<https://github.com/malecki/qmssviz/tree/hw4/hw4>

## HW4: make a plot in your browser

As usual, this is an open ended assignment: make a plot of some data. You get to pick the data, decide what you want to do with it, and then fight with d3 and svg to make it show up in your browser.

This directory contains a template that will work. You may copy the contents of these files into a new plunk, or create a gist that will live on blocks.

Again following the pattern of other homeworks, there are several parts to the assignment. The first NNN are the minimum needed for the assignment to be accepted. Adding additional complexity will earn additional credit,

#### Data sources

You may choose data of your own from your project or from homework 3. Format it for consumption by javascript, either as json or csv, with names you can explain and parameterize.

### Requirements

1. Describe the graph(s) you want to make. When pseudocode helps for succinctness and clarity, use it. You may also include or link to a bitmap prototype generated in R, or even svg generated by gridsvg.
2. Describe your data shape (a typical element that will be bound to some element's \_\_data\_\_ property).
3. Create data, as json or csv, that meets the required data contract.
4. Load your data in js using the convenience d3.csv(file, error, data) or d3.json(file, error, data) methods and use the data callback to indicate success loading the data, either in the console or on the page.
5. Write an ordered list of comments in js describing the steps you will take to map your data to svg elements. Commit this.
6. Write code to perform each of the steps you described in comments. Your graph should render on the page. Commit this.

This is enough to complete the assignment. You should be able to do this in a few hours.

1. Iterate: add comments, then code, to enrich your graph. Add aesthetics, tweak sizes or color scales, add components like grid lines, axes, or annotations.
2. Interact: add comments, then code, using the event listeners for events such as mouseover, mouseout, click, mousedown, mouseup, and mousemove. Write empty callback functions to be executed when these events are raised. Commit this.
3. Fill in one event callback.
4. Fill in any other event callbacks you described in comments.
5. Add a component to choose subsets of your data