# [D3中常用的比例尺](https://segmentfault.com/a/1190000011006780)

D3中有个重要的概念就是比例尺。比例尺就是把一组输入域映射到输出域的函数。映射就是两个数据集之间元素相互对应的关系。比如输入是1，输出是100，输入是5，输出是10000，那么这其中的映射关系就是你所定义的比例尺。

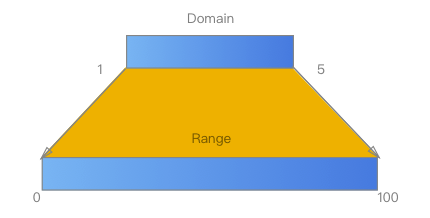
D3中有各种比例尺函数，有连续性的，有非连续性的，本文对于常用比例尺进行一一介绍。

1. d3.scaleLinear() 线性比例尺

使用d3.scaleLinear()创造一个线性比例尺，而domain()是输入域，range()是输出域，相当于将domain中的数据集映射到range的数据集中。

**let** scale = d3.scaleLinear().domain([1,5]).range([0,100])

映射关系：



接下来，我们来研究这个比例尺的输入和输出。

scale(1) // 输出:0

scale(4) // 输出:75

scale(5) // 输出:100

刚才的输入都是使用了domain区域里的数据，那么使用区域外的数据会得出什么结果呢？

scale(-1) // 输出:-50

scale(10) // 输出:225

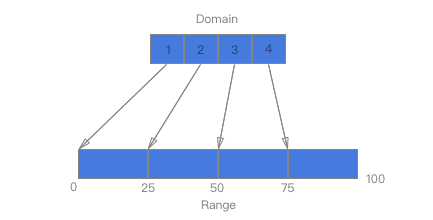
所以这只是定义了一个映射规则，映射的输入值并不局限于domain()中的输入域。

2. d3.scaleBand() 序数比例尺

d3.scaleBand()并不是一个连续性的比例尺，domain()中使用一个数组，不过range()需要是一个连续域。

**let** scale = d3.scaleBand().domain([1,2,3,4]).range([0,100])

映射关系：



看一下输入与输出：

scale(1) // 输出:0

scale(2) // 输出:25

scale(4) // 输出:75

当输入不是domain()中的数据集时：

scale(0) // 输出:undefined

scale(10) // 输出:undefined

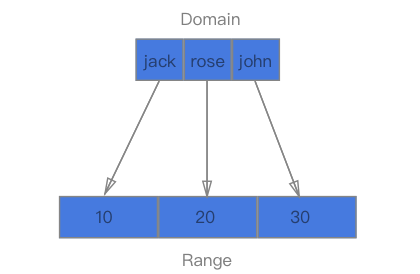
由此可见，d3.scaleBand()只针对domain()中的数据集映射相应的值。

3. d3.scaleOrdinal() 序数比例尺

d3.scaleOrdinal()的输入域和输出域都使用离散的数据。

**let** scale = d3.scaleOrdinal().domain(['jack', 'rose', 'john']).range([10, 20, 30])

映射关系：



输入与输出：

scale('jack') // 输出:10

scale('rose') // 输出:20

scale('john') // 输出:30

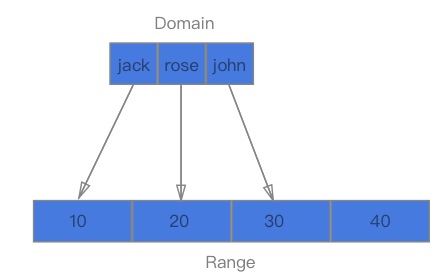
当输入不是domain()中的数据集时：

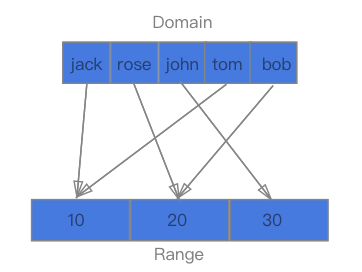
scale('tom') // 输出:10

scale('trump') // 输出:20

输入不相关的数据依然可以输出值。所以在使用时，要注意输入数据的正确性。

我们从上面的映射关系中可以看出，domain()和range()的数据是一一对应的，如果两边的值不一样呢？下面两张图说明这个问题：





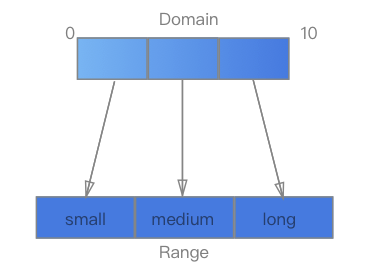
domain()的值按照顺序循环依次对应range()的值。

4. d3.scaleQuantize() 量化比例尺

d3.scaleQuantize()也属于连续性比例尺。定义域是连续的，而输出域是离散的。

**let** scale = d3.scaleQuantize().domain([0, 10]).range(['small', 'medium', 'long'])

映射关系：



输入与输出：

scale(1) // 输出:small

scale(5.5) // 输出:medium

scale(8) // 输出:long

而对于domain()域外的情况：

scale(-10) // 输出：small

scale(10) // 输出：long

大概就是对于domain()域的两侧的延展。

5. d3.scaleTime() 时间比例尺

d3.scaleTime()类似于d3.scaleLinear()线性比例尺，只不过输入域变成了一个时间轴。

**let** scale = d3.scaleTime()

.domain([**new** Date(2017, 0, 1, 0), **new** Date(2017, 0, 1, 2)])

.range([0,100])

输入与输出：

scale(**new** Date(2017, 0, 1, 0)) // 输出:0

scale(**new** Date(2017, 0, 1, 1)) // 输出:50

时间比例尺较多用在根据时间顺序变化的数据上。另外有一个d3.scaleUtc()是依据世界标准时间(UTC)来计算的。

6. 颜色比例尺

D3提供了一些颜色比例尺，10就是10种颜色，20就是20种：

d3.schemeCategory10

d3.schemeCategory20

d3.schemeCategory20b

d3.schemeCategory20c

// 定义一个序数颜色比例尺

**let** color = d3.scaleOrdinal(d3.schemeCategory10)

7. 其他比例尺

另外有一些函数比例尺的功能，从名称上就可见一斑。

d3.scaleIdentity() // 恒等比例尺

d3.scaleSqrt() // 乘方比例尺

d3.scalePow() // 类似scaleSqrt的乘方比例尺

d3.scaleLog() // 对数比例尺

d3.scaleQuantile() // 分位数比例尺

8. invert()与invertExtent()方法

上述的各种使用比例尺的例子都相当于一个正序的过程，从domain的数据集映射到range数据集中，那么有没有逆序的过程呢？D3中提供了invert()以及invertExtent()方法可以实现这个过程。

**let** scale = d3.scaleLinear().domain([1,5]).range([0,100])

scale.invert(50) // 输出:3

**let** scale2 = d3.scaleQuantize().domain([0,10]).range(['small', 'big'])

scale2.invertExtent('small') // 输出:[0,5]

不过，值得注意的是，这两种方法只针对连续性比例尺有效，即domain()域为连续性数据集的比例尺。那么非连续性的比例尺就没有invert()方法了吗？