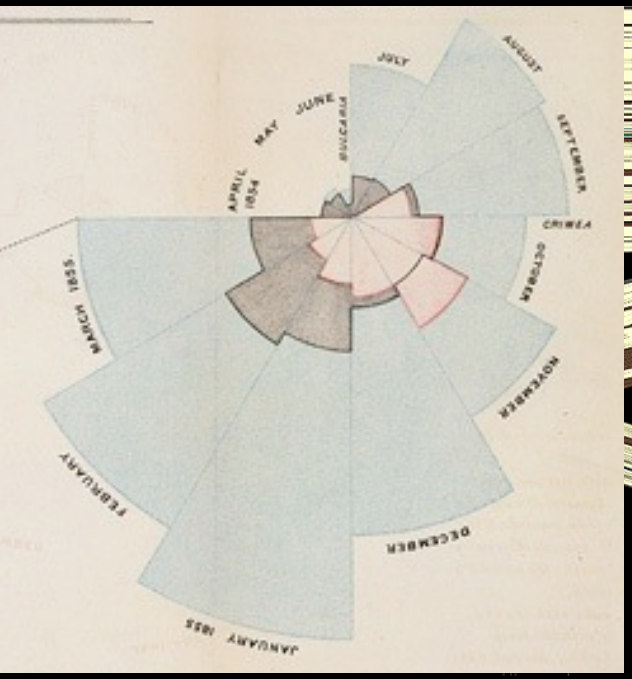
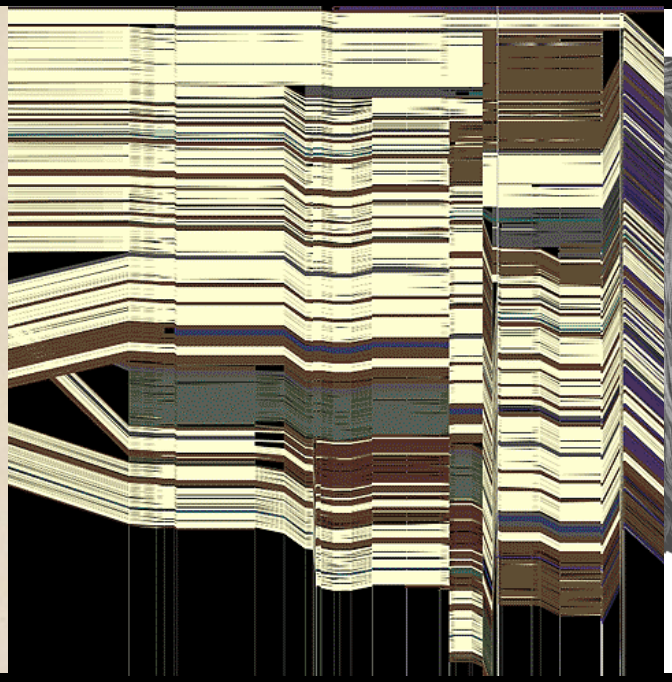
CSE 412 - Intro to Data Visualization

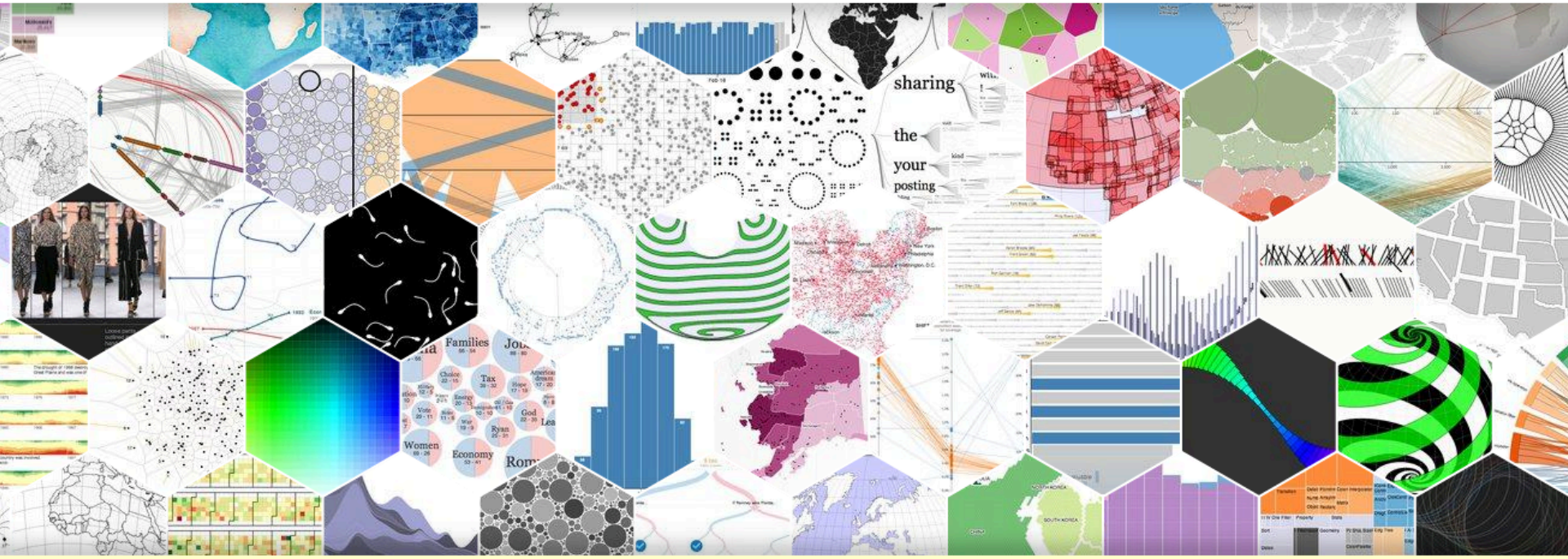
Intro to D3.js





Jane Hoffswell University of Washington

Data-Driven Documents



Like visualization and creative coding? Try interactive JavaScript notebooks in **Observable!**

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.

Download the latest version (6.5.0) here:

- d3.zip

To link directly to the latest release, copy this snippet:

See more examples
Chat with the community
Follow announcements
Report a bug
Ask for help

$\mathbb{D}^3$: Data-Driven Documents

Michael Bostock, Vadim Ogievetsky and Jeffrey Heer

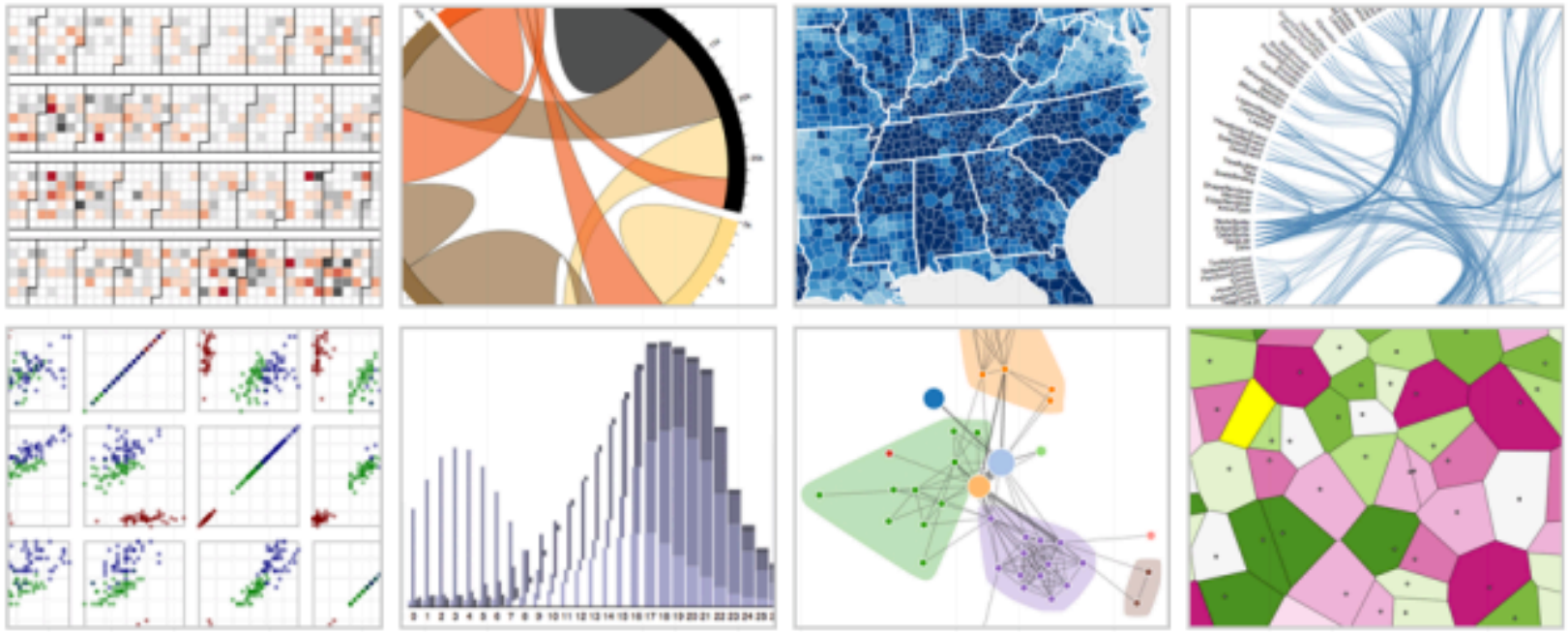


Fig. 1. Interactive visualizations built with D3, running inside Google Chrome. From left to right: calendar view, chord diagram, choropleth map, hierarchical edge bundling, scatterplot matrix, grouped & stacked bars, force-directed graph clusters, Voronoi tessellation.

Abstract—Data-Driven Documents (D3) is a novel representation-transparent approach to visualization for the web. Rather than hide the underlying scenegraph within a toolkit-specific abstraction, D3 enables direct inspection and manipulation of a native representation: the standard *document object model* (DOM). With D3, designers selectively bind input data to arbitrary document elements, applying dynamic transforms to both generate and modify content. We show how representational transparency improves expressiveness and better integrates with developer tools than prior approaches, while offering comparable notational efficiency and retaining powerful declarative components. Immediate evaluation of operators further simplifies debugging and allows iterative development. Additionally, we demonstrate how D3 transforms naturally enable animation and interaction with dramatic performance improvements over intermediate representations.

Index Terms—Information visualization, user interfaces, toolkits, 2D graphics.

Introduction

D3 allows you to bind arbitrary data to a Document Object Model (DOM), and then apply data-driven transformations to the document. For example, you can use D3 to generate an HTML table from an array of numbers. Or, use the same data to create an interactive SVG bar chart with **smooth transitions and interaction**.

D3 is not a monolithic framework that seeks to provide every conceivable feature. Instead, D3 solves the crux of the problem: **efficient manipulation of documents based on data**. This avoids proprietary representation and affords extraordinary flexibility, exposing the **full capabilities of web standards** such as HTML, SVG, and CSS. With minimal overhead, D3 is extremely fast, supporting large datasets and dynamic behaviors for interaction and animation. D3's functional style allows code reuse through a diverse collection of official and community-developed modules.

Quoted from the D3.js homepage, color emphasis added here.

Week 6 Topics:

Today - Intro to D3.js
Overview of important D3 concepts.

Thursday Quiz Section - D3 Tutorial, Part 1
Hands on experience with a static D3 chart.

Friday Lecture - D3 Tutorial, Part 2
Hands on experience with interaction & animation.

Week 7, Thursday Quiz Section: HTML/CSS/GitHub
Hands on experience with web development.

Zoom Poll:
Familiarity with D3/HTML/CSS

D3 is a declarative language.

What is a Declarative Language?

Programming by describing *what*, not *how*

Separate **specification** (*what you want*) from **execution** (*how it should be computed*)

In contrast to **imperative programming**, where you must give explicit steps.

Declarative Programming in D3

```
d3.selectAll("p").style("color", "blue");
```

"I want all paragraphs to have the color blue."

Declarative Programming in D3

```
d3.selectAll("p").style("color", "blue");
```

"I want all paragraphs to have the color blue."

Compared to imperative programming:

```
var paragraphs = document.getElementsByTagName("p");
for (var i = 0; i < paragraphs.length; i++) {
  var paragraph = paragraphs.item(i);
  paragraph.style.setProperty("color", "blue", null);
}
```

"Get all the paragraphs, then take each one one-at-a-time and set the color to be blue."

Why Declarative Languages?

Faster iteration. Less code. Larger user base.

Better visualization. *Smart defaults.*

Reuse. *Write-once, then re-apply.*

Performance. *Optimization, scalability.*

Portability. *Multiple devices, renderers, inputs.*

Programmatic generation.
Write programs which output visualizations.
Automated search & recommendation.

D3 leverages web standards.

DOM / HTML / CSS / SVG

Click title elements for web resources.

"D3 allows you to bind arbitrary data to a **D**ocument **O**bject **M**odel"

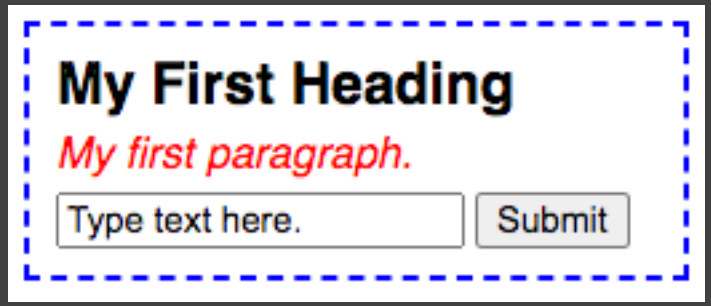
HTML: Structure of items on a page.

```
<h1>My First Heading</h1>
<p>My first paragraph.</p>
<input type="text" value="Type text here.">
<input type="submit">
```

CSS: Visual style of items on a page.

```
body {
  border: blue 2px dashed;
  padding: 10px;
  width: 225px;
}
h1 {
  font-family: sans-serif;
  font-size: 16pt;
  margin: 0px;
}
```

```
p {
  color: red;
  font-family: sans-serif;
  font-style: italic;
  margin: 5px 0px;
}
```



DOM / HTML / CSS / SVG

Click title elements for web resources.

"D3 allows you to bind arbitrary data to a **D**ocument **O**bject **M**odel"

HTML for Course Website:

```
<!DOCTYPE html>
<html xml:lang="en" lang="en">
▶ <head>…</head>
▼ <body data-spy="scroll" data-target=".navbar" id="about" style>
  ▼ <div class="content">
    ▶ <div class="title">…</div>
    ▶ <div class="band">…</div>
    ▶ <script>…</script>
      <br>
    ▶ <div class="sidebar">…</div>
… ▼ <div class="article"> == $0
      ▶ <p>…</p>
      ▶ <p>…</p>
      ▶ <p>…</p>
        <h2 id="texts">Textbooks</h2>
      ▶ <ul>…</ul>
        <h2 id="objectives">Learning Goals & Objectives</h2>
        "
        This course is designed to provide students with the foundatio
        data visualization. By the end of the course, students will ha
        "
        <br>
      ▶ <ul>…</ul>
        <h2 id="schedule">Schedule & Readings</h2>
      ▶ <script type="text/javascript">…</script>
        <h3 id="week1">Week 1</h3>
      ▶ <div class="day" id="lec-value">…</div>
      ▶ <div class="day" id="lec-dmodels">…</div>
      ▶ <div class="day" id="qz-wrangling">…</div>
      ▶ <div class="day" id="lec-imodels">…</div>
        <h3 id="week2">Week 2</h3>
      ▶ <div class="day" id="lec-eda">…</div>
      ▶ <div class="day" id="lec-eda2">…</div>
      ▶ <div class="day" id="qz-tableau">…</div>
      ▶ <div class="day" id="lec-a1">…</div>
```

CSS for Course Website:

```
411 .section small {
412   padding: 0.5px 10.5px;
413   background: #7533f4;
414 }
415
416 .optional small {
417   color: #666;
418 }
419
420 li.assignment, li.duetoday, li.section {
421   color: white;
422 }
423
424 .duedate {
425   font-size: 0.85em;
426   color: red;
427 }
428
429 .duedate small {
430   position: relative;
431   top: -.15em;
432   font-size: 0.75em;
433   background: red;
434   padding: 0.5px 3.5px;
435   border-radius: 4px;
436   color: white;
437 }
```

Hint: You can use the Chrome Developer Tools to view webpage details, inspect or modify the structure, and debug using the JavaScript console. From Chrome, select "View" > "Developer" > "Inspect Elements" to see the HTML.

DOM / HTML / CSS / SVG

Click title elements for web resources.

"D3 allows you to bind arbitrary data to a **D**ocument **O**bject **M**odel"

SVG: Scalable Vector Graphics - shapes and lines!

```
<svg version="1.1"
     baseProfile="full"
     width="300" height="200"
     xmlns="http://www.w3.org/2000/svg">

  <rect width="100%" height="100%" fill="red" />

  <circle cx="150" cy="100" r="80" fill="green" />

  <text x="150" y="125" font-size="60" text-anchor="middle" fill="white">SVG</text>

</svg>
```

The core abstraction
in D3 is a *selection*.

D3 Selections

The core abstraction in D3 is a ***selection***.

D3 Selections

The core abstraction in D3 is a ***selection***.

```
// Add and configure an SVG element (<svg width="500" height="300">)
var svg = d3.create("svg") // add new SVG to page body
   .attr("width", 500)     // set SVG width to 500px
   .attr("height", 300);   // set SVG height to 300px
```

D3 Selections

The core abstraction in D3 is a ***selection***.

```
// Add and configure an SVG element (<svg width="500" height="300">)
var svg = d3.create("svg") // add new SVG to page body
   .attr("width", 500)     // set SVG width to 500px
   .attr("height", 300);   // set SVG height to 300px

// Select & update existing rectangles contained in the SVG element
svg.selectAll("rect")                 // select all SVG rectangles
   .attr("width", 100)                // set rect widths to 100px
   .style("fill", "steelblue");       // set rect fill colors
```

D3 allows you to bind arbitrary data to the DOM.

Data Binding

Selections can ***bind*** **data and DOM elements**.

```
var values = [ {…}, {…}, {…}, … ]; // input data as JS objects
```

Data Binding

Selections can ***bind*** **data and DOM elements**.

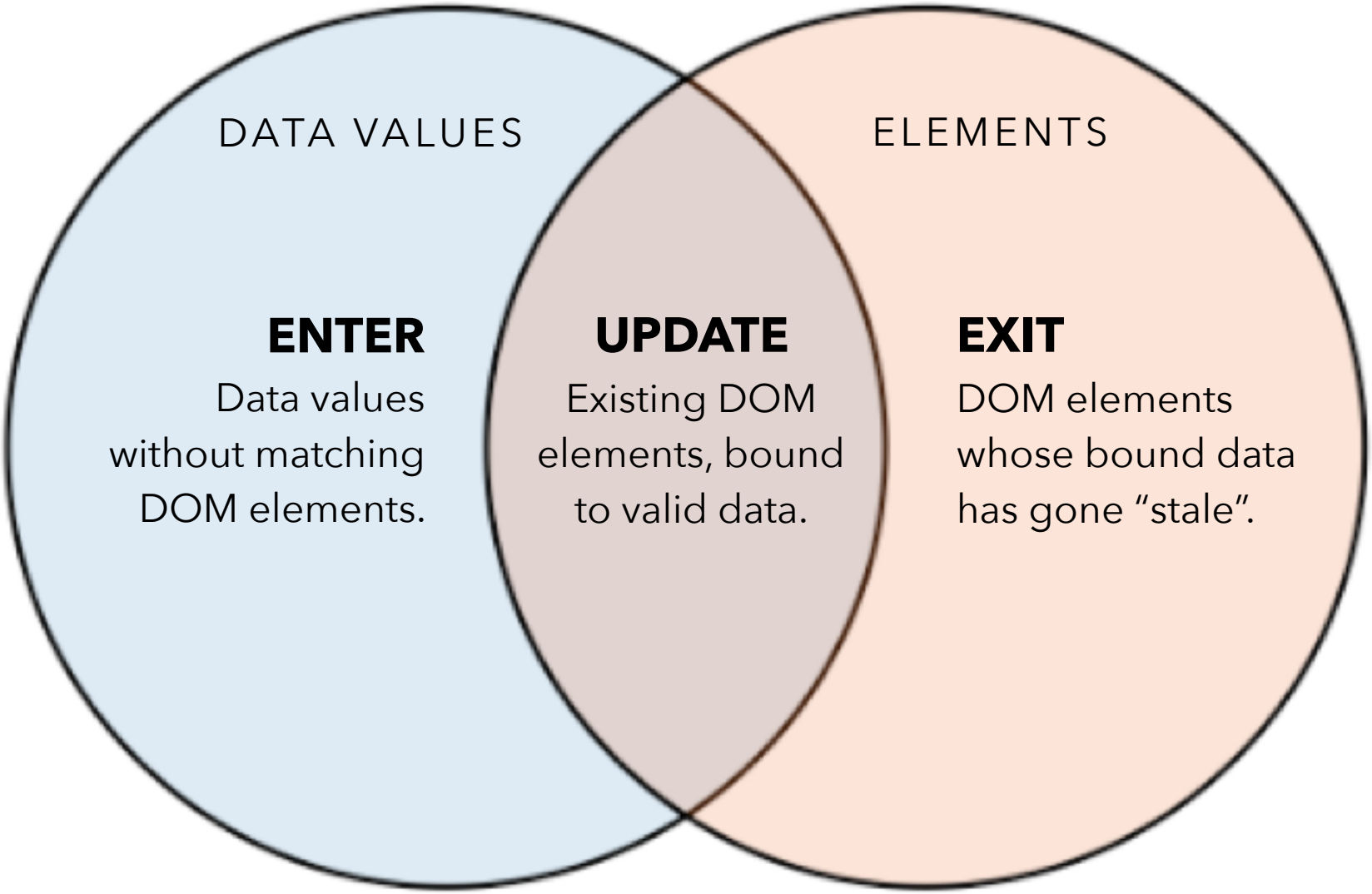
```
var values = [                                  // input data as JS objects
    {"x": 0, "y": 28},
    {"x": 1, "y": 55},
    {"x": 2, "y": 43},
    {"x": 3, "y": 91},
    {"x": 4, "y": 81},
    {"x": 5, "y": 53},
    {"x": 6, "y": 19},
    {"x": 7, "y": 87},
    {"x": 8, "y": 52}
];
```

Data Binding

Selections can ***bind*** **data and DOM elements**.

```
var values = [ {…}, {…}, {…}, … ]; // input data as JS objects

// Select SVG rectangles and bind them to data values.
var bars = svg.selectAll("rect.bars").data(values);
```

The Data Join

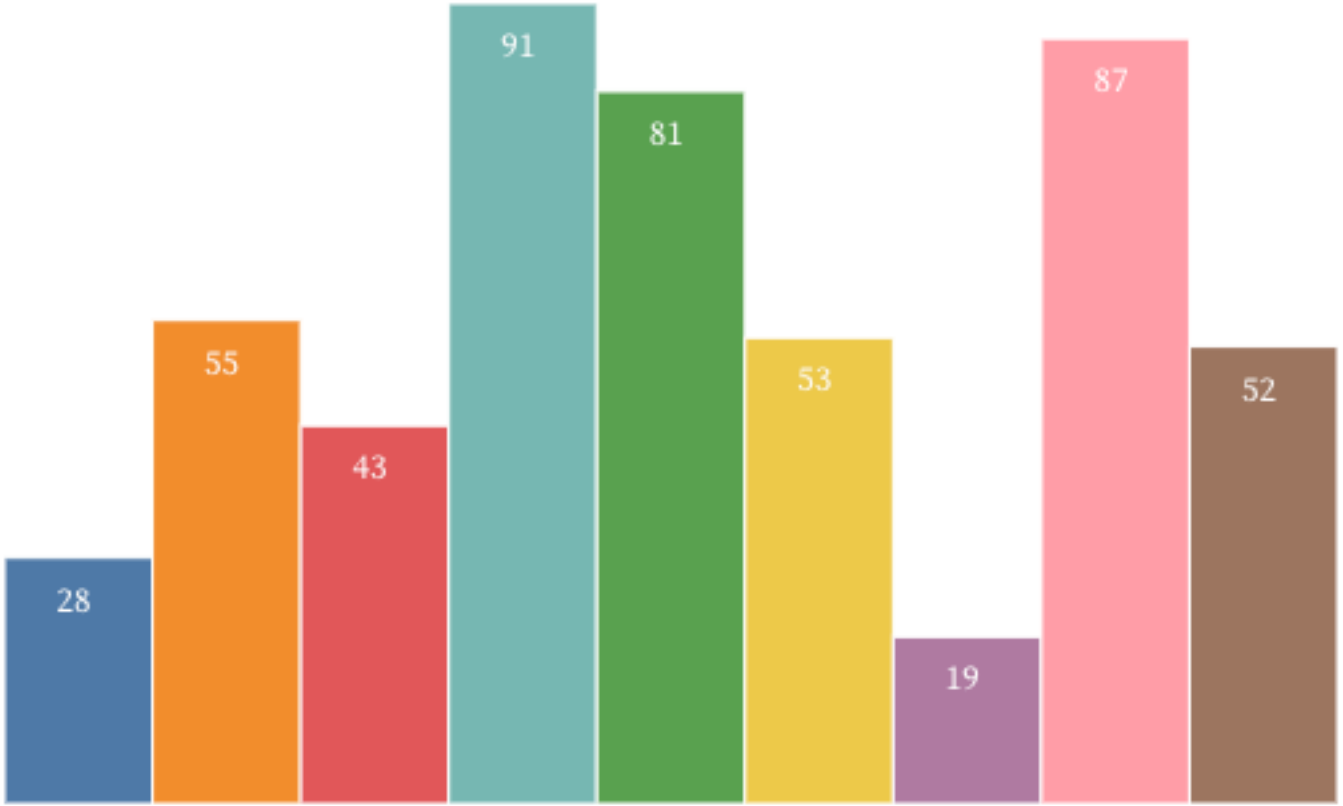


Data Binding

Selections can ***bind*** **data and DOM elements**.

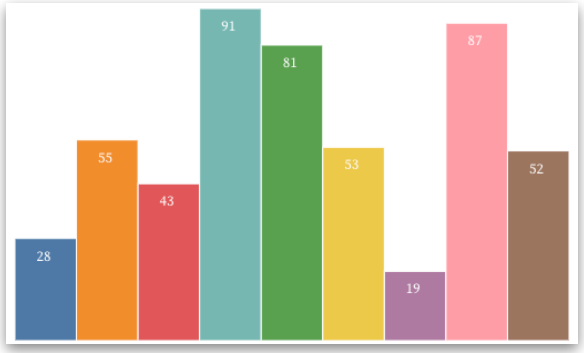
```
var values = [ {…}, {…}, {…}, … ]; // input data as JS objects

// Select SVG rectangles and bind them to data values.
var bars = svg.selectAll("rect.bars").data(values)
    .join(
        // create new SVG rect marks with class "bars"
        enter => enter.append("rect").attr("class","bars"),
        // update the existing marks to change their style
        update => update,
        // remove outdated marks from the view
        exit => exit.remove()
    )
```


▼<svg width="500" height="300"> == $0
<rect class="bars"></rect>
<rect class="bars"></rect>
<rect class="bars"></rect>
<rect class="bars"></rect>
<rect class="bars"></rect>
<rect class="bars"></rect>
<rect class="bars"></rect>
<rect class="bars"></rect>
<rect class="bars"></rect>
</svg>

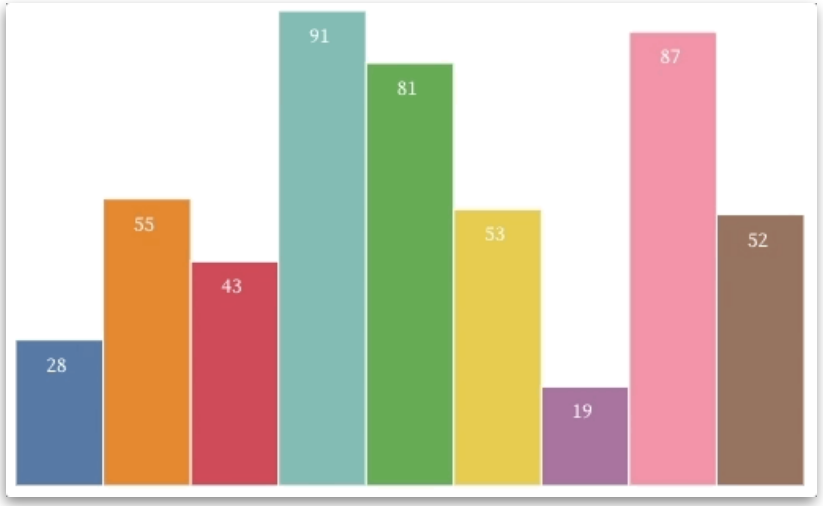

28
55
43
91
81
53
19
87
52

Updating Data Example

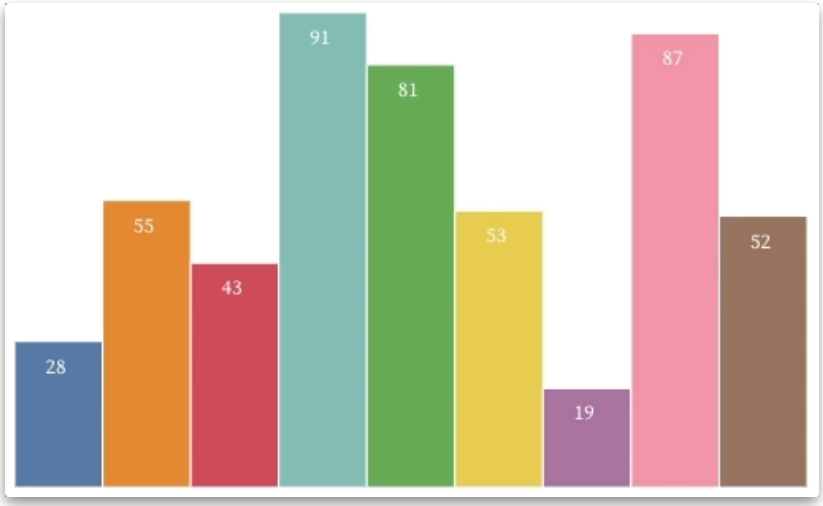
```
// round the y value down to the tens place (e.g., 58 => 50, 91 => 90)
function tensplace(array) {
  return array.map(function(obj) {
    return {
      "x": obj.x,
      "y": obj.y - obj.y%10
    };
  });
}


// randomly shuffle the order of the input array
function shuffle(array) {…}


// update our data values
values = shuffle(tensplace(values))
```



`shuffle(tensplace(values))`



(a)



(b)

Data Binding

Selections can ***bind*** **data and DOM elements**.

```
var values = [ {…}, {…}, {…}, … ]; // input data as JS objects

// Select SVG rectangles and bind them to data values.
var bars = svg.selectAll("rect.bars").data(values)
    .join(
        // create new SVG rect marks with class "bars"
        enter => enter.append("rect").attr("class","bars"),
        // update the existing marks to change their style
        update => update,
        // remove outdated marks from the view
        exit => exit.remove()
    )
```

Data Binding

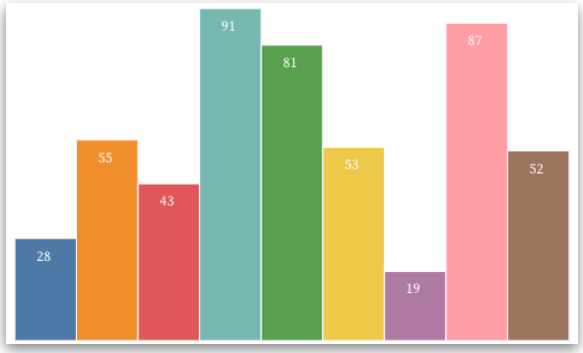
Selections can ***bind*** **data and DOM elements**.

```
var values = [ {…}, {…}, {…}, … ]; // input data as JS objects

// Select SVG rectangles and bind them to data values.
var bars = svg.selectAll("rect.bars")
      .data(values)
    .join("rect")
      .attr("class", "bars")
      .attr("x", d => xscale(d.x))
      // more code for styling the bars…
```

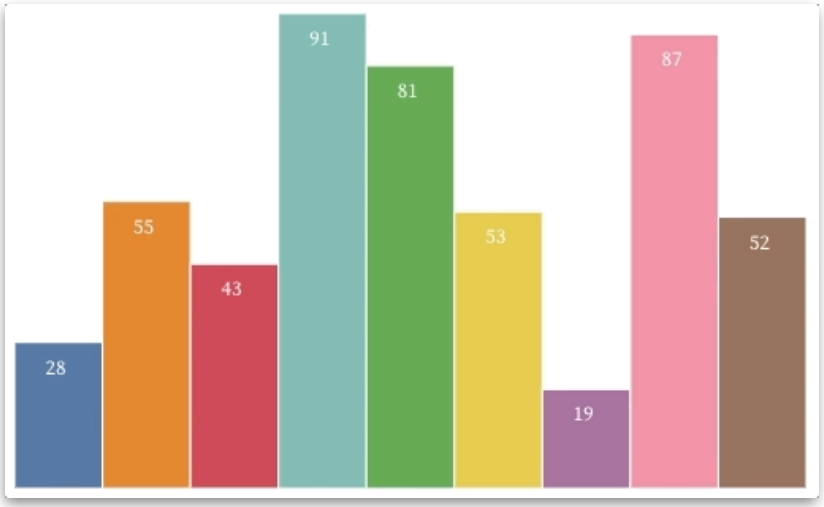
Data Binding with Key Functions

Selections can ***bind*** **data and DOM elements**.

```
var values = [ {…}, {…}, {…}, … ]; // input data as JS objects

// Select SVG rectangles and bind them to data values.
var bars = svg.selectAll("rect.bars")
      .data(values, d => d.x)
    .join("rect")
      .attr("class", "bars")
      .attr("x", d => xscale(d.x))
      // more code for styling the bars…
```




```
shuffle(tensplace(values))
```



Administrivia

Reminders!

Final Project Proposal Due **Fri 2/12, 11:59pm**

https://courses.cs.washington.edu/courses/cse412/21wi/fp.html

Four Peer Evaluations Due **Mon 2/15, 11:59pm**

https://courses.cs.washington.edu/courses/cse412/21wi/a3b.html

D3 is modular.

All of the images in this next part have links!

D3 Modules

Data Parsing / Formatting (JSON, CSV, …)
Shape Helpers (arcs, curves, areas, symbols, …)
Scale Transforms (linear, log, ordinal, …)
Color Spaces (RGB, HSL, LAB, …)
Animated Transitions (tweening, easing, …)
Geographic Mapping (projections, clipping, …)
Layout Algorithms (stack, pie, force, trees, …)
Interactive Behaviors (brush, zoom, drag, …)

Many of these correspond to future lecture topics!

Data Parsing / Formatting

Load file and process data in callback function.

```
d3.csv("path/to/file.csv", function(data) { … });

d3.json("path/to/file.json", function(data) { … });

d3.tsv("path/to/file.tsv", function(data) { … });

d3.xml("path/to/file.xml", function(data) { … });
```

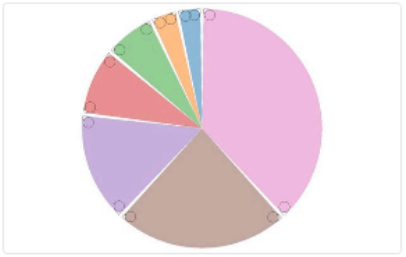
Shape Helpers: Arc, Curve, etc.

"Graphical primitives for visualization"

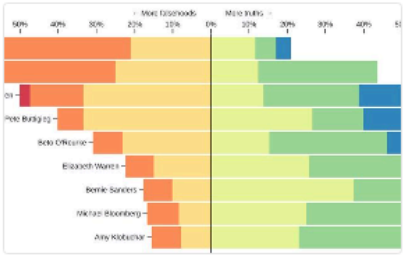
d3-shape

Graphical primitives for visualization, such as lines and areas.

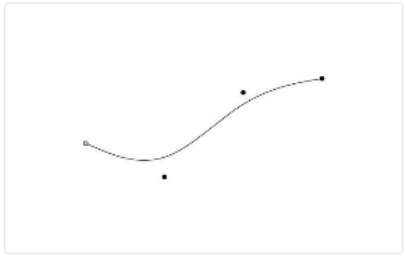
Showing all 29 listings



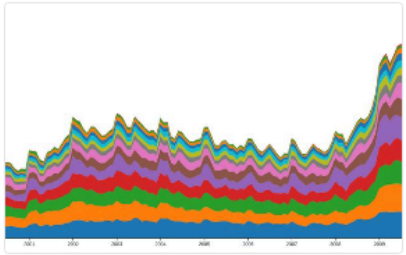
Pie settings
Fil in D3
Aug 19, 2020 • ♡ 9 • 1



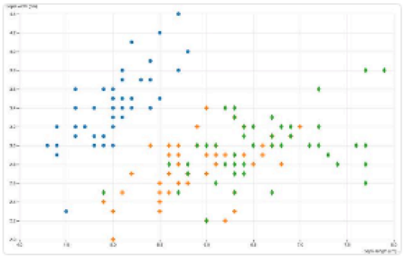
Diverging Stacked Bar Chart
Mike Bostock in D3
Mar 12, 2020 • ♡ 40



Spline Editor
Mike Bostock in D3
Dec 21, 2018 • ♡ 26

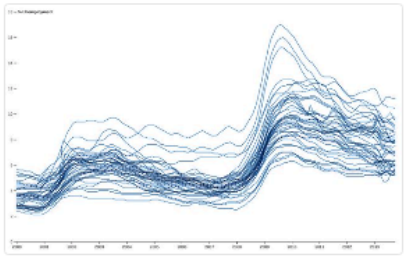


Tidy Stacked Area Chart
Mike Bostock in D3
Aug 7, 2019 • ♡ 9

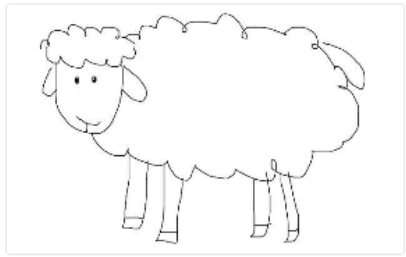


Scatterplot with Shapes
Mike Bostock in D3
Dec 13, 2019 • ♡ 22

Context-to-Curve
Mike Bostock in D3
Nov 4, 2019 • ♡ 24



Multi-Line Chart
Mike Bostock in D3
Oct 20, 2018 • ♡ 167



Draw Me
Mike Bostock in D3
May 23, 2017 • ♡ 34 • 4

Scale Transforms

"Encoding that map abstract data to visual"

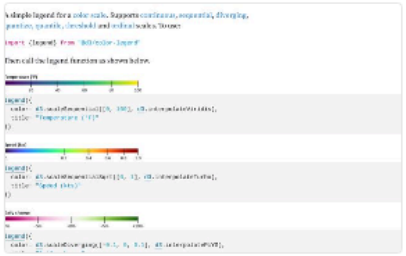
d3-scale

Encodings that map abstract data to visual representation.

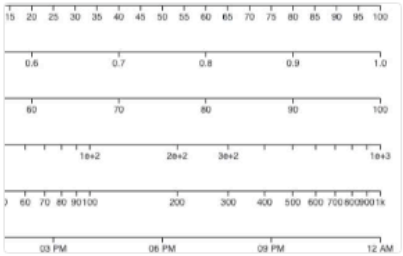
Showing all 15 listings

start	end	timestamp[1]
1880-01-01	1881-04-09	-2840140800
1881-04-09	1887-08-11	-2800000000
1887-08-11	1893-12-12	-2600000000
1893-12-12	1900-04-15	-2400000000
1900-04-15	1906-08-16	-2200000000
1906-08-16	1912-12-17	-2000000000
1912-12-17	1919-04-20	-1800000000
1919-04-20	1925-08-21	-1600000000
… etc.		

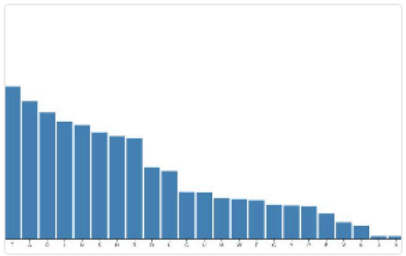
Time thresholds for d3.bin
Fil in D3
Jun 24, 2020 • ♡ 5



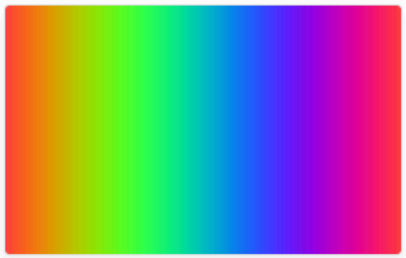
Color Legend
Mike Bostock in D3
Sep 1, 2019 • ♡ 162



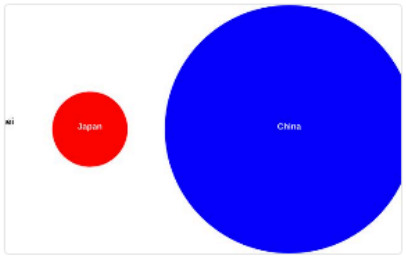
scale.ticks
Fil in D3
Jun 28, 2019 • ♡ 26



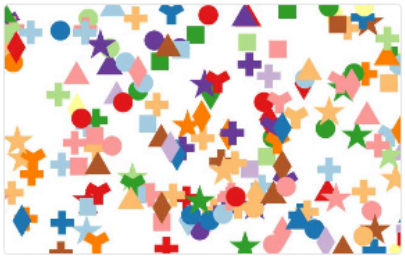
d3.scaleBand
Fil in D3
Jun 26, 2019 • ♡ 19 💬 3



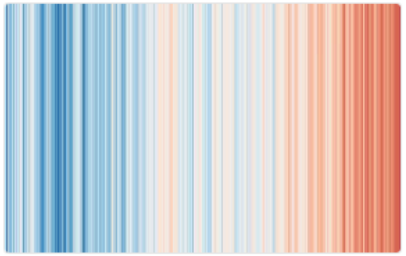
Sequential scales
Fil in D3
Jun 28, 2019 • ♡ 22



Introduction to D3's scales
Fil in D3
Jun 24, 2019 • ♡ 14



d3.scaleOrdinal
Fil in D3
Jun 24, 2019 • ♡ 14



Diverging scales
Fil in D3
Jun 26, 2019 • ♡ 16

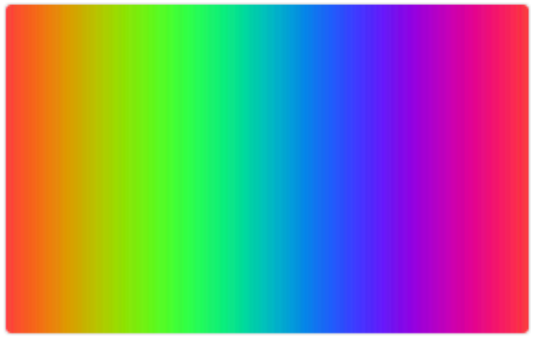
Color Spaces

"RGB, HSL, Cubehelix, Lab (CIELAB) and HCL"

d3-color

Color spaces! RGB, HSL, Cubehelix, Lab (CIELAB) and HCL (CIELCH).

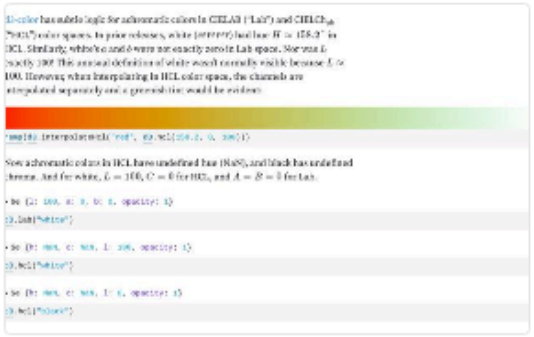
Showing all **3** listings



Sequential scales

Fil in D3

Jun 28, 2019 • ♡ 22



Achromatic Interpolation

Mike Bostock in D3

Apr 17, 2018 • ♡ 6

Working with Color

Mike Bostock in D3

May 7, 2018 • ♡ 59

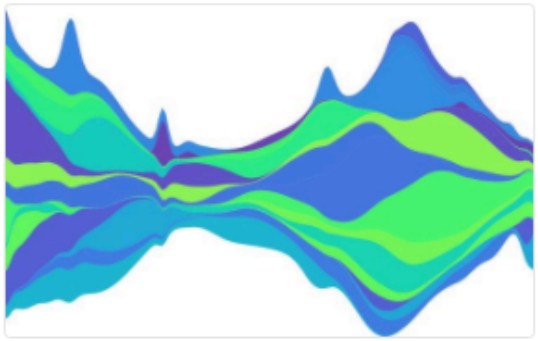
Animated Transitions

d3-transition

Animated transitions for D3 selections.

transition.easeVarying

Fil in D3
Aug 23, 2020 • ♡ 3



Streamgraph Transitions

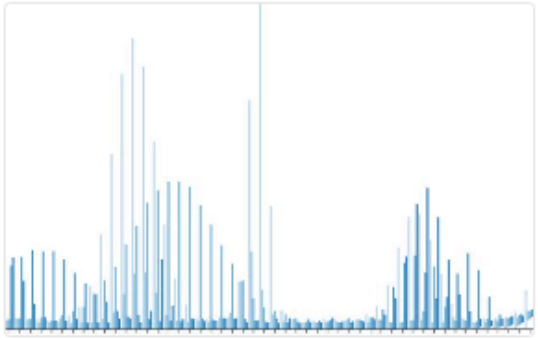
Mike Bostock in D3
Sep 6, 2018 • ♡ 52

transition.textTween

Mike Bostock in D3
Nov 17, 2019 • ♡ 15 💬 3

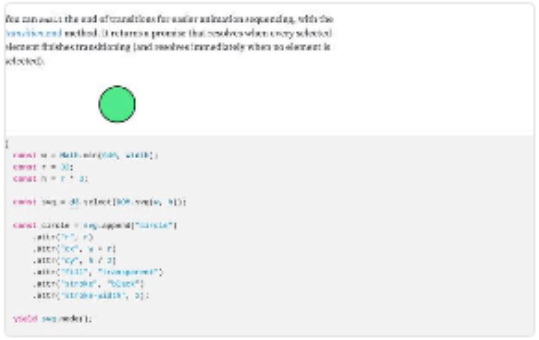
Easing Animations

Mike Bostock in D3
Aug 17, 2019 • ♡ 29



Stacked-to-Grouped Bars

Mike Bostock in D3
Oct 22, 2018 • ♡ 74

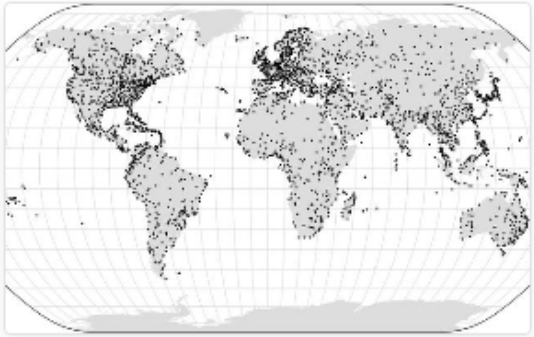


transition.end

Mike Bostock in D3
Jan 24, 2019 • ♡ 27

Geographic Mapping

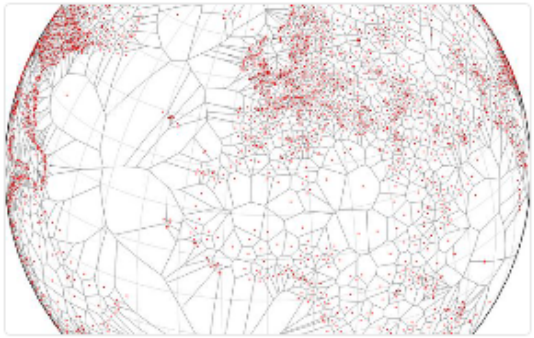
d3-geo



World Airports

Mike Bostock in D3

Nov 3 • ♡ 6

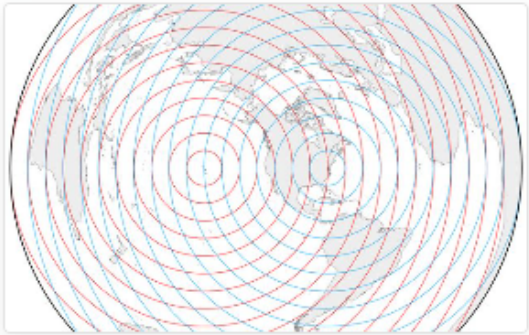


World Airports Voronoi

Mike Bostock in D3

Aug 28, 2018 • ♡ 28

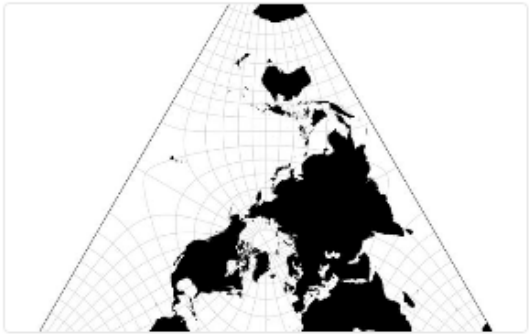
d3-geo-polygon



Two Point Equidistant

Fil in D3

May 22, 2020 • ♡ 6

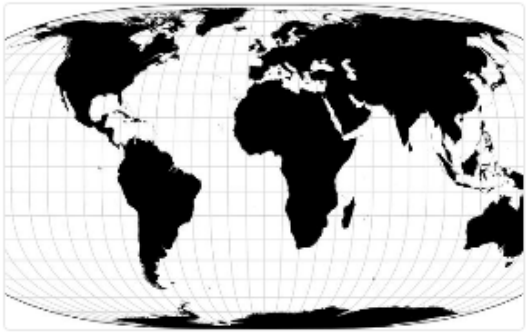


Lee's Tetrahedral

Mike Bostock in D3

Mar 15, 2019 • ♡ 1

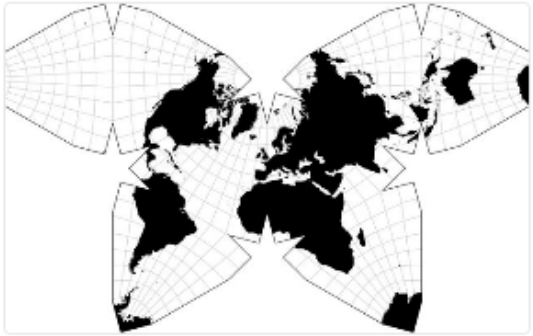
d3-geo-projection



Hufnagel

Fil in D3

Jul 27, 2019



Waterman's Butterfly

Mike Bostock in D3

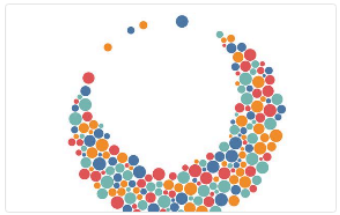
Mar 15, 2019 • ♡ 4

Layout Algorithms

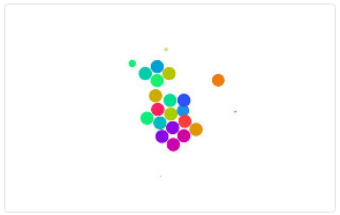
d3-force

Force-directed graph layout using velocity Verlet integration.

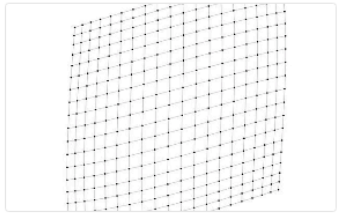
Showing all **16** listings



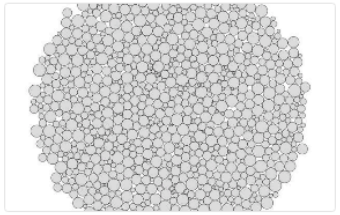
Collision Detection
Mike Bostock in D3
Sep 16 • 55



forceCenter.strength
Fil in D3
Aug 25, 2020 • 11

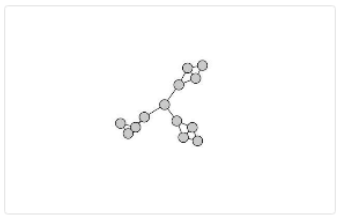


Force-Directed Lattice
Fil in D3
Sep 1 • 13



Collision Detection
Fil in D3
Sep 1 • 5

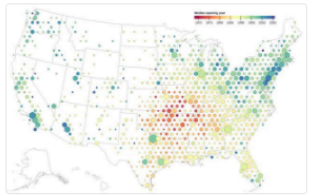
Temporal Force-Directed Graph
Mike Bostock in D3
Jul 8, 2020 • 41 • 2



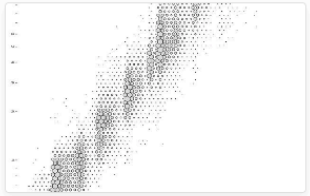
Sticky Force Layout
Fil in D3
Sep 2 • 15 • 1

d3-hexbin

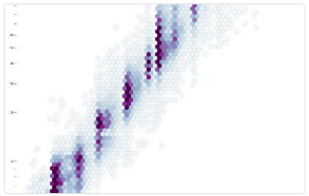
Group two-dimensional points into hexagonal bins.



Hexbin Map
Mike Bostock in D3
Feb 26, 2019 • 55



Hexbin (Area)
Mike Bostock in D3
Oct 21, 2018 • 13

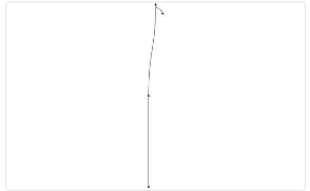


Hexbin
Mike Bostock in D3
Oct 21, 2018 • 42

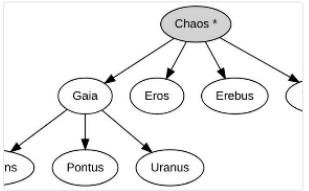
d3-hierarchy

2D layout algorithms for visualizing hierarchical data.

d3.groups as a hierarchy
Fil in D3
Sep 15 • 38 • 4



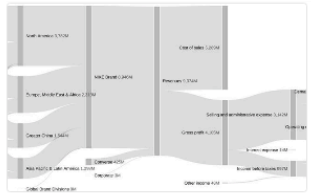
Random Tree
Mike Bostock in D3
Sep 29, 2018 • 28



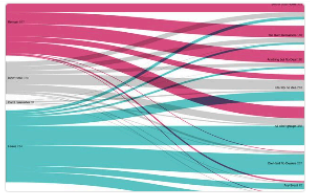
Hierarchy traversal, animated
Fil in D3
Jul 6, 2020 • 12

d3-sankey

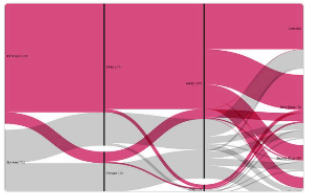
Visualize flow between nodes in a directed acyclic network.



Nike Quarterly Statement
Mike Bostock in D3
Mar 3, 2019 • 7 • 2



Brexit Voting
Mike Bostock in D3
Mar 11, 2019 • 18 • 1



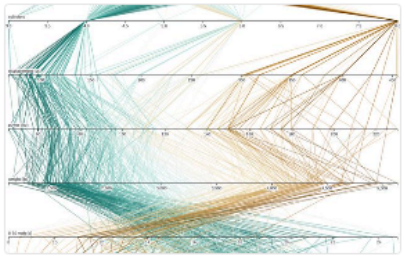
Parallel Sets
Mike Bostock in D3
Mar 10, 2019 • 58

Interactive Behaviors

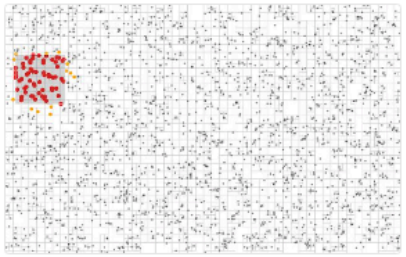
d3-brush

Select a one- or two-dimensional region using the mouse or touch.

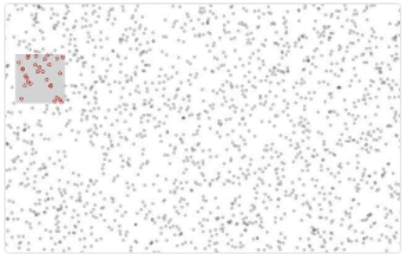
Showing all **15** listings



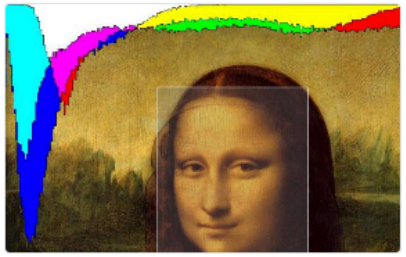
Brushable Parallel Coordinates
Kerry Rodden in D3
Sep 10 • ♡ 18



Quadtree Brush
Fil in D3
Sep 2 • ♡ 14 💬 7



brush.filter
Mike Bostock in D3
Aug 20, 2019 • ♡ 12



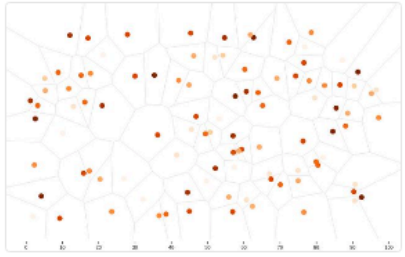
Mona Lisa Histogram
Mike Bostock in D3
Aug 8, 2019 • ♡ 11

d3-zoom

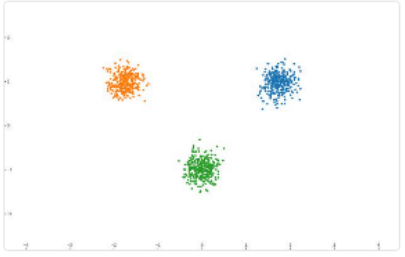
Pan and zoom SVG, HTML or Canvas using mouse or touch input.

Showing all **20** listings

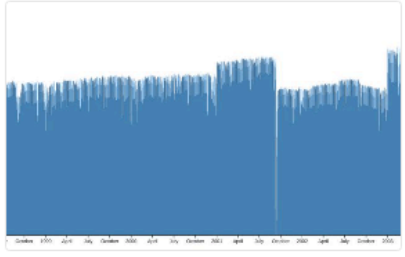
delaunay.find & zoom
Fil in D3
Oct 14 • ♡ 28 💬 1



X/Y Zoom
Fil in D3
Jun 25, 2020 • ♡ 12 💬 1



Scatterplot Tour
Mike Bostock in D3
Apr 3, 2020 • ♡ 17



Zoomable Area Chart
Mike Bostock in D3
Jan 14, 2020 • ♡ 28

D3 is flexible!