ITMD 462/562

Web Site Application Development

Lecture 1

Fall 2015 – August 26, 2015

Course Introduction

Web Site Application Development

ITMD 462/562

Contact

- Professor: Brian Bailey
- Telephone: 312.567.6937
- Email: bbailey4@iit.edu (Email Preferred)
- Office Hours: Before Class and By Appointment

Lecture Time

- Days: Wednesdays
- Time: 6:25pm to 9:05pm
- Where: Stuart Building, Room 111, IIT Main Campus

Course Description

Programming the Common Gateway Interface (CGI) for Web pages is introduced with emphasis on creation of interfaces to handle HTML form data. CGI programming is taught in multiple languages. Security of Web sites is covered with an emphasis on controlled access sites. Setup, administration and customization of content management systems including blog and portal sites is introduced. Students design and create a Web site including basic CGI programs with Web interfaces and process data flows from online forms with basic database structures. Prerequisites: [(ITMD 461)] Credit: (2-2-3) (C)

Course Format

- Lecture Based
- Lectures consist of discussions on the concepts and practical examples
- Slides should not be considered full notes!
  - Take notes during class
  - Don’t tune out and browse the web, you may miss an important concept that is needed for an assignment
- Demos will be shown in class for most topics we discuss.
  - These examples will demonstrate the basics of the topics we discuss. Expect assignments to ask you to go further than the demos in class and require some research on your time.

Assignment Types

- Two main types
  - smaller lab type assignments expecting to take a small amount of time.
  - Larger multi-week project based assignments
- You will be expected to apply class concepts to solve a simulated problem
- I will deliver a narrative of the problem and a description of what I expect as a deliverable. How you solve the problem has some flexibility.
- Sometimes I will supply some skeleton code or specific libraries to use.
- Graduate students will have extended requirements for each assignment.
- Expect assignment to ask you to do things that may not have completely been covered in class. You will need to take the concepts we discuss and do some additional research to complete the solution.
- Follow the assignment specification document for submission expectations. You will be graded on following those guidelines.
- Assignments are individual (not group) work.

Knowledge Expectations

- This is a programming course. I expect everyone in this course has been exposed to programming at some level, ideally an object-oriented language.
- I also expect that everyone in the course has at least been exposed to basic database concepts.
- We will discuss object-oriented programming concepts and use basic SQL to work with databases but this is not a full course in either. We will cover the basics but not go into deep detail. You may need to do extra outside research if these topics are not familiar to you.
- You are expected to have a solid knowledge of HTML and CSS since it is a prerequisite to the course.

Course Policies

Reading

- There is no required text book, that doesn't mean there is no required reading.
- Our main resource will be the online php.net official documentation. This will serve as our primary language API guide and reference.
- I will also assign various web resources I find that may be applicable throughout the course.
- I may suggest optional books that would be worthwhile for you to read as additional information and reference.
- If you find interesting articles relative to our class please share.

Technology Requirement

- This is a programming class. You will need access to a computer and you will need to be able to install software.
  - I have been told the computers in some of the OTS labs will have the software you need. I will check the software list when the new one is published.
- You should be familiar with installing development tools and setting up your development environment. As a developer you would be expected to know how to configure your development environment.
- Any computer operating system will do, Windows, Mac, or Linux. All the work in this class can be done with free software.
- We will discuss software needed and environment setup later.

Attendance

- If you are registered for the live class you are expected to attend
- Lectures will be recorded and available online. If you miss a class you are expected to watch the lecture online through IIT Online or my screen recording.
- I will be giving occasional quizzes as a factor of attendance and participation
- Online students will get a different timed online quiz, live students are expected to take the quiz in class
  - Sometimes an online quiz may be assigned to both online students and live students.
- There will be discussion forums in blackboard for major class topics and each assignment. Please discuss issues with the class there.
- Quizzes and discussion forum postings are factored into the participation component of the grade.
- See syllabus for detailed policy on these topics.

Academic Honesty

- Plagiarism and Unauthorized Collaboration
- Do Not Do It. You will receive a Zero for the assignment and be reported for an Academic Honesty violation.
- I am on a lookout for it, it tends to happen every semester to some degree.
- There will be NO EXCEPTIONS to this policy.
- See syllabus for detailed policy
- Plagiarism
  - Copying Code or Words without Attribution
- Unauthorized Collaboration
  - Sharing Code with Classmates
- Your assignments need to be your own code. 80% of your code should be your own. No more than 20% attributed code for the graded component of a project. (primary application libraries not included)

Grading

- Grading policy is detailed in the syllabus
- The final grade for the class will be calculated as follows:

Assignments/Labs	40%
Final Project	20%
Mid-term Exam	10%
Final Exam	20%
Class Participation	10%

- Late Policy
  - Detailed in the syllabus grading section

Syllabus

- The syllabus is my contract with you as to what I will deliver and what I expect from you. If I change the syllabus, I will issue a revised version of the syllabus; the latest version will always be available on Blackboard. Revisions to readings and assignments will be communicated via Blackboard.
- By staying registered in this class you agree to the policies outlined in the syllabus.
- If you ask me a question that is outlined in the syllabus I may answer "It is in the syllabus".
- I will be very firm enforcing the policies in the syllabus.
- Let's review the syllabus.

Web Technology Review

HTML

- Hypertext Markup Language
- Proposed by Tim Berners-Lee in 1989 and defined in 1990. HTML, HTTP, URI
- XML like structure to describe the contents of a page in a tree of elements. Originally based on SGML (standard generalized markup language) used within CERN.
- CSS specification added in 1996 to allow for styling and separation of content from presentation in HTML pages.
- We currently should be using the HTML5 specification for all our HTML we do in class.
- *(Demo – Basic HTML Page)*

URI/URL

- Uniform Resource Identifier
- Uniform Resource Locator
- URI is a unique string that identifies a location of a resource.
- URL typically added the method of locating the resource by adding the protocol component.
- https://danielmiessler.com/study/url_vs_uri/
- http://www.w3.org/TR/uri-clarification/
- https://en.wikipedia.org/wiki/Uniform_resource_locator
- https://en.wikipedia.org/wiki/Uniform_resource_identifier

HTTP

- Hypertext Transport Protocol
- Protocol for reliable transfer of data from client to server using TCP
- HTTP is the protocol that is used by the browser to send requests and for the server to deliver responses when a web resource is asked for.
- Request – Response model
- HTTPS adds TLS (transport layer security) or SSL (secure sockets layer) encryption to the request/response transfer
- HTTPS created at Netscape in 1994

HTTP

- HTTP v0.9 – 1991
  - First documented version
- HTTP/1.0 – 1996
  - Officially introduced and recgonized
  - http://www.w3.org/Protocols/HTTP/1.0/spec.html
- HTTP/1.1 - 1999
  - http://www.w3.org/Protocols/rfc2616/rfc2616.html
- 1.0 vs 1.1
  - 1.0 only had GET, POST, HEAD Methods
  - 1.1 requires host header
  - 1.1 adds some cacheing and persistence and more
- http://www2.research.att.com/~bala/papers/h0vh1.html
- HTTP/2 – May 2015 published
  - https://en.wikipedia.org/wiki/HTTP/2

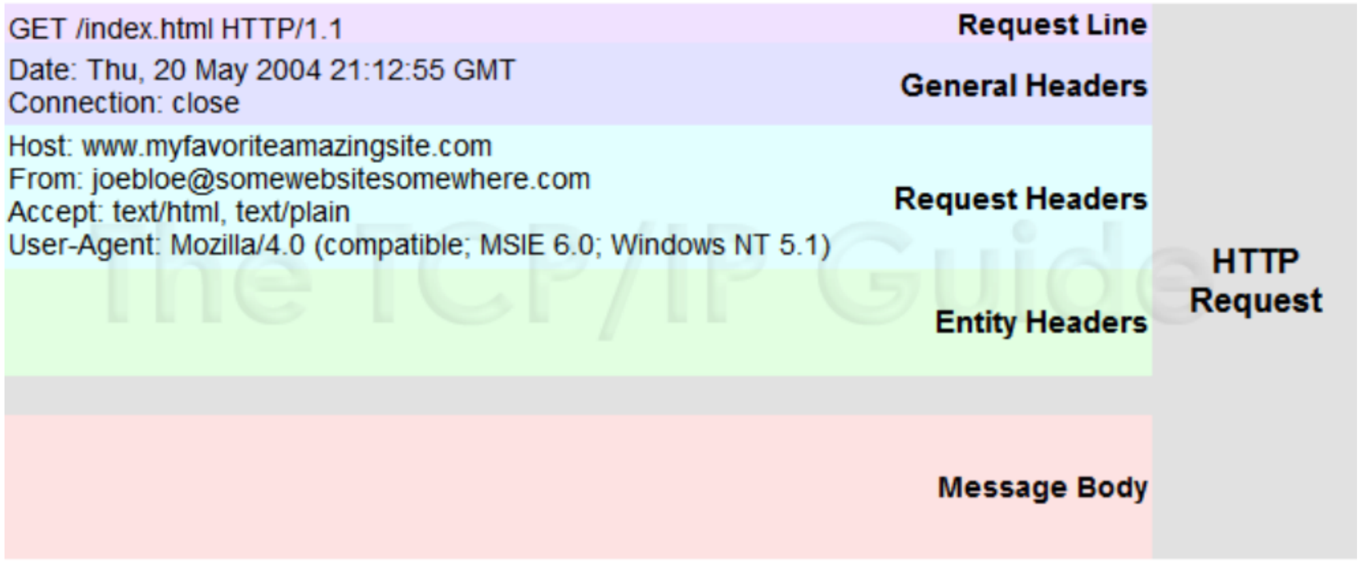
HTTP Request

- Client Parses the URI
  - protocol://server/request
- Client sends request to Server
  - Usually HTTP protocol

```
[METH] [REQUEST-URI] HTTP/[VER]
[fieldname1]: [field-value]
...
[request body, if any (used for POST and PUT)]
```

- Example - GET /index.html HTTP/1.1

HTTP Request



```
GET /index.html HTTP/1.1
Date: Thu, 20 May 2004 21:12:55 GMT
Connection: close
Host: www.myfavoriteamazingsite.com
From: joebloe@somewebsitesomewhere.com
Accept: text/html, text/plain
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1)
```

http://www.tcpipguide.com/free/t_HTTPRequestMessageFormat.htm

HTTP Methods

- HTTP Methods
  - GET, POST, PUT, DELETE, HEAD, TRACE, CONNECT, OPTIONS
  - First 4 are the common ones. Mostly GET.
  - http://www.w3.org/Protocols/rfc2616/rfc2616-sec9.html
- GET
  - Most common, Basically get me this document
  - Any variable or form data is sent as part of the URL
  - http://www.domain.com/?q=232&name=joe
  - Data q=232 and name=joe is available to target page

HTTP Methods

- POST
  - Second most common method
  - Used often to send form data
  - Any variable or form data is sent in the request body and not appended to the URL
- PUT & DELETE
  - Used mostly with web programming frameworks
  - Used in Ruby on Rails
- HEAD
  - Returns only the Response headers from server
- OPTIONS
  - Used to ask the server for information about the communication options available.

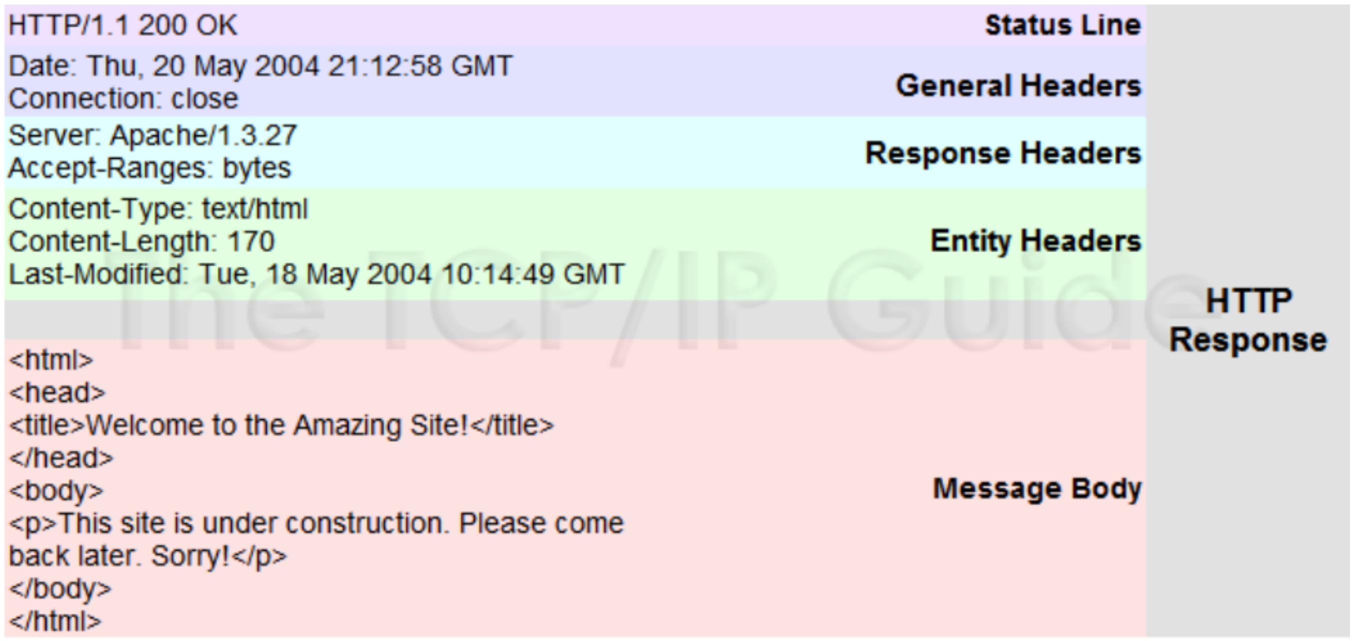
HTTP Response

- Server sends response to client
  - Usually HTTP Protocol

```
HTTP/[ver] code text
[fieldname1]: [field-value]
...
[response body]
```

- First line is status of request
- Then multiple header fields can follow
- Lastly the response body follows

HTTP Response



```
HTTP/1.1 200 OK
Date: Thu, 20 May 2004 21:12:58 GMT
Connection: close
Server: Apache/1.3.27
Accept-Ranges: bytes
Content-Type: text/html
Content-Length: 170
Last-Modified: Tue, 18 May 2004 10:14:49 GMT

<html>
<head>
<title>Welcome to the Amazing Site!</title>
</head>
<body>
<p>This site is under construction. Please come
back later. Sorry!</p>
</body>
</html>
```

http://www.tcpipguide.com/free/t_HTTPResponseMessageFormat.htm

HTTP Response

- First line includes status code. You should know the common codes. Some will be important to our app development.
  - http://www.w3.org/Protocols/rfc2616/rfc2616-sec6.html
- Many headers can be set in both the response and request. We will be setting some of these headers in our web applications.

Sessions

- A series of requests from one user
- The first request that is not recognized by the server starts a new session and the server considers it a new person.
- Subsequent requests from that person are kept together in the server
- Variables can be stored on the server in the users session
- The session is forcibly cleared by the server

Dynamic Web Pages and Server Based Web Applications

Moving beyond static HTML

- Why don't we just write HTML for all our pages?

Technologies - CGI

- In 1993 at the National Center for Supercomputing Applications (NCSA) a team wrote the first specification for a way of running command line executables in the context of a web request.
- Many other developers used the specification and it became a virtual standard called CGI (common gateway interface)
- A working group was formed in 1997, led by Ken Coar, to get the NCSA CGI definition formally defined and standardized.
- CGI was formally specified as version 1.1 in RFC 3875
  - https://tools.ietf.org/html/rfc3875
- https://en.wikipedia.org/wiki/Common_Gateway_Interface

Technologies - PHP

- Originally written by Rasmus Lerdorf as a set of CGI scripts in 1994.
- He named the set of scripts "Personal Home Page Tools" often shortened to PHP Tools.
- PHP 3 was a rewrite of the text processor and is the first version that looks similar to today's PHP.
- PHP 3 also got a new name. It was now a recursive acronym "PHP: Hypertext Preprocessor"
- Read the history here
  - http://php.net/manual/en/history.php.php
  - https://en.wikipedia.org/wiki/PHP
- Current released version: 5.6
  - 7.0 is in RC and 6.0 was skipped

PHP in a webpage

- PHP can be executed various ways these days. The most common way is with an Apache module which executes the code directly in the Apache web server.
- PHP files should be plain text files saved with a .php extension
- The extension triggers its execution in the webserver as php.
- At its most basic it is HTML content with PHP tags added where you want to run PHP code.
- PHP Tags in HTML
  - `<?php` `//php goes here` `?>`
- PHP must run through a configured web server. You can not open a file on your computer directly in your web browser.
- *(Demo – Adding PHP to our HTML Page)*

Development Environment

- Programming Based Text Editor
  - Notepad++
  - Sublime Text
  - Atom
- https://en.wikipedia.org/wiki/Comparison_of_text_editors
- We will look into PHP IDEs later.
  - Netbeans
  - PHPStorm
  - And others

- Local LAMP Server Package
  - XAMPP
  - MAMP
  - WAMP
- PHP Built in Server
  - Little more involved to install and configure on Windows
  - Built in on the Mac
  - Probably doesn’t have all the needed drivers for databases and such out of the box in all environments.

Assignments

Reading

- Read the wikipedia page on CGI and the two links in the PHP section.
- Go to http://php.net/manual/en/
  - Read the getting started section
  - Start reading the language reference to get ahead for next week

Assignment

- No formal assignment
- Try to get your local development environment installed.
  - Programming based text editor
  - Local LAMP stack application
- See if you can get a basic PHP page to be served by the server

ITMD 462/562
Web Site Application Development

Lecture 2

Fall 2015 – September 2, 2015

Tonight’s Agenda

- Servers and Responses
- Setting up your local development environment
- Introduction to the PHP Language

Servers and the Request-Response Model

Servers

- Why do we need a server?
- Server hardware/OS is a fixed location resource that provides an operating environment for applications to run which listen for network communications and respond.
- A Server application/container is a specific software application that is running on the server OS that listens and responds to specific types of requests.
- Servers run program code logic to form the response

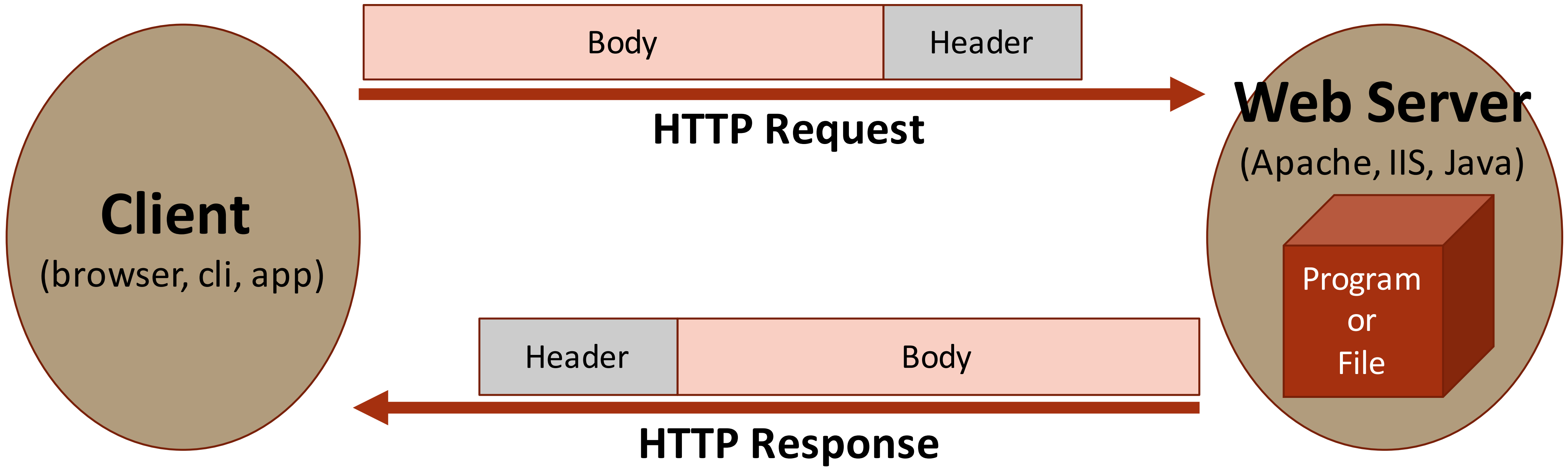
Servers/Applications

- There are several types of servers/containers
- HTTP/Web Server Container
  - Apache
  - IIS
  - nginx
- Java Application Server Container
  - Tomcat
  - Jboss
  - Glassfish
- Many others like FTP, SSH, GAMES

Web Servers

- Web servers listen for HTTP or HTTPS protocol requests on a specific port.
- What is a port?
- HTTP default port is 80
- HTTPS default port is 443
- Web servers job is to listen on the port for the HTTP request and route the request to the folder or file on the file system or another type of resource.

HTTP Request-Response Model



Server Logic

- If the requested resource points to a normal file the web server will then respond with that file and MIME type
- If the requested resource points to a program then the web server routes the request to the program for processing then responds to the client with the results of the program output.
- We will be writing our logic in PHP
- What kinds of things can we write using that logic?
  - Blogs, Social Networks, Forums, Cloud Services, Business Portals, Shopping Carts, Games, and more

Client Logic

- What is a client?
  - Typically a web browser but can be anything that sends an HTTP request. Browsers, apps, command line tools
- What does our clients need to do?
  - Using a URL compose an HTTP GET request
  - Compose a HTTP POST request using form data
- Web Services and REST APIs
  - Using a resource URI send a PUT, DELETE, POST, GET request get data response

Server Programs and Code

- Web server (Apache) receives the request and if it is for a program based resource sends it to the program engine for processing.
- Our program creates the response content and sends it back to the web server and then the web server sends the response to the client.

Receive Request
Decipher Request
Web Server
Access Resources / Run Logic
Our Program
Create Response
Send Response

Local Server

Web Server and PHP Interpreter is required

- The browser can not interpret PHP
- We need a web server to receive the request and hand off the request to our PHP interpreter and then get the response back from the PHP engine and send it to the client
- We need to run a local PHP enabled web server for development

LAMP, WAMP, MAMP, XAMPP

- Typical development environment is a LAMP stack
  - Linux, Apache, MySQL, PHP
- Prepackaged LAMP stacks for all major operating systems
  - WAMP – Windows
  - MAMP – Mac, Windows
  - XAMPP – Windows, Mac, Linux
- [OS] Apache MySQL PHP

PHP Built in Server

- Recent versions (>5.4) of PHP have a built in server.
- Not always preconfigured for your operating system especially database drivers

```
php -S localhost:8080
```

Introduction to PHP

PHP is a programming language

- PHP is a full programming language, not a framework or application
- PHP is an interpreted language at runtime (no complier)
- Syntax similar to C based languages
- Has all your standard programming language data types and control structures
- PHP is very popular and easy to learn and use in the context of web programming
- Can also be use outside of web programming like command line apps but it is much less common.

PHP Tags

- The file should be a plain text file and needs to end in a .php extension to tell the web server that it is PHP and needs to be processed by the PHP engine.
- Any PHP code needs to be inside the PHP tags
- The PHP tags tell the interpreter what code needs to be evaluated.

```
<?php          ?>
```

PHP Info Function

- This is a built in function that can check that PHP is working in the server correctly.
- Displays a web page showing all the PHP configuration

```
<?php
   phpinfo();
?>
```

PHP Variables

- Variables are a way to store and access data in memory in your application.
- All variables in PHP are denoted by a leading dollar sign ($)
- Variables are assigned a value with the = operator. Variable on left, expression on the right.
- Do not need to be declared first.
- Do not have intrinsic types.
- Value is equal to its most recent assignment

PHP Variables

- Cannot start with a number. Need to start with A-Za-z_ then number can follow
- No Spaces allowed in name. Use camelCase or underscores

```
<?php
$myVariable  = 7;
~~myBadVariable  = 8;~~
~~$34myVariable  = 123;~~
$_4myNewVariable  = 'This is fine';
$_4myNewVariable  = 7;
$another_variable_name  = array(5,6,7);
?>
```

PHP Constants

- An identifier for a simple value.
- Can not change during script execution
- Does not start with dollar sign ($). By convention should be all upper case with underscores for spaces.
- Typical variable naming conventions apply
- One useful built-in constant is PHP_EOL

```
Define("FOO", "somevalue");
```

PHP Comments

- Use comments to annotate your work for others and yourself but try not to overuse when the code is clear.

```
// Single line comment


/*
Multi line comment
*/
```

PHP Data Types

- Here is the listing of PHP data types from the php.net manual

http://php.net/manual/en/language.types.php

PHP Numbers

```
$myVar = 8;
$myFloatingPointNum = 3.1415;
$myScientificNum = 4E23;
$myVar++;
$myVar--;
```

PHP Boolean

- Used to hold a true/false value
- No Quotes! Case-insensitive

```
$myVal = true;
$myVal = false;
```

See PHP docs for examples of what coverts to true/false

PHP Strings

- A series of characters.
- Single Quoted vs Double Quoted
- Special characters or same quotes need to be escaped with \

```
$myVal = 'hello’;
$myVal = "hello”;
$myVal = "hello $myName”;
$myVal = 'hello $myName’;
$myVal = "hello \"World\" $myName";
```

String Manipulation

- PHP has many string manipulation functions built-in
- http://php.net/manual/en/ref.strings.php
- The web has a lot of text and HTML is all text based so it is essential to be able to manipulate strings.

```
print, echo
printf
sprintf
htmlspecialchars
rtrim, ltrim
nl2br
strtolower, strtoupper
substr
```

PHP Arrays

- An array is an ordered list or map
- Simple arrays are indexed by a number
- Associative arrays are similar to a hashmap and indexed by a string
- PHP arrays can contain both types of keys

http://php.net/manual/en/language.types.array.php

Indexed Array

- Using numbers as keys

```
$var = array('a', 'b', 'c');
echo $var[0]     // prints out a
$var = array(
0 => 'a',
1 => 'b'
)
```

Associative Arrays

- Similar to a Java HashMap
- Uses string keys

```
$foo = array('bar' => 'baz', 'rc' => 'car' );
$value = $foo['bar'];
```

Advanced Arrays

- Arrays can contain other arrays
- Multidimensional arrays
- Can be very deeply nested and pretty fast in PHP
- `print_r()` function can be helpful in visualizing a deeply nested array.

PHP Operators

- Operators take one or more values and yields a different value
- See the PHP documentation for all the operators and precedence

http://php.net/manual/en/language.operators.php

PHP Math

- PHP has a built-in math function library
- http://php.net/manual/en/ref.math.php
- Working with numeric data is very important and easy.
- Basic Math uses the Math operators, complex math uses these functions.

PHP Control Structures

- We often need to branch code depending on a condition or repeating code
- Language for branching might sound like:
  - Do this if that happens
  - If the data is this value – do this otherwise do that
  - Do this 5 times.
  - Do this for every item in this list.
- Control structures are detailed in the PHP docs

http://php.net/manual/en/language.control-structures.php

Assignments

- Get you local environment installed and working.
- Test the local environment with a phpinfo file and also some basic php to make sure everything is working.
- Possible discussion board posting will be coming. I will email if I want something posted.
- Assignment 1 will be posted soon

Reading

- Read over PHP documentation regarding topics covered tonight. Don't need to read all the pages for each function. Just the high level pages.
- Take a look at the additional resources I posted in blackboard so you know what they contain.

ITMD 462/562
Web Site Application Development

Lecture 3

Fall 2015 – September 9, 2015

Tonight's Agenda

- PHP Built-in Functions
- Custom Functions
- Form Processing

PHP Built-in Functions

String Manipulation Functions

- Functions to use and modify strings and their contents
- Very necessary to most applications. Most form data comes in as strings.
- http://php.net/manual/en/ref.strings.php

```
echo
print
sprintf, fprintf, printf
explode, implode
trim
htmlspecialchars
nl2br
strlen
substr
```

Math Functions

- Functions and constants for numeric manipulation, cleanup, and display
- http://php.net/manual/en/ref.math.php
- http://php.net/manual/en/math.constants.php

```
rand
mt_rand
min
max
ceil
floor
```

Variable Handling

- Ways to test for and convert variable values
- Since PHP doesn’t require a specific declaration of a variable you need to check for them before using them
- http://php.net/manual/en/ref.var.php

```
isset
empty
is_numeric
is_string
intval
floatval
strval
serialize, unserialize
```

Array Functions

- Functions used to count, manipulate, filter, merge, sort, and more
- http://php.net/manual/en/ref.array.php

```
count, sizeof
in_array
key_exists
array_pop
array_push
array_merge
sort
array_slice
```

Loop over elements in arrays:

```
foreach($array as $value){

}

foreach($array as $key=>$value){

}
```

http://php.net/manual/en/control-structures.foreach.php

Date / Time functions

- There is a set of functions for dealing with dates and times
- PHP also has an object-oriented interface for DateTime but we will look at that later. Better to use the classes for date time than the functional code ultimately
- http://php.net/manual/en/ref.datetime.php

```
date
getdate
time
date_default_timezone_set
strtotime
date_parse
```

Custom Functions

User Functions

- A function is a block of statements that can be reused in a program.
- Functions are executed by calling the function by adding parenthesis after the name
- User functions are declared with the `function` keyword

```
function functionName() {
    code statements to execute;
}
```

User Functions

- Any valid PHP code can be inside the function, including function and class definitions
- Same naming rules as variables. Start with a letter or underscore, then followed by letters, underscores, or numbers
- Functions don't necessarily have to be declared before they are used but must exist in the scope of the script before they are used.
- http://php.net/manual/en/functions.user-defined.php

Function Arguments

- You can pass information to the function via an argument.
- Arguments are comma-delimited lists of expressions which evaluate left to right

```
function echoString($aString) {
     echo $aString;
}

echoString('test');    // outputs test
```

- Arguments can have default values
- http://php.net/manual/en/functions.arguments.php

Function Return Values

- You can return a value from a function by using the `return` statement.
- Any data type can be returned by the function.
- The `return` statement causes the function to end immediately and pass back the value.
- If there is no `return` statement NULL will be returned

```
function squareNum($aNum) {
    return $aNum * $aNum;
}

$val = squareNum(4);
echo $var;  // outputs 16
echo squareNum(5); // outputs 25
```

http://php.net/manual/en/functions.returning-values.php

Anonymous Functions

- PHP does support anonymous functions, or closures.
- These are functions without a specific name defined.
- Most useful for values of callback parameters.
- http://php.net/manual/en/functions.anonymous.php

```
$helloWorld = function($aName) {
    echo ‘Hello’ . $aName;
}

$helloWorld(‘Brian’); // outputs Hello Brian
```

Forms

Processing user input

Structure

- Forms are the way for users to enter information into a web page and send it to the server for processing
- Doesn't always have to be a strict 'form' in the display. It can be hidden or disguised as a button, field, or other item
- HTML Form Tag

```
<form action="somescript.php" method="POST">
   <input>
    …
</form>
```

Form Action

- The action is the name or URL of the resource/script that you want to process the form input
- Often can be the same page
- If we detect if the page was called by a GET or POST request we could conditionally do different things or display different pieces of content using a conditional like an `if` statement

Form Method

- This will be the method the browser sends the request and form data to the server.
- Forms typically use POST, URLs typically use GET
- GET request has the form data or parameters as components of the URL
  - http://www.iit.edu/form.php?name=brian&school=sat
  - Need to be careful to URL encode your parameters
- POST requests encode the form data or parameters in the body of the request

Form enctype

- There is an HTML attribute on the form tag called enctype
- It specifies how the form data will be encoded when submitted to the server
- Only used for POST method
- Default Value – Ensures that all characters are encoded before they're sent to the server. This is what is used if you leave it off the form tag.
    - `application/x-www-form-urlencoded`
- If your form has a file upload control you must use a different enctype. This is required and not optional. It ensures no character conversions take place and transfers the form data as a compound MIME doc.
    - `multipart/form-data`

Input tags

- There are various form input controls to submit data such as text, password, radio, checkboxes, select lists, and others.
- The `name` attribute on the form control will be the variable name in PHP

```
<input type="text" name="firstName">
```

- These should be review from your HTML class
- https://developer.mozilla.org/en-US/docs/Web/Guide/HTML/Forms
- http://www.tutorialspoint.com/html/html_forms.htm
- http://www.w3schools.com/html/html_forms.asp
- http://www.w3schools.com/html/html_form_input_types.asp

Handle form input in PHP

Global Input Arrays

- The users input is placed in these arrays based on request method

`$_GET[]`

`$_POST[]`

`$_REQUEST[]`

- Request is the combination of the two other arrays, GET and POST. It is a copy of the data and not a reference to the data.
- They are global variables
- Files come in via the `$_FILES[]` array
- http://php.net/manual/en/reserved.variables.php

Type Conversion

- All form data comes in to the server in string form or arrays of strings.
- You needs some of the type conversion and variable functions we looked at earlier.
- Things to do
  - Check if the variable exists
  - Validate that the variable is in the correct format
  - Process the data to protect your site from security perspective
  - Things like isset, empty, strlen, is_numeric, floatval, ==, all help you

Input Safety and Security

- The User can input anything in the form controls. There is no guarantee that the user input didn't contain SQL, HTML, JavaScript
- JavaScript validation is commonly used to prevent the submission but it can be disabled in the browser and doesn't exist in CLI. YOU MUST DO SERVER SIDE VALIDATION TO BE SAFE
- You must clean and sanitize user input depending how it will be used.

`htmlspecialchars`

`addslashes`

`str_replace`

- Database specific functions exist and more

Testing for POST

- Best way to test if the request is POST is to use the SERVER array
- https://secure.php.net/manual/en/reserved.variables.php

```
if ($_SERVER[‘REQUEST_METHOD’] == ‘POST’)
```

- Other options include
- Adding a hidden form field and then look for that field in the post array

```
<input type=“hidden” name=“submitcheck” value=“submit”>
if (isset($_POST[‘submitcheck’]))
```

Assignments

- Assignment 1 is Posted
- Due **Sunday, September 20, 2015 at 11:59pm Chicago Time**
- We will discuss in class on September 23 so no submissions will be accepted after 6:00pm on September 23.

ITMD 462/562
Web Site Application Development

Lecture 4

Fall 2015 – September 16, 2015

Tonight's Agenda

- Error Reporting
- Using external PHP Files
- Classes and OOP
- Namespacing

Error Reporting

Displaying Errors

- You may notice that your pages do or do not output and display errors based on the configuration
- Often set in php.ini
- Settings can be overridden in your script

```
ini_set(‘display_errors’, 1);
error_reporting(E_ALL);
```

- http://php.net/manual/en/function.error-reporting.php

Using external files

Why put everything in a single file?

External File

- We need to extract some of our code to external files
- This allows us to reuse code and better maintainability
- Common Code or Class Files are typical

```
include

require

include_once

require_once
```

- http://php.net/manual/en/function.include.php

Classes, Objects, and OOP

Classes and Objects

- Object Oriented Programming (OOP) is a pattern that abstracts and groups similar functionality into classes.
- Classes and Objects are custom data types in your application that typically represent something.
- Classes are the blueprint or template for the object
- Objects are the instantiated concrete item produced from the class
- Greatly increases maintainability and reuse of components (DRY)
- http://php.net/manual/en/language.oop5.php

Class Definition

- Class definitions begin with the class keyword, then the name of the class, followed by curly braces enclosing the contents of the class
- Can not be a reserved word and must follow standard naming. Start with a letter or underscore and then can contain letters, numbers, or underscores.
- Classes contain
  - Constants
  - Variables – called properties in classes
  - Functions – called methods in classes

Class Definition

- By convention Class names typically start with a capital letter and then camelcase

  `ExampleClass`

- Classes have an internal `$this` variable that references the current object.

```
class SampleClass {
  // properties and methods go here
}
```

- Instantiated with the `new` keyword

```
$obj = new SampleClass();
```

http://php.net/manual/en/language.oop5.basic.php

Class Properties

- Class member variables are called properties
- Must be defined with one of the keywords `public`, `private`, `protected`, `static`
- May be initialized when defined with a value (no variable)
- Non-static properties are accessed with the arrow/object operator

  `$this->property` or `$obj->property`
- Static properties are accessed with the double colon operator

  `self::$property` or `MyClass::$property`
- http://php.net/manual/en/language.oop5.properties.php

Class Constants

- Defined with the `const` keyword. Not the define function.
- Do not use the `$` symbol

```
const MY_CONSTANT = ‘something’;
```

- Accessed with the double colon operator

```
self::MY_CONSTANT    or    MyClass::MY_CONSTANT
```

Visibility Keywords

- Three visibility keywords which must be used on properties and may be used on methods.
- Methods without a visibility keyword will be considered public

`public`

`private`

`protected`

- http://php.net/manual/en/language.oop5.visibility.php

Class Methods

- Functions defined within a class
- Can use visibility keywords, defaults to public
- `$this` variable is available to reference the current object instance.

```
class MyClass {
   public function myMethod() {
      echo ‘my method’;
   }
}

$obj = new MyClass();

$obj->myMethod(); // prints my method
```

Constructors, Destructors, and Magic Methods

- Constructor method for a class is defined with the 2 underscore construct name

```
function __construct(){
}
```

- Is called when the object is instantiated
- http://php.net/manual/en/language.oop5.decon.php
- Other magic methods exist like __toString
- http://php.net/manual/en/language.oop5.magic.php

Inheritance

- The is a way to extend another class
- Subclass inherits all the public and protected methods from parent
- It allows you to override methods in the parent class
- Uses the extends keyword

```
class Bar extends Foo {
}
```

- http://php.net/manual/en/language.oop5.inheritance.php

Static Keyword

- If you use the static keyword when declaring a property or method it is available as a class property or method without instantiating the object
- Access with the double colon (Scope Resolution Operator)

```
print Foo::$my_static
```

- http://php.net/manual/en/language.oop5.static.php

Abstract Class, Interface, Traits

- Classes defined as abstract can not be instantiated
- http://php.net/manual/en/language.oop5.abstract.php

- Interfaces are code which specifies what methods a class must implement
- http://php.net/manual/en/language.oop5.interfaces.php

- Traits, >= 5.4, a way for code reuse in single inheritance languages, like a class but can’t be instantiated
- http://php.net/manual/en/language.oop5.traits.php

Final Keyword

- Prevents child subclasses from overriding a method.
- If the class is declared final it can not be sub classed.
- http://php.net/manual/en/language.oop5.final.php

Much More

- Please read through the API docs classes and objects section
- http://php.net/manual/en/language.oop5.php

Namespace

Too many classes and methods in the global namespace

Namespaces

- Namespaces are a way to organize code so everything is not defined in the global namespace.
- Similar to a Java package
- Solution for problems with naming collisions
- Uses the `namespace` keyword to define the namespace for the following code
- Uses the backslash character `\` as a separator
- When using namespaced code you can use the `use` keyword to alias the namespace
- http://php.net/manual/en/language.namespaces.php

Namespace Tutorials

- Demo
- Check out these resources for additional info
- https://knpuniversity.com/screencast/php-namespaces-in-120-seconds
- http://daylerees.com/php-namespaces-explained/
- http://code.tutsplus.com/tutorials/namespacing-in-php--net-27203
- http://www.sitepoint.com/php-53-namespaces-basics/

Assignments

Reading

- Posted in blackboard in the assignments section.
- Read object oriented PHP for beginners tutorial
- Read through php.net documentation on classes and objects that we did not cover.
- Read some of the namespace documentation links listed on the previous slide. You need to understand the namespace concept. At a minimum watch the PHP namespaces in 120 seconds video.
- https://knpuniversity.com/screencast/php-namespaces-in-120-seconds

Assignments

- Assignment 1 is Due this weekend
- Due **Sunday, September 20, 2015 at 11:59pm Chicago Time**
- We will discuss in class on September 23 so no submissions will be accepted after 6:00pm on September 23.

ITMD 462/562
Web Site Application Development

Lecture 5

Fall 2015 – September 23, 2015

Tonight’s Agenda

- Uploading files with POST
- Review AS1 Possible Solution
- Working with the file system
- Saving/reading PHP data to/from a file

Uploading Files

POST File Uploads

- Tutorial examples on how to do file uploads with a POST form
- http://php.net/manual/en/features.file-upload.post-method.php
- http://www.sitepoint.com/file-uploads-with-php/
- Let’s look at the php.net example.

POST File Uploads

- Important things to note.
- Form tag must have the enctype attribute with the value being "multipart/form-data" or it will not work
- MAX_FILE_SIZE hidden field can be used to save the user the trouble up uploading a file bigger than php is configured for but can be tricked.
- File is uploaded to a temporary directory on the server
- All file metadata is stored in the PHP superglobal $_FILES['fieldname'] array
- Use the move_uploaded_file function to move the file from the temp directory to your final location
- http://php.net/manual/en/features.file-upload.post-method.php

Assignment 1

Possible approach to Assignment 1

- Demo

PHP File System Use

Reading, Writing, Moving, and any other file needs

File System Related Extensions

- These functions and objects are used to interact with the underlying file system on the server
- http://php.net/manual/en/refs.fileprocess.file.php
- We will concentrate on the Filesystem extension tonight
- Often we need to read the contents of a file to process the data
- Often we want to save the state of our application into a file for later use
- Sometimes we just want to write lines to a file for logging or other purposes

Opening a file

- First need to open a file which returns a resource handle. fopen function

```
<?php
   $handle = fopen("data.txt", "r");
?>
```

- First parameter is the file path, second is the file mode.
- http://php.net/manual/en/function.fopen.php

```
<?php
   if(file_exists("data.txt")){
      $file = fopen("data.txt", "r");
   } else{
      die("Error: The file you are trying to access doesn't exist.");
   }
?>
```

Closing a file

- When finished working with the file it needs to get closed. `fclose` function

```
<?php
   $file = "data.txt”;

   // Open the file for reading
   $handle = fopen($file, "r") or die("ERROR: Cannot open the file");

   // Some code to be executed

   // Closing the file handle
   fclose($handle);
?>
```

- http://php.net/manual/en/function.fclose.php

Reading Strings of Characters

- Many PHP functions for reading in data from a file. 1 character to whole file.
- The `fread` function can read strings of characters.

```
fread(file handle, length in bytes)
```

- http://php.net/manual/en/function.fread.php

Reading a String of fixed length

```php
<?php
$file = "data.txt";

// Open the file for reading
$handle = fopen($file, "r") or die("ERROR: Cannot open the file");

// Read in the entire file
$content = fread($handle,"20");

// Closing the file handle
fclose($handle);

// Display the file content
echo $content;
?>
```

Reading an Entire File

- The `fread` function can be used with the `filesize` function to read the whole file at one time.
- Use the `filesize` function in place of the length in bytes parameter of `fread`

```
$content = fread($handle, filesize($file));
```

Easier way to read entire file

- You can use the `file_get_contents` function to read the entire file into a string.
- http://php.net/manual/en/function.file-get-contents.php

```
<?php
$file = "data.txt";

// Reading the entire file into a string
$content = file_get_contents($file) or die("ERROR: Cannot
open the file");

// Display the file content
echo $content;
?>
```

Yet another way

- You can use the php file function
- Similar to file_get_contents but it reads the contents into an array of lines vs a single string
- http://php.net/manual/en/function.file.php

```
<?php
$file = "data.txt";

// Reading the entire file into an array
$arr = file($file) or die("ERROR: Cannot open the file");
   foreach($arr as $line){
   echo $line;
}
?>
```

Basic file writing

- Many PHP functions for writing data to a file.
- The `fwrite` function can write strings to a file.
- Make sure `fopen` mode is writable

```
fwrite(file handle, string data)
```

- http://php.net/manual/en/function.fwrite.php

Basic writing example

```
 <?php
$file = "note.txt";

// String of data to be written
$data = "The quick brown fox jumps over the lazy dog.";

// Open the file for writing
$handle = fopen($file, "w") or die("ERROR: Cannot open the file");

// Write data to the file
fwrite($handle, $data) or die ("ERROR: Cannot write the file");

// Closing the file handle
fclose($handle);

echo "Data written to the file successfully";
?>
```

Alternative way

- You can use the `file_put_contents` to write data to file without opening it
- Can use the `FILE_APPEND` parameter to append vs overwrite
- http://php.net/manual/en/function.file-put-contents.php

```
<?php
$file = "note.txt";

// String of data to be written
$data = "The quick brown fox jumps over the lazy dog.";

// Write data to the file
file_put_contents($file, $data) or die("ERROR: Cannot write the
file");

echo "Data written to the file successfully";
?>
```

Many other filesystem function

- http://php.net/manual/en/ref.filesystem.php

fread()	Reads a string of characters from a file.
fwrite()	Writes a string of characters to a file.
fgetc()	Reads a single character at a time.
feof()	Checks to see if the end of the file has been reached.
fgets()	Reads a single line at a time.
fgetcsv()	Reads a line of comma - separated values.
fputcsv()	Format line as CSV and write to file pointer
file()	Reads an entire file into an array.
file_get_contents()	Reads an entire file into a string without needing to open it.
file_put_contents()	Writes a whole string to a file without needing to open it.
fpassthru()	Displays the contents of an open file.
readfile()	Displays the contents of a file without needing to open it.
fseek()	Moves the file pointer to a specific location within an open file.
ftell()	Returns the position of the file pointer.
rewind()	Moves the file pointer to the start of the file.

Reading and Writing Data in CSV

- Comma Separated Values (CSV) is a common data format.
- Values are stored in a text file, each row is one item, each items columns are separated by comma.
- Might have a header row or not
- PHP has a couple functions that help us with this task
- `fgetcsv` and `fputcsv`
- http://php.net/manual/en/function.fgetcsv.php
- http://php.net/manual/en/function.fputcsv.php

Object Serialization

- Serialization is a way to represent an Object as a string of text for storage in a file or database.
- Two primary functions `serialize` and `unserialize`
- Serialize creates the data string from the object and unserialize recreates the object from the string.
- http://php.net/manual/en/language.oop5.serialization.php
- http://php.net/manual/en/function.serialize.php
- http://php.net/manual/en/function.unserialize.php

Assignments

Assignments

- Assignment 2 will be finalized and released this weekend.

ITMD 462/562
Web Site Application Development

Lecture 6

Fall 2015 – September 30, 2015

Tonight’s Agenda

- Sessions & Cookies
- CRUD
- CRUD Application Demo

Sessions & Cookies

Session Basics

- Sessions are a way to store simple data for individual users based on a unique session ID.
- This can persist data between application pages.
- Session IDs are usually sent to the browser and server via cookies
- The actual session data is stored on the server
- $_SESSION superglobal is where the data is stored.
- session_start() function manually starts the session
- http://php.net/manual/en/session.examples.basic.php

Cookies

- Cookies are a way of storing simple text data in the users browser.
- Returned to the server that set it when a new request is sent to the server.
- Only the domain that set the cookie can read it.
- If you want to set a cookie it must be called before any output is sent to the browser.
- Cookie data is in the $_COOKIE superglobal
- set_cookie() function sets a cookie.
- http://php.net/manual/en/features.cookies.php
- http://php.net/manual/en/function.setcookie.php

CRUD

CRUD

- Create, Read, Update, Delete
- 4 Basic operations of persistent storage
- Can be implemented in numerous persistence layers
  - SQL Databases
  - Object Databases
  - XML DB, NoSQL, and other database formats
  - Flat Files
  - Custom Formats and more

CRUD

- Typically map to database or application layer operations
- Most often SQL or database language
- Also maps to specific HTTP verbs for REST model
- Fundamental operations to interact with an application and data

Operation	SQL	HTTP	DDS
Create	INSERT	PUT / POST	write
Read (Retrieve)	SELECT	GET	read / take
Update (Modify)	UPDATE	PUT / PATCH	write
Delete (Destroy)	DELETE	DELETE	dispose

Data Access

Data Access Layer

- You could write direct data access code in your application everywhere it is needed.
- This leads to a lot of locations to test and change if you data access model changes.
- One Data Access Pattern that has emerged is the Repository Pattern

Repository Pattern

- Adds a separation layer between the data mapping and the domain layers of an application and acts like an in-memory collection of domain objects.
- Objects are added and modified through the repository
- This encapsulates the specific data access methods behind the scenes.
- Helps to enforce a clean separation between data mapping and domain logic layers
- If done well, the data storage layer can be changed with minimal changes to application or domain layer logic.
- http://martinfowler.com/eaaCatalog/repository.html
- http://shawnmc.cool/the-repository-pattern

Repository Pattern

- Commonly a more generic interface is defined in a repository object.
- That interface defines the methods that will need to be implemented by any class that implements that interface.
- Specific Classes will be created that implement that interface which implement the methods based on the underlying data storage method
- http://shawnmc.cool/the-repository-pattern

Assignments

Assignments

- Assignment 2 will be posted tonight.

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.

ITMD 462/562
Web Site Application Development

Lecture 8

Fall 2015 – October 14, 2015

Tonight's Agenda

- Next Week
- Database
- SQL Introduction

Next Week

- No Physical Class Next Week
- I will be uploading a replacement video (auth demo)
- Read/Watch the SQL tutorials
- Play with MySQL and SQLite. Get environment working
- Midterm Exam will be assigned Sunday Night at midnight. You must complete by Wednesday Oct 21 11:59pm Chicago time. **NO EXCEPTIONS**
  - Once you start you will have a time limit, probably 2 hours.
  - You must finish it once you start it.
  - Email will be sent from blackboard when it is available with all the details

Databases

Database

- Databases are collections of data stored in an organized way.
  - Tables
  - Queries
  - Reports
  - Views
  - Other Objects
- Access to this data is provided by a DBMS (Database Management System)

Databases

- Technically the database is the data and how it is organized. The DBMS is how you access the data.
- DBMS provides functions to allow 4 basic groups of functionality: Data Definition, Update, Retrieval, and administration.
- Read about the history of DBs here
  - https://en.wikipedia.org/wiki/Database
- Why Use a Database?
  - http://arstechnica.com/information-technology/2013/05/why-use-a-database-instead-of-just-saving-your-data-to-disk/

DBMS

- Database Management System (DBMS)
  - Computer software that has interactions with the user, other software, the database itself to allow for storage, and analysis of data.
  - Database itself is not portable between different DBMS
  - Examples include MySQL, PostgreSQL, Microsoft SQL Server, Oracle, Sybase, DB2, SQLite
  - Typically client-server model, except SQLite
- We will look at SQLite and MySQL over the coming weeks
- Most follow a Relational Model using a SQL based language, but there are others including NoSQL DBMS that do not use SQL.

Relational Database

- Based on a relation model invented by E. F. Codd of IBM in 1970.
  - Presents data as relations. Collections of tables made up of rows and columns.
  - Provides operations to manipulate the tabular data
  - Codd's 12 Rules https://en.wikipedia.org/wiki/Codd's_12_rules
- Basic definition is that it presents a view of data as a collection of rows and columns. Using this definition RDBMS don't always implement all of the 12 rules.
- Some say for a database to truly Relational it needs to implement all 12 rules.
- Most RDBMS use SQL as the query language
- Most popular are Oracle, Microsoft SQL, IBM DB2, and MySQL
- https://en.wikipedia.org/wiki/Relational_database_management_system

NoSQL Database

- Data is modeled in a way other than relational tables
- Doesn't use SQL but some systems do support some SQL like languages
- Some advantages or RDBMS with scaling and performance for certain tasks
- Data model based on Column, Document, Key-value, Graph, Multi-model
- Document based hold individual documents in collections. Subclass of a Key-Value model. MongoDB and CouchDB two popular Document based. Often modeled/stored as JSON data.
- Key-value model works like an associative array (map or dictionary) where data is stored in a key.
- https://en.wikipedia.org/wiki/NoSQL

MySQL

- Open source RDBMS written in C and C++
- Runs on many platforms
- Most widely used open source RDBMS and a top DB overall
- Originally created by a Swedish company MySQL AB and Michael Widenius
- Bought by Sun in 2008 and then Oracle in 2010 when they bought Sun
- https://en.wikipedia.org/wiki/MySQL
- Some concern when Oracle aquired MySQL led Michael Widenius to fork a GPL only version called MariaDB. Same codebase as MySQL 5.5 and aims to maintain compatibility with Oracle versions.
- https://en.wikipedia.org/wiki/MariaDB

SQLite

- RDBMS written and used as a C programming Library
- Designed in 2000 by D. Richard Hipp
- It is not a client-server database system. No outside process.
- It is used as a library, linked, and embedded in the application
- It stores the database as a cross platform compatible file on the hosts machine / application. The file is locked while writing.
- No Users. Access control by file system.
- Supports most standard SQL with some limits
- Bindings for all major programming languages
- https://en.wikipedia.org/wiki/SQLite

SQL

A quick introduction to basic SQL

Basic SQL

- This lecture is not meant to be a compressive lesson on SQL.
- We will talk about the basics. There are two database courses that go into much more detail
- If you have not taken those courses you need to read/watch one or more of the following tutorials
- https://www.codeschool.com/courses/try-sql
- http://www.w3schools.com/sql/default.asp
- http://www.tutorialspoint.com/sql/

Basic SQL

- SQL – Structured Query Language
- Special programming language for accessing and managing data stored in a RDBMS.
- Based on relational algebra and tuple calculus
- One of the first languages developed for Codd's relational model and has become the most widely used language for databases.
- Became standardized in 1986-87 by ANSI and ISO
- Not completely portable between RDBMS. Sometimes some statements need slight modifications but mostly compatible.
- Consists of a data definition language, data manipulation language, and a data control language.
- https://en.wikipedia.org/wiki/SQL

Basic SQL Components

- data definition language
  - Syntax for defining data structures and especially database schemas
  - CREATE, DROP, ALTER, rename
  - https://en.wikipedia.org/wiki/Data_definition_language
- data manipulation language
  - Syntax for selecting, inserting, deleting, and updating data
  - SELECT, INSERT, UPDATE, DELETE
  - SELECT is technically outside DML because it is read only but considered DML
  - https://en.wikipedia.org/wiki/Data_manipulation_language
- data control language
  - Syntax to control access to data
  - GRANT, REVOKE
  - https://en.wikipedia.org/wiki/Data_control_language

Data Definition Language

- CREATE – Used to create Databases and Tables
  - `CREATE DATABASE dbname;`
  -
    ```
    CREATE TABLE table_name
    (
    column_name1 data_type(size),
    column_name2 data_type(size),
    column_name3 data_type(size),
    ....
    );
    ```
- DROP – Used to delete indexes, tables, and databases
  - `DROP TABLE table_name`
- ALTER - Used to add, delete, or modify columns in an existing table.
  - `ALTER TABLE table_name ADD column_name datatype`

Data Manipulation Language - SELECT

- Simple Query with SELECT - Used to select data from a database.
  - Results are stored in a result table or result set

  -
    ```
    SELECT column_name,column_name
    FROM table_name;
    ```

  -
    ```
    SELECT * FROM table_name;
    ```
- Can add a WHERE clause to filter results or ORDER BY to sort

  -
    ```
    SELECT column_name,column_name
    FROM table_name
    WHERE column_name operator value;
    ```
  - Single quotes required around non-numeric values

  -
    ```
    SELECT column_name, column_name
    FROM table_name
    ORDER BY column_name ASC|DESC, column_name ASC|DESC;
    ```

Data Manipulation Language - INSERT

- INSERT - Used to insert new records in a table.
- Do not specify column names

-
  ```
  INSERT INTO table_name
  VALUES (value1,value2,value3,...);
  ```
- Specify column names

-
  ```
  INSERT INTO table_name
  (column1,column2,column3,...)
  VALUES (value1,value2,value3,...);
  ```
- Remember to single quote all non-numeric values

Data Manipulation Language - UPDATE

- UPDATE - Used to update records in a table

-
  ```
  UPDATE table_name
  SET column1=value1,column2=value2,...
  WHERE some_column=some_value;
  ```
- The WHERE clause selects which record or records that should be updated. If there is no WHERE clause, **all records will be updated!**
- Remember to single quote all non-numeric values

Data Manipulation Language - DELETE

- DELETE - Used to delete records in a table.
- Deletes rows in a table

-
  ```
  DELETE FROM table_name
  WHERE some_column=some_value;
  ```
- The WHERE clause selects which record or records that should be deleted. If there is no WHERE clause, **all records will be deleted!**
- Remember to single quote all non-numeric values

Misc SQL

- LIKE clause - Used to search for a specified pattern in a column

  -
    ```
    SELECT column_name(s)
    FROM table_name
    WHERE column_name LIKE pattern;
    ```
- IN operator - Specify multiple values in a WHERE clause

  -
    ```
    SELECT column_name(s)
    FROM table_name
    WHERE column_name IN (value1,value2,...);
    ```
- JOINS - Combine rows from two or more tables
- NOT NULL constraint enforces a column to NOT accept NULL values
- SELECT DISTINCT - Used to return only distinct (different) values
  - `SELECT DISTINCT column_name,column_name FROM table_name;`

Misc SQL

- PRIMARY KEY constraint uniquely identifies each record in a database table.
- A FOREIGN KEY in one table points to a PRIMARY KEY in another table.
- SQL includes functions to calculate certain results for example: sum, avg, count, min, max, group by, and more.
- SQL Datatypes, functions, and some syntax may vary from DBMS, consult specific documentation

SQL Injection

- Be careful to prevent SQL injection so a hacker cannot destroy or alter your database in ways you don't expect.
- Be cautious to allow user entered data directly in your SQL string.
- Either carefully sanitize the data or use another method.
  - `mysqli_real_escape_string`
  - http://php.net/manual/en/mysqli.real-escape-string.php
- Using SQL bound parameters, prepared statements, or Specific Object-based APIs will help to prevent this.
- http://wiki.hashphp.org/Validation
- http://www.w3schools.com/sql/sql_injection.asp
- http://www.tutorialspoint.com/sql/sql-injection.htm

SQL and PHP

SQL in PHP

- We are going to talk about MySQL and SQLite in class
- There are numerous native PHP APIs and also the newer PHP PDO API
- PDO is a lightweight, consistent interface for accessing databases in PHP. Often you can change database technology by just changing the connection.
  - PDO - http://php.net/manual/en/intro.pdo.php
  - mysqli - http://php.net/manual/en/book.mysqli.php
  - SQLite3 - http://php.net/manual/en/book.sqlite3.php
  - sqlite - http://php.net/manual/en/book.sqlite.php
- PHP Database Access Tutorial - http://code.tutsplus.com/tutorials/php-database-access-are-you-doing-it-correctly--net-25338

SQLite & MySQL PDO

- Here are three SQLite tutorials for PHP to help you
- http://zetcode.com/db/sqlitephp/
- http://henryranch.net/software/ease-into-sqlite-3-with-php-and-pdo/
- http://www.tutorialspoint.com/sqlite/sqlite_php.htm
- http://wiki.hashphp.org/A_PDO_Example_Using_SQLite
- Here is a MySQL PDO Tutorial
- http://wiki.hashphp.org/PDO_Tutorial_for_MySQL_Developers

Tools to work with databases

- These tools will let you interact with the underlying database
- SQLite
  - SQLite Browser
    - http://sqlitebrowser.org/
  - Firefox Extension SQLite Manager
    - https://addons.mozilla.org/en-US/firefox/addon/sqlite-manager/
- MySQL
  - MySQL Workbench – Official Oracle App
    - https://www.mysql.com/products/workbench/
  - phpMyAdmin – (should be already installed in your local environment)
    - https://www.phpmyadmin.net/
- There are others for both. These are just a couple.

Assignments

Reading

- If you have not taken those courses you need to read/watch one or more of the following tutorials
  - https://www.codeschool.com/courses/try-sql
  - http://www.w3schools.com/sql/default.asp
  - http://www.tutorialspoint.com/sql/
- Read PHP PDO Tutorial
  - http://code.tutsplus.com/tutorials/php-database-access-are-you-doing-it-correctly--net-25338
- Attempt to create a SQLite Database and MySQL database using the database management applications we discussed in class.
- Good PHP resource -> http://wiki.hashphp.org/Main_Page

Assignments

- Assignment 2 is up and due this Sunday Oct 18 11:59 Chicago time.
- Remember to use the discussion board. There is a very small amount of points allocated to participating in the discussion board.
- Take Quiz 1 – online – before Sunday Oct 18 11:59 Chicago time.
- Midterm will be assigned next Monday. You will have until Wednesday October 21 11:59pm Chicago Time to take.
  - Once you start a timer will start. Probably 2 hours.
  - Can take it anytime between Monday and Wednesday
  - I will send an email with details as soon as it is posted.
  - You will not be able to take the midterm after this time.