METIS DATA SCIENCE BOOK CAMP

DATA VISUALIZATION & D3 SUPER POWERS

Sebastian Gutierrez
DashingD3js.com
@dashingd3js

Metis Data Science Bootcamp
May 12th, 2015

SEBASTIAN GUTIERREZ

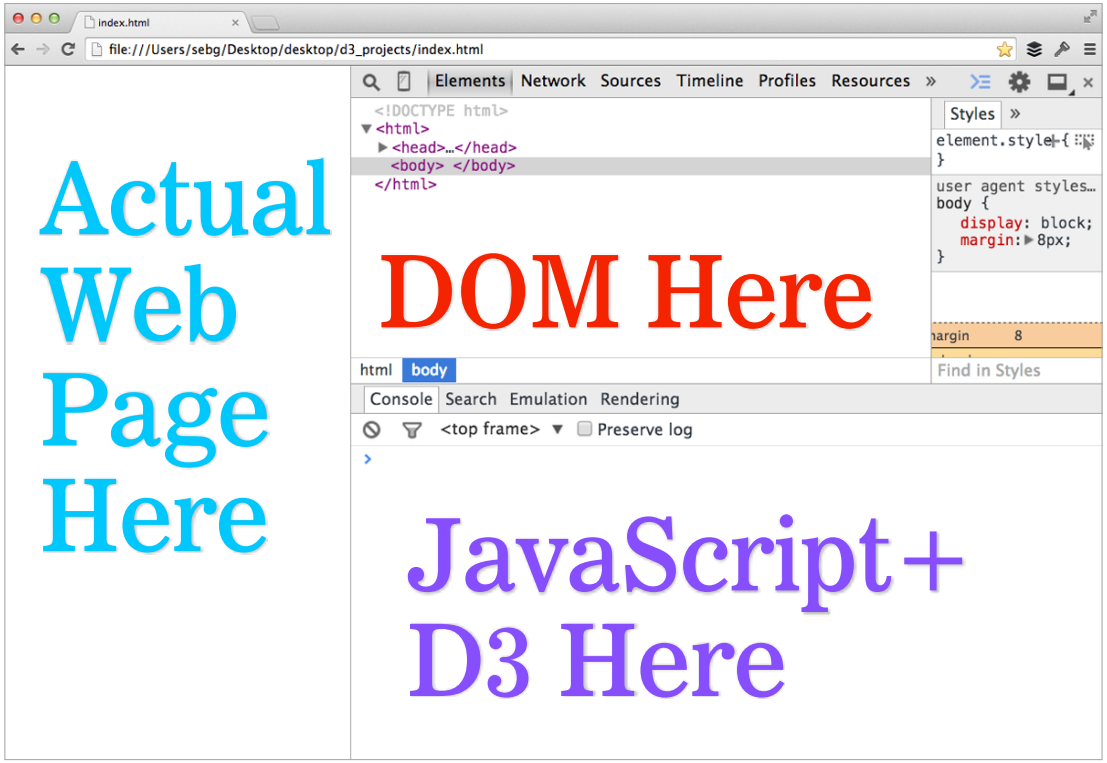
- Specializes in Actionable Data Visualizations
- Organizes of NYC D3.js Meetup
- Corporate training & consulting in D3 & Data Viz
- Author of “Data Scientists at Work” book
- DashingD3js.com (D3.js training website)
- DataScienceWeekly.org (weekly newsletter)

@dashingd3js
sebastian@dashingd3js.com

WEBSITE FOR LINKS, FILE, & CODE

- PDF of this presentation: http://bit.ly/20150512-metis-d3-pdf
- Files for this presentation: http://bit.ly/20150512-metis-d3-files

CHROME + DEVELOPER TOOLS



EXERCISE: CHROME

1. Get Google Chrome

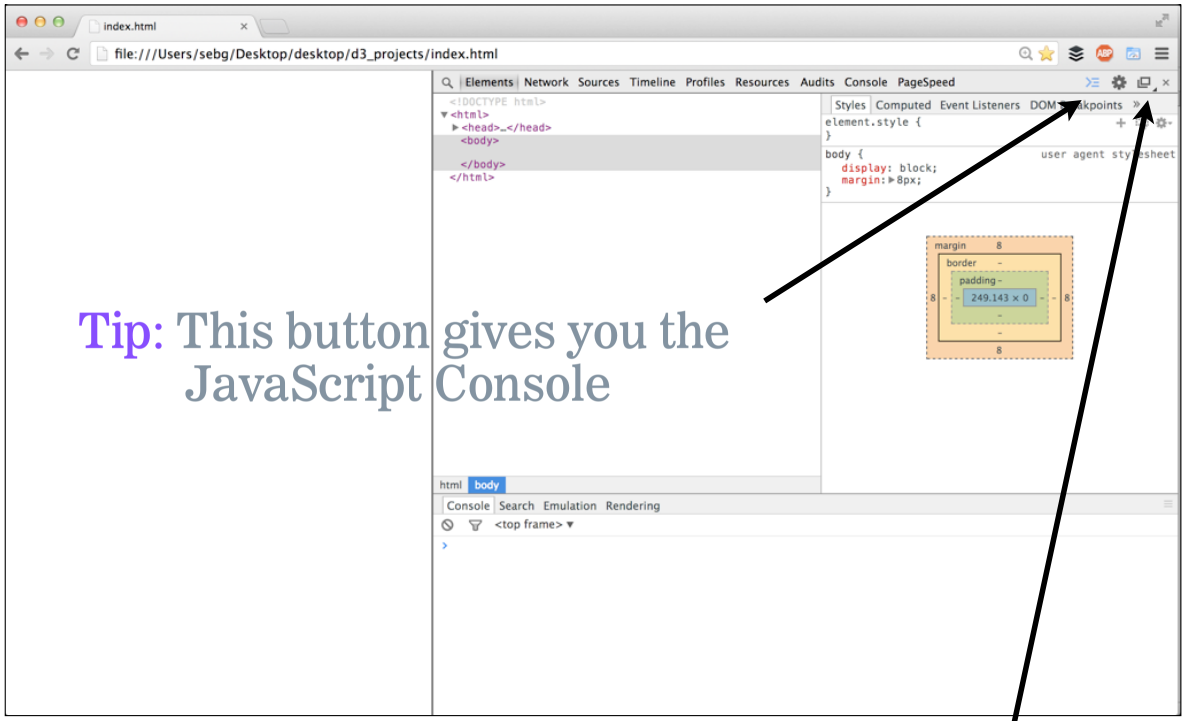
https://www.google.com/chrome

2. Open Up Chrome Dev Tools

View -> Developer -> Developer Tools

3. Setup Screen As Pictured

a) Press the Elements button
b) Press the button for the JS Console
c) Press and hold the button for the way to orient the JS Console / Elements



Tip: Hold this button down to see different ways to setup JavaScript Console

AGENDA

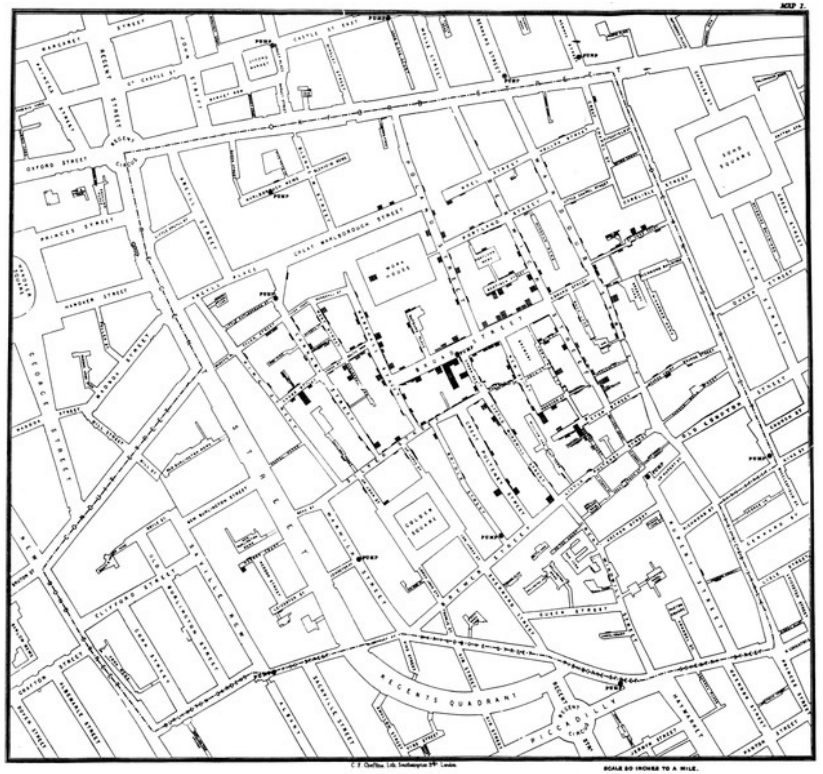
1. What is Data Visualization
2. D3 as a Data Visualization Tool
3. D3 Examples & bl.ocks.org
4. D3 Layouts
5. Conclusion

SECTION 1
WHAT IS DATA VISUALIZATION

PUBLIC HEALTH TOOL

John Snow's Map of the 1854 Broad Street Cholera Outbreak

Showed Cholera was being spread by water coming from a Broad Street Water Pump.

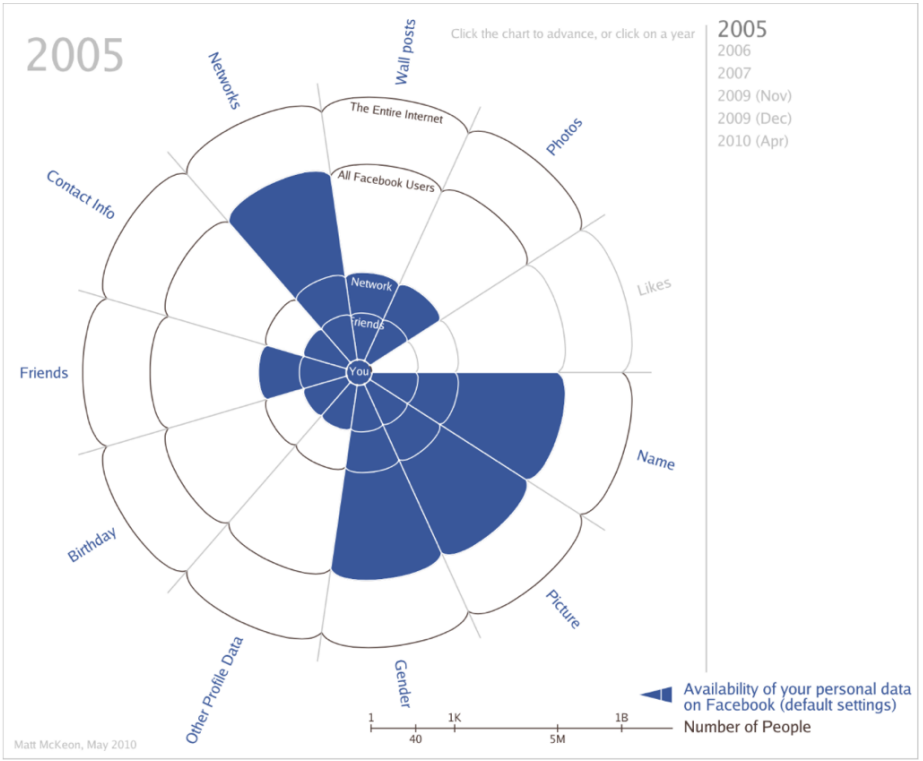


Source:
http://en.wikipedia.org/wiki/1854_Broad_Street_cholera_outbreak

PUBLIC / PRIVACY TOOL

How Public Your Personal Data is On Facebook By Default

Evolution of Facebook Default Privacy Settings



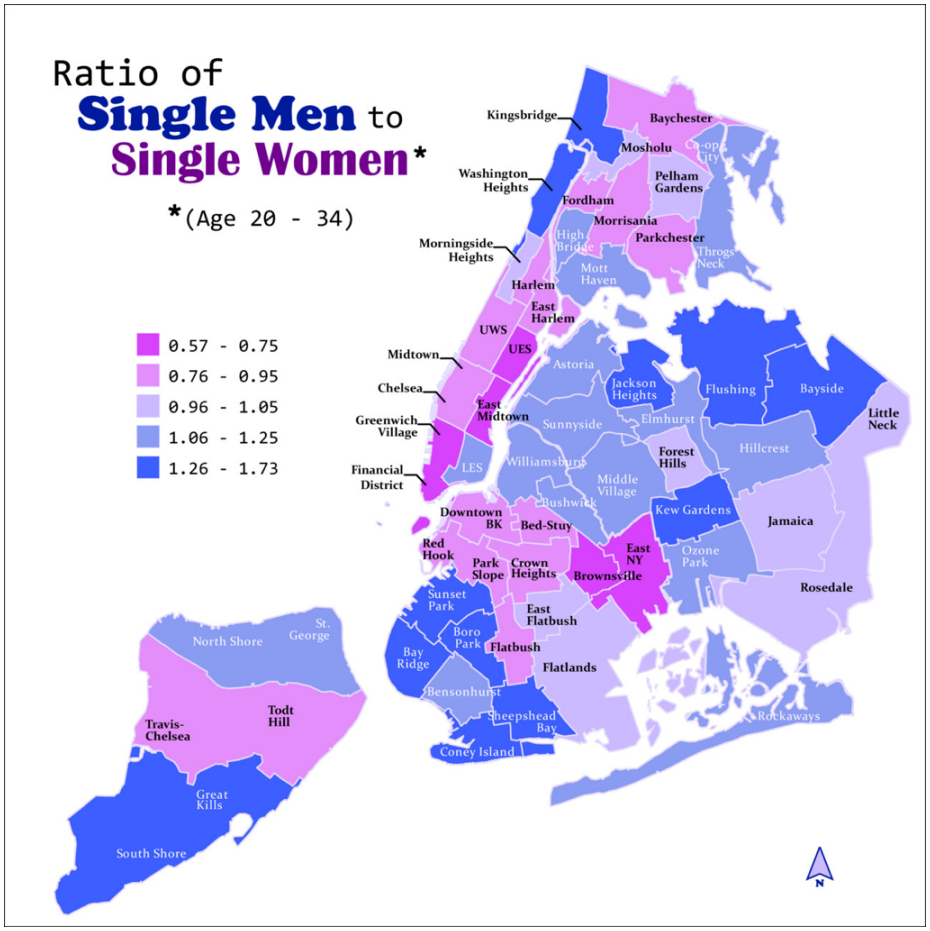
Source:
http://mattmckeon.com/facebook-privacy/

DATING TOOL

NYCEDC

Ratio of Single Men to Single Women (Age 20 - 34)

New York City's population is 53% female and 47% male. Using Census data, NYCEDC analyzed only the population who are never married singles between the ages of 20 and 34.



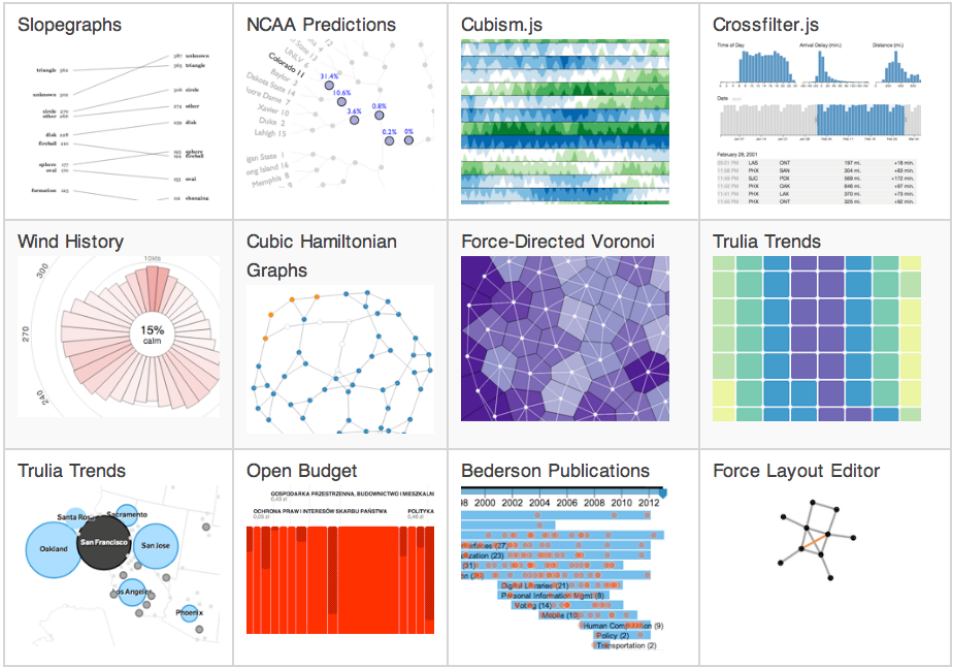
Source:
http://nycedc.tumblr.com/post/16175652438/ratio-of-single-men-to-single-women-in-nyc-new

DATA VISUALIZATION IS A TOOL

Everything Generates Data

Visualizing this data leads to understanding.

- Sports
- Commerce
- Weather
- Real Estate
- Publications
- Social Media
- Etc.....



Source:
https://github.com/mbostock/d3/wiki/Gallery

DATA VIZ GUIDES THINKING

Data in Columns & Rows

Audience has to think hard

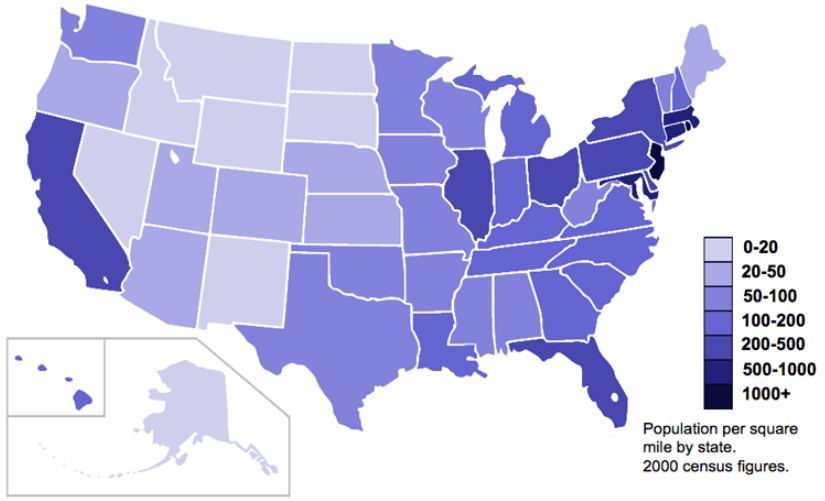
A39 | Maryland

	A	B	C	D	E	F	G	H
20	Alaska	D-3	R-3	R-3	R-3	R-3	R-3	R-3
21	Arizona	R-5	R-5	R-6	R-6	R-6	R-7	R-7
22	Arkansas	D-6	(\1)	R-6	D-6	R-6	R-6	R-6
23	California	D-40	R-40	R-45	R-45	R-45	R-47	R-47
24	Colorado	D-6	R-6	R-7	R-7	R-7	R-8	R-8
25	Connecticut	D-8	D-8	R-8	R-8	R-8	R-8	R-8
26	Delaware	D-3	R-3	R-3	D-3	R-3	R-3	R-3
27	District of Columbia	D-3	D-3	D-3	D-3	D-3	D-3	D-3
28	Florida	D-14	R-14	R-17	D-17	R-17	R-21	R-21
29	Georgia	R-12	(\1)	R-12	D-12	D-12	R-12	R-12
30	Hawaii	D-4	D-4	R-4	D-4	D-4	R-4	D-4
31	Idaho	D-4	R-4	R-4	R-4	R-4	R-4	R-4
32	Illinois	D-26	R-26	R-26	R-26	R-26	R-24	R-24
33	Indiana	D-13	R-13	R-13	R-13	R-13	R-12	R-12
34	Iowa	D-9	R-9	R-8	R-8	R-8	R-8	D-8
35	Kansas	D-7	R-7	R-7	R-7	R-7	R-7	R-7
36	Kentucky	D-9	R-9	R-9	D-9	R-9	R-9	R-9
37	Louisiana	R-10	(\1)	R-10	D-10	R-10	R-10	R-10
38	Maine	D-4	D-4	R-4	R-4	R-4	R-4	R-4
39	Maryland	D-10	D-10	R-10	D-10	D-10	R-10	R-10
40	Massachusetts	D-14	D-14	D-14	D-14	R-14	R-13	D-13
41	Michigan	D-21	D-21	R-21	R-21	R-21	R-20	R-20
42	Minnesota	D-10	D-10	R-10	D-10	D-10	D-10	D-10
43	Mississippi	R-7	(\1)	R-7	D-7	R-7	R-7	R-7
44	Missouri	D-12	R-12	R-12	D-12	R-12	R-11	R-11
45	Montana	D-4	R-4	R-4	R-4	R-4	R-4	R-4
46	Nebraska	D-5	R-5	R-5	R-5	R-5	R-5	R-5
47	Nevada	D-3	R-3	R-3	R-3	R-3	R-4	R-4
48	New Hampshire	D-4	R-4	R-4	R-4	R-4	R-4	R-4
49	New Jersey	D-17	R-17	R-17	R-17	R-17	R-16	R-16
50	New Mexico	D-4	R-4	R-4	R-4	R-4	R-5	R-5

Source:
http://www.census.gov

Data Visualization

Audience has to think less hard



Source:
http://commons.wikimedia.org/wiki/File:US_2000_census_population_density_map_by_state.svg

"RAW DATA" IS HARD TO READ

Anscombe's quartet

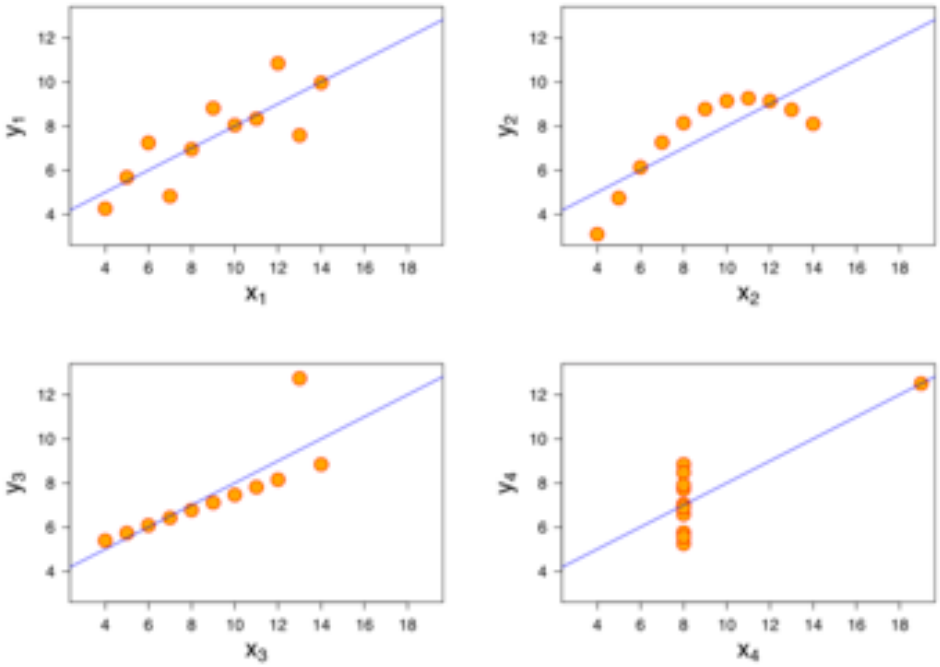
I		II		III		IV	
x	y	x	y	x	y	x	y
10.0	8.04	10.0	9.14	10.0	7.46	8.0	6.58
8.0	6.95	8.0	8.14	8.0	6.77	8.0	5.76
13.0	7.58	13.0	8.74	13.0	12.74	8.0	7.71
9.0	8.81	9.0	8.77	9.0	7.11	8.0	8.84
11.0	8.33	11.0	9.26	11.0	7.81	8.0	8.47
14.0	9.96	14.0	8.10	14.0	8.84	8.0	7.04
6.0	7.24	6.0	6.13	6.0	6.08	8.0	5.25
4.0	4.26	4.0	3.10	4.0	5.39	19.0	12.50
12.0	10.84	12.0	9.13	12.0	8.15	8.0	5.56
7.0	4.82	7.0	7.26	7.0	6.42	8.0	7.91
5.0	5.68	5.0	4.74	5.0	5.73	8.0	6.89

- Mean of x: 9
- Variance of x: 11
- Mean of y: 7.50
- Variance of y: 4.12
- Correlation between x and y: 0.816
- Linear regression line for each case:
  $y = 3.00 + 0.500 * x$

DATA VIZ DRIVES INTUITION OF DATA

Anscombe's quartet

I		II		III		IV	
x	y	x	y	x	y	x	y
10.0	8.04	10.0	9.14	10.0	7.46	8.0	6.58
8.0	6.95	8.0	8.14	8.0	6.77	8.0	5.76
13.0	7.58	13.0	8.74	13.0	12.74	8.0	7.71
9.0	8.81	9.0	8.77	9.0	7.11	8.0	8.84
11.0	8.33	11.0	9.26	11.0	7.81	8.0	8.47
14.0	9.96	14.0	8.10	14.0	8.84	8.0	7.04
6.0	7.24	6.0	6.13	6.0	6.08	8.0	5.25
4.0	4.26	4.0	3.10	4.0	5.39	19.0	12.50
12.0	10.84	12.0	9.13	12.0	8.15	8.0	5.56
7.0	4.82	7.0	7.26	7.0	6.42	8.0	7.91
5.0	5.68	5.0	4.74	5.0	5.73	8.0	6.89

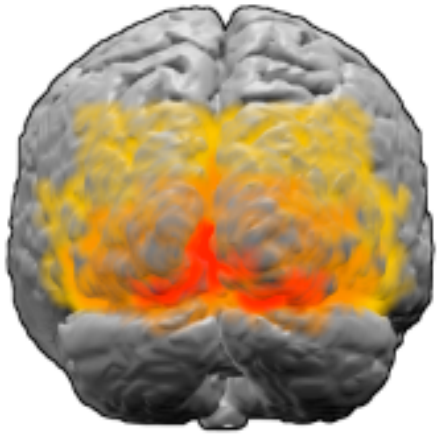


DATA VIZ USES VISUAL CORTEX

66% of stimuli reaching the brain are visual (Zaltman 1996)

50% of brain devoted to processing visual images (Bates & Cleese 2001)

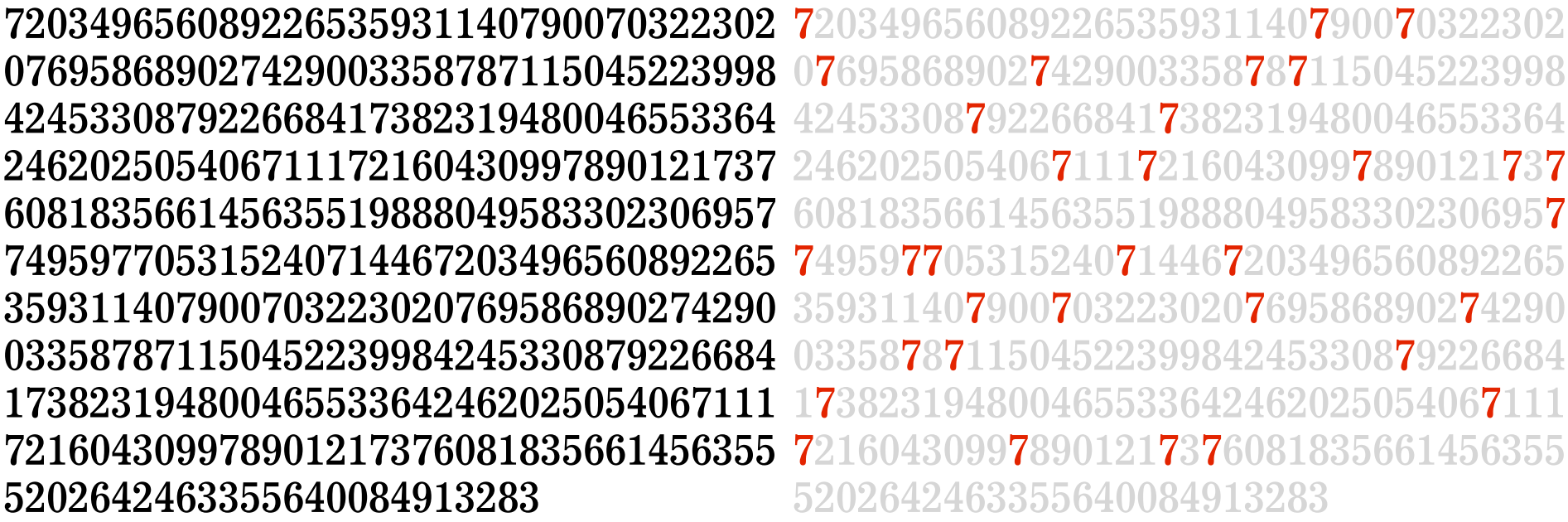
80% of learning is visually based (American Optometric Assoc. 1991)



Eye Image Source:
http://www.flickr.com/photos/orangeacid/234358923/

Brain Image Source:
http://en.wikipedia.org/wiki/File:Brodmann_areas_17_18_19.png

DATA VIZ USES BRAIN'S SHORTCUTS



BRAIN'S PRE-ATTENTIVE PROPERTIES

COLOR HUE

ORIENTATION

TEXTURE

POSITION & ALIGNMENT

COLOR BRIGHTNESS

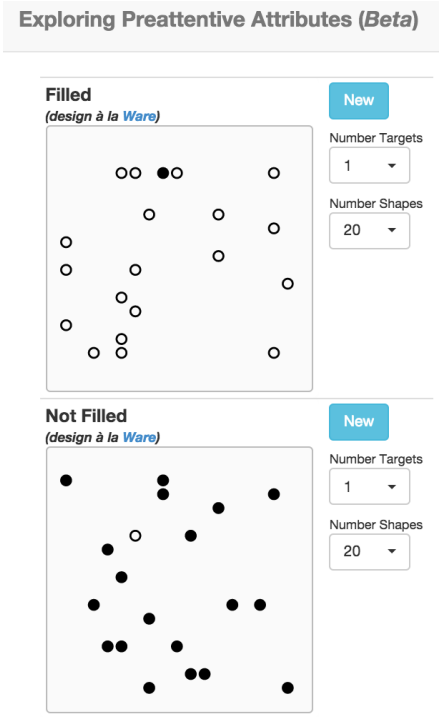
COLOR SATURATION

SIZE

SHAPE

D3 EXAMPLES OF PRE-ATTENTIVE PROPERTIES

http://learnforeverlearn.com/preattentive/



DATA VISUALIZATION IS A...

- Tool
- That helps guide thinking
- When trying to “understand” / “develop intuition about” data
- That uses the Visual Cortex
- To take advantage of the brain’s shortcuts
- And pre-attentive properties
- To achieve a goal.

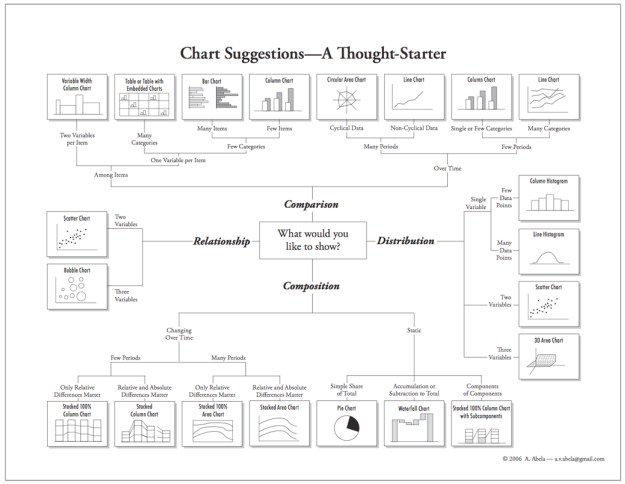
SEVEN MAIN TYPES OF DATA VIZ GOALS

Visual Encoding Depends on your goal -

- Time / Evolution
- Drill down
- Zoom out
- Contrast
- Intersections
- Factors
- Outliers

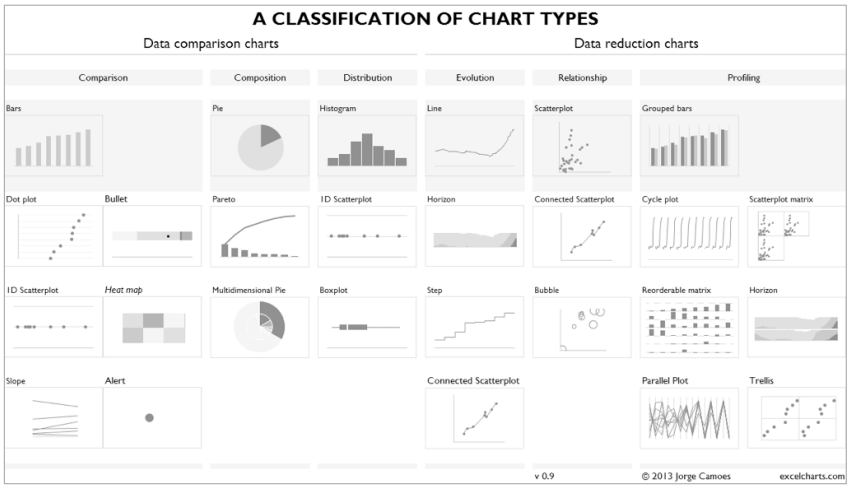
Source:
https://twitter.com/dataremixed

VISUALLY ENCODING YOUR GOAL



- Comparison
- Distribution
- Composition
- Relationship

Source:
© A. Abela - a.v.abela@gmail.com



Comparison
- Comparison
- Composition
- Distribution

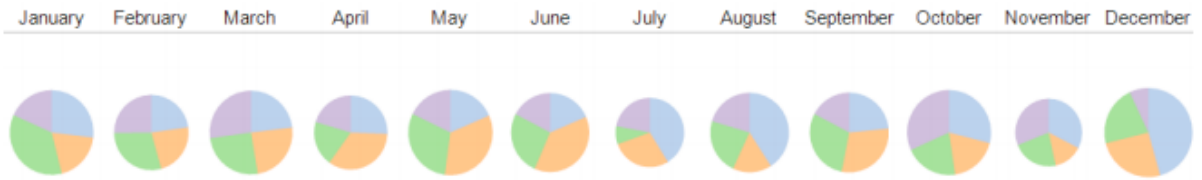
Reduction
- Evolution
- Relationship
- Profiling

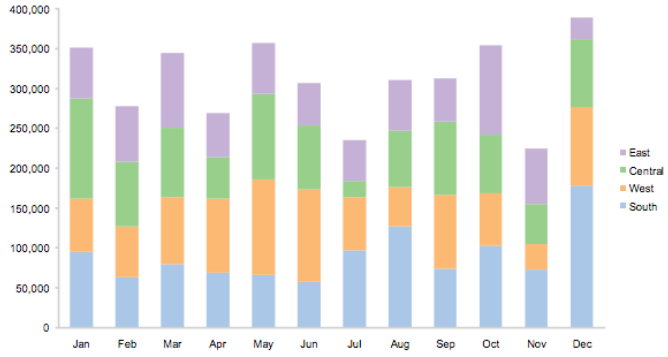
Source:
excelcharts.com

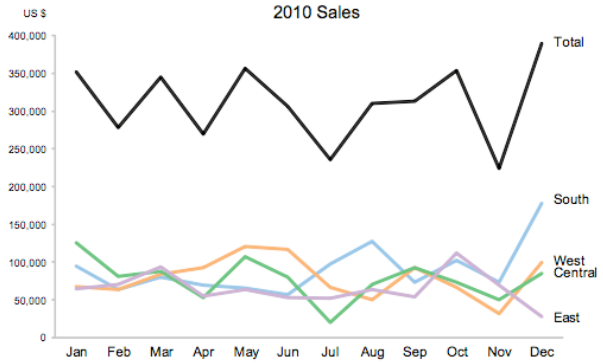
VISUAL ENCODING EFFECTIVENESS

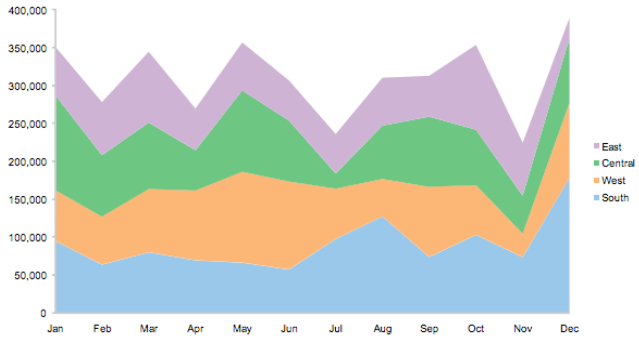
Depends on you goal

12 Months of Sales
- By Region
- By Month









Source:
Enrico Bertini, Assistant Professor at NYU-Poly (@filwd)

PURPOSE, DATA, AND AUDIENCE

Seth Godin

Three questions to ask your marketing team

Q1 - Who are you trying to reach?

Q2 - Why do they decide to support us?

Q3 - What do you need in order to make this happen more often?

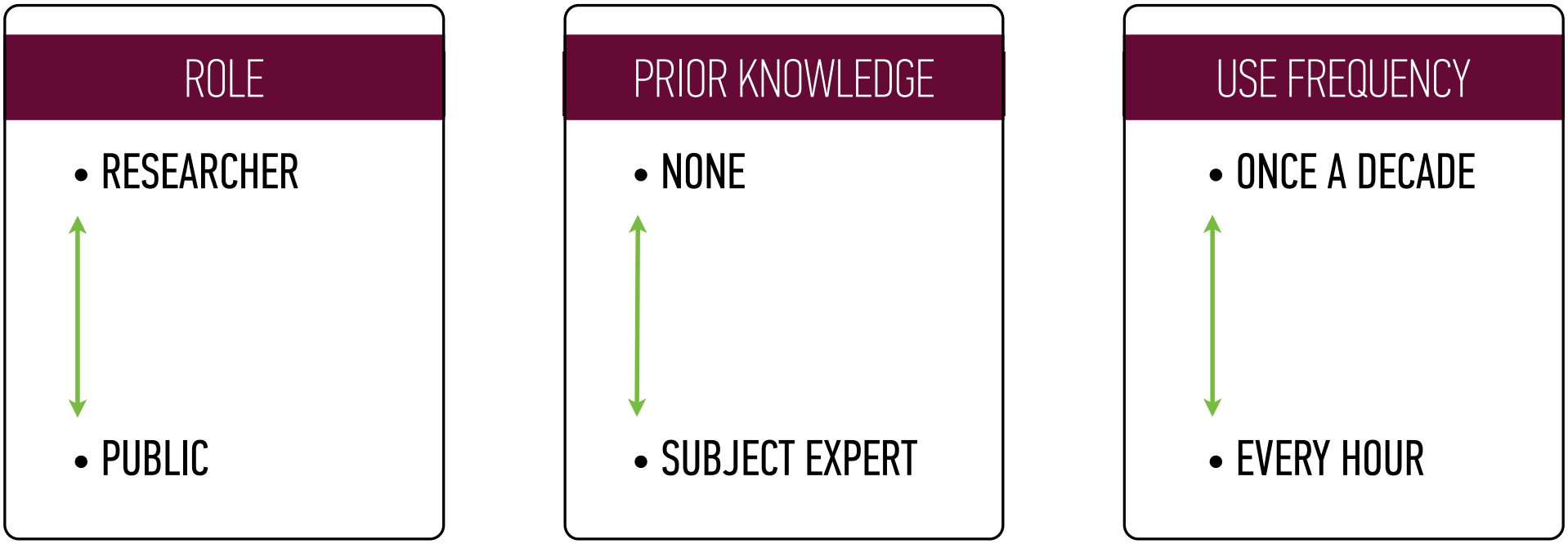
Three questions to ask of your data visualization

Q1 - Who are you trying to reach? (Audience)

Q2 - Why do they decide to support us? (Purpose)

Q3 - What do you need in order to make this happen? (Data)

NEED TO KNOW – AUDIENCE



NEED TO KNOW – HOW IT IS VIEWED

PRINT	WEB	VIDEO	PRESENTATION
• BLACK AND WHITE? • SOME COLOR? • ALL COLOR?	• INTERACTIVE? • NON-INTERACTIVE?	• NEWS SEGMENT? • COMMERCIAL? • SHOW?	• GUIDED? • UNGUIDED?

NEED TO KNOW – AVAILABLE DATA

PRIMARY	SECONDARY	GENERATED
• YOU COLLECT IT • YOU OWN IT • NOBODY ELSE HAS IT	• OTHERS COLLECT IT • OTHERS OWN IT • EVERYONE HAS IT	• FROM PRIMARY • FROM SECONDARY • FROM COMBINATION

NEED TO KNOW – GOAL / PURPOSE / WIN

HYPOTHESIS	GOAL	PARAMETERS
WHAT ARE WE TRYING TO SHOW?	HOW DO WE KNOW IF WE ACHIEVED IT?	WHAT ARE THE BOUNDARIES?

AUDIENCE VIEWS DATA VIZ, THEN WHAT

Seth Godin

- What are you trying to tell me?
- What do you want me to do now?

Source:
http://www.sethgodin.com/

DATA VISUALIZATION IS A...

- Tool
- That helps guide thinking
- When trying to “understand” / “develop intuition about” data
- That uses the Visual Cortex
- To take advantage of the brain’s shortcuts (pre-attentive properties)
- So that you / your audience can achieve a goal / purpose
- Without thinking too much

SECTION 2
D3 AS A DATA VISUALIZATION TOOL

D3 MADE BY DATA VIZ EXPERTS

2005: Prefuse (Java, Heer @ Berkeley)

2007: Flare (ActionScript, Heer @ Berkeley)

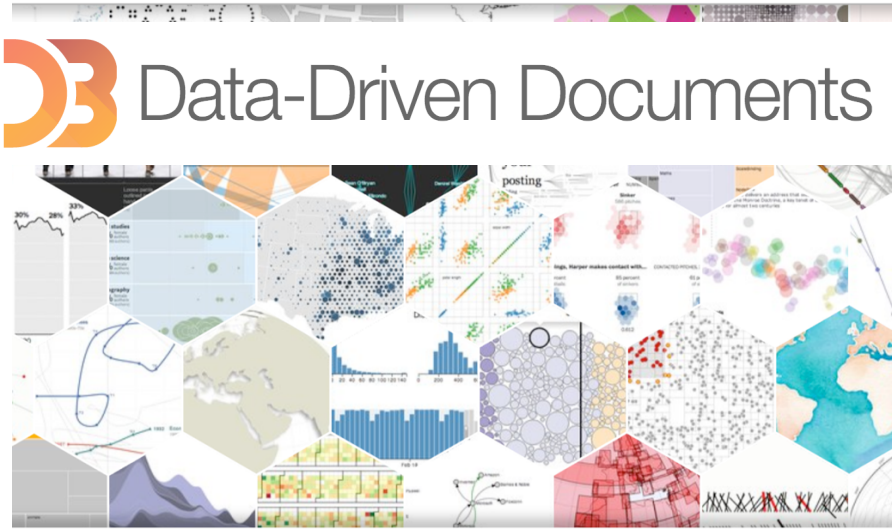
2009: Protovis (JavaScript, Heer & Bostock @ Stanford)

2011: D3 (JavaScript, Heer & Bostock @ Stanford)

TLDR:
very smart people thought very hard
about data visualization for a very long time

Source:
http://anna.ps/talks/fel/#/4

D3.JS KEY POINTS



- Manipulate Document (DOM)
- Data-Driven
- Visualization Components

Source: http://d3js.org

DOCUMENT MANIPULATION

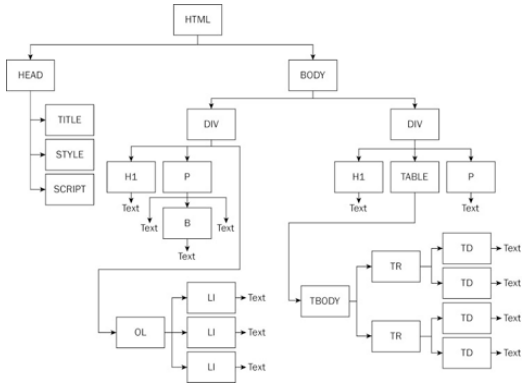
- Manipulate Document (DOM)
- Data-Driven
- Visualization Components

→

- Create Instructions
- Modify Instructions
- Remove Instructions

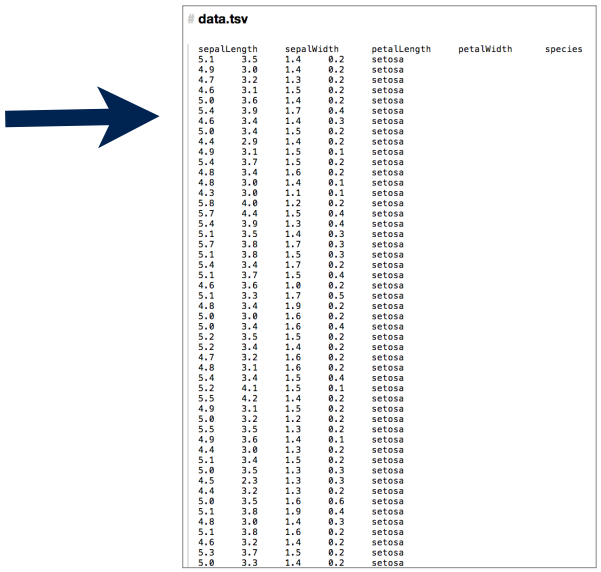
XML, HTML, & SVG

<tag attribute=value ...> ... </tag>

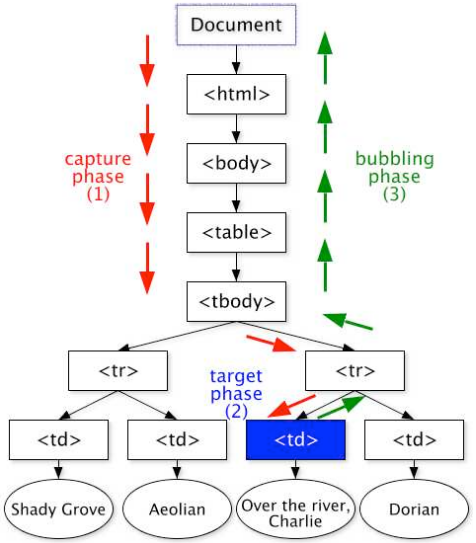


DATA-DRIVEN

- Manipulate Document (DOM)
- Data-Driven
- Visualization Components

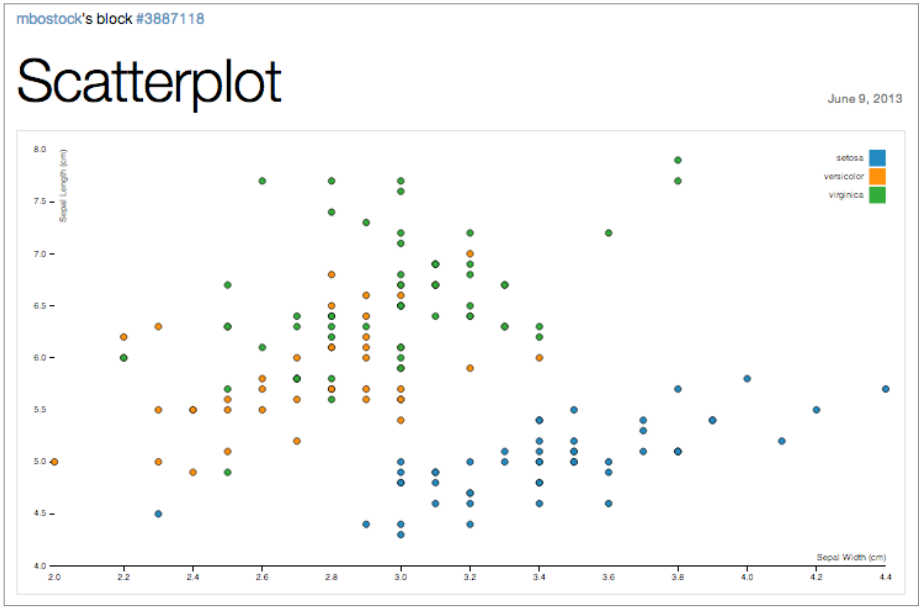

data.tsv

sepalLength	sepalWidth	petalLength	petalWidth	species
5.1	3.5	1.4	0.2	setosa
4.9	3.0	1.4	0.2	setosa
4.7	3.2	1.3	0.2	setosa
4.6	3.1	1.5	0.2	setosa
5.0	3.6	1.4	0.2	setosa
5.4	3.9	1.7	0.4	setosa
4.6	3.4	1.4	0.3	setosa
5.0	3.4	1.5	0.2	setosa
4.4	2.9	1.4	0.2	setosa
4.9	3.1	1.5	0.1	setosa
5.4	3.7	1.5	0.2	setosa
4.8	3.4	1.6	0.2	setosa
4.8	3.0	1.4	0.1	setosa
4.3	3.0	1.1	0.1	setosa
5.8	4.0	1.2	0.2	setosa
5.7	4.4	1.5	0.4	setosa
5.4	3.9	1.3	0.4	setosa
5.1	3.5	1.4	0.3	setosa
5.7	3.8	1.7	0.3	setosa
5.1	3.8	1.5	0.3	setosa
5.4	3.4	1.7	0.2	setosa
5.1	3.7	1.5	0.4	setosa
4.6	3.6	1.0	0.2	setosa
5.1	3.3	1.7	0.5	setosa
4.8	3.4	1.9	0.2	setosa
5.0	3.0	1.6	0.2	setosa
5.0	3.4	1.6	0.4	setosa
5.2	3.5	1.5	0.2	setosa
5.2	3.4	1.4	0.2	setosa
4.7	3.2	1.6	0.2	setosa
4.8	3.1	1.6	0.2	setosa
5.4	3.4	1.5	0.4	setosa
5.2	4.1	1.5	0.1	setosa
5.5	4.2	1.4	0.2	setosa
4.9	3.1	1.5	0.2	setosa
5.0	3.2	1.2	0.2	setosa
5.5	3.5	1.3	0.2	setosa
4.9	3.6	1.4	0.1	setosa
4.4	3.0	1.3	0.2	setosa
5.1	3.4	1.5	0.2	setosa
5.0	3.5	1.3	0.3	setosa
4.5	2.3	1.3	0.3	setosa
4.4	3.2	1.3	0.2	setosa
5.0	3.5	1.6	0.6	setosa
5.1	3.8	1.9	0.4	setosa
4.8	3.0	1.4	0.3	setosa
5.1	3.8	1.6	0.2	setosa
4.6	3.2	1.4	0.2	setosa
5.3	3.7	1.5	0.2	setosa
5.0	3.3	1.4	0.2	setosa



PRE-BUILT VISUALIZATION HELPERS

- Manipulate Document (DOM)
- Data-Driven
- Visualization Components



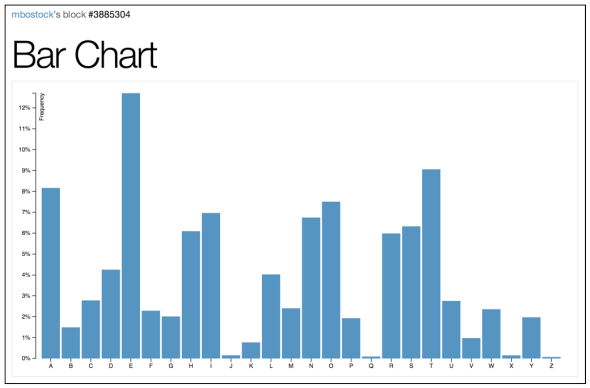
http://bl.ocks.org/mbostock/3887118

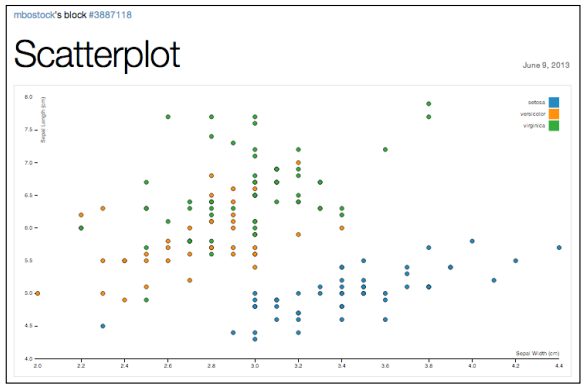
MANY LIBRARIES BUILT ON / USE D3

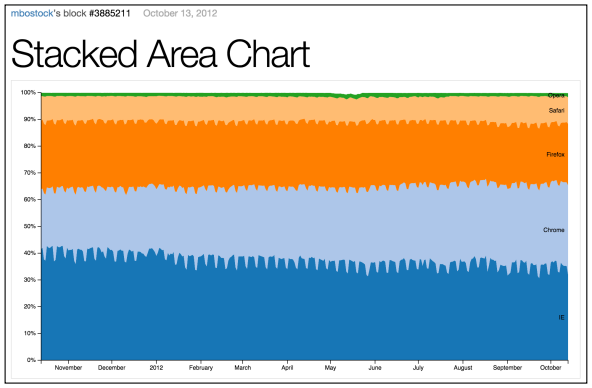
Cubism.js	n3-charts	Bokeh
NVD3.js	C3.js	Dimple.js
d3.chart	Visual Sedimentation	xCharts
Graphene	Raw	DVL
Rickshaw	rCharts	DC.js
Crossfilter.js	d4.js	Many Others...

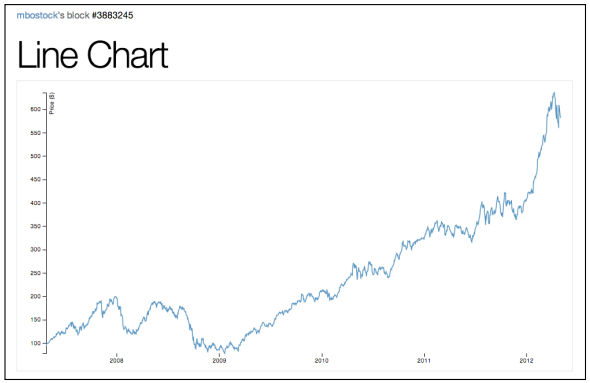
Source:
http://mikemcdearmon.com/portfolio/techposts/data-visualization-base-on-d3-js

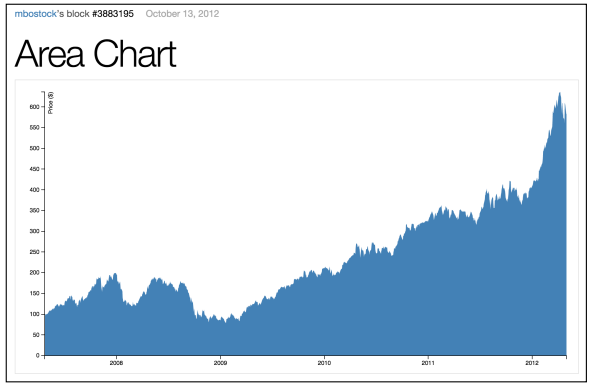
BASIC CHARTS

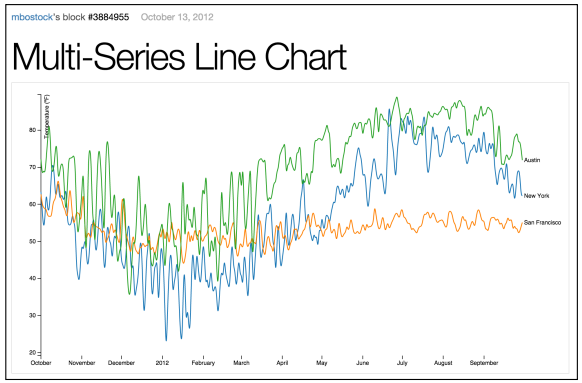








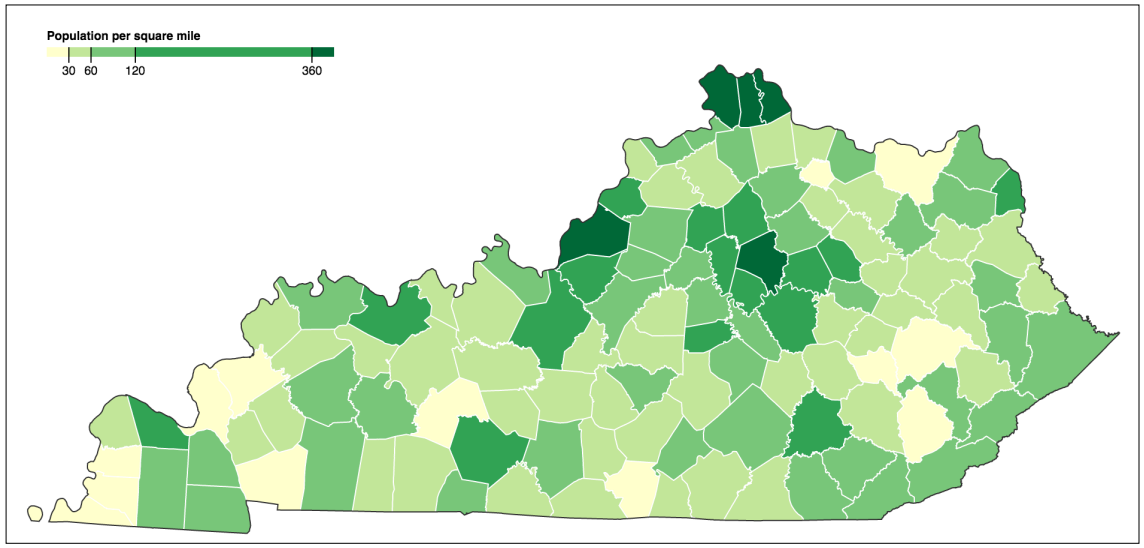




CHOROPLETH MAPS

Kentucky Population Density

http://bl.ocks.org/mbostock/5144735



D3 ALLOWS US TO EXPLOIT THE VISUAL INFORMATION SEEKING MANTRA

Source:
Ben Shneiderman, The Eyes Have It: A Task by Data Type Taxonomy for Information Visualizations. In Proceedings of the IEEE Symposium on Visual Languages, pages 336-343, Washington. IEEE Computer Society Press, 1996.

ONLINE CHARTS / DATA VISUALIZATION

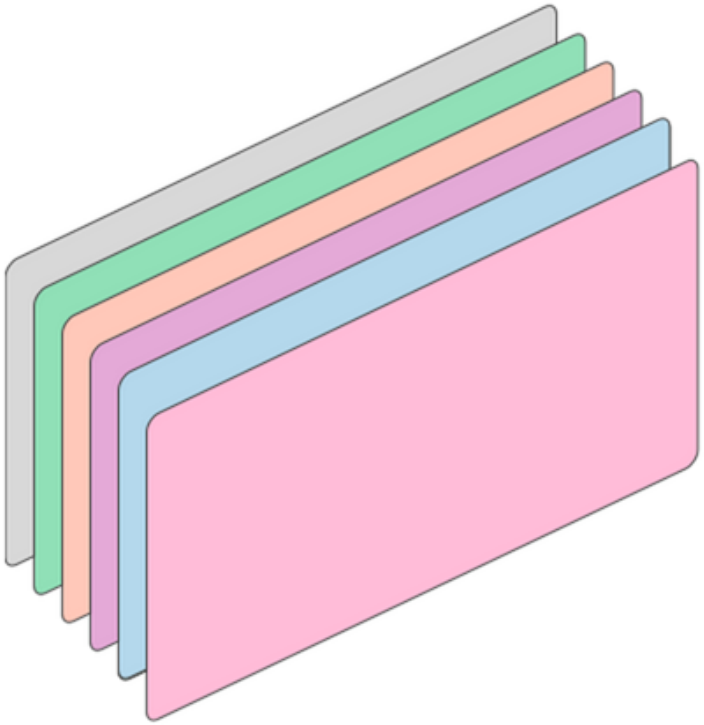


Chart +
Mouseover / tool tips
Data filters (ie date pickers...)
Different modes of data display
Data export
Drilldown for more specific charts

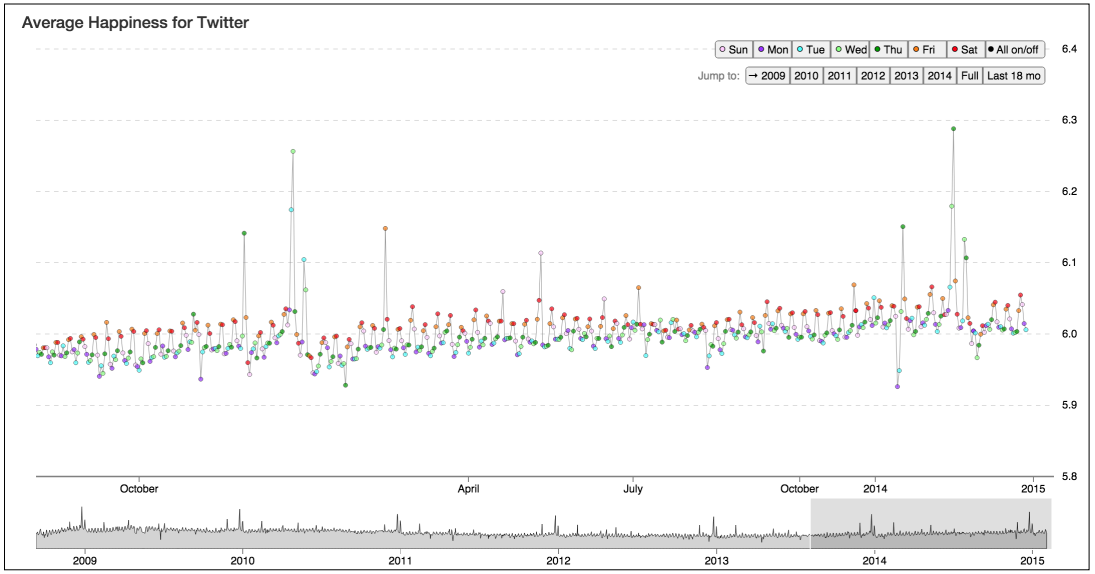
Source:
http://www.jeromecukier.net/blog/2015/02/07/charts-in-the-age-of-the-web/

INTERACTIVE LINE / AREA CHARTS

Hedonometer - Average happiness for twitter

With hedonometer.org we've created an instrument that measures the happiness of large populations in real time.

http://hedonometer.org/index.html

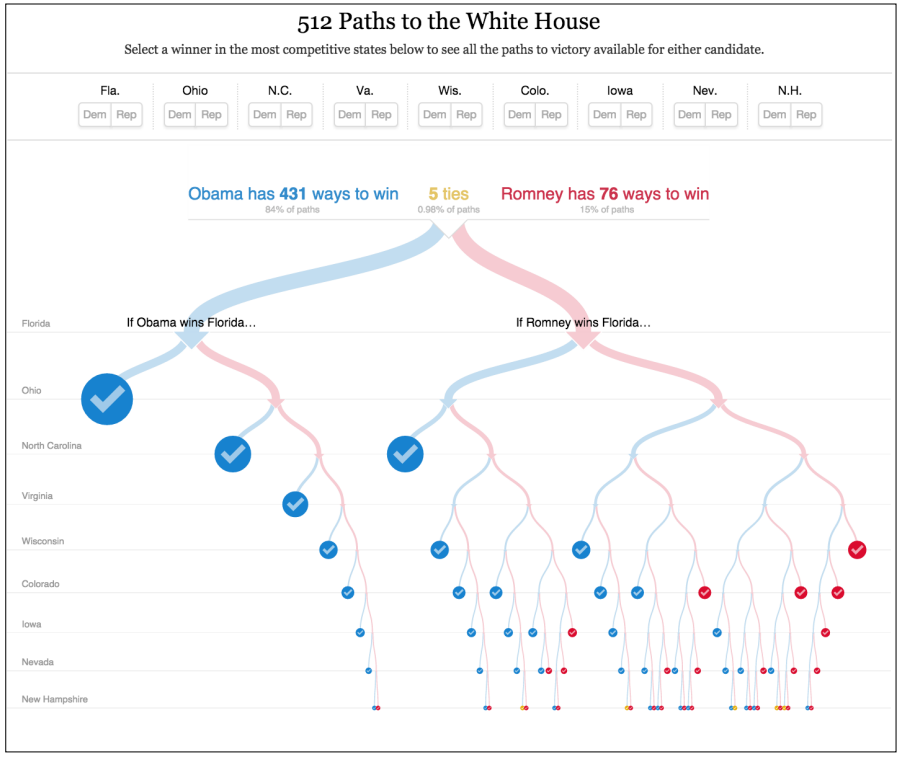


EXPLORATORY VISUALIZATIONS

512 Paths To The White House

2012 Political Tool to select a winner in the most competitive states below to see all the paths to victory available for either candidate

http://www.nytimes.com/interactive/2012/11/02/us/politics/paths-to-the-white-house.html

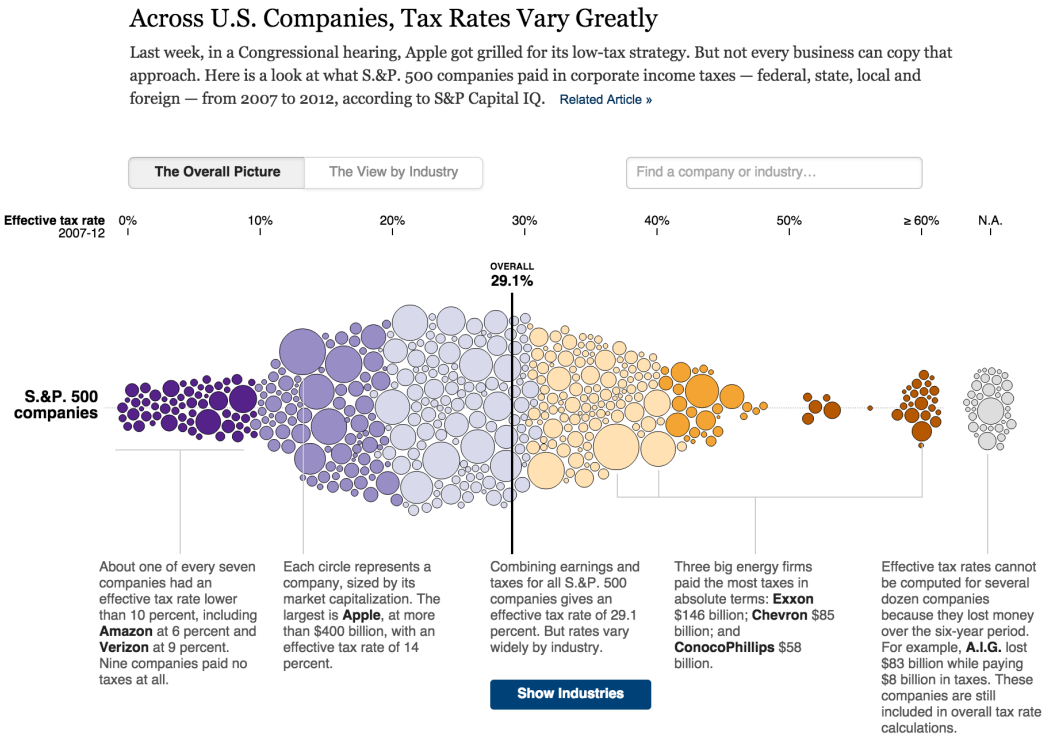


EXPLANATORY VISUALIZATIONS

S&P Corporate Tax Rate Chart

Here is a look at what S.&P. 500 companies paid in corporate income taxes — federal, state, local and foreign — from 2007 to 2012, according to S&P Capital IQ

http://www.nytimes.com/interactive/2013/05/25/sunday-review/corporate-taxes.html

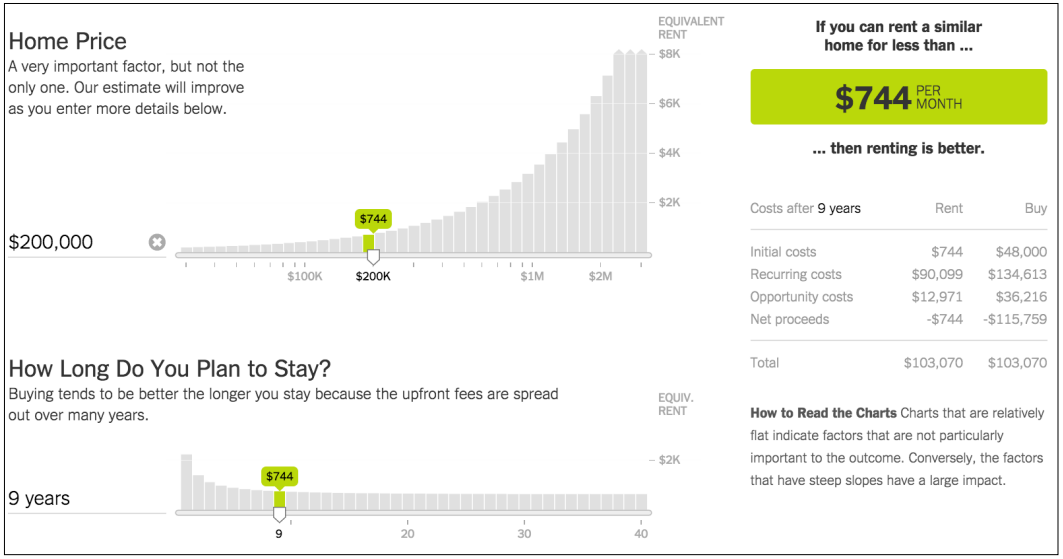


EXPERIMENTAL VISUALIZATIONS

Is It Better to Rent or Buy?

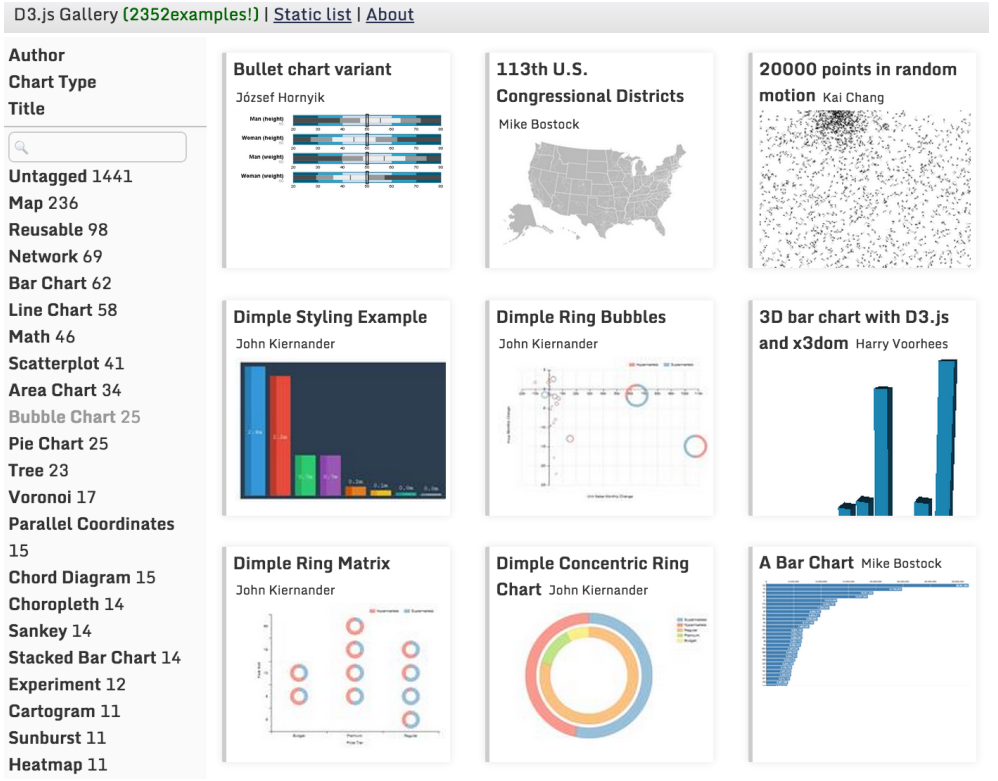
The choice between buying a home and renting one is among the biggest financial decisions that many adults make…

http://www.nytimes.com/interactive/2014/upshot/buy-rent-calculator.html



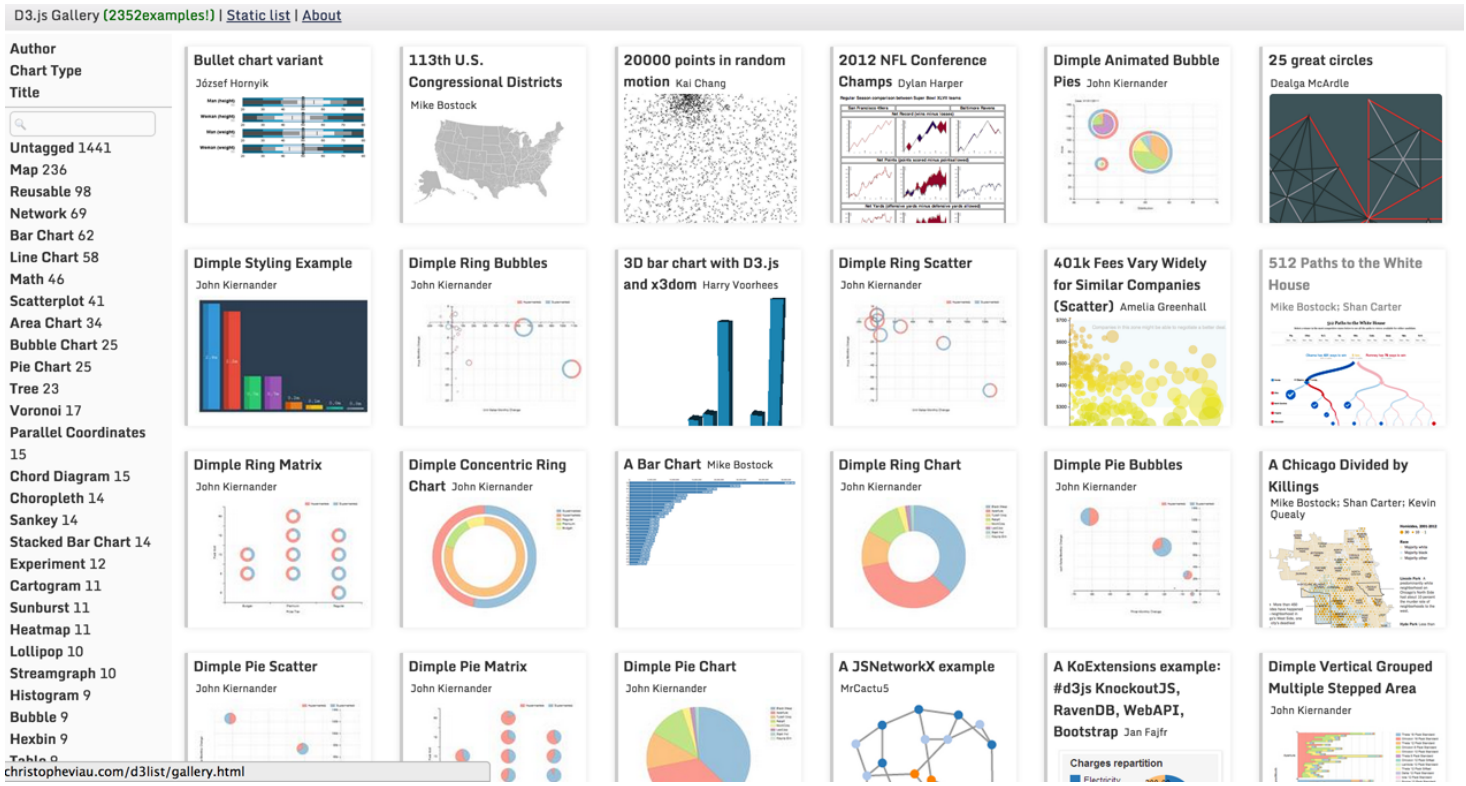
1000's OF EXAMPLES AVAILABLE

http://christopheviau.com/d3list/gallery.html



SECTION 3
D3 EXAMPLES & bl.ocks.org

D3 EXAMPLES = shortcut!



BL.OCKS.ORG

1. Mike Bostock Scatterplot Example
   http://bl.ocks.org/mbostock/raw/3887118/
2. See the code, data files, png at GitHub:
   https://gist.github.com/mbostock/3887118/

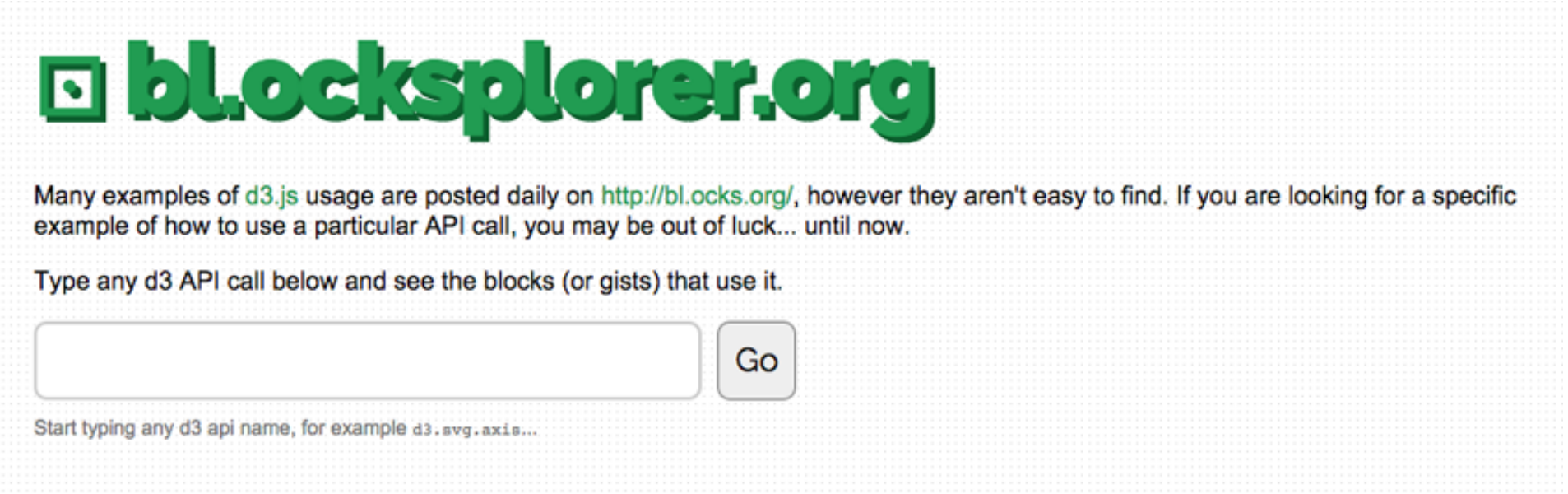
Discuss:
How to get the files
How to run the files

MIKE'S BL.OCKS.ORG BLOCK'S

1. Mike Bostock Scatterplot Example
   http://bl.ocks.org/mbostock/
2. See the code, data files, png at GitHub:
   https://gist.github.com/mbostock/

BL.OCKSPLORER.ORG

=> bl.ocksplorer.org, lets you search blocks by d3 API Call…



Source:
http://bl.ocksplorer.org/

EXERCISE: BL.OCKS.ORG SCATTERPLOT

Go to https://gist.github.com/
Name file index.html
Paste base.html code into it from http://bit.ly/20150512-metis-d3-files
Click add file button
Name file aquarium.json
Paste aquarium.json data into it from http://bit.ly/20150512-metis-d3-files
Click on "Create secret Gist" (yellow) button
URL will be => https://gist.github.com/anonymous/[XXXXXXXXXXXXXXXXXXXX]
Go to http://bl.ocks.org/
paste the "anonymous/[XXXXXXXXXXXXXXXXXXXX]" part after the "/" of the URL
Click on "Open in a new window." link underneath empty rectangle
In developer window:

```
d3.json("aquarium.json", function(error, data) {
  console.log(data);
});
```

Discuss:
Why go to the trouble?

SECTION 4
D3 LAYOUTS

DATA VISUALIZATION EXPERTS

2005: Prefuse (Java, Heer @ Berkeley)

2007: Flare (ActionScript, Heer @ Berkeley)

2009: Protovis (JavaScript, Heer & Bostock @ Stanford)

2011: D3 (JavaScript, Heer & Bostock @ Stanford)

TLDR:
very smart people thought very hard
about data visualization for a very long time

Source:
http://anna.ps/talks/fel/#/4

D3 LAYOUTS

Layouts

Wiki ▸ API Reference ▸ Layouts

See one of:

- Bundle - apply Holten's *hierarchical bundling algorithm* to edges.
- Chord - produce a chord diagram from a matrix of relationships.
- Cluster - cluster entities into a dendrogram.
- Force - position linked nodes using physical simulation.
- Hierarchy - derive a custom hierarchical layout implementation.
- Histogram - compute the distribution of data using quantized bins.
- Pack - produce a hierarchical layout using recursive circle-packing.
- Partition - recursively partition a node tree into a sunburst or icicle.
- Pie - compute the start and end angles for arcs in a pie or donut chart.
- Stack - compute the baseline for each series in a stacked bar or area chart.
- Tree - position a tree of nodes tidily.
- Treemap - use recursive spatial subdivision to display a tree of nodes.

LAYOUT: BUNDLE

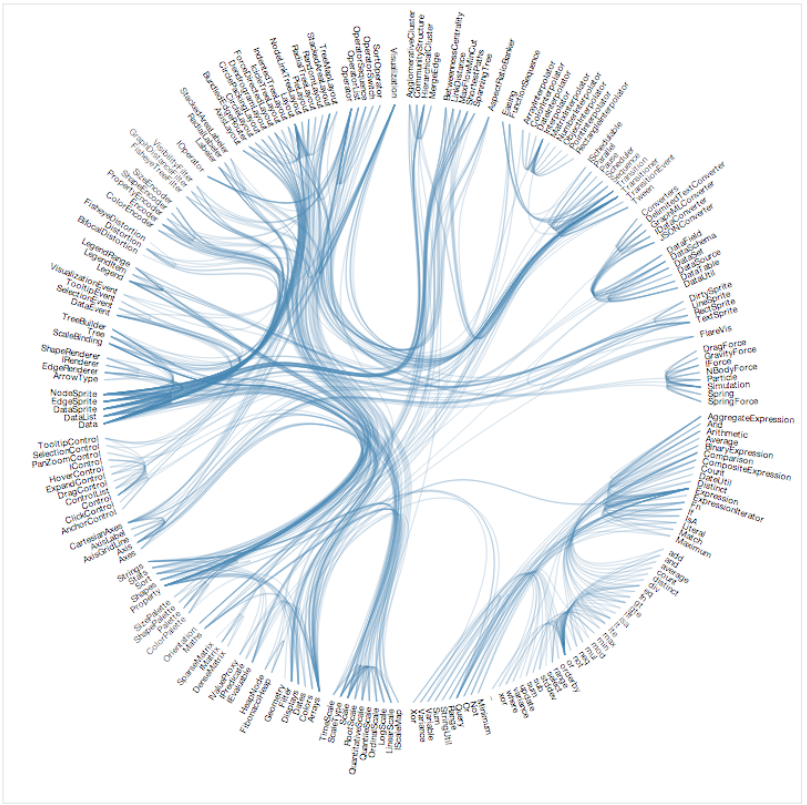
Layouts

Wiki ▸ API Reference ▸ Layouts

See one of:

- Bundle - apply Holten's *hierarchical bundling algorithm* to edges.
- Chord - produce a chord diagram from a matrix of relationships.
- Cluster - cluster entities into a dendrogram.
- Force - position linked nodes using physical simulation.
- Hierarchy - derive a custom hierarchical layout implementation.
- Histogram - compute the distribution of data using quantized bins.
- Pack - produce a hierarchical layout using recursive circle-packing.
- Partition - recursively partition a node tree into a sunburst or icicle.
- Pie - compute the start and end angles for arcs in a pie or donut chart.
- Stack - compute the baseline for each series in a stacked bar or area chart.
- Tree - position a tree of nodes tidily.
- Treemap - use recursive spatial subdivision to display a tree of nodes.

Hierarchical Edge Bundling



Source:
http://bl.ocks.org/mbostock/1044242

LAYOUT: CHORD

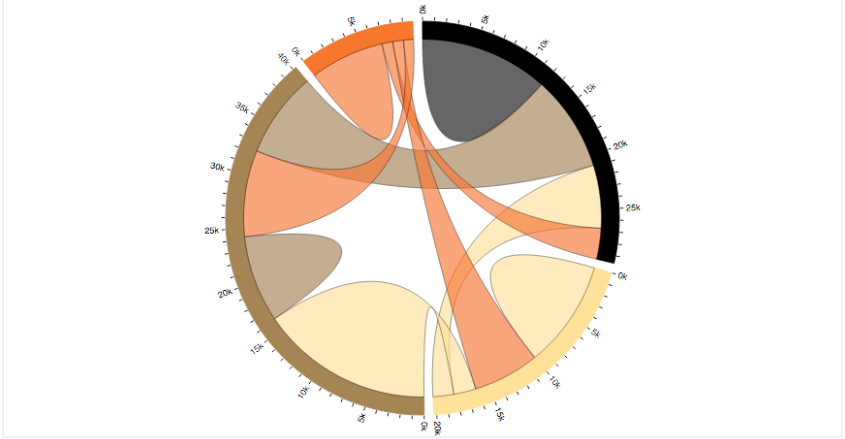
Layouts

Wiki ▸ API Reference ▸ Layouts

See one of:

- Bundle - apply Holten's *hierarchical bundling algorithm* to edges.
- Chord - produce a chord diagram from a matrix of relationships.
- Cluster - cluster entities into a dendrogram.
- Force - position linked nodes using physical simulation.
- Hierarchy - derive a custom hierarchical layout implementation.
- Histogram - compute the distribution of data using quantized bins.
- Pack - produce a hierarchical layout using recursive circle-packing.
- Partition - recursively partition a node tree into a sunburst or icicle.
- Pie - compute the start and end angles for arcs in a pie or donut chart.
- Stack - compute the baseline for each series in a stacked bar or area chart.
- Tree - position a tree of nodes tidily.
- Treemap - use recursive spatial subdivision to display a tree of nodes.

Chord Diagram



Source:
http://bl.ocks.org/mbostock/4062006

LAYOUT: CLUSTER

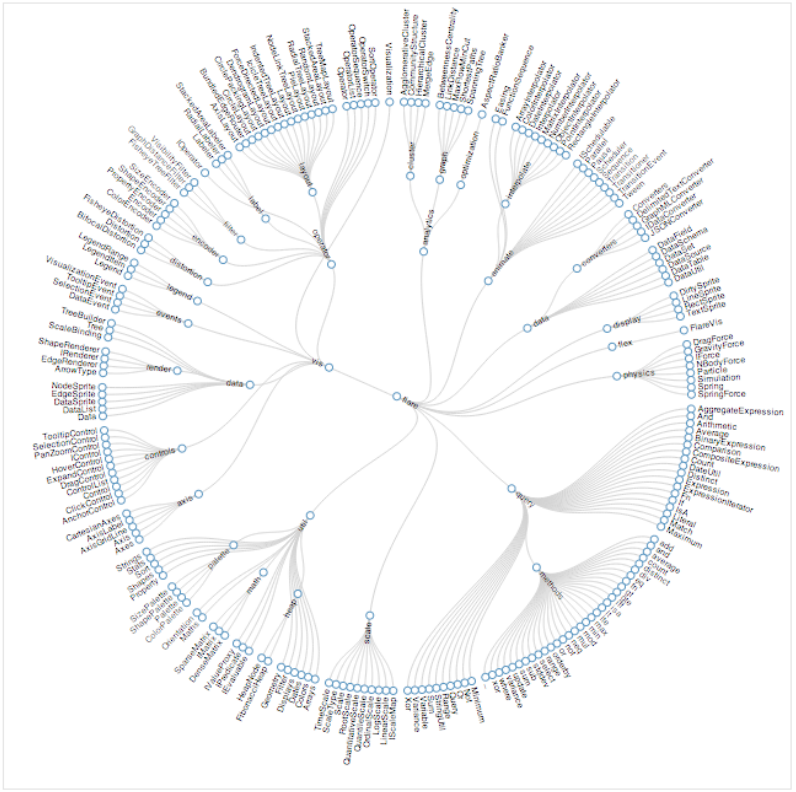
Layouts

Wiki ▸ API Reference ▸ Layouts

See one of:

- Bundle - apply Holten's *hierarchical bundling algorithm* to edges.
- Chord - produce a chord diagram from a matrix of relationships.
- Cluster - cluster entities into a dendrogram.
- Force - position linked nodes using physical simulation.
- Hierarchy - derive a custom hierarchical layout implementation.
- Histogram - compute the distribution of data using quantized bins.
- Pack - produce a hierarchical layout using recursive circle-packing.
- Partition - recursively partition a node tree into a sunburst or icicle.
- Pie - compute the start and end angles for arcs in a pie or donut chart.
- Stack - compute the baseline for each series in a stacked bar or area chart.
- Tree - position a tree of nodes tidily.
- Treemap - use recursive spatial subdivision to display a tree of nodes.

Cluster Dendrogram



Source:
http://bl.ocks.org/mbostock/4339607

LAYOUT: FORCE

Layouts

Wiki ▸ API Reference ▸ Layouts

See one of:

- Bundle - apply Holten's *hierarchical bundling algorithm* to edges.
- Chord - produce a chord diagram from a matrix of relationships.
- Cluster - cluster entities into a dendrogram.
- Force - position linked nodes using physical simulation.
- Hierarchy - derive a custom hierarchical layout implementation.
- Histogram - compute the distribution of data using quantized bins.
- Pack - produce a hierarchical layout using recursive circle-packing.
- Partition - recursively partition a node tree into a sunburst or icicle.
- Pie - compute the start and end angles for arcs in a pie or donut chart.
- Stack - compute the baseline for each series in a stacked bar or area chart.
- Tree - position a tree of nodes tidily.
- Treemap - use recursive spatial subdivision to display a tree of nodes.

Force-Directed Graph

Source:
http://bl.ocks.org/mbostock/4062045

LAYOUT: HIERARCHY

Layouts

Wiki ▸ API Reference ▸ Layouts

See one of:

- Bundle - apply Holten's *hierarchical bundling algorithm* to edges.
- Chord - produce a chord diagram from a matrix of relationships.
- Cluster - cluster entities into a dendrogram.
- Force - position linked nodes using physical simulation.
- Hierarchy - derive a custom hierarchical layout implementation.
- Histogram - compute the distribution of data using quantized bins.
- Pack - produce a hierarchical layout using recursive circle-packing.
- Partition - recursively partition a node tree into a sunburst or icicle.
- Pie - compute the start and end angles for arcs in a pie or donut chart.
- Stack - compute the baseline for each series in a stacked bar or area chart.
- Tree - position a tree of nodes tidily.
- Treemap - use recursive spatial subdivision to display a tree of nodes.

- Cluster
- Pack
- Partition
- Tree
- Treemap

Source:
https://github.com/mbostock/d3/wiki/Hierarchy-Layout

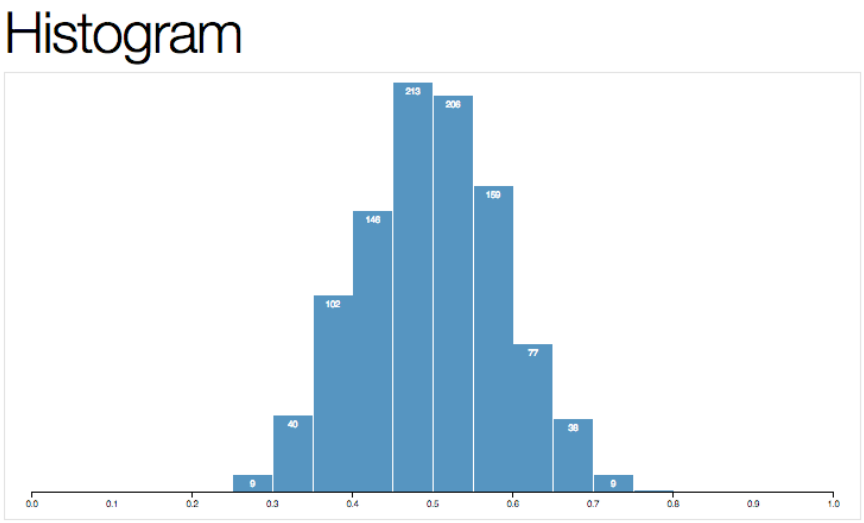
LAYOUT: HISTOGRAM

Layouts

Wiki ▸ API Reference ▸ Layouts

See one of:

- Bundle - apply Holten's *hierarchical bundling algorithm* to edges.
- Chord - produce a chord diagram from a matrix of relationships.
- Cluster - cluster entities into a dendrogram.
- Force - position linked nodes using physical simulation.
- Hierarchy - derive a custom hierarchical layout implementation.
- Histogram - compute the distribution of data using quantized bins.
- Pack - produce a hierarchical layout using recursive circle-packing.
- Partition - recursively partition a node tree into a sunburst or icicle.
- Pie - compute the start and end angles for arcs in a pie or donut chart.
- Stack - compute the baseline for each series in a stacked bar or area chart.
- Tree - position a tree of nodes tidily.
- Treemap - use recursive spatial subdivision to display a tree of nodes.



Source:
http://bl.ocks.org/mbostock/3048450

LAYOUT: PACK

Layouts

Wiki ▸ API Reference ▸ Layouts

See one of:

- Bundle - apply Holten's *hierarchical bundling algorithm* to edges.
- Chord - produce a chord diagram from a matrix of relationships.
- Cluster - cluster entities into a dendrogram.
- Force - position linked nodes using physical simulation.
- Hierarchy - derive a custom hierarchical layout implementation.
- Histogram - compute the distribution of data using quantized bins.
- **Pack** - produce a hierarchical layout using recursive circle-packing.
- Partition - recursively partition a node tree into a sunburst or icicle.
- Pie - compute the start and end angles for arcs in a pie or donut chart.
- Stack - compute the baseline for each series in a stacked bar or area chart.
- Tree - position a tree of nodes tidily.
- Treemap - use recursive spatial subdivision to display a tree of nodes.

Circle Packing

Source:
http://bl.ocks.org/mbostock/4063530

LAYOUT: PARTITION

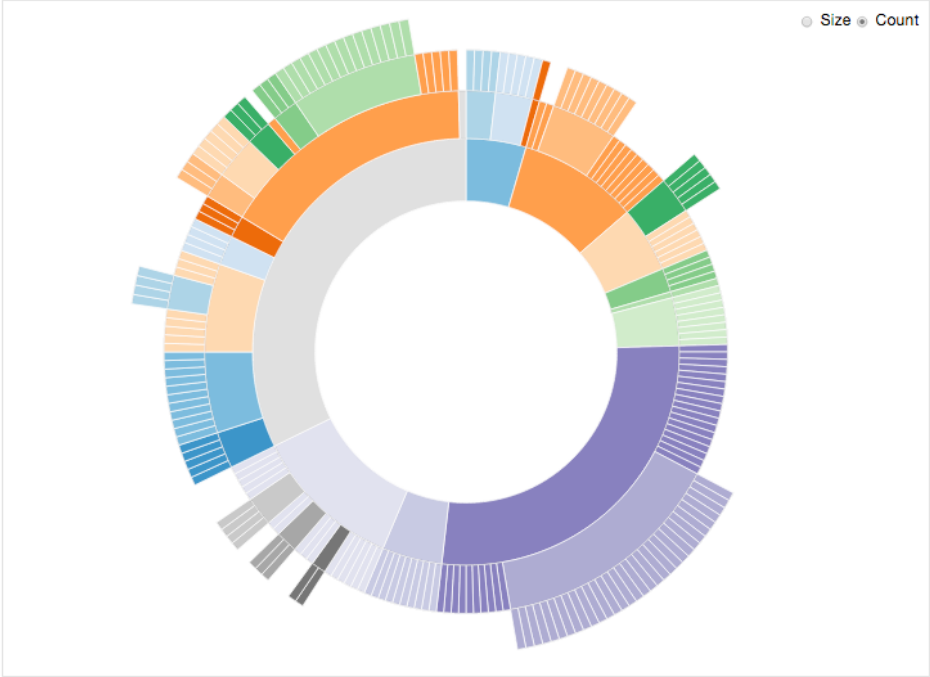
Layouts

Wiki ▸ API Reference ▸ Layouts

See one of:

- Bundle - apply Holten's *hierarchical bundling algorithm* to edges.
- Chord - produce a chord diagram from a matrix of relationships.
- Cluster - cluster entities into a dendrogram.
- Force - position linked nodes using physical simulation.
- Hierarchy - derive a custom hierarchical layout implementation.
- Histogram - compute the distribution of data using quantized bins.
- Pack - produce a hierarchical layout using recursive circle-packing.
- Partition - recursively partition a node tree into a sunburst or icicle.
- Pie - compute the start and end angles for arcs in a pie or donut chart.
- Stack - compute the baseline for each series in a stacked bar or area chart.
- Tree - position a tree of nodes tidily.
- Treemap - use recursive spatial subdivision to display a tree of nodes.

Sunburst Partition



Source:
http://bl.ocks.org/mbostock/4063423

LAYOUT: PIE

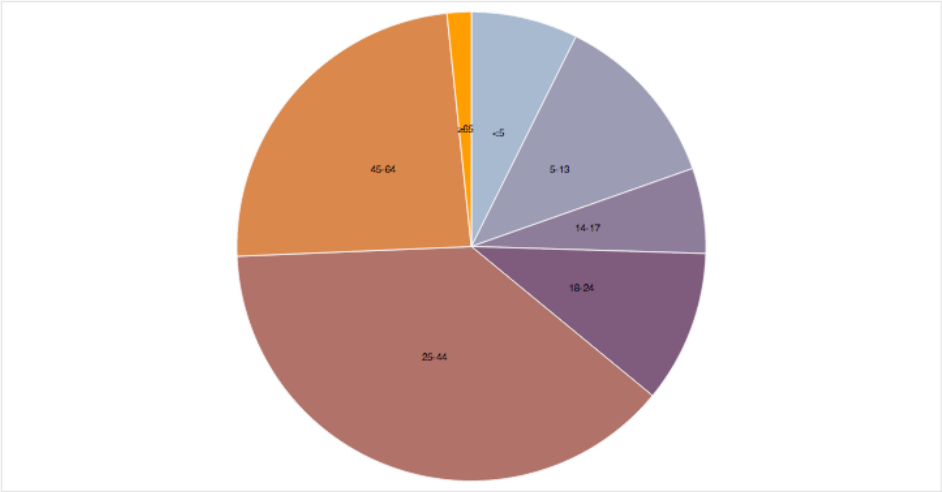
Layouts

Wiki ▸ API Reference ▸ Layouts

See one of:

- Bundle - apply Holten's *hierarchical bundling algorithm* to edges.
- Chord - produce a chord diagram from a matrix of relationships.
- Cluster - cluster entities into a dendrogram.
- Force - position linked nodes using physical simulation.
- Hierarchy - derive a custom hierarchical layout implementation.
- Histogram - compute the distribution of data using quantized bins.
- Pack - produce a hierarchical layout using recursive circle-packing.
- Partition - recursively partition a node tree into a sunburst or icicle.
- **Pie - compute the start and end angles for arcs in a pie or donut chart.**
- Stack - compute the baseline for each series in a stacked bar or area chart.
- Tree - position a tree of nodes tidily.
- Treemap - use recursive spatial subdivision to display a tree of nodes.

Pie Chart



Source:
http://bl.ocks.org/mbostock/3887235

LAYOUT: STACK

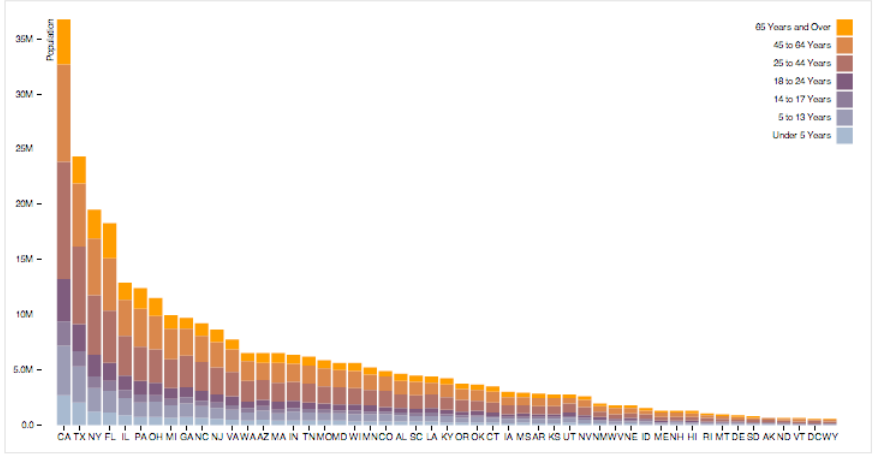
Layouts

Wiki ▸ API Reference ▸ Layouts

See one of:

- Bundle - apply Holten's *hierarchical bundling algorithm* to edges.
- Chord - produce a chord diagram from a matrix of relationships.
- Cluster - cluster entities into a dendrogram.
- Force - position linked nodes using physical simulation.
- Hierarchy - derive a custom hierarchical layout implementation.
- Histogram - compute the distribution of data using quantized bins.
- Pack - produce a hierarchical layout using recursive circle-packing.
- Partition - recursively partition a node tree into a sunburst or icicle.
- Pie - compute the start and end angles for arcs in a pie or donut chart.
- Stack - compute the baseline for each series in a stacked bar or area chart.
- Tree - position a tree of nodes tidily.
- Treemap - use recursive spatial subdivision to display a tree of nodes.

Stacked Bar Chart



Source:
http://bl.ocks.org/mbostock/3886208

LAYOUT: TREE

Layouts

Wiki ▸ API Reference ▸ Layouts

See one of:

- Bundle - apply Holten's *hierarchical bundling algorithm* to edges.
- Chord - produce a chord diagram from a matrix of relationships.
- Cluster - cluster entities into a dendrogram.
- Force - position linked nodes using physical simulation.
- Hierarchy - derive a custom hierarchical layout implementation.
- Histogram - compute the distribution of data using quantized bins.
- Pack - produce a hierarchical layout using recursive circle-packing.
- Partition - recursively partition a node tree into a sunburst or icicle.
- Pie - compute the start and end angles for arcs in a pie or donut chart.
- Stack - compute the baseline for each series in a stacked bar or area chart.
- Tree - position a tree of nodes tidily.
- Treemap - use recursive spatial subdivision to display a tree of nodes.

Radial Reingold–Tilford Tree

Source:
http://bl.ocks.org/mbostock/4063550

LAYOUT: TREEMAP

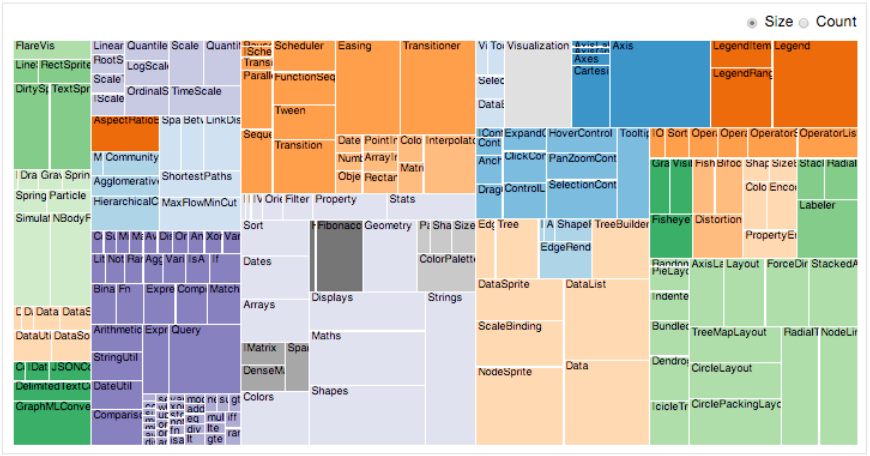
Layouts

Wiki ▸ API Reference ▸ Layouts

See one of:

- Bundle - apply Holten's *hierarchical bundling algorithm* to edges.
- Chord - produce a chord diagram from a matrix of relationships.
- Cluster - cluster entities into a dendrogram.
- Force - position linked nodes using physical simulation.
- Hierarchy - derive a custom hierarchical layout implementation.
- Histogram - compute the distribution of data using quantized bins.
- Pack - produce a hierarchical layout using recursive circle-packing.
- Partition - recursively partition a node tree into a sunburst or icicle.
- Pie - compute the start and end angles for arcs in a pie or donut chart.
- Stack - compute the baseline for the series in a stacked bar or area chart.
- Tree - position a tree of nodes tidily.
- Treemap - use recursive spatial subdivision to display a tree of nodes.

Treemap



Source:
http://bl.ocks.org/mbostock/4063582

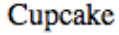
D3 PLUGINS

box	Add missing semicolons.
bullet	Link to vertical demo.
chernoff	Remove redundant svg: prefixes.
cubehelix	Restore example.
fisheye	Remove a few unused variables.
force_labels	Add missing semicolons.
geo	Slightly prettier link.
geodesic	Fix bug with flipped triangles.
geom	Update geom/contour/README.md
graph	removed traverse
hexbin	Include hexbin grid coordinates.
hive	Move to top-level directory.
horizon	Remove horizon.duration.
interpolate-zoom	Add deprecation notice.
jsonp	Fix token replacement in jsonp plugin
keybinding	Update keybinding with improvements from iD project
longscroll	Shorten.
qq	Add qq plugin.
rollup	Create README.md
sankey	Add demo links.
simplify	Remove obsolete d3.simplify plugin.
superformula	Add superformula plugin.
urlencode	Add missing semicolons.
LICENSE	Update copyright year.
README.md	Add an example (d3.svg.hive).

D3 & SVG BASIC SHAPES

Shape	SVG Construction	D3 Construction
	<svg width="100" height="100"> <circle cx="35" cy="35" r="25" /> </svg>	d3.select("svg").append("circle") .attr("cx", "35").attr("cy", "35").attr("r", "25");
	<svg width="100" height="100"> <rect x="10" y="10" width="50" height="50" /> </svg>	d3.select("svg").append("rect") .attr("x", "10").attr("y", "10").attr("height", "50").attr("width", "50");
	<svg width="100" height="100"> <ellipse cx="25" cy="25" rx="15" ry="10" /> </svg>	d3.select("svg").append("ellipse") .attr("cx", "35").attr("cy", "35").attr("rx", "15").attr("ry", "25");

	<svg width="50" height="50">
	  <line x1="10" y1="10" x2="50" y2="50"
	  stroke-width="3" stroke="black" />
	</svg>
		d3.select("svg").append("line")
		  .attr("x1", 10).attr("y1", 10).attr("x2", 50).attr("y2", 50)
		  .attr("stroke-width", 3).attr("stroke", "black");

	<svg width="50" height="50"> <text x="25" y="25">Cupcake</text> </svg>	d3.select("svg").append("text") .attr("x","25").attr("y","25").text("Cupcake");

	<svg width="50" height="50">
	  <path d="M10,10L50,50" stroke-width="3" stroke="black">
	  </path>
	</svg>
		d3.select("svg").append("path")
		  .attr("d", "M10,10L50,50")
		  .attr("stroke-width", 3).attr("stroke", "black");

SVG PATH MINI-LANGUAGE

Command	Parameters	Repeatable	Explanation
Pen Command			
M (m)	x, y	Yes	**moveto** Move the pen to a new location. No line is drawn. All path data must begin with a 'moveto' command.
Line Commands			
L (l)	x, y	Yes	**lineto** Draw a line from the current point to the point (x,y).
H (h)	x	Yes	**horizontal lineto** Draw a horizontal line from the current point to x.
V (v)	y	Yes	**vertical lineto** Draw a horizontal line from the current point to y.
Cubic Bezier Curve Commands			
C (c)	x1 y1 x2 y2 x y	Yes	**curveto** Draw a cubic Bézier curve from the current point to the point (x,y) using (x1,y1) as the control point at the beginning of the curve and (x2,y2) as the control point at the end of the curve.
S (s)	x2 y2 x y	Yes	**shorthand/smooth curveto** Draw a cubic Bézier curve from the current point to (x,y). The first control point is assumed to be the reflection of the last control point on the previous command relative to the current point. (x2,y2) is the second control point (i.e., the control point at the end of the curve).

Command	Parameters	Repeatable	Explanation
Quadratic Bezier Curve Commands			
Q (q)	x1 y1 x y	Yes	**quadratic Bézier curveto** Draw a quadratic Bézier curve from the current point to (x,y) using (x1,y1) as the control point.
T (t)	x y	Yes	**Shorthand/smooth quadratic Bézier curveto** Draw a quadratic Bézier curve from the current point to (x,y). The control point is assumed to be the reflection of the control point on the previous command relative to the current point.
Elliptical Arc Curve Command			
A (a)	rx ry x-axis-rotation large-arc-flag sweep-flag x y	Yes	**elliptical arc** Draws an elliptical arc from the current point to (x, y). The size and orientation of the ellipse are defined by two radii (rx, ry) and an x-axis-rotation, which indicate how the ellipse as a whole is rotated relative to the current SVG coordinate system. The center (cx, cy) of the ellipse is calculated automatically to satisfy the constraints imposed by the other parameters. large-arc-flag and sweep-flag contribute to the automatic calculations and help determine how the arc is drawn.
End Path Command			
Z (z)	none	No	**closepath** Closes the path. A line is drawn from the last point to the first point drawn.

SET OF POINTS -> GENERATE A PATH

Shape	SVG Construction	D3 Construction
	<svg width="50" height="50"> <path d="M10,10L50,50" stroke-width="3" stroke="black"> </path> </svg>	d3.select("svg").append("path") .attr("d", "M10,10L50,50") .attr("stroke-width", 3).attr("stroke", "black");

```
var lineData = [ { "x": 10,   "y": 10},  { "x": 50,  "y": 50}];

// "M10,10L50,50"
```

D3 SVG LINE GENERATOR

```
var lineFunction = d3.svg.line()
        .x(function(d) { return d.x; })
        .y(function(d) { return d.y; })
        .interpolate("linear");

var lineData = [ { "x": 10,   "y": 10},  { "x": 50,  "y": 50}];

lineFunction(lineData);
// returns "M10,10L50,50"
```

ACCESSOR FUNCTIONS

D3 SVG PATH DATA GENERATOR OPTIONS

- linear - piecewise linear segments, as in a polyline.
- linear-closed - close the linear segments to form a polygon.
- step - alternate between horizontal and vertical segments, as in a step function.
- step-before - alternate between vertical and horizontal segments, as in a step function.
- step-after - alternate between horizontal and vertical segments, as in a step function.
- basis - a B-spline, with control point duplication on the ends.
- basis-open - an open B-spline; may not intersect the start or end.
- basis-closed - a closed B-spline, as in a loop.
- bundle - equivalent to *basis*, except the *tension* parameter is used to straighten the spline.
- cardinal - a Cardinal spline, with control point duplication on the ends.
- cardinal-open - an open Cardinal spline; may not intersect the start or end, but will intersect other control points.
- cardinal-closed - a closed Cardinal spline, as in a loop.
- monotone - cubic interpolation that preserves monotonicity in *y*.

Source:
https://github.com/mbostock/d3/wiki/SVG-Shapes#wiki-line

USING D3 SVG LINE PATH GENERATOR

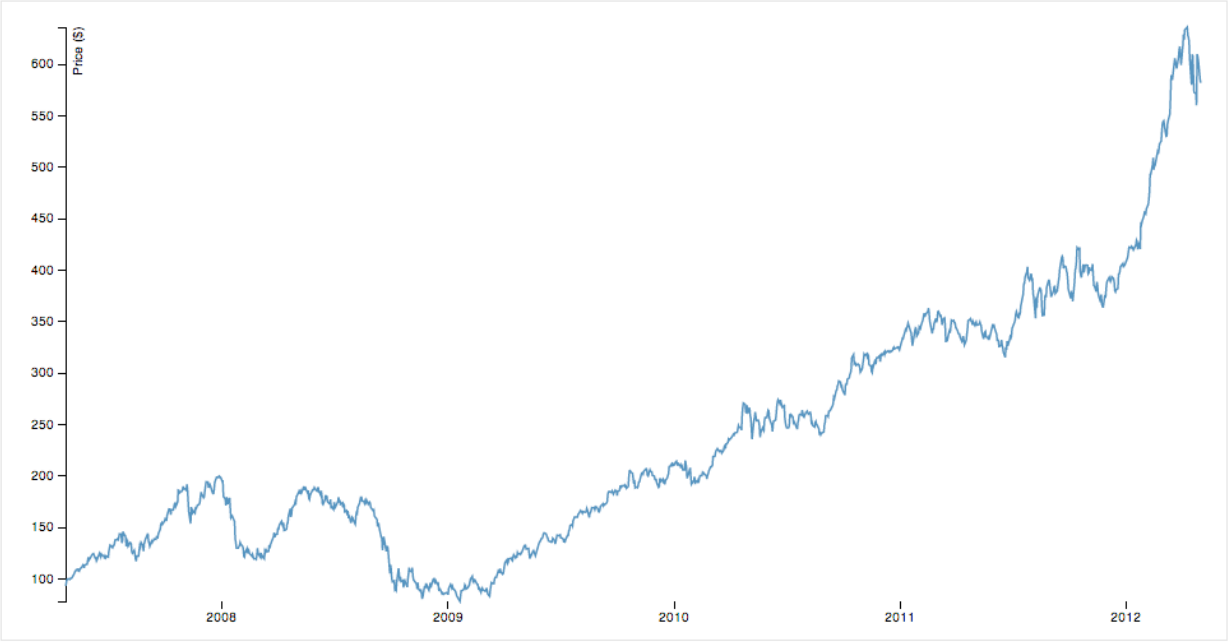
Shape	SVG Construction	D3 Construction
	<svg width="50" height="50"> <path d="M10,10L50,50" stroke-width="3" stroke="black"> </path> </svg>	d3.select("svg").append("path") .attr("d", "M10,10L50,50") .attr("stroke-width", 3).attr("stroke", "black");

```
var lineFunction = d3.svg.line()
      .x(function(d) { return d.x; })
      .y(function(d) { return d.y; })
      .interpolate("linear");

var lineData = [ { "x": 10,   "y": 10},  { "x": 50,  "y": 50}];

lineFunction(lineData);
// returns "M10,10L50,50"

d3.select("svg").append("path")
   .attr("d", lineFunction(lineData))
   .attr("stroke-width", 3).attr("stroke", "black");
```

SVG PATH DATA GENERATOR EXAMPLE

mbostock's block #3883245 October 13, 2012

Line Chart



1280
Point Objects
And Connecting
Lines
Graphed Easily

Source:
http://bl.ocks.org/mbostock/3883245

D3 SVG PATH GENERATOR – PATH TAG

```
<!DOCTYPE html>
▼<html>
  ▶<head>…</head>
  ▼<body style>
    <script src="http://d3js.org/d3.v3.js"></script>
    ▶<script>…</script>
    ▼<svg width="960" height="500">
      ▼<g transform="translate(50,20)">
        ▶<g class="x axis" transform="translate(0,450)">…</g>
        ▶<g class="y axis">…</g>
         <path class="line" d=
         "M890,43.626686737272166L889.5147219193021,42.1348314(
       </g>
     </svg>
   </body>
</html>
```

Instead of 1279 SVG <line ... /> elements
It is One SVG <path ... /> element

Source:
http://bl.ocks.org/mbostock/3883245

MAKING A PIE CHART

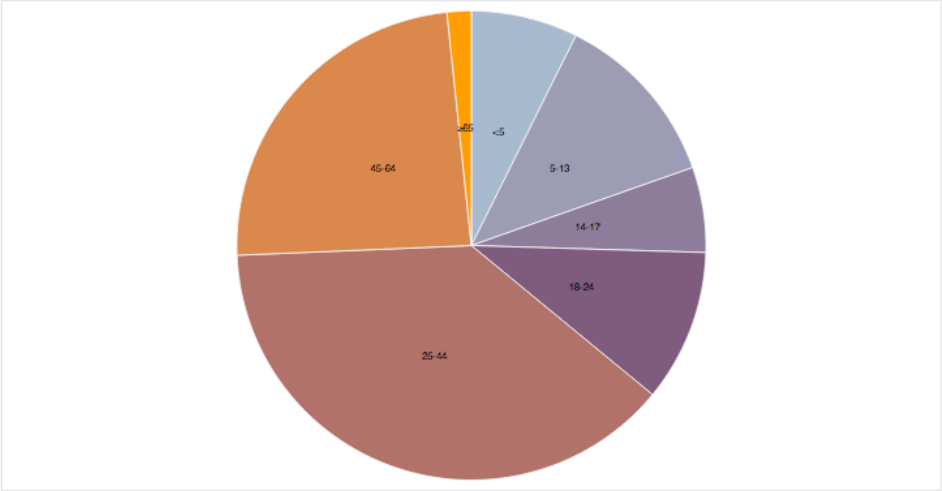
Layouts

Wiki ▸ API Reference ▸ Layouts

See one of:

- Bundle - apply Holten's *hierarchical bundling algorithm* to edges.
- Chord - produce a chord diagram from a matrix of relationships.
- Cluster - cluster entities into a dendrogram.
- Force - position linked nodes using physical simulation.
- Hierarchy - derive a custom hierarchical layout implementation.
- Histogram - compute the distribution of data using quantized bins.
- Pack - produce a hierarchical layout using recursive circle-packing.
- Partition - recursively partition a node tree into a sunburst or icicle.
- Pie - compute the start and end angles for arcs in a pie or donut chart.
- Stack - compute the baseline for each series in a stacked bar or area chart.
- Tree - position a tree of nodes tidily.
- Treemap - use recursive spatial subdivision to display a tree of nodes.

Pie Chart



Source:
http://bl.ocks.org/mbostock/3887235

D3 SVG SHAPE GENERATOR – ARC

```
var arc = d3.svg.arc()
    .innerRadius(20)
    .outerRadius(100)
    .startAngle(function(d, i) { return d.start; })
    .endAngle(function(d, i) { return d.start + d.size; });

var data = [{start: 0, size: 1.57079633}];

arc(data[0]);
// returns "M0,-100A100,100 0 0,1 100,3.2051035159241787e-7L20,6.410207031848357e-8A20,20 0 0,0 0,-20Z"
```

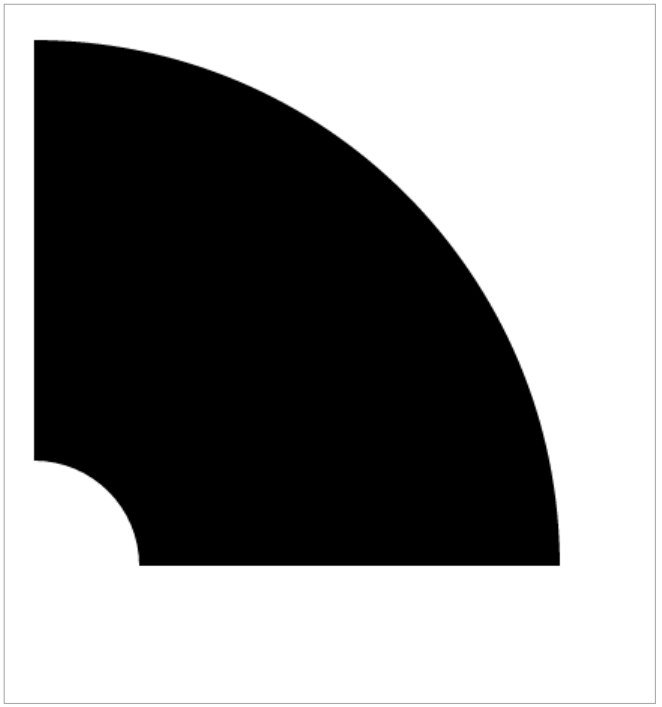
Pixels from Center of "Circle"

Radians From Top of "Circle"

Source: https://github.com/mbostock/d3/wiki/SVG-Shapes

D3 SVG ARC EXAMPLE

```
var data = [{start: 0, size: 1.57079633}];

var arc = d3.svg.arc()
    .innerRadius(20)
    .outerRadius(100)
    .startAngle(function(d, i) { return d.start; })
    .endAngle(function(d, i) { return d.start + d.size; });

var chart = d3.select("body").append("svg")
    .attr("class", "chart")
  .append("g")
    .attr("transform", "translate(0,100)");

chart.selectAll("path")
    .data(data)
  .enter().append("path")
    .attr("d", arc);
```



Discuss
Why transform, translate?

EXERCISE: SVG ARC GENERATOR – 1 ARC

```
var data = [{start: 0, size: 3.14159}];

var arc = d3.svg.arc()
    .innerRadius(80)
    .outerRadius(100)
    .startAngle(function(d, i) { return d.start; })
    .endAngle(function(d, i) { return d.start + d.size; });

var chart = d3.select("body").append("svg").attr("height", 600)
    .attr("class", "chart")
  .append("g")
    .attr("transform", "translate(50,100)");

chart.selectAll("path")
    .data(data)
  .enter().append("path")
    .attr("d", arc);
```

Discuss:
Radius
Angles, size, arc…

Use base.html

DATA-DRIVEN SVG SHAPE GENERATOR

```
var arc = d3.svg.arc()
    .innerRadius(function(d, i) { return ...; })
    .outerRadius(function(d, i) { return ...; })
    .startAngle(function(d, i) { return ...; })
    .endAngle(function(d, i) { return ...; });
```

EXERCISE: SVG ARC GENERATOR – RADII

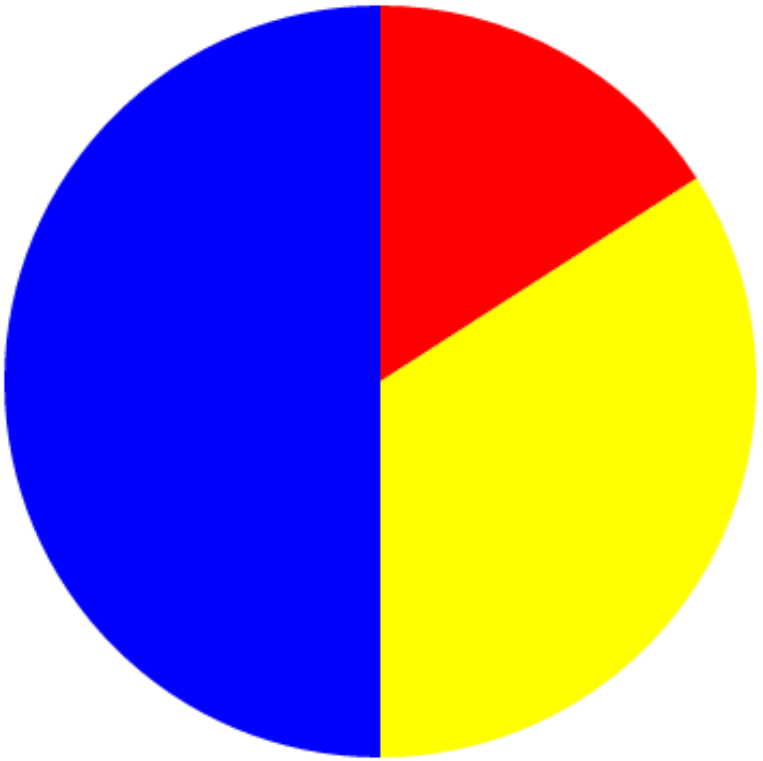
```
var data = [ {start: 0, size: 2,    inner: 50, outer: 100, color: "green"},
             {start: 2, size: 2,    inner: 25, outer: 150, color: "blue"},
             {start: 4, size: 2.28, inner: 75, outer: 125, color: "red"}];

var arc = d3.svg.arc()
   .innerRadius(function(d, i) { return d.inner; })
   .outerRadius(function(d, i) { return d.outer; })
   .startAngle(function(d, i) { return d.start; })
   .endAngle(function(d, i) { return d.start + d.size; });

var chart = d3.select("body").append("svg").attr("height", 600)
   .attr("class", "chart")
  .append("g")
   .attr("transform", "translate(150,200)");

chart.selectAll("path")
   .data(data)
  .enter().append("path")
   .style("fill", function(d, i) { return d.color; })
   .attr("d", arc);
```

Discuss:
inner
outer

Use base.html

PIE CHART IS MADE UP OF ARCS

- 3 Arcs
- 1 arc - red
- 1 arc - yellow
- 1 arc - blue
- All have same inner radius
- All have same outer radius
- 3 Arcs add up to 100% of circle
- Each arc has a different starting and ending point



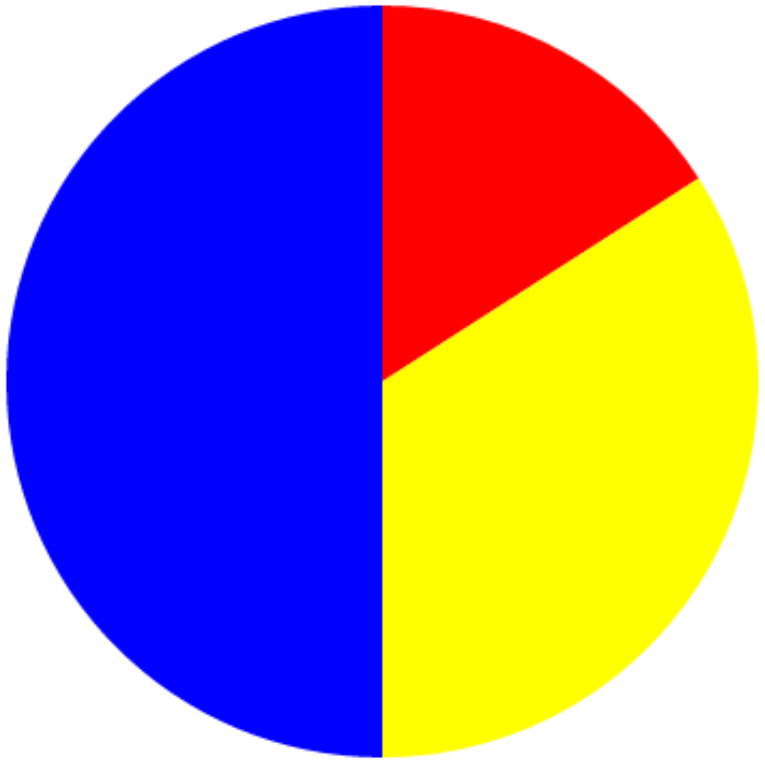
PIE CHART LAYOUT DOES MATH FOR US

Layouts

Wiki ▸ API Reference ▸ Layouts

See one of:

- Bundle - apply Holten's *hierarchical bundling algorithm* to edges.
- Chord - produce a chord diagram from a matrix of relationships.
- Cluster - cluster entities into a dendrogram.
- Force - position linked nodes using physical simulation.
- Hierarchy - derive a custom hierarchical layout implementation.
- Histogram - compute the distribution of data using quantized bins.
- Pack - produce a hierarchical layout using recursive circle-packing.
- Partition - recursively partition a node tree into a sunburst or icicle.
- Pie - compute the start and end angles for arcs in a pie or donut chart.
- Stack - compute the baseline for each series in a stacked bar or area chart.
- Tree - position a tree of nodes tidily.
- Treemap - use recursive spatial subdivision to display a tree of nodes.



PIE LAYOUT – COMPUTES ARC ANGLES FROM DATA

```
> var data = [ {amount: 1, color: "red"  },
               {amount: 2, color: "green"},
               {amount: 3, color: "blue" }];
  undefined
> var pie = d3.layout.pie()
      .sort(null)
      .value(function(d) { return d.amount; });
  undefined
> pie(data)
  [▼ Object i                          , ▼ Object i                          , ▼ Object i                          ]
     ▶ data: Object                       ▶ data: Object                       ▶ data: Object
       endAngle: 1.0471975511965976         endAngle: 3.141592653589793          endAngle: 6.283185307179586
       startAngle: 0                        startAngle: 1.0471975511965976       startAngle: 3.141592653589793
       value: 1                             value: 2                             value: 3
     ▶ __proto__: Object                  ▶ __proto__: Object                  ▶ __proto__: Object
>
```

EXERCISE: PIE LAYOUT ARC GENERATOR

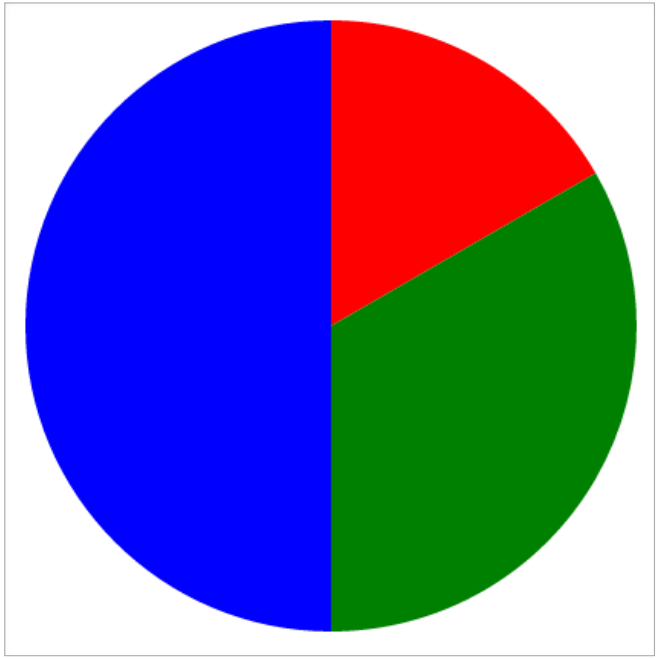
```
var data = [
    {amount: 0.75},
    {amount: 2.5},
    {amount: 3.75}
];

var pie = d3.layout.pie()
      .sort(null)
      .value(function(d) { return d.amount; });

console.table(pie(data));
```

Discuss:
sort & null
console.table

Use base.html

PIE LAYOUT + D3 SVG ARC GENERATOR

```
var data = [ {amount: 1, color: "red"  },
             {amount: 2, color: "green"},
             {amount: 3, color: "blue" }];

var arc = d3.svg.arc()
    .innerRadius(0)
    .outerRadius(100);

var pie = d3.layout.pie().sort(null)
    .value(function(d) { return d.amount; });

var chart = d3.select("body").append("svg")
    .attr("width", 400).attr("height", 400)
  .append("g")
    .attr("transform", "translate(100,100)");

var arcGs = chart.selectAll(".arc")
    .data(pie(data))
  .enter().append("g")
    .attr("class", "arc");

arcGs.append("path")
    .attr("d", arc)
    .style("fill", function(d, i) { return d.data.color; });
```



Discuss
arc vs d.data.color

EXERCISE: PIE CHART GENERATION

```
var data = [ {amount: 10, color: "red"}, {amount: 20, color: "green"}, {amount: 5, color: "blue"}];

var arc = d3.svg.arc()
    .innerRadius(0)
    .outerRadius(100);

var pie = d3.layout.pie()
    .sort(null)
    .value(function(d) { return d.amount; });

var chart = d3.select("body").append("svg").attr("height", 600).append("g")
    .attr("transform", "translate(100,100)");

var arcGs = chart.selectAll(".arc")
    .data(pie(data))
  .enter().append("g");

arcGs.append("path")
    .attr("d", arc)
    .style("fill", function(d, i) { return d.data.color; });
```

Discuss:
Radius, Angles, Size, arc
d.data.color

Use base.html

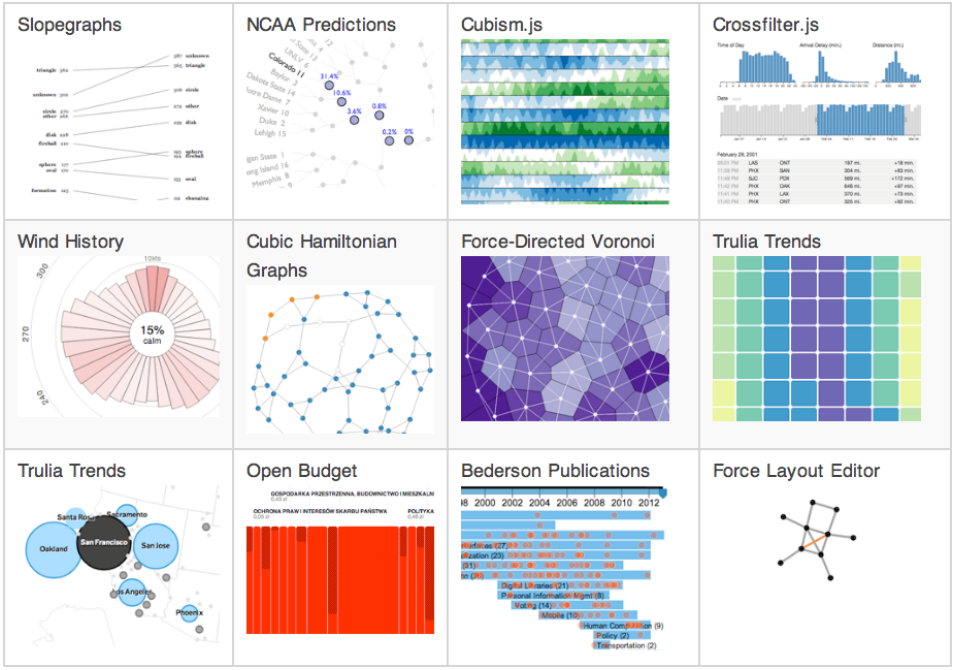
SECTION 5
CONCLUSION

DATA VISUALIZATION IS A TOOL

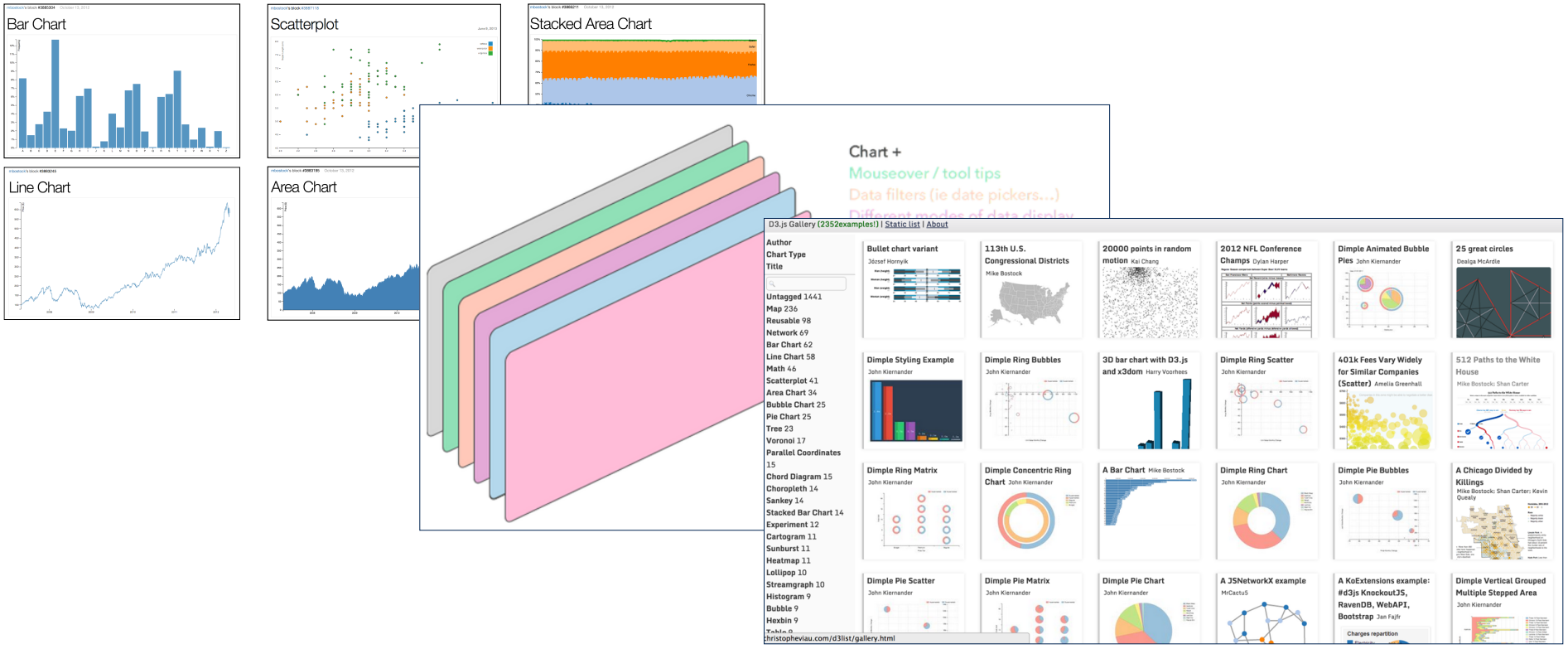
Everything Generates Data

Visualizing this data leads to understanding.

- Sports
- Commerce
- Weather
- Real Estate
- Publications
- Social Media
- Etc.....

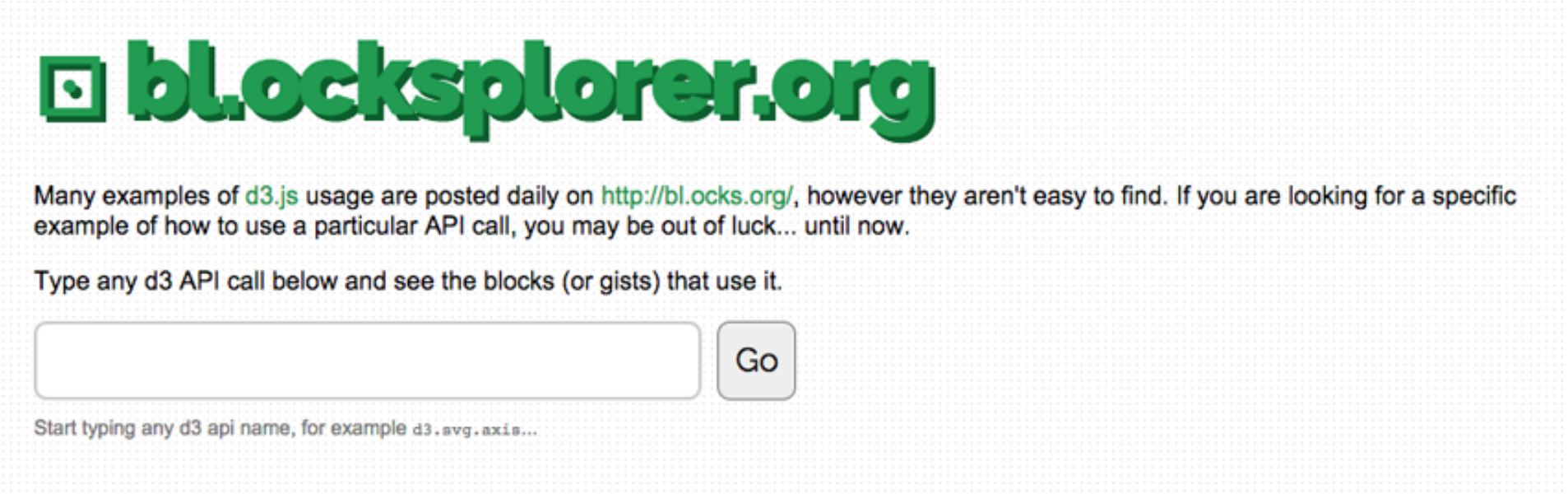


D3 IS A DATA VISUALIZATION TOOL



D3 EXAMPLES & bl.ocks.org

=> bl.ocksplorer.org, lets you search blocks by d3 API Call…



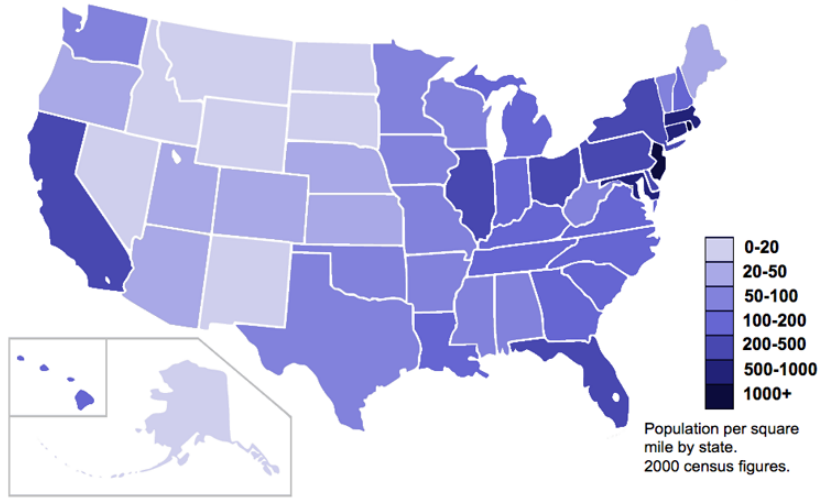
D3 LAYOUTS

box	Add missing semicolons.
bullet	Link to vertical demo.
chernoff	Remove redundant svg: prefixes.
cubehelix	Restore example.
fisheye	Remove a few unused variables.
force_labels	Add missing semicolons.
geo	Slightly prettier link.
geodesic	Fix bug with flipped triangles.
geom	Update geom/contour/README.md
graph	removed traverse
hexbin	Include hexbin grid coordinates.
hive	Move to top-level directory.
horizon	Remove horizon.duration.
interpolate-zoom	Add deprecation notice.
jsonp	Fix token replacement in jsonp plugin
keybinding	Update keybinding with improvements from iD project
longscroll	Shorten.
qq	Add qq plugin.
rollup	Create README.md
sankey	Add demo links.
simplify	Remove obsolete d3.simplify plugin.
superformula	Add superformula plugin.
urlencode	Add missing semicolons.
LICENSE	Update copyright year.
README.md	Add an example (d3.svg.hive).

DON'T MAKE YOUR AUDIENCE THINK

A39 Maryland

	A	B	C	D	E	F	G	H
20	Alaska	D-3	R-3	R-3	R-3	R-3	R-3	R-3
21	Arizona	R-5	R-5	R-6	R-6	R-6	R-7	R-7
22	Arkansas	D-6	(\1)	R-6	D-6	R-6	R-6	R-6
23	California	D-40	R-40	R-45	R-45	R-45	R-47	R-47
24	Colorado	D-6	R-6	R-7	R-7	R-7	R-8	R-8
25	Connecticut	D-8	D-8	R-8	R-8	R-8	R-8	R-8
26	Delaware	D-3	R-3	R-3	D-3	R-3	R-3	R-3
27	District of Columbia	D-3	D-3	D-3	D-3	D-3	D-3	D-3
28	Florida	D-14	R-14	R-17	D-17	R-17	R-21	R-21
29	Georgia	R-12	(\1)	R-12	D-12	D-12	R-12	R-12
30	Hawaii	D-4	D-4	R-4	D-4	D-4	R-4	D-4
31	Idaho	D-4	R-4	R-4	R-4	R-4	R-4	R-4
32	Illinois	D-26	R-26	R-26	R-26	R-26	R-24	R-24
33	Indiana	D-13	R-13	R-13	R-13	R-13	R-12	R-12
34	Iowa	D-9	R-9	R-8	R-8	R-8	R-8	D-8
35	Kansas	D-7	R-7	R-7	R-7	R-7	R-7	R-7
36	Kentucky	D-9	R-9	R-9	D-9	R-9	R-9	R-9
37	Louisiana	R-10	(\1)	R-10	D-10	R-10	R-10	R-10
38	Maine	D-4	D-4	R-4	R-4	R-4	R-4	R-4
39	Maryland	D-10	D-10	R-10	D-10	D-10	R-10	R-10
40	Massachusetts	D-14	D-14	D-14	D-14	R-14	R-13	D-13
41	Michigan	D-21	D-21	R-21	R-21	R-21	R-20	R-20
42	Minnesota	D-10	D-10	R-10	D-10	D-10	D-10	D-10
43	Mississippi	R-7	(\1)	R-7	D-7	R-7	R-7	R-7
44	Missouri	D-12	R-12	R-12	D-12	R-12	R-11	R-11
45	Montana	D-4	R-4	R-4	R-4	R-4	R-4	R-4
46	Nebraska	D-5	R-5	R-5	R-5	R-5	R-5	R-5
47	Nevada	D-3	R-3	R-3	R-3	R-3	R-4	R-4
48	New Hampshire	D-4	R-4	R-4	R-4	R-4	R-4	R-4
49	New Jersey	D-17	R-17	R-17	R-17	R-17	R-16	R-16
50	New Mexico	D-4	R-4	R-4	R-4	R-4	R-5	R-5



Source:
http://www.census.gov

Source:
http://commons.wikimedia.org/wiki/File:US_2000_census_population_density_map_by_state.svg

D3 RESOURCES

http://dashingd3js.com

http://bost.ocks.org/mike/

http://www.jasondavies.com

http://vallandingham.me/vis/

http://alignedleft.com/tutorials/d3/

http://christopheviau.com/d3list/gallery.html

https://github.com/mbostock/d3/wiki/API-Reference

http://www.jeromecukier.net/wp-content/uploads/2012/10/d3-cheat-sheet.pdf

DATA VISUALIZATION RESOURCES

http://wtfviz.net/

http://stamen.com/

http://thumbsupviz.com/

http://idl.cs.washington.edu/

https://twitter.com/DashingD3js

http://annkemery.com/dataviz-checklist/

http://www.visual-literacy.org/periodic_table/periodic_table.html

http://visualoop.com/13484/the-30-data-viz-blogs-you-cant-miss-in-one-place

QUESTIONS: ASK – HAPPY TO HELP!

CONTACT:

SEBASTIAN GUTIERREZ

SEBASTIAN@DASHINGD3JS.COM

@DASHINGD3JS