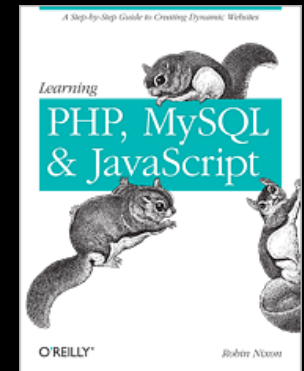


Introduction to PHP

Chapter 3

Dr. Charles Severance

To be used in association with the book:
PHP, MySQL, and JavaScript by Robin Nixon



open.michigan

Unless otherwise noted, the content of this course material is licensed under a Creative Commons Attribution 3.0 License.

<http://creativecommons.org/licenses/by/3.0/>.

Copyright 2011, Charles Severance



About the PHP Language

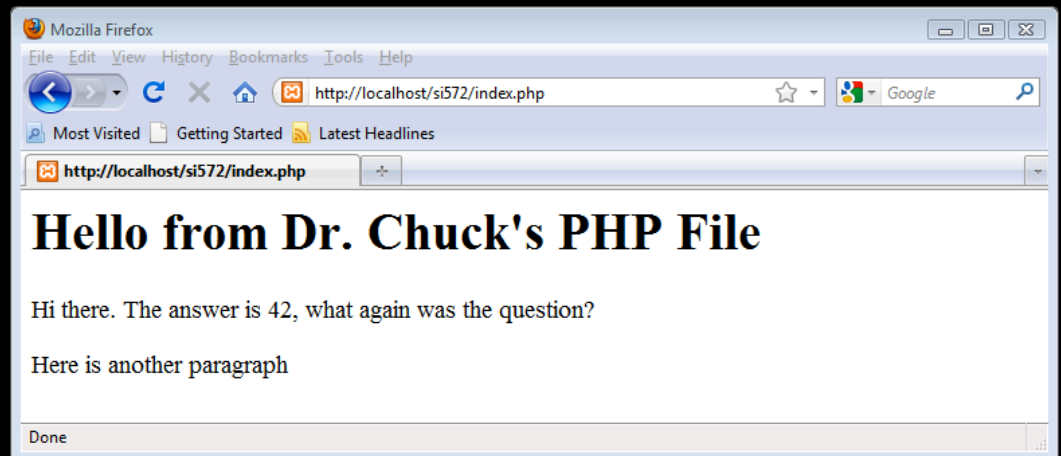
- Syntax is inspired by C
 - Curly braces, semicolons, no significant whitespace
- Syntax inspired by perl
 - Dollar signs to start variable names, associative arrays
- Extends HTML to add segments of PHP within an HTML file.

Philosphy of PHP

- You are a responsible and intelligent programmer
- You know what you want to do
- Some flexibility in syntax is OK - style choices are OK
- Lets make this as convenient as possible
- Sometimes errors fail silently

```
<h1>Hello from Dr. Chuck's HTML Page</h1>
<p>
<?php
    echo "Hi there.\n";
    $answer = 6 * 7;
    echo "The answer is $answer, what ";
    echo "was the question again?\n";
?>
</p>
<p>Yes another paragraph.</p>
```

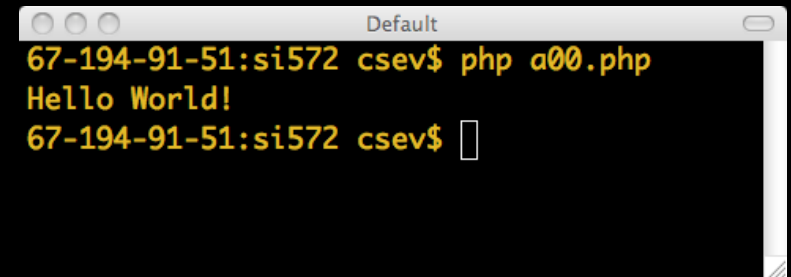
```
<h1>Hello from Dr. Chuck's HTML Page</h1>
<p>
<?php
    echo "Hi there.\n";
    $answer = 6 * 7;
    echo "The answer is $answer, what ";
    echo "was the question again?\n";
?>
</p>
<p>Yes another paragraph.</p>
```



PHP From the Command Line

- You can run PHP from the command line - the output simply comes out on the terminal
- It does not have to be part of a request-response cycle

```
<?php
    echo("Hello World!");
    echo("\n");
?>
```

A terminal window titled "Default" with a light gray title bar and three window control buttons (red, yellow, green) on the left. The terminal content is as follows:

```
67-194-91-51:si572 csev$ php a00.php
Hello World!
67-194-91-51:si572 csev$ █
```

Key Words

abstract and array() as break case catch class
clone const continue declare default do else
elseif end declare endfor endforeach endif
endswitch endwhile extends final for foreach
function global goto if implements interface
instanceof namespace new or private protected
public static switch \$this throw try use var
while xor

<http://php.net/manual/en/reserved.php>

Variable Names

- Start with a dollar sign (\$) followed by a letter or underscore, followed by any number of letters, numbers, or underscores
- Case matters


```
$abc = 12;  
$total = 0;  
$largest_so_far = 0;
```

```
abc = 12;  
$2php = 0;  
$bad-punc = 0;
```

<http://php.net/manual/en/language.variables.basics.php>

Expressions

- Completely normal like other languages (+ - / *)
- More aggressive implicit type conversion

```
<?php
    $x = "21" * 2;
    echo($x);  42
    echo("\n");
?>
```

Output

- **echo** is a language construct - can be treated like a function with one parameter. Without parenthesis, it accepts multiple parameters.
- **print** is a function - only one parameter but parenthesis are optional so it can look like a language construct

```
<?php
    $x = "21" * 2;
    echo $x;
    echo ("\n");
    echo $x, "\n";
    print $x;
    print "\n";
    print($x);
    print("\n");
?>
```

Conditional - if

- Logical operators (== != < > <= >= and or !)
- Curly braces

```
<?php
    $ans = 42;
    if ( $ans == 42 ) {
        print "Hello world!\n";
    } else {
        print "Wrong answer\n";
    }
?>
```

—————→ Hello World!

Whitespace does not matter

```
<?php
    $ans = 42;
    if ( $ans == 42 ) {
        print "Hello world!\n";
    } else {
        print "Wrong answer\n";
    }
?>
```

```
<?php $ans = 42; if ( $ans == 42 ) { print
"Hello world!\n"; } else { print "Wrong answer\n"; }
?>
```

Associative Arrays

- Like Python Dictionaries - but more powerful
- Can be key => value or simply indexed by numbers
- Ignore two-dimensional arrays for now...

Integer Indices

```
<?php
    $stuff = array("Hi", "There");
    echo $stuff[1], "\n";
?>
```

There

Key / Value

```
<?php
    $stuff = array("name" => "Chuck",
                  "course" => "SI572");
    echo $stuff["course"], "\n";
?>
```

SI572

Dumping an Array

- The function `print_r()` dumps out PHP data - it is used mostly for debugging

```
<?php
    $stuff = array("name" => "Chuck",
                  "course" => "SI572");
    print_r($stuff);
?>
```

```
Array
(
    [name] => Chuck
    [course] => SI572
)
```

Looping Through an Array

```
<?php
    $stuff = array("name" => "Chuck",
                  "course" => "SI572");
    foreach($stuff as $k => $v ) {
        echo "Key=", $k, " Val=", $v, "\n";
    }
?>
```

```
Key=name Val=Chuck
Key=course Val=SI572
```

Strings

- String literals can use single quotes or double quotes
- The backslash (\) is used as an "escape" character
- Strings can span multiple lines - the newline is part of the string
- In double-quoted strings variable values are expanded

<http://php.net/manual/en/language.types.string.php>

```
<?php
echo 'this is a simple string';

echo 'You can also have embedded newlines in
strings this way as it is
okay to do';

// Outputs: Arnold once said: "I'll be back"
echo 'Arnold once said: "I\'ll be back" ';

// Outputs: This will not expand: \n a newline
echo 'This will not expand: \n a newline';

// Outputs: Variables do not $expand $either
echo 'Variables do not $expand $either';
?>
```

```
<?php
echo "this is a simple string\n";

echo "You can also have embedded newlines in
strings this way as it is
okay to do";

// Outputs: This will expand:
//           a newline
echo "This will expand: \na newline";

// Outputs: Variables do 12
$expand = 12;
echo "Variables do $expand\n";
?>
```

```
<?php
    echo 'This is a test'; // This is a c++ style comment
    /* This is a multi line comment
       yet another line of comment */
    echo 'This is yet another test';
    echo 'One Final Test'; # This is a shell-style comment
?>
```

<http://php.net/manual/en/language.basic-syntax.comments.php>

Summary

- This is a sprint through the language features of PHP