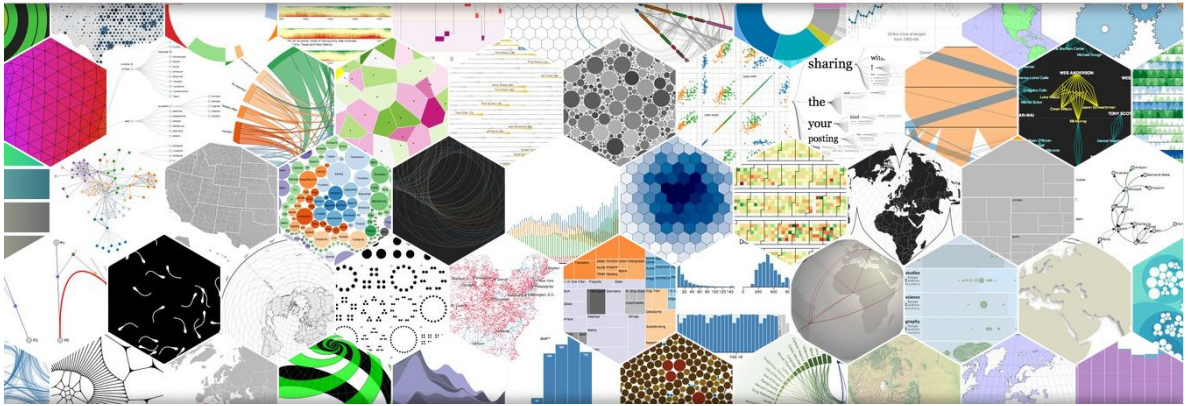
Data-Driven Documents



CSc 337
SVG and D3 Intro

Samantha Cox (referenced a lot from Dr. Levine's CSC 444 slides)

Announcements

- SCS Response Rate: 65.65%
- Final project due December 6th at 11:59pm
  - Don’t forget to include video.txt and live.txt in your submission!
- Final exam is December 13th from 3:30-5:30pm
- Feel free to send “instructor” questions to me (you can include Ben if you want to as well)
  - samtastic8@arizona.edu

What will we be talking about today?

- Intro to data visualization!
- We will be using HTML, CSS, and JS in combination with SVG and D3 to visualize data on a web application
- Interested in learning more? CSC 444

SVG: Scalable Vector Graphics

- XML Language
- “Vector graphics” refers to graphics drawn by points, lines, and curves defined by mathematical equations as opposed to pixels
- Allows us to draw graphical content in a procedural way
- Examples of SVG files

Using SVG in HTML

- Use the svg tag directly in the HTML:

```
<svg width="..." height="...">
    ... instructions with SVG elements ...
</svg>
```

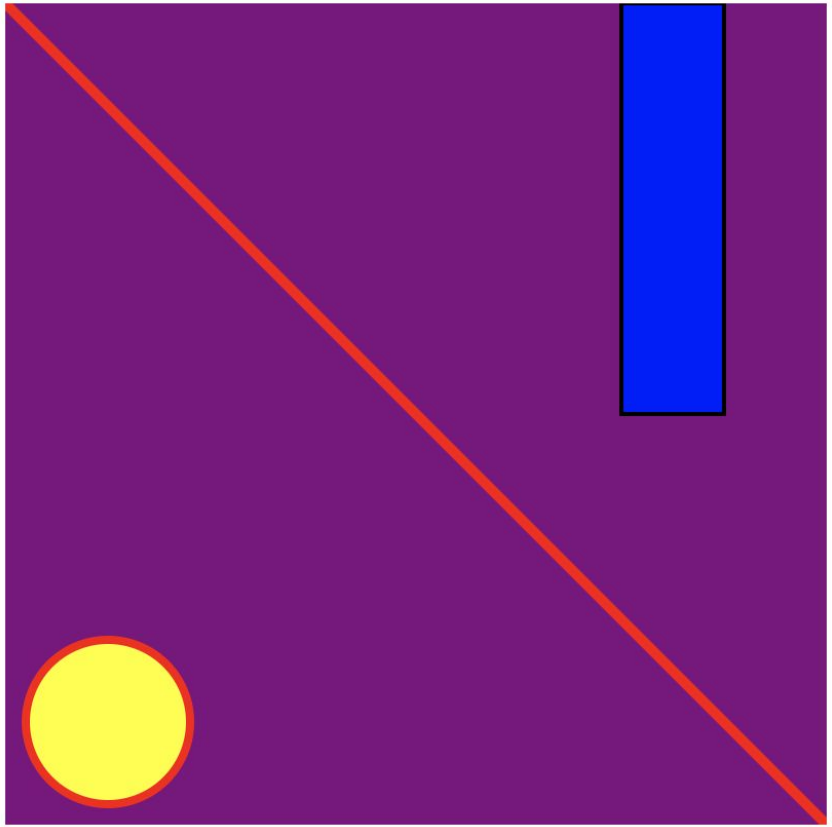
- Instructions are included in a set of tags and allow us to draw simple shapes (circles, rectangles, lines, text, etc)
- SVG elements

Drawing SVG Elements

- Instructions are applied one at a time
- New tags are drawn over existing ones
- Uses a 2D coordinate system to specify location of points for most drawings
    - Note: The top left corner is (0,0)

Drawing SVG Elements

- Let’s attempt to draw the following SVG canvas
- *Note: The canvas is the purple area, which is **400 x 400***

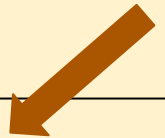


Activity

What is drawn on the SVG canvas?

```
... assume in <body> of HTML page ...
<svg id="canvas" width="500" height="500">
    <rect x="0" y="0" width="500" height="100" fill="gray" />
    <rect x="0" y="150" width="500" height="300" fill="red" />
    <circle cx="250" cy="300" r="100" fill="blue" />
</svg>
<script src="script.js"></script>
...
```

script.js



```
let canvas = document.getElementById('canvas');
let rectangles = "";
for (let i = 0; i < 500; i+=50) {
  rectangles += "<rect x='"+i+"' y='"+i+"' width='50' height='50'"+
  "stroke='black' />";
}
canvas.innerHTML += rectangles;
```

D3: Data-Driven Documents

- JavaScript library that allows us to design dynamic/interactive data visualizations for web applications
- Binds data and graphical elements to the DOM
  - Create, select, and modify
- Most recent version: d3.js

Selecting Elements

- `d3.select()` and `d3.selectAll()` accept a CSS selector and return elements that match the selector
  - No longer need to worry about `document.getElementById()`, `document.getElementsByClassName()`, etc.
- `.append()` can insert elements at the current selection

```
var selection = d3.select("...").append("...");
```

Setting Attributes

- With a selection
  - `.attr(attribute, value)` can be used set attributes
  - `.style(attribute, value)` can be used set CSS styles
- Both accept anonymous (lambda) functions

```
var svg = d3.select("div").append("svg")
            .attr("width", 500)
            .attr("height", 500);
```

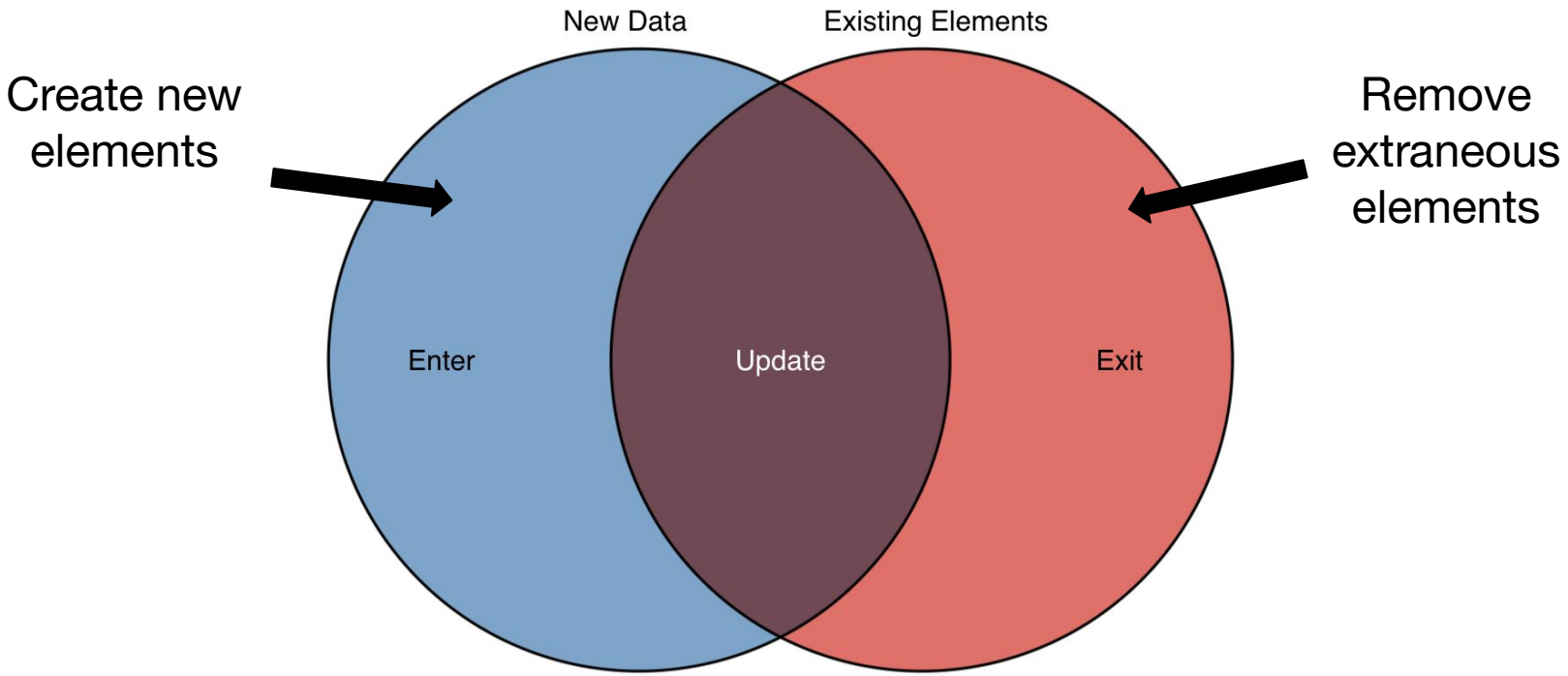
Binding Data

- With a selection, can bind data using `.data()`
- Creates a mapping between each element in the selection and each data element
- Once bound, we can define attributes for the data
  - ```
    .attr("stroke", function(d) {
        return "green";
    });
    ```
  - Sets the stroke color to `green` for all elements `d` in the selection
- `function(d, i)` allows you to access index `i` of data element `d`
    ```

What Happens to Data Selections?

- Three possibilities:
  - Update: ***same number*** of elements in selection and data
  - Enter: ***less*** elements in selection than in data
    - `.enter()` returns enter selection
    - `.append()` used to add new elements
  - Exit: ***more*** elements in selection than in data
    - `.exit()` returns exit selection
    - `.remove()` to remove existing elements

Data Join: Three Parts



https://observablehq.com/@d3/learn-d3-joins

General Update Pattern

- From https://bost.ocks.org/mike/join/:

```
var circle = svg.selectAll("circle")
              .data(data);

circle.exit().remove();

circle.enter().append("circle")
      .attr("r", 2.5)
      .merge(circle)
      .attr("cx", function(d) { return d.x; })
      .attr("cy", function(d) { return d.y; });
```

Make selection

Bind data

Remove extraneous elements

Create new elements

Update elements

Activity

What is drawn on the page?

```
var svg = d3.select("body").append("svg")
            .attr("width", 400)
            .attr("height", 400);

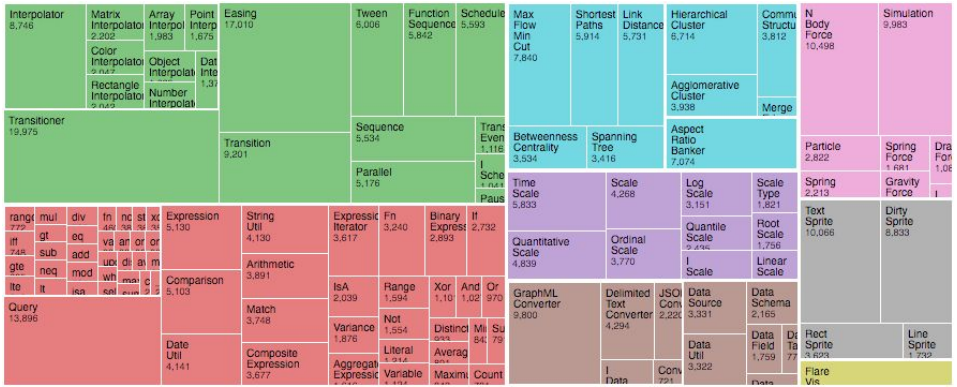
var rect = svg.selectAll("rect")
              .data(data);

rect.enter().append("rect")
    .attr("x", function(d) { return d.x; })
    .attr("y", function(d) { return d.y; })
    .attr("width", 30)
    .attr("height", 30)
    .attr("fill", "purple");
```

```
var data = [
  {x:50,  y:200},
  {x:200, y:100},
  {x:350, y:300}
];
```

Other Key Features of D3

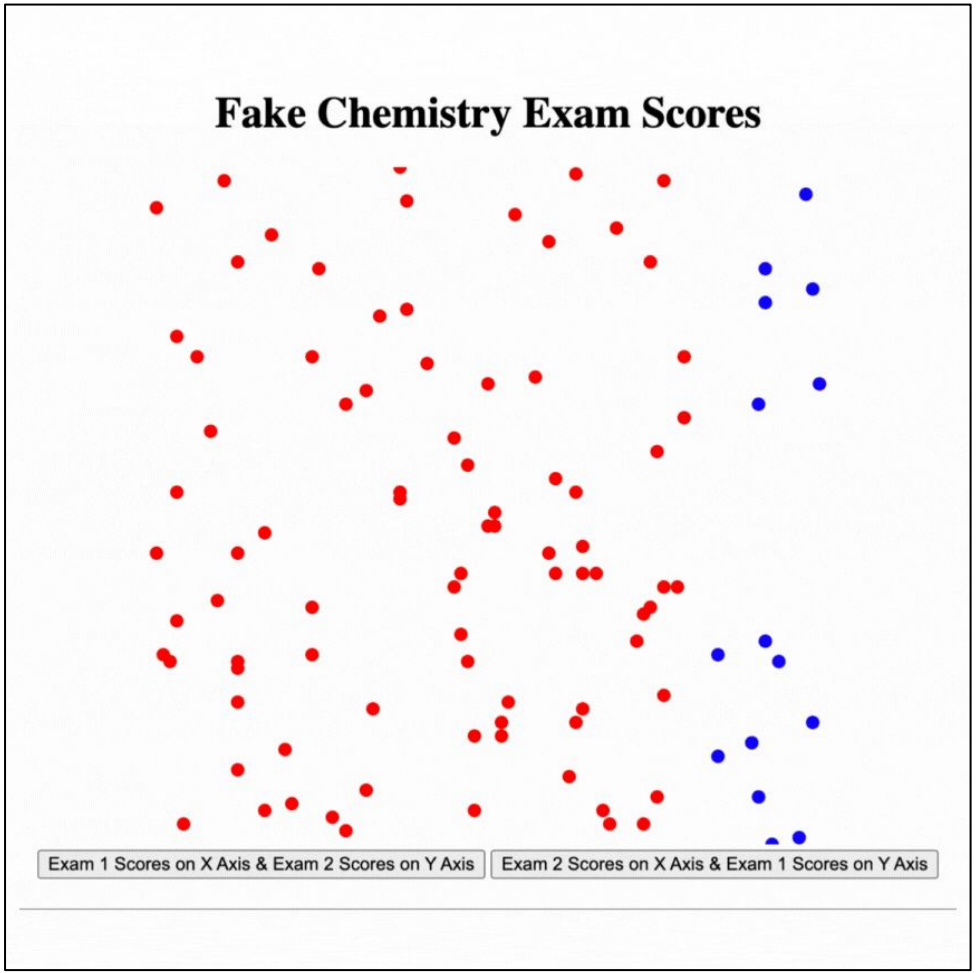
- Animate changes using `.transition()`
- Create scales that allow you to map data across different spaces
- Create complex charts like treemaps, packed circles, and geographic maps



D3 Demo

Can follow along with starter code!

Note: The data in this demo was randomly generated using https://json-generator.com/.



D3 Demo

Now with axes and scales!

Note: The data in this demo was randomly generated using https://json-generator.com/.