Web Site Design

Stanford University Continuing Studies CS 03

Mark Branom
branom@alumni.stanford.edu
http://web.stanford.edu/people/markb/

Course Web Site: http://web.stanford.edu/group/csp/cs03

Week 5 Agenda

- Unfinished business
- Cascading Style Sheets

CSS Review – why called CSS?

- Cascading
  - Rules are interpreted in a top-to-bottom, cascading fashion.
  - Rules that aren't conflicting are aggregated (e.g., `body { background: red; color: white; }` produces a background of red with a font color of white.
  - For rules that conflict, last one generally "wins" (e.g., `{background: red; background: white}` produces a background of white, not red.
- Style
  - Refers to the look and feel
- Sheet
  - Set of rules
  - External, Internal, or Inline

CSS Review: Sheet Types (External)

- External
  - Rules are listed in a separate text file (e.g., style.css)
  - Web pages are attached to the style using the <link> or the @import method:

```
<link href="URL-to-style-sheet-file.css"
rel="stylesheet" type="text/css"
media="all | print | only screen and
(max-width: 640px)">

<style type="text/css" media="all | print | only
screen and (max-width: 640px)">
@import url("URL-to-style-sheet-file.css")
</style>
```

CSS Review: Sheet Types (Internal)

- Internal
  - Rules are listed in the web page itself (usually in the <head>)

```
<head>
…
<style type="text/css">
   body { background: pink;}
</style>
…
</head>
```

CSS Review: Sheet Types (Inline)

- Inline
  - Rules are listed in the element (“tag”) itself

```
<p style="text-align: center;">
     paragraph that's centered
</p>
```

  - Generally used only in email newsletters and computer scripts

CSS Review: CSS Rules

- Rules are defined as a selector and declaration:

- selector { property: value; }

$\uparrow$

“declaration”

Comments

- CSS Comments:

```
/* Enter comment here */

/****************
Main section
***************/
```

- HTML Comments:

```
<!-- enter comment here -->
```

CSS Selectors: Redefining Elements

- You can use CSS to redefine an element ("tag"):

```
body { background: pink;}

p { font-family: Verdana;}
```

CSS Selectors: Create Own Style (Class/ID)

- You can use CSS to create your own style:

```
.class { property: value; }

#id { property: value;}
```

```
 .highlight { background: #eee; }
 #headerarea { border: thin dotted #000; }
```

CSS Selectors: "Advanced" "Stuff": Multiple Styles/Selectors

- Multiple styles – Each rule can include multiple styles, using semicolons to separate them.
  - `h2 {color: darkblue; font-style: italic;}`
- Multiple selectors that have the same styles can be grouped, using commas to separate them.
  - `h1, h2, h3 {font-style: italic;}`

CSS Selectors: “Advanced” “Stuff”: Contextual Selectors

- Contextual selectors allow you to specify that something will occur only when it is used in conjunction with something else.
- In the style below, em will display in red, but only when it occurs within li within ul:
  - `ul li em {color: red;}`
- Elements being modified by contextual selectors need not appear immediately inside one another. For example, using the style above with the HTML below, blah would still be red text:
  - `<ul><li><strong><em> blah </em></strong></li></ul>`

CSS Selectors: "Advanced" "Stuff": Direct Child Selectors

- Direct child selectors allow you to specify that something will change, but only when immediately inside another element.
- With the style below, only those strong elements that are directly inside h1 will be purple font; no strong tags deeper within the sheet will be purple:
  - `h1 > strong {color: purple;}`

CSS Selectors: “Advanced” “Stuff”: Adjacent Selectors

- Adjacent selectors allow you to specify that something will change only when preceded by something else.

  In the style below, only those links that are preceded by an h2 will be green:
  - `h2 + a {color: green;}`
- Elements being modified by adjacent selectors appear immediately after one another.
- Using the style above, this link would be green:

  -
    ```
    <h2>Visit Stanford!</h2>
    <a href="http://www.stanford.edu">click here</a>.
    ```
- But this link would not:

  -
    ```
    <h2>Visit Stanford!
    <a href="http://www.stanford.edu">click here</a></h2>.
    ```

CSS and Fonts

- Some of the things you can do to change fonts in CSS:
  - font-family
    ```
    p { font-family: Verdana;}
    ```
  - font-style
    ```
    h1 { font-style: italic;}
    ```
  - font-weight
    ```
    .important { font-weight: bolder;}
    ```
  - font-size
    ```
    body { font-size: large;}
    ```
  - font-variant
    ```
    em { font-variant: small-caps;}
    ```
  - text-shadow
    ```
    .cooltext { text-shadow: #f00 3px 3px 5px; }
    ```

Colors/Backgrounds

- Colors:
  - 6 digit hex code: `#ff0000`
  - 3 digit hex code: `#f00`
  - rgb method (integer/percentage): `rgb(255,0,0)` or `rgb(100%,0%,0%)`
  - keyword color name: `red`
- Backgrounds:
  - `body { background-color: #f00; }`
  - `table { background-image: url("http://web.stanford.edu/csp/cs03/images/bg.gif"); }`

Span/Div

- The <span> tag is an inline element and is used to apply style to a small amount of content.
  - `Only the word <span class="hightlight"> highlight</span> is highlit.`

- The <div> tag is a block element and is used to apply style to a large amount of content (e.g., regions of a page).

  -
    ```
    <div id="footerarea">
      [ footer content ]
    </div>
    ```

Pseudo-elements, Pseudo-classes, and Generated Content

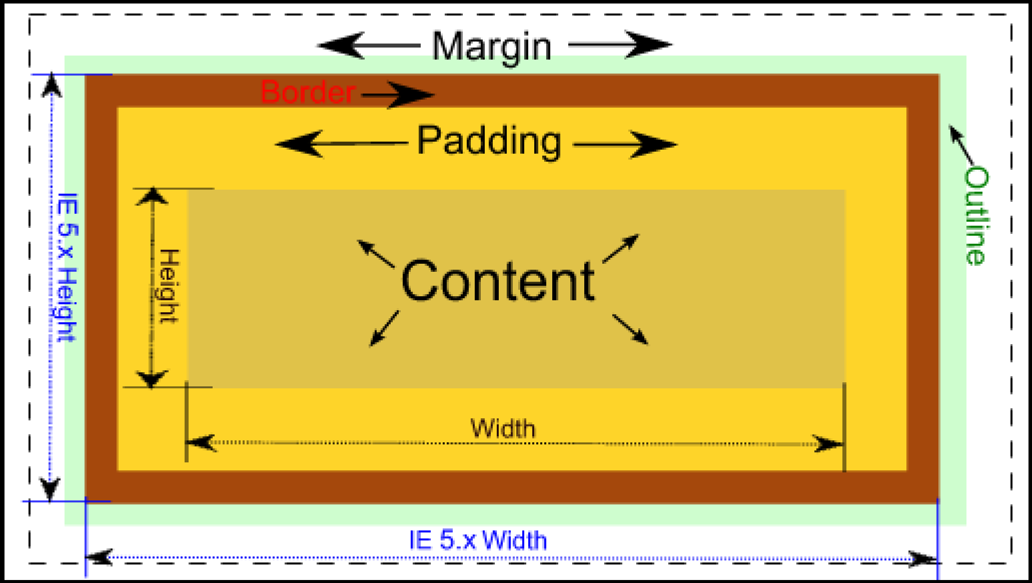
- Pseudo-elements:
  - To affect the first letter, use the style `:first-letter`
  - To affect the first line, use the style `:first-line`
- Pseudo-classes:
  - To affect a link, use the style a`:link`
  - To affect a visited link, use the style a`:visited`
  - To affect an active link, use the style a`:active`
  - To affect a hovered link, use the style a`:hover`
- Generated Content:
  - To make content appear before an element, use the style :before with the property content

    ```
    a.msw:before { content: url('msword.gif'); }
    ```
  - To make content appear after an element, use the style :after with the property content

    ```
    a.acrobat:after { content: url('adobeacrobat.gif'); }
    ```

Box model

- When a browser draws an object on a page, it places it into an invisible rectangular space called a “bounding box.”
- You can specify the size, look, and feel of the margins, the padding, the border, the outline, and the content of that bounding box.



Making content “centered”

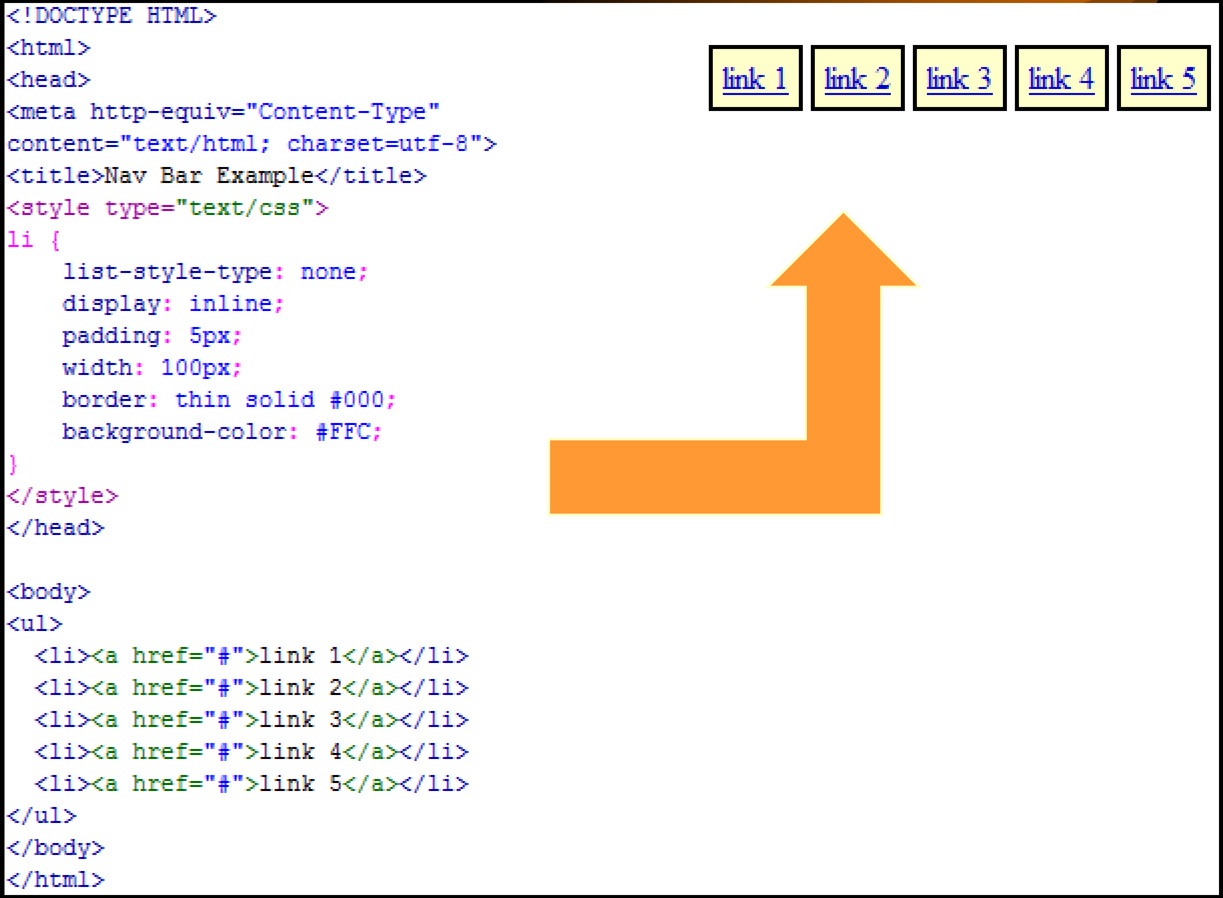
- To make a block centered, you need to first give it a width, and then auto-margin the left and right margins:
  - `#main { width: 80%; margin: 0 auto 0 auto; }`

Block vs. Inline

- Elements are either `block` or `inline`.
- Using CSS, you can change their default status:
  - `em { display: block; }`
  - `p { display: inline; }`
- You can even make them disappear:
  - `.hidden { display: none; }`

Creating a Nav Bar

- To create a nav bar using a list of links and CSS:



```
<!DOCTYPE HTML>
<html>
<head>
<meta http-equiv="Content-Type"
content="text/html; charset=utf-8">
<title>Nav Bar Example</title>
<style type="text/css">
li {
    list-style-type: none;
    display: inline;
    padding: 5px;
    width: 100px;
    border: thin solid #000;
    background-color: #FFC;
}
</style>
</head>

<body>
<ul>
  <li><a href="#">link 1</a></li>
  <li><a href="#">link 2</a></li>
  <li><a href="#">link 3</a></li>
  <li><a href="#">link 4</a></li>
  <li><a href="#">link 5</a></li>
</ul>
</body>
</html>
```

Absolute Positioning

- You can also use CSS to define exactly where you want an object to be located, using the property position with a value of absolute:
- `#headersection { position: absolute; top: 0px; left: 0px; width: 100%; }`

Fixed Positioning

- You can also use CSS to keep content static and never move:
- `#linksarea { position: fixed; top: 20px; left: 0px; width: 200px; height: 400px; }`

Layering

- If two objects are positioned using absolute or fixed positioning, you can layer one on top of the other. To determine which is on top, use the property z-index. The rule with the higher z-index will be on top.

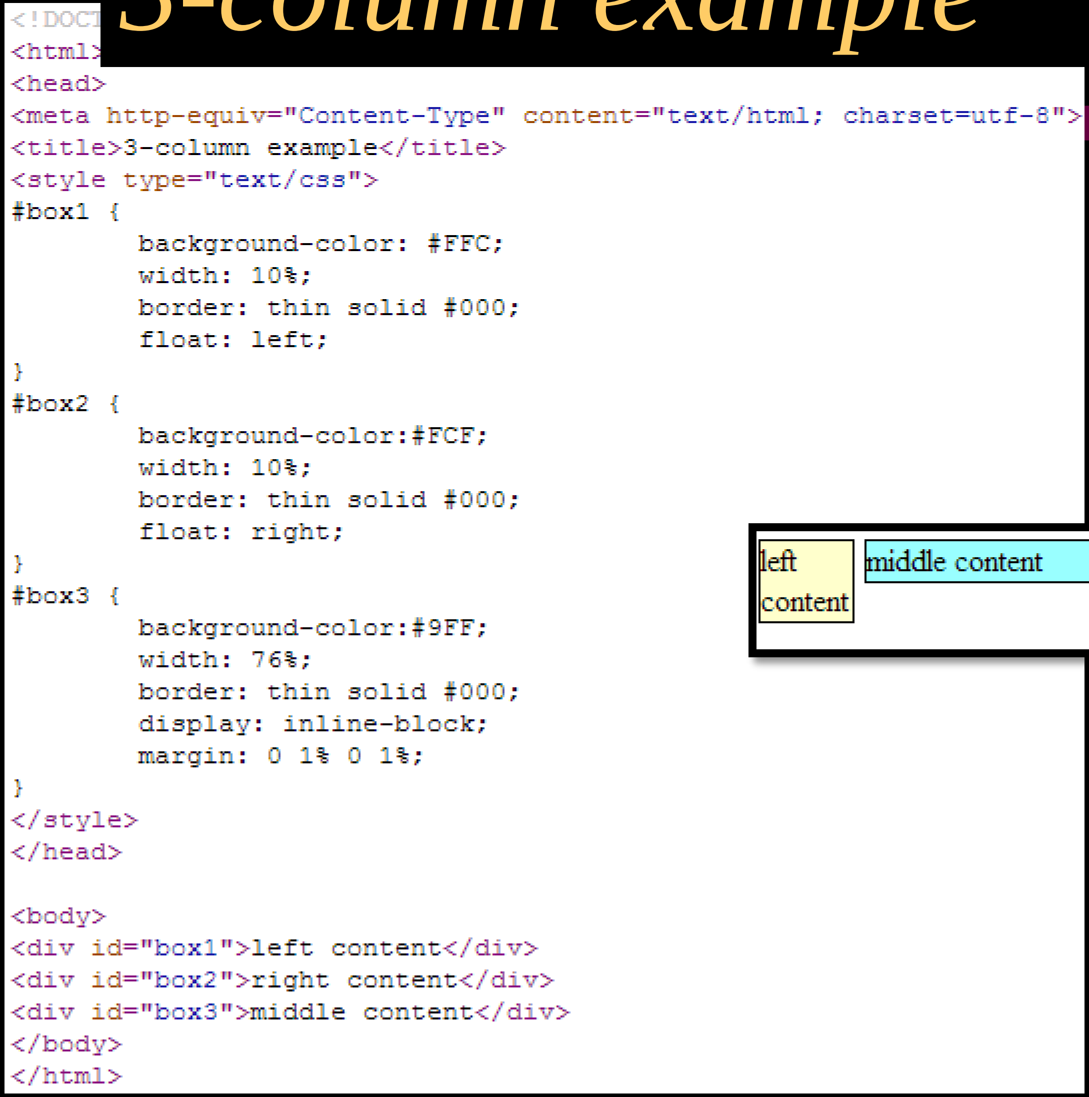
-
  ```
  #box1 { position: absolute;
  bottom: 0px; left: 0px; z-index: 5; }
  #box2 { position: absolute;
  bottom: 3px; left: 3px; z-index: 1; }
  ```

Float/Clear

- You can also have content float to the right or left of another object.
  - `float: left;`
  - `float: right;`

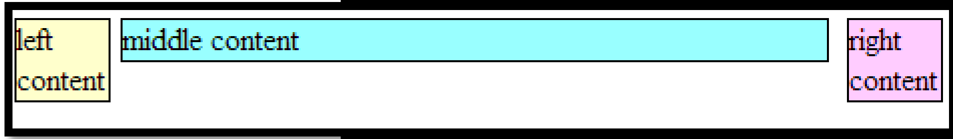
- To end a float, use the clear property:
  - `clear: right;`
  - `clear: left;`
  - `clear: both;`

3-column example



```
<!DOCT
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8">
<title>3-column example</title>
<style type="text/css">
#box1 {
        background-color: #FFC;
        width: 10%;
        border: thin solid #000;
        float: left;
}
#box2 {
        background-color:#FCF;
        width: 10%;
        border: thin solid #000;
        float: right;
}
#box3 {
        background-color:#9FF;
        width: 76%;
        border: thin solid #000;
        display: inline-block;
        margin: 0 1% 0 1%;
}
</style>
</head>

<body>
<div id="box1">left content</div>
<div id="box2">right content</div>
<div id="box3">middle content</div>
</body>
</html>
```



Inline Frames

- Inline frames allow a webmaster to insert more than one web page into a single window.
- They allow one part of the screen stay up and running at all time regardless of what's loaded onto the rest of the page.

The iframe tag

- The tag iframe lets you insert a frame within the regular flow of an HTML document.
- Many websites, including Google (e.g., Gmail & YouTube) use iframes to dynamically display content stored on multiple servers at once.
- Syntax:

  ```
  <iframe src="location.html" width="xx"
  height="yy">Browser can't process frames?
  <a href="location.html">Go to the framed page</a>
  </iframe>
  ```
- An iframe example:

  ```
  <iframe width="560" height="315"
  src="http://www.youtube.com/embed/fpMZbT1tx2o"
  frameborder="0" allowfullscreen></iframe>
  ```

Internal Links

- What if you wanted to let visitors jump around your web page? Suppose you have a long page, and want users to have the ability to jump back to the top of the page or right to the bottom of the page?
- Or suppose you're putting a paper on the web and you want to have footnotes? Web designers can use an internal link to satisfy these types of needs.

- Internal links let visitors be pointed to a specific spot on a web page.

```
<a name="specificspot" id="specificspot"></a>
```

- The text in the NAME/ID attribute can be anything, but using one word is best. Point users to this spot by using the number sign with whatever was used in the NAME/ID attribute.

```
<a href="#specificspot">Go to this location.</a>
```

Meta Tags

- Meta tags are mostly used by webmasters to provide information about their website to search engines.

  ```
  <meta
        name="xxx"
        content="yyy"
        dir="ltr|rtl"
        http-equiv="yyy" >
  ```

name: specifies the type of information

content: sets the content

dir: sets the direction of the text (left to right or right to left)

http-equiv: affects the way the browser & server react to the webpage

Meta tag examples

```
<meta name="description" content="This is where I would put a one-paragraph description of this website" />

<meta name="keywords" content="this, is, where, I, would, put, keywords, related, to, the, website, separated, by, commas" />
```

Client-pull

- A main use of the meta tag is to force the browser to open a new page in a different location (or to refresh an existing page after a set number of seconds).

```
<meta http-equiv="refresh"
content="10; url=newlocation.html" />
```

- You can also create a text file called .htaccess to define and process the redirect.
  - http://en.wikipedia.org/wiki/URL_redirection#Using_.htaccess_for_Redirection

Cache control

- Webmasters also use meta tags to force browsers to show current pages, rather than viewing them from the cache.

```
<meta http-equiv="Expires" content="0" />
```

Causes browser to NEVER view the page in the cache

```
<meta http-equiv="expires" content="Wed, 12
Dec 2012 12:12:12 GMT" />
```

Causes browser to check for new content starting on 12/12/12, otherwise it'll show what's in the cache.

Preventing Search Engines from indexing your site

```
<meta name="robots" content="noindex" />
```

Don't take any info from this page

```
<meta name="robots" content="noindex, nofollow" />
```

Don't take any info from this page, and don't go looking at any links

```
<meta name="robots" content="noindex, follow" />
```

Don't take any info from this page, but go ahead and index links.

You can also create a text file called robots.txt specifying your exclusion desires.

http://en.wikipedia.org/wiki/Robots.txt

Forcing handheld devices (phones/tablets) to use their real size

- By default, handheld devices “shrink” a processed web page to fit the device’s screen. But if you want to make a mobile-friendly version of a page or site, you need to force the device’s browser to use the real width:

  ```
  <meta name="viewport"
  content="width=device-width,
  initial-scale=1.0, user-scalable=yes">
  ```

Content Management Systems (CMS)

- Wiktionary Definition: http://en.wiktionary.org/wiki/content_management_system

  A computer software system for organizing and facilitating collaborative creation of documents and other content, especially for loading to a website

Content Management Systems (continued)

- In a content management system, content pieces (articles, stories, bios, documents, etc.) are stored in a database, while computer scripts automatically display and organize the information presented based on the actions of the visitor to the web site. This means that content producers do NOT need to be experts in programming or HTML; they can focus their attention onto the actual content they are writing.

- Examples :
  – WordPress (http://wordpress.org)
  – Drupal (http://drupal.org)
  – Joomla! (http://joomla.org)

Web Site Hosting

- Options:
  - Internet Service Providers
    - Often will give you a certain amount of disk space to play with on the web (often 20-100 MB)
    - Address would be: http://www.isp.com/userid/
  - Free Web Pages
    - We've been using freezoy.com
    - Others also exist: tripod.com, 50megs.com, etc.

Setting up a domain

- Right now, your web address looks something like: http://yourname.freezoy.com/
- If you'd rather have it look something like http://www.yourname.com, you will need to register your own domain name with one of the Internet's domain registrars.
- Most registrars charge around $10-50/year to register a domain name.
- http://www.internic.net
- http://www.internic.net/alpha.html

Learning More...

- Many local colleges and universities offer a variety of classes on Web topics. Here are some suggestions:

Stanford University's Continuing Studies:
    http://continuingstudies.stanford.edu/

Canada College:
    http://canadacollege.net/

College of San Mateo:
    http://gocsm.net/

Foothill College:
    http://www.foothill.fdha.edu/

Learning More (cont.)

- Books and Websites -- check the resources page; although they can be great resources, books are often "outdated" before they're even published.
- Practice! Most web designers learned how to "do" HTML by looking at the source codes of others and playing around with tags