Data Visualization (DSC 530/CIS 568)

Web Programming

Dr. David Koop
Definition of Visualization

“Computer-based visualization systems provide visual representations of datasets designed to help people carry out tasks more effectively.”

— T. Munzner
Why Visual?

[F. J. Anscombe]
Why Visual?
Why Computers?

[Cerebral, Barsky et al., 2007]
**Design Space: Think Broad**

[Design Study Methodology, Sedlmair et al., 2012]

**Know**

**Consider**

**Propose**

**Select**
A Nine-Stage Framework

PRECONDITION
personal validation

CORE
inward-facing validation

ANALYSIS
outward-facing validation

[Design Study Methodology, Sedlmair et al., 2012]
Assignment 1

- Due Friday, Feb. 8
- HTML, CSS, SVG, JavaScript
- Start now!
Languages of the Web

• HTML
• CSS
• SVG
• JavaScript
  - Versions of Javascript: ES6, ES2015, ES2017…
  - Specific frameworks: react, jQuery, bootstrap, D3
Web Programming Tools

• Basic: Text editor and Modern Browser
• Developer Tools: Built in to browsers (e.g. Chrome Developer Tools)
• Web Environments: JSFiddle, Liveweave, CodePen, etc.
• IDEs: WebStorm, etc.
Hyper Text Markup Language (HTML)

• Markup languages allow users to encode the **semantics** of text

• Tags define the boundaries of the structures of the content
  - Tags are enclosed in angle brackets (e.g. `<html>`)
  - Most of the time, you have a start and end tag
  - End tags are just like start tags except that they have forward slash after the open bracket (e.g. `</html>`)
  - Tags may be nested but not mismatched
    - `<p>A <strong><em>very</em></strong> cool example</p>`
    - `<p>A <strong>very <em>cool</em></strong> example</p>`
  - What about `<img src="mypicture.png" alt="My Image">`?
HTML Elements and Attributes

• Tags denote **elements** of the content (e.g. sections, paragraphs, images)
• Each element may have **attributes** which define other information about the element
  - An attribute has a **key** and **value** (*key=“value”*)
  - e.g. `<img src=“mypicture.png” alt=“My Image”>`
• Many different elements available
  - Common: headers (**h1**, ..., **h6**), paragraph (**p**), lists (**ul**, **ol**, **li**), emphasis (**em**, **strong**), link (**a**), spans & divisions (**span**, **div**)
  - Lots more (e.g. abbr): see [MDN Documentation](https://developer.mozilla.org)
• Many different attributes available
  - See [MDN Documentation](https://developer.mozilla.org): note that some are legacy due to CSS
HTML Element Structure & Naming

• Elements structure a document
  - Document Object Model (DOM)
  - We can visualize this information
  - More importantly, we can navigate this tree

• Identifying and Classifying elements: id and class attributes
  - id identifies a single element—use for a unique case
  - class may identify multiple elements—use for common cases
  - Each element may have multiple classes, separate by spaces
  - Use normal identifiers: don’t start the name with a number
Other HTML Trivia

• `<`, `>`, `&`, and " are special characters, escape with `<`, `>`, `&`, and `"` (note the semi-colon)
• Comments are enclosed by `<!--` and `-->`
  - `<!-- This is a comment -->`
• HTML Documents begin with a `DOCTYPE` declaration
  - For HTML5, this is easier `<!DOCTYPE html>`
• `meta` tag: `<meta charset="UTF-8"/>`
• HTML has audio and video tags, math equation support, and more
Basic HTML File

```html
<!DOCTYPE html>
<html>
  <head>
    <title>A Basic Web Page</title>
  </head>
  <body>
    <h1>My Wicked Awesome Web Page</h1>
    <p><em>This is <strong>cool</strong>. What about <u><strong>this?</strong></u></em></p>
  </body>
</html>
```

- [https://codepen.io/dakoop/pen/PdRKE](https://codepen.io/dakoop/pen/PdRKE)
What is CSS?
Cascading Style Sheets (CSS)

- Separate from content, just specifies how to style the content
- Style information can appear in three places:
  - External file
  - In a style element at the beginning of the HTML file
  - In a specific element in the body of a document (least preferable)
- Why Cascading?
  - Don’t want to have to specify everything over and over
  - Often want to use the same characteristics in a region of the DOM
  - Use inheritance: properties that apply to children cascade down
CSS Selectors

• How do we specify what part(s) of the page we want to style?

• The **element types** themselves (the HTML tag)
  - strong { color: red; }

• Classes of elements (ties to HTML class attribute)
  - .cool { color: blue; }

• A specific element (ties to HTML id attribute)
  - #main-section { color: green; }

• Relationships
  - Descendant: p em { color: yellow; }
  - Child: p > em { color: orange; }

• Pseudo-classes: a:hover { color: purple; }
Other CSS Bits

- Comments: /* This is a comment in CSS */
- Grouping Selectors: p, li { font-size: 12pt; }
- Multiple Classes: .cool.temp { color: blue; }
- Colors:
  - Names (Level 1, 2, & 3): red, orange, antiquewhite
  - Dash notation (3- & 6-character): #fff, #00ff00
  - Integer or % RGB and HSL Functions: rgb(255, 0, 0), rgb(50%, 50%, 0%), hsl(120, 100%, 50%)
  - Also background-color
- Watch out for multiple rules (look at how a web browser parses)
- Again, much more documentation at MDN
Example CSS

body {
  font-face: sans-serif;
  font-size: 12pt;
}
em { color: green; }
em u { color: red; }
em > strong { color: blue; }
img { border: 4px solid red; }

• What colors are displayed for this HTML (with the above stylesheet)?
  - <em>This is <strong>cool</strong>. What about </em>
  <u><strong>this?</strong></u></em>
• [https://codepen.io/dakoop/pen/ErNJvJ](https://codepen.io/dakoop/pen/ErNJvJ)
CSS Specificity

- **Example:**
  - **CSS:**
    ```css
    p.excitng { color: red; }
    p { color: blue; }
    ```
  - What is the color of the paragraph `<p class="exciting">Cool</p>`?

- Generally, last rule listed overrides previous rules
- …but anytime a selector is **more specific**, it has precedence
- `p.excitng` is a more specific selector
- When in doubt, test it in a browser
- [https://codepen.io/dakoop/pen/MLbRQz](https://codepen.io/dakoop/pen/MLbRQz)
How to add CSS to HTML

• External: a separate file via a link element (in the `<head>` section):
  - `<link rel="stylesheet" href="styles.css">`

• Embedded: in the header:
  - `<style type="text/css"> ... </style>`

• Inline: for a specific element: (Discouraged!)
  - `<p style="font-weight: bold;">Some text</p>`
What is the difference between vector and raster graphics?
Scalable Vector Graphics (SVG)

- Vector graphics vs. Raster graphics
- Drawing commands versus a grid of pixels
- Why vector graphics?
SVG Background

- Another markup language:
  - Describe the shapes and paths by their endpoints, characteristics
- SVG can be embedded into HTML5 documents!
- Pixel Coordinates: **Top-left** origin

\[(0,0) \quad (\text{width},0) \quad (\text{width},\text{height})\]
SVG Elements

• Drawing primitives:
  - Lines, Circles, Rects, Ellipses, Text, Polylines, Paths
  - Work by specifying information about how to draw the shape
  - Lots more: see MDN Documentation

• Ordering/Stacking:
  - SVG Elements are drawn in the order they are specified

• Paths: directions for drawing
SVG Styles

• We can specify styles or SVG elements in CSS!
• Example:

```css
circle { fill: green; stroke: black; stroke-width: 4px; }
.normal { fill: red; stroke: blue; stroke-width: 2px; }
#p1 { fill: none; stroke: red; stroke-width: 3px; }
```
SVG Example

- [Codepen](http://codepen.io/dakoop/pen/yexVXb)

```xml
<svg id="mysvg" width="300" height="600">
  <circle cx="50" cy="50" r="50"/>
  <rect class="lego" x="150" y="150" width="50" height="20"/>
  <path id="triangle" d="M 20 200 L 120 200 L 120 250 Z"/>
</svg>

circle { fill: green; stroke: black; stroke-width: 4px; }
.lego { fill: red; stroke: blue; stroke-width: 2px; }
#triangle { fill: none; stroke: orange; stroke-width:3px; }
```