PHP

Syntax, Variables, If-Statements, and Loops

COS216
AVINASH SINGH
DEPARTMENT OF COMPUTER SCIENCE
UNIVERSITY OF PRETORIA

PHP – OVERVIEW

- Personal Home Page (PHP)
  - Acronym changed to: PHP Hypertext Preprocessor (PHP)

- Server-side scripting language
- Specifically design for web development

- File extension: .php
- MIME type: text/php
  - Not official, there are many other unofficial MIME type for PHP

PHP – OVERVIEW

- Released in 1994/1995
- Current version is 8.2
- Maintained by the PHP Group

- **Wheatley uses PHP 7.3**

- Typically used with embedded HTML
- An interpreter is needed to interpret/execute the PHP code
  - Common Gateway Interface (CGI) module handles this
  - Typically, part of the webserver, such as Apache

PHP – TYPING

- PHP is dynamic weakly typed
- Variables can change their type over time
- The underlying interpreter manages memory on behalf of PHP

PHP – SCRIPTS

- PHP has opening and closing tags
- Only code within these tags will be executed

```
<?php
        // Some PHP code
?>
```

PHP – COMMENTS

- Standard single- and multi-line comments

```
<?php
        // Single-line comment

        /*
                Multi-line
                comment
        */
?>
```

PHP – ECHO

- Echo prints text which is then returned by the interpreter
- Everything that echo will be displayed in the client's browser
- Can be used with or without parentheses (brackets)
- Echo and print is almost identical (print has a return value, echo does not)

```
<?php
        echo "Hello";
        echo("world!");
        print "Hello";
        print("world!");
?>
```

PHP – HTML

- PHP can be embedded into HTML
- Note that an interpreter is required to execute the PHP code
  - Hence the page has to be opened through Apache
  - Manually opening the HTML document (double-clicking on the .html file) will not execute the PHP code

```
<!DOCTYPE html>
<html>
        <body>
                <h1>COS216</h1>
                <?php echo "Hello World!"; ?>
        </body>
</html>
```

PHP – HTML

- Entire HTML pages can be echoed by PHP

```php
<?php
        echo "<!DOCTYPE html>";
        echo "<html>";
        echo "<body>";
        echo "<h1>COS216</h1>";
        echo "Hello World!";
        echo "</body>";
        echo "</html>";
?>
```

PHP – VARIABLES

- Variables are dynamically typed
- Variables start with the dollar ($) sign
- Supports primitive types (bool, int, float, string, array, object, null)
- Note that Booleans can be any case (true, True, TRUE), depending on the PHP version

```php
<?php
        $someBool = True;
        $someInt = 5;
        $someFloat = 10.5;
        $someString = "Hello world!";
        $someNull = null;
?>
```

PHP – DUMPS

- Variables can be echoed or printed
- Echoed variables are converted to a string
- For debugging purposes, rather dump the variables
- Dumps will preserve the structure of variables, such as arrays or objects

```php
<?php
        $x = 1234;
        var_dump($x);
?>
```

PHP – LOGS

- Everything through echo, print, var_dump is printed to screen
- Hence the end user is able to see it
- If you want to log errors without letting the user see it, write to the error log
- Everything is written to the error log file on the server

```
<?php
        error_log("Some debugging statement");
?>
```

PHP – OPERATORS

- Operators are similar to JavaScript
- Arithmetic operators (+ - * / %)
- Can be combined with the assignment operator (+= -= *= /= %=)
- Comparison operators (== === != !== < > <= >= <>)
  - == and != type casting (checks if value is the same even if type is different)
  - === and !== without type casting (checks if value and type is the same)
  - <> not equal (same as !=)

PHP – OPERATORS

- Logical operators (! && ||)
  - PHP also allows some operators as words (and or xor)
- Increment and decrement operators ($x++ ++$x $x-- --$x)
- String operators (. .=)
- Array operators (== === != !== <> +)

PHP – STRINGS

- Strings can be encapsulated in single quotes (') or double quotes (")
- Strings do not use the + operator, but the . operator for concatenation

```php
<?php
        $someString = "Hello" . "world";
        $someString .= "!";

        // The string length
        $length = strlen($someString);

        // The position of the substring
        $position = strpos($someString, "world");

        // Replaces substrings
        $someString = str_replace("world", "earth", $someString);
?>
```

PHP – STRINGS

- Note that PHP has many inconstancies that might be confusing
- Naming convention
  - Example: strlen vs str_replace
- Parameter order
  - Often the subject variable is placed last instead of first like other languages
  - Example: The haystack parameter in str_replace is placed last

PHP – ARRAYS

- Array initialization

```php
<?php
        $array1 = array();
        $array2 = [];
        $array3 = array("value1", "value2");
        $array4 = ["value1", "value2"];
?>
```

PHP – ARRAYS

- Access or updating elements

```php
<?php
        $array = array("value1", "value2");
        echo "First element: " . $array[0];
        $array[0] = "new value";
        $array[1] .= " extension;
?>
```

- Getting the number of elements in the array

```php
<?php
        $array = array("value1", "value2");
        echo "Array size: " . count($array);
?>
```

PHP – ARRAYS

- Adding and removing elements

```php
<?php
        $array = array("value1", "value2");
        array_push($array, "value3");
        unset($array[0]);
?>
```

PHP – ARRAYS

- Arrays can be unnamed (indexed arrays) or named (associative arrays)
- Indexed arrays

```php
<?php
        $array = array(256, 850);
        echo $array[1];
        $array[0] += 42;
?>
```

- Associative arrays

```php
<?php
        $array = array("val1" => 256, "val2" => 850);
        echo $array["val2"];
        $array["val1"] += 42;
?>
```

PHP – ARRAYS

- Associative arrays can be used as object equivalents in JavaScript/JSON
- JavaScript

```
var obj = {
        "prop1" : "value",
        "prop2" : 0.123
};
```

- Associative arrays

```
<?php
        $obj = array(
                "prop1" => "value",
                "prop2" => 0.123
        );
?>
```

PHP – IF STATEMENTS

- If-elseif-else statements
- With or without scope parentheses ({})

```php
<?php
        $value = 20;
        if($value < 0)
        {
                echo "invalid";
        }
        else
        {
                if($value > 100) echo "large";
                else if($value > 50) echo "medium";
                else echo "small";
        }
?>
```

PHP – IF STATEMENTS

- If-else statements with HTML mixed.

```
<?php
        $value = 20;
        if($value < 0) :?>

        <h1>invalid</h1>

        <? else: ?>
        <div>some more HTML code here </div>
        <? endif; ?>
?>
```

PHP – SWITCH STATEMENTS

- Switch statements

```php
<?php
        $value = 1;
        switch($value)
        {
                case 0:
                        echo "no";
                        break;
                case 1:
                        echo "yes";
                        break;
                default:
                        echo "invalid";
        }
?>
```

PHP – FOR LOOPS

- For loops

```
<?php
        for($i = 0; $i < 10; ++$i)
        {
                echo "Counter: $i <br/>";
        }
?>
```

```
<?php
        for($i = 0; $i < 10; ++$i):?>
                <div> Counter: <?=$i?> <br/></div>
        <? endfor;
?>
```

PHP – FOREACH LOOPS

- For versus foreach loops

```php
<?php
        $array = [1, 2, 3, 4, 5];
        for($i = 0; $i < count($array); ++$i)
        {
                echo $array[$i];
        }
?>
```

```php
<?php
        $array = [1, 2, 3, 4, 5];
        foreach($array as $value)
        {
                echo $value;
        }
?>
```

PHP – WHILE LOOPS

- While loops

```
<?php
        $x = 0;
        while($x <= 5)
        {
                ++$x;
        }
?>
```

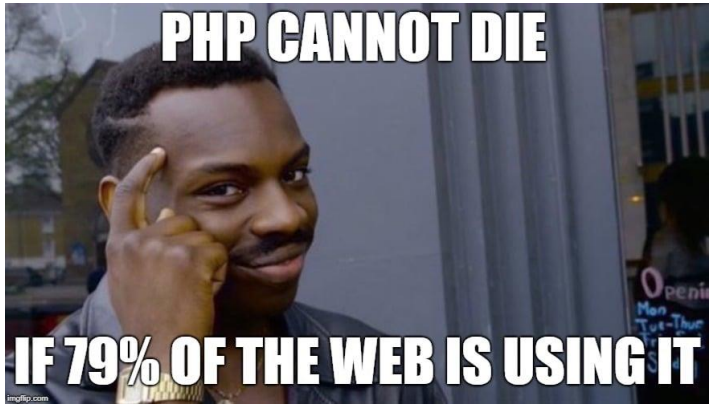
PHP – DO WHILE LOOPS

- Do while loops

```php
<?php
        $x = 5;
        do
        {
                $x--;
        }
        while($x >= 0);
?>
```


PHP
PHP EVERYWHERE


ONE DOES NOT SIMPLY
UPGRADE PHP VERSION
imgflip.com


PHP CANNOT DIE
IF 79% OF THE WEB IS USING IT


You are without doubt the worst programming language i have ever heard of.
php
But you HAVE heard of me!


PHP7
<?php
if($age > 19){
echo "Adult";
} else{
echo "Teenager";
}
?>