

Ordinal Scales

(IDVW, Ch. 9)

Ordinal scales

```
var ordscale = d3.scaleBand()  
  .domain(["cold", "warm", "hot"])  
  .range([0, 600]);
```

```
> ordscale("cold");
```

0

```
> ordscale("warm");
```

200

```
> ordscale("hot");
```

400



d3.range() .length

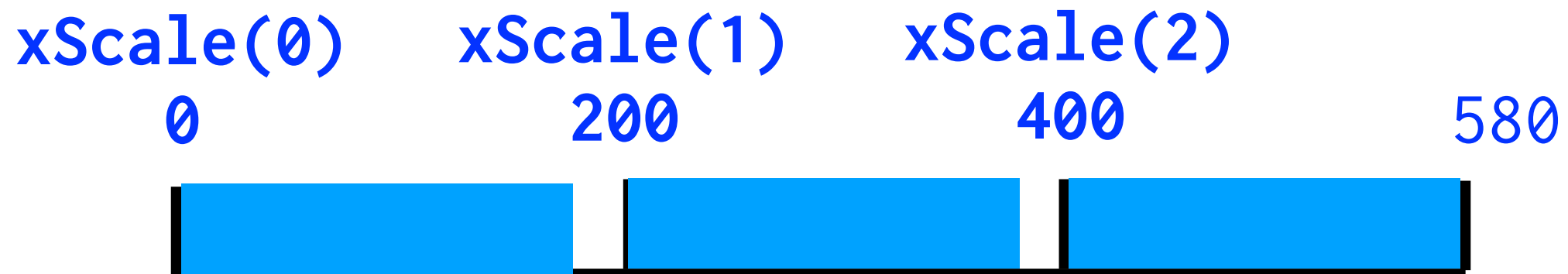
```
var ordscale = d3.scaleBand()  
  .domain([0, 1, 2, 3, 4])  
  .range([0, 600]);
```

d3.range(5) *returns* [0, 1, 2, 3, 4]

```
.domain(d3.range(dataset.length))
```

Ordinal scales

```
var xScale = d3.scaleBand()  
  .domain(d3.range(dataset.length))  
  .range([0, 580])  
  .paddingInner([.1]);
```



```
> xScale.bandwidth();  
180
```

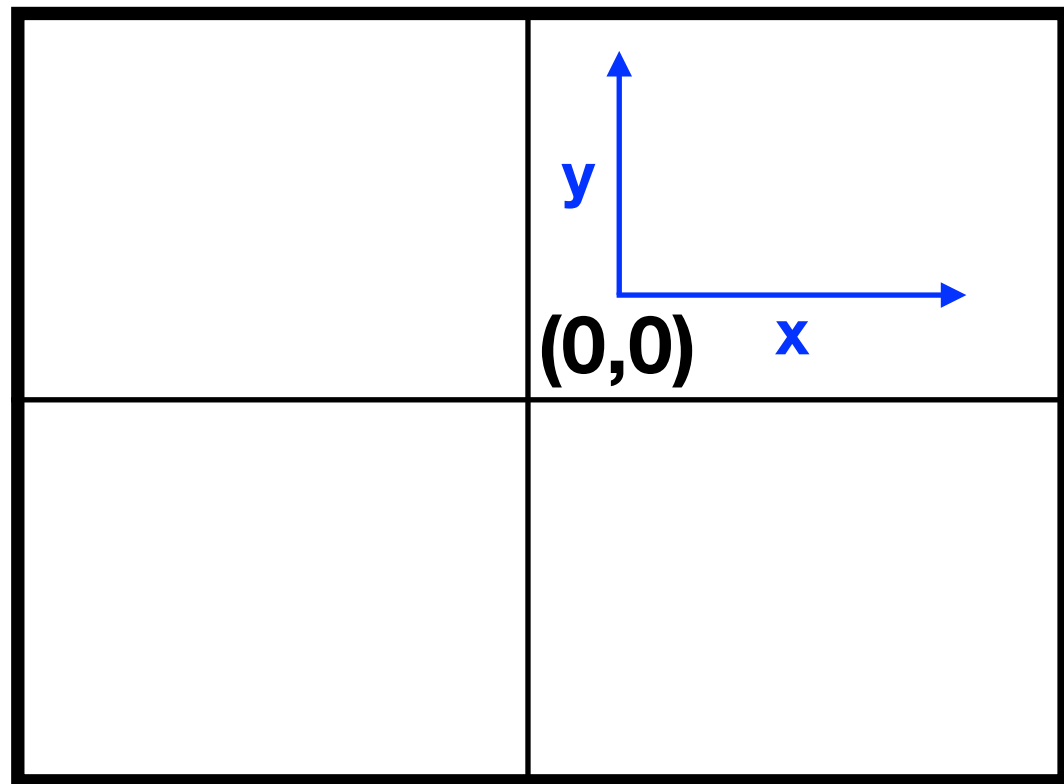
EDAV5_4_scaleBand.html

**ordinal scale
for x-axis**

Linear Scales

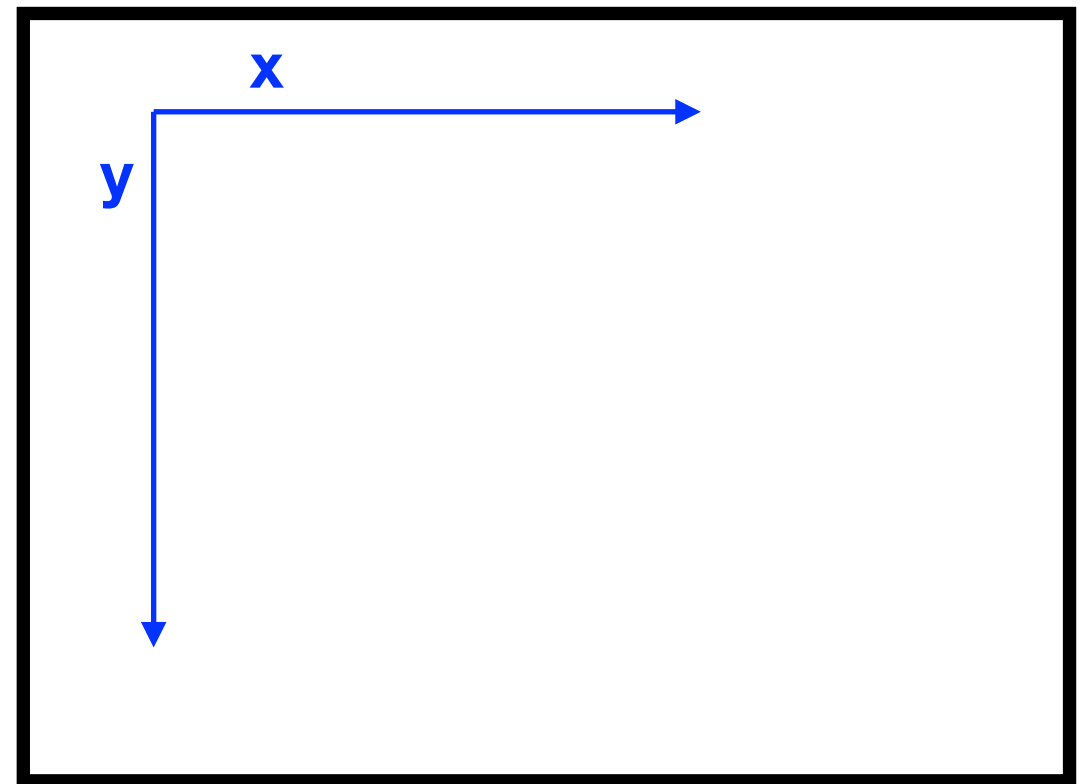
(IDVW, Ch. 6)

Cartesian Coordinates



SVG

$(0,0)$



X

Dealing with negative values

```
d3.scaleLinear()  
  .domain([-100, 100])  
  .range([0, 500])
```


y

so far...

```
.attr("height", d => d)
```

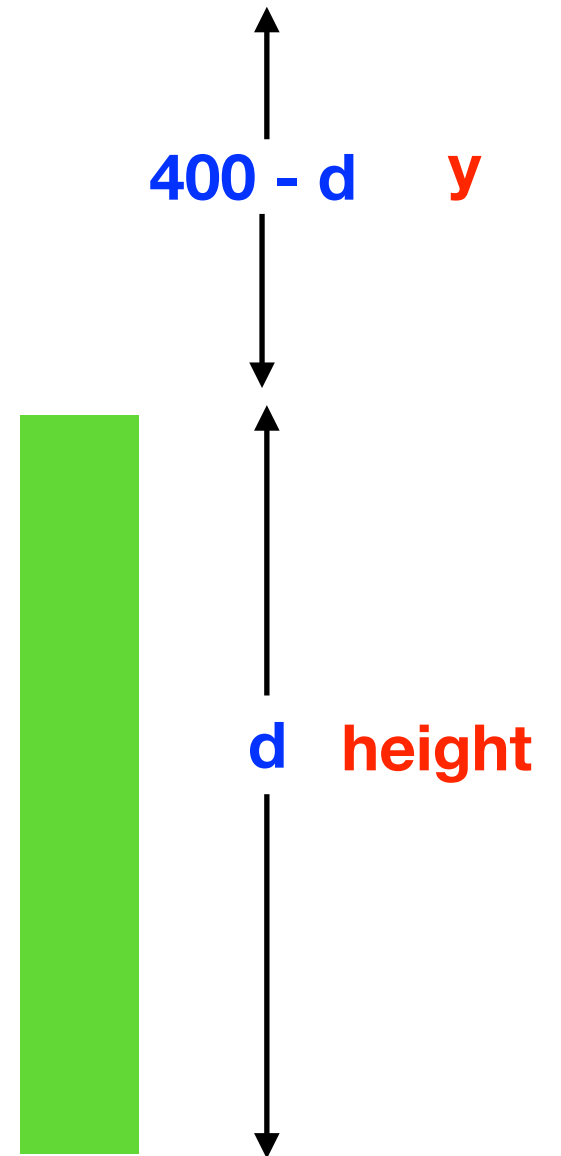
```
.attr("y", d => 400 - d)
```

1. the data (d) is the bar height

2. y is the gap on top

3. $y + \text{bar height} = \text{svg height}$

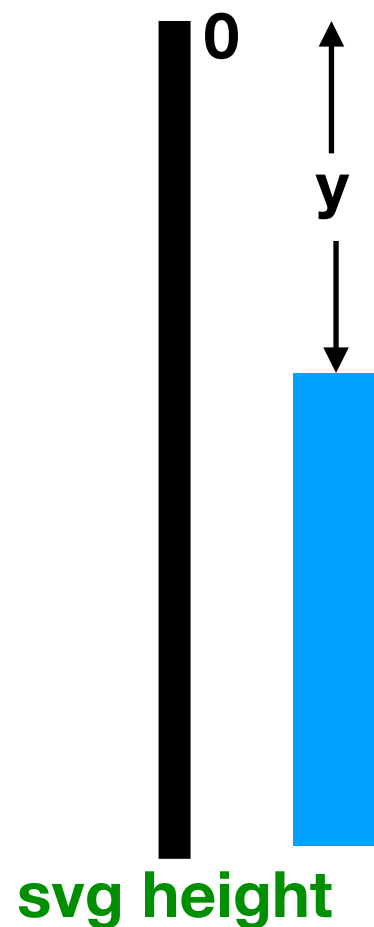
4. conceptual leap: y starts at the top



Scales (one approach)

```
var yScale = d3.scaleLinear()  
  .domain([0, datamax])  
  .range([0, svgheight]);
```

range



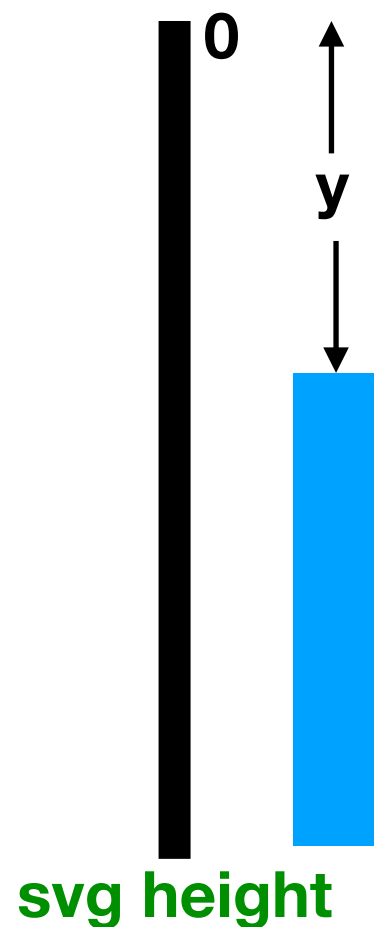
```
.attr("y", d => h - yScale(d));
```

```
.attr("height", d => yScale(d));
```

Scales (approach generally used with axes)

```
var yScale = d3.scaleLinear()  
  .domain([0, datamax])  
  .range([svgheight, 0]);
```

range



```
.attr("y", d => yScale(d));
```

```
.attr("height", d => h - yScale(d));
```

EDAV5_5_scaleLinear.html

**linear scale
for y-axis**