Information Technologies

Anselm Spoerri PhD (MIT)
SC&I @ Rutgers University
aspoerri@rutgers.edu
anselm.spoerri@gmail.com

Lecture 4 – Overview

CSS

- **Cascade**: Inheritance, Specificity and Location
- Constructing Complex Selectors
- Box Model
- Floating Element | Positioning Elements: Absolutely Positioned Element
- CSS Reset: CSS "Clean Slate", HTML5 Backwards Compatible

CSS3 Key New Features

Exercise 2 Demos

Web Design

- Basic Principles and Steve Krug's Suggestions

Testing & Debugging Web Pages
➔ Check Easy Stuff First

Lectures – Week 4 Content

http://comminfo.rutgers.edu/~aspoerri/Teaching/InfoTech/Lectures.html#week4

Cascade: Inheritance, Specificity and Location.

Inheritance

- **Html = Hierarchical Structure** of the content

Specificity

- The **more specific** the selector, **the stronger** the rule

Location

- Rules that **appear later** have **more weight**

CSS: Cascade

Cascade: Inheritance, Specificity and Location.

Inheritance

- **Html = Hierarchical Structure** of the content

  Elements are contained / appear with other elements (h1 resides inside div).

- **Many properties**, but not all, **inherited by descendants** of elements

  h1 is blue and has red border … blue is inherited but not red border by elements residing inside h1 element.

- **Inherited rules** are considered the **most general of all** and are overruled by any other rule.

Specificity

- The more specific the selector, the stronger the rule

  h1 with class X rule will overrule h1 rule.

- The **id** attribute is considered the **most specific**.

CSS: Cascade

Cascade: Inheritance, Specificity and Location.

Location

- Rules that **appear later** have **more weight**.

  Browser ➔ External CSS ➔ Internal CSS ➔ Inline Style

- You can declare a rule to be more important than others by adding **!important** at the end of the rule.

Summary

In the absence of a rule, many styles are inherited from parent element to child. With two competing rules, the **more specific the rule**, **the more weight** or importance it has – regardless of its location. With two rules of equal specificity, **the one that appears later wins**.

Note: Some CSS properties allow you to specify the URL of another file: if it is a relative URL, then it **needs to relative to the external style sheet**.

CSS: External & Internal Style Sheets

Linking to External Style Sheet

- Place **<link rel="stylesheet" href="mystyle.css"** type="text/css" **/>** inside **<head> tag**.
- Linking to several external style sheets: the **later ones take precedence** over earlier ones (location principle).

Creating Internal Style Sheet

- **<style** type="text/css"**>** … **</style>**

Internal style sheet **overrides external** style sheets **if and only** if style tag comes after link tag.

Applying styles **locally** is **not recommended**.

To view other designer's CSS code: **view source code** of Web page and look at internal style sheet and load URLs for external style sheets.

CSS: Selectors

Selector determines which elements the formatting will be applied to.

Declarations define the formatting.

Constructing Complex Selectors that apply formatting based on:

- **type** or **name** of element (e.g. tag)

  h1 {color: red;}

- **context** element is found

  h1 em {color: red;} **h1.**news {color: red;}

  div#gaudi p = any p element contained in div whose id is equal to gaudi

- **class** (.name) or **id** (#name) of an element

  strong**.news** {color: red;} div**#gaudi** {color: red;}

- **pseudo-class**

  tag:**first-line** tag**:first-letter**

Specifying Groups of Elements: h1, h2 {color: red;}

Combining Selectors

1. Define Context

div#intro

2. Spell out Element's Name

div#intro **p**

3. Specify Class or Id of desired element

div#intro p.**firstP**

4. Specify Pseudo-class or Pseudo-element

div#intro p.firstP**:first-letter** example

CSS Validator

http://jigsaw.w3.org/css-validator/

Recap – CSS: Cascade

Cascade: Inheritance, Specificity and Location.

Inheritance

example

- Html = Hierarchical Structure
- Many properties, but not all, inherited by descendants of elements

Specificity

example

- The more specific the selector, the stronger the rule

How do you create a specific CSS rule?

- **tag** **class** (.name) **id** (#name) ➔ **context**

Location

example

- Rules that appear later have more weight.

➔ **Inherit** from Parent

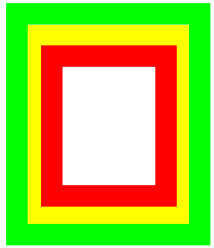
➔ The **More Specific** the rule, the **More Weight**

➔ The one that **Appears Later Wins**.

CSS – Box Model

Box Model = every element is enclosed in **Invisible Box**

Width and **Height** can be specified



Padding = space surrounding content inside of box
(all four sides can be specified separately)

Border : can specify all four borders separately
also specify border-style: type; border-width: n; border-color: color;

Margin = invisible space around border of box
(-top, -bottom,-left, -right, auto)

w3schools: Box Model Demo

box-sizing: border-box; width, border and padding fall within

Overflow : elements are not always contained in their boxes; can be: visible; hidden, scroll; auto (scroll appears when needed).

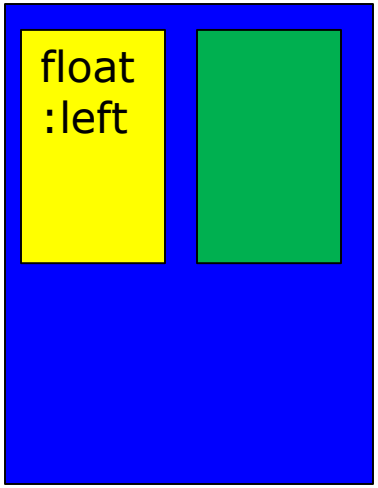
If width, margin, border and padding don't equal size of containing parent element ➔ **something's got to give** :)

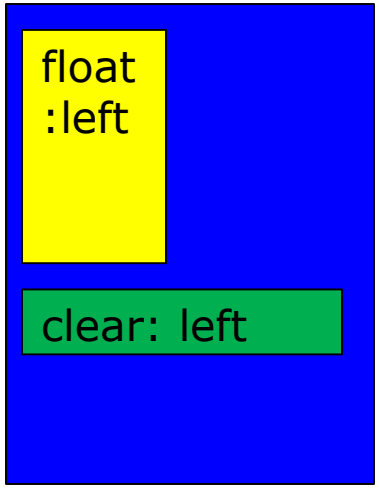
CSS – Floating Elements

Making Elements Float so that they float in a sea of text

- **float: left / right** ➔ element is on left / right
  ➔ text flows on right / left.

- **clear: left / right / both / none** to keep elements from floating on left / right / both sides and none lets elements float on either side.
  - clear property stops affected element from displaying **until designated side is free**.
  - **Add clear property** to elements whose sides you want to be **clear of floating elements**.





http://comminfo.rutgers.edu/~aspoerri/Teaching/InfoTech/Lectures/Lec4/Steps/float_and_clear.html

CSS – AP Element = AP Div

AP Element = **Absolutely Positioned Element** = **AP Div**

- Specify **exact coordinates** with respect to:
  - **Body / AP parent** element (position: **absolute**)
  - **Browser** window (position: **fixed**) [not all browsers support it]

  then set top, right, bottom and/or left: value;

  ➔ takes element out of "natural/normal flow"
- **More Layout Control & Flexibility**
  - **Layout flexibility** like in print design
  - **Change visibility** of AP Elements
- **Nesting AP Elements**
  - Inherit properties from parent AP Element such as visibility
- **AP elements can overlap ➔** specify a **stacking order** (z-index) to position elements in 3D.
- **Vertical-align**: baseline/ middle / sub / super / top / bottom / text-top / text-bottom

CSS – Position and Display Properties

position property of an element

- position: **static**; default | appear in document / linear flow
- position: **relative**; positioned relative to its normal position
- position: **absolute**; positioned relative to 1st non-static parent
- position: **fixed**; relative to browser window

Learn CSS Positioning in 10 Steps

display property of an element

- display: **inline**; Default. Displays element as inline element (like)
- display: **block**; Displays element as block element (like <p>)
- display: **none**; Element will not be displayed (no effect on layout)

CSS – Reset = "Clean Slate" + HTML5 Backwards Compatible

CSS "Clean Slate"

```
for html and html5 tags … {
 margin: 0;  padding: 0;  border: 0;
 font-size: 100%;  font-weight: inherit; vertical-align: baseline;
}
```

HTML5 Reset for older browsers

```
article, aside, details, figcaption, figure, footer, header, hgroup,
   menu, nav, section {
display: block;
}
```

HTML5 shiv for IE < 9 browsers

```
<!--[if lt IE 9]>
<script src="http://html5shiv.googlecode.com/svn/trunk/html5.js">
</script>
<![endif]-->
```

CSS3 – Intro

http://www.w3schools.com/css3/default.asp

CSS3 backwards compatible, don't have to change existing designs.

Borders Rounded, add **shadow** and use **image** as border.

Backgrounds new background properties and **greater control**.

Text Effects new features such as **text-shadow** or **word-wrap**.

Fonts can **use font you like** by including font file on server.

2D Transforms move, **scale**, **turn**, **spin** and **stretch** elements.

3D Transforms format elements using **3D transforms**.

Transitions add effect when changing from one style to another, such as on mouseover or mouseout events.

Animations specify CSS style inside **@keyframes rule** and animation will gradually change from the current style to the new style.

Multiple Columns can create **multiple columns** for laying out text.

User Interface user can **resize** elements and other features supported.

Exercise 2: CSS – Enhance Information Resource with CSS

- Create at least **three pages** (you can use content from Ex1)
- **Develop External Style Sheet**: controls appearance of pages
  - Have **"CSS Reset"** code at start of css file and **HTML5 shiv** in head
  - **DIVs** to control presentation: *pageContent*, *mainContent*, *rightSidebar*
    - Specify float and clear as needed
    - Specify borders, margins and padding as needed
  - **HTML5**: header, nav, main, article, figure, aside, footer
    - Specify float, clear as needed as well as padding etc as needed
  - **Tags**: p, h1, h2, a
  - Use **sans-serif** font + **other Ex2 requirements**

  Task: Design Compact and Visually Appealing Site
- **Layout**
  - **Floating sidebar** on the right
  - **Image floating** on left and **enough text** so that it flows around it
  - **Footer does not float** next to anything
- **Create screencast related to page content**
- **Embed YouTube video and Screencast**

Exercise 2 – Demo Steps

Step-by-Step files:

http://comminfo.rutgers.edu/~aspoerri/Teaching/InfoTech/Lectures/Lec4/Steps/**ex2**

Step 1 – **Link to External CSS file**

Step 2 – **Create DIVs to control presentation**

Step 3 – **Create HTML5 to describe semantics**

Step 4 – **Add Floating Sidebar**

Step 5 – **Non-floating Footer and Floating Image**

Step 6 – **Embed YouTube Video / Screencast**

Step 1 – Link to External CSS file

Step 1

- **Download** from http://comminfo.rutgers.edu/~aspoerri/Teaching/InfoTech/Lectures/Lec4/Steps/**ex2**
  - "lec4.html" and "ex2styles.css"
- **"ex2styles.css" file contains "CSS Reset" code**
  - Place **/*** before "Clean Slate" code and ***/** after it (code goes gray)
- **Cut & Paste** "lec4.html" **Internal CSS Code** into external CSS file
- **Save ex2styles.css**
- **Save "lec4.html" As "ex2_step1.html"** (page loses formatting)
- **Create link to external CSS file** in "ex2_step1.html" page
  - <link rel="stylesheet" href="ex2styles.css" />
  - page formatting is back

Step 2 – Create DIVs to control presentation

Step 2

- **Create DIVs** to control presentation
  - pageContent div already exists
  - **Create DIV with id="mainContent"** is child of pageContent div
    - mainContent div contains main element
- **Create CSS rules for DIVs**

  Specify CSS Internally in <head> and then later move to external file

  <style> </style>
  - CSS rule for id="mainContent"

Step 3 – Create HTML5 to describe semantics

Step 3

- **Create HTML5** to describe semantics
  - HTML5 elements in CSS demo:
    - <header> inside of DIV with id="pageContent"
    - <nav> inside of DIV with id="pageContent"
    - <main> element inside of DIV with id="mainContent"
    - <footer> element inside of DIV with id="pageContent"
  - **Create HTML5 <article> element** inside of main element
    - <article> contains h1, p tags

Step 4 – Add Floating Sidebar

Step 4

- **Create DIV** with **id="rightSidebar"**
  - Place this div in HTML hierarchy as **child of "pageContent" div** and **before "mainContent" div**
- **Create CSS code** for **DIV** with **id="rightSidebar"**

```
float:right;
width:120px;
height:100px;
margin-top:20px;
margin-left:10px;
margin-bottom:10px;
padding:5px;
border-width:thin;
border-style:solid;
border-color:#C1F3BC;
border-top: 20px solid #C1F3BC;
```

- **Create HTML5 <aside> element** in rightSidebar div

Step 5 – Non-floating Footer and Floating Image on Left

Step 5

- **Specify CSS code clear:both for "footer" element**

  clear:both;

  margin-top:10px;

- **Create HTML5 <figure> element and Insert Image**

  Create <figure> element after h1

  Insert image <img src="116.jpg" alt="Intro Image" width="100" height="100">

- **Specify class="floatLeft" and apply to figure** element

  float:left;
  margin-left: 5px;
  margin-right:10px;
  margin-top: 5px;
  margin-bottom:10px;
  border:medium;
  border-style:solid;
  border-color:black;

- **Add enough text** in paragraph so it wraps around image and sidebar

Step 6 – Embed YouTube Video / Screencast

Step 6

- **Create HTML5 <figure> element for Video / Screencast**

  Create <figure> after opening paragraph

- **Insert Table**: **Single Row and Two Cells**
- **Copy & Paste YouTube Embed Code into Cell**

  http://www.youtube.com/watch?v=h9bwDx1Vrm4

  Make sure to set width = 200 and height = 150

- **Copy & Paste Screencast Embed Code into Cell**

  Same steps as for YouTube video

Next Steps

Move Internal CSS code to External CSS file
and **remove comments for "clean slate" code**
and add **HTML tags** and **specify CSS rules that are needed**
and **format page to create your visual look**
Read Ex2 Requirements carefully

Web Design – Basic Principles

Alignment

- **Don't Mix Alignment Styles** – Simplicity and **Left-Aligned**
- Create Sufficient Left Margin
- Constrain Total Width of Page

Proximity

- Related Things Close Together

- Spatial Separation = Conceptual Separation

Repetition & Consistency

- Navigation, Graphics Color Coding, Typeface
- Creates Ease of Use

Contrast

- Bigger, Bolder, Color, Spatial Distance
- Guide the Eye

Design for Scanning, not reading

- **Visual Hierarchy**
  - Visual contrast - size, bold, color
  - Important things = Visually prominent
  - Break pages up into clearly defined areas
  - Related things = Spatially close, Nested
- **Don't mix Alignment Styles**: left-aligned text
- **Avoid "visual noise"**
- **Leverage Conventions**
- **Clear what's clickable**
  - Use underline and/or **color coding**

Make each click a "mindless" choice

Cut ½ of words, then cut ½.

Testing & Debugging Web Pages

- Before looking for a big problem, **check common little problems** :)
- **Work incrementally**
- **Use process of elimination**
  (use comments to make code active / inactive).

- Be careful about **typos**.
- In CSS, not sure if the problem is with the property or the selector, **use a very simple declaration** (color: red).

Check Easy Stuff First - General

- **Refresh browser** so that latest file is shown
- **Upload actual file** and **refresh browser** so that latest file is shown
- **Upload file** in the **correct location**
- Make sure you **save file**
- **Upload any related files**: CSS, images, SWF etc.
- Make sure **spelling of URL = spelling of filename**.
- **Test** in **multiple browsers**
- **Test** on **different computer** than the one used to create the files

Check Easy Stuff First – HTML

HTML

- Make sure you used **correct spelling of tags**
- **Be careful about nesting** and make sure you have **closing tags**
- All elements should have **opening and closing tags**
  (always put space before / for " />" closing tag).
- Include # when specifying hexadecimal colors.

Check Easy Stuff First – CSS

- **Use colon (:)** to separate your properties from value (color: red;).
- **Complete** each **property-value pair** with **semicolon (;)**
- **No spaces** between number and their units (16px).
- **Close brackets**.
- Don’t quote values.
- **Use accepted value**.
- Don’t forget closing **< /style> tag**.
- Make sure **linked HTML document** to the **proper CSS file(s)**.
- **Watch** the **spaces** and **punctuation** between selectors.

Check Easy Stuff First – Testing Your Page

1. **Validate** HTML and CSS.
2. **Open** in **Browser**
3. **Formatting correct**?
4. **Hyperlinks work** & correct?
5. **CSS file referenced properly**?
6. **All images appear**?
   If not, check the easy stuff first, especially spelling of filenames and don't use spaces in filenames and saved as GIF or JPEG.
7. **Upload files to server** (and **set permissions** if needed).
8. **View** pages in **different browsers**.
9. Still Stuck **➔ check for typos and check easy stuff first :)**