INFO I-CE9224: Introduction to PHP Programming

Session 5
July 11, 2012

Resources

http://davehauenstein.com/nyu/INFO1-CE9224-2012-Summer

Username: nyuscps
Password: $nyuscps$

Class 5 Agenda

- User Defined Functions
- HTTP - Request/Response Headers & More
- HTTP and PHP
  - Cookies
  - Sessions
  - More...
- Sending Email
- Class 4 Lab Review

User Defined Functions

User Defined Functions

- Self contained block of code to perform a specific task.
- Modular: Once defined, can be called many times, anywhere in a script.
- Avoid duplication of code.
- Easier to eliminate errors.
- Help break down a big project into smaller pieces.

User Defined Functions

- Defining your own functions
- Return values
- Function arguments
- Optional arguments
- Variable Scope
- Creating Anonymous Functions
- Passing arguments by reference
- Recursion

User Defined Functions

```
function functionName ()
{
    // function body here...
    return expression;
}

 function functionName ( $arg1, $optionalArg = '')
{
    // function body here...
    return expression;
}
```

User Defined Functions

```php
<?php

function add($num1, $num2)
{
    if (!is_int($num1) || !is_int($num2)) {
        return false;
    }

    return $num1 + $num2;
}
```

User Defined Functions

```php
echo add(67, 32); // prints (int) 99

$val = add(22, 74);
echo $val; // prints (int) 96

$val = add('hi', 51); // $val is (bool) false
```

User Defined Functions

- A function's return value is an expression.
- A function's return value can be used anywhere any expression can be used.

```php
<?php

function add($num1, $num2, $num3 = 0)
{
    if (!is_int($num1) ||
        !is_int($num2) ||
        !is_int($num3)
    ) {
        return false;
    }

    return $num1 + $num2 + $num3;
}

echo add(67, 32); // prints (int) 99
echo add(67, 32, 1); // prints (int) 100
echo add(67, 32, '1'); // add returns false
```

User Defined Functions

Order of arguments matters...

```php
function printFullName($first, $last)
{
    $first = ucfirst($first);
    $last  = ucfirst($last);
    return $first . ' ' . $last;
}

// prints Bill Cosby
echo printFullName('bill', 'cosby');
```

Echo vs. Return

```php
function getTextInput($name, $val = '')
{
    $template = '<input type="text" name="%s" value="%s" />';
    return sprintf($template, $name, $val);
}

// prints <input type="text" name="name" value="dave" />
echo getTextInput('name', 'dave');

// set it to a variable for later use
$input = getTextInput('name', 'dave');
```

Echo vs. Return

```php
function printTextInput($name, $val = '')
{
    $template = '<input type="text" name="%s" value="%s" />';
    printf($template, $name, $val);
}

// prints <input type="text" name="name" value="dave" />
echo printTextInput('name', 'dave');

// prints <input type="text" name="name" value="dave" />
$input = printTextInput('name', 'dave');

var_dump($input); // Value of $input is NULL
```

Functions and Scope

- Variables can be created within a function for use within in function.
- Variables created within a function are not accessible outside of the function.

Functions and Scope

```php
$greeting = "Hello, old friend.";

function greetFriend()
{
    $greeting = "Good to see you!";
    return $greeting;
}

echo $greeting;
echo greetFriend();
echo $greeting;
```

Functions and Scope

```php
$greeting = "Hello, old friend.";

function greetFriend()
{
    $greeting = "Good to see you!";
    return $greeting;
}

echo $greeting;      // prints: Hello, old friend.
echo greetFriend(); // prints: Good to see you!
echo $greeting;      // still prints: Hello, old friend.
```

Functions and Scope

```php
$greeting = "Hello, old friend.";

function greetFriend()
{
    global $greeting;

    return $greeting;
}

echo $greeting;     // prints: Hello, old friend.
echo greetFriend(); // prints: Hello, old friend.
```

Functions and Scope

```php
$greeting = "Hello, old friend.";

function greetFriend()
{
    global $greeting;
    $greeting = 'Good to see you!';

    return $greeting;
}

echo $greeting;     // prints: Hello, old friend.
echo greetFriend(); // prints: Good to see you!
```

Functions and Scope

```php
define('GREETING', 'Hello, old friend.');

function greetFriend()
{
    $greeting = GREETING;

    return $greeting;
}

echo GREETING;      // prints: Hello, old friend.
echo greetFriend(); // prints: Hello, old friend.
```

Passing Arguments by Reference

- We've learned how to pass arguments by value.
- Function can now alter original value.
- After function call, variable is modified without a return statement or assignment expression.
- Place an ampersand (&) before argument in function declaration.

Passing By Value

```php
function arrayValuesToUpper($values)
{
    foreach($values as $key => $value) {
        $values[$key] = strtoupper($value);
    }

    return $values;
}

$cars = array('Ford', 'Chevy', 'Dodge');

// Scenario 1
arrayValuesToUpper($cars);
print_r($cars);

// Scenario 2
$cars = arrayValuesToUpper($cars);
print_r($cars);
```

Passing By Value

```php
function arrayValuesToUpper($values)
{
    foreach($values as $key => $value) {
        $values[$key] = strtoupper($value);
    }

    return $values;
}

$cars = array('Ford', 'Chevy', 'Dodge');

// Scenario 1
arrayValuesToUpper($cars);
print_r($cars);

// Scenario 2
$cars = arrayValuesToUpper($cars);
print_r($cars);
```

Important

Passing By Value

```
Array
(
    [0] => Ford
    [1] => Chevy
    [2] => Dodge
)
Array
(
    [0] => FORD
    [1] => CHEVY
    [2] => DODGE
)
```

Passing By Value

- Scenario 1 - calling arrayValuesToUpper does not modify $cars. $cars holds the same value as it was originally assigned.
- Scenario 2 - assigning the result of the function call, arrayValuesToUpper, to $cars finally modifies the value of $cars.

Passing By Reference

```php
function arrayValuesToUpper(&$values)
{
    foreach($values as $key => $value) {
        $values[$key] = strtoupper($value);
    }
}

$cars = array('Ford', 'Chevy', 'Dodge');

// Scenario 1
arrayValuesToUpper($cars);
print_r($cars);

// Scenario 2
$cars = arrayValuesToUpper($cars);
print_r($cars);
```

Passing By Reference

```php
function arrayValuesToUpper(&$values)
{
    foreach($values as $key => $value) {
        $values[$key] = strtoupper($value);
    }
}
```

```php
$cars = array('Ford', 'Chevy', 'Dodge');

// Scenario 1
arrayValuesToUpper($cars);
print_r($cars);

// Scenario 2
$cars = arrayValuesToUpper($cars);
print_r($cars);
```

Important

Passing By Reference

```
Array
(
    [0] => FORD
    [1] => CHEVY
    [2] => DODGE
)


NULL
```

Passing By Reference

- Scenario 1 - calling arrayValuesToUpper DOES modify $cars. $cars holds the UPDATED value. It was modified by the function.

- Scenario 2 - assigning the result of the function call, arrayValuesToUpper, to $cars sets $cars to NULL. Why?

- Because... If a function does not have a return statement, by default it returns NULL. That is why the second $cars prints out NULL.

Passing By Reference

PHP built-in array functions,

sort() and rsort(), use pass by reference.

```php
$cars = array('Ford', 'Chevy', 'Dodge');
sort($cars);

print_r($cars);
```

```
Array
(
    [0] => Chevy
    [1] => Dodge
    [2] => Ford
)
```

Passing By Reference

- Functions receives reference to value rather than a copy of the value.
- Function can modify value and will be seen by it's caller.
- Provides additional channel of communication between function and caller.
- Makes it more difficult to track effects of function call and harder to track down bugs.

But...WHY?

Passing By Reference

- Since the value is not copied, the memory footprint is lower. Function *doesn't have to* modify contents.
- Returning multiple values.

```php
function isAgeValid($age, &$messages)
{
    if(is_numeric($age) && $age > 0 && $age < 122) {
        return true;
    }
    $messages[] = 'Age is Invalid';
    return false;
}

function isNameValid($name, &$messages)
{
    $match = preg_match('/^[a-z]{3,20}$/i', $name);
    if(is_numeric($match) && 0 < $match) {
        return true;
    }
    $messages[] = 'Name is Invalid';
    return false;
}

$messages  = array();
$ageValid  = isAgeValid(231, $messages);
$nameValid = isNameValid('a1b2', $messages);

var_dump($ageValid);
var_dump($nameValid);
print_r($messages);
```

```
bool(false)

bool(false)

Array
(
    [0] => Age is Invalid
    [1] => Name is Invalid
)
```

Anonymous Functions

- New to PHP 5.3.
- Functions with no name.
- Can be assigned to variables.
- Customized code w/in a function at the time it's created = flexibility.
- Short-term disposable functions: usually if another function takes a "callback".

Callbacks

- A callback is an answer to a question.
- It's used by a function doing something generic, to perform a more specific and customized task.

Callback Example

array_map (*callable* $callback , *array* $arr1)

- Looping over array is done by array_map.
- Looping over array is the generic behavior.
- *callable* allows you to add customization.

Callback Example

array_map (*callable* $callback , *array* $arr1)

- For each element in the array, do something.
- What to do is defined by the callable, aka anonymous, function.
- Example: We want to convert integers to the string value in the array, 1 should become one, etc... Define a function to do it.

Callback Example

```php
$conversion = function($integer) {
    $map = array(
        1 => 'one',
        2 => 'two',
        3 => 'three',
        4 => 'four',
        5 => 'five',
        // etc...
    );
    return (isset($map[$integer]))
        ? $map[$integer]
        : false;
};

$integers = array(2, 3, 4);
$strings  = array_map($conversion, $integers);

print_r($integers);
print_r($strings);
```

```php
$conversion = function($integer) {
    $map = array(
        2 => 'two',
        3 => 'three',
        4 => 'four',
        // etc...
    );
    return (isset($map[$integer]))
        ? $map[$integer]
        : false;
};

function customArrayMap(Closure $function, array $array)
{
    $updatedArray = array();
    foreach($array as $key => $value) {
        $updatedArray[$key] = $function($value);
    }
    return $updatedArray;
}

$integers = array(2, 3, 4);
$strings  = array_map($conversion, $integers);

print_r($integers);
print_r($strings);
```

HTTP

HTTP Request

- JUST STRINGS!
- Parts of the Request:
  - Request line
  - List of HTTP Headers
  - An empty line
  - An optional body

HTTP Request

```
GET / HTTP/1.1
Accept:text/html,application/xhtml+xml,*/*;q=0.8
Accept-Charset:ISO-8859-1,utf-8;q=0.7,*;q=0.3
Accept-Encoding:gzip,deflate,sdch
Accept-Language:en-US,en;q=0.8
Cache-Control:max-age=0
Connection:keep-alive
Cookie:__utma=132259099.7...
Host:davehauenstein.com
User-Agent:Mozilla/5.0 (Macintosh; Intel Mac...
```

HTTP Response

```
HTTP/1.1 200 OK
Date: Thu, 12 Apr 2012 16:40:21 GMT
Server: Apache/2.2.16 (Ubuntu)
Vary: Accept-Encoding
Content-Length: 5407
Content-Type: text/html

<?xml version="1.0" encoding="utf-8" ?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 ...
<html xmlns="http://www.w3.org/1999/xhtml" xm...
    <head>
        <title>Dave Hauenstein | Programmer</...
        <link href="favicon.png" rel="shortcu...
        <style type="text/css" media="screen"...
            *{
                margin: 0;
                padding: 0;
            }
```

Some Response Codes

- 200 OK
- 201 Created
- 301 Moved Permanently
- 400 Bad Request
- 404 Not Found
- 500 Internal Server Error

HTTP Request Methods

- GET + *
- POST
- PUT *
- DELETE *
- HEAD + *
- OPTIONS
- TRACE
- CONNECT
- PATCH

+ Safe Methods
* Idempotent Methods

Modifying the Response with PHP

- Returning a status code other than 200
- Redirecting a user: Location header
- Forcing a download in the browser
- Setting Cookies

Setting Response Headers with PHP

This file is called slides.php

```php
<?php

header('X-Example-Header: Hello Class!');

echo "I set a header";
```

Response

```
GET /slides.php HTTP/1.1
User-Agent: curl/7.21.4 (universal-apple-da...
Host: localhost:8888
Accept: */*

HTTP/1.1 200 OK
Date: Thu, 12 Apr 2012 16:44:17 GMT
Server: Apache/2.2.21 (Unix) mod_ssl/2.2.21...
X-Powered-By: PHP/5.3.6
X-Example-Header: Hello Class!
Content-Length: 14
Content-Type: text/html

I set a header
```

Different Status Codes

```php
<?php

$action  = ($_GET && isset($_GET['action'])) ? $_GET['action'] : false;
$content = '';

switch ($action) {
    case 'home':
        $content = 'Welcome Home!';
        break;
    case 'bio':
        $content = 'I am Dave, a developer.';
        break;
    case 'info':
        header('HTTP/1.1 301 Moved Permanently');
        header('Location: ' . $_SERVER['PHP_SELF'] . '?action=bio');
        break;
    default:
        header('HTTP/1.1 404 Not Found');
        $content = '404 Not Found';
}

echo $content;
```

Forcing a Download

```
<?php

header('Pragma: public');
header('Expires: 0');
header('Cache-Control: must-revalidate, post-check=0, pre-check=0');
header('Cache-Control: private');
header('Content-Transfer-Encoding: binary');
header('Content-Type: text/x-vcard');
header('Content-Disposition: attachment; filename="hauenstein_dave.vcf";');

?>
BEGIN:VCARD
NAME:Dave Hauenstein - Programmer\, New York City
VERSION:3.0
N;LANGUAGE=en;CHARSET=UTF-8:Hauenstein;Dave;;;
FN;LANGUAGE=en;CHARSET=UTF-8:Dave Hauenstein
TITLE;LANGUAGE=en;CHARSET=UTF-8:Software Engineer, Programmer
URL:http://davehauenstein.com/
URL:http://www.businessinsider.com/
URL:http://www.flickr.com/photos/davehauenstein
URL:http://twitter.com/davehauenstein
EMAIL:davehauenstein@gmail.com
ADR;LANGUAGE=en;CHARSET=UTF-8:;;45 Wall St.;New York;NY;10005;United States
TEL;TYPE=cell:732 754 3870
END:VCARD
```

HTTP: A Stateless Protocol

- What does this mean?
- What if we want state?

Giving HTTP State

- Cookies
- Sessions

Setting/Getting Cookies

```php
<?php

setcookie( 'name', 'Mark' );
```

```php
<?php

if($_COOKIE['name']) {
    echo $_COOKIE['name']; // prints: Mark
}
```

Setting/Getting Cookies

```
setcookie(
    'recently_viewed',          // Name of the cookie.
    'New Balance Sneakers',     // Value of the cookie.
    time()+60*60*24*30,         // Time in Unix seconds
                                // until the cookie expires.
    '/',                        // Path in which cookie is
                                // valid.
    'www.example.com',          // Domain in which cookie is
                                // valid.
);

/*
If a script on the subdomain blog.example.com tried
to access the recently_veiwed cookie, it would not be
set. If we changed www.example.com to example.com then
it would be available.
*/
```

Deleting Cookies

Set the expiry time to the past.

```php
setcookie(
    'recently_viewed',
    'New Balance Sneakers',
    time() - 3600
);

// still prints: 'New Balance Sneakers'
echo $_COOKIE['recently_viewed'];

// on subsequent requests
// $_COOKIE['recently_viewed']
// will no longer be available.
```

```php
$action  = ($_GET && isset($_GET['action'])) ? $_GET['action'] : false;
$content = '';

switch ($action) {
    case 'home':
        $content = 'Welcome Home!';
        break;
    case 'bio':
        $content = 'I am Dave, a developer.';
        break;
    case 'info':
        header('HTTP/1.1 301 Moved Permanently');
        header('Location: ' . $_SERVER['PHP_SELF'] . '?action=bio');
        break;
    default:
        header('HTTP/1.1 404 Not Found');
        $content = '404 Not Found';
}

if (isset($_GET['name'])) {
    $name = $_GET['name'];
    setcookie('name', $name);
} else if (isset($_COOKIE['name'])) {
    $name = $_COOKIE['name'];
} else {
    $name = false;
}
?>
<h1><?php echo $content; ?></h1>
<?php if($name) { ?>
<p>Hello, <?php echo $name; ?></p>
<?php } ?>
```

Setting/Getting Session Data

```php
session_start();

$_SESSION['views'] = (isset($_SESSION['views']))
    ? $_SESSION['views'] + 1
    : 1;

echo $_SESSION['views'];
```

Removing Session Data

```php
// Destroying a session,
// for example, on log-out

// Session first has to be started.
session_start();

// Once started, calling this function
// erases all information stored on disc.
session_destroy();

// The data in $_SESSION will still be
// available for the duration of the
// request. To ensure it's gone, reinitialize
// the $_SESSION variable.
$_SESSION = array();
```

Header, Cookie, Session Common Error

Remember that **header()** must be called before any actual output is sent, either by normal HTML tags, blank lines in a file, or from PHP.

Warning: session_start() [function.session-start]: Cannot send session cache limiter - headers already sent (output started at /Applications/MAMP/htdocs/slides.php:1) in **/Applications/MAMP/htdocs/slides.php** on line **3**

Sending Email

Sending Email: *mail()*

- Uses built in Mail Transfer Agent (MTA); for example *sendmail*
- Sometimes uses built in SMTP server (on Windows)
- Sendmail is a program that uses SMTP to send email
- SMTP is a protocol for sending email
- SMTP is to HTTP as Sendmail is to Apache

mail() - EASY!

```php
$to      = 'dah16@nyu.edu';
$subject = 'Hi, Dave!';
$message = 'Just saying hi!';
$message = wordwrap($message, 70);

$result = mail ( $to, $subject, $message );

var_dump($result); // will be boolean true or false
```

From: "Jetsetter" <reminder@mail.jetsetter.com>
To: <Davehauenstein@gmail.com>
Subject: The Weekender: Summer Camp for Grown U...
Date: Thu, 12 Apr 2012 13:05:49 -0600
List-Unsubscribe: <mailto:leave-fdb8171f2d20592...
MIME-Version: 1.0
Reply-To: "Jetsetter" <reply-ff2d12707c67-3_HTM...
x-job: 1032069_6093
Message-ID: <5ba0cd34-ec13-4ad4-8cec-b5e82d8a4e...
Content-Type: multipart/alternative;

mail() - Setting Headers

```php
$to      = 'dah16@nyu.edu';
$subject = 'Hi, Dave!';
$message = 'Just saying hi!';
$message = wordwrap($message, 70);
$headers = array(
    'From: Bill Cosby <me@billcosby.com>',
    'X-Priority: 1 (Highest)',
);
$headers = implode("\r\n", $headers);

$result = mail ( $to, $subject, $message, $headers );

var_dump($result); // will be boolean true or false
```

mail() - Sending HTML Email

```php
$to      = 'dah16@nyu.edu';
$subject = 'Hi, Dave!';
$message = '<h1>Just saying hi!</h1>';
$headers = array(
    'From: Bill Cosby <me@billcosby.com>',
    'X-Priority: 1 (Highest)',
    'MIME-Version: 1.0',
    'Content-Type: text/html; charset=utf-8'
);
$headers = implode("\r\n", $headers);

mail ( $to, $subject, $message, $headers );
```

Lab Assignment!