

Data Visualization on the Web with D3 and Leaflet

Shruti Mukhtyar

Web Developer

Geospatial Innovation Facility, UC Berkeley

Projects: [Cal-Adapt](#), [VTM](#)

Twitter: [@mapchitra](#) [@cal_adapt](#)

Today's Plan

- Brief overview of Data Visualization
- Where does D3 fit in?
- Whirlwind introduction to Web Development
- Explore D3 Core Concepts and build a bar chart
- Web mapping with Leaflet
- Discussion (Visualization, Web Development, Learning, Deeper dive into some examples, etc.)

Brief Overview of Data Visualization

What is Data Visualization?

- Graphical display of abstract information for communication and sense-making

Goals

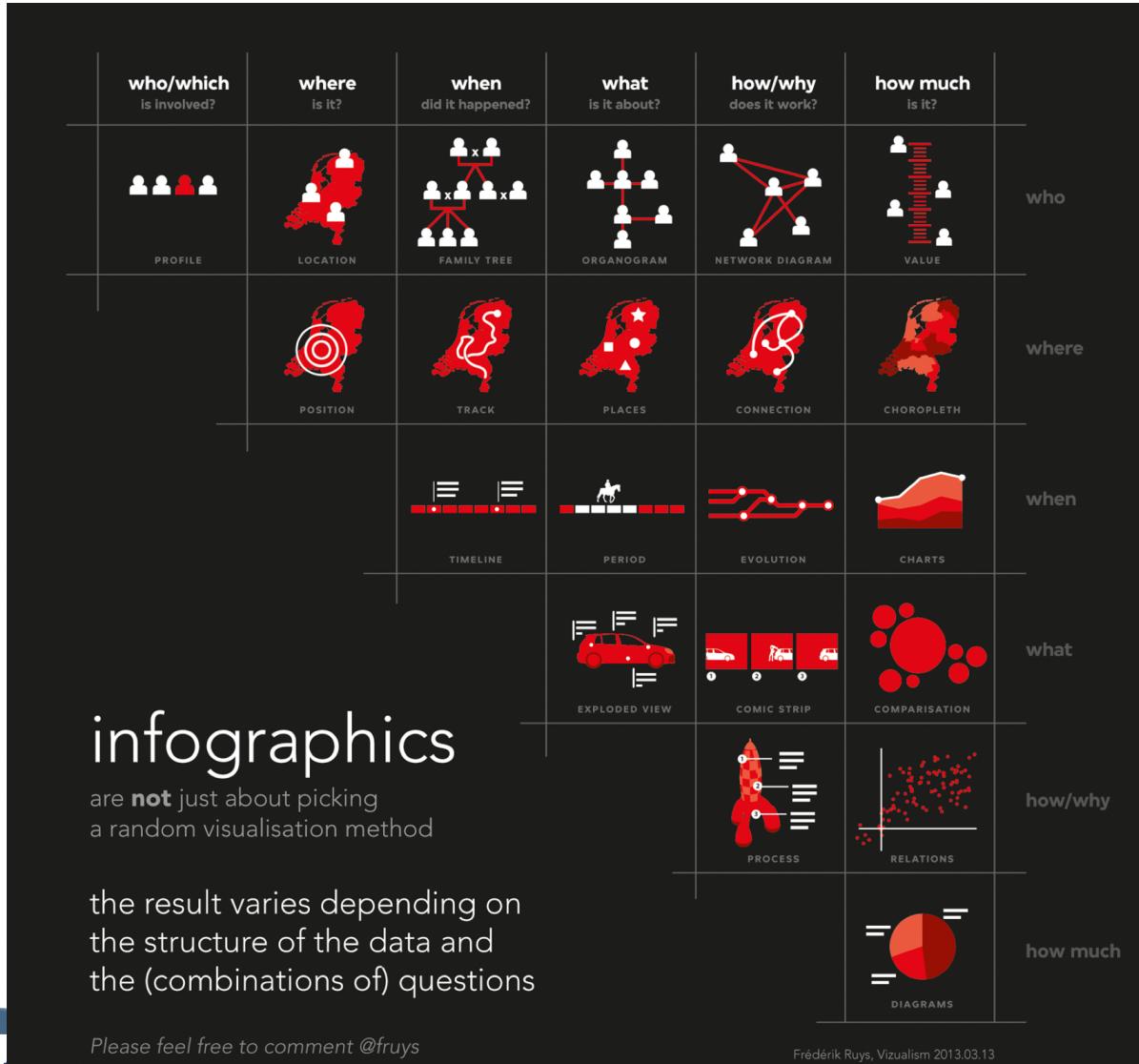
- Using graphics with or without explanatory texts for **Communication**
- Displaying data for exploratory **Analysis**
- **The Changing Goals of Data Visualization**

Types of Quantitative Messages (Stephen Few)

- Timeseries: A single variable is captured over a period of time
- Ranking: Categorical subdivisions are ranked in ascending or descending order
- Part-to-whole: Categorical subdivisions are measured as a ratio to the whole
- Deviation: Categorical subdivisions are compared against a reference
- Frequency Distribution: Number of observations of a particular variable for given interval
- Correlation: Comparison between observations represented by two variables
- Nominal Comparison: Comparing categorical subdivisions in no particular order
- Geographic: Comparison of a variable across a map or layout

Form follows function

[Link](#)



Today's goal

- What kind of visualization to build? ✗
 - Book list
 - Alberto Cairo, Edward Tufte, Stephen Few, Tamara Munzner, Cole Nussbaumer Knaflic, Andy Kirk
- How do you go about building the visualization you want? ✓

Why publish on the web?

- Fastest way to reach a larger audience
- Working with web-standard technologies means that your work can be seen and experienced by anyone using a recent web browser, regardless of the operating system and device type.
- Avoiding proprietary software (e.g. GIS)

Technology Strategy

- Who's your target audience?
- How much data do you have? Where is it?
- Is the web map/dataviz going to part of an existing website?
- Do you need to support tablets and mobiles?
- Do you need to support offline access?
- Public or private?

Data Strategy

- Data has a longer life-cycle
- Web presentations of data age quickly

Visualization Tools

[Link](#)

+ DATAVISUALIZATION.CH **SELECTED TOOLS**

All **Maps** Charts Data Color

Search..

CartoDB
A web service for mapping, analyzing and building applications with data.

D3.js
An small, flexible and efficient library to create and manipulate interactive documents based on data.

GeoCommons
A public community and set of tools to access, visualize and analyze data with compelling map visualizations.

Google Chart Tools
A collection of simple to use, customizable and free to use interactive charts and data tools.

Google Fusion Tables
A web application that makes it easy to host, manage, collaborate on, visualize, and publish data tables.

Kartograph
A simple and lightweight framework for creating beautiful, interactive vector maps.

Leaflet
A lightweight JavaScript library for making tile-based interactive maps for desktop and mobile browsers.

Many Eyes
A web application to build, share and discuss graphic representation of user uploaded data.

Visualization Tools

[Link](#)

↑ DATAVISUALIZATION.CH SELECTED TOOLS

All **Maps** Charts Data Color

Search...

MapBox
A web platform for hosting custom designed map tiles and a set of open source tools to produce them.

Modest Maps
A display and interaction library for tile-based maps in Flash, JavaScript and Python.

Polymaps
A library for making dynamic, interactive maps with image- and vector-based tiles.

Tableau Public
A desktop application to build and post interactive graphs, dashboards, maps and tables to the web.

Unfolding
A library to create interactive maps and geovisualizations in Processing and Java.

Visualization Tools

[Link](#)

+ DATAVISUALIZATION.CH SELECTED TOOLS

Search...

All Maps **Charts** Data Color



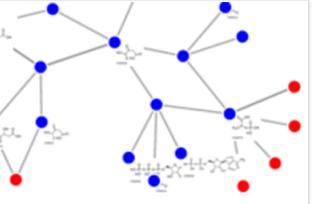
Arbor.js

A library of force-directed layout algorithms plus abstractions for graph organization and refresh handling.



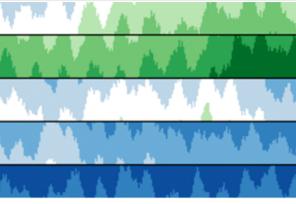
Circos

A software package for visualizing data in a circular layout.



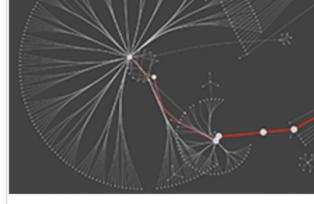
Cola.js

A library for arranging networks using constraint-based optimization techniques.



Cubism.js

A library for creating interactive time series and horizon graphs based on D3.js



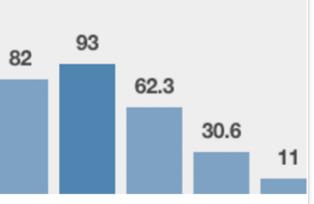
Cytoscape

An application for visualizing complex networks and integrating these with any type of attribute data.



D3.js

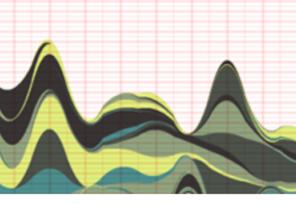
An small, flexible and efficient library to create and manipulate interactive documents based on data.



Dance.js

A simple data-driven visualization framework based on Data.js and Underscore.js

82	93	62.3	30.6	11
----	----	------	------	----



Degrafa

A powerful declarative graphics framework for rich user interfaces, data visualizations and mapping.

Visualization Tools

[Link](#)

↑ DATAVISUALIZATION.CH SELECTED TOOLS

All Maps **Charts** Data Color

Search.

Envision.js
A library for creating fast, dynamic and interactive time series visualizations.

Flare
A set of software tools for creating rich interactive data visualizations in ActionScript.

Gephi
A visualization and exploration platform for networks with dynamic and hierarchical graphs.

Google Chart Tools
A collection of simple to use, customizable and free to use interactive charts and data tools.

Google Fusion Tables
A web application that makes it easy to host, manage, collaborate on, visualize, and publish data tables.

JavaScript InfoVis Toolkit
A JavaScript library that provides tools for creating interactive data visualizations for the web.

Many Eyes
A web application to build, share and discuss graphic representation of user uploaded data.

NVD3.js
A collection of re-usable charts and chart components for d3.js.

Visualization Tools

Link

↑ DATAVISUALIZATION.CH SELECTED TOOLS

All Maps **Charts** Data Color

Search...

The screenshot shows a grid of nine visualization tools under the 'Charts' category:

- NodeBox**: A desktop application for generative, static, animated, or interactive visuals. Example image: a dense network of brown lines forming a branching structure.
- Paper.js**: A vector graphics scripting framework. Example image: concentric red and blue spirals.
- Peity**: A jQuery plugin for simple charts. Example image: a green pie chart labeled 'insect'.
- Prefuse**: Software tools for Java-based data visualization. Example image: a complex circular sunburst chart with labels like 'arthropod' and 'insect'.
- Processing**: An open-source programming language for images, animations, and interactions. Example image: a grid of small circles connected by lines.
- Processing.js**: The sister project of Processing for web standards. Example image: a cluster of colored circles of varying sizes.
- Protovis**: A library for custom data views. Example image: a chart with orange bars and a blue base layer.
- R**: A software environment for statistical computing and graphical techniques. Example image: a candlestick chart with a bar chart overlay.

Where does D3 fit in?

What is D3?

- From d3js.org:
 - 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.

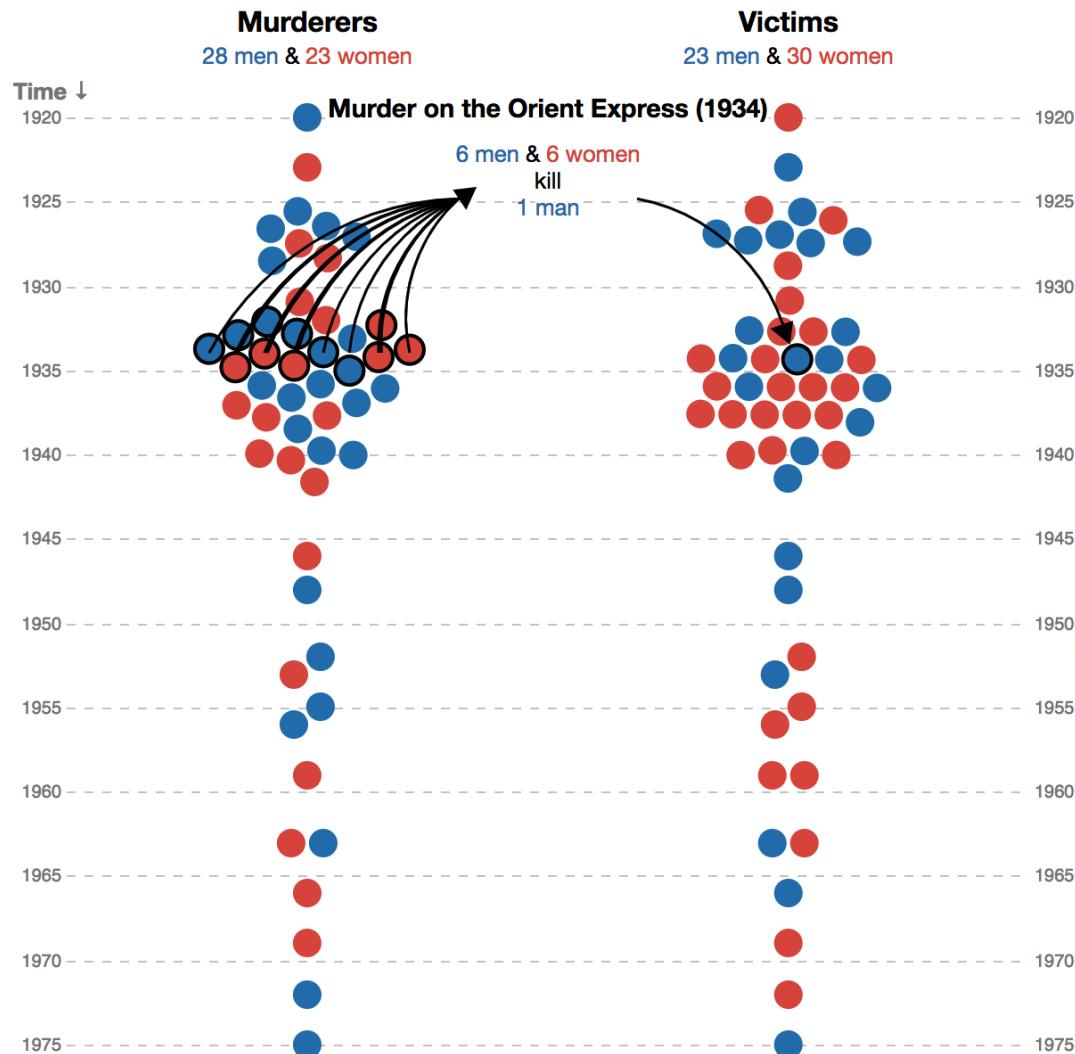
Why learn D3?

- Create highly customizable and interactive visualizations on the web
- Knowing D3 = Hirable skills
- Opportunity to learn web development
- Thousands of examples and a great open source community
- [5 reasons to learn D3](#)

Linked Beeswarm Plot

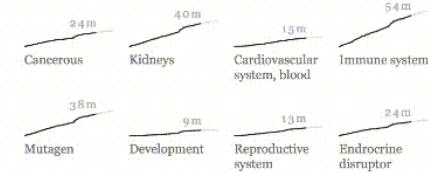
[Link](#)

Each **circle** represents a murderer or a victim. Earlier stories are on **top**; later ones are on **bottom**. Hover or tap on a circle for more information.



Face of Fracking

[Link](#)

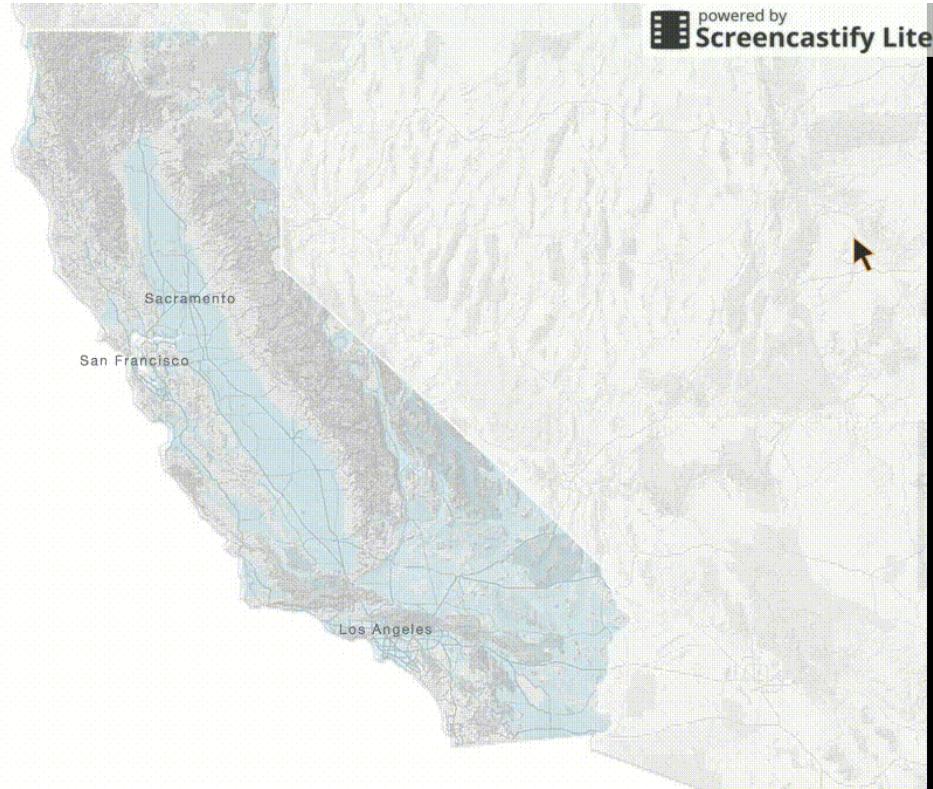


That's a total of 72 million pounds of chemicals in a single year.

These chemicals can cause serious medical ailments, including severe organ damage, infertility, birth defects, and cancer.

The LA Basin is not the only place in California where oil and gas companies are carrying out these high-intensity production techniques.

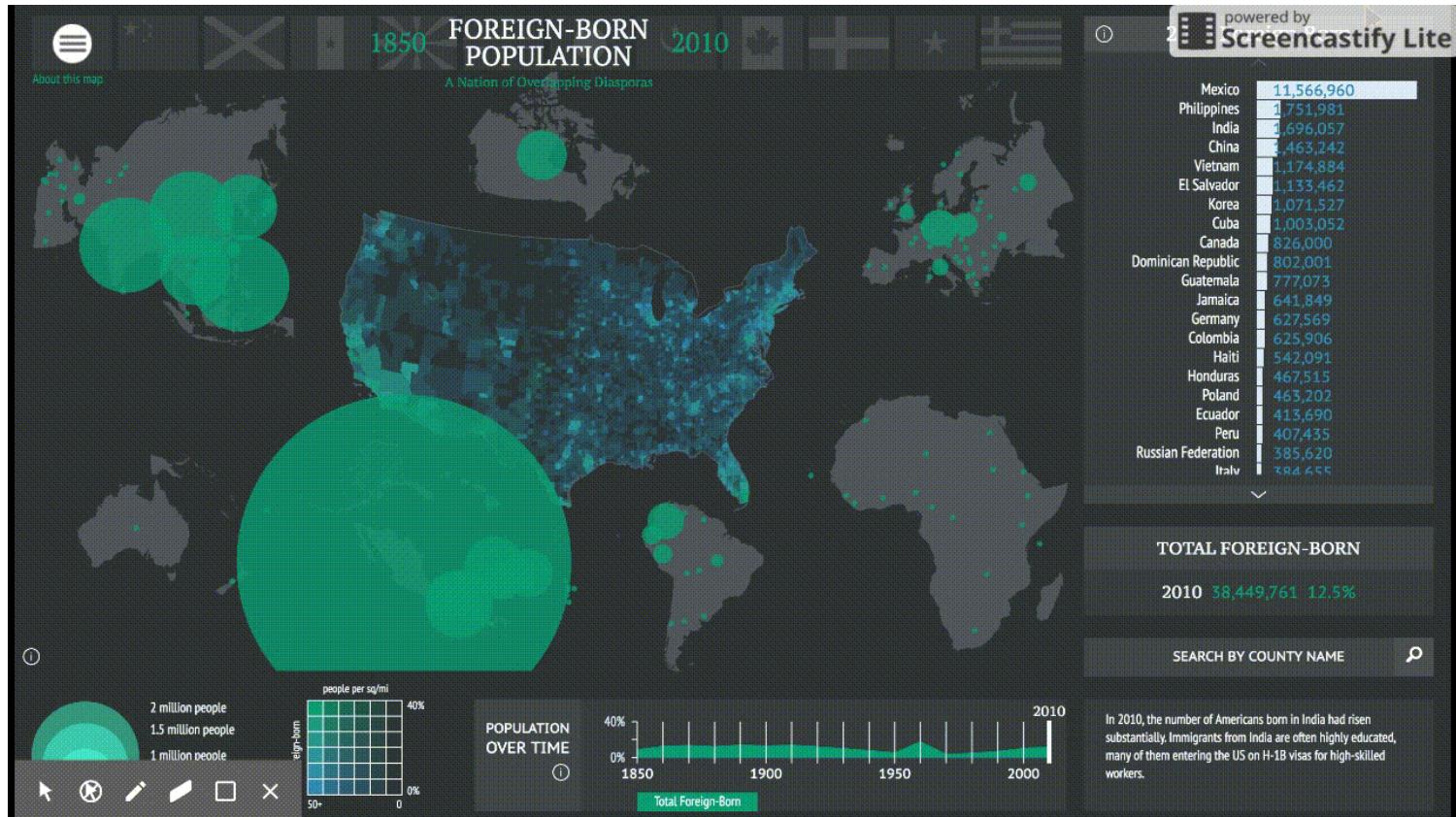
Here's a map of the 3,014 wells across California where high-intensity



powered by
Screencastify Lite

American Panorama - Foreign Born Population

[Link](#)



Simpson's Paradox

[Link](#)

Proper Pooling

By "properly pooled," the investigators at Berkeley meant "broken down by department." Men more often applied to science departments, while women inclined towards humanities. Science departments require special technical skills but accept a large percentage of qualified applicants. In contrast, humanities departments only require a standard undergrad curriculum but have fewer slots.

The authors concluded that any sexism occurred before Berkeley ever saw the applications:

Women are shunted by their socialization and education toward fields of graduate study that are generally more crowded, less productive of completed degrees, and less well funded, and that frequently offer poorer professional employment prospects.

— (p.403)

To the right are data on the six largest departments, but the names have been changed to protect the innocent.



Illustration

Suppose there are two departments: one easy, one hard ('hard' as in 'hard to get into'). The sliders below set what percentage each gender applies to the easy department. Both departments prefer women, but if too many women apply to the hard one, their acceptance rate drops below the men's.



departments	# applied		# admitted		% admitted	
	men	women	men	women	men	women
"Easy"	1,076	1,340	672	1,068	62%	80%
"Hard"	1,615	495	413	131	26%	74%
Combined	2,691	1,835	1,085	1,199	40%	65%

Simpson's paradox? no

Generative Art with D3

[Link](#)



Optionally...

- Software
 - Tableau, Excel, GIS software
- Data visualization services
 - [Datawrapper](#), [RAW Graphs](#)
- R or Python
 - Associated visualization packages such as ggvis, rCharts, and Shiny
- Create interactive graphics but less customization options and/or limited pre-existing graphics types (e.g., bar charts).

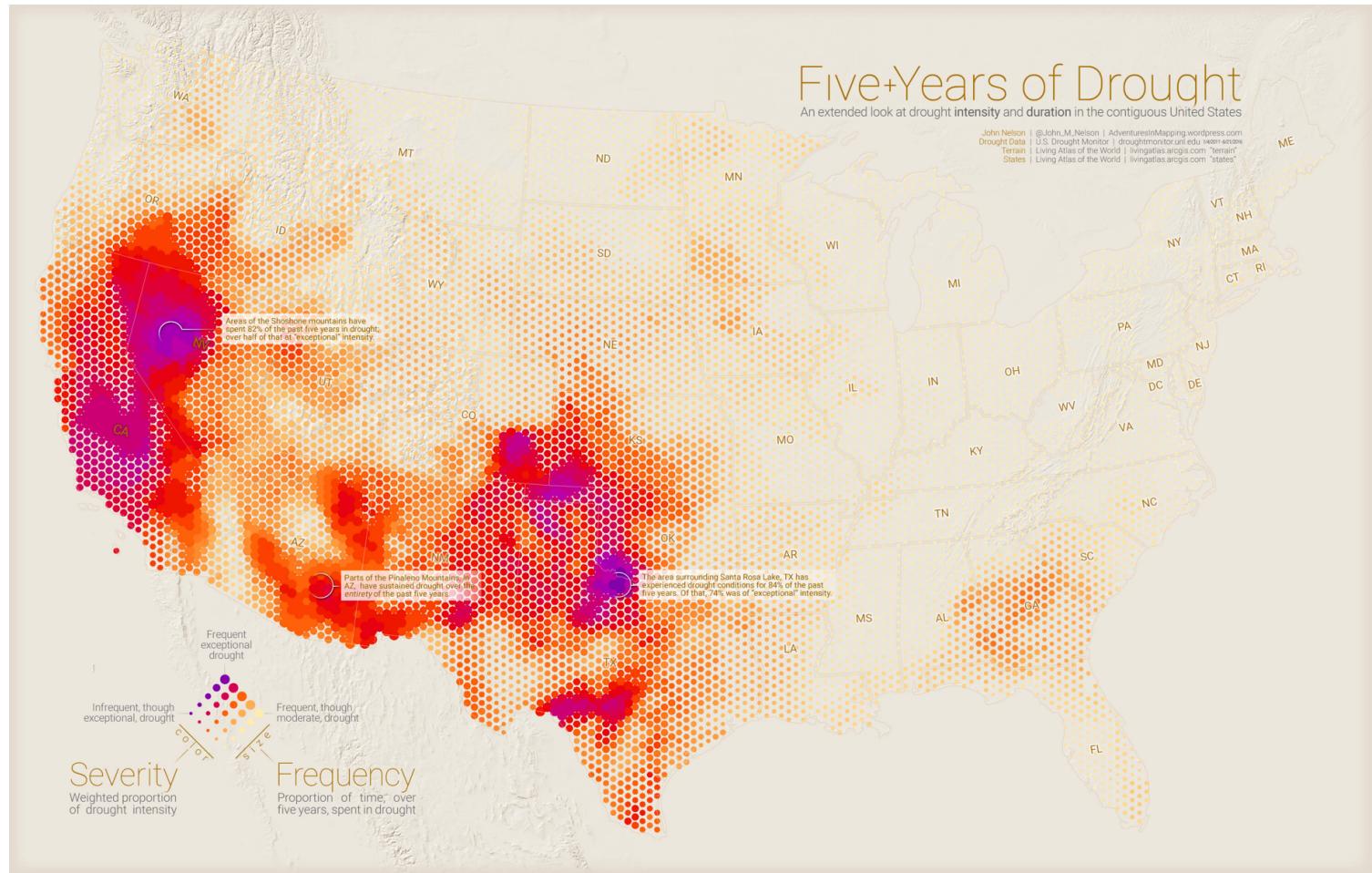
Visualizing Friendships - R

[Link](#)



Five Years of Drought - ArcGIS Pro

[Link](#)

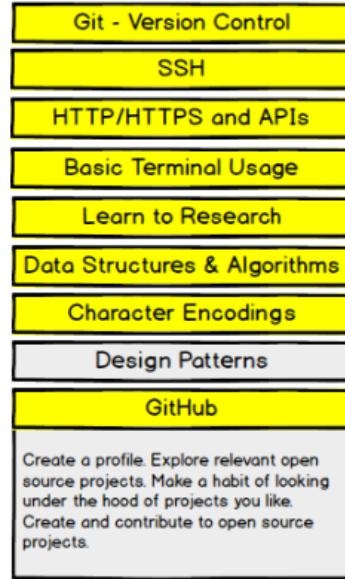


Whirlwind Introduction to Web Development

Roadmap for Learning Web Development

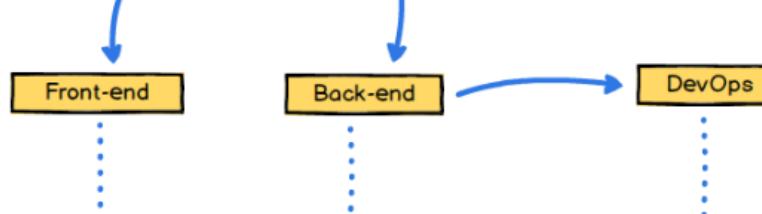
[Link](#)

Required for any path

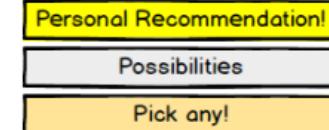


Web Developer in 2018

Choose your path



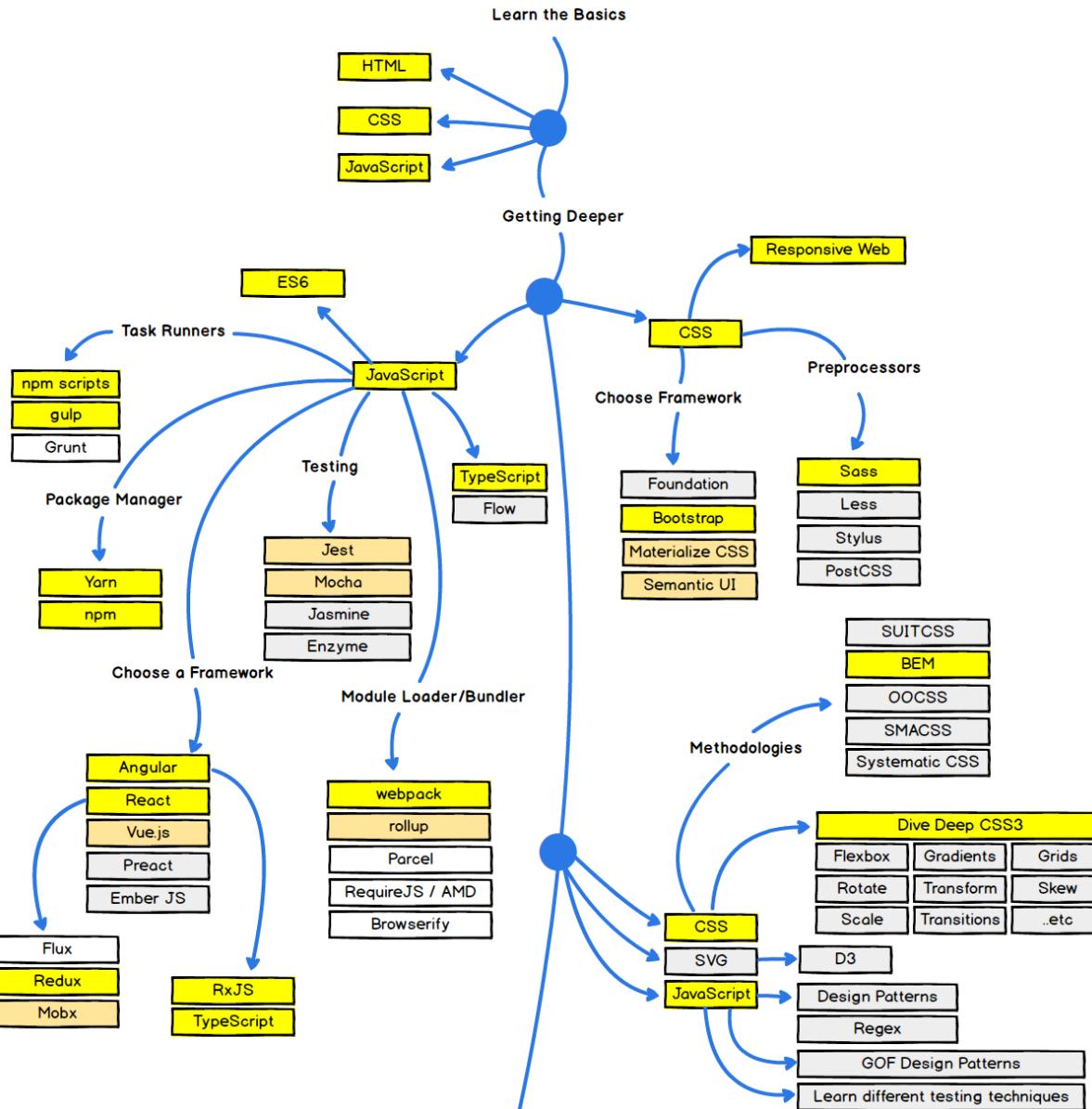
Legends



Now build something

- Frontend
 - HTML5, CSS3, JavaScript
 - Frontend frameworks
 - SVG, Canvas, WebGL

- Backend
 - Python, Java
 - Backend frameworks
 - Web Servers



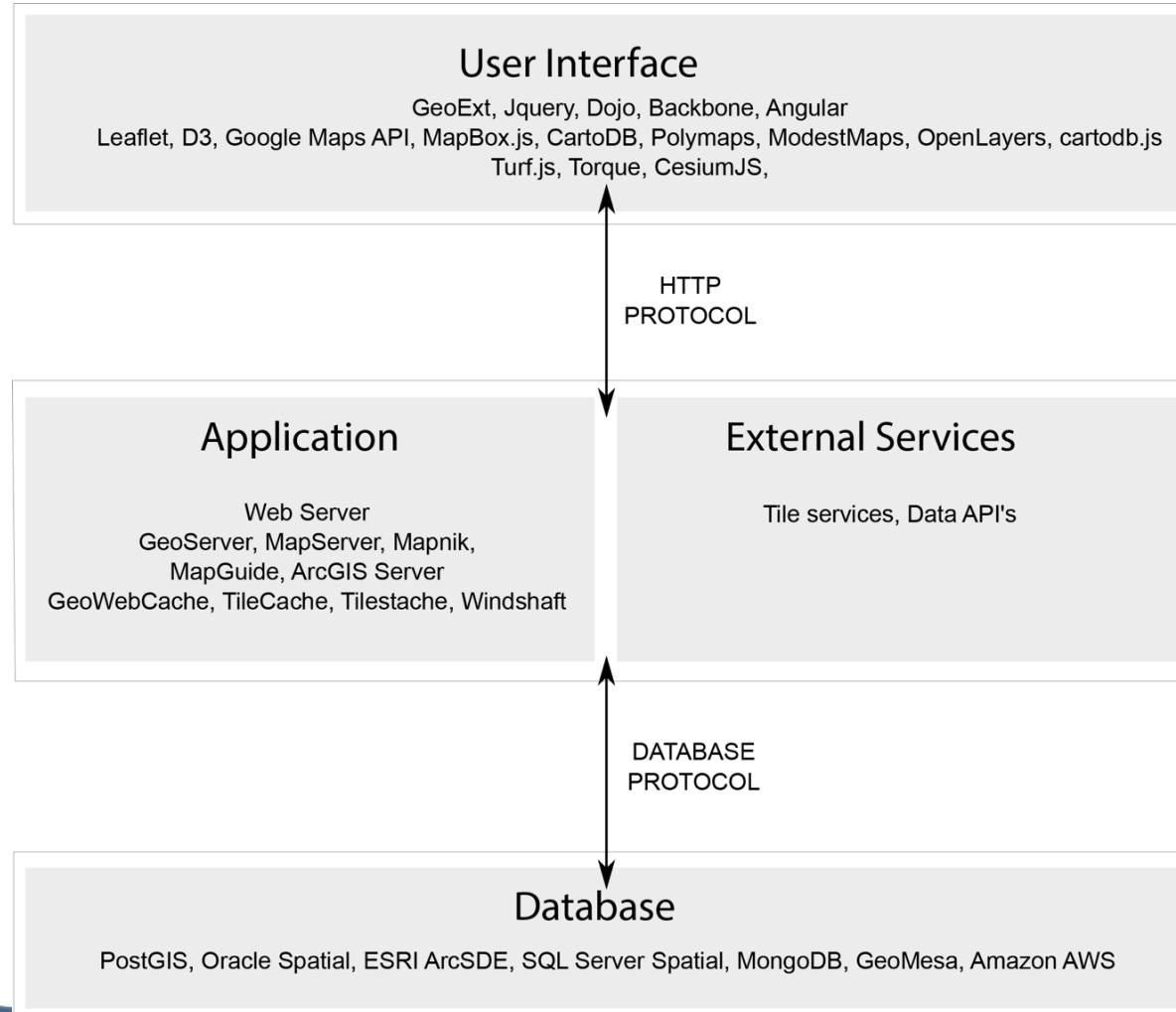
Quick Start Technology Stack

- Data, Middleware & User Interface
 - ArcGIS Online
 - CARTO
 - MapBox
 - Google
 - Tableau, RStudio, [Datawrapper](#), [RAW Graphs](#)
- Embed widgets into your own website

Hybrid Technology Stack

- Don't want to maintain database/mapping servers?
 - Use data from Web API's
 - Use web services from ArcGIS Online, CARTO, MapBox, Google, etc.
 - Build custom user interface with JavaScript libraries
- Don't want to write Javascript/HTML/CSS?
 - Build your web app with Rshiny; host your data and app on a server running Shiny Server
 - Build your web app with Python packages (Folium, Bokeh), deploy on a web server
- Interactive maps and data with Google Sheets

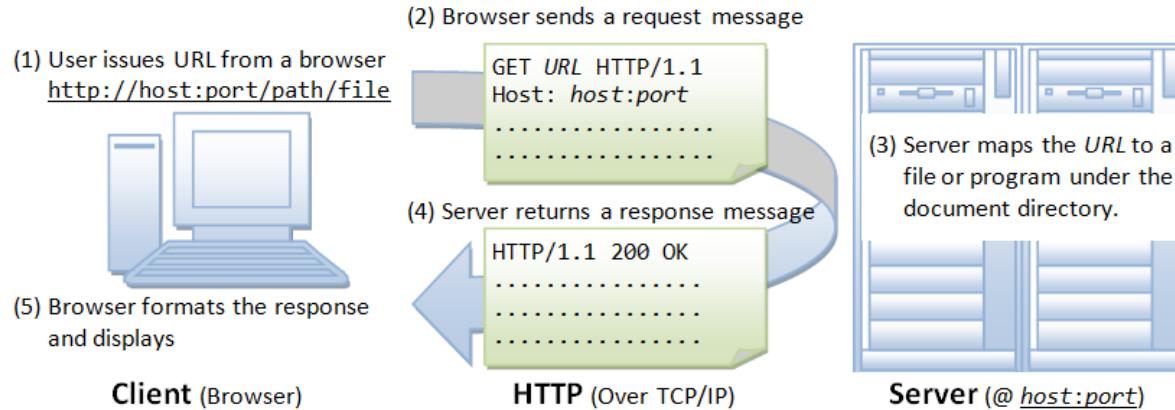
Custom Technology Stack



Quick Review

- Client & Server
- Web Developer Tools
- HTML
- CSS
- DOM
- JavaScript
- Web Graphic Formats - SVG and Canvas

Client & Server



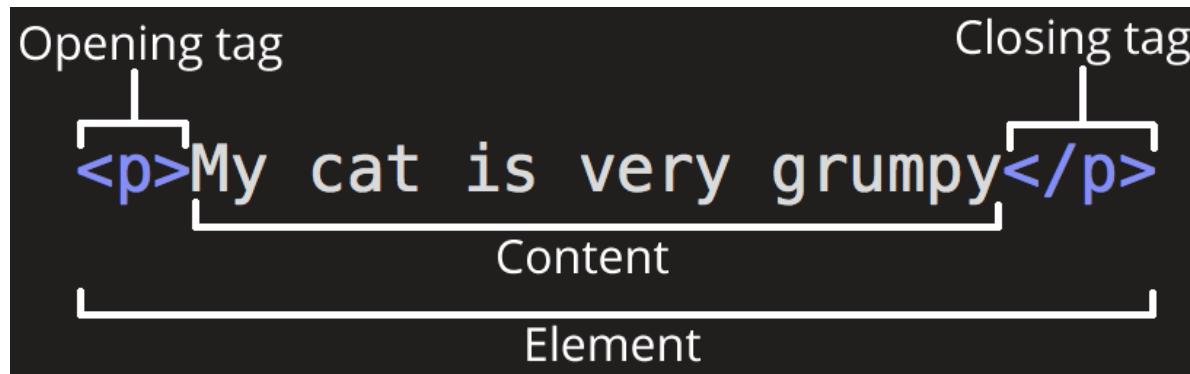
- HTTP, language of the web
- Browser sends HTTP GET Request. Server sends response for each request with content and/or status message.
- **HTTP Basics**

Web Developer Tools

- Code Editor
 - Sublime Text, Atom, PyCharm, Vim, Notepad++, IDLE, Gedit, TextMate
 - Syntax highlighting, indentation, autocomplete, bracket matching
 - Run interpreters, debuggers
- Browser
 - Developer Tools - press `Ctrl+Shift+I` (or `F12`) in Chrome
 - More info on Chrome Developer tools [here](#)
- [Github](#)
 - Code hosting platform for version control and collaboration

Hyper Text Markup Language (HTML)

- HTML is not a programming language; it is a markup language
- Role » Describes content
- HTML Elements consist of a opening tag, the content, and a closing tag



```
<ul>
  <li><a href="gif.berkeley.edu">Home Page</a></li>
  <li><a href="gif.berkeley.edu/people">Staff</a></li>
</ul>
<div id="map" class="center" style="height:500px;"></div>
```

- HTML5 Resources
 - [Dive into HTML5](#)
 - [Mozilla Developer Network](#)

Cascading Style Sheet (CSS)

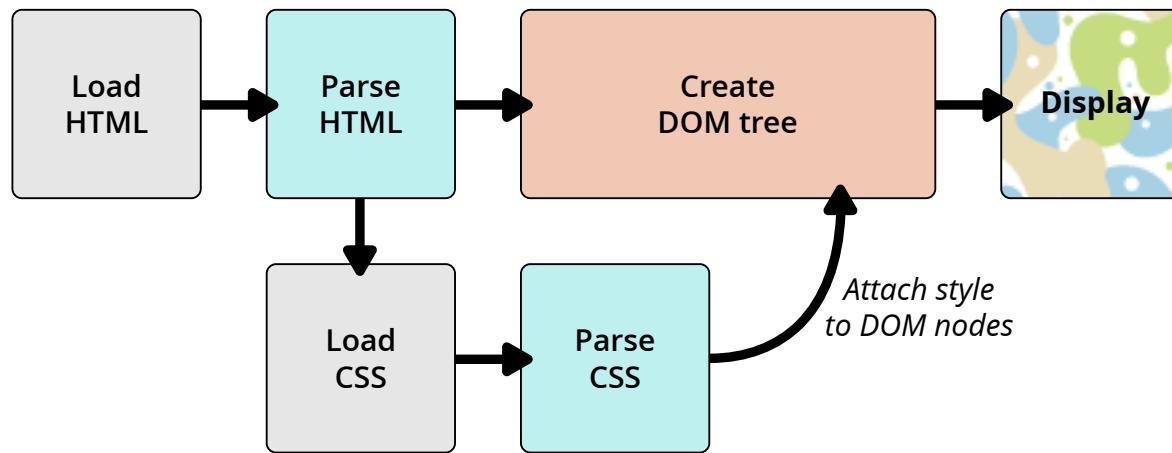
- Not a programming or markup language
- Role » Describes presentation of a document written in HTML or XML (e.g. SVG)
- Selectors, properties, and values
- Properties and Attributes

```
#map {  
    height: 500px;  
}  
h1 {  
    font-family: "Helvetica", "sans-serif";  
    font-weight: bold;  
    background-color: #000;  
}  
.chart {  
    float: left;  
}
```

- CSS3 Resources
 - [Mozilla Developer Network](#)

Document Object Model (DOM)

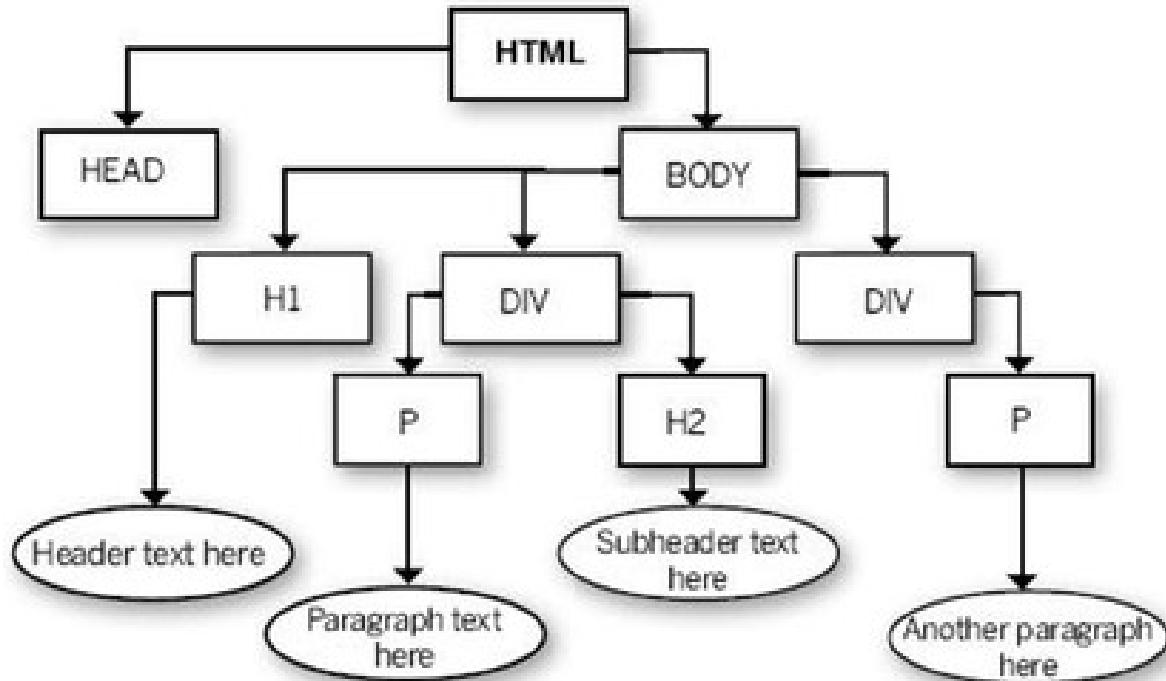
- Structured representation of the document (a tree) created by the browser
- HTML you write is parsed by the browser and turned into the DOM
- Programming interface for HTML and SVG documents
- JavaScript manipulates the DOM



Conceptual Web Page

```
<!DOCTYPE html>
<html>
<head>
    <!-- head content -->
    <style type="text/css">
        div {
            color: red;
        }
    </style>
</head>
<body>
    <h1>Header text here</h1>
    <div>
        <h2>Subheader text here</h2>
        <p>Paragraph text here</p>
    </div>
    <div>
        <p>Another paragraph here</p>
    </div>
    <script>
        <!-- JavaScript content -->
    </script>
</body>
</html>
```

The DOM for conceptual web page



- Understanding the DOM

JavaScript

- "Easy to learn, hard to master"
- Role » Creating interaction
- Interpreted by your browser
- Asynchronous (code executes in background after client-browser receives data from server)
- Get familiar with:
 - Working with json objects
 - Array functions (forEach, filter, map)
 - Method chaining, Callbacks, Closures, Modules
- JS Resources
 - [Mozilla Developer Network](#)

SVG

```
<svg width="300" height="180">
  <rect x="10" y="20" width="20" height="50" fill="blue"
        stroke="red" stroke-width="1"/>

  <circle cx="100" cy="100" r="25" fill="red"
        stroke="#ddd" stroke-width="5"></circle>

  <g transform="translate(5, 15)">
    <text x="0" y="0">My graphic</text>
  </g>

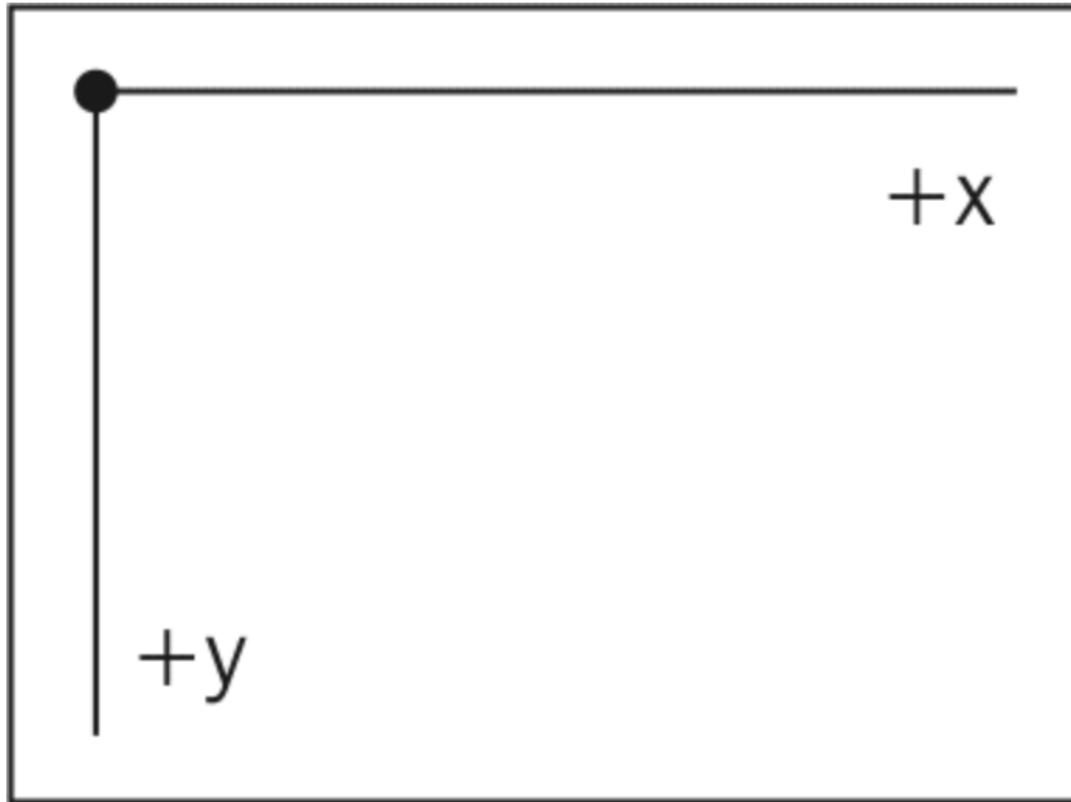
  <g transform="translate(5, 55)">
    <!-- M: move to (jump)
        L: line to
        Q: curve to (quadratic) -->
    <path d="M0,50 L50,0 Q100,0 100,50"
          fill="none" stroke-width="3" stroke="black" />
  </g>

  <g transform="translate(5, 105)">
    <!-- C: curve to (cubic)
        Z: close shape -->
    <path d="M0,100 C0,0 25,0 125,100 Z" fill="black" />
  </g>
</svg>
```

- Play with this code on [SVG Graphic Primitives - JSFiddle](#)

SVG (contd.)

- Coordinate system, top-left is (0, 0)
- Depth depends on order in which shapes are drawn



SVG v/s Canvas

- HTML5 Canvas is a raster based format for drawing on the web
 - You can draw raster graphics with D3
 - You can only get pixel values on click event
 - Faster
- SVG (Scalable Vector Graphics) is a vector based format for drawing on the web
 - HTML has div and span, etc.; SVG has circle and rect, etc.
 - SVG is a DOM for graphical elements
 - You can attach events to SVG elements
 - Most D3 examples work with SVG

HTML, CSS and JavaScript playgrounds

- [Codepen](#), [JSFiddle](#), [JS Bin](#)
- Open this [Codepen link](#) in a new tab to explore how HTML, CSS and JavaScript work together.
- In-browser code editor built for creating and sharing d3.js examples
 - [Blockbuilder](#)
 - Integrated with Github and [blocks.org](#)
- Or set up a local testing server on your machine

Start a local testing server and review final project

- Set up a simple local HTTP server on Windows
 - [How do you set up a local testing server?](#)
 - Python library comes with a `SimpleHTTPServer` module
 - For today we will use Python installed by ArcGIS on the lab computers
- Open Windows Command Prompt
- Change directory to the workshop folder

```
cd Documents/workshop-intro-d3-leaflet
```

- Start the local server using python library installed by ArcGIS

```
C:\Python27\ArcGIS10.4\python -m SimpleHTTPServer
```

- Open a browser. Type `localhost:8000` into the url bar at top. You should see a list of directories & files. Click on the `final` directory

Web Development Resources

- If you are going to develop medium to large scale web apps learn more about using front-end development tools.
 - [Getting Started Web Development Guide](#)
 - [Front-end Handbook](#)
- Google Search
 - Favor results from Stack Exchange, Mozilla Developer Network, CSS-Tricks
- [caniuse](#) - Check which browsers support what features
- Web Design
 - [A List Apart](#)
 - [Webdesigner News](#) - curated stories
- [Eloquent Javascript](#)
- [Egghead.io](#) - Bite size web dev video training
- Lynda.com, Frontend Masters, CodeAcademy, Udacity, Coursera, etc.

Explore D3 Core Concepts and build a bar chart



Data-Driven Documents



D3 (Data Driven Documents)

- D3.js is a javascript for producing dynamic, interactive data visualizations in web browsers.
- Built on the fundamental web standards of HTML, CSS, SVG, and JavaScript.



- Created by [Mike Bostock](#), Vadim Ogievetsky and Jeffrey Heer (all at that time were part of the Stanford Visualization Group).

What does it do?

- It is *not* a graphics library, it is *not* a charting library, it is a general purpose data visualization library
- Has everything you need for visualizing complex data BUT you will not find commands like barchart, scatterplot, or piechart, or even map
- Visualizes data using web standards (HTML, Javascript, CSS, SVG)
- Heavily used in data journalism and visualization. New York Times has been the pioneer in data visualization and data journalism.
- Most and foremost a DOM manipulation library (in this regard similar to jquery).
- Provides handy utilities for processing data (array, time series, geo data)
- Comes with a lot of functions and methods to create mapping functions that will map your data directly to properties on html elements
- Filled with algorithms (voronoi, quadtrees, circle fitting, convex hull, projections)

Learning D3

One of the most interesting aspects about d3 is that it is the intersection between design and code, math and art, data and story - Ian Johnson

- Very active user community in Bay Area.
- [Bay Area d3 User Group](#).
- Rich ecosystem of examples showing visualization types, coding techniques:
 - [blockbuilder.org/](#)
 - [blocks.org/](#)
- [Navigating the D3 Ecosystem](#)
- How do I learn D3.js? [Quora thread](#)
- D3 version 3 not the same as [D3 version 4](#)

Resources

- Books
 - [Interactive Data Visualization for the Web](#)
 - [D3 Tips and Tricks](#)
- Curated lists of data viz and data viz research
 - [visualisingdata.com](#)
 - [The Chartmaker Directory](#)
- [Mike Bostock Blog](#)
- For geographic visualizations
 - [Learning D3.js 4 Mapping](#)
 - [Jason Davies](#), [Jason Davies Code Blocks](#)
- [API documentation](#)

Core Concepts - Declarative API

- **Declarative:** Tell D3 what you want, not how you want it done
- **API:** Methods and objects you use to interact with software
 - e.g. keyboard interface for a computer
- Describe a relationship between data and UI (User Interface) elements, D3 handles implementation details

```
d3.selectAll('p').style('color', 'red');
```

- Example: [D3- Declarative API](#)

Core Concepts - Selections

- D3 provides shorthand for selecting and manipulating DOM objects (similar to jQuery)
 - `d3.select()`, `d3.selectAll()`
- Create/Read/Update/Delete (CRUD) properties of selections
- Getters and setters
 - `selection.attr(name[, value])`
 - `selection.style(name[, value])`
 - `selectionclassed(name[, value])`
 - `selection.text(name[, value])`
 - `selection.html(name[, value])`
- Example: [D3 - Selections](#)

Core Concepts - Selections (contd.)

- `selection.append(type) && selection.insert(type[, before])`

```
d3.select('body').append('p');
```

- `selection.remove()`

```
d3.select('body').remove('p');
```

- Handle events - `selection.on()`

```
d3.selectAll('button').on('click', handleButtonClicks);
d3.select('body').on('mouseover', trackMousePosition);
```

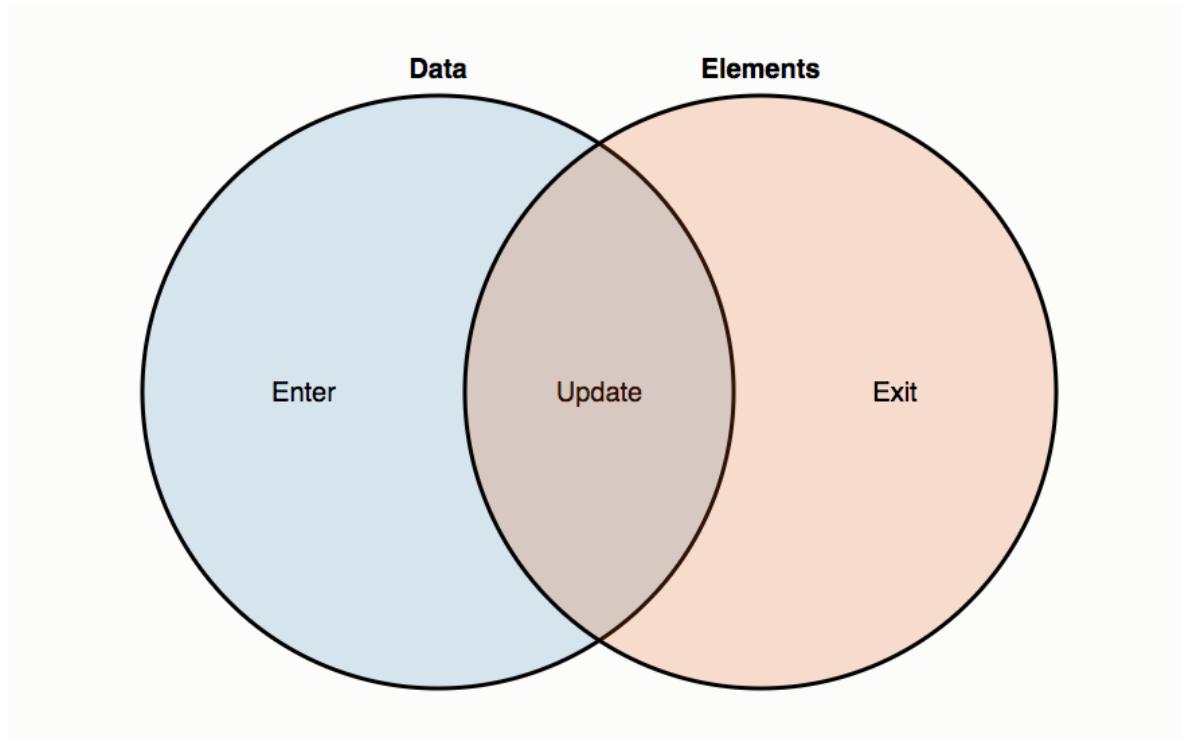
- Read documentation [d3-selection](#)

Core Concepts - Data Binding

- Real power of D3 is joining data and selections
- `selection.data()` - accepts an array of data items and joins them to the DOM elements in the selection
- Example: [D3 - Data Binding 1](#)
- Example with SVG: [D3 - Data Binding 2](#)

Core Concepts - Data Binding (contd.)

- What if we have more data than existing elements? Enter...
- ENTER, UPDATE and EXIT SELECTIONS
 - Mike Bostock - [Thinking with Joins](#)
 - Mike Bostock - [General Update Pattern examples](#)
 - Also [D3 In Depth book](#) (in progress) is great!



Core Concepts - Data Binding (contd.)

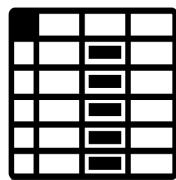
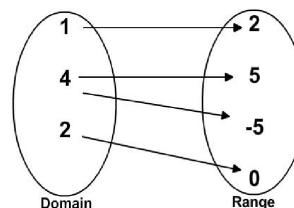
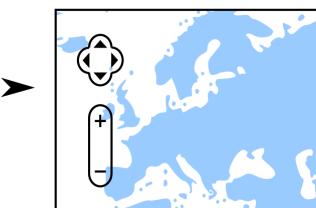
- Use with dynamic data
- Example: [D3 - Enter, Update, Exit](#)

Core Concepts - Scales

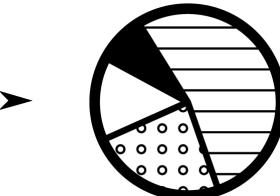
- Mapping: An operation that associates each element of a given set (the domain) with one or more elements of a second set (range)
- Any (computer) visualization problem can be described as a mapping problem.



Mapping function
Pixel coordinates =
 $f(\text{geographic coordinates})$
f aka projection



Mapping functions
 $\text{angle} = f(\text{data})$
 $\text{color} = f(\text{data})$



horse
bunny
bird
cow

Pferd
Kaninnchen
Vogel
Kuh

```
const percentToDecimal = d3.scaleLinear()  
  .domain([0, 100])  
  .range([0, 1])  
  
percentToDecimal(0)    // 0  
percentToDecimal(50)   // 0.5  
percentToDecimal(100)  // 1
```

Core Concepts - Scales (contd.)

- Scales in practice

```
// specify chart dimensions
const width = 600;
const height = 400;

// create a scale to map scores to bar widths
// test scores are stored as percentages
// a score of 100 should create a full-width bar
const xScale = d3.scaleLinear()
  .domain([0, 100])
  .range([0, width]);

d3.select('#chart')
  .selectAll('div')
  .data(data)
  .enter()
  .append('div')
  .style('width', function(d) {
    // pass the score through the linear scale function
    return xScale(d[subject]) + 'px'
  });
});
```

Core Concepts - Scales (contd.)

- Lot of other scales, e.g. `d3.bandScale()`

```
// specify chart dimensions
const width = 600;
const height = 400;

// create a scale to map scores to bar widths
// test scores are stored as percentages
// a score of 100 should create a full-width bar
const xScale = d3.scaleLinear()
  .domain([0, 100])
  .range([0, width]);

const yScale = d3.scaleBand()
  .domain(data.map(function(d) { return d.name }))
  .range([0, height]);

d3.select('#chart')
  .selectAll('div')
  .data(data)
  .enter()
  .append('div')
  .style('width', function(d) {
    // pass the score through the linear scale function
    return xScale(d[subject]) + 'px'
  })
  .style('height', function(d) {
    return yScale.bandwidth() + 'px'
  });
});
```

D3 - Other concepts

- Lots of convenience functions for loading and manipulating data, working with arrays
- Axes, Scales, Colors
- Manipulate structure of your data - d3.nest()
- D3 [API](#) Reference
- Writing reusable components
 - A good place to start is by reading [Towards Resuable Charts](#) where Mike Bostock proposes a convention for creating reusable charts.
 - [This blog post](#) and [this book](#) do a great job on clarifying the reusable charts api with examples and explanations.

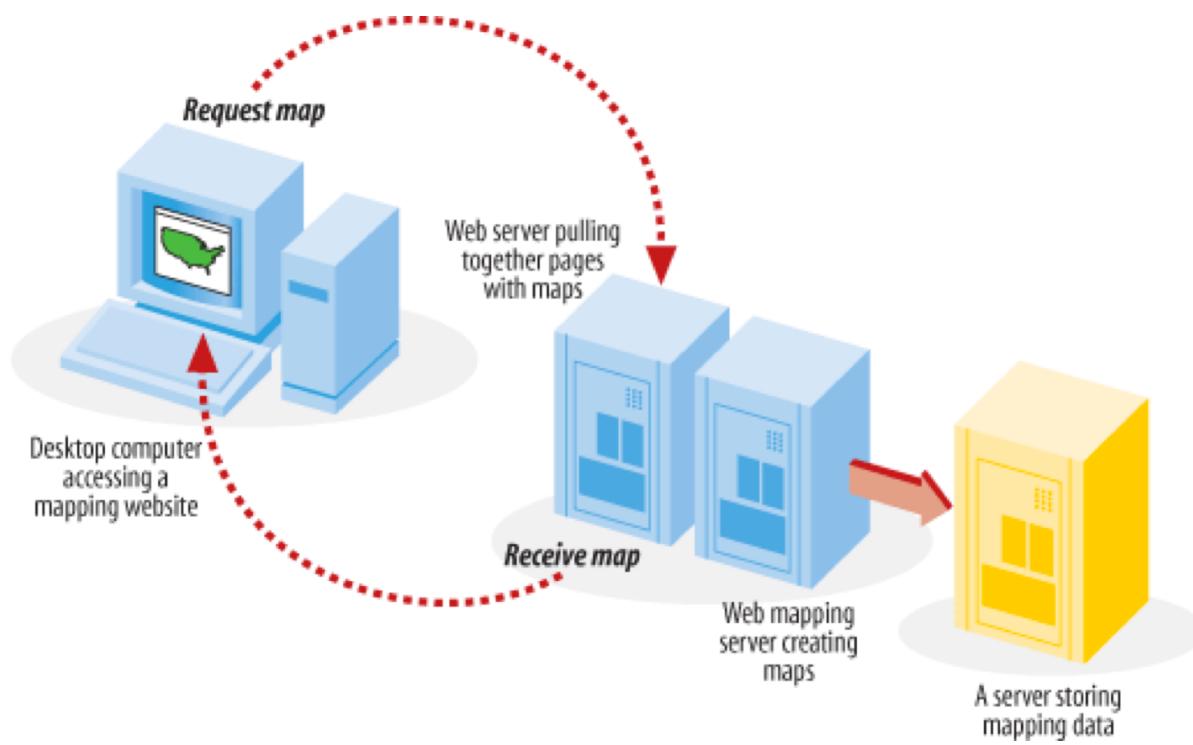
Other options for using D3

- Rickshaw, Highcharts, NVD3 are libraries built on top of D3
- D3 and [Crossfilter](#) (Fast Multidimensional Filtering for Coordinated Views)
- Open-source tools binding D3 to R, Python
- [Vega](#) - higher-level visualization specification language on top of D3

Web Mapping

What is a Web Map?

- Digital map that works in a web browser (as opposed to PDF/Image files)
 - Interactive
 - Developed using web application technologies – Not specialized mapping software



A Quick Recap of Web Mapping History

- 1996 MapQuest

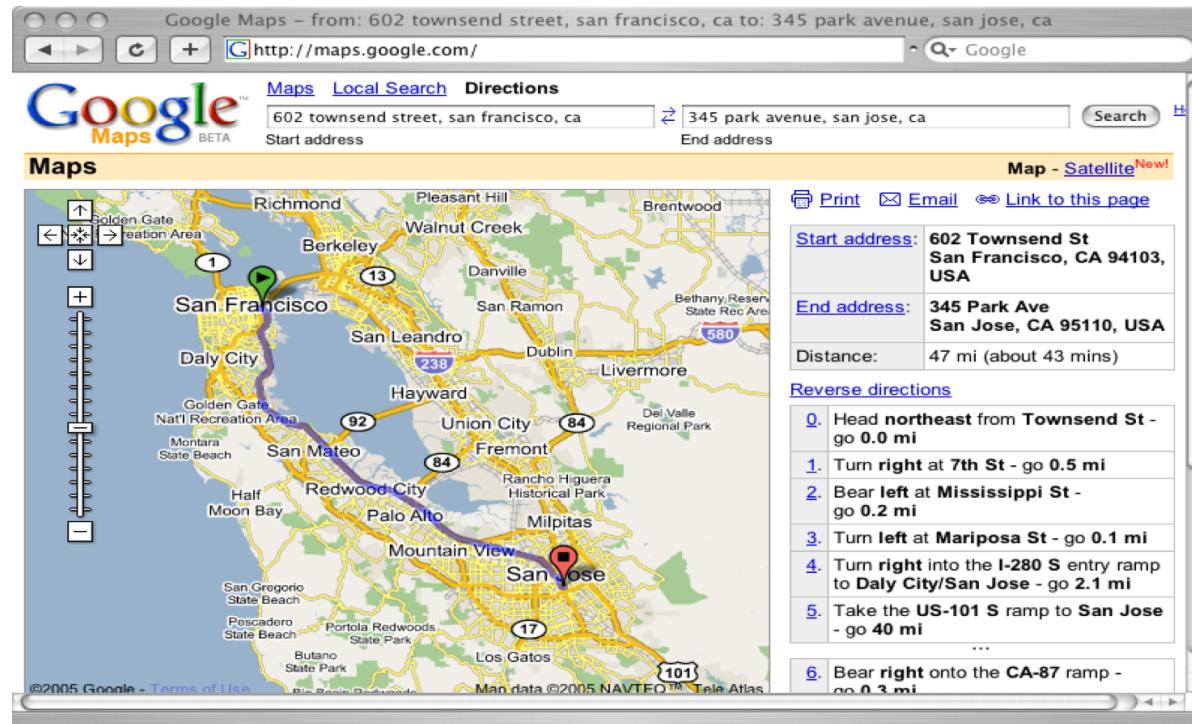


A Quick Recap of Web Mapping History (contd.)

- 1997 - 2004
 - MapInfo, AOL buys MapQuest
 - GPS Selective Availability ends (2000)
 - Microsoft MapPoint
 - Open Street Map (OSM) founded in UK (2004)
 - NASA releases WorldWind
 - Yahoo! Maps
 - Google buys Where2, Keyhold & ZipDash

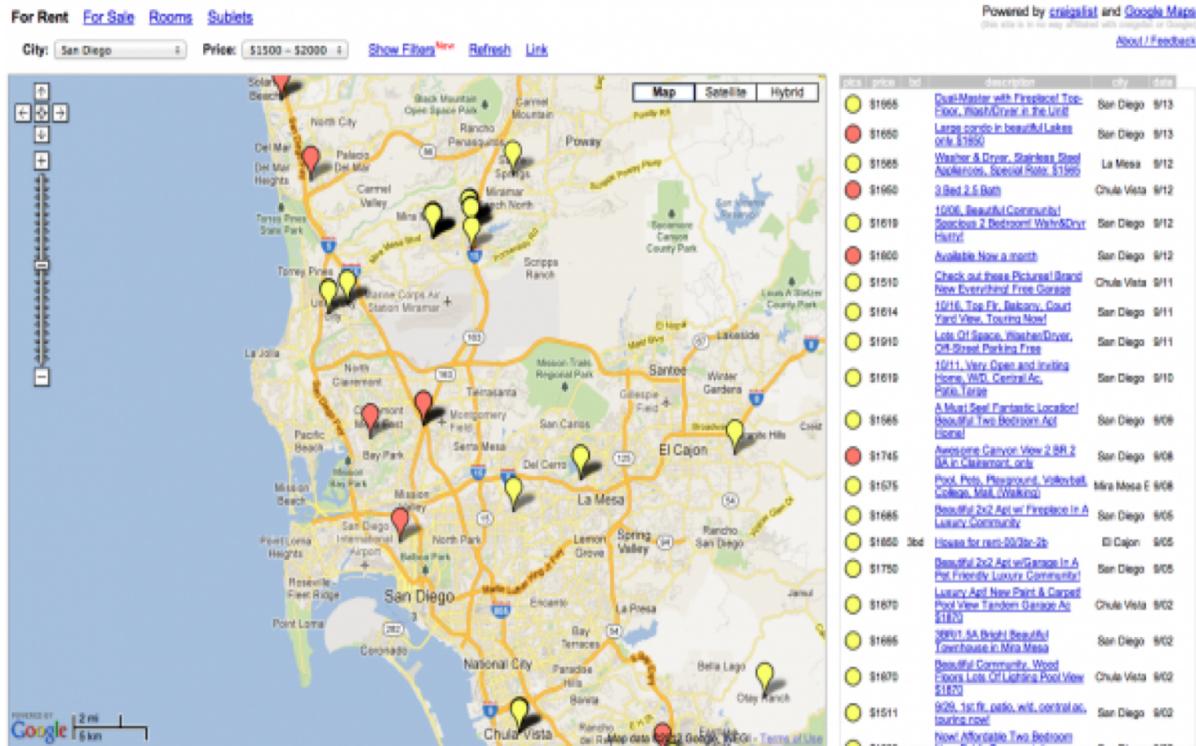
A Quick Recap of Web Mapping History (contd.)

- 2005 - Google Maps



A Quick Recap of Web Mapping History (contd.)

- 2005 - First Google Maps Mashup

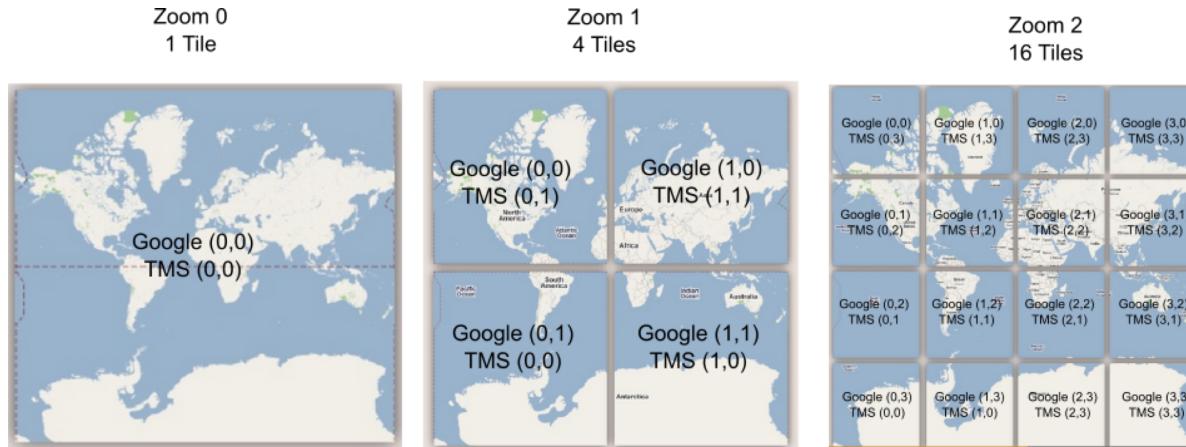


Technological Breakthroughs

- Slicing a square map into smaller square image tiles – Web Mercator projection
- Slippy maps with smooth zoom and pan – Web development techniques (AJAX) used for requesting data (map tiles) from server asynchronously (in the background)
- Open API
- No plugins, seamless user experience
- Mapping as a service

Map Tiles

- 256 pixels x 256 pixels image files
- Pre-rendered or rendered on the fly
- 360 billion tiles to cover the whole world at 20 zoom levels



Fast forward to present

- Web Services, Open Data, Open Standards and Formats, Free and Open Source Software



Web Services

- APIs (Application Programming Interface)
 - Basemaps, Data, Geocoding, Navigation, Spatial Analysis • Commercial
 - Google, ESRI, CARTO, MapBox, Planet ... many more – Free and paid tiers
- Non-commercial
 - Data.gov, NPMAP, SF Open Data, NASA
 - Cal-Adapt, Landcarbon, Ecoengine @berkeleyGIF

Open Data

- Free
- Limited or no copyright restrictions
- Contributed by government, researchers, non-profits, citizens
 - World Bank, OpenStreetMap, NASA
- Pay attention to:
 - Attribution requirements
 - Restrictions on redistribution
 - Data Quality

Data - OpenStreetMap

- Collaborative project to create a free and editable map of the world
- Volunteered geographic information
- Primary output = DATA not the map
- Open Database License (ODBL)
 - License/Use cases
- Getting the data (planet.osm.org & other)
 - Planet extract (666 GB uncompressed; 30-40 GB compressed)
 - Smaller extracts for countries, metro areas available
 - Updated regularly
 - Tools for importing into PostgreSQL/PostGIS

Open Standards

- Data Formats
 - GeoJSON, TopoJSON, KML, Shapefile, WaterML, NetCDF...
- Specifications
 - Web Map Services, Web Feature Services, Vector Tiles

Open Standards - GeoJSON

- File format for spatial data common on the web
- Subset of JSON format (JavaScript Object Notation)
- Converting to GeoJSON
 - GDAL, geojson.io, fiona (python package), QGIS, ArcGIS
- TopoJSON
 - Extension that encodes topology – More compact than GeoJSON

Open Standards - Vector Tiles

- Instead of raster images data is returned in vector format
- More efficient
- Contains
 - Data and geometry
 - Instructions on styling, instructions on stitching data back together
- Style on the fly in browser instead of server (no need to pre-render)
- Formats
 - Mapbox Vector Tile (MVT), GeoJSON/TopoJSON vector tiles

FOSS (Free and Open Source Software)

- Dekstop
 - GRASS, QGIS
- Geo-enabled databases
 - PostgreSQL/PostGIS
- Software for exposing spatial data to the web
 - MapServer, Mapnik, GeoServer, Tilecache
- JavaScript mapping and data visualization libraries
- System libraries
 - GDAL, PROJ4

Leaflet

- Lightweight, simple & flexible open source JavaScript mapping library
- Created by [Vladimir Agafonkin](#). Great speaker, checkout some of his talks on Leaflet
- Mobile-friendly. Well documented [API](#), huge amount of [plugins](#).
- Use with other JS mapping libraries (like Esri-Leaflet) or by itself. Similar libraries - OpenLayers, ModestMaps, Polymaps.
- Carto.js and Mapbox.js libraries are built on top of Leaflet.
- [Leaflet FAQ](#)

What does it do?

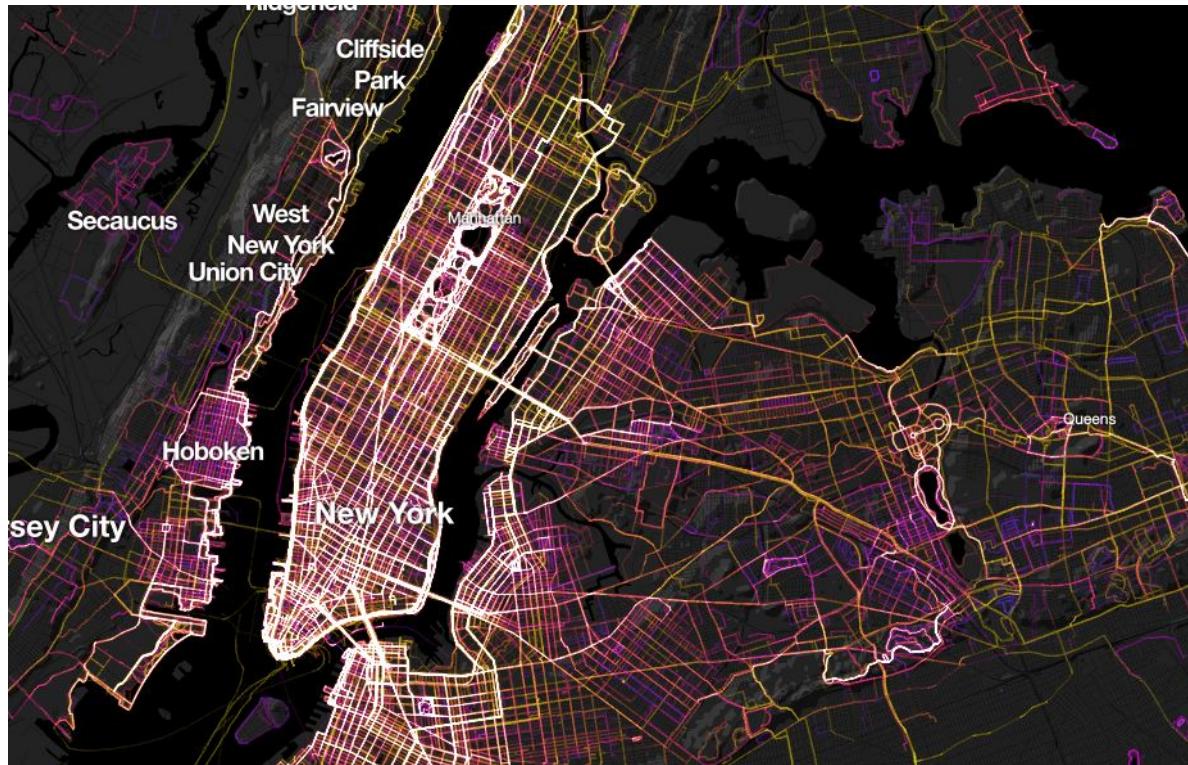
- Slippy maps with panning and zooming
- Provides functions for converting data into map layers
- Provides mouse interaction
- Does not provide any data
- You provide tile basemaps and data for overlays
- Easy to extend with plugins

Overlays - GeoJSON

```
var obj =  
{ "type": "FeatureCollection",  
  "features": [  
    { "type": "Feature",  
      "geometry": {"type": "Point", "coordinates": [102.0, 0.5]},  
      "properties": {"prop0": "value0"}  
    },  
    { "type": "Feature",  
      "geometry": {  
        "type": "Polygon",  
        "coordinates": [  
          [ [100.0, 0.0], [101.0, 0.0], [101.0, 1.0],  
            [100.0, 1.0], [100.0, 0.0] ]  
        ]  
      },  
      "properties": {  
        "prop0": "value0",  
        "prop1": {"this": "that"}  
      }  
    }  
  ]  
}
```

- [GeoJSON Specification](#), [Validate your geojson](#)
- Convert from shapefile - [geojson.io](#), [Mapshaper](#), GIS
- GeoJSON data coordinates should be in latitude, longitude (WGS 84)!!!

Basemaps & Overlays - Map Tiles



- How Web Maps Work
- Web Mercator - Standard projection for map tiles

Leaflet Exercises

- [Link to Github Repo](#)
- Before we begin:
 - Start a local server
 - Code Editor and Browser
 - Workflow for the exercises

Exercise 01 - Make a map and add overlay

- This exercise covers the basics for creating a Leaflet map element and adding a basemap to it.
- Four things you absolutely need for adding Leaflet map to a web page
 - Leaflet CSS file
 - Leaflet JavaScript file
 - A div element with an `id` attribute
 - A `css` height for your map div

Exercise 02 - Add and style data

- Add an overlay of tick collection point locations and style it

Exercise 03 - Working with Larger Datasets

- Geojson files with lots of data can take longer to load depending on size and network speed. And on the web you never want to keep your user waiting :-) There are couple of ways you can deal with larger files.
- Option 1 - Use the topojson format. Topojson is an extension of GeoJSON that encodes topology.
 - Convert geojson to topojson using mapshaper.org. You can also further reduce file size by simplifying the geometry. It depends on your app, how much detail you need to show.
- Option 2 - Host the data on service providers like CARTO, ArcGIS Online, Mapbox. These companies also provide their own web mapping libraries (some built on top of Leaflet)
- Great [mapmakers cheatsheet](#) on other options for dealing with large files with Leaflet and web maps in general.

Review Final Project with Leaflet Map and D3 chart

- In browser go to `localhost:8000/final`

Additions

- Content and styling
- Legend
- An example of using the Turf.js library for spatial analysis
- Integrated D3 bar chart

Discussion

You Pick!

Visualization, Web Development, Learning,

Deeper dive into some examples, ...