

Intro to D3

Slides adapted from...

Maneesh Agrawala

Jessica Hullman

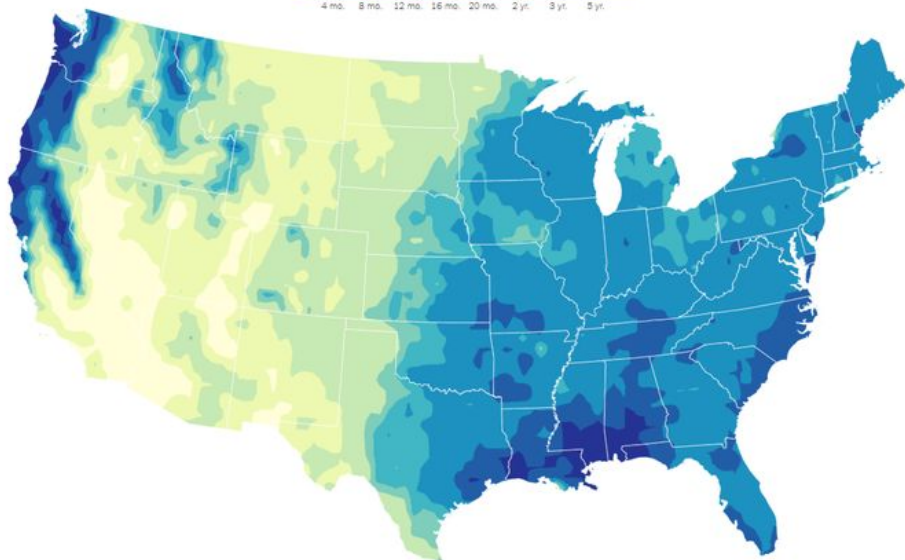
Ludwig Schubert

Peter Washington

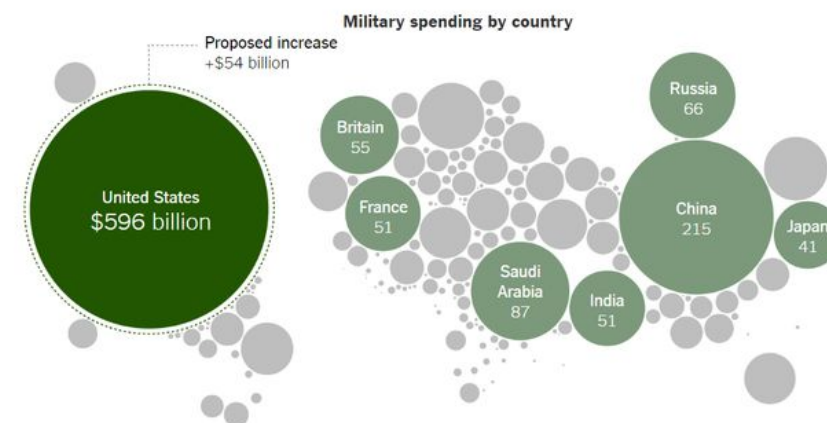
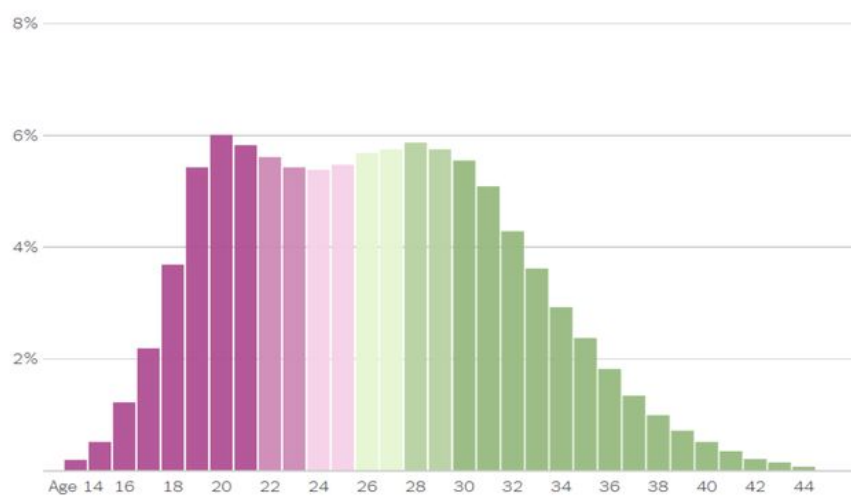
Alec Glassford and Zach Maurer

Gracie Young and Vera Lin

How long it took 50 inches of rain to accumulate



Ages of first-time mothers in **2016**

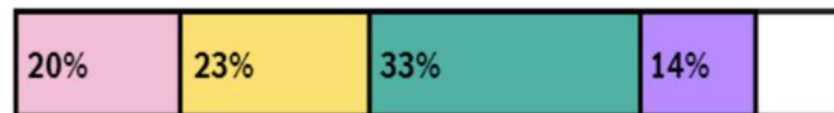


2018 music consumption by genre...

INCLUDING STREAMING

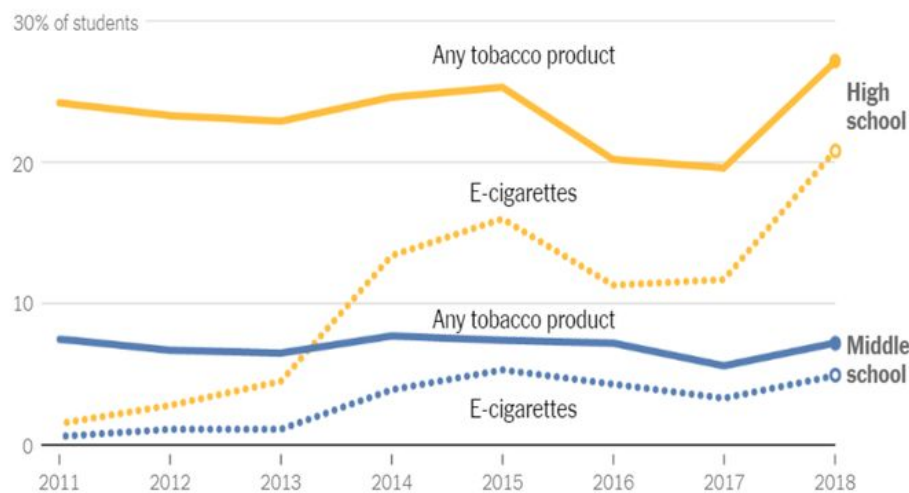


TRADITIONAL SALES ONLY



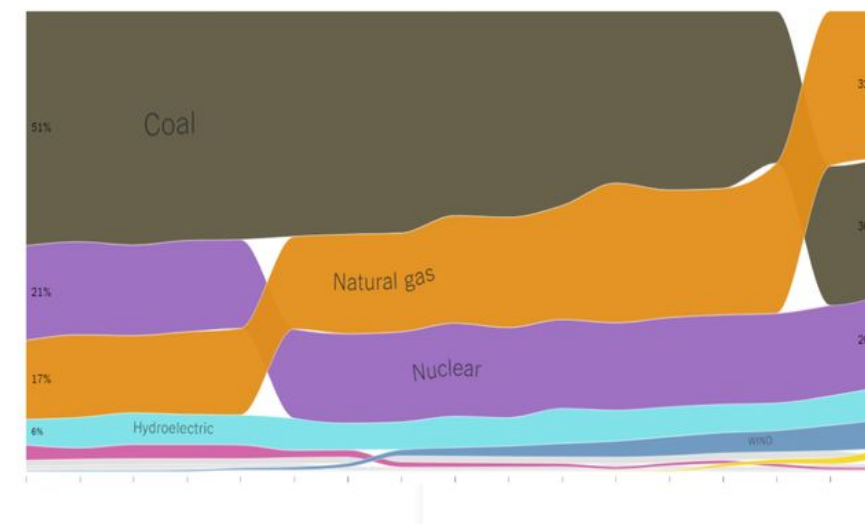
R&B/HIP-HOP
 POP
 ROCK
 COUNTRY

Tobacco Consumption Among Students



How the United States generated electricity from 2001 to 2017

Percentage of power produced from each energy source



D3.js is a JavaScript library for manipulating documents based on data. D3 helps you bring data to life using HTML, SVG, and CSS. D3's emphasis on web standards gives you the full capabilities of modern browsers without tying yourself to a proprietary framework, combining powerful visualization components and a data-driven approach to DOM manipulation.

D3.js is a **JavaScript library** for manipulating documents based on data. D3 helps you bring data to life using HTML, SVG, and CSS. D3's emphasis on web standards gives you the full capabilities of modern browsers without tying yourself to a proprietary framework, combining powerful visualization components and a data-driven approach to DOM manipulation.

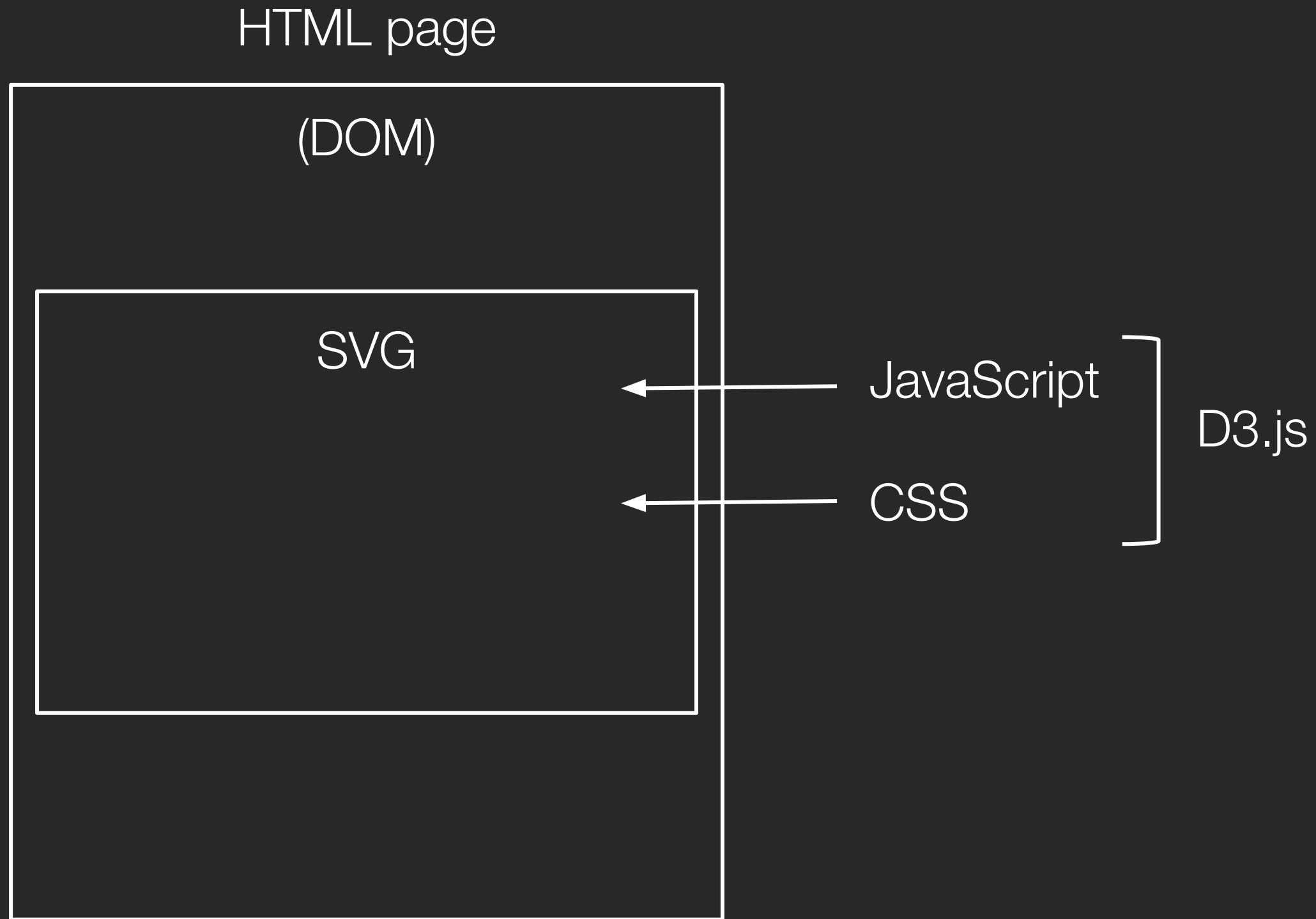
D3.js is a JavaScript library for **manipulating documents based on data**. D3 helps you bring data to life using HTML, SVG, and CSS. D3's emphasis on web standards gives you the full capabilities of modern browsers without tying yourself to a proprietary framework, combining powerful visualization components and a data-driven approach to DOM manipulation.

D3.js is a JavaScript library for manipulating documents based on data. D3 helps you bring data to life using **HTML, SVG, and CSS**. D3's emphasis on web standards gives you the full capabilities of modern browsers without tying yourself to a proprietary framework, combining powerful visualization components and a data-driven approach to DOM manipulation.

What we will cover today (and Wednesday)...

very customizable

built for web



I. HTML

```
<html>
```

```
  <head>
```

```
  </head>
```

```
  <body>
```

```
    Hello world! <br>
```

```
  </body>
```

```
</html>
```

I. HTML

```
<html>
```

```
  <head>
```

```
</head>
```

```
<body>
```

```
  Hello world!<br>
```

```
  <svg id='vis'></svg>
```

```
</body>
```

```
</html>
```

I. HTML

```
<html>
```

```
  <head>
```

```
    <link rel='stylesheet'  
         type='text/css'  
         href='styles.css'>
```

```
  </head>
```

```
  <body>
```

```
    Hello world!<br>
```

```
    <svg id='vis'></svg>
```

```
  </body>
```

```
</html>
```

I. HTML

```
<html>
```

```
  <head>
```

```
    <link rel='stylesheet'  
          type='text/css'  
          href='styles.css'>
```

```
  </head>
```

```
  <body>
```

```
    Hello world!<br>
```

```
    <svg id='vis'></svg>
```

```
  </body>
```

```
  <script src='makevis.js'></script>
```

```
</html>
```

I. HTML

```
<html>
```

```
  <head>
```

```
    <link rel='stylesheet'  
          type='text/css'  
          href='styles.css'>
```

```
    <script src='https://d3js.org/d3.v5.min.js'></script>
```

```
  </head>
```

```
  <body>
```

```
    Hello world!<br>
```

```
    <svg id='vis'></svg>
```

```
  </body>
```

```
  <script src='makevis.js'></script>
```

```
</html>
```

Running your code

```
cd path/to/directory
```

```
python -m http.server (3.x)
```

```
python -m SimpleHTTPServer (2.x)
```

shell

```
http://localhost:8000/
```

```
http://localhost:8000/mainpage.html
```

browser

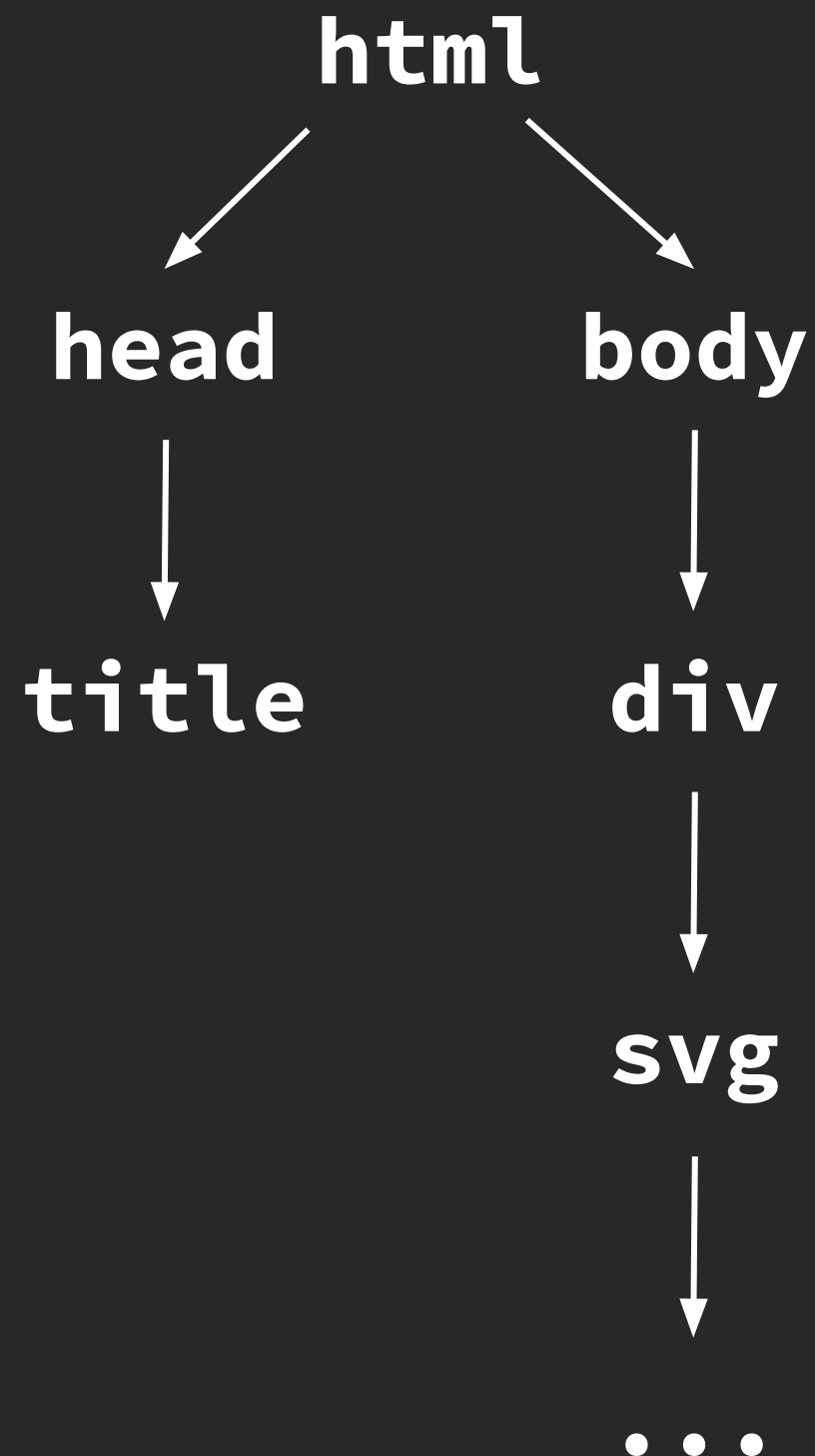
II. JavaScript

II. JavaScript

DOM (Document Object Model)

II. JavaScript

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



II. JavaScript

Add to and manipulate DOM with D3

II. JavaScript

```
<script>
```

```
  var svg = d3.select('#vis');
```

```
</script>
```

II. JavaScript

```
<script>
```

```
  var svg = d3.select('#vis');
```

```
  var circle = svg.append('circle');
```

```
</script>
```

II. JavaScript

```
<script>
```

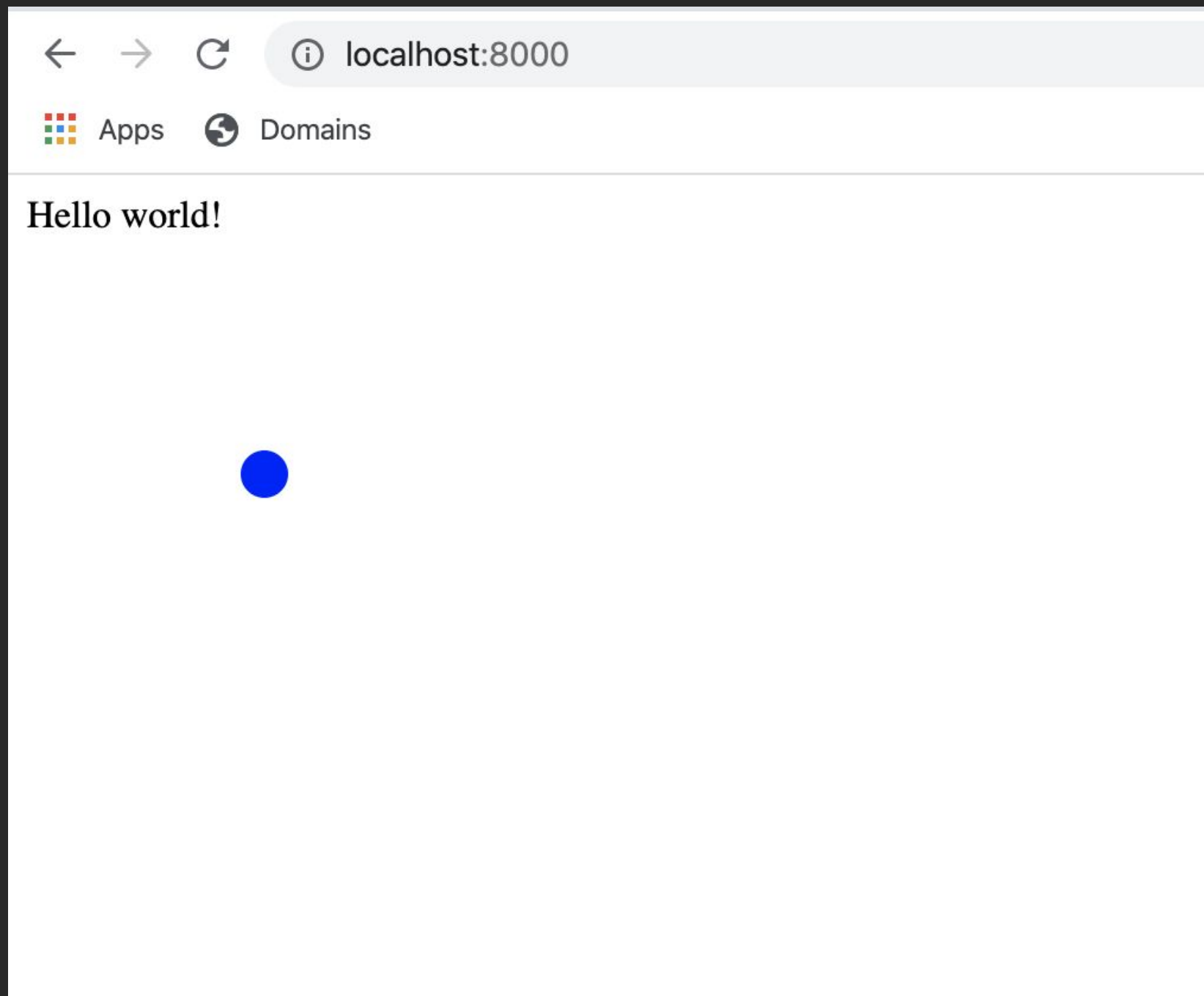
```
  var svg = d3.select('#vis');  
  var circle = svg.append('circle');
```

```
  circle.attr('r', 10)  
          .attr('cx', 100)  
          .attr('cy', 100)  
          .style('fill', 'blue');
```

```
</script>
```

II. JavaScript

what shows up



II. JavaScript

id	animal	weight	height
1	cat	10	3
2	cat	3	3
3	cat	9	9
4	cat	3	5
5	cat	6	5
6	cat	5	6
7	cat	1	8
8	dog	9	2
9	dog	8	9
10	dog	7	7
11	dog	2	3
12	dog	8	3

```
<html>
```

```
  <head>
```

```
    <script src='https://d3js.org/d3.v5.min.js'></script>
```

```
  </head>
```

```
  <body>
```

```
    This is our plot.
```

```
  </body>
```

```
  <script>
```

```
  </script>
```

```
</html>
```



```
<html>

  <head>
    <script src='https://d3js.org/d3.v5.min.js'></script>
  </head>

  <body>
    This is our plot.
    <svg id='vis'></svg>
  </body>

  <script>

  </script>

</html>
```

```
<html>

  <head>
    <script src='https://d3js.org/d3.v5.min.js'></script>
  </head>

  <body>
    This is our plot.
    <div id='vis'></div>
  </body>

  <script>
    /* Here we will make our visualization magic */
  </script>

</html>
```

```
<script>
```

```
  var plotWidth = 800;
```

```
  var plotHeight = 600;
```

```
  var buf = 15;
```

```
</script>
```

```
<script>
```

```
  var plotWidth = 800;
```

```
  var plotHeight = 600;
```

```
  var buf = 15;
```

```
  var svg = d3.select('#vis').append('svg')
```

```
    .attr('width', plotWidth)
```

```
    .attr('height', plotHeight);
```

```
</script>
```

...

```
var allData;  
d3.csv('animals.csv', function(d) {  
    // process data
```

```
}).then(function(data) {  
    // do something with data
```

```
});
```

```
</script>
```

...

```
var allData;  
d3.csv('animals.csv', function(d) {  
  return {  
    id: +d.id,  
    species: d.animal,  
    weight: +d.weight,  
    height: +d.height  
  };  
}).then(function(data) {  
  // do something with data  
  
});
```

```
</script>
```

...

```
var allData;  
d3.csv('animals.csv', function(d) {  
  return {  
    id: +d.id,  
    species: d.animal,  
    weight: +d.weight,  
    height: +d.height  
  };  
}).then(function(data) {  
  allData = data;  
  drawAxes();  
  drawPoints(data);  
});
```

```
</script>
```

...

```
function drawAxes() {  
  var xaxis = svg.append('line')  
    .attr('x1', buf)  
    .attr('x2', plotWidth - buf)  
    .attr('y1', plotHeight - buf)  
    .attr('y2', plotHeight - buf)  
    .attr('stroke', black);  
  var yaxis = svg.append('line')  
    .attr('x1', buf)  
    .attr('x2', buf)  
    .attr('y1', buf)  
    .attr('y2', plotHeight - buf)  
    .attr('stroke', black);  
}  
</script>
```


...

```
function drawPoints(data) {  
    var points = svg.selectAll('circle');
```

```
}
```

```
</script>
```


...

```
function drawPoints(data) {  
  var points = svg.selectAll('circle');  
  var pointsData = points.data(data, d=>d.id);  
  
}
```

```
</script>
```

...

```
function drawPoints(data) {  
  var points = svg.selectAll('circle');  
  var pointsData = points.data(data, d=>d.id);  
  var circles = pointsData.enter().append('circle')  
  
}
```

```
</script>
```

...

```
function drawPoints(data) {  
  var points = svg.selectAll('circle');  
  var pointsData = points.data(data, d=>d.id);  
  var circles = pointsData.enter().append('circle')  
    .attr('r', 5)  
    .attr('cx', function(d) { return x(d.weight); })  
    .attr('cy', function(d) { return y(d.height); })  
  
}
```

```
</script>
```

...

```
function drawPoints(data) {  
  var points = svg.selectAll('circle');  
  var pointsData = points.data(data, d=>d.id);  
  var circles = pointsData.enter().append('circle')  
    .attr('r', 5)  
    .attr('cx', function(d) { return x(d.weight); })  
    .attr('cy', function(d) { return y(d.height); })  
    .style('fill', function(d) {  
      return d.species == 'cat' ? 'orange' : 'blue';  
    })  
}
```

```
</script>
```

...

```
function drawPoints(data) {  
  var points = svg.selectAll('circle');  
  var pointsData = points.data(data, d=>d.id);  
  var circles = pointsData.enter().append('circle')  
    .attr('r', 5)  
    .attr('cx', function(d) { return x(d.weight); })  
    .attr('cy', function(d) { return y(d.height); })  
    .style('fill', function(d) {  
      return d.species == 'cat' ? 'orange' : 'blue';  
    })  
  pointsData.exit().remove();  
}
```

</script>

...

```
var x = d3.scaleLinear()  
  .domain([0, 11])  
  .range([buf, plotWidth - buf]);
```

```
var y = d3.scaleLinear()  
  .domain([0, 10])  
  .range([plotHeight - buf, buf]);
```

```
</script>
```


...

```
function drawPoints(data) {
```

...

```
  circles
```

...

```
}
```

...

```
function drawPoints(data) {
```

...

```
  circles
```

```
    .on('mouseover', function(d) {
```

```
    })
```

...

```
}
```

...

```
function drawPoints(data) {  
  ...  
  circles  
    .on('mouseover', function(d) {  
      svg.append('text')  
        .attr('x', x(d.weight) + 10)  
        .attr('y', y(d.height) + 5)  
        .text(d.species);  
    })  
}
```

...

```
}
```

...

```
function drawPoints(data) {  
  ...  
  circles  
    .on('mouseover', function(d) {  
      svg.append('text')  
        .attr('x', x(d.weight) + 10)  
        .attr('y', y(d.height) + 5)  
        .text(d.species);  
    })  
    .on('mouseout', function(d) {  
      svg.selectAll('text').remove();  
    });  
  ...  
}
```

```
<html>

  <head>
    <script src='https://d3js.org/d3.v5.min.js'></script>
  </head>

  <body>
    This is our plot.
    <div id='vis'></div>

  </body>

  <script>
    /* Here we will make our visualization magic */

  </script>
</html>
```

```
<html>

  <head>
    <script src='https://d3js.org/d3.v5.min.js'></script>
  </head>

  <body>
    This is our plot.
    <div id='vis'></div>
    <button data-filter='dog'>Remove cats</button>
  </body>

  <script>
    /* Here we will make our visualization magic */

    </script>
</html>
```

```
<html>

  <head>
    <script src='https://d3js.org/d3.v5.min.js'></script>
  </head>

  <body>
    This is our plot.
    <div id='vis'></div>
    <button data-filter='dog'>Remove cats</button>
  </body>

  <script>
    var button = document.querySelector('button');
    button.addEventListener('click', function() {
      var newData = allData.filter(d =>
        d.species == button.dataset.filter);
      drawPoints(newData);
    });
  </script>
</html>
```

Tips for D3

Change not showing up? Clear your cache.

Use the console to see what's happening.

Too slow? Made a mistake? Kill and restart your http server.

Utilize HTML to add input fields.

Can use pure CSS if something doesn't have to change.

Learn by example.

What D3 can do for you

Scales

Axes

Paths, shapes, areas

Maps

Hierarchical layouts

... all have subpackages dedicated to them.

Resources

D3 API Reference v5.0

<https://github.com/d3/d3/blob/master/API.md>

<https://github.com/d3>



D3

Data-Driven Documents

San Francisco, CA <https://d3js.org>

Repositories 52

Packages

People 5

Pinned repositories

d3

Bring data to life with SVG, Canvas and HTML.



JavaScript ★ 89.7k 🍴 21.7k

d3-shape

Graphical primitives for visualization, such as lines and areas.

JavaScript ★ 1.9k 🍴 237

d3-scale

Encodings that map abstract data to visual representation.

JavaScript ★ 1.2k 🍴 235

d3-geo-projection

Extended geographic projections for d3-geo.

JavaScript ★ 794 🍴 171

d3-force

Force-directed graph layout using velocity Verlet integration.

JavaScript ★ 756 🍴 237

d3-hierarchy

2D layout algorithms for visualizing hierarchical data.

JavaScript ★ 459 🍴 217

Resources

d3-csv

d3-scale

d3-geo

Lots of examples at...

<https://github.com/d3/d3/wiki/Gallery>

<https://observablehq.com/@d3>

Helpful tiny examples (i.e. making shapes) at...

<https://www.dashingd3js.com/table-of-contents>