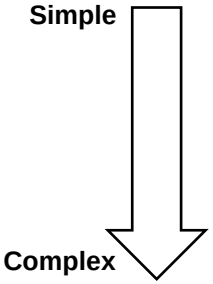
Lecture Handout
Week 15: Dec. 7 / Dec 9, 2015

Reminder: CS 105 Final Exam
Thursday, December 17, 1:30pm – 4:30pm
Rooms and TA Review Session TBA on CS 105 website

A major theme in CS 105 has been how you process data. **Data processing** can be done with several different tools, ranging from extremely simple to overwhelmingly complex:



	Tool	**Maximum Data Size** *Significant performance degradation after approximately:*
1.		
2.		
3.		
4.		

One of the most impactful ways to share the results of your data processing is through **data visualization**.

- Excel **provides** basic data visualization through simple charts: bar charts, pie charts, line charts, and their variations.
- Excel **limits** your ability to create a custom visualization.

- d3.js is a _____.
- d3.js requires **explicit instructions** to draw each element of a visualization: every line, box, arc, text and circle corresponds to a line of source code.

Terminology: A **JavaScript library** is JavaScript code that:

1.

2.

3.

Every d3.js visualization is made up of three major components:

Component #1: _____
- The processed data you intend to visualize
- Nearly always an array of objects or a format that is easily converted to an array of objects (eg: CSV exported from Excel)

Component #2: _____
- JavaScript source code that sets up the visualization in a standard way; often copied/pasted between projects.
- Will be provided in CS 105.

Component #3: _____
- JavaScript source code that draws the individual points of data on the visualization.
- Every line, box, arc, text, and circle corresponds to a series of JavaScript statements that **render the visualization**.

Snippets of each of these three components are shown here:

```
 1  var games = [
 2    { score: [4, 1],  opponent: "Oakland" },
 3    { score: [1, 0],  opponent: "Illinois State" },
 4    { score: [5, 2],  opponent: "TCU" },
..  ...
15  ];
16
17  var margin = { top: 50, right: 0,
                   bottom: 100, left: 150 },
18      width = 3000 - margin.left - margin.right,
19      height = 3200 - margin.top - margin.bottom;
20
21  var svg = d3.select("#chart")
22              .append("svg")
23              .attr("width", width + margin.left
                                     + margin.right)
25              .attr("height", height + margin.top
                                       + margin.bottom)
..  ...
76  svg.selectAll()
77     .data( games )
78     .enter()
79     .append("rect")
80     .attr("x", function (d, i) { return d.score[0]; })
81     .attr("y", function (d, i) { return d.score[1]; })
..  ...
```

The code on lines 76-81 is central to the third component, actually drawing the visualization. All d3.js visualizations will follow the same pattern of three steps:

Step #1: Select the area of the svg (using `.selectAll`), pass the data that is going to be visualized (using `.data(games)`, where `games` is the array of objects that will be visualized), and begin processing that data (using `.enter`).
…as a general rule, this code will always be the same except changing out `games` with the variable name that contains your data:

```
76  svg.selectAll()
77      .data( games )
78      .enter()
```

Step #2: Select the shape you want to draw about each point in your dataset. In CS 105, we will cover rectangles (`"rect"`) and circles (`"circle"`).

```
79      .append("rect")
```

Step #3: Add data-point specific attributes about each data point from your data set about how you want it to appear on the visualization. This is done via the `.attr` function, which takes in two arguments:

- The name of the attribute (eg: `"x"`)
- A function that computes the value of the attribute based on the data point. This function has two arguments itself:
  - `d`: Your data point, a single element in your data array
  - `i`: The index your data point appears in the data array

```
80      .attr("x", function(d, i) { return ____________; })
81      .attr("y", function(d, i) { return ____________; })
..  ...
```

Step 3 – Example: In our example, a single element of our data is:

```
 2    { score: [4, 1],  opponent: "Oakland" },
```

Therefore, a valid function would be anything that returns information from that JavaScript object:

```
      function (d, i) { return d.score[0]; }  // returns 4
      function (d, i) { return d.score[1]; }  // returns 1
      function (d, i) { return d.opponent; }  // "Oakland"
```

Attributes that are applied to a rectangle (`"rect"`):

Attribute Name	Value

Attributes that are applied to a rectangle (`"circle"`):

Attribute Name	Value

Style attributes that can be applied to either a rectangle or a circle (using `.style` instead of `.attr`)

Attribute Name	Value

MPx: An Extra Credit MP!
Released on Monday, Dec. 7; due before the start of the final exam.
Allows the replacement of a single MP (or +5 points in CS 105).

Reminder: CS 105 Final Exam
Thursday, December 17, 1:30pm – 4:30pm
Rooms and TA Review Session TBA on CS 105 website