PHP & MySQL

An Introduction to Databases

Databases are just containers that hold information. In this article , we will look at MySQL databases. These hold information for things like forums, chatrooms, usernames and passwords etc.
I don't want to bore you with all the technical information about them so I'll just get right into the coding bit.

Connecting to Databases

Too connect to MySQL the PHP mysql_connect() function requiers three arguments to connect, a vaild domain name,a username and a password. In this example these arguments are localhost, username and password. If the connection is successful is returns true therefore displaying "Congratulations! You connected to MySQL" and if the connection fails then it returns false and will display nothing on the screen.

connect.php

```
<?php
$conn = mysql_connect("localhost", "username", "password");
if($conn)
{
 $msg = "Congratulations! You connected to MySQL!";
}
?>
<html><body><?php echo($msg); ?></body></html>
```

This simply connects to MySQL on localhost using the user account username with the password password. Then if the connection attempt is successful, it will display "Congratulations! You connected to MySQL", if the connection attempt was not successful then it will not output anything.
The user details specified in the code, e.g. username, are case sensitive meaning if you were to type USERNAME instead of username then the connection attempt would fail.

Listing Databases

The mysql_list_dbs() function returns a result set of all the information about all of the databases and in this case, stores them into a variable, $rs.
All the details of a database are held in a separate row from another database's details. You can use the mysql_num_rows() function to find out how many rows there are in the result set.
The name of each database and table can be taken from a result set using the mysql_tablename() function. The arguents for this function are a result set and a row number.

list_dbs.php

```
<?php
$conn = @mysql_connect("localhost", "username", "password") or die("Sorry, could not connect to MySQL");
$rs = mysql_list_dbs($conn);
for($row = 0; $row < mysql_num_rows($rs); $row++)
{
 $db_list .= mysql_tablename($rs, $row) . "<br>";
```

```
}
?>
<html><body><?php echo($db_list); ?></body></html>
```

This connects to MySQL then gets the names of the databases and stores them in $rs. It then adds the database name to $db_list using a for loop.

Listing Database Tables

The function mysql_list_tables() is very similar to the mysql_list_dbs() function although it returns a result set of information about all the tables in a specific database.
To get a table name you need to specify a result set and it's row number in the result set. See the example below.

list_tables.php

```
<?php
$conn = @mysql_connect("localhost", "username", "password") or die("Sorry, could not connect to MySQL");
$rs1 = mysql_list_dbs($conn);
for($row = 0; $row < mysql_num_rows($rs1); $row++)
{
 $db = mysql_tablename($rs1, $row);
 $list .= "<b>" . $db . "</b><br>";
 if($db != "mysql")
{
  $rs2 = mysql_list_tables($db);
  for($num = 0; $num < mysql_num_rows($rs2); $num++)
  { $list .= " - " . mysql_tablename($rs2, $num) . "<br>"; }
 }
}
?>
<html><body><?php echo($list); ?></body></html>
```

This script just outputs the table names underneath their database.

Thank you for reading my quick article on MySQL and PHP.

In the next installment, I will show you how to create MySQL databases and tables and how to insert data into them.

Tutorial One: Beginning PHP

Welcome Note

Thank me, for redoing this whole article that I've submitted to Xavier. =) I couldn't transfer the whole document into this site for Microsoft Word exporting the whole document with styles. But in any case, I hope you guys would still enjoy it, as much as I did this. Some introduction from me: I'm thirteen, and is only beginning to learn PHP. Like some of you out there, you want to learn, but do not know where to start off, you've come to the right place. Note that this tutorial assumes that you have a general idea on programming language, or coding languages such as HTML, etc.

Things to Prepare

Well, it'll be hard to find a site which supports PHP and mySQL, so we'll do PHP in *our* personal computer; or workstation. So, your next question might be, "what exactly do I prepare?"

What in the world would you do if you can't find any host? Well, load PHP files in your computer! How, you ask? Simple, first, I recommend you get a PHP code editor, can't find one? Get what I'll be using in every PHP tutorial: **PHP Designer 2005**. Next, you'll have to get a PHP interpreter, of course, at the **PHP Official Site**, or if you're too lazy to do it, download **this**, and **this**, and extract them to your PHP Designer 2005 folder after you have downloaded it.

Don't quote me on the above (I can already sense someone doing so. =P), don't ask me why am I using PHP Designer, it's entirely my personal opinion.

Conventions

Throughout this tutorial series, you'll find some important conventions that you'll need to take note of. One of them is simply a box, and below each box contains the figure reference number/letter.

```
//This is how a code will be shown for the first time, or this is will be a
//reference to codes that is previously shown.

//And codes that are in bold and bigger of size are lines that are
//added, edited or modified which you should take note of.
```

Fig. 1-A

Red coloured text is important stuff that you should be taking note of, to avoid errors and mistakes that people commonly make. Green, are tips and stuff that you might want to try out on your own. Violet for clarification of new unknown terms to you.

When I speak of loading up of the *.php file in your browser, I am telling you to load it from PHP Designer 2005. Read the short but complete article on loading *.php files from PHP Designer **here**.

Need Help?

In each tutorial, anything that you are left unclear of, should you have emergency in clarification of matters, please contact me via **bluezane@hotmail.co.uk**. Please use proper English while clarifying your unclear mind to me, don't go: Oh my Jesus lord, I can't do this!!!!1111111111oneoneoneoneoneone! Errors will be addressed in the next tutorial released.

Introduction to PHP

Oh no, I'm not going to do a long boring introduction if you can understand me in a few words. What in the world is PHP? Hypertext Pre-processor, it is a powerful server-side scripting language that is used commonly in websites and message boards. What about "MySQL" that I hear commonly together with PHP? MySQL is the database construct that enables PHP and Apache to work together to access and display data in a readable format to a browser. It is a Structured Query Language server designed for heavy loads and processing of complex queries. So, you've got the general idea of the whole thing? I certainly hope so.

Before I start giving you the first few lines of the code, you might be thinking – why do I spend my bloody time on writing all these stuff? Well, I'm killing to birds with one stone, I must admit – one, an upgrade to the writing feature; two, I am still on my way in learning PHP, so while writing, people *hopefully* will correct my mistakes and I can improve on the way.

Tutorial Coverage

By the end of this tutorial, you should have a general idea on:

- Echo – to display text;
- Formatting text with PHP and HTML;
- Constants and variables;
- Mathematics in PHP, and;
- Passing variable values.

And of course, some activities and rounding up for you to apply your newly acquired PHP knowledge.

PHP Syntax

Always remember, that whenever you are about to summon the PHP language, you have to use:

```
<?php                                   // tell the server to execute the codes
echo "Text goes here.";                 // each line in PHP ends with a semicolon
//comments                              // lines that start with double slashes
//comments                              // are your comment reference
?>                                      // ending the execution
```

Fig. 2-A

Also, you might want to practice indenting your codes like in RPGCode.

```
if ($_POST["…"] == "…")
        {
        …;
        }
        else
        {
        …;
        }
```

Fig. 2-B

It might not seem that important to do that in your codes, but it makes it more readable, and mistakes are easier to be spotted. That, I conclude the organization of codes.

First Program

To start it off, "Hello World" is commonly used in programming language, the first essential step to help

beginners delve into the PHP world. Open up your PHP editor (Notepad is also another alternative, you just have to save it as *.php.) and type this code in:

```
<HTML>
<HEAD>
<TITLE>First Program</TITLE>
</HEAD>
<BODY>

<?php
echo "Hello World!";
?>

</BODY>
</HTML>
```

Fig. 3-A

Save it as firstprog.php and load it up in the browser. You should get something like Figure 1-1 below.

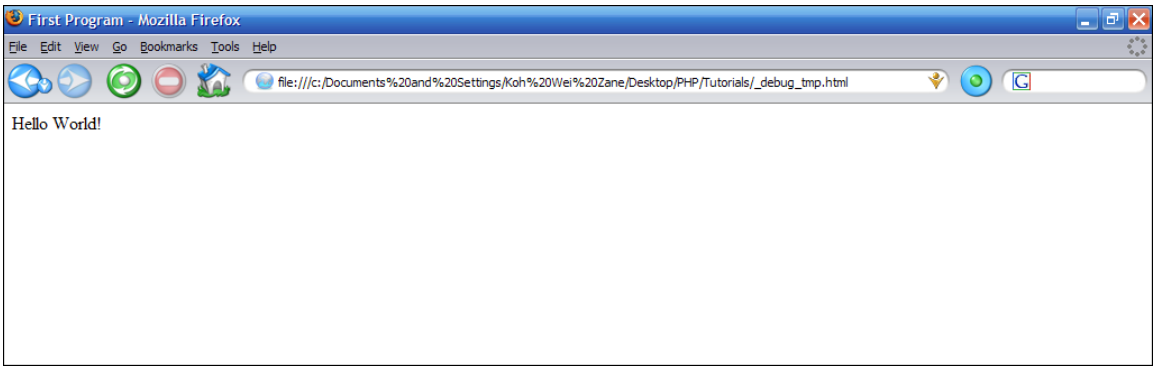

Fig. 3-1

Got the hang of it? Now, try putting "Hello World 02!" under the "Hello World!" line in the above figure; use PHP, don't you try using HTML. So, before you look at the example below, you might want to do a trial and error.

Always put a semicolon after each line in PHP coding.

```
<HTML>
<HEAD>
<TITLE>First Program</TITLE>
</HEAD>
<BODY>

<?php
echo "Hello World!";
echo "<br>";
echo "Hello World 02!";
?>

</BODY>
</HTML>
```

Fig. 3-B

 : A break in line. (Line spacing.)
<p>: Double break lines make a paragraph.
Remember your semicolons; missing them out will risk your PHP program from being executed.

Okay, you've probably tried and had a failed attempt because of the break line which you might have missed out. Now open firstprog.php and type the codes in bold (Fig. 3-B), save it. If you get a result like the one shown in figure 3-2 below, you're on your way in becoming the next PHP coder!

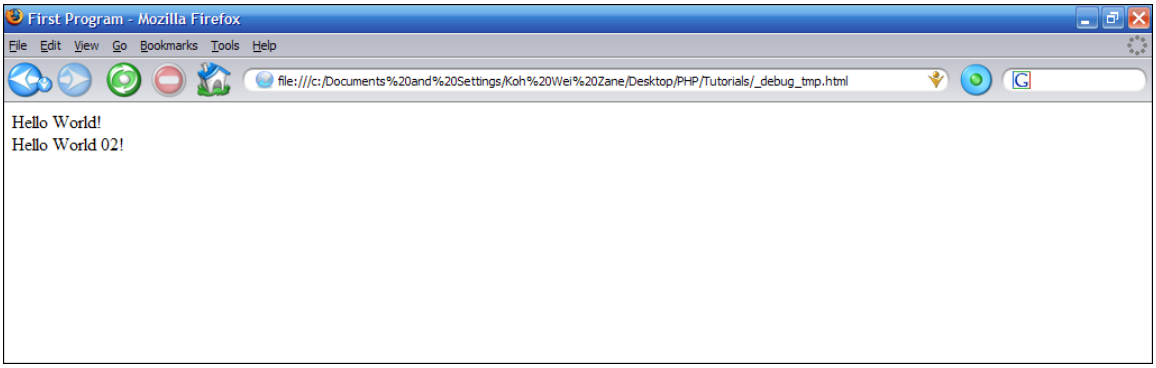

Fig. 3-2

Now, as I've warned you twice above, to always end a sentence in `<?php` and `?>` php, if you didn't, you will get the error as shown in Fig. 3-3 below.

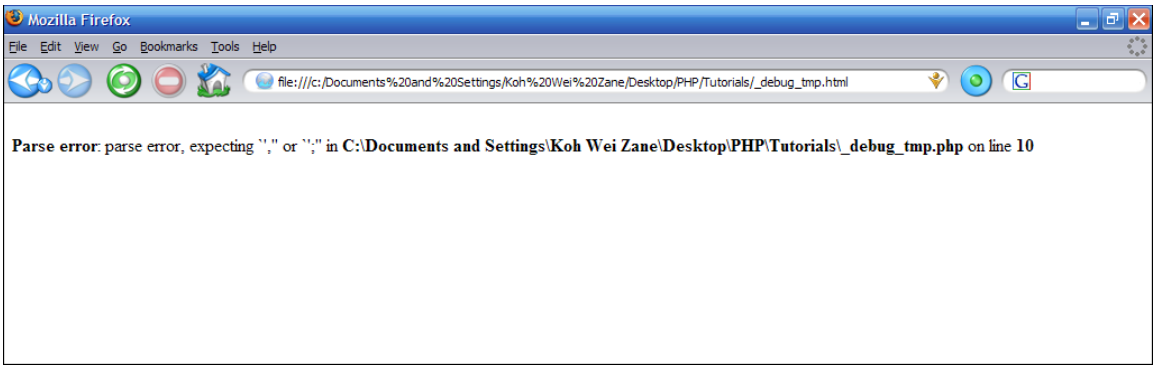

Fig. 3-3

Do not, attempt the code below. It generates the above error.

```
<HTML>
<HEAD>
<TITLE>First Program</TITLE>
</HEAD>
<BODY>

<?php
echo "Hello World!";
echo "<br>"                  // notice that the semicolon is not added. " ; "
echo "Hello World 02!";
?>

</BODY>
</HTML>
```

Fig. 3-C

So, a lesson learnt: always heed advices. Oh, you did? That's good. =P

Now we've done printing out the words on a page, what about formatting it now? Wouldn't you like to see different sizes and fonts? Well, we can do it in both HTML and PHP. But since this is a PHP tutorial, we'll stick to PHP.

Now, you'll get your first task. It's not much, but at least you get some hands on now. Task one: using the codes from Fig. 3-B, make Hello World! change its font to Arial, font size as normal, and make Hello World 02! change its font size to size 10.

The end result should reveal the formatting as shown below (fig. 3-4).

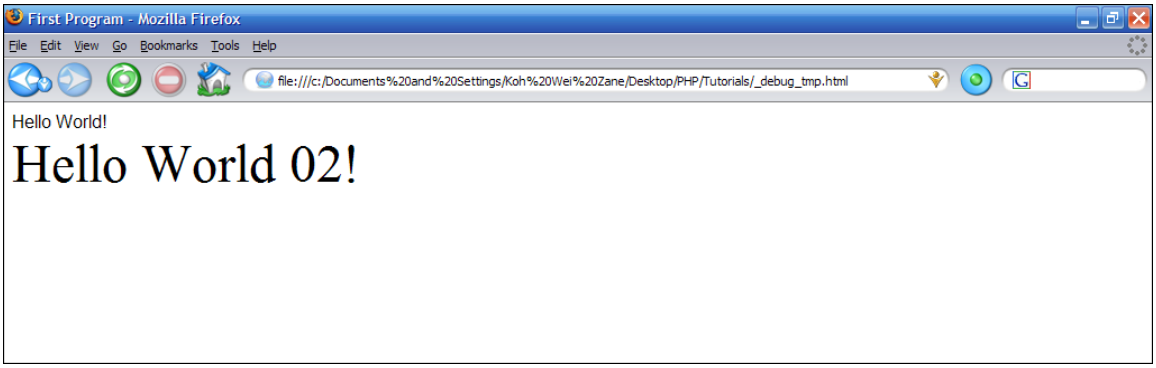

Fig. 3-4

I'll reveal the code in just a moment, just to check, have you really done your coding? Looking at the PHP codes below without trying won't gain you any knowledge...

Let's go:

```
<HTML>
<HEAD>
<TITLE>First Program</TITLE>
</HEAD>
<BODY>

<?php
echo "<font face=\"Arial\">Hello World!</font>";
echo "<br>";
echo "<font size=\"10\">Hello World 02!</font>";
?>

</BODY>
</HTML>
```

Fig. 3-D

The above, is a good example for me to demonstrate how each line is executed.

```
<HTML>
<HEAD>
<TITLE>First Program</TITLE>
</HEAD>
<BODY>
```

The above, simply gets the page rolling, as in, showing "First Program" in the browser title bar itself. The "BODY" is actually where we begin to place all our content to make it publicly viewable.

```
<?php
```

Opening of the PHP statement and content…

```
echo "<font face=\"Arial\">Hello World!</font>";
```

Showing “Hello World!” in the web page with the font as “Arial”.

```
echo "<br>";
```

Break line, as defined above.

```
echo "<font size=\"10\">Hello World 02!</font>";
```

Showing “Hello World 02!” with a font size of ten.

```
?>
```

Closing of PHP statement… And the below, is the ending of the whole .php script.

```
</BODY>
</HTML>
```

If you’ve noticed, you see some back slashes (\ ) before “”. So, what do they mean?

As you may have noted when you worked through the previous example, using the echo function involves the use of double quotation marks. Because HTML also uses double quotes, you can do one of two things to avoid problems:

- Use single quotes inside your HTML.
- Escape your HTML double quotes with a backslash, as in the following:
  `echo “<font size=\”10\”>”;`

This is especially useful if you want to display double quotes in your text, such as in the above example.

Constants and Variables

Constants -

Constants, are generally a placeholder for a value that you reference with your code. Normally one, including me, won’t understand much about all these until you see a working example. Figure 4-A will demonstrate how a function is defined and called. Constants are also case sensitive.

Now, load up your PHP editor and type the codes in:

```
<HTML>
<HEAD>
<TITLE>Constants</TITLE>
</HEAD>
<BODY>

<?php
define ("FAVCOLOUR", "Blue");
echo "My favourite colour is <u>";
echo Blue;
echo "</u>.";
?>
```

```
</BODY>
</HTML>
```

Fig. 4-A

Save it as constants.php, and you should see something like the one below when you load up your browser.

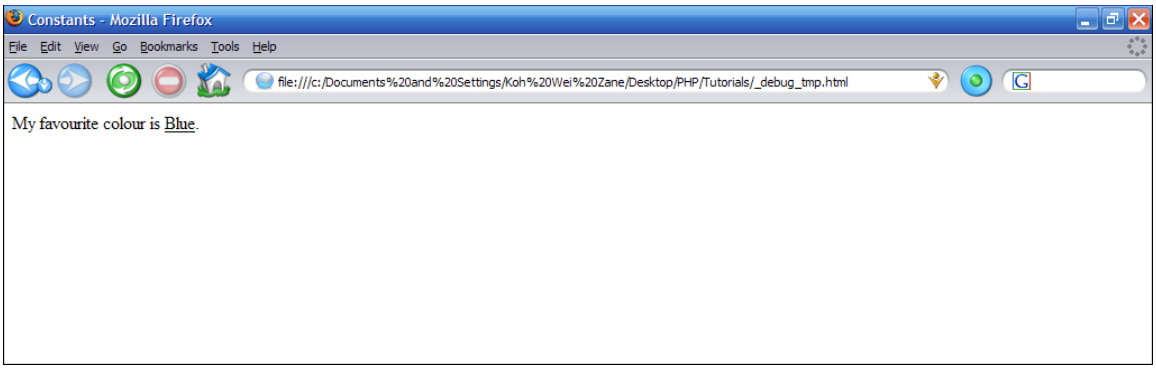

Fig. 4-1

Well, obviously you can modify the codes above to tweak "Blue" to your true favourite colour. Let's break it down, shall we?

```
<?php
define ("FAVCOLOUR", "Blue");
```

Inside "define", the first constant name is FAVCOLOUR, and you have defined it, to be "Blue". So whenever you call it, it will reveal your favourite colour as "Blue".

```
echo "My favourite colour is <u>";
```

This isn't anything new, I just left a spacing after "is" and entered the HTML underline tag, `<u>`.

```
echo Blue;
```

Now, while calling a constant, you should not be placing any inverted commas along with it. Eg. `echo "Blue";` Don't try this, unless you are not calling out the constant, alright?

```
echo "</u>.";
?>
```

The closing of the underline tag, a full-stop after it, and the closing of the PHP codes.

Variables -

Phew, coming this far, let's take a break, drink some water and relax yourself before you continue to read.

Unlike Constants, you can change and assign a new value into the Variable itself at any point of time. Variables are denoted with a $ sign and are not case sensitive, and the first character of the variable must be either a letter or an underscore.

Boot up your editor again, and this time, we'll do another experimental web page on completed surveys and results of it.

```
<HTML>
<HEAD>
<TITLE>Variables</TITLE>
</HEAD>
<BODY>

<?php
define ("FAVCOLOUR", "Blue");
echo "My favourite colour is <u>";
echo Blue;
echo "</u>.";
echo "<br>";
$survey1=10;
$survey2=20;
$survey3=6;
echo "A total of <u>";
echo $survey1;
echo "</u> schoolmates like the above colour.";
echo "<p>";
echo "Red is another general colour and <u>";
echo $survey2;
echo "</u> of my schoolmates like red.";
echo "<p>";
echo "The final main colour is Yellow and <u>";
echo $survey3;
echo "</u> of my schoolmates like yellow.";
?>

</BODY>
</HTML>
```

Fig. 4-B

Save that as variables.php. The end result is:

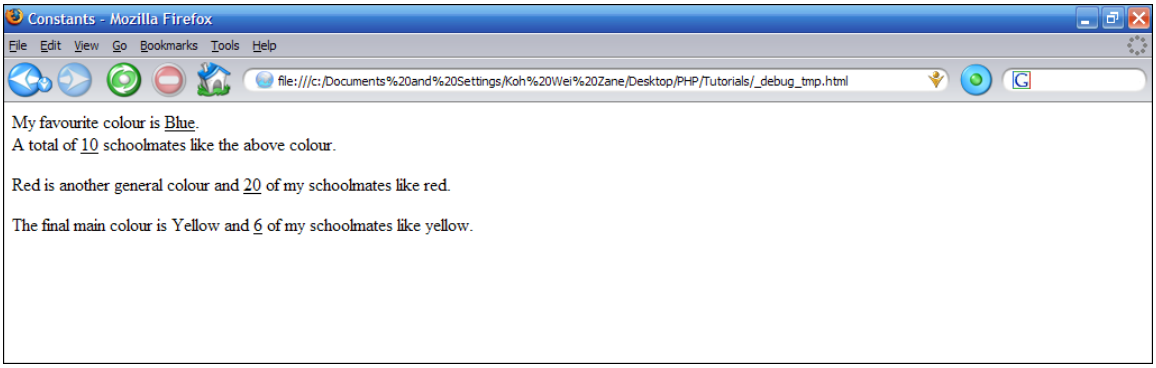

Fig. 4-2

Now, another challenge, see if you can define each PHP-line and figure out what it does. So before you look at the answer below, I hope you already have learnt something, and you know what each line signifies.

```
$survey1=10;
$survey2=20;
$survey3=6;
```

Like RPGcode, PHP has more or less the same format of representing variables. Take `$survey` as the colours, red, blue, and yellow that is being voted for popularity. And [x] as the number of people who actually has that colour as their favourites.

That's basically it, and to round up this topic, I'd like to give out another warning… Do not, in any way, forget the semicolon character after each PHP sentence.

Mathematics in PHP

Easily, PHP can do mathematics for you, as any general regular expressions that are use in the real world. I'll set a simple example: an English test.

So what we need, are the pupils' name, and their scores, and we'll find their average.

Pupil Number… (pupil[x])	Pupil's Score (=[y])
1	60 / 100
2	78 / 100
3	42 / 100

Now, take this down on a piece of rough paper if you like… And we'll begin, PHP-ing… *evil grin* Let's start by, of course, loading your editor.

Just to let you know, that I'll be storing the pupil reference number like this: $pupil1, $pupil2, $pupil3 and their score like this: $pupil1=60… And so on. Start a brand new file and type the below in.

```
<HTML>
<HEAD>
<TITLE>Mathematics in PHP</TITLE>
</HEAD>
<BODY>

<?php
$pupil1=60;
$pupil2=78;
$pupil3=42;
echo "Pupil 1 scored <u>";
echo $pupil1;
echo "</u> in a English test. ";
echo "While Pupil 2 and three scored <u>";
echo $pupil2;
echo "</u> and <u>";
echo $pupil3;
echo "</u> respectively.";
echo "<p>";
echo "Their average score is: <u>";
$average=(($pupil1+$pupil2+$pupil3)/3);
echo $average;
echo "</u>.";
?>

</BODY>
</HTML>
```

Fig. 4-C

Save this as math.php. Now if you run it, you'll get this:

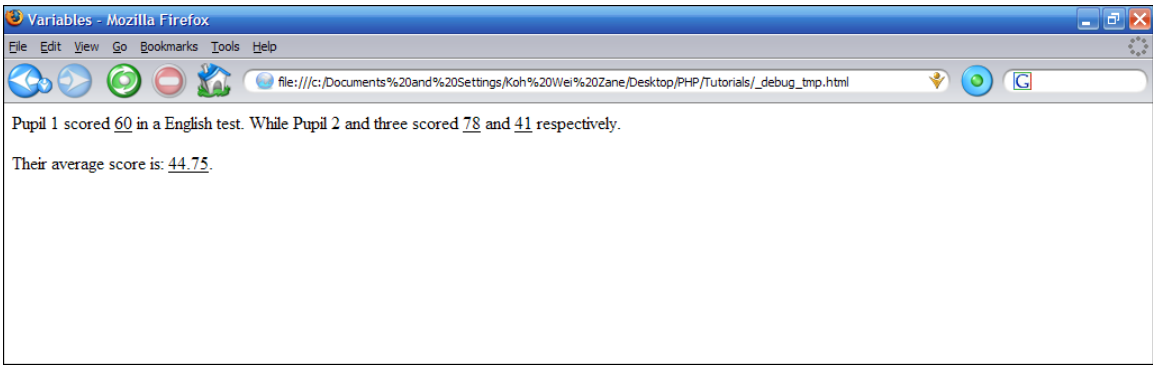

Fig. 4-3

I'll suppose you need me to break this down for you, for further understanding.

```
Line 01 <?php
Line 02 $pupil1=60;
Line 03 $pupil2=78;
Line 04 $pupil3=42;
Line 05 echo "Pupil 1 scored <u>";
Line 06 echo $pupil1;
Line 07 echo "</u> in a English test. ";
Line 08 echo "While Pupil 2 and three scored <u>";
Line 09 echo $pupil2;
Line 10 echo "</u> and <u>";
Line 11 echo $pupil3;
Line 12 echo "</u> respectively.";
Line 13 echo "<p>";
Line 14 echo "Their average score is: <u>";
Line 15 $average=(($pupil1+$pupil2+$pupil3)/3);
Line 16 echo $average;
Line 17 echo "</u>.";
Line 18 ?>
```

Line 01 and 18, I hope by now you do not need me to explain. Line 02 to 04, I featured the pupil number (`$pupil[x]`) and their scores (`=[x]`). The part where I want to go further into, is line fifteen. It will not look complex if you see things this way:

`$average` All scores added together, and finding its average.
`+` You add up all three pupil's score using this character.
`/` The division character.

In mathematics, you have to express yourself like this: **(2+2+2) / 3** to get the average, and same goes to PHP. So…

```
$pupil1=60
$pupil2=78
$pupil3=42

60 + 78 + 42 / 3 // This won't get you the average!
(60 + 78 + 42) / 3 // Isn't this better?
```

Final Words

Now, I wish you pure good luck with PHP, now go try and experiment it, till the next tutorial, then! Again, if you have any enquiries or spotted any errors, please ask/correct and contact me at **bluezane@hotmail.co.uk**. Stay healthy.

Sprite Tutorial
by Flicky

Introduction

This tutorial is for people who would like to learn a different approach to creating a character sprite. All you need is a pre-made character sprite and some time. This technique has used the ideas of other people's sprite tutorials and my own to create a character sprite.

So now lets get on with the tutorial!

Step One: Basic Shape

First off get your pre-made sprite. This sprite was created at **Charas**. If you want you can take the easy route and create a random generated character there. If you were like me and took this sprite you can skip this step.

If you took a pre-made sprite like this you will have to put in a little bit of extra work by changing it into a basic sprite like the example before. To do this select the brightest colour of the face and colour the whole image in it. Your end figure may not look exactly right.

To fix this problem you're going to have to figure out the proper proportions of your figure. In mine I have to find the proper position of the legs. This can be quite tricky.

The second figure is better isn't it? Now if you want you can erase the hair. Lastly to put in the skin tones copy the tones on the image at the top of this page and place them appropriately on your character sprite.

Step Two: Clothing and Hair

In this step we are going to create our sprites hair and clothing. To do this you must have it in your mind what it will look like on your character. Your character may look something like this with the clothing. It only has the outline of the clothing leave like this for now.

Hair is quite harder. If you don't proportion the hair correctly your characters' hair will look really stupid. To fix this put the start of the hair on the scalp where the red dots are. The dots should give you an outline of where the hair can start. You don't need to start it right there. The next image is what the hair should look like on the scalp. From there create the outline of the hair. Then colour everything that needs to be. Don't fill the skin areas or the darker areas on the sprite!! Now you've completed step 2!

Step Three: Shading

The last image in step 2 is a basic image. Now we must shade our sprite according to the darker areas of the skin is where the shading should go. So choose a darker colour of the colours in the area that you need to fill and place them in the dark areas of the skin. That should be what your sprite looks like now. It should have shading in the clothes and in skin.

Now on to the last bit of the shading the hair. I have found hair quite hard when I have done sprites. It is because of the source of the light. On my character sprite the light is hitting him head on. So all the shading should be done on the edge of the hair like so. You have almost completed your character sprite all that is left is the eyes and a few touches here and there.

Step Four: Touching Up

We have now come to the end of my tutorial all that is left are the eyes and a bit of shading. If you want you can leave the last bit of shading but if you want it looking the best it possibly can than I'd advise you to take the next steps.

The eyes are the easiest part on the sprite. The only thing you have to know is where to put them. Usually it is just above the ears, but on mine they will be in line with them. To make the eyes just put a 3 pixels long black line, then select the colours of the eyes and make them 2 pixels long going down. Now all you have to do is put in the white. At the bottom have the brightest white and at the top make it a darker shade for a better look.

Finally you'll probably notice that we could do a bit more to the shading. To fix this use

your perspective on where the light is coming from and mostly work on that. Mine because it is coming from the front means that all my shading is based on the border. If your light is coming from the left most of your shading should be on the right, if your light is coming from the right your shading is mostly on the left. And finally if your light is coming from behind shade most of the area except around the borders. Oh! And if you want you can also add a mouth.

Conclusion

Now I have come to the end of my tutorial. Remember these are only the basics. There are many combinations out there and I recommend you try them all. They will all give you something new to learn. I also recommend you create your own style of sprite creation my tutorial is only made to guide you and give you advice. Lastly if you wish to further research on sprite tutorials I suggest you go and look at Khins'.

Thank you for using my tutorial,
Flicky.