Scales

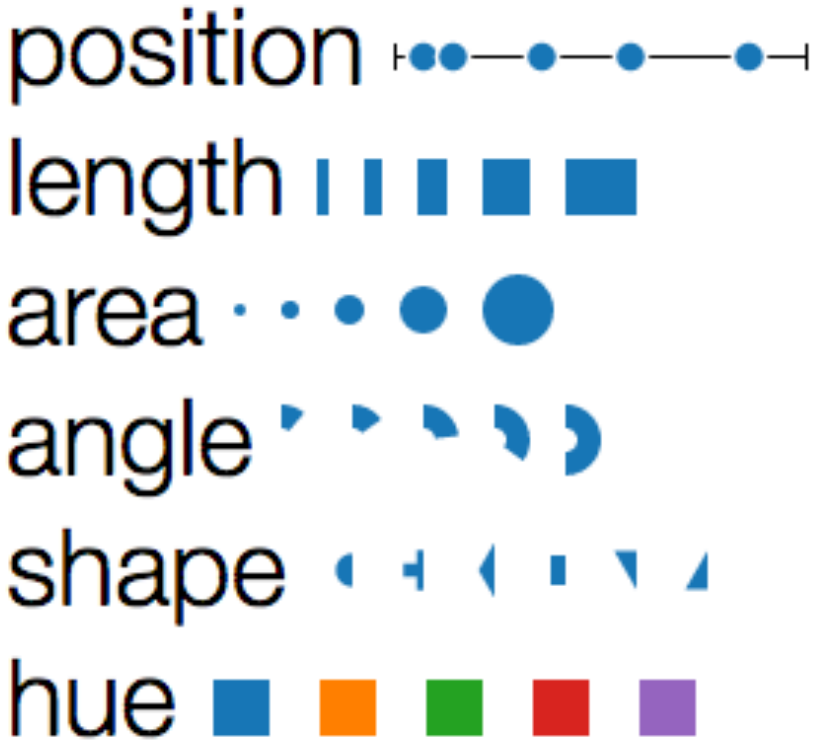
VISUALIZATION ON THE WEB

Visualization and Data Graphics

Data Types

- Categorical
- Ordinal
- Quantitative

Visual Variables



Visual Variables -> Documents

- Datum -> Element
  - Associate a graphical mark to each data point
- Data Attribute -> Element Attribute
  - Adjust properties of mark to encode properties of datum

Datum -> Element

```
// perform join and add missing
elements
var circles =
svg.selectAll("circle")
   .data(data)
   .enter().append("circle");
```

Data Attributes -> Element Attributes

```
// Update previous and new
elements at once
svg.selectAll("circle")
  .attr("cx", 10)
  .attr("cy", 25)
  .attr("r", 2.5);
```

An example

```
join.enter()
    .append("rect")
    .attr("x", function(d,i){
        return i*barw;
    })
    .attr("y",function(d){
        return height - d*4;
    })
    .attr("width", barw)
    .attr("height",function(d){
        return d*4;
    });
```

Data attributes (i.e.value and position) are mapped to element attributes (i.e. [x,y] and height)

SCALES FUNCTION

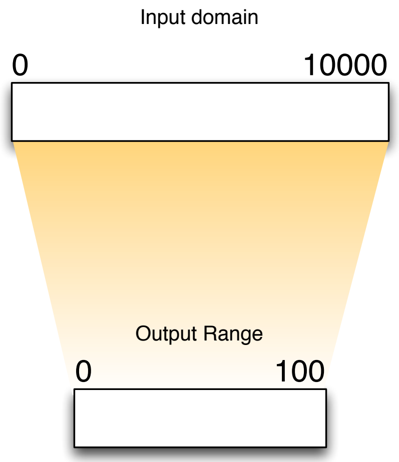
Scales

- Data values do not correspond to pixel coordinates
- We need to map data values to new values to meet visualization constraints
- Scales are **functions** that map from an input domain to an output range
- More details available at D3.js documentation: https://github.com/mbostock/d3/wiki/Quantitative-Scales

Manual Mapping

1. For input domain
   1. Select the largest number in original interval (10000)
   2. Select the smallest number in original interval (0)
   3. Select the difference of the two values (10000)
2. For output range
   1. Select the largest number in the new interval (100)
   2. Select the minimum number in the new interval (0)
   3. Select the difference of the two values (100)
3. Compute the ratio of the two intervals'range (10000/100 = 100)

- This is an example of a linear scaling
  - y = mx +b, where b=0 and m=1/100
  - 100 units in the original interval correspond to 1 unit in the destination interval



An example – an alternative solution

```
join.enter()
    .append("rect")
    .attr("x", function(d,i){
        return i*barw;
    })
    .attr("y",function(d){
        return height - d*4;
    })
    .attr("width", barw)
    .attr("height",function(d){
        return d*4;
    });
```

```
join.enter()
    .append("rect")
    .attr("x", function(d,i){
        return x(i);
    })
    .attr("y",function(d){
        return y(d);
    })
    .attr("width", barw)
    .attr("height",function(d){
        return h(d);
    });
function x(d){return m*d + b};
function y(d){return m'*d + b'};
function h(d){return m''*d + b''};
```

D3.js Scales generator

- D3 provides several scale types
  - Quantitative
    - Continuous
      - Identity
      - Linear (y=mx+b)
      - Power (y=mx^k+b)
      - Log (y=m log(x) + b)
    - Discrete
      - Quantize
      - Quantile
      - Threshold
  - Ordinal
  - Time

Creating a scale

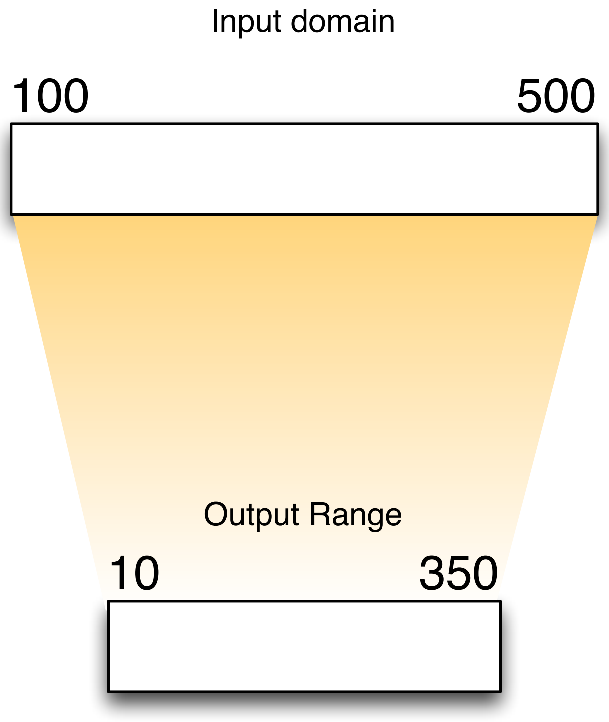
```
var scale = d3.scale.linear();
```

- Default scale uses
  - Domain is [0,1]
  - Range is [0,1]
  - Function is Identity
  - `scale(2.5); //returns 2.5`

Creating a scale – setting domain and range

```
var scale = d3.scale.linear()
   .domain([100,500]);
   .range([10,350]);
```

- Default scale uses
  - `scale(100); //returns 10`
  - `scale(300); //returns 180`
  - `scale(500); //returns 350`



Quantitative power scale – circle radius

Previous example

```
g.append("circle")
    .attr("fill","pink")
    .attr("stroke","red")
    .attr("r",function(d){
        return Math.sqrt(d*100);
    })
```

Refined solution

```
var r = d3.scale.sqrt()
    .domain([0,20])
    .range([0,30];

g.append("circle")
    .attr("fill","pink")
    .attr("stroke","red")
    .attr("r",function(d){
        return r(d);
    })
```

Utility functions: d3.min, d3.max, d3.extent

- To determine the domain and range interval we should know min and max of the two intervals
- D3.js provides utility functions to access such values
  - `d3.min(array[,accessor])`
  - `d3.max(array[,accessor])`
  - `d3.extent(array[,accessor])`

Utility Functions: examples

```
d3.min([10,30,40,70,100]) //returns 10
d3.max([10,30,40,70,100]) //returns 100
d3.extent([10,30,40,70,100]) //returns
[10,100]
```

Bar Chart

Exercise

Scatterplot Chart

Exercise

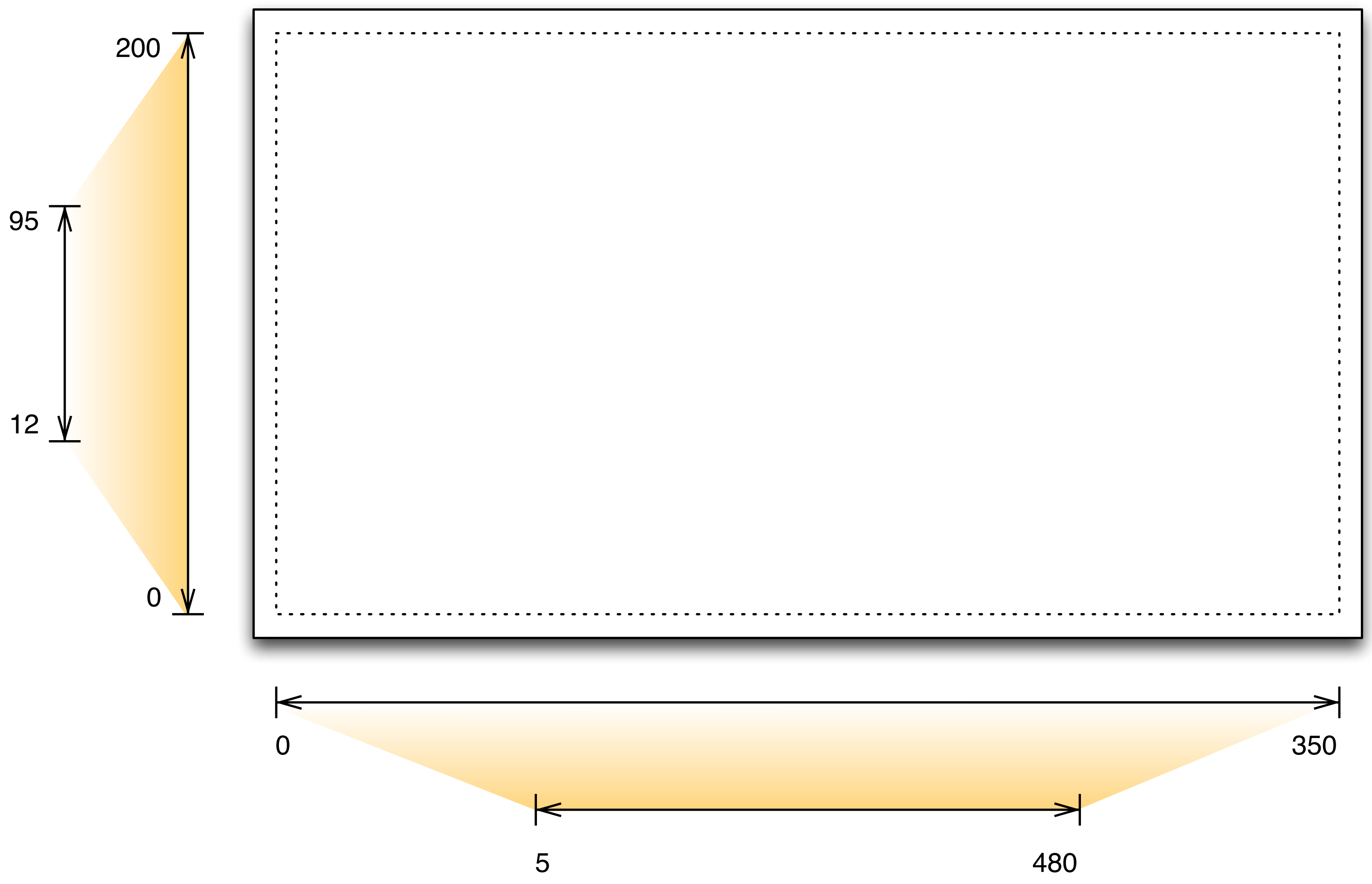
Utility Functions: examples

```
var dataset = [
         [ 5,      20 ],
         [ 480,    90 ],
         [ 250,    50 ],
         [ 100,    33 ],
         [ 330,    95 ],
         [ 410,    12 ],
         [ 475,    44 ],
         [ 25,     67 ],
         [ 85,     21 ],
         [ 220,    88 ]
     ];

d3.min(dataset, function(d){return d[0]}) //returns 5
d3.max(dataset, function(d){return d[0]}) //returns 480
d3.extent(dataset, function(d){return d[0]}) //returns [5,480]

d3.min(dataset, function(d){return d[1]}) //returns 12
d3.max(dataset, function(d){return d[1]}) //returns 95
d3.extent(dataset, function(d){return d[1]}) //returns [12,95]
```

Linear scales for (x,y) coordinates



200
95
12
0
0
350
5
480