php[training]

Introduction to PHP

In the beginning

“Personal Home Page” - Perl

PHP/FI - written in C,

- v1 - released 1995
- v2 - released 1997

PHP3 - “PHP: Hypertext Processor”

- Rewritten Parser, launched 1998

PHP4

- Introduces basic OOP, May 2000

PHP today

5.0	• Public/private/protected class methods & properties
5.1	• PDO: common database interface
5.2	• DateTime classes
5.3	• Namespaces • Lambda/Anonymous functions
5.4	• Traits • Short array syntax [ ]
5.5	• Generators • Native password hashing

PHP's Strengths

Easy (and cheap) to deploy

- Low barrier to entry

C-like syntax

Built for web-scripting

- Makes cookies, server, request info available easily.
- Handle form input directly

Duck-typing

- "If it walks, swims, quacks like a duck..."

Inline PHP inside of HTML

PHP's weaknesses

Not designed

- Inconsistent function signatures

No benevolent dictator, fragmented community

Poor & insecure early coding practices

Servers may configure it differently

Duck-typing

Inline PHP inside of HTML

Script execution

PHP interpreter works within a web server.

- Files matching configure file extensions are parsed.
- Usually its ".php"

Opening and closing tags indicate code to be executed

- Opening tag: "<?php"
- Closing tag: "?>" is optional if your file is all PHP
- Short echo tag "<?= $name ?>"

Can mix PHP and HTML in same file.

Anatomy of a script

```php
<?php
// Opening tags tell the php engine to compile the code

// assigning some strings
$title = “Example”;
$message = “Hello World”;

// output an HTML fragment
?>
<h1><?= $title ?></h1>
<p><? echo $message ?>. Today's is <?= date('Y-m-d') ?></p>
```

Constants

Constants hold values that cannot be changed and can be accessed globally. By convention, they are uppercase.

```php
<?php
define(“DELAY”, 60);
sleep(DELAY);
```

Variables

Starts with a dollar sign ($)

Alpha numeric names, starts with a letter or underscore

- `$total`
- `$_max`
- `$first_name`
- `$address_2`
- `$is_parsed`

Variables - beware

PHP populates “Super-globals” arrays. Don’t trust user input - Filter on Input / Escape on Output

- $GLOBALS
- $_SERVER
- $_COOKIES
- $_SESSION
- $_POST / $_GET / $_REQUEST
- $_FILES
- Full list: http://www.php.net/manual/en/reserved.variables.php

Types - Strings

Single Quotes

```
$color = ‘Green’;
```

Double Quotes

```
// outputs Green Eggs and Ham
$food = “$color Eggs and Ham”;
echo $food;
```

Types - Booleans

A variable that is TRUE or FALSE

- `$is_parsed = FALSE;`

False equivalents

- The numbers 0 or 0.0
- Empty strings “” and “0”
- Empty arrays
- NULL value

Types - Numbers

Integers

- Whole numbers, positive or negative
- `$max = 100`
- `$height = -123`

Floats (or doubles)

- Numbers with a fractional component, positive or negative
- `$book_price = 49.99`
- `$distance = 1.2e4`

Types - Arrays

Numerically indexed, starting at 0

```
$zoo = array('Lion', 'Tiger', 'Bear');
echo $zoo[0]; // outputs Lion
```

Elements within an array can be any type.

```
$bag = [5, 3.2, array('one', 'two'), 'Banana'];
echo $bag[2][0]; // outputs 'One'
```

Types - Associative Arrays

Like plain arrays, but specify keys and values.
Keys can be strings or integers.

```
// short array syntax
$address = ['city' => 'Fairfax', 'state' => 'VA'];

// adding a new key, value pair
$address['country'] = 'United States';
```

Flow Control - IF/ELSE

```php
<?php
if ($speed == 88) {
   echo “We’re going back to the future!”;
} elseif ($speed > 75) {
      echo “Almost there, keep accelerating.”;
} else if ($speed > 25) {
      echo “You need to go faster.”;
} else {
      echo “Please press the accelerator”;
}
```

Flow Control - Switch

```php
<?php
switch (strtolower($color)) {
   case "blue":
      $turtle = "Leonardo";
      break;
   case "orange":
      $turtle = "Michelangelo";
      break;
   case "purple":
      $turtle = "Donatello";
      break;
   case "red":
      $turtle = "Raphael";
      break;
   default:
      trigger_error("Unknown turtle.", E_USER_WARNING);
}
```

For Loop

Basic loop, specify starting & ending conditions and an increment.

```
for ($i = 1; $i < 5; $i++) {
    echo $i * 2 . “\n”;
}
```

Output:

```
2
4
6
8
```

Foreach loop

Useful for traversing associative arrays and using keys & values within the loop.

```
$zoo = [“Lions” => 3, “Tigers” => 2, “Bears” => 5];
foreach ($zoo as $animal => $count) {
    echo $animal . “: “ . $count;
}
```

Output:

```
Lions: 3

Tigers: 2

Bears: 5
```

While loop

Execute while a condition is TRUE.

```
$x = 'PHP';
while (strlen($x) < 10) {
  $x = $x . $x;
  echo $x;
}
```

Output:

```
PHPPHP
PHPPHPPHPPHP
```

Functions

Most basic reusable block of code.

- Name follows same rules as variables.
- Can accept zero or more arguments.
- Can only return a single value.

```php
<?php
function largest($a, $b) {
    if ($a > $b) {
        return $a;
    }
    return $b;
}
echo largest(7, "23"); // outputs 23
echo largest(120.2, -15); // outputs 120.2
```

Objects

Objects can have properties and methods that act on those properties.

```php
<?php
$student = new StdClass;
$student->name="Alex";
$student->email="alex@example.com";
$student->graduation_year = 2010;

echo $student->name . ' - ' . $student->email;
```

Reusing PHP scripts

Can load another PHP script in your current script. The code in the included file inherits your current variable scope.

```php
<?php
// sets $year, $author from a file in same directory
include(__DIR__ . "/copyright.php");
?>

<p>&copy;<?= $year; ?> <?= $author ?>
```

Include versus Requires

Four ways to load another script

- `include("lib/foo.php");`
  Loads the file, script continues if file is not found.
- `include_once("lib/foo.php");`
  Loads the file if it hasn't been loaded previously, script continues if file is not found.
- `require("lib/foo.php");`
  Loads the file, triggers fatal error if file not found
- `require_once("lib/foo.php");`
  Loads the file if it hasn't been loaded already, triggers fatal error if file not found

Worst Practices

If you learn nothing else today...

- Don’t trust user input, filter input & sanitize output.
- Don’t use global scope in your functions
  - http://en.eikipedia.org/wiki/Global_variable
- Don’t suppress error output with @.
- Don’t use eval statement.
- Don’t use “or die()” to handle errors.
- Don’t use ereg_*, mysql_* functions (or other deprecated functions)

Handling form input

```
<p>Please answer a few questions.</p>
<form action="./survey.php" method="post">
   <input type="text" name="name" placeholder="Your Name" required="required" />
   <input type="email" name="email" placeholder="Your e-mail" required="required" />

   <p><b>Check each animal to visit at the zoo:</b></p>
   <label><input type="checkbox" name="animals[]" value="lions" />Lions</label>
   <label><input type="checkbox" name="animals[]" value="tigers" />Tigers</label>
   <label><input type="checkbox" name="animals[]" value="bears" />Bears</label>
   <label><input type="checkbox" name="animals[]" value="zebras" />Zebras</label>
   <label><input type="checkbox" name="animals[]" value="monkeys" />Monkeys</label>

   <div><input type="submit" value="Save" /></div>
</form>
```

Please answer a few questions.

Your Name Your e-mail

Check each animal to visit at the zoo:

Lions Tigers Bears Zebras Monkeys

Save

Handling form input

```php
<?php
// sanitize incoming name and email
$name  = filter_var($_POST['name'], FILTER_SANITIZE_STRING);
$email = filter_var($_POST['email'], FILTER_SANITIZE_EMAIL);
if (!filter_var($email, FILTER_VALIDATE_EMAIL)) {
    trigger_error('Invalid email address', E_USER_ERROR);
}

// ensure animal options are valid
$valid = ['lions', 'tigers', 'bears', 'zebras', 'monkeys'];

// Filter by looping through the array
$animals = [];
foreach ($_POST['animals'] as $test) {
    if (in_array($test, $valid)) {
        array_push($animals, $test);
    }
}

// OR filter with an array_* function, keep the ones in common
$animals = array_intersect($_POST['animals'], $valid);

// We now have values for $name, $email, and $animals that we can use safely.
// We can email them, store them in a database, and/or echo them back to the user.
```

Making an HTTP request

```php
<?php
// get the weather based on a city name
// name is supplied via query string parameter "city"
$city = null;
if (isset($_GET['city'])) {
  $city = filter_var($_GET['city'], FILTER_SANITIZE_STRING);
}

if (empty($city)) {
  trigger_error('City must be specified', E_USER_ERROR);
}

// build URL to query weather
$params = ['q' => $city ];
$url = 'http://api.openweathermap.org/data/2.5/weather'?' . http_build_query($params);

// make the request and decode the response, assumes allow_url_fopen is enabled
if ($result = file_get_contents($url)) {
  $weather = json_decode($result);
  var_dump($weather);
}
```

Making an HTTP request

```
stdClass Object
(
    [coord] => stdClass Object
        (
            [lon] => -77.031997680664
            [lat] => 38.890399932861
        )

    [sys] => stdClass Object
        (
            [country] => United States of America
            [sunrise] => 1378723498
            [sunset] => 1378769117
        )

    [weather] => Array
        (
            [0] => stdClass Object
                (
                    [id] => 802
                    [main] => Clouds
                    [description] => scattered clouds
                    [icon] => 03d
                )

        )

    [base] => gdps stations
    [main] => stdClass Object
        (
            [temp] => 295.46
            [humidity] => 63
            [pressure] => 1021
            [temp_min] => 293.71
            [temp_max] => 297.15
        )

    [wind] => stdClass Object
        (
            [speed] => 1.54
            [gust] => 1.54
            [deg] => 360
        )

    [clouds] => stdClass Object
        (
            [all] => 32
        )

    [dt] => 1378743982
    [id] => 4140963
    [name] => Washington
    [cod] => 200
)
```

Output a result set

```php
<?php
// $dbh is a connection to our database
$sth = $dbh->prepare("SELECT name, email, room, phone, fax, photo FROM employees");
$sth->execute();

// Return next row as an array indexed by column name
while ($row = $sth->fetch(PDO::FETCH_ASSOC)) {
    include('card.php');
}
```

card.php:

```
<dl class="staff">
    <dt>Name</dt>
        <dd><?= $row['name'] ?></dd>
    <dt>E-mail</dt>
        <dd><?= $row['email'] ?></dd>
    <dt>Room</dt>
        <dd><?= $row['room'] ?></dd>
    <dt>Phone</dt>
        <dd><?= $row['phone'] ?></dd>
    <dt>Fax</dt>
        <dd><?= (!empty($row['fax']) ? $row['fax'] : '-' ?></dd>
    <dt>Photo</dt>
        <img src="/images/staff/<?= $row['photo'] ?>" alt="<?= $row['name'] ?>/>
</dl>
```

Where do you go from here?

Further reading

- http://php.net - beware User Notes
- http://www.phptherightway.com/
- https://phpbestpractices.org/

Community

- http://www.meetup.com/DC-PHP/
- http://phpmentoring.org
- http://phpwomen.org/

php[architect] @phparch

Web Summits - http://summits.phparch.com

- Sept. 17th - What's new in PHP5.5

Online Training - PHP, Wordpress, more...

- Sign-up before Sept 15th to get an iPod Shuffle.

Monthly Electronic Magazine

- 25% off a year subscription: **AZ42-W1JJ-D57Z**

Books too

- http://phparch.com