VISUALIZATION ON THE WEB

tableau.com

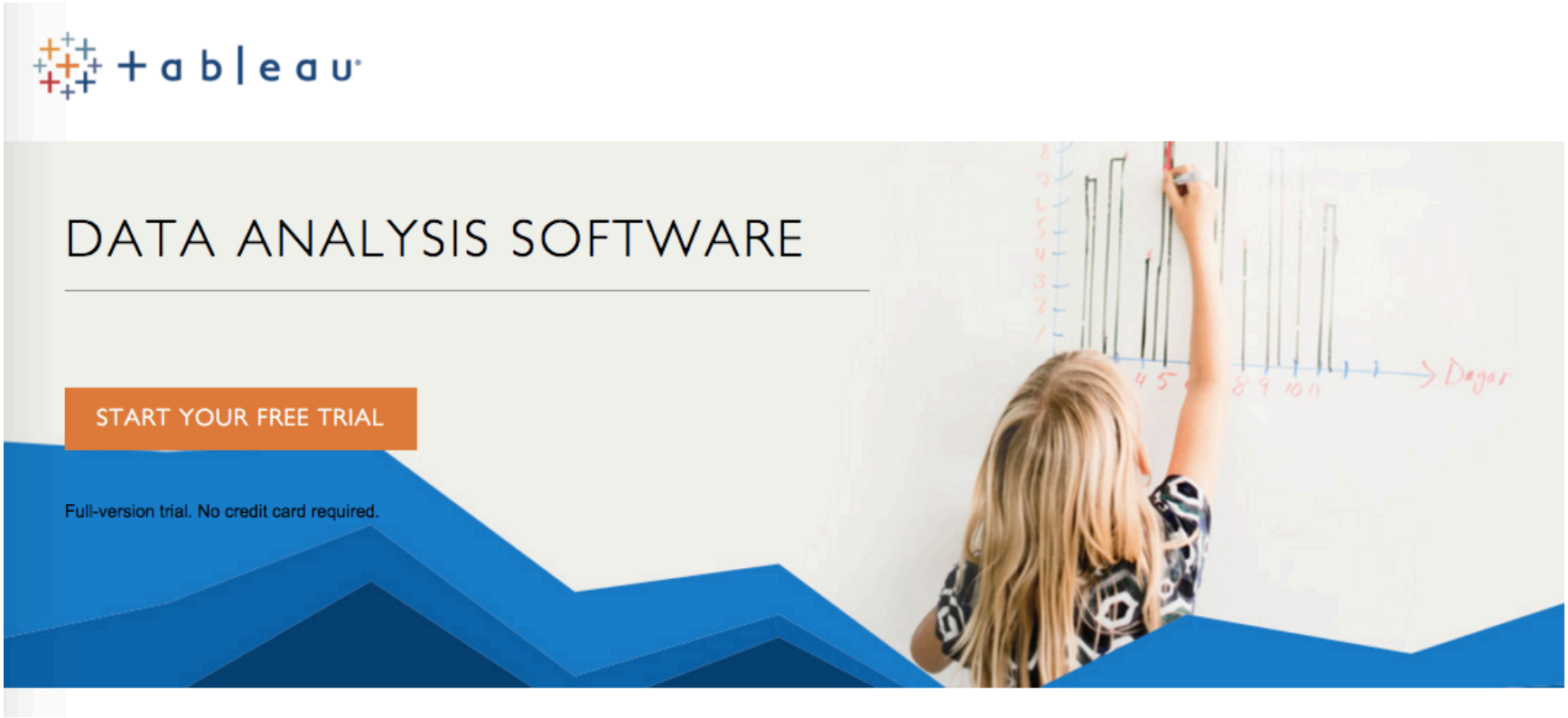
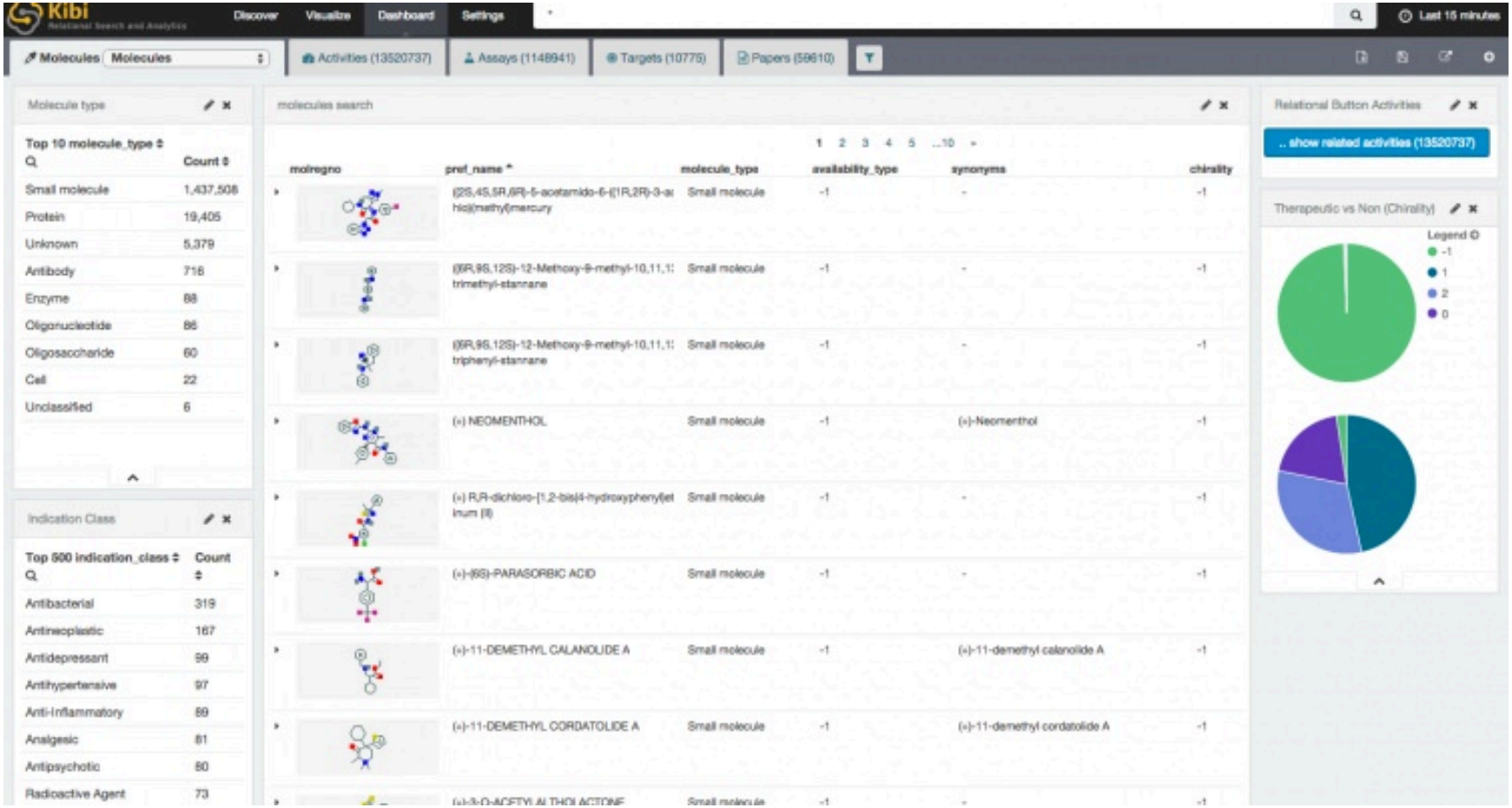


tableau
DATA ANALYSIS SOFTWARE
START YOUR FREE TRIAL
Full-version trial. No credit card required.

Kibana GA

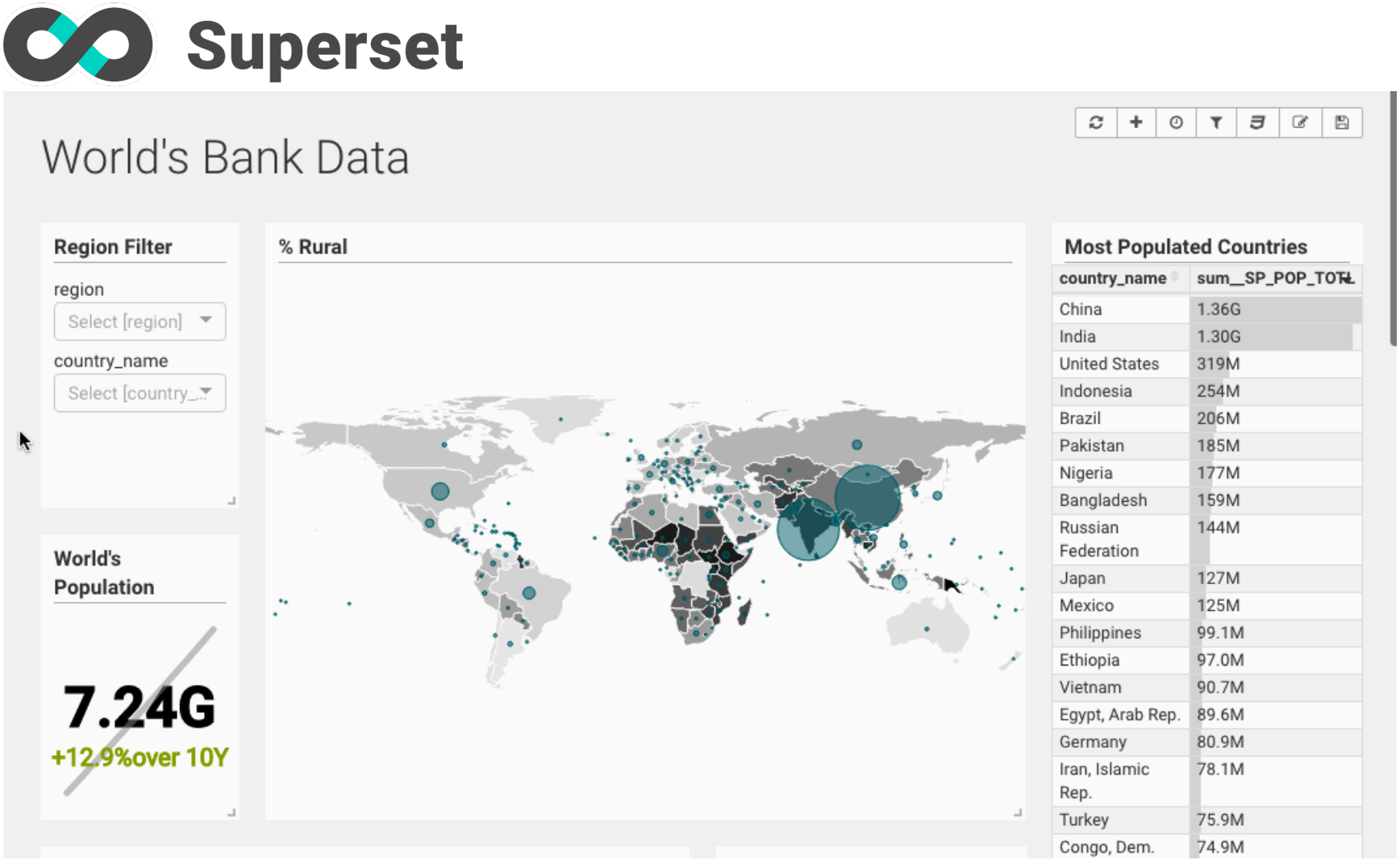

Logstash Search
2 days ago to 3 minutes ago
QUERY
bytes:[0 TO 4000000] AND @tags:success
FILTERING
EVENTS OVER TIME
MAP
REVENUE
VOLUME
TOP DESTINATIONS
TOP SOURCES
GEO PAIRS
ALL EVENTS
Fields
Type to filter...
@message
@tags
0 to 100 of 500 available for paging
@tags
geo.srcdst
extension
clientip
bytes
id
phpmemory
response
success,security
MY:VN
html
167.12.22.189
8540
1066
200
success,info
IT:MM
png
164.87.170.73
2045
1903
200
success,info
AR:ES
html
222.23.102.238
1801
1133
200
success,info
IN:DZ
html
138.226.66.81
7029
1801
200

Kibi



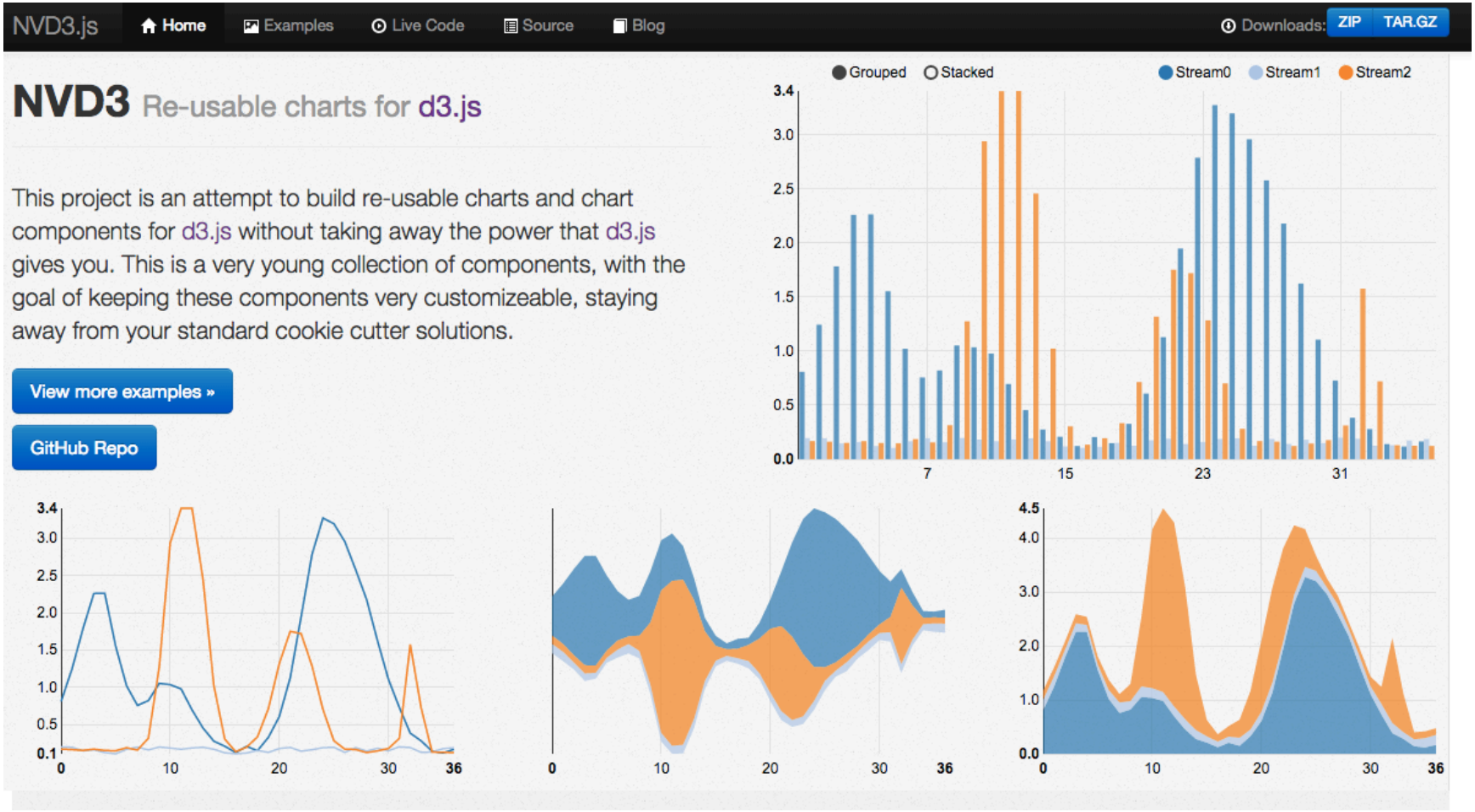
https://siren.solutions/kibi/

Superset

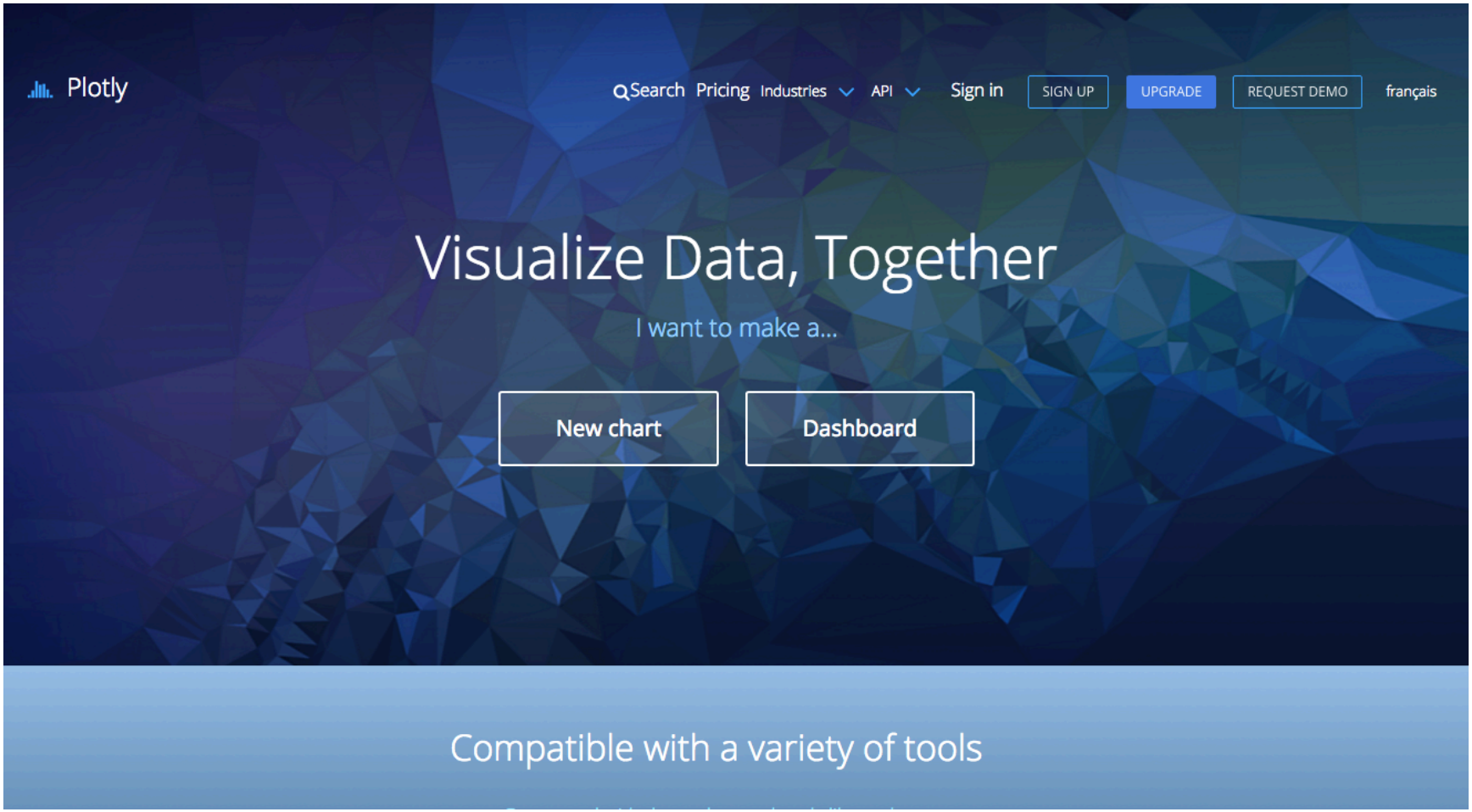


https://github.com/airbnb/superset

NVD3.js


NVD3.js
Home
Examples
Live Code
Source
Blog
Downloads:
ZIP
TAR.GZ
NVD3 Re-usable charts for d3.js
This project is an attempt to build re-usable charts and chart components for d3.js without taking away the power that d3.js gives you. This is a very young collection of components, with the goal of keeping these components very customizeable, staying away from your standard cookie cutter solutions.
View more examples »
GitHub Repo
Grouped
Stacked
Stream0
Stream1
Stream2

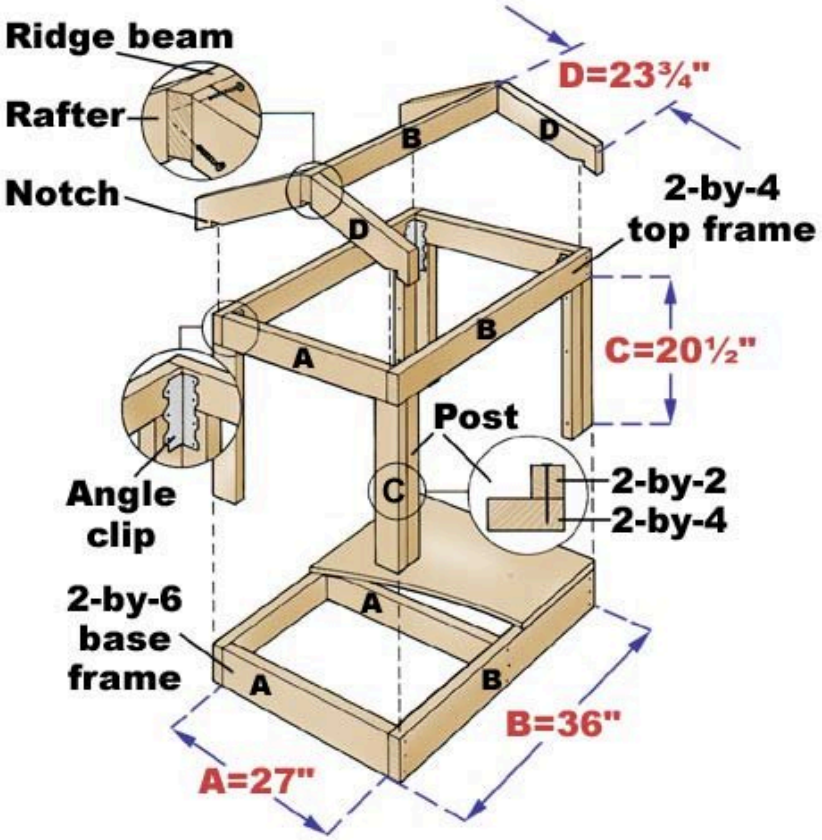
Plot.ly


Plotly
Search
Pricing
Industries
API
Sign in
SIGN UP
UPGRADE
REQUEST DEMO
français
Visualize Data, Together
I want to make a...
New chart
Dashboard
Compatible with a variety of tools

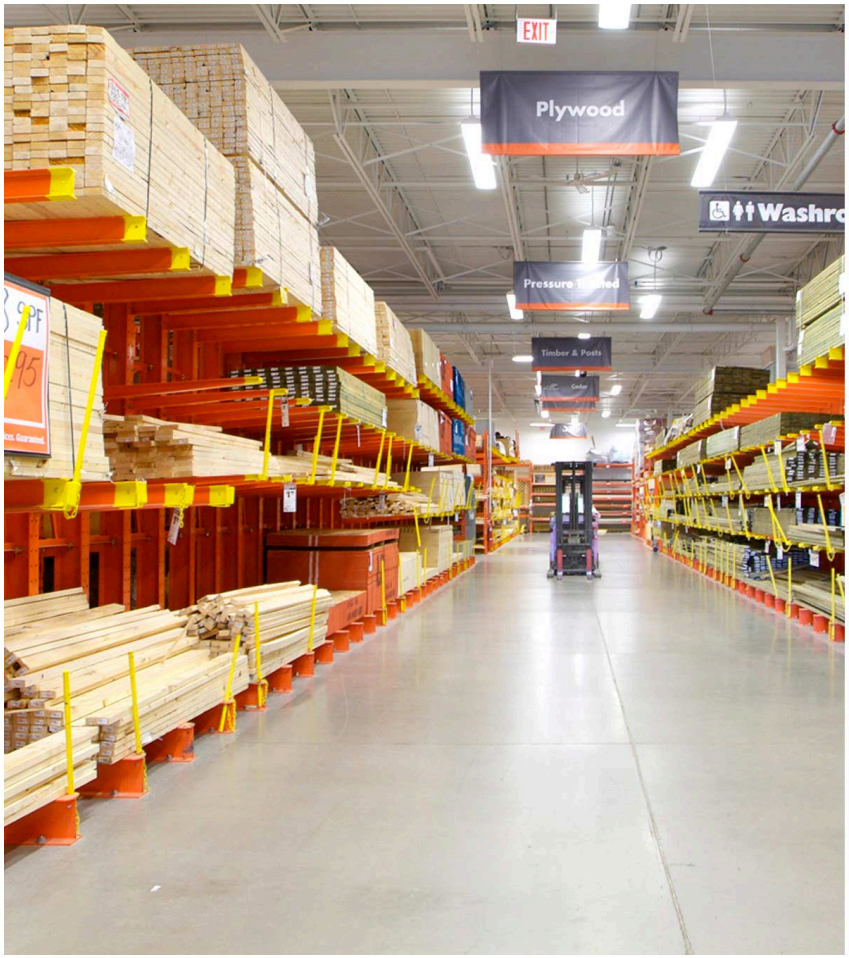
Data-Driven Documents

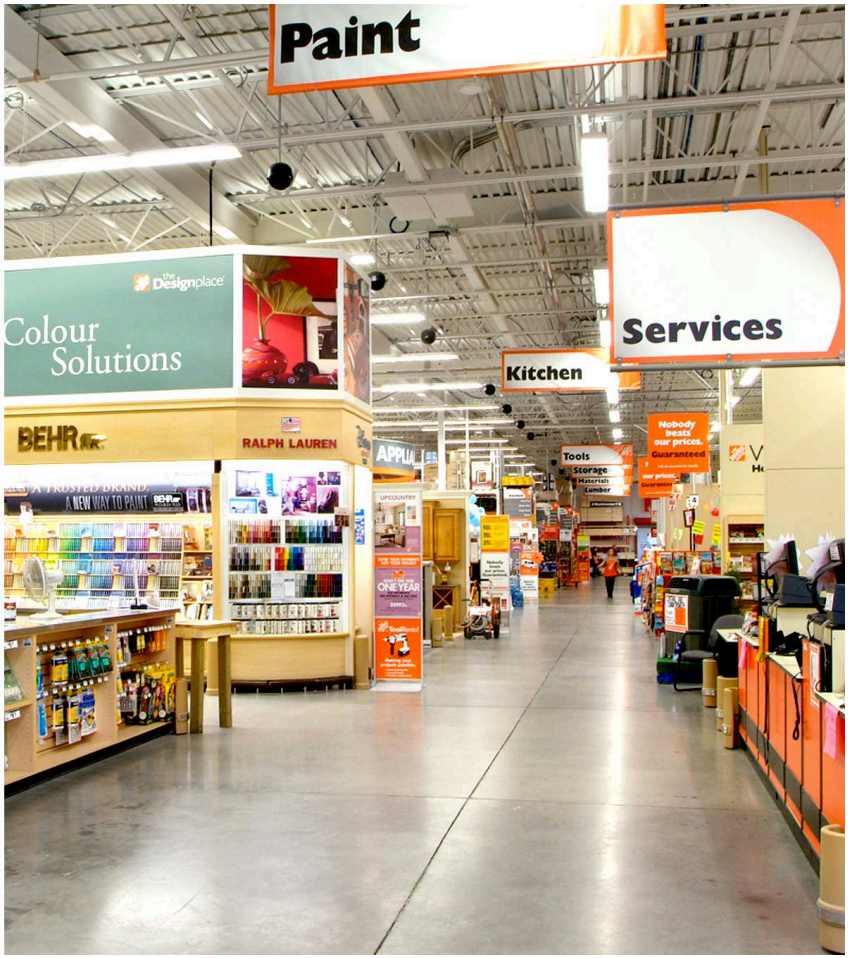
VISUAL ANALYTICS
D3.JS

What is D3?


Ridge beam
Rafter
Notch
D=23¾"
2-by-4
top frame
C=20½"
Post
2-by-2
2-by-4
Angle
clip
2-by-6
base
frame
A=27"
B=36"
A
B
C
D

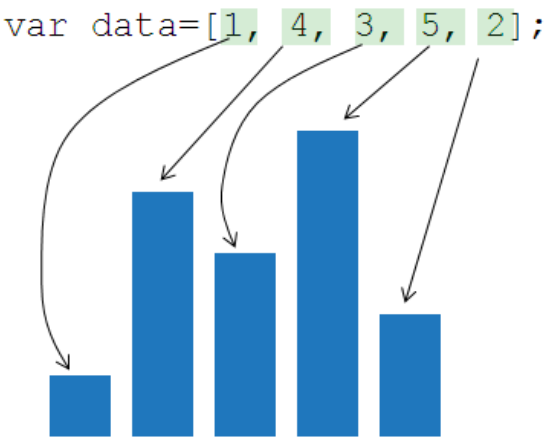
What is D3?


EXIT
Plywood
Timber & Posts


Paint
Colour Solutions
Designplace
BEHR
RALPH LAUREN
Services
Kitchen
Tools
Storage
Materials
Lumber
Nobody beats our prices.
Guaranteed

What is D3?

- JavaScript library to make beautiful, interactive, browser-based data visualizations.
- D3 stands for **D**ata **D**riven **D**ocuments
- D3.js is a low level visualization library based on Web standards (HTML, CSS, JS, SVG)
- D3.js is Open Source library written by Mike Bostok
- Mike Bostock Github Profile
- d3js.org

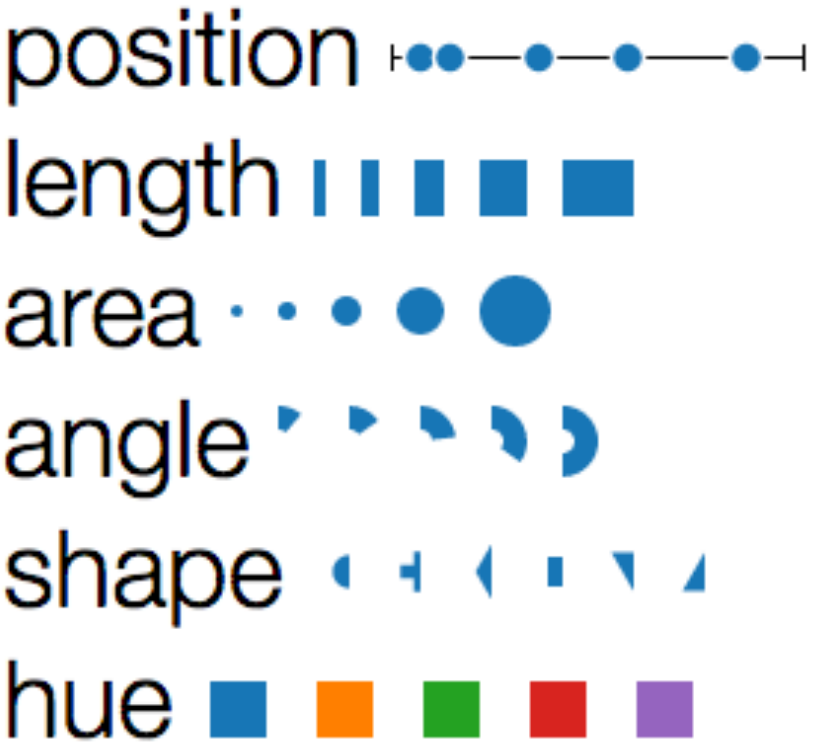


Visualization and Data Graphics

Data Types

- Categorical
- Ordinal
- Quantitative

Visual Variables

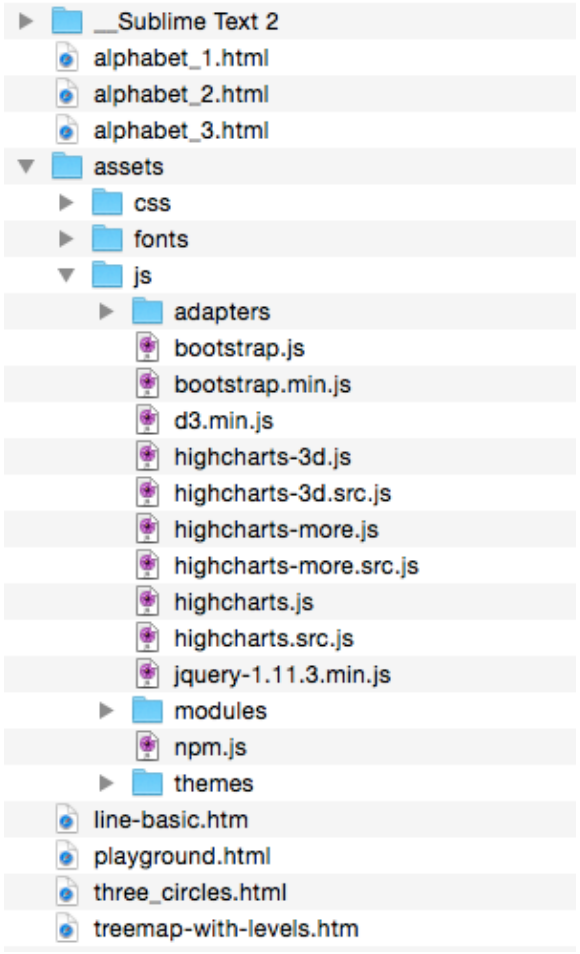


Visual Variables -> Documents

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

GETTING STARTED

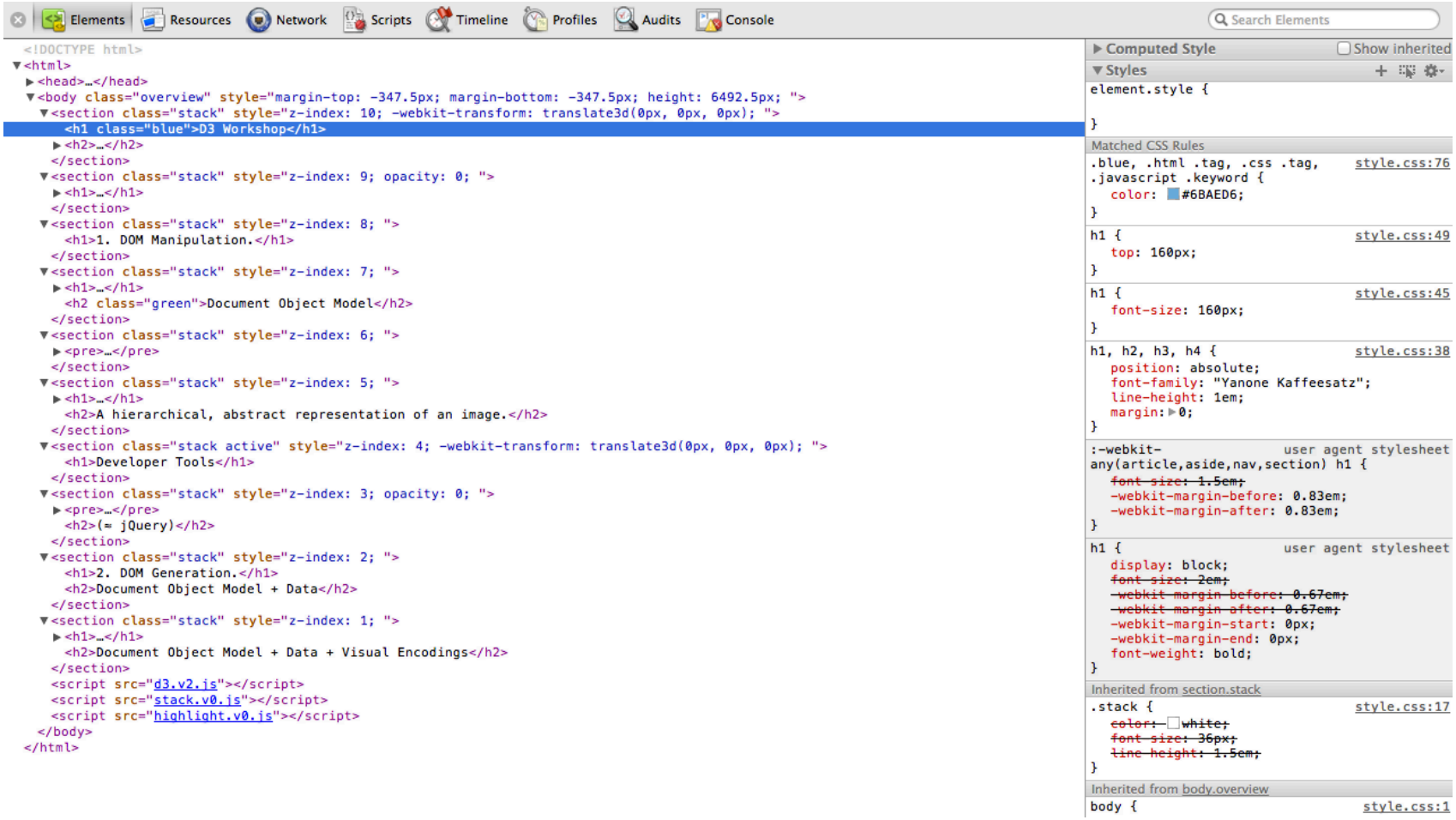
Your toolbox



Web Server

- A web server is required when loading external data
- Python
  - > python -m SimpleHTTPServer 8888
- LAMP/MAMP/WAMP
- Other webserver…

Developer Tools (Safari, Chrome, Firefox)



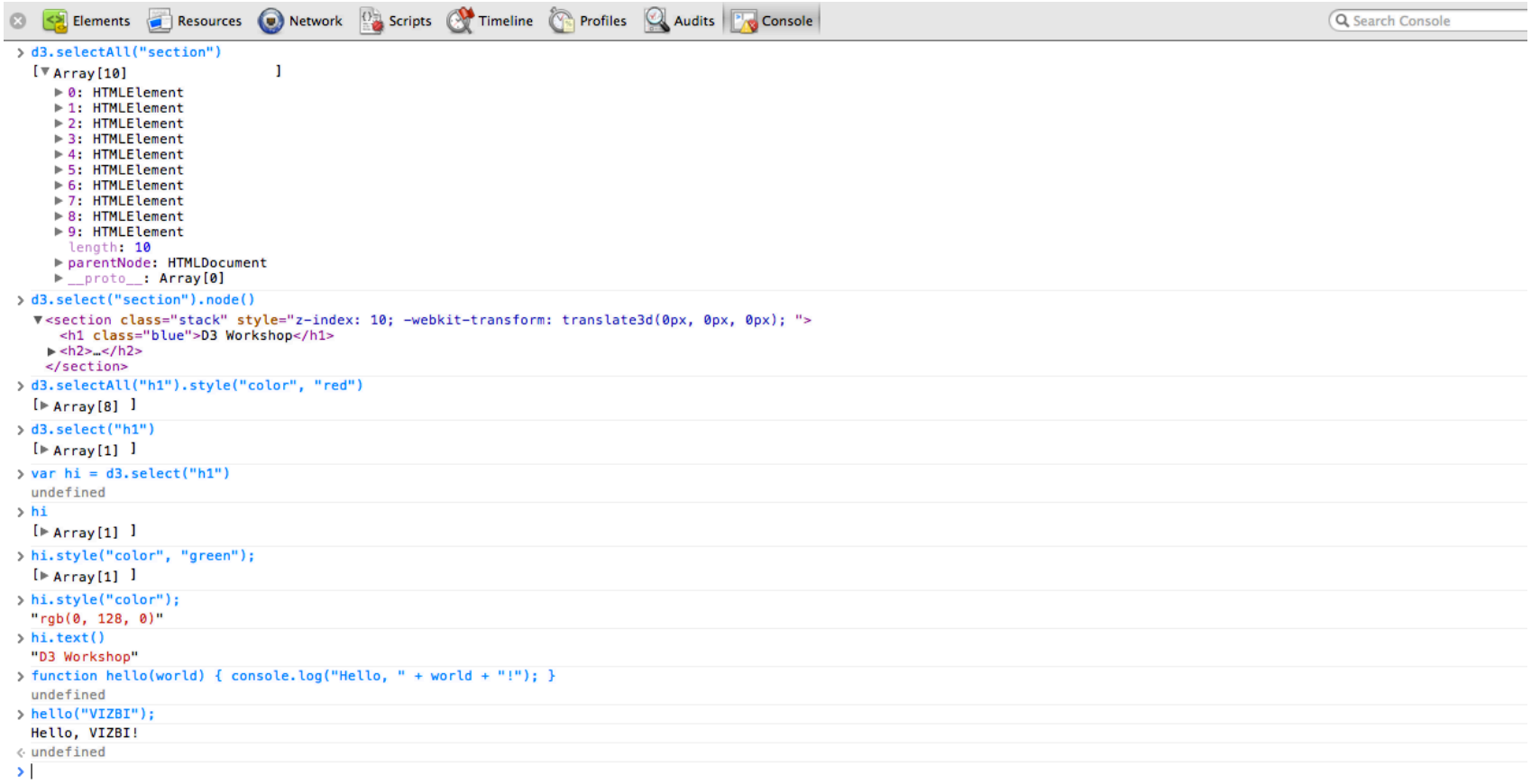
Getting started – Anatomy of an HTML Page

```
<!DOCTYPE html>
<html>
   <head>
      <script src="http://d3js.org/d3.v3.min.js"></script>
      <script src="js/main.js" ></script>
   </head>
   <body>
      ...
   </body>
</html>
```

main.js

```
d3.select("body").append("p").text("Hello
World!");
```

Javascript Console (Safari, Chrome, Firefox)



SELECTIONS

CSS Selectors

- CSS provides an efficient way to refer to specific elements in a DOM

- `#foo // <any id="foo">`
- `foo  // <foo>…</foo>`
- `.foo // <any class="foo">`
- `[foo=bar] // <any foo="bar">`
- `foo bar //<foo><bar/></foo>`

Selector Functions

W3C

- `document.querySelectorAll("h1")`

D3.js / JQuery

- `d3.selectAll("h1")`

Selections are Arrays.
Explore selections with Developer Tools

attr and style methods

```
// select all <h1> elements
var H1s = d3.selectAll("H1");

H1s.attr("class","newClass");
H1s.style("fill","yellow");
H1s.style("font-color","black");
```

Chaining methods

```
d3.selectAll(“H1”)
  .attr(“class”,”newClass”)
  .style(“fill”,”yellow”)
  .style(“font-color”,”black”);
```

Append new elements

```
var body = d3.select("body");

var h1 = body.append("h1");
h1.text("Hello!");
```

Modify existing elements

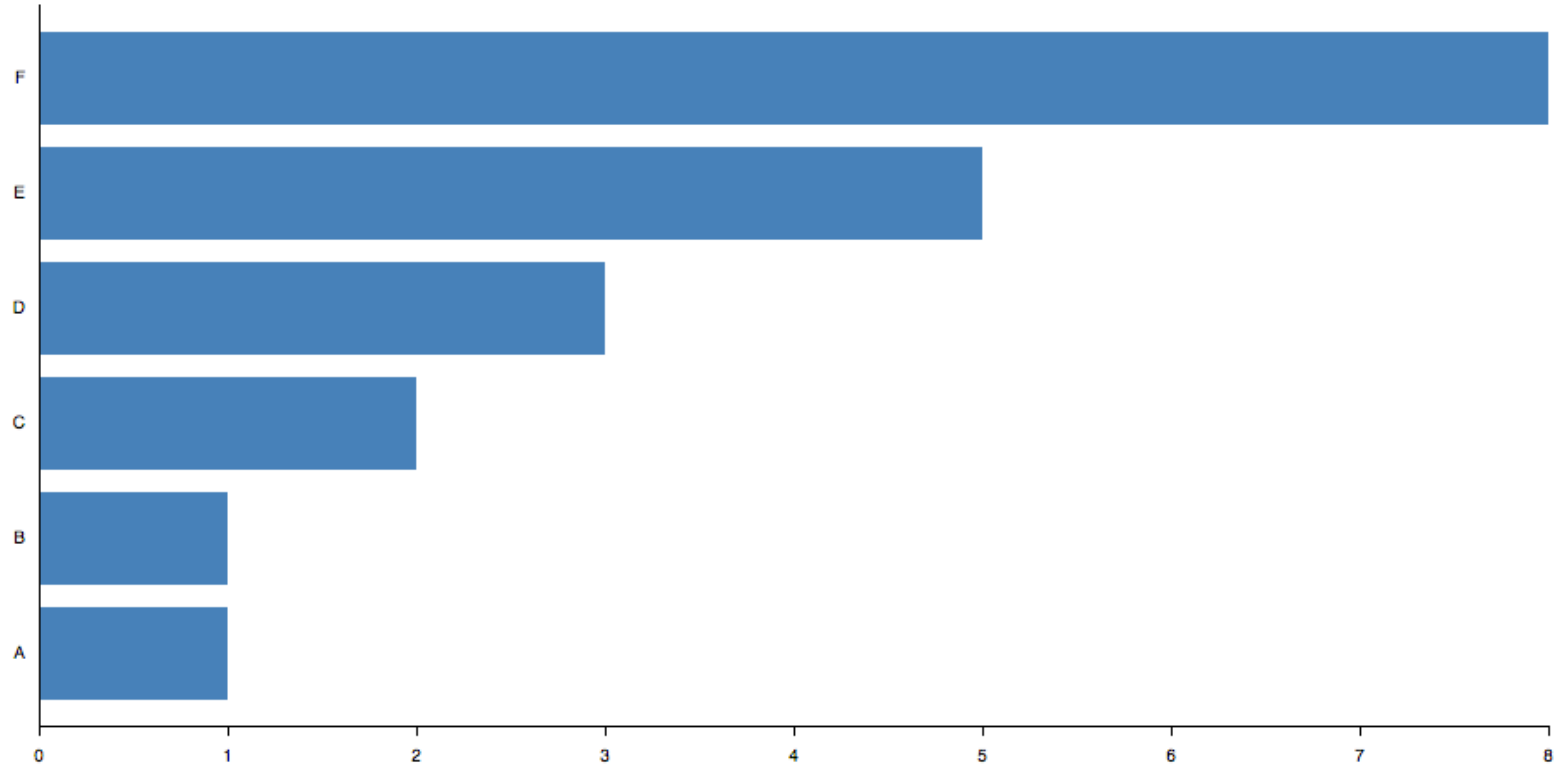
```
var section = d3.selectAll(“section”);

var h1 = section.append(“h1”);
h1.text(“Hello!”);
```

Exercise: Playground

See example playground.html

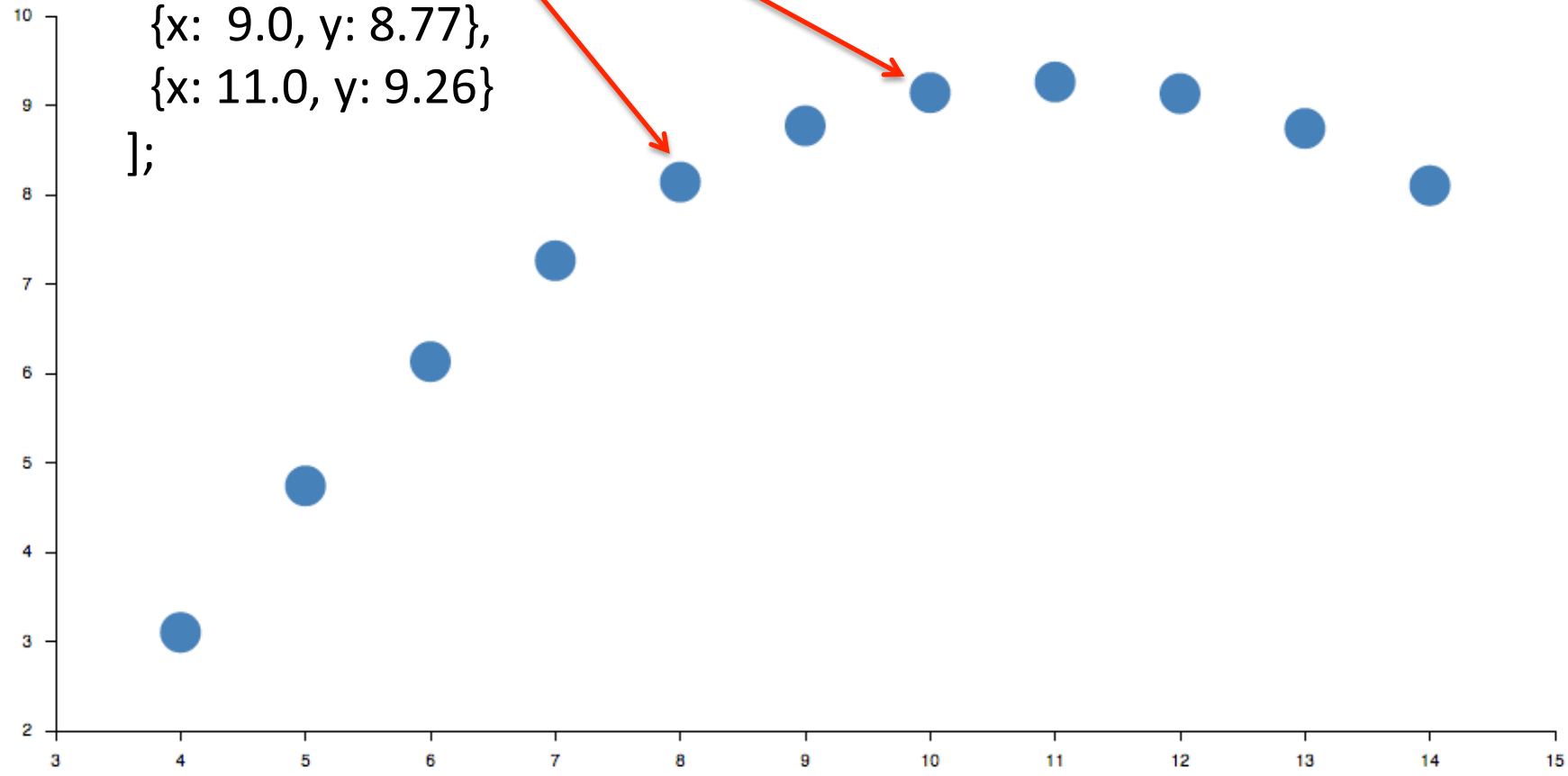
Data can be numbers

```
var numbers= [1, 1, 2, 3, 5, 8];
```



Data can be objects.

```
var data = [
  {x: 10.0, y: 9.14},
  {x:  8.0, y: 8.14},
  {x: 13.0, y: 8.74},
  {x:  9.0, y: 8.77},
  {x: 11.0, y: 9.26}
];
```



DATA TO ELEMENTS

Selection should correspond to data

```
svg.selectAll("circle")
   .data(data)
   .enter().append("circle")
   .attr("cx", x)
   .attr("cy", y)
   .attr("r", 2.5);
```

Selection should correspond to data

```
svg.selectAll("circle")
   .data(data)
   .enter().append("circle")
   .attr("cx", x)
   .attr("cy", y)
   .attr("r", 2.5);
```

Method data joins data with document elements

Selection should correspond to data

```
svg.selectAll("circle")
   .data(data)
   .enter().append("circle")
   .attr("cx", x)
   .attr("cy", y)
   .attr("r", 2.5);
```

Method enter specifies the action for missing elements

Selection should correspond to data

```
svg.selectAll("circle")
   .data(data)
   .enter().append("circle")
   .attr("cx", x)
   .attr("cy", y)
   .attr("r", 2.5);
```

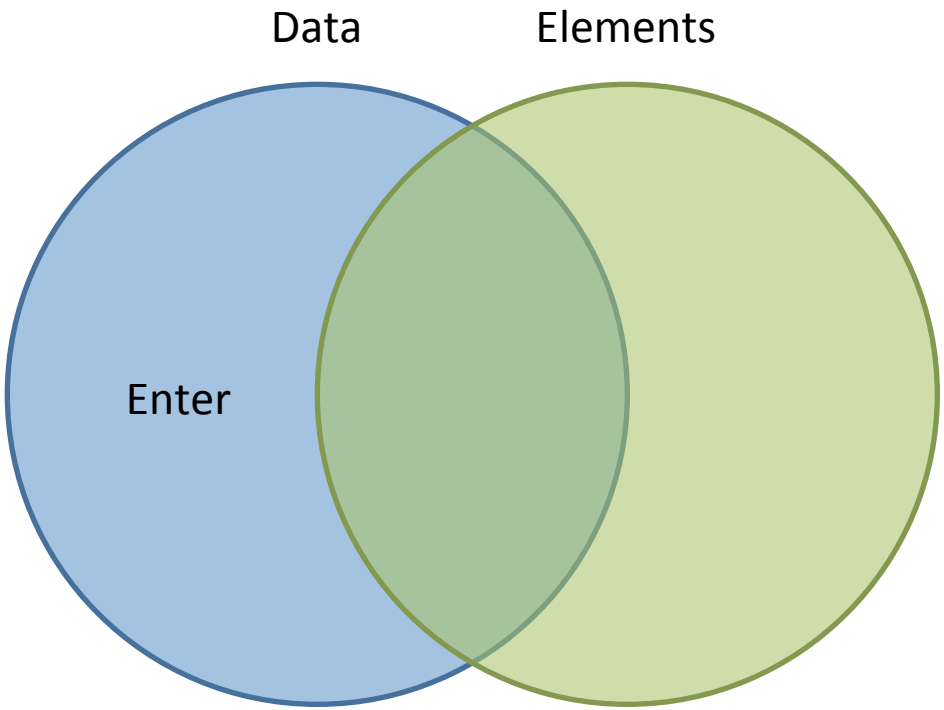
The new elements are bound to data. Data can be used to compute attributes

Thinking with Joins

ENTER, UPDATE, AND EXIT

Enter

- New data, for which there were no existing elements.

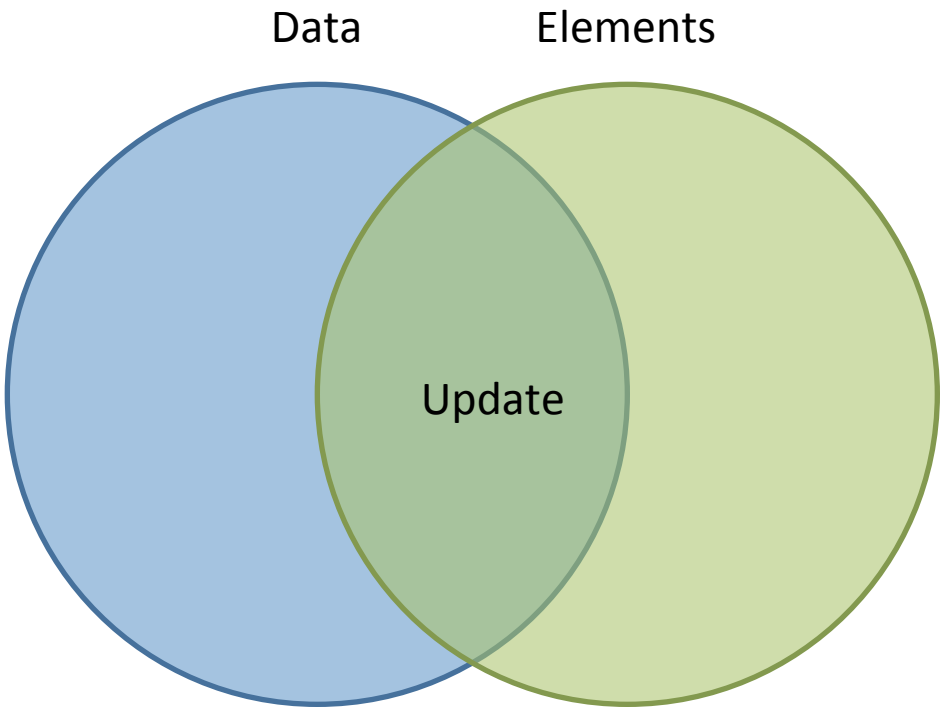


Entering new elements

```
svg.selectAll("circle")
   .data(data)
   .enter().append("circle")
   .attr("cx", x)
   .attr("cy", y)
   .attr("r", 2.5);
```

Update

- Data that was joined previously to an existing element

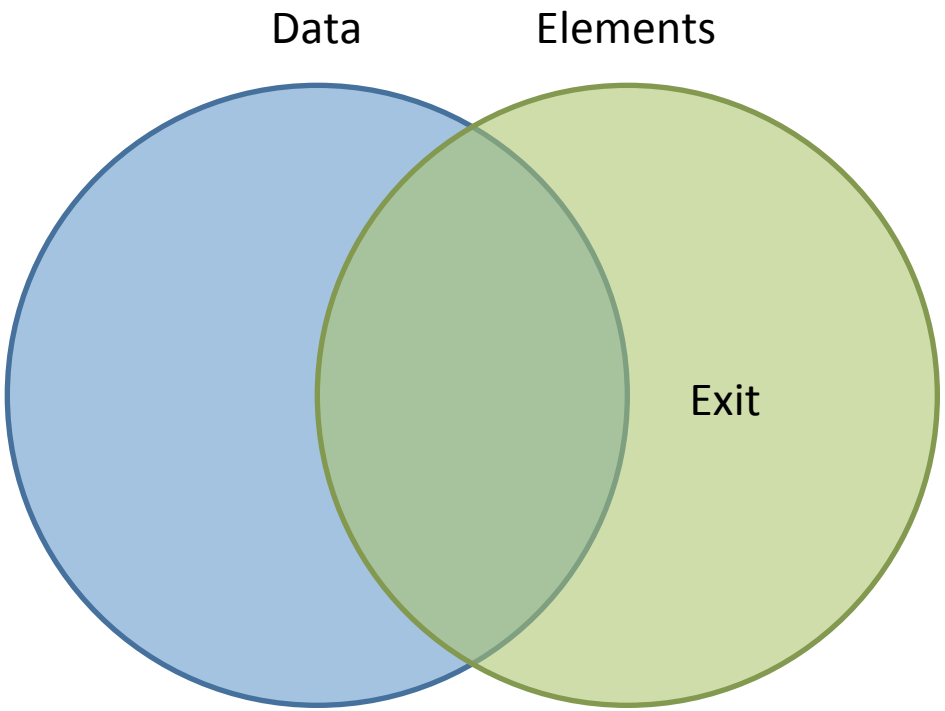


Updating existing elements

```
svg.selectAll("circle")
   .data(data)
   .attr("cx", x)
   .attr("cy", y)
   .attr("r", 2.5);
```

Exit

- Data that is not associated to any data

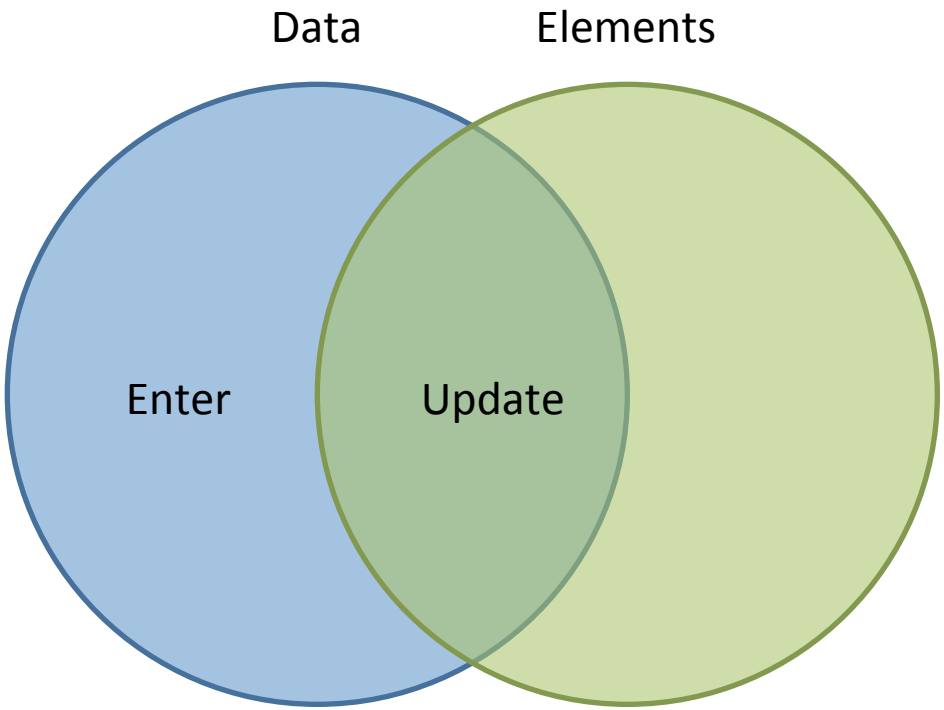


Removing unlinked elements

```
svg.selectAll("circle")
   .data(data)
   .exit().remove();
```

Enter + Update

- Often, we need to perform both Enter and Update



Entering and then update

```
// perform join and add missing elements
var circles = svg.selectAll("circle")
   .data(data)
   .enter().append("circle");
// Update previous and new elements at
once
svg.selectAll("circle")
   .attr("cx", x)
   .attr("cy", y)
   .attr("r", 2.5);
```

Playground.html text example

Exercise