CSE 412 - Intro to Data Visualization

Intro to D3.js

Jane Hoffswell  University of Washington
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:
D³: 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.
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.
"D3 allows you to bind arbitrary data to a Document Object Model"

**HTML:** Structure of items on a page.

```html
<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.

```css
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;
}
```

---

Click title elements for web resources.
"D3 allows you to bind arbitrary data to a Document Object Model"

HTML for Course Website:

```html
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8">
    <title>Course Website</title>
    <link rel="stylesheet" href="styles.css">
  </head>
  <body>
    <div class="navbar">
      <h2>Textbooks</h2>
      <h2>Learning Goals & Objectives</h2>
      <p>This course is designed to provide students with the foundation for data visualization. By the end of the course, students will be able to...<br>
      <a href="schedule.html">Schedule & Readings</a>|<a href="assignments.html">Assignments</a>
    </div>
    <div class="content">
      <h3>Week 1</h3>
      <h4>Lecture 1</h4>
      <p>Introduction to JavaScript</p>
      <h4>Lecture 2</h4>
      <p>Web Development Fundamentals</p>
      <h4>Lecture 3</h4>
      <p>Data Visualization Tools</p>
    </div>
  </body>
</html>
```

CSS for Course Website:

```css
* { padding: 0.5px 10.5px; }
body { background: □#7533f4; }
ul { font-size: 0.85em; color: □#666; }
li { display: none; font-size: 0.75em; background: □#333; border-radius: 4px; color: □#000; }
li.signed { padding: 0.5px 3.5px; background: □#000; color: □#666; }
section { position: relative; top: -15em; font-size: 0.75em; background: □#666; border-radius: 4px; color: □#000; }
```

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.
"D3 allows you to bind arbitrary data to a Document Object Model"

**SVG**: Scalable Vector Graphics - shapes and lines!

```xml
<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`.

```javascript
// 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.

```javascript
// 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.

```javascript
var values = [ {...}, {...}, {...}, ... ]; // input data as JS objects
```
Data Binding

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

```javascript
var values = [
    {"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}
];
```
Selections can **bind** data and DOM elements.

```javascript
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

**ENTER**
Data values without matching DOM elements.

**UPDATE**
Existing DOM elements, bound to valid data.

**EXIT**
DOM elements whose bound data has gone “stale”.
Data Binding

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

```javascript
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(
    enter => enter.append("rect").attr("class","bars"),
    update => update,
    exit => exit.remove()
  )
```

Data Binding
<svg width="500" height="300">
  <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>
  <rect class="bars"></rect>
</svg>
## 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))
Data Binding

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

```javascript
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()
    )
```
Selections can **bind** data and DOM elements.

```javascript
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.

```javascript
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…
```

// input data as JS objects
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.
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
- Fill 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 - 24

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

<table>
<thead>
<tr>
<th>Time thresholds for d3.bin</th>
<th>Color Legend</th>
<th>scale.ticks</th>
<th>d3.scaleBand</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mike Bostock in D3</td>
<td></td>
<td>Fill in D3</td>
<td>Fill in D3</td>
</tr>
<tr>
<td>Jun 24, 2020</td>
<td>Sep 1, 2019</td>
<td>Jun 28, 2019</td>
<td>Jun 26, 2019</td>
</tr>
<tr>
<td>5</td>
<td>162</td>
<td>26</td>
<td>3</td>
</tr>
</tbody>
</table>

**Sequential scales**

Fill in D3

Jun 28, 2019 - 22

**Introduction to D3’s scales**

Fill in D3

Jun 24, 2019 - 14

**d3.scaleOrdinal**

Fill in D3

Jun 24, 2019 - 14

**Diverging scales**

Fill 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 likes

**Achromatic Interpolation**

Mike Bostock in D3

Apr 17, 2018 • 6 likes

**Working with Color**

Mike Bostock in D3

May 7, 2018 • 59 likes
Animated Transitions

**d3-transition**

Animated transitions for D3 selections.

- **transition.easeVarying**
  - Mike Bostock in D3
  - Aug 23, 2020 • 💬 3

- **Streamgraph Transitions**
  - Mike Bostock in D3
  - Sep 6, 2018 • 💬 52

- **transition.textTwee**
  - 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
Layout Algorithms

**d3-force**
Force-directed graph layout using velocity Verlet integration.

Showing all 16 listings

- Collision Detection
  - Mike Bostock in D3
  - Sep 15 - 38

- forceCenter strength
  - Mike Bostock in D3
  - Aug 29, 2020 - 11

- Force-Directed Lattice
  - Mike Bostock in D3
  - Jul 8, 2020 - 41

- Collision Detection
  - Mike Bostock in D3
  - Sep 1 - 5

**d3-hexbin**
Group two-dimensional points into hexagonal bins.

- Hexbin Map
  - Mike Bostock in D3
  - Feb 26, 2019 - 58

- 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
  - Mike Bostock in D3
  - Sep 15 - 38

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

- Hierarchy traversal, animated
  - Mike Bostock in D3
  - Jul 8, 2010 - 12

**d3-sankey**
Visualize flow between nodes in a directed acyclic network.

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

- Brexit Voting
  - Mike Bostock in D3
  - Mar 11, 2010 - 18

- Parallel Sets
  - Mike Bostock in D3
  - Mar 10, 2010 - 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
  - Oct 29, 2018
  - Heart 18

- Quadtree Brush
  - Fil in D3
  - Sep 2
  - Heart 14
  - Comments 7

- brush.filter
  - Mike Bostock in D3
  - Aug 20, 2019
  - Heart 12

- Mona Lisa Histogram
  - Mike Bostock in D3
  - Aug 8, 2019
  - Heart 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, 2018
  - Heart 28
  - Comments 1

- X/Y Zoom
  - Fil in D3
  - Jun 29, 2020
  - Heart 12
  - Comments 1

- Scatterplot Tour
  - Mike Bostock in D3
  - Apr 3, 2020
  - Heart 17

- Zoomable Area Chart
  - Mike Bostock in D3
  - Jan 14, 2020
  - Heart 28
D3 is flexible!