<script type="text/javascript" src="d3.js"></script>

**选择元素：**  d3.select() 选中第一个

D3.selectAll() 选中所有

**添加元素：** [selection]. append(“p”) 在选择集末尾插入p元素

[selection]. insert(“p”, “#id”) 在选择集（id为id的元素）前面插入p元素

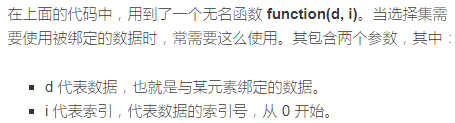
**删除元素：** d3.select(“body”).select(“#id”).remove() 删除制定id的元素

**绑定数据：** dataum() 绑定一个数据到选择集上 多了个\_data\_属性

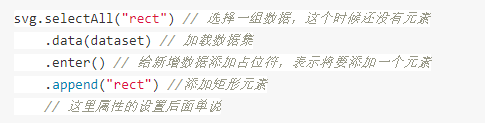
data() 绑定一个数组到选择集上，分别对应绑定

**enter()** 数组长度比选择集的DOM元素多，用enter()创建一个新元素的占位符。再append( )在占位符的DOM中插入一个元素

**function(d, i)**

****

**D3编程套路：** 选择元素-绑定数据-添加新元素的占位符-添加元素-设置属性或样式。



Q??????

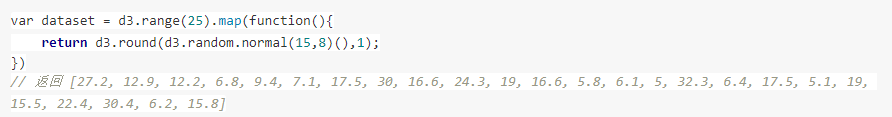
Enter().append()之后选中的元素就只有append添加的这几个了。之前的不选中。

**添加画布** d3.select(“body”).append(“svg”)

.attr(“width”, WIDTH)

.attr(“height”, HEIGHT);

**生成测试数据**



* d3.range(25) 生成25个数字：[0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24]。当然我们没有用它生成的数字，而只是用它来循环。真正生成数字的地方在map函数里。
* map函数，顾名思义映射，也就是生成一个对应该位置的一个数字（当然这里强大之处不仅仅是一个数字，还可以是对象）。
* d3.round函数对随机数保留一位小数四舍五入。V4没有了这个函数
* d3.randomNormal(15,8)()生成一个满足期望是15（默认0），方差是8（默认1）的正态分布的随机数。

**比例尺**

**线性** var x = **d3.scaleLinear().**domain([10, 130]).range([0, 960])

x(20) //80 d3.scaleLinear()返回的事函数，因此可以用x(20)

x.invert(80) //20 相当于是知道y值算x值

**domain([domain])**  一般是两个数字指定domain，也可以两个以上

.domain([-1, 0, 1]).range(“red”, “white”, “green”) [-1,0]对应红白，[0, 1]对应白绿

**rangeRound()**  取整后的值域，也就是相应的y值取整

**clamp(true)** 超出范围的会被归到边界值

**序数** var index = [0, 1, 2, 3, 4];

var color = [“red”, “blue”, “green”, “yellow”, “black”];

**d3.scaleOrdinal()**.domain(index).range(color)

**坐标轴**



生成的坐标轴实质是line和text的组合，所以设置样式的时候，可以.axis line .axis text。

**d3.axisTop/axisRight/axisBottom/axisLeft** 创建一个上/右/底/左部轴生成器

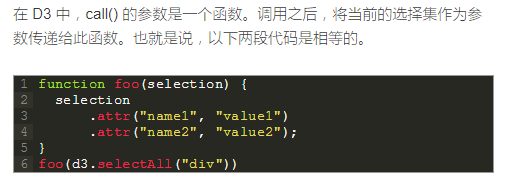
**指定比例尺** var axis = d3.axisLeft(scale);

**指定刻度数量** axis.ticks(8);

**在svg中添加坐标轴** svg.append(“g”).call(axis);

**刻度格式** axis.tickFormat(d3.format(",.0f"));

或 axis.ticks(10, “,.0f”);



**动态效果**

d3.selectAll( ).**transition()** //启动过渡

.duration(1000) //过渡持续的时间1000ms

.ease(d3.easeLinear) //指定过渡的方式

.delay(1000) //延迟1000ms后开始转变

.attr(); / .style();

或者 var t = d3.transition()

.duration(750)

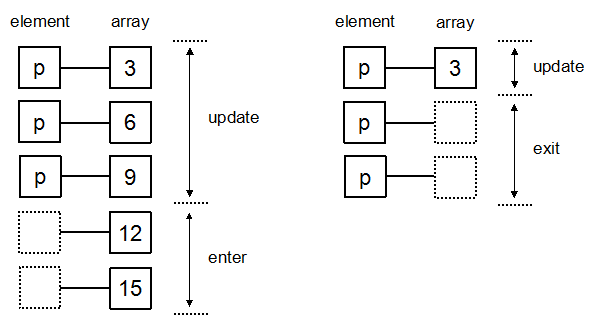
.ease(d3.easeLinear);

d3.selectAll(".apple").**transition(t)**

.style("fill", "red");

**选择集与数据数量关系不确定 Update/Enter/Exit**

如果数组为 [3, 6, 9, 12, 15]，将此数组绑定到三个 p 元素的选择集上。可以想象，会有两个数据没有元素与之对应，这时候 D3 会建立两个空的元素与数据对应，这一部分就称为 Enter。而有元素与数据对应的部分称为 Update。如果数组为 [3]，则会有两个元素没有数据绑定，那么没有数据绑定的部分被称为 Exit。示意图如下所示。



var p = d3.select(“body”).selectAll(“p);

**绑定数据数量 > 对应元素**

var update = p.data(dataset);

var enter = update.enter(); 也就是 p.data(dataset).enter();

update一般用于更新属性值

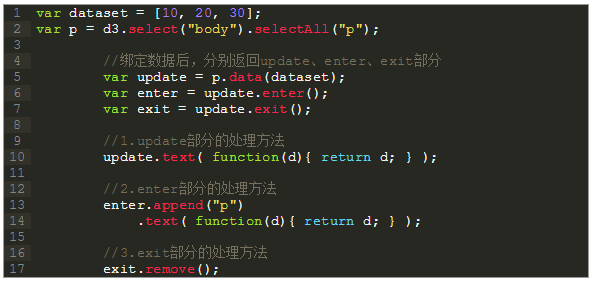
enter 一般用于添加元素后赋予属性值

**绑定数据数量 < 对应元素**

var update = p.data(dataset);

var exit = update. exit(); 也就是 p.data(dataset).exit();

exit 这里只是表明哪一部分是exit，并没有删除多余元素，但一般用于删除元素



**交互**

[selection].**on**(监听事件”click”, 响应函数function(d, i){

d3.select(this)…..

});

事件被监听到后，会存到d3.event对象，可以console.log查看

**鼠标常用的事件有：**

click：鼠标单击某元素时，相当于 mousedown 和 mouseup 组合在一起。

mouseover：光标放在某元素上。

mouseout：光标从某元素上移出来时。

mousemove：鼠标被移动的时候。

mousedown：鼠标按钮被按下。

mouseup：鼠标按钮被松开。

dblclick：鼠标双击。

**键盘常用的事件有三个：**

keydown：当用户按下任意键时触发，按住不放会重复触发此事件。该事件不会区分字母的大小写，例如“A”和“a”被视为一致。

keypress：当用户按下字符键（大小写字母、数字、加号、等号、回车等）时触发，按住不放会重复触发此事件。该事件区分字母的大小写。

keyup：当用户释放键时触发，不区分字母的大小写。

**触屏常用的事件有三个：**

touchstart：当触摸点被放在触摸屏上时。

touchmove：当触摸点在触摸屏上移动时。

touchend：当触摸点从触摸屏上拿开时。

**d3.json()**

通过 Ajax 请求服务器的数据，然后用 JSON.parse() 将返回的字符串转换成 JavaScript 对象的结果。——异步请求

**d3.csv()**

保存了一个数组，数组中的每个元素都是一个对象。

表格文件的读取其实都是用了d3.dsv(“,”, filename), 根据分隔符判断

乱码解决：更改d3.dsv(“,”, “text/csv; charset=gb2312”);的第二项参数

**Datum()**

对于选择集中的每一个元素，增加了一个\_\_data\_\_属性，值为datum(value)的参数value. 此处的value并非一定要是number（数值）型，也可以是string（字符串）、boolean（布尔型）和object（对象）。如果使用undefined和null作为参数，则将不会创建\_\_data\_\_属性。

var p = d3.select(“body”).selectAll(“p”);

p.datum(7); //\_\_data\_\_值为7

p.datum(); //\_\_data\_\_值还是7

p.datum(8); //\_\_data\_\_值变为8；

p.datum(8).text(function(d, i){

return d + “ “ + I;

});

p.datum(9).append(“span”).text(function(d, i){

return d + “ “ + I; //被绑定的数据传递给子元素，p和span都有绑定数据9

}); //text操作是针对span

**Data(array, [function()])**

默认情况下，data()函数根据索引号依次绑定数组各项

也可以使用键函数指定绑定顺序

**颜色插值和线性渐变**

<http://www.ourd3js.com/wordpress/1111/>

**Category Scales 用于颜色 和 d3.scaleOrdinal一起使用**

var color = d3.scaleOrdinal(d3.schemeCategory20);

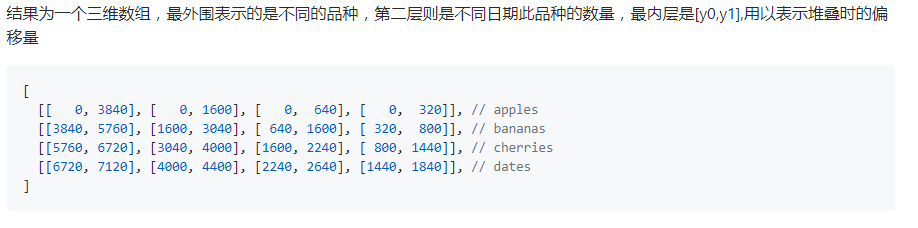
**力导向图**

alphaTarget以及alpha的含义是什么？

猜想alpha值与平衡状态相关，因为alpha<0.001时，认为平衡，停止计算。但是在拖动一个点时，将simulation.alphaTarget(0.3).restart();这个时候alpha也不是从0.3开始或者结束的。

经过试验，大致能理解是指在拖曳过程中alpha值不能大于0.3。整个拖曳过程，其他节点位置有所变化，为了保证alpha值不能大于0.3。所以当拖曳结束要将alphaTarget设回0，即最终还是alpha<0.001时平衡。

**D3.stack()**



**D3.area()**

会执行一个类似data()的操作。

生成的是path的d属性数据 .attr(“d”, area)