Basic PHP Syntax

INTRODUCTION TO HTML AND PHP

Shano Solanki
Asst. professor, Comp. Sc.
NITTTR, Chandigarh

Presentation outline

ÅIntroduction to HTML

ÅIntroduction to PHP

ÅVariables

ÅFunctions

ÅControl Structure

ÅLooping

ÅArrays

HTML INTRODUCTION

- HTML document should contain certain standard HTML tags. Each document consists of head and body text. The head contains the title, and the body contains the actual text that is made up of paragraphs, lists, and other elements. Browsers expect specific information because they are programmed according to HTML specifications.
- *HTML is not case sensitive.*
- Not all tags are supported by all World Wide Web browsers. If a browser does not support a tag, it will simply ignore it.

HTML

- **WHAT DO WE MEAN BY HTML LANGUAGE?**

  The language used to develop webpages is called Hyper Text Markup Language(HTML). HTML is a language interpreted by a browser. Web pages are also called HTML documents.

- **WHAT ARE TAGS?**

  HTML is specified as TAGS in HTML document. By convention all the HTML TAGS starts with an open angle bracket(<) and end with and close angle bracket(>).

  Note : HTML files are saved as .htm or .html

Basic Structure for HTML

```
<HTML>
<HEAD>
<TITLE> WELCOME TO MY WEBPAGE </TITLE>
</HEAD>

<BODY>


</BODY>
</HTML>
```

Example 1 Introhtml.htm

```
<HTML>
<HEAD>
<TITLE> WELCOME TO MY WEBPAGE </TITLE>
</HEAD>
<BODY BGCOLOR=CYAN TEXT=BLUE>
<H1> INTRODUCTION TO HTML LANGUAGE </H1>
<H3> WHAT DO WE MEAN BY HTML LANGUAGE?
</H3>
The language used to develop webpages is called Hyper Text
Markup Language(HTML).
HTML is a language interpreted by a browser. Webpages are also
called HTML documents.
<P ALIGN=LEFT>
```

```
<H3> WHAT ARE TAGS? </H3>
HTML is specified as TAGS in HTML document. By convention all the
HTML TAGS starts with an open angle bracket(<) and end with and close
angle bracket(>).<BR><BR><BR>
<HR>
<B><U><FONT COLOR=BLACK>FOR ANY FURTHER CHANGES
IN THIS DOCUMENT CONTACT : </FONT></U></B>
<BR><BR>
<CENTER>
<FONT SIZE=2 COLOR=RED FACE="ARIAL"><I>
DEPARTMENT OF COMPUTER SCIENCE<BR>
NITTTR, SEC-26, CHANDIGARH-160019<BR>
PHONE NO.'S 0172-2791349/51 EXTN. 496</I>
</FONT>
</CENTER>
</BODY>
</HTML>
```

introhtml2.htm HTML file

```
<HTML>
<HEAD>
<TITLE> LEARN ABOUT SOME ADDITIONAL TEXT EFFECTS
</TITLE>
</HEAD>

<BODY BGCOLOR=PINK TEXT=RED>

<! COMMENT TAGS ARE NOT MENT FOR DISPLAY IN BROWSER -->

<FONT COLOR=BLUE SIZE=3><B>FONT TAG CAN BE USED TO
SET FONT FACE, SIZE AND COLOR ETC.<BR><BR>
SIZE ATTRIBUTE CAN TAKE VALUES BETWEEN 1 TO 7. THE
DEFAULT SIZE IS 3.
```

```
</B></FONT><BR><BR>

<FONT FACE = "ARIAL" SIZE=4 COLOR=RED >
ADDITIONAL TEXT EFFECTS </FONT><BR><BR>

SUPERSCRIPT TAG<BR>
<I>4X<SUP>2</SUP> + 2X + 3 = 5</I><BR><BR>

SUBSCRIPT TAG<BR>
<I> H<SUB>2</SUB>SO<SUB>4</SUB> </I>

</BODY>
</HTML>
```

LISTS IN HTML

- ORDERED LISTS
- UNORDERED LISTS
- DEFINITION LISTS

```
<HTML>
<HEAD>
<TITLE> LEARN ABOUT LISTS OF DIFFERENT TYPES AND THEIR ATTRIBUTES </TITLE>
</HEAD>
 <BODY BGCOLOR=LAVENDER TEXT=BLACK>
 <FONT FACE = "ARIAL" SIZE=4 COLOR=RED > ORDERED LISTS </FONT> <BR><BR>
LIST OF STANDARD  COLORS TO BE USED IN WEBPAGES<BR><BR>
<OL>
<LI>BLACK
<LI>GREEN
<LI>SILVER
</OL>
<HR>
<OL TYPE="A" START=3>
<LI>BLACK
<LI>GREEN
<LI>SILVER
</OL>
<UL TYPE=FILLROUND>
 <LI>MARRON
 <LI>NAVY
<LI>RED
</UL>
</BODY> </HTML>
```

DEFINITION LIST

```
<DL>
   <DT> NCSA
   <DD> NCSA, the National Center for Supercomputing
     Applications, is located on the campus of the
     University of Illinois at Urbana-Champaign.
   <DT> Cornell Theory Center
   <DD> CTC is located on the campus of Cornell
     University in Ithaca, New York.
</DL>
```

```
<HTML>
 <HEAD>
<TITLE>DEFINITION LIST</TITLE>
 </HEAD>
<BODY BGCOLOR=LAVENDER TEXT=BLACK>
<FONT FACE = "ARIAL" SIZE=4 COLOR=RED > LISTS OF DIFFERENT TYPES AND
THEIR ATTRIBUTES </FONT><BR><BR>
 <H4>DEFINITION LIST</H4><BR>
 <H4>Definition list consists of two parts:</H4><BR><BR>
<B>DEFINTION TERM:</B> APPEARS AFTER THE TAG <DT><BR>
<B>DEFINITION DESCRIPTION :</B> APPEARS AFTER THE TAG <DD><BR><BR>
<HR>
<H5> EXAMPLE :</H5><BR><BR>
<DL>
<DT><B> ORDERED LISTS :</B> <DD> THESE LISTS ARE TYPICALLY USED TO
INDICATE A SEQUENCE OF EVENTS OR PRIORITIES. <BR><BR> <DT><B>
UNORDERED LISTS:</B>
<DD> AN UNORDERED LIST IS TYPICALLY USED TO DISPLAY A GROUP OF ITEMS THAT
ARE SOMEHOW RELATED, BUT NECESSARILY IN A HIERARCHICAL
FASHION.<BR><BR>
 </DL>
</BODY>
</HTML>
```

Image tag <img>

ALT="mas"
If the image can not be displayed, then the text associated with the ALT attribute will be displayed instead of the image. This occurs, if for example the display of images has been turned off in the browser, or for some reason the browser can not display the image.

ALIGN=TOP
Causes any following text to be displayed aligned with the top of the picture.
ALIGN=BOTTOM.
Causes any following text to be displayed aligned with the bottom of the picture.
ALIGN=MIDDLE.
Causes any following text to be displayed aligned with the middle of the picture.
ALIGN=LEFT
Cause the image to be left aligned on the page. Text is flowed around the image on the right hand side.
ALIGN=RIGHT
Cause the image to be right aligned on the page. Text is flowed around the image on the left hand side.
HEIGHT=n
Set the height of the image to be n pixels.
WIDTH=n

Set the width of the image to be n pixels.

Example

Example

<IMG ALT="A picture of Olivia and Elena" WIDTH="210" HEIGHT="136" SRC="http://www.sonic.net/~roy/girls96a.jpeg">

Server-Side Scripting

- A "script" is a collection of program or sequence of instructions that is interpreted or carried out by another program rather than by the computer processor.
  - Client-side
  - Server-side
- In server-side scripting, (such as PHP, ASP) the script is processed by the server Like: Apache, ColdFusion, Microsoft's IIS on Windows.
- Client-side scripting such as JavaScript runs on the web browser.

What is PHP?

- PHP stands for **P**HP: **H**ypertext **P**reprocessor
- Developed by Rasmus Lerdorf in 1994
- 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 a PHP File?

- PHP files can contain text, HTML, JavaScript code, and PHP code
- PHP code are executed on the server, and the result is returned to the browser as plain HTML
- PHP files have a default file extension of ".phpñ
- A PHP file can be included using include statement

include ώabc.phpộ; // this statement is added in any php file for including another php file.

What Can PHP Do?

- PHP can generate dynamic page content
- PHP can create, open, read, write, 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 restrict users to access some pages on your website
- PHP can encrypt data

Why PHP?

- PHP runs on different platforms (Windows, Linux, Unix, Mac OS X, etc.)
- PHP is compatible with almost all servers used today (Apache, IIS, etc.)
- PHP has support for 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

Editors for PHP

PHP programs can be written using text editors :

- notepad
- notepadplusplus(freely available)
- adobe dreamweaver
- NetBeans(integrated development environment)
- eclipse etc.

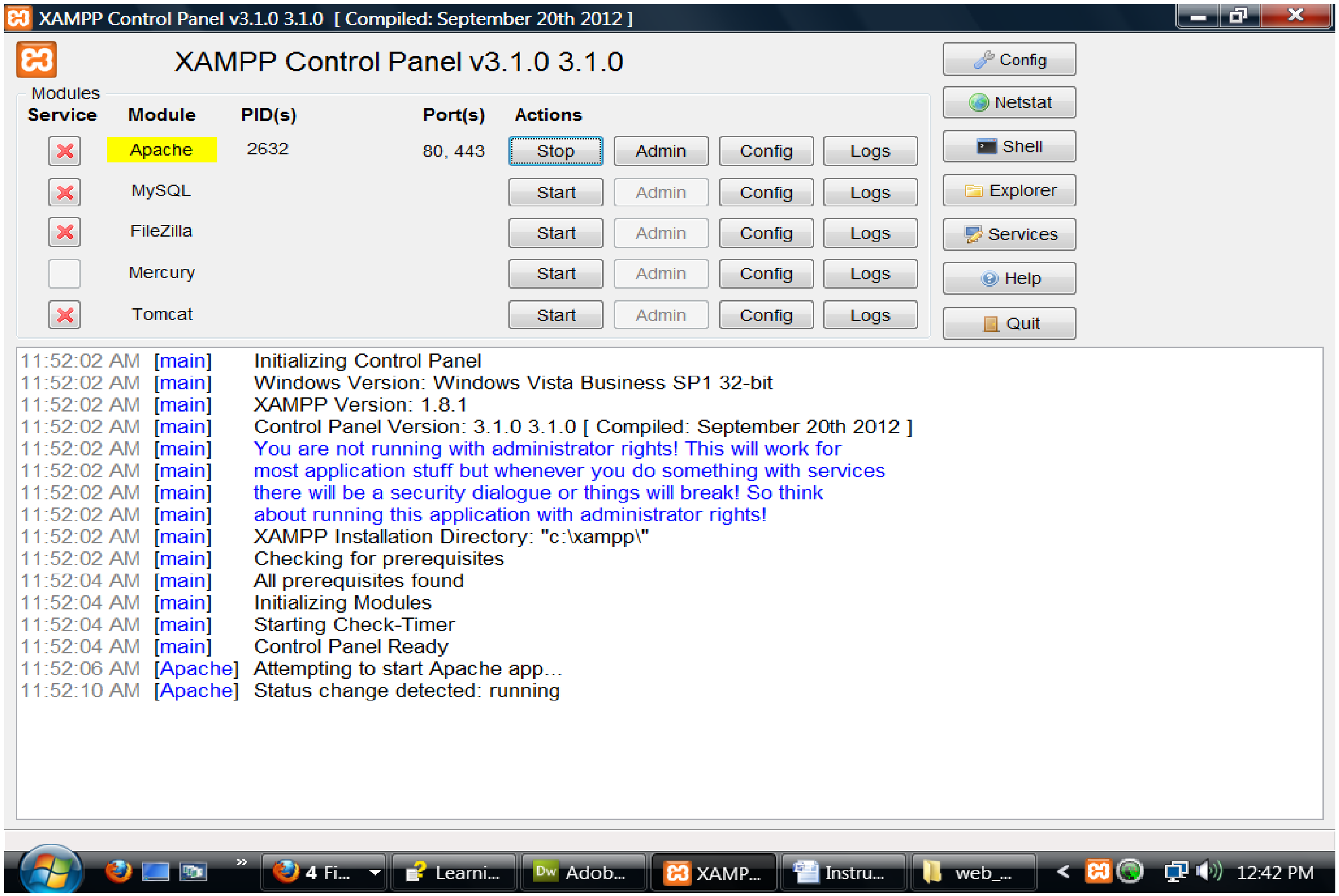
Set Up PHP On Your Own PC

Install Xampp for windows from website
https://www.apachefriends.org/download.html

XAMPP is an easy to install Apache distribution containing MariaDB, PHP, and Perl. Just download and start the installer. It's that easy.

How to run php file

- Save the .php file in htdocs folder (root directory) and in this case we can run the php program from any browser like Internet Explorer, Mozilla Firefox, Google Chrome etc.
- Before running a program we need to start Apache and Mysql from XAMPP control panel. Run the program by writing a command http://localhost/php1.php in address bar in web browser.
- Save the .php file in any user defined folder in htdocs folder. In this case for running a php program we need to run a command http://localhost/php_programs/php1.php in address bar in web browser.



Service	Module	PID(s)	Port(s)	Actions			
✗	Apache	2632	80, 443	Stop	Admin	Config	Logs
✗	MySQL			Start	Admin	Config	Logs
✗	FileZilla			Start	Admin	Config	Logs
	Mercury			Start	Admin	Config	Logs
✗	Tomcat			Start	Admin	Config	Logs

```
11:52:02 AM  [main]     Initializing Control Panel
11:52:02 AM  [main]     Windows Version: Windows Vista Business SP1 32-bit
11:52:02 AM  [main]     XAMPP Version: 1.8.1
11:52:02 AM  [main]     Control Panel Version: 3.1.0 3.1.0 [ Compiled: September 20th 2012 ]
11:52:02 AM  [main]     You are not running with administrator rights! This will work for
11:52:02 AM  [main]     most application stuff but whenever you do something with services
11:52:02 AM  [main]     there will be a security dialogue or things will break! So think
11:52:02 AM  [main]     about running this application with administrator rights!
11:52:02 AM  [main]     XAMPP Installation Directory: "c:\xampp\"
11:52:02 AM  [main]     Checking for prerequisites
11:52:04 AM  [main]     All prerequisites found
11:52:04 AM  [main]     Initializing Modules
11:52:04 AM  [main]     Starting Check-Timer
11:52:04 AM  [main]     Control Panel Ready
11:52:06 AM  [Apache]   Attempting to start Apache app...
11:52:10 AM  [Apache]   Status change detected: running
```

Fig 1.1 XAMPP control panel from where we can start or stop the various services provided by different modules.

Basic PHP Syntax

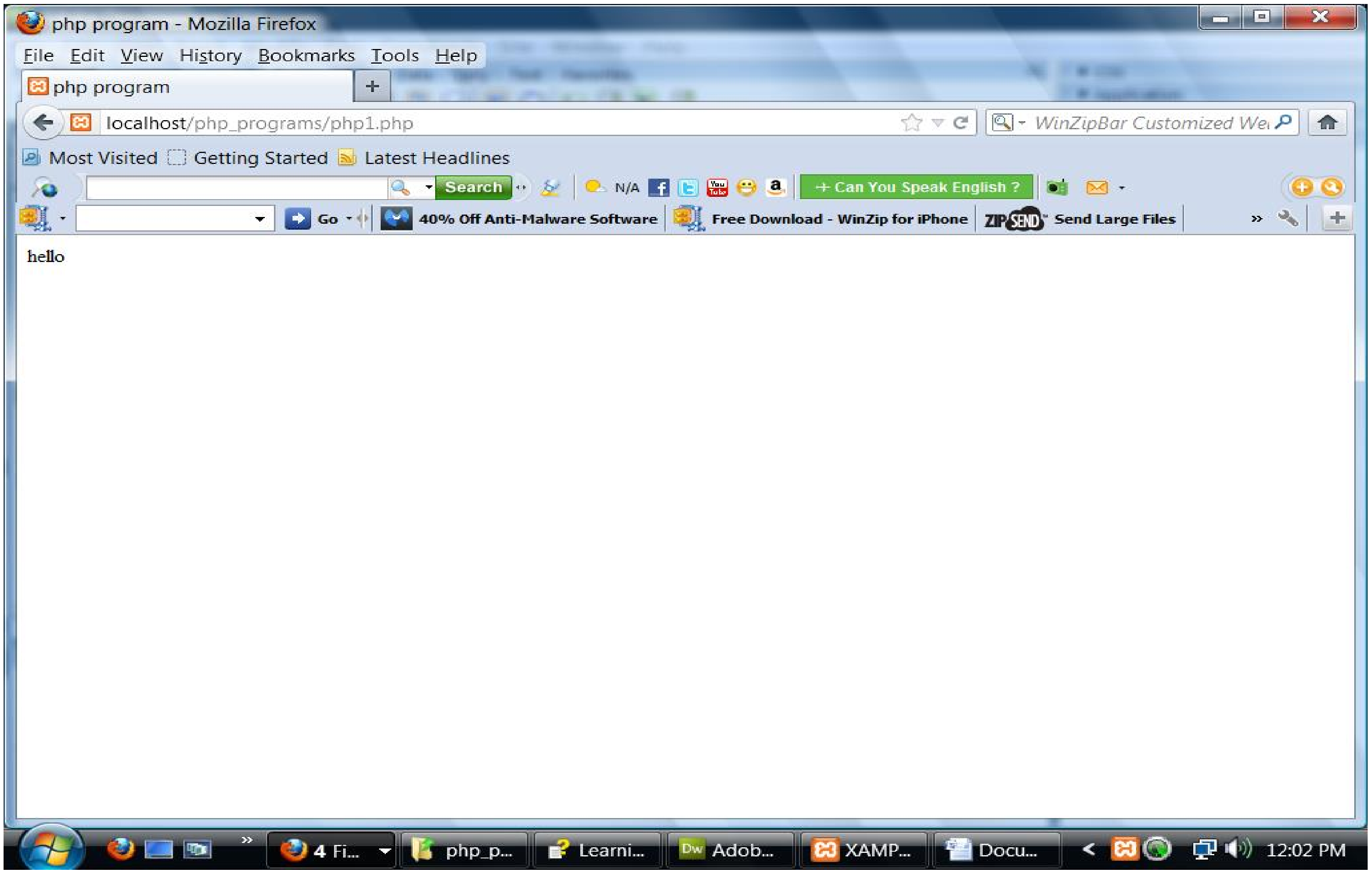
A PHP script starts with **<?php** and ends with **?>**:

- <?php
  // PHP code goes here
  ?>
- we have an example of a simple PHP file, with a PHP script that sends the text "Hello" back to the browser. The php file is saved with the name php1.php in user defined folder say php_programs

```
<html>
<body>
<h1>My first PHP
page</h1>

<?php
echo "Hello";
?>

</body>
</html>
```


php program - Mozilla Firefox
File Edit View History Bookmarks Tools Help
php program
localhost/php_programs/php1.php
Most Visited Getting Started Latest Headlines
Search
+ Can You Speak English ?
40% Off Anti-Malware Software
Free Download - WinZip for iPhone
Send Large Files
12:02 PM

```
hello
```

Fig 1.2 Above figure shows how to run a .php program in web browser

Certain coding practices while using PHP

- PHP statements end with semicolon.
- Do not embed too much of HTML code within PHP code. If we need some more complex HTML code then end the php tag and start writing normal HTML tags.
- We can add single comment line using double forward slashes // and multiple comment lines with /*   */. Commenting facilitates in debugging as well helps in further expansion and modifications in code. Moreover it enhances the readability of code also.
- You can put multiple PHP statements on the same line of a program as long as they are separated with a semicolon.

Certain coding practices while using PHP

- You can put as many blank lines between statements as you want. The PHP interpreter ignores them. The semicolon tells the interpreter that one statement is over and another is about to begin. No whitespace at all or lots and lots of whitespace between statements doesn't affect the program's execution. To the server, the code will be one continuous line, regardless of tabs, indents and line returns.
- In practice, it's good style to put one statement on a line and to put blank lines between statements only when it improves the readability of your source code.
- Make use of consistent formatting style while writing code and write line no. if text editor does not automatically provides for better debugging.

PHP Variables:

ÄAs with algebra, PHP variables can be used to hold values (x=5) or expressions (z=x+y).

ÄVariable can have short names (like x and y) or more descriptive names (age, Firstname, totalcost).

➢**Rules for PHP variables:**

ÄA variable starts with the $ sign, followed by the name of the variable.

ÄA variable name must begin with a letter or the underscore character.

ÄA variable name can only contain alpha-numeric characters and underscores (A-z, 0-9, and _)

ÄA variable name should not contain spaces

ÄVariable names are case sensitive ($y and $Y are two different variables)

Example:

```
<?php
$var1 = 'PHP'; // Assigns a value of 'PHP' to
$var1
$var2 = 5; // Assigns a value of 5 to $var2
$var3 = $var2 + 1; // Assigns a value of 6 to
$var3
$var2 = $var1; // Assigns a value of 'PHP' to
$var2
echo $var1; // Outputs 'PHP‘
echo "<br />";
echo $var2; // Outputs 'PHP'
echo "<br />";
echo $var3; // Outputs '6'
echo "<br />";

echo $var1 . ' rules!'; // Outputs 'PHP rules!'
echo "$var1 rules!"; // Outputs 'PHP rules!'
echo '$var1 rules!'; // Outputs '$var1 rules!‘
?>
```

PHP Variable Scopes

- The scope of a variable is the part of the script where the variable can be referenced/used.
- PHP has four different variable scopes:
- local
- global
- Static
- parameter

Local Scope

Å A variable declared **within** a PHP function is local and can only be accessed within that function.

Global Scope

ÅA variable that is defined outside of any function, has a global scope.
ÅGlobal variables can be accessed from any part of the script, EXCEPT from within a function.
ÅTo access a global variable from within a function, use the **global** keyword.

Static Scope

ÅWhen a function is completed, all of its variables are normally deleted. However, sometimes you want a local variable to not be deleted.
ÅTo do this, use the **static** keyword when you first declare the variable

Local Scope

```
<?php
$x=5; // global
scope
function myTest()
{
$x=3;
echo $x; // local
scope
}
myTest();
?>
```

Global Scope

```
<?php
$x=5; // global scope
$y=10; // global scope
function myTest()
{
global $x,$y;
$y=$x+$y;
}
myTest();
echo $y; // outputs 15
?>
```

Static Scope

```
<?php
function myTest()
{
static $x=0;
echo $x;
$x++;
}
myTest();
myTest();
myTest();
?>
```

Escaping quotes with in quotes

Example 1:

```
<?php
$str = "\"This is a PHP string examples quotes\"";
echo $str;
?>
```

Example 2

```
<?php
$str = 'It\'s a nice day today.';
echo $str;
?>
```

The PHP Concatenation Operator

- The concatenation operator (.) is used to join two strings.

```
<?php
$txt1="Hello  Nikita!";
$txt2="Are you there!";
echo $txt1 . " " . $txt2;
?>
```

// Output will be: Hello Nikita! Are you there!

Defining and Referencing a Function

Syntax

function functionname *( ) {* your code *}*

Example:

```
<html> <body>

<?php
Function Name()
{
echo "Ben John";
}
Name();
?>
</body> </html>
```

PHP Functions - Adding parameters

- A parameter is just like a variable.
- The parameters are specified inside the parentheses.

Syntax:

```
<?php
     function
  function_name(param_1
  , ... , param_n)
     {
          statement_1;
          statement_2;
          ...
          statement_m;

          return
  return_value;
     }
  ?>
```

Functions can also be used to return values.

Example:

```
<html>
<body>

<?php
function add($x,$y)
  {
  $total = $x + $y;
  return $total;
  }

echo "1 + 16 = " .
  add(1,16);
?>

</body>
</html>
```

The PHP strlen() function

Å The strlen() function returns the length of a string, in characters.

```
<?php
echo  strlen("Welcome to
NITTTR!");
?>
```

//The output of the code above will be: 18

The PHP strpos() function

- The strpos() function is used to search for a character or a specific text within a string.
- If a match is found, it will return the character position of the first match.

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

//The output of the code above will be: 6.

Date Function Formatting

- **DAYS**
  d - day of the month 2 digits (01-31)
  j - day of the month (1-31)
  D - 3 letter day (Mon - Sun)
  l - full name of day (Monday - Sunday)
  N - 1=Monday, 2=Tuesday, etc (1-7)
  S - suffix for date (st, nd, rd,th)
  w - 0=Sunday, 1=Monday (0-6)
  z - day of the year (1=365)

- **WEEK**
  W - week of the year (1-52)

- **MONTH**
  F - Full name of month (January - December)
  m - 2 digit month number (01-12)
  n - month number (1-12)
  M - 3 letter month (Jan - Dec)
  t - Days in the month (28-31)

- **YEAR**
  L - leap year (0 no, 1 yes)
  o - ISO-8601 year number (Ex. 1979, 2006)
  Y - four digit year (Ex. 1979, 2006)
  y - two digit year (Ex. 79, 06)

Å **TIME**

a - am or pm
A - AM or PM
B - Swatch Internet time (000 - 999)
g - 12 hour (1-12)
G - 24 hour c (0-23)
h - 2 digit 12 hour (01-12)
H - 2 digit 24 hour (00-23)
i - 2 digit minutes (00-59)
s - 2 digit seconds (00-59)

Using Built-in Function

PHP **Date() function formatting characters:**

Example 1:

```
<?php

$theDate = date("m/d/y");

echo "Today's date is: $theDate";

?>
```

Example 2

```
<?php
print date("m/d/y") . "<br />";

print date("D, F jS") . "<br />";

print date("l, F jSY") . "<br />";

?>
```

Control Structure

- Control structures are the building blocks of any programming language. PHP provides all the control structures that you may have encountered anywhere. The syntax is the same as C or Perl.
- Making computers think has always been the goal of the computer architect and the programmer. Using control structures computers can make simple decisions and when programmed cleverly they can do some complex things.

Conditional Statements

1. **The If...Else Statement**

Syntax

if (*condition*) *code to be executed if condition is true;*

else *code to be executed if condition is false;*

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

If more than one line should be executed if a condition is true/false, the lines should be enclosed within curly braces:

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

Conditional Statements

2. The ElseIf Statement

- If you want to execute some code if one of several conditions is true use the elseif statement

Syntax

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

```
<html><head> <title>good ……</title>
    </head>
<body>
<?php
$hour = date("H");
 if ($hour <= 11)
 {
 echo "good morning my friend";
 }
elseif ($hour > 11 && $hour < 18)
  {
echo "good afternoon my friend";
 }
else  {
  echo "good evening my friend";
  }
?>
</body></html>
```

PHP Switch Statement

- If you want to select one of many blocks of code to be executed, use the Switch statement.
- The switch statement is used to avoid long blocks of if..elseif..else code.

Syntax

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;*

}

```
$textcolor= "black";

switch ($textcolor)
{
case "black":
     echo "My eyes are black";
     break;
case "blue":
     echo " My eyes are blue";
     break;
case “brown”:
     echo " My eyes are brown";
     break;
default:
     echo "too bad!!, My eyes are
     something else";
}
```

PHP Looping

- Looping statements in PHP are used to execute the same block of code a specified number of times.
- In PHP we have the following looping statements:
  - **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

Syntax

```
while (condition)
    {
    // statements
    }
```

Example

```
<html>  <head>  <title>Let us count
   !!!</title></head>
 <body>
<?php
 $limit = 10;
 echo "<h2> Let us count from 1 to $limit </h2><br
   />";
 $count = 1;
 while ($count <= $limit)
 {
  echo "counting $count of $limit <br>";
  $count++;
 }
?>
   </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.

Syntax

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

Example

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

The for Statement

- It is used when you know how many times you want to execute a statement or a list of statements.

Syntax

- for (*init*; *cond*; *incr*) { *code to be executed;* }
  Parameters:

- **init**: Is mostly used to set a counter, but can be any code to be executed once at the beginning of the loop statement.
- **cond**: Is evaluated at beginning of each loop iteration. If the condition evaluates to TRUE, the loop continues and the code executes. If it evaluates to FALSE, the execution of the loop ends.
- **incr**: Is mostly used to increment a counter, but can be any code to be executed at the end of each loop.

Example

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

The foreach Statement

- The foreach statement is used to loop through arrays.
- For every loop, the value of the current array element is assigned to $value (and the array pointer is moved by one) - so on the next loop, you'll be looking at the next element.

Syntax

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

Example

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

PHP Arrays

- An array can store one or more values in a single variable name.
- There are three different kind of arrays:
  - **Numeric array** - An array with a numeric ID key
  - **Associative array** - An array where each ID key is associated with a value
  - **Multidimensional array** - An array containing one or more arrays

Numeric Array

- A numeric array stores each element with a numeric ID key.
- There are different ways to create a numeric array:

Example 1

- In this example the ID key is automatically assigned:
- $names = array("Peter“, "john“ ,"Joe");

Example 2

- In this example we assign the ID key manually:

```
$names[0] = "Peter";
$names[1] = “john";
$names[2] = "Joe";
```

Example 3:

```
<?php
$names[0] = "Peter";
$names[1] = "john";
$names[2] = "Joe";
echo $names[1] . " and " .
    $names[2] . " are ".
    $names[0] . "'s
    neighbors";
?>
```

Associative Arrays

- Each ID key is associated with a value.
- When storing data about specific named values, a numerical array is not always the best way to do it.
- There are two ways of creating Associative Array:

Example 1

- $ages = array("Peter"=>32, “john"=>30, "Joe"=>34);

Example 2

- $ages['Peter'] = "32"; $ages[‘john'] = "30"; $ages['Joe'] = "34";

Example 3:

```
<?php
$ages['Peter'] = "32";
$ages['john']= "30";
$ages['Joe'] = "34";
echo "Peter is " .
    $ages['Peter'] . " 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.

Example 1

- with automatically assigned ID keys:

```
$families = array
( "Griffin"=>array
    ( "Peter", "Lois", "Megan" ),
"Quagmire"=>array
    ( "Glenn" ),
"Brown"=>array
    ( "Cleveland", "Loretta", "Junior" )
);
```

Example 2:

```
$families = array
(
'Griffin' => array
(
0 => 'Peter' ,
1 => 'Lois' ,
2 => 'Megan'
),
'Quagmire' => array
(
0 => 'Glenn'
) ,
'Brown' => array
(
0 => 'Cleveland',
1 => 'Loretta',
2 => 'Junior' )
);

echo "Is " . $families['Griffin'][2] . " a part of the
Griffin family?";
```

THANK YOU