

# Week 3

## Chapter 3: Introduction to PHP

---

CIS 86 • PHP and MySQL • Mission College

# Getting Started Tonight

- Start WAMP/MAMP on your computer.
  - You can use the classroom computers if you want, but it will be painful.
- Create a PHP file.
  - Put the PHP file in your WAMP/MAMP folder.
  - Use your text editor to add the basic HTML code from the class blog.
  - <http://cis086.blogspot.com/>

# USB Thumb Drive

- Carry your data back and forth between class and home.
- Your thumb drive should never be the only copy of your code.
- Format your thumb drive for PC so you can use it on the PC's in the classroom. (You can use your PC-formatted drive on a Mac at home if you need to.)
- Establish a routine of syncing your thumb drive with your primary code base at home.
- You could also use Google Drive or Dropbox or other cloud services for this purpose.
- *(In the old days, we had the same rules for floppy disks.)*

# Running PHP Code

- Your code **must** be in the WAMP or MAMP server directory.
- The WAMP or MAMP server **must** be running.
- You **must** access the code using http / localhost.
- Windows / WAMP:
  - Files: `c:\wamp\www`
  - Website: `http://localhost/index.php`
- Mac / MAMP:
  - Files: `/Applications/MAMP/htdocs`
  - Website: `http://localhost:8888/index.php`

# File naming rules

- If your file has only HTML in it, but no PHP code, you may name it *filename.html*.
- If your file has PHP code in it, you should **(must)** name it *filename.php*.
  - Exceptions
- If your file has only HTML in it, but no PHP code, you **may** name it *filename.php*.
- File names should be all lower case unless the book or the assignment instructs you otherwise.
- File names should not contain spaces or punctuation.

# Directories

- We want the user to type in as little of the URL as possible.
- If the user types in the directory but no file path, **and** if your directory has a file with the name `index.html` or `index.php`, the Apache server will display that index file by default.
- So name the home page in any folder with the file name `index.html` or `index.php`.
- I will expect your home folder to have a file with the name `index.html` or `index.php`.
- Don't make me guess what your file name is.

# index.html vs. index.php

- If your directory has both an `index.html` file and an `index.php` file, the Apache server must choose which file to display.
- By default, the Apache server is supposed to choose `index.html` first.
- However, sometimes it seems as if the Mission College PHP server chooses the `index.php` file first. (So its configuration is altered from the default.)
- Solution: make sure there is only an `index.html` file or an `index.php` file in the directory, but not both.

# PHP Code Blocks

PHP code blocks can go directly inside HTML code.

```
<?php
    echo "I wrote this in code block one.\n";
?>
<p>This paragraph is HTML.</p>
<?php
    echo "I wrote this in code block two.\n";
?>
```



# Comments

```
// This one line is commented out
```

```
/*
```

```
    This whole block is commented out.
```

```
    The comment can span many lines.
```

```
    Fourscore and seven years ago.
```

```
*/
```

# Constants

```
define ("FAVORITE_LANGUAGE", "PHP");
```

```
echo "My favorite programming language is " .  
    FAVORITE_LANGUAGE;
```

# PHP Language Constructs

- echo
- print
- include
- require
- isset
- Language constructs don't need parentheses (but you may use them; they are not prohibited either)
- <http://www.phpknowhow.com/basics/language-constructs-vs-built-in-functions/>

# Commonly-used functions

- `print_r ()`
- `substr ()`
- `trim ()`
- `array ()`
- `header ()`
- `sprintf ()`
- `file ()`
- `strlen ()`
- `strpos ()`
- `count ()`
- `str_replace()`
- `preg_match()`
- `date ()`
- `printf ()`
- `min ()`
- `exec ()`

- <http://www.boyter.org/2011/03/list-of-most-commonly-used-php-functions/>
- <http://php.net/manual/en/function.print-r.php>

# PHP Data Types

- Primitive or Scalar
  - Integer
  - Floating Point
  - Boolean
  - String
- Compound
  - Array
  - Object
- Special
  - Resource
  - NULL
- <http://php.net/manual/en/language.types.intro.php>

# Exponential Notation & Floating Point

- `$myPie = 3.14e2;`
  - $3.14e2 = 3.14 \times 10^2 = 314$ , but it is stored as a floating point value because it has digits after the decimal point
- `$myPie = 314;`
  - 314 is stored as an integer
- `$myPie = 314.0;`
  - Another way to store a whole number value as floating point is to add .0 to give it a number after the decimal point

# Reassigning variable values

```
$myStuff = 1; // integer  
print_r ($myStuff);
```

```
$myStuff = 1.2; // floating point  
print_r ($myStuff);
```

```
$myStuff = "my stuff"; // string  
print_r ($myStuff);
```

```
$myStuff = fopen ("mystuff.txt"); // resource  
print_r ($myStuff);
```

```
$myStuff = array (1, 1.2, "my stuff",  
                 fopen ("mystuff.txt")); // array of 4  
print_r ($myStuff);
```

# Operators and Operands

`$z = $a * $x + $b * $y; // z = ax + by`

- Operands: `$z`, `$a`, `$x`, `$b`, `$y`
- Operators: `=` `*` `+` `*`
- Each operator has 2 operands:

- `=`

- `$z`
- `$a * $x + $b * $y;`

- `+`

- `$a * $x`
- `$b * $y;`

- `*`

- `$a`
- `$x`

- `*`

- `$b`
- `$y`



# Binary and Unary Operators

```
$x = 3;
```

```
$y = 2;
```

```
// - is a BINARY operator that subtracts
```

```
$z = $x - $y;
```

```
// - is a UNARY operator that negates
```

```
$w = -$x;
```

# Operator Precedence

```
$a = 5;  
$b = 6;  
$x = 7;  
$y = 8;  
$z = $a * $x + $b * $y; // z = ax + by  
print_r ($z);
```

- First : \*
- Second : +
- Last : =

# Logical Operator Precedence

```
$a = true;  
$b = true;  
$x = true;  
$y = false;  
$z = $a && $x || $b && $y;  
print_r ($z);
```

- First : && (both of them)
- Second : | |
- Last : =

# Associativity

- **Left to Right**

- `$a + $b + $c + $d; //` is the same as

- `(( $a + $b ) + $c) + $d;`

- `$a - $b - $c - $d; //` is the same as

- `(( $a - $b ) - $c) - $d;`

- `// BUT NOT` the same as

- `$a - ($b - ($c - $d));`

- **Right to Left**

- `$x = $y = $z; //` is the same as

- `$y = $z; // THEN`

- `$x = $y;`

# Increment and Decrement

```
$x = 3;
```

```
$y = 2;
```

```
$x = $y++; // POST-increment is the same as
```

```
$x = y; // THEN
```

```
$y = $y + 1;
```

```
$x = ++$y; // PRE-increment is the same as
```

```
$y = $y + 1; // THEN
```

```
$x = $y;
```

# Conditional Operator

```
$BlackjackPlayer1 = 20;  
$Result = "Player 1 is " .  
    ($BlackjackPlayer1 <= 21 ?  
        "still in the game." :  
        "out of the action.");  
echo "<p>", $Result, "</p>";
```

- Prototype:
- **condition** ? **value\_if\_true** : **value\_if\_false**
- **value\_if\_true** and **value\_if\_false** may be full statements but they may also be simply expressions

# Precedence Associativity Example

```
$subtotals = array ( 42, 314 );  
$index = 0;  
$x = 1; $y = 2; $z = 3; $w = 4;  
$a = 4; $b = 5; $c = 6;  
$d = 7; $e = 8; $f = 9;  
$g = 10; $h = 11; $j = 12;
```

```
// Precedence High to Low: [ ] ++ -- * / % + - ?: =  
// All go Left to Right except =
```

```
$sum = $subtotals[$index++] + $x * $y / $z * $w  
+ $a - $b + $c * $d % $e + --$f  
+ ($g>0 ? $h : $j);
```

# Variable names

```
// Must start with a letter or underscore  
// May use letters, underscores, and numbers  
// http://php.net/manual/en/language.variables.basics.php
```

```
$xyz123 = 17; // ok to start with a letter and have number  
$_startsWithUnderscore = 42; // ok to start w/underscore
```

```
$123startsWithNumber = "xyz"; // NO, cannot start w/number
```

```
$inside_underscore = "Can I do it?"; // ok underscores  
$double__underscore = "Really"? // ok to have 2 underscore
```

```
$embedded space = 3.14e2; // NO cannot have space inside
```

```
 $"embedded space" = 3.14e2; // NO this does not help
```



# Arrays

```
$cities = array (  
    "Alviso", "Belmont", "Campbell",  
    "Danville", "Eureka", "Fremont",  
    "Gilroy", "Hayward", "Irvine",  
    "Jackson" );  
print_r ($cities);
```

# Multidimensional Arrays

```
$countySeats = array (  
    array ("Alameda", "Oakland"),  
    array ("Contra Costa", "Martinez"),  
    array ("San Benito", "Hollister")  
    array ("San Joaquin", "Stockton"),  
    array ("Santa Clara", "San Jose"),  
    array ("Solano", "Fairfield"),  
    array ("Yolo", "Woodland"),);  
print_r ($countySeats);
```

# Bracket Syntax

(PHP 5.4+ only)

```
$countySeats = [  
    [ "Alameda", "Oakland" ],  
    [ "Alpine", "Markleeville" ],  
    [ "Amador", "Jackson" ],  
    [ "Butte", "Oroville" ],  
    [ "Calaveras", "San Andreas" ],  
    [ "Colusa", "Colusa" ],  
    [ "Contra Costa", "Martinez" ],  
    [ "Del Norte", "Crescent City" ] ];  
print_r ($countySeats);
```

# Functions

```
$numberOfCities = count ($cities);
```

```
echo $numberOfCities;
```

# Variable Scope

- A variable declared inside a function is visible only within that function.
- A variable that is a function parameter is visible only within that function.
- Global variables, declared outside any function, are visible outside all functions.
- Global variables can be made visible within a function by using the keyword *global*.

# Global variable example

```
$languages = [ "php", "python", "perl", "pascal",  
              "prolog" ];
```

```
function print_languages () {  
    global $languages;
```

```
    for ($i=0; $i<count($languages); $i++) {  
        echo $languages . "<br />\n";
```

```
    }
```

```
}
```

# Superglobals

Basic information	\$_SERVER
Forms	\$_GET \$_POST
Uploading files	\$_FILES
State information	\$_COOKIE \$_SESSION