CSCB20 – Week 7

Introduction to Database and Web Application Programming

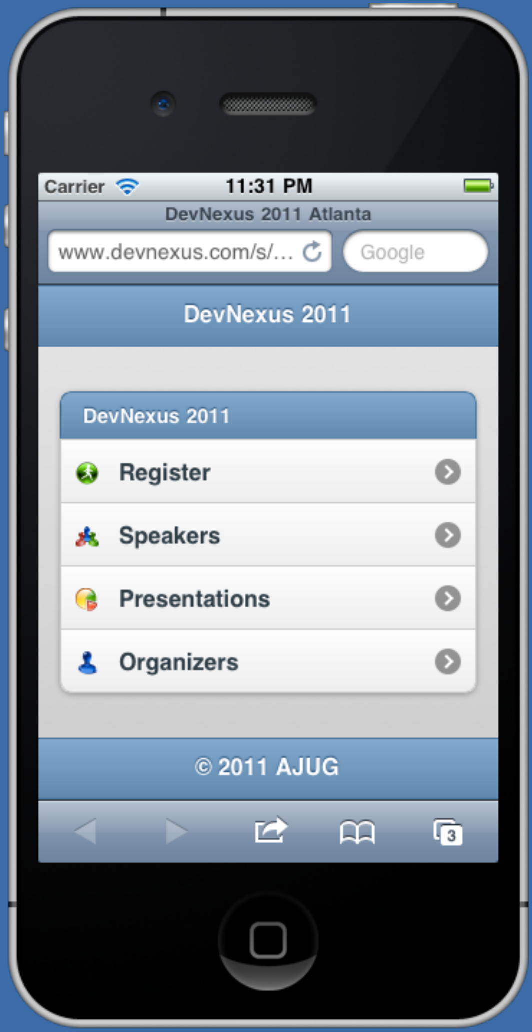
Anna Bretscher
Winter 2017

Cascading Style Sheets (CSS)

Examples of CSS

CSS provides a powerful and still-evolving toolkit of style properties to develop advanced Web apps, e.g.:

- http://andrew-hoyer.com/experiments/rain/
- http://vlog.it/
- http://andrew-hoyer.com/experiments/walking/



Motivation

The “Style” in Cascading Style Sheets (CSS) refers to

- appearance or
- “layout” of a document for viewing,
- is distinct from it’s structure or meaning

How would you design style support into HTML documents?

What are the advantages and disadvantages of your approach?

Separation of Concerns

HTML (and it's cousin XML) are supposed to be about document structure and meaning.

Early in the development of HTML, structure and style were intermixed – special tags to control style.

Q. What's wrong with this approach?

- style is highly localized – difficult to apply changes across entire document or set of documents
- style is not reusable across elements without copying (repeating) style code
- difficult to change style or structure without risking changing the other

Separation of Concerns

Burdening HTML with style details leads to tight coupling between style and structure

- hard to maintain – style information spread across documents, changing style requires numerous consistent html tag changes (error prone)
- difficult to divide responsibility (labour) between designers and document-content creators, who, in the case of dynamic documents, are software developers, not stylists

Style and Document Elements

Want a flexible way to bind style(s) to various subsets of document elements

Desirable Properties?

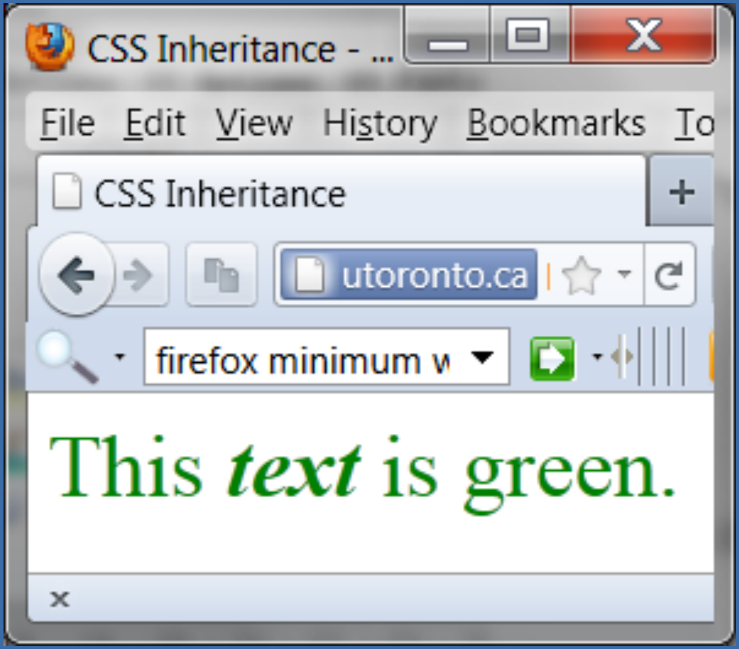
- DRY (don't repeat yourself): should be able to compactly define style to apply to a whole class of elements
- Inheritance: styles of outer elements should be inherited by inner/nested elements
- Structural independence: should be able to apply style to document components where ever they occur
- Context awareness: applied style may vary depending on the structural context in which an element appears

Inheritance

Most styles are inherited into nested elements.

One way to set a "default" document style is by setting style property values for the `<body>` element.

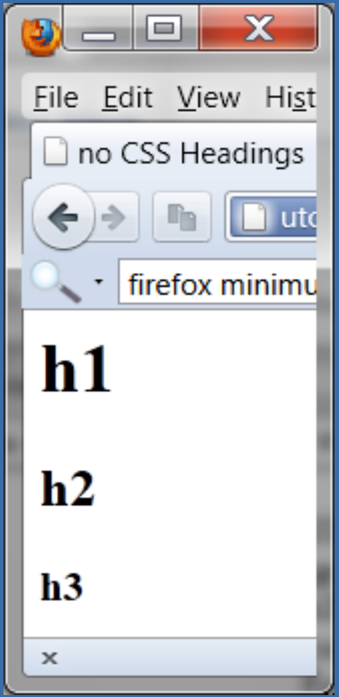
```
<style type="text/css">
    body { color: green;
           font-size: 200%
    }
    em { font-weight: bold }
</style>
   ...
<p>This <em>text</em> is
   green.</p>
```



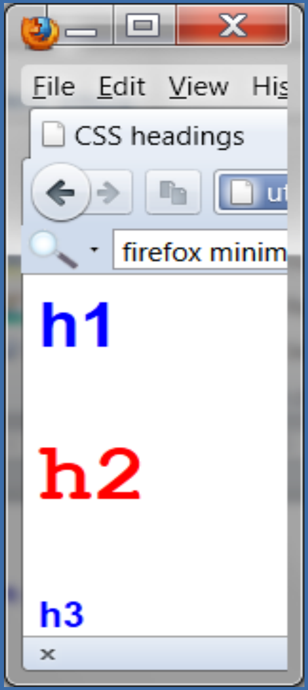
This ***text*** is green.

Grouping as Selector

Multiple comma-separated elements can be grouped, with common style applied to all.



without CSS

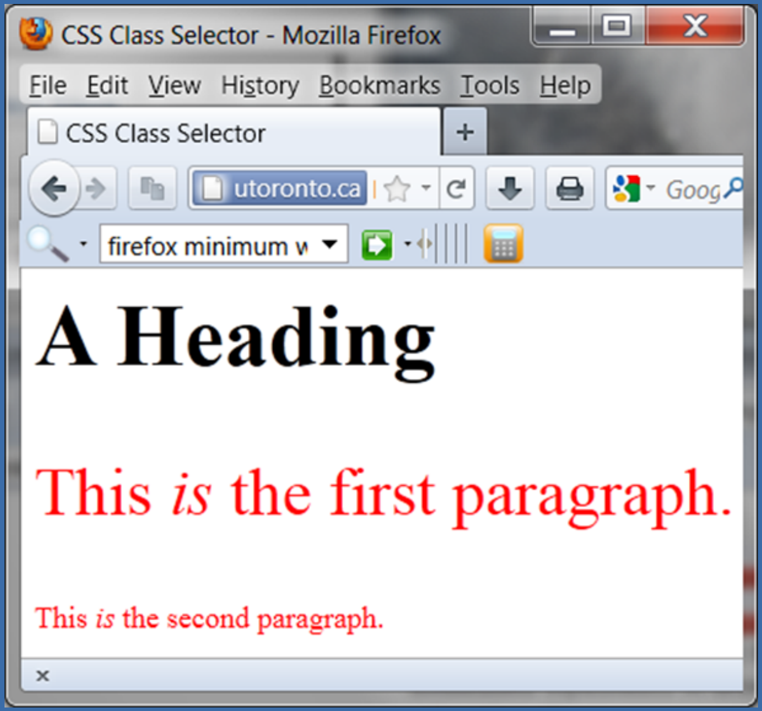


with CSS

```
<style type="text/css">
   h1, h3 { color: blue;
         font-family:
   helvetica
   }
   h2 { font-size: 36pt;
         color: red;
         font-family: courier
   new
   }
</style>
<body>
    <h1>h1</h1>
    <h2>h2</h2>
    <h3>h3</h3>
</body>
```

Class Attribute as Selector

- Special case of attribute selector for HTML documents
- HTML elements can be tagged with possibly many classes
- Style properties can be set for all elements of a given class


CSS Class Selector - Mozilla Firefox
File Edit View History Bookmarks Tools Help
CSS Class Selector
utoronto.ca
Goog
firefox minimum w

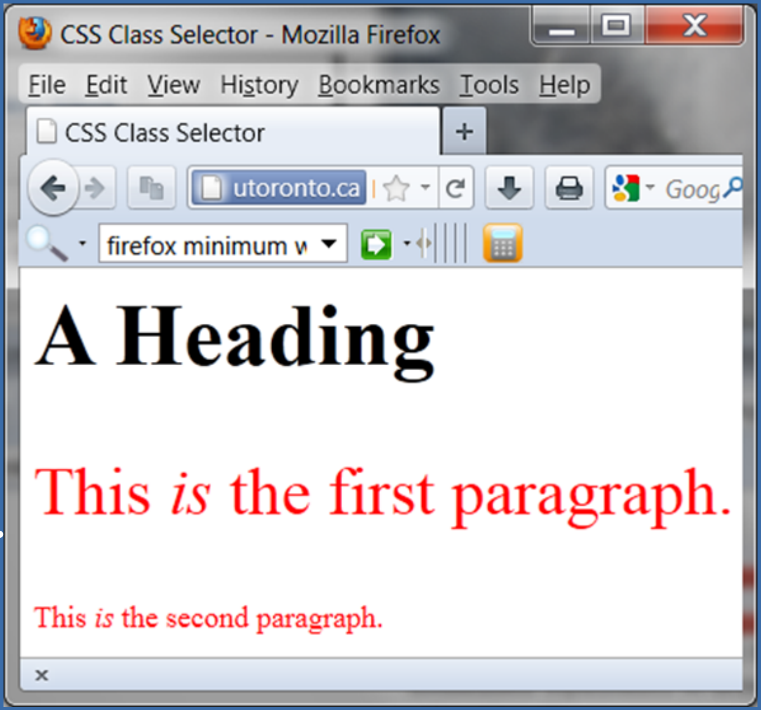
A Heading

This *is* the first paragraph.

This *is* the second paragraph.

class Attribute as Selector

```
<style type="text/css">
   .red { color: red }
   .large { font-size: 30pt }
  h1.large { font-size: 40pt }
</style>...
<h1 class="large">A Heading<h1>
<p class="large red">This
<em>is</em> the first paragraph.
</p>
<p class="red">This <em>is</em>
the second paragraph.</p>
```


CSS Class Selector - Mozilla Firefox
File Edit View History Bookmarks Tools Help
CSS Class Selector
utoronto.ca
firefox minimum w

A Heading

This *is* the first paragraph.

This *is* the second paragraph.

id Attribute as Selector

Any element can have an "id" attribute whose value is unique within the document

- can be used as the target for a hyperlink.
- used by JavaScript to identify a unique element, e.g. to place dynamic content retrieved from a server.
- used to associate style properties with a particular element, e.g. one paragraph from among a list of them.

```
<style type="text/css">
   #myid { color:red }  </style>
<p id="myid">Paragraph text in red.</p>
```

Whereas a class selector may apply to several elements, an id selector applies to at most one element.

Contextual Selection

CSS can match a search-pattern

- on a stack of open elements
- designated by a white space - separated list of selectors

Ancestors, not just parents can affect style

Can mix and match the various types of selectors into selector "sentences":

```
div.chapter  p.first { font-size: 120% }
```

Apply font-size style to paragraph elements with class "first" that occur as descendants of div elements with class "chapter"

```
#x23a  p  .foo  { color: red }
```

CSS Style Properties

Sufficiently expressive for fine-grained control

- doesn't give designers a reason to cheat

Too many properties to cover comprehensively (many dozens)

We'll focus on a few of the most important ones

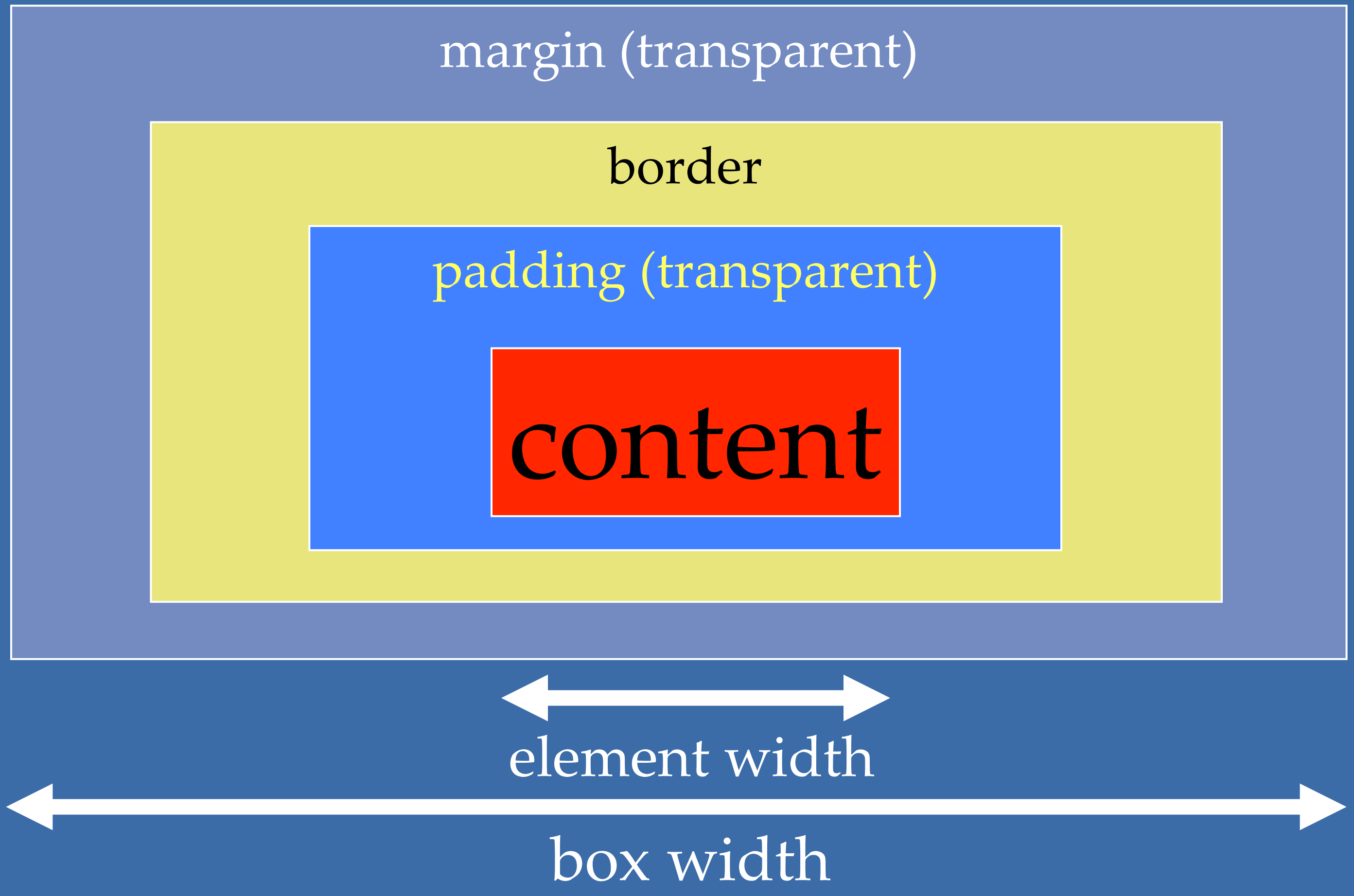
- text properties
- layout and positional control, including the "box model"

CSS Style Properties: color

- color used to specify foreground element color, especially text
- background-color used to specify element background color
- color specified with
  - a predefined name, e.g. "red"
  - RGB (red-green-blue) value expressed in Hex as 6-digit values, e.g. #A0B0C0, or
  - decimal as 3 values in the range 0-255, e.g. rgb(10,255,100)

```
p  { color: green;  background-color: #D0E0F0; }
q  { color: rgb(100,200,10); }
```

Box Formatting Model



Provides a means to control the spatial layout of elements; the basis for position-oriented properties

Box Properties

width, height of box:

```
    .small { width: 100px; height: 20px };
```

margin, margin-top, margin-right, margin-bottom, margin-left, and similar properties for border and padding

abbreviated way to specify margin, padding: list of values that provide the top/right/bottom/left widths, e.g.

```
    span  { margin: 1em 2em 3em 4em; }
```

Box Properties

Units can be expressed

- o "px" (pixels), "cm" (centimeters)
- o "em" (width of M character)
- o "%" (relative to surrounding text)

Generally preferable to use relative units such as "em" or "%" rather than absolute units like "cm", e.g.:

```
div { padding: 1em 2px;  margin 2px, 1%, 4cm; }
```

Box Properties: borders

Border property has values to control thickness, style, size of border on each of 4 sides of element

border-color, border-width, border-style, border, and many properties for specific sides such as border-bottom-color, e.g.

```
h3 {  border-color:  blue;
    border-style: solid;
    border-width: thin;
}
h4 { border: #E100D3 dashed 5px; }
```

CSS3 adds a new border-radius property for curved borders

Font and Text Properties

font-style: normal | italic | oblique

- italic matches if keyword italic or oblique found in known font list
- else must match exactly

font-variant: normal | small-caps

- small-caps satisfied if font labelled as such or can be synthesized

font-weight: normal | bold | bolder | lighter | 100-900

- always matches

Font and Text Properties

font-size: absolute | relative | length | percentage

text-align: left | center | right

CSS3 text-shadow

- horiz-shadow vert-shadow color, e.g.

```
h2 { text-shadow:  5px 5px #EEDDEE }
```

Font Properties

font-family: [family-name | generic-family] [, [family-name | generic-family]]*

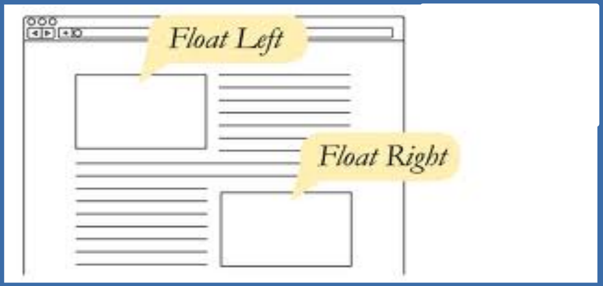
generic-families:

- serif
- sans-serif
- cursive
- fantasy
- Monospace

```
body { font-family: gill, helvetica, sans-
serif }
```

Floats

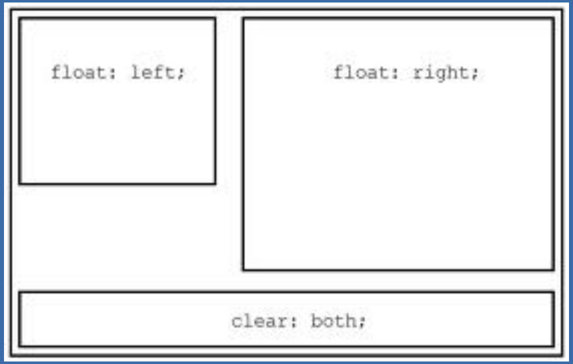
A floated element shifts out of the normal document left-to-right layout flow



If there is text beside a float, the text will wrap around the floated element

```
.right_img { float:  right; width: 200px; }
<img class="right_img" src="…" alt="…">
```

To escape the wrapping behavior, use the "clear" property, which prevents overlap of floating elements



```
q  { clear: right; }
```

Clear property possible values:

```
left, right, both, none (default)
```

Backgrounds

background-color: color used too fill the background for an element

```
body  { background-color:  #1A2B3C ; }
```

background-image: image used for background

```
div.main {
   background-image: url(“movie_poster.jpg”);
}
```

background-repeat: repeat a background image, like tiling

```
div.main {
   background-image: url(“movie_banner.jpg”);
   background-repeat: repeat-x; /* horizontal */
}
```

How do I *use* CSS?

CSS in a separate document, e.g.:

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

CSS in same document as HTML:

Global: style defined within the document head applies to the entire document

```
<style type="text/css"> style definitions … </style>
```

Local: style attributes on elements apply only to individual elements

```
<p style="color:blue"> paragraph text in blue </p>
```

Which form is best?

Using CSS

```
<html>
  <head>
      <title>CSS Linking Example</title>
      <link rel="stylesheet" type="text/css"
        href="http://www.utsc.utoronto.ca/style.css"/>
      <style type="text/css">
         h1 { color: blue }  /* local to document */
      </style>
  </head>
  <body>
    <h1>Heading in blue</h1>
    <p style="color:green">Paragraph in green.</p>
  </body>
</html>
```