Data Visualization in the Browser

Rob Larsen

2013.4.1

Download (https://github.com/roblarsen/dataviz)

What we're going to talk about

- About me
- SVG & Canvas
- Libraries like Raphael, D3, and Highcharts

Me

Work

- I've been making web sites since 1997
- I've been primarily focused on HTML, CSS & JavaScript that whole time
- Formerly an agency guy/consultant. Lots of big brand stuff (Adidas, Motorola, Samsung, etc.)
- Nowadays I'm the client & I come with with 100% more suits

Seriously, suits

Work

- I'm roblarsen on Github
- I'm @robreact on Twitter
- I have a blog @ htmlcssjavascript.com
- I wrote this book. It just came out.
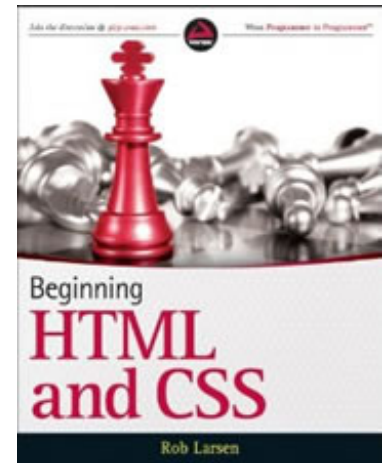
Art

DrunkenFist.com and (coming soon) Java+++

Data Visualization?

Rhymes with crufty. Photo from Andreas_MB on Flickr

Basically

Taking this

{"username": "rob_react", "weeks": {"1152446400": {"Pink": 2, "The Dece
mberists": 3, "Jerry Jeff Walker": 7, "The Clash": 2, "Jim Carroll Band
": 2, "Big Audio Dynamite II": 1, "The Beatles": 12, "Cheap Trick": 1,
"Rod Stewart": 3, "Ninja Tunes": 1, "Nas": 1, "Boogie Down Productions"
: 1, "Everclear": 8, "DANGERDOOM": 2, "Paul Simon": 1, "Macy Gray": 1,
"Jem": 1, "Terror Danjah Ft. Riko, Bruza, D Double E And Hyper": 1, "Ke
lly Willis": 1, "Herman's Hermits": 1, "Tool": 1, "Sound Tracks": 1, "T
he Cure": 1, "Belle and Sebastian": 9, "Tom Waits": 2, "Nick Drake": 1,
"Jammer Ft. Wiley, D Double E, Kano and Goodz": 2, "Oasis": 4, "Acid H
ouse Kings": 2, "Ralph Stanley & Friends": 1, "Joe Jackson": 1, "The Li
bertines": 7, "Wilco": 1, "Sublime": 5, "The Cardigans": 1, "Bill Hicks
": 3, "Harry Nilsson": 1, "Operation Ivy": 1, "Bob Dylan": 11, "Joni Mi
tchell": 1, "Led Zeppelin": 1, "50 Cent": 1, "Black Crowes & Wilco": 1,
"Kano": 1, "Mason Jennings": 7, "Dave Matthews Band": 1, "The White St
ripes": 1, "Michael Penn": 4, "Ralph Stanley": 1, "Kirsty MacColl / The
Pogues": 1, "Billy Bragg & Wilco": 2, "Misfits": 3, "The Pretenders":
2, "The Pogues": 12, "Sangue Misto": 1, "Rufus": 1, "The Beach Boys": 2

And turning it into this
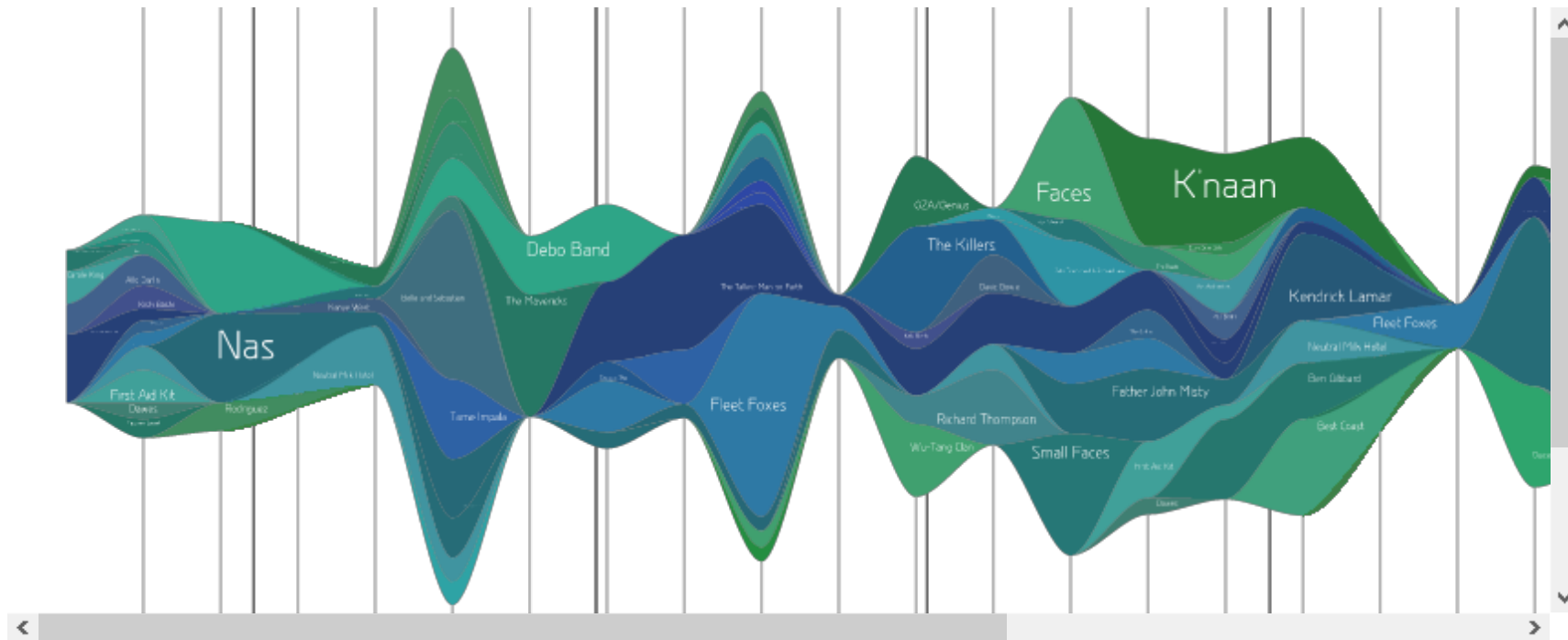
Generated with the awesome http://lastgraph3.aeracode.org/

The Technologies

Scalable Vector Graphics (SVG)

SVG as a "new" thing

SVG has had a long and strange journey to the world of "emerging technologies." SVG was actually defined in 1999, so it's not exactly the new kid on the block.

Still, it took a long while to catch on, so I don't get bent out of shape when people lump it in with "HTML5 and Friends."

What is SVG

SVG is vector graphics mixed up with XML.

When created in the context of an HTML document SVG elements are funky DOM elements-- properties are stored as part of the regular DOM and access to individual elements is available using traditional DOM access methods like `document.geElementById` and `document.getElementsByTagName` (or, if you only speak jQuery `$()`)

It can also be styled with CSS.

SVG is better than sliced bread

SVG can also be output from a vector based drawing program and used as the `src` of an `<img>` element or as the background of an element as defined by CSS.
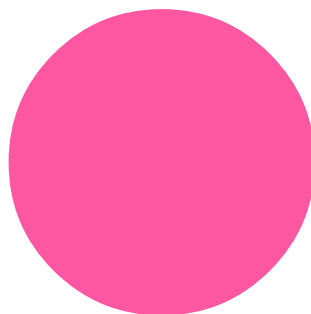
Vector means: it scales!

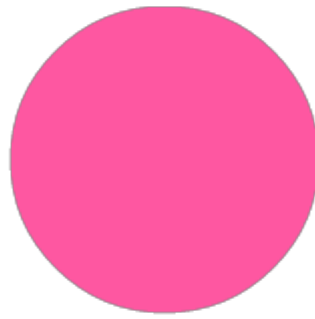# It looks like this

Viewing Source

```
<svg width="200" height="200"
    viewPort="0 0 200 200" version="1.1"

    xmlns="http://www.w3.org/2000/svg">

    <circle cx="100" cy="100" r="75" fill="#fe57a1"/>

</svg>
```

It can also look like this

Viewing Source

```
<img src="img/circle.svg">
```

Support

| 5.5 | 6 | 7 | 8 | 9 | 10 |

| 2 | 3 | 3.5 | 3.6 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 |

| 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 |

| 3.1 | 3.2 | 4 | 5 | 5.1 | 6 |

| 9 | 9.5-9.6 | 10.0-10.1 | 10.5 | 10.6 | 11 | 11.1 | 11.5 | 11.6 | 12 | 12.1 | 12.5 |

| 3.2 | 4.0-4.1 | 4.2-4.3 | 5.0-5.1 | 6 |

| 5.0-7.0 |

| 2.1 | 2.2 | 2.3 | 3 | 4 | 4.1 | 4.2 |

| 7 | 10 |

| 10 | 11 | 11.1 | 11.5 | 12 | 12.1 |

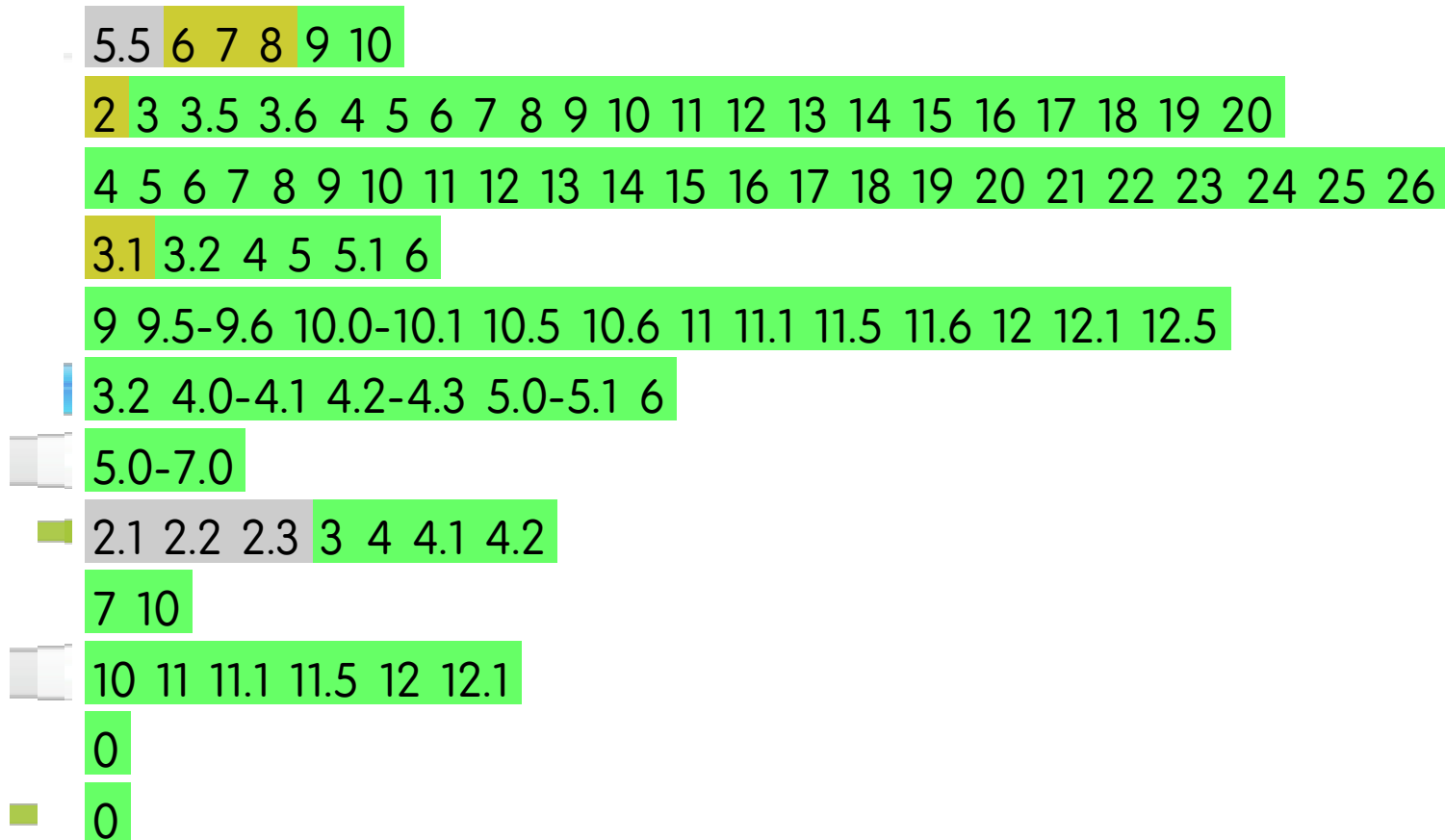| 0 |

| 0 |

data from caniuse.com

Canvas

What's Canvas?

The canvas element and associated API started life as an Apple extension to HTML. From there it blossomed into one of the early stars of the HTML5 era.

The canvas element provides a scriptable interface for drawing two-dimensional images in the browser. Think... dynamic PNGs. Even without full browser support on the desktop, developers have embraced canvas fully.

Stay in control

It's an extremely low level API. This means you often have to do a bit of work but you have complete, pixel level control.

It looks like this

Viewing Source

```
<canvas id="circle" width="800" height="300"></canvas>
```
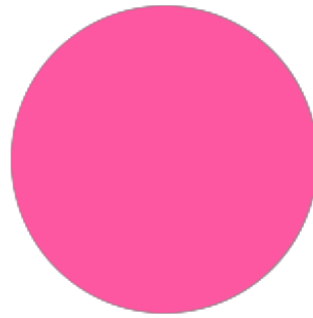
```javascript
    var ctx = document.getElementById( "circle" ).getContext( "2d" );
    ctx.beginPath();
    ctx.arc( 400, 150, 75, 0, Math.PI*2, true );
    ctx.fillStyle = "#fe57a1";
    ctx.closePath();
    ctx.fill();
```
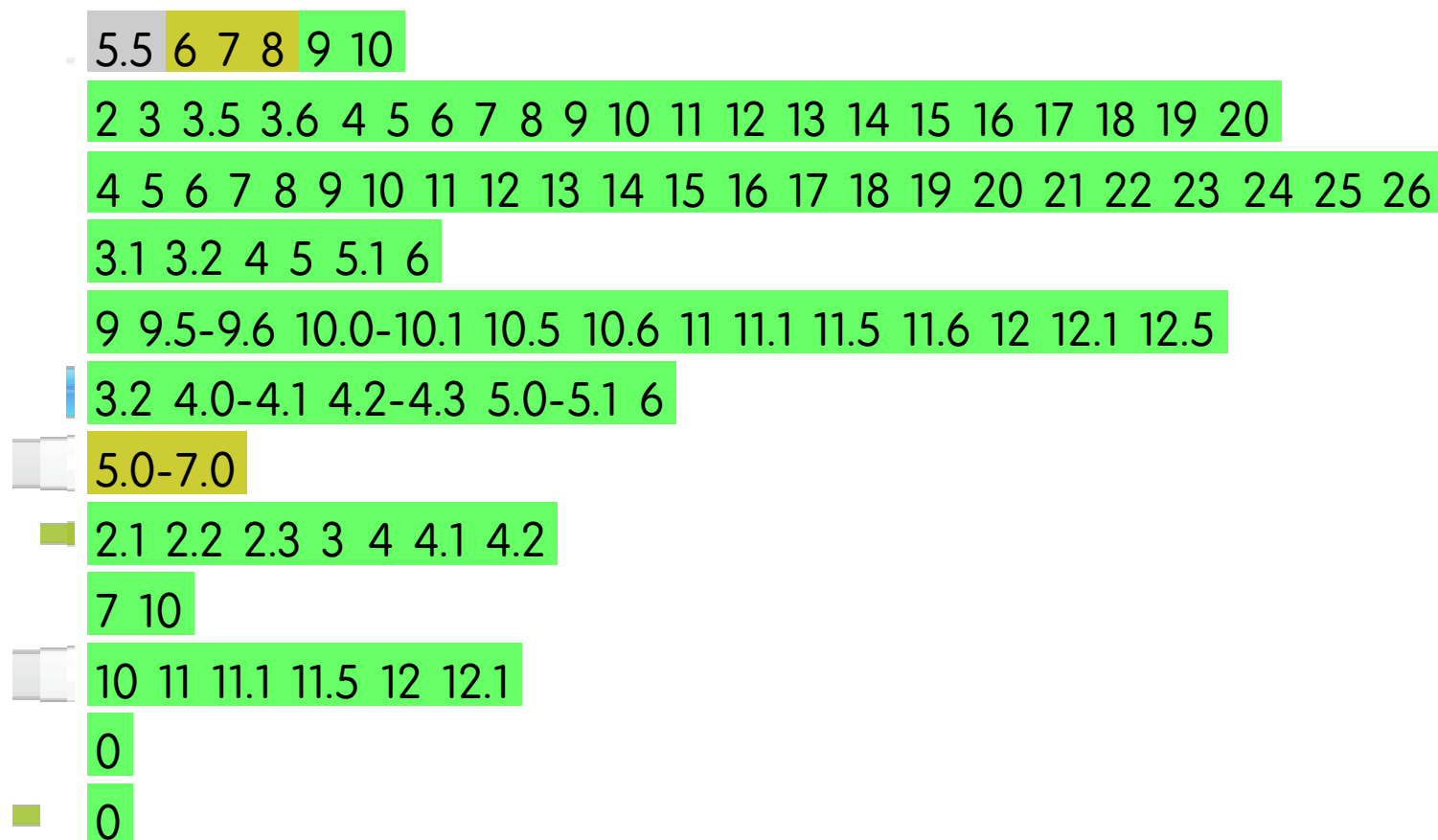
Did I say low level API?

I sure did. Everyone knows `arc` is all you need when you have `Math.PI*2`

```
ctx.arc( 400, 150, 75, 0, Math.PI*2, true );
```

Not the friendliest API

Support

| 5.5 | 6 | 7 | 8 | 9 | 10 |

| 2 | 3 | 3.5 | 3.6 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 |

| 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 |

| 3.1 | 3.2 | 4 | 5 | 5.1 | 6 |

| 9 | 9.5-9.6 | 10.0-10.1 | 10.5 | 10.6 | 11 | 11.1 | 11.5 | 11.6 | 12 | 12.1 | 12.5 |

| 3.2 | 4.0-4.1 | 4.2-4.3 | 5.0-5.1 | 6 |

| 5.0-7.0 |

| 2.1 | 2.2 | 2.3 | 3 | 4 | 4.1 | 4.2 |

| 7 | 10 |

| 10 | 11 | 11.1 | 11.5 | 12 | 12.1 |

| 0 |

| 0 |

data from caniuse.com

Comparing Canvas & SVG

At a High Level

Generally... Go SVG when dealing with a limited number of elements, lower animation requirements and higher interactivity needs. Because it's got traditional access methods and traditional event registration, it's more familiar to the average front end engineer. It's much easier to implement in an accessible way.

Two big SVG libraries are also polyfilled by default (Raphaël and Highcharts.) That's super cool.

Canvas Strengths

Canvas is better for applications that require intense animation, real-time interaction and / or many more elements. SVG elements are DOM elements so thousands of them is kind of scary. Thousands of "elements" in the canvas is still just the one element being manipulated pixel by pixel.

megaDOM

Of course, things can bog down with Canvas too- it just won't be because you've got a MEGADOM.

Scaling matters

One important note: If different resolutions and form factors are the kind of thing that keeps you up at night, make nice with SVG. SVG is like a miracle cure for what ails you. Canvas is bitmapped. Bitmaps don't scale. Vector scales all day.

Your Mileage May Vary

This issue could be a whole presentation all by itself. It's a nuanced topic. To help, here are a couple of articles which cover this question in greater depth:

- How To Choose Between SVG and Canvas
- How to Choose Between Canvas and SVG for your Site

In Practice

What follows are the tools and approaches I've used on projects over the past few years and can speak about with some authority. While these aren't the only solutions to these problems, these are ones that have been successful for me.

Focus

The focus here is the intersection between ease of use, browser compatibility and power.

SVG: Highcharts

"Highcharts is a charting library written in pure JavaScript, offering intuitive, interactive charts to your web site or web application. Highcharts currently supports line, spline, area, areaspline, column, bar, pie, scatter, angular gauges, arearange, areasplinerange, columnrange and polar chart types."

SVG: Highstock

"Highstock lets you create stock or general timeline charts in pure JavaScript, including sophisticated navigation options like a small navigator series, preset date ranges, date picker, scrolling and panning."

These are commercial products, but they're really great.

I almost always have a budget. So for me, any free solution would have to be insanely good. YMMV.

SVVG:Highcharts +-

- Pluses:
  - Super powerful
  - Well Documented
  - Old IE support via VML
  - Rock solid
- Minuses:
  - Not free

Viewing Source

```
var seriesOptions = [],
    yAxisOptions = [],
    seriesCounter = 0,
    names = ['AAPL', 'BBRY'],
    colors = Highcharts.getOptions().colors;
$.each( names, function( i, name ) {
  $.getJSON('../data/'+names[i].toLowerCase()+'.json', function( data )
 {
    for (var j = 0, len = data.length; j < len; j++) {
      data[j][0] = Date.parse( data[j][0] ).getTime();
    }
    data = data.reverse();
    seriesOptions[i] = {
       name: name,
       data: data
    };
```

SVG: D3

"D3.js is a JavaScript library for manipulating documents based on data. D3 helps you bring data to life using HTML, SVG and CSS. D3's emphasis on web standards gives you the full capabilities of modern browsers without tying yourself to a proprietary framework, combining powerful visualization components and a data-driven approach to DOM manipulation."

data from the Hubway Data Visualization Challenge

SVG: D3+-

- Pluses:
  - Super powerful
  - Under active development

  - Many examples
- Minuses:
  - No support for old IE

View Source

```
[
  [2689, 508, 1170, 189, 1007, 187,  745,  248,  263,  2311],

  [1064, 121, 830,  323, 2473, 393,  453,  312,  533,  599],
  [506,  296, 813,  530, 988,  540,  1936, 578,  747,  268],
  [706,  311, 1568, 526, 1273, 371,  618,  694,  481,  227],
  [178,  701, 277,  176, 663,  227,  379,  284,  330,  111],
  [550,  270, 548,  445, 196,  769,  868,  317,  1477, 195],
  [344,  141, 468,  955, 172,  346,  502,  388,  415,  97],
  [333,  207, 455,  545, 196,  1322, 618,  254,  659,  62],
  [655,  120, 301,  90,  2368, 108,  226,  99,   229,  875],
  [270,  221, 625,  436, 239,  278,  548,  1158, 320,  90]]
```

View Source

```
//from bl.ocks.org/mbostock/4062006 etc.
var chord = d3.layout.chord()
    .padding( .05 )
    .sortSubgroups( d3.descending )
    .matrix( matrix );
var width = 900,
    height = 500,
    innerRadius = Math.min( width, height ) * .35,
    outerRadius = innerRadius * 1.1;

var fill = d3.scale.ordinal()
            .domain( d3.range(4) )
            .range(["#336699", "#99ccff", "#6699cc", "#0066cc"]);

var svg = d3.select( "body" )
            .append( "svg" )
```

SVG: Raphaël

"Raphaël is a small JavaScript library that should simplify your work with vector graphics on the web. If you want to create your own specific chart or image crop and rotate widget, for example, you can achieve it simply and easily with this library."

Background photo from stintje on flickr

SVG: Raphaël +-

- Pluses:
  - Fun, intuitive API
  - Built-in OLD IE support via VML
  - Lower Level API
- Minuses:
  - Lower Level API
  - Hopefully not dead, just resting:

View Source

```
var paper = Raphael( 0, 0, 900, 550 );
var alpe = paper.path( "M71.5,538.641l29.937-15.665l30.633-14.271l30.98
1-16.014 l31.329-14.968l30.633-12.88l30.981-10.791l29.937-10.443l32.026
-12.184l29.936-11.835l30.285-11.488l30.633-10.442l30.981-12.88 l29.937-
11.488l30.633-13.924l30.285-10.792L562.672,335l29.937-11.835l31.329-9.7
47l29.937-12.88l32.374-11.487l30.285-12.879  l30.285-11.835l29.938-12.1
84l29.937-11.836l31.678-10.095L868.307,225h13.576  L886,538 L71.5,538"
)
    .attr( 'fill' , 'url(../slides/img/alpe.jpg)' );

function climb( name, time, hex, endpos ){
   var rider = paper.text( 10, 10, name )
                    .attr( "fill", hex )
                    .attr( "font-size", "14px" );
   rider.animateAlong({
    path: "M71.5,538.641l29.937-15.665l30.633-14.271l30.981-16.014 l31.
```

SVG: Other Libraries

- SVG.js
  "a lightweight JavaScript library for manipulating and animating svg using an uncluttered syntax"
- Polymaps
  "a free JavaScript library for making dynamic, interactive maps in modern web browsers."

Canvas: Core Canvas

"This specification defines the 2D Context, Level 2 for the HTML canvas element. The 2D Context provides objects, methods, and properties to draw and manipulate graphics on a canvas drawing surface."

Canvas: Core Canvas +-

- Pluses:
  - Not a library
  - POWERFUL
- Minuses:
  - Lower level API

View Source

```
function draw( data ) {
  ctx.fillRect( 0, 0, 900, 300 );
  var len = data.length,
      width = 900/len,
      gradient = ctx.createLinearGradient( 0, 0, 0, 300 );
      gradient.addColorStop( ".5", "#003366" );
      gradient.addColorStop( "1.0", "#ccff00" );
      ctx.fillStyle = gradient;
  for ( var i=0; i<len; i++ ) {
      ctx.fillRect( i, 300, width, -data[i] );
  }
}
```

Canvas: Canvas.js

"CanvasJS is a small helper library for the Canvas 2d API. The goal is to extend and enhance the basic API while still remaining familiar"

I'm partially to blame (also, yay @bobholt and @marcneuwirth)

Canvas: Canvas.js +-

- Pluses:
  - Familiar API, just enhanced
  - Tiny (4k gzipped)
  - It's got a really nice personality
  - Chaining
- Minuses:
  - Full of youthful exuberance
  - Remains low level
  - Not a fancy game developer tool

View Source

```
function draw(data) {
  ctx.reset();
  var len = data.length,
      width = 900/len,
      fill;
  for ( var i=0; i<len; i = i + 10 ) {
    var dataLen = data[i];
      for ( var j = 0; j < dataLen; j = j+10 ){
        fill = "rgb("+j+","+(255-j)+",255)";
        ctx.fillCircle({
          x : i,
          y : 300 - j,
          radius : 4,
          fillStyle : fill
        });
      }
```

Canvas: JavaScript InfoVis Toolkit

"The JavaScript InfoVis Toolkit provides tools for creating Interactive Data Visualizations for the Web."

Canvas:JIT +-

- Pluses:
  - Strong foundation
  - Straightforward API
- Minuses:
  - Out of the box visualizations are skeletal

View Source

```
icicle = new $jit.Icicle({
    injectInto : 'infovis',
    animate : animate,
    offset : 1,
    cushion : false,
    constrained : true,
    levelsToShow : 4,
    Tips : {
      enable : true,
      type : 'Native',
      offsetX : 20,
      offsetY : 20,
      onShow : function(tip, node){
        var count = 0;
        node.eachSubnode(function(){
          count++;
```

Canvas: Other Libraries

- **sigma.js**
  "an open-source lightweight JavaScript library to draw graphs, using the HTML canvas"

- **paper.js** "Scriptographer ported to JavaScript and browser, using HTML5 Canvas"

- **Peity** "progressive <canvas> piecharts"

- **arbor.js** "a graph visualization library using web workers and jQuery"

- **envision** "a library for creating fast, dynamic and interactive HTML5 visualizations.

- **CanvasQuery** "use HTML5 Canvas with jQuery syntax"

- **Processing.js**
  "the sister project of the popular Processing visual programming language"

Bonus: Google Maps

"The Google Maps Javascript API lets you embed Google Maps in your own web pages. Version 3 of this API is especially designed to be faster and more applicable to mobile devices, as well as traditional desktop browser applications. "

About Google Maps

The Maps API is already rich, adding a layer of data on top of it opens up a wide range of possibilities
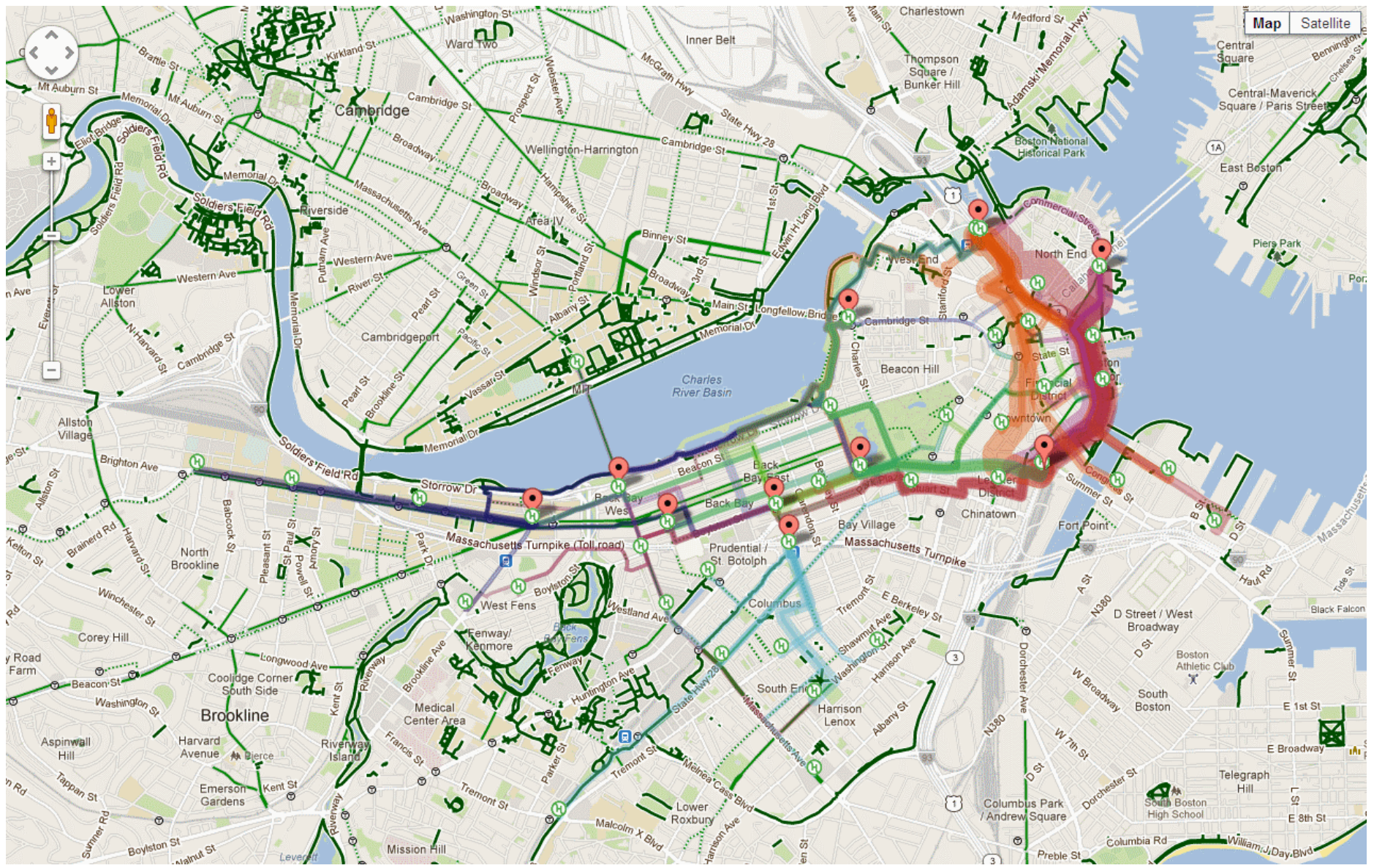
Google Maps +-

- Pluses:
  - Powerful, well documented API
- Minuses:
  - If you get really popular it costs $$

View Source

```
;( function( window, document, $, undefined ) {
  "use strict";
  if( window.HW === undefined ) {
    window.HW = {};
  };
  var HW = window.HW;
  HW.common = {
    init: function() {
      $( "#map" ).height( $( window ).height() );
      var GM = google.maps,
        defaultPosition = new GM.LatLng( 42.3520, -71.0560 ),
        mapOptions = {
          zoom: 12,
          center: defaultPosition,
          mapTypeId: GM.MapTypeId.ROADMAP
        },
```

My Columbo Moment

just one more thing...

Datavisualization.ch Selected Tools

A list of all these tools, plus more.

Thanks!