PHP

Tyler Harley

What is PHP?

- General purpose server-side scripting language
- Developed in 1994 out of a set of Perl scripts designed to maintain a webpage
- Rewrote as CGI binaries in C so it could work with databases and more web tools
- Interpreted through Zend Engine built into browsers
- Dynamically produces content on web pages

Symbols & Syntax

Variables

```
<?php
    $var = 10;
    $var = "string";

    $arrayA = array(2, 3.14,
    "cat");

    $arrayB = array(
                "ten" -> 10,
                "hundred" -> 100,
                20 -> 42
    );

    echo $var; //Prints "string"
?>
```

Functions

```
// Style A
function add($a, $b){
    return $a + $b;
}

// Style B
$add = function($a, $b){
    return $a + $b;
};

echo add(3, 5) . "\n";
echo $add(3, 5) . "\n";
```

Control Statements

For loops:

```
for($i=0; $i<10; $i++){
    echo $i;
}
```

```
for($i=0; i<10; echo $i, $i++);
```

If statements

```
if($a > $b)
    echo "Greater than";
elseif($a < $b)
    echo "Less than";
else
    echo "Equal";
```

While loops

```
while($i<10){
    echo "printing";
    $i++;
}
```

Expressions

```
if($a > 10 && ($b - $c) > 0)
    echo "Expression success";
```

Scope

```
$first = 20;
$second = 8;
echo $first . "\n"; // outputs 20

// general scoping test
function scopeTest(){
     echo $first . "\n";
     $first = 15;
     echo $first . "\n"; // outputs 15
     global $second;
     $third = $first + $second;
     echo $third . "\n"; // outputs 23
}

// tests the static variable qualities
function staticTest(){
     static $fourth = 0;
     $fourth++;
     echo $fourth . "\n"; // outputs 1-10
}

scopeTest();
for($i = 0; $i < 10; $i++){
     staticTest();
}
```

Functions cont.

```
function add1($a, $b){
    return $a + $b;
}
$func1 = "add1"; // func1 can be used as a function
echo $func1 . ": " . $func1(3, 5); // outputs "add1: 8";

$add2 = function($a, $b){
    return $a + $b;
};
$func2 = "add2"; // add2 has not yet been defined as a function

////////////////////////////////////

// Sum an array
function foo(){
    $list = func_get_args(); // Gets a list of all the arguments
    $sum = 0;
    foreach($list as $value){ // Loops through each $value passed to $list
        $sum += $value;
    }
    return $sum;
}
```

Classes

```
class Foo{
    // Instance data
    public $array1;
    public $array2;

    // Constructor
    public function __construct($a, $b){
        $this->array1 = $a;
        $this->array2 = $b;
    }

    // Method
    public function printArray($array){
        var_dump($array);
        echo "\n";
    }
}

$a = array("string", 20, 3.14);
$b = array("and another string, 6, 3.14);
$bar = new Foo($a, $b);

$bar->printArray($bar->array1);
```

Memory Management

- Garbage collected using reference counting
- Everything stored in symbol tables as zvars, storing reference status and reference numbers
- PHP 5.3 introduced algorithm to deal with reference loops
  1. Root buffer
  2. Depth-first search #1
  3. Depth-first search #2
  4. Free all marked zvars

Interesting Facts

- PHP includes abstract classes and interfaces
- Compilers do exist for PHP
  - HipHop
- Works alongside HTML
- A lot of easy, available functionality