# Learning D3.js(1) 学习制作一个柱形图/直方图

https://www.douban.com/note/267039793/

Learning D3 是俺准备推出的原创+翻译教程~作者主要是俺与各位d3大牛们。。。  
数据可视化是个热门话题，而D3.js则是前端数据可视化的超级工具。俺作为一个前端，有志于推动D3.js在中国的发展。。  
但是中文网络上关于D3.js的资料比较少，因此本系列教程将会以翻译外文D3教程及我个人撰写的形式来进行。不过虽然是翻译，但肯定不会照着抄，我会加入很多自己的理解。也希望大家学习的时候，多看d3.js的api与实例 <https://github.com/mbostock/d3/wiki/API-Reference>  
然后呢，俺翻译完后，会注明各种链接。俺的原创文章若有参考，也会注明链接。  
第一篇是简单易学的  <http://mbostock.github.com/d3/tutorial/bar-1.html>     学习制作一个柱形图/直方图  
现在有个数组  
var data = [4, 8, 15, 16, 23, 42];  
我们尝试利用柱形图可视化展现这些数据。 接下来的教程呢，会从简单的html与复杂点的svg来讲如何创建一个柱形图。  
HTML  
我们首先需要一个容器来装这个图表  
var chart = d3.select("body").append("div")  
    .attr("class", "chart");  
这段代码可以选择document body，然后以此作为新图表容器的父元素。除了文档根元素之外每个可见节点都是需要有父元素的。  
然后div作为图表的容器创建并追加到body中。append会把元素追加到父元素末尾，因此图表会在body末尾被展现。  
D3可以使用链式调用的设计模式，这个写过jq的童鞋都不会陌生了。上面代码使用链式调用有啥好处呢？  
上面的attr方法可以设置属性，之后返回当前的选择元素集合,这样，chart变量就指向图表的容器。(因为你是给div进行的设置)。这种方法可读性很高，代码也很干净。  
attr方法设置了图表容器的class属性，这样我们就可以给图表元素应用样式表了。这对一些静态样式比如字体啊，背景啥的设置是非常方便的。我们可以使用style标签进行设置，也可以link外联css样式来设置。比如这样  
.chart div {  
  font: 10px sans-serif;  
  background-color: steelblue;  
  text-align: right;  
  padding: 3px;  
  margin: 1px;  
  color: white;  
}  
之后我们给图表容器里面添加几个div元素，并依据开始的数组值来给这些元素规定宽度。  
chart.selectAll("div")  
    .data(data)  
  .enter().append("div")  
    .style("width", function(d) { return d \* 10 + "px"; })  
    .text(function(d) { return d; });  
demo可以见 <http://www.d3js.cn/demo/bar1.html>  
上面的代码可能不是很好理解。  
首先呢我们使用selectAll方法会选择chart内部所有的div元素。但注意，这个集合是空的，因为内部并没有div元素。  
为什么要创建一个空的集合呢，下面的data方法，可以给这个集合绑定一组数据，绑定后比较有意思了，返回的是一系列的“空房间”:占位节点。数组中的每个元素占据一个节点。这样你就可以给这些节点追加元素了  
注意，chart.selectAll('div')的作用，经过我的测试，确实是仅仅创建了一个空的集合，这个集合是为了给下面的数据切分用的。以我的理解这很类似于一个文档碎片之类。就比如你买了个大房子，然后人家给你个数组，你按照这个数组切分成 两室一厅啥的。  
你当然也可以使用chart.selectAll('a'),chart.selectAll('abc')等等。  
空房间创建好后，我们使用enter()来“进入”这个房间，之后每个房间追加一个div，然后用style给房间“装修”一下。  
那么text()方法可以设置每个柱形图的文字。这个方法对于数字，会使用javascript自己的字符串转换，类似于toString.这个方法是有问题的，比如说  
3.0000000000000000000005.toString()//返回3  
然后d3提供了d3.format类,很类似python的string formatting。这个类对数字格式的控制更有效，包括支持千分位，支持更精确的小数位数之类。  
d3.format可以参考  <https://github.com/mbostock/d3/wiki/Formatting>  
我们的图表如图所示：  
上面你会注意到一个问题，这里我们使用.style("width", function(d) { return d \* 10 + "px"; })，根据数值乘以10来设置宽度，这是比较“不健壮”的做法。  
我们的想法是，容器宽度总宽定死，为了避免突然出现一个很大的值来撑开，不需要计算“比例”，我们可以使用一个非常好用的类：D3’s linear scale (我翻译为直线标度)  
var x = d3.scale.linear()  
    .domain([0, d3.max(data)])  
    .range(["0px", "600px"]);  
这个x是一个动态的对象，可以把数据转换为我们想要的宽度，这里利用domain方法来设置需要匹配的数值范围(可以使用d3.max方法拿到最大值)，之后利用range来规定容器的宽度。这样做，会使图表画起来更加的灵活，你不需要考虑具体的数据，只需要专注于图表即可。  
举例来讲，当我们遇到了42（数组最大值），x返回600px,同样遇到data=21时，可以返回300px  
之后我们就可以使用下面的写法  
chart.selectAll("div") .data(data) .enter().append("div") .style("width", x) .text(String);  
demo2 <http://www.d3js.cn/demo/bar2.html>  
HTML写起来简洁,但是不太灵活。比如我们需要在背景显示参考线,或生成列时,很难使用纯HTML实现。当然如饼图和流图之类的图表更不可能做出来了。。幸运的是,有一个方便的选择:可缩放矢量图形(SVG)。我们在下一部分会讲到。  
d3的api <https://github.com/mbostock/d3/wiki/API-Reference>  
 SVG  
注意下面svg的例子请用现代浏览器看，比如chrome  
其实使用svg与html很类似，但svg更加灵活。首先，我们使用svg标签来代替div标签，表示一个图表容器.  
var chart = d3.select("body").append("svg")  
    .attr("class", "chart")  
    .attr("width", 420)  
    .attr("height", 20 \* data.length);  
svg是矢量图形。这里估计你也注意到了，svg单位不需要加px后缀。而且通过像素调节svg的大小，也不会损失图像质量，你也可以使用百分比做相对定位。  
接下来我们跟上文一样，使用D3’s linear scale来给x轴划定下直线标度。  
var x = d3.scale.linear()  
    .domain([0, d3.max(data)])  
    .range([0, 420]);  
SVG不像HTML一样拥有自动的流式布局。svg图形的定位相对于左上角,称为原点。因此,默认情况下,柱形图也会从那个点开始绘制。为了解决这个问题,我们设置明确的y坐标和高度:  
chart.selectAll("rect")  
    .data(data)  
  .enter().append("rect")  
    .attr("y", function(d, i) { return i \* 20; })  
    .attr("width", x)  
    .attr("height", 20);  
注意了，我们仍然使用了selectAll方法，而这里仍然仅仅是为了创建一个空的集合，见前文。然后我们这里利用rect(矩形)代替了上文的div，用来装柱形图。上面其实是按照顺序制作了六个rect:矩形。注意attr方法中，i表示这个元素在数组中的序列。  
对于svg来讲css有些不同。fill属性决定了svg图形的颜色，而你也可以使用stroke属性设置图形的边框。  
.chart rect { stroke: white; fill: steelblue; }  
效果如下：  
  
你可能会注意到，我们上面用了20表示一个绝对距离，而每个矩形距离原点的距离是0,20,40,60...有没有什么办法，可以类似scale.linear一样，规定一个最大值，但是这个值是一个累加的等差数列呢(注意与linear的区别，linear是一个比例相关的)  
var y = d3.scale.ordinal()  
    .domain(data)  
    .rangeBands([0, 120]);  
使用以上方法即可，利用d3.scale.ordinal(),仍然使用domain来设置需要匹配的数值范围，rangeBands我们可以理解为开始坐标与结束坐标。  
这样y跟上面的x类似，变成了一个动态对象，可以根据数据改变返回值  
我们重新改写函数  
chart.selectAll("rect")  
    .data(data)  
  .enter().append("rect")  
    .attr("y", y)  
    .attr("width", x)  
    .attr("height", y.rangeBand());  
对于输入的数  
 [4, 8, 15, 16, 23, 42]  
y分别返回0,20,40,60,80,100  
而y.rangeBand返回的是总长度/数组的元素个数 = 20.这样我们的布局就更加的灵活了，只需要关注图标的长宽即可。  
我们也可以使用一些新的属性来渲染图表的标签  
chart.selectAll("text")  
    .data(data)  
  .enter().append("text")  
    .attr("x", x)  
    .attr("y", function(d) { return y(d) + y.rangeBand() / 2; })  
    .attr("dx", -3) // padding-right 右边距  
    .attr("dy", ".35em") // vertical-align: middle 标签垂直居中  
    .attr("text-anchor", "end") // text-align: right 文字水平居右  
    .text(String);  
上面我们用text创建标签，注意标签的位置:框的高度+框高度的一半，以及我加了注释的那些属性。你可以自己试验一下  
这些dx,dy之类属性都在svg标准里规定了 <http://www.w3.org/TR/SVG/text.html> 标准非常的复杂。  
当然我们只需要了解一些布局用的常用属性，比如对齐，边距等就够用了。  
效果如下图所示，现在跟html的已经基本一样了  
  
demo见<http://www.d3js.cn/demo/bar3.html>  
之后我们给图表增加新的功能-纵向的标度。我们使用svg的标签，g与line来完成。  
var chart = d3.select("body").append("svg")  
    .attr("class", "chart")  
    .attr("width", 440)  
    .attr("height", 140)  
  .append("g")  
    .attr("transform", "translate(10,15)");  
这里的g类似于html标签的div，是一个容器元素。设置transform属性可以影响元素子元素的定位。这里我们利用translate，实质是设置边距，可以给上面的文字留下空白。当然你也可以利用这个属性使用丰富多彩的图形效果，比如缩放，旋转，裁剪等。详细参考 transform  
注意了，这时候chart返回的是g容器，而不是svg了  
接下来我们要考虑一个因素，图标的标注都是有一定的刻度的，比如说我们的数据从0-40，那么刻度就应该类似0,5,10,15,20这样。  
d3提供了ticks方法来方便我们制作这种周期性的刻度。他会根据具体的数据，与给定的刻度区间数量来计算刻度。比如  
    chart.selectAll("line")  
            .data(x.ticks(10))  
这样，他会给你把数据分为十个刻度展示。其实这个方法让我比较纠结，实验了很多次也没得到什么理想的结果。d3官网对此方法的解释：  
Returns approximately count representative values from the scale's input domain. The returned tick values are uniformly spaced, have human-readable values (such as multiples of powers of 10), and are guaranteed to be within the extent of the input domain. Ticks are often used to display reference lines, or tick marks, in conjunction with the visualized data. The specified count is only a hint; the scale may return more or fewer values depending on the input domain.  
大概意思是说返回的数是个近似的数，代表数据的一个范围。返回值是均匀间隔,而且是人类可读的值(比如10的倍数)。  
这个方法通常结合可视化数据用于显示参考线,或标记。而你传入的这个值只是一个提示,返回间隔的多少取决于输入的数据。  
大家可以自己尝试下。  
加上参考线  
    chart.selectAll("line")  
            .data(x.ticks(10))  
            .enter().append("line")  
            .attr("x1", x)  
            .attr("x2", x)  
            .attr("y1", 0)  
            .attr("y2", 120)  
            .style("stroke", "#ccc");  
绘制线非常简单，只需要指定x1,x2,y1,y2 分别是起点与终点的横纵坐标。  
然后我们如法炮制，加上标注线上面的文字  
    chart.selectAll("a")  
            .data(x.ticks(10))  
            .enter().append("text")  
            .attr("class", "rule")  
            .attr("x", x)  
            .attr("y", 0)  
            .attr("dy", -3)  
            .attr("text-anchor", "middle")  
            .text(String);  
这里跟上面的参考线注意selectAll里面随便搞个就行，只要不是已经有的元素。前文也说了。  
最终效果：<http://www.d3js.cn/demo/bar4.html>  
  
上面的教程信息量比较大，涉及了d3.js几个核心的概念：数据绑定，属性，文字，svg等等。  
作者：新浪微博 @宇宙一片小囧  
  
原文链接：<http://www.civn.cn/p/12311.html>