Unit 4 Module 1: Introduction to PHP

Dylan Lane McDonald

CNM STEMulus Center
Web Development with PHP

September 5, 2014

Outline

Introduction to PHP

PHP, or **P**HP **H**ypertext **P**rocessor is one of the most popular server side languages in use today. PHP is a hybrid imperative/object oriented language. Although it's most common use case is in dynamic web sites, it has evolved into a general purpose programming language. One of PHP's biggest advantages is its vast array of libraries and extensions that can accomplish almost any task.

Introduction to PHP

PHP, or **P**HP **H**ypertext **P**rocessor is one of the most popular server side languages in use today. PHP is a hybrid imperative/object oriented language. Although it's most common use case is in dynamic web sites, it has evolved into a general purpose programming language. One of PHP's biggest advantages is its vast array of libraries and extensions that can accomplish almost any task.

In class, we will be authoring PHP pages and uploading them via SCP to a Linux server running PHP/Apache/mySQL. This is a very realistic use case and is commonly implemented in industry.

PHP vs. JavaScript: In Common

Despite the fact they're on opposite ends of the server/client relationship, PHP & JavaScript have a lot in common:

- Dynamically typed variables
- Hybrid imperative/object oriented paradigm
- Basic language constructs: functions, branching, loops, exception handling
- Full unit testing libraries available
- Web optimized programming language

In sum, both languages have grown beyond the bounds of their original design to fill other roles.

PHP vs. JavaScript: Not In Common

The obvious difference between the two languages is that JavaScript is a client-side and PHP is server. Naturally, there are advantages and disadvantages to each:

- PHP is server side. This requires more network traffic. **Advantage**: JavaScript
- JavaScript engines differ in behavior and consistent behavior cannot be guaranteed for the same page across browser. **Advantage**: PHP
- Servers are typically configured with better hardware and can do heavier processing than desktops or mobile devices. **Advantage**: PHP

Typically, the general rule is to do as much processing on the server side and use JavaScript to improve the user experience.

Associative Arrays

PHP has arrays just as JavaScript does. In JavaScript, we only used arrays with numeric indexes. PHP supports **associative arrays**.[1]

```
$car              = array();
$car["tires"]     = 4;
$car["cylinders"] = 6;
$car["make"]      = "Toyota";
```

Listing 1: Creating an Associative Array

Notice the array indexes can be any data type. Also note the items in the array can also be of mixed data type.

[1] Well, so does JavaScript...

Traversing an Associative Array

PHP does support the classic `for` loop for traversing arrays with all numeric indexes. With an associative array, such a loop would not be very useful.

```
foreach($car as $part => $item)
{
    echo "The car's $part is $item";
}
```

Listing 2: Traversing an Associative Array

Using the `foreach` loop, one can traverse an associative array with ease. If the array keys aren't of interest, the loop can be simplified to:
`foreach($car as $item)`

Superglobals

PHP has global variables that are accessible to every single PHP program. These are called **superglobals** and are enumerated in Table 1.

`$GLOBALS`	global variables in the program
`$_SERVER`	server side variables
`$_GET`	URL parameters
`$_POST`	form data
`$_FILES`	uploaded files
`$_COOKIE`	cookie data
`$_SESSION`	session data
`$_REQUEST`	`$_GET`, `$_POST`, and `$_COOKIE`
`$_ENV`	environmental variables

Table 1: PHP Superglobals

Unit Testing in PHP

```
<?php
require_once("/usr/lib/php5/simpletest/autorun.php");
class CanAddUp extends UnitTestCase
{
    function testOneAndOneMakesTwo()
    {
        $this->assertEqual(1 + 1, 2);
    }
}
?>
```

Listing 3: Unit Testing in PHP