Cascading Style Sheets (CSS)

Mendel Rosenblum
Driving problem behind CSS

What font type and size does <h1>Introduction</h1> generate?

Answer: Some default from the browser (HTML tells **what** browser **how**)

Early HTML - Override defaults with attributes

```
<table border="2" bordercolor="black">
```

**Style sheets** were added to address this:

Specify style to use rather than browser default

Not have to code styling on every element
Key concept: Separate style from content

Content (what to display) is in HTML files

Formatting information (how to display it) is in separate style sheets (.css files).

Use an element attribute named class to link (e.g. `<span class="test">`) 

Result: define style information once, use in many places

Consider can you make all the text in the app slightly bigger?
Or purple is our new company color.

DRY principle: Don't Repeat Yourself
Style sheet contain one or more **CSS Rules**

```
body {
    font-family: Tahoma, Arial, sans-serif;
    color: black;
    background: white;
    margin: 8px;
}
```
<table>
<thead>
<tr>
<th>CSS Selector</th>
<th>CSS</th>
<th>HTML</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tag name</td>
<td>h1 {</td>
<td>&lt;h1&gt;Today’s Specials&lt;/h1&gt;</td>
</tr>
<tr>
<td></td>
<td>color: red;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>}</td>
<td></td>
</tr>
<tr>
<td>Class attribute</td>
<td>.large {</td>
<td>&lt;p class=&quot;large&quot;&gt;...</td>
</tr>
<tr>
<td></td>
<td>font-size: 16pt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>}</td>
<td></td>
</tr>
<tr>
<td>Tag and Class</td>
<td>p.large {...}</td>
<td>&lt;p class=&quot;large&quot;&gt;...</td>
</tr>
<tr>
<td>Element id</td>
<td>#p20 {</td>
<td>&lt;p id=&quot;p20&quot;&gt;...</td>
</tr>
<tr>
<td></td>
<td>font-weight: bold;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>}</td>
<td></td>
</tr>
</tbody>
</table>
CSS Pseudo Selectors

**hover** - Apply rule when mouse is over element (e.g. tooltip)
```css
p:hover, a:hover {
    background-color: yellow;
}
```

**a:link, a:visited** - Apply rule when link has been visited or not visited (link)
```css
a:visited {
    color: green;
}
a:link {
    color: blue;
}
```
CSS Properties

Control many style properties of an element:

- Coloring
- Size
- Position
- Visibility
- Many more: (e.g. `p: { text-decoration: line-through; }`)

- Also used in animation
Color - Properties: color & background_color

Must ultimately turn into red, green, and blue intensities between 0 and 255:

- Predefined names: red, blue, green, white, etc. (140 standard names)
- 8-bit hexadecimal numbers for red, green, blue: #ff0000
- 0-255 decimal intensities: rgb(255,255,0)
- Percentage intensities: rgb(80%,80%,100%)

Example: h1: { color: red; }
CSS Box Model

![CSS Box Model Diagram]

**Total element width** =
- width +
- left padding +
- right padding +
- left border +
- right border +
- left margin +
- right margin

**Margin & Padding**
- Transparent
### CSS distance units

<table>
<thead>
<tr>
<th>Absolute</th>
<th>Relative</th>
</tr>
</thead>
<tbody>
<tr>
<td>2px</td>
<td>pixels</td>
</tr>
<tr>
<td>1mm</td>
<td>millimeters</td>
</tr>
<tr>
<td>2cm</td>
<td>centimeters</td>
</tr>
<tr>
<td>0.2in</td>
<td>inches</td>
</tr>
<tr>
<td>3pt</td>
<td>printer point 1/72 inch</td>
</tr>
<tr>
<td>2em</td>
<td>2 times the element’s current font size</td>
</tr>
<tr>
<td>3rem</td>
<td>3 times the root element’s current font size</td>
</tr>
</tbody>
</table>
Size Properties - Element, pad, margin, border

width - Override element defaults
height

padding-top
padding-right
padding-bottom
padding-left

margin-top
margin-right
margin-bottom
margin-left

border-bottom-color
border-bottom-style
border-bottom-width
border-left-color
border-left-style
border-left-width
border-right-color
border-right-style
border-right-width
etc.

p {
  border: 5px solid red;
}

CS142 Lecture Notes - CSS
position property

code: position: \texttt{static}; \quad \text{(default) - Position in document flow}

code: position: \texttt{relative}; \quad \text{Position relative to default position via top, right, bottom, and left properties}

code: position: \texttt{fixed}; \quad \text{Position to a fixed location on the screen via top, right, bottom, and left properties}

code: position: \texttt{absolute}; \quad \text{Position relative to ancestor absolute element via top, right, bottom, and left properties}

Fixed position (0,0) is top left corner
Some more common properties

background-image: image for element's background

background-repeat: should background image be displayed in a repeating pattern (versus once only)

font, font-family, font-size, font-weight, font-style: font information for text

text-align, vertical-align: Alignment: center, left, right

cursor - Set the cursor when over element (e.g. help)
Element visibility control properties

display: none; - Element is not displayed and takes no space in layout.
display: inline; - Element is treated as an inline element.
display: block; - Element is treated as a block element.
display: flex; - Element is treated as a flex container.
display: grid; - Element is treated as a grid container.

visibility: hidden; - Element is hidden but space still allocated.
visibility: visible; - Element is normally displayed
Flexbox and Grid layout

- `display: flex;` (Flexbox)
- `display: grid;` (Grid) newer layout method
  - Items flex to fill additional space and shrink to fit into smaller spaces.
  - Useful for web app layout:
    - Divide up the available space equally among a bunch of elements
    - Align of different sizes easily
    - Key to handling different window and display sizes

- Flexbox - Layout one dimension (row or column) of elements
- Grid - Layout in two dimensions (rows and columns) of elements
- Covered in discussion section
Some other CSS issues

● Inheritance
  ○ Some properties (e.g. font-size) are inherited from parent elements
  ○ Others (border, background) are not inherited.

● Multiple rule matches
  ○ General idea: most specific rule wins

```
<span>Text1</span>       span.test { color: green }
<span class="test">Text2</span>  span { color: red }
```
Adding Styles to HTML

```html
<html>
  <head>
    <link rel="stylesheet" type="text/css" href="myStyles.css" />
    <style type="text/css">
      body {
        font-family: Tahoma, Arial, sans-serif;
      }
    </style>
  </head>
  <body>
    <div style="padding:2px; ... ">
    </div>
  </body>
</html>
```
body {
  font-family: Tahoma, Arial, sans-serif;
  font-size: 13px;
  color: black;
  background: white;
  margin: 8px;
}
h1 {
  font-size: 19px;
  margin-top: 0px;
  margin-bottom: 5px;
  border-bottom: 1px solid black
}
.shaded {
  background: #d0d0ff;
}

<body>
  <h1>First Section Heading</h1>
  <p>
    Here is the first paragraph, containing text that really doesn't have any use or meaning; it just prattles on and on, with no end whatsoever, no point to make, really no purpose for existence at all.
  </p>
  <div class="shaded">
    <h1>Another Section Heading</h1>
    <p>
      Another paragraph.
    </p>
  </div>
</body>
Example Output

First Section Heading

Here is the first paragraph, containing text that really doesn't have any use or meaning; it just prattles on and on, with no end whatsoever, no point to make, really no purpose for existence at all.

Another Section Heading

Another paragraph.
CSS in the real world

- CSS preprocessors (e.g. less) are commonly used
  - Add variable and functions to help in maintaining large collections of style sheets
  - Apply scoping using the naming conventions

- Composition is a problem
  - It can be really hard to figure out what rule from which stylesheet is messing things up