Aula 2- Gráficos de barras, Interações e Transições

Resumo

- Características principais da biblioteca d3.js
- Elementos SVG em d3
- Gráfico de barras (bar chart)
- Interações e transições

2.1 Elementos SVG

No exemplo da aula anterior (d3_1_7.htm), remova todo código dentro do script e adicione uma `div` no `body` com o nome "`div_d3`" antes do script.

```
<div class="div_d3">    </div>
```

No script, selecione essa `div` com o método select a adicione um elemento SVG a mesma.

```
var svg = d3.select('.div_d3').append('svg')
```

Em alternativa pode adicioanr diretamente o elemento svg ao body sem definir nenhum div.

```
var svg = d3.select('body').append('svg')
```

Modifique os attributos width e height para 600 e 300 assim como a cor de fundo (style com o attributo background background-color) para a cor hexadecimal #ffa07a ou outra a sua escolha.

2.2 Desenho de um círculo

Acrescente agora um elemento SVG circle na posição 300, 150 com raio 50 e cor vermelha.
https://www.w3schools.com/graphics/svg_circle.asp

2.3 Desenhos usando SVG

Modifique o exemplo anterior para utilizar o código seguinte.

```
var w = 500;
var h = 50;
```

```
var svg = d3.select("body")
   .append("svg")
   .attr("width", w)
   .attr("height", h);
```

Modifique a cor de fundo para 'teal'.
Adicione o código seguinte e veja o que ocorre.

```
var dataset = [5, 10, 15, 20, 25];

var circles = svg.selectAll("circle")
   .data(dataset)
   .enter()
   .append("circle");

circles.attr("cx", "10")
   .attr("cy", "10")
   .attr("r", "10");
```

Modifique agora o código para que a coordenada em x dos círculos seja atualizada de acordo com o índice do círculo a desenhar, a coordenada y seja metade da altura da janela e o raio seja proporcional ao valor dos dados. Nesta caso as funções tem um argumento adicional, uma variável contadora gerada automaticamente pelo d3. Por exemplo para o atributo cx, pode usar o código:

```
circles.attr("cx", function (d, i) {
   return (i * 50) + 25;
});
```

No final deve obter algo parecido com:

Modifique os dados e veja o que ocorre. Para melhor perceber o que ocorre pode colocar o código "debugger" na função associada a variável cx e ver o valor da variável d.

2.4 Círculos com cores diferentes (opcional)

Utilize o método `attr` nos círculos para modificar as propriedades dos círculos (use uma cor para o `fill`, outra para `stroke` e modifique o `stroke-width` de acordo com metade do valor dos dados). Veja o resultado. Modifique agora o código para especificar cores diferentes para cada círculo e use um conjunto de dados com um maior número de elementos. Verifique o que acontece.

2.5 Gráfico de barras com SVG

Vamos revisitar o exemplo do gráfico de barras mas agora utilizando o elemento `rect` do SVG em vez do div. No exemplo 2.3 altere os dados para:

```
var dataset = [5, 10, 13, 19, 21, 25, 22, 18, 15, 13, 11, 12, 15, 20, 18,
17, 16, 18, 23, 25];
```

Substitua o código dos círculos pelo código seguinte para desenhar um retângulo.

```
var rects = svg.selectAll("rect")
    .data(dataset)
```

```
        .enter()
        .append("rect")
        .attr("x", 0)
        .attr("y", 0)
        .attr("width", 20)
        .attr("height", 100);
```

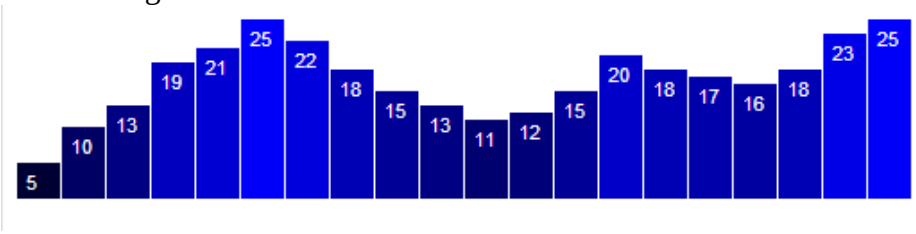
Modifique agora o código para que a posição x dos retângulos seja calculada em função do indice, da largura da janela e do número de elementos no `array` de dados (`i * (w / dataset.length)`.
Modifique também a largura para que a mesma dependa também do número de elementos a representar (`w / dataset.length - 1)`.
Modifique o numero de valores no array de dados e veja o resultado.
Modifique agora o código para que a altura dos retângulos seja proporcional a 4 vezes o valor do array (modifique a altura da janela se necessário). O que observa? Esse problema deve-se ao facto que o SVG define a coordenada `0,0` do `rect` como a coordenada superior esquerda. Pode resolver isso calculando a coordenada superior em y como a altura menos o valor a representar.
Finalmente para adicionar alguma cor, pode associar ao atributo `fill` dos retângulos o valor `"rgb(0, 0, " + (d * 10) + ")".`

2.6 Etiquetas

Adicione o código seguinte para permitir visualizar texto junto de cada retângulo no gráfico:

```
svg.selectAll("text")
   .data(dataset)
   .enter()
   .append("text")
   .text(function (d) {
          return d;
   })
   .attr("x", function (d, i) {
         return i * (w / dataset.length);
   })
   .attr("y", function (d) {
          return h - (d * 4);
   });
```

Modifique os atributos do texto (nomeadamente o text-anchor) para obter um gráfico semelhante ao seguinte:



2.7 Interação

Adicione o código seguinte ao elemento svg com todos os rectângulos para associar o evento clique a função de `callback` e veja o resultado.

```
rects.on("click", function(d) {
       alert("O valor é "+ d);
});
```

Modifique agora o exemplo para responder ao clique ordenando as barras por ordem crescente ou decrescente aproveitando o código seguinte. Este código utiliza uma função `sort`, especificando na função as regras para comparação entre os elementos.

```
//Define sort order flag
var sortOrder = false;
//Define sort function
var sortBars = function() {
	//Flip value of sortOrder
	sortOrder = !sortOrder;

	svg.selectAll("rect")
		.sort(function(a, b) {
			if (sortOrder) {
				return d3.ascending(a, b);
			} else {
				return d3.descending(a, b);
			}
		})
		.attr("x", function(d, i) {
		return i * (w / dataset.length);
		})
	};
```

2.8 Transição

Ao utilizar a palavra chave `transition()`, o d3 deixa de atualizar as variáveis instantaneamente e introduz um elemento temporal, permitindo a interpolação entre os valores iniciais e finais quando ocorrem modificações dos mesmos.
Adicione no exemplo anterior o elemento `transition()` dentro da função `sortBars` antes do `.attr("x"…)` e veja o resultado. Note que transição deve ser colocada antes do elemento a animar. Por omissão o tempo da animação é de 250 ms. Utilize a seguir ao elemento `.transition()` o elemento `.duration(tempo)` para especificar outro tempo para a animação. Experimente ainda o elemento `ease( )` que permite configurar o tipo de animação. Experimente alguma das opções: "cubic-in-out","linear","circle","elastic " ou "bounce".
Modifique ainda o exemplo para que todos os elementos não sejam animados ao mesmo tempo, permitindo as barras realizarem a animação com um atraso proporcional a sua posição no array. Para tal utilize o elemento `.delay` definindo uma função adequada.
Ver:
https://github.com/d3/d3-transition/blob/master/README.md#transition_delay

Opcional: Modifique o exemplo para visualizar as etiquetas (ver o exemplo 2.6) e animar as mesmas juntas com as barras.