Data Driven Documents

Kevin Lynagh

A talk for the PDX Data Visualization group, 19 October 2011

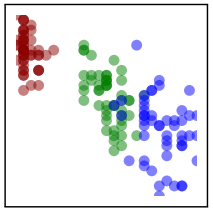
Slides & notes: `http://keminglabs.com/talks/d3_pdx`
Email: `kevin@keminglabs.com`
Office: 888.502.1042

Overview

D3 is a JavaScript library that binds arbitrary data to HTML & SVG elements, allowing you to build rich, interactive visualizations on the web. Rather than try to serve as a monolitic library supporting every visualization imaginable, D3 simply provides a lightweight, data-driven interface to the web's underlying HTML5, CSS3, and SVG technologies. Developers can use D3 immediately with their current development tools and knowledge of web standards.

Anatomy of a scatterplot

A scatterplot of a dataset is just a collection of dots (one for each datum) with horizontal and vertical positions determined from the data. The code listing on the right draws a scatterplot of the famous Iris dataset collected by the botanist Edgar Anderson. The plot of sepal length vs. petal width is given below, with a subset of the data (in centimeters):



species	sepal		pedal	
	width	length	width	length
setosa	5.3	3.7	1.5	0.2
versicolor	6.4	3.2	4.5	1.5
virginica	6.2	3.4	5.4	2.3

```
var iris = [{species:"setosa",
             sepal_width:5.3, sepal_length:3.7,
             pedal_width:1.5, pedal_length:0.2}
            //and so on...
           ], width = 800;

var scales = {};
d3.keys(iris[0]).forEach(function(dim){
  var values = function(x){return x[dim];};
  scales[dim] = d3.scale.linear()
    .domain([d3.min(iris, values),
             d3.max(iris, values)])
    .range([0, width]);
});

d3.select("#a_scatterplot")
  .append("svg:svg")
  .attr("width", width).attr("height", width)
  .style("border", "2px solid black")

  .selectAll("circle").data(iris)
  .enter().append("svg:circle")
  .attr("class", "little")
  .attr("r", 12)
  .attr("opacity", 0.5)
  .attr("cx", function(d){
    return scales.petal_width(d.petal_width);
  })
  .attr("cy", function(d){
    return scales.sepal_length(d.sepal_length);
  })
  .attr("fill", function(d){
    switch(d.species){
     case "setosa": return "darkRed";
     case "virginica": return "blue";
     case "versicolor": return "green";
    }});
```

Initial data array and variable setup. It is common to use JSON, but D3 includes helpers for loading CSV, XML, and text files.

Generate a linear scale for each dimension. This is a JavaScript function that maps from the data domain (the min and max of, e.g., *petal width*) to the visualization domain (e.g., 0–800 px).

Create an SVG element, append it to the node that matches the CSS selector `#a_scatterplot`, and use its `style` attribute to add a border.

Select all of the `circle` elements, and bind them to the elements of the array `iris`. For each new datum, add a `circle` element; set its `radius` and `opacity` attributes to constants, set `cx` & `cy` according to the *petal width* and *sepal length* scales, and color it according to *species*.