**Announcements**

- **Class Web Site:**
  - You can find this link at the end of the main passport site

- **E-mail Account**
  - Get your own e-mail account if you don’t have one

- Announcements’ section in web site

- Rules regarding Forum Use

- Academic Integrity

- Reminder to your parents

- Be on time

- Slides
Fundamentals: Client/Server

- Client and server are two terms frequently used
- Client/Server Model
- Client/Server model when talking about software
- Client/Server model when talking about hardware
Fundamentals: IP Addresses

- **IP Address** - Unique address for machine on internet
  - Get from ISP when connecting to internet
  - Allows network to find your machine

- **Format**
  - 32-bit unsigned integer ➔ 128.8.128.8

- **Domain Name**
  - Text name corresponding to the numeric IP address
  - Example: wikipedia.org

- **Name and address for local machine**
  - localhost
  - 127.0.0.1

- **Running out of 32-bit IP addresses**
Fundamentals: Web Server

- Web Server –
  - computer program that delivers (serves up) web pages.
  - It is like a person that is in charge of a warehouse

- Four popular Web Server Programs
  - IIS – Internet Information Services
  - Sun Java System Web Server

- Web server statistics

- Some web server statistics (by domain)
Fundamentals: DNS

- DNS – Domain Name Systems
- Protocol for translating domain names to IP addresses
  - Example: cs.umd.edu → 128.8.128.44
- Multiple DNS servers on internet
- DNS server may need to query other DNS servers
  - edu DNS server queries umd.edu server to find cs.umd.edu
Fundamentals: URLs

- URL – Uniform Resource Locators
- Represent web resources
  - Arbitrary files
  - Web pages
- Examples
  - https://login.yahoo.com/
  - file://dir/my.txt
Fundamentals: URL Structure

- URL consists of
  - Protocol
    - http
    - ftp
    - https (secure http)
    - file
    - ...
  - IP address (or domain name)
  - Port (optional most of the time)
  - path
Firefox

- Browser we will use

- Extensions we would like to have
  - Error Console
HTML

- Language used to define web pages
- What the server sends to the browser
- Browser reads HTML and renders the page
  - May require downloading data from server (e.g., images)
HTTP

- Hypertext Transfer Protocol (HTTP) – protocol that defines how user agents (e.g., browser) and web server can communicate
- HTTP is a request/response protocol between clients and servers
- Some methods (operations) defined as part of the protocol
  - GET – Use to download a resource (e.g., image, web page). Most common method used.
  - HEAD – Returns only the header
  - POST – Submits data (e.g., form data) to the server
- Do not confuse with HTML
- Demo
Creating Web Pages

- HTML - Hypertext Markup Language
- HTML Standard
  - Developed by the World Wide Web Consortium (W3C)
  - http://www.w3.org
  - Latest version HTML 4.01
- Document is described through a series of commands and directives present in a text file.
- HTML goal is to describe structure only. Presentation should be left to cascading style sheets.
- When interpreted by an HTML viewer, those commands determine the appearance of the page
- HTML documents are entirely ASCII text
- Commands are explicitly inserted
- Great HTML/CSS tutorial site:
  - http://www.htmldog.com/
Three versions of HTML

- HTML 4.01 Strict (excludes deprecated tags and attributes)
- HTML 4.01 Transitional (less restrictive including appearance elements)
- HTML 4.01 Frameset (identical to transitional but allows <body> to be replaced with <frameset>)

Web Standards Project (www.webstandards.org)

- Industry watchdog convincing web browsers developers to adhere to web standards.

HTML 4.01 is the last version for HTML. Next version is XHTML 1.0

XHTML

- Uses same tags as HTML 4.01
- Enforces rules like closing tags, tags in lowercase, and others.

We will use XHTML in this class
Validation

- You can use W3C Markup Validation Service (http://validator.w3.org/) to validate your html.
- Also through firefox you can use tidy for html validation. Tidy also provides suggestions for code that cannot be validated.
HTML Tags

- Tag – specifies a command or directive. It surrounds content and apply meaning to that content

- General format: `<elementName attributes>`

- Most HTML elements have two tags:
  - start tag and end tag
  - Example: `<h1> text </h1>`

- Tags and attributes will be in lowercase (XHTML requirement)

- Some tags are self-closed (ending them in `/`):
  - `<hr />`  `<br />`  `<meta />`  `<img />`
Attributes

- An attribute extends or modify a tag
- Attributes
  - Only appear in the start tag
  - You can have several attributes in one tag each separated by spaces
  - Order is immaterial
  - Some take values which are specified after an =
- General format
  - `<ELEM ATTR="attrValue">Displayed Text</ELEM>`
- Example
  - `<img src="bear.gif" width="100" height="75" alt="bear image" />`
- All attribute values will be enclosed in “ “ for XHTML compliance.
Nested Tags / Spaces / Comments

- Nested tags are possible but don’t overlap sets of them. Avoid the following:
  \[
  \langle i \rangle \langle b \rangle \text{Message} \langle /i \rangle \langle /b \rangle
  \]

- Browser Processing
  - Multiple spaces are converted to one space
    John
    Mary
    Peter
    John Mary Peter
  - Line returns are ignored
  - Unrecognized tags are ignored
  - Comments
    - Represented by \langle !-- --> \rangle Note: (two sets of double -)
    - Examples
      - \langle !--The html code example starts at this point--\rangle
    - Comments can not be nested
HTML Basic Skeleton

- An html document has two main parts.
  - **Header** – provides information about the document
  - **Body** – contents of the page
- **Example 1 (htmlDoc1.html)**

```html
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
 "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" lang="en" xml:lang="en">
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1"/>
    <title>Template</title>
  </head>
  
  <body>
    <!--HTML CODE HERE-->
  </body>
</html>
```

- Let’s validate the above document
How to Develop HTML Documents

- Text Editor
  - Any text editor (e.g., wordpad, notepad, pico, etc.)
- HTML Editors
  - Utilities designed to write HTML
  - Examples: CoffeeCup HTML Editor, HTMLjive
- Authoring tools
  - Frontpage – Good for beginners
  - Dreamweaver – Fairly complex but powerful
  - NVU – Free and available for (Windows, Linux, Mac)
- List of editors can be found at:
  http://dir.yahoo.com/Computers_and_Internet/Software/Internet/World_Wide_Web/HTML_Editors/
- Recommended:
  - Komodo Edit -
Frequently Used Tags

- `<head> </head>`
  - It does not generate displayed contents
  - Contains other tags (e.g., `<title> </title>`
- `<title> </title>`
  - Part of the header
  - It is required
  - Search engines depend on it, so use meaningful titles
Frequently Used Tags

- Heading tags
  - `<h1> text </h1>`
  - `<h2> text </h2>` … and so on until
  - `<h6> text </h6>`
  - Higher numbers imply smaller headers
- Paragraph tag
  - `<p> paragraph </p>`
- Code – Use to define computer code
  - `<code> </code>`
- Horizontal Line - `<hr />`
Frequently Used Tags

- Emphasis
  - `<em> text here </em>` Text usually rendered in italics
  - `<strong> text here </strong>` Text usually rendered in bold
- Super/Sub script
  - `<sub> text here </sub>`
  - `<sup> text here </sup>`
- Quotations
  - `<q> quote here </q>`
- Line Breaks
  - `<br />`
- Verbatim (text displayed exactly as it appears)
  - `<pre> text here </pre>`
- **Example:** HtmlIDoc.html
Lists

- Unordered lists
  - `<ul> </ul>` tags to represent beginning and end
  - `<li> </li>` to represent elements in the list
  - **Example: Lists.html**

- Ordered lists
  - `<ol> </ol>` tags to mark beginning and end
  - `<li> </li>` to represent elements in the list

- Definition lists
  - Consist of terms and definitions like in a glossary
  - Tags - `<dl> </dl>`
  - Terms specified using `<dt> </dt>` and definitions with `<dd> </dd>`
  - **Example: DefNestedLists.html**

- Nested lists (See previous example)
Image Inclusion

- We can include an image using the img tag
  &lt;img src="testudo.jpg" alt="Testudos' image" /&gt;

- **Example: Image.html**

- Although the width and height attributes are not required they are highly recommended. (They can also be set through CSS).
Links

- Link – connection between web resources
- Hypertext links are created using the <a> (anchor) tag
- The link can be text:
  - Notice that you need to specify the protocol (http://)
  - **Example: Link.html**
- The URL can be absolute or relative
- The link can be an image:
  - `<a href="http://www.umd.edu"><img src="testudo.jpg" alt="Testudos' image" /></a>`
Tables

- To define a table we use the `<table>` tag
  - Border attribute controls table’s border
  - By default borders are not visible
- The following tags are associated with tables
  - `<tr>` - defines a row
  - `<td>` - defines a data element
  - `<th>` - define a header data element
  - `<caption>` - provides a caption for the table
    - Must appear after the `<table>` tag
    - Must be used only once
- Example: Table.html
Character Entity References

- Special Characters can be specified by
  - Name specification - &name
  - Numeric specification - &##xxx
- Commonly used characters
  
  \[\text{Copyright} \quad \text{Registered Trademark} \quad \& \quad \&lt; \quad \&gt; \quad \text{Non break space}\]

- Example: CharacterReferences.html
- Complete list at:
  \[\text{http://www.w3.org/TR/html4/sgml/entities.html}\]
Block Elements/Inline Elements

- **Comparison**
  - Block elements begin on new lines whereas inline elements don’t
  - Block elements create larger structures (allow you to define the large structure of your document) whereas inline elements don’t

- **Block Elements Examples**
  - Paragraphs (<p>), Headings, Lists, Tables, Division (<div>), Block Quotations, Preformatted Text (<pre>)

- **Inline Elements Examples**
  - Anchors (<a>), Images (<img>), Line Breaks (<br />)

- Block elements may contain other block elements, inline elements, and data. Some block elements may not contain other block elements.

- Inline elements may contain inline elements and data.
Inline Elements in Block Elements

- Why the following example does not validate?
- Example: validationProblem.html
Suggestions for Writing HTML Code

- Add the corresponding end tag immediately
- Use indentation
- Have a consistent style
- Use comments to separate sections of your code.
- Validate your code as you develop it.
**Googles Page Creator**

- You need a gmail account
- Provides free hosting
- Your address will be:
  - [http://YOURGMAILID/googlepages.com](http://YOURGMAILID/googlepages.com)
CSS (Cascading Style Sheets)

- Official W3C standard for controlling presentation
- Specification: [http://www.w3.org/TR/CSS21/](http://www.w3.org/TR/CSS21/)
- Style Sheets
  - Text file with rules. It includes no html.
  - Style sheets files use a .css extension
  - Allows you to apply typographic styles (font size, line spacing, etc.)
  - Allows you to apply spacing instructions
  - Allows you to have page layout control
  - Allows you to generate smaller html files by avoiding redundancy in style specification
  - Allows you to easily update a collection of pages by updating only a single file
- Why CSS? Demo
Rules

- Rule - Basic element of a style sheet
- Rule - describes the formatting associated with a page element
- Rule format

**selector declaration**

**selector** – identifies what should be styled in a web document (e.g., h1, p)

**declaration** – what and how that portion of the web document should be modified.

- declaration - consists of *property: value* pair(s) enclosed in `{ }`
- Examples:

  ```
  h1 {color: green}
  p  {font-size: 10px,
      color: red;
  }
  ```

- Notice there is a space after the colon (`;`)
- Popular properties – color, font-family, font-size, text-decoration
- HTML Dog CSS Properties –
Types of Style Sheets

- **Inline**
  - Style information applied to specific tag (e.g., `<p style=…”)
  - Avoid if possible.

- **Internal**
  - Using the `<style>` tag in the header of the html document
  - Convenient to provide own style to a specific page
  - Example: internalStyle.html

- **External**
  - External style sheet which web pages link to
  - Preferred approach
  - Example: externalFile.html and externalFile.css
CSS

- Why cascading?
  - Rules can come from different sources (inline, external file, etc.). The final set of rules that apply to a document comes from cascading all the sources.

- Rule Conflict Resolution
  - To resolve conflicts, styles defined at a specific level override those set at a higher level
    - Example: you can set the color of body text to be blue but you can override to red the text in a list
  - When multiple style files are linked or imported the last will take precedence

- A child element inherits the same properties of its parent element (unless otherwise specified).
CSS Validator

- [http://jigsaw.w3.org/css-validator/](http://jigsaw.w3.org/css-validator/)

Notice you have three choices
- by URI
- by File Upload
- by direct input
Colors

- You can specify colors using one of the following predefined colors:
  - yellow, white, teal, silver, red, purple, orange, olive, navy, maroon, lime, green, gray, fuchsia, blue, black, aqua
- Source for colors
- You can specify a color by indicating the red, green and blue components. For example, all the following are equivalent:
  - red
  - rgb(255,0,0)
  - #ff0000
Kinds of Selectors

- **Type Selectors** – Those based on the name of an HTML tag
  - p { color: red; }

- **Pseudo-classes** – attached to selectors to specify a state. Four popular pseudo-classes are
  - a:link – initial color of a link
  - a:visited – color for a visited link
  - a:hover – color when mouse hover over link
  - a:active – color during the clicking of the link

- **Example**: selectors.html, selectors.css
Kinds of Selectors

- **Class Selectors** – Allow us to apply the same CSS rule to different elements
  - Use when you need to apply a style to many times in your document
  - Created with a period (also known as full stop)
  - Example: classIdSelectors.html, classIdSelectors.css

- **ID Selectors** – Like class selectors but appear only once in the document
  - Used when you need to apply a style only once in your document
  - Created using #
  - Example: classIdSelectors.html, classIdSelectors.css

- **Others** (will see them later on)
  - Descendant Selectors, child selectors, attribute selectors, universal selectors

- **Example**: selectors.html, selectors.css