Announcements

- Password protected sections
- Do not post PHP projects' code on the web

htdocs

- How to access htdocs in the different platforms
  - Windows → C:\xampp\htdocs
  - Mac → /Applications/XAMPP/htdocs
- http://localhost
  - Provides access to your data
  - You may need to specify the port if different from 80
- You can define your own folders in htdocs
- You can rename index.php to allow listing of folder contents

index.html

- An index.html file in a folder represents the main index page for that folder
- You will not see the contents of a folder if an index.html is present
  - You need to type the file/folder name to see a particular element in the folder
- Which file represents the main index page can be defined in the Apache configuration file

Comments/echo/print

- PHP Code
  - Written in files that end with .php extension
  - Web server that has the php module will be able to process these files
  - PHP could appear embedded in html
  - Different delimeters for php code. We will use:

```
<?php
  // PHP CODE HERE
?>
```

  - Comments –

```
// everything until end of the line is a comment
# everything until end of the line is a comment
/*
  Multiple lines comment
 */
```

  - echo and print are language constructs used to send data to the browser
  - Strings can appear in single or double quotes
- **Example**: commentsEchoPrint.php

echo vs. print

- Echo vs. Print
  - print returns a value; echo does not
  - echo can print expressions separated by commas

```
echo "Hello",    "goodbye",   "aloha";  /* valid */
print "Hello",    "goodbye",   "aloha";  /* invalid */
```

Escaping into PHP Mode

- PHP Parser escapes into PHP Mode when it sees <?php
- In a PHP script you are either in PHP Mode or in HTML
- Everything within <?php ?> tags is understood by the parser to be PHP code and anything outside (e.g., html, JavaScript, etc.) is simply sent to the browser
- You can switch between modes as many times as you want. (We will have more to say about this later on)

Variables

- Variable names start with $ and can be followed by _ (underscore), letter, number
- We can use = operator to assign values to variables
- Statements are terminated with a semicolon
- No declaration or specification of variable types (implied from assignment)
- We can see variables contents by using echo or print
- Scope
  - Usually a variable is visible within the script is declared or within the function is declared
  - You can have global variables that can be "shared" across functions or scripts
- **Example**: variables.php

Superglobals

- Variables that are always present and whose values are available to all your scripts. (Each variable is actually an array of other variables)
- Global Variables
  - $_GET → variables provided to a script through the GET method
  - $_POST → variables provided to a script through the POST method
  - $_COOKIE → variables provided to a script via a cookie
  - $_FILES → variables provided to a script through file uploads
  - $_SERVER → provides information regarding headers, script locations, file paths
  - $_ENV → variables associated with server environment
  - $_REQUEST → variables provided to a script via user input mechanisms
  - $_SESSION → has variables currently registered in a session
- **Example**: globals.php

Data Types

- Types
  - integer (e.g., 4,5)
  - float/double (e.g., 8.9, 10.5)
  - string (e.g., "Mary", 'Happy Hour')
  - boolean (true or false)
  - array → ordered set of keys and values
  - object → class instance
  - resource → external resource (e.g., database)
  - NULL → uninitialized variable
- You can tell the variable's type by using gettype or is_* family of functions
  - is_bool, is_int, is_string, is_double,
  - is_numeric, is_resource, is_null, is_array
- When printing booleans true is represented as 1 and false as an empty string
- In the context of a test expression, zero, an undefined variable, or an empty string will be converted to false. All other values will evaluate to true
- **Example**: types.php

Operators

- Assignment Operator (=)
  - We have an expression when we assign a value to a variable
  - **Example:**

    echo $age = 10; // assigns 10 and expression evaluates to 10
- Typical Mathematical operators are available

  +, - , * , / , %
- String concatenation through a period
  - **Example:** “Mary”.”land”
- Typical compound assignment operators are available

  (+=, -=,*=,/=,%=,.=)
- Increment/decrement operators (++/--) in both pre/post configuration are available
- **Example:** operators.php

Operators

- Following Operators are available

  <, <=, >, >=, !=, ==

- Strings can be compared using == or ===

Example: "john" == "john“

- **==** vs. **===**

  === → Equivalency with regards to both value and type

- **Example:** comparisons.php

Casting

- settype() can be used to change the type of a variable

  Example: settype($myVariable, 'double')

- You can cast from one type to another using

  (<targetType>)$variable

  <targetType> can assume the values

  double, string, integer, boolean

- **Example**: casting.php

Constants

- You can define constants by using php's define() method
- **Example**:

```
define("PI", 3.14);
echo "Value of our constant is: ".PI;  // no need for $
```

- **Predefined Constants**
  - __FILE__ name of the file the PHP engine is currently reading
  - __LINE__ current line number of the file
  - PHP_VERSION version of PHP interpreting the script

Logical Operators

- Typical logical operators available
  - || (logical or)
  - && (logical and)
  - ! (negation)
- **xor** → left or right is true but not both (one must be true for the expression to be true)
- **and** *and* **or** correspond to && and || (the only difference is the precedence) and are available to make expressions clearer

Precedence Table

++, --,
(cast)
!
/,*,%
+,-
<,<=,=>,>
==, ===, !=
&&
||
=,+=,-=,/=*=,%=,.=
and
xor
or

- **Complete Precedence Table at:**
  - **http://us2.php.net/manual/en/language.operators.precedence.php**

Conditional Statements

- **if statements** → syntactically and semantically as in Java
- **if else statements** → syntactically and semantically as in Java
- **Cascaded if statements** → syntactically and semantically as in Java. The only difference if that elseif can replace else if
- **ternary operator (? :)** → syntactically and semantically as in Java.
- **Example:** cascadedIf.php
- **switch statements** → syntactically and semantically as in Java with the exception that the controlling expression does not need to be an integer/char expression (e.g., in php, it can be a string)
- **Example:** switch.php

Iteration Statements

- while loop → syntactically and semantically as in Java
- **Example:** while.php
- do while loop → syntactically and semantically as in Java
- **Example:** doWhile.php
- for loop → syntactically and semantically as in Java
- **Example**: multiplicationTable.php

Functions

- Two types: built-in and user-defined
- **Built-in Examples:**
  - **strtoupper** → takes a string as an argument and returns the string in uppercase
  - **abs** → returns the absolute value
- General Format for User-Defined Function

```
function <functionName>($param1, $param2, ...) {
    // function body
}
```

- Unlike variables, function names are not case sensitive
- **Example:** function.php
- We can use return to return values from a function
- **Example:** maximum.php
- You can check the existence of a function by using the function **function_exists** (returns true if the function exists and false otherwise)

Functions

- **Variable Passing mechanism**
  - PHP uses pass-by-value by default
  - You can pass a variable by reference by using an &
  - **Example:** passByValueRef.php
- **Variable Scope**
  - Variables declared in a function are local to the function (cannot be accessed outside of the function)
  - We can access a variable declared outside of a function by using the **global** construct
  - **Example:** scope.php
- **Static**
  - Allows you to preserve the value of a local variable across multiple function calls
  - **Example:** static.php
- **Default Arguments**
  - You can provide default argument values for parameters by assigning a value
  - Keep in mind that once you have given an argument a default value, all subsequent arguments should also be given default values
  - **Example:** defaultArguments.php
- **Recursion is supported**