

Why this is a PDF

This presentation was built with Reveal.js.
Unfortunately, reveal does not have an export to
PowerPoint option. I chose to use reveal because it
better displays the sections of my presentation that
display code



ACTIVATE.



LABORATE.



INNOVATE.



Unlocking the Potential of I with Qlik Sense APIs

Kevin McGovern

Slalom Consulting

Who am I? Kevin McGovern



Kevin McGovern

- Data Visualization Consultant at Slalom
- Qlik Branch Contributor
- Builds things with D3

Questions we will cover

- What is D3?
- How is D3 used?
- Where would you use D3 in Qlik Sense Mashups?
- What do you need to do differently for Extensions?

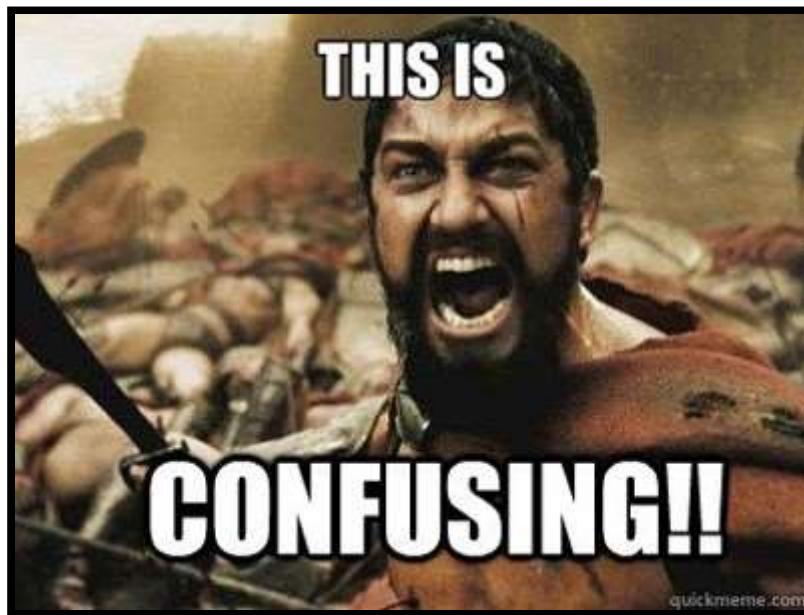
What is D3?

What D3 isn't

- Not a charting library
- Not a drawing library
- Not optimized for canvas and WebGL
- Not a framework

What D3 is

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.



quickmeme.com

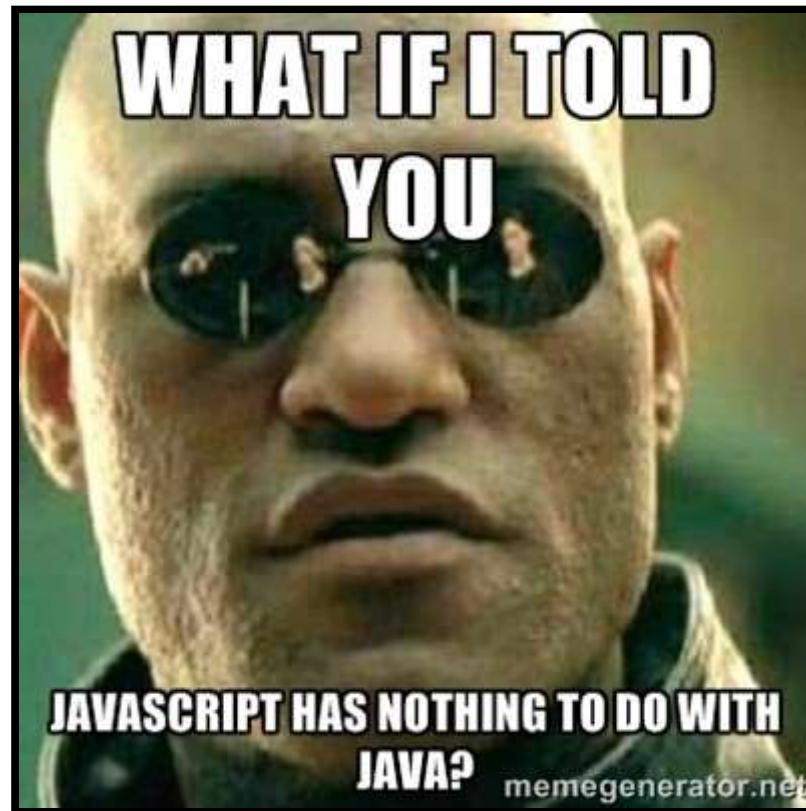
PATIENCE YOU MUST HAVE

WORTH IT WILL BE

meme crunch.com

Backing Up "A JavaScript Library"

What is JavaScript?



D3: "A JavaScript Library"

What is JavaScript?

- HTML sets document structure
- CSS sets styling
- JS informs your browser how to display content dynamically

D3: "A JavaScript Library"

What is a js library?

Extending what native js can do, makes developers lives easier

- jQuery
- Angular
- React
- Ember
- Three.js
- Node.js
- D3.js

D3: "manipulating documents based on data"

Browser renders HTML, CSS, JS to create Document Object Model

...or DOM



The DOM

More than just a fast and furious character
API that lets you interact with rendered web pages

Where does D3 come in?

- Bind data to elements in the DOM
- Change existing elements based on bound data
- Add new elements when more data added
- Remove elements when data changes
- Basically...

HTML Magic



D3 in Action

Finding elements, binding data, creating new elements

Using Scott Murray's Awesome Tutorial

An Example

```
var dataset = [ 5, 10, 15, 20, 25 ];  
  
d3.select("body")  
  .selectAll("p")  
  .data(dataset)  
  .enter()  
  .append("p")  
  .text("New paragraph!");
```

An Example

```
var dataset = [ 5, 10, 15, 20, 25 ];  
  
d3.select("body")  
    .selectAll("p")  
    .data(dataset)  
    .enter()  
    .append("p")  
    .text("New paragraph!");
```

An Example

```
var dataset = [ 5, 10, 15, 20, 25 ];  
  
d3.select("body")  
    .selectAll("p")  
    .data(dataset)  
    .enter()  
    .append("p")  
    .text("New paragraph!");
```

An Example

```
var dataset = [ 5, 10, 15, 20, 25 ];  
  
d3.select("body")  
    .selectAll("p")  
    .data(dataset)  
    .enter()  
    .append("p")  
    .text("New paragraph!");
```

An Example

```
var dataset = [ 5, 10, 15, 20, 25 ];  
  
d3.select("body")  
    .selectAll("p")  
    .data(dataset)  
    .enter()  
    .append("p")  
    .text("New paragraph!");
```

An Example

```
var dataset = [ 5, 10, 15, 20, 25 ];  
  
d3.select("body")  
    .selectAll("p")  
    .data(dataset)  
    .enter()  
        .append("p")  
        .text("New paragraph!");
```

An Example

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

d3.select("body")
    .selectAll("p")
    .data(dataset)
    .enter()
    .append("p")
    .text("New paragraph!");
```

An Example - Results

The screenshot shows a browser's developer tools with the "Elements" tab selected. On the left, there is a preview area containing six identical paragraphs: "New paragraph!". On the right, the DOM tree is displayed:

```
<!DOCTYPE html>
<html lang="en">
  <head>...</head>
  <body>
    <script type="text/javascript">...</script>
    ...<p>New paragraph!</p> == $0
    <p>New paragraph!</p>
    <p>New paragraph!</p>
    <p>New paragraph!</p>
    <p>New paragraph!</p>
  </body>
</html>
```

The sixth paragraph node under the body element is highlighted with a grey background, indicating it is the result of the JavaScript code execution.

```
console.log(d3.selectAll("p"))
```

An Example - Exploring the Elements

```
> console.log(d3.selectAll("p"))

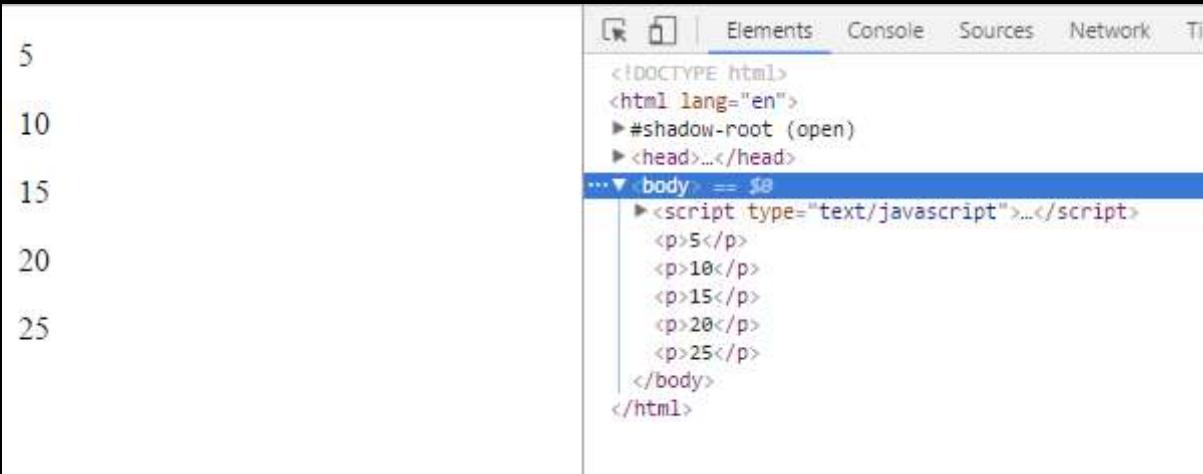
▼ [Array(5)] □
  ▼ 0: Array(5)
    ▼ 0: p
      accessKey: ""
      align: ""
      assignedSlot: null
      attributes: NamedNodeMap
```

```
translate: true
webkitdropzone: ""
__data__: 5
► __proto__: HTMLParagraphElement
▶ 1: p
▶ 2: p
▶ 3: p
▶ 4: p
▶ parentNode: document
length: 5
► __proto__: Array(0)
length: 1
► __proto__: Array(0)
```

Modified Example - Using Data to Add to Elements

```
var dataset = [ 5, 10, 15, 20, 25 ];  
  
d3.select("body")  
    .selectAll("p")  
    .data(dataset)  
    .enter()  
    .append("p")  
        .text(function(d) { return d; });
```

Modified Example - Results



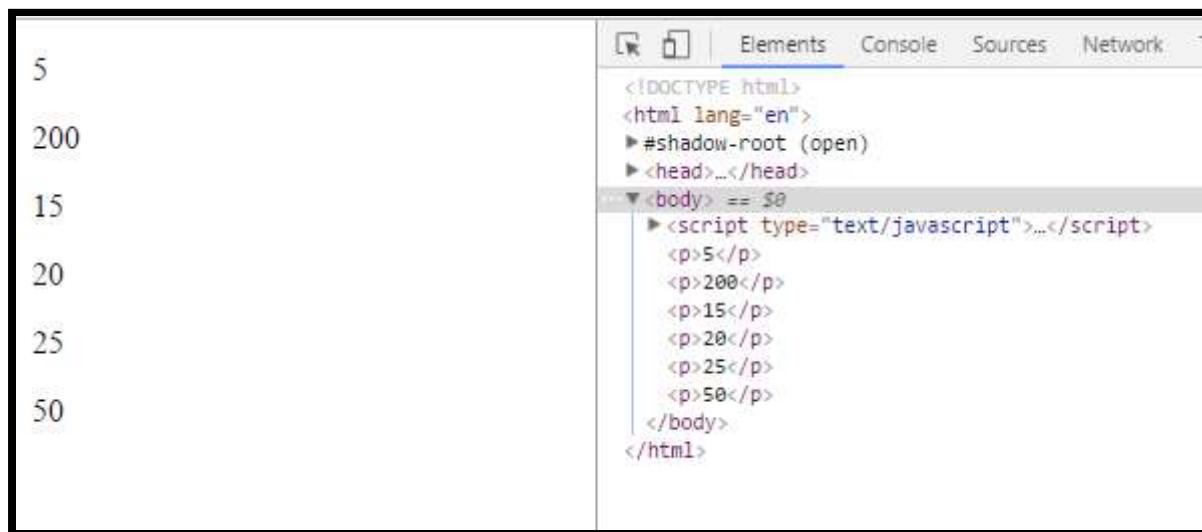
The screenshot shows the browser's developer tools with the "Elements" tab selected. On the left, there is a list of numbers: 5, 10, 15, 20, and 25. To the right, the DOM tree is displayed:

```
<!DOCTYPE html>
<html lang="en">
  >#shadow-root (open)
    ><head>...</head>
    ...<body> == $0
      ><script type="text/javascript">...</script>
      ><p>5</p>
      ><p>10</p>
      ><p>15</p>
      ><p>20</p>
      ><p>25</p>
    </body>
  </html>
```

Modified Example - Changing the Data

```
var dataset = [ 5, 200, 15, 20, 25, 50 ];  
  
d3.select("body")  
    .selectAll("p")  
    .data(dataset)  
    .enter()  
    .append("p")  
    .text(function(d) { return d; });
```

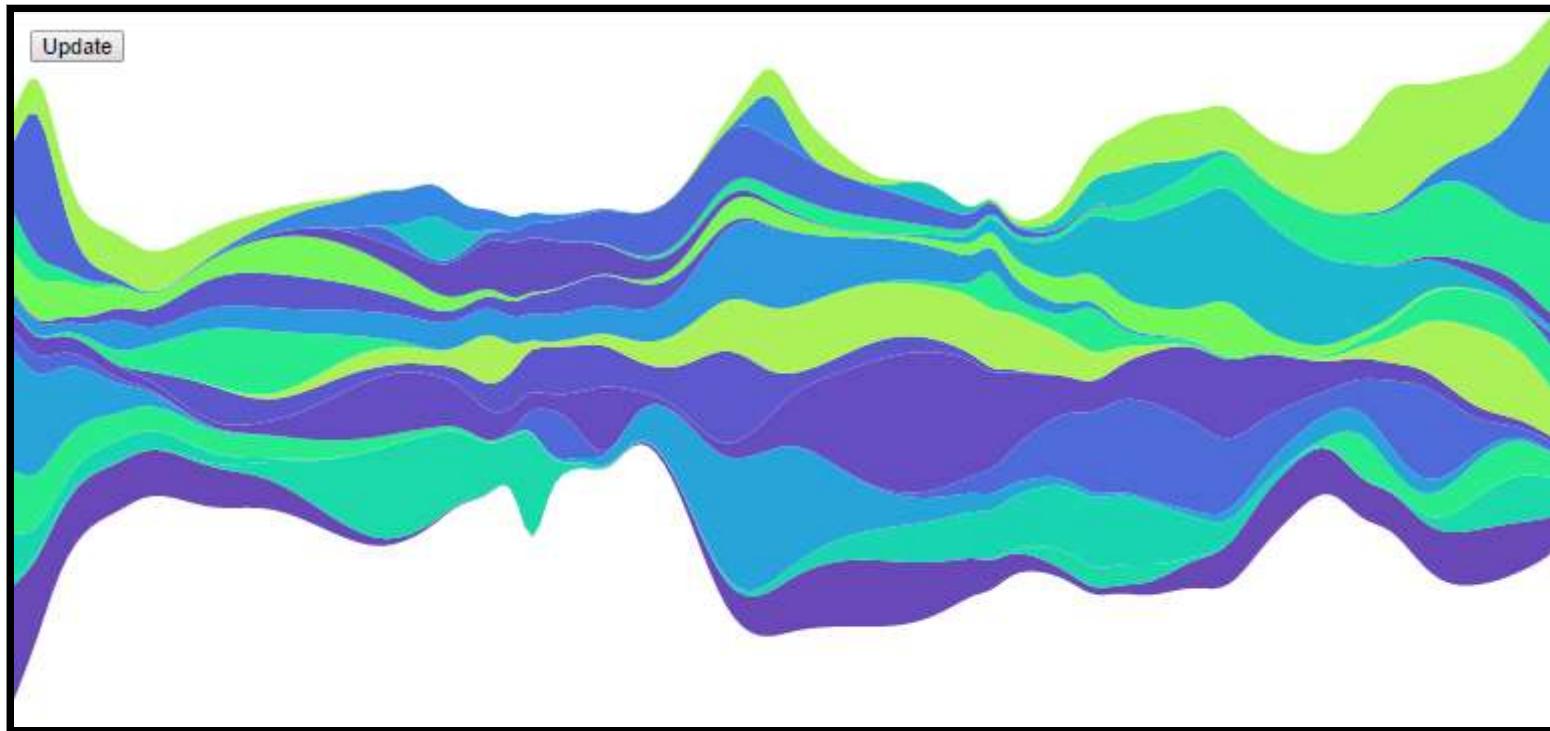
Modified Example - Results



The screenshot shows a browser's developer tools with the "Elements" tab selected. On the left, there is a list of numbers: 5, 200, 15, 20, 25, and 50. To the right, the browser's DOM tree is displayed, showing the following structure:

```
<!DOCTYPE html>
<html lang="en">
  >#shadow-root (open)
    ><head>...</head>
    ><body> == $0
      ><script type="text/javascript">...</script>
      ><p>5</p>
      ><p>200</p>
      ><p>15</p>
      ><p>20</p>
      ><p>25</p>
      ><p>50</p>
    </body>
</html>
```

Cool chart but why do I care?

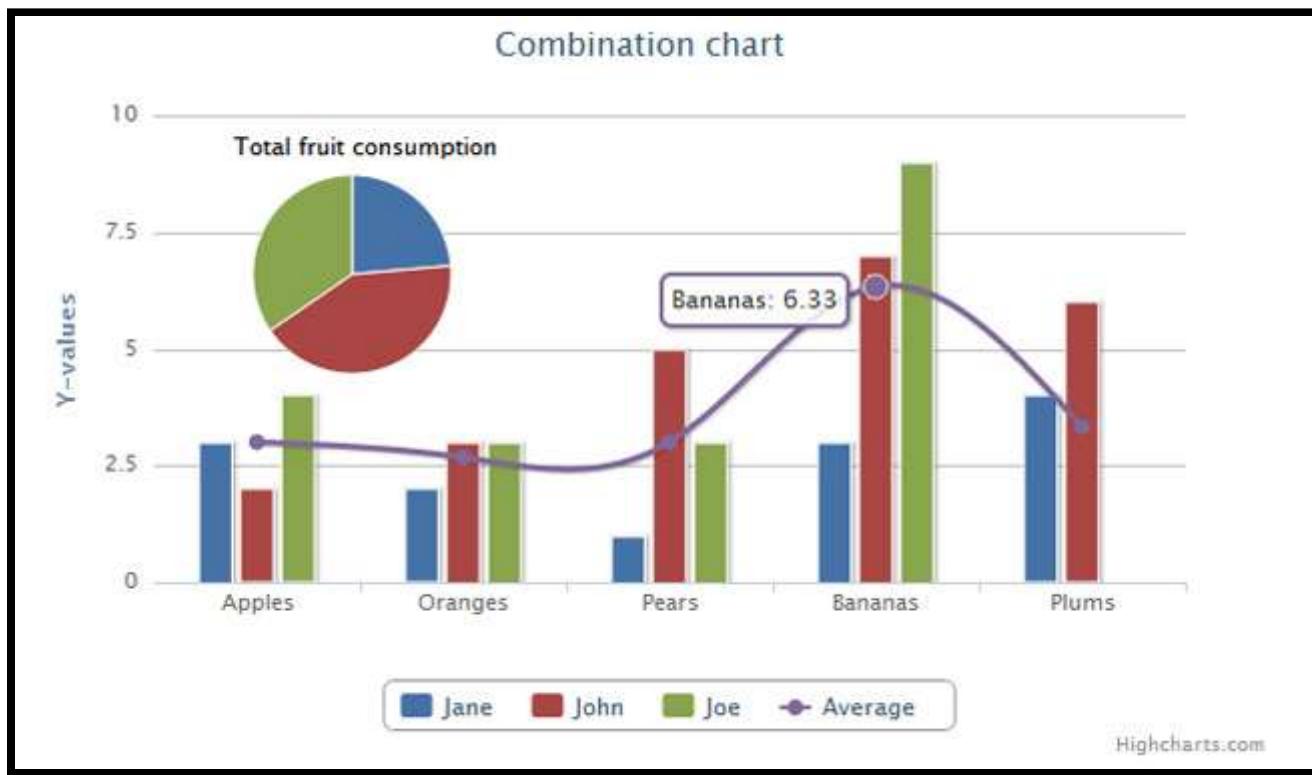


Dynamic data manipulation is really useful for data visualization

- Data changes over time
- User interactions
 - Clicks
 - Hover
 - Time-based delays

D3 looks hard, what about charting libraries?

- Few chart options
- Customization is difficult
- Limited page interaction



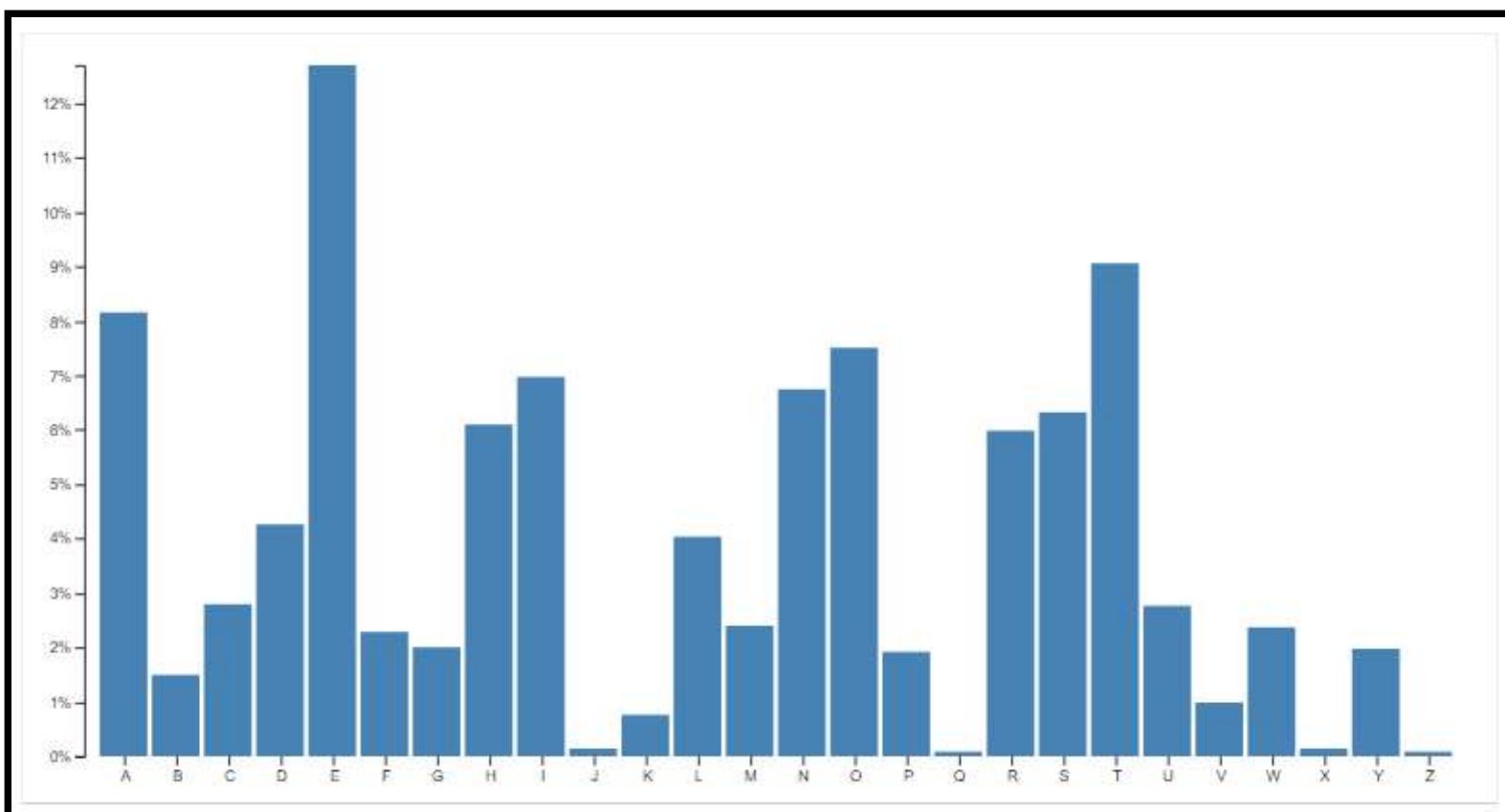
D3 is more than a way to just change
data on the page dynamically

- Create interactive charts
- Visualize hierarchical data
- Leverage open source geo libraries
- Make super slick transitions
- Visualize relationship-base data
- Integrate with the newest JS standards and libraries

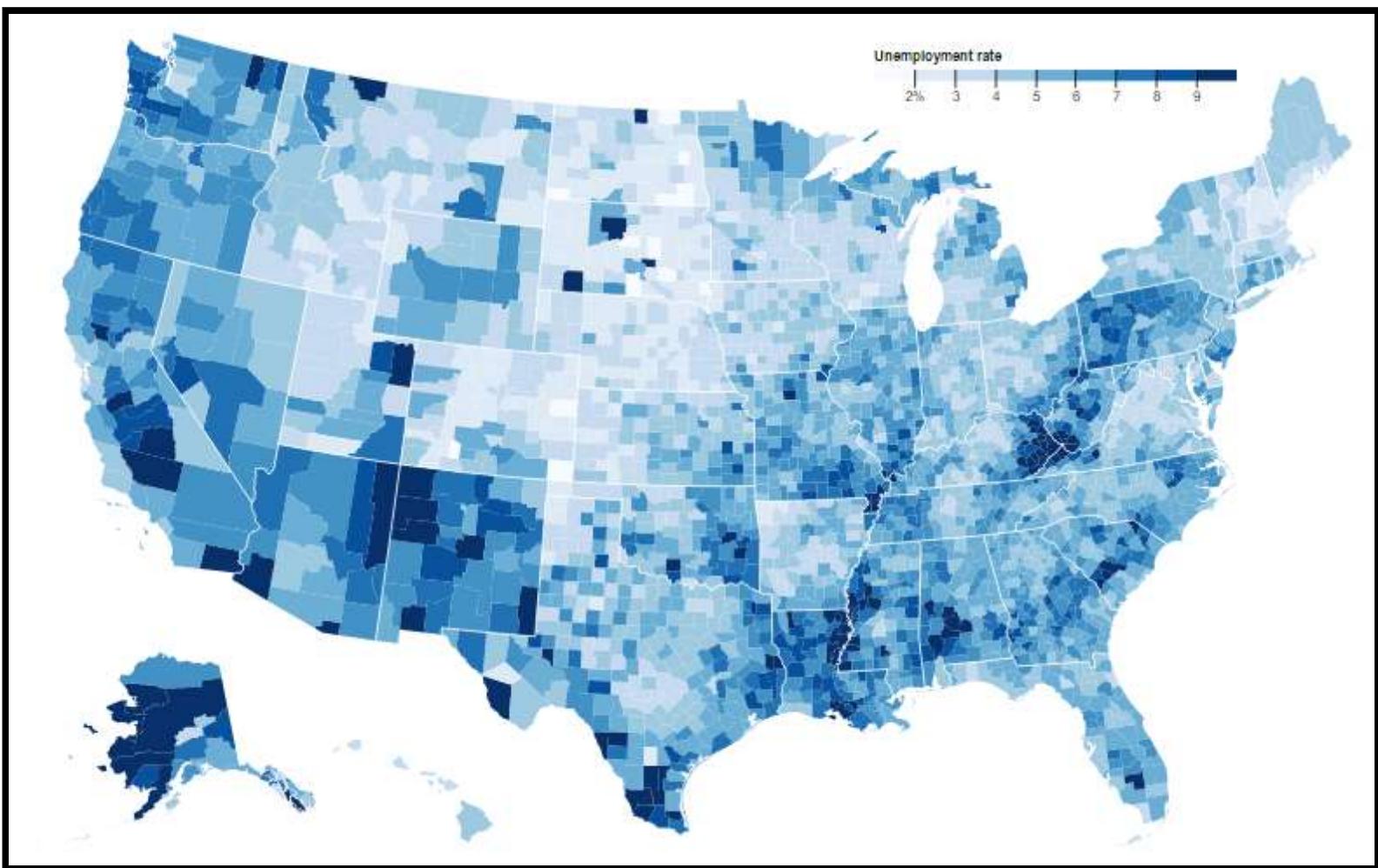
D3 in the wild



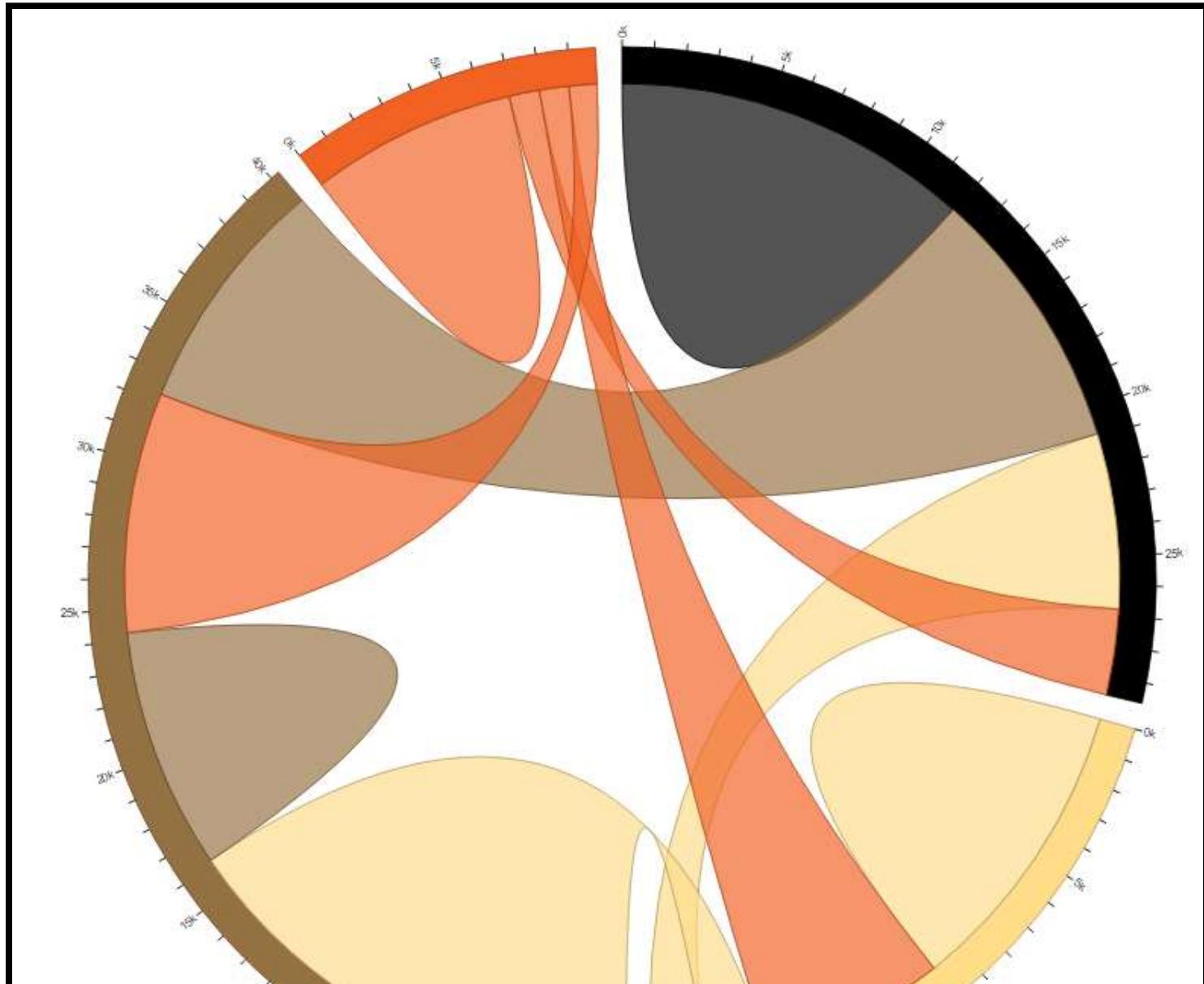
The most basic (source)

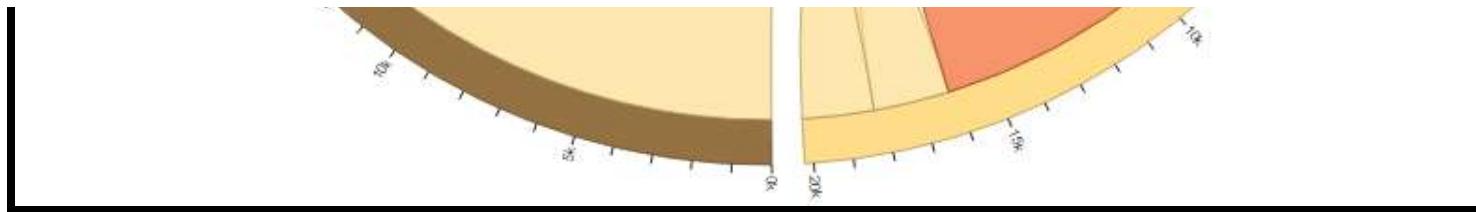


Maps (source)

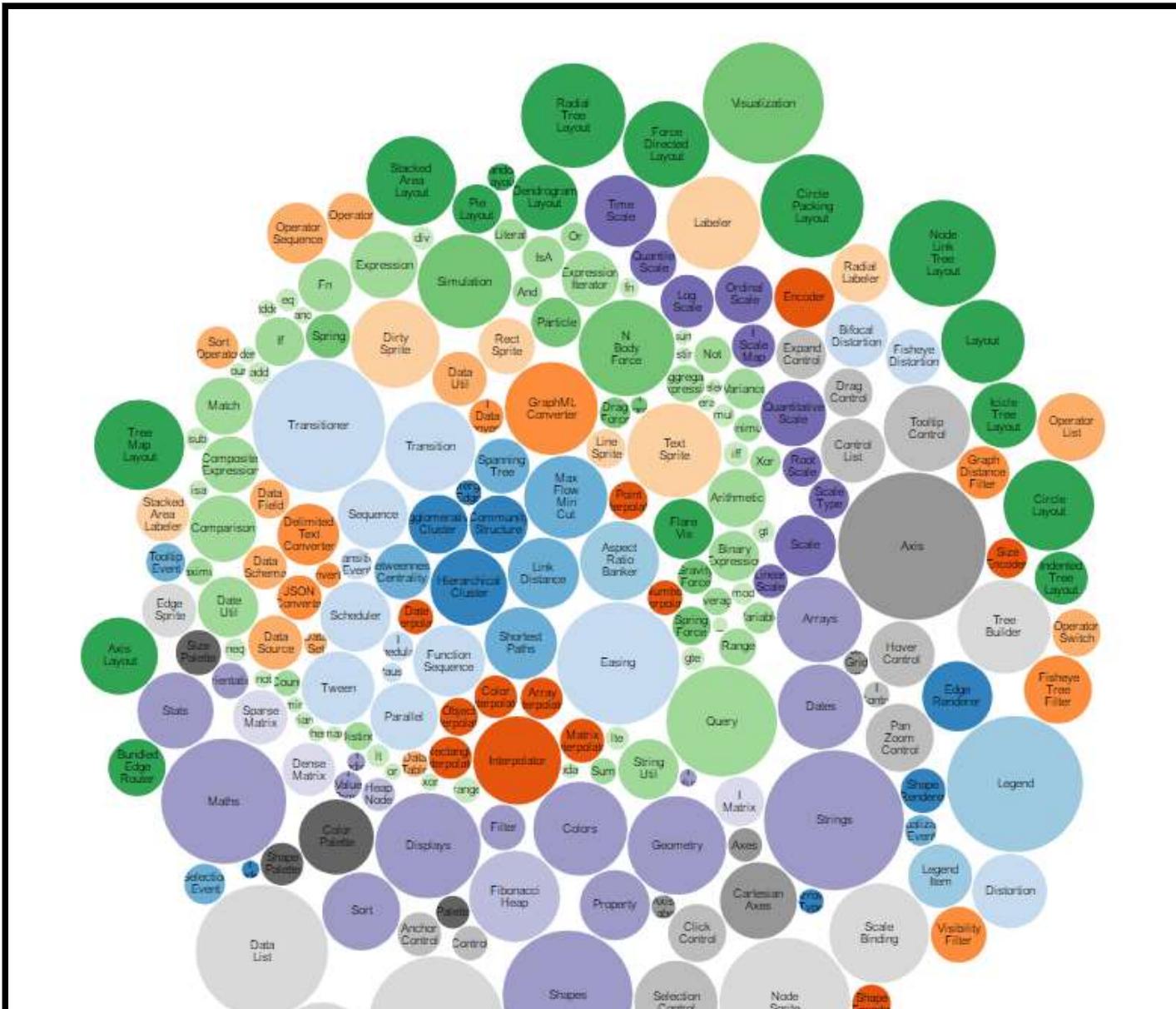


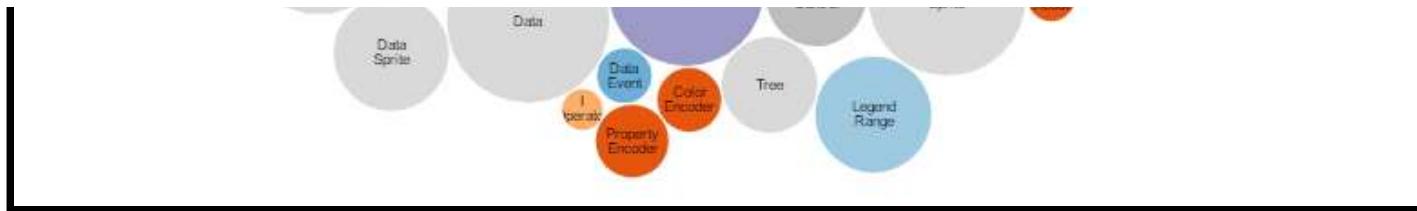
Chords (source)



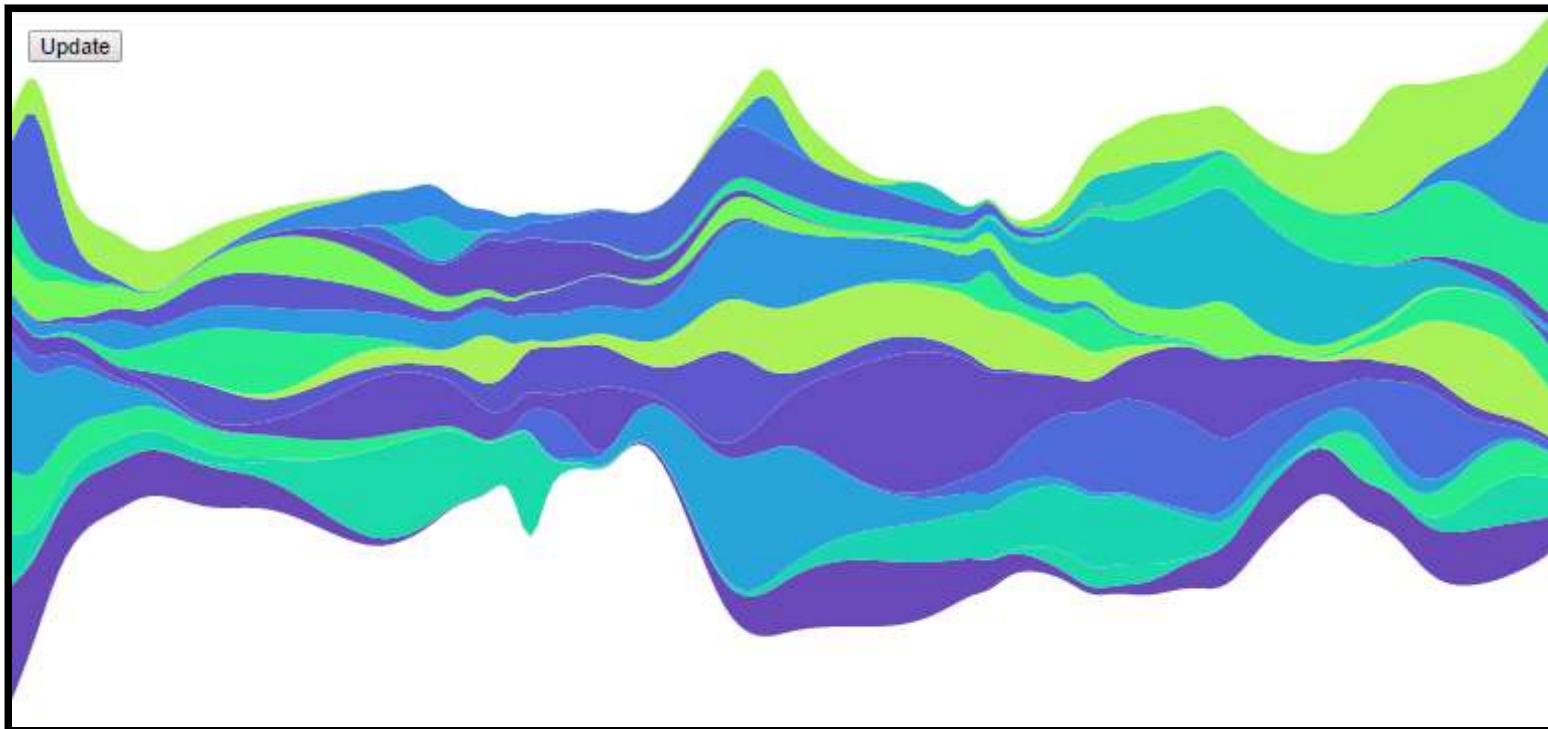


Bubble (source)

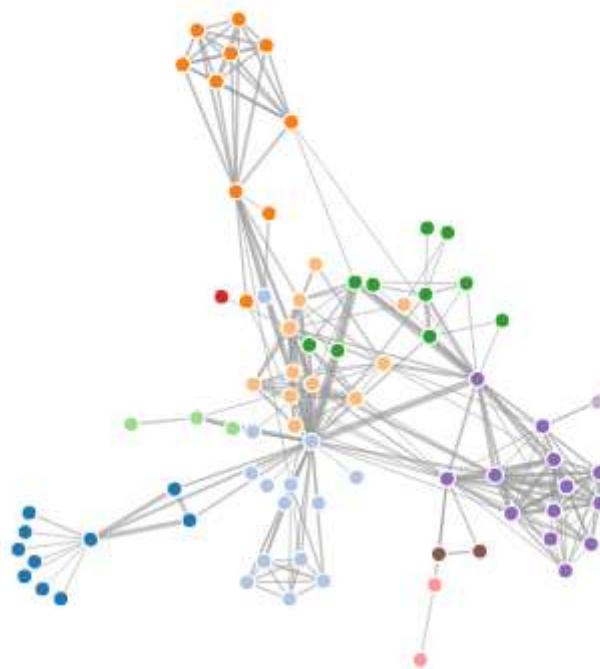




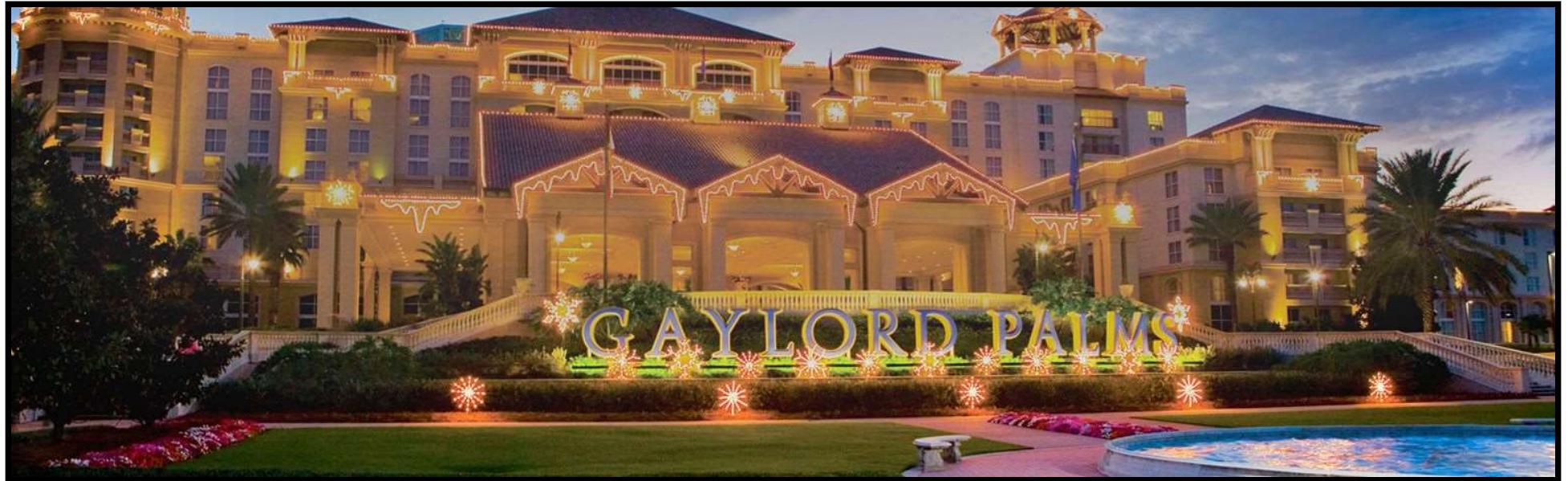
Stream (source)



Network (source)



Why are we talking about this at Qonnections?



Other than to show off appropriate gif/memes

Benefits of Using D3 with Qlik Sense

Specifically how does D3 integrate with Qlik Sense

A couple different reasons

Mashups

- Blend styles with the rest of your page
- Stand on the shoulders of other developers

Extensions

- Extend the capabilities of Qlik Sense
- Get an appreciation for bar charts

Ok Qlik Sense needs D3 but why does D3 benefit from Qlik Sense?

- Meta data
 - Min and Max
 - Glyph counts
 - Aggregations in D3 are a pain
 - Limits and methods on number of rows
 - Sorting and formatting

Branch blog post

Alright, let's integrate in Qlik Sense



Using D3 in Mashups

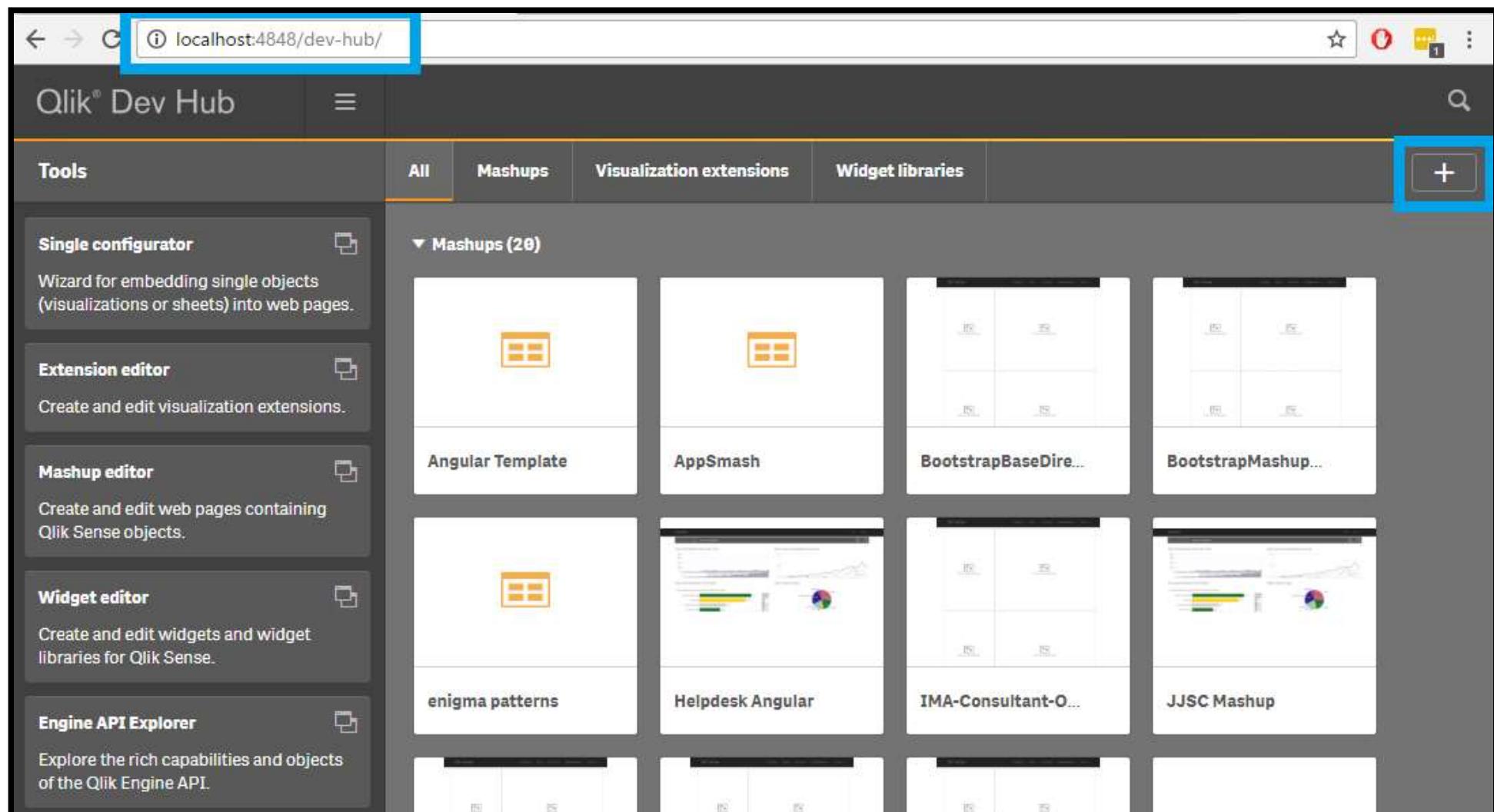
or how to use the capabilities APIs with D3.js



What are we building?



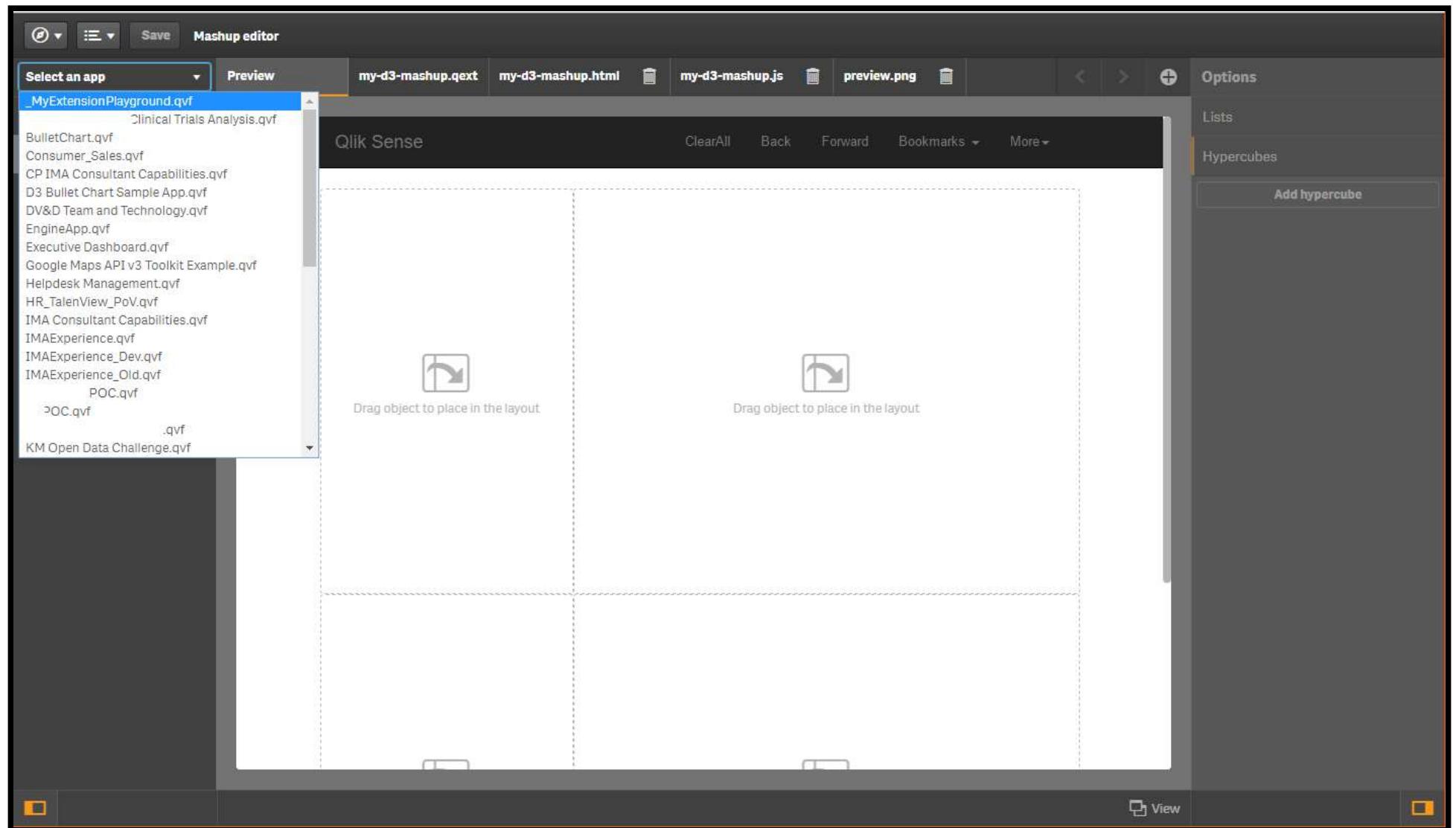
Create a new mashup



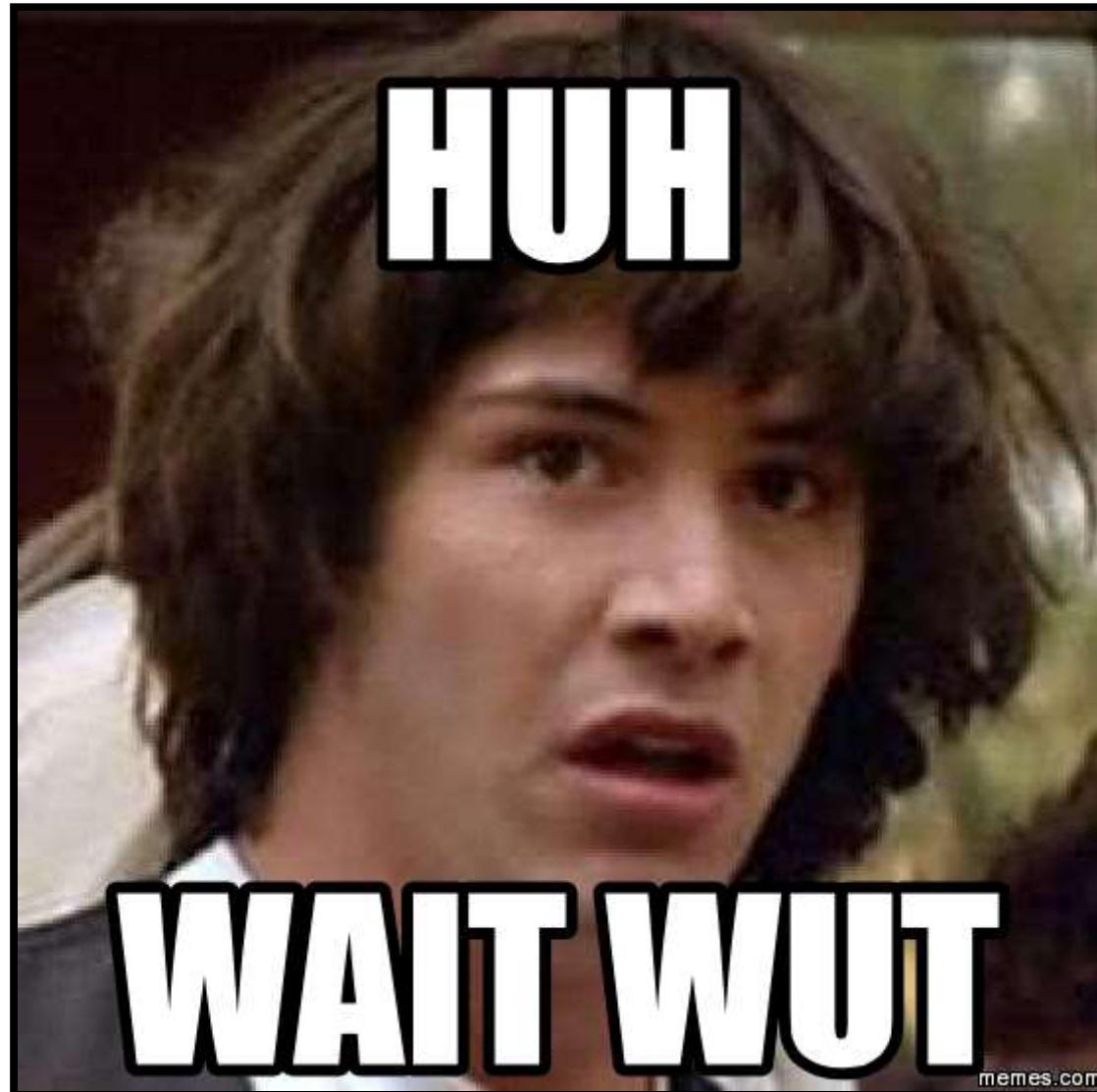
Create a new mashup

The screenshot shows the Qlik Dev Hub interface. The top navigation bar includes 'Qlik® Dev Hub' and a search icon. Below it, a toolbar has tabs for 'Tools' (selected), 'All', 'Mashups' (highlighted in orange), 'Visualization extensions', and 'Widget libraries'. A '+' button is in the top right corner. On the left, a sidebar lists tools: 'Single configurator', 'Extension editor', 'Mashup editor', 'Widget editor', and 'Engine API Explorer'. The main area displays a grid of 28 existing mashups, each with a preview thumbnail and name. A modal window titled 'Create new mashup' is open in the center. It contains fields for 'Name' (with 'my-d3-mashup' typed in) and 'Template' (set to 'Grid mashup template'). At the bottom of the modal are 'Cancel' and 'Create & edit' buttons.

Connect to a Qlik Sense app



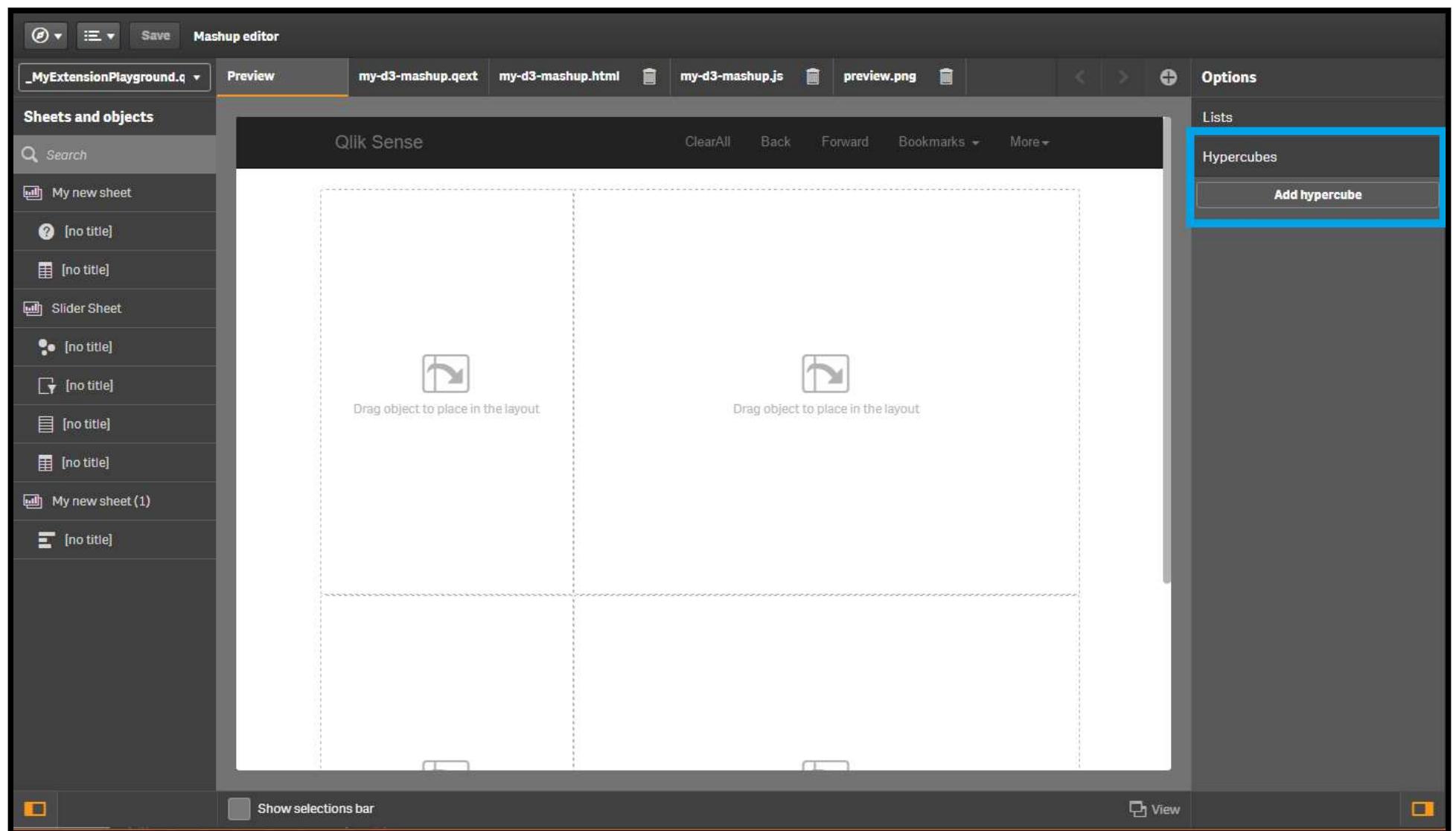
Add a hypercube



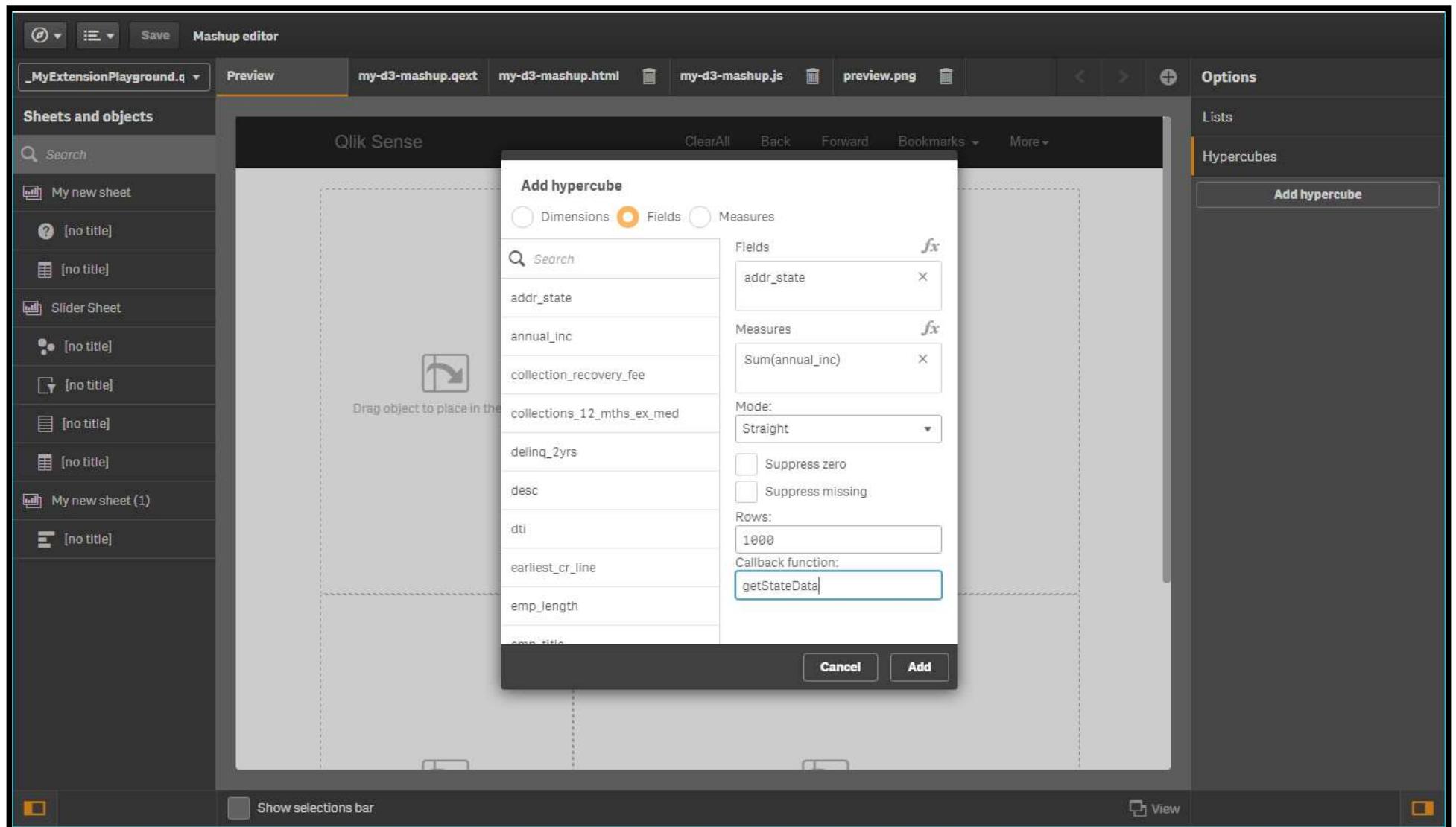
What in tarnation is a hypercube

- Define dimensions and measures
- Tell Qlik the max number of rows expected
- Create a callback function to handle the data returned
- User selects data, callback function is called again

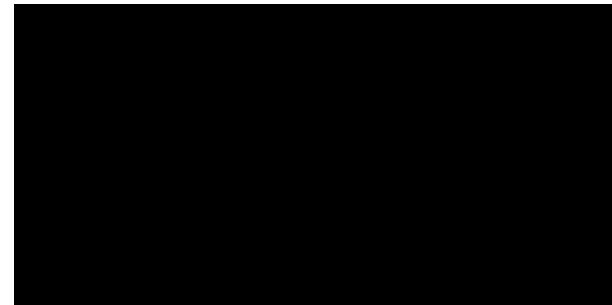
So about that hypercube....



Configuring the hypercube



What did you do!?



What did all that do?

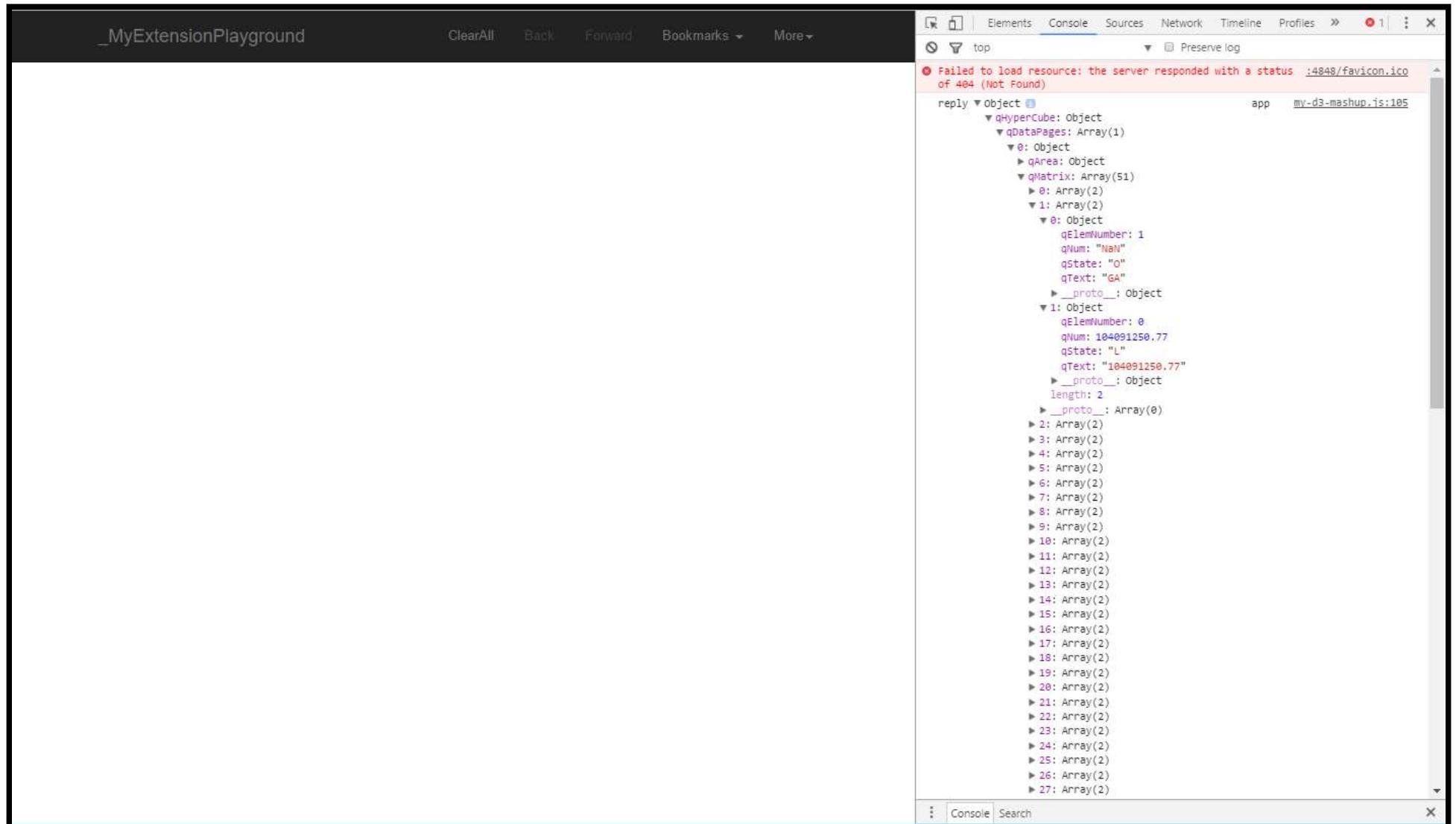
The screenshot shows the Qlik Sense Mashup editor interface. The top menu bar includes Save and Mashup editor. The left sidebar lists "Sheets and objects" such as "My new sheet", "[no title]", "Slider Sheet", and "[no title]". The main area is a code editor with tabs for "my-d3-mashup.qext", "my-d3-mashup.html", "my-d3-mashup.js" (which is selected), and "preview.png". The code in "my-d3-mashup.js" is as follows:

```
//create cubes and lists -- inserted here --
app.createCube({
    "qInitialDataFetch": [
        {
            "qHeight": 1000,
            "qWidth": 2
        }
    ],
    "qDimensions": [
        {
            "qDef": {
                "qFieldDefs": [
                    "addr_state"
                ]
            },
            "qNullSuppression": true,
            "qOtherTotalSpec": {
                "qOtherMode": "OTHER_OFF",
                "qSuppressOthers": true,
                "qOtherSortMode": "OTHER_SORT_DESCENDING",
                "qOtherCounted": {
                    "qv": "5"
                },
                "qOtherLimitMode": "OTHER_GE_LIMIT"
            }
        }
    ],
    "qMeasures": [
        {
            "qDef": {
                "qDef": "Sum(annual_inc)"
            },
            "qLabel": "Sum(annual_inc)",
            "qLibraryId": null,
            "qSortBy": {
                "qSortByState": 0,
                "qSortByFrequency": 0,
                "qSortByNumeric": 0,
                "qSortByAscii": 1,
                "qSortByLoadOrder": 0,
            }
        }
    ]
});
```

The right side of the interface shows "Lists" and "Hypercubes" sections, with "Add hypercube" being the active tab. A preview pane is visible on the right, showing a small thumbnail of the generated visualization.

Let's see the return values

The developer panel

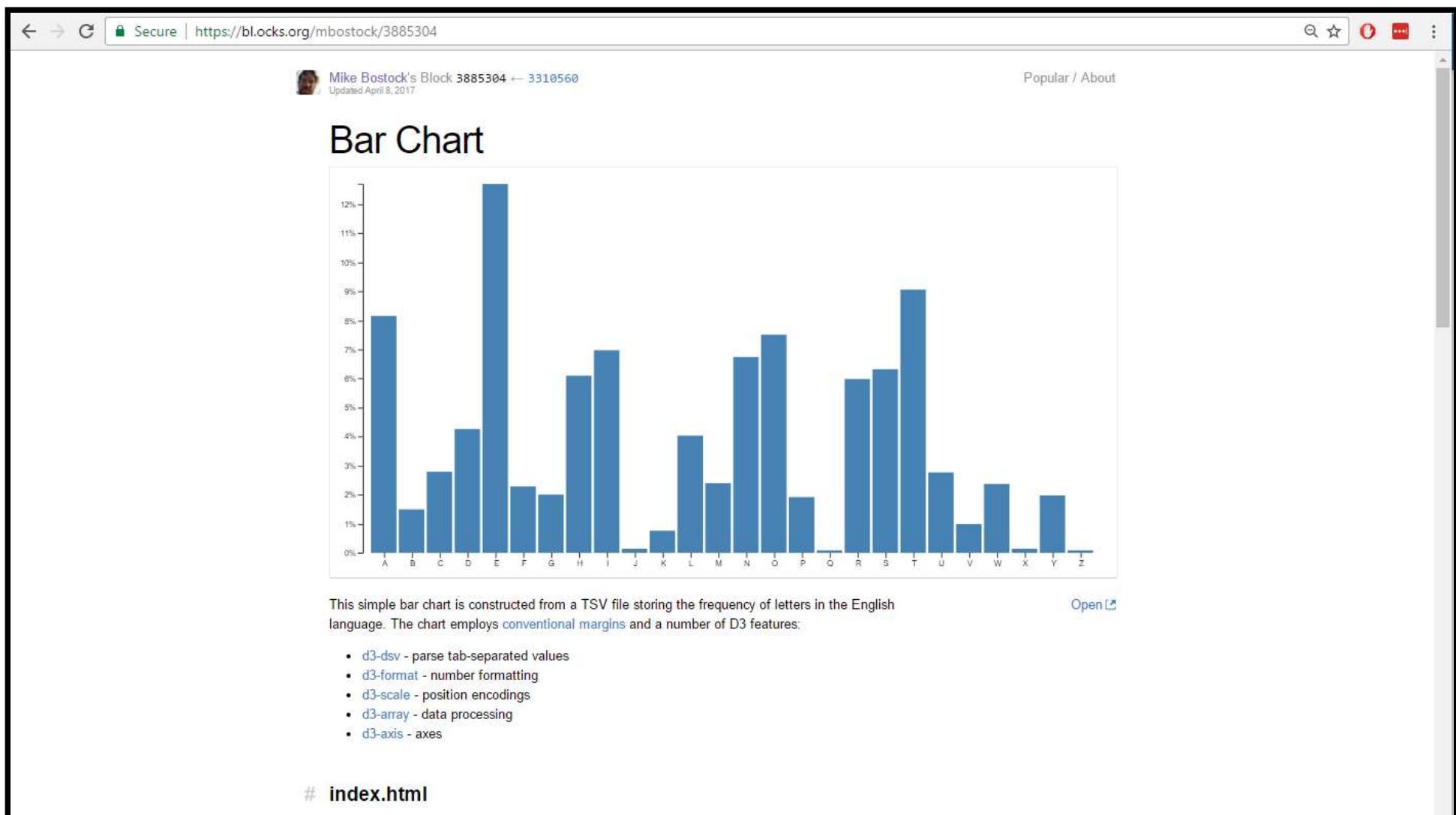


OH YEAH!!!!!!

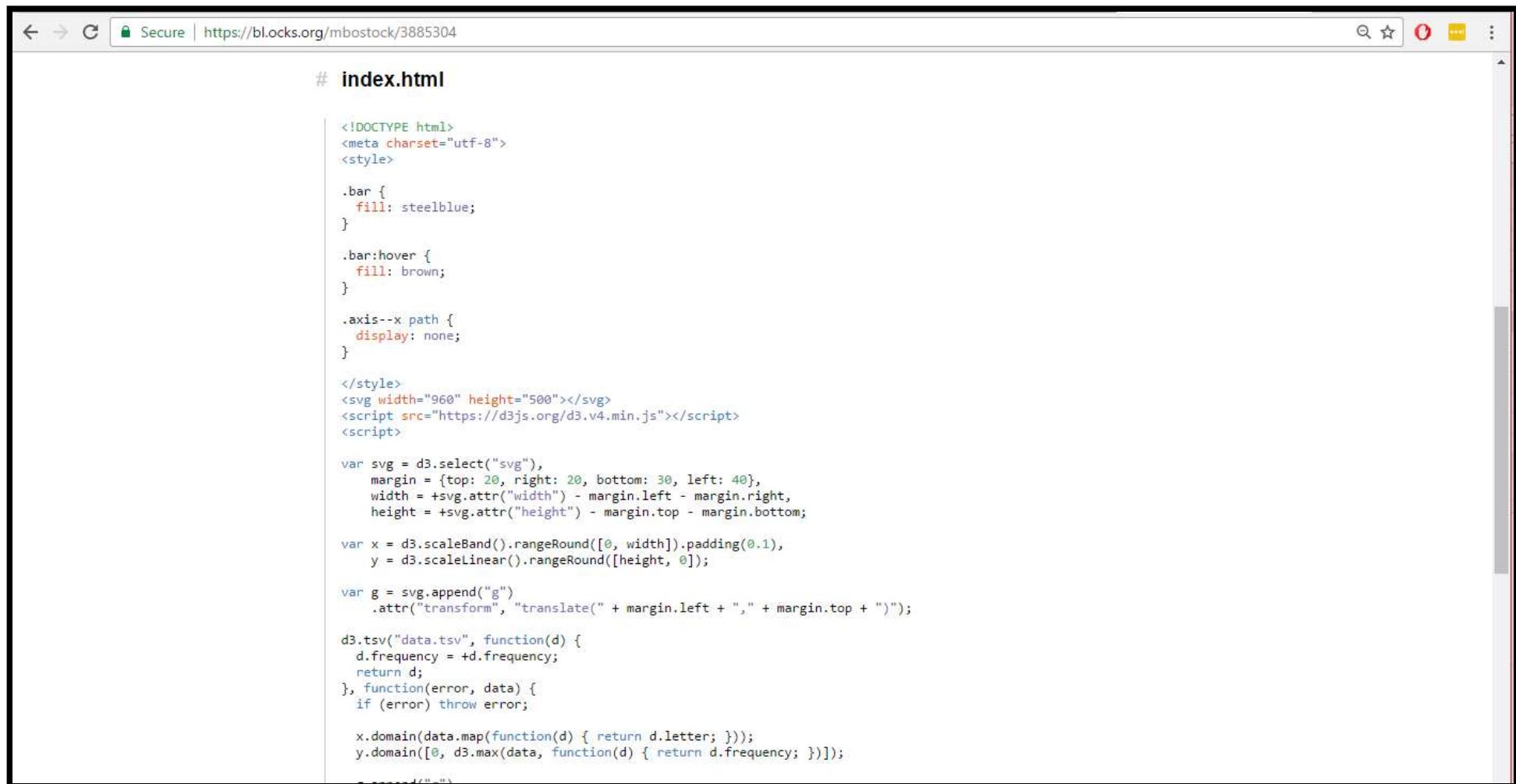


quickmeme.com

Go grab an example (source)



Get HTML/CSS/JS from example



The screenshot shows a web browser window with the URL <https://bl.ocks.org/mbostock/3885304>. The page title is "# index.html". The content of the page is the source code for a D3.js visualization. The code includes CSS styles for bars, an SVG element with dimensions, a script to load d3.js, and a JavaScript block to set up the visualization.

```
# index.html

<!DOCTYPE html>
<meta charset="utf-8">
<style>

.bar {
  fill: steelblue;
}

.bar:hover {
  fill: brown;
}

.axis--x path {
  display: none;
}

</style>
<svg width="960" height="500"></svg>
<script src="https://d3js.org/d3.v4.min.js"></script>
<script>

var svg = d3.select("svg"),
    margin = {top: 20, right: 20, bottom: 30, left: 40},
    width = +svg.attr("width") - margin.left - margin.right,
    height = +svg.attr("height") - margin.top - margin.bottom;

var x = d3.scaleBand().rangeRound([0, width]).padding(0.1),
    y = d3.scaleLinear().rangeRound([height, 0]);

var g = svg.append("g")
  .attr("transform", "translate(" + margin.left + "," + margin.top + ")");

d3.tsv("data.tsv", function(d) {
  d.frequency = +d.frequency;
  return d;
}, function(error, data) {
  if (error) throw error;

  x.domain(data.map(function(d) { return d.letter; }));
  y.domain([0, d3.max(data, function(d) { return d.frequency; })]);
});
```

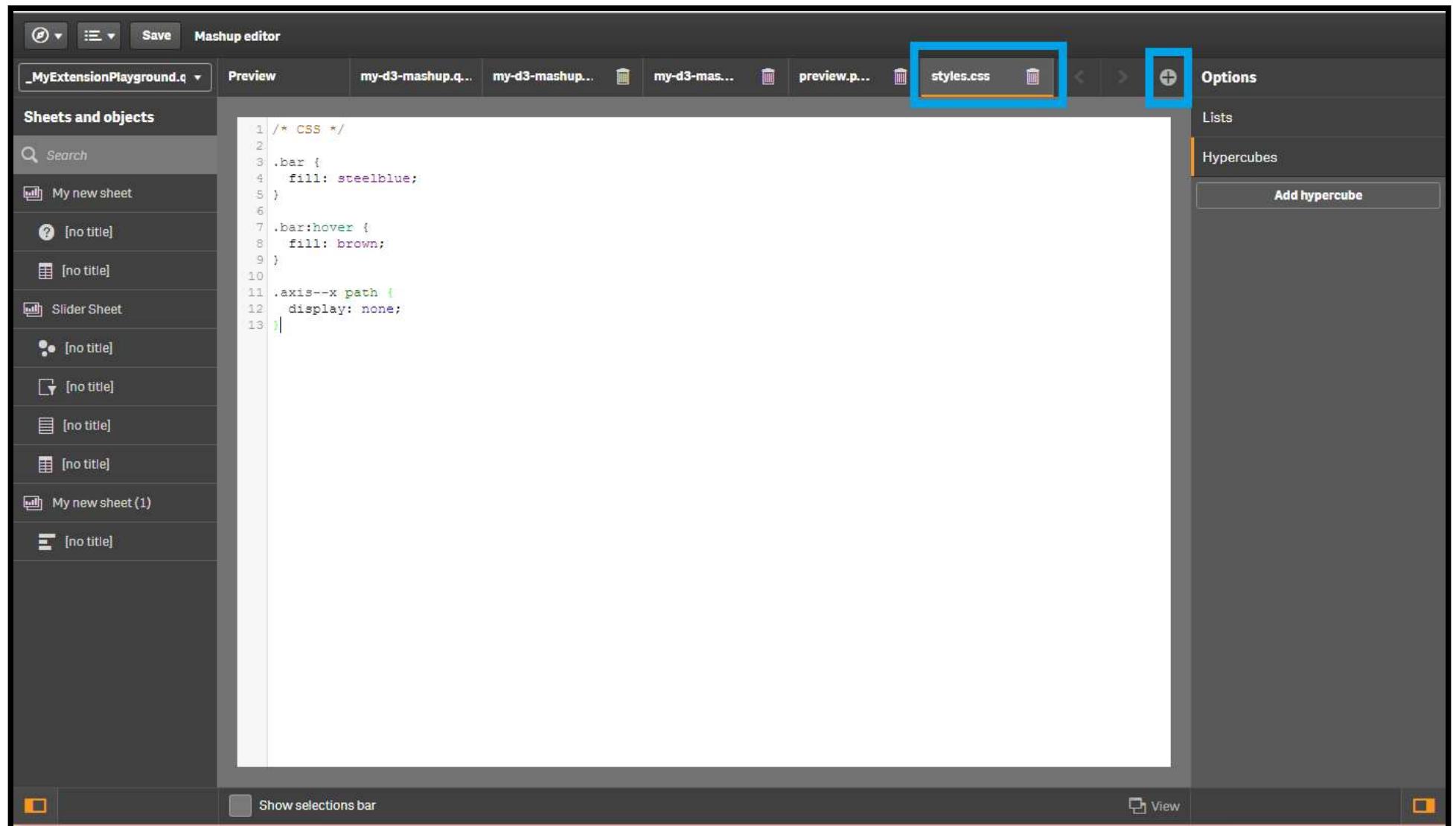
Alter HTML

Screenshot of the Mashup editor interface showing the 'my-d3-mashup.q...' tab selected. The code editor displays an HTML file with several rows and columns of div elements. A large blue arrow points from the right side of the screen towards the line of code containing the SVG element at line 118.

```
107 </div>
108
109 <div class="container " id="main" role="main">
110   <div class="alert alert-danger alert-dismissible" role="alert" style="display:none">
111     <button type="button" class="close" id="closeerr" aria-label="Close"><span aria-hidden="true">&amptimes</span>
112     <span id="errormsg"></span>
113   </div>
114   <div class="row">
115     <div class="col-sm-4 qvplaceholder" id="QV01">
116     </div>
117     <div class="col-sm-8 qvplaceholder" id="QV02">
118       <svg width="960" height="500"></svg>
119     </div>
120   </div>
121   <div class="row">
122     <div class="col-sm-4 qvplaceholder" id="QV03">
123     </div>
124     <div class="col-sm-8 qvplaceholder" id="QV04">
125     </div>
126   </div>
127   <!-- add more rows here if you want more visualizations -->
128 </div>
129
130 <!-- Bootstrap Modals -->
131 <div class="modal " id="createBmModal">
132   <div class="modal-dialog">
133     <div class="modal-content">
134       <div class="modal-header">
135         <button type="button" class="close" data-dismiss="modal" aria-label="Close"><span aria-hidden="true">&amptimes</span>
136         <h4 class="modal-title">Create bookmark</h4>
137       </div>
138       <div class="modal-body">
139         <form>
140           <label for="bmtitle" class="control-label">Title:</label>
141           <input type="text" class="form-control" id="bmtitle">
142           <label for="bmdesc" class="control-label">Description:</label>
143           <input type="text" class="form-control" id="bmdesc">
144         </form>
145       </div>
146       <div class="modal-footer">
```

The 'Options' panel on the right shows sections for 'Lists' and 'Hypercubes', with a button labeled 'Add hypercube'.

Insert css into script



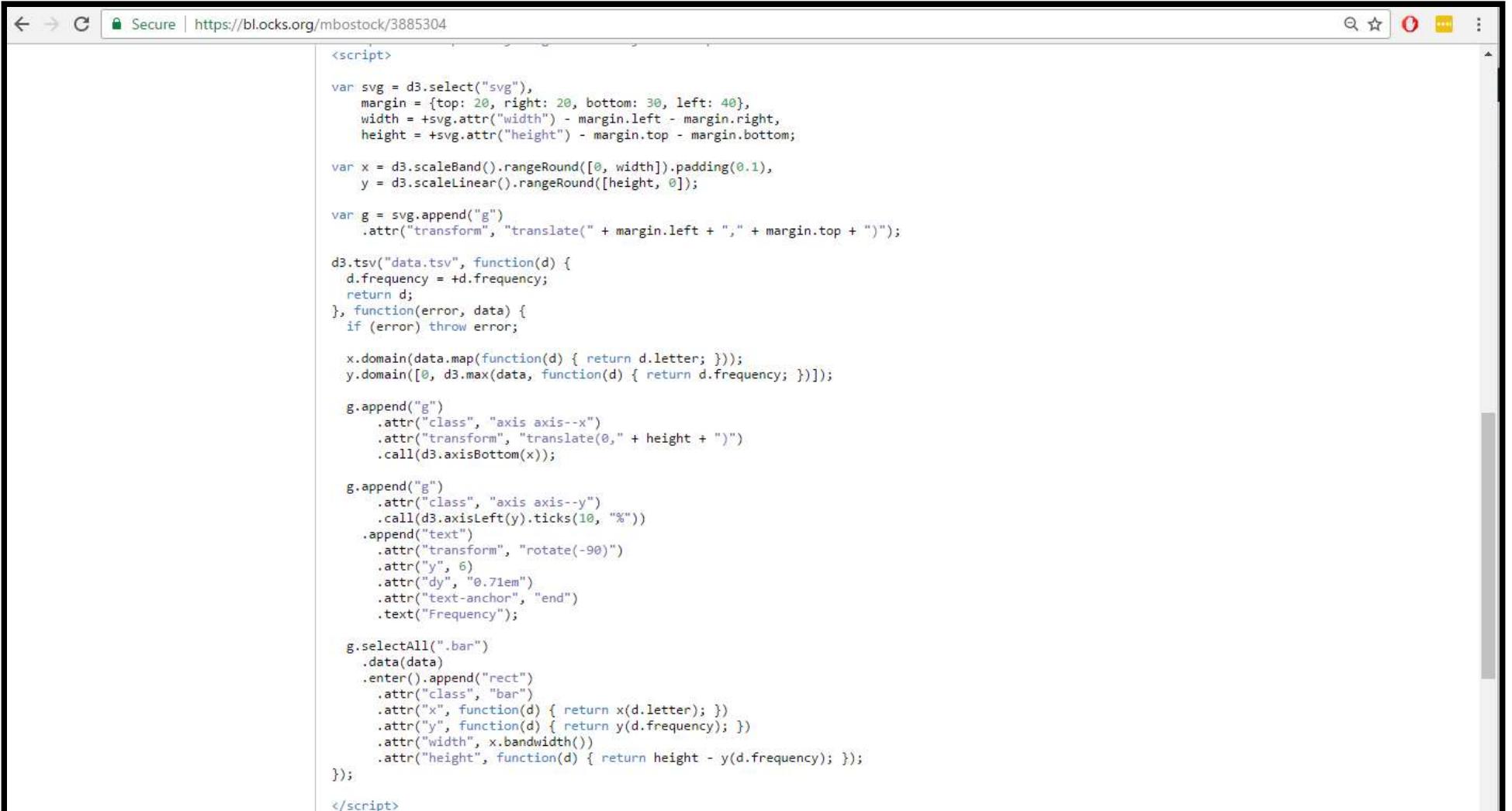
Load D3 library in

The screenshot shows the Qlik Sense Mashup editor interface. The top navigation bar includes Save and Options buttons. The main workspace displays a script file named "preview.qmashup". The script content is as follows:

```
1 /*  
2  * Bootstrap-based responsive mashup  
3  * @owner Enter your name here (xxx)  
4  */  
5 /*  
6  * Fill in host and port for Qlik engine  
7  */  
8 var prefix = window.location.pathname.substr( 0, window.location.pathname.toLowerCase().lastIndexOf( "/extension" ) );  
9  
10 var config = {  
11    host: window.location.hostname,  
12    prefix: prefix,  
13    port: window.location.port,  
14    isSecure: window.location.protocol === "https:"  
15 };  
16 //to avoid errors in workbench: you can remove this when you have added an app  
17 var app;  
18 require.config({  
19   baseUrl: (config.isSecure ? "https://" : "http://") + config.host + (config.port ? ":" + config.port : "")  
20 } );  
21  
22 require( [ "js/qlik", "//d3js.org/d3.v4.min.js" ], function ( qlik, d3 ) {  
23  
24   var control = false;  
25   qlik.setOnError( function ( error ) {  
26     $( '#popupText' ).append( error.message + "<br>" );  
27     if ( !control ) {  
28       control = true;  
29       $( '#popup' ).delay( 1000 ).fadeIn( 1000 ).delay( 11000 ).fadeOut( 1000 );  
30     }  
31   } );  
32  
33   $( "#closePopup" ).click( function () {  
34     $( '#popup' ).hide();  
35   } );  
36   if ( $( 'ul#qbmlist li' ).length === 0 ) {  
37     $( '#qbmlist' ).append( "<li><a>No bookmarks available</a></li>" );  
38   }  
39   $( "body" ).css( "overflow: hidden;" );  
40   function bookUi ( app ) {
```

The code uses RequireJS to load the Qlik Sense API and the D3.js library. It then sets up an error handler and a click event for a close button. If there are no bookmarks, it adds a placeholder message. Finally, it hides the scroll bar.

Get D3 from example



The screenshot shows a browser window displaying a data visualization. The URL in the address bar is <https://bl.ocks.org/mbostock/3885304>. The page content is a single script block containing D3.js code. The code defines a margin, scales for x and y axes, and creates a root group 'g'. It then reads data from 'data.tsv' and creates an axis at the bottom. A text label 'Frequency' is placed at the top right. Finally, it enters data to create bars.

```
<script>
var svg = d3.select("svg"),
    margin = {top: 20, right: 20, bottom: 30, left: 40},
    width = +svg.attr("width") - margin.left - margin.right,
    height = +svg.attr("height") - margin.top - margin.bottom;

var x = d3.scaleBand().rangeRound([0, width]).padding(0.1),
    y = d3.scaleLinear().rangeRound([height, 0]);

var g = svg.append("g")
    .attr("transform", "translate(" + margin.left + "," + margin.top + ")");

d3.tsv("data.tsv", function(d) {
  d.frequency = +d.frequency;
  return d;
}, function(error, data) {
  if (error) throw error;

  x.domain(data.map(function(d) { return d.letter; }));
  y.domain([0, d3.max(data, function(d) { return d.frequency; })]);

  g.append("g")
    .attr("class", "axis axis--x")
    .attr("transform", "translate(0," + height + ")")
    .call(d3.axisBottom(x));

  g.append("g")
    .attr("class", "axis axis--y")
    .call(d3.axisLeft(y).ticks(10, "%"))
    .append("text")
      .attr("transform", "rotate(-90)")
      .attr("y", 6)
      .attr("dy", "0.71em")
      .attr("text-anchor", "end")
      .text("Frequency");

  g.selectAll(".bar")
    .data(data)
    .enter().append("rect")
      .attr("class", "bar")
      .attr("x", function(d) { return x(d.letter); })
      .attr("y", function(d) { return y(d.frequency); })
      .attr("width", x.bandwidth())
      .attr("height", function(d) { return height - y(d.frequency); });
});

</script>
```

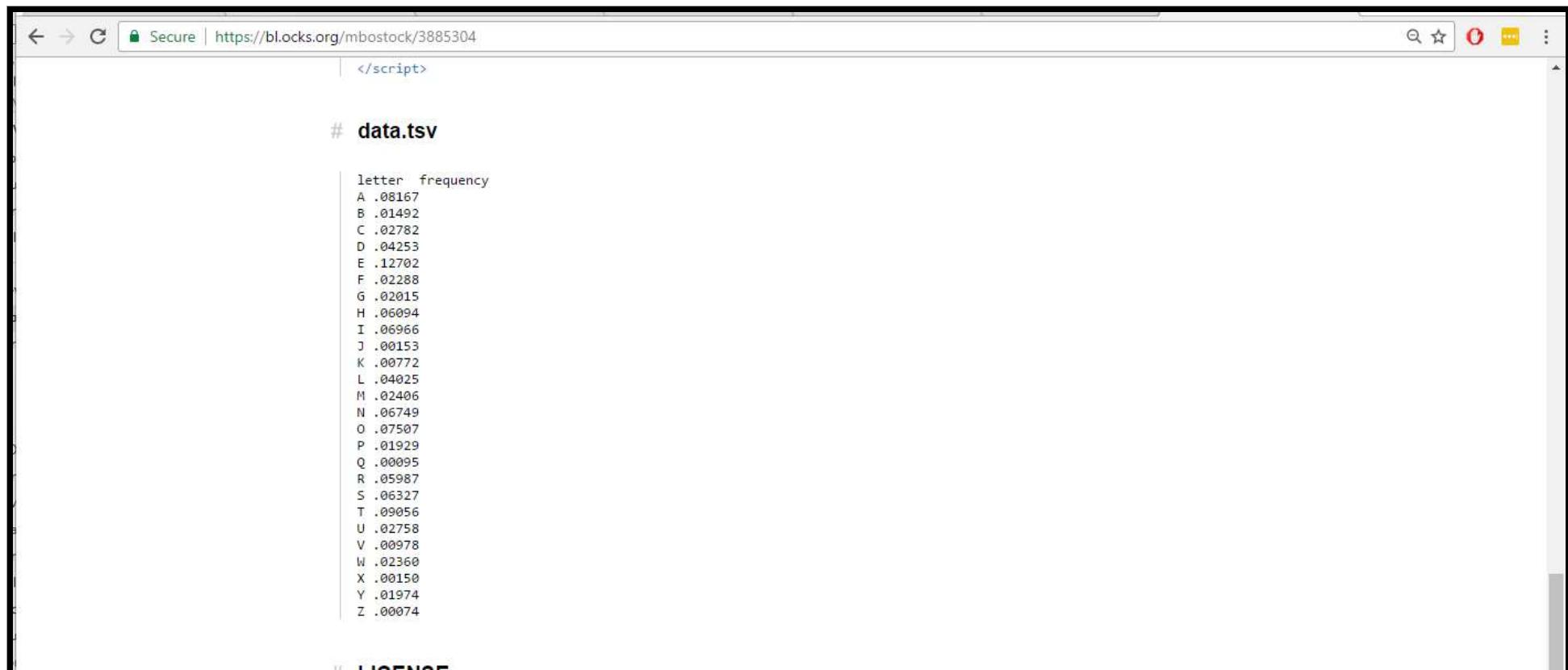
Copy D3 code into script

The screenshot shows the 'Mashup editor' interface with the title bar 'Mashup editor'. The left sidebar is titled 'Sheets and objects' and lists several sheets: 'MyExtensionPlayground.q...', 'Preview', 'my-d3-mashup.q...', 'my-d3-mashup...', 'my-d3-mash...', 'preview.p...', 'styles.css', and '+ Options'. The main area displays a block of D3.js code with line numbers from 100 to 139. The code defines margins, scales, and axes for an SVG visualization. The right sidebar includes sections for 'Lists' and 'Hypercubes', with a button labeled 'Add hypercube'.

```
100     } );
101
102
103     var svg = d3.select("svg"),
104         margin = {top: 20, right: 20, bottom: 30, left: 40},
105         width = +svg.attr("width") - margin.left - margin.right,
106         height = +svg.attr("height") - margin.top - margin.bottom;
107
108     var x = d3.scaleBand().rangeRound([0, width]).padding(0.1),
109         y = d3.scaleLinear().rangeRound([height, 0]);
110
111     var g = svg.append("g")
112         .attr("transform", "translate(" + margin.left + "," + margin.top + ")");
113
114     d3.tsv("data.tsv", function(d) {
115         d.frequency = +d.frequency;
116         return d;
117     }, function(error, data) {
118         if (error) throw error;
119
120         x.domain(data.map(function(d) { return d.letter; }));
121         y.domain([0, d3.max(data, function(d) { return d.frequency; })]);
122
123         g.append("g")
124             .attr("class", "axis axis--x")
125             .attr("transform", "translate(0," + height + ")")
126             .call(d3.axisBottom(x));
127
128         g.append("g")
129             .attr("class", "axis axis--y")
130             .call(d3.axisLeft(y).ticks(10, "%"))
131             .append("text")
132                 .attr("transform", "rotate(-90)")
133                 .attr("y", 6)
134                 .attr("dy", "0.7em")
135                 .attr("text-anchor", "end")
136                 .text("Frequency");
137
138         g.selectAll(".bar")
139             .data(data)
```

Before we connect to hypercube...

let's get the sample data



A screenshot of a web browser window displaying a list of data points. The URL in the address bar is <https://bl.ocks.org/mbostock/3885304>. The page content shows a script block containing a comment "# data.tsv" and a table of letter frequencies.

letter	frequency
A	.08167
B	.01492
C	.02782
D	.04253
E	.12702
F	.02288
G	.02015
H	.06094
I	.06966
J	.00153
K	.00772
L	.04025
M	.02406
N	.06749
O	.07507
P	.01929
Q	.00095
R	.05987
S	.06327
T	.09056
U	.02758
V	.00978
W	.02360
X	.00150
Y	.01974
Z	.00074

Drop the data file in the directory

C:\Users\kevin.mcgowan\Documents\Qlik\Sense\Extensions\my-d3-mashup\data....

File Edit Selection Find View Goto Tools Project Preferences Help

FOLDERS

my-d3-mashup

data.tsv

letter frequency

1 letter frequency

2 A .08167

3 B .01492

4 C .02782

5 D .04253

6 E .12702

7 F .02288

8 G .02015

9 H .06094

10 I .06966

11 J .00153

12 K .00772

13 L .04025

14 M .02406

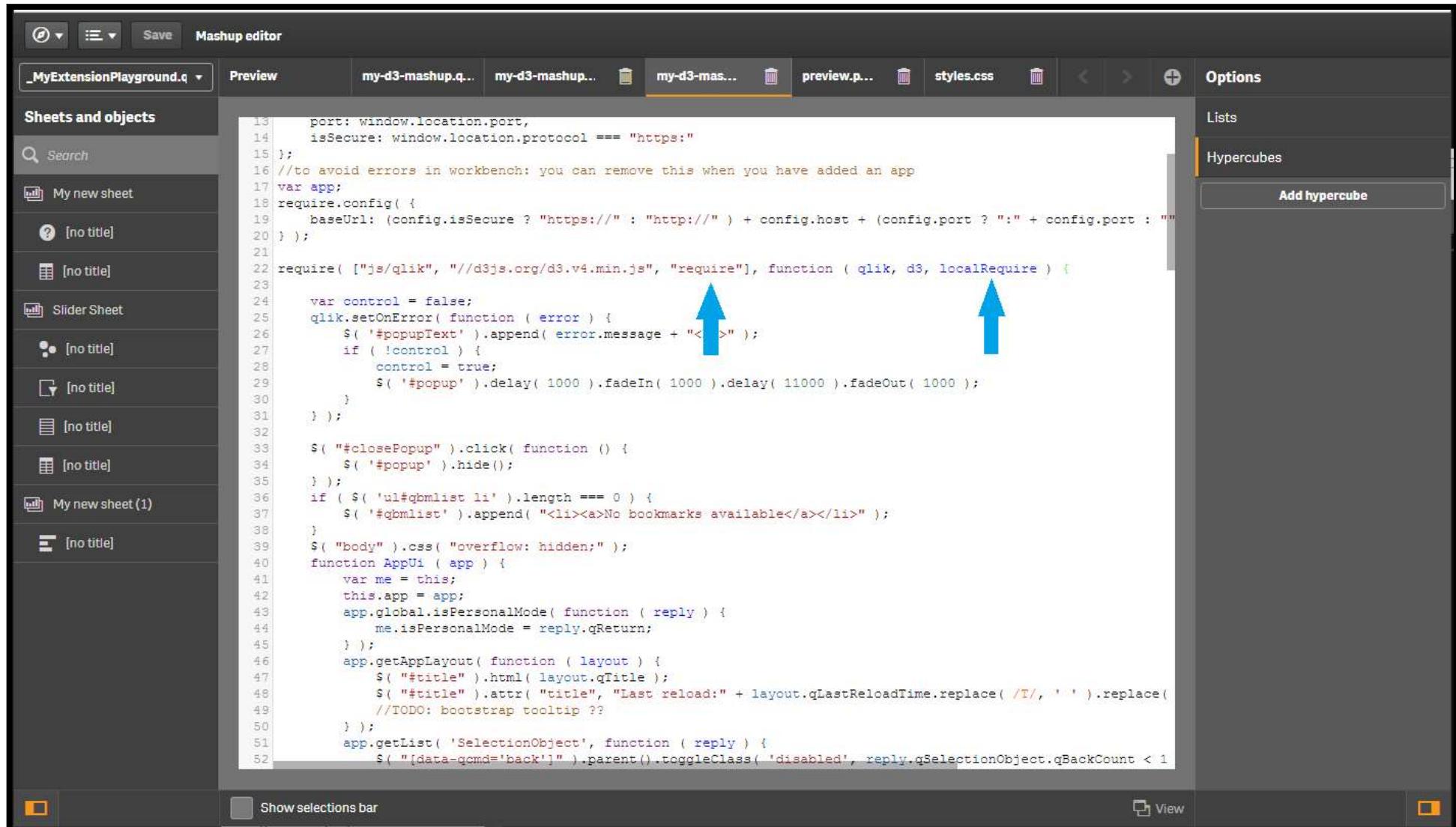
15 N .06749

Organize Open

This PC > Documents > Qlik > Sense > Extensions > my-d3-mashup

<input type="checkbox"/> Name	Date modified	Type	Size
<input checked="" type="checkbox"/> data.tsv	4/20/2017 11:23 AM	TSV File	1 KB
<input checked="" type="checkbox"/> my-d3-mashup.html	4/19/2017 3:38 PM	HTML File	5 KB
<input checked="" type="checkbox"/> my-d3-mashup.js	4/19/2017 3:35 PM	JS File	6 KB
<input type="checkbox"/> my-d3-mashup.qext	4/18/2017 8:24 AM	QEXT File	1 KB
<input checked="" type="checkbox"/> preview.png	4/18/2017 8:24 AM	PNG File	14 KB
<input checked="" type="checkbox"/> styles.css	4/19/2017 3:35 PM	CSS File	1 KB
<input type="checkbox"/> wbfolder.wbl	4/19/2017 3:16 PM	WBL File	1 KB

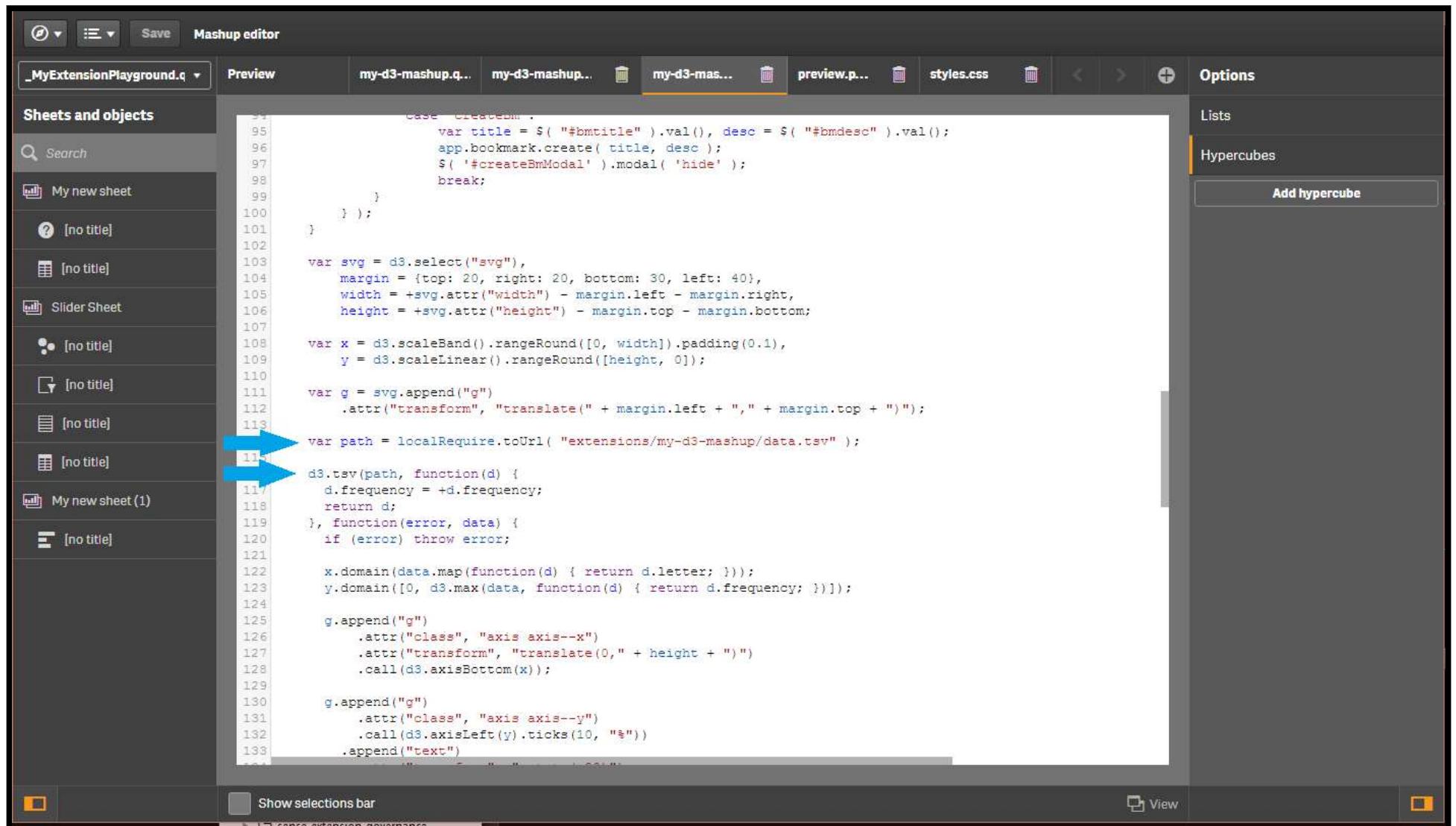
Load an instance of require.js



The screenshot shows the Qlik Sense Mashup editor interface. The main window displays a code editor with a RequireJS configuration file. The file contains code for setting up a Qlik app, including URLs for Qlik and D3, and logic for handling errors and bookmarks. Two blue arrows point upwards from the bottom of the code area towards the top of the editor window, likely indicating a specific section of the code or a step in the process.

```
13 port: window.location.port,
14   isSecure: window.location.protocol === "https:"
15 };
16 //to avoid errors in workbench: you can remove this when you have added an app
17 var app;
18 require.config( {
19   baseUrl: (config.isSecure ? "https://" : "http://") + config.host + (config.port ? ":" + config.port : "")
20 } );
21
22 require( ["js/qlik", "//d3js.org/d3.v4.min.js", "require"], function ( qlik, d3, localRequire ) {
23
24   var control = false;
25   qlik.setOnError( function ( error ) {
26     $( '#popupText' ).append( error.message + "<br>" );
27     if ( !control ) {
28       control = true;
29       $( '#popup' ).delay( 1000 ).fadeIn( 1000 ).delay( 11000 ).fadeOut( 1000 );
30     }
31   });
32
33   $( "#closePopup" ).click( function () {
34     $( '#popup' ).hide();
35   });
36   if ( $( 'ul#qbmList li' ).length === 0 ) {
37     $( '#qbmList' ).append( "<li><a>No bookmarks available</a></li>" );
38   }
39   $( "body" ).css( "overflow: hidden;" );
40   function AppUi ( app ) {
41     var me = this;
42     this.app = app;
43     app.global.isPersonalMode( function ( reply ) {
44       me.isPersonalMode = reply.qReturn;
45     });
46     app.getAppLayout( function ( layout ) {
47       $( "#title" ).html( layout.qTitle );
48       $( "#title" ).attr( "title", "Last reload:" + layout.qLastReloadTime.replace( /T/, ' ' ).replace(
49         //TODO: bootstrap tooltip ???
50       );
51       app.getList( 'SelectionObject', function ( reply ) {
52         $( "[data-qcmd='back']" ).parent().toggleClass( 'disabled', reply.qSelectionObject.qBackCount < 1 );
```

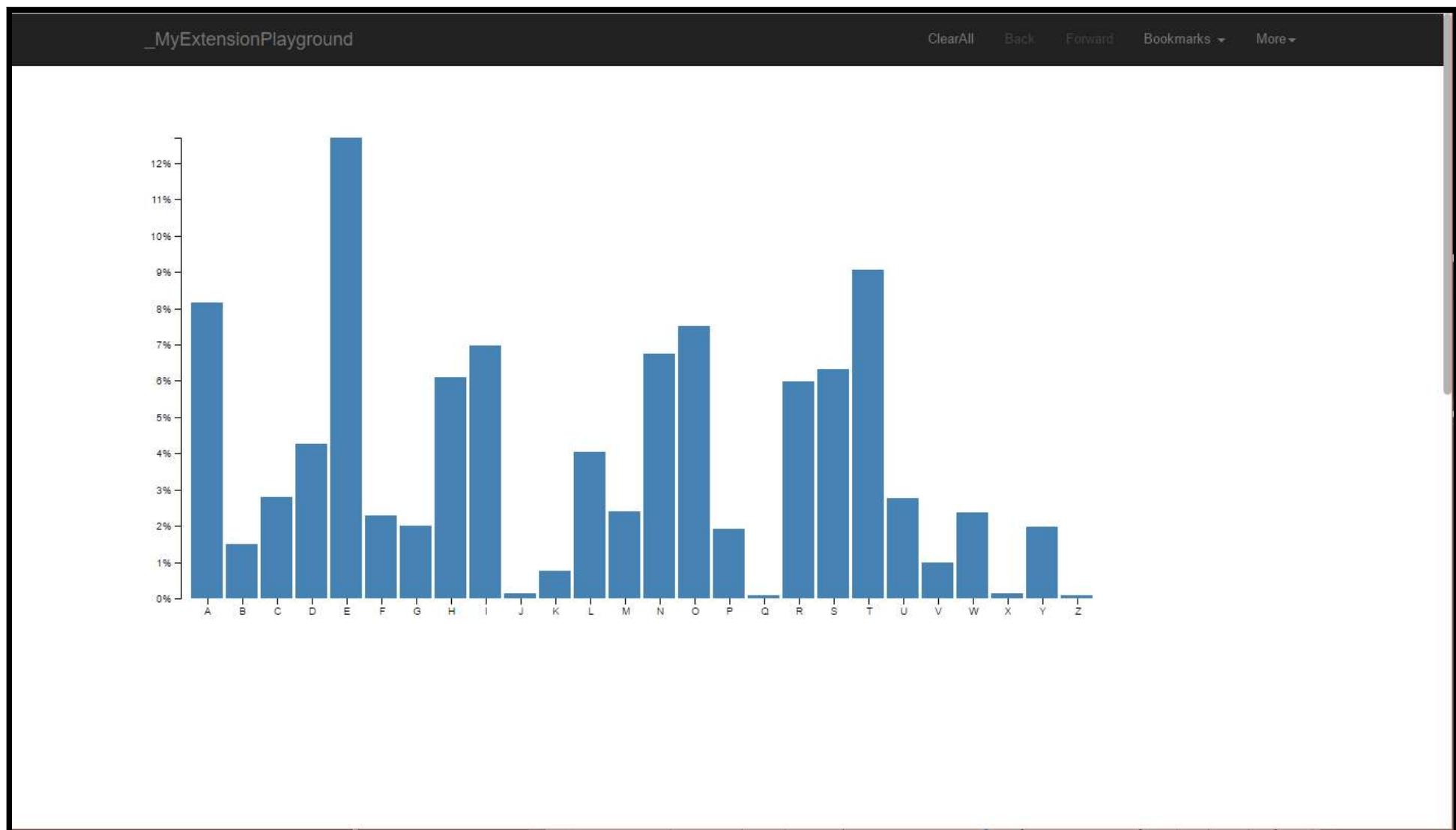
Reference the tsv file in js

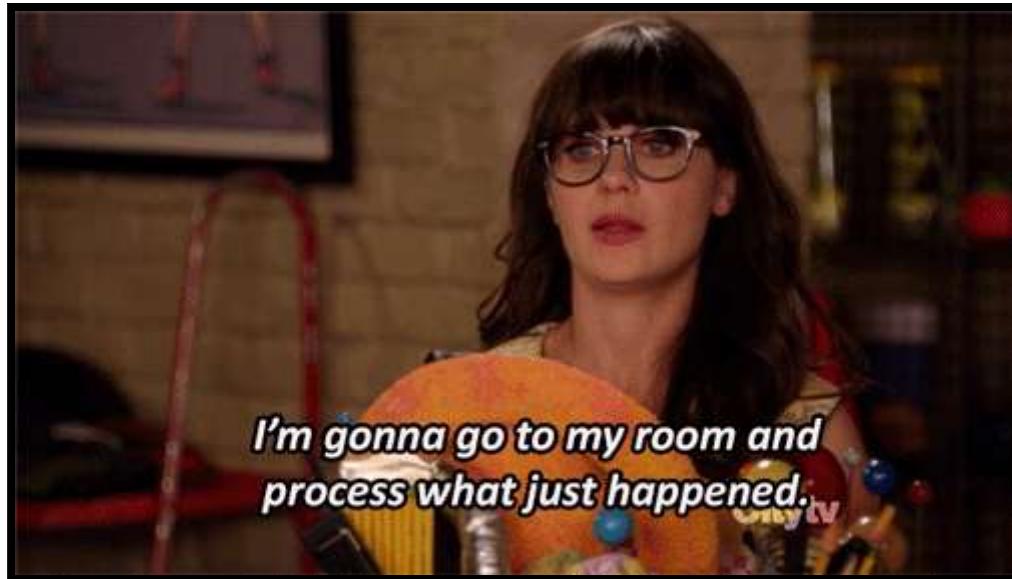


The screenshot shows the SAP Fiori Mashup editor interface. The left sidebar lists various sheets and objects, including "My new sheet" and several unnamed sheets. The main area displays a portion of a JavaScript file with line numbers 94 through 133. A blue arrow points to line 114, which contains the line `var path = localRequire.toUrl("extensions/my-d3-mashup/data.tsv");`. Another blue arrow points to line 115, which contains the line `d3.tsv(path, function(d) {`. The right side of the screen shows the "Lists" and "Hypercubes" panels.

```
94     case "createBm":  
95         var title = $( "#bmtitle" ).val(), desc = $( "#bmdesc" ).val();  
96         app.bookmark.create( title, desc );  
97         $( '#createBmModal' ).modal( 'hide' );  
98         break;  
99     }  
100 } );  
101  
102  
103 var svg = d3.select("svg"),  
104     margin = {top: 20, right: 20, bottom: 30, left: 40},  
105     width = +svg.attr("width") - margin.left - margin.right,  
106     height = +svg.attr("height") - margin.top - margin.bottom;  
107  
108 var x = d3.scaleBand().rangeRound([0, width]).padding(0.1),  
109     y = d3.scaleLinear().rangeRound([height, 0]);  
110  
111 var g = svg.append("g")  
112     .attr("transform", "translate(" + margin.left + "," + margin.top + ")");  
113  
114 var path = localRequire.toUrl( "extensions/my-d3-mashup/data.tsv" );  
115 d3.tsv(path, function(d) {  
116     d.frequency = +d.frequency;  
117     return d;  
118 }, function(error, data) {  
119     if (error) throw error;  
120  
121     x.domain(data.map(function(d) { return d.letter; }));  
122     y.domain([0, d3.max(data, function(d) { return d.frequency; })]);  
123  
124     g.append("g")  
125         .attr("class", "axis axis--x")  
126         .attr("transform", "translate(0," + height + ")")  
127         .call(d3.axisBottom(x));  
128  
129     g.append("g")  
130         .attr("class", "axis axis--y")  
131         .call(d3.axisLeft(y).ticks(10, "%"))  
132         .append("text")  
133             .attr("dy", -10)  
             .attr("dx", 10)  
             .text("Frequency")  
);  
});
```

Eureka!





I'm gonna go to my room and process what just happened.

Citytv

Link to source so far

Get data from your hypercube

But first prep data (source)

The screenshot shows a GitHub repository page for 'mcgovey/QlikSenseD3Utils'. The repository was forked from 'brianwmunz/QlikSenseD3Utils'. The main navigation bar includes links for 'Pull requests', 'Issues', and 'Gist'. Below the navigation bar, there are buttons for 'Unwatch' (1), 'Star' (0), 'Fork' (7), and a 'Code' tab which is currently selected. Other tabs include 'Pull requests' (0), 'Projects' (0), 'Wiki', 'Pulse', 'Graphs', and 'Settings'. A summary section below the tabs shows '28 commits', '1 branch', '0 releases', and '4 contributors'. The 'Branch: master' dropdown is set to 'master'. There are buttons for 'New pull request', 'Create new file', 'Upload files', 'Find file', and 'Clone or download'. A note states 'This branch is 5 commits ahead, 1 commit behind brianwmunz:master.' A commit history entry shows 'mcgovey committed on GitHub updated readme for createJSONObj changes' with a 'Latest commit 7e8bdae on Feb 7'. Another entry shows '.gitattributes' with an 'Initial Commit' and a date of '3 years ago'.

Minor edits

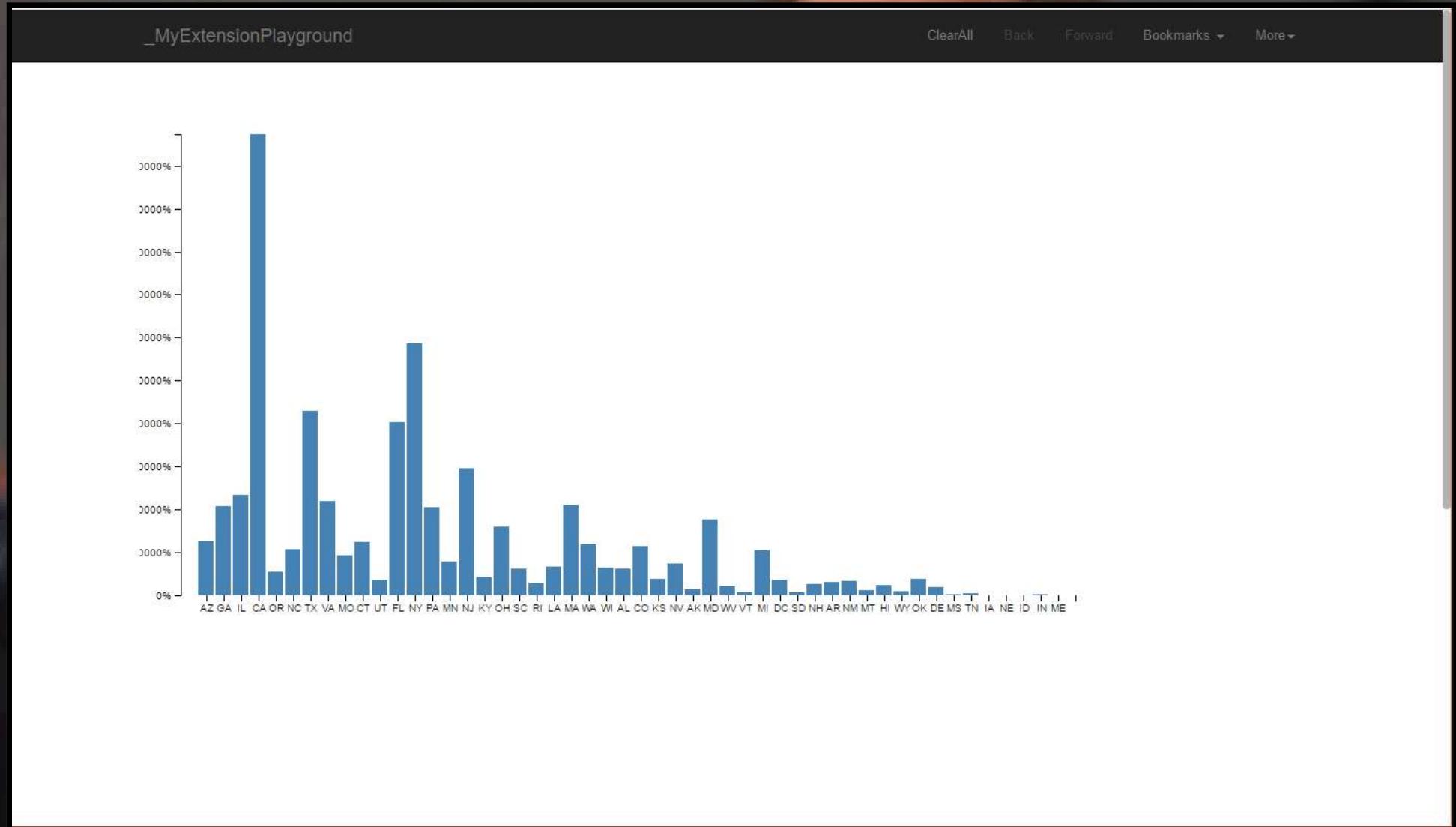
- Put the D3 code inside the callback function
- Clear the SVG before writing to it
- Comment out the call to the tsv
- Point the chart data to the hypercube variable

[Source with these changes](#)

Modified Example - Changing the Data

```
function getStateData(reply, app) {
    //remove all d3 elements from svg
    $('#QV01 svg').empty();
    //store cleansed data
    var data = senseD3.createJSONObj(reply);
    ...
    g.selectAll(".bar")
        .data(data)
        .enter().append("rect")
            .attr("class", "bar")
            .attr("x", function(d) { return x(d.dim_0); })
            .attr("y", function(d) { return y(d.meas_0); })
            .attr("width", x.bandwidth())
            .attr("height", function(d) { return height - y(d.meas_0); })
}
```

Refresh and...





We're done right? Right!?

[Link to source](#)

Well kinda

Some other things we would want to handle

- Window size changes
- Smoothing data changes
- Fix hard coded D3 selections
- Add selection listeners to the chart
- Leverage hypercube data

Selections (using the capability apis)

```
.on("click", function(d) {  
    app.field('addr_state').select([d.dim_0_id], true, true);  
}) ;
```

Documentation

Capabilities API

What else can we do?

- Make/clear selections
- Forward/Backward through Selections
- Navigate to an app page
- Alternative States

Much More

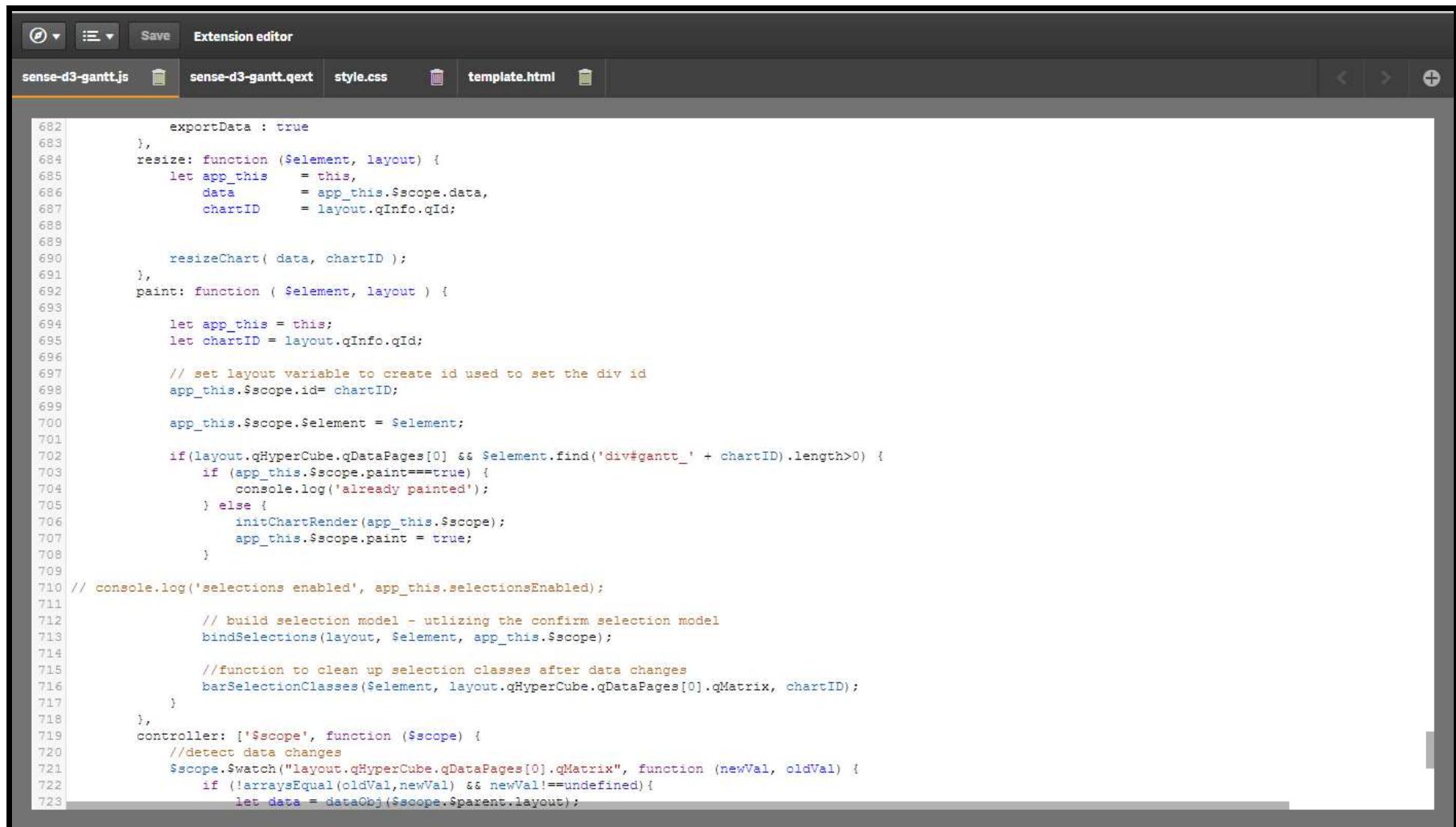
Mashup source code
source

Using D3 in Extensions

Mostly the same as mashups, a few differences

- No hypercube creation, controlled by properties
- More error handling and data validation
- Handle data changes in a different way

What's different?



The screenshot shows a code editor interface with the following tabs:

- sense-d3-gantt.js
- sense-d3-gantt.qext
- style.css
- template.html

The `sense-d3-gantt.js` tab is active and displays the following code:

```
682     exportData : true
683 },
684     resize: function ($element, layout) {
685         let app_this = this,
686             data = app_this.$scope.data,
687             chartID = layout.qInfo.qId;
688
689         resizeChart( data, chartID );
690     },
691     paint: function ( $element, layout ) {
692
693         let app_this = this;
694         let chartID = layout.qInfo.qId;
695
696         // set layout variable to create id used to set the div id
697         app_this.$scope.id= chartID;
698
699         app_this.$scope.$element = $element;
700
701         if(layout.qHyperCube.qDataPages[0] && $element.find('div#gantt_' + chartID).length>0) {
702             if (app_this.$scope.paint==true) {
703                 console.log('already painted');
704             } else {
705                 initChartRender(app_this.$scope);
706                 app_this.$scope.paint = true;
707             }
708         }
709
710 //        console.log('selections enabled', app_this.selectionsEnabled);
711
712         // build selection model - utilizing the confirm selection model
713         bindSelections(layout, $element, app_this.$scope);
714
715         //function to clean up selection classes after data changes
716         barSelectionClasses($element, layout.qHyperCube.qDataPages[0].qMatrix, chartID);
717     }
718 },
719     controller: ['$scope', function ($scope) {
720         //detect data changes
721         $scope.$watch("layout.qHyperCube.qDataPages[0].qMatrix", function (newVal, oldVal) {
722             if (!arraysEqual(oldVal,newVal) && newVal!==undefined){
723                 let data = dataObj($scope.$parent.layout);
```

How does our data look different

- 'Element' is roughly equal to 'App' from Mashup
(different contexts)
- 'Layout' is roughly equal to 'Reply' from HC
 - Different data types could come in
- Varying numbers of dimensions and measures

What happens when our data changes in an extension?

- Redraw the full chart (cop out)
- Use Angular to detect data changes (difficult to pull off)
 - Too heavy for this but...[source](#)

What about using APIs with D3 extensions?

- Extension APIs used to set extension properties
 - Color picker
 - SelectValues
- Capability APIs from earlier still available

So what's possible?

An example!

Where can we go from here?

- D3 Project Page
- Scott Murray Tutorial***
- Blocks
- D3 Documentation
- Sample Slideshare Presentation

Where can we go from here?

These slides at:

<https://mcgovey.github.io/qonnections-2017-d3>

Supporting repo:

<https://github.com/mcgovey/qonnections-2017-d3>