machine. A log is basically used to document events that occur over a period of time, rather than all at once. Logs: a perfect use for append.

PHP - File Truncate
When an existing file is opened for writing with the paramater 'w' it completely wipes all data from that file. This is called “truncating" a file.The process is called File Truncate.

Truncating is most often used on files that contain data that will only be used for a short time, before needing to be replaced. These types of files are most often referred to as temporary files.

For example, An online word processor that automatically saves every thirty seconds. Every time it saves it would take all the data that existed within some HTML form text box and save it to the server. This file, say tempSave.txt, would be truncated and overwritten with new, up-to-date data every thirty seconds.This might not be the most efficient program, but it is a nice usage of truncate.

Mostly Used String Functions
PHP - String Position - strpos
The *strpos* function, finds string or substring within a string.

Searching a String with strpos:
strpos(source_string,search_string) – Searches for search_string in source_string and returns the index of the first match and stops once it finds the first match.Searcg starts from index 0.
strpos (source_string,search_string, pos)

Searches for search_string in source_string and returns the index of the first match and stops once it finds the first match.Searcg starts from index pos.

```
<?php
$main_string = "vit , vellore ";
$pos = strpos($main_String, "v");
echo "The position of 5 in our string was $pos";
?>
```

The value of pos is 0

```
<?php
$main_string = "vit , vellore ";
$pos = strpos($main_String, "v" , 3);
echo "The position of v in our string is $pos";
?>
```

The value of pos is 6.

Finding All Occurrences in a String with Offset

One of the limitations of strpos is that it only returns the position of the very first match. There is a third (optional) argument to strpos that is used to specify where search of the string should begin. If the position of the last match is stored and use that + 1 as an offset, then all the occurrences can be found. (Use any iterative control structure).

```
<?php
$main_string = "vit , vellore ";
$pos = 0;
$c=0;
while($pos = (strpos($main_String, "v" , $pos + 1))
    $c++;
echo "The number of occurences " . $c;
?>
```

PHP - String Explode

The PHP function explode is similar to split function of java. The function splits(explodes) the string into an array of strings based on punctuator / string/character.

Phone.html

```
<html>
<head><title>USAGE OF EXPLODE</title></head>
<body>
<form method="post" action="PhonePro.php">
Enter Ur Phone Number:<input type="text" name="pno"/>
<input type="submit" name="submit" value="submit">
</form>
```

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

PhonePro.php

```
<html>
<head><title>Display Phone and Area</title></head>
<body>
<?php
function splitNcheck($f_data)
{
      $s_data = explode("-",$f_data);
      switch($s_data[0])
      {
            case "011":
               echo "STD CODE OF Delhi"."<br\>";
            break;
            case "022":
               echo "STD CODE OF Mumbai"."<br\>";
            break;
            case "033":
               echo  "STD CODE OF Calcutta"."<br\>";
            break;
            case "044":
               echo "STD CODE OF Chennai"."<br\>";
            break;
            case "0416":
               echo "STD CODE OF Vellore"."<br\>";
            break;
            default:
            echo "STD CODE NOT RECOGONIZED" . "<br\>";
            break;
      }
      echo  "Phone Number = " .$s_data[ 1 ]. "<br\>";
}
$f = $_POST['pno'];
splitNcheck($f);
?>
</body>
</html>
```

PHP - Array implode

The PHP function implode is known as the "undo" of explode or equivalent to JavaScript join

```
<?php
```

```
$arr = array("IPL","Season 5","Favourites","Chennai Super Kings");
$arr_str = implode(" ", $arr);
echo $arr_str."<br/>";
?>
```

Other String Functions in PHP

FUNCTION	SYNTAX	EXPLANATION
chr	string **chr** (int $ascii)	Returns a one-character string containing the character specified by **ascii**. Returns a one-character string containing the character specified by **ascii**.
ord	int **ord** (string $string)	Returns the ASCII value of the first character of **string**.
chop	string **rtrim** (string $str [, string $charlist])	This function is an alias of: rtrim().
str_split	array **str_split** (string $string [, int $split_length = 1])	Converts a string to an array. Where each array element is size 1 (default) or of length,given by the second argument.
str_replace	mixed **str_replace** (mixed $search , mixed $replace , mixed $subject [, int &$count]) * mixed is any type	This function returns a string or an array with all occurrences of **search** in **subject** replaced with the given **replace** value.

strlen	int **strlen** (string $string)	Returns the length of the given **string**.
strcmp	int **strcmp** (string $str1 , string $str2)	Returns < 0 if **str1** is less than **str2**; > 0 if **str1** is greater than **str2**, and 0 if they are equal
strcasecmp	int **strcasecmp** (string $str1 , string $str2)	Returns < 0 if **str1** is less than **str2**; > 0 if **str1** is greater than **str2**, and 0 if they are equal
trim	string **trim** (string $str [, string $charlist])	This function returns a string with whitespace stripped from the beginning and end of **str**. Without the second parameter, **trim()** will strip these characters:
ucwords	string **ucwords** (string $str)	Returns a string with the first character of each word in **str** capitalized, if that character is alphabetic. The definition of a word is any string of characters that is immediately after a whitespace (These are: space, form-feed, newline, carriage return, horizontal tab, and vertical tab).
strtolower	string **strtolower** (string $str)	Returns **string** with all alphabetic characters converted to lowercase.
strtoupper	string **strtoupper** (string $string)	Returns **string** with all alphabetic characters converted to uppercase.
substr	string **substr** (string $string	Returns the portion of

	, int $start [, int $length])	**string** specified by the **start** and **length** parameters. If **start** is non-negative, the returned string will start at the **start**'th position in **string**, counting from zero. For instance, in the string '**abcdef**', the character at position **0** is '**a**', the character at position **2** is '**c**', and so forth. If **start** is negative, the returned string will start at the **start**'th character from the end of **string**. If **string** is less than or equal to **start** characters long, **FALSE** will be returned.
strripos	int **strripos** (string $source_String , string $ser_string [, int $offset = 0])	Find the numeric position of the last occurrence of ser_string in the source_string. **strripos()** is case-insensitive.
print	int **print** (string $arg)	**print** is not actually a real function (it is a language construct) so you are not required to use parentheses with its argument list.
count_chars	mixed **count_chars** (string	Counts the number of

	$string [, int $mode = *0*])	occurrences of every byte-value (0..255) in **string** and returns it in various ways. ▪ - an array with the byte-value as key and the frequency of every byte as value. ▪ 1 - same as 0 but only byte-values with a frequency greater than zero are listed. ▪ 2 - same as 0 but only byte-values with a frequency equal to zero are listed. ▪ 3 - a string containing all unique characters is returned. ▪ 4 - a string containing all not used characters is returned.
printf	int **printf** (string $format [, mixed $args [, mixed $...]])	Prints datta in preferred format

```
<?php
$data = "Two Ts and one F.";

foreach (count_chars($data, 1) as $i => $val) {
   echo "There were $val instance(s) of \"" , chr($i) , "\"
 in the string.\n";
}
?>
```

Example of printf

```
<?php
$n =  43951789;
$u = -43951789;
$c = 65; // ASCII 65 is 'A'

//notice the double %%, this prints a literal '%' character
printf("%%b = '%b'\n", $n);// binary representation
printf("%%c = '%c'\n", $c);// print the ascii character
printf("%%d = '%d'\n", $n);//standard integer
printf("%%e = '%e'\n", $n); // scientific notation
printf("%%u = '%u'\n", $n);// unsigned integer
printf("%%u = '%u'\n", $u); // unsigned integer
printf("%%f = '%f'\n", $n); // floating point
printf("%%o = '%o'\n", $n); // octal representation
printf("%%s = '%s'\n", $n); // string representation
printf("%%x = '%x'\n", $n); // hexadecimal representation
printf("%%X = '%X'\n", $n); // hexadecimal representation
printf("%%+d = '%+d'\n", $n); // sign specifier
printf("%%+d = '%+d'\n", $u); // sign specifier
$s = 'monkey';
$t = 'many monkeys';
printf("[%s]\n",      $s);//standard string output
printf("[%10s]\n",    $s);//right-justification with spaces
printf("[%-10s]\n",   $s);// left-justification with spaces
printf("[%010s]\n",   $s);//zero-padding works on strings
printf("[%'#10s]\n",  $s);//use the custom padding char '#'
printf("[%10.10s]\n", $t);//left-justify of 10 characters
$isodate = sprintf("%04d-%02d-%02d", $year, $month, $day);
$money1 = 68.75;
$money2 = 54.35;
$money = $money1 + $money2;
// echo $money will output "123.1";
$formatted = sprintf("%01.2f", $money);
// echo $formatted will output "123.10"
$number = 362525200;
echo sprintf("%.3e", $number); // outputs 3.625e+8
?>
```

PHP Date - date

The date function uses letters of the alphabet to represent various parts of a typical date and time format. The letters that are used are:

- **d**: The day of the month. The type of output is 01 through 31.
- **m**: The current month, as a number. The type of output is 01 through 12.

• **y**: The current year in two digits ##. That is value 00 through 99

The characters like a slash "/" can be inserted between the letters to add additional formatting.

r: Displays the full date, time and timezone offset. It is equivalent to manually entering date ("D, d M Y H: i: s O")

PHP date function takes 2 arguments. The first argument is format of date and second argument is Time Stamp (optional)

PHP Timestamp - mktime

The function mktime is used for TimeStamp.The mktime function can be used to create a timestamp for tomorrow. To go one day in the future, simply add one to the day argument of mktime.

• mktime (hour, minute, second, month, day, year, daylight savings time)

Time:

• **a**: am or pm depending on the time
• **A**: AM or PM depending on the time
• **g**: Hour without leading zeroes. Values are 1 through 12.
• **G**: Hour in 24-hour format without leading zeroes. Values are 0 through 23.
• **h**: Hour with leading zeroes. Values 01 through 12.
• **H**: Hour in 24-hour format with leading zeroes. Values 00 through 23.
• **i**: Minute with leading zeroes. Values 00 through 59.
• **s**: Seconds with leading zeroes. Values 00 through 59.

Day:

- **d**: Day of the month with leading zeroes. Values are 01 through 31.
- **j**: Day of the month without leading zeroes. Values 1 through 31
- **D**: Day of the week abbreviations. Sun through Sat
- **l**: Day of the week. Values Sunday through Saturday
- **w**: Day of the week without leading zeroes. Values 0 through 6.
- **z**: Day of the year without leading zeroes. Values 0 through 365.

Month:

- **m**: Month number with leading zeroes. Values 01 through 12
- **n**: Month number without leading zeroes. Values 1 through 12
- **M**: Abbreviation for the month. Values Jan through Dec
- **F**: Normal month representation. Values January through December.
- **t**: The number of days in the month. Values 28 through 31.

Year:

- **L**: 1 if it's a leap year and 0 if it isn't.
- **Y**: A four digit year format
- **y**: A two digit year format. Values 00 through 99.

Other Formatting:

- **U**: The number of seconds since the Unix Epoch (January 1, 1970)
- **O**: This represents the Timezone offset, which is the difference from Greenwich Meridian Time (GMT). 100 = 1 hour, -600 = -6 hours

PHP Code:

```
<?php
 echo date("d/m/y");
 echo date("D/M/Y");
 $tomo   =   mktime(0,   0,   0,   date("m"),   date("d")+1,
date("y"));
echo "Tomorrow is ".date("d/m/y", $tomo);
echo date("r");
?>
```

PHP/MySQL

Open source has brought a lot more than Linux to the computing world. It has also given us PHP and MySQL. PHP and MySQL are the world's best combination for creating data-driven sites. MySQL is a small, compact database server ideal for small - and not so small - applications. In addition to supporting standard SQL (ANSI), it compiles on a number of platforms and has multithreading abilities on Unix servers, which make for great performance. For non-Unix people, MySQL can be run as a service on Windows NT and as a normal process in Windows 95/98 machines. PHP is a server-side scripting

language.PHP script is processed by the Web server like servlet/JSP/Perl/ASP. After processing the server returns plain old HTML back to the browser.
In addition to being free (MySQL does have some licensing restrictions though), the PHP-MySQL combination is also cross-platform, which means you can develop in Windows and serve on a Unix platform.Also, PHP can be run as an external CGI process, a stand-alone script interpreter, or an embedded Apache module.
If you're interested, PHP also supports a massive number of databases, including Informix, Oracle, Sybase,Solid, and PostgreSQL - as well as the ubiquitous ODBC.
PHP supports a host of other features right at the technological edge of Internet development. These include authentication, XML, dynamic image creation, WDDX, shared memory support, and dynamic PDF document creation to name but a few.

Example : PHP with MySQL

mysql> use shop;
Database changed
mysql> desc shopcart;

```
+--------+----------------+------+---------+----------+--------+
| Field  | Type           | Null | Key     | Default | Extra |
+--------+----------------+------+---------+----------+--------+
| bname  | varchar(30) | YES |          | NULL   |         |
| bcount | int(11)        | YES |          | NULL   |         |
| price  | int(11)        | YES |          | NULL   |         |
+----------+--------------+------+---------+----------+-------+
```

3 rows in set (0.14 sec)

mysql> select * from shopcart;

```
+---------------------------------------+---------+---------+
| bname                                 | bcount | price   |
+---------------------------------------+---------+---------+
| Java Black Book                       |    17   |   500   |
| C++ Complete Reference                |    39   |   420   |
| Design Of Algorithms                  |    40   |   450   |
| Database Management Systems |    17   |   500   |
| Internet Programming                  |    15   |   350   |
| Datastructures                        |    19   |   360   |
+---------------------------------------+--------+--------+
```

6 rows in set (0.11 sec)

PHP Code: Retrieval Query- Displaying first record of table

```
<html>
<body>
<?php
$db = mysql_connect("localhost", "root","amirtha");
```

```
mysql_select_db("shop",$db);
$result = mysql_query("SELECT * FROM shopcart",$db);
echo "BookName:".mysql_result($result,0,"bname") ."<br/>";
echo "Available". mysql_result($result,0,"bcount")."<br/>";
echo "Price: ". mysql_result($result,0,"price")."<br/>";
?>
</body>
</html>
```

The mysql_connect() function opens a link to a MySQL server on the specified host (in this case it's localhost) along with a username (root) and password, if any other wise the last argument is optional.The result of the connection is stored in the variable $db.mysql_select_db () then tells PHP that any queries we make are against the shop database. We could create multiple connections to databases on different servers. Next, mysql_query () does all the hard work. Using the database connection identifier, it sends a line of SQL to the MySQL server to be processed. The results that are returned are stored in the variable $result (result set i.e, set of all tuples).Finally, mysql_result () is used to display the values of fields from our query. Using $result, we go to the first row, which is numbered 0, and display the value of the specified fields.

The result of the above PHP code is:

Book Name: Java Black Book
Available: 17
Price: 500

PHP Code: To display all the records of the table shopcart in database shop

```
<html>
<body>
<?php
$db = mysql_connect("localhost", "root","amirtha");
mysql_select_db("shop",$db);
$result = mysql_query("SELECT * FROM shopcart",$db);
echo "<table border=1>\n";
echo
"<tr><td>Name</td><td>Available</td><td>price</td></tr>\n";
while ($myrow = mysql_fetch_row($result)) {
echo "<tr><td>". $myrow [0]. "</td><td>“. $myrow
[1]."</td><td>". $myrow [2]."</td></tr>";
}
echo "</table>\n";
?>
</body>
</html>
```

The output of the previous PHP Code is given below

Name	Available	price
Java Black Book	17	500
C++ Complete Reference	39	420
Design Of Algorithms	40	450
Database Management Systems	17	500
Internet Programming	15	350
Datastructures	19	360

The mysql_fetch_row () function fetches a row/tuple/record from the result set object (variable) $result, which has the records/tuples/rows fetched from the table the values in the variable $myrow. Each field of the tuple is referenced as $myrow [0], $myrow [0] and so on.The above example supports only numeric references to the individual fields. So the first field is referred to as 0, the second as 1, and so on. With complex queries this can become something of a nightmare.

Alternate Way of accessing fields of the tuples in PHP Code

```
<html>
<body>
<?php
$db = mysql_connect("localhost", "root","amirtha");
mysql_select_db("shop",$db);
$result = mysql_query("SELECT * FROM shopcart",$db);
echo "<table border=1>\n";
echo
"<tr><td>Name</td><td>Available</td><td>price</td></tr>\n";
while ($myrow = mysql_fetch_row($result)) {
echo "<tr><td>.$myrow["bname"]. "</td><td>"
.$myrow["bcount"]."</td><td>".
$myrow["price"]."</td></tr>";
}
echo "</table>\n";
?>
</body>
</html>
```

The mysql_fetch_row () function fetches a row/tuple/record from the result set object (variable) $result, which has the records/tuples/rows fetched from the table the values in the variable $myrow. Each field of the tuple is referenced as $myrow [0], $myrow [0]

and so on.The above example supports string references to the individual fields. So the first field is referred to as bname, the second as bcount, and third by price, where bname, bcount and price are the names of fields of the table shopcart.This is more informative as the fields are referred by name instead of numerical references.

PHP Code for selecting specific rows/tuples

```
<html>
<body>
<?php
$db = mysql_connect("localhost", "root","amirtha");
mysql_select_db("shop",$db);
$result = mysql_query("SELECT * FROM shopcart where bname='Java Black Book' ",$db);
echo "<table border=1>\n";
echo "<tr><td>Name</td><td>Available</td><td>price</td></tr>\n";
while ($myrow = mysql_fetch_row($result)) {
echo "<tr><td>".$myrow[0]. "</td><td>" .$myrow[1]."</td><td>". $myrow[2]."</td></tr>";
}
echo "</table>\n";
?>
</body>
</html>
```

In the above program the where condition of select query can be checked against constant value or variable value.if the value/variable is number it can be given directly.If it s a string it should be given within quotes.

$i = 1; if the value/variable is number

mysql_query("SELECT * FROM employee where eno= $i,$db);
(or)
mysql_query("SELECT * FROM employee where eno= 1,$db);

$n="chitra" ; if the value/variable is number

mysql_query("SELECT * FROM employee where ename= '$n',$db);
(or)
mysql_query("SELECT * FROM employee where ename= 'chitra',$db);

PHP interaction with the form

The client sends the information/requisition as form data through the user agent. The request(request object) is sent to the server. The server then invokes the server side script, which in this case is PHP script. The script processes this request and sends the response to the server. The server then sends the response(response object) back to the client through the user agent.

HTML Form for getting client information

```
<html>
<head><title>Form Registration</title>
</head>
<body>
<form method="post" action="regisp.php">
<pre>
Enter Ur Regno    :<input type="text" name="Regno"/>
Enter Ur Name     :<input type="text" name="Name"/>
Enter Ur Hobbies  :<input type="text area" name="hobby">
Choose Ur Membership:
<input type="radio" name="member" value="CSI"/>  :CSI
<input type="radio" name="member" value="ISTE"/> :ISTE
<input type="radio" name="member" value="ACM"/> :ACM
<select name="choice">
<option>Web Technologies</option>
<option>Open Source Programming</option>
<option>Others</option>
</select>
<input type="submit" name="submit"/>
</pre>
</form>
</body>
</html>
```

regis.php for displaying the form information

```
<html>`
<head><title>Form Registration Details</title>
</head>
<body>
<?php

   foreach( $_POST as $k => $v)
   {
      if ( $k != "submit" )
         echo "<b>".$k.":" ."</b>".$v."<br>";
   }
?>
</body>
```

```
</html>
```

Survey.html

```
<html>
<head><title>REGISTER Ur Vote</title></head>
<body bgcolor="#000000" text="#FFFFFF">
<center>
<h1><b>POLL YOUR VOTES FOR UR FAVOURITE BOOK</b></h1>
</center>
<form method = "get" action = "surveypoll.php">
<pre>

<font color="red"  size="7px"  name="times roman">

Which is Ur favourite book?
<input type="radio" name = "b_name" value = "if tommorrow
comes" >If Tommorrow Comes
<input type="radio" name = "b_name" value = "coma" >Coma
<input type="radio" name = "b_name" value = "love story"
>Love Story
<input type="radio" name = "b_name" value = "prodigal
daughter" >Prodigal Daughter
<input type="radio" name = "b_name" value = "Jane Ayre"
>Jane Ayre
<input type ="submit" name = "vote" value ="VOTE"><input
type ="reset" value = "CLEAR" >

</font>
</pre>
</body>
</html>
```

surveypoll.php

```
<!--Survey Poll- PHP can have a mix of named
blocks(functions) and un-named blocks.
can be embedded any where and the php code is called from
html file(survey.html)-->

<html>
<body>
<?php
function updateDB($book , $vct ) //php func to updt db
```

```
{

   $db1 = mysql_connect("localhost", "root","amirtha");
   mysql_select_db("e",$db1);
   $ar = "UPDATE favbook SET ct=$vct WHERE trim(bookname) =
'$book' ";
   $r = mysql_query($ar);
   mysql_close($db1);

}
function display($_a) //php function to display the results
{

   echo "<table border=1 width=\"25%\" bgcolor=\"orange\"
align=\"center\">";

   echo "<tr><td><h1>Book
Name</h1></td><td><h1>Votes</h1></tr>";
   foreach($_a as $key=>$value)
   {

      echo
"<tr><td><h3>".$key."</h3></td><td><h3>".$value."</h3></td>
</tr>\n";
   }

   echo "</table>\n";
}

$v = $_GET['b_name'];  //getting form data
                       // php unamed block start

$db = mysql_connect("localhost", "root","amirtha");
mysql_select_db("e",$db);
$result = mysql_query("SELECT * FROM favbook",$db);

/*fetching the data from the table row-wise,$myrow[0] is
the first field and $myrow[1] is the second field
*/

while ($myrow = mysql_fetch_row($result))
$_arr[$myrow[0]] =$myrow[1];

foreach($_arr as $key=>$value)
{
   if ($key == $v )
```

```
   {
      $c = $value+ 1;
      $_arr[$key]=$c;
      break;
   }
}
mysql_close($db);
display($_arr);
updateDB($v,$c);
// php unamed block end

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

```
mysql> select * from favbook;
+--------------------+------+
| bookname           | ct      |
+--------------------+------+
| if tommorrow comes |    5 |
| coma               |    8 |
| love story         |    8 |
| prodigal daughter  |   13 |
| Jane Ayre          |   10 |
+--------------------+------+
```

```
$_arr[$myrow[0]] =$myrow[1];

Storing the fields as key value pairs.field[0] is key and
field[1] as value.

Eg:
$_arr["if tommorrow comes "] = 5;
```

Connecting to MS Access

```
<html>
<body>

<?php


$conn=odbc_connect('demo'," "," "); //demo is the dsn
if (!$conn)
  {exit("Connection Failed: " . $conn);}
$sql="select * from student";
```

```
$rs=odbc_exec($conn,$sql);
if (!$rs)
  {exit("Error in SQL");}
echo "<table><tr>";
echo "<th>Student regno</th>";
echo "<th>Name</th></tr>";
while (odbc_fetch_row($rs))
  {
  $r=odbc_result($rs,"regno");
  $n=odbc_result($rs,"name");
  echo "<tr><td>$r</td>";
  echo "<td>$n</td></tr>";
  }
odbc_close($conn);
echo "</table>";
?>

</body>
</html>
```

PHP Sessions

As a website becomes more sophisticated, so must the code that backs it. When data needs to be passed from one page to another, PHP sessions can be used. A normal HTML website will not pass data from one page to another. In other words, all information is forgotten when a new page is loaded. This makes it quite a problem for tasks like a shopping cart, which requires data (the user's selected product) to be remembered from one page to the next.

A PHP session solves this problem by storing user information on the server for later use (i.e. username, shopping cart items, etc). However, this session information is temporary and is usually deleted very quickly after the user has **left** the website that uses sessions. If the data has to be stored permanently then files or database can be used.Sessions work by creating a unique identification (UID) number for each visitor and storing variables based on this ID. This helps to prevent two users' data from getting confused with one another when visiting the same webpage.

login.html

```
<html>
<head>
<title>PHPMySimpleLogin</title>
</head>
<body>
<h3>User Login</h3>
<form method="POST" action="loginpro.php">
Username:<input type="text" name="username" size="20">
Password:<input type="password" name="password" size="20">
```

```
<input type="submit" value="Login">
</form>
</table>
</body>
</html>
```

loginpro.php

```
<html>
<head>
<title>PHPMySimpleLogin</title>
</head>
<body>
<h3>User Login</h3>
<form method="POST" action="loginpro.php">
Username:<input type="text" name="username" size="20">
Password:<input type="password" name="password" size="20">
<input type="submit" value="Login">
</form>
</table>
</body>
</html>
```

nextpage.php

```
<html>
<head>
<title>PHP Page</title>
</head>
<body>

<?php
session_start();
echo "<h1>hello"  . "   ". $_SESSION['username']."</h1>";
?>

</body>
</html>
```