Faculty of Science, Engineering and Technology

COS30045 Data Visualisation

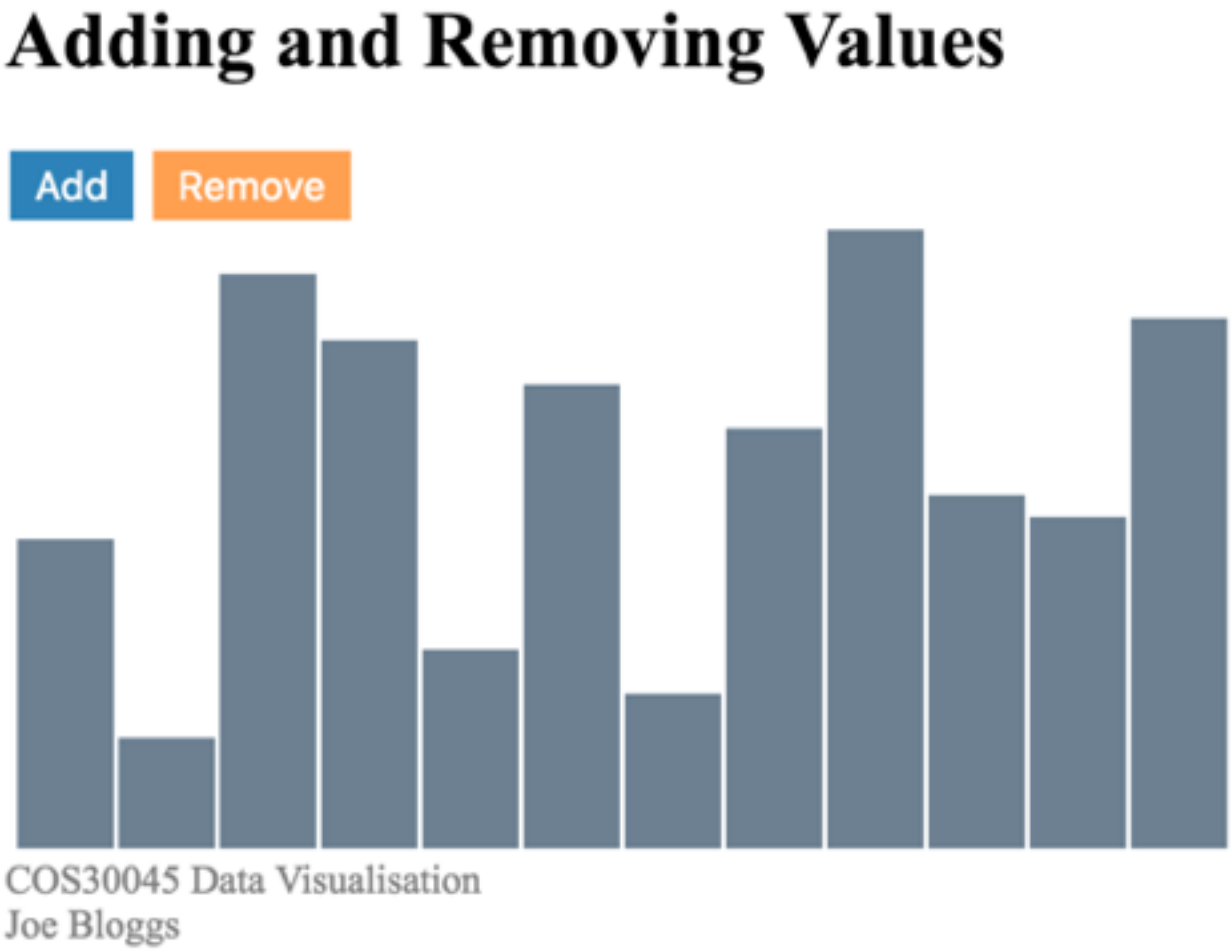
Task 5.3 D3 Adding and Removing Values

ILO	Create web-based interactive visualisations using real-world data sets.
Aim:	Use D3 to generate add and remove data from a bar chart
Resources:	*Textbook:* Murray Ch 9 Murray on ProQuest Murray on Safari (Make sure you use v2 of Murray as per links above)
To be marked as Complete your submission must:	Submit working code that meets the requirements specified in document below. Demonstrate appropriate use of HTML, CSS and D3. Properly formatted code Well commented code with references to code sourced from web, stack overflow etc. where appropriate. Demonstrate and explain code to tutor in class.
Submission	Submit to Doubtfire • code that allows the user to add and remove data from a bar chart Bring code to class to demonstrate to tutor

Note: The functions handling scale have changed between D3 v3 and D3 v4. This is something to be aware of if you are doing your on research into this topic. Make sure you use Murray Ed 2. Code examples from Ed 1 will not work.

Overview

At the end of Task 5.2 we had a bar chart that updated with random values up to 25 and was able to demonstrate a few different transitions. Now we want to try adding and removing data from the bar chart.



Step 1: Start with the code from Task 5.2

In this task we are going to focus on adding (and eventually removing) data from the set. First, delete or comment out the code for generating a random data set from Task 5.2.

Step 2: Generate a new data

First, delete or comment out the code for generating a random data set from Task 5.2. Replace it with some code that generates one new random number:

```
var newNumber = Math.floor(Math.random()* maxValue);
dataset.push(newNumber);

xScale.domain(d3.range(dataset.length));
```

Because we have added a value to the data set we also need to update xScale to recognise that we have one extra value in the set.

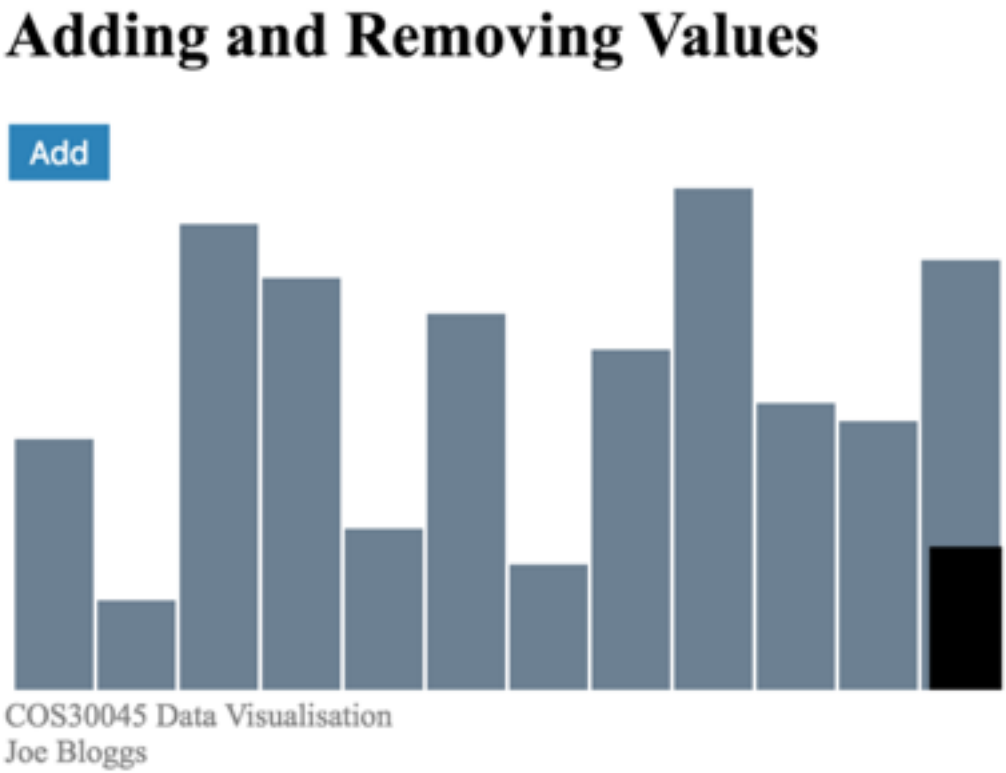
Step 3: Bind data to variable

We can use the `data()` method to select the existing rectangles to a new variable ('bars') and rebind the existing rectangles to it.

```
var bars = svg1.selectAll("rect")
    .data(dataset)
```

However, we don't have enough rectangles at the moment because we just added a new value to the data set. So we use `enter()` to do this. If you remember from the first D3 task, `enter()` creates elements for any data that doesn't have an element already. Then `append()` creates the new rectangle. After that we need to update the x and bar width values with the new data.

If you make these changes and run your code, you will notice that the new bar just sits at the end of the chart, and every subsequent bar is just laid on top of it. It is not being integrated into the chart.



Step 3: Merge the elements

We need to use `merge()` to actually integrate it in with the other existing bars.

```
bars.enter()
    .append("rect")
    .merge(bars)
    .attr("x", function(d, i) {
        return xScale(i)
    })
    .attr("y", function(d) {
        return h - yScale(d);
    })
    .attr("width", xScale.bandwidth())
    .attr("height", function(d) {
        return yScale(d);
    });
```

Step 4: Smooth the transition

Now we have the bars being merged together. However, the transition is not very smooth.

```
bars.enter()
    .append("rect")
    .merge(bars)
    .transition()
    .duration(500)
```

Now we have a smooth transition, but it's flying in from the left. It might be nicer to have it slide in from the right.

We can do this by specifying the start x and y values before the merge.

```
bars.enter()
    .append("rect")
    .attr("x",w)
    .attr("y", function(d) {
        return h - yScale(d);
    })
    .merge(bars)
```

Now it should come in smoothly from the right. Finally, you may have noticed that the new bars are coming in the default colours (i.e., black). Make it so they come in with the same colour as your initial bars.

Step 5: Removing values

Firstly update your button to say ‘Remove’. Then comment out the code for adding values and use `shift()` to remove the first element of the array.

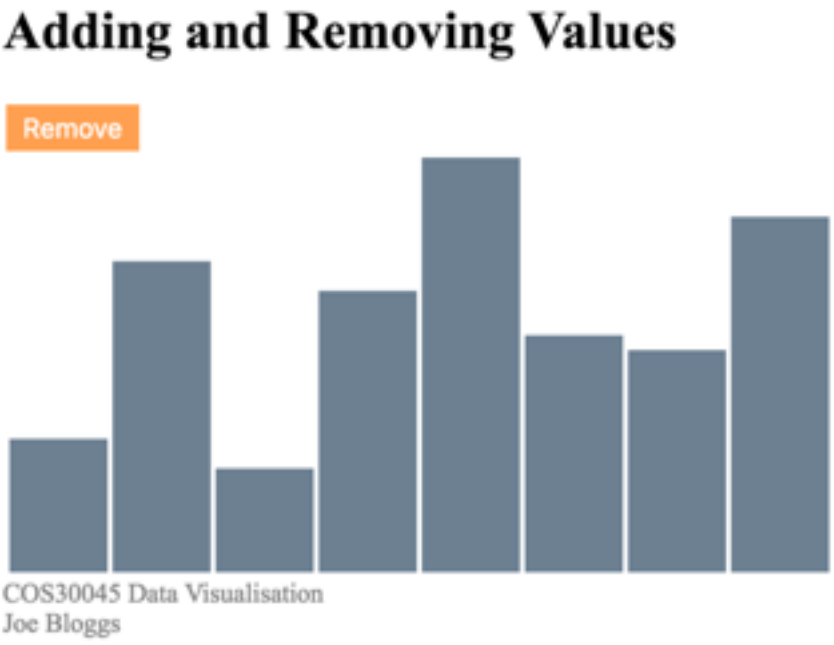
```
d3.selectAll("button")
  .on("click", function() {

    // var newNumber = Math.floor(Math.random()* maxValue);
    // dataset.push(newNumber);

    dataset.shift();
```

We can make the bars exit gracefully by using `.exit()` to transition the element to the right (i.e., to `.attr("x", w)` then we can use `remove()` to take the element from the DOM.

```
    bars.exit()
        .transition()
        .duration(500)
        .attr("x",w)
        .remove();
```



If you have time make it so the user can both add and remove bars from the chart.

Be ready to demonstrate your code in class!