Computer Science & Engineering 120
Learning to Code

Presenting Data II – CSS

Christopher M. Bourke
cbourke@cse.unl.edu

Part I: Intro to CSS

Topic Overview

Part I:

- Intro to Cascading Style Sheets
- Inline Styling
- Span Element
- External Style Sheets

Overview

- It is important to put data into a standard, universal format: raw data can be processed in any number of ways, but data in a static table or program has limited use
- Similarly: *content* should be separated from *presentation*
- HTML semantically marks up a document, *Cascading Style Sheets* can be used to add stylistic elements
- Cascading: HTML elements are nested and *inherit* style *properties* which can be overridden or modified
- You can define "general" rules/properties that apply to all elements in a document to the most specific which can apply to a single element
- Ultimately rules are general; browsers decide how styles are rendered (and often create their own, nonstandard properties)

Overview

- CSS rules follow a key-value pair syntax
- Key is a CSS *property*
- Value depends on the type of property
- Syntax: `property: value`

Inline Styles

- Styles can be applied to particular HTML elements
- Style rules are placed "inline" with HTML tags
- The `style` attribute of any element can contain CSS style rules
- Multiple rules are separated by a semicolon

```
1 <h1 style="color: red">My Home Page</h1>
2
3 <p style="font-family: helvetica; font-size: 14pt;
4   margin: 1em;">Greetings, this is my webpage.</p>
```

Inheritance

- A style rule applies to an element and *all of its children*
- Unless it is overridden by another style

```
<div style="color: red">
<p>This paragraph inherits its red color
from the div tag.</p>
<p style="color: green">But this one is green
as it has been overridden with another
inline style</p>
<p>This paragraph is again red</p>
</div>

<p>This paragraph's color inherits from its parent
HTML element's style</p>
```

Overview

- Often want to apply a style to a part of a document that is not a distinct HTML element
- Italicize or bold a word within a paragraph
- The `<span>` tag was designed to allow this
- Doesn't apply any particular semantic meaning to the document
- Used to apply style rules that "span" a particular part of the document

```
<p>This course is <span style="font-weight: bold">CSCE 120</span>
&ndash; Learning to Code.  We meet every Tuesday and Thursday.
Please be sure to <span style="font-style: italic;">bring your
laptop.</span></p>
```

External Style Sheets

- Inline style logically separates content and presentation, but still places presentation within the document
- Better to use *external* style sheets
- A separate source file (usually `.css`) that contains presentation rules
- Including the style sheet in the header:
  `<link href="myStyles.css" rel="stylesheet" type="text/css">`
- Similar syntax, but rules are enclosed with curly brackets

External Style Sheets

Example

```
/*
  myStyles.css
  Multiline comments are similar to JavaScript
*/

/* style for body element, all children will inherit */
body {
  font-family: helvetica;
  color: rgb(20, 20, 20);
}

/* rule for all paragraphs */
p {
  margin-left: 2em;
}

h1 {
  padding: 10px;
  color: black;
}
```

Part II: Common Style Elements

Font Properties I

Property	Values
`font-style`	`normal` `italic`

`font-weight`	`normal`
	`bold`
	`bolder`
	`lighter`

`font-size`	`medium`
	`pt, px, em`
	`x%`
	`small`
	`large`

`font-family`	`"Times New Roman"`
	`Helvetica`
	`Serif`

Font Properties II

- Font family availability depends on browser/system; best to include generic family
- Multiple properties can be set at once: `font: bold 3em serif;`
- Units: percentage, pixels, points, em

Margins, Borders & Padding I

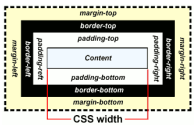

Figure : Margins, Borders, Padding

Margins, Borders & Padding II

- Elements (usually `<div>`s) have margins, borders, and padding around their content
- Each can be set independently: `margin-top`, `margin-right`, `margin-bottom`, `margin-left`
- Can be set all at once using `margin: 10px;` or `margin: 10px 20px 10px 20px;` (top, right, bottom, left)
- Similar for padding: `padding-top`, `padding-right`, `padding-bottom`, `padding-left` or `padding`

Margins, Borders & Padding III

- Borders similar but have more components: `-width`, `-style`, `-color`
- Styles: `none`, `dotted`, `dashed`, `solid`, etc.
- Shortcut: `border: 5px solid green`
- Units: `px`, `em`, etc.

Colors I

- Foreground color (text): `color`
- Background color: `background-color`
- Many *named* colors (see `http://www.w3schools.com/cssref/css_colorsfull.asp`)
- Main color method: specify red, green, blue color levels
- Numeric: integer value 0 through 255: `color: rgb(255, 255, 0);` (full red, full green, no blue; yellow)
- Additive, so white: `color: rgb(255, 255, 255);`, black: `color: rgb(0, 0, 0);`
- Hexadecimal: same range, but expressed in base-16: `00`, `01`, ... `09`, `0a`, ... `0f` `10`, ..., `ff`
- Use a hash mark: `color: #ffff00;`

Colors II

- *Opacity* is a measure of how transparent an element is
- 1.0: completely opaque; 0.0: completely transparent
- `opacity: 0.5;`
- RGBA ("alpha channel"): `color: rgba(255, 0, 0, 0.5);`

Visibility

- Some elements can be "hidden" so they are not displayed
- Common when using animation: hide, fade-in, out, etc.
- Visibility: `visibility: visible` (or `hidden`)
- Display: `display: inline` (or `block` or `none`)
- Inline: does not start on a new line, takes only as much width as necessary (like `<span>`)
- Block: starts a new line, takes up the full width of parent element (like `<div>`)
- Visibility vs display: `display: none` means the element does not affect the layout `visibility: hidden` does not show the element, but the element still takes up as much space as if it were visible

Other Style Elements

Resources

- List of (315) properties: `http://meiert.com/en/indices/css-properties/`
- CSS Playground: `http://css3.mikeplate.com/`
- Another: `http://playground.webflow.com/`

Part III: Selectors, Combinators, etc.

Topic Overview

- Classes, Identifiers, Selectors
- Advanced Combinators, etc.
- Frameworks

Classes

- CSS rules can be made more fine-grained
- You can use the `class` attribute to give an element one or more "classes" of style rules
- HTML:

```
<p class="leading">...</p>
<p class="opening important">...</p>
```

- Rules can be applied to classes in a style file using `.class` syntax
- Can be combined with other *selectors*

Classes

Example

```
/* applies to ANY element with a leading class */
.leading {
  font-size: 120%;
  font-weight: 500;
}

/* applies only to div elements with an important class */
div.important {
  color: red;
}

/* applies to elements with BOTH classes */
.leading .important {
  font-family: serif;
}

/* applies to elements with EITHER class */
.leading, .important {
  background-color: rgb(25, 25, 25);
}
```

Identifiers

- Any element can have an identifier, `id` attribute
- Style rules can be applied to that element only, avoiding inline styling
- Syntax: use a hash, `#elementId`
- Can do "conditional" style when combined with classes

Identifiers

Example

```
/* applies only to the element with id="contactInfo" */
#contactInfo {
  margin: 1em;
  border: solid black 1px;
}

/* applied to id="contactInfo" only if it has a highlighted class */
#contactInfo.highlighted
div.important {
  color: red;
}
```

Combinators I

- Universal Selector:
  `*`
  Applies the rule to *every* element
- Descendent selector: combinations of elements can select nested elements
  `div p`
  Applies the rule to any paragraph that is a descendent of a `<div>` element
- Child Selector:
  `div > p`
  Applies the rule to any paragraph that is an *immediate child* of a `<div>` element

Combinators II

- Adjacent Sibling Selector:
  `div + p`
  Applies the rule to any paragraph that *immediately follow* a `<div>` element
- General Sibling Selector:
  `div ~ p`
  Applies the rule to any paragraph that is a sibling (not just following) of a `<div>` element

Attribute Selectors

Can specify rules to apply to elements with particular attributes and attribute values

```
/* applies to any anchor with an href */
a[href] { ... }

/* applies to any anchor that links to hello.pdf */
a[href="hello.pdf"] { ... }

/* applies to any anchor that links to resources that
   begin with http */
a[href^="http"] { ... }

/* applies to any anchor that links to resources that
   ends with jpg */
a[href$="jpg"] { ... }

/* applies to any anchor that links to resources that
   contains cse */
a[href*="cse"] { ... }
```

Advanced CSS

More:

- Animations & Transitions
- Calculated Values (`calc()`)
- Gradients
- Webfonts
- Media Queries for adaptive styles (mobile, desktop, etc.)
- CSS4: more programming-like capabilities

Resources:

`http://www.w3.org/community/webed/wiki/Advanced_CSS_selectors`

`http://tutorialzine.com/2013/10/12-awesome-css3-features-you-can-finally-use/`

HTML & CSS Frameworks I

- Coding is hard, graphic design is hard
- Styling is ever more important with various mobile devices
- Many tools, frameworks and libraries have been developed to help get you started

HTML & CSS Frameworks II

Tools:

- Syntactically Awesome Stylesheets (Sass): `http://sass-lang.com/`
- LESS: `http://lesscss.org/`
- CSS Next: `http://cssnext.io/`

Frameworks/Libraries:

- HTLM5 Boilerplate (`https://html5boilerplate.com/`)
- jQuery UI (`http://jqueryui.com`)
- Bootstrap (`http://getbootstrap.com/`)