

Section 1:

The Easy Stuff

Covering ECHO, STRING, DATE, VARIABLES, INCLUDE, REQUIRE, and HEADER

Before we get into learning how to use PHP, we need to briefly explore what PHP is...but I'm not going to go into some long drawn out history of PHP and why it's here. Also, it is good to be familiar with HTML before you proceed, as you will need to be able to copy and paste code in web pages.

In the world of the wide web, there are two general types of coding languages: "Server side" and "Client side". This means that one type is run, or interpreted, on the web server itself, and the other is run in your web browser.

Think of it this way...

Server side: You order a sandwich at a restaurant. It comes to you already prepared and ready to eat.

Client side: You order a sandwich at a restaurant. Individual sandwich parts (bread, meat, lettuce, tomatoes, condiments) are delivered to you separately and you have to assemble them before you eat.

PHP is a Server side language. All of the processing is done on the web server itself, and the result is delivered to your web browser as HTML (which, by the way, is a Client side language). Your web server must also have PHP installed in order for it to work! Most web hosting providers have PHP installed, so if you are in doubt, simply ask them.

Things To Note:

There are three things that need to happen before the web server correctly runs a PHP command.

1. The filename must be "file.php" instead of the usual "file.htm" or "file.html". If it has the HTM or HTML extension, the PHP engine on the web server will ignore it. *(There are exceptions to this, but we won't go into them right now.)*
2. The PHP code must be within the <?php and ?> tags. If it is not, the PHP engine on the web server will ignore it.
3. Each PHP line must end with a Semicolon. If it does not, you're going to get some cryptic error messages.

Now, Let's Code!

We're going to tackle the basics to start with. Getting PHP to write things to the web browser, what variables are and how to use them...examples you can use in just about every web page you create.

Let's start with the most basic one.

The PHP Echo command

We're going to tell PHP to output something to the screen. Keep in mind that PHP can be used in conjunction with HTML, but we are not showing the code in this example, to keep it simple:

```
<?php echo "Creamy Bagels"; ?>
```

(I hate the "Hello World" sample that every other book in the world uses, so I'm using "Creamy Bagels" instead.)

Let's dissect the command example bit by bit, shall we? It helps to do this when you are looking at a LOT of PHP code...because, trust me, it can look like a jumbled blurry mess sometimes if you don't take it piece by piece! Next page, please, if you will...

<?php - tells the server to process this as php code...

echo "Creamy Bagels" ; - tells the server to write what's in the quotes to the screen, and that the semicolon is ending this particular command...

?> - tells the webserver, "OK, I'm done with PHP for now. Back to regular HTML".

Pretty simple when you look at it that way, yes?

"OK, that's cool..but what if I want to see quotes on my screen?"

This can be done by "escaping" the PHP code for what you want to show up in quotes. Let's use the example from above...

```
<?php echo "Creamy Bagels"; ?>
```

If you wanted to see quotes around Creamy Bagels, you would use the following code instead:

```
<?php echo "\"Creamy Bagels\""; ?>
```

Using `\` tells PHP that you want a quotation mark to appear. Remember this - we're going to use it later!

Pull some Strings...

In PHP, as a general rule, a String is any line of text contained within quotation marks. You can use either double quotation marks (") or single quotation marks also known as apostrophes (') in a string. Strings could be considered the building blocks of PHP, considering most all data is going to come from what's in a string.

```
<?php
$double = "quotation marks.";
$single = 'single quotes.';
?>
```

When using single quotes, you need to "escape" the apostrophe with the slash (just like you would with double quotation marks) if you wish to display it in the output text...

```
<?php
echo 'Wouldn\'t you like a bagel?';
?>
```

Special commands within strings...

There are some "secret commands" you can use within strings to manipulate the output text:

\n: makes a new line

\r: a carriage return

\t: a tab

\\$: shows a dollar sign - remember PHP will be looking for a variable if you want to display a dollar sign and don't use a slash...and throw an ugly error!

Using Variables

A Variable in PHP, simply put, is one thing that means another thing or things - a "container" if you will. It can represent text, numbers, calculations, and more.

Variables are quite powerful, and if you mess 'em up, they'll come get you in the middle of the night.

Declaring a variable is easy. No spaces in the variable name, please - PHP doesn't like that...

```
<?php
$this_thing = "The Other Thing";

?>
```

Now before you go off saying "What the heck would I need THAT for?", remember that variables are very useful...especially if you are PHP Include-ing other files (Like the foreshadowing there? Do ya?)

Real World Usage For Variables

Here's an example of how you can use a variable in the real world: show the current date on your website.

```
<?php
$today = date("F j, Y");
echo "$today";
?>
```

This example sets the date command as a variable called "\$today", and uses echo to display it on the screen.



And now, for a quick tangent...

More about the "DATE" command - it is very versatile and flexible - see the guide below to use it to it's potential!

The DATE command

Expanding on the above example, here are the options for DATE and TIME display:

Time:

- a: am or pm
- A: AM or PM
- g: Hour without leading zeroes (1-12)
- G: Hour in military time without leading zeroes (0-23)
- h: Hour with leading zeroes (01-12)
- H: Hour in military time with leading zeroes (00-23)
- i: Minute with leading zeroes (00-59)
- s: Seconds with leading zeroes (00-59)

Days:

- d: Day of the month with leading zeroes (01-31)
- j: Day of the month without leading zeroes (1-31)
- D: Day of the week abbreviations (Sun – Sat)
- l: Day of the week (Sunday – Saturday)
- w: Day of the week without leading zeroes (0-6)
- z: Day of the year without leading zeroes (1-365)

Months:

- m: Month of the year with leading zeroes (01-12)
- n: Month of the year without leading zeroes (1-12)
- M: Month abbreviations (Jan – Dec)
- F: Month names (January – December)
- t: Number of days in the month (28-31)

Years:

- L: Displays 1 if it is a leap year, 0 if not
 - Y: Year in 4-digit format (2006)
 - y: Year in 2-digit format (06)
- Other Date Formats:
- r: Full date, including timestamp and timezone offset (O)
 - U: Number of seconds since the Unix Epoch (Jan. 1, 1970)
 - O: Offset difference from Greenwich Meridian Time (GMT). 100 = 1 hour, -100 = -1 hour

And now...back to Variables!