

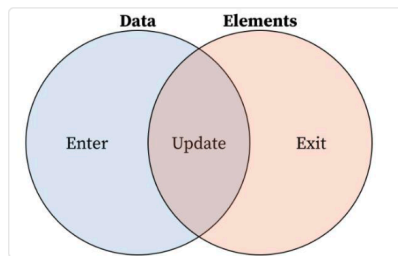
Introduction to D3

Maneesh Agrawala

CS 448B: Visualization
Fall 2020

1

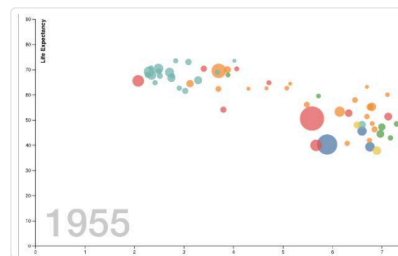
D3 Notebooks



Team - Published

Introduction to D3

You republished 14 hours ago



You - Published

Making D3 Charts Interactive

You republished 14 hours ago

2

Last Time: Interaction

3

Dynamic Queries

6

Query and results

SELECT house
FROM east bay
WHERE price < 1,000,000 AND bedrooms > 2
ORDER BY price

Dynamic Browser : DC Home Finder

IdNumber	Dwelling	Address	City
2	House	5256 S. Capitol St.	Beltsville, MD
4	House	5536 S. Lincoln St.	Beltsville, MD
5	House	5165 Jones Street	Beltsville, MD
8	House	5007 Jones Street	Beltsville, MD
9	House	4872 Jones Street	Beltsville, MD
17	House	5408 S. Capitol St.	Beltsville, MD
20	House	5496 S. Capitol St.	Beltsville, MD
85	Condo	5459 S. Lincoln St.	Laurel, MD
86	Condo	5051 S. Lincoln St.	Laurel, MD
88	Condo	5159 Hamilton Street	Laurel, MD
92	Condo	5132 Hamilton Street	Laurel, MD
93	Condo	5221 S. Lincoln St.	Laurel, MD
94	Condo	5043 S. Lincoln St.	Laurel, MD
95	Condo	4970 Jones Street	Laurel, MD
97	Condo	4677 Jones Street	Laurel, MD
98	Condo	4896 S. Capitol St.	Laurel, MD
99	Condo	5048 S. Capitol St.	Laurel, MD
100	Condo	4597 31st Street	Laurel, MD
101	Condo	5306 S. Lincoln St.	Laurel, MD
103	Condo	5562 Glass Road	Laurel, MD
105	Condo	5546 Hamilton Street	Laurel, MD
152	House	7670 31st Street	Upper Marlboro, MD

7

HomeFinder

Dynamic HomeFinder

Reset Quit

Save Print

Dist to A: 30

Dist to B: 30

Bedrooms: 7

Cost: \$50k \$500k

Look at: Hse TH Cnd

Features: Grg Fp1 CAC New

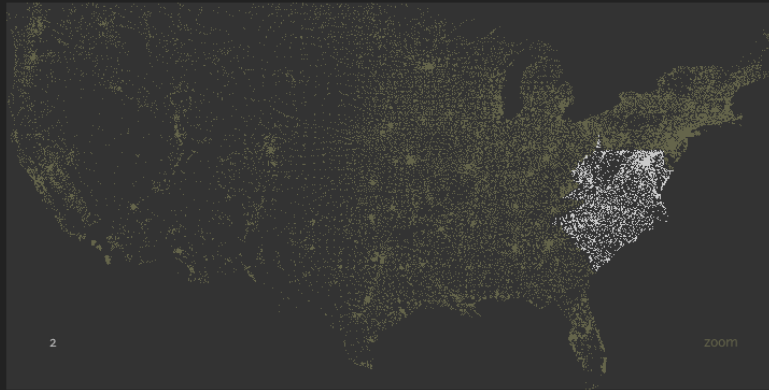
The yellow dots above are homes in the DC area for sale. You may get more information on a home by selecting it. You may drag the 'A' and 'B' distance markers to your office or any other location you want to live near. Select distances, bedrooms, and cost ranges by dragging the corresponding slider boxes on the right. Select specific home types and services by pressing the labeled buttons on the right.

[Ahlberg and Schneiderman 92]

9

Zipcode [from Fry 04]

<< ben fry

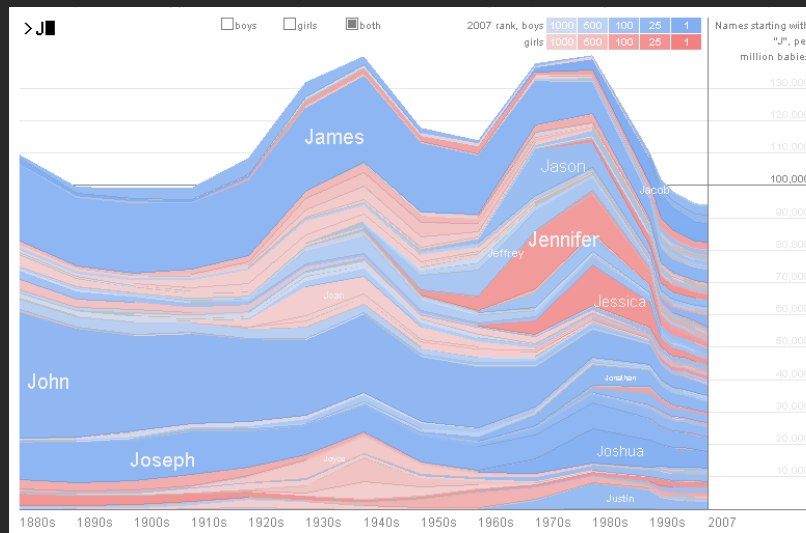


Hit the letter z, or click the word zoom to enable or disable zooming.
Hold down shift while typing a number to replace the previous number
(U.S. keyboards only).

<https://benfry.com/zipcode/>

17

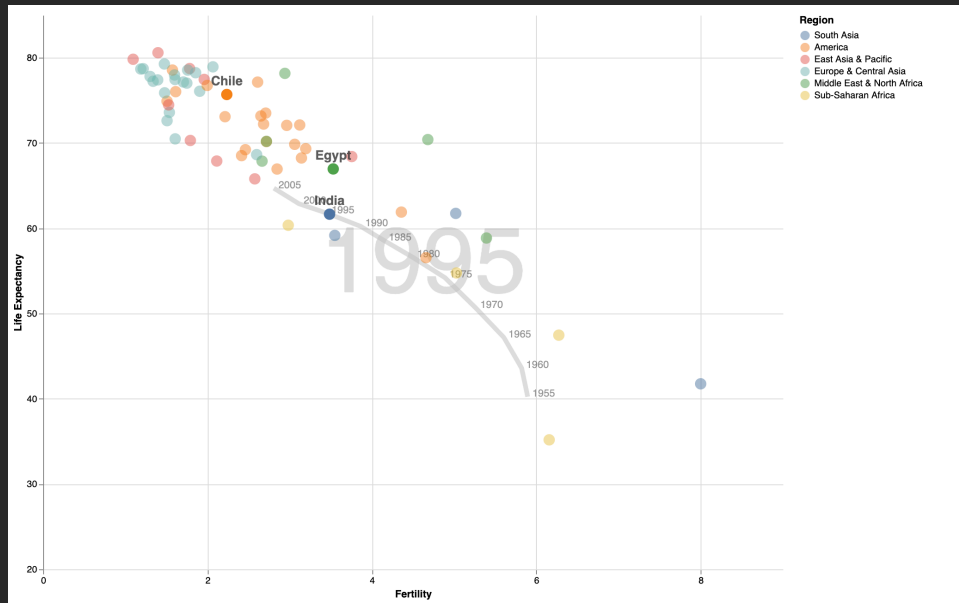
NameVoyager



<http://www.babynamewizard.com/voyager>

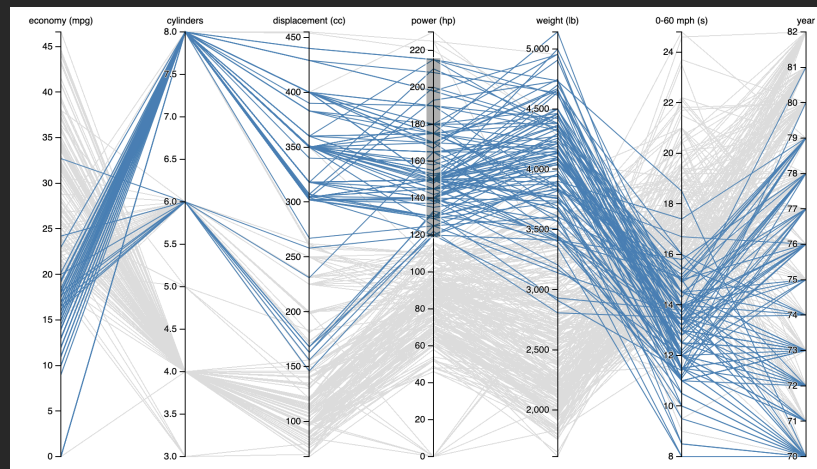
18

DimpVis [Kondo 14]



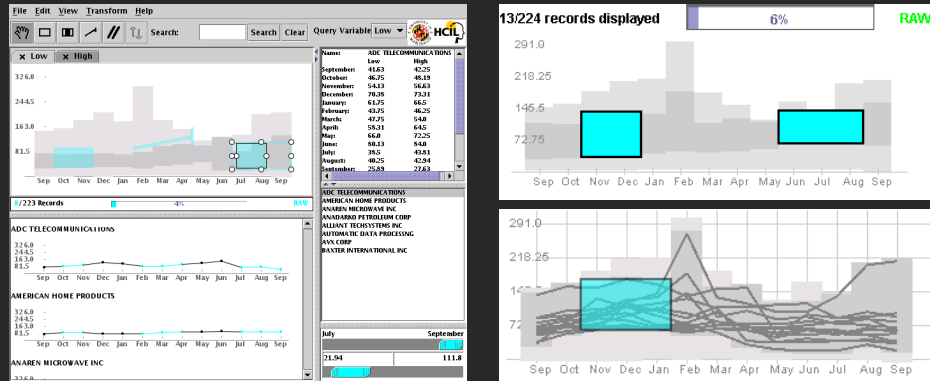
19

Parallel Coordinates [Inselberg]



20

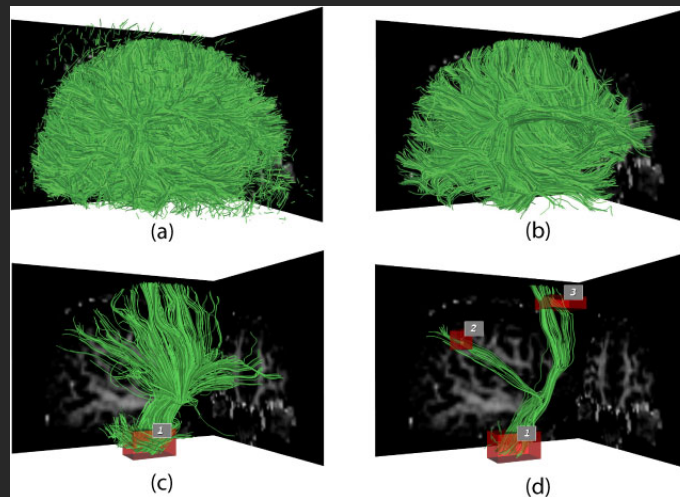
TimeSearcher [Hochheiser & Schneiderman 02]



Based on Wattenberg's [2001] idea for sketch-based queries of time-series data.

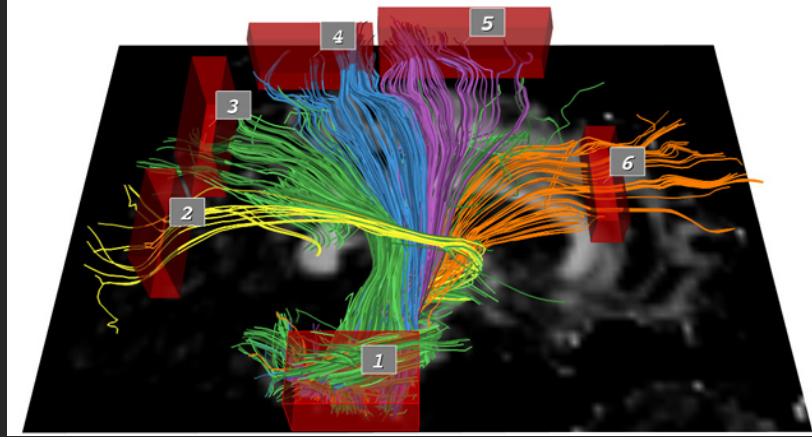
21

3D dynamic queries [Akers et al. 04]



22

3D dynamic queries [Akers et al. 04]



23

Pros and cons

Pros

- Controls useful for both novices and experts
- Quick way to explore data

Cons

- Simple queries
- Lots of controls
- Amount of data shown limited by screen space

24

Summary

Most visualizations are interactive

- Note that even passive media elicit interactions

Good visualizations are task dependent

- Pick the right interaction technique

Fundamental interaction techniques

- Selection, Brushing & Linking, Dynamic Queries

25

Announcements

26

A2: Exploratory Data Analysis

Use **Tableau** to formulate & answer questions

First steps

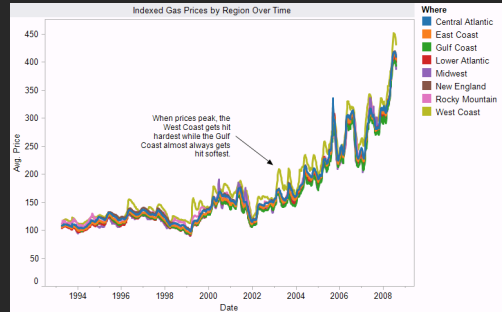
- Step 1: Pick domain & data
- Step 2: Pose questions
- Step 3: Profile data
- Iterate as needed

Create visualizations

- Interact with data
- Refine questions

Author a report

- Screenshots of most insightful views (10+)
- Include titles and captions for each view



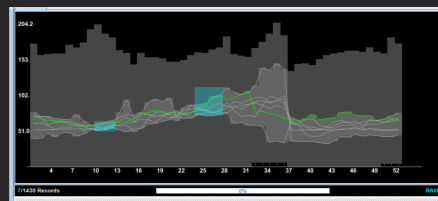
Due before class on **Oct 6, 2020**

27

Assignment 3: Dynamic Queries

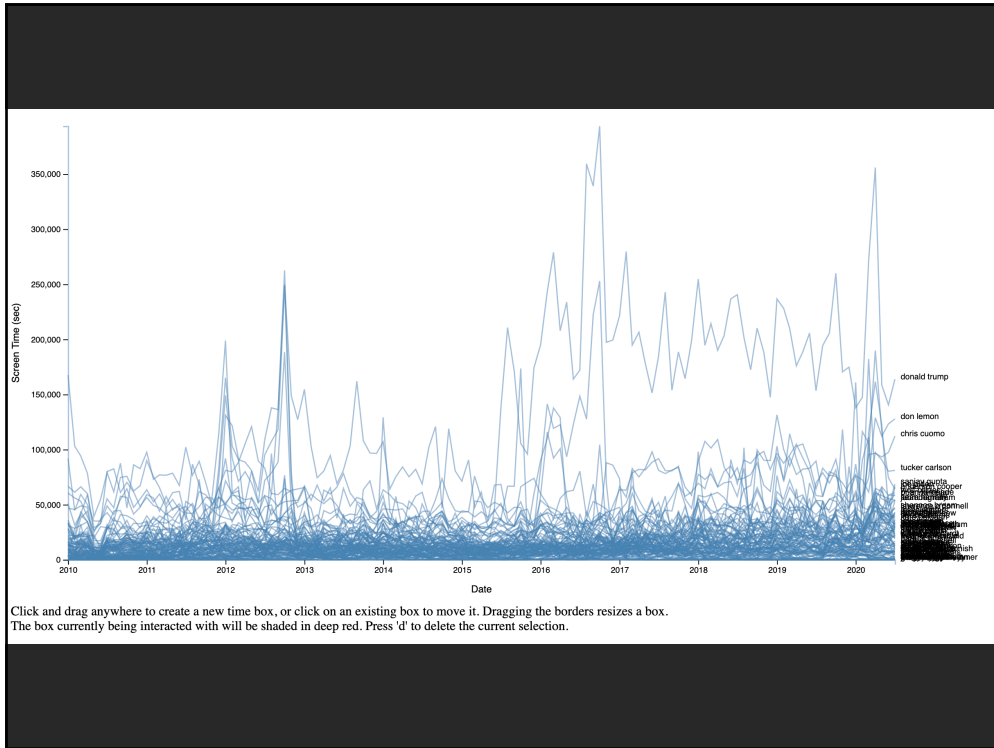
Create a **small** interactive dynamic query application similar to TimeSearcher, but for top 100 personalities on Cable TV News.

1. Implement timeboxes interface
2. Submit the application and a short write-up on canvas



Can work alone or in pairs
Due before class on **Oct 20, 2020**

28



29

Introduction to D3

30

What is D3?

D3: "Data-Driven Documents"

Data visualization built on top of HTML, CSS, JavaScript, and SVG

Pros:

- Highly-customizable**
- Developing and debugging tools**
- Documentation, resources, community**
- Integrates with the web!**

Cons:

- Very "low-level"**

31

hello-world.html

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

<body>
  Hello, world!
</body>

</html>
```

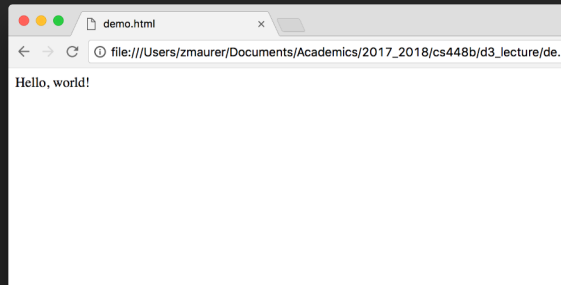
32

hello-world.html

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

<body>
  Hello, world!
</body>

</html>
```



33

hello-svg.html

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8">
  <style> /* CSS */ </style>
</head>

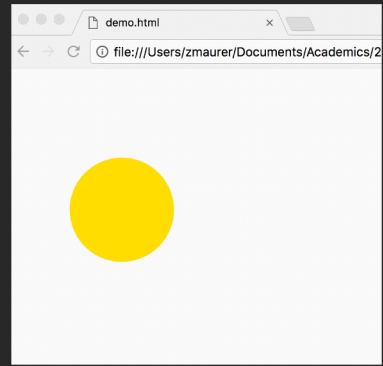
<body>
  <svg width="960" height="500">
    <circle cx='120' cy='150' r='60' style='fill: gold;'>
      <animate
        attributeName='r'
        from='2' to='80' begin='0' dur='3'
        repeatCount='indefinite' />
    </circle>
  </svg>
</body>
</html>
```

36

hello-svg.html

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<style> /* CSS */ </style>
</head>

<body>
<svg width="960" height="500">
  <circle cx='120' cy='150' r='60' style='fill:
    gold;'>
    <animate attributeName='r'
      from='2' to='80' begin='0' dur='3'
      repeatCount='indefinite' />
  </circle>
</svg>
</body>
</html>
```



37

hello-javascript.html

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<style> /* CSS */ </style>
</head>

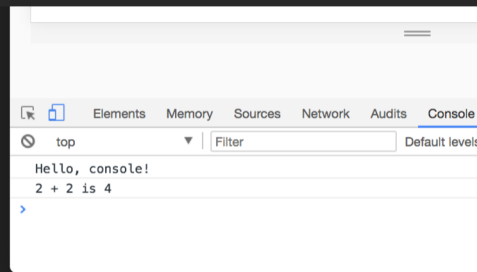
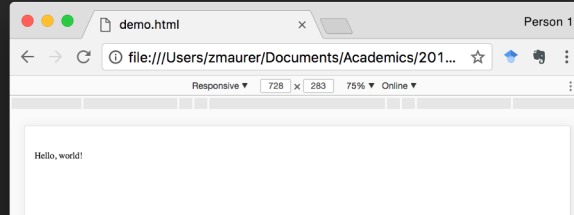
<body>
Hello, world!
<script>
  console.log("Hello, world!");
  function add2(x) {
    return x + 2;
  }
  console.log("2 + 2 is " + add2(2));
</script>
</body>
</html>
```

38

hello-javascript.html

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8">
  <style> /* CSS */ </style>
</head>

<body>
Hello, world!
<script>
  console.log("Hello, world!");
  function add2(x) {
    return x + 2;
  }
  console.log("2 + 2 is " + add2(2));
</script>
</body>
</html>
```



39

hello-d3.html

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8">
  <style> /* CSS */ </style>
</head>

<body>
  <script src="https://d3js.org/d3.v6.min.js"></script>
  <script>

    // JavaScript code that handles the logic of adding SVG elements
    // that make up the visual building blocks of your data visualization

  </script>
</body>
</html>
```

40

DOM: Document Object Model

```
<html>
  <head>

    <title></title>
  </head>
  <body>
    <h1></h1>
    <div>

      <p></p>
    </div>
  </body>
</html>
```

[Adapted from Victoria Kirst's [cs193x slides](#).]

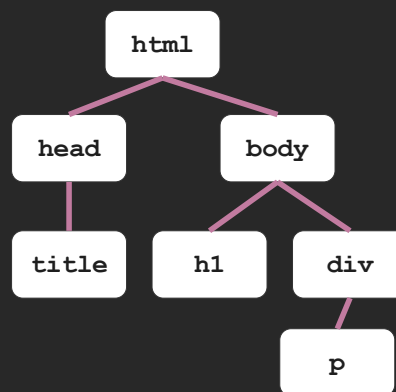
41

DOM: Document Object Model

```
<html>
  <head>

    <title></title>
  </head>
  <body>
    <h1></h1>
    <div>

      <p></p>
    </div>
  </body>
</html>
```

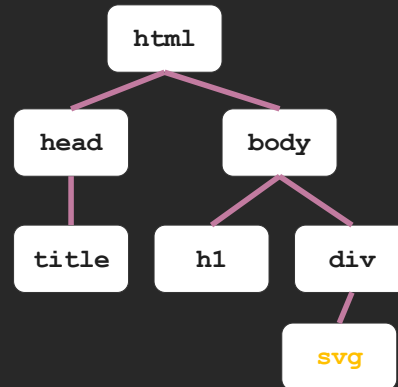


[Adapted from Victoria Kirst's [cs193x slides](#).]

42

DOM: Document Object Model

```
<html>
  <head>
    <title></title>
  </head>
  <body>
    <h1></h1>
    <div>
      <svg></svg>
    </div>
  </body>
</html>
```

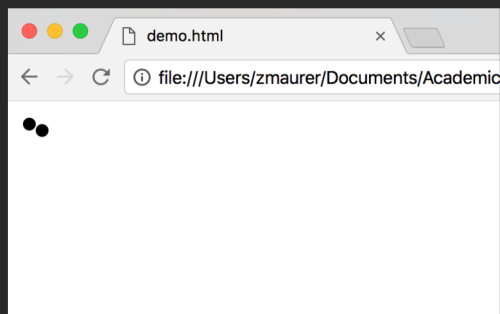


[Adapted from Victoria Kirst's [cs193x slides](#).]

43

D3: Selection

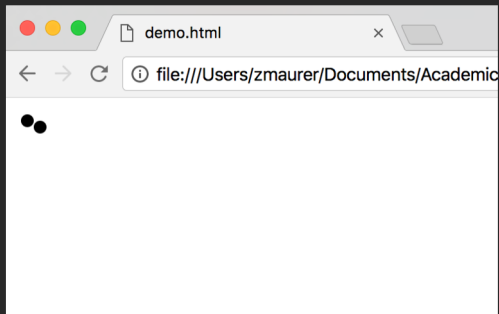
```
<html>
...
<svg width="960" height="500">
  <circle cx="10" cy="10" r="5"></circle>
  <circle cx="20" cy="15" r="5"></circle>
</svg>
```



44

D3: Selection

```
<html>
...
<svg width="960" height="500">
  <circle cx="10" cy="10" r="5"></circle>
  <circle cx="20" cy="15" r="5"></circle>
</svg>
```



```
<script>
// select all SVG circle
  elements
var circles =
  d3.selectAll("circle");
```

```
</script>
```

45

D3: Selection & Manipulation

```
<html>
...
<svg width="960" height="500">
  <circle cx="10" cy="10" r="5"></circle>
  <circle cx="20" cy="15" r="5"></circle>
</svg>
```

```
<script>
// select all SVG circle
  elements
var circles =
  d3.selectAll("circle");

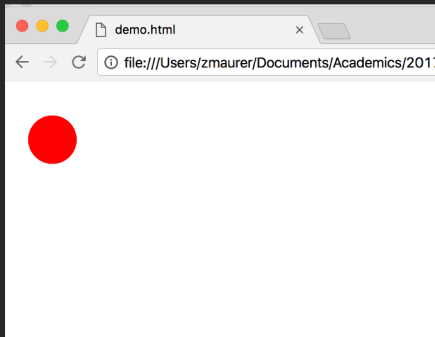
// set attributes and styles
circles.attr("cx", 40);
circles.attr("cy", 50);
circles.attr("r", 24);
circles.style("fill", "red");

</script>
```

46

D3: Selection & Manipulation

```
<html>
...
<svg width="960" height="500">
  <circle cx="10" cy="10" r="5"></circle>
  <circle cx="20" cy="15" r="5"></circle>
</svg>
```



```
<script>
// select all SVG circle
  elements
var circles =
  d3.selectAll("circle");
```

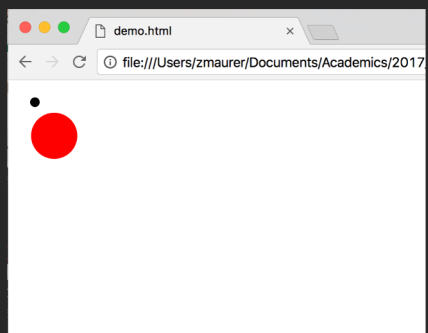
```
// set attributes and styles
circles.attr("cx", 40);
circles.attr("cy", 50);
circles.attr("r", 24);
circles.style("fill", "red");
```

```
</script>
```

47

D3: Selection & Manipulation

```
<html>
...
<svg width="960" height="500">
  <circle cx="10" cy="10" r="5"></circle>
  <circle cx="20" cy="15" r="5"></circle>
</svg>
```



```
<script>
// select all SVG circle
  elements
var circles =
  d3.select("circle");
```

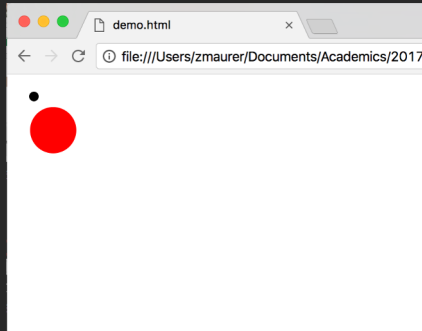
```
// set attributes and styles
circles.attr("cx", 40);
circles.attr("cy", 50);
circles.attr("r", 24);
circles.style("fill", "red");
```

```
</script>
```

49

D3: Selection & Manipulation

```
<html>
...
<svg width="960" height="500">
  <circle cx="10" cy="10" r="5"></circle>
  <circle cx="20" cy="15" r="5"></circle>
</svg>
```



```
<script>
// all together!!
d3.select("circle")
  .attr("cx", 40)
  .attr("cy", 50)
  .attr("r", 24)
  .style("fill", "red");
</script>
```

50

Binding Data & Joining DOM Elements

```
1. China: 1303182268
2. India: 1080264388
3. United States: 295734134
4. Indonesia: 218465000
5. Brazil: 186112794
6. Pakistan: 162419946
7. Bangladesh: 144319628
8. Nigeria: 128765768
9. Japan: 127417244
10. Mexico: 106202903
```

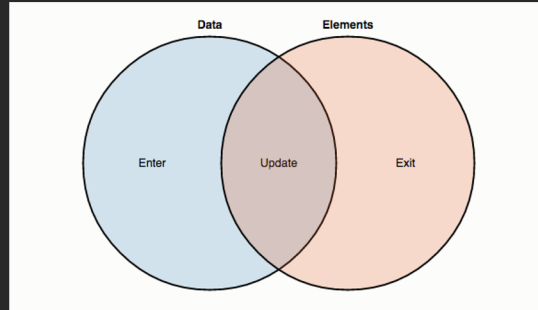
```
{
  const ol = d3.create('ol');

  ol.selectAll('li') // select all list elements (orange circle above)
  .data(listData) // bind all our data values (blue circle above)
  .join(
    enter => enter.append('li'), // append an li element for each entering item
    update => update, // do nothing with items that match an existing element
    exit => exit.remove() // remove li elements whose backing data is now gone
  )
  .text(d => `${d.country}: ${d.pop}`);

  return ol.node();
}
```

51

Binding Data & Joining DOM Elements



A *join* creates three sub-selections:

- **Enter:** selection containing placeholders for every data value that did not have a corresponding DOM element in the original selection
- **Update:** selection containing existing DOM elements that match a bound data value
- **Exit:** selection that also contains existing DOM elements, but for which a matching data value was not found

53

Binding Data & Joining DOM Elements

Exercise

Modify the *enter*, *update*, and *exit* functions in the code below such that entering items are colored green, updating items are colored blue, and exiting items are not removed but rather colored red.

```
1. China: 1303182368
2. India: 1080264388
3. United States: 295734134
4. Indonesia: 218465000
5. Brazil: 186112794
6. Pakistan: 162419946
7. Bangladesh: 144319628
8. Nigeria: 128765768
9. Japan: 127417244
10. Mexico: 106202903
```

```
undefined
{
  const ol = d3.select('ol#enter-update-exit');
  // use a new dataset, manipulable with the variables defined below
  const newData = ganninder
    .filter(d => d.year === year)
    .sort((a, b) => b.pop - a.pop)
    .slice(0, n);
  ol.selectAll('li') // select all list elements (orange circle above)
    .data(newData) // bind all our data values (blue circle above)
    .join(
      enter => enter.append('li').style('color', 'green'),
      update => update.style('color', 'blue'),
      exit => exit.remove()
    )
    .text(d => `${d.country}: ${d.pop}`);
}
```

You can use the variables below to change the elements in the *newData* extracted in the cell above to test your changes to *enter*, *update*, and *exit*. Note that we are using Observable's ability to automatically figure out the dependency structure between cells so that changes to the cells below correctly update the cells above.

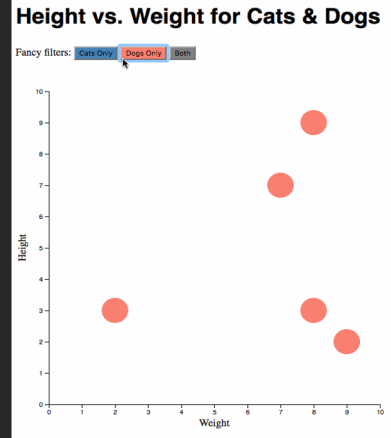
```
year = 2085
year = 2085
n = 10
n = 10
```

54

Let's make a scatterplot



```
id,animal,weight,height,name
1,cat,10,3,phyllis
2,cat,3,3,oreo
3,cat,9,9,sam
4,cat,3,5,dog
5,cat,6,5,fred
6,cat,5,6,jane
7,cat,1,8,esmerelda
8,dog,9,2,garfield
9,dog,8,9,alpha
10,dog,7,7,omega
11,dog,2,3,zeta
12,dog,8,3,cupcake
```



<https://observablehq.com/@stanfordvis/lets-make-a-scatterplot>