ITMD 462/562
Web Site Application Development

Lecture 7

Fall 2015 – October 7, 2015

Tonight's Agenda

- Exceptions and Handling
- Output Buffering
- Headers and Redirection
- Passwords, Hashing, & Authentication

Exceptions

Exception Handling

- Exceptions are basically PHP errors
- Exceptions are objects that are thrown when the error happens
- Using try / catch blocks allow you to test for an exception and if one happens react without an error in your program
- Built-in and custom Exceptions available

Try

```
try {
    //run your code
} catch (Exception $e) {
    //handle any exceptions
    echo 'Caught exception: ',  $e->getMessage(), "\n";
} finally {
    //always run after try/catch
    echo "First finally.\n";
}
```

- http://php.net/manual/en/language.exceptions.php
- http://code.tutsplus.com/tutorials/php-exceptions--net-22274

Output Buffering

Output buffering

- Normally as soon as you send any output to the response stream, HTML, echo, print statements, etc., it is directly sent to the stream as it is ready
- All headers must be sent and some other functions run before any output is sent to the browser.
- For example a header must be set before anything is sent to the browser including a single space.
- Output buffering allows you to buffer your output and then flush it to the response stream later.

Output Buffering

- Three functions to use basic output buffering.

ob_start ()

ob_end_flush ()

ob_end_clean ()

- Start begins the buffer, Flush outputs the buffer to the page, Clean deletes the buffer content without putting it to the page

ob_get_contents ( )

- Gets the buffer contents as a string
- Buffers can be stacked

Output Buffering

- http://php.net/manual/en/ref.outcontrol.php
- http://php.net/manual/en/function.ob-start.php
- http://php.net/manual/en/function.ob-end-flush.php
- http://php.net/manual/en/function.ob-end-clean.php
- http://www.hackingwithphp.com/13/0/0/output-buffering
- http://www.devshed.com/c/a/PHP/Output-Buffering-With-PHP/

Response Headers

Headers

- Sometimes you need to set the HTTP response headers yourself
- PHP has a function to send a header to the browser
- **`header()`** is used to send a raw HTTP header.
- Must be called before any actual output is sent, either by normal HTML tags, blank lines in a file, or from PHP, including require or include files.
- Parameter is a string header
- Location header is used for redirection to another page.
- `header("Location: http://$host$uri/$extra");`
- http://php.net/manual/en/function.header.php

Authentication

HTTP Basic Authentication

- Sends an *"Authentication Required"* message to the browser which pops up a user and password prompt.
- Values in

```
$_SERVER['PHP_AUTH_USER']
```

```
$_SERVER['PHP_AUTH_PW']
```

- Handles HTTP Basic and Digest Authentication
- http://php.net/manual/en/features.http-auth.php

Password Hashing

- Never store plain text passwords in your database
- Use hashing and salt to store a one-way encrypted password.
- PHP 5.5 has dedicated password hashing functions – use them
- PHP < 5.5 you need to use the crypt functions
- When the user supplies their password from a web form hash it again and then compare the hashes.
- To keep someone persistently logged in store something like their login name in a session variable.

Password Hashing

- http://php.net/manual/en/faq.passwords.php
- http://php.net/manual/en/book.password.php
- http://php.net/manual/en/function.crypt.php
- http://www.formget.com/login-form-in-php/

- Let's look at some of these examples

Assignments

Assignments

- Assignment 2 is up.