NEW YORK UNIVERSITY

D3 Lab

Bowen Yu

April 11, 16

Big Data Analysis

Work in Groups

- Please team up in groups of 3.
- You need to have npm installed on your machine. Check if you already have npm

```
npm --version
```

- If not, an easy way to install it is to use the Node Version Manager (NVM)

```
curl -o- https://raw.githubusercontent.com/creationix/nvm/v0.31.0/install.sh | bash
```

- After you have nvm, you may simply install npm along with nodejs:

```
nvm install v4.0
```

NPM Installation FAQ

- After you install nvm with the curl command, if you get nvm not found, please run the following and then try again.

```
source ~/.bashrc
```

Getting the Packages

- Now you need to install bower.

```
npm install -g bower
```

- Create new repository on your GitHub. Clone it to local disk. Then initialize this repo with bower. Under the cloned folder, run:

```
bower init
```

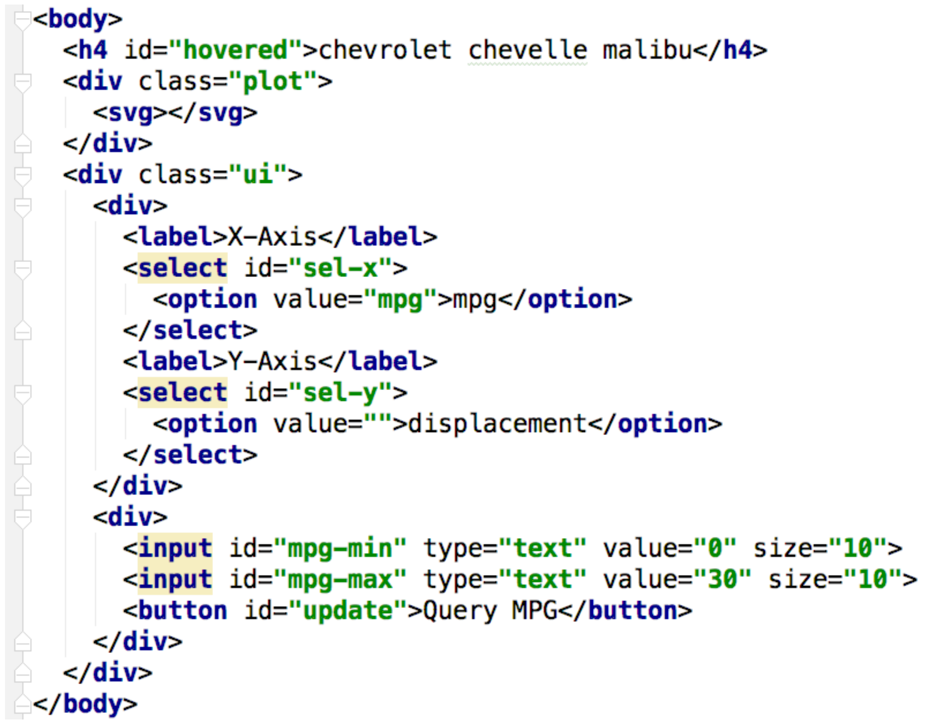
- Then install jQuery and D3 with bower

```
bower install --save jQuery d3
```

- Commit your bower.json to Git!

Task 1 – Create a Webpage

- Follow the slides and create a webpage that properly includes the jQuery and D3 libraries.
- Let index.html body contain the following content (a template has been provided for you)



```
<body>
  <h4 id="hovered">chevrolet chevelle malibu</h4>
  <div class="plot">
    <svg></svg>
  </div>
  <div class="ui">
    <div>
      <label>X-Axis</label>
      <select id="sel-x">
        <option value="mpg">mpg</option>
      </select>
      <label>Y-Axis</label>
      <select id="sel-y">
        <option value="">displacement</option>
      </select>
    </div>
    <div>
      <input id="mpg-min" type="text" value="0" size="10">
      <input id="mpg-max" type="text" value="30" size="10">
      <button id="update">Query MPG</button>
    </div>
  </div>
</body>
```

NEW YORK UNIVERSITY

Task 2– Play with JS in the Console

- Now play with JS in your browser console
  - If you included jQuery properly, you should be able to access directly the jQuery namespace “$”.
- Try the following:
  - Define variables and functions in the console.
  - Do arithmetic on variables and functions.
  - Define arrays and objects in the console.
  - Add/remove DOMs using jQuery. For example, you can set the text of the ui div by

```
$(‘.ui’).text(‘hello!’);
```

Task 2– JS Challenge

- Answer the **JS Challenge**:

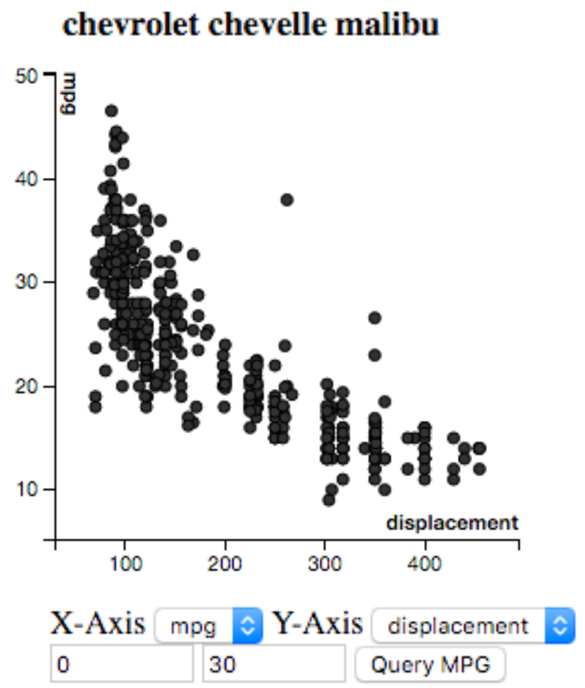
  How to get numerical value **123** in JS in one command with no more than 50 characters, if you can only use 4 types of characters: `[ ] ! +`

- That is,
  - You are writing one line of string containing only `[ ] ! +`
  - After copy-and-paste your line into the JS console and press enter, the result shall be exactly the numerical value 123.
    - Note that the value must be numerical. Therefore the string “123” is not a correct output.

NEW YORK UNIVERSITY

Task 3 – Interactive Scatterplot

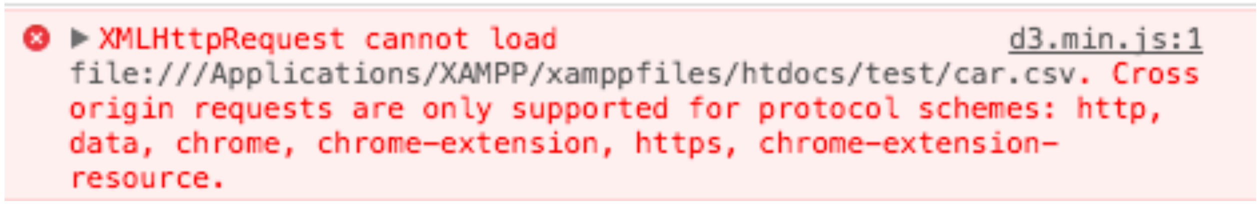
- Download the data from https://s3.amazonaws.com/yubowenok/car.csv
- Save the data at the root of your repo.
  - The data contains 392 cars with their info on 9 dimensions.
  - Do not commit the data to GitHub!
- Render an interactive scatterplot of the data:
  - Dimensions for x-axis and y-axis can be selected.
  - Filtering on MPG is supported.
  - Hovered car name is shown as the h4 header.



NEW YORK UNIVERSITY

Cross Origin Policy

- Your index.html and other source files shall be hosted on a web server. If you open the index.html directly, you may not be able to retrieve the CSV file at the same folder.



```
XMLHttpRequest cannot load                                    d3.min.js:1
file:///Applications/XAMPP/xamppfiles/htdocs/test/car.csv. Cross
origin requests are only supported for protocol schemes: http,
data, chrome, chrome-extension, https, chrome-extension-
resource.
```

- You need to start a server in this case in the index.html directory:

```
python -m SimpleHTTPServer
```

- Then access your webpage by

```
http://localhost:8000/
```

Task 3 – Detailed Requirements

- The final output shall look similar to the snapshot from the previous slide. Axes shall be properly labeled with ticks, and dimension names.
- Your must use JS to read the data from car.csv located on the server. You shall not hard code the data.
- Complete the dropdown lists in the UI div for selecting the data dimensions to be plotted as x-axis and y-axis.
  - Note that you do not need to include the categorical dimensions 'name' and 'origin'
  - You shall not hard code the dimensions.
- After the "Query MPG" button is pressed, the visualization is updated to render only those cars with MPG in the range defined by the mpg-min, mpg-max input box.
- D3 rendering must use the enter/exit/update data mapping. You shall not clear the entire svg every time before rendering.
- When the mouse hovers a scatterplot point, the <h4> header is updated to show the hovered car name.
  - You do not need to worry about point overlapping (some points cannot be hovered)

D3 Lab - Deliverable

- Submit to NYU Classes by **Monday, Apr 18** the following:
  - Inline text submission (write in the text box):
    - Your team members' netIds (including yourself).
    - The answer to the JS challenge.
    - The link to your GitHub for the interactive scatterplot.
  - Attachment: a zip file containing your source codes for the interactive scatterplot, containing the following:
    - index.html
    - bower.json
    - js or css files written by you

Do not include bower_components, or car.csv in the zip!

- After we downloading your zip, unzipping it and running bower install, the index.html shall properly present the interactive scatterplot in a browser.