

# Unlocking the Potential of D3.js with Qlik Sense APIs

*Using a popular JavaScript library to  
build things beyond the Qlik Sense  
out-of-box capabilities*

# 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?

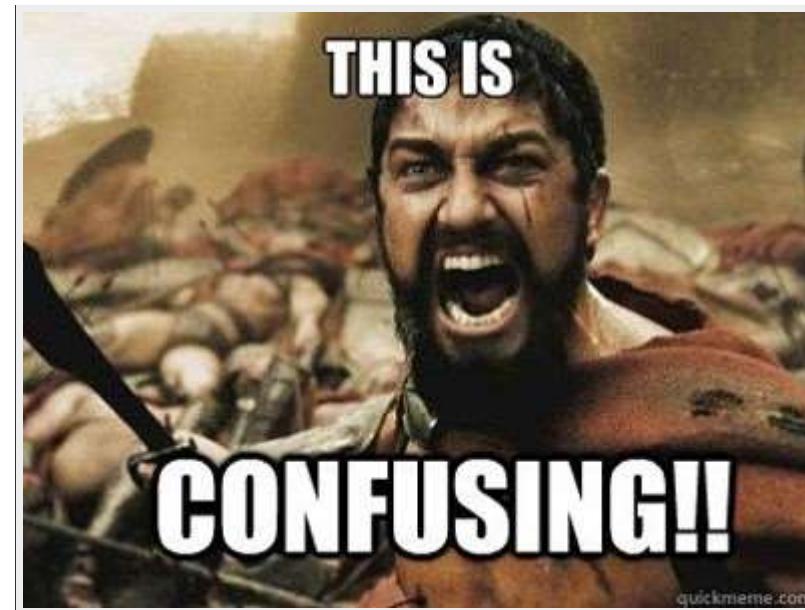
# What is D3?

# 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.*





# 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 open to the 'Elements' tab. On the left, there is a preview pane displaying six identical paragraphs, each containing the text 'New paragraph!'. On the right, the DOM tree is visible, showing the following structure:

```
<!DOCTYPE html>
<html lang="en">
  >#shadow-root (open)
    ><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 first paragraph node under the body element is highlighted with a grey background, indicating it is the result of the `d3.selectAll("p")` selection.

```
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
```

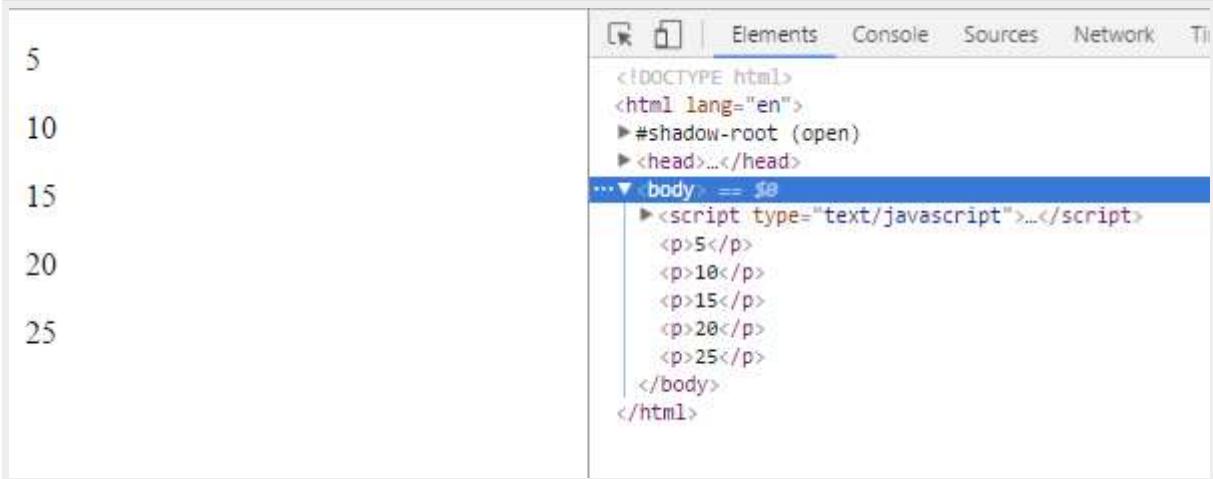
```
translate: true
webkitdropzone: ""
__data__: 5
► __proto__: HTMLParagraphElement
► 1: p
► 2: p
► 3: p
► 4: p
► parentNode: document
length: 5
► __proto__: Array(0)
length: 1
► __proto__: Array(0)
```

# 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 a browser's developer tools with the "Elements" tab selected. The DOM tree on the right side displays the following structure:

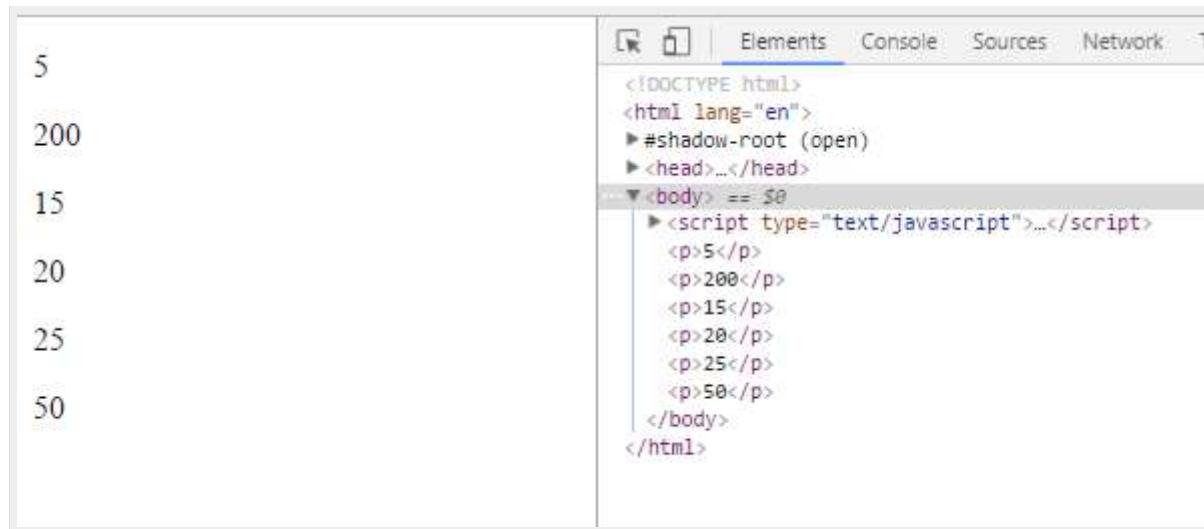
```
<!DOCTYPE html>
<html lang="en">
  >#shadow-root (open)
    ><head>...</head>
    ...▼ body == $0
      ><script type="text/javascript">...</script>
      ><p>5</p>
      ><p>10</p>
      ><p>15</p>
      ><p>20</p>
      ><p>25</p>
    </body>
</html>
```

The number 5 is visible on the left side of the screenshot, corresponding to the first item in the list.

# 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; }) ;
```

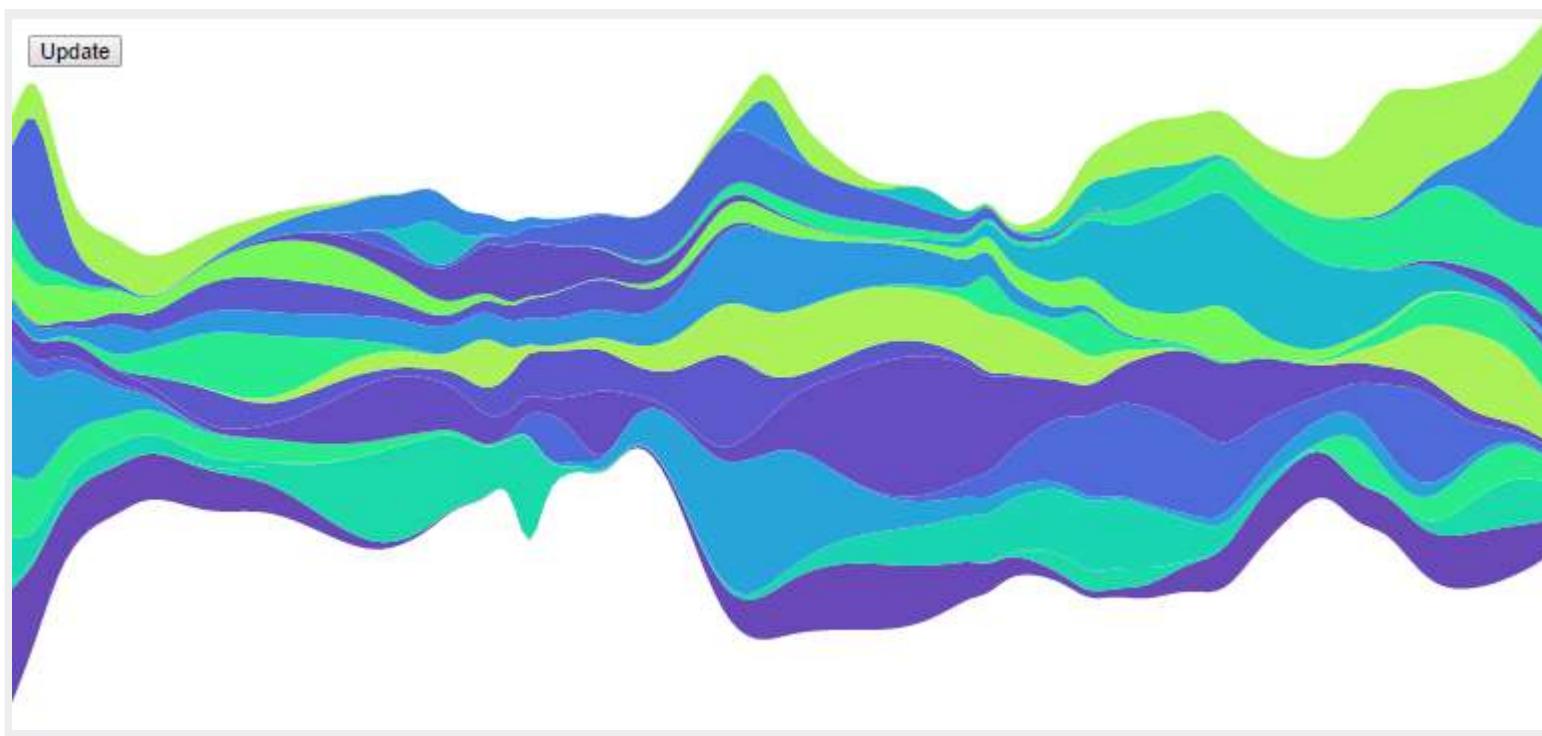
# Modified Example - Results



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 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>200</p>
      ><p>15</p>
      ><p>20</p>
      ><p>25</p>
      ><p>50</p>
    </body>
</html>
```

# Cool chart but why do I care?

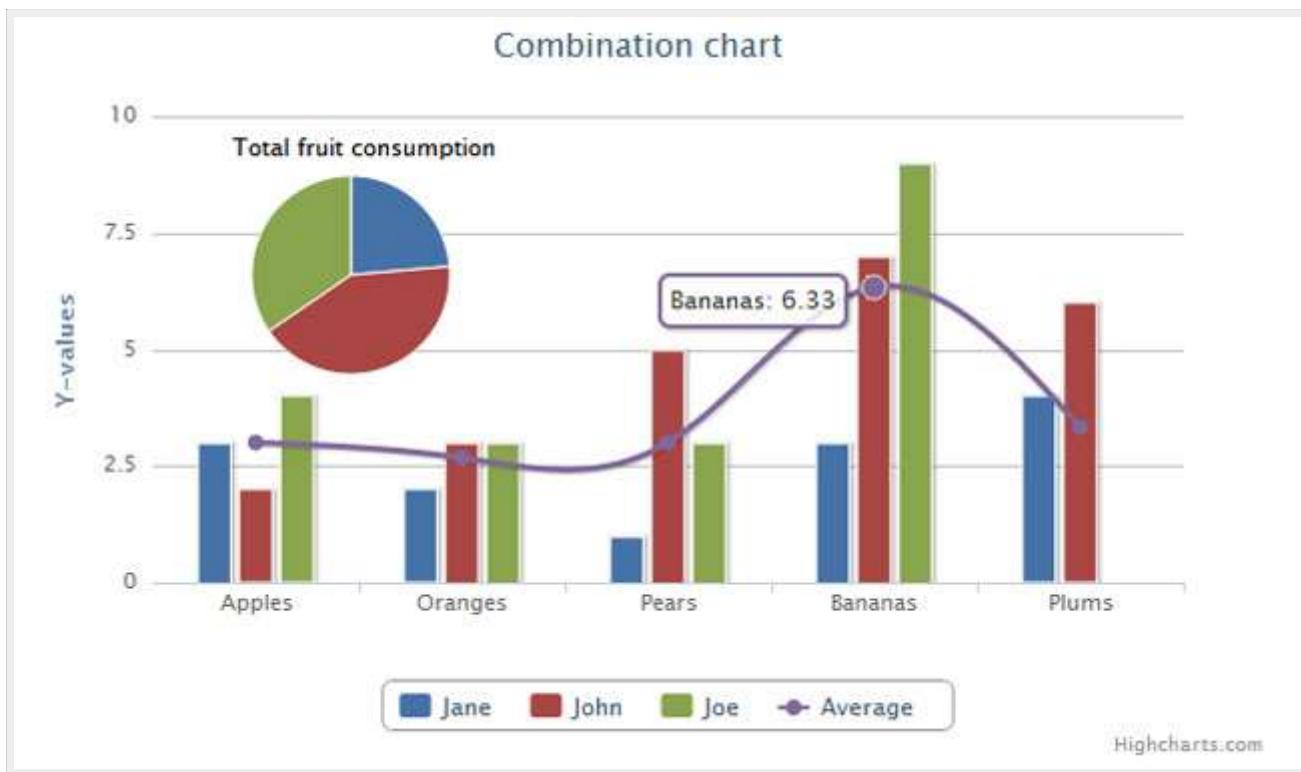


# Dynamic data manipulation is really useful for data visualization

- 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



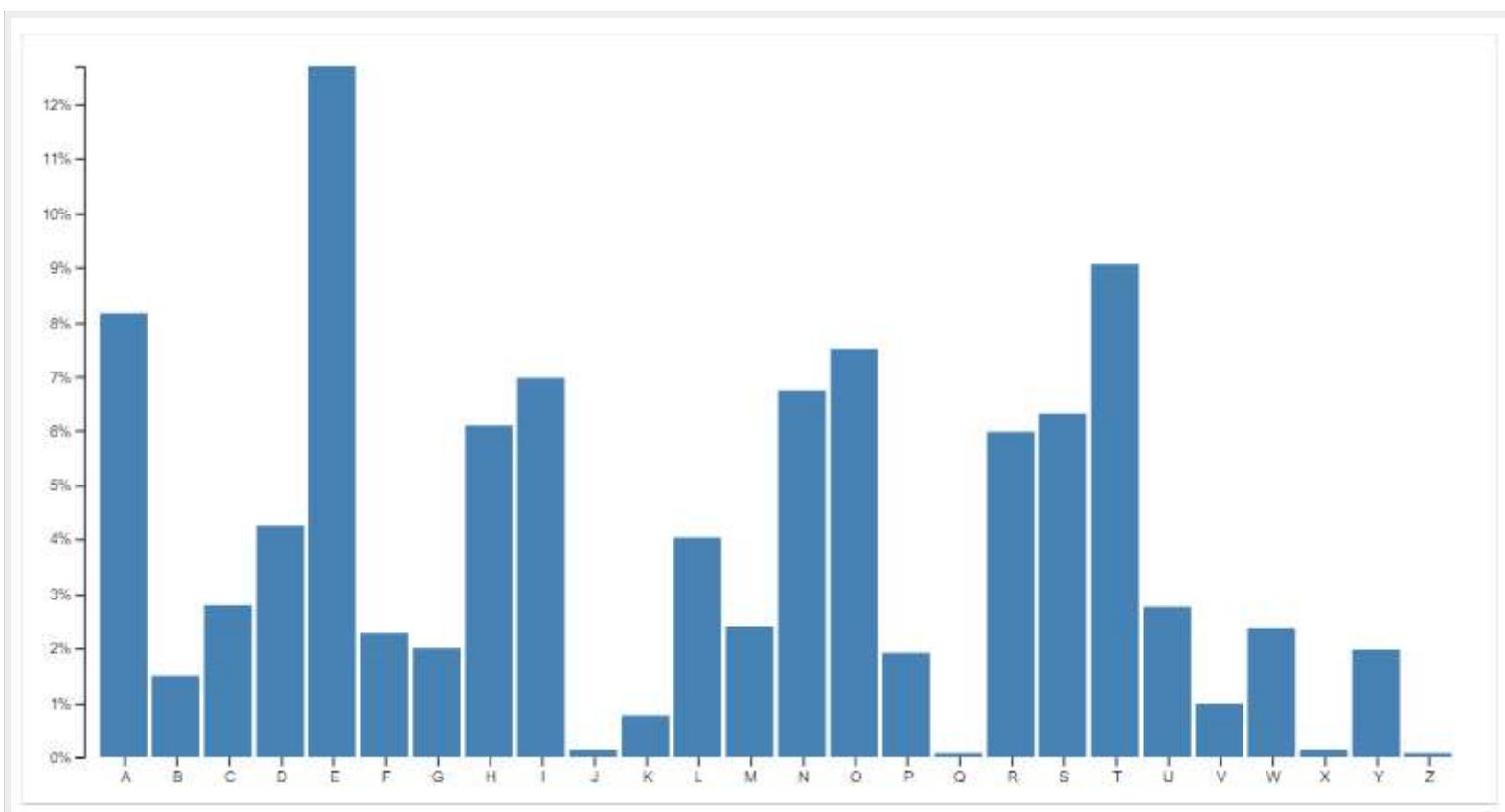
# D3 is more than a way to just change data on the page dynamically

- 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

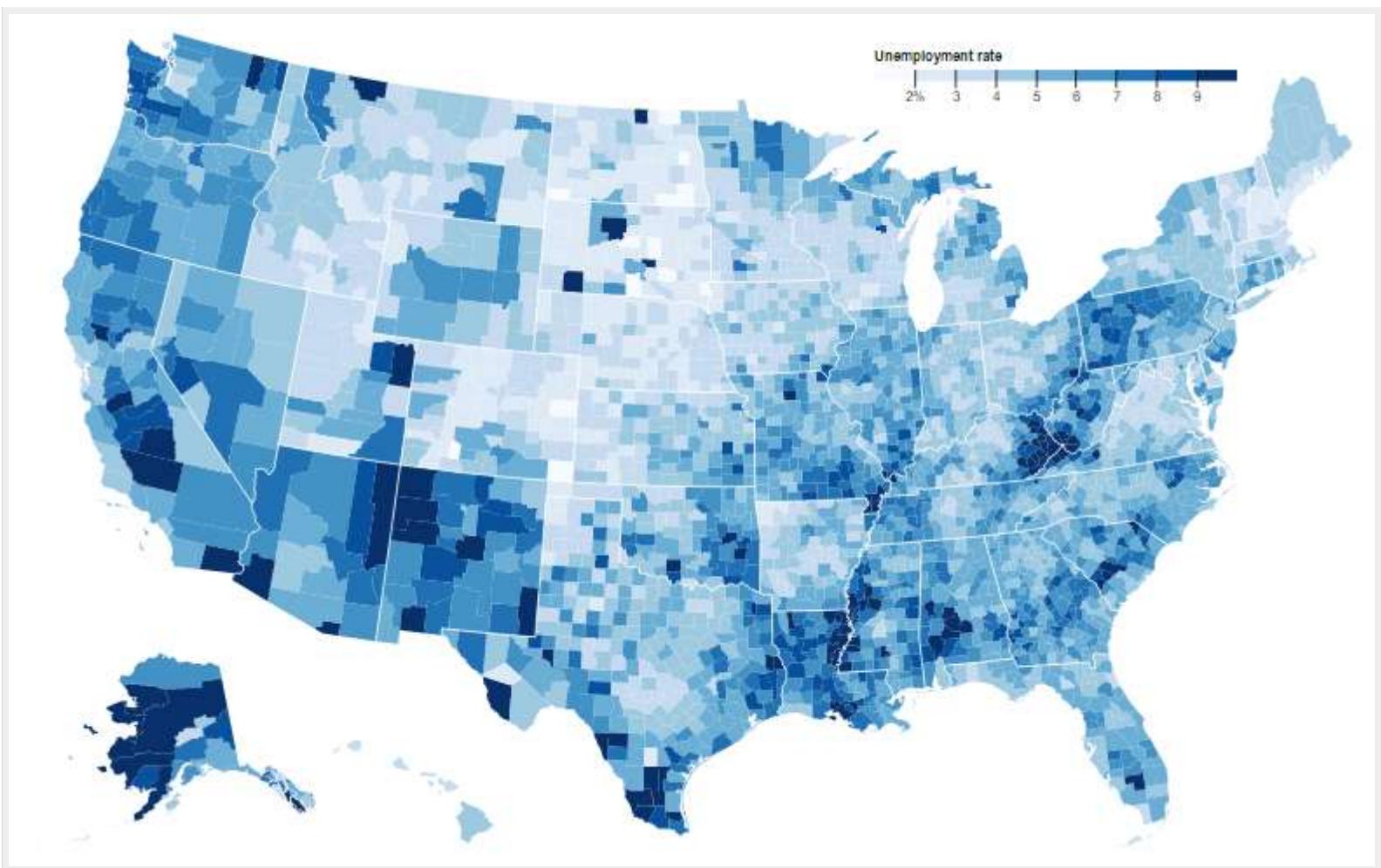
# D3 in the wild



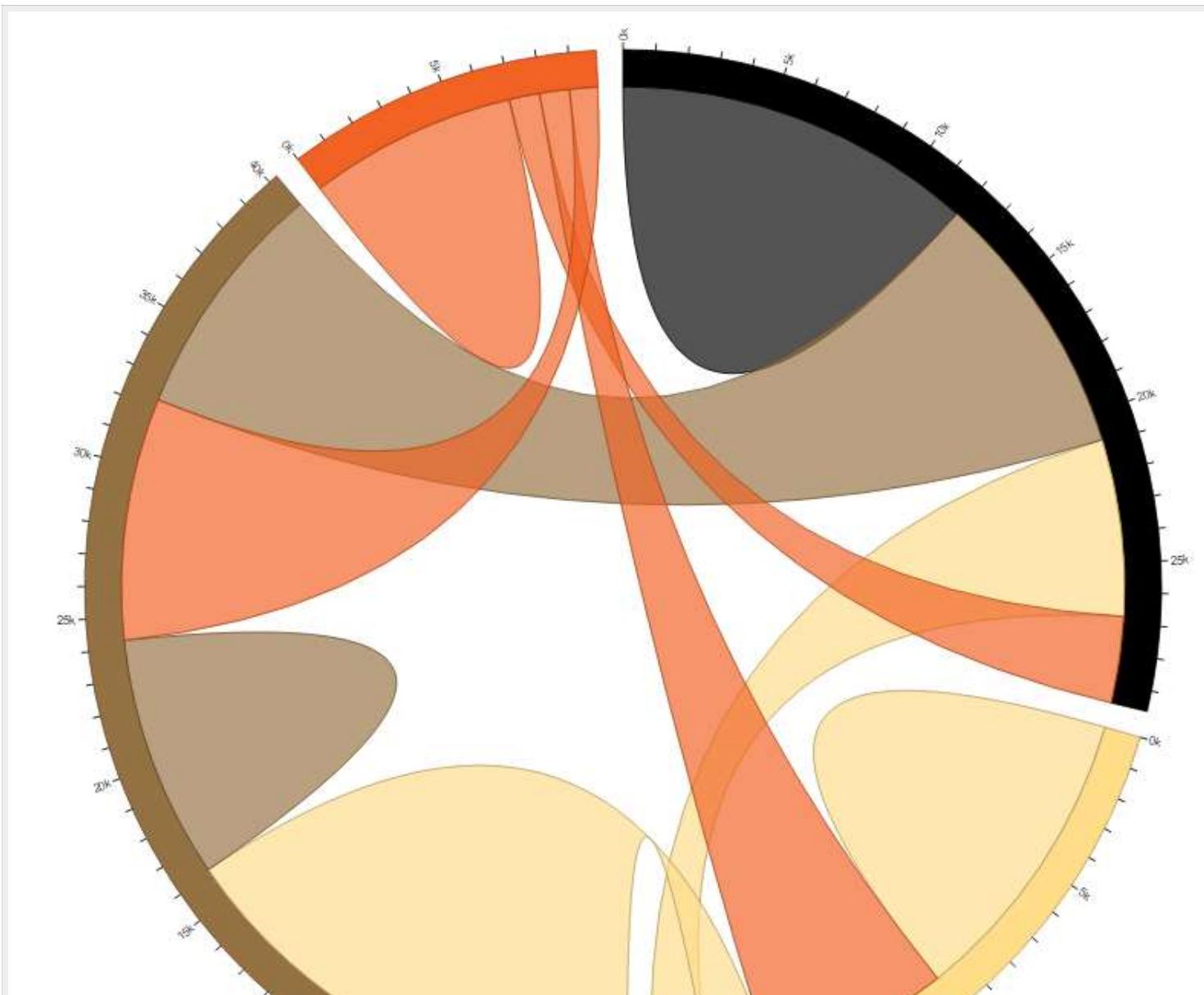
# The most basic (source)

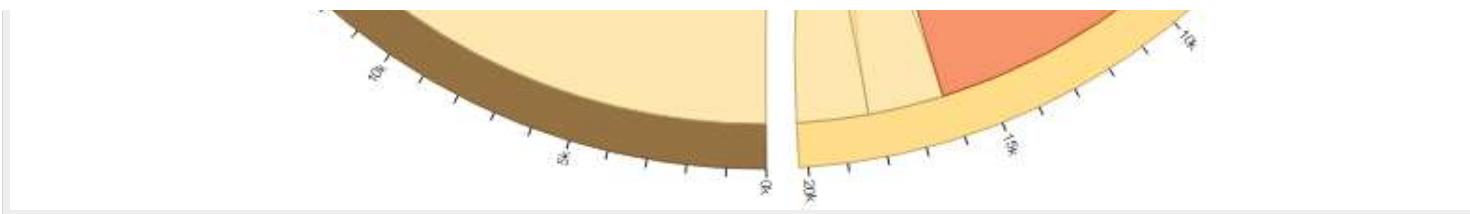


# Maps (source)

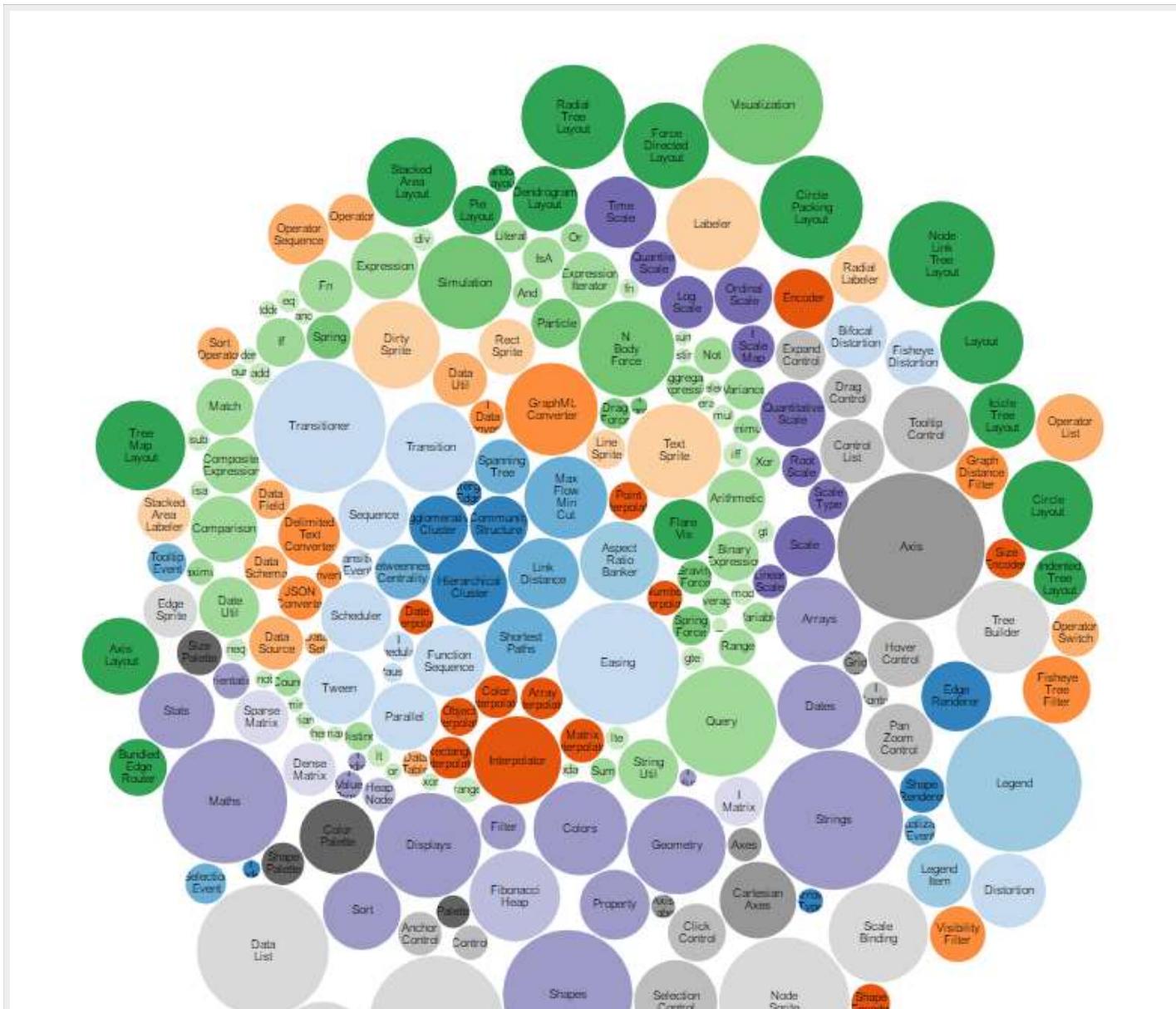


# Chords (source)



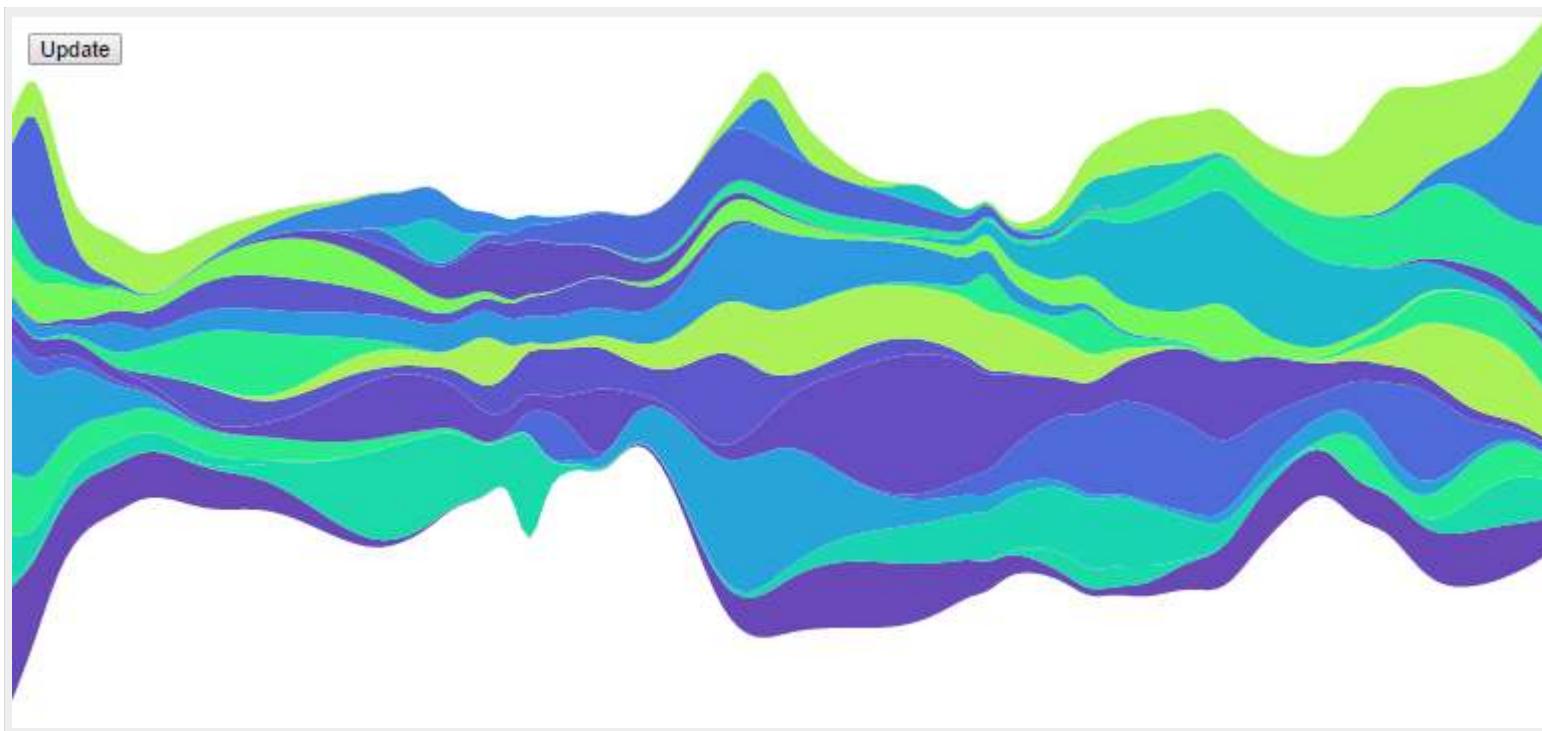


# Bubble (source)

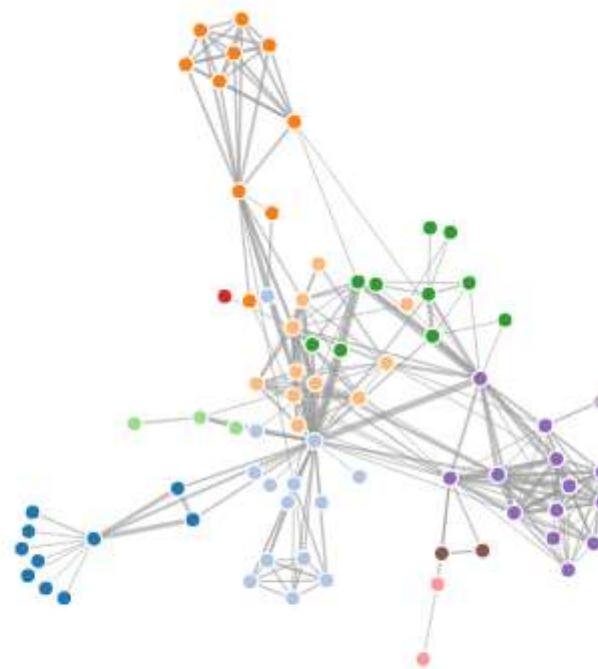




# Stream (source)



# 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

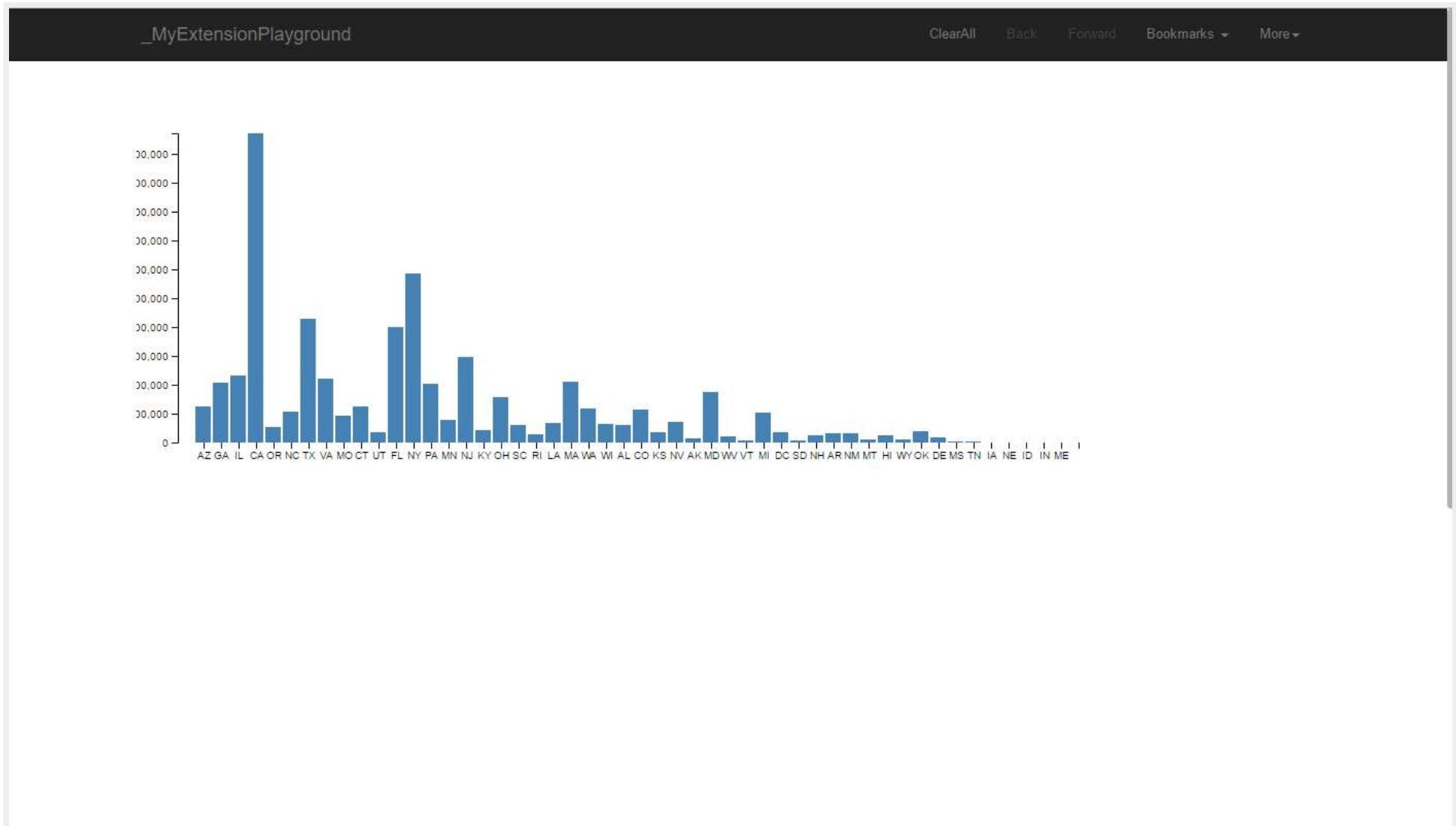


# Using D3 in Mashups

## or how to use the capabilities APIs with D3.js



# What are we building?



# Create a new mashup

The screenshot shows the Qlik Dev Hub interface at [localhost:4848/dev-hub/](http://localhost:4848/dev-hub/). The main navigation bar includes back, forward, and search icons, along with a star, refresh, and notifications icon. The title bar says "Qlik® Dev Hub". The top menu bar has tabs: All (highlighted), Mashups, Visualization extensions, and Widget libraries. A large "+" button is located in the top right corner of the main content area. The left sidebar, titled "Tools", contains five items: Single configurator, Extension editor, Mashup editor, Widget editor, and Engine API Explorer. The "Mashups" tab is selected, displaying a grid of 20 available mashups. Each item in the grid has a thumbnail icon and a name: Angular Template, AppSmash, BootstrapBaseDire..., BootstrapMashup..., enigma patterns, Helpdesk Angular, IMA-Consultant-O..., and JJSC Mashup.

# Create a new mashup

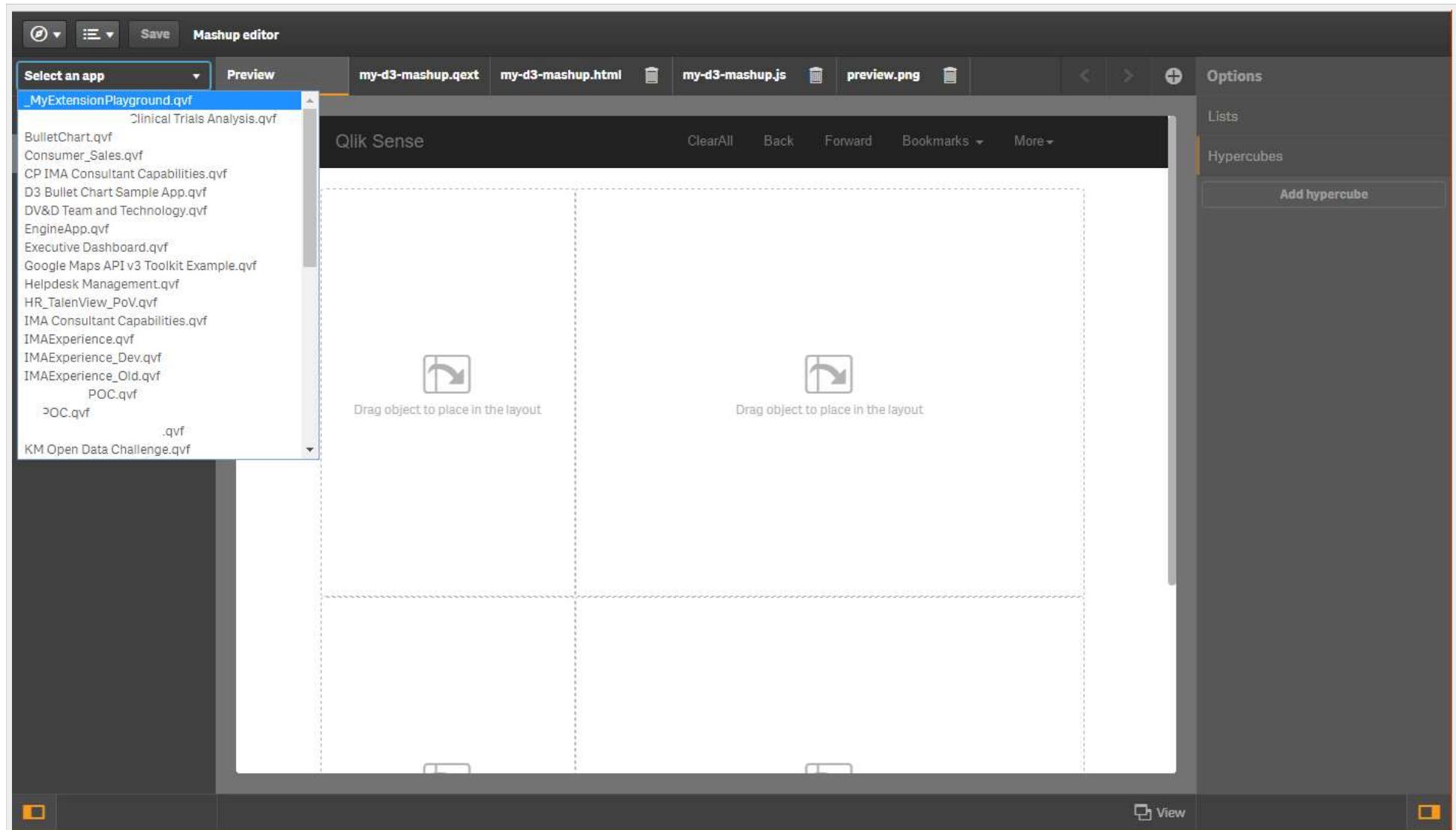
The screenshot shows the Qlik Dev Hub interface. The top navigation bar includes 'Qlik® Dev Hub', a menu icon, and a search icon. Below the navigation is a toolbar with tabs: 'Tools' (selected), 'All', 'Mashups' (highlighted in orange), 'Visualization extensions', and 'Widget libraries'. A '+' button is located in the top right corner of the toolbar.

The main area displays a grid of 20 existing mashups. One mashup, 'Angular Template', is highlighted with a light gray background. A modal dialog box titled 'Create new mashup' is overlaid on the grid. The dialog contains fields for 'Name' (containing 'my-d3-mashup') and 'Template' (set to 'Grid mashup template'). At the bottom of the dialog are 'Cancel' and 'Create & edit' buttons.

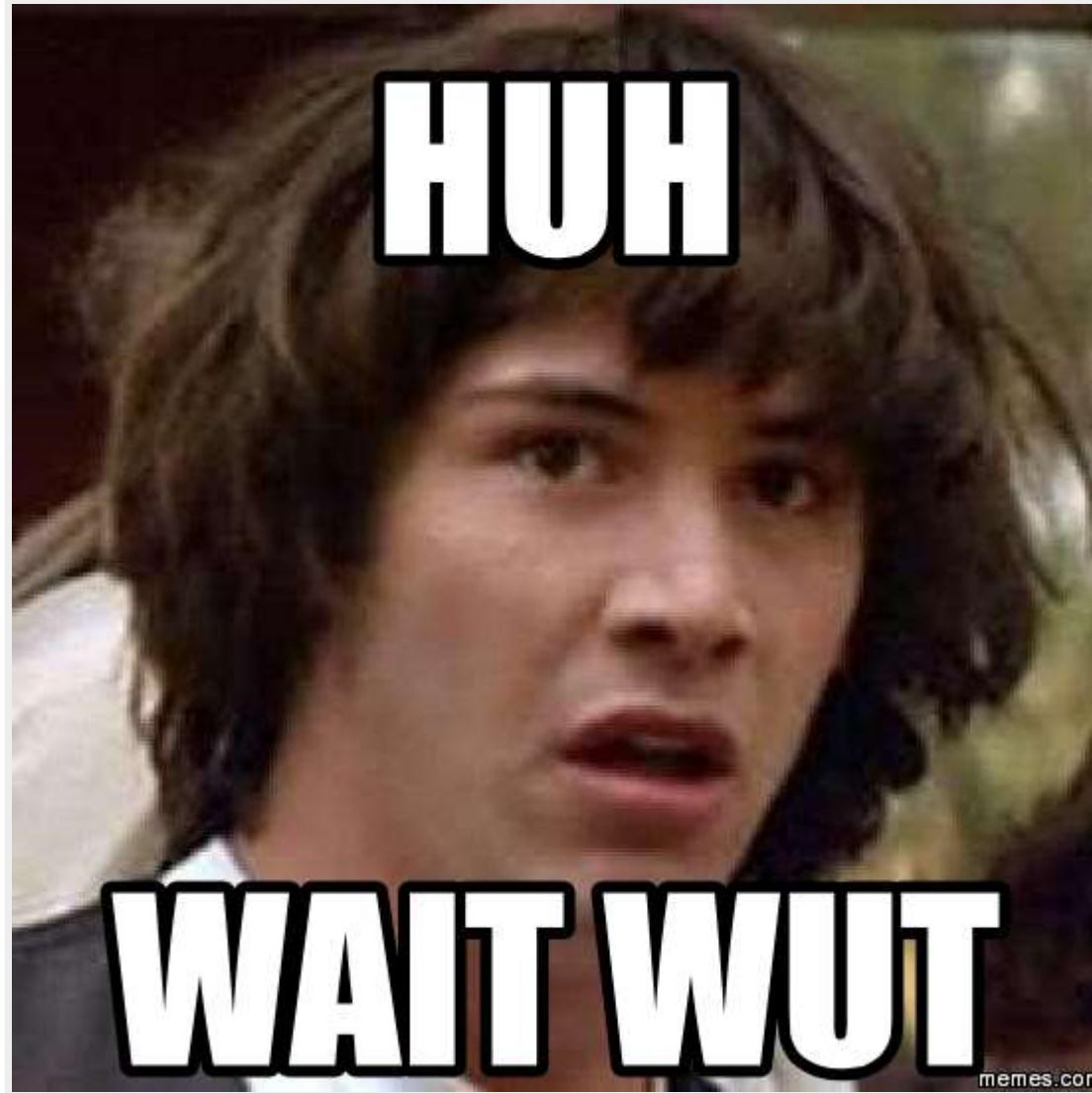
The left sidebar lists several tools:

- Single configurator**: Wizard for embedding single objects (visualizations or sheets) into web pages.
- Extension editor**: Create and edit visualization extensions.
- Mashup editor**: Create and edit web pages containing Qlik Sense objects.
- Widget editor**: Create and edit widgets and widget libraries for Qlik Sense.
- Engine API Explorer**: Explore the rich capabilities and objects of the Qlik Engine API.

# 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...

The screenshot shows the Qlik Sense Mashup editor interface. At the top, there's a toolbar with icons for refresh, save, and other functions. Below the toolbar, the title bar displays the file name '\_MyExtensionPlayground.qext'. The main area is divided into several sections: 'Sheets and objects' on the left containing a search bar and a list of sheets; a central workspace with a large dashed rectangle labeled 'Drag object to place in the layout'; and a right sidebar titled 'Options' with a 'Lists' section. In the 'Lists' section, 'Hypercubes' is listed, and below it is a button labeled 'Add hypercube'. This 'Add hypercube' button is highlighted with a blue rectangular selection.

# Configuring the hypercube

The screenshot shows the Qlik Sense Mashup editor interface. The top navigation bar includes 'Save' and 'Mashup editor' buttons. Below the navigation is a tab bar with 'Preview' selected, followed by 'my-d3-mashup.qext', 'my-d3-mashup.html', 'my-d3-mashup.js', and 'preview.png'. On the left, a sidebar titled 'Sheets and objects' lists various sheets like 'My new sheet', '[no title]', and 'Slider Sheet'. The main area displays a 'Qlik Sense' dashboard with a placeholder message 'Drag object to place in the area'. A central modal window is open, titled 'Add hypercube'. It has three radio buttons for 'Dimensions', 'Fields' (which is selected), and 'Measures'. Under 'Fields', 'addr\_state' is listed. Under 'Measures', 'Sum(annual\_inc)' is listed. The 'Mode' dropdown is set to 'Straight'. There are two unchecked checkboxes: 'Suppress zero' and 'Suppress missing'. The 'Rows' input field contains '1000'. The 'Callback function' input field contains 'getStateData'. At the bottom of the modal are 'Cancel' and 'Add' buttons.

# What did you do!?!?

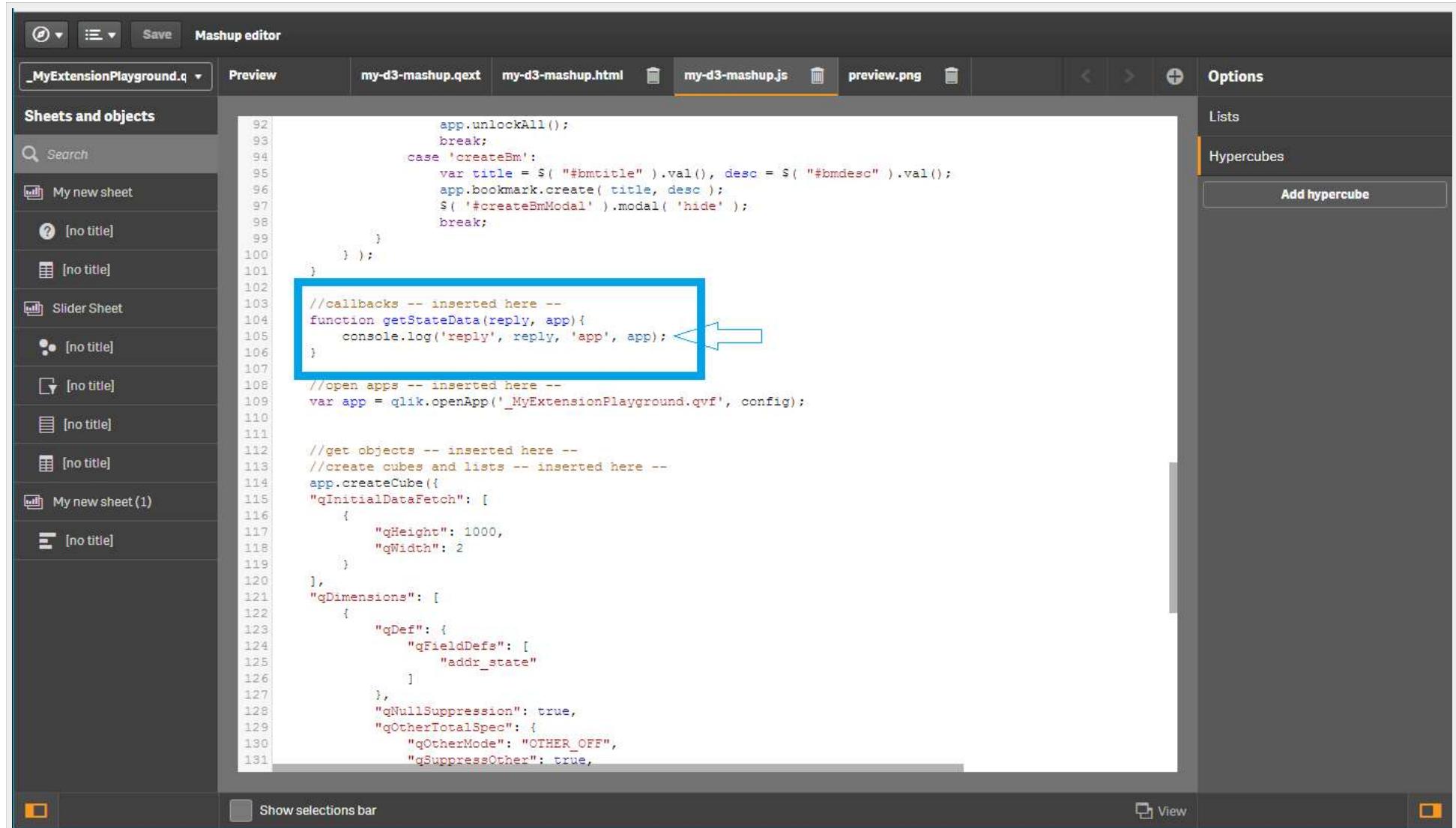


# What did all that do?

The screenshot shows the Qlik Sense Mashup editor interface. The title bar includes 'Save' and 'Mashup editor'. The left sidebar lists 'Sheets and objects' with items like 'MyExtensionPlayground.q', 'My new sheet', '[no title]', 'Slider Sheet', and several unnamed sheets. The main area displays a JSON configuration file named 'my-d3-mashup.js'. The code defines a cube with dimensions, measures, and various properties for data handling. On the right, there are tabs for 'Lists' and 'Hypercubes', with an 'Add hypercube' button. The bottom of the screen has standard window controls and a 'View' button.

```
//create cubes and lists -- inserted here --
app.createCube({
    "qInitialDataFetch": [
        {
            "qHeight": 1000,
            "qWidth": 2
        }
    ],
    "qDimensions": [
        {
            "qDef": {
                "qFieldDefs": [
                    "addr_state"
                ]
            },
            "qNullSuppression": true,
            "qOtherTotalSpec": {
                "qOtherMode": "OTHER_OFF",
                "qSuppressOther": 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,
                "qSortByLabel": 0
            }
        }
    ]
});
```

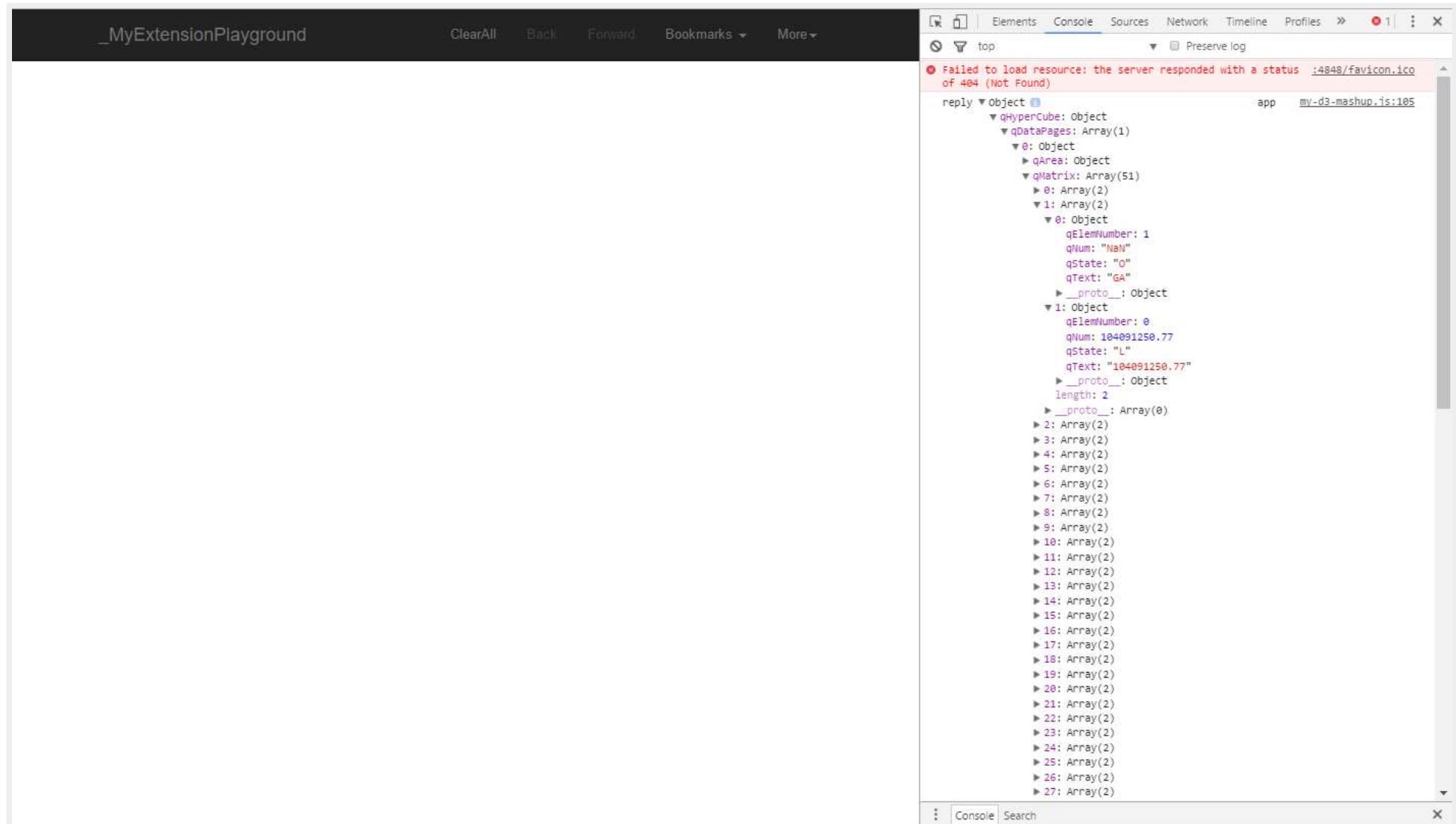
# Let's see the return values



The screenshot shows the Qlik Mashup editor interface. The title bar includes 'Save' and 'Mashup editor'. The tabs at the top are 'myExtensionPlayground.qvext', 'my-d3-mashup.html', 'my-d3-mashup.js' (which is selected), and 'preview.png'. The main area is a code editor with syntax highlighting for JavaScript. A blue box highlights a function call 'getStateData' with parameters 'reply', 'app', and 'app'. An arrow points from the word 'app' in the parameter list to the variable 'app' in the line above it, which is part of a 'for' loop. The code editor has a dark theme with light-colored code. On the left, there's a sidebar titled 'Sheets and objects' listing various sheets like 'My new sheet', 'Slider Sheet', and multiple '[no title]' entries. On the right, there are sections for 'Lists' and 'Hypercubes' with a button 'Add hypercube'.

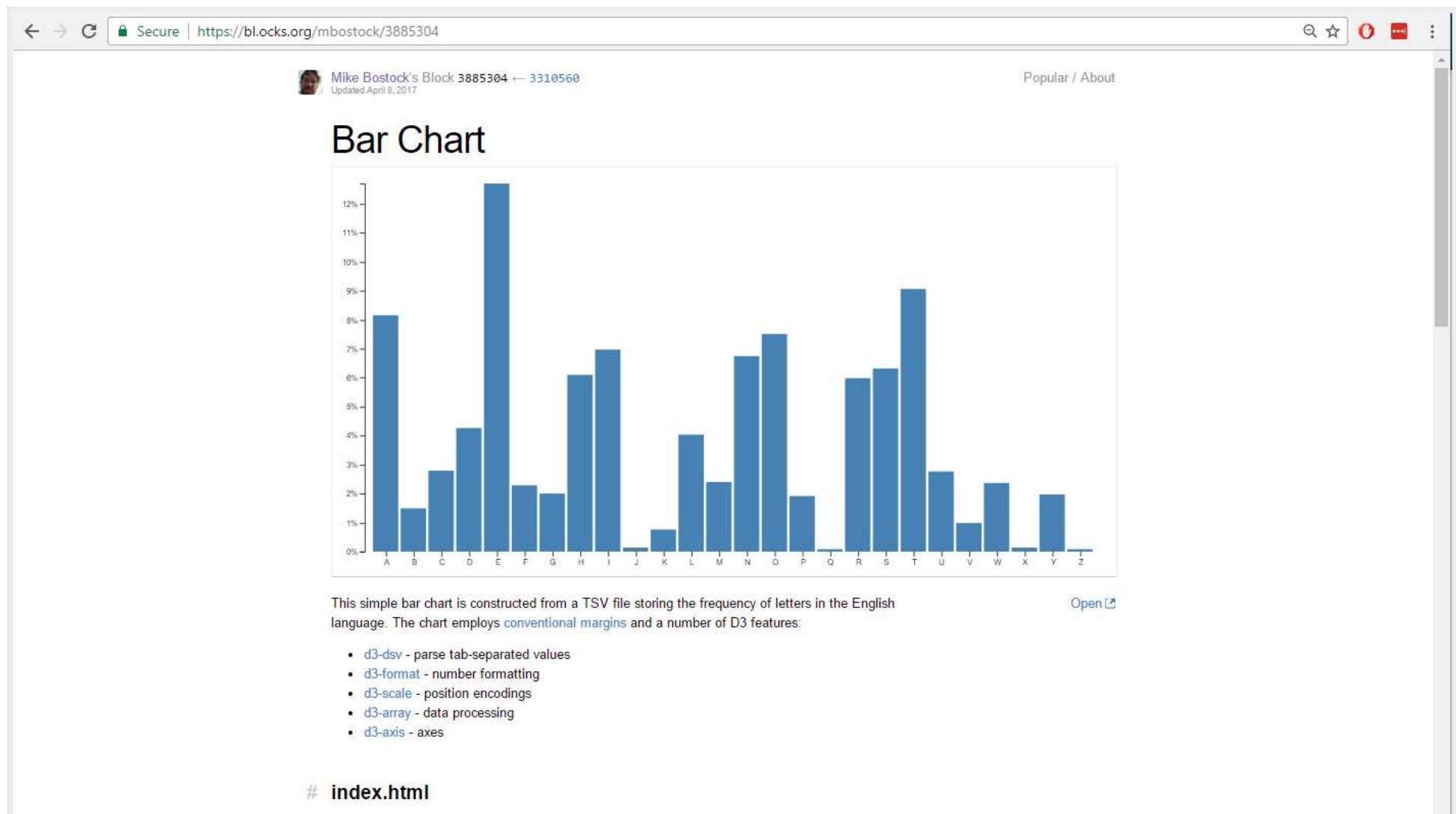
```
92         app.unlockAll();
93         break;
94     case 'createBm':
95         var title = $( "#bmtitle" ).val(), desc = $( "#bmdesc" ).val();
96         app.bookmark.create( title, desc );
97         $( '#createBmModal' ).modal( 'hide' );
98         break;
99     }
100 }
101
102 //callbacks -- inserted here --
103 function getStateData(reply, app){
104     console.log('reply', reply, 'app', app);
105 }
106
107 //open apps -- inserted here --
108 var app = qlik.openApp('_MyExtensionPlayground.qvf', config);
109
110
111 //get objects -- inserted here --
112 //create cubes and lists -- inserted here --
113 app.createCube({
114     "qInitialDataFetch": [
115         {
116             "qHeight": 1000,
117             "qWidth": 2
118         }
119     ],
120     "qDimensions": [
121         {
122             "qDef": {
123                 "qFieldDefs": [
124                     "addr_state"
125                 ]
126             },
127             "qNullSuppression": true,
128             "qOtherTotalSpec": {
129                 "qOtherMode": "OTHER_OFF",
130                 "qSuppressOther": true,
131             }
132         }
133     ]
134 }
```

# The developer panel

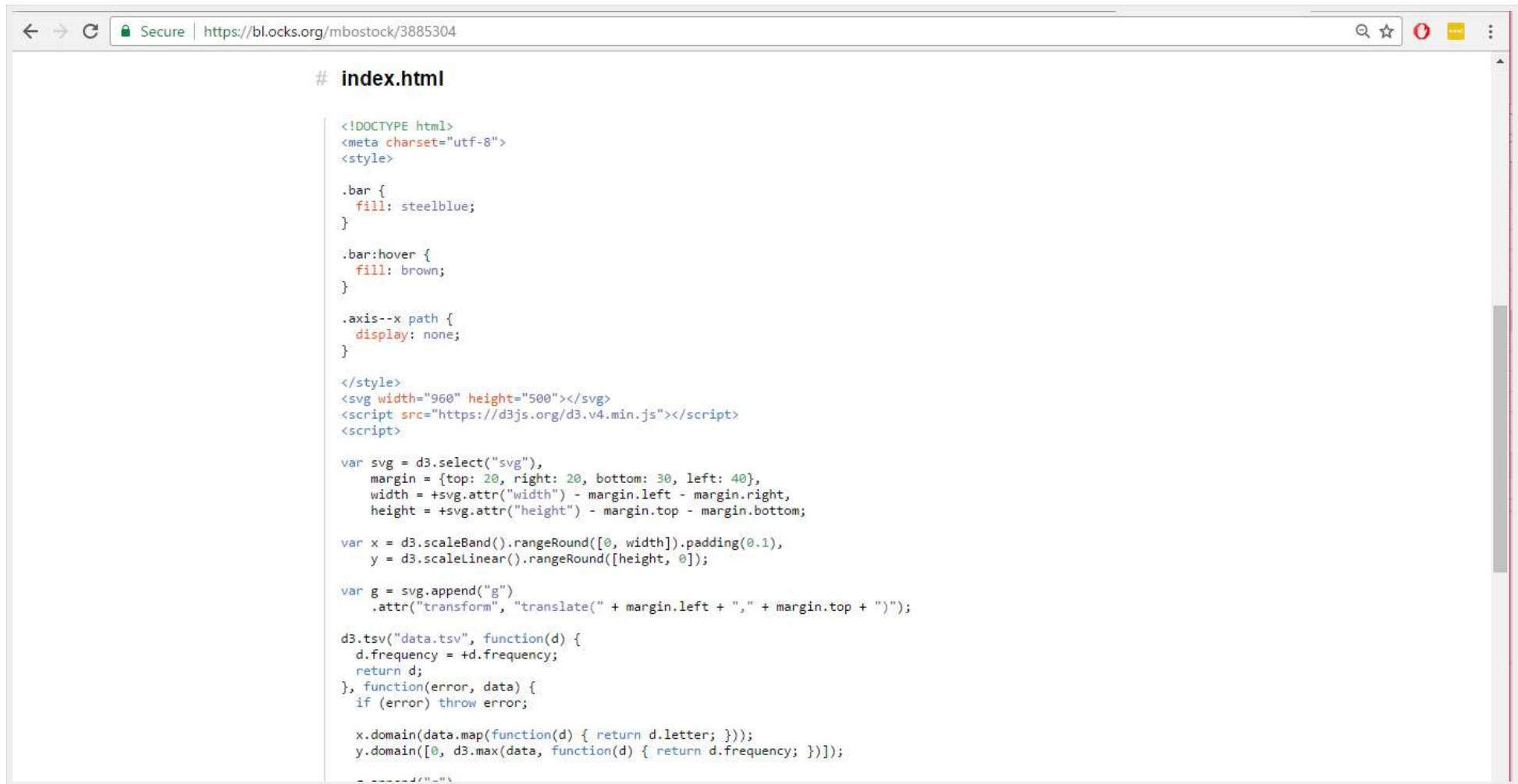




# Go grab an example ([source](#))



# Get HTML/CSS/JS from example



The screenshot shows a browser window with the URL <https://bl.ocks.org/mbostock/3885304>. The page title is "# index.html". The content displays the source code of a D3.js visualization. The code includes CSS styles for bars, an SVG element with dimensions, and a script block containing D3.js code to load data from "data.tsv" and create a bar chart.

```
<!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

Mashup editor

Sheets and objects

Search

My new sheet

[no title]

[no title]

Slider Sheet

[no title]

[no title]

[no title]

[no title]

[no title]

My new sheet(1)

[no title]

Preview my-d3-mashup.q... my-d3-mashup... my-d3-mashup... preview.p... styles.css Options

Lists

Hypercubes

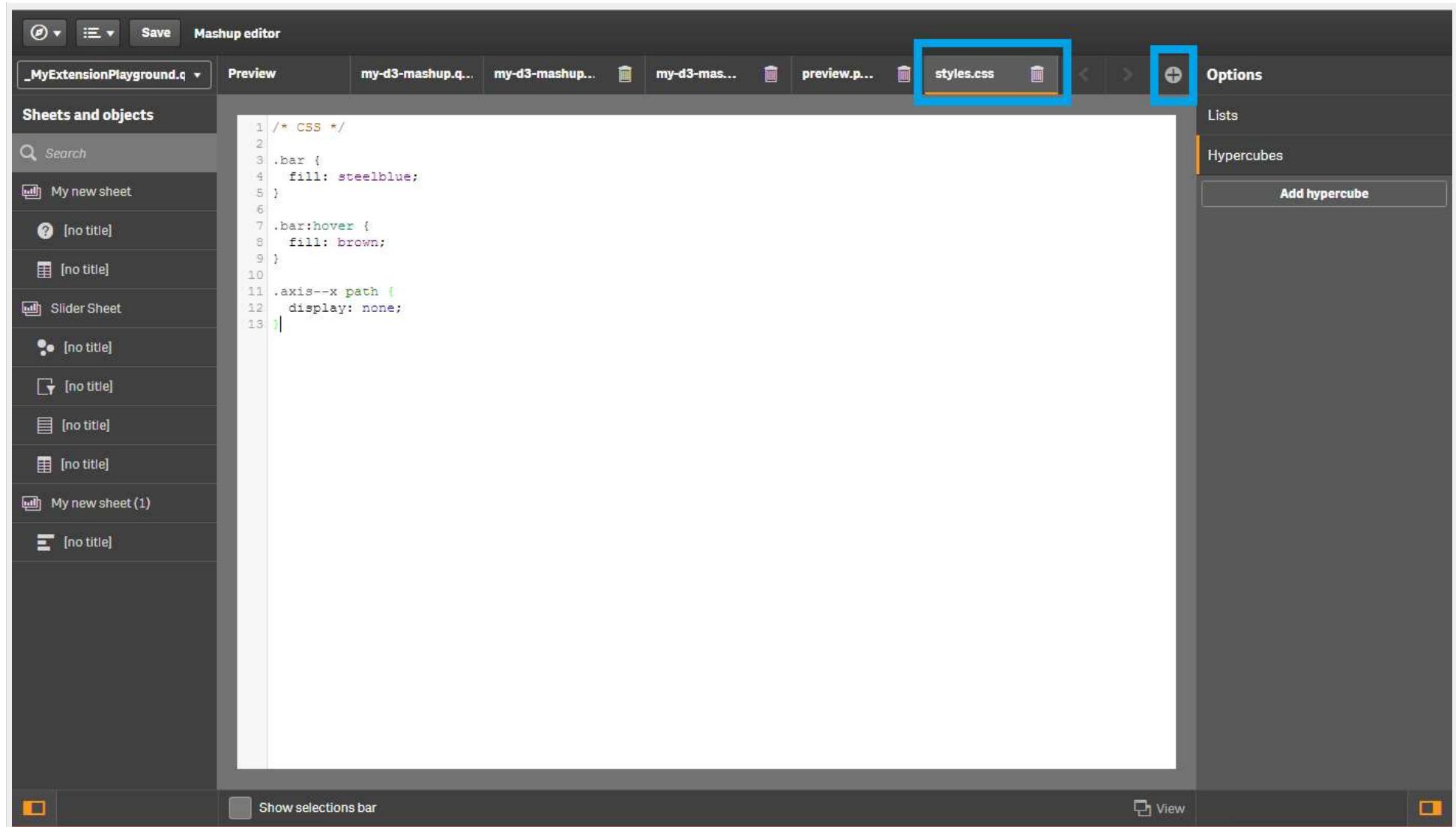
Add hypercube

```
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">×</span>
112     <span id="errmsg"></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">×

Show selections bar View


```

# Insert css into script



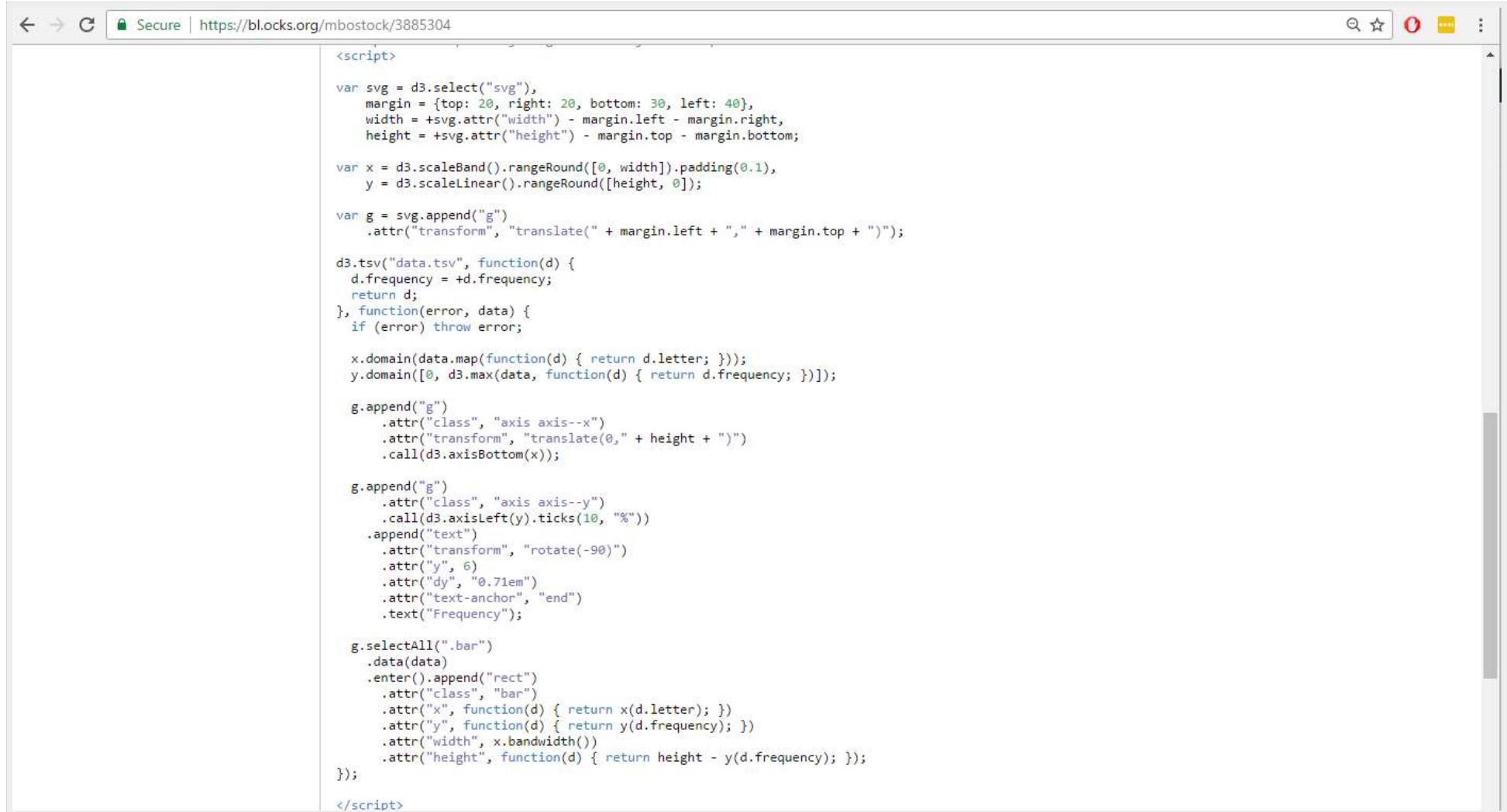
# Load D3 library in

The screenshot shows the Qlik Sense Mashup editor interface. The title bar reads "Mashup editor". The left sidebar lists various sheets and objects, including "My new sheet", "[no title]", "Slider Sheet", and "[no title]". The main area displays a JavaScript code snippet:

```
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( "/extensi  
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 AnnLi ( ann ) {
```

The code loads the Qlik Sense API and the D3.js library. It handles errors by displaying them in a modal. It also checks for bookmarks and adds a message if none are found. The body is set to scroll hidden.

# Get D3 from example

A screenshot of a web browser window displaying a block of JavaScript code. The URL in the address bar is https://bl.ocks.org/mbostock/3885304. The code is a D3.js script for creating a histogram. It includes imports for d3.js, a data.tsv file, and a style.css file. The script sets up an SVG container with margins and scales for both axes. It reads data from 'data.tsv' and maps letters to the x-axis and frequency to the y-axis. The visualization features a bottom axis with ticks and a text label 'Frequency', and bars representing the data.

```
<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. At the top, there's a toolbar with icons for file operations like 'Save' and tabs for different files: '\_MyExtensionPlayground.q...', 'my-d3-mashup.q...', 'my-d3-mashup...', 'my-d3-mashup...', 'preview.p...', 'styles.css', and 'Options'. The 'Options' tab is currently selected.

The main area is titled 'Sheets and objects' and contains a 'Search' bar and a list of sheets: 'My new sheet', '[no title]', '[no title]', 'Slider Sheet', '[no title]', '[no title]', '[no title]', '[no title]', '[no title]', 'My new sheet (1)', and '[no title]'. The 'My new sheet (1)' sheet is currently active.

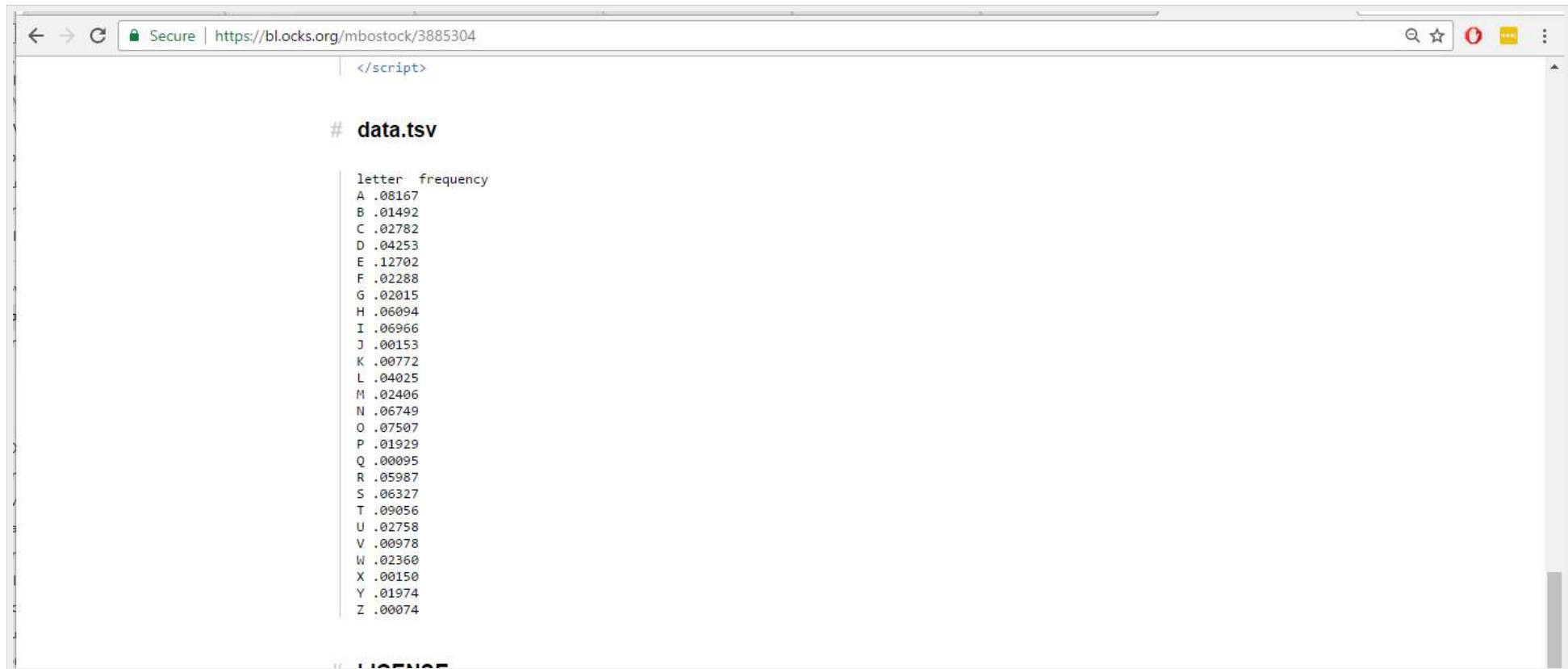
The code editor displays the following D3.js code:

```
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)
```

On the right side of the interface, there are sections for 'Lists' and 'Hypercubes'. The 'Hypercubes' section has a button labeled 'Add hypercube'.

# Before we connect to hypercube...

## let's get the sample data



A screenshot of a web browser window showing a terminal-like interface. The address bar indicates a secure connection to <https://bl.ocks.org/mbostock/3885304>. The content area displays a list of data points from a file named `data.tsv`. The data consists of two columns: `letter` and `frequency`, separated by a tab character.

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

The screenshot shows a Windows file explorer window with the following details:

**File Explorer Title Bar:** C:\Users\kevin.mcgovern\Documents\Qlik\Sense\Extensions\my-d3-mashup\data....

**File Menu:** File Edit Selection Find View Goto Tools Project Preferences Help

**Folders:** my-d3-mashup

**File List:** data.tsv, my-d3-mashup.html, my-d3-mashup.js, my-d3-mashup.qext, preview.png, styles.css, wbfolder.wbl

**Content Preview of data.tsv:**

	letter	frequency
1	A	.08167
2	B	.01492
3	C	.02782
4	D	.04253
5	E	.12702
6	F	.02288
7	G	.02015
8	H	.06094
9	I	.06966
10	J	.00153
11	K	.00772
12	L	.04025
13	M	.02406
14	N	.06749
15		

**Address Bar:** This PC > Documents > Qlik > Sense > Extensions > my-d3-mashup

**File List Table:**

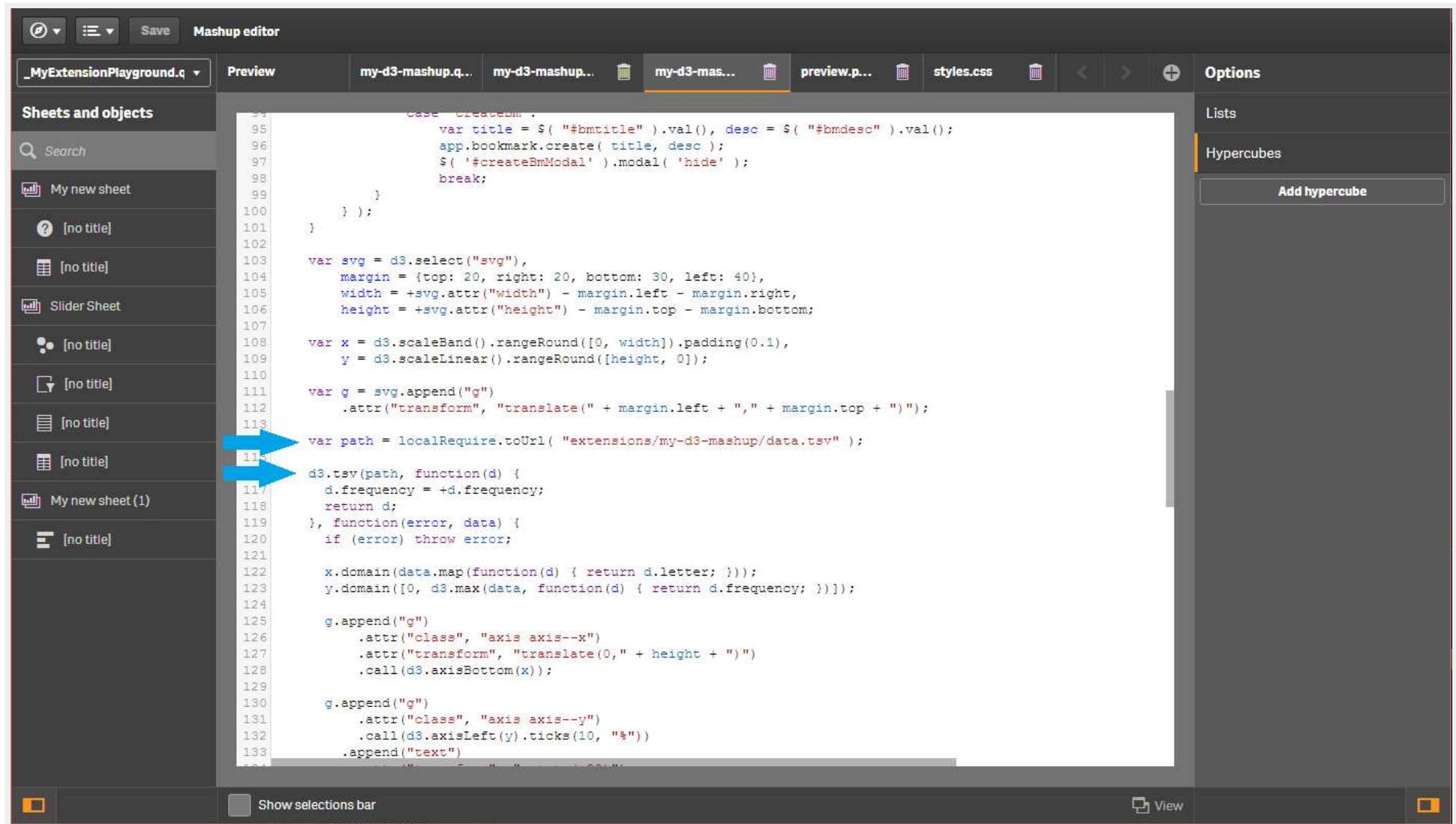
Name	Date modified	Type	Size
data.tsv	4/20/2017 11:23 AM	TSV File	1 KB
my-d3-mashup.html	4/19/2017 3:38 PM	HTML File	5 KB
my-d3-mashup.js	4/19/2017 3:35 PM	JS File	6 KB
my-d3-mashup.qext	4/18/2017 8:24 AM	QEXT File	1 KB
preview.png	4/18/2017 8:24 AM	PNG File	14 KB
styles.css	4/19/2017 3:35 PM	CSS File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

# Load an instance of require.js

The screenshot shows a Qlik Sense Mashup editor interface. The title bar includes 'Save' and 'Mashup editor'. The left sidebar lists 'Sheets and objects' with items like 'MyExtensionPlayground.q', 'My new sheet', '[no title]', 'Slider Sheet', and 'My new sheet(1)'. The main area displays a code editor with a Require.js configuration script. Two blue arrows point upwards from the bottom of the code editor towards the top of the window, likely indicating scrollable content.

```
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 ( $( '#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 );
53       } );
54     } );
55   }
56 } );
```

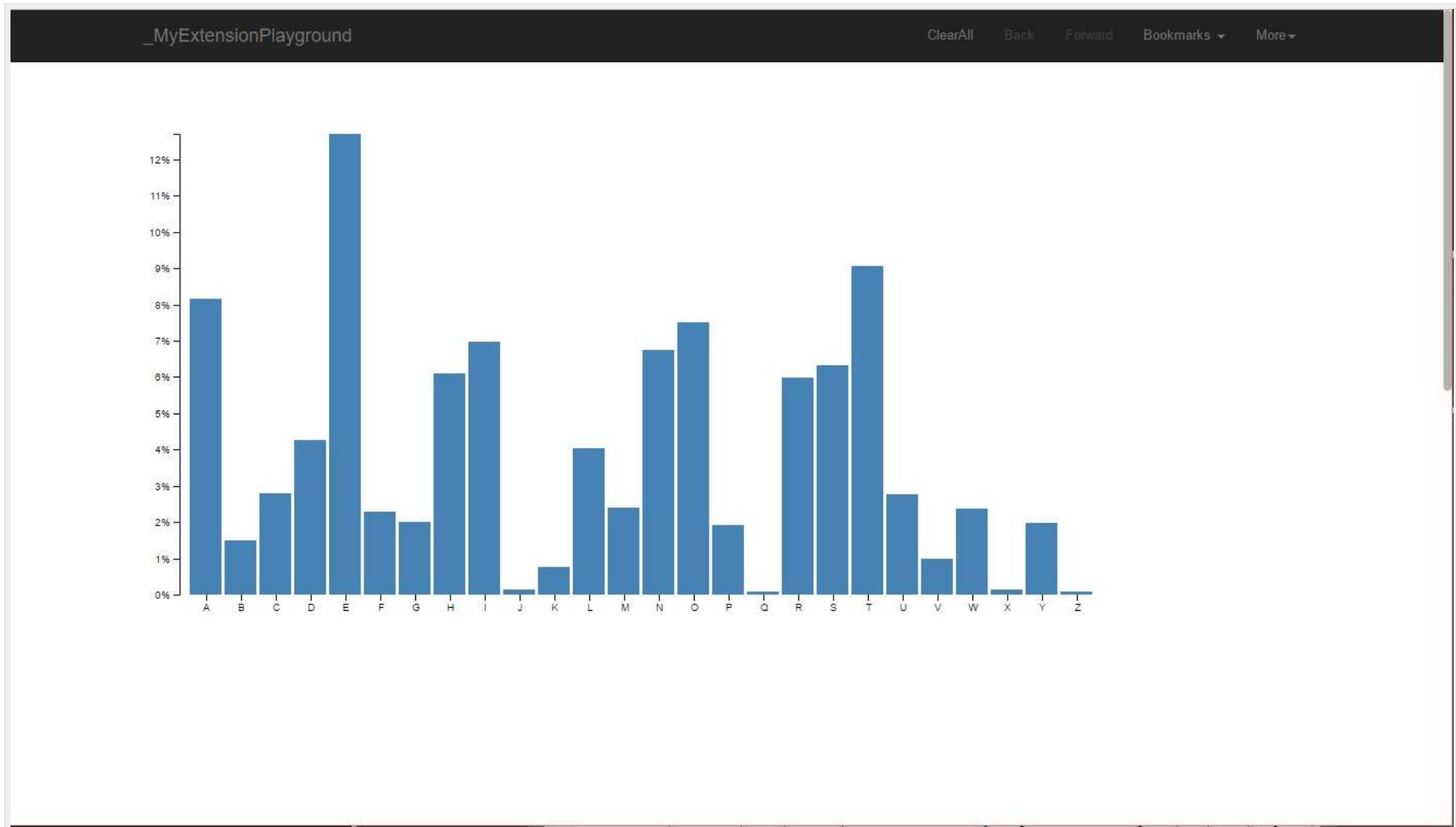
# Reference the tsv file in js



The screenshot shows the Google Sheets Mashup editor interface. The left sidebar lists various sheets and objects, including "My new sheet" and "Slider Sheet". The main area displays a portion of a JavaScript file with line numbers 95 through 133. Two blue arrows point to line 114, which contains the code `var path = localRequire.toUrl('extensions/my-d3-mashup/data.tsv');`. The right sidebar shows sections for "Lists" and "Hypercubes", with a button labeled "Add hypercube".

```
95
96
97
98
99
100
101
102
103     case "createBm":
104         var title = $( "#bmtitle" ).val(), desc = $( "#bmdesc" ).val();
105         app.bookmark.create( title, desc );
106         $( '#createBmModal' ).modal( 'hide' );
107         break;
108     }
109 }
110
111 var svg = d3.select("svg"),
112     margin = {top: 20, right: 20, bottom: 30, left: 40},
113     width = +svg.attr("width") - margin.left - margin.right,
114     height = +svg.attr("height") - margin.top - margin.bottom;
115
116 var x = d3.scaleBand().rangeRound([0, width]).padding(0.1),
117     y = d3.scaleLinear().rangeRound([height, 0]);
118
119 var g = svg.append("g")
120     .attr("transform", "translate(" + margin.left + "," + margin.top + ")");
121
122 var path = localRequire.toUrl('extensions/my-d3-mashup/data.tsv');
123
124 d3.tsv(path, function(d) {
125     d.frequency = +d.frequency;
126     return d;
127 }, function(error, data) {
128     if (error) throw error;
129
130     x.domain(data.map(function(d) { return d.letter; }));
131     y.domain([0, d3.max(data, function(d) { return d.frequency; })]);
132
133     g.append("g")
134         .attr("class", "axis axis--x")
135         .attr("transform", "translate(0," + height + ")")
136         .call(d3.axisBottom(x));
137
138     g.append("g")
139         .attr("class", "axis axis--y")
140         .call(d3.axisLeft(y).ticks(10, "%"))
141         .append("text")
142             .attr("transform", "rotate(-90)");
```

# Eureka!





**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 page includes navigation links for Code, Pull requests (0), Projects (0), Wiki, Pulse, Graphs, and Settings. Below the navigation, there's a summary of repository statistics: 28 commits, 1 branch, 0 releases, and 4 contributors. A 'Create new file', 'Upload files', 'Find file', and 'Clone or download' button are also present. The main content area displays a commit history starting with an initial commit by 'mcgovey' on Feb 7, 2017.

This branch is 5 commits ahead, 1 commit behind brianwmunz:master.

mcgovey committed on GitHub updated readme for createJSONObj changes

.gitattributes Initial Commit 3 years ago

Latest commit 7e8bdæ on Feb 7

# Minor edits

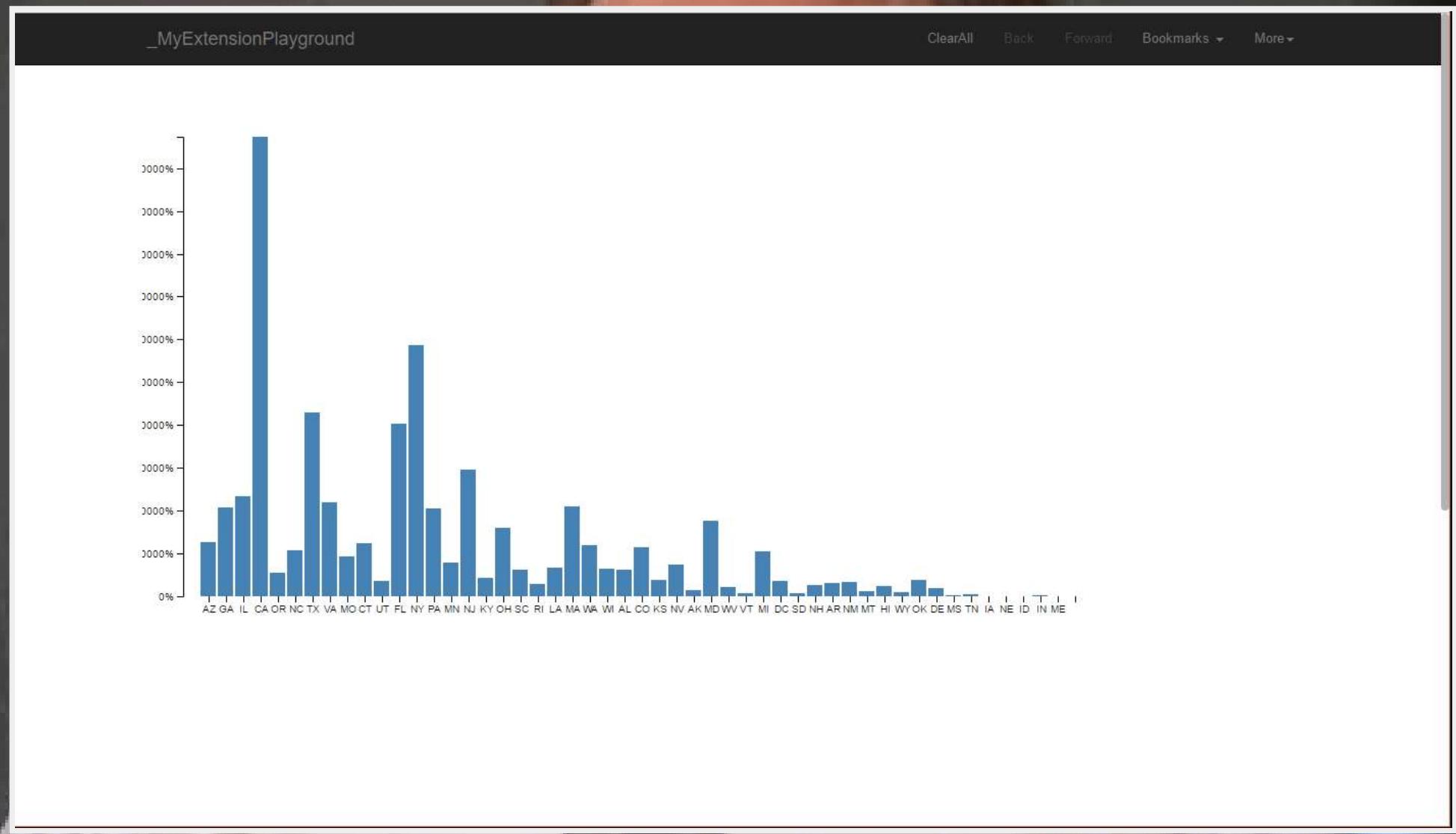
- Put the D3 code inside the callback function
- Clear the SVG before writing to it
- Comment out the call to the tsv
- Point the chart data to the hypercube variable

**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....





**THAT'S PERFECT**

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?



# 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\*\*\*
- Bl.ocks
- D3 Documentation
- Sample Slideshare Presentation

# Where can we go from here?

These slides at:

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

Supporting repo:

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