Institiúid Teicneolaíochta Chorcaí
Cork Institute of Technology

SOFT7008
Server Side Web Development

Introduction to PHP

Méabh O'Connor
meabh.oconnor@cit.ie

Server Side Scripting

- PHP – is a server side scripting language

  Think of it as a *plug-in to your web server* that enables you to do more than just send exact copies of the files that the web browser asks for.

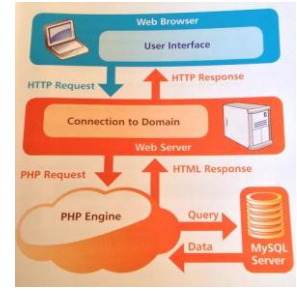
- PHP scripts are computer programs

  These scripts can do tasks like retrieve up-to-the-minute information from a database and use it to *generate a web page on the fly* before sending it to the browser that requested it.

Server Side Scripting

- Key difference between a client side script e.g. JavaScript and a server side script like PHP is the **stage of loading the web page** at which these **scripts are executed**.
- Client side scripts are executed by the *browser* **after** downloading the web page from the web server.
- Server side scripts like PHP are executed by the *web server* **before** the web page is sent to the browser.

Server Side Scripting



source: PHP & MySQL *In easy steps* - Mike McGrath

Introduction to PHP

- PHP script code is embedded within a web page
- PHP code is executed by web server to produce HTML
- Produced HTML *replaces* PHP script in web page
- **So new modified HTML web page is sent to browser**

Introduction to PHP

- A web page containing PHP code must be saved with the *file extension* of **.php** to be processed by the scripting engine
- PHP code is never sent to a client's web browser; *only the output of the processing is sent to the browser*

Introduction to PHP

Snippet of PHP code embedded in the body of a HTML web page:

```
<body>
  <p>Today&rsquo;s date (according to this web server) is
    <?php

    echo date('l, F dS Y.');

    ?>
  </p>
</body>
```

executed

```
<body>
  <p>Today&rsquo;s Date (according to this web server) is
    Wednesday, April 1st 2009.    </p>
</body>
```


Introduction to PHP

- **<?php** marks the *start* of an embedded PHP script
- **?>** marks the *end* of a PHP script

- The web server is asked to *interpret* everything between these two delimiters, and to convert it to regular HTML code *before* it sends the web page to the requesting browser.

Introduction to PHP

Most of this code is HTML but the lines between
<?php
.....
?> are PHP

```
<body>
  <p>Today&rsquo;s date (according to this web server) is
    <?php

    echo date('l, F dS Y.');

    ?>
  </p>
</body>
```


Introduction to PHP

- The actual HTML that is sent to the browser after the PHP has been executed on the web server is listed below :

```
<body>
  <p>Today&rsquo;s Date (according to this web server) is
    Wednesday, April 1st 2009.    </p>
</body>
```

- Notice that all signs of PHP code have been **replaced** by standard HTML

Advantages of Server Side Scripting

1. **No browser compatibility issues**
   PHP scripts are interpreted by the web server alone, so no concerns about whether the language you are using is supported by the visitor's browser.

2. **Access to Server Side Resources**
   If we insert the date on a web page using JavaScript, we would only be able to display the date *according to the computer on which the web browser is running*.
   But of greater importance – is the ability to extract data from the web server database(s) and insert it into the web page.

Advantages of Server Side Scripting

3. Reduced load on the client.
   JavaScript can delay the display of a web page on slower computers significantly, as the browser must run the script *before* it can display the web page. With server-side code, this burden is passed to the web server.

PHP – Basic Syntax

- PHP will be familiar to anyone with an understanding of C, C++, C#, Java, JavaScript, Perl or any other C derived language.
- Each script contains a series of commands or statements.
- Each statement is an instruction that must be followed by the web server before it can proceed to the next statement.
- PHP statements are terminated with a semi-colon ;

PHP – Basic Syntax

- This is a typical PHP stamement:

```
echo 'This is a <strong>test</strong>!';
```

- An `echo` statement simply takes the text it is given, and inserts it into the page's HTML code at the position of the PHP script that contains it.

PHP – Basic Function Calls

- A **function** is a logical *group* of individual statements that *performs a specific task*
  - To execute a function, you must invoke, or **call**, it from somewhere in the script
- A **function call** is the function name followed by any data that the function needs
  - The data (in parentheses following the function name) are called **arguments** or **actual parameters**
- Sending data to a called function is called **passing arguments**

PHP – Basic Function Calls

```
echo date('l, F dS Y.');
```

- This statement *invokes* a ***built-in PHP function*** called **date()**
- A list of all the built in PHP functions and their requirements is available in the function reference at :

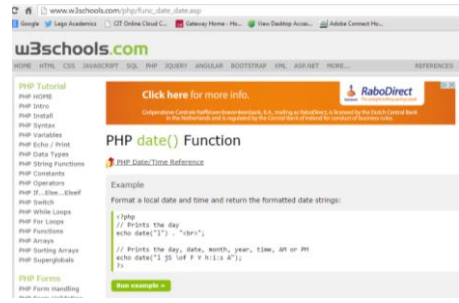
http://www.w3schools.com/php/func_date_date.asp

- In this call it is passed a string of text **'l, F dS Y.'** indicating that the date string returned should be in the following format:

PHP – Basic Function Calls

- l (lowercase 'L') - A full textual representation of a day
- F - A full textual representation of a month (January through December)
- d - The day of the month (from 01 to 31)
- S - The English ordinal suffix for the day of the month (2 characters st, nd, rd or th
- Y - A four digit representation of a year

PHP – Basic Function Calls



PHP – Basic Function Calls

- For functions that take more than 1 parameter – we separate each parameter with a comma **,**
- For functions that take no parameter – we still need to specify the parentheses **()**

PHP – Creating Multiple Code Declaration Blocks

- For *multiple* script sections in a web document, *include a separate code declaration block for each section*

```
...
</head>
<body>
<h1>Multiple Script Sections</h1>
<h2>First Script Section</h2>
<?php echo "<p>Output from the first script section.</p>";
?>
<h2>Second Script Section</h2>
 <?php echo "<p>Output from the second script
 section.</p>";?>
</body>
</html>
```


PHP – Creating Multiple Code Declaration Blocks

- PHP code declaration blocks execute **on the web server** before a web page is sent to a client

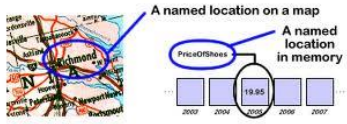
HTML after PHP has been executed:

```
...
</head>
<body>
<h1>Multiple Script Sections</h1>
<h2>First Script Section</h2>
<p>Output from the first script section.</p>
<h2>Second Script Section</h2>
<p>Output from the second script section.</p>
</body>
</html>
```


PHP – Creating Multiple Code Declaration Blocks

Final output as it is displayed in web browser.

PHP – Variables



- The data used during the execution of a program can be stored in *named* memory locations called **variables.**
- Associating a name with a memory location allows us to choose names that give *meaning* to the contents of the memory location

```
$PriceOfShoes = 19.95;
```


PHP – Variables

- The name you assign to a variable is called an **identifier**
- The following rules and conventions must be followed when naming a variable:
  - Identifiers must begin with a dollar sign ($)
  - Identifiers may contain uppercase and lowercase letters, numbers, or underscores (_). The first character after the dollar sign must be a letter or an underscore character.
  - Identifiers cannot contain spaces
  - Identifiers are ***case sensitive***

PHP – Variables

- In PHP – variable names are **case sensitive**. So:
  - $name
  - $Name
  - $NAME
  - refer to 3 **different** variables.

PHP – Case Sensitivity

- However – most programming language constructs in PHP are **case insensitive**

```
<?php
echo "<p>Explore <strong>Africa</strong>, <br />";
Echo "<strong>South America</strong>, <br />";
ECHO " and <strong>Australia</strong>!</p>";
?>
```

- It is good programming practice to be consistent in your use of case throughout your program.

PHP –Declaring and Initializing Variables

- Specifying and creating a variable name is called **declaring the variable**
- Assigning a first value to a variable is called **initializing the variable**
- In PHP, you must *declare and initialize* a variable in the same statement:

```
$votingAge = 18;
```

  - If you declare a variable in PHP without initializing it – an error will be generated.

PHP –Modifying Variables

- You can modify a variable's value at any point in a script

```
$SalesTotal = 40;
echo "<p>Your sales total is
        $SalesTotal</p>";
$SalesTotal = 50;
echo "<p>Your new sales total is
       $SalesTotal</p>";
```


PHP – Comments

Comments are nonprinting lines placed in code that *do not get executed*, but provide helpful information

```
$testvariable = 1 + 1;  // Assigns a value of 2
$testvariable = 1 - 1;  // Assigns a value of 0
$testvariable = 2 * 2;  // Assigns a value of 4
$testvariable = 2 / 2;  // Assigns a value of 1
```

comments

Comments are an important way of improving the readability of your code - vital when code is in the maintenance phase of the software life cycle

PHP Comments

- For comments that span multiple lines of text e.g a code file header – start the comment with **/*** and end it with ***/**

```
/*   this is a multi – line comment
     to make the code self – documenting
*/
```

- The PHP interpreter will ignore everything between these 2 delimiters.