Course Notes for

CS 1520
Programming Languages for Web Applications

By
Matt Bowytz
Department of Computer Science
University of Pittsburgh

- These notes are intended for use by students in CS1520 at the University of Pittsburgh and no one else
- These notes are provided free of charge and may not be sold in any shape or form
- These notes are NOT a substitute for material covered during course lectures. If you miss a lecture, you should definitely obtain both these notes and notes written by a student who attended the lecture.
- Material from these notes is obtained from various sources, including, but not limited to, the following:
    - Programming the World Wide Web, various editions, by Robert W. Sebesta
    - Online documentation at: http://www.php.net and in particular the PHP Manual at http://www.php.net/manual/en/
    - PHP and MySQL Web Development, by Luke Welling and Laura Thomson (Developer's Library)
    - The Web Wizard's Guide to PHP by David A. Lash (AW)

- To understand and be able to program web servers and web clients to accomplish useful tasks
- Web servers are ubiquitous and many / most organizations now have elaborate / interactive web sites
- These web sites require programming on the server side
  - Scripts that execute on the server and return resulting documents to the client
  - For server-side scripts we will primarily utilize the **PHP language**

These web sites require programming on the client side

- Scripts that execute locally on a user's machine via the web browser
- For client-side scripts we will primarily utilize the **Javascript language**
- We will also utilize **JQuery**, a library of Javascript functions that simplifies much of the client-side programming
- We may also look at an API, such as Google Maps that can be integrated into our sites

These web sites often require interaction between web servers and web clients

- More and more sites contain resident client code which accesses the server asynchronously and dynamically updates the client page
- We will utilize **DOM** and **AJAX** for this interaction
  - Document Object Model
  - Asynchronous Javascript And XML

- To examine and understand the differences in various programming languages, focusing especially on those geared toward Web programming
- Many programming languages are used for general and Web programming
  - See just the previous few slides for examples
- How do these languages differ?
  - Why is one better suited to a specific purpose than another?
- Now some more fine-grained goals:

- To learn and utilize the PHP language, and its applications
- PHP syntax, constructs and basic language features
- Using PHP for server-side embedded scripts
    - Writing and processing HTML forms with PHP
    - Access and utilizing system state variables
    - Maintaining individual states via cookies and sessions

  Regular expression handling in PHP

  Object-oriented features of PHP

Using PHP for Web database access

- With a MySQL database
  - We will cover the very basics of using MySQL and DB development as well
  - Simple database setup and querying

Simple authentication and authorization

- Multiple methods
  - Within the site
  - External Services (fb, GitHub, google, etc.)

- To learn and utilize the Javascript language in conjunction with html files
- Learning the basics of the language
    - Quickly
- Using Javascript in conjunction with DOM to add dynamic content to web pages
    - Ex: Doing client-side processing of forms
        - Checking form input, dynamic formatting, etc.
- Using AJAX techniques to make web apps more like stand-alone applications
    - Updating a web page without requiring a full refresh

- To learn and utilize the JQuery library
  - Many client-side actions can be simplified greatly utilizing JQuery rather than straight Javascript
  - JQuery has many powerful plugins
- To learn and utilize XML and how it can be used in Web applications
  - Basic idea / syntax of XML
  - Parsing / formatting XML documents

- If time permits…
- To learn the basics of other languages or frameworks
- To look at an API such as Google Maps and how it can be integrated into our sites
- To examine mobile web apps and how they differ from traditional web apps
    - Content and formatting differences
    - Responsive vs. Adaptive
    - Native vs. Web

- What is assumed of you:
  - You should be familiar with Java and have used it in CS0401/445
- You are expected to know object-oriented programming and basic event-driven programming
  - You are expected to know the basics of html
- What is not assumed of you:
  - Detailed knowledge / experience with any of the languages we will be using

- How do Web Servers work?
  - Client specifies document at a specific web address that is desired (specified by a URL)
  - Ex: http://www.cs.pitt.edu/
  - If the document is HTML or text, the server simply forwards it back to the client
  - If it is text, it is shown unaltered in the browser
  - If it is HTML it is rendered in the client's browser
    - HTML tags are interpreted and result is shown to the user
    - For info on HTML/XHTML, read Ch. 2 in the Sebesta text

- However, the requested document may be an executable script (PHP), or it may be HTML with an embedded script(JS)
  - The script could be written in any of many different web scripting languages
  - PHP, JS, ASP, JSP, Perl, Python
- In these cases, the server executes the script
  - If the entire document was a script, the server simply sends the output back to the client
  - If the document had an embedded script, the script sections are replaced with the output and the modified document is then sent to the client

- Note that **the client never sees the server-side script code**
  - This is important – typically client should not see logic / code that the server executes to process requests
  - The server may be accessing files whose names should not be seen, or preprocessing data that it does not want the client to see
- See ex1.txt
- See ex1.html
- See ex1.php

- In order to be useful, web servers need access to the directories from which they serve and process files
- Minimally the server must have read access, but in many cases, write access is also needed and/or useful
    - In case server creates / modifies files
        - Ex: User submits a file
        - Ex: Various user data is stored / updated in a file
- This leads to an issue for a course like CS 1520:

- How do we give the server access to students' individual directories in isolation?
    - For student X the server should have read/write access to X's directory
    - However, when executing a script in X's directory, the server should not be able to access files in the directory of some other student, Y
- There are ways of doing this but they are not simple
- For now we are eliminating this issue entirely

- Rather than using a shared Web server, you will each use your own server
- The server you will use is XAMPP
- This is a full-featured Apache / MySQL / PHP / Perl development environment
- It can be installed on a laptop
    - If you have one
- It can be installed on a flash drive
    - Makes it very portable
- See: https://www.apachefriends.org/

- You will develop your projects on your own servers, and then run them for your TA via a demonstration
  - You will need to bring either your laptop or your flash drive (with the server on it) to the demonstration
- **Assignment:**
  - Your first in-class exercise will be to run a simple script on your own XAMPP server
  - You have from now until then to get it installed on either your laptop or on a flash drive

- If you are putting it on your laptop the installation is fairly straightforward
- If you are putting it on a flash drive, you don't actually "install" it – just download the .zip file, unzip it and run a .bat script
- See documentation for where to put your scripts so that they will run
- If you have any trouble with this, see your TA or me before next class!

- HTML is a mark-up language
- Idea is that extra characters / symbols in the text provide information to a parser, which uses that information to render the document in a certain way
    - Ex:

      <h1>Hello There</h1>

      –The tags do not appear in the rendered document

      –The parser utilizes them to alter the appearance of the text
- We will discuss mark-up languages in more detail when we discuss XML later in the term

- HTML has evolved greatly over the years
  - New tags have been added
  - Some obsolete tags have been removed
  - Syntax has been standardized
- The current version is HTML 5
  - Still not universally used
  - However, recently updated browsers should all support it for the most part
  - Allows the language to be more semantic

- Fundamentally, each HTML 5 document has the following shell:

```
<!DOCTYPE html>
<html>
    <head>
    </head>
    <body>
    </body>

</html>
```

  - There are a LOT of other tags / attributes and variations
  - We will discuss more of these in detail later, when we focus on client-side programming

- HTML can be generated in several ways:
  - Can by typed in directly, using a text editor
    - Ex: TextWrangler, TextPad, vi
  - Can be generated through software
    - Ex: MS Word, IDEs
  - Can be generated dynamically via scripts
    - Ex: PHP, Perl
  - In this course we will be either typing in HTML directly or generating dynamically via scripts
    - Ex: Using PHP or JavaScript

- CSS - Cascading Style Sheets
  - CSS allows you to change and enhance the appearance of elements in a web site
  - Can happen inline (sparingly, or will happen inline with js manipulation)
  - Can live in an external, linked file
  - Target elements by class, id and other attributes
  - Pre-compilers take styling to a whole new level
    - SASS, LESS, Stylus

- Conclusion
    - Course Logistics
    - Course Objectives
    - PHP, SQL, HTML, JAVAScript, …