

# What is and Why D3?



- Data-Driven Documents (D3)
  - Mike Bostock
- Let's see some examples!
  - <http://d3js.org>
  - <http://bl.ocks.org/mbostock>

# What is D3 Doing?



- HTML elements => Webpage
- Manipulate HTML elements dynamically with Javascript based on input data to create visualizations
- Style with CSS

# Reference Tutorial



- Let's Make a Bar Chart by Mike Bostock
- Use examples in this tutorial
- Part 2: <http://bost.ocks.org/mike/bar/2/>
- Part 3: <http://bost.ocks.org/mike/bar/3/>

```

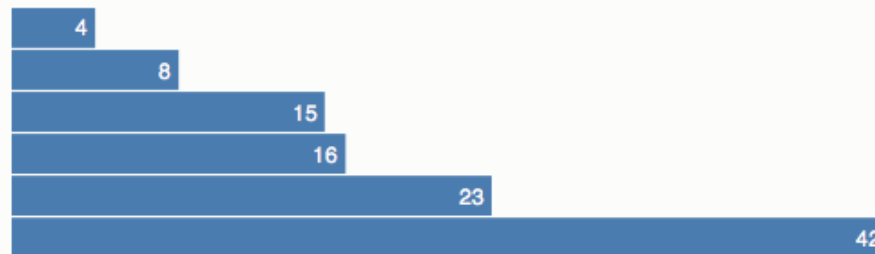
<!DOCTYPE html>
<meta charset="utf-8">
<style>

.chart rect {
  fill: steelblue;
}

.chart text {
  fill: white;
  font: 10px sans-serif;
  text-anchor: end;
}

</style>
<svg class="chart"></svg>
<script src="http://d3js.org/d3.v3.min.js"></script>
<script>

```



```

var width = 420,
    barHeight = 20;

var x = d3.scale.linear()
    .range([0, width]);

var chart = d3.select(".chart")
    .attr("width", width);

d3.tsv("data.tsv", type, function(error, data) {
  x.domain([0, d3.max(data, function(d) { return d.value; })]);

  chart.attr("height", barHeight * data.length);

  var bar = chart.selectAll("g")
    .data(data)
    .enter().append("g")
    .attr("transform", function(d, i) { return "translate(0," + i * barHeight + ")"; });

  bar.append("rect")
    .attr("width", function(d) { return x(d.value); })
    .attr("height", barHeight - 1);

  bar.append("text")
    .attr("x", function(d) { return x(d.value) - 3; })
    .attr("y", barHeight / 2)
    .attr("dy", ".35em")
    .text(function(d) { return d.value; });
});

function type(d) {
  d.value = +d.value; // coerce to number
  return d;
}

</script>

```

name	value
Locke	4
Reyes	8
Ford	15
Jarrah	16
Shephard	23
Kwon	42

```
<script>
```

```
var width = 420,  
    barHeight = 20;
```

```
var x = d3.scale.linear()  
    .range([0, width]);
```

```
var chart = d3.select(".chart")  
    .attr("width", width);
```

```
d3.tsv("data.tsv", type, function(error, data) {  
    x.domain([0, d3.max(data, function(d) { return d.value; })]);
```

```
    chart.attr("height", barHeight * data.length);
```

```
    var bar = chart.selectAll("g")  
        .data(data)  
        .enter().append("g")  
        .attr("transform", function(d, i) { return "translate(0," + i * barHeight + ")"; });
```

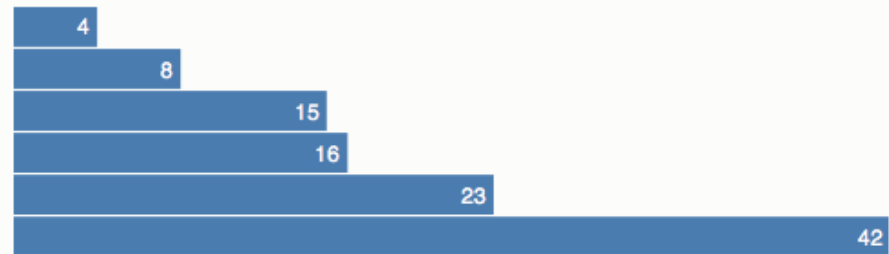
```
    bar.append("rect")  
        .attr("width", function(d) { return x(d.value); })  
        .attr("height", barHeight - 1);
```

```
    bar.append("text")  
        .attr("x", function(d) { return x(d.value) - 3; })  
        .attr("y", barHeight / 2)  
        .attr("dy", ".35em")  
        .text(function(d) { return d.value; });
```

```
});
```

```
function type(d) {  
    d.value = +d.value; // coerce to number  
    return d;  
}
```

```
</script>
```



name	value
Locke	4
Reyes	8
Ford	15
Jarrah	16
Shephard	23
Kwon	42

# Selections



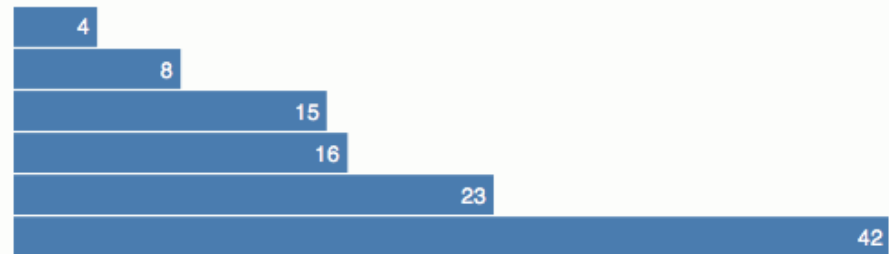
- Acquire html element to manipulate
- CSS selector (e.g. “.class” and “#id”)
- `d3.select(“.chart”)`
  - Select first element of class “chart”
  - `d3.selectAll(selector)` - select all elements
  - Can also select a specific node
    - e.g. `d3.select(this)`
- <https://github.com/mbostock/d3/wiki/Selections>

```
<script>
```

```
var width = 420,  
    barHeight = 20;
```

```
var x = d3.scale.linear()  
    .range([0, width]);
```

```
var chart = d3.select(".chart")  
    .attr("width", width);
```



```
d3.tsv("data.tsv", type, function(error, data) {  
    x.domain([0, d3.max(data, function(d) { return d.value; })]);
```

```
    chart.attr("height", barHeight * data.length);
```

```
    var bar = chart.selectAll("g")  
        .data(data)  
        .enter().append("g")  
        .attr("transform", function(d, i) { return "translate(0," + i * barHeight + ")"; });
```

```
    bar.append("rect")  
        .attr("width", function(d) { return x(d.value); })  
        .attr("height", barHeight - 1);
```

```
    bar.append("text")  
        .attr("x", function(d) { return x(d.value) - 3; })  
        .attr("y", barHeight / 2)  
        .attr("dy", ".35em")  
        .text(function(d) { return d.value; });
```

```
});
```

```
function type(d) {  
    d.value = +d.value; // coerce to number  
    return d;  
}
```

```
</script>
```

name	value
Locke	4
Reyes	8
Ford	15
Jarrah	16
Shephard	23
Kwon	42

# Loading Data



- Tab-separated values (TSV)
- `d3.tsv(url[, accessor][, callback])`
  - See also `d3.csv()`, `d3.json()`
  - Asynchronous
- Callback: `function(error, data)`
  - After data are loaded
- <https://github.com/mbostock/d3/wiki/Requests>

name	value
Locke	4
Reyes	8
Ford	15
Jarrah	16
Shephard	23
Kwon	42



```
var data = [  
  {name: "Locke", value: 4},  
  {name: "Reyes", value: 8},  
  {name: "Ford", value: 15},  
  {name: "Jarrah", value: 16},  
  {name: "Shephard", value: 23},  
  {name: "Kwon", value: 42}  
];
```



```

<script>
var width = 420,
    barHeight = 20;

var x = d3.scale.linear()
    .range([0, width]);

var chart = d3.select(".chart")
    .attr("width", width);

d3.tsv("data.tsv", type, function(error, data) {
    x.domain([0, d3.max(data, function(d) { return d.value; })]);

    chart.attr("height", barHeight * data.length);

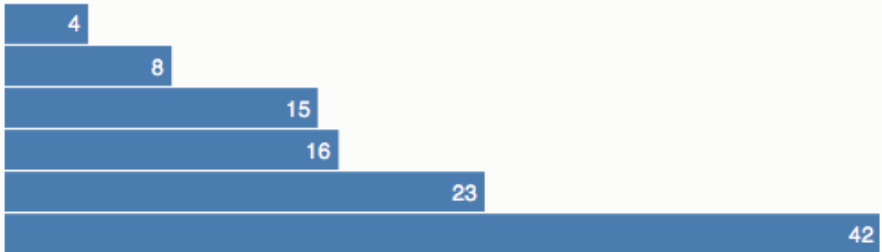
    var bar = chart.selectAll("g")
        .data(data)
        .enter().append("g")
        .attr("transform", function(d, i) { return "translate(0," + i * barHeight + ")"; });

    bar.append("rect")
        .attr("width", function(d) { return x(d.value); })
        .attr("height", barHeight - 1);

    bar.append("text")
        .attr("x", function(d) { return x(d.value) - 3; })
        .attr("y", barHeight / 2)
        .attr("dy", ".35em")
        .text(function(d) { return d.value; });
});

function type(d) {
    d.value = +d.value; // coerce to number
    return d;
}
</script>

```



name	value
Locke	4
Reyes	8
Ford	15
Jarrah	16
Shephard	23
Kwon	42

# Local Web Server



- Needed if you were to load data locally with d3.xhr (e.g. d3.tsv)
- > `python -m SimpleHTTPServer 8888`
  - Run at folder with vis source code
- Access <http://localhost:8888/>

```

<script>
var width = 420,
    barHeight = 20;

var x = d3.scale.linear()
    .range([0, width]);

var chart = d3.select(".chart")
    .attr("width", width);

d3.tsv("data.tsv", type, function(error, data) {
    x.domain([0, d3.max(data, function(d) { return d.value; })]);

    chart.attr("height", barHeight * data.length);

    var bar = chart.selectAll("g")
        .data(data)
        .enter().append("g")
        .attr("transform", function(d, i) { return "translate(0," + i * barHeight + ")"; });

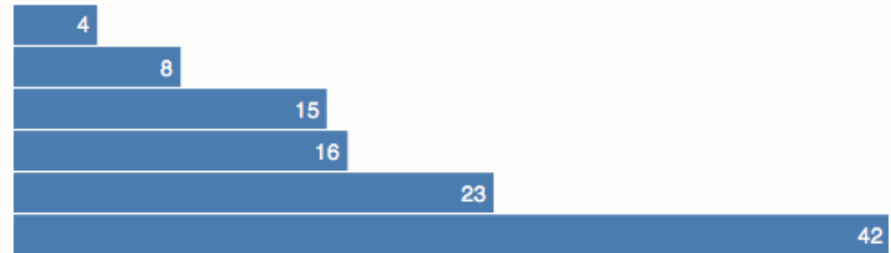
    bar.append("rect")
        .attr("width", function(d) { return x(d.value); })
        .attr("height", barHeight - 1);

    bar.append("text")
        .attr("x", function(d) { return x(d.value) - 3; })
        .attr("y", barHeight / 2)
        .attr("dy", ".35em")
        .text(function(d) { return d.value; });

    });

function type(d) {
    d.value = +d.value; // coerce to number
    return d;
}
</script>

```



name	value
Locke	4
Reyes	8
Ford	15
Jarrah	16
Shephard	23
Kwon	42

# Linear Scale



- Dynamically rescale variable to fit on screen



Data	1	2	3	4	5	<b>Domain</b>
Screen	0	?	?	?	420	<b>Range</b>

A horizontal line with arrows at both ends spans from the '1' column to the '5' column, indicating the mapping of the domain to the range.

- `var x = d3.scale.linear()  
 .domain([1,5]).range([0,420]);`
- `x(3); // 210`
- Find domain from data: `d3.min()`, `d3.max()`
- <https://github.com/mbostock/d3/wiki/scales>

# Ordinal Scales



- `var x = d3.scale.ordinal()  
 .domain(["a", "b", "c"]).range([0, 210, 420]);`
- `x("b"); // 210`
- `var color = d3.scale.category10()  
 .domain(["a", "b", "c"]);` 
- `color("a"); // #1f77b4` 

```

<script>

var width = 420,
    barHeight = 20;

var x = d3.scale.linear()
    .range([0, width]);

var chart = d3.select(".chart")
    .attr("width", width);

d3.tsv("data.tsv", type, function(error, data) {
    x.domain([0, d3.max(data, function(d) { return d.value; })]);
    chart.attr("height", barHeight * data.length);

    var bar = chart.selectAll("g")
        .data(data)
        .enter().append("g")
        .attr("transform", function(d, i) { return "translate(0," + i * barHeight + ")"; });

    bar.append("rect")
        .attr("width", function(d) { return x(d.value); })
        .attr("height", barHeight - 1);

    bar.append("text")
        .attr("x", function(d) { return x(d.value) - 3; })
        .attr("y", barHeight / 2)
        .attr("dy", ".35em")
        .text(function(d) { return d.value; });

    });

function type(d) {
    d.value = +d.value; // coerce to number
    return d;
}

</script>

```

```

<svg class="chart" width="420" height="120">
</svg>

```



```

<script>

var width = 420,
    barHeight = 20;

var x = d3.scale.linear()
    .range([0, width]);

var chart = d3.select(".chart")
    .attr("width", width);

d3.tsv("data.tsv", type, function(error, data) {
    x.domain([0, d3.max(data, function(d) { return d.value; })]);

    chart.attr("height", barHeight * data.length);

    var bar = chart.selectAll("g")
        .data(data)
        .enter().append("g")
        .attr("transform", function(d, i) { return "translate(0," + i * barHeight + ")"; });

    bar.append("rect")
        .attr("width", function(d) { return x(d.value); })
        .attr("height", barHeight - 1);

    bar.append("text")
        .attr("x", function(d) { return x(d.value) - 3; })
        .attr("y", barHeight / 2)
        .attr("dy", ".35em")
        .text(function(d) { return d.value; });

});

function type(d) {
    d.value = +d.value; // coerce to number
    return d;
}

</script>

```

# Binding Data to Elements



- `d3.selectAll("g").data(data);`
- Attach the array items to SVG group elements `<g>`
- Question: where are they?



```

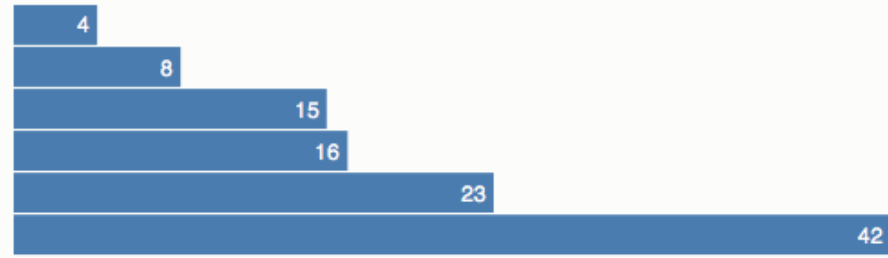
<!DOCTYPE html>
<meta charset="utf-8">
<style>

.chart rect {
  fill: steelblue;
}

.chart text {
  fill: white;
  font: 10px sans-serif;
  text-anchor: end;
}

</style>
<svg class="chart"></svg>
<script src="http://d3js.org/d3.v3.min.js"></script>
<script>

```



```

var width = 420,
    barHeight = 20;

var x = d3.scale.linear()
    .range([0, width]);

var chart = d3.select(".chart")
    .attr("width", width);

d3.tsv("data.tsv", type, function(error, data) {
  x.domain([0, d3.max(data, function(d) { return d.value; })]);

  chart.attr("height", barHeight * data.length);

  var bar = chart.selectAll("g")
    .data(data)
    .enter().append("g")
    .attr("transform", function(d, i) { return "translate(0," + i * barHeight + ")"; });

  bar.append("rect")
    .attr("width", function(d) { return x(d.value); })
    .attr("height", barHeight - 1);

  bar.append("text")
    .attr("x", function(d) { return x(d.value) - 3; })
    .attr("y", barHeight / 2)
    .attr("dy", ".35em")
    .text(function(d) { return d.value; });
});

function type(d) {
  d.value = +d.value; // coerce to number
  return d;
}

</script>

```

**They don't exist yet!**

```

<script>

var width = 420,
    barHeight = 20;

var x = d3.scale.linear()
    .range([0, width]);

var chart = d3.select(".chart")
    .attr("width", width);

d3.tsv("data.tsv", type, function(error, data) {
    x.domain([0, d3.max(data, function(d) { return d.value; })]);

    chart.attr("height", barHeight * data.length);

    var bar = chart.selectAll("g")
        .data(data)
        .enter().append("g")
        .attr("transform", function(d, i) { return "translate(0," + i * barHeight + ")"; });

    bar.append("rect")
        .attr("width", function(d) { return x(d.value); })
        .attr("height", barHeight - 1);

    bar.append("text")
        .attr("x", function(d) { return x(d.value) - 3; })
        .attr("y", barHeight / 2)
        .attr("dy", ".35em")
        .text(function(d) { return d.value; });

});

function type(d) {
    d.value = +d.value; // coerce to number
    return d;
}

</script>

```

# Enter()



- Select new elements from data
- `var group = chart.selectAll("g").data(data).enter();`
  - Enter group: array of elements that do not exist
- `group.append("g").attr("transform", ...);`

```
var data = [  
  {name: "Locke", value: 4},  
  {name: "Reyes", value: 8},  
  {name: "Ford", value: 15},  
  {name: "Jarrah", value: 16},  
  {name: "Shephard", value: 23},  
  {name: "Kwon", value: 42}  
];
```



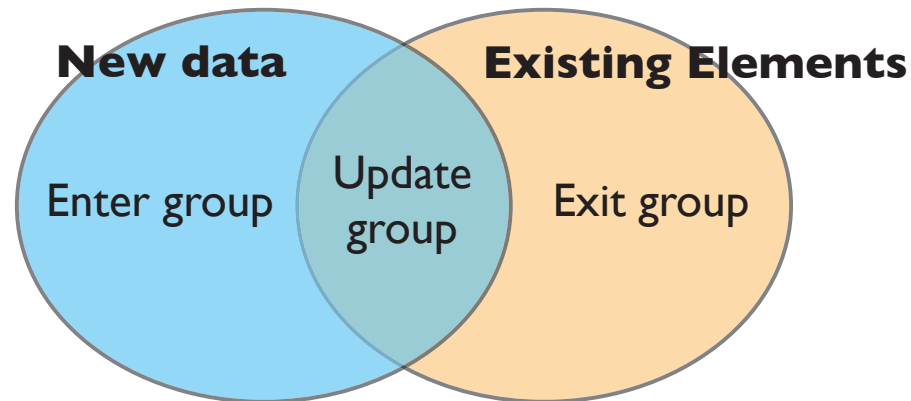
```
<svg class="chart" width="420" height="120">  
  <g transform="translate(0,0)">...</g>  
  <g transform="translate(0,20)">...</g>  
  <g transform="translate(0,40)">...</g>  
  <g transform="translate(0,60)">...</g>  
  <g transform="translate(0,80)">...</g>  
  <g transform="translate(0,100)">...</g>  
</svg>
```

What if we need to  
update the data?

# Enter, Update, Exit



- Three groups
  - **Enter** group - elements that don't exist yet
  - **Update** group - elements that exist
  - **Exit** group - elements that should be removed
- <http://bost.ocks.org/mike/join/>



# Enter, Update, Exit



- `var newData = [1,2,3,4,5];`
- If there exist 5 elements bounded to data: `[3,4,5,6,7]`
- `var group = d3.selectAll("g").data(newData);`
  - Update group `[3,4,5]`
- `group.enter()`
  - Enter group `[1,2]`
- `group.exit()`
  - Exit group `[6,7]`

# Enter, Update, Exit



- `chart.selectAll("g")  
 .data(data)  
 .enter().append("g")  
 .attr("transform", ...);`
- Notice the indention
  - 2 spaces when the selected group changed
  - 4 spaces if not



# Looping through Data



- `var bar = chart.selectAll("g").data(data).enter().append("g").attr("transform", function(d, i){ return "translate(0," + i*barHeight + ")");`
- Callback loop through each item in the array **data**
  - `d` is the item and `i` is the index

```
Index  var data = [  
0      {name: "Locke",    value: 4},  
1      {name: "Reyes",   value: 8},  
2      {name: "Ford",    value: 15},  
3      {name: "Jarrah",   value: 16},  
4      {name: "Shephard", value: 23},  
5      {name: "Kwon",    value: 42},  
6      ];
```



```
<svg class="chart" width="420" height="120">  
▶ <g transform="translate(0,0)">...</g>  
▶ <g transform="translate(0,20)">...</g>  
▶ <g transform="translate(0,40)">...</g>  
▶ <g transform="translate(0,60)">...</g>  
▶ <g transform="translate(0,80)">...</g>  
▶ <g transform="translate(0,100)">...</g>  
</svg>
```

```

<script>

var width = 420,
    barHeight = 20;

var x = d3.scale.linear()
    .range([0, width]);

var chart = d3.select(".chart")
    .attr("width", width);

d3.tsv("data.tsv", type, function(error, data) {
    x.domain([0, d3.max(data, function(d) { return d.value; })]);

    chart.attr("height", barHeight * data.length);

    var bar = chart.selectAll("g")
        .data(data)
        .enter().append("g")
        .attr("transform", function(d, i) { return "translate(0," + i * barHeight + ")"; });

    bar.append("rect")
        .attr("width", function(d) { return x(d.value); })
        .attr("height", barHeight - 1);

    bar.append("text")
        .attr("x", function(d) { return x(d.value) - 3; })
        .attr("y", barHeight / 2)
        .attr("dy", ".35em")
        .text(function(d) { return d.value; });

});

function type(d) {
    d.value = +d.value; // coerce to number
    return d;
}

</script>

```

# What is “bar?”



- **var bar** = chart.selectAll(“g”).data(data)  
 .enter().append(“g”)  
 .attr(“transform”, function(d, i){  
 return “translate(0,” + i\*barHeight + “)”;  
 });

# Adding Rects



- bar is a group of `<g>` elements with data!
- `bar.append("rect")`  
    `.attr("width", function(d){`  
        `return x(d.value);`  
    `})`  
    `.attr("height", barHeight);`
- <http://www.w3.org/TR/SVG/shapes.html>

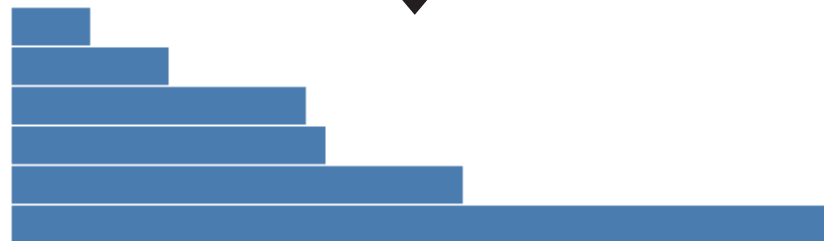
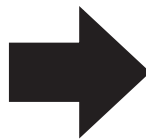
```
var data = [
  {name: "Locke",    value: 4},
  {name: "Reyes",   value: 8},
  {name: "Ford",    value: 15},
  {name: "Jarrah",  value: 16},
  {name: "Shephard", value: 23},
  {name: "Kwon",    value: 42}
];
```



```
▼ <svg class="chart" width="420" height="120">
  ▼ <g transform="translate(0,0)">
    <rect width="40" height="19"></rect>
  </g>
  ▼ <g transform="translate(0,20)">
    <rect width="80" height="19"></rect>
  </g>
  ▼ <g transform="translate(0,40)">
    <rect width="149.9999999999997" height="19"></rect>
  </g>
  ▼ <g transform="translate(0,60)">
    <rect width="160" height="19"></rect>
  </g>
  ▼ <g transform="translate(0,80)">
    <rect width="229.9999999999997" height="19"></rect>
  </g>
  ▼ <g transform="translate(0,100)">
    <rect width="420" height="19"></rect>
  </g>
</svg>
```



```
<style>
.chart rect {
  fill: steelblue;
}
.chart text {
  fill: white;
  font: 10px sans-serif;
  text-anchor: end;
}
</style>
```



How was the color chosen?

# Method Chaining



- The return object is the same as the caller object
- D3 extensively uses it
  - `group.attr(...);`  
`group.attr(...);`
  - `group.attr(...).attr(...);`
- Useful for setting attributes of the same object

```

<script>

var width = 420,
    barHeight = 20;

var x = d3.scale.linear()
    .range([0, width]);

var chart = d3.select(".chart")
    .attr("width", width);

d3.tsv("data.tsv", type, function(error, data) {
    x.domain([0, d3.max(data, function(d) { return d.value; })]);

    chart.attr("height", barHeight * data.length);

    var bar = chart.selectAll("g")
        .data(data)
        .enter().append("g")
        .attr("transform", function(d, i) { return "translate(0," + i * barHeight + ")"; });

    bar.append("rect")
        .attr("width", function(d) { return x(d.value); })
        .attr("height", barHeight - 1);

    bar.append("text")
        .attr("x", function(d) { return x(d.value) - 3; })
        .attr("y", barHeight / 2)
        .attr("dy", ".35em")
        .text(function(d) { return d.value; });

});

function type(d) {
    d.value = +d.value; // coerce to number
    return d;
}

</script>

```

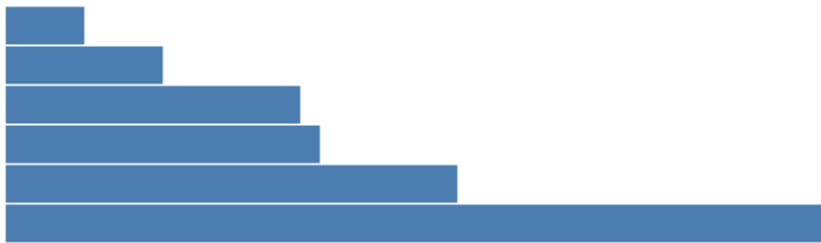


# Adding Text



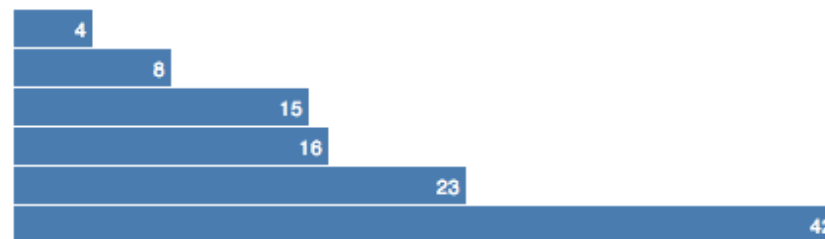
```
bar.append("text")
  .attr("x", function(d) { return x(d.value) - 3; })
  .attr("y", barHeight / 2)
  .attr("dy", ".35em")
  .text(function(d) { return d.value; });
```

<http://www.w3.org/TR/SVG/text.html>



```
.chart text {
  fill: white;
  font: 10px sans-serif;
  text-anchor: end;
}
```

```
var data = [
  {name: "Locke",    value: 4},
  {name: "Reyes",   value: 8},
  {name: "Ford",    value: 15},
  {name: "Jarrah",  value: 16},
  {name: "Shephard", value: 23},
  {name: "Kwon",    value: 42}
];
```





```

<!DOCTYPE html>
<meta charset="utf-8">
<style>

.chart rect {
  fill: steelblue;
}

.chart text {
  fill: white;
  font: 10px sans-serif;
  text-anchor: end;
}

</style>
<svg class="chart"></svg>
<script src="http://d3js.org/d3.v3.min.js"></script>
<script>

var width = 420,
    barHeight = 20;

var x = d3.scale.linear()
    .range([0, width]);

var chart = d3.select(".chart")
    .attr("width", width);

d3.tsv("data.tsv", type, function(error, data) {
  x.domain([0, d3.max(data, function(d) { return d.value; })]);

  chart.attr("height", barHeight * data.length);

  var bar = chart.selectAll("g")
    .data(data)
    .enter().append("g")
    .attr("transform", function(d, i) { return "translate(0," + i * barHeight + ")"; });

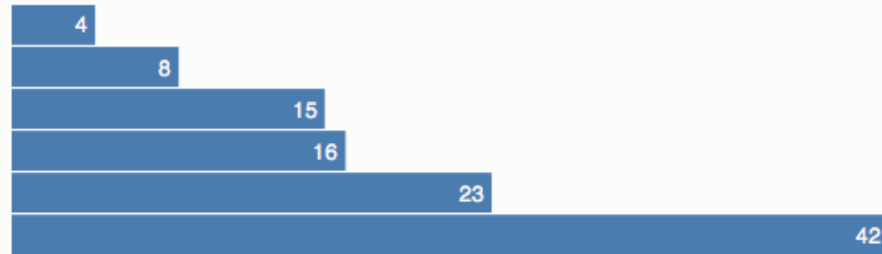
  bar.append("rect")
    .attr("width", function(d) { return x(d.value); })
    .attr("height", barHeight - 1);

  bar.append("text")
    .attr("x", function(d) { return x(d.value) - 3; })
    .attr("y", barHeight / 2)
    .attr("dy", ".35em")
    .text(function(d) { return d.value; });
});

function type(d) {
  d.value = +d.value; // coerce to number
  return d;
}

</script>

```

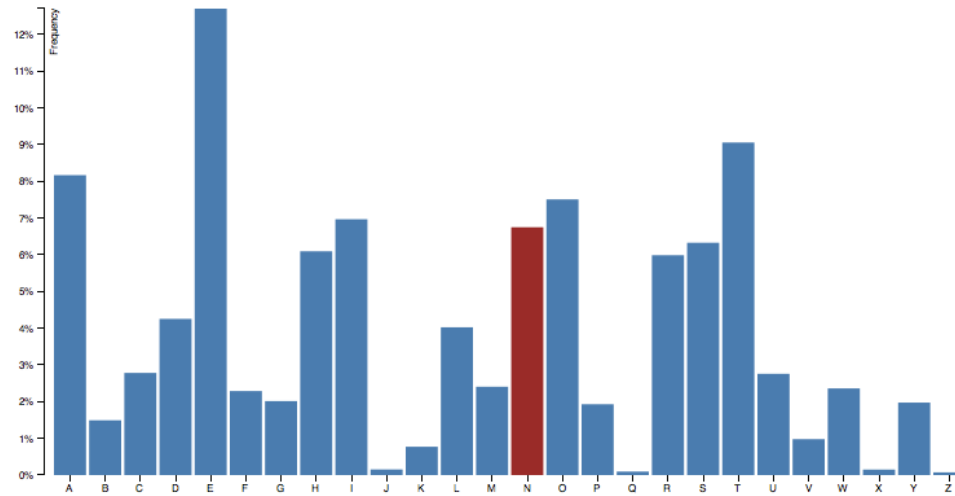


name	value
Locke	4
Reyes	8
Ford	15
Jarrah	16
Shephard	23
Kwon	42

# Another Example

# Relative frequency of English letters

letter	frequency
A	.08167
B	.01492
C	.02782
D	.04253
E	.12702
F	.02288
G	.02015
H	.06094
I	.06966
J	.00153
K	.00772
L	.04025
M	.02406
N	.06749
O	.07507
P	.01929
Q	.00095
R	.05987
S	.06327
T	.09056
U	.02758
V	.00978
W	.02360
X	.00150
Y	.01974
Z	.00074



```
<!DOCTYPE html>
<meta charset="utf-8">
<style>

.bar {
  fill: steelblue;
}

.bar:hover {
  fill: brown;
}

.axis {
  font: 10px sans-serif;
}

.axis path,
.axis line {
  fill: none;
  stroke: #000;
  shape-rendering: crispEdges;
}

.x.axis path {
  display: none;
}

</style>
<body>
<script src="http://d3js.org/d3.v3.min.js"></script>
<script>
```

```
var margin = {top: 20, right: 20, bottom: 30, left: 40},
    width = 960 - margin.left - margin.right,
    height = 500 - margin.top - margin.bottom;
```

```
var x = d3.scale.ordinal()
    .rangeRoundBands([0, width], .1);
```

```
var y = d3.scale.linear()
    .range([height, 0]);
```

```
var xAxis = d3.svg.axis()
    .scale(x)
    .orient("bottom");
```

```
var yAxis = d3.svg.axis()
    .scale(y)
    .orient("left")
    .ticks(10, "%");
```

```
var svg = d3.select("body").append("svg")
    .attr("width", width + margin.left + margin.right)
    .attr("height", height + margin.top + margin.bottom)
    .append("g")
    .attr("transform", "translate(" + margin.left + "," + margin.top + ")");
```

```
d3.tsv("data.tsv", type, function(error, data) {
  x.domain(data.map(function(d) { return d.letter; }));
  y.domain([0, d3.max(data, function(d) { return d.frequency; })]);
```

```
  svg.append("g")
    .attr("class", "x axis")
    .attr("transform", "translate(0," + height + ")")
    .call(xAxis);
```

```
  svg.append("g")
    .attr("class", "y axis")
    .call(yAxis)
    .append("text")
    .attr("transform", "rotate(-90)")
    .attr("y", 6)
    .attr("dy", ".71em")
    .style("text-anchor", "end")
    .text("Frequency");
```

```
  svg.selectAll(".bar")
    .data(data)
    .enter().append("rect")
    .attr("class", "bar")
    .attr("x", function(d) { return x(d.letter); })
    .attr("width", x.rangeBand())
    .attr("y", function(d) { return y(d.frequency); })
    .attr("height", function(d) { return height - y(d.frequency); });
```

```
});

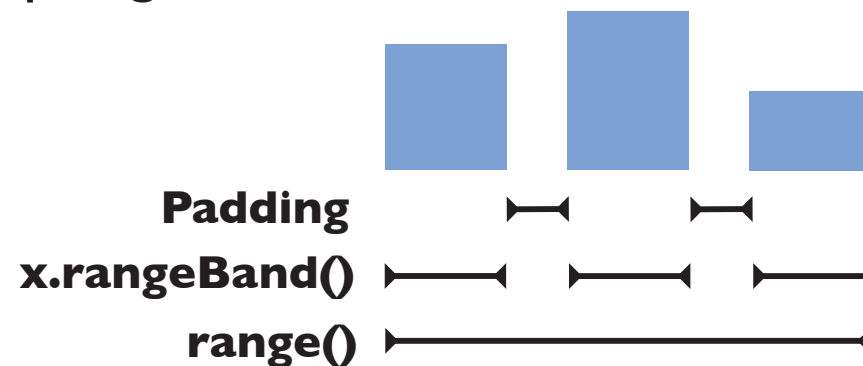
function type(d) {
  d.frequency = +d.frequency;
  return d;
}
```

```
</script>
```

# RangeBand()

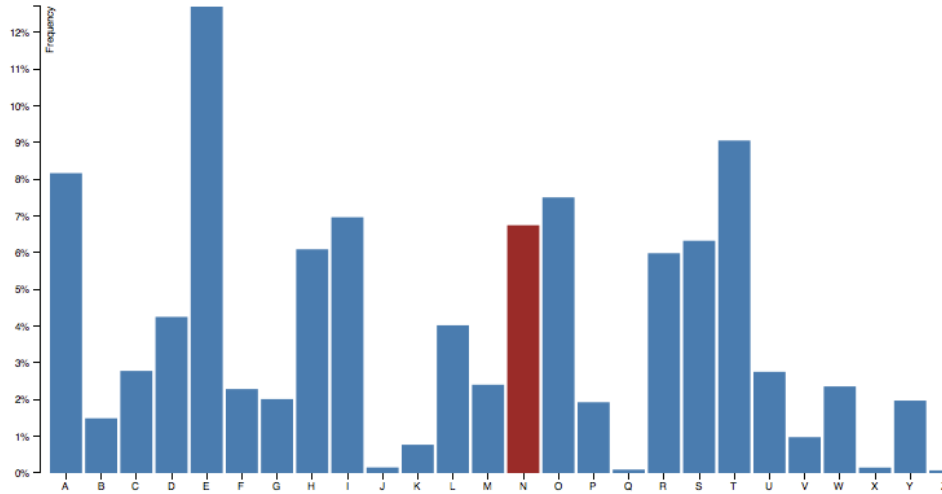


- For ordinal scales
- ```
var x = d3.scale.ordinal()  
  .domain("A", "B", "C")  
  .rangeBands([0,width], 0.3); // range with padding
```
- ```
x.rangeBand(); // bar width
```
- <https://github.com/mbostock/d3/wiki/Ordinal-Scales>



# Data: relative frequency of English letters

letter	frequency
A	.08167
B	.01492
C	.02782
D	.04253
E	.12702
F	.02288
G	.02015
H	.06094
I	.06966
J	.00153
K	.00772
L	.04025
M	.02406
N	.06749
O	.07507
P	.01929
Q	.00095
R	.05987
S	.06327
T	.09056
U	.02758
V	.00978
W	.02360
X	.00150
Y	.01974
Z	.00074



```
<!DOCTYPE html>
<meta charset="utf-8">
<style>
```

```
.bar {
  fill: steelblue;
}
```

```
.bar:hover {
  fill: brown;
}
```

```
.axis {
  font: 10px sans-serif;
}
```

```
.axis path,
.axis line {
  fill: none;
  stroke: #000;
  shape-rendering: crispEdges;
}
```

```
.x.axis path {
  display: none;
}
```

```
</style>
<body>
<script src="http://d3js.org/d3.v3.min.js"></script>
<script>
```

```
var margin = {top: 20, right: 20, bottom: 30, left: 40},
    width = 960 - margin.left - margin.right,
    height = 500 - margin.top - margin.bottom;
```

```
var x = d3.scale.ordinal()
    .rangeRoundBands([0, width], .1);
```

```
var y = d3.scale.linear()
    .range([height, 0]);
```

```
var xAxis = d3.svg.axis()
    .scale(x)
    .orient("bottom");
```

```
var yAxis = d3.svg.axis()
    .scale(y)
    .orient("left")
    .ticks(10, "%");
```

```
var svg = d3.select("body").append("svg")
    .attr("width", width + margin.left + margin.right)
    .attr("height", height + margin.top + margin.bottom)
    .append("g")
    .attr("transform", "translate(" + margin.left + "," + margin.top + ")");
```

```
d3.tsv("data.tsv", type, function(error, data) {
  x.domain(data.map(function(d) { return d.letter; }));
  y.domain([0, d3.max(data, function(d) { return d.frequency; })]);
```

```
svg.append("g")
    .attr("class", "x axis")
    .attr("transform", "translate(0," + height + ")")
    .call(xAxis);
```

```
svg.append("g")
    .attr("class", "y axis")
    .call(yAxis)
    .append("text")
    .attr("transform", "rotate(-90)")
    .attr("y", 6)
    .attr("dy", ".71em")
    .style("text-anchor", "end")
    .text("Frequency");
```

```
svg.selectAll(".bar")
    .data(data)
    .enter().append("rect")
    .attr("class", "bar")
    .attr("x", function(d) { return x(d.letter); })
    .attr("width", x.rangeBand())
    .attr("y", function(d) { return y(d.frequency); })
    .attr("height", function(d) { return height - y(d.frequency); });
```

```
});
function type(d) {
  d.frequency = +d.frequency;
  return d;
}
```

```
</script>
```

# Axes



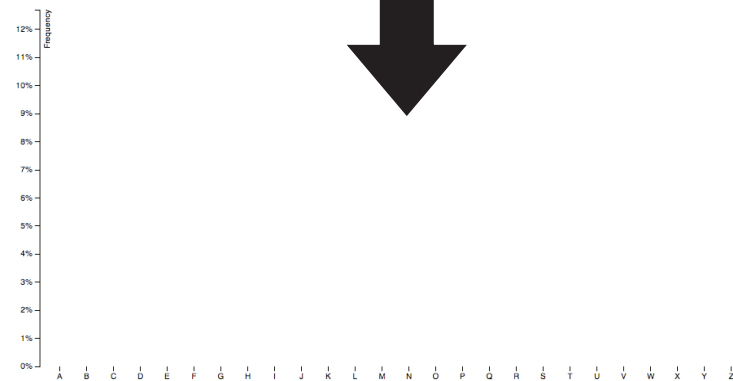
```
var xAxis = d3.svg.axis()  
  .scale(x)  
  .orient("bottom");  
  
var yAxis = d3.svg.axis()  
  .scale(y)  
  .orient("left")  
  .ticks(10, "%");
```



Number and format of tick marks

```
svg.append("g")  
  .attr("class", "x axis")  
  .attr("transform", "translate(0," + height + ")")  
  .call(xAxis);  
  
svg.append("g")  
  .attr("class", "y axis")  
  .call(yAxis)  
  .append("text")  
  .attr("transform", "rotate(-90)")  
  .attr("y", 6)  
  .attr("dy", ".71em")  
  .style("text-anchor", "end")  
  .text("Frequency");
```

```
.axis text {  
  font: 10px sans-serif;  
}  
  
.axis path,  
.axis line {  
  fill: none;  
  stroke: #000;  
  shape-rendering: crispEdges;  
}  
  
.x.axis path {  
  display: none;  
}
```



<https://github.com/mbostock/d3/wiki/SVG-Axes>



# Data: relative frequency of English letters

```

<!DOCTYPE html>
<meta charset="utf-8">
<style>

.bar {
  fill: steelblue;
}

.bar:hover {
  fill: brown;
}

.axis {
  font: 10px sans-serif;
}

.axis path,
.axis line {
  fill: none;
  stroke: #000;
  shape-rendering: crispEdges;
}

.x.axis path {
  display: none;
}

</style>
<body>
<script src="http://d3js.org/d3.v3.min.js"></script>
<script>

```

```

var margin = {top: 20, right: 20, bottom: 30, left: 40},
    width = 960 - margin.left - margin.right,
    height = 500 - margin.top - margin.bottom;

var x = d3.scale.ordinal()
    .rangeRoundBands([0, width], .1);

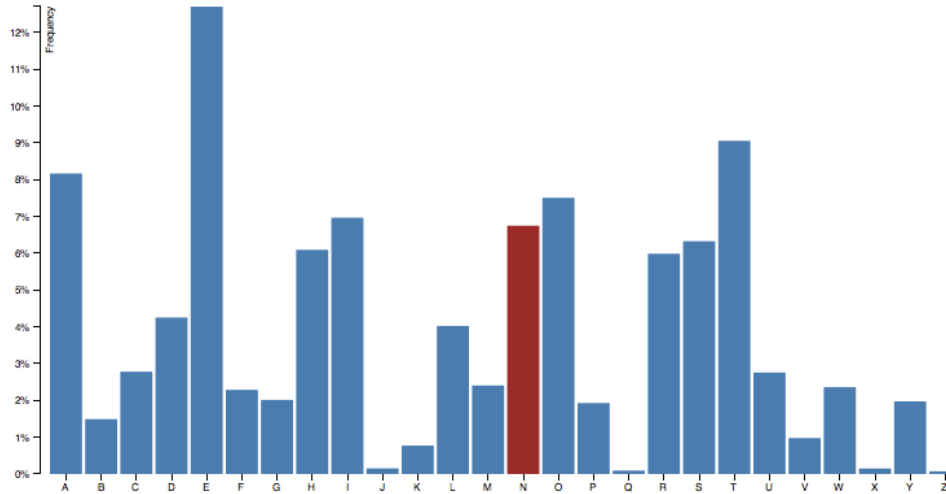
var y = d3.scale.linear()
    .range([height, 0]);

var xAxis = d3.svg.axis()
    .scale(x)
    .orient("bottom");

var yAxis = d3.svg.axis()
    .scale(y)
    .orient("left")
    .ticks(10, "%");

var svg = d3.select("body").append("svg")
    .attr("width", width + margin.left + margin.right)
    .attr("height", height + margin.top + margin.bottom)
    .append("g")
    .attr("transform", "translate(" + margin.left + "," + margin.top + ")");

```



Letter	Frequency
A	.08167
B	.01492
C	.02782
D	.04253
E	.12702
F	.02288
G	.02015
H	.06094
I	.06966
J	.00153
K	.00772
L	.04025
M	.02406
N	.06749
O	.07507
P	.01929
Q	.00095
R	.05987
S	.06327
T	.09056
U	.02758
V	.00978
W	.02360
X	.00150
Y	.01974
Z	.00074

```

d3.tsv("data.tsv", type, function(error, data) {
  x.domain(data.map(function(d) { return d.letter; }));
  y.domain([0, d3.max(data, function(d) { return d.frequency; })]);

  svg.append("g")
    .attr("class", "x axis")
    .attr("transform", "translate(0," + height + ")")
    .call(xAxis);

  svg.append("g")
    .attr("class", "y axis")
    .call(yAxis)
    .append("text")
    .attr("transform", "rotate(-90)")
    .attr("y", 6)
    .attr("dy", ".71em")
    .style("text-anchor", "end")
    .text("Frequency");

  svg.selectAll(".bar")
    .data(data)
    .enter().append("rect")
    .attr("class", "bar")
    .attr("x", function(d) { return x(d.letter); })
    .attr("width", x.rangeBand())
    .attr("y", function(d) { return y(d.frequency); })
    .attr("height", function(d) { return height - y(d.frequency); });

  });

function type(d) {
  d.frequency = +d.frequency;
  return d;
}

</script>

```

# Mouse Interactions

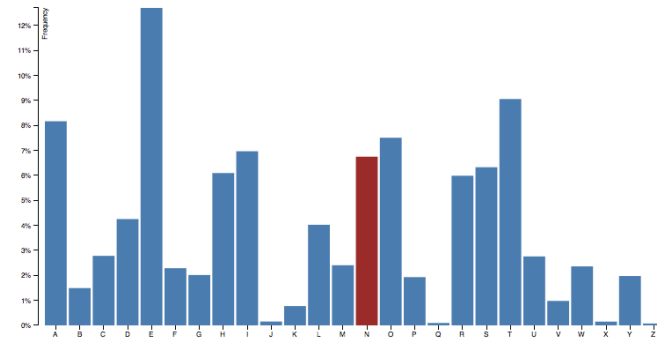


CSS

```
.bar:hover {  
  fill: brown;  
}
```

D3

```
d3.selectAll(".bar")  
  .on("mouseover", function(){  
    d3.select(this).style("fill", "brown");  
  })  
  .on("mouseout", function(){  
    d3.select(this).style("fill", "steelblue");  
  });
```



<https://github.com/mbostock/d3/wiki/Selections>



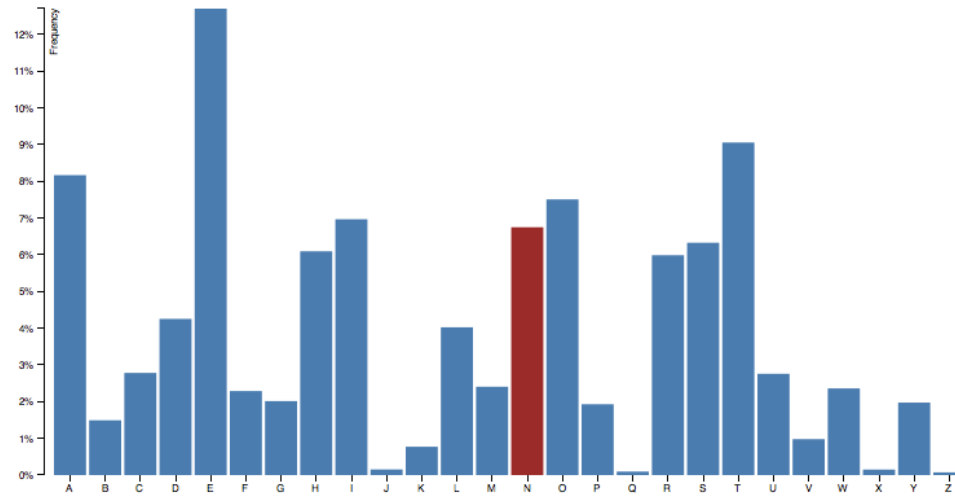
# Transitions



- ```
d3.selectAll(".bar")  
  .on("mouseout", function(){  
    d3.select(this)  
      .transition()  
      .duration(500)  
      .style("fill", "steelblue");  
  });
```
- Bar fades into steelblue over 500 milliseconds when mouse moved out of it
- <https://github.com/mbostock/d3/wiki/Transitions>

# Relative frequency of English letters

| letter | frequency |
|--------|-----------|
| A      | .08167    |
| B      | .01492    |
| C      | .02782    |
| D      | .04253    |
| E      | .12702    |
| F      | .02288    |
| G      | .02015    |
| H      | .06094    |
| I      | .06966    |
| J      | .00153    |
| K      | .00772    |
| L      | .04025    |
| M      | .02406    |
| N      | .06749    |
| O      | .07507    |
| P      | .01929    |
| Q      | .00095    |
| R      | .05987    |
| S      | .06327    |
| T      | .09056    |
| U      | .02758    |
| V      | .00978    |
| W      | .02360    |
| X      | .00150    |
| Y      | .01974    |
| Z      | .00074    |



```

<!DOCTYPE html>
<meta charset="utf-8">
<style>

.bar {
  fill: steelblue;
}

.bar:hover {
  fill: brown;
}

.axis {
  font: 10px sans-serif;
}

.axis path,
.axis line {
  fill: none;
  stroke: #000;
  shape-rendering: crispEdges;
}

.x.axis path {
  display: none;
}

</style>
<body>
<script src="http://d3js.org/d3.v3.min.js"></script>
<script>

var margin = {top: 20, right: 20, bottom: 30, left: 40},
    width = 960 - margin.left - margin.right,
    height = 500 - margin.top - margin.bottom;

var x = d3.scale.ordinal()
    .rangeRoundBands([0, width], .1);

var y = d3.scale.linear()
    .range([height, 0]);

var xAxis = d3.svg.axis()
    .scale(x)
    .orient("bottom");

var yAxis = d3.svg.axis()
    .scale(y)
    .orient("left")
    .ticks(10, "%");

var svg = d3.select("body").append("svg")
    .attr("width", width + margin.left + margin.right)
    .attr("height", height + margin.top + margin.bottom)
    .append("g")
    .attr("transform", "translate(" + margin.left + "," + margin.top + ")");

```

```

d3.tsv("data.tsv", type, function(error, data) {
  x.domain(data.map(function(d) { return d.letter; }));
  y.domain([0, d3.max(data, function(d) { return d.frequency; })]);

  svg.append("g")
    .attr("class", "x axis")
    .attr("transform", "translate(0," + height + ")")
    .call(xAxis);

  svg.append("g")
    .attr("class", "y axis")
    .call(yAxis)
    .append("text")
    .attr("transform", "rotate(-90)")
    .attr("y", 6)
    .attr("dy", ".71em")
    .style("text-anchor", "end")
    .text("Frequency");

  svg.selectAll(".bar")
    .data(data)
    .enter().append("rect")
    .attr("class", "bar")
    .attr("x", function(d) { return x(d.letter); })
    .attr("width", x.rangeBand())
    .attr("y", function(d) { return y(d.frequency); })
    .attr("height", function(d) { return height - y(d.frequency); });

});

function type(d) {
  d.frequency = +d.frequency;
  return d;
}
</script>

```

# Debugging



- Google Chrome Developer Tools
- Inspect HTML elements
- Debug Javascript

# Examples



- Example 1
  - <http://bl.ocks.org/mbostock/7341714>
- Example 2
  - <http://bl.ocks.org/mbostock/3885304>