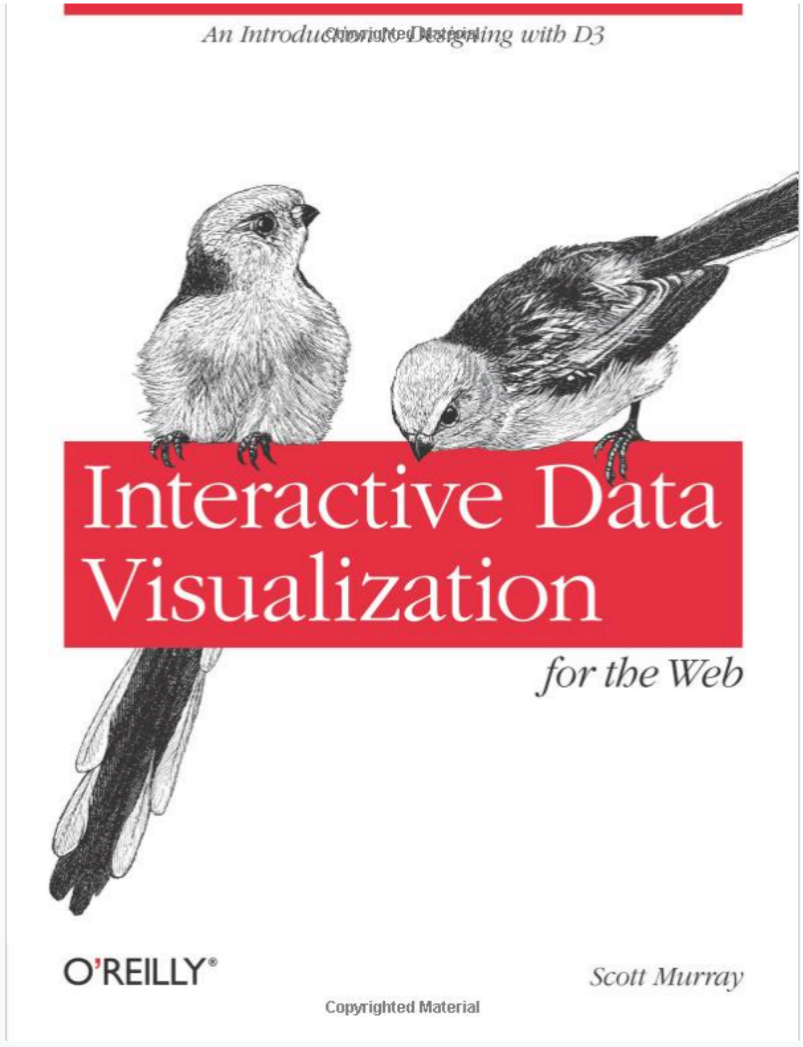


An Introduction to Designing with D3

Interactive Data Visualization

for the Web

O'REILLY®

Scott Murray

Interactive Data Visualization for the Web

Scott Murray

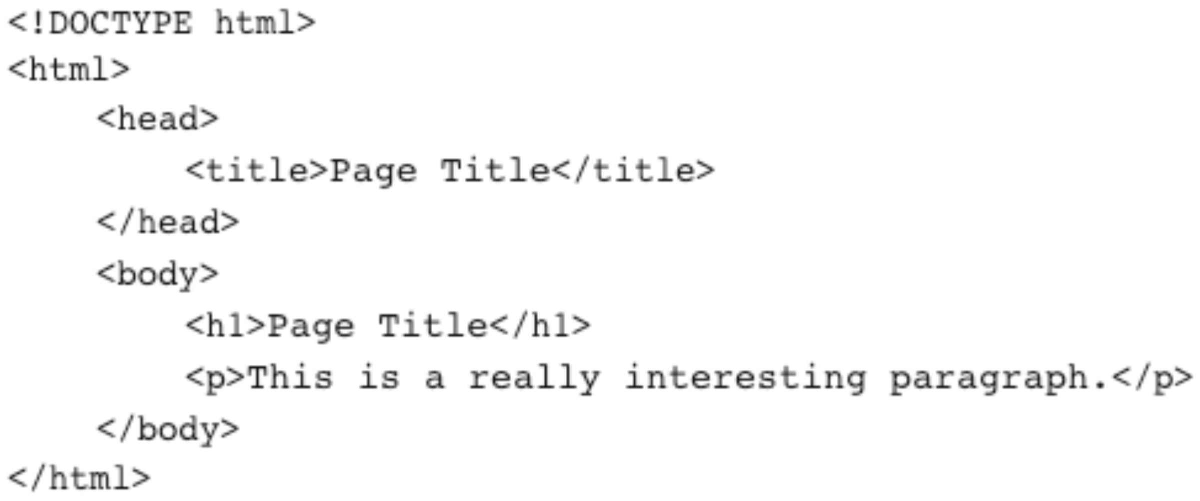
Technology Foundations

- Web technologies
  - HTML
  - CSS
  - SVG
  - Javascript

HTML
(Hypertext Markup Language)

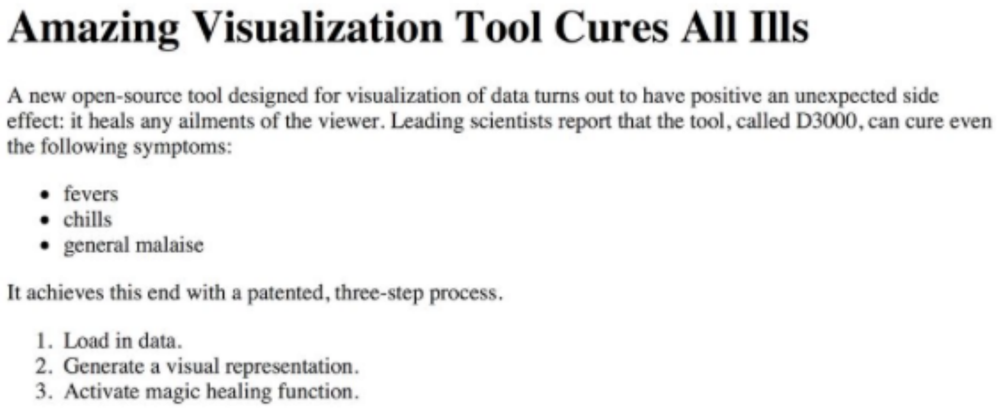
- Used to mark up the content of a web page by adding a structure to the elements in the web page
- Elements
  - – Paragraph, division, ordered and unordered list, headings, links, body, head, title, etc., and the root html
  - – Elements are created by tags, for example,
    - <p> defines the beginning of a paragraph
    - </p> closes the paragraph

A Simple HTML



```
<!DOCTYPE html>
<html>
    <head>
        <title>Page Title</title>
    </head>
    <body>
        <h1>Page Title</h1>
        <p>This is a really interesting paragraph.</p>
    </body>
</html>
```

1. Can you create a web page like the following, with your own content
2. Then enhance your web page with tables and images



Amazing Visualization Tool Cures All Ills

A new open-source tool designed for visualization of data turns out to have positive an unexpected side effect: it heals any ailments of the viewer. Leading scientists report that the tool, called D3000, can cure even the following symptoms:

- fevers
- chills
- general malaise

It achieves this end with a patented, three-step process.

1. Load in data.
2. Generate a visual representation.
3. Activate magic healing function.

A List of Common Elements

`<!DOCTYPE html>`

The standard document type declaration. Must be the first thing in the document.

`html`

Surrounds all HTML content in a document.

`head`

The document `head` contains all metadata about the document, such as its `title` and any references to external stylesheets and scripts.

`title`

The title of the document. Browsers typically display this at the top of the browser window and use this title when bookmarking a page.

`body`

Everything not in the `head` should go in the `body`. This is the primary visible content of the page.

`h1`, `h2`, `h3`, `h4`

These let you specify headings of different levels. `h1` is a top-level heading, `h2` is

`p`

A paragraph!

`ul`, `ol`, `li`

Unordered lists are specified with `ul`, most often used for bulleted lists. Ordered lists (`ol`) are often numbered. Both `ul` and `ol` should include `li` elements to specify list items.

`em`

Indicates emphasis. Typically rendered in *italics*.

`strong`

Indicates additional emphasis. Typically rendered in **boldface**.

`a`

A link. Typically rendered as underlined, blue text, unless otherwise specified.

`span`

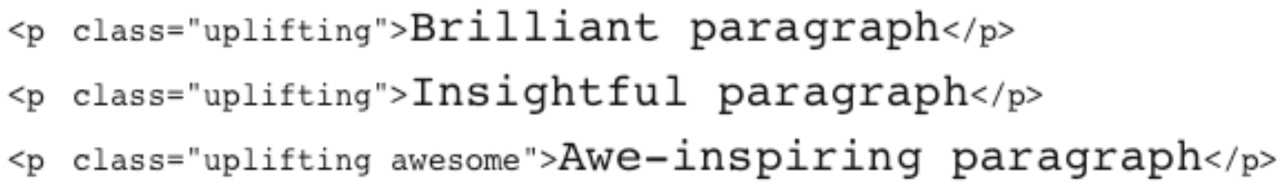
An arbitrary `span` of text, typically within a larger containing element like `p`.

`div`

An arbitrary *division* within the document. Used for grouping and containing related elements.

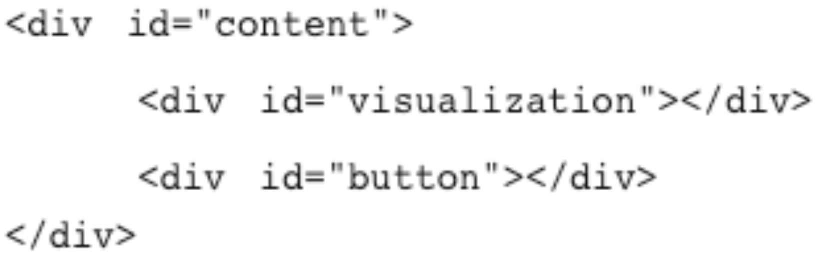
Comments, Classes, and IDs

- You can add comments to your html document with <!– this is a comment -->
- Elements can be identified by their classes or IDs (important for CSS and Javascript)
- Classes:



```
<p class="uplifting">Brilliant paragraph</p>
<p class="uplifting">Insightful paragraph</p>
<p class="uplifting awesome">Awe-inspiring paragraph</p>
```

- IDs: (only used for one element and only once in a page)



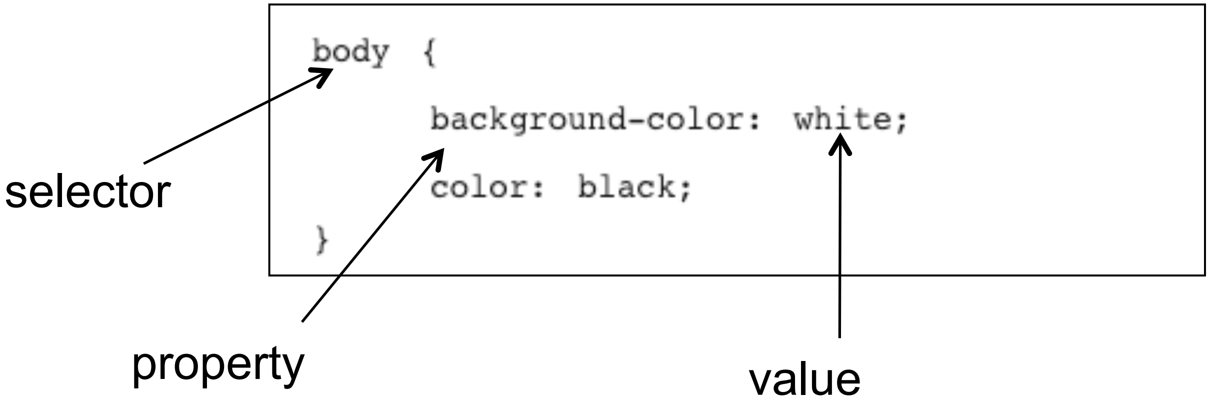
```
<div id="content">
    <div id="visualization"></div>
    <div id="button"></div>
</div>
```

Document Object Model (DOM)

- Describes the hierarchical structure of HTML
  - The parent, child, sibling, ancestor, descendant relationships among the HTML elements
- Open the development tool of your browser to check the DOM of the page you just created

Cascading Style Sheets (CSS)

- To style the visual presentation of DOM elements



```
body {
     background-color: white;
     color: black;
}
```

selector

property

value

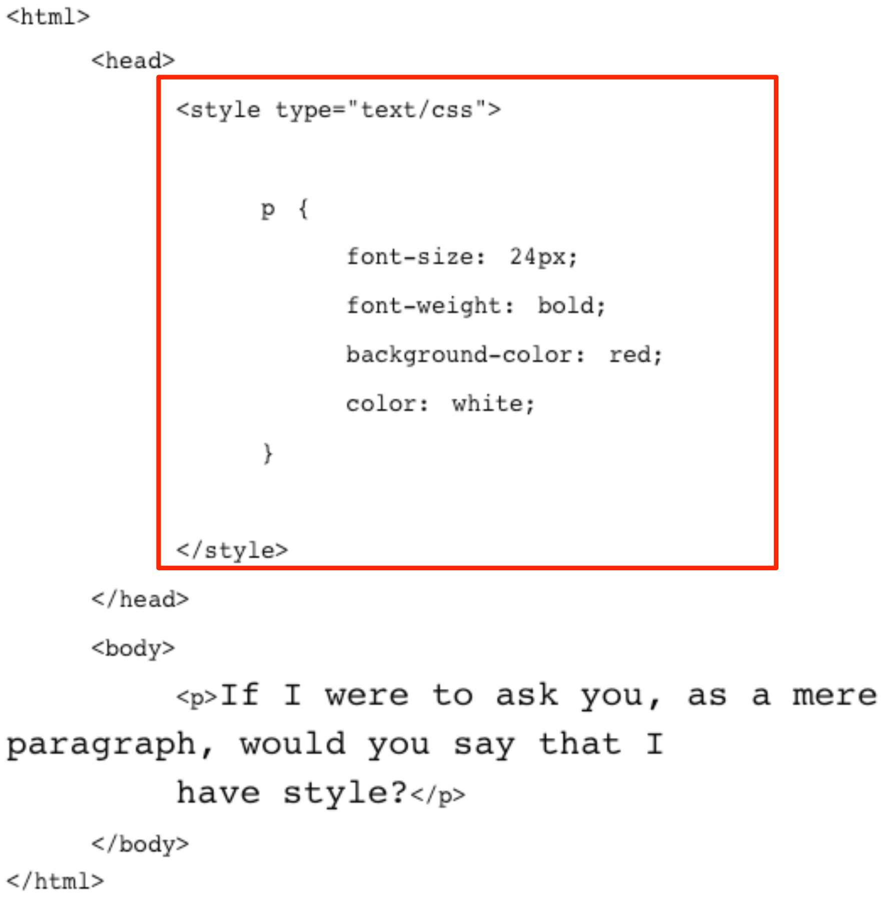
- Selectors:
  - DOM elements : body, h1, p, div, em, etc.
  - Descendant selectors: div p /* p elements contained in a div
  - Class selectors: example: .caption, .label, .axis (caption, label, and axis are class names
  - You can string the classes together: e.g. .bar.highlight
  - ID selectors: e.g. #nav #export

Properties

- There are tons of properties in CSS
- Common properties: font-family, font-size, background-color, background-image, border, etc. http://tech.journalism.cuny.edu/documentation/css-cheat-sheet/)
- An exhaustive list of CSS properties: https://developer.mozilla.org/en-US/docs/Web/CSS/Reference

Apply CSS rules

- Embed CSS in HTML



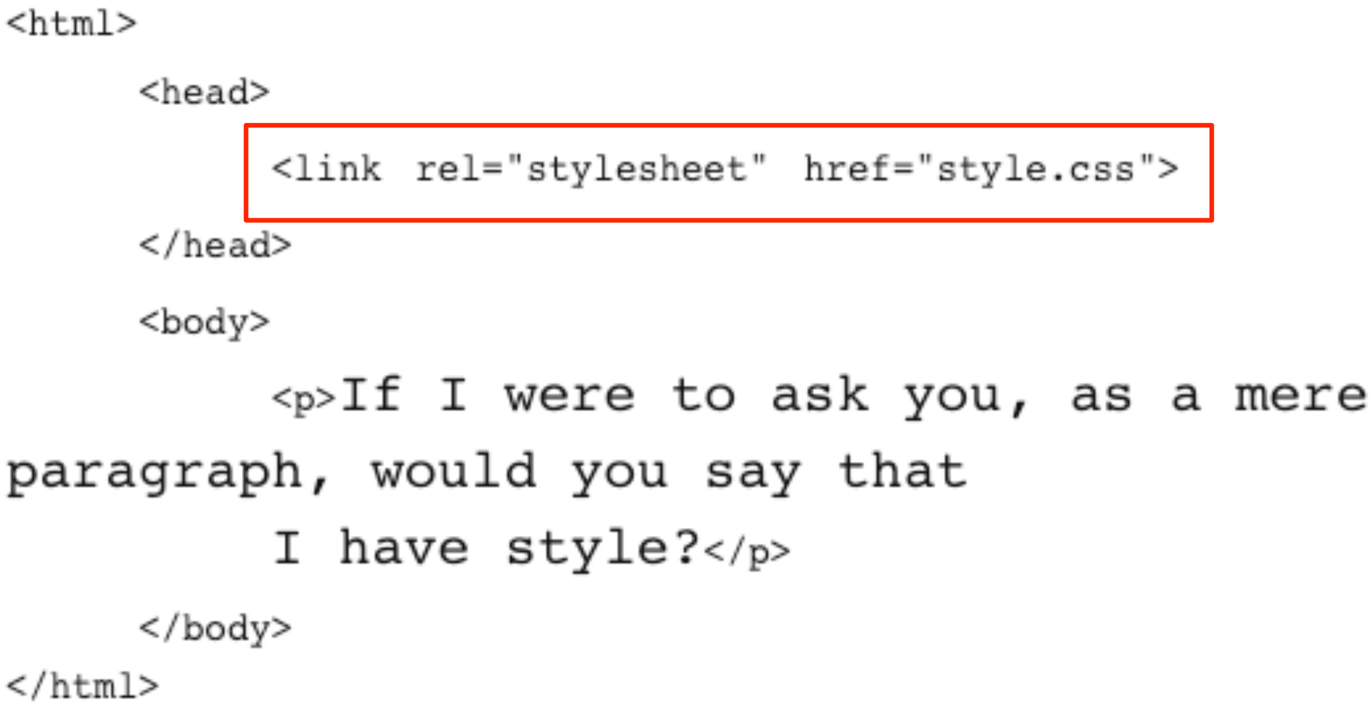
```
<html>
      <head>
            <style type="text/css">

                  p {
                        font-size: 24px;
                        font-weight: bold;
                        background-color: red;
                        color: white;
                  }

            </style>
      </head>
      <body>
            <p>If I were to ask you, as a mere
paragraph, would you say that I
            have style?</p>
      </body>
</html>
```

Apply CSS rules

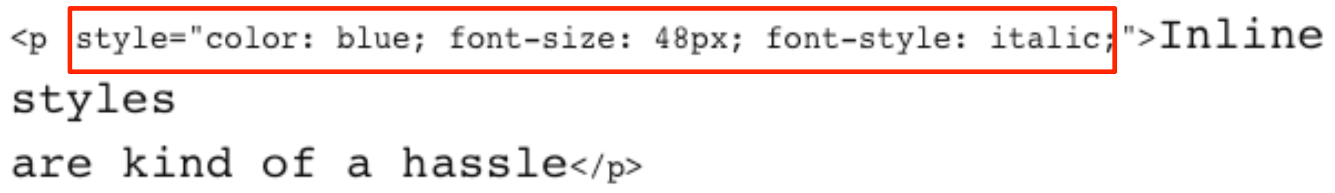
- Reference an external file



```
<html>
      <head>
            <link rel="stylesheet" href="style.css">
      </head>
      <body>
            <p>If I were to ask you, as a mere
paragraph, would you say that
            I have style?</p>
      </body>
</html>
```

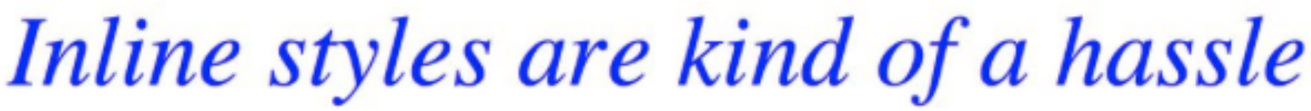
Apply CSS rules

- Attach inline styles



```
<p style="color: blue; font-size: 48px; font-style: italic;">Inline
styles
are kind of a hassle</p>
```

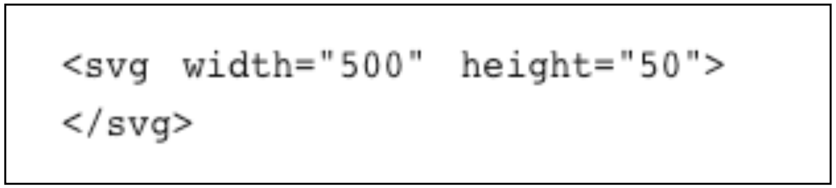
⇩



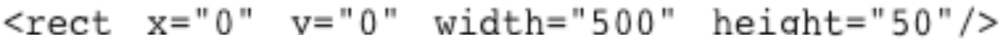
Inline styles are kind of a hassle

Scalable Vector Graphics (SVG)

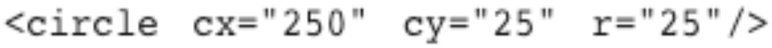
- Use D3 to produce SVG
- SVG can be directly included in a HTML document
- How to write SVG?
  - Create a SVG element



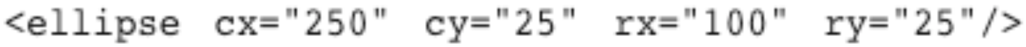
```
<svg width="500" height="50">
</svg>
```

  - Between the svg tags, include your visual elements
    - rect,circle, elliopse, line, text, and path
  - (0,0) is the top left corner
  - rect 

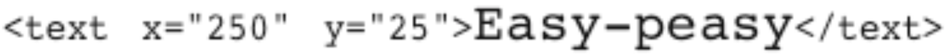
```
<rect x="0" y="0" width="500" height="50"/>
```

  - circle 

```
<circle cx="250" cy="25" r="25"/>
```

  - ellipse 

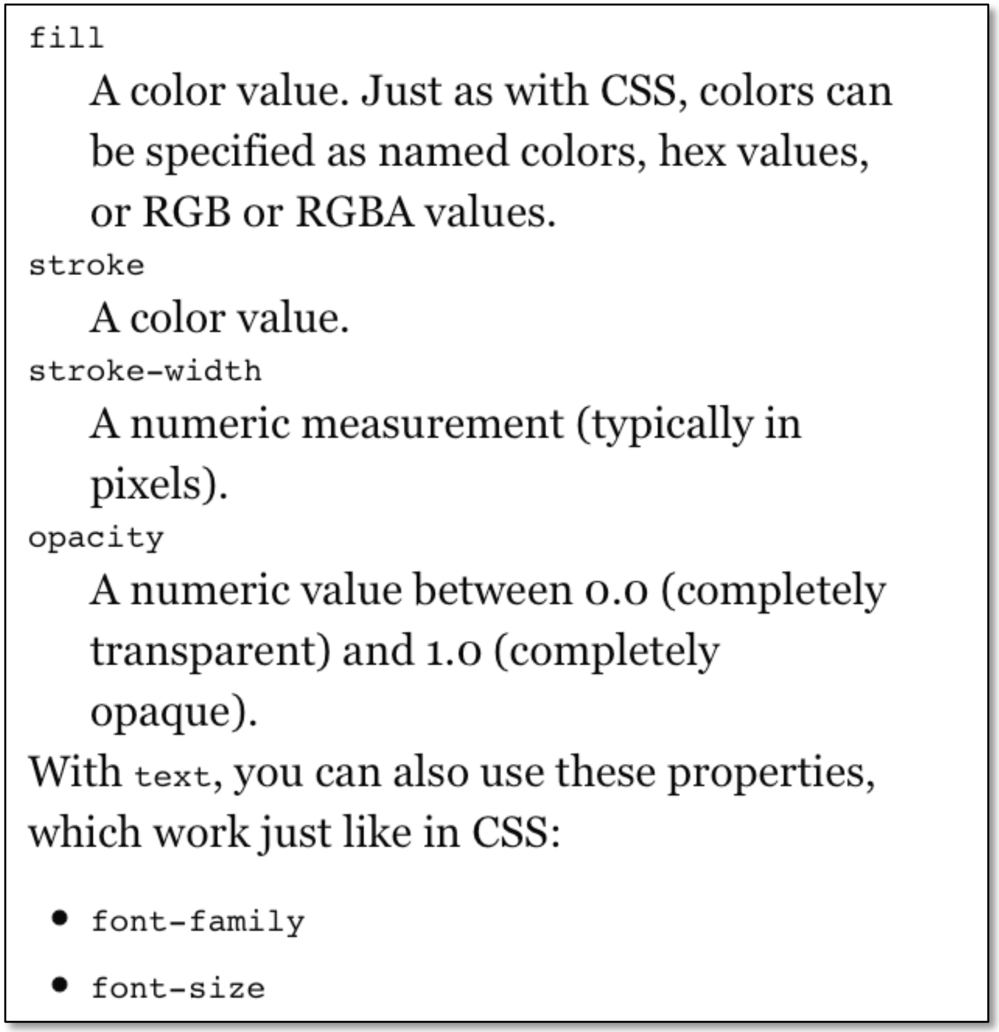
```
<ellipse cx="250" cy="25" rx="100" ry="25"/>
```

  - text 

```
<text x="250" y="25">Easy-peasy</text>
```

  - path anything more complex then the preceding shapes

Styling SVG



`fill`

A color value. Just as with CSS, colors can be specified as named colors, hex values, or RGB or RGBA values.

`stroke`

A color value.

`stroke-width`

A numeric measurement (typically in pixels).

`opacity`

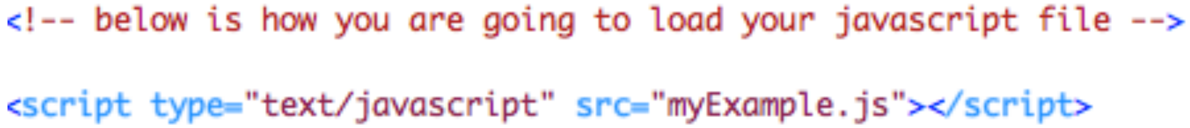
A numeric value between 0.0 (completely transparent) and 1.0 (completely opaque).

With `text`, you can also use these properties, which work just like in CSS:

- `font-family`
- `font-size`

Javascript

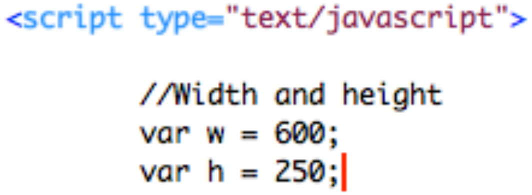
- Putting javascript code in your HTML
  - External source file:



```
<!-- below is how you are going to load your javascript file -->

<script type="text/javascript" src="myExample.js"></script>
```

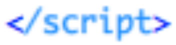
  - Direct put in your HTML:



```
<script type="text/javascript">

        //Width and height
        var w = 600;
        var h = 250;
```

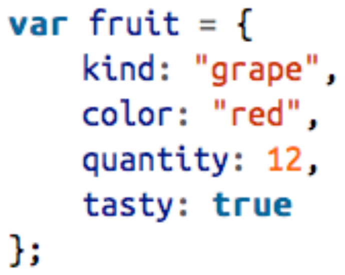
…



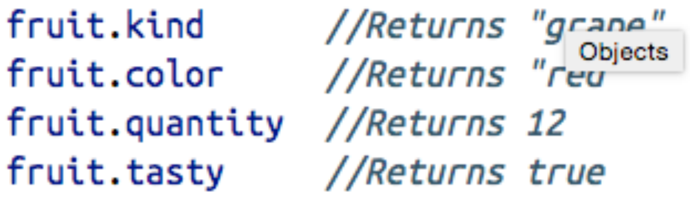
```
</script>
```

Quick Review of JS syntax

- Print message to the console (in the development window)
  - console.log(“hello world!”);
- Declare a variable
  - var number = 5;
  - You can later change the variable content to a value of different type
    - number = “hello”;
  - JS is a losely typed language
- Declare an array (useful for you to try some visualization)
  - var numbers = [1,2,3,4,5];
- Objects



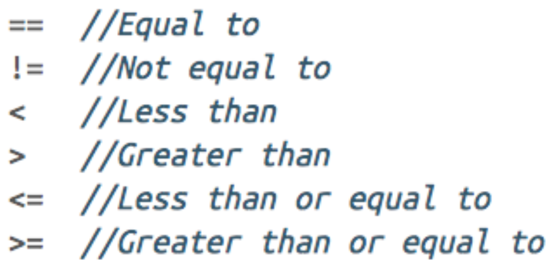
```
var fruit = {
    kind: "grape",
    color: "red",
    quantity: 12,
    tasty: true
};
```



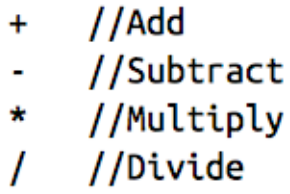
```
fruit.kind        //Returns "grape"
fruit.color       //Returns "red"
fruit.quantity    //Returns 12
fruit.tasty       //Returns true
```

Quick Review of JS syntax

- Mathematical Operators

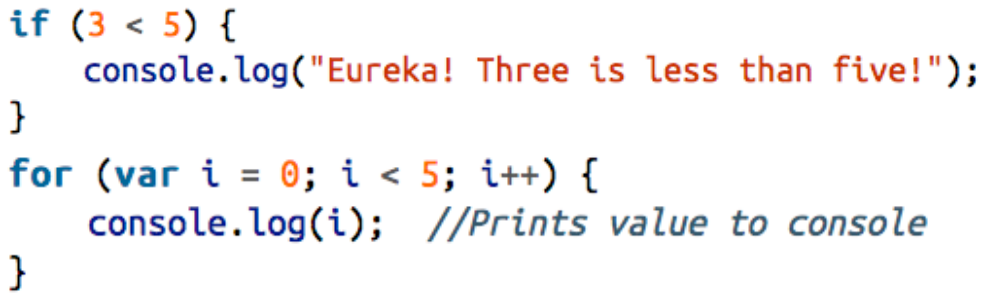


```
==  //Equal to
!=  //Not equal to
<   //Less than
>   //Greater than
<=  //Less than or equal to
>=  //Greater than or equal to
```



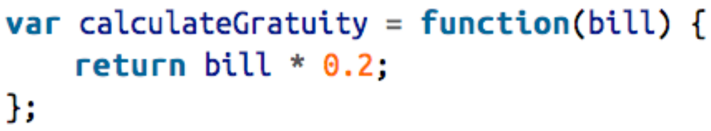
```
+   //Add
-   //Subtract
*   //Multiply
/   //Divide
```

- Control structures



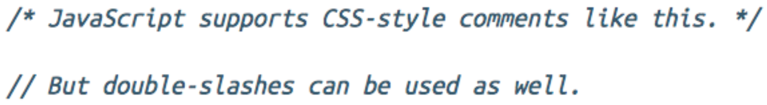
```
if (3 < 5) {
    console.log("Eureka! Three is less than five!");
}
for (var i = 0; i < 5; i++) {
    console.log(i);  //Prints value to console
}
```

- Functions (a chunk of reusable code)



```
var calculateGratuity = function(bill) {
    return bill * 0.2;
};
```

- Comments



```
/* JavaScript supports CSS-style comments like this. */

// But double-slashes can be used as well.
```

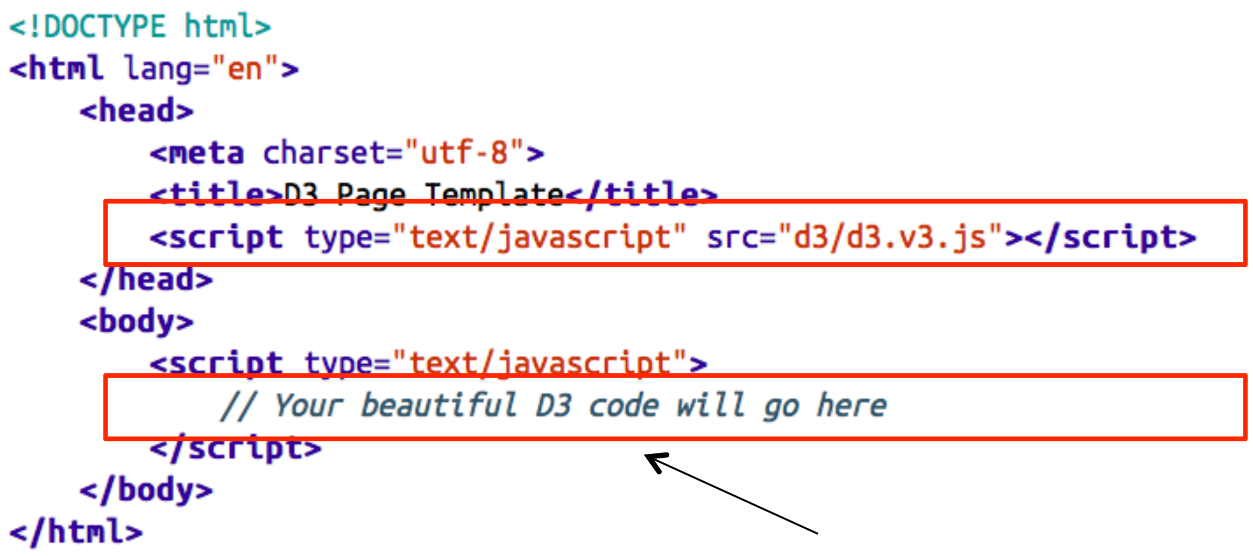
Javascript Tutorials

- Codecademy
  http://www.codecademy.com/tracks/javascript
- Other resources:
  - Overview: http://javascript.crockford.com/survey.htmlTutorial: http://www.w3schools.com/js/
  - Tutorial:http://www.w3schools.com/js/
  - Reference book: The Definitive Guide, 6th Edition

Do this!

Data Driven Document (D3)

- Downloading D3 - http://d3js.org
- Unzip the download and create a sub-folder called d3 in the folder you put your HTML/D3 code
- Include D3 in your HTML



```
<!DOCTYPE html>
<html lang="en">
    <head>
        <meta charset="utf-8">
        <title>D3 Page Template</title>
        <script type="text/javascript" src="d3/d3.v3.js"></script>
    </head>
    <body>
        <script type="text/javascript">
            // Your beautiful D3 code will go here
        </script>
    </body>
</html>
```

Include or directly place
Your javascript/D3 code here

Learning D3

- Before running D3 code, you need to start a local web server by doing the following:
  - Start a command line window
  - Change to the folder that you will place your HTML code
  - Run the following command

    python –m SimpleHTTPServer 8888 &
  - Open your browser, and type the address:

    http://localhost:8888
- Watch the online tutorial

  https://github.com/curran/screencasts/tree/gh-pages/introToD3
- Also start to read chapter 5 and follow the examples