Objectives

- Review: Web, HTML
- CSS: Presentation of Web Pages
- Project discussion/planning
Web Review

• What made the WWW possible?
• What are the main applications that enable the Web?
  ➢ What protocol do they use to communicate?
• How does the process of retrieving a page work?
HTML Review

• What is used to markup a document?
  ➢ What are its components?

• What are the two main types of elements?
  ➢ How are they different?

• How do we make...
  ➢ A heading
  ➢ A link
  ➢ An image
  ➢ A table
  ➢ A list
Lab 0

• How did Lab 0 go?
  ➢ Wiki?
  ➢ Validating your page?

• Anything tricky?

• Any questions?
cs.wlu.edu’s Web Server Set Up

• How ~user directs to user’s public_html directory

```html
<IfModule mod_userdir.c>
  #
  # UserDir is disabled by default since it can confirm the presence
  # of a username on the system (depending on home directory
  # permissions).
  #
  # UserDir disable

  #
  # To enable requests to /~user/ to serve the user's public_html
  # directory, remove the "UserDir disable" line above, and uncomment
  # the following line instead:
  #
  #UserDir public_html
  UserDir /home/www/users
</IfModule>
```
cs.wlu.edu’s Web Server Set Up

- How ~user directs to user’s public_html directory

  \texttt{public\_html} -> \texttt{/home/www/users/sprenkle/}
cs.wlu.edu’s Web Server Set Up

• Location of “main” web pages

```
# DocumentRoot: The directory out of which you will serve your
# documents. By default, all requests are taken from this directory, but
# symbolic links and aliases may be used to point to other locations.
# DocumentRoot "/var/www/html"
```
cs.wlu.edu’s Web Server Set Up

- Why when you go to a directory in the browser, you see index.html

```
# DirectoryIndex: sets the file that Apache will serve if a directory is requested.
#
# The index.html.var file (a type-map) is used to deliver content-negotiated documents. The MultiViews Option can be used for the same purpose, but it is much slower.
#
DirectoryIndex index.html index.html.var
```
CSS: CASCADING STYLE SHEETS
Presentation of Web Pages

• Talked mostly about structure and content of HTML pages

• Want presentation to be *separate*
  - In general, don’t encode style into the HTML page itself
  - Easier to apply different styles to a set of web pages or a whole web site

Cascading Style Sheets (CSS)

• Describe the **appearance**, **layout**, and **presentation** of information on a web page
  - **How** information is to be displayed, not what is being displayed

• CSS is designed to specify style
  - **HTML** is not

• Can be embedded in HTML document or placed into separate **.css** file
  - Separate **.css** file advantage: one style sheet can be *shared* across many HTML documents
Why *Cascading* Style Sheets?

- **Cascading** because the attributes of an element cascade together in this order:
  - Browser’s default styles
  - external style sheet files (in a `<link>` tag)
  - internal style sheets (inside a `<style>` tag in the page’s header)
  - inline style (the `style` attribute of the HTML element)
Attaching a CSS File: `<link>`

- **link** appears in **head** element
- Can link to multiple style sheet files
  - When > 1 style sheet defines a style for the same HTML element, latter sheet's properties are applied

```
<link rel="stylesheet" type="text/css" href="filename"/>
```

- **Example from W&L site:**

```
<link rel="stylesheet" type="text/css" href="http://www.wlu.edu/prebuilt/v2css/gateway.css">
<link rel="stylesheet" type="text/css" href="http://www.wlu.edu/prebuilt/shadowbox-3.0.3/shadowbox.css">
```

Takes precedence
Basic CSS Rule Syntax

• A CSS file consists of one or more rules
• Each rule starts with a selector that specifies an HTML element
  ➢ Applies style properties to the element
  ➢ Properties have values

```css
selector {
  property: value;
  property: value;
  ...
  property: value;
}

p {
  font-family: sans-serif;
  color: blue;
}
```
What Can You Specify Styles For?

- CSS Categories
  - Colors
  - Fonts
  - Lists
  - Alignment of Text
  - Backgrounds
  - Borders
  - Margins

Provide Overview of Properties
Resources on Wiki

April 26, 2016  Sprenkle - CSCI335
CSS Properties for Colors

• **color**: color of the element’s text
• **background-color**: color that will appear behind the element

```css
p {
  color: red;
  background-color: black;
}
```

This paragraph uses the above style.
Specifying Colors

• Color names recognized by all browsers:
  ➢ aqua, black, blue, fuchsia, gray, green, lime, maroon, navy, olive, purple, red, silver, teal, (white), yellow

• RGB codes: red, green, and blue values from 0 (none) to 255 (full)

• Hex codes: RGB values in base-16 from 00 (0, none) to FF (255, full)
Specifying Colors Examples

- Use Color Names, RGB code, or Hex Code

```css
p { color: red; }
h2 { color: rgb(128, 0, 196); /* purple */ }
h3 { color: #FF8800; /* orange */ }
```

This paragraph uses the first style.
This heading uses the second style.
This heading uses the third style.

- Color references on Wiki Resources page
CSS Comments

- Use /* */ style comments
- CSS (and HTML) are not commented as rigorously as programming language code
- The // single-line comment is NOT supported in CSS

```css
/* CSS Comment. Can span multiple lines. */
p { color: red; }
```
FONTS, TEXT
CSS Properties for Fonts

<table>
<thead>
<tr>
<th>Property</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>font-family</td>
<td>which font will be used</td>
</tr>
<tr>
<td>font-size</td>
<td>how large the letters will be drawn</td>
</tr>
<tr>
<td>font-style</td>
<td>used to enable/disable italic style</td>
</tr>
<tr>
<td>font-weight</td>
<td>used to enable/disable bold style</td>
</tr>
</tbody>
</table>
font-family

• Examples:

```css
p { font-family: "Georgia"; }
h2 { font-family: "Arial Narrow"; }
```

• Multi-name font names should be in quotes

This paragraph uses the first style.
This heading uses the second style.
font-family

• Can specify multiple font names from highest to lowest priority
  ➢ Use generic font name last

p { font-family: "Garamond", "Times New Roman", serif; }

This paragraph uses the above style.

• Generic font names:
  ➢ serif, sans-serif, cursive, fantasy, monospace
  ➢ Keywords, so no quotation marks

In Times New Roman b/c Garamond not installed
Possible Values for font-size

• Vague font sizes: xx-small, x-small, small, medium, large, x-large, xx-large

• Relative font sizes: smaller, larger

• Percentage font sizes, e.g., 90% or 120%

• Units: pixels (px), points (pt), m-size (em), x-height (ex)

  ➢ 16px, 16pt, 1.16em, 1.16ex (no spaces)

```css
p { font-size: large; }
```

This paragraph uses the above style.
em

- Defines the *proportion* of the letter width and height with respect to the point size of the current font
  - Scalable measurement
- Originally derived from the width of the capital "M" in a particular typeface
- Not defined in terms of any specific typeface
  - Same for all fonts at a given point size
  - Example: 1 em in a 16 point typeface = 16 points
- Not an acronym or initialism and is pronounced the same as the letter it refers to, the letter "M"
- ex is similar but the height of the lower-case x

font-weight and font-style

• Either can be set to **normal** to turn them off
  ➢ Such as for heading tags

```html
p {  font-weight: bold;
     font-style: italic;
 }
```

*This paragraph uses the above style.*
**body** Style

- Apply a style to the `body` element to apply a style to the entire body of your page
- Advantage: don’t need to apply a style to each element

```css
body {
  color: #666666;
  font-size: 14px;
}
```

**Example: Course Web page**
W3C CSS Validator

- jigsaw.w3.org/css-validator/
  - Or use WebDeveloper Tool

- Checks your CSS to make sure it meets the official CSS specifications
  - May need to change the CSS version to CSS3
    - Default seems to be CSS2.1

- More picky than the web browser, which may render malformed CSS correctly
Practice Problem: Simpsons

• Add a style sheet to the page
• Entire page should have a Simpsons-yellow background and use 14 pt font
• Main heading should use “Comic Sans MS” font
• Lists should appear in “Lucida Console” font
• Link text should be red
• List bullets should have a blue background
• List items should have a green background
Why `<em>` and `<strong>`, not `<i>` and `<b>`?
Why `<em>` and `<strong>`, not `<i>` and `<b>`?

- **strong** and **em** describe attributes of the content
  - “This is something important in the document.”
- **b** and **i** describe formatting and presentation
  - “I want this to be bold.”
- Add style to **strong** and **em** to do something other than bold or italics
  - What would this do?

```css
strong { font-weight: normal; color: red; }
em { font-style: normal; color: #ff00ff; }
```
## CSS Text Properties Subset

<table>
<thead>
<tr>
<th>Property</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>text-align</strong></td>
<td>Alignment of text within its element, e.g., left, right, center, or justify</td>
</tr>
<tr>
<td><strong>text-decoration</strong></td>
<td>Decorations, such as underline, line-through, blink Can be combined</td>
</tr>
<tr>
<td><strong>line-height, word-spacing, letter-spacing</strong></td>
<td>Gaps between the various portions of text</td>
</tr>
<tr>
<td><strong>text-indent</strong></td>
<td>Indents the first letter of each paragraph</td>
</tr>
</tbody>
</table>
CSS Properties for Dimensions

- **width, height:**
  - How wide or tall to make this element
  - Specified as percentage of frame or in pixels

- **max-width, max-height, min-width, min-height:**
  - Maximum or minimum size of this element in the given dimension
Grouping Styles

- A style can select multiple elements separated by commas
- The given properties will be applied to all of the elements

```css
p, h1, h2 { color: blue; }
h2 { background-color: yellow; }
```

This paragraph uses the above style.

This heading uses the above style.

- Individual elements can also have their own styles (like `h2` above)
Document Tree

• HTML document’s elements can be viewed as a tree

```html
<html>
  <head><title>My Web Page</title></head>
  <body>
    <h1>My Web Page</h1>
    <p>My Favorite Movies: </p>
    <ul>
      <li>Tombstone</li>
      <li>The Muppet Movie</li>
    </ul>
  </body>
</html>
```
Document Tree

• HTML document’s elements can be viewed as a tree

```html
<html>
  <head><title>My Web Page</title></head>
  <body>
    <h1>My Web Page</h1>
    <p>My Favorite Movies: </p>
    <ul>
      <li>Tombstone</li>
      <li>The Muppet Movie</li>
    </ul>
  </body>
</html>
```
Inheriting Styles

• Elements inherit their parents’ styles
• A more tightly matching rule can override a more general inherited rule
• Not all properties are inherited
  ➢ Example: Borders are not inherited
  ➢ Some have default, overriding styles
Simpsons CSS Practice

• All headings should be centered, bolded
• Images should take up 1/3 of the width of the screen
• List items should only take up 1/2 of the width of the screen
• The text should be spaced so that the lines are further apart
• Links should be slightly larger than the other text on the page
CSS Classes

• Selectively apply a CSS rule to only elements of a specific class
  ➢ Give a style to some occurrences of an element

• From course schedule page:
  ➢ Set the background color for a row in the table, if its class is “even”

```html
tr.even { background: #D8DFE7; }

<table>
  <tr class="even"><td>…</td></tr>
  <tr class="odd"><td>…</td></tr>
</table>
```
CSS Class Selector Without Element

- Selectively applies a style to any element that is part of the class

```
.smallCaps { font-variant: small-caps; }
```

```html
<h2 class="smallCaps">Heading 2</h2>
<p class="smallCaps">Paragraph Example</p>
```
CSS ID Selectors

• Selectively applies a CSS rule to only the elements that have a particular id
• Differs from class selector in that an id can only be used once in the HTML document
  ➢ Page won’t validate otherwise
• HTML element can be omitted
  ➢ Rule will apply to any element with given ID

```html
element#id { ... }
```
# CSS ID Selectors

- **Course Web Page Example:**

  ```html
  #sidebar {
    color: rgb(117,144,174));
    background-color: transparent;
    width: 8em;
    padding: 1ex 0;
    border: 1px solid rgb(204,204,204);
    position: absolute;
    left: 4px;
    top: 141px;
  }
  
  <div id="sidebar"><!-- sidebar --></div>
  ```
Logical Divisions in HTML: `<div>`

- Denotes a section or division of an HTML document (block-level)
- Has no on-screen appearance
- Can apply a style or id to it
  - Inherited by all elements inside the `div`
- Powerful for layouts, presentation
Inline Styling Sections: `<span>`

- Has no onscreen appearance
- Can apply a style or ID to it
  - applied to the text inside the `span`

```html
<p>Here is some text in <span class="smallCaps">Small Caps</span>. </p>
```

Here is some text in **SMALL CAPS**.
Grouping Tags

• Can group together some elements and give them a style
• Similar to use of `div` tag but for specific type of elements
• Example: `colgroup`
  ➢ Groups together columns with same style

• More grouping tags on Thursday...
Embedding Style Sheets: `<style>`

- Placed within a page’s `head` element
- Preferred: linking to an external style sheet
  - Especially when many styles

```html
<head>
<style type="text/css">
<!--
/* hide from browsers that can’t handle */
p { font-family: sans-serif }
h2 { color: red }
-->
</style>
</head>
```
Inline Styles with the **style** Attribute

- Higher precedence than embedded or linked styles
- Useful for one-time overrides

```html
<p style="font-family: sans-serif; color: red;"> This is a red paragraph. </p>
```
Practice Problem

• Modify the Simpsons’ CSS and HTML so that the second list item belongs to the “even” class
• An element in the “even” class has a gray background
## CSS Background Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Meaning/Values</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>background-color</code></td>
<td>Color to fill background</td>
</tr>
<tr>
<td><code>background-image</code></td>
<td>Image to place in background</td>
</tr>
<tr>
<td><code>background-position</code></td>
<td>Placement of bg image within an element</td>
</tr>
<tr>
<td><code>background-repeat</code></td>
<td>Whether/how bg image should be repeated; values=<code>repeat</code> (default), <code>repeat-x</code>, <code>repeat-y</code>, or <code>no-repeat</code></td>
</tr>
<tr>
<td><code>background-attachment</code></td>
<td>Whether bg image scrolls within the page</td>
</tr>
<tr>
<td><code>background</code></td>
<td>Shorthand to set all background properties</td>
</tr>
</tbody>
</table>
Advanced Selection

• Applies given properties to `selector2` only if it is *inside* a `selector1` on the page

```
selector1 selector2 { 
  properties
}
```

• Applies given properties to `selector2` only if `selector1` is *directly* inside `selector2`

  ➢ no intermediate tags

```
selector1 > selector2 { 
  properties
}
```
# Pseudo Classes

<table>
<thead>
<tr>
<th>Class Name</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>active</td>
<td>An activated or selected element</td>
</tr>
<tr>
<td>focus</td>
<td>An element that has the keyboard focus</td>
</tr>
<tr>
<td>hover</td>
<td>An element that has the mouse over it</td>
</tr>
<tr>
<td>link</td>
<td>A link that has not been visited</td>
</tr>
<tr>
<td>visited</td>
<td>A link that has already been visited</td>
</tr>
<tr>
<td>first-child</td>
<td>An element that is the first child of another</td>
</tr>
</tbody>
</table>
Pseudo Classes

• Example uses:

```css
a:link {color:#ff0000;} /* unvisited link */
a:visited {color: #00FF00} /* visited link */
a:hover {color: #FF00FF} /* mouse over link*/
a:active {color: #0000FF} /* selected link */
```

Modify so that unvisited links are blue, but only if they’re within a `paragraph` inside of the `div` with id `sidebar`

• Course Web page Example
Other Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Meaning, Values</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>list-style-type</strong></td>
<td>Use with <code>ol</code> or <code>ul</code>. Some possible values: <code>none</code>, <code>decimal</code>, <code>upper-roman</code>, <code>lower-alpha</code>, <code>square</code>, ...</td>
</tr>
<tr>
<td><strong>display</strong></td>
<td>Sets the type of CSS box model an element is displayed with. Values: <code>none</code>, <code>inline</code>, <code>block</code>, <code>run-in</code>, <code>compact</code>, ... Use sparingly--can radically alter page layout</td>
</tr>
<tr>
<td><strong>visibility</strong></td>
<td>Sets whether an element should be shown onscreen. Element will still take up space onscreen but will not be shown; to make it not take up any space, set <code>display</code> to <code>none</code> instead. Values: <code>visible</code> (default) or <code>hidden</code>. Can be used to show/hide dynamic HTML content on the page in response to events</td>
</tr>
</tbody>
</table>
LAYOUT USING BOX MODEL
Layout Using CSS: Box Model

- For layout, every element is composed of:
  - element's content
  - border around the element
  - padding between the content area (inside)
  - margin between border and other content (outside)

- width = content width + L/R padding + L/R border + L/R margin

- height = content height + T/B padding + T/B border + T/B margin
  - IE6 doesn't implement these correctly
Border Properties

• Use **border** property to set borders on all 4 sides

• Properties specified in this order:

<table>
<thead>
<tr>
<th>thickness</th>
<th>specified in px, pt, em, %, or a general widths: thin, medium, thick</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>style</strong></td>
<td>One of none, hidden, dotted, dashed, double, groove, inset, outset, ridge, solid</td>
</tr>
<tr>
<td><strong>color</strong></td>
<td>specified as seen previously for text and background colors</td>
</tr>
</tbody>
</table>
Border Properties

• Use **border** property to set borders on all 4 sides
  
  \[
  \text{border: } \text{<thickness>} \ \text{<style>} \ \text{<color>};
  \]

• To set specific properties of border on all 4 sides:
  
  \[
  \text{border-color, border-width, border-style}
  \]

• All properties of a border on a particular side:
  
  \[
  \text{border-bottom, border-left, border-right, border-top}
  \]

• A specific property on a particular side:
  
  \[
  \text{E.g., border-bottom-color, border-bottom-style, border-bottom-width}
  \]
Border Example

h1, h2 {
  font-family: sans-serif;
  color: gray;
  border-bottom: 1px solid black;
}

Unlike underline, border extends to edge of element’s width
Padding

- **padding**: padding on all 4 sides
  - If one value: all 4 sides
  - 2 values: top/bottom right/left
  - 3 values: top right/left bottom
  - 4 values: top right bottom left

- **padding-bottom**: padding on bottom side only

- **padding-left**: padding on left side only

- **padding-right**: padding on right side only

- **padding-top**: padding on top side only

You may have TRouBLe remembering the order at first
Padding Example

```
p { padding: 20px; border: 3px solid black; }
h2 { padding: 0px; background-color: yellow; }
```

This is the first paragraph

This is the second paragraph

This is a heading

Padding shares the element’s background color
Padding Example

Can set padding for each side separately:

```css
p {
  padding-left: 200px;
  padding-top: 30px;
  background-color: fuchsia;
}

This is the first paragraph

This is the second paragraph
Margins

- **margin**: margin on all 4 sides
  - If one value: all 4 sides
  - 2 values: top/bottom right/left
  - 3 values: top right/left bottom
  - 4 values: top right bottom left

- **margin-bottom**: margin on bottom side only

- **margin-left**: margin on left side only

- **margin-right**: margin on right side only

- **margin-top**: margin on top side only
Margin Example

```css
p {
  margin: 70px;
  background-color: fuchsia;
}
```

This is the first paragraph

Margin: Space between elements

This is the second paragraph
Margin Example

```css
p { margin-left: 200px;
    background-color: fuchsia;
}
```

This is the first paragraph
This is the second paragraph
FLOAT & CLEAR
**float Property**

- **float** can have values *left*, *right*, or *none* (default)
- Floating elements are removed from normal document flow
- Underlying text wraps around floating element as necessary
- Usually has a **width** property
  - Otherwise, default is 100% width
  - Other text can’t wrap around
Practice Problem

• Make Simpsons image float to the right and text wraps around
It was the best of times, it was the worst of times, it was the age of wisdom, it was the age of foolishness, it was the epoch of belief, it was the epoch of incredulity, it was the season of Light, it was the season of Darkness, it was the spring of hope, it was the winter of despair, we had everything before us, we had nothing before us, we were all going direct to heaven, we were all going direct the other way - in short, the period was so far like the present period, that some of its noisiest authorities insisted on its being received, for good or for evil, in the superlative degree of comparison only.
clear Property

• Disallows any floating elements from overlapping this element
  ➢ This element will start “below” floating elements

• clear can be left, right, both, or none (default)
POSITIONING
### position Property

<table>
<thead>
<tr>
<th>Property</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>static</strong></td>
<td>default position</td>
</tr>
<tr>
<td><strong>relative</strong></td>
<td>offset from its normal static position, relative to block element that contains it</td>
</tr>
<tr>
<td><strong>absolute</strong></td>
<td>at a fixed position <em>within its containing element</em></td>
</tr>
<tr>
<td><strong>fixed</strong></td>
<td>at a fixed position <em>within the browser window</em></td>
</tr>
</tbody>
</table>
**fixed Position**

- At a fixed position *within the browser window*
- *top, bottom, left, right* properties specify positions of box's corners
  - Can be negative to create an element that sits outside the visible browser window
Those Annoying Ads: \texttt{z-index}

- Sets which absolute positioned element will appear on top of another that occupies the same space
- Higher \texttt{z-index} wins
- Can be \texttt{auto} (default) or a number
Using WebDeveloper

- Using Outlines
- View CSS Style Information
Bootstrap

- “most popular HTML, CSS, and JS framework for developing responsive, mobile first projects on the web”
- Free front-end framework for faster and easier web development
- Includes HTML and CSS based design templates for typography, forms, buttons, ...
  - optional JavaScript plugins
- Easily create responsive designs

http://getbootstrap.com/
http://www.w3schools.com/bootstrap/
TODO

• Lab 1: CSS
  ➢ Practice using plugins
  ➢ Create your own home page

• Readings/Summaries on Sakai forums

• Project – more on Thursday