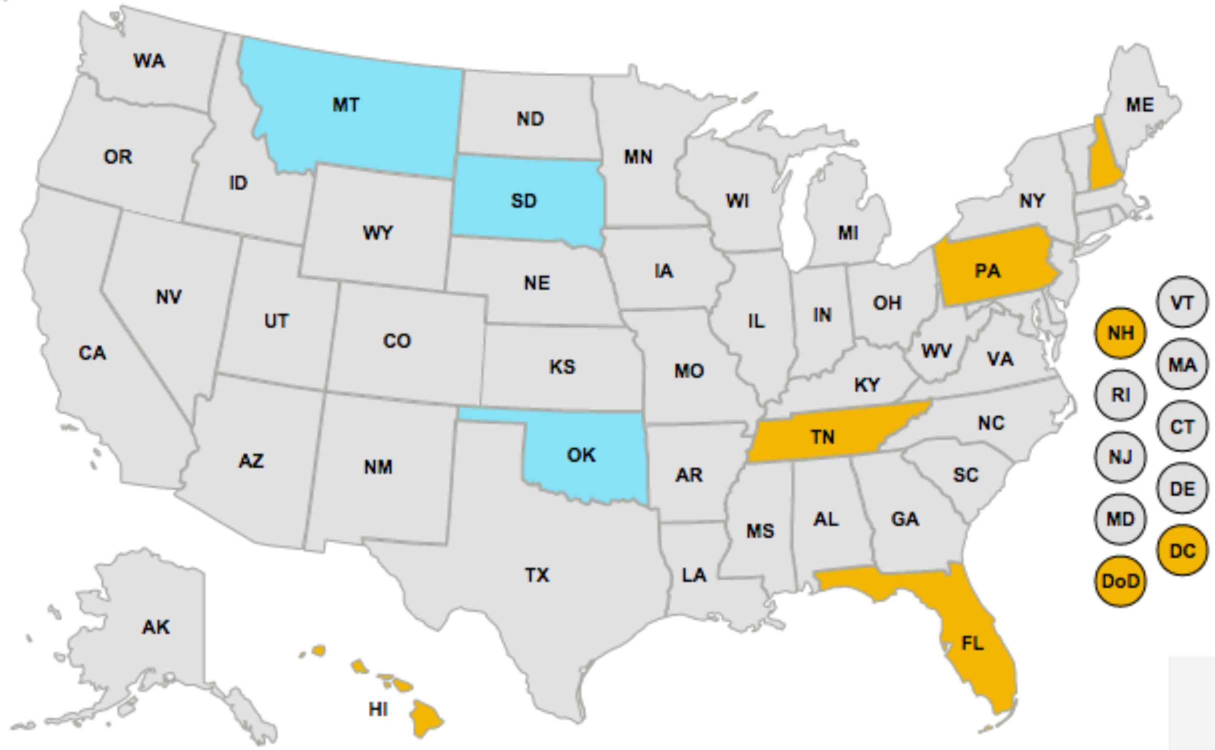
D3 Data Visualization

USE CASES

HOME
SHOPPING
CONTACT
SIGN UP
LOG IN
Development Diversity
Fullwidth
Boxed

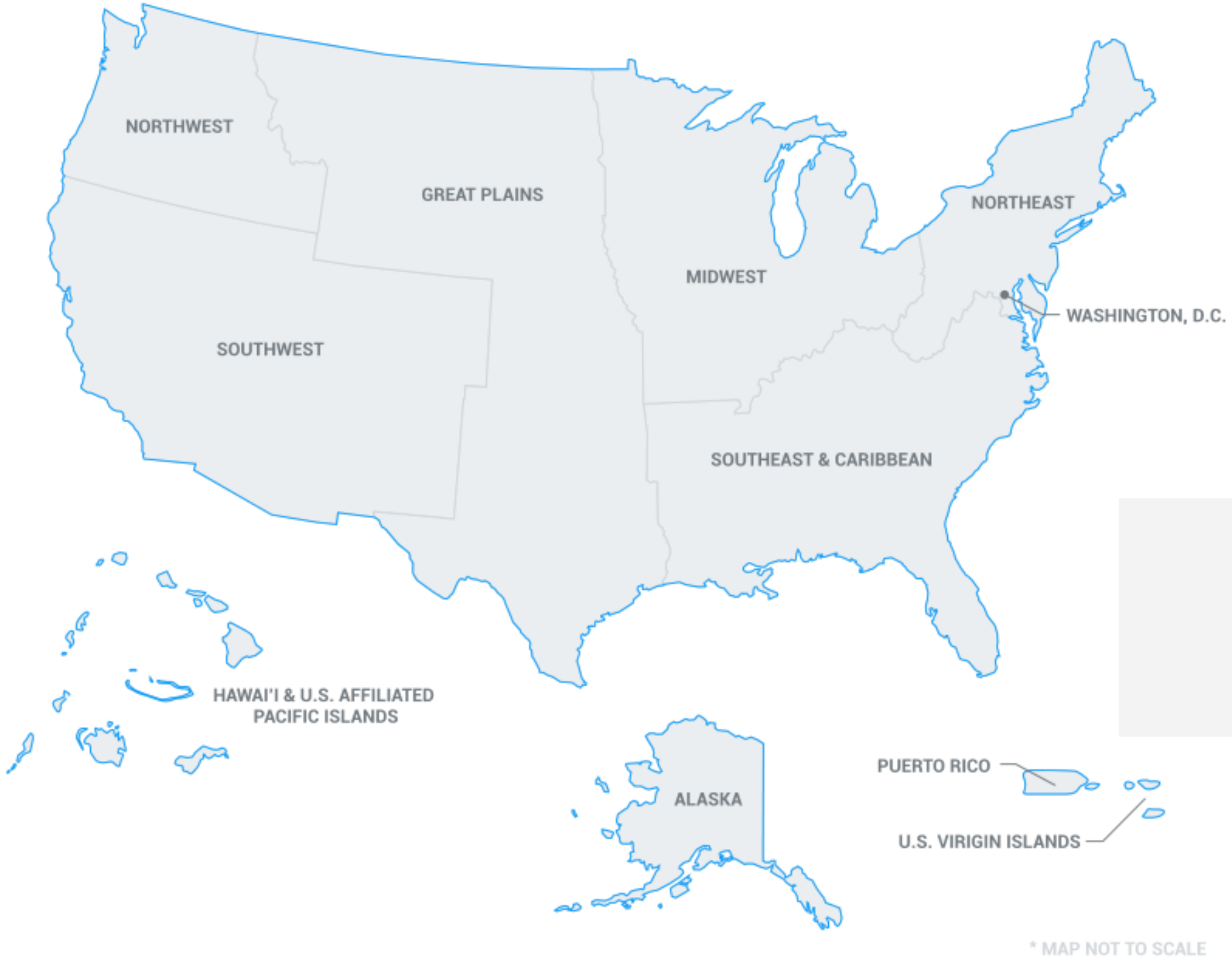


NOTE: DoD = Department of Defense Education Activity (DoDEA).

🟧 Gain 🟩 No change 🟦 Loss ⬜ Not available/applicable

Nation's Report Card.gov

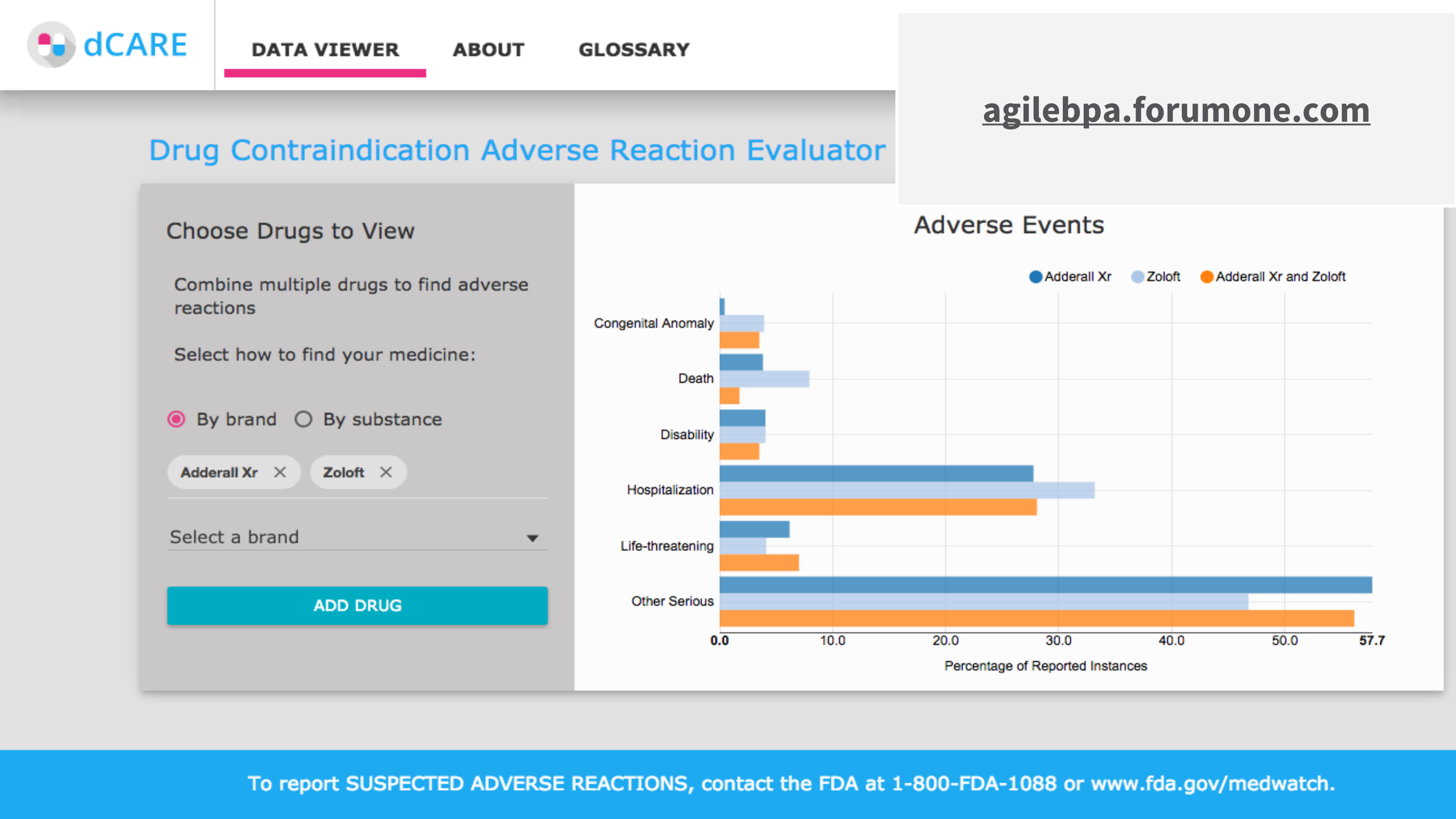
Shapes: Natural Earth
Projection: Albers USA



GlobalChange.gov

Shapes: custom

agilebpa.forumone.com



Drug Contraindication Adverse Reaction Evaluator

Choose Drugs to View

Combine multiple drugs to find adverse reactions

Select how to find your medicine:

[✓ By brand] By substance

To report SUSPECTED ADVERSE REACTIONS, contact the FDA at 1-800-FDA-1088 or www.fda.gov/medwatch.

Overview

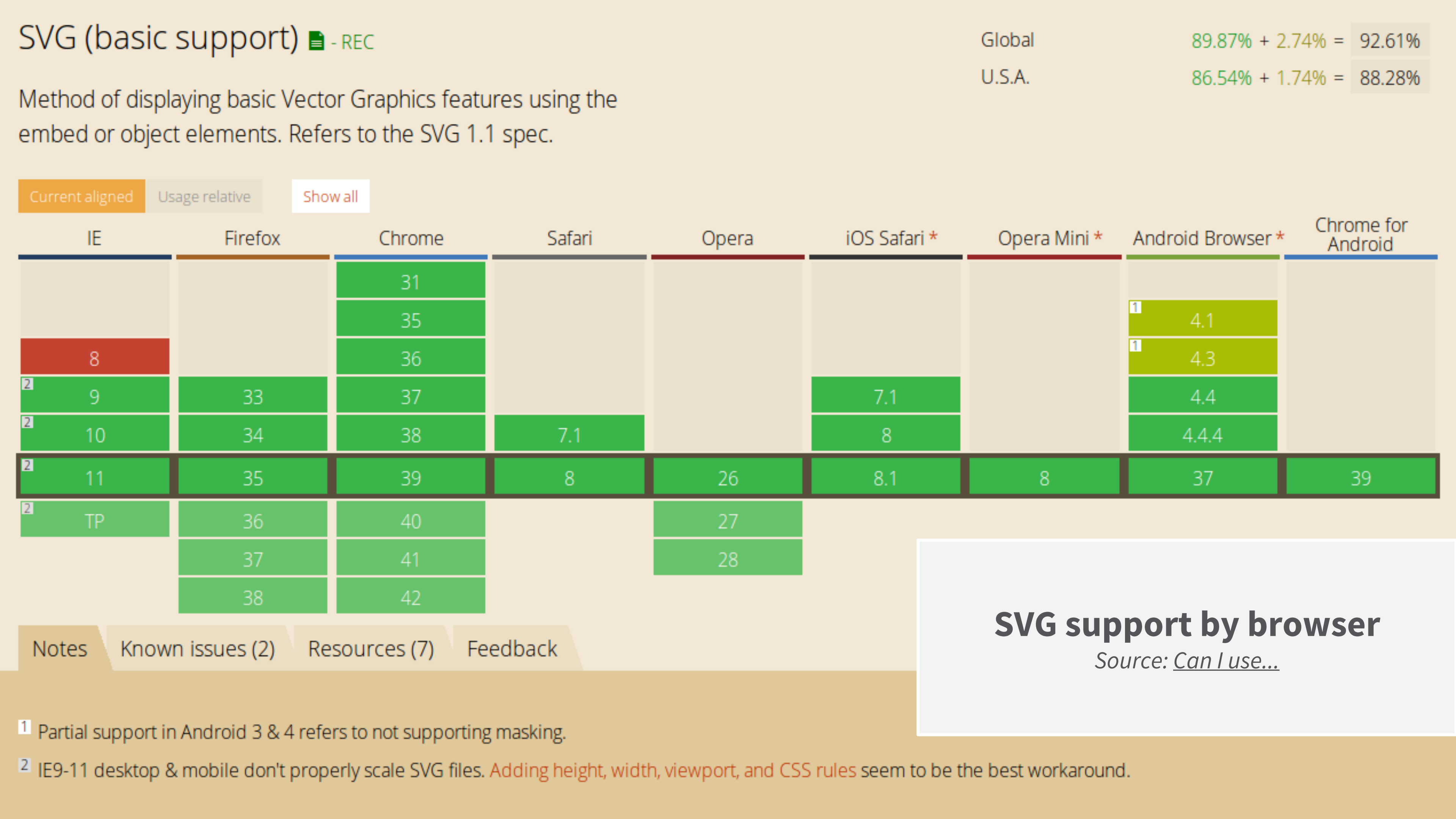
1. Scalable Vector Graphics (SVG)
2. D3 Overview
3. D3 Foundation
4. Drupal and D3 Module
    a. Views with Basic D3 Module Libraries
    b. Views with Custom D3 Module Libraries
5. Custom D3 WITHOUT D3 Module

Scalable Vector Graphics (SVG)

Scalable Vector Graphics (SVG) is an XML-based vector image format for two-dimensional graphics with support for interactivity and animation.

SVG Highlights

- DOM API
- Defines vector-based graphics for the Web
- Supports CSS styling
- Element grouping
- Hyperlinks
- Accessibility support (ARIA, etc)
- Path elements for arbitrary drawing



SVG (basic support) - REC

Global 89.87% + 2.74% = 92.61%

U.S.A. 86.54% + 1.74% = 88.28%

Method of displaying basic Vector Graphics features using the embed or object elements. Refers to the SVG 1.1 spec.

Current aligned | Usage relative | Show all

IE	Firefox	Chrome	Safari	Opera	iOS Safari *	Opera Mini *	Android Browser *	Chrome for Android
		31						
		35					[1] 4.1	
8		36					[1] 4.3	
[2] 9	33	37			7.1		4.4	
[2] 10	34	38	7.1		8		4.4.4	
[2] 11	35	39	8	26	8.1	8	37	39
[2] TP	36	40		27				
	37	41		28				
	38	42						

Notes | Known issues (2) | Resources (7) | Feedback

[1] Partial support in Android 3 & 4 refers to not supporting masking.

[2] IE9-11 desktop & mobile don't properly scale SVG files. Adding height, width, viewport, and CSS rules seem to be the best workaround.

SVG support by browser

Source: Can I use...

Common SVG Elements

svg
- Container element

circle, rect, line, ...
- Various shape elements

path
- Arbitrary drawing paths
- ‘d’ attribute for path data

g
- Used for grouping

a
- Links of course

text
- Textual content

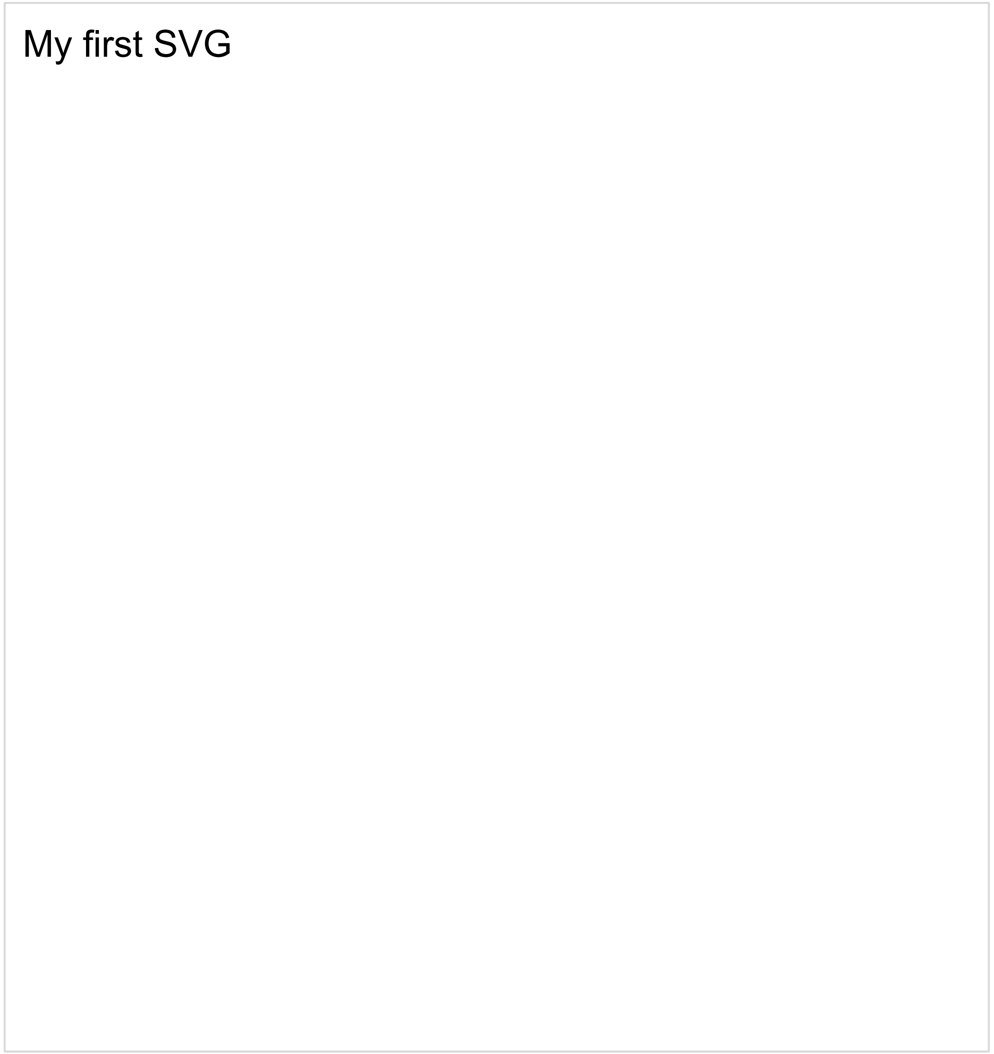
SVG Attributes

`fill`
- color of the inside of an element

`stroke`
- color of the border of an element

`stroke-width`
- width of the border

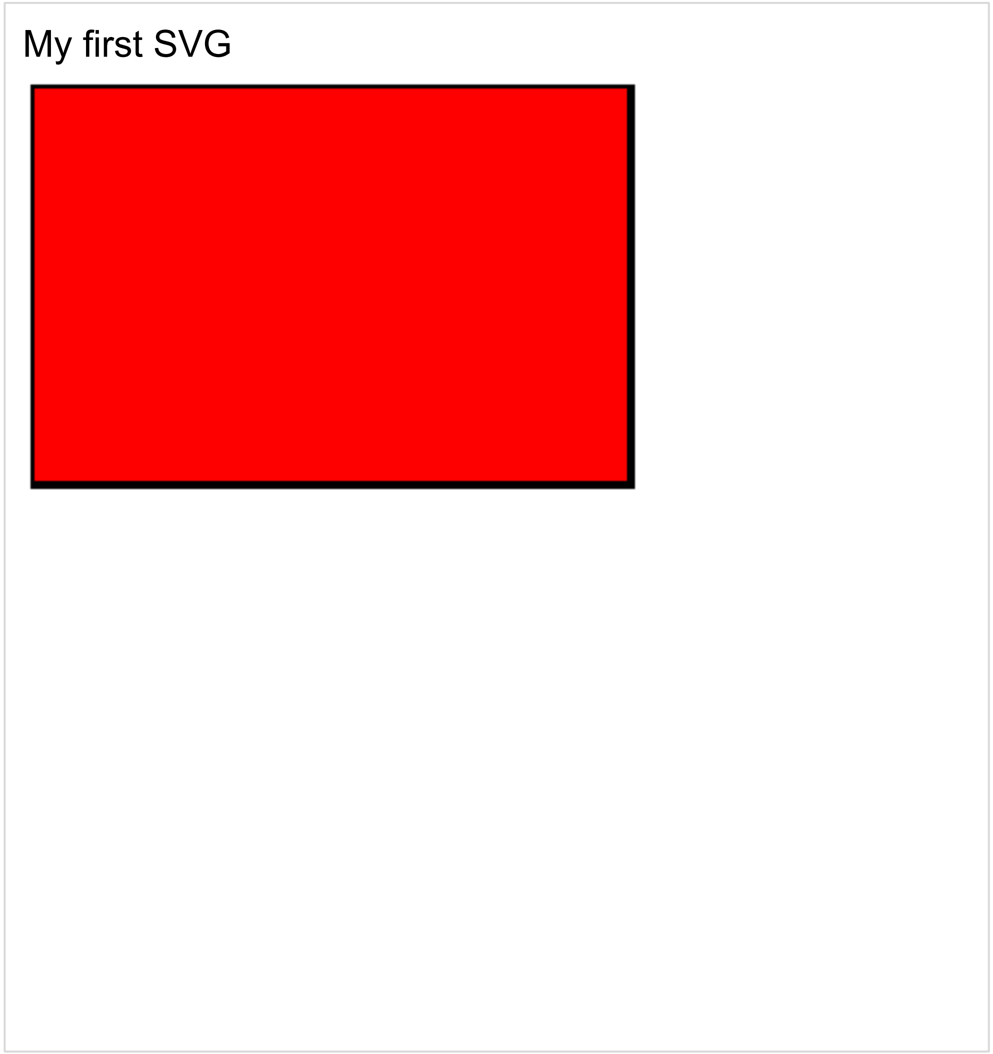
`stroke-dasharray`
- customizable dashes for lines


My first SVG

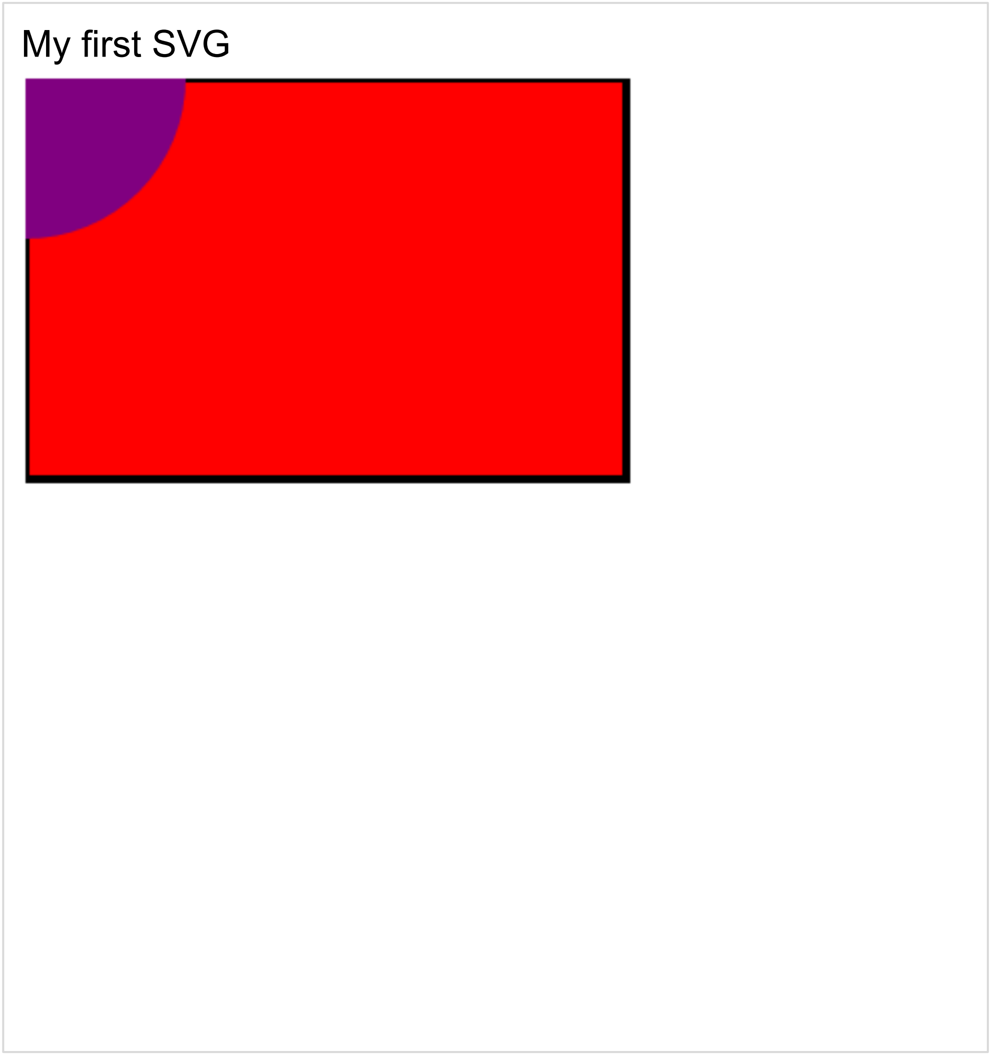
```
<html>
<body>

<h1>My first SVG</h1>

<svg width="600" height="600">
</svg>

</body>
</html>
```

My first SVG

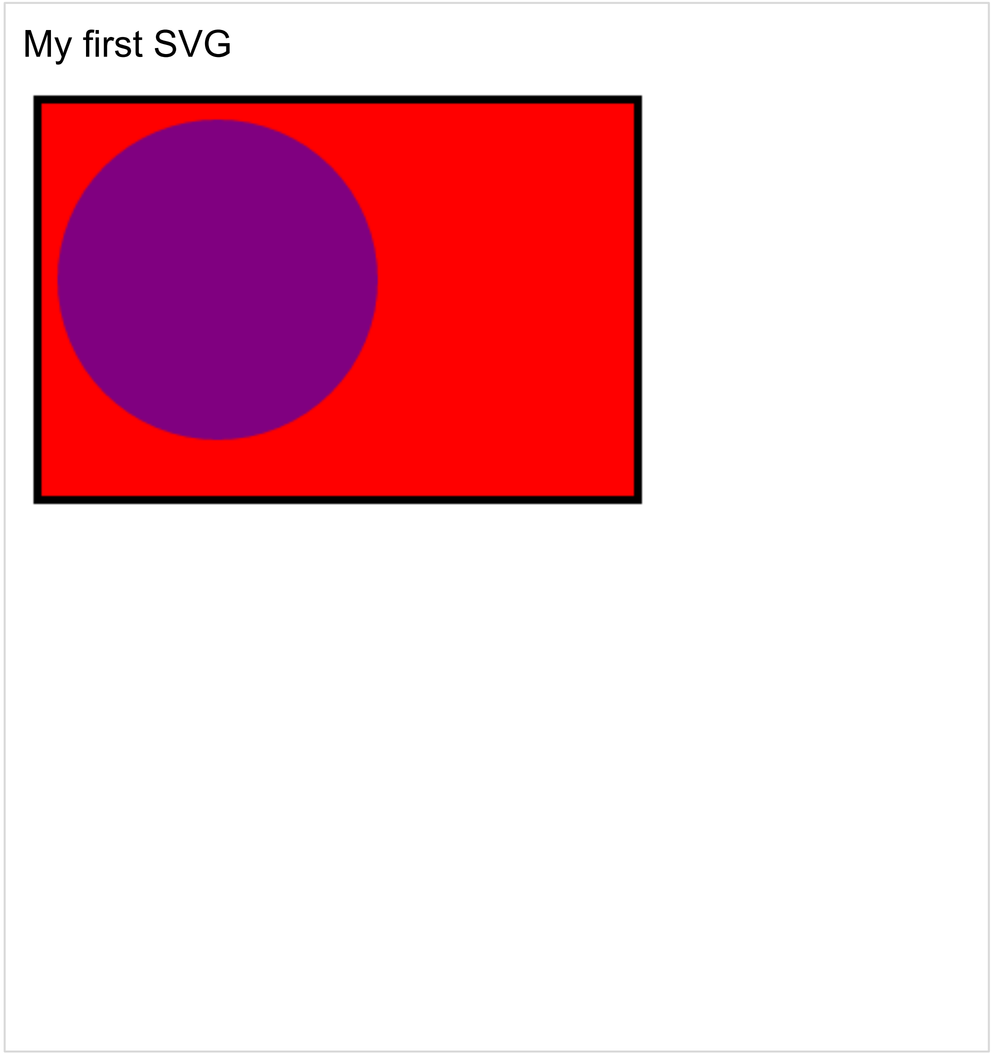
```
<html>
<body>

<h1>My first SVG</h1>

<svg width="600" height="600">
  <rect width="300" height="200">
</svg>

</body>
</html>
```



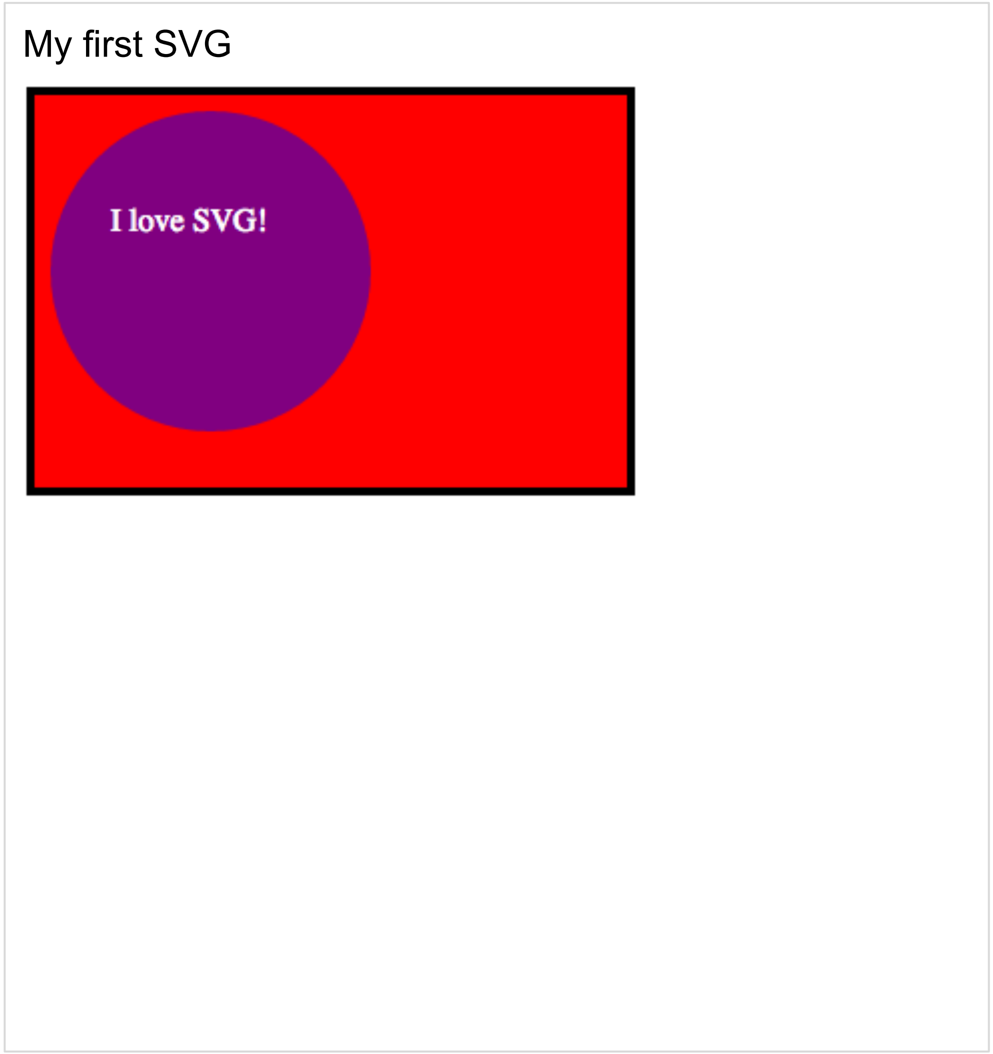
```
<html>
<body>

<h1>My first SVG</h1>

<svg width="600" height="600">
  <rect width="300" height="200" fill="red"
stroke="black" stroke-width="4">
</svg>

</body>
</html>
```


My first SVG

```
<html>
<body>

<h1>My first SVG</h1>

<svg width="600" height="600">
  <rect width="300" height="200" fill="red"
stroke="black" stroke-width="4">
  <circle r="80" fill="purple" />
</svg>

</body>
</html>
```


My first SVG

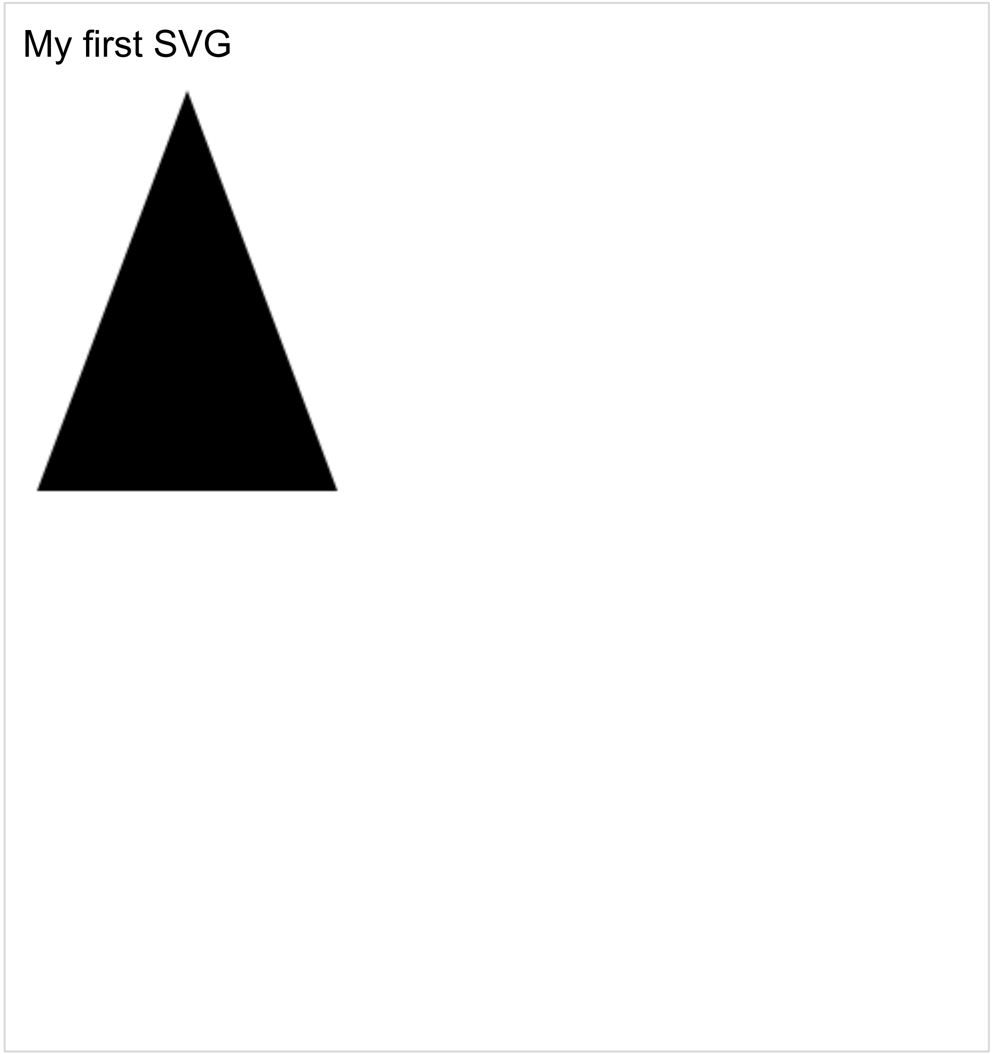
```
<html>
<body>

<h1>My first SVG</h1>

<svg width="600" height="600">
   <rect width="300" height="200" fill="red"
stroke="black" stroke-width="4" x="10" y="10" />
   <circle r="80" fill="purple" cx="100" cy="100" />
</svg>

</body>
</html>
```


My first SVG

```
<html>
<body>

<h1>My first SVG</h1>

<svg width="600" height="600">
   <rect width="300" height="200" fill="red"
stroke="black" stroke-width="4" x="10" y="10" />
   <circle r="80" fill="purple" cx="100" cy="100" />
   <text x="70" y="80" fill="white">I love SVG!
</text>
</svg>

</body>
</html>
```

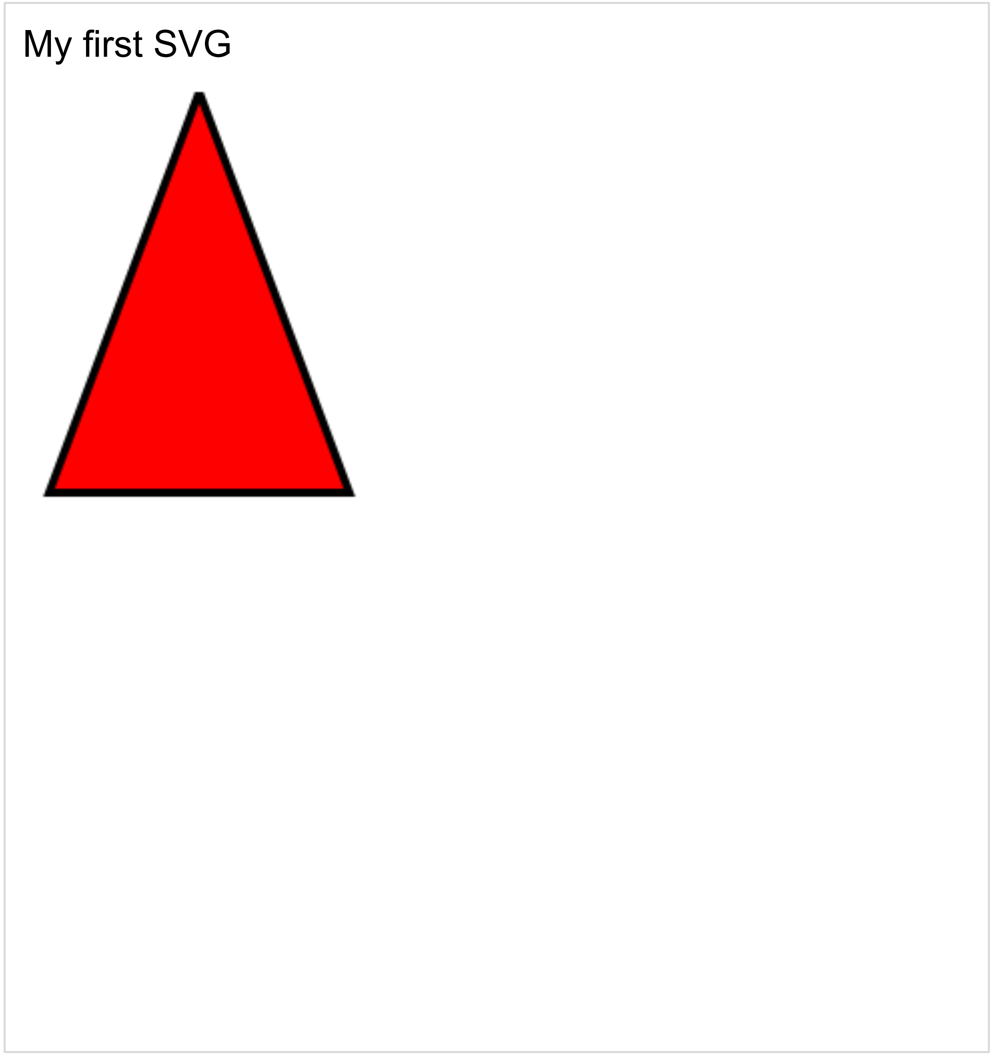
SVG Path

- M = moveto
- L = lineto
- H = horizontal lineto
- V = vertical lineto
- C = curveto
- S = smooth curveto
- Q = quadratic Bézier curve
- T = smooth quadratic Bézier curveto
- A = elliptical Arc
- Z = closepath


My first SVG

```
<html>
<body>

<h1>My first SVG</h1>

<svg width="600" height="600">
   <path d="M150 0 L75 200 L225 200 Z" />
</svg>

</body>
</html>
```

SVG + CSS



```
<html>
<body>

<style>
svg path {
   fill:red;
   stroke-width:4;
   stroke:black;
}
</style>

<h1>My first SVG</h1>

<svg width="600" height="600">
   <path d="M150 0 L75 200 L225 200 Z" />
</svg>

</body>
</html>
```

Data Driven Documents (D3)

What is D3?

D3.js is a JavaScript library for manipulating documents based on data.

- D3js.org

D3js is the glue between data and SVG (or other DOM elements).

Why should I use it?

- Cross Browser Compatibility
- Easy to learn API
- Good documentation and examples
- Expansive library of data visualizations
- Out-of-the-box functions:
  - XHR data loading
  - Geo data conversion

Other Options Besides D3?

Highcharts

Pro: Easy to use and customizable visualizations.
Con: Limited to available visualizations.

CanvasJS

Pro: Best contender to D3js flexibility. Faster than SVG.
Con: HTML5 based. Pixel based visualizations.

Google Charts

Pro: Easy to use predefined visualizations & Customized.
Con: Limited and requires live server linking.

D3 Code Foundation

D3 Selections

d3.select(*selector*: string)

- query one element

d3.selectAll(*selector*: string)

- query multiple elements

D3 Selection Actions

selection.append(name)

- Appends to the current selection

selection.attr(name[, value])

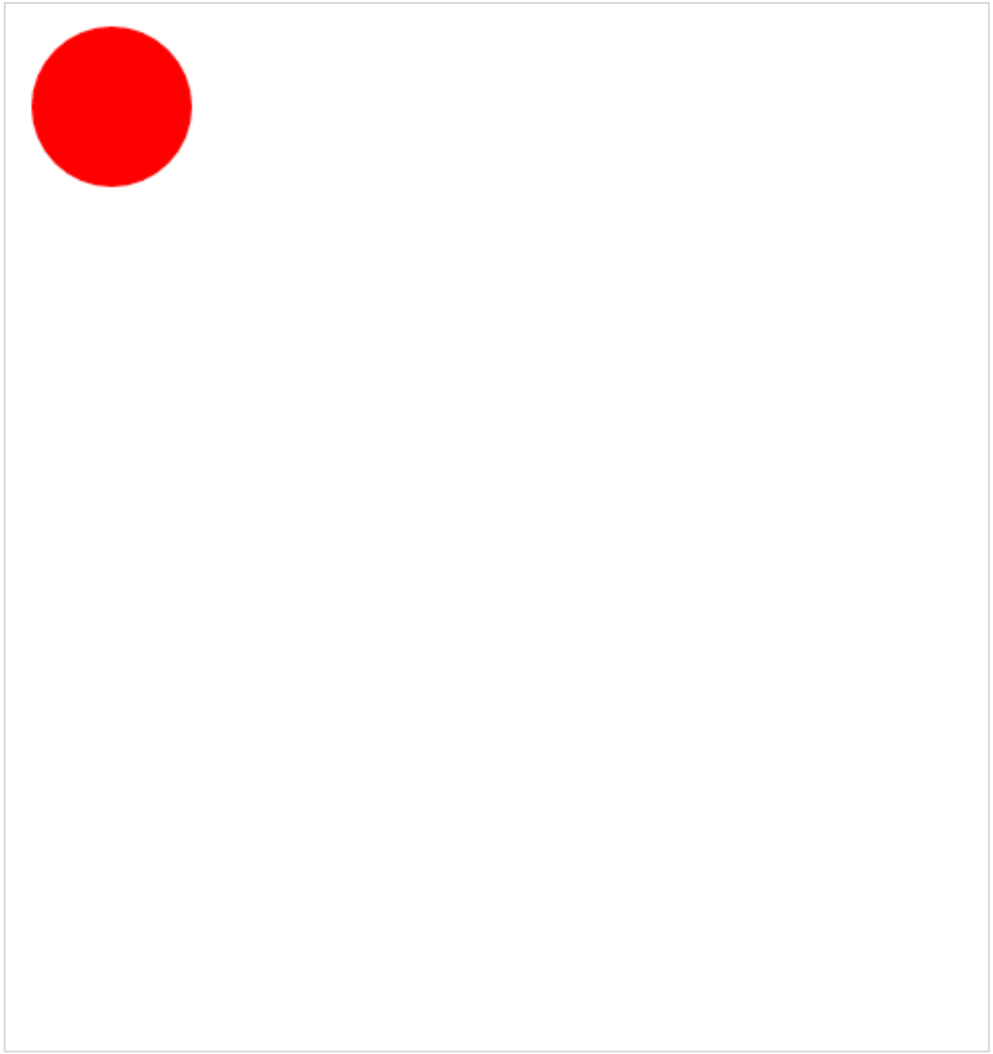
- Adds/sets attributes for the current selection

selection.on(type[, listener[, capture]])

- adds or removes event listeners

Example: SVG Element

```
var svg = d3.select(‘body’)
    .append(‘svg’)
    .attr(‘width’, 960)
    .attr(‘height’, 500)
    .on('mouseover', someFunction);
```



```
<div id="viz"/>

<script>
  //Create a sized SVG surface within viz:
  var vizsvg = d3.select("#viz")
    .append("svg")
    .attr("width", 600)
    .attr("height", 600);

  //Add to the svg surface a circle
  var circle =
    vizsvg.append("circle")
    .attr("fill", "red")
    .attr("r", 40)
    .attr("cx", 50)
    .attr("cy", 50);
</script>
```

SVG Transform

Applies transformations to an element and it’s children

- matrix(...)
- translate(...)
- scale(...)
- skewX(...)
- skewY(...)
- rotate(...)

Example: SVG Transform

```
// rotate(degrees, x, y)
var svg = d3.select(‘body’)
    .append(‘svg’)
    .attr(‘width’, 960)
    .attr(‘height’, 500)
    .append(‘g’)
    .attr(‘transform’, ‘rotate(-45 100
100)’);
```

Bring in the Data

Data Joins

update = selection.data(*data*)

Bind array of data to selection.

update.enter()

Iterates over data points that don’t have associated nodes.

update.exit()

Data nodes without matching data trigger this.

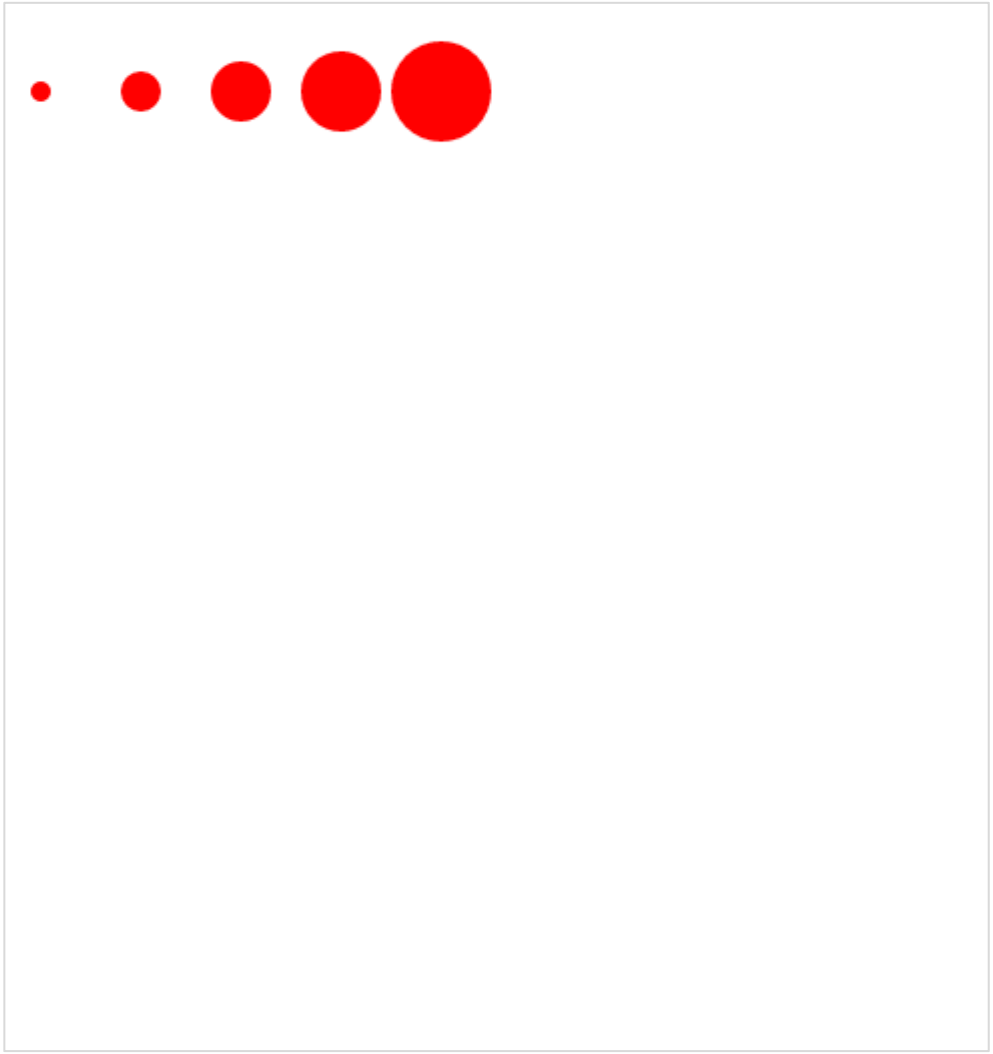
D3 Data Actions

d3.max(array[, accessor])

- Returns the maximum value of a given array

d3.map([object][, key])

- Create a new array with the result of calling a function on every element in the array.



```
...

var dataset = [ 5, 10, 15, 20, 25 ];

var circles = svg.selectAll("circle")
   .data(dataset)
   .enter()
   .append("circle")
   .attr("r", function(d) {
         return d;
     })
   .attr("cx", function(d, i) {
         // i is the current data node index
         return (i * 50) + 25;
     })
   .attr("cy", h/2)
   .attr("fill","red");
```

D3 Scales

D3.js provides functions to perform data transformations.

These functions map an input domain to an output range.

Said another way, these functions take an interval and transform it into a new interval.

Example: Linear Scale

```
//Initial Data
var dataset = [ 100, 200, 300, 400, 500 ];

// Domain 0 to 500, Range 0 to canvas width
var xScale = d3.scale.linear()
   .domain([0, d3.max(dataset, function(d) {
return d; })])
   .range([0, output_width]);
```

DRUPAL & D3 MODULE

Common Integration Methods

Views with Basic D3 Module Libraries

Views with Custom D3 Module Libraries

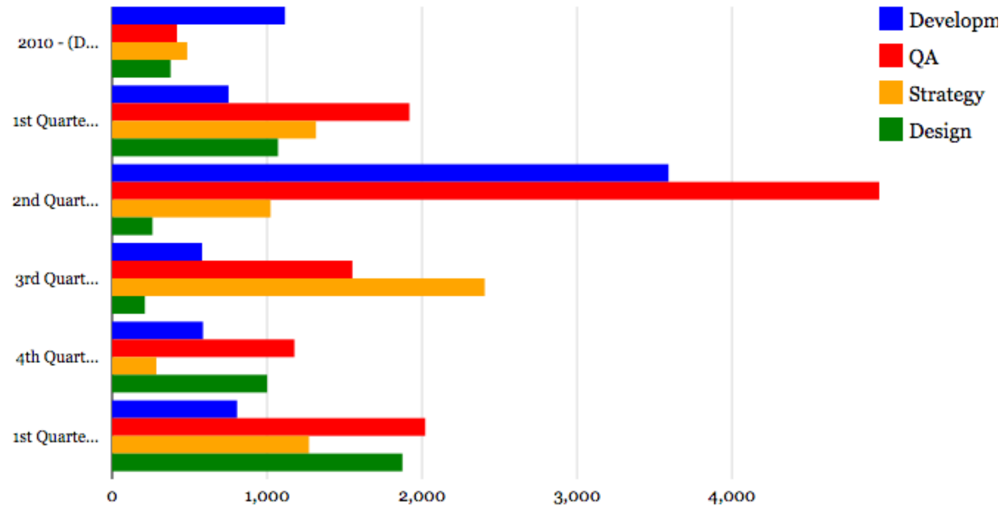
Custom D3 WITHOUT D3 Module

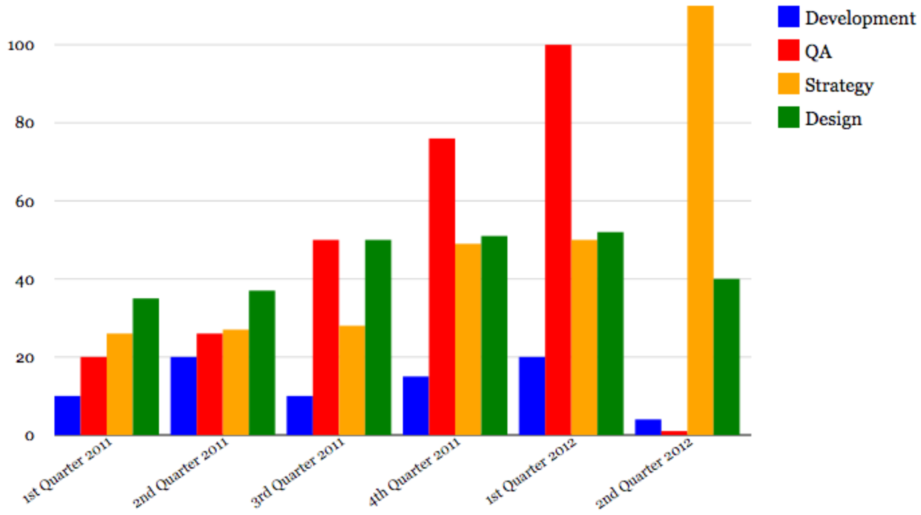
Why use the D3 module?

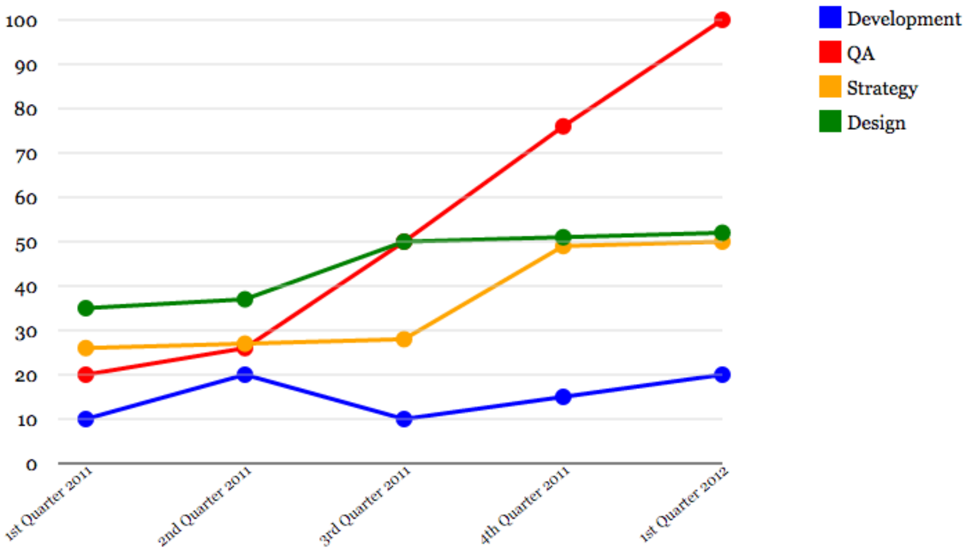
D3 Module Features

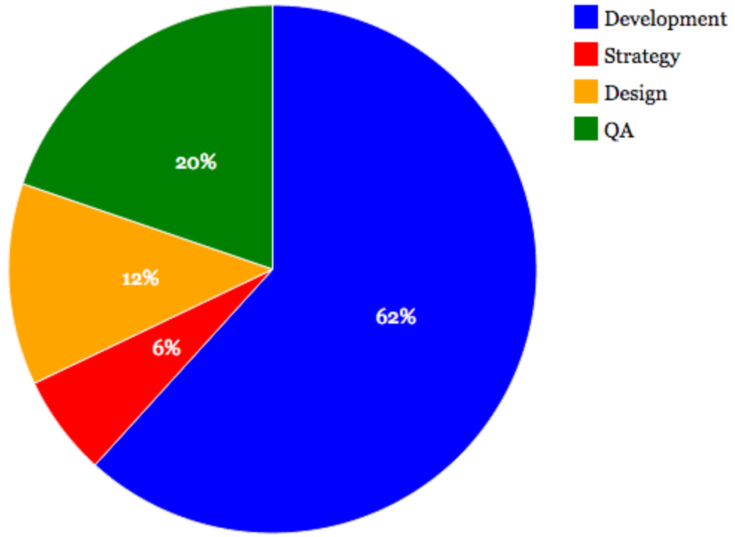
- Simple API
- Out-of-the-box basic visualizations
- Custom library support
- Simplified data binding through Views integration and custom library

Views with Basic D3 Libraries

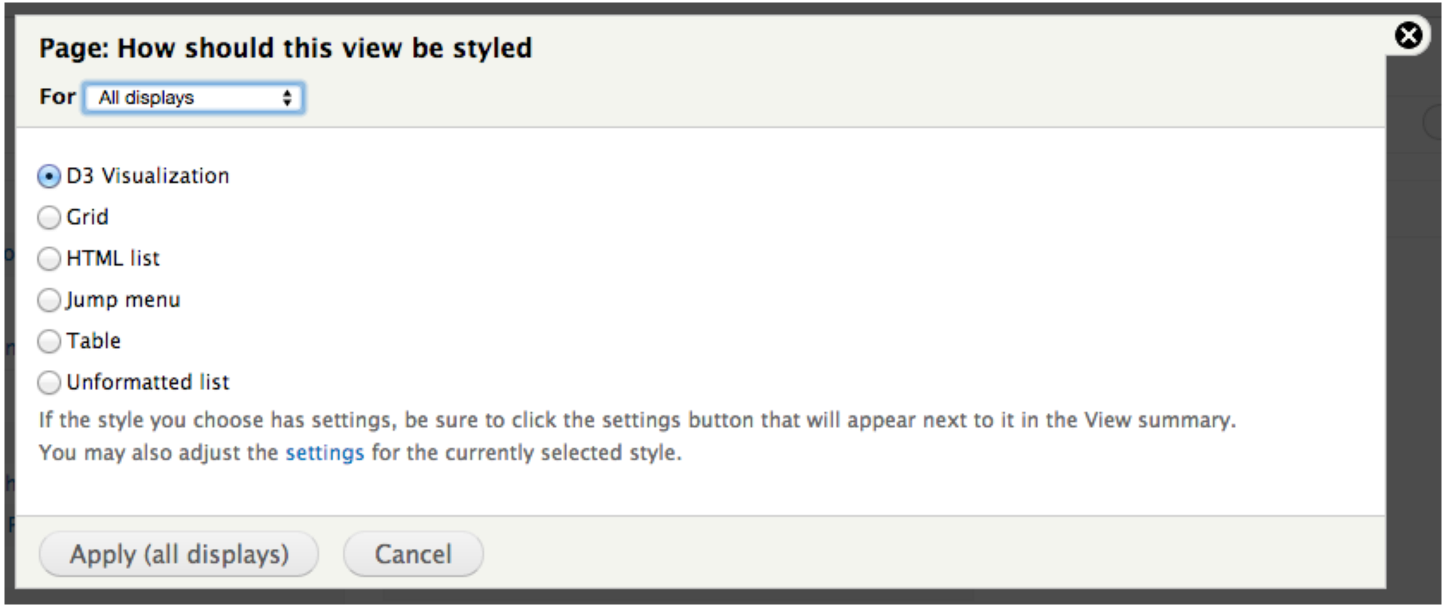

2010 - (D...
1st Quarte...
2nd Quart...
3rd Quart...
4th Quart...
1st Quarte...
0
1,000
2,000
3,000
4,000
Developm
QA
Strategy
Design


0
20
40
60
80
100
1st Quarter 2011
2nd Quarter 2011
3rd Quarter 2011
4th Quarter 2011
1st Quarter 2012
2nd Quarter 2012
Development
QA
Strategy
Design


0
10
20
30
40
50
60
70
80
90
100
1st Quarter 2011
2nd Quarter 2011
3rd Quarter 2011
4th Quarter 2011
1st Quarter 2012
Development
QA
Strategy
Design


20%
12%
6%
62%
Development
Strategy
Design
QA

Select the Views Format



Page: How should this view be styled

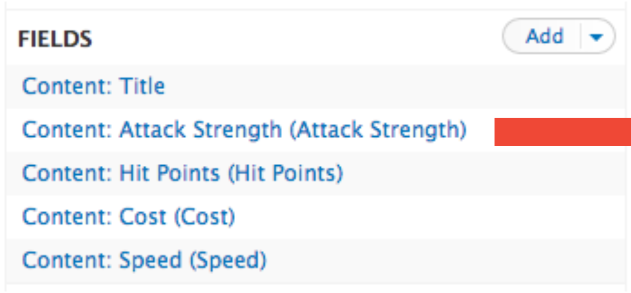
For All displays

[✓ D3 Visualization]
Grid
HTML list
Jump menu
Table
Unformatted list

If the style you choose has settings, be sure to click the settings button that will appear next to it in the View summary.

You may also adjust the settings for the currently selected style.

Add Your Fields



FIELDS

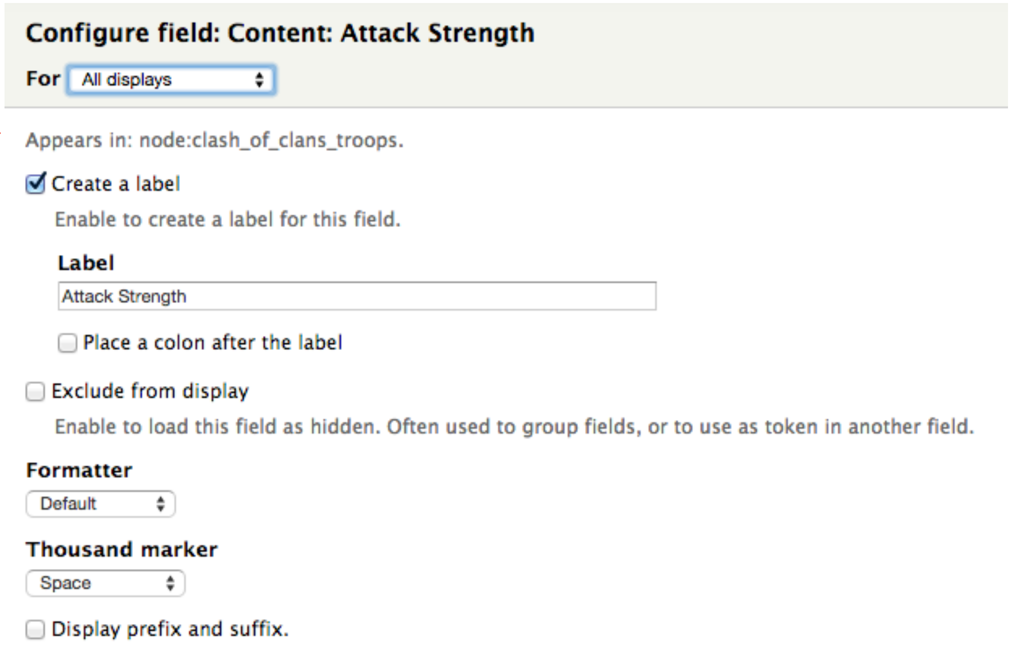
Content: Title

Content: Attack Strength (Attack Strength)

Content: Hit Points (Hit Points)

Content: Cost (Cost)

Content: Speed (Speed)



Configure field: Content: Attack Strength

For: All displays

Appears in: node:clash_of_clans_troops.

[✓] Create a label

Enable to create a label for this field.

Label: Attack Strength

[ ] Place a colon after the label

[ ] Exclude from display

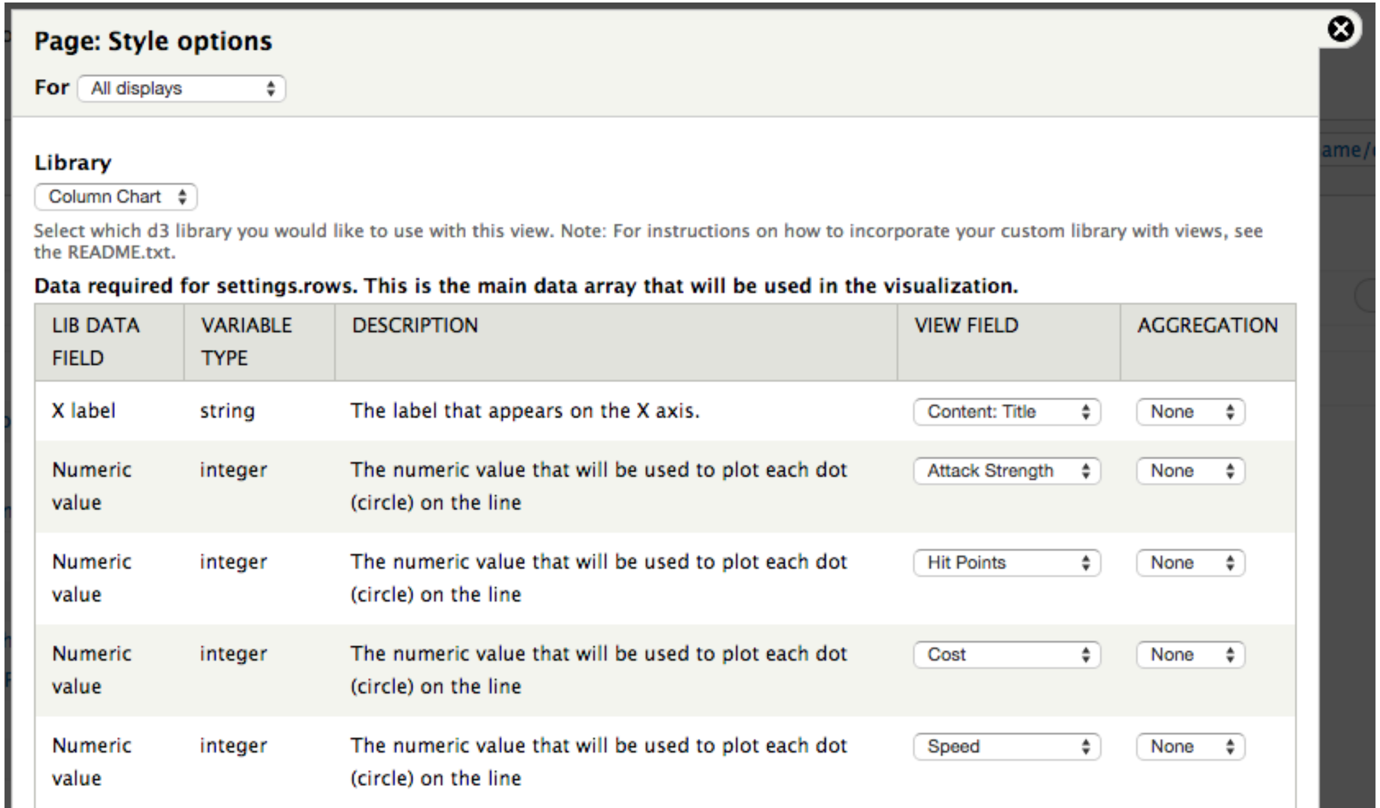
Enable to load this field as hidden. Often used to group fields, or to use as token in another field.

Formatter: Default

Thousand marker: Space

[ ] Display prefix and suffix.

Select a D3 Library and Configure Data



Page: Style options

For All displays

Library

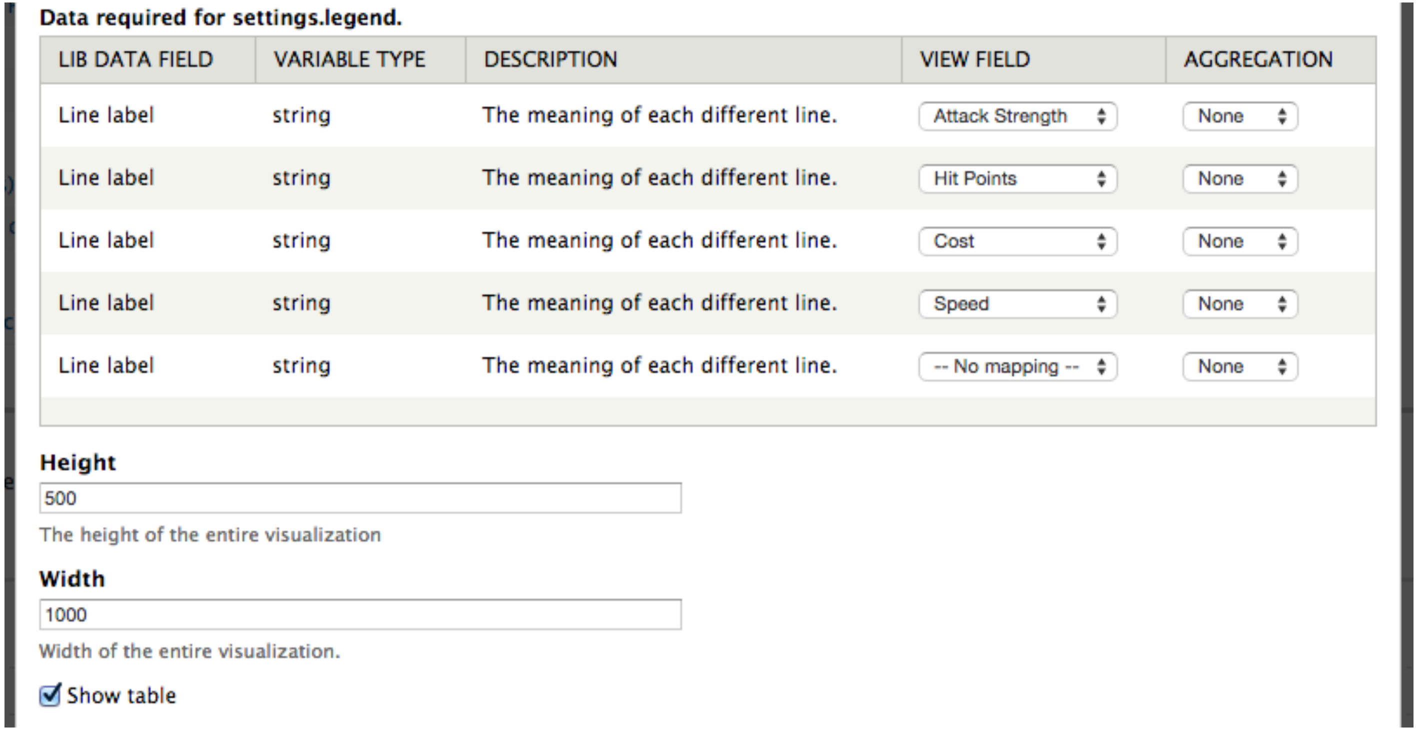
Column Chart

Select which d3 library you would like to use with this view. Note: For instructions on how to incorporate your custom library with views, see the README.txt.

Data required for settings.rows. This is the main data array that will be used in the visualization.

LIB DATA FIELD	VARIABLE TYPE	DESCRIPTION	VIEW FIELD	AGGREGATION
X label	string	The label that appears on the X axis.	Content: Title	None
Numeric value	integer	The numeric value that will be used to plot each dot (circle) on the line	Attack Strength	None
Numeric value	integer	The numeric value that will be used to plot each dot (circle) on the line	Hit Points	None
Numeric value	integer	The numeric value that will be used to plot each dot (circle) on the line	Cost	None
Numeric value	integer	The numeric value that will be used to plot each dot (circle) on the line	Speed	None

Configure D3 Library Display



Data required for settings.legend.

LIB DATA FIELD	VARIABLE TYPE	DESCRIPTION	VIEW FIELD	AGGREGATION
Line label	string	The meaning of each different line.	Attack Strength	None
Line label	string	The meaning of each different line.	Hit Points	None
Line label	string	The meaning of each different line.	Cost	None
Line label	string	The meaning of each different line.	Speed	None
Line label	string	The meaning of each different line.	-- No mapping --	None

Height

500

The height of the entire visualization

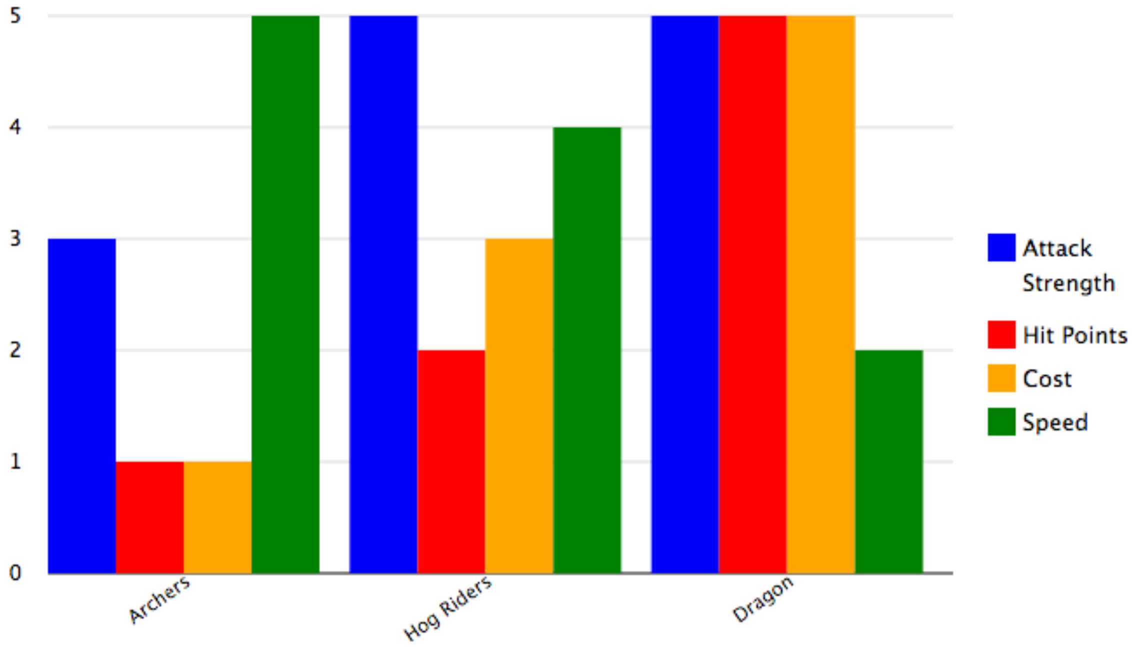
Width

1000

Width of the entire visualization.

☑ Show table

Content



TITLE	FIELD_CCT_ATTACK_STRENGTH	FIELD_CCT_HIT_POINTS	FIELD_CCT_COST	FIELD_CCT_SPEED
Archers	3	1	1	5
Hog Riders	5	2	3	4
Dragon	5	5	5	2

Views with Custom D3 Libraries

Custom D3 Library Files

- d3.myLibrary *(folder at sites/SITE/libraries)*
    - d3.myLibrary.libraries.info (contains info and dependencies)
    - myLibrary.css *(contains custom CSS)*
    - myLibrary.js *(contains custom D3 js)*
    - views-setting.php *(contains views info)*

d3.myViz.libraries.info

- Same as out the box D3 module custom library info file with some exceptions

```
name = My Vis
description = My Vis custom D3 display
files[js][] = myvis.js
files[js][] = nv.d3.min.js
files[css][] = myvis.css
version = 0.1
dependencies[] = d3.extend
dependencies[] = d3.tooltip
views[version] = 3.0
views[fields][rows][__data_type] = 2dnnv
views[fields][rows][x_label][label] = X label
views[fields][rows][x_label][type] = string
views[fields][rows][x_label][description] = The label that
appears on the X axis.
views[fields][rows][value] = { __repeated: TRUE, __cardinality: 0,
label: Numeric value, type: integer, description: The numeric
value that will be used to plot each dot (circle) on the line }
views[fields][legend][__cardinality] = 0
views[fields][legend][__data_type] = 1dn
views[fields][legend][label] = Line label
views[fields][legend][description] = The meaning of each
different line.
views[fields][legend][type] = string
views[settings] = views-settings.php
```

myViz.js

- Load data and configuration from views into JS variables for later use

```
(function($) {

 Drupal.d3.myvis = function (select, settings) {

   // Get the name of the DIV ID to place the visualization in
   var div = (settings.id) ? settings.id : 'visualization';
   // Get the height/width from the views settings
   var height = (settings.height) ? settings.height : '400';
   var width = (settings.width) ? settings.width : '600';
   // Get the rows of content
   var rows = settings.rows;
   // Shift the yLabel names from the rows array
   var yLabels = rows.map(function(d) { return d.shift(); });
   // Get key names for content grouping
   var key = settings.legend;

...
```

myViz.js (cont.)

- Rearrange data structure to match the following structure:

```
"key": "Dragon",
"color": "#E80CC8",
"values": [
   {
     "label" : "Cost" ,
     "value" : 5
   },
   {
     "label" : "Strength" ,
     "value" : 5
   }
]
```

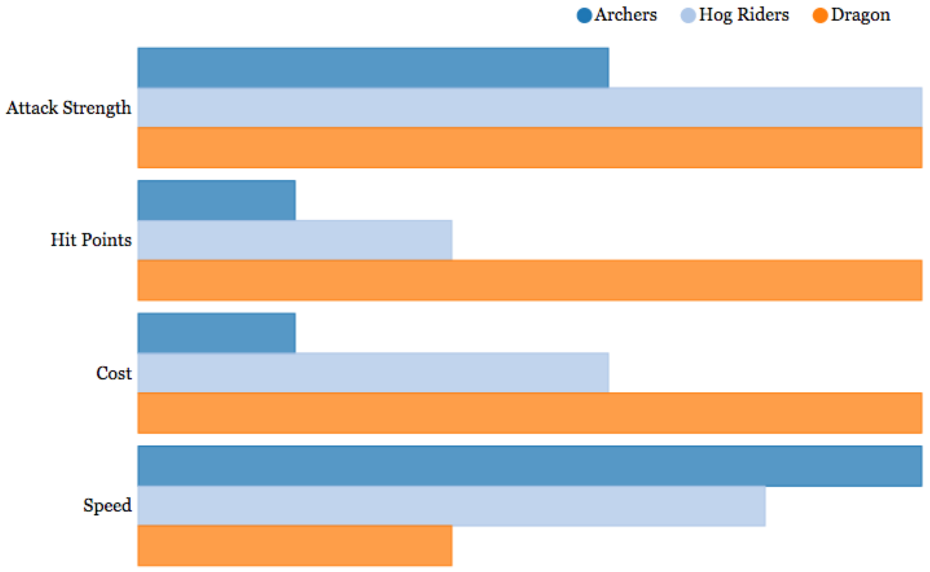
```
...

var troopData = [];
   // Force rows array into data structure noted above
   for (var i in rows) {
     var row = rows[i];
     var items = [];
     for (var j in row) {
       var item = row[j];
       items.push({
         "label" : key[j],
         "value" : item
       });
     }
     troopData.push({
       "key" : yLabels[i],
       "values" : items
     });
   }

...
```

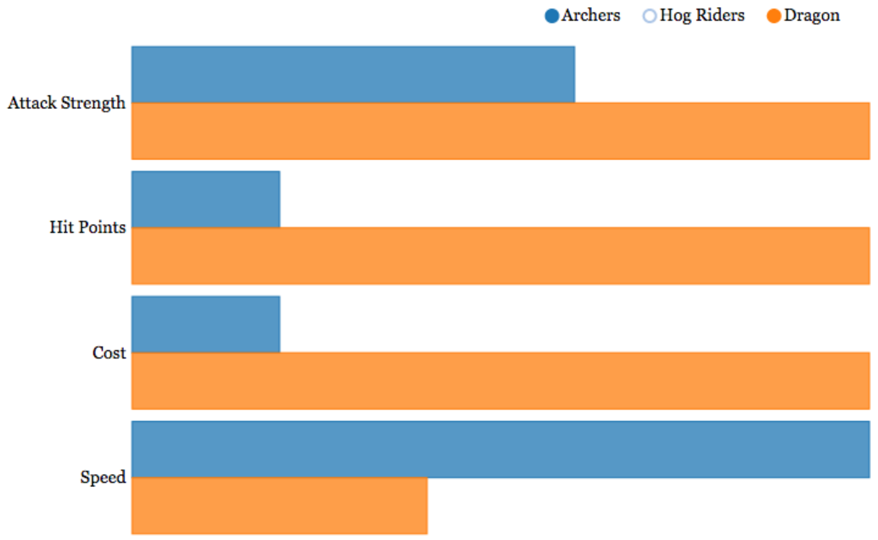
myViz.js (cont.)

- Create an NVD3 Multi-Bar-Horizontal-Chart data visualization with the data variables set previously
- Create an SVG element inside the Views content DIV
- NVD3 chart will be injected into the SVG element

```
...

// Render the NVD3 Multi-Bar Horizontal Chart in the proper DIV
element
   var chart;
   nv.addGraph(function() {
     chart = nv.models.multiBarHorizontalChart()
        .x(function(d) { return d.label })
        .y(function(d) { return d.value })
        .margin({top: 30, right: 20, bottom: 50, left: 175})
        .showValues(false) //Hide bar values
        .showControls(false) //Hide group toggle option
        .showYAxis(false); // Do not show yaxis values

     d3.select('#' + div).append("svg") // Append SVG to views div
        .attr("width",width) // Set SVG width and height
        .attr("height",height)
        .datum(troopData) // Inject Data
        .call(chart); // Inject the chart call

     nv.utils.windowResize(chart.update); // ensure proper sizing of
window
     return chart;
   });
  }
})(jQuery);
```

Resulting Custom Data Visualization





Custom D3 WITHOUT D3 Module

D3 Fully Custom

1. Get data in a structured format (JSON, CSV, Array, etc.)
   a. Usually Views
2. Create custom D3js Drupal library
3. Create custom Drupal module/hook to invoke library where needed
   a. Commonly injected into the content of blocks, panes and custom pages
4. Additional theming if needed

THANK YOU!

Keenan Holloway
Senior Developer
kholloway@forumone.com
www.forumone.com