Faculty of Science, Engineering and Technology

COS30045 Data Visualisation

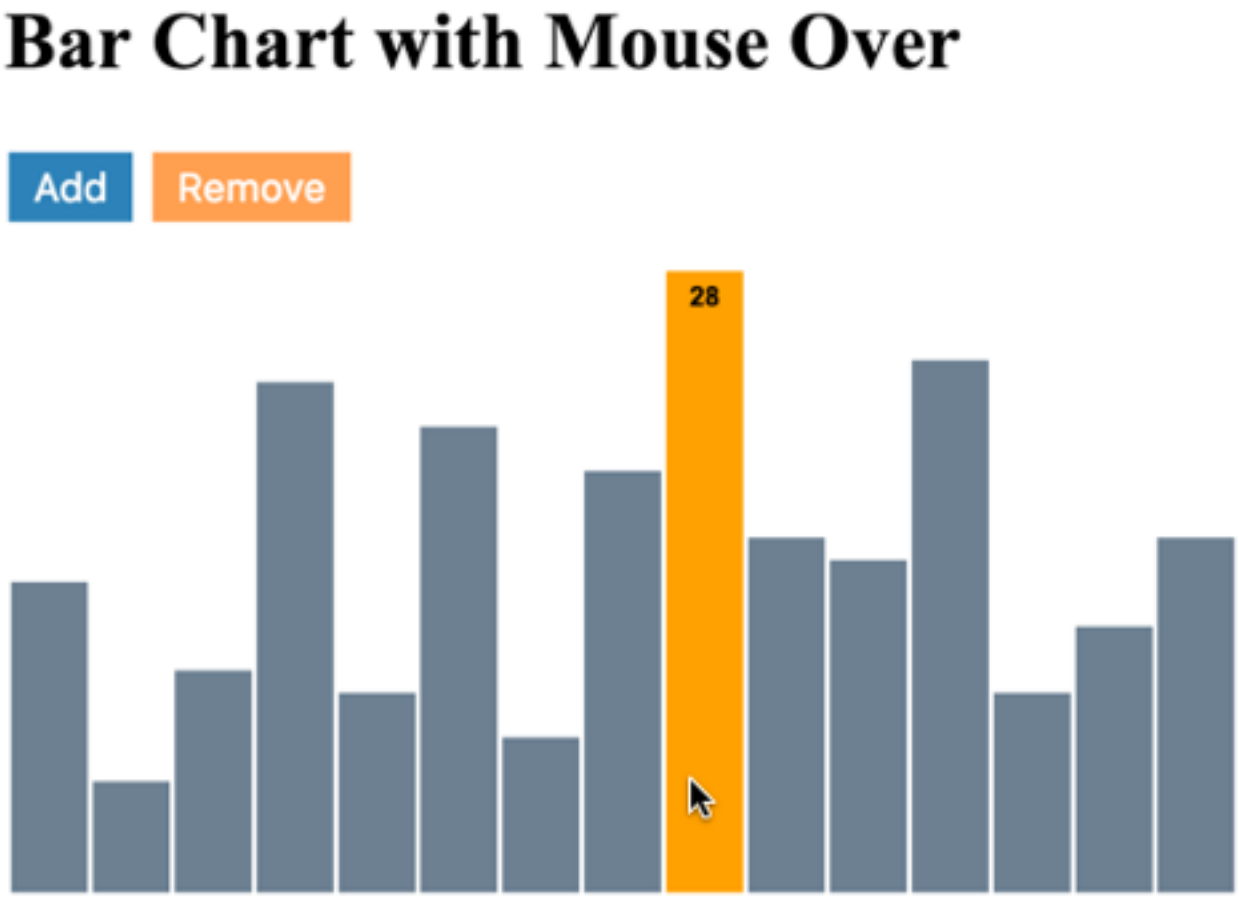
Task 6.1 D3 Interactivity - mouse over effects

ILO	Create web-based interactive visualisations using real-world data sets.
Aim:	Bind event listeners to D3 selections
Resources:	*Textbook:* Murray Ch 10 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 demonstrates a mouse over effects (e.g., hover effect, tooltip) 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 5.3 we had a bar chart that we could add and remove data from. In this task we will add a mouse over effect so that the colour of the bar changes and the data is displayed when the user mouses over it. The effects will disappear when the user moves their mouse away.



Step 1: Start with the code from Task 5.3

In Task 5.3 we used D3 to bind an event lister for mouse click to a button. In this task we will use a similar procedure to generate a mouse over effect.

Step 2: Add mouse over effect

Add the `on()` to your main SVG to change the colour of the selected `rect`. Save and test. You find that when the mouse goes over a rectangle it turns orange. While this is nice, it might also be good for it go back to it's original colour when the mouse has left the rectangle.

```
        .attr("fill", "slategrey")
        .on("mouseover", function() {
            d3.select(this)
             .attr("fill", "orange");
        });
```

Add a mouse out effect to return the colour of the rectangle back to it's original colour. Add a transition, such as a delay to make the mouse in/out smoother.

Step 3: Mouse effect on added bars

Try adding some bars. Does the mouse over effect work on the new bars? Make it work.

Step 4: Add a browser tool tip

Add a browser tool tip that displays the data value. As before, make sure that any new bars also display the tool tip.

```
        .append("title")
        .text(function(d) {
             return "This value is " +d;
         });
```

Step 5: Add a SVG tool tip

We don't have much control over the tool tip generated by a browser. To get more control we can use SVG elements. Start by commenting out the code for the browser tool tip, and the highlight mouse overs.

As for the bar highlight mouse over effect, we will use .on() to trigger the effect. We will read the value of the bar and print the value at the top of the bar.

First identify the x and y positions of where you want the text to appear.

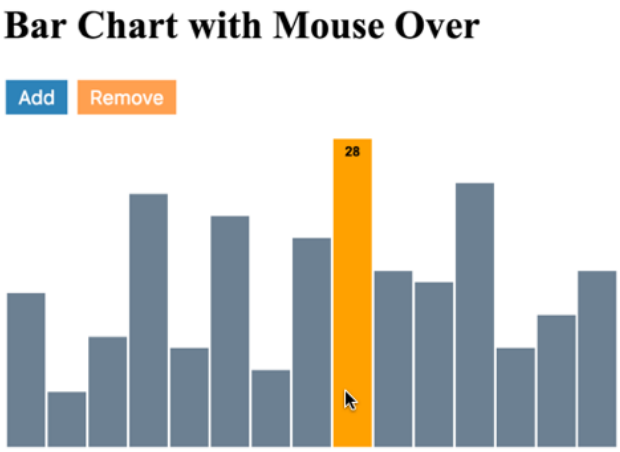
```
    .on("mouseover", function(d) {

        var xPosition = parseFloat(d3.select(this).attr("x"))
        var yPosition = parseFloat(d3.select(this).attr("y"))
```

Then append the text into position.

```
        svg.append("text")
            .attr("id", "tooltip")
            .attr("x", xPosition)
            .attr("y", yPosition)
            .text(d);
```

Save and run your code. You will notice that the number is not well placed or well styled. Also the bar highlight mouse over is no longer evident.

Arrange so that the tool tip appears similar to as follows:



To remove the tool tip when the user removes the mouse, use `remove()`.

```
        .on("mouseout", function(d) {

            d3.select("#tooltip").remove();

        });
```

Save your work and be ready to demonstrate in class.