CSS: Cascading Style Sheets

Lecture 9
Evolution of CSS

- MIME type: text/css
- CSS 1 ('96): early recognition of value
- CSS 2 ('98): improvements in language
  - Adding media types (screen vs print)
  - Inconsistent support by browsers
- CSS 2.1 ('11)
  - In practice since '04
  - Took forever to standardize
- CSS 3
  - Breaks standard into many (50?) modules
  - Various modules already adopted & supported
Key Idea

- Separate content and style
  - Different languages (syntax): HTML vs CSS
  - Different documents

- Goal: Single-point-of-control-over-change
  - Change font of every word in paragraph?
  - Change font of every `<em>` element in document?
  - Change font of every `<em>` element in every document on a site?
  - Change font of every `<em>` element which is part of instructions, but not finalized, on site?
CSS Syntax

- CSS is *declarative* (not *procedural*)
  - Describe a thing, not how to do compute it
  - Example: RE matching
- CSS = list of *rules* (order can matter)
- Rule = a *location* & the *style* to use there
- Basic syntax of a rule

```css
selector {
    property1: style1;
    property2: style2;
    . . .
}
/*comments always help*/
Example CSS

h2 {
/* draconian OSU visual identity */
  color: darkred;
  background: gray;
/* additional gratuitous styling */
  font-style: italic;
}

Available Properties (Styles)

- **Background**
  - `background-color`, `background-image`
- **Text, font**
  - `line-height`, `text-align`
  - `font-size`, `font-style`
- **Border, margin, padding**
  - `border-left-width`, `border-bottom-color`
- **Positioning**
  - `clear`, `display`, `float`
- **Dimension**
- **List, table**
  - `list-style-type`
  - `border-collapse`, `caption-side`
- **Generated content and other fancy stuff**
Shorthand Properties

- Example: Margins have 4 sides
  
  ```
  margin-top: 3px;
  margin-right: 5px;
  margin-bottom: 7px;
  margin-left: 9px;
  ```

- Shorthand property: margin
  
  ```
  margin: 3px 5px 7px 9px; /*TRBL*/
  margin: 7px 9px; /*TB sides*/
  margin: 2px 6px 8px; /*T sides B*/
  ```

- Other shorthand properties:
  
  - Border-width, padding, font, border, background...
Including CSS: Mechanics

- Embed directly in element
  
  \(<p\ style="\text{color:red; background:gray}"\>\)

- Place in style element in head
  
  \(<\text{head}>\)

    \(<\text{style} \text{media}=\"\text{screen}\" \text{type}=\"\text{text/css}\"\>\)

    \(<p\ \{\text{color:red; background:gray;}}\}\)\n
  \(<\text{/style}>\)\n
  \(<\text{/head}>\)\n
- Link to separate CSS file in head
  
  \(<\text{head}>\)

    \(<\text{link} \text{type}=\"\text{text/css}\" \text{rel}=\"\text{stylesheet}\" \text{href}=\"3901Style.css\" \text{media}=\"\text{screen}\" /\>\)

  \(<\text{head}>\)
Example CSS

```css
h2 {
    color: darkred;
    background: gray;
    font-style: italic;
}

em {
    font-style: normal;
    font-weight: bold;
}
```
Tree (Rooted at Body)

- `body`
  - `h1`
    - `a`
      - `href: planet.html`
      - `em`
  - `h2`
    - `p`
      - `img`
        - `src: pic.png`
        - `alt: a globe`
        - `em`
  - `h2`
    - `p`
      - `em`
      - `q`
Tree (sans Attributes)
Tree (sans Attributes)
Selectors Applied to Tree

- body
  - h1
  - h2
  - p
  - img
  - h2
  - p
    - em
    - q
  - em
  - h2 {...}
Selectors Applied to Tree
Multiple Selectors

h1 {
    color: darkred;
    background: gray;
    font-style: italic;
    border-bottom-style: solid;
}

h2 {
    color: darkred;
    background: gray;
    font-style: italic;
}
Multiple Selectors: SPOCOC

```css
h1, h2 {
  color: darkred;
  background: gray;
  font-style: italic;
}

h1 {
  border-bottom-style: solid;
}
```
Layout: The Box Model

- Block and inline elts
  - Small differences
- Border appearance
  - Style, width, color, radius
- Margins & padding
  - Transparent
  - 4 independent sides
- Padding is *part* of it
  - *Content* background shows through
- Margins gives space
  - Some adjacent margins "collapse"
The Box Model As Layers

http://geek.focalcurve.com/crashcourse-part2/
Examples

- Content
- Content
- Content
- Content
- Content
- Content
Box Sizing

```css
p {
    margin: 10px 100px 10px 10px;
    border-width: 5px 1px 5px;
    width: 200px;
    padding: 2px;
}
```

- Total width = ?
- CSS3 adds box-sizing
  - content-box (width sizes content only)
  - border-box (width includes border & padding)
Box Sizing

\[
p \{ \\
    margin: 10px 100px 10px 10px; \\
    border-width: 5px 1px 5px; \\
    width: 200px; \\
    padding: 2px; \\
}\]

- Total width = ?
- CSS3 adds box-sizing
  - content-box (width sizes content only)
  - border-box (width includes border & padding)
Collapsing Vertical Margins

Content1

Content2

Content1

Content2
Collapsing Nested Margins

Content

Content

Content
Preventing Margin Collapse

Content

padding: 1px

Content
Inheritance for SPOCOC

- A child inherits many properties from parent by default
  - Font weight, color, family, etc
  - Can be overridden in child

- Set global styles in root
  ```
  body {
    font-family: sans-serif;
  }
  ```
  - Contrast this with having to set property in all possible elements!

- Generally, text-related properties are inherited, box-related aren't (eg border)
Example Cascade

body {
  font-family: sans-serif;
  background: lightgray;
}

h2 {
  color: darkred;
  background: gray; /*new backgrnd*/
  font-style: italic;
  /*inherits font family*/
}

Cascading Properties

All nodes are sans-serif
Most nodes have light gray background
Summary

- CSS separates style from structure
  - Syntax: Rules with selectors, properties
  - Link to CSS file from HTML document
- Selectors for picking elements in tree
- Box Model
  - Content, padding, border, margin
  - Margins can collapse when overlapping
- Inheritance
  - Parent passes (font) properties to child
  - Box-related properties aren't inherited
Classes

- Not all paragraphs created equally
  - Some paragraphs are not finalized (draft), so want them styled differently

- Solution: class attribute
  
  ```html
  <p class="draft">... </p>
  ```

- CSS syntax for selector: `elt.class`
  
  ```css
  p.draft { color: gray; }
  ```

- Wildcard (any element): `.class`
  
  ```css
  .draft { font-style: italic; }
  ```

- An element can be in multiple classes
  
  ```html
  <p class="draft even">... </p>
  ```
Classes Add to Tree Structure
Notes on Classes

- When an element belongs to multiple classes, which style gets applied?
  - Different properties are combined
  - Conflicts on same property need to be resolved (more later)

- Classes should reflect semantics or structure, not visual formatting
  - Bad class name: green
  - Good class name: draft

- Example: csstest.html
Problem

- Multiple block elements that need to be styled together
  - Header and paragraph are both part of a warning that needs to be highlighted
  - `<h2 class="warning">...</h2>`
  - `<p class="warning"> ... </p>`

- This approach is awkward
  - Every block element in group needs to be decorated in this way
  - Difficult to style the entire unit (eg add a border around the whole warning)
Solution: Div Element

- Div gives a *logical* block element
- Can be styled just like any other block element
  - Font, dimension, border, margin, etc
    ```
    .warning { border: thick; }
    ```
- Can have block elements as children
  - Style inherited by children
    ```
    <div class="warning">
      <h2> ... </h2>
      <p> ... </p>
    </div>
    ```
Divs in the Tree

```
body
  ↘
  h1
  ↘
  h2
  ↘
div
    ↘
    warning
  ↘
  p
    ↘
  h2
    ↘
p
draft
  ↘
  img
  ↘
a
  ↘
  em
  ↘
  em
  ↘
  em
  ↘
ex
  ↘
q
draft
  ↘
element
  ↘
class name
```
Span

- Div is a (logical) block level element
  - Gives line breaks
- Sometimes styling/semantics belongs to *inline* elements
  - Text discussing different textbooks, where titles appear here and there
- Solution: Span tag
  - `<p>One book to consider is the <span class="book">Book of Ruby</span>, ...`
- Now all book titles can be styled consistently
- Like div, span often used with classes
Adding Spans to the Tree

- **body**
  - **h1**
  - **div**
    - **warning**
  - **h2**
  - **p**
    - **span**
      - **university**
        - University of Michigan
    - **draft**
    - **em**
    - **img**
    - **em**
  - **q**
    - **draft**
    - **Ohio State University**
    - The
Ancestors in Selectors

- Sometimes you care about *where* in the tree an element occurs
  - University names appearing *somewhere inside* warnings need a different styling

- CSS syntax: `ancestor ancestor ancestor... elt .warning .university`

- Note: *big* difference between
  - `.warning em .university`
  - `.warning em, .university`
  - `.warning, em .university`
Your Turn
More Exotic Paths in Selectors

- Child: >
  .warning > p
  .warning li > em

- Adjacent sibling: +
  h1 + p /*only first p after h1*/

- General sibling: ~
  h1 ~ p /*all sibling p's after h1*/

- Attributes: [attr="value"], *=, $=
  input[type="button"]
  a.[href$=".pdf"]
Your Turn: Select Shaded Node

- body
  - h1
  - div
    - warning
  - h2
  - p
    - span
      - university
    - em
  - a
  - img
  - p
    - draft
  - q
    - draft

- body > h2
- h1 ~ h2
- warning + h2
- h2
- h2 + p

University of Michigan
Ohio State University
Id: Class Plus Invariant

- Some classes are meant to be unique
  - At most one such element per page
    <div class="sponsors">
    </div>

- Solution: id attribute
  <div id="sponsors">
  </div>

- CSS syntax for selector: \texttt{elt\#id}
  p#sponsors { color: red; }

- Wildcard (any element): \texttt{#id}
  #headline { box-style: thin; }

- An element can have at most one id
Summary

- Classes and Ids
  - Class gives an extra dimension to tree
  - ID is unique: at most one per page
  - CSS selector syntax (. vs #)

- Divs and Spans
  - Div is a logical block element
  - Span is a logical inline element
  - Often used together with classes/ids

- Selectors with ancestors, siblings
  - CSS selector syntax (space, >, +, ~)
To Ponder

Which one wins?

```html
.draft div .warning li {  }
draft div #main li {  !important;  }
div #main ul li {  }
draft .warning ul li {  }
```