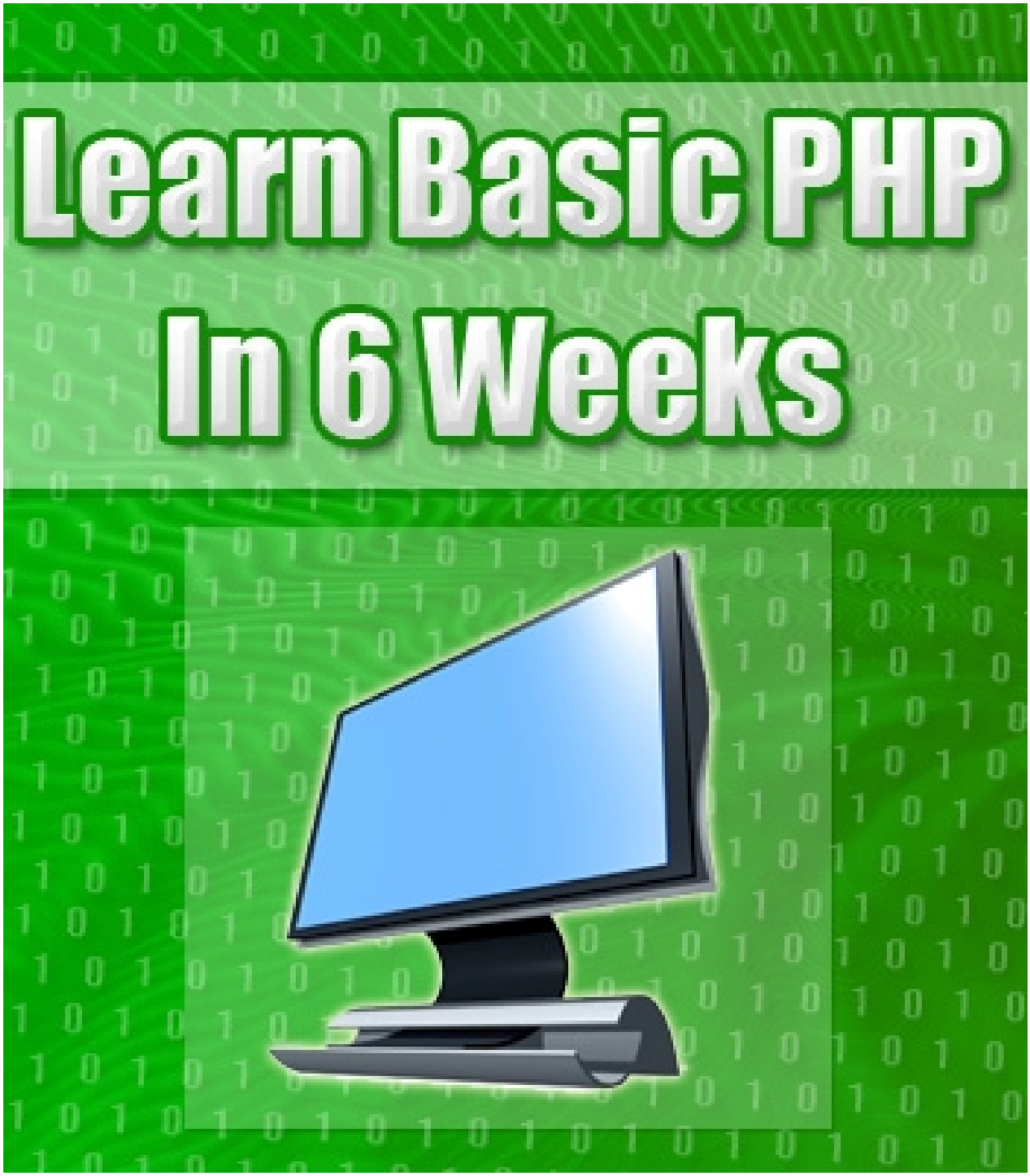

Learn basic PHP in 6 weeks

www.learn-php-tutorial.com
post@learn-php-tutorial.com

www.web-freelancer.com
Web-Freelancer / Flavus Data AS
Grønvoldvn 705
N-3830 Ulefoss
Norway

Table of content

1. INTRODUCTION:

PHP (recursive acronym for "PHP: Hypertext Preprocessor").

PHP started out as a small open source project that evolved as more and more people found out how useful it was. Rasmus Lerdorf unleashed the first version of PHP way back in 1994. Now it has version 4.0.3 with numerous improvements and refinements over the original release.

PHP is a server side scripting language that is embedded in HTML. It is used to manage dynamic content, databases, session tracking, even build entire e-commerce sites. It is integrated with a number of popular databases, including MySQL, PostgreSQL, Oracle, Sybase, Informix, and Microsoft SQL Server.

Common uses of PHP:

- It performs system functions, i.e. from files on a system it can create, open, read, write, and close them.
- It can handle forms, i.e. gather data from files, save data to a file, thru email you can send data, return data to the user.
- You can add, delete, modify elements within your database thru PHP.
- Access cookies variables and set cookies.
- Using PHP, you can restrict users to access some pages of your website.
- It can encrypt data.

Now, we will walk through a few simple examples so you can have an idea. Keep in mind that this is just a quick starter.

"Hello, World!"

To get a feel for PHP, first start with simple PHP scripts. Since "Hello, World!" is an essential example, first we will create a friendly little "Hello, World!" script.

As mentioned earlier, PHP is embedded in HTML. (You could have a file that contains almost no HTML, but usually it's a mixture.) That means that in amongst your normal HTML (or XHTML if you're cutting-edge) you'll have PHP statements like this:

```
<body bgcolor="white">
  <strong>How to say "Hello, World!"</strong>
        <?php echo "Hello, World!";?>
  <br>
It is simple.
```

</body>
It is simple. Just an "echo" statement, and that's it. But it does teach us something about the language. (By the way, if you examine the HTML output, you'll notice that the PHP code is not present in the file sent from the server to your Web browser. All of the PHP present in the Web page is processed and stripped from the page; the only thing returned to the client from the Web server is pure HTML output.)

2. COMMENTING YOUR CODE

It is a very good habit to enter comment in your code. Entering comment in code helps you to keep track of the area and later you can easily debug them (if required). The browser ignores comments in HTML. In HTML we enter comments with in following tags:

<!- - and - >

For example the following comment reminds you that here you have inserted a Flash file.

<!- - Insert Flash file here. - - >

Same case is with PHP. Thru parsing engine, comments are ignored. In PHP we write comments in the following format:

// Your PHP comment goes here

PHP uses other types of commenting such as:

kirang@giasdl01.vsnl.net.in # This is shell-style style comment

and

/* This begins a C-style comments that runs on to
two lines */

If you write comments in your code (HTML or PHP), you do not have to work hard to figure out what your code is trying to do.

3. LANGUAGE REFERENCE

TYPES:

There are eight primitive types in PHP:

- boolean
- integer
- float
- string
- array
- object
- resource
- NULL

VARIABLES:

Variables represent data. For example,

The variable “city” holds the literal value “New York” when appearing in your script as:

$city = “New York”;

Variables begin with a dollar sign ($) and are followed by a concise, meaningful name. The variable cannot begin with a numeric character.

Sometimes it is convenient to be able to have variable variable names. That is, a variable name which can be set and used dynamically. A normal variable is set with a statement such as:

```
<?php
$one = "hi";
?>
```

Variables in PHP are represented by a dollar sign followed by the name of the variable. PHP supports the basic data types of strings, integers, double precision floating point numbers, etc. Variables are not tied to a specific data type, and may take any of the data types.

A variable stored in an area of memory, which is set aside to store information. Variables in PHP can be recognize by a prefixed with the dollar ($) sign . To assign a variable you use the assignment operator (=) . Here is an example of this.

```
$name = "Chris Oak";
$intDaysInWeek = 7;
```

In the above example the variable identifier is $name and the string value "Chris Oak" has been assigned to it. In the second example the variable identifier is $intDaysInWeek and the number 7 is assigned to it. In the days in week example we do not surround the variable with quotes, as PHP treats this as a numeric value but if we had put quotes round it then PHP would have treated it as a string.

Variable Naming

Rules for naming a variable is:

1. Variable names must begin with a letter or underscore character.
2. A variable name can consist of numbers, letters, underscores but you cannot use characters like + , - , % , (,) . & , etc

There is no size limit for variables.

Case Sensitivity

PHP is a case sensitive languages, whereas some languages are not. Here is an example.

```
<?php
$mynameis = "Chris Oak";
echo $Mynameis";
?>
```

Now we want to declare a variable, assign it the value of "Chris Oak" and then print this on the screen but in this example as we have mis-spelt the variable name, when we will run the program, following error will be displayed on the screen.

Warning: Undefined variable: Mynameis in D:\samplecode.php on line 3

Example

Using your HTML editor enter the following

```
<?php
$url = "http://www.mywebsite.com";
$number = 1;
echo "Our favorite site is ".$url;
echo "<br>";
echo "It is number ".$number;
?>
```

You will get the following output:

Our favorite site is http://www.mywebsite.com

It is number 1

CONSTANTS:

A constant is a name or an identifier for a simple value. In constants value cannot change during the execution of the script. By default a constant is case-sensitive by default. By convention, constant identifiers are always uppercase. A constant name starts with a letter or underscore, followed by any number of letters, numbers, or underscores. If you have defined a constant, it can never be changed or undefined.

To retrieve the value of a constant, you have to simply specifying its name. Unlike with variables, you do not need to have a constant with a $. You can also use the function constant() to read a constant's value if you wish to obtain the constant's name dynamically.

constant(): As indicated by the name, this function will return the value of the constant.

constant(): This is useful when you want to retrieve value of a constant, but you do not know its name, i.e. It is stored in a variable or returned by a function.

mixed constant(string $name)

constant() example

```
<?php

define("MINSIZE", 50);

echo MINSIZE;
echo constant("MINSIZE"); // same thing as the previous line

?>
```

Only scalar data (boolean, integer, float and string) can be contained in constants.

Differences between constants and variables are:

- There is no need to write a dollar sign ($) before a constant, where as in Variable one has to write a dollar sign.
- Constants cannot be defined by simple assignment, they may only be defined using the define() function.
- Constants may be defined and accessed anywhere without regard to variable scoping rules.
- Once the Constants have been set, may not be redefined or undefined.

Valid and invalid constant names:

```
<?php

// Valid constant names
define("ONE",     "first thing");
define("TWO2",    "second thing");
define("THREE_3", "third thing")

// Invalid constant names
define("2TWO",    "second thing");

define("__THREE__", "third value");

?>
```

OPERATORS:

Using different types of operators we values are assigned to variables. Different types of operators are as follows:

a. Arithmetic operator:

+, -, *, /, and % are assignment operators.

+ means **addition**
Example: $z = $x + $y;

- means **subtraction**
Example: $z = $x - $y;

***** means **multiplication**
Example: $z = $x * $y;

/ means **division**
Example: $z = $x / $y;

% means **modulus or remainder**
Example: $z = $x % $y;

b. Assignment operator:

The basic assignment operator is (=) sign.

Single equal sign means “assigned to”. It does not mean, “equal to”. Double equal to sign (==) means “equal to”.

Other assignment operators include +=, -=, and .=:

$x += 1 -> It assigns the value of ($x + 1) to x.

$x -= 1 -> It assigns the value of ($x - 1) to x.

$x .= “is Mary” -> It concatenates a string. If $x = “My name ”, then ($x .= “is Mary”) is “My name is Mary”.

c. Comparison operator:

This operator is used to compare to values. Most of the comparison operators are:

== equal to
!= not equal to
> greater than
< less than
>= greater than or equal to
<= less than or equal to

Let us see the following examples. If $x = 4 and $ y = 10, then

Is $x == $y? The comparison is false.
Is $x != $y? The comparison is true.
Is $x > $y? The comparison is false.
Is $x < $y? The comparison is true.
Is $x >= $y? The comparison is false.

Is $x <= $y? The comparison is true. Though 4 is not equal to 10, but 4 is less than 10.

d. Logical operator:

This operator allows your script to determine the status of the conditions. Mostly this operator is used in if…. else and while control statements.
Following are the logical operators:

! -> not
Example: !$x. This means TRUE if $x is not true.

&& -> and
Example: $x && $y. This means TRUE if $x and $y are true.

|| -> or
Example: $x || $y. This means TRUE if either $x or $y is true.

e. Increment or decrement operator:

This operator adds or subtracts from a variable.

Pre-increment: ++$x
It means increment by 1 and returns $x

Post-increment: $x++
It means return $x and then increment $x by 1

Pre-decrement: --$x
It means decrement by 1 and returns $x

Post-decrement: $x--
It means return $x and then decrement $x by 1

FOR AND WHILE LOOP:

Repetitive tasks are always a burden to us. Deleting spam email, sealing 50 envelopes, and going to work are all examples of tasks that are repeated. The nice thing about programming is that you can avoid such repetitive tasks with a little bit of extra thinking. Most often, these repetitive tasks are conquered in the loop.

The idea of a loop is to do something over and over again until the task has been completed. Before we show a real example of when you might need one, let's go over the structure of the PHP while loop.

Simple **While Loop** Example
==========
The function of the while loop is to do a task over and over, as long as a specified conditional statement is true. This logical check is the same as the one that appears in PHP if statements to determine if it is true or false. Here is the basic structure of a PHP while loop

```
$pen_price = 5;
$counter = 10;

echo "<table border=\"1\" align=\"center\">";
echo "<tr><th>Quantity</th>";
echo "<th>Price</th></tr>";
while ( $counter <= 100 ) {
echo "<tr><td>";
echo $counter;
echo "</td><td>";
echo $pen_price * $counter;
echo "</td></tr>";
$counter = $counter + 10;
}
echo "</table>";
```

The **for loop** is simply a while loop with a bit more code added to it. The common tasks that are covered by a for loop are:

Set a counter variable to some initial value.
Check to see if the conditional statement is true.
Execute the code within the loop.
Increment a counter at the end of each iteration through the loop.

The **for loop** allows you to define these steps in one easy line of code. It may seem to have a strange form, but we think you will find the for loop will grow on you the more you use it.

```
$pen_price = 5;

echo "<table border=\"1\" align=\"center\">";
echo "<tr><th>Quantity</th>";
echo "<th>Price</th></tr>";
for ( $counter = 10; $counter <= 100; $counter += 10) {
echo "<tr><td>";
echo $counter;
echo "</td><td>";
echo $pen_price * $counter;
echo "</td></tr>";
}
echo "</table>";
```

5. HANDLING FORMS

Handling HTML Forms and Variables with PHP

First powerful feature of PHP that we are going to introduce here is the way how PHP handles HTML forms and variables from such forms. With these variables, you may accomplish many features, such as: sending web-based email, outputting information in Web browser, pass data to and from a database. In order to demonstrate these capabilities, let's assume we have the following HTML form:

```
---------------------------------------------------------------------------
<html>
<head>
<title>General  Preferences</title>
</head>
<body>
<center>
Would you like to tell us your preferences?
<p>
<table width = 400>
<tr>
<td align = right>
<form action = "action1.php3" method = "POST">
Your name:
<br>
<input type = "text" name = "name" size = "20" maxlength = "30">
<p>
Your age:
<br>
<input type = "text" name = "age" size = "20" maxlength = "30">
<p>
I prefer:
<select name = "preference">
<option value = News>News
<option value = Movies >Movies
<option value = Sports>Sports
<option value = Editor>Editor
</select>
<p>
<input type = "submit" value = "Send">
</form>
</td>
</tr>
```

```
</table>
</center>
</body>
</html>
```

Note, that the action of this form points to a PHP file called action1.php3. This file contains a number of commands that will be executed after submitting the form. In such a script file we may handle the variables from a particular form. These variables are passed to the script. Their names in the script corresponds to the names of input fields from the form. However, names of variables in the script have $ as prefix for the names from the form. In our example we may manipulate the following variables in our script:

$name corresponds to the "name" input field from the form
$age corresponds to the "age" input field from the form
$preference corresponds to the "preference" input field from the form
The action1.php3 script may simply dump the variables, as entered by a particular user:

To get the value entered in the form we first look that in the form what we have defined the method of the form.
If the method is "post" then to get the value we have to write
$HTTP_POST_VARS["field_name"]

and
if it is "Get" then we will write $HTTP_GET_VARS["field_name"].

```
<?
/* this script will handle the variables passed from the action1.html file */
PRINT "<center>";
PRINT "Hi, $name.";
PRINT "<br><br>";
PRINT "You are $age years old and you like $preference.<br><br>";
PRINT "Thank you.";
PRINT "</center>";
?>
```

6. SENDING EMAILS

To send an email in PHP, use PHP's built-in mail() function.

==========
mail (string to, string subject, string message [, string additional_headers [, string additional_parameters]])
==========

You'll notice from the return type of the function that a Boolean is returned upon completion. If PHP successfully passed the email to the SMTP server then true is returned. If an error occurred then false is returned. Please note that even if true is returned the mail may not be sent! (For example if the SMTP is incorrectly configured. In this case you should consult your SMTP server logs to see what went wrong).

To send the email from the previous section - with error checking - we use the following code:

```
<?php
   $to = "Mary@xyz.com";
   $from = "jsss@www.com";
   $subject = "This is a test email";
   $message = "Dear Mary,\n\n\n This is just a test email.\n\n From Chris.";

   $headers  = "From: $from\r\n";

   $success = mail($to, $subject, $message, $headers);
   if ($success)
      echo "The email to $to from $from was successfully sent";
   else
      echo "An error occurred when sending the email to $to from $from";
?>
```

It is also possible to send an email to multiple recipients with a single call to the mail() function. Here is an example. All you need to do is modify the recipient variable as follows:

```
<?php
   $to = " Mary@xyz.com, def@wer.com";
?>
```

7. DATE AND TIME FUNCTIONS

These functions allow you to get the date and time from the server where your PHP scripts are running. You can use these functions to format the date and time in many different ways.

This lesson is a bit long, but look at it more like a reference. You don't have to read every line of code in this chapter.

1.checkdate -- Validate a Gregorian date
==========
bool checkdate (int month, int day, int year)

Returns TRUE if the date given is valid; otherwise returns FALSE. Checks the validity of the date formed by the arguments. A date is considered valid if:

year is between 1 and 32767 inclusive

month is between 1 and 12 inclusive

Day is within the allowed number of days for the given month. Leap years are taken into consideration.

```
<?php
var_dump(checkdate(12, 31, 2000));
var_dump(checkdate(2, 29, 2001));
?>
```

The above example will output:

```
bool(true)
bool(false)
```

==========
2.date -- Format a local time/date
==========

string date (string format [, int timestamp])

Returns a string formatted according to the given format string using the given integer timestamp or the current local time if no timestamp is given.

Note: The valid range of a timestamp is typically from Fri, 13 Dec 1901 20:45:54 GMT to Tue, 19 Jan 2038 03:14:07 GMT. (These are the dates that correspond to the minimum and maximum values for a 32-bit signed integer.)

To generate a timestamp from a string representation of the date, you may be able to use strtotime(). Additionally, some databases have functions to convert their date formats into timestamps (such as MySQL's UNIX_TIMESTAMP function).

The following characters are recognized in the format string:

a - "am" or "pm"

A - "AM" or "PM"

B - Swatch Internet time

d - day of the month, 2 digits with leading zeros; i.e. "01" to "31"

D - day of the week, textual, 3 letters; i.e. "Fri"

F - month, textual, long; i.e. "January"

g - hour, 12-hour format without leading zeros; i.e. "1" to "12"

G - hour, 24-hour format without leading zeros; i.e. "0" to "23"

h - hour, 12-hour format; i.e. "01" to "12"

H - hour, 24-hour format; i.e. "00" to "23"

i - minutes; i.e. "00" to "59"

I (capital i) - "1" if Daylight Savings Time, "0" otherwise.

j - day of the month without leading zeros; i.e. "1" to "31"

l (lowercase 'L') - day of the week, textual, long; i.e. "Friday"

L - boolean for whether it is a leap year; i.e. "0" or "1"

m - month; i.e. "01" to "12"

M - month, textual, 3 letters; i.e. "Jan"

n - month without leading zeros; i.e. "1" to "12"

O - Difference to Greenwich time in hours; i.e. "+0200"
r - RFC 822 formatted date; i.e. "Thu, 21 Dec 2000 16:01:07 +0200" (added in PHP 4.0.4)

s - seconds; i.e. "00" to "59"

S - English ordinal suffix for the day of the month, 2 characters; i.e. "th", "nd"

t - number of days in the given month; i.e. "28" to "31"

T - Timezone setting of this machine; i.e. "MDT"

U - seconds since the epoch

w - day of the week, numeric, i.e. "0" (Sunday) to "6" (Saturday)

W - ISO-8601 week number of year, weeks starting on monday (added in PHP 4.1.0) (Saturday)

Y - year, 4 digits; i.e. "1999"

y - year, 2 digits; i.e. "99"

z - day of the year; i.e. "0" to "365"

Z - timezone offset in seconds (i.e. "-43200" to "43200"). The offset for timezones west of UTC is always negative, and for those east of UTC is always positive.

example
echo date ("l dS of F Y h:i:s A");

==========
3.getdate -- Get date/time information
==========

array getdate ([int timestamp])

Returns an associative array containing the date information of the timestamp, or the current local time if no timestamp is given, as the following array elements:

"seconds" - seconds

"minutes" - minutes

"hours" - hours

"mday" - day of the month

"wday" - day of the week, numeric : from 0 as Sunday up to 6 as Saturday

"mon" - month, numeric

"year" - year, numeric

"yday" - day of the year, numeric; i.e. "299"

"weekday" - day of the week, textual, full; i.e. "Friday"

"month" - month, textual, full; i.e. "January"

Example 1. getdate() example

```
$today = getdate();
$month = $today['month'];
$mday = $today['mday'];
$year = $today['year'];
echo "$month $mday, $year";
```

==========
4.gettimeofday -- Get current time
==========
array gettimeofday (void)

This is an interface to gettimeofday(2). It returns an associative array containing the data

"sec" - seconds

"usec" - microseconds

"minuteswest" - minutes west of Greenwich

"dsttime" - type of dst correction

==========
5.gmdate -- Format a GMT/UTC date/time
==========

string gmdate (string format [, int timestamp])

Identical to the date() function except that the time returned is Greenwich Mean Time (GMT). For example, when run in Finland (GMT +0200), the first line below prints "Jan 01 1998 00:00:00", while the second prints "Dec 31 1997 22:00:00". Example 1. gmdate() example

```
echo date ("M d Y H:i:s", mktime (0,0,0,1,1,1998));
echo gmdate ("M d Y H:i:s", mktime (0,0,0,1,1,1998));
```

==========
6.gmmktime -- Get Unix timestamp for a GMT date
==========

int gmmktime (int hour, int minute, int second, int month, int day, int year [, int is_dst])

==========
7.gmstrftime -- Format a GMT/CUT time/date according to locale settings
==========

string gmstrftime (string format [, int timestamp])

Behaves the same as strftime() except that the time returned is Greenwich Mean Time (GMT). For example, when run in Eastern Standard Time (GMT -0500), the first line below prints "Dec 31 1998 20:00:00", while the second prints "Jan 01 1999 01:00:00". Example 1. gmstrftime() example

```
setlocale ('LC_TIME', 'en_US');
echo strftime ("%b %d %Y %H:%M:%S", mktime (20,0,0,12,31,98))."\n";
echo gmstrftime ("%b %d %Y %H:%M:%S", mktime (20,0,0,12,31,98))."\n";
```

==========
8.localtime -- Get the local time
==========

array localtime ([int timestamp [, bool is_associative]])

The localtime() function returns an array identical to that of the structure returned by the C function call. The first argument to localtime() is the timestamp, if this is not given the

current time is used. The second argument to the localtime() is the is_associative, if this is set to 0 or not supplied than the array is returned as a regular, numerically indexed array. If the argument is set to 1 then localtime() is an associative array containing all the different elements of the structure returned by the C function call to localtime. The names of the different keys of the associative array are as follows:

"tm_sec" - seconds

"tm_min" - minutes

"tm_hour" - hour

"tm_mday" - day of the month

"tm_mon" - month of the year, starting with 0 for January

"tm_year" - Years since 1900

"tm_wday" - Day of the week

"tm_yday" - Day of the year

"tm_isdst" - Is daylight savings time in effect

==========
9.microtime -- Return current Unix timestamp with microseconds
==========

string microtime (void)

Returns the string "msec sec" where sec is the current time measured in the number of seconds since the Unix Epoch (0:00:00 January 1, 1970 GMT), and msec is the microseconds part. This function is only available on operating systems that support the gettimeofday() system call.

Both portions of the string are returned in units of seconds. Example 1. microtime() example

```
function getmicrotime(){
   list($usec, $sec) = explode(" ",microtime());
   return ((float)$usec + (float)$sec);
   }
```

```
$time_start = getmicrotime();

for ($i=0; $i < 1000; $i++){
   //do nothing, 1000 times
   }

$time_end = getmicrotime();
$time = $time_end - $time_start;

echo "Did nothing in $time seconds";
```

==========

10.mktime -- Get Unix timestamp for a date

==========

int mktime (int hour, int minute, int second, int month, int day, int year [, int is_dst])

Warning: Note the strange order of arguments, which differs from the order of arguments in a regular UNIX mktime() call and which does not lend itself well to leaving out parameters from right to left (see below). It is a common error to mix these values up in a script.

Returns the Unix timestamp corresponding to the arguments given. This timestamp is a long integer containing the number of seconds between the Unix Epoch (January 1 1970) and the time specified.

Arguments may be left out in order from right to left; any arguments thus omitted will be set to the current value according to the local date and time.

is_dst can be set to 1 if the time is during daylight savings time, 0 if it is not, or -1 (the default) if it is unknown whether the time is within daylight savings time or not. If it's unknown, PHP tries to figure it out itself. This can cause unexpected (but not incorrect) results.

Note: is_dst was added in 3.0.10.

mktime() is useful for doing date arithmetic and validation, as it will automatically calculate the correct value for out-of-range input. For example, each of the following lines produces the string "Jan-01-1998". Example 1. mktime() example

```
echo date ("M-d-Y", mktime (0,0,0,12,32,1997));
echo date ("M-d-Y", mktime (0,0,0,13,1,1997));
echo date ("M-d-Y", mktime (0,0,0,1,1,1998));
```

echo date ("M-d-Y", mktime (0,0,0,1,1,98));
Year may be a two or four digit value, with values between 0-69 mapping to 2000-2069 and 70-99 to 1970-1999 (on systems where time_t is a 32bit signed integer, as most common today, the valid range for year is somewhere between 1902 and 2037).

The last day of any given month can be expressed as the "0" day of the next month, not the -1 day. Both of the following examples will produce the string "The last day in Feb 2000 is: 29".

==========
11.strftime -- Format a local time/date according to locale settings
==========

string strftime (string format [, int timestamp])

Returns a string formatted according to the given format string using the given timestamp or the current local time if no timestamp is given. Month and weekday names and other language dependent strings respect the current locale set with setlocale().

The following conversion specifiers are recognized in the format string:

%a - abbreviated weekday name according to the current locale

%A - full weekday name according to the current locale

%b - abbreviated month name according to the current locale

%B - full month name according to the current locale

%c - preferred date and time representation for the current locale

%C - century number (the year divided by 100 and truncated to an integer, range 00 to 99)

%d - day of the month as a decimal number (range 01 to 31)

%D - same as %m/%d/%y

%e - day of the month as a decimal number, a single digit is preceded by a space (range ' 1' to '31')

%g - like %G, but without the century.

%G - The 4-digit year corresponding to the ISO week number (see %V). This has the same format and value as %Y, except that if the ISO week number belongs to the previous or next year, that year is used instead.

%h - same as %b

%H - hour as a decimal number using a 24-hour clock (range 00 to 23)

%I - hour as a decimal number using a 12-hour clock (range 01 to 12)

%j - day of the year as a decimal number (range 001 to 366)

%m - month as a decimal number (range 01 to 12)

%M - minute as a decimal number

%n - newline character

%p - either `am' or `pm' according to the given time value, or the corresponding strings for the current locale

%r - time in a.m. and p.m. notation

%R - time in 24 hour notation

%S - second as a decimal number

%t - tab character

%T - current time, equal to %H:%M:%S

%u - weekday as a decimal number [1,7], with 1 representing Monday

Warning
Sun Solaris seems to start with Sunday as 1 although ISO 9889:1999 (the current C standard) clearly specifies that it should be Monday.

%U - week number of the current year as a decimal number, starting with the first Sunday as the first day of the first week

%V - The ISO 8601:1988 week number of the current year as a decimal number, range 01 to 53, where week 1 is the first week that has at least 4 days in the current year, and

with Monday as the first day of the week. (Use %G or %g for the year component that corresponds to the week number for the specified timestamp.)

%W - week number of the current year as a decimal number, starting with the first Monday as the first day of the first week

%w - day of the week as a decimal, Sunday being 0

%x - preferred date representation for the current locale without the time

%X - preferred time representation for the current locale without the date

%y - year as a decimal number without a century (range 00 to 99)

%Y - year as a decimal number including the century

%Z - time zone or name or abbreviation

%% - a literal `%' character

Note: Not all conversion specifiers may be supported by your C library, in which case they will not be supported by PHP's strftime(). This means that %T and %D will not work on Windows.

==========
12.strtotime -- Parse about any English textual datetime description into a Unix timestamp
==========

int strtotime (string time [, int now])

The function expects to be given a string containing an English date format and will try to parse that format into a UNIX timestamp relative to the timestamp given in now, or the current time if none is supplied. Upon failure, -1 is returned.

Because strtotime() behaves according to GNU date syntax, have a look at the GNU manual page titled Date Input Formats. Described there is valid syntax for the time parameter.

Example 1. strtotime() examples

```
echo strtotime ("now"), "\n";
echo strtotime ("10 September 2000"), "\n";
```

```
echo strtotime ("+1 day"), "\n";
echo strtotime ("+1 week"), "\n";
echo strtotime ("+1 week 2 days 4 hours 2 seconds"), "\n";
echo strtotime ("next Thursday"), "\n";
echo strtotime ("last Monday"), "\n";
```

Example 2. Checking for failure

```
$str = 'Not Good';
if (($timestamp = strtotime($str)) === -1) {
    echo "The string ($str) is bogus";
} else {
    echo "$str == ". date('l dS of F Y h:i:s A',$timestamp);
}
```

Note: The valid range of a timestamp is typically from Fri, 13 Dec 1901 20:45:54 GMT to Tue, 19 Jan 2038 03:14:07 GMT. (These are the dates that correspond to the minimum and maximum values for a 32-bit signed integer.)

==========
13.time -- Return current Unix timestamp
==========

int time (void)

Returns the current time measured in the number of seconds since the Unix Epoch (January 1 1970 00:00:00 GMT).

==========

8. PHP INCLUDE FUNCTION

The include() statement includes and evaluates the specified file.

The include statement enables you to include the PHP files that will be parsed and their output placed in the same place as the statement itself. You can also include files that are of different language.

The documentation below also applies to require(). The two constructs are identical in every way except how they handle failure. include() produces a Warning while require() results in a Fatal Error.
Basic include() example

vars.php

```
<?php

$color = 'green';
$fruit = 'apple';

?>
```

test.php

```
<?php

echo "A $color $fruit"; // A

include 'vars.php';

echo "A $color $fruit"; // A green apple

?>
```

If the include occurs inside a function within the calling file, then all of the code contained in the called file will behave as though it had been defined inside that function. So, it will follow the variable scope of that function.

Example including within functions

```
<?php

function foo()
{
   global $color;
```

```
   include 'vars.php';

   echo "A $color $fruit";
}

/* vars.php is in the scope of foo() so     *
 * $fruit is NOT available outside of this  *
 * scope.  $color is because we declared it *
 * as global.                               */

foo();                  // A green apple
echo "A $color $fruit";  // A green

?>
```

9. HOW TO SET UP A MYSQL DATABASE

MySQL is a lightweight, fast database put by T.c.X. because of its speed and simplicity, MySQL is an ideal component to PHP.

To set up the MySQL database containing the grant tables is the part of the MySQL installation process.

Windows distributions contain preinitialized grant tables that are installed automatically. The grant tables on UNIX are populated by the mysql_install_db program. Some installation methods run this program for you. Others require that you execute it manually. The grant tables define the initial MySQL user accounts and their access privileges.

Two accounts are created with a username of root. These are superuser accounts that can do anything. Anyone can connect to the MySQL server as root without a password and be granted all privileges, as the initial root account passwords are empty.

On Windows, one root account is for connecting from the local host and the other allows connections from any host. On Unix, both root accounts are for connections from the local host. Connections must be made from the local host by specifying a hostname of localhost for one account, or the actual hostname or IP number for the other.
Two anonymous-user accounts are created, each with an empty username. The anonymous accounts have no passwords, so anyone can use them to connect to the MySQL server.

On Windows, one anonymous account is for connections from the local host, which has all privileges, just like the root accounts. The other is for connections from any host and has all privileges for the test database or other databases with names that start with test. On Unix, both anonymous accounts are for connections from the local host. Connections must be made from the local host by specifying a hostname of localhost for one account, or the actual hostname or IP number for the other. These accounts have all privileges for the test database or other databases with names that start with test_.
None of the initial accounts have passwords.

You should either assign passwords to the anonymous accounts or else remove them if you want to prevent clients from connecting as anonymous users without a password.

You should assign passwords to the MySQL root accounts.

How to set up passwords for the initial MySQL accounts

The following instructions describe how to set up passwords for the initial MySQL accounts, first for the anonymous accounts and then for the root accounts. Replace ‘’newpwd'' in the examples with the actual password that you want to use. The instructions also cover how to remove the anonymous accounts, should you prefer not to allow anonymous access at all.

You might want to defer setting the passwords until later, so that you don't need to specify them while you perform additional setup or testing. However, be sure to set them before using your installation for any real production work.

To assign passwords to the anonymous accounts, you can use either SET PASSWORD or UPDATE. In both cases, be sure to encrypt the password using the PASSWORD() function.

To use SET PASSWORD in Windows, write:

```
shell> mysql -u root
mysql> SET PASSWORD FOR ''@'localhost' = PASSWORD('newpwd');
mysql> SET PASSWORD FOR ''@'%' = PASSWORD('newpwd');
```

To use SET PASSWORD on Unix, do this:

```
shell> mysql -u root
mysql> SET PASSWORD FOR ''@'localhost' = PASSWORD('newpwd');
mysql> SET PASSWORD FOR ''@'host_name' = PASSWORD('newpwd');
```

In the second SET PASSWORD statement, replace host_name with the name of the server host. This is the name that is specified in the Host column of the non-localhost record for root in the user table. If you don't know what hostname this is, issue the following statement before using SET PASSWORD:

```
mysql> SELECT Host, User FROM mysql.user;
```

Look for the record that has root in the User column and something other than localhost in the Host column. Then use that Host value in the second SET PASSWORD statement.

The other way to assign passwords to the anonymous accounts is by using UPDATE to modify the user table directly. Connect to the server as root and issue an UPDATE statement that assigns a value to the Password column of the appropriate user table

records. The procedure is the same for Windows and Unix. The following UPDATE statement assigns a password to both anonymous accounts at once:

```
shell> mysql -u root
mysql> UPDATE mysql.user SET Password = PASSWORD('newpwd')
    ->     WHERE User = '';
mysql> FLUSH PRIVILEGES;
```

After you update the passwords in the user table directly using UPDATE, you must tell the server to re-read the grant tables with FLUSH PRIVILEGES. Otherwise, the change will go unnoticed until you restart the server.

If you prefer to remove the anonymous accounts instead, do so as follows:

```
shell> mysql -u root
mysql> DELETE FROM mysql.user WHERE User = '';
mysql> FLUSH PRIVILEGES;
```

The DELETE statement applies both to Windows and to Unix. On Windows, if you want to remove only the anonymous account that has the same privileges as root, do this instead:

```
shell> mysql -u root
mysql> DELETE FROM mysql.user WHERE Host='localhost' AND User='';
mysql> FLUSH PRIVILEGES;
```

This account allows anonymous access but has full privileges, so removing it improves security.

You can assign passwords to the root accounts in several ways. The following discussion demonstrates three methods:

Use the SET PASSWORD statement
Use the mysqladmin command-line client program
Use the UPDATE statement
To assign passwords using SET PASSWORD, connect to the server as root and issue two SET PASSWORD statements. Be sure to encrypt the password using the PASSWORD() function.

For Windows, do this:

```
shell> mysql -u root
mysql> SET PASSWORD FOR 'root'@'localhost' = PASSWORD('newpwd');
```

```
mysql> SET PASSWORD FOR 'root'@'%' = PASSWORD('newpwd');
```

For Unix, do this:

```
shell> mysql -u root
mysql> SET PASSWORD FOR 'root'@'localhost' = PASSWORD('newpwd');
mysql> SET PASSWORD FOR 'root'@'host_name' = PASSWORD('newpwd');
```

In the second SET PASSWORD statement, replace host_name with the name of the server host. This is the same hostname that you used when you assigned the anonymous account passwords.

To assign passwords to the root accounts using mysqladmin, execute the following commands:

```
shell> mysqladmin -u root password "newpwd"
shell> mysqladmin -u root -h host_name password "newpwd"
```

These commands apply both to Windows and to Unix. In the second command, replace host_name with the name of the server host. The double quotes around the password are not always necessary, but you should use them if the password contains spaces or other characters that are special to your command interpreter.

If you are using a server from a very old version of MySQL, the mysqladmin commands to set the password will fail with the message parse error near 'SET password'. The solution to this problem is to upgrade the server to a newer version of MySQL.

You can also use UPDATE to modify the user table directly. The following UPDATE statement assigns a password to both root accounts at once:

```
shell> mysql -u root
mysql> UPDATE mysql.user SET Password = PASSWORD('newpwd')
    ->     WHERE User = 'root';
mysql> FLUSH PRIVILEGES;
```

The UPDATE statement applies both to Windows and to Unix.

After the passwords have been set, you must supply the appropriate password whenever you connect to the server. For example, if you want to use mysqladmin to shut down the server, you can do so using this command:

```
shell> mysqladmin -u root -p shutdown
Enter password: (enter root password here)
```

10. HOW TO SET UP TABLES IN THE INTERFACE

In our interface we have 4 buttons and 1 screen to the right of those buttons. The screen spans the width of all 4 rows of buttons.

Start out by pasting the code generated into your html editor:

```
<html><head><title>::Tables in the Interface::</title>
<script language="JavaScript">
<!--

if (document.images) {
  image1on = new Image();
  image1on.src = "0.jpg";

  image2on = new Image();
  image2on.src = "1.jpg";

  image3on = new Image();
  image3on.src = "2.jpg";

  image4on = new Image();
  image4on.src = "3.jpg";

  image1off = new Image();
  image1off.src = "off0.jpg";

  image2off = new Image();
  image2off.src = "off1.jpg";

  image3off = new Image();
  image3off.src = "off2.jpg";

  image4off = new Image();
  image4off.src = "off3.jpg";

  otherImageDefault = new Image();
  otherImageDefault.src = "off1.jpg";

  otherImage1 = new Image();
  otherImage1.src = "hist1.jpg";

  otherImage2 = new Image();
```

```
  otherImage2.src = "type1.jpg";

  otherImage3 = new Image();
  otherImage3.src = "bene1.jpg";

  otherImage4 = new Image();
  otherImage4.src = "link1.jpg";

}

function changeImages() {
  if (document.images) {
    for (var i=0; i<changeImages.arguments.length; i+=2) {
      document[changeImages.arguments[i]].src = eval(changeImages.arguments[i+1] +
".src");
    }
  }
}

// -->
</script>
</head>

<body>
****TABLE WILL GO HERE****

<a href="history.htm" onMouseOver="changeImages('image1', 'image1on', 'otherImage',
'otherImage1')" onMouseOut="changeImages('image1', 'image1off', 'otherImage',
'otherImageDefault')"><img name="image1" src="off0.jpg" alt="" width=118 height=77
border=0></a>
<a href="types.htm" onMouseOver="changeImages('image2', 'image2on', 'otherImage',
'otherImage2')" onMouseOut="changeImages('image2', 'image2off', 'otherImage',
'otherImageDefault')"><img name="image2" src="off1.jpg" alt="" width=118 height=76
border=0></a>
<a href="benefits.htm" onMouseOver="changeImages('image3', 'image3on',
'otherImage', 'otherImage3')" onMouseOut="changeImages('image3', 'image3off',
'otherImage', 'otherImageDefault')"><img name="image3" src="off2.jpg" alt="" width=118
height=75 border=0></a>
<a href="links.htm" onMouseOver="changeImages('image4', 'image4on', 'otherImage',
'otherImage4')" onMouseOut="changeImages('image4', 'image4off', 'otherImage',
'otherImageDefault')"><img name="image4" src="off3.jpg" alt="" width=118 height=72
border=0></a>
```

```
<img name="otherImage" src="link1.jpg" alt="" width=182 height=300 border=0>

</body>
</html>
```

This should end up looking like this:

We have mouseovers, still need to get them built into a table. What we will do is build our table, into our html, and just cut and paste the JavaScript into it...
So start out with your basic table tags, and put them between the <body> tag and the first <a href> tag.

```
<table>
<tr> ***row 1***
<td> </td> ***History button***
<td> </td> ***Default Screen***
</tr>
<tr> ***row 2***
<td> </td> ***Types button***
</tr>
<tr> ***row 3***
<td> </td> ***Benefits button***
</tr>
<tr> ***row 4***
<td> </td> ***Links button***
</tr>
</table>
```

Now we must tell our table how to act. So we will add the BLUE parts....

```
<table border="0" cellpadding="0" cellspacing="0">
<tr> ***row 1***
<td> </td> ***History button***
<td rowspan="4"> </td> ***Default Screen***
</tr>
<tr> ***row 2***
<td> </td> ***Types button***
</tr>
<tr> ***row 3***
<td> </td> ***Benefits button***
</tr>
<tr> ***row 4***
<td> </td> ***Links button***
```

```
</tr>
</table>
```

Now we just need to cut and paste the Javascript into our table. First of all, leave the part within the <script> </script> tags alone. That stays put. The part we'll be moving are the <a href> tags and the screen image, which I have just copied and pasted (and now color-coded) from above....

```
<a href="history.htm" onMouseOver="changeImages('image1', 'image1on',
'otherImage', 'otherImage1')" onMouseOut="changeImages('image1', 'image1off',
'otherImage', 'otherImageDefault')"><img name="image1" src="off0.jpg" alt=""
width=118 height=77 border=0></a>
<a href="types.htm" onMouseOver="changeImages('image2', 'image2on',
'otherImage', 'otherImage2')" onMouseOut="changeImages('image2', 'image2off',
'otherImage', 'otherImageDefault')"><img name="image2" src="off1.jpg" alt=""
width=118 height=76 border=0></a>
<a href="benefits.htm" onMouseOver="changeImages('image3', 'image3on',
'otherImage', 'otherImage3')" onMouseOut="changeImages('image3', 'image3off',
'otherImage', 'otherImageDefault')"><img name="image3" src="off2.jpg" alt=""
width=118 height=75 border=0></a>
<a href="links.htm" onMouseOver="changeImages('image4', 'image4on',
'otherImage', 'otherImage4')" onMouseOut="changeImages('image4', 'image4off',
'otherImage', 'otherImageDefault')"><img name="image4" src="off3.jpg" alt=""
width=118 height=72 border=0></a>

<img name="otherImage" src="link1.jpg" alt="" width=182 height=300 border=0>
```

Move the parts in PINK, GREEN, PURPLE, ORANGE, and RED above to the following spots within your table....

```
<table border="0" cellpadding="0" cellspacing="0">
<tr>
<td><a href="history.htm" onMouseOver="changeImages('image1', 'image1on',
'otherImage', 'otherImage1')" onMouseOut="changeImages('image1', 'image1off',
'otherImage', 'otherImageDefault')"><img name="image1" src="off0.jpg" alt=""
width=118 height=77 border=0></a></td>
<td rowspan="4"><img name="otherImage" src="link1.jpg" alt="" width=182
height=300 border=0></td>
</tr>
<tr>
<td><a href="types.htm" onMouseOver="changeImages('image2', 'image2on',
'otherImage', 'otherImage2')" onMouseOut="changeImages('image2', 'image2off',
'otherImage', 'otherImageDefault')"><img name="image2" src="off1.jpg" alt=""
width=118 height=76 border=0></a></td>
</tr>
<tr>
```

```
<td><a href="benefits.htm" onMouseOver="changeImages('image3', 'image3on',
'otherImage', 'otherImage3')" onMouseOut="changeImages('image3', 'image3off',
'otherImage', 'otherImageDefault')"><img name="image3" src="off2.jpg" alt=""
width=118 height=75 border=0></a></td>
</tr>
<tr>
<td><a href="links.htm" onMouseOver="changeImages('image4', 'image4on',
'otherImage', 'otherImage4')" onMouseOut="changeImages('image4', 'image4off',
'otherImage', 'otherImageDefault')"><img name="image4" src="off3.jpg" alt=""
width=118 height=72 border=0></a></td>
</tr>
</table>
```

Once you have your table completed (it should now be in between your <body> and </body> tags), you are now ready to test your html. Save your changes, and open up in your browser.

11. COOKIES AND SESSIONS

Cookies

PHP transparently supports HTTP cookies. Cookies are a mechanism for storing data in the remote browser and thus tracking or identifying return users. You can set cookies using the setcookie() or setrawcookie() function. Cookies are part of the HTTP header, so setcookie() must be called before any output is sent to the browser.

Session support in PHP consists of a way to preserve certain data across subsequent accesses. This enables you to build more customized applications and increase the appeal of your web site.

A visitor accessing your web site is assigned an unique id, the so-called session id. This is either stored in a cookie on the user side or is propagated in the URL.

The session support allows you to register arbitrary numbers of variables to be preserved across requests. When a visitor accesses your site, PHP will check automatically (if session.auto_start is set to 1) or on your request (explicitly through session_start() or implicitly through session_register()) whether a specific session id has been sent with the request. If this is the case, the prior saved environment is recreated.

There is a relationship between Sessions and Cookies -- they serve somewhat the same purpose, and are, to a certain extent, usable interchangeably. Sessions, which were integrated into PHP in version 4 of the language, are a means to store and track data for a user while they travel through a series of pages, or page iterations, on your site.

The most significant differences between the two are that cookies are stored on the client, while the session data is stored on the server. As a result, sessions are more secure than cookies (no information is being sent back and forth between the client and the server) and sessions work even when the user has disabled cookies in their browser. Cookies, on the other hand, can be used to track information even from one session to another by setting it's time( ) parameter (see http://www.htmlgoodies.com/php/p14cookies.html)

How Sessions Work

Sessions in PHP are started by using the session_start( ) function. Like the setcookie( ) function, the session_start( ) function must come before any HTML, including blank lines, on the page. It will look like this:

```
<?php
session_start( );
```

```
?>
<html>
<head> ..... etc
```

The session_start( ) function generates a random Session Id and stores it in a cookie on the user's computer (this is the only session information that is actually stored on the client side.) The default name for the cookie is PHPSESSID, although this can be changed in the PHP configuration files on the server (most hosting companies will leave it alone, however.) To reference the session Id in you PHP code, you would therefore reference the variable $PHPSESSID (it's a cookie name; remember that from Cookies?)

Your sharp mind may be wondering what happens when you come to the second pass through your page and reach the session_start( ) function again. PHP knows that there is already a session on progress and so ignores subsequent instances of the session_start( ) -- phew!!

Using Session Data

Having established a session, you can now create, store and retrieve information pertaining to that session. You might want, for example, to keep track of items in your visitor's shopping cart. Information for sessions is stored in a special directory on the server; the path of that directory is specified in the server's PHP configuration files.

Information to be stored for a session is kept in session variables. Session variables are created by registering them for the session, using the session_register( ) function. To use that information (on any page iteration in the session) you simply reference the variable just like you would any other variable. Here's an example:

```
<?php
session_start( );
?>
<html>
<head>
<title>Using a session variable</title>
</head>
<body>
<?php
print "Welcome to session number: ";
print $PHPSESSID;
?>
 <br />
```

```
<?php
session_register("username");
$username = "Goody";
print "Your name is: ";
print $username;
?>
</body>
</html>
```

In this example we have created a session and displayed the session number. We then registered a session variable called username (notice the quotes around the variable's name in the call to the session_register( ) function.)

12. HOW TO CREATE TABLE THROUGH PHP

PHP has built-in functions for easily handling database connections. One such database is MySQL. MySQL is very popular in the *nix world for its generous license, and in many cases it is free.MySQL does not offer many of the features many commercial database servers have (sub-queries, transactions), its best attributes are speed and light weight on system resources.

The following php program will demonstrate:

1.opening a MySQL connection,
2.selecting a database,
3.adding a table,
4.inserting records,
5.finding records,
6.updating records,
7.deleting records,
8.deleting a table,
9.closing a database connection

Normally tables will not be created every time you access a database, but is done at least once for initialization.

```
<HTML>
<HEAD>
<TITLE><?php echo $TITLE ?></TITLE>
</HEAD>
<BODY>
<CENTER><BIG><B>
<?php echo $TITLE ?>
<BR><BR>

<?php

# Open MySQL connection.

$hostname = "localhost";
$username = "youruserid"; //generally when we are working offline we did not

give username as root and                    password="".
$password = "yourpassword";
$dbName = "demo";
```

```
if($myDB = mysql_connect("$hostname","$username","$password",))
{ echo "Connection to MySQL server 'localhost' successful!<br>\n"; }

# "localhost" can be replaced with an IP address or a domain name.
# If your MySQL server is not using the standard MySQL port,
# add a port (example localhost:7777).
# $myDB saves the present connection.

mysql_select_db("yourdatabasename",$myDB);
echo "Database selected<br>\n";

$query = "CREATE TABLE test (";
$query .= "id INT NOT NULL PRIMARY KEY AUTO_INCREMENT,";
$query .= "date DATE);";
if(mysql_query($query,$myDB))
{ echo "Table 'test' created<br>\n"; }

# $query contains the SQL statements that will be executed by MySQL

$query = "INSERT INTO test (id,date) VALUES(NULL,\"";
$query .= date("Y-m-d");
$query .= "\");";
if(mysql_query($query,$myDB))
{ echo "Insert successful<br>\n"; }

# $query contains the SQL statements that will be executed by MySQL

$query = "INSERT INTO test (id,date) VALUES(NULL,\"";
$query .= date("Y-m-d");
$query .= "\");";
if(mysql_query($query,$myDB))
{ echo "Insert successful<br>\n"; }

# NULL is required for the id, because it will be inserted automatically.
# date command formats a date from current time.
# Y indicates a 4-digit year
# m indicates a 2-digit month
# d indicates a 2-digit day

$id = mysql_insert_id();
echo $id . " is the id number.<br>\n";

# mysql_insert_id retrieves the id value generated in the
```

```
# sql insert statement.
# For PHP4 compatibility, leave the contents of the mysql_insert_id
# function blank.  Adding a database identifier will create an error
# in PHP4.

$query = "SELECT * FROM test;";
$result = mysql_query($query,$myDB);
while($row = mysql_fetch_array($result,1))
{ echo "id = " . $row['id'] . "; date = " . $row['date'] . ";<br>\n"; }

# This is a very basic search/find statement that loops through the
# contents of 'test' table.

# $result stores the query affected instance of the database.
# mysql_fetch_array retrieves a record.
# The "1" option returns an array with real name indexes.
# A "2" option will return an array with number indexes.
# A "3" option will return an array with both real name and number indexes.
# $row stores the array/record.

$query = "UPDATE FROM test SET date = \"1995-01-10\" WHERE id = 1;";
if(mysql_query($query,$myDB))
{ echo "Record updated;<br>\n"; }

$query = "SELECT * FROM test;";
$result = mysql_query($query,$myDB);
while($row = mysql_fetch_array($result,1))
{ echo "id = " . $row['id'] . "; date = " . $row['date'] . ";<br>\n"; }

$query = "DELETE FROM test WHERE id = 1;";
if(mysql_query($query,$myDB))
{ echo "Record deleted<br>\n"; }

$query = "SELECT * FROM test;";
$result = mysql_query($query,$myDB);
while($row = mysql_fetch_array($result,1))
{ echo "id = " . $row['id'] . "; date = " . $row['date'] . ";<br>\n"; }

$query = "DROP TABLE test;";
if(mysql_query($query,$myDB))
{ echo "Table 'test' deleted<br>\n"; }

mysql_close($myDB);
```

```
echo "The MySQL database connection is now closed<br>\n";

# After going through a doorway, you close the door after yourself...
# Technically PHP should automatically close the database connection at
# the end of the script, but I like to re-emphasize that the connection
# is being closed.

?>

</B></BIG></CENTER>
</BODY>
</HTML>
```

13. STEP-BY-STEP EXPLANATION AND SOURCE CODE FOR A SIMPLE BLOG-SITE OR CMS (CONTENT MANAGEMENT SYSTEM)

What is a blog

A blog is basically a journal or simple content management system on the web. An simple solution to easily update content regularly on a website, without having to open FrontPage or Dreamweaver and create a new static webpage every single time.

The activity of updating a blog is called "blogging" and someone who has a blog is a "blogger." Postings on a blog are almost always arranged in sequential order with the most recent additions featured most prominently.

Let us pull together what we have learnt in this PHP-tutorial by making a simple blog solution together.

Let us first create the following files:

1. For adding the data to the databse (addstudent.php)
2. Editing the data.(editstudent.php)
3. View all the records under that table.(viewstudent.php)

First of all create a database with the name test, and under that database create

a table with the name students.

```
CREATE TABLE `Student` (
  `slno` int(11) NOT NULL auto_increment,
  `student_name` varchar(50) ,
  `address` text,
  `phone_number` varchar(50),
  `emailid` varchar(50),
  'created_date' date
    PRIMARY KEY  (`slno`)
) ;
```

Above code has to be written in the mysql.

Steps are as follows:

databaseconnection.php

```
<?

$hostname = "localhost";
$username = "root";
$password = "";
$dbName = "test1";
mysql_connect($hostname,$username,$password) or die(mysql_error());
mysql_select_db($dbName) or DIE("Table unavailable");
?>
```

==========

index.php

```
<html>
<head>

	<title>Student</title>

</head>
<STYLE type=text/css>
BODY {
	FONT-SIZE: 10pt; COLOR: #000000; FONT-FAMILY: Arial, Helvetica, sans-serif;
BACKGROUND-COLOR: #ffffff
}
A:link {
	COLOR: #000099; TEXT-DECORATION: none
}
A:visited {
	COLOR: #660099; TEXT-DECORATION: none
}
A:active {
	COLOR: #000099; TEXT-DECORATION: none
}
A:hover {
	COLOR: red; TEXT-DECORATION: none
}
ADDRESS {
	FONT-SIZE: 10pt; COLOR: #000000; FONT-STYLE: italic; FONT-FAMILY: Arial,
Helvetica, sans-serif
}
DL {
	FONT-SIZE: 10pt; COLOR: #000000; FONT-FAMILY: Arial, Helvetica, sans-serif
}
OL {
	FONT-SIZE: 10pt; COLOR: #000000; FONT-FAMILY: Arial, Helvetica, sans-serif
}
TD {
	FONT: 9pt Verdana, Arial, Helvetica, sans-serif
```

```
}
TH {
	FONT: 9pt Verdana, Arial, Helvetica, sans-serif
}
UL {
	FONT-SIZE: 10pt; COLOR: #000000; FONT-FAMILY: Arial, Helvetica, sans-serif
}
P {
	FONT-SIZE: 10pt; COLOR: #000000; FONT-FAMILY: Arial, Helvetica, sans-serif
}
.caption {
	FONT-WEIGHT: bold; FONT-SIZE: 9pt; COLOR: #000000; FONT-FAMILY: Arial,
Helvetica, sans-serif
}
.date {
	FONT-SIZE: 8pt; COLOR: #000000; FONT-FAMILY: Arial, Helvetica, sans-serif
}
.emphasis {
	FONT-WEIGHT: bold; FONT-SIZE: 10pt; COLOR: #000000; FONT-STYLE: italic; FONT-
FAMILY: Arial, Helvetica, sans-serif
}
.location {
	FONT-SIZE: 8pt; COLOR: #ff6600; FONT-FAMILY: Arial, Helvetica, sans-serif
}
.mousetype {
	FONT-SIZE: 8pt; COLOR: #000000; FONT-FAMILY: Arial, Helvetica, sans-serif
}
.navigation {
	FONT-SIZE: 9pt; COLOR: #000000; FONT-FAMILY: Arial, Helvetica, sans-serif
}
.smalltext {
	FONT-SIZE: 9pt; COLOR: #000000; FONT-FAMILY: Arial, Helvetica, sans-serif
}
.subtitle {
	FONT-WEIGHT: bold; FONT-SIZE: 12pt; COLOR: #000000; FONT-FAMILY: Arial,
Helvetica, sans-serif
}
.text {
	FONT-SIZE: 10pt; COLOR: #000000; FONT-FAMILY: Arial, Helvetica, sans-serif
}
.title {
	FONT-WEIGHT: bold; FONT-SIZE: 14pt; COLOR: #ff9900; FONT-FAMILY: Arial,
Helvetica, sans-serif
}
</STYLE>
<body bottommargin="0" leftmargin="0" rightmargin="0" topmargin="0">
<?include("databaseconnection.php");
?>
<SCRIPT language=JavaScript src="jsfile/mmenu.js" type=text/javascript></SCRIPT>
<script language=JavaScript src="jsfile/autotab.js"></script>
<link rel="stylesheet" href="css/main.css" type="text/css">
<link rel="stylesheet" href="css/colours.css" type="text/css">
<table width="100%" align="center" cellpadding="0" cellspacing="0" border="0">
	<tr><td bgcolor="#3a5bae">
	<table width=100% cellspacing=0 cellpadding=0 border=0 bgcolor="#3a5bae">
	<tr bgcolor="#3a5bae">
```

```
		<td width=229 bgcolor="#3a5bae"> </td>
		<td align="center" valign="middle" bgcolor="#3a5bae"></td>
		</tr>
		</table>
		</td></tr>
		<tr><td valign="top" height="100%" class="tablebody">
		<table width="100%" align="center" cellpadding="0" cellspacing="0" border="0">
		<tr>
			<td width="23%"  bgcolor="#91adf5"> </td>
			<td width="77%" valign="top" class="tablebody">
			<table align="center" width="100%" border=0 height=350>
			<tr>
				<td valign="middle" align="center"  class="headertext" height="20">
				</td>
			</tr>
			<tr>
			<td valign="top" align="center">
<?
$sqltot = "select * from student order by created_date limit 1,10";
$result = mysql_query($sqltot) or die("database eror2");
$TotalRecord=mysql_num_rows($result);

?>
<form name="form1" action="index.php" method="post">

<table align="center" width="100%" height="290">
<tr><td align="center"><b>View Student</b></td></tr>
<tr>
		<td valign="top" align="center">
		<table border=0 cellspacing=1 cellpadding=1 width=100%>
    <?if($TotalRecord != 0)
		 {
		 ?>
		 <tr bgcolor="Aqua">
		<td width=10% class=tableviewheader align="center"><span
class=tableheadfont> <b>Edit</b></span></td>
		<td width=20% class=tableviewheader align="center"><span
class=tableheadfont> <b>Student Name</b></span></td>
		<td width=40% class=tableviewheader align="center"><span
class=tableheadfont> <b>Phone Number</b></span></td>
		<td width=10% class=tableviewheader align=center nowrap><span
class=tableheadfont> <b>Email id</b></span></td>
   <td width=10% class=tableviewheader align=center nowrap><span
class=tableheadfont> <b>Created Date</b></span></td>
		</tr>
		 <?
		 $kk=0;
		 for($i=0;$i<$TotalRecord;$i++)
		{
		$slno=trim(mysql_result($result,$i,"slno"));
		$student_name=trim(mysql_result($result,$i,"student_name"));
		$phone_number=trim(mysql_result($result,$i,"phone_number"));
		$emailid=trim(mysql_result($result,$i,"emailid"));
		$create_date=trim(mysql_result($result,$i,"created_date"));
		$j=$i+1;
		 ?>
```

```
    <tr>
            <td  width=10% align="center"><span
class=tableBodyfont> <?=$j?></span></td>
     <td  width=20% align="center"><span class=tableBodyfont> <? echo
$student_name;?></span></td>
            <td  width=40% align="center"><span class=tableBodyfont> <? echo
$phone_number;?></span></td>
            <td  width=10% align="center"><span class=tableBodyfont> <? echo
$emailid;?></span></td>
            <td  width=10% align="center"><span class=tableBodyfont> <? echo
$create_date;?></span></td>
        </tr>
          <?}?>

          <?}?>
           </table>
          </td>
</tr>
</table>
</form>
          </td>
</tr>
</table>
          </td>
          </tr>
          </table>
          </td></tr>
          <tr><td bgcolor="#3a5bae"> </td></tr>
</table>
</body>
</html>
```

==========

archive.php

```
<html>
<head>

          <title>Student</title>

</head>
<STYLE type=text/css>
BODY {
          FONT-SIZE: 10pt; COLOR: #000000; FONT-FAMILY: Arial, Helvetica, sans-serif;
BACKGROUND-COLOR: #ffffff
}
A:link {
          COLOR: #000099; TEXT-DECORATION: none
}
A:visited {
          COLOR: #660099; TEXT-DECORATION: none
}
A:active {
```

```
	COLOR: #000099; TEXT-DECORATION: none
}
A:hover {
	COLOR: red; TEXT-DECORATION: none
}
ADDRESS {
	FONT-SIZE: 10pt; COLOR: #000000; FONT-STYLE: italic; FONT-FAMILY: Arial,
Helvetica, sans-serif
}
DL {
	FONT-SIZE: 10pt; COLOR: #000000; FONT-FAMILY: Arial, Helvetica, sans-serif
}
OL {
	FONT-SIZE: 10pt; COLOR: #000000; FONT-FAMILY: Arial, Helvetica, sans-serif
}
TD {
	FONT: 9pt Verdana, Arial, Helvetica, sans-serif
}
TH {
	FONT: 9pt Verdana, Arial, Helvetica, sans-serif
}
UL {
	FONT-SIZE: 10pt; COLOR: #000000; FONT-FAMILY: Arial, Helvetica, sans-serif
}
P {
	FONT-SIZE: 10pt; COLOR: #000000; FONT-FAMILY: Arial, Helvetica, sans-serif
}
.caption {
	FONT-WEIGHT: bold; FONT-SIZE: 9pt; COLOR: #000000; FONT-FAMILY: Arial,
Helvetica, sans-serif
}
.date {
	FONT-SIZE: 8pt; COLOR: #000000; FONT-FAMILY: Arial, Helvetica, sans-serif
}
.emphasis {
	FONT-WEIGHT: bold; FONT-SIZE: 10pt; COLOR: #000000; FONT-STYLE: italic; FONT-
FAMILY: Arial, Helvetica, sans-serif
}
.location {
	FONT-SIZE: 8pt; COLOR: #ff6600; FONT-FAMILY: Arial, Helvetica, sans-serif
}
.mousetype {
	FONT-SIZE: 8pt; COLOR: #000000; FONT-FAMILY: Arial, Helvetica, sans-serif
}
.navigation {
	FONT-SIZE: 9pt; COLOR: #000000; FONT-FAMILY: Arial, Helvetica, sans-serif
}
.smalltext {
	FONT-SIZE: 9pt; COLOR: #000000; FONT-FAMILY: Arial, Helvetica, sans-serif
}
.subtitle {
	FONT-WEIGHT: bold; FONT-SIZE: 12pt; COLOR: #000000; FONT-FAMILY: Arial,
Helvetica, sans-serif
}
.text {
	FONT-SIZE: 10pt; COLOR: #000000; FONT-FAMILY: Arial, Helvetica, sans-serif
```

```
}
.title {
	FONT-WEIGHT: bold; FONT-SIZE: 14pt; COLOR: #ff9900; FONT-FAMILY: Arial,
Helvetica, sans-serif
}
</STYLE>
<body bottommargin="0" leftmargin="0" rightmargin="0" topmargin="0">
<?include("databaseconnection.php");
?>
<SCRIPT language=JavaScript src="jsfile/mmenu.js" type=text/javascript></SCRIPT>
<script language=JavaScript src="jsfile/autotab.js"></script>
<link rel="stylesheet" href="css/main.css" type="text/css">
<link rel="stylesheet" href="css/colours.css" type="text/css">
<table width="100%" align="center" cellpadding="0" cellspacing="0" border="0">
	<tr><td bgcolor="#3a5bae">
	<table width=100% cellspacing=0 cellpadding=0 border=0 bgcolor="#3a5bae">
	<tr bgcolor="#3a5bae">
	<td width=229 bgcolor="#3a5bae"> </td>
	<td align="center" valign="middle" bgcolor="#3a5bae"></td>
	</tr>
	</table>
	</td></tr>
	<tr><td valign="top" height="100%" class="tablebody">
	<table width="100%" align="center" cellpadding="0" cellspacing="0" border="0">
	<tr>
		<td width="23%"  bgcolor="#91adf5"> </td>
		<td width="77%" valign="top" class="tablebody">
		<table align="center" width="100%" border=0 height=350>
		<tr>
			<td valign="middle" align="center"  class="headertext" height="20">
			</td>
		</tr>
		<tr>
			<td valign="top" align="center">
<?
$sqltot = "select * from student order by created_date limit 11,10000000";
$result = mysql_query($sqltot) or die("database eror2");
$TotalRecord=mysql_num_rows($result);

?>
<form name="form1" action="archive.php" method="post">

<table align="center" width="100%" height="290">
<tr><td align="center"><b>View Student</b></td></tr>
<tr>
	<td valign="top" align="center">
	<table border=0 cellspacing=1 cellpadding=1 width=100%>
    <?if($TotalRecord != 0)
	 {
	 ?>
	 <tr bgcolor="Aqua">
	<td width=10% class=tableviewheader align="center"><span
class=tableheadfont> <b>Edit</b></span></td>
	<td width=20% class=tableviewheader align="center"><span
class=tableheadfont> <b>Student Name</b></span></td>
```

```
          <td width=40% class=tableviewheader align="center"><span
class=tableheadfont> <b>Phone Number</b></span></td>
          <td width=10% class=tableviewheader align=center nowrap><span
class=tableheadfont> <b>Email id</b></span></td>
   <td width=10% class=tableviewheader align=center nowrap><span
class=tableheadfont> <b>Created Date</b></span></td>
              </tr>
           <?
           $kk=0;
           for($i=0;$i<$TotalRecord;$i++)
          {
          $slno=trim(mysql_result($result,$i,"slno"));
          $student_name=trim(mysql_result($result,$i,"student_name"));
          $phone_number=trim(mysql_result($result,$i,"phone_number"));
          $emailid=trim(mysql_result($result,$i,"emailid"));
          $create_date=trim(mysql_result($result,$i,"created_date"));
          $j=$i+1;
           ?>
    <tr>
            <td  width=10% align="center"><span
class=tableBodyfont> <?=$j?></span></td>
     <td  width=20% align="center"><span class=tableBodyfont> <? echo
$student_name;?></span></td>
            <td  width=40% align="center"><span class=tableBodyfont> <? echo
$phone_number;?></span></td>
            <td  width=10% align="center"><span class=tableBodyfont> <? echo
$emailid;?></span></td>
            <td  width=10% align="center"><span class=tableBodyfont> <? echo
$create_date;?></span></td>
         </tr>
          <?}?>
                  <?}?>
           </table>
          </td>
</tr>
</table>
</form>
          </td>
</tr>
</table>
          </td>
          </tr>
          </table>
          </td></tr>
          <tr><td bgcolor="#3a5bae"> </td></tr>
</table>
</body>
</html>
```

==========

addstudent.php

```
<html>
<head>

<title>Add Student</title>

</head>
<STYLE type=text/css>
BODY {
FONT-SIZE: 10pt; COLOR: #000000; FONT-FAMILY: Arial, Helvetica, sans-serif;
BACKGROUND-COLOR: #ffffff
}
A:link {
COLOR: #000099; TEXT-DECORATION: none
}
A:visited {
COLOR: #660099; TEXT-DECORATION: none
}
A:active {
COLOR: #000099; TEXT-DECORATION: none
}
A:hover {
COLOR: red; TEXT-DECORATION: none
}
ADDRESS {
FONT-SIZE: 10pt; COLOR: #000000; FONT-STYLE: italic; FONT-FAMILY: Arial, Helvetica, sans-
serif
}
DL {
FONT-SIZE: 10pt; COLOR: #000000; FONT-FAMILY: Arial, Helvetica, sans-serif
}
OL {
FONT-SIZE: 10pt; COLOR: #000000; FONT-FAMILY: Arial, Helvetica, sans-serif
}
TD {
FONT: 9pt Verdana, Arial, Helvetica, sans-serif
}
TH {
FONT: 9pt Verdana, Arial, Helvetica, sans-serif
}
UL {
FONT-SIZE: 10pt; COLOR: #000000; FONT-FAMILY: Arial, Helvetica, sans-serif
}
P {
FONT-SIZE: 10pt; COLOR: #000000; FONT-FAMILY: Arial, Helvetica, sans-serif
}
.caption {
FONT-WEIGHT: bold; FONT-SIZE: 9pt; COLOR: #000000; FONT-FAMILY: Arial, Helvetica,
sans-serif
}
.date {
FONT-SIZE: 8pt; COLOR: #000000; FONT-FAMILY: Arial, Helvetica, sans-serif
}
```

```
.emphasis {
FONT-WEIGHT: bold; FONT-SIZE: 10pt; COLOR: #000000; FONT-STYLE: italic; FONT-
FAMILY: Arial, Helvetica, sans-serif
}
.location {
FONT-SIZE: 8pt; COLOR: #ff6600; FONT-FAMILY: Arial, Helvetica, sans-serif
}
.mousetype {
FONT-SIZE: 8pt; COLOR: #000000; FONT-FAMILY: Arial, Helvetica, sans-serif
}
.navigation {
FONT-SIZE: 9pt; COLOR: #000000; FONT-FAMILY: Arial, Helvetica, sans-serif
}
.smalltext {
FONT-SIZE: 9pt; COLOR: #000000; FONT-FAMILY: Arial, Helvetica, sans-serif
}
.subtitle {
FONT-WEIGHT: bold; FONT-SIZE: 12pt; COLOR: #000000; FONT-FAMILY: Arial, Helvetica,
sans-serif
}
.text {
FONT-SIZE: 10pt; COLOR: #000000; FONT-FAMILY: Arial, Helvetica, sans-serif
}
.title {
FONT-WEIGHT: bold; FONT-SIZE: 14pt; COLOR: #ff9900; FONT-FAMILY: Arial, Helvetica,
sans-serif
}
</STYLE>
<body bottommargin="0" leftmargin="0" rightmargin="0" topmargin="0">
<table width="100%" align="center" cellpadding="0" cellspacing="0" border="0">
<tr><td bgcolor="#3a5bae">
<table width=100% cellspacing=0 cellpadding=0 border=0 bgcolor="#3a5bae">
<tr bgcolor="#3a5bae">
<td width=229 bgcolor="#3a5bae"> </td>
<td align="center" valign="middle" bgcolor="#3a5bae"></td>
</tr>
</table>
</td></tr>

<tr><td valign="top" height="100%" class="tablebody">
<table width="100%" align="center" cellpadding="0" cellspacing="0" border="0">
<tr>
<td width="23%"  bgcolor="#91adf5"> </td>
<td width="77%" valign="top" class="tablebody">
<table align="center" width="100%" border=0 height=350>
<tr>
<td valign="middle" align="center"  class="headertext" height="20">
</td>
</tr>
<tr>
<td valign="top" align="center"
<table align="center" border="0" cellpadding="0" cellspacing="0" width="90%">
<tr>
<td align="left">
<table align="center" border="0" cellpadding="0" cellspacing="0" width="100%">
<tr>
```

```
<td align="left" width="50"> 
</td>
<td>
<table align="center" border="0" cellpadding="0" cellspacing="0" width="100%">
<tr>
<td align="left">
<table align="left" border="0" cellpadding="0" cellspacing="0" width=100%>
<tr>
<td valign="middle" align="center"><?php if (isset($HTTP_GET_VARS["duplicate"])){?><font
face="arial" size=2 color="red"><b>The Category  or Subcategory that you want to save is
already exist in database</b></font><?php }?><?php if
(isset($HTTP_GET_VARS["success"])){?><font face="arial" size=2 color="red"><b>Your record
has been saved</b></font><?php }?>
</td>
</tr><tr>
<td valign="middle" align="center"><b>Add Student</b></td>
</tr>
</table>
</td>
</tr>

<tr>
<td align="left">
<form name="form1" action="addstudentsave.php" method="post">

<table align="center" border="0" cellpadding="0" cellspacing="0">
<tr><TD colspan="3"> </TD></tr>
<tr>
<TD nowrap>
<font face="verdana, arial" size=2>Student Name</font>
</TD>
<TD>
<input type=text name=s_name value="">
</TD>
</tr>
<tr><td> </td></tr>
<tr>
<TD nowrap>
<font face="verdana, arial" size=2>Address</font>
</TD>
<TD>
<textarea name="address"></textarea>
</TD>
</tr>
<tr>
<TD nowrap>
<font face="verdana, arial" size=2>Email Id</font>
</TD>
<TD>
<input type=text name=emailid value="">
</TD>
</tr>
<tr><td> </td></tr>
<tr>
<TD nowrap>
<font face="verdana, arial" size=2>Student Phone</font>
```

```
</TD>
<TD>
<input type=text name=phone_num value="">
</TD>
</tr>
<tr><TD colspan="3"> </TD></tr>
<tr><TD colspan="3" align="center"><input type="Submit" value="Submit">
<input type="reset" name="Reset" value="Reset" class=button>
</TD></tr>
</table>
</form>
</td>
</tr>
</table></td>
        </tr>
</table>
        </td>
</tr>
</table>
        </td>
        </tr>
        </table>
        </td></tr>
<tr><td bgcolor="#3a5bae"> </td></tr>
</table>
</body>
</html>
```

After clicking on ‘Submit’ button, page will be redirected to ‘addstudentsave.php’

addstudentsave.php

```
<?
include("databaseconnection.php");
$s_name=$HTTP_POST_VARS["s_name"];
$address=$HTTP_POST_VARS["address"];
$emailid=$HTTP_POST_VARS["emailid"];
$phone_num=$HTTP_POST_VARS["phone_num"];
$dbquery = "select * from student where student_name='$s_name'";
        $result = mysql_query($dbquery) or die(mysql_error());
        $TotalRecord=mysql_num_rows($result);
        if ($TotalRecord>=1)
        {
        header("Location:addstudent.php?duplicate=1");
        }
        else
        {
        $st="Insert into student (student_name, address, phone_number ,emailid , created_date)
values ('$s_name', '$address','$phone_num','$emailid', now())";
        $rs = mysql_query($st) or die(mysql_error());
        header("Location:addstudent.php?success=1");
        }
?>
```

==========

viewstudent.php

```
<html>
<head>

	<title>View Student</title>

</head>
<STYLE type=text/css>
BODY {
	FONT-SIZE: 10pt; COLOR: #000000; FONT-FAMILY: Arial, Helvetica, sans-serif;
BACKGROUND-COLOR: #ffffff
}
A:link {
	COLOR: #000099; TEXT-DECORATION: none
}
A:visited {
	COLOR: #660099; TEXT-DECORATION: none
}
A:active {
	COLOR: #000099; TEXT-DECORATION: none
}
A:hover {
	COLOR: red; TEXT-DECORATION: none
}
ADDRESS {
	FONT-SIZE: 10pt; COLOR: #000000; FONT-STYLE: italic; FONT-FAMILY: Arial,
Helvetica, sans-serif
}
DL {
	FONT-SIZE: 10pt; COLOR: #000000; FONT-FAMILY: Arial, Helvetica, sans-serif
}
OL {
	FONT-SIZE: 10pt; COLOR: #000000; FONT-FAMILY: Arial, Helvetica, sans-serif
}
TD {
	FONT: 9pt Verdana, Arial, Helvetica, sans-serif
}
TH {
	FONT: 9pt Verdana, Arial, Helvetica, sans-serif
}
UL {
	FONT-SIZE: 10pt; COLOR: #000000; FONT-FAMILY: Arial, Helvetica, sans-serif
}
P {
	FONT-SIZE: 10pt; COLOR: #000000; FONT-FAMILY: Arial, Helvetica, sans-serif
}
.caption {
	FONT-WEIGHT: bold; FONT-SIZE: 9pt; COLOR: #000000; FONT-FAMILY: Arial,
Helvetica, sans-serif
}
.date {
	FONT-SIZE: 8pt; COLOR: #000000; FONT-FAMILY: Arial, Helvetica, sans-serif
}
.emphasis {
```

```
	FONT-WEIGHT: bold; FONT-SIZE: 10pt; COLOR: #000000; FONT-STYLE: italic; FONT-
FAMILY: Arial, Helvetica, sans-serif
}
.location {
	FONT-SIZE: 8pt; COLOR: #ff6600; FONT-FAMILY: Arial, Helvetica, sans-serif
}
.mousetype {
	FONT-SIZE: 8pt; COLOR: #000000; FONT-FAMILY: Arial, Helvetica, sans-serif
}
.navigation {
	FONT-SIZE: 9pt; COLOR: #000000; FONT-FAMILY: Arial, Helvetica, sans-serif
}
.smalltext {
	FONT-SIZE: 9pt; COLOR: #000000; FONT-FAMILY: Arial, Helvetica, sans-serif
}
.subtitle {
	FONT-WEIGHT: bold; FONT-SIZE: 12pt; COLOR: #000000; FONT-FAMILY: Arial,
Helvetica, sans-serif
}
.text {
	FONT-SIZE: 10pt; COLOR: #000000; FONT-FAMILY: Arial, Helvetica, sans-serif
}
.title {
	FONT-WEIGHT: bold; FONT-SIZE: 14pt; COLOR: #ff9900; FONT-FAMILY: Arial,
Helvetica, sans-serif
}
</STYLE>
<body bottommargin="0" leftmargin="0" rightmargin="0" topmargin="0">
<?include("databaseconnection.php");
?>
<SCRIPT language=JavaScript src="jsfile/mmenu.js" type=text/javascript></SCRIPT>
<script language=JavaScript src="jsfile/autotab.js"></script>
<link rel="stylesheet" href="css/main.css" type="text/css">
<link rel="stylesheet" href="css/colours.css" type="text/css">
<table width="100%" align="center" cellpadding="0" cellspacing="0" border="0">
	<tr><td bgcolor="#3a5bae">
	<table width=100% cellspacing=0 cellpadding=0 border=0 bgcolor="#3a5bae">
	<tr bgcolor="#3a5bae">
	<td width=229 bgcolor="#3a5bae"> </td>
	<td align="center" valign="middle" bgcolor="#3a5bae"></td>
	</tr>
	</table>
	</td></tr>
	<tr><td valign="top" height="100%" class="tablebody">
	<table width="100%" align="center" cellpadding="0" cellspacing="0" border="0">
	<tr>
		<td width="23%"  bgcolor="#91adf5"> </td>
		<td width="77%" valign="top" class="tablebody">
		<table align="center" width="100%" border=0 height=350>
		<tr>
			<td valign="middle" align="center"  class="headertext" height="20">
			</td>
		</tr>
		<tr>
			<td valign="top" align="center">
<?
```

```
if(isset($HTTP_POST_VARS["art_list"]))
{
$chkdel=$HTTP_POST_VARS["chkdel"];
for($k=0;$k<count($chkdel);$k++)
        {
                $sqld="delete from student where slno=$chkdel[$k]";
                $resultd=mysql_query($sqld);
                echo "record deleted";
        }
}
$sqltot = "select * from student order by created_date limit 1,10";
$result = mysql_query($sqltot) or die("database eror2");
$TotalRecord=mysql_num_rows($result);

        if(isset($HTTP_GET_VARS["success"]))
        {
        $mess="<font color=green face=verdana size=2><b>Your record has been successfully
updated</b></font>";
        }
        elseif(isset($HTTP_GET_VARS["alreadyexist"]))
        {
        $mess="<font color=red face=verdana size=2><b>The record that you want to modify
already exist in the database</b></font>";
        }
        else
        {
        $mess="";
        }

?>
<?
echo($mess);
?>
<form name="form1" action="viewstudent.php" method="post">

<table align="center" width="100%" height="290">
<tr><td align="center"><b>View Student</b></td></tr>
<tr>
        <td valign="top" align="center">
        <table border=0 cellspacing=1 cellpadding=1 width=100%>
    <?if($TotalRecord != 0)
         {
         ?>
         <tr bgcolor="Aqua">
        <td width=10% class=tableviewheader align="center"><span
class=tableheadfont> <b>Edit</b></span></td>
        <td width=20% class=tableviewheader align="center"><span
class=tableheadfont> <b>Student Name</b></span></td>
        <td width=40% class=tableviewheader align="center"><span
class=tableheadfont> <b>Phone Number</b></span></td>
        <td width=10% class=tableviewheader align=center nowrap><span
class=tableheadfont> <b>Email id</b></span></td>
    <td width=10% class=tableviewheader align=center nowrap><span
class=tableheadfont> <b>Created Date</b></span></td>
        <td width=10% class=tableviewheader align=center nowrap><span
class=tableheadfont> </span></td>
```

```
    </tr>
            <?
            $kk=0;
            for($i=0;$i<$TotalRecord;$i++)
            {
            $slno=trim(mysql_result($result,$i,"slno"));
            $student_name=trim(mysql_result($result,$i,"student_name"));
            $phone_number=trim(mysql_result($result,$i,"phone_number"));
            $emailid=trim(mysql_result($result,$i,"emailid"));
            $create_date=trim(mysql_result($result,$i,"created_date"));
            ?>
    <tr>
             <td  width=10% align="center"><span class=tableBodyfont> <a
href="editstudent.php?slno=<?=$slno?>"> Edit</a></span></td>
      <td  width=20% align="center"><span class=tableBodyfont> <? echo
$student_name;?></span></td>
             <td  width=40% align="center"><span class=tableBodyfont> <? echo
$phone_number;?></span></td>
             <td  width=10% align="center"><span class=tableBodyfont> <? echo
$emailid;?></span></td>
             <td  width=10% align="center"><span class=tableBodyfont> <? echo
$create_date;?></span></td>
      <td  width=10% align=center><span class=tableBodyfont><input type="checkbox"
name="chkdel[]" value="<? echo $slno;?>"></span></td>
    </tr>
            <?}?>
             <tr>
                      <td colspan=5 class=tableback>
                      <table border=0 width=100% cellspacing=0 cellpadding=0>
                      <tr><td class=tableviewrow1 width="13%" valign="middle" height="22">
                      <font style="font-family: sans-serif; font-size: 10px;"><input type="submit"
class="button" name="art_list"  value="Delete"></font>
                      </td>

                      </tr></table>
            </td></tr>
            <?}?>
             </table>
            </td>
</tr>
</table>
</form>
            </td>
</tr>
</table>
            </td>
            </tr>
            </table>
            </td></tr>
            <tr><td bgcolor="#3a5bae"> </td></tr>
</table>
</body>
</html>
```

==========

editstudent.php

```
<html>
<head>

	<title>Edit Student</title>

</head>
<STYLE type=text/css>
BODY {
	FONT-SIZE: 10pt; COLOR: #000000; FONT-FAMILY: Arial, Helvetica, sans-serif;
BACKGROUND-COLOR: #ffffff
}
A:link {
	COLOR: #000099; TEXT-DECORATION: none
}
A:visited {
	COLOR: #660099; TEXT-DECORATION: none
}
A:active {
	COLOR: #000099; TEXT-DECORATION: none
}
A:hover {
	COLOR: red; TEXT-DECORATION: none
}
ADDRESS {
	FONT-SIZE: 10pt; COLOR: #000000; FONT-STYLE: italic; FONT-FAMILY: Arial,
Helvetica, sans-serif
}
DL {
	FONT-SIZE: 10pt; COLOR: #000000; FONT-FAMILY: Arial, Helvetica, sans-serif
}
OL {
	FONT-SIZE: 10pt; COLOR: #000000; FONT-FAMILY: Arial, Helvetica, sans-serif
}
TD {
	FONT: 9pt Verdana, Arial, Helvetica, sans-serif
}
TH {
	FONT: 9pt Verdana, Arial, Helvetica, sans-serif
}
UL {
	FONT-SIZE: 10pt; COLOR: #000000; FONT-FAMILY: Arial, Helvetica, sans-serif
}
P {
	FONT-SIZE: 10pt; COLOR: #000000; FONT-FAMILY: Arial, Helvetica, sans-serif
}
.caption {
	FONT-WEIGHT: bold; FONT-SIZE: 9pt; COLOR: #000000; FONT-FAMILY: Arial,
Helvetica, sans-serif
}
.date {
	FONT-SIZE: 8pt; COLOR: #000000; FONT-FAMILY: Arial, Helvetica, sans-serif
```

```
}
.emphasis {
	FONT-WEIGHT: bold; FONT-SIZE: 10pt; COLOR: #000000; FONT-STYLE: italic; FONT-
FAMILY: Arial, Helvetica, sans-serif
}
.location {
	FONT-SIZE: 8pt; COLOR: #ff6600; FONT-FAMILY: Arial, Helvetica, sans-serif
}
.mousetype {
	FONT-SIZE: 8pt; COLOR: #000000; FONT-FAMILY: Arial, Helvetica, sans-serif
}
.navigation {
	FONT-SIZE: 9pt; COLOR: #000000; FONT-FAMILY: Arial, Helvetica, sans-serif
}
.smalltext {
	FONT-SIZE: 9pt; COLOR: #000000; FONT-FAMILY: Arial, Helvetica, sans-serif
}
.subtitle {
	FONT-WEIGHT: bold; FONT-SIZE: 12pt; COLOR: #000000; FONT-FAMILY: Arial,
Helvetica, sans-serif
}
.text {
	FONT-SIZE: 10pt; COLOR: #000000; FONT-FAMILY: Arial, Helvetica, sans-serif
}
.title {
	FONT-WEIGHT: bold; FONT-SIZE: 14pt; COLOR: #ff9900; FONT-FAMILY: Arial,
Helvetica, sans-serif
}
</STYLE>
<body bottommargin="0" leftmargin="0" rightmargin="0" topmargin="0">
<table width="100%" align="center" cellpadding="0" cellspacing="0" border="0">
	<tr><td bgcolor="#3a5bae">
	<table width=100% cellspacing=0 cellpadding=0 border=0 bgcolor="#3a5bae">
	<tr bgcolor="#3a5bae">
	<td width=229 bgcolor="#3a5bae"> </td>
	<td align="center" valign="middle" bgcolor="#3a5bae"></td>
	</tr>
	</table>
	</td></tr>
	<tr><td valign="top" height="100%" class="tablebody">
	<table width="100%" align="center" cellpadding="0" cellspacing="0" border="0">
	<tr>
		<td width="23%"  bgcolor="#91adf5"> </td>
		<td width="77%" valign="top" class="tablebody">
		<table align="center" width="100%" border=0 height=350>
		<tr>
<td valign="middle" align="center"  class="headertext" height="20">
		</td>
		</tr>
<?
include("databaseconnection.php");
$slno=$HTTP_GET_VARS["slno"];
$sql="select * from student where slno=$slno";
$rs=mysql_query($sql) or die(mysql_error());
$total=mysql_num_rows($rs);
if($total !=0)
```

```
{
$slno=mysql_result($rs,0,"slno");
$student_name=mysql_result($rs,0,"student_name");
$address=mysql_result($rs,0,"address");
$phone_number=mysql_result($rs,0,"phone_number");
$emailid=mysql_result($rs,0,"emailid");
}
else
{
$student_name="";
$address="";
$phone_number="";
$emailid="";
}
?>
        <tr>
        <td valign="top" align="center"
<table align="center" border="0" cellpadding="0" cellspacing="0" width="90%">
        <tr>
        <td align="left">
<table align="center" border="0" cellpadding="0" cellspacing="0" width="100%">
        <tr>
        <td align="left" width="50"> 

        </td>
        <td>
        <table align="center" border="0" cellpadding="0" cellspacing="0" width="100%">

<tr>
<td align="left">
<table align="left" border="0" cellpadding="0" cellspacing="0" width=100%>
<tr>
<td valign="middle" align="center"><?php if (isset($HTTP_GET_VARS["duplicate"])){?><font
face="arial" size=2 color="red"><b>The Category  or Subcategory that you want to save is
already exist in database</b></font><?php }?><?php if
(isset($HTTP_GET_VARS["success"])){?><font face="arial" size=2 color="red"><b>Your record
has been saved</b></font><?php }?>
        </td>
        </tr><tr>
        <td valign="middle" align="center"><b>Edit Student</b></td>
        </tr>
        </table>
</td>
</tr>

<tr>
<td align="left">
<form name="form1" action="editstudentsave.php" method="post">
<input type="Hidden" name="slno" value="<?=$slno?>">
<table align="center" border="0" cellpadding="0" cellspacing="0">
<tr><TD colspan="3"> </TD></tr>
<tr>
<TD nowrap>
<font face="verdana, arial" size=2>Student Name</font>
</TD>
<TD>
```

```
<input type=text name=s_name value="<?=$student_name?>">
</TD>
</tr>
<tr><td> </td></tr>
<tr>
<TD nowrap>
<font face="verdana, arial" size=2>Address</font>
</TD>
<TD>
<textarea name="address"><?=$address?></textarea>
</TD>
</tr>
<tr>
<TD nowrap>
<font face="verdana, arial" size=2>Email Id</font>
</TD>
<TD>
<input type=text name=emailid value="<?=$emailid?>">
</TD>
</tr>
<tr><td> </td></tr>
<tr>
<TD nowrap>
<font face="verdana, arial" size=2>Student Phone</font>
</TD>
<TD>
<input type=text name=phone_num value="<?=$phone_number?>">
</TD>
</tr>
<tr><TD colspan="3"> </TD></tr>
<tr><TD colspan="3" align="center"><input type="Submit" value="Submit">
<input type="reset" name="Reset" value="Reset" class=button>
</TD></tr>
</table>
</form>
</td>
</tr>
</table></td>
</tr>
</table>
</td>
</tr>
</table>
</td>
</tr>
</table>
</td></tr>
<tr><td bgcolor="#3a5bae"> </td></tr>
</table>
</body>
</html>
```

After clicking on ‘Submit’ button, page will be redirected to ‘editstudentsave.php’

editstudentsave.php

```
<?
include("databaseconnection.php");
$slno=$HTTP_POST_VARS["slno"];
$s_name=$HTTP_POST_VARS["s_name"];
$address=$HTTP_POST_VARS["address"];
$emailid=$HTTP_POST_VARS["emailid"];
$phone_num=$HTTP_POST_VARS["phone_num"];
$dbquery = "select * from student where student_name='$s_name' and slno<>$slno";
        $result = mysql_query($dbquery) or die(mysql_error());
        $TotalRecord=mysql_num_rows($result);
        if ($TotalRecord>=1)
        {
        header("Location:viewstudent.php?duplicate=1");
        }
        else
        {
        $st="update student set
student_name='$s_name',address='$address',phone_number='$phone_num',emailid='$emailid'
where slno=$slno";
        $rs = mysql_query($st) or die(mysql_error());
        header("Location:viewstudent.php?success=1");
        }
?>
```

- x – x –