D3.js

UC Berkeley Graduate School of Journalism

Review

Review — DataTypes

`324230293420`	**Number**
`"Hello World"`	**String**
`false`	**Boolean**
`[3, 4, 3, 2, 1, 0]`	**Array**
`{ hair:"black" }`	**Object Literal**
`function(){}`	**Function**
`d3.select("#someid")`	**d3 object**

Review — Calling Functions

This is calling a function:

```
somefunction();
```

This is also calling a function, sending data:

```
somefunction("hey there");
```

Review — SVG transform

```
<g transform="">
</g>

<g transform="">
</g>

<g transform="">
</g>
```

Review — SVG transform

```
<g transform="translate(30, 50)">
</g>

<g transform="">
</g>

<g transform="">
</g>
```

Review — SVG transform

```
<g transform="translate(30, 50)">
</g>
```

```
<g transform="rotate(-40, 10, 10)">
</g>
```

```
<g transform="">
</g>
```

Review — SVG transform

```
<g transform="translate(30, 50)">
</g>

<g transform="rotate(-40, 10, 10)">
</g>

<g transform="scale(1.5)">
</g>
```

Review — SVG transform

or combine them

```
<g transform="translate(30, 50)rotate(-40, 10, 10)scale(1.5)">
</g>
```

D3

d3 — Selecting elements on the webpage

```
<div id="something">
</div>
```

d3 — Selecting elements on the webpage

```
<div id="something">
</div>
```

d3

d3 — Selecting elements on the webpage

```
<div id="something">
</div>
```

```
d3.select("#something")
```

d3 — Selecting elements on the webpage

```
<div id="something">
    <p></p>
</div>
```

```
d3.select("#something")
    .append("p")
```

d3 — Selecting elements on the webpage

```
<div id="something">
    <p>Hello World</p>
</div>
```

```
d3.select("#something")
   .append("p")
   .text("Hello World")
```

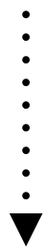
D3 Select

always start with d3 variable

```
d3.select("#title")
```

D3 Select

select is the first command to select the element



```
d3.select("#title")
```

D3 Select

a string which contains the CSS selector of the element

```
d3.select("#title")
```

D3 Select

dot means it's a class

```
d3.select(".headlines")
```

This will only return the first element
with **`class="headline"`**

D3 Recap

selects id="title"

```
.select("#title")
```

selects everything with class="headlines"

```
.selectAll(".headlines")
```

appends an element, like a div

```
.append("div")
```

places text within currently selected element

```
.text("Here is text")
```

D3 Recap

sets an attribute on the currently selected tag

```
.attr("width", 300)
```

set some CSS styles on the currently selected tag

```
.style({"fill":"red"})
```

just like text, but renders html

```
.html("<p>hey</p>")
```

adds or removes a class attribute

```
.classed("headlines", false)
```

Using Data with D3

D3 wants an Array of data

this is an array of numbers

```
[39, 239, 12, 32, 42, 24]
```

or an array of objects

```
[
  {name:"Apples", value:39},
  {name:"Orange", value:23},
  {name:"Pears",  value:30},
  {name:"Bananas",value:40}
]
```

Data in D3

```
var svg = d3.select("body")
          .append("svg")
          .attr("width", 400)
          .attr("height", 300);
```

this variable now has the SVG

Data in D3

```
svg.selectAll(".bubbles")
```

there is NO .bubbles

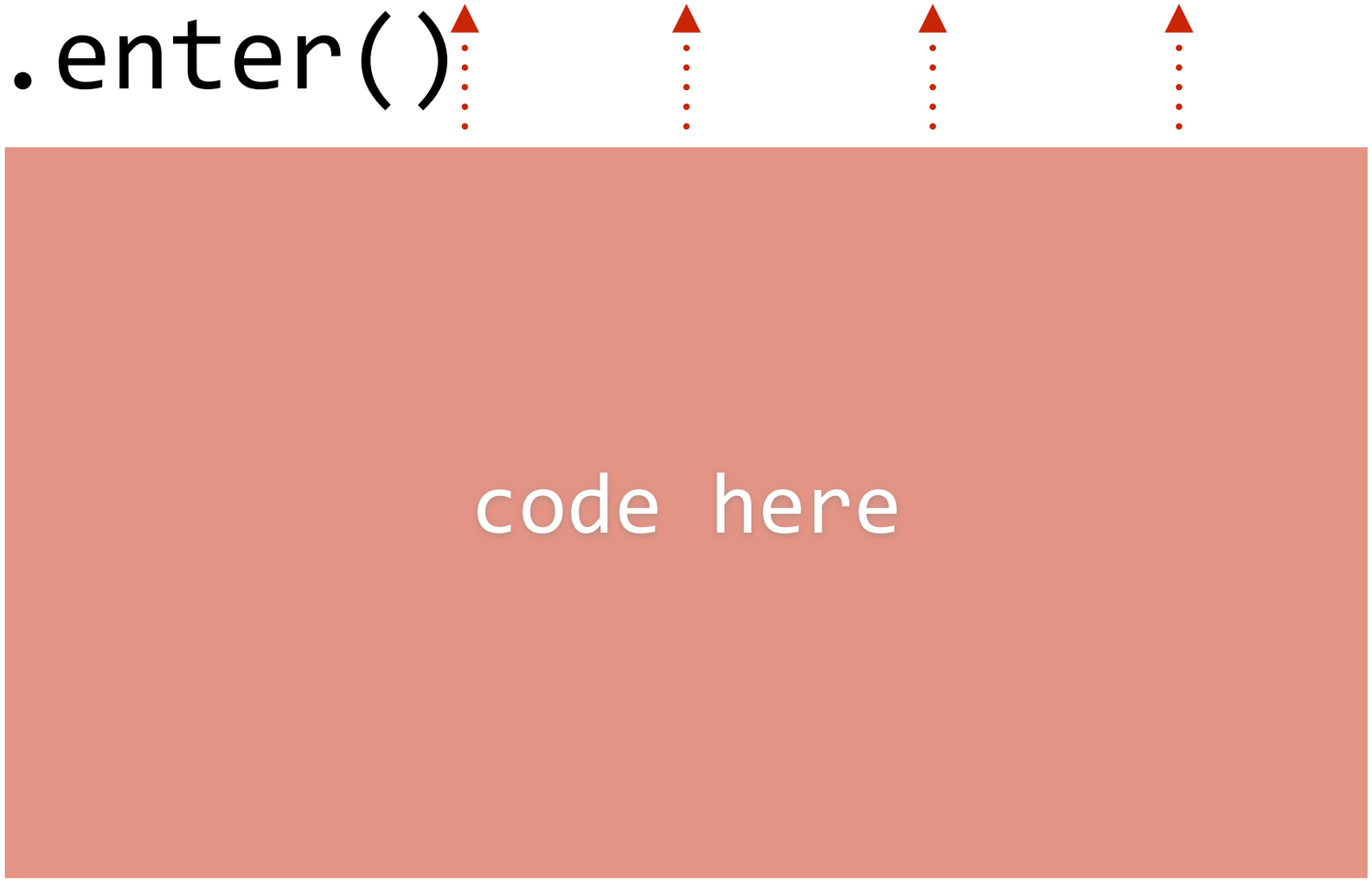
Data in D3

```
var data = [32,23,24,45];

svg.selectAll(".bubbles")
    .data(data)
    .enter()
```

◀ Magic!

Data in D3

```
svg.selectAll(".bubbles")
    .data([32, 43, 23, 54])
    .enter()
```



Data in D3

```
svg.selectAll(".bubbles")
    .data([32, 43, 23, 54])
    .enter()
    .append("circle")
```

Data in D3

```
svg.selectAll(".bubbles")
    .data([32, 43, 23, 54])
    .enter()
    .append("circle")
    .attr("cy", 100)
```

Data in D3

```
svg.selectAll(".bubbles")
    .data([32, 43, 23, 54])
    .enter()
    .append("circle")
.attr("cy", 100)
.attr("cx", function(d, i){
    return i * 50;
 })
```

Data in D3

```
svg.selectAll(".bubbles")
    .data([32, 43, 23, 54])
    .enter()
    .append("circle")
    .attr("cy", 100)
    .attr("cx", function(d, i){
        return i * 50;
    })
    .attr("r", function(d, i){
        return d;
    })
```

Data in D3

```
.attr("r", function(d, i){ return d; })
```

d is the value of the data array during this iteration of the loop.

i is the index (how many times) this loop has run.

return is what you want to return

Maps

Mapping Data

```
var myArray = [
    {name: "Bill", age: "45"},
    {name: "Joe", age: "37"},
    {name: "Jane", age: "25"},
    {name: "Jill", age: "9"}
]
```

```
myArray.map()
```

Mapping Data

```
var myArray = [
    {name: "Bill", age: 45},
    {name: "Joe", age: 37},
    {name: "Jane", age: 25},
    {name: "Jill", age: 9}
]
```

```
myArray.map(function(d){})
```

Mapping Data

```
var myArray = [
    {name: "Bill", age: 45},
    {name: "Joe", age: 37},
    {name: "Jane", age: 25},
    {name: "Jill", age: 9}
        ]

var newArray = myArray
    .map(function(d){ return d.name;})

    ["Bill","Joe","Jane", "Jill"]
```

Mapping Data

```
var myArray = [
    {name: "Bill", age: 45},
    {name: "Joe", age: 37},
    {name: "Jane", age: 25},
    {name: "Jill", age: 9}
          ]

var newArray = myArray
    .map(function(d){ return d.age;})
```

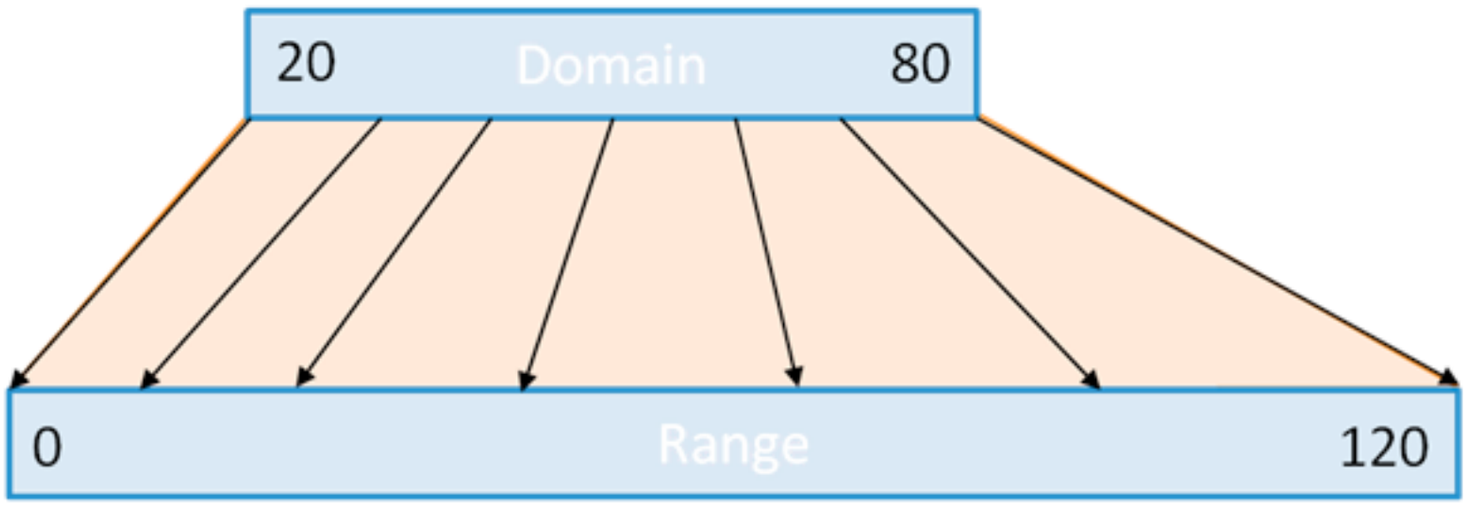
```
[45, 37, 25, 9]
```

Scales

Scales

domain - The span of your data set.

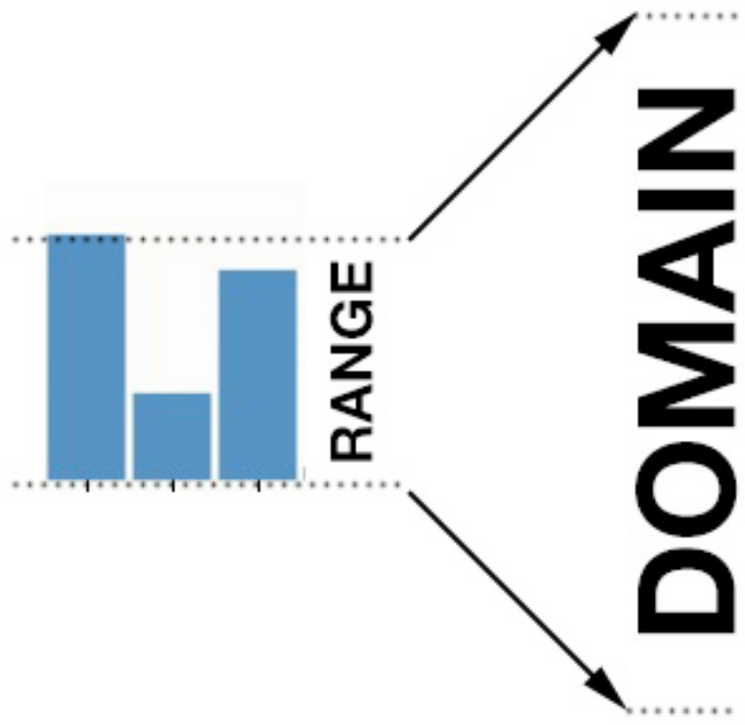
range - The range of the chart you want to stuff it into



Scales

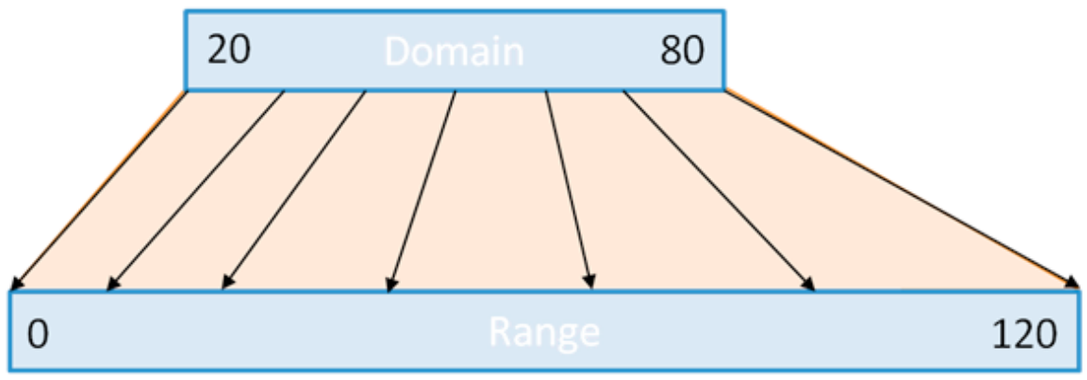
domain - The span of your data set.

range - The range of the chart you want to stuff it into



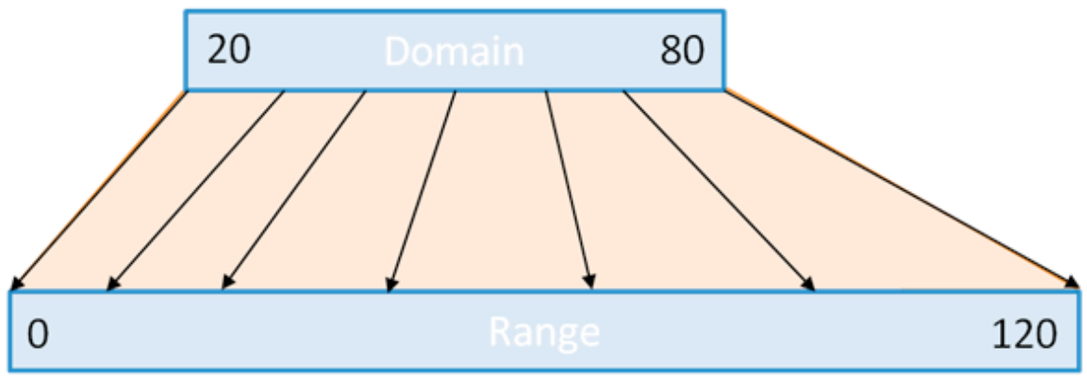
Linear Scales

```
var y = d3.scaleLinear()
          .range([0, 100])
          .domain([0, 5])

y(1)  //returns 20
y(4)  //returns 80
y(0)  //returns 0
```



Linear Scales

```
var y = d3.scaleLinear()
          .range([0, 100])
          .domain([350, 100000])

y(5410)  //returns 5.077
y(20343) //returns 20.063
y(100000)//returns 100
```



Ordinal (Band) Scales

```
var x = d3.scaleBand()
    .range([0, 100])
    .domain(["apple", "orange", "pear"])
    .padding(0.1);
```

```
x("apple")  //returns 0
x("orange") //returns 33
x("pear")   //returns 66
```