School of Software and Electrical Engineering

COS30045 Data Visualisation

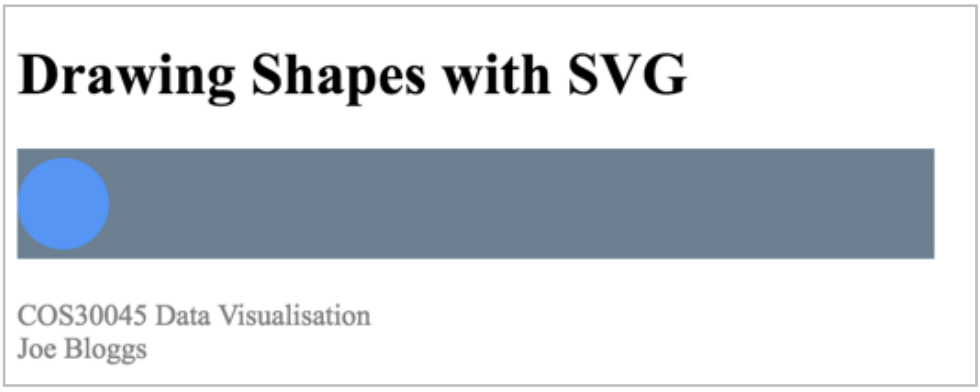
Exercise 1.3 Technology Fundamentals - SVG

ILO	Create web-based interactive visualisations using real-world data sets.
Requirements:	Demonstrate basic familiarity with creating and formatting SVG's by creating a webpage that displays an original arrangement of a variety of SVG shapes (i.e., a pattern created by you) .
Resources:	*Textbook:* Chapter 3 SVG Murray (2017) Interactive Data Visualisation (2nd Ed) on ProQuest Murray on Safari *Web Resources:* *W3Schools: SVG* *codeacademy* *Videos:*
Demonstration:	Demonstrate appropriate and well formatted use of HTML, CSS and SVG code Be prepared to demonstrate Code live to tutor when requested: - code is appropriate for exercise, well formatted and commented - code runs correctly and meets the requirements specified in this exercise - explain programming features and concepts in the code - successfully edit code to change a specified feature of the program

Note: This Exercise Guide is not meant to be fully explanatory. Unless you are already familiar with SVG you will need to read the text book and/or examine web resources. This is something you need to get used to doing when you get out into the 'real' world!

Overview

In this unit you will learn to use D3 to create web-based visualisations. Scaleable Vector Graphics (SVG's) are the building blocks of D3 visualisations. In this exercise you will create and style some SVGs (see screenshot below). In the later exercises we will get D3 to generate the SVGs using some data to specify the SVG's characteristics (e.g., size, shape, colour)



Requirement Set 1: Draw SVG shapes

- ☑ draw a set of at least 3 SVG shapes in various positions
- ☑ colour of shapes is customised (i.e., not default) (e.g., different stroke and fill colours, transparency, stroke width

Step 1 Draw SVG shapes

Start with a basic html template with appropriately labeled meta data and title. The first step to creating an SVG is to create an SVG element which will act as a canvas on which our shapes will sit.

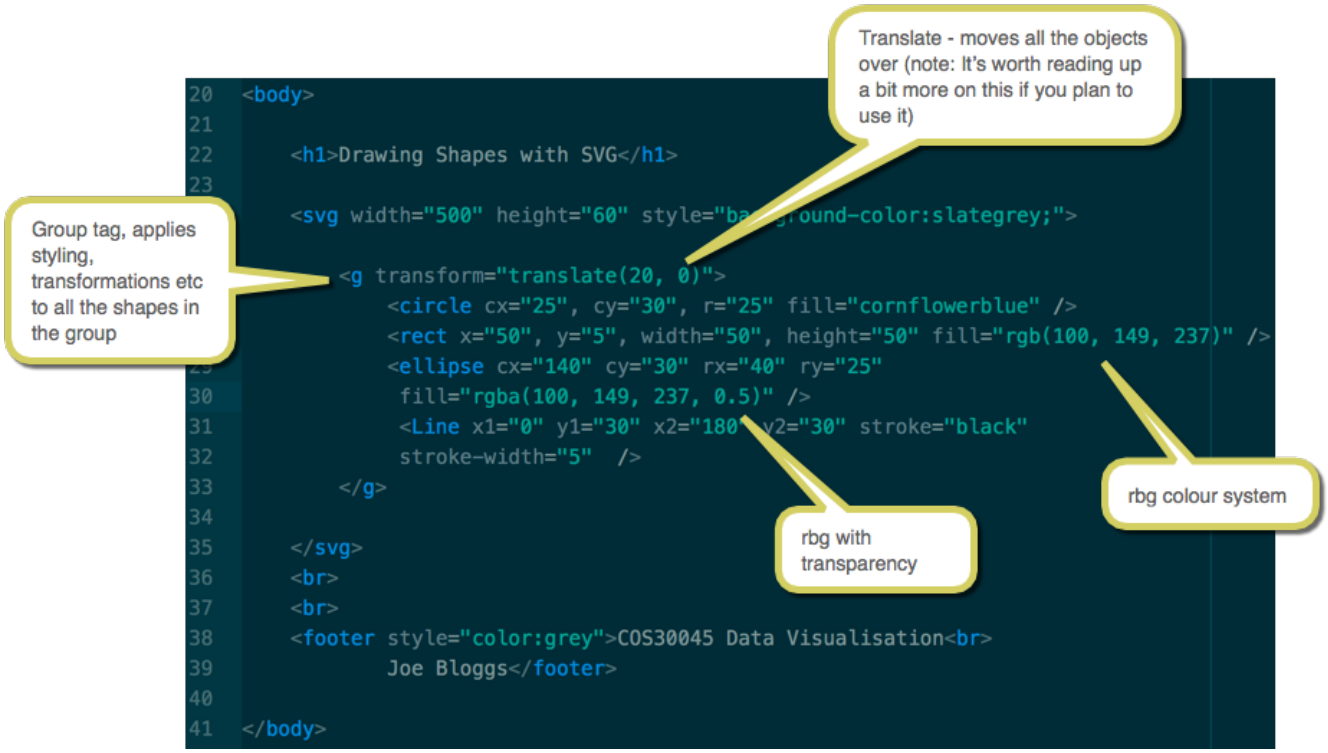
```
20  <body>
21
22      <h1>Drawing Shapes with SVG</h1>
23
24      <svg width="500" height="50">
25
26          <circle cx="25", cy="25", r="25"/>
27
28      </svg>
29
30      <footer>COS30045 Data Visualisation<br>
31              Joe Bloggs</footer>
32
33  </body>
34  </html>
35
```

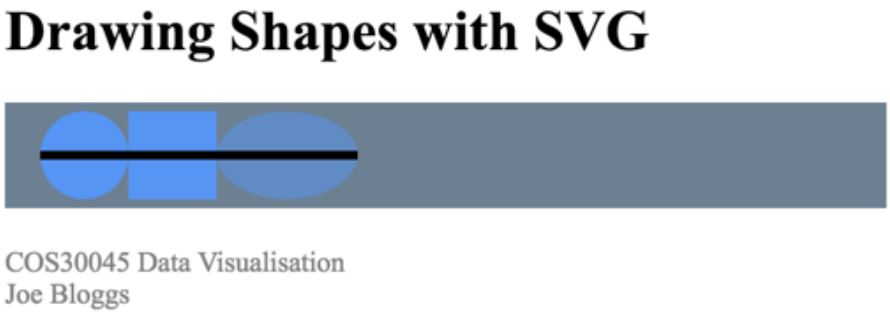
In the body of the code, first create a 500 x 50 px SVG element with the SVG tag. To draw a circle you need to specify the type of shape you want to draw (i.e., a circle), the x and y of the *centre* of the circle and the radius.

If you run the code above it will become apparent that the default colour is black. Change the colour to something a bit more exciting. There are are a number of different ways to specify colour, to start with use a simple colour name which will specify a standard websafe colour (i.e., cornflowerblue). In the example below a background colour is also specified for the SVG element.

```
20  <body>
21
22      <h1>Drawing Shapes with SVG</h1>
23
24      <svg width="500" height="60" style="background-color:slategrey;">
25
26              <circle cx="25", cy="30", r="25" fill="cornflowerblue" />
27
28      </svg>
29      <br>
30      <br>
31      <footer style="color:grey">COS30045 Data Visualisation<br>
32              Joe Bloggs</footer>
33
34  </body>
```

There are a large number of attributes that you can specify for a SVG including animations. Make more circles demonstrating different styling. Do the same with some other shapes such as rectangles, elapses and lines.

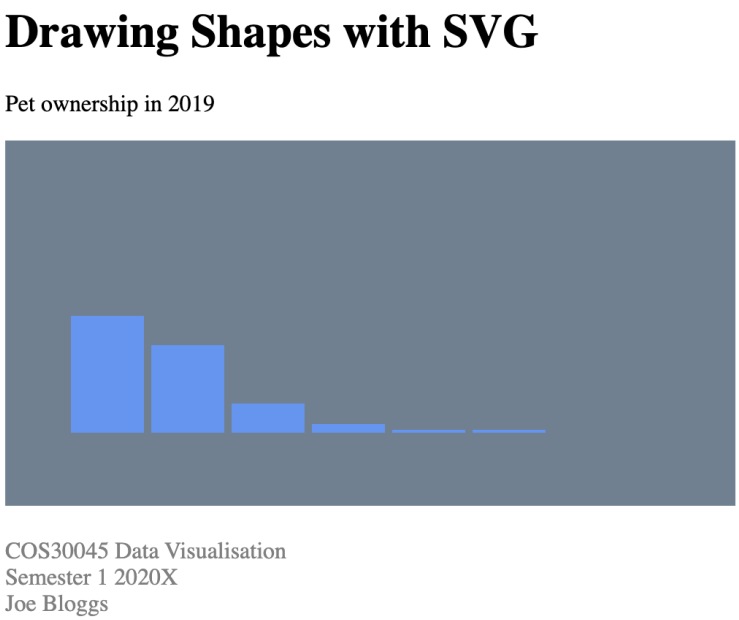




Colour is very important in data visualisation. Try to make your colour combinations harmonious! Try using colorbewer for inspiration. Or w3schools' HTML Colour 'Picker'.

Tip: If you position a shape outside of the main SVG canvas it won't be shown.

If you are feeling adventurous you could try hard coding our 2019 pet data using SVG rectangle shapes as per below.



Next week we will start using D3 to bind data to SVG elements so that we can use the the data to help draw the SVG without hard coding it or creating extra variables. For example, we can take a rectangle SVG and create a ‘bar’ with a height that correlates with the bound data value.