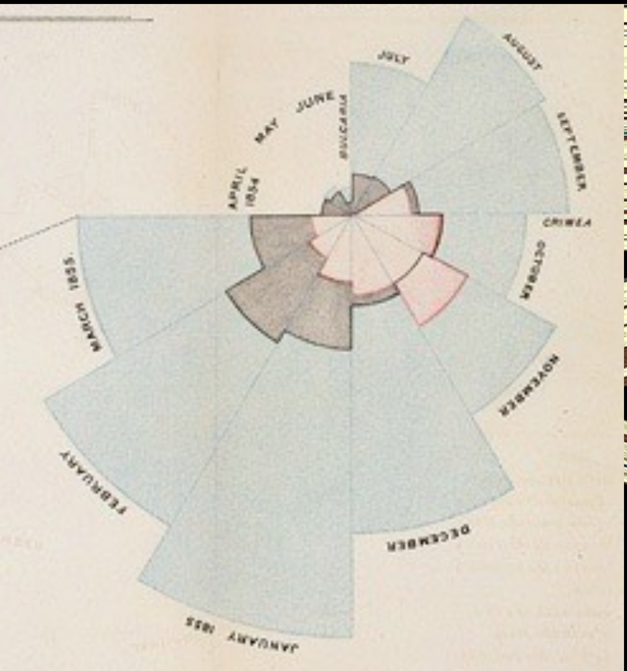
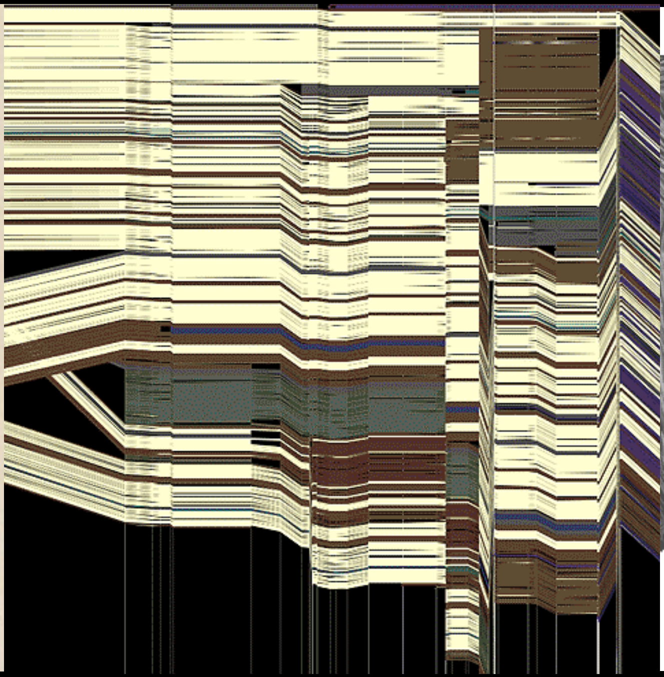
CSE 442 - Data Visualization

Visualization Tools





Jeffrey Heer University of Washington

How do people create visualizations?

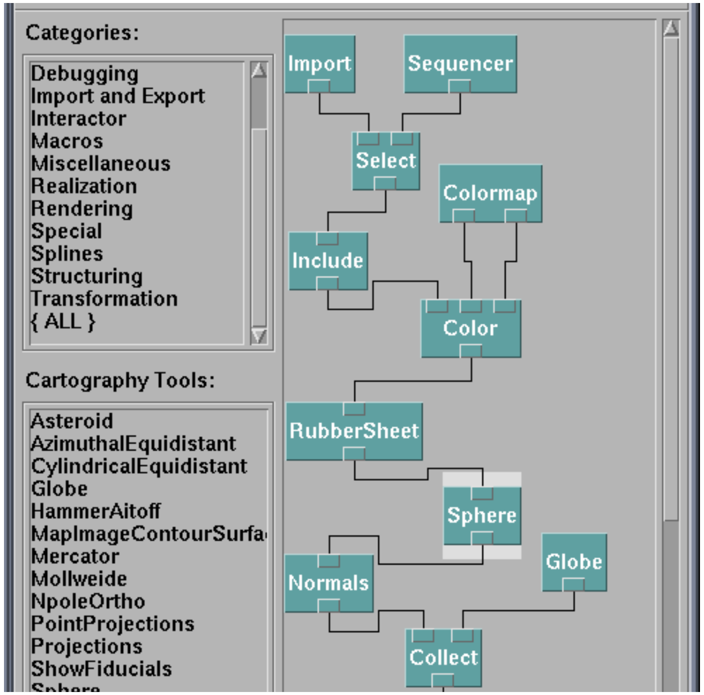


Chart Typology

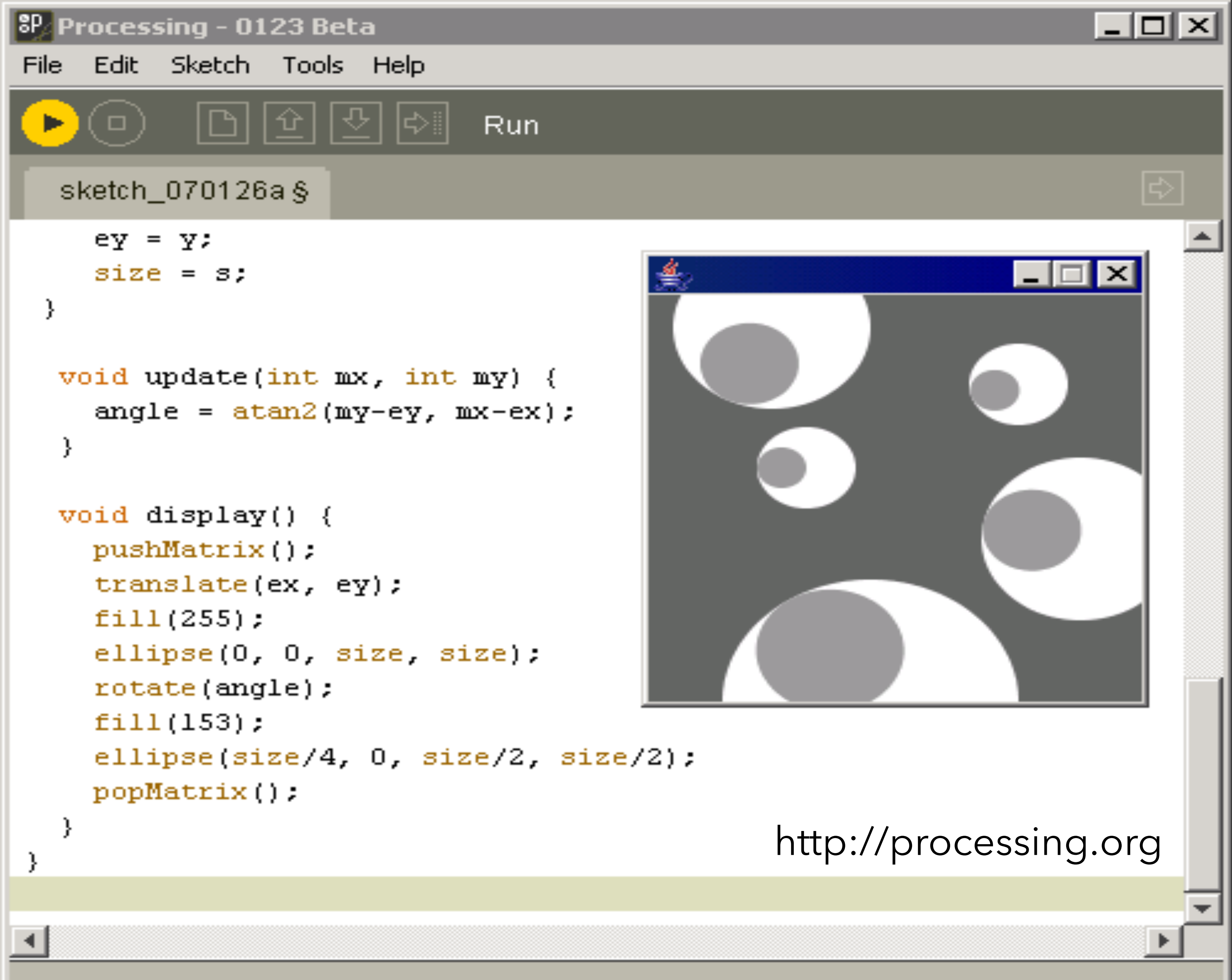
Pick from a stock of templates
Easy-to-use but limited expressiveness
Prohibits novel designs, new data types

Component Architecture

Permits more combinatorial possibilities
Novel views require new operators,
which requires software engineering

Graphics APIs

Canvas, OpenGL, Processing



```
    ey = y;
    size = s;
  }

  void update(int mx, int my) {
    angle = atan2(my-ey, mx-ex);
  }

  void display() {
    pushMatrix();
    translate(ex, ey);
    fill(255);
    ellipse(0, 0, size, size);
    rotate(angle);
    fill(153);
    ellipse(size/4, 0, size/2, size/2);
    popMatrix();
  }
}
```

http://processing.org

US Air Traffic, Aaron Koblin

Graphics APIs

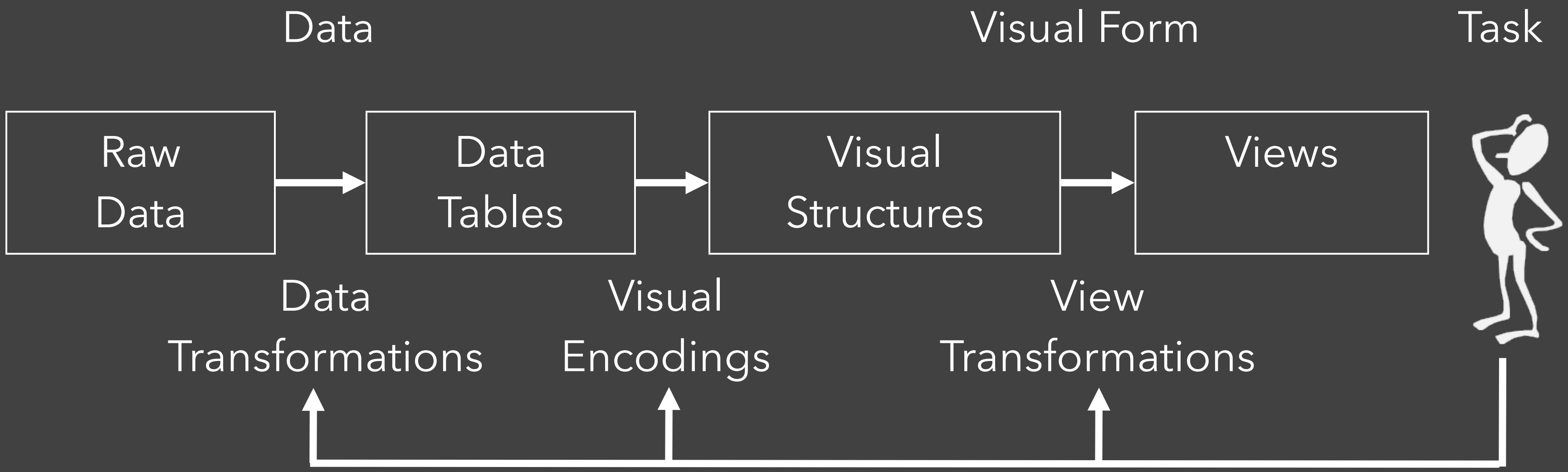
Canvas, OpenGL, Processing

Component Architectures

Prefuse, Flare, Improvise, VTK

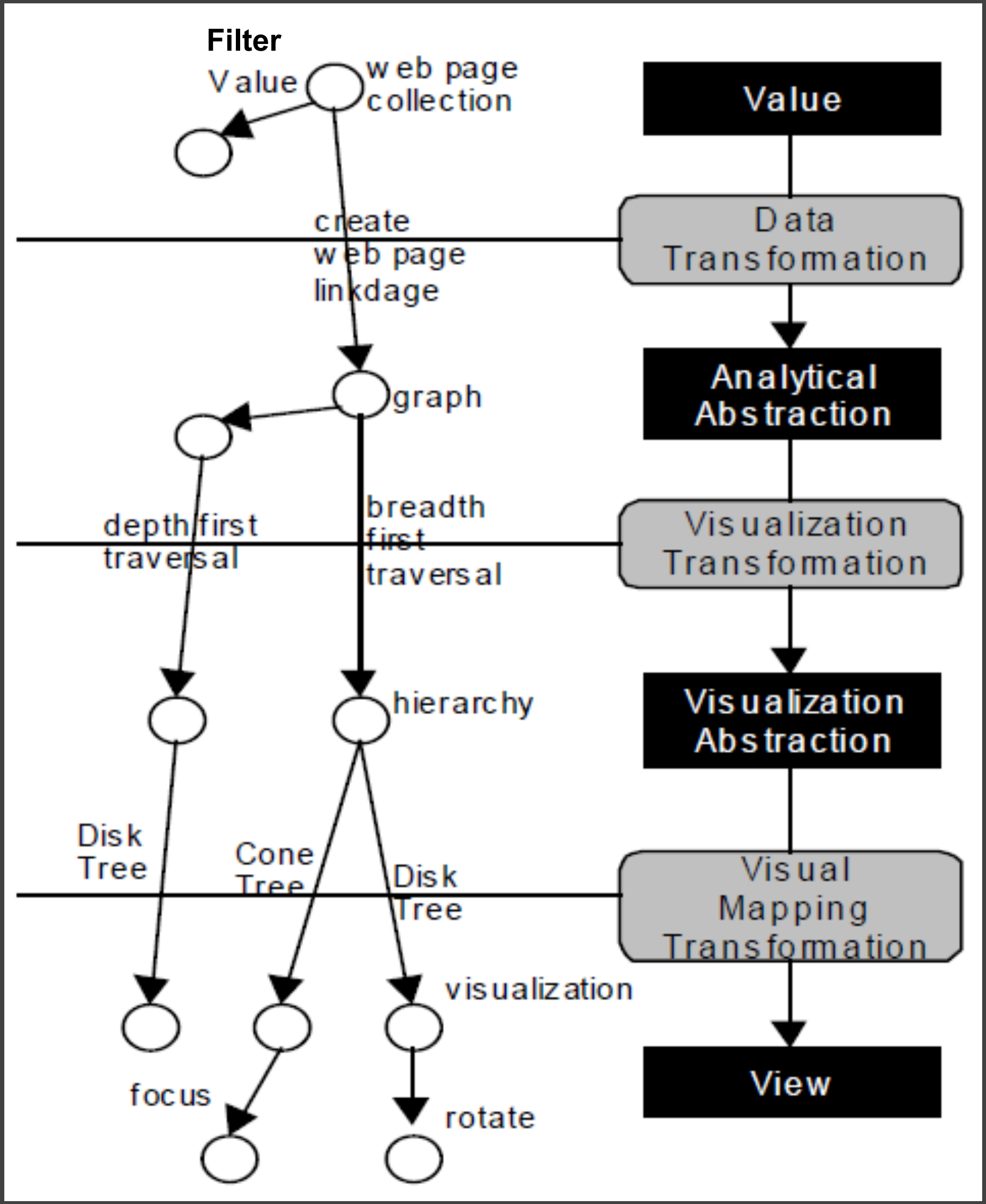
Graphics APIs

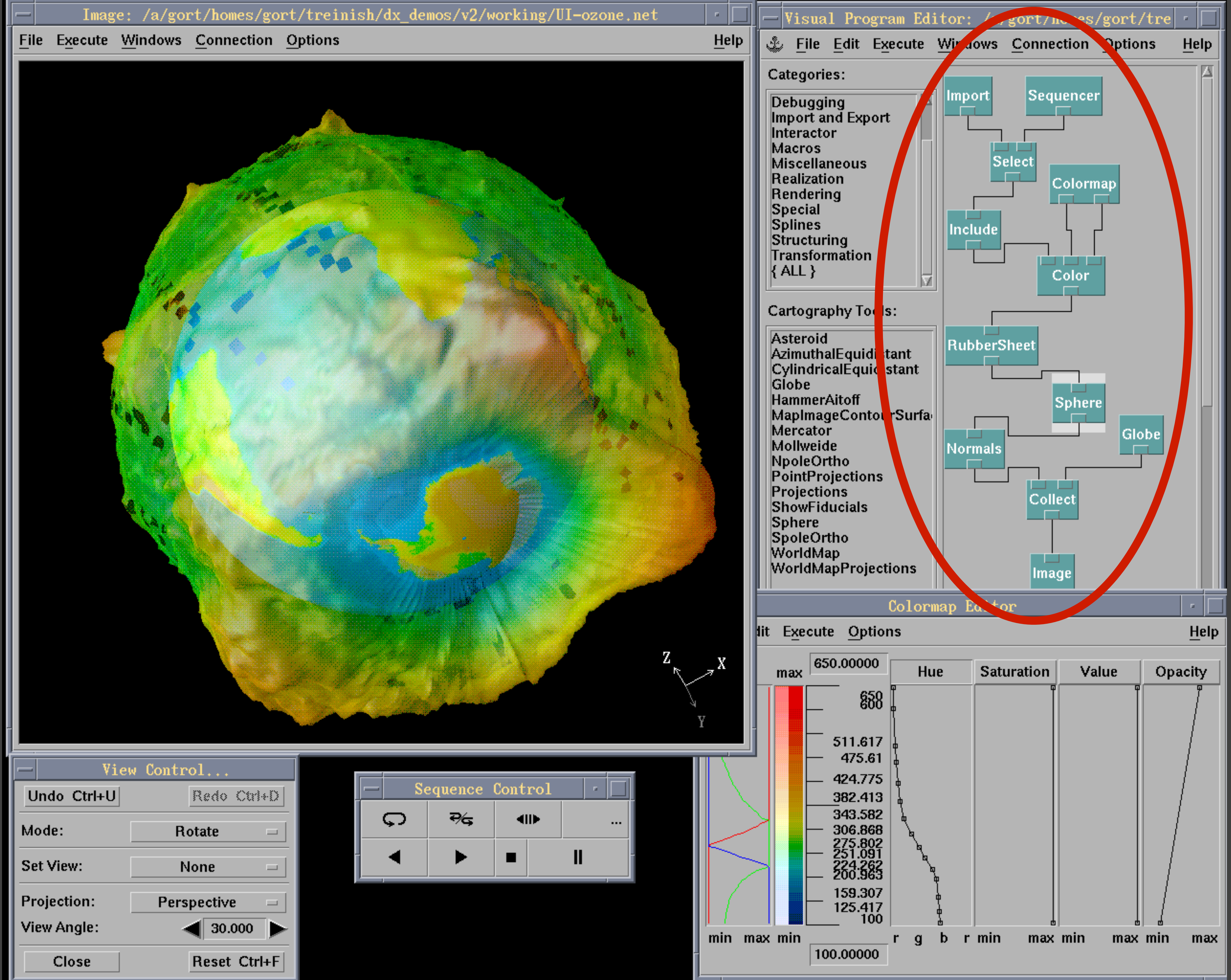
Canvas, OpenGL, Processing

Data
Visual Form
Task
Raw Data
Data Tables
Visual Structures
Views
Data Transformations
Visual Encodings
View Transformations

Data State Model

[Chi 98]





Mode: Rotate

Set View: None

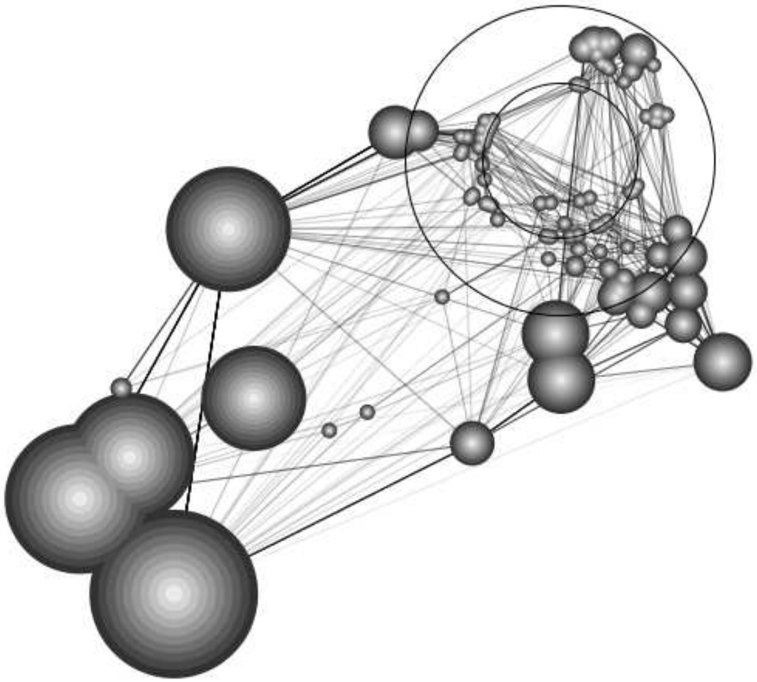
Projection: Perspective

View Angle: 30.000

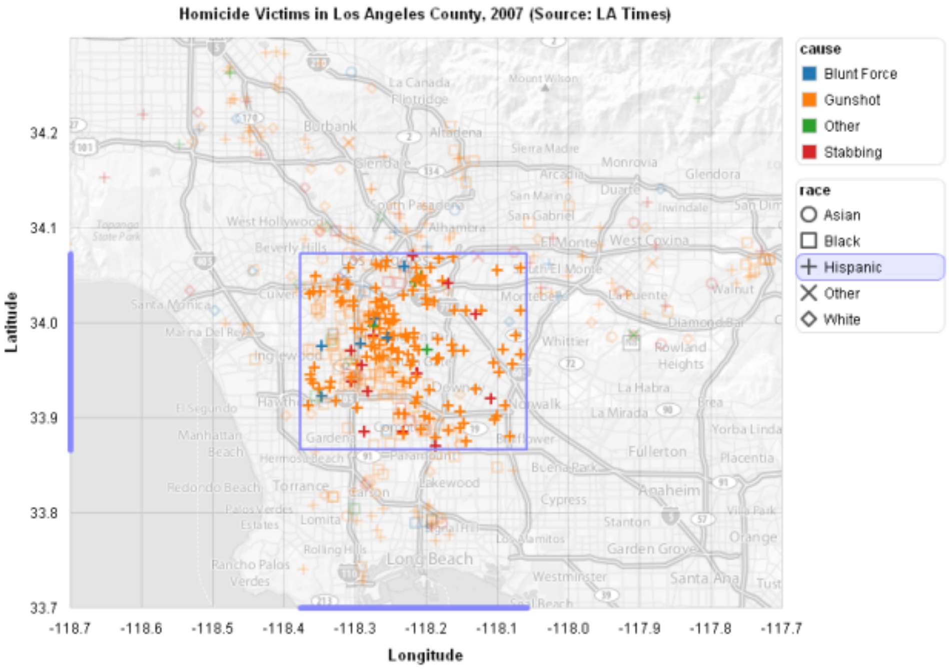
Prefuse & Flare

Operator-based toolkits for visualization design

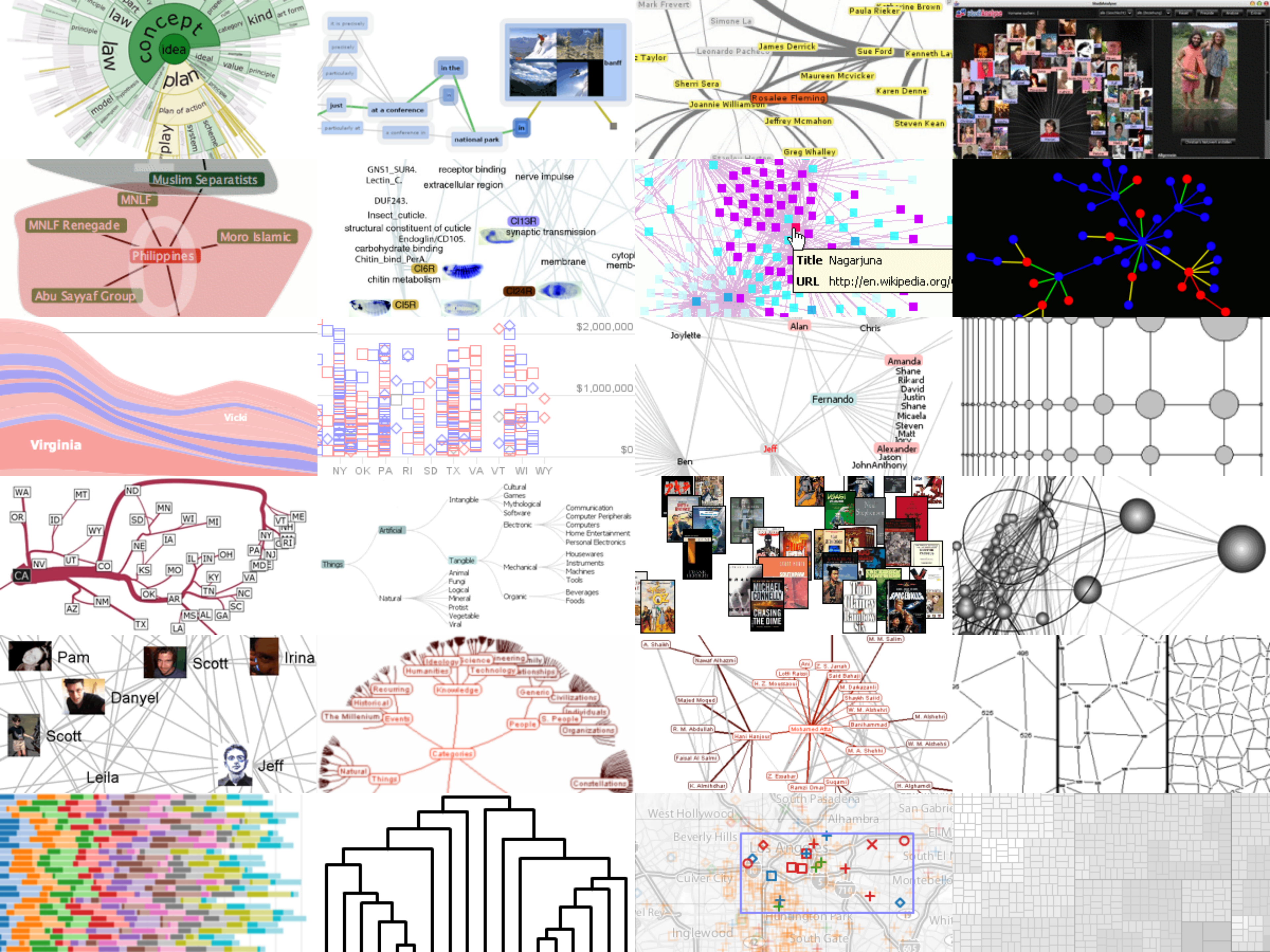
Vis = (Input Data -> Visual Objects) + Operators



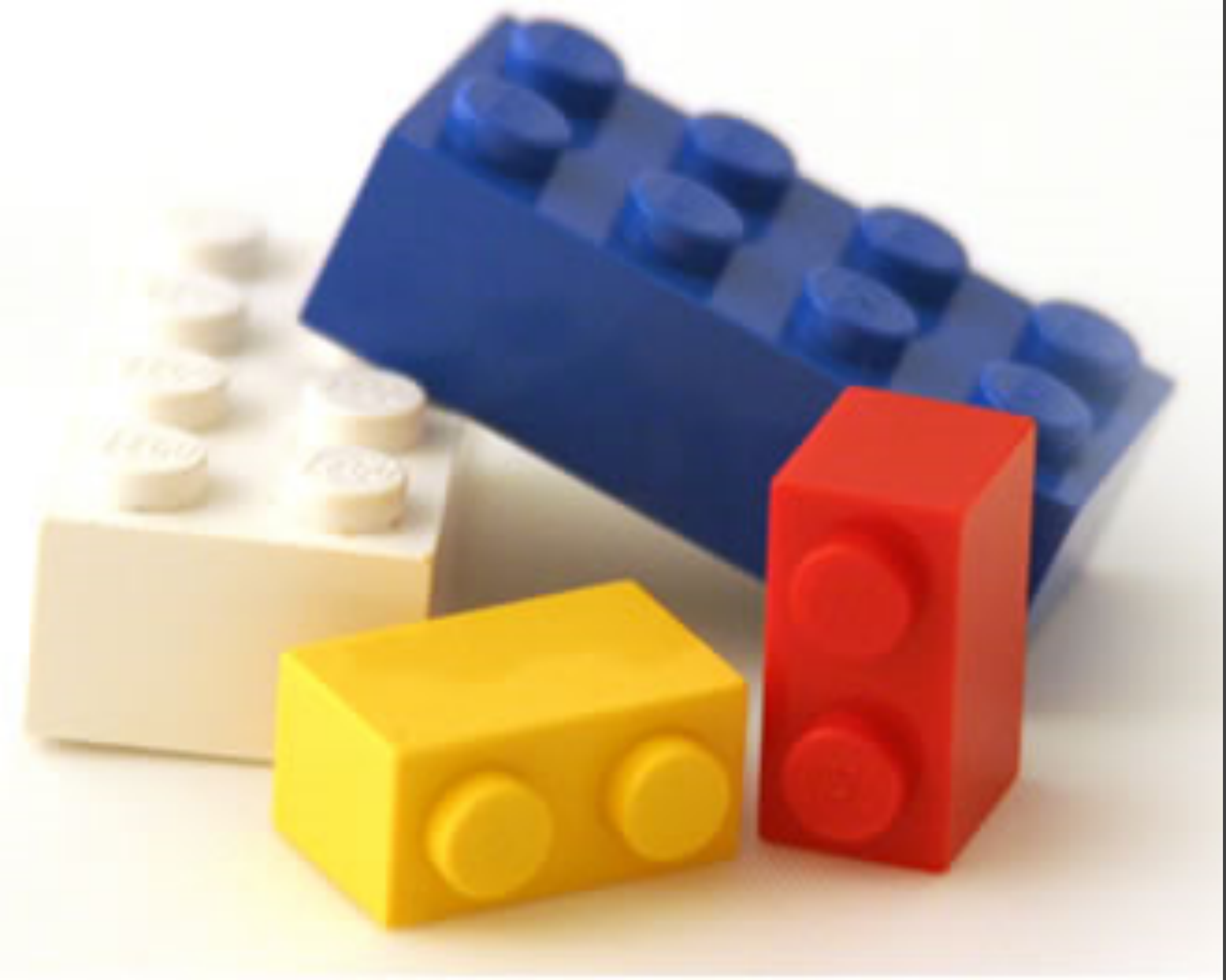
Prefuse (http://prefuse.org)



Flare (http://flare.prefuse.org)



concept
idea
law
plan
kind
principle
ideal
value
model
plan of action
play
system
scheme
in the
at a conference
national park
just
in
banff
Mark Frevert
Simone La
Paula Rieker
Katherine Brown
James Derrick
Sue Ford
Kenneth Lay
Leonardo Pacheco
Taylor
Maureen Mcvicker
Sherri Sera
Karen Denne
Rosalee Fleming
Joannie Williamson
Jeffrey Mcmahon
Steven Kean
Greg Whalley
Muslim Separatists
MNLF
MNLF Renegade
Moro Islamic
Philippines
Abu Sayyaf Group
GNS1_SUR4.
receptor binding
Lectin_C.
extracellular region
nerve impulse
DUF243.
Insect_cuticle.
CI13R
structural constituent of cuticle
synaptic transmission
Endoglin/CD105.
carbohydrate binding
Chitin_bind_PerA.
membrane
CI6R
chitin metabolism
CI24R
CI5R
Title Nagarjuna
URL http://en.wikipedia.org/
Virginia
Vicki
$2,000,000
$1,000,000
$0
NY
OK
PA
RI
SD
TX
VA
VT
WI
WY
Joylette
Alan
Chris
Amanda
Shane
Rikard
David
Justin
Fernando
Shane
Micaela
Steven
Matt
Jeff
Alexander
Jason
Ben
JohnAnthony
WA
MT
ND
OR
ID
MN
SD
WI
MI
WY
VT
ME
NH
IA
NY
RI
NE
CT
PA
NJ
NV
UT
CO
IL
IN
OH
CA
KS
MO
MD
DE
KY
VA
OK
TN
NC
AZ
NM
AR
SC
MS
AL
GA
TX
LA
Things
Artificial
Intangible
Cultural
Games
Mythological
Software
Electronic
Communication
Computer Peripherals
Computers
Home Entertainment
Personal Electronics
Tangible
Mechanical
Housewares
Instruments
Machines
Tools
Natural
Animal
Fungi
Logical
Mineral
Protist
Vegetable
Viral
Organic
Beverages
Foods
MICHAEL CONNELLY
CHASING THE DIME
Pam
Scott
Irina
Danyel
Scott
Leila
Jeff
Ideology
Science
Engineering
Humanities
Technology
Relationships
Recurring
Knowledge
Historical
Generic
Civilizations
The Millenium
Events
Individuals
People
S. People
Organizations
Categories
Natural
Things
Constellations
A. Shaikh
M. M. Salim
Nawaf Alhazmi
Ani
Z. S. Jarrah
Lotfi Raissi
Said Bahaji
H. Z. Moussaoui
M. Darkazanli
Majed Moqed
Shaykh Saiid
W. M. Alshehri
Mohamed Atta
M. Alshehri
R. M. Abdullah
Hani Hanjour
Banihammad
Faisal Al Salmi
M. A. Shehhi
W. M. Alshehri
Z. Essabar
Suqami
K. Almihdhar
Ramzi Omar
H. Alghamdi
498
526
526
West Hollywood
South Pasadena
San Gabriel
Alhambra
Beverly Hills
Los Angeles
El M
South El M
Culver City
Montebello
Rey
Huntington Park
Inglewood
South Gate
Whit



Component Architectures

Prefuse, Flare, Improvise, VTK

Graphics APIs

Canvas, OpenGL, Processing

Chart Typologies

Excel, Google Charts

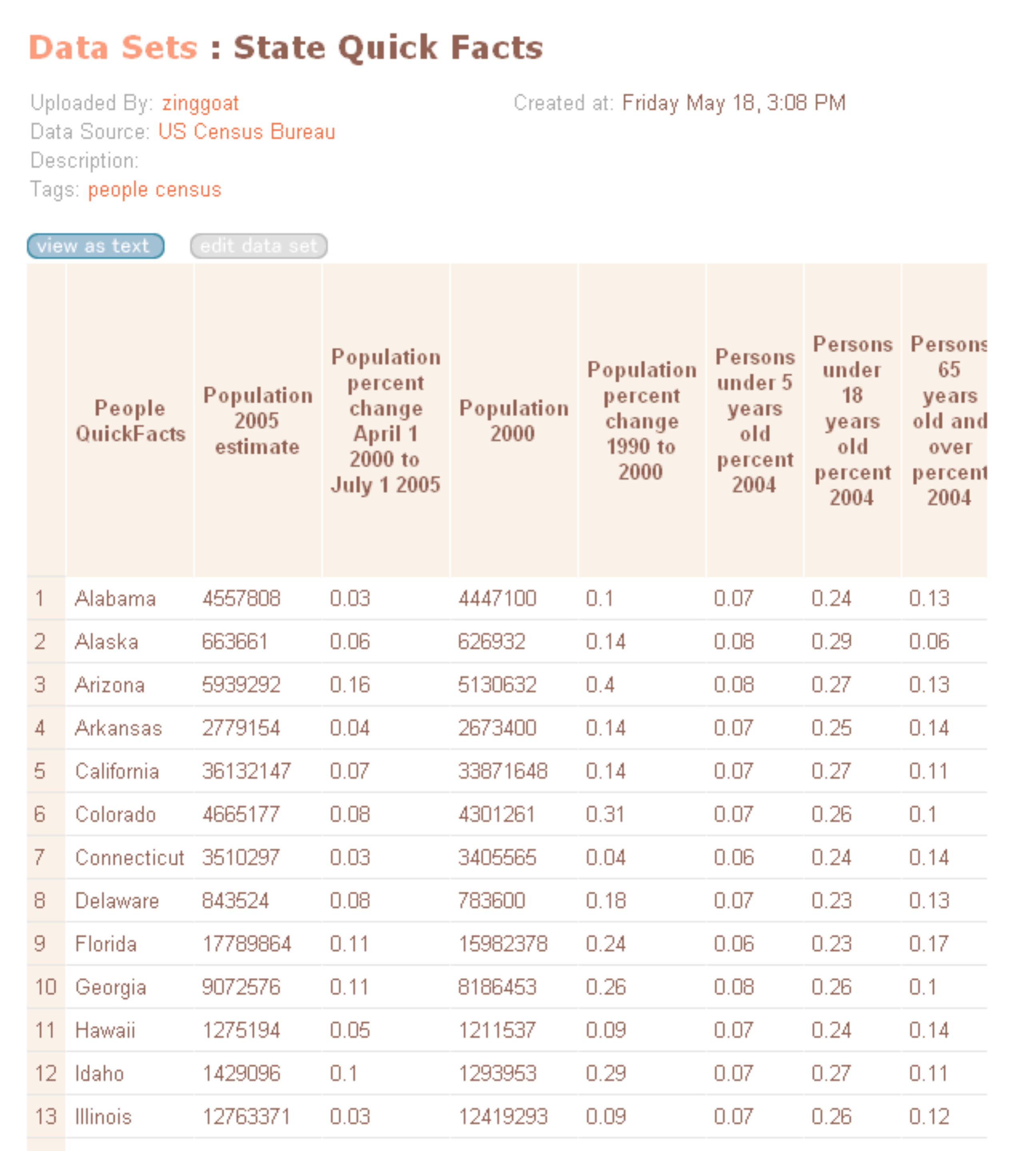
Component Architectures

Prefuse, Flare, Improvise, VTK

Graphics APIs

Canvas, OpenGL, Processing

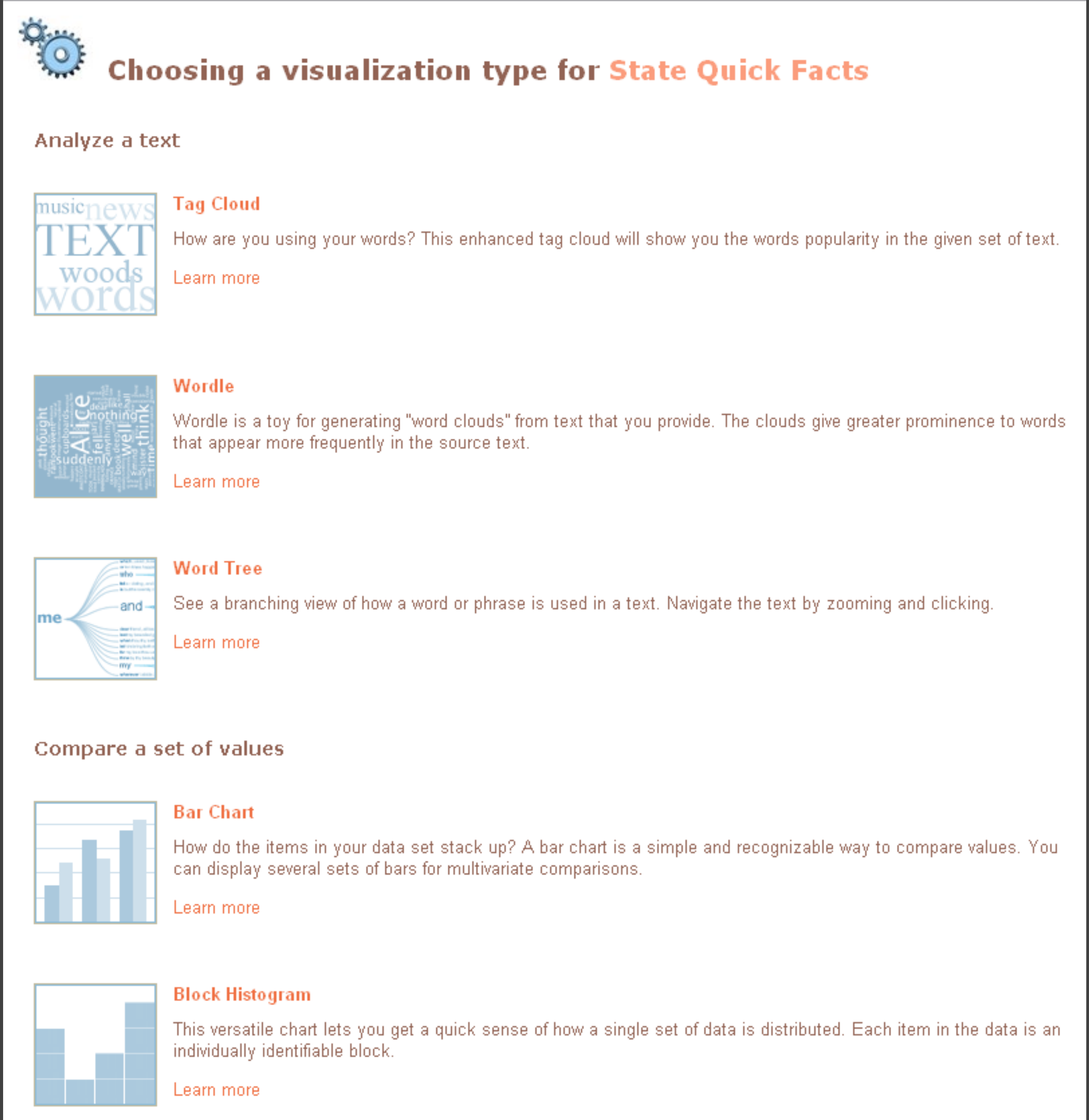
Chart Typologies



Data Sets : State Quick Facts

Uploaded By: zinggoat Created at: Friday May 18, 3:08 PM
Data Source: US Census Bureau
Description:
Tags: people census

	People QuickFacts	Population 2005 estimate	Population percent change April 1 2000 to July 1 2005	Population 2000	Population percent change 1990 to 2000	Persons under 5 years old percent 2004	Persons under 18 years old percent 2004	Persons 65 years old and over percent 2004
1	Alabama	4557808	0.03	4447100	0.1	0.07	0.24	0.13
2	Alaska	663661	0.06	626932	0.14	0.08	0.29	0.06
3	Arizona	5939292	0.16	5130632	0.4	0.08	0.27	0.13
4	Arkansas	2779154	0.04	2673400	0.14	0.07	0.25	0.14
5	California	36132147	0.07	33871648	0.14	0.07	0.27	0.11
6	Colorado	4665177	0.08	4301261	0.31	0.07	0.26	0.1
7	Connecticut	3510297	0.03	3405565	0.04	0.06	0.24	0.14
8	Delaware	843524	0.08	783600	0.18	0.07	0.23	0.13
9	Florida	17789864	0.11	15982378	0.24	0.06	0.23	0.17
10	Georgia	9072576	0.11	8186453	0.26	0.08	0.26	0.1
11	Hawaii	1275194	0.05	1211537	0.09	0.07	0.24	0.14
12	Idaho	1429096	0.1	1293953	0.29	0.07	0.27	0.11
13	Illinois	12763371	0.03	12419293	0.09	0.07	0.26	0.12



Choosing a visualization type for State Quick Facts

Analyze a text

Tag Cloud

How are you using your words? This enhanced tag cloud will show you the words popularity in the given set of text.

Learn more

Wordle

Wordle is a toy for generating "word clouds" from text that you provide. The clouds give greater prominence to words that appear more frequently in the source text.

Learn more

Word Tree

See a branching view of how a word or phrase is used in a text. Navigate the text by zooming and clicking.

Learn more

Compare a set of values

Bar Chart

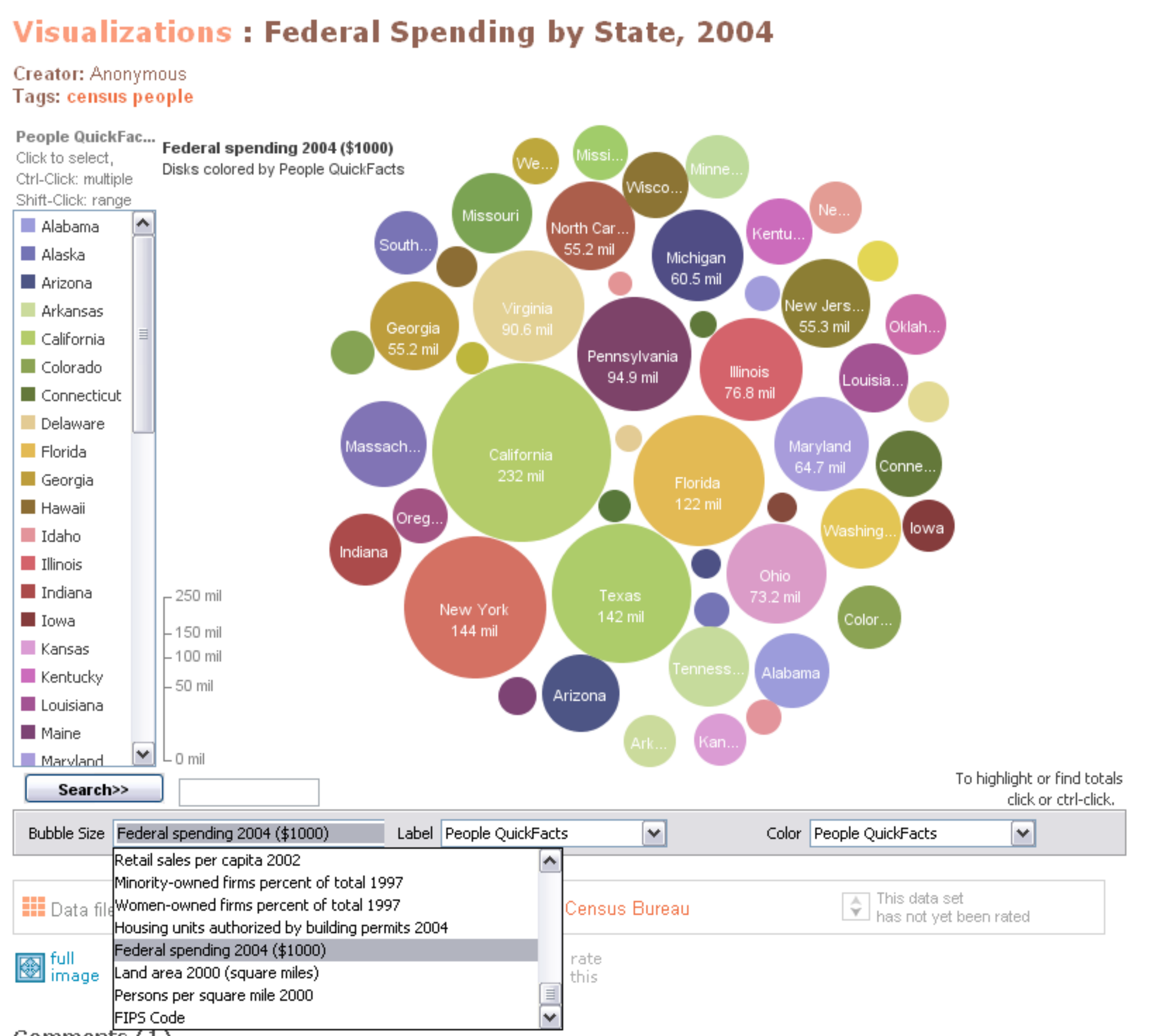
How do the items in your data set stack up? A bar chart is a simple and recognizable way to compare values. You can display several sets of bars for multivariate comparisons.

Learn more

Block Histogram

This versatile chart lets you get a quick sense of how a single set of data is distributed. Each item in the data is an individually identifiable block.

Learn more



Visualizations : Federal Spending by State, 2004

Creator: Anonymous
Tags: census people

Bubble Size: Federal spending 2004 ($1000)
Label: People QuickFacts
Color: People QuickFacts

Retail sales per capita 2002
Minority-owned firms percent of total 1997
Women-owned firms percent of total 1997
Housing units authorized by building permits 2004
Federal spending 2004 ($1000)
Land area 2000 (square miles)
Persons per square mile 2000
FIPS Code
Data file
Census Bureau
This data set
has not yet been rated
full
image
rate
this
Comments (1)

MAD LIBS®

MY MUSIC LESSON

Every Wednesday, when I get home from school, I have a piano
lesson. My teacher is a very strict house. Her name is
NOUN
Hillary Clinton. Our piano is a Steinway Concert tree
CELEBRITY (FEMALE) NOUN
and it has 88 ~~[illegible]~~ cups. It also has a soft pedal and a/an
PLURAL NOUN
Smily pedal. When I have a lesson, I sit down on the piano
ADJECTIVE
AlBERTO and play for 16 Minuts. I do scales to
NOUN PERIOD OF TIME
exercise my cats, and then I usually play a minuet by
PLURAL NOUN
Johann Sebastian washington. Teacher says I am a natural
CELEBRITY (LAST NAME)
Haunted House and have a good musical leg. Perhaps
NOUN PART OF THE BODY
when I get better I will become a concert vet and give
PROFESSION
a recital at Carnegie hospital.
TYPE OF BUILDING

[M]ost charting packages channel user requests into a **rigid array of chart types**. To atone for this lack of flexibility, they offer a kit of post-creation editing tools to return the image to what the user originally envisioned. **They give the user an impression of having explored data rather than the experience**.

Leland Wilkinson
The Grammar of Graphics, 1999

Chart Typologies

Excel, Many Eyes, Google Charts

Component Architectures

Prefuse, Flare, Improvise, VTK

Graphics APIs

Canvas, OpenGL, Processing

Chart Typologies

Excel, Many Eyes, Google Charts

Visual Analysis Grammars

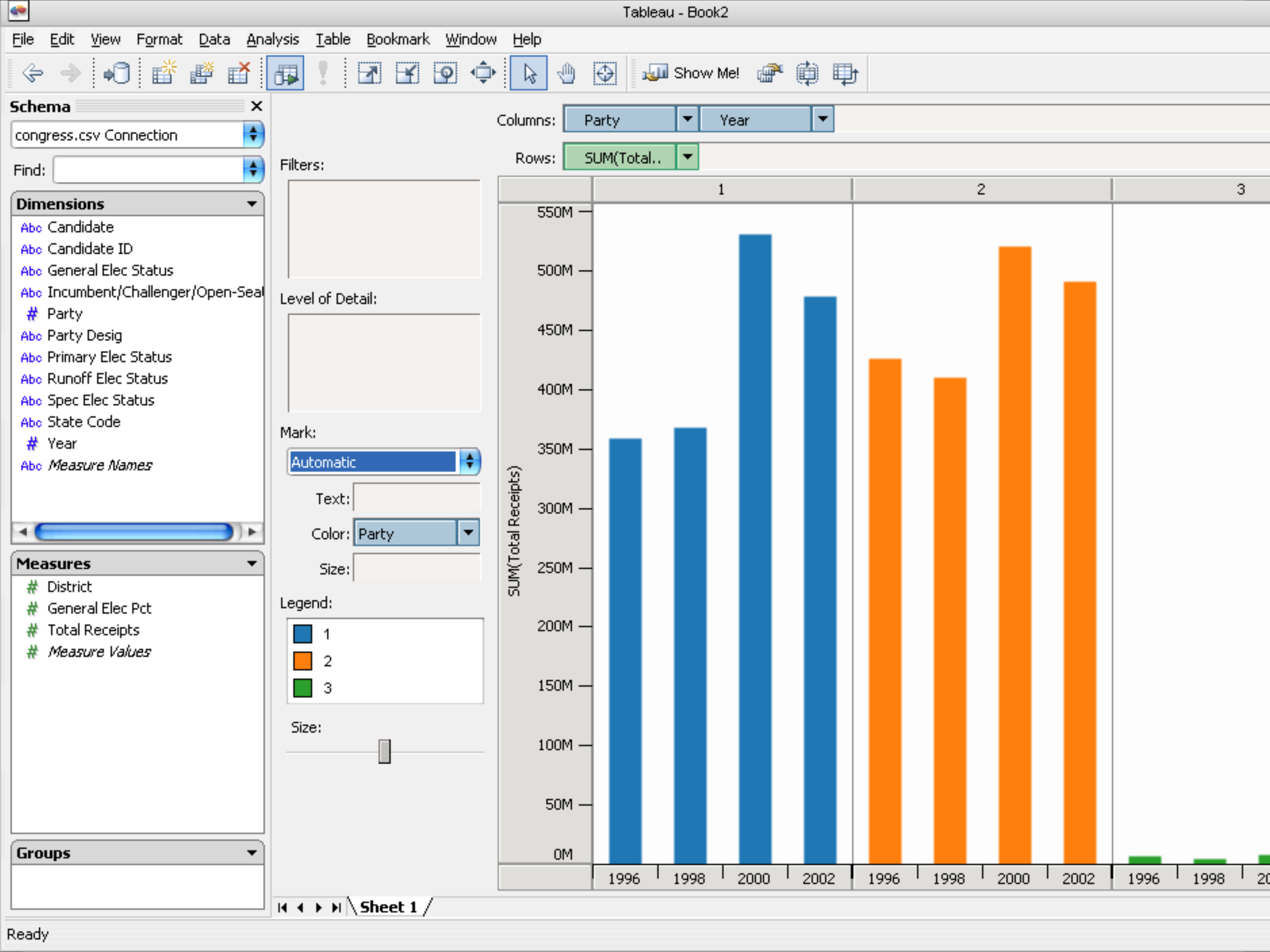
VizQL, ggplot2, Vega-Lite

Component Architectures

Prefuse, Flare, Improvise, VTK

Graphics APIs

Canvas, OpenGL, Processing



congress.csv Connection

Find:

Dimensions:
- Abc Candidate
- Abc Candidate ID
- Abc General Elec Status
- Abc Incumbent/Challenger/Open-Seat
- # Party
- Abc Party Desig
- Abc Primary Elec Status
- Abc Runoff Elec Status
- Abc Spec Elec Status
- Abc State Code
- # Year
- Abc *Measure Names*

Measures:
- # District
- # General Elec Pct
- # Total Receipts
- # *Measure Values*

Columns: Party | Year

Rows: SUM(Total..

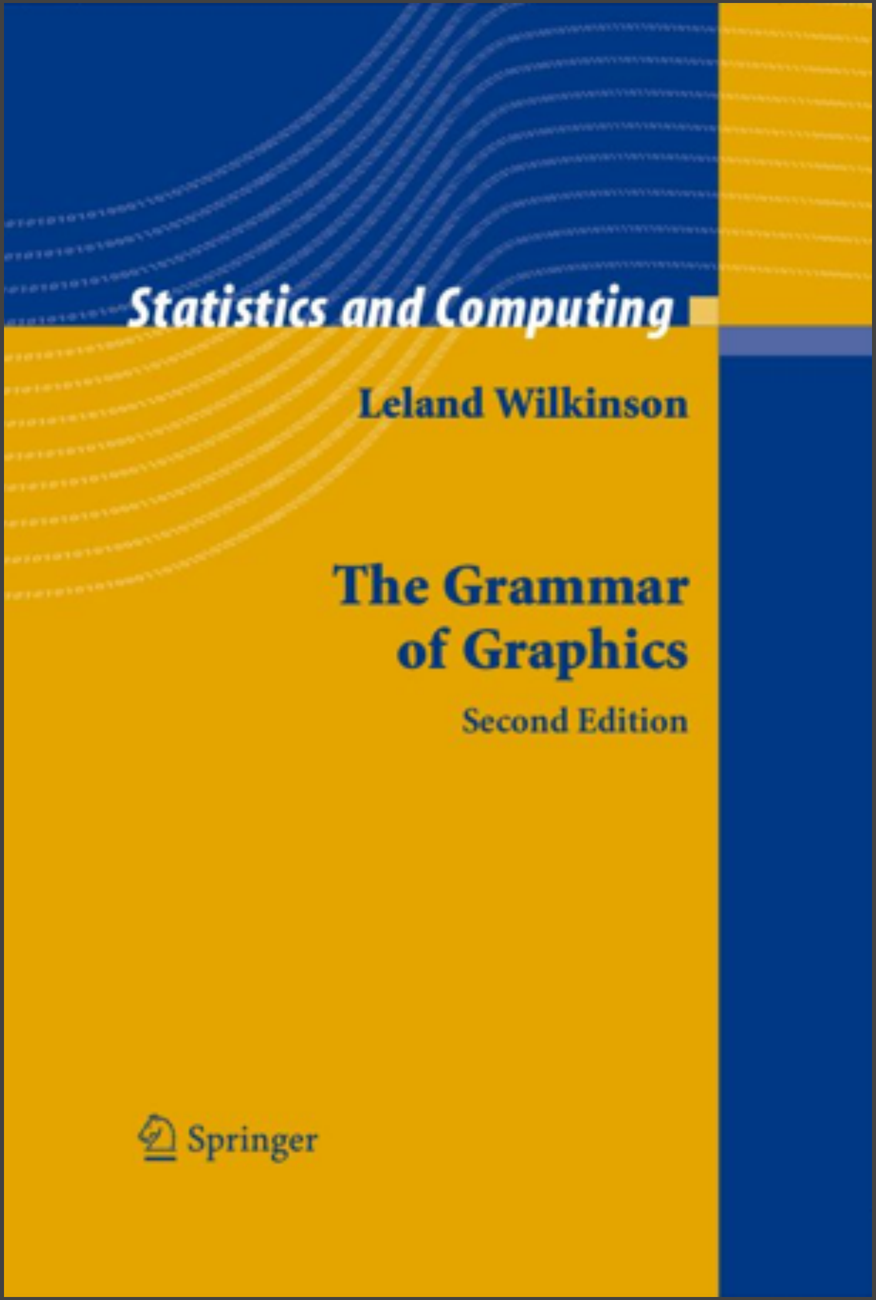
Mark: Automatic

Text:

Color: Party

Size:

Legend:
- 🟦 1
- 🟧 2
- 🟩 3



Statistics and Computing

Leland Wilkinson

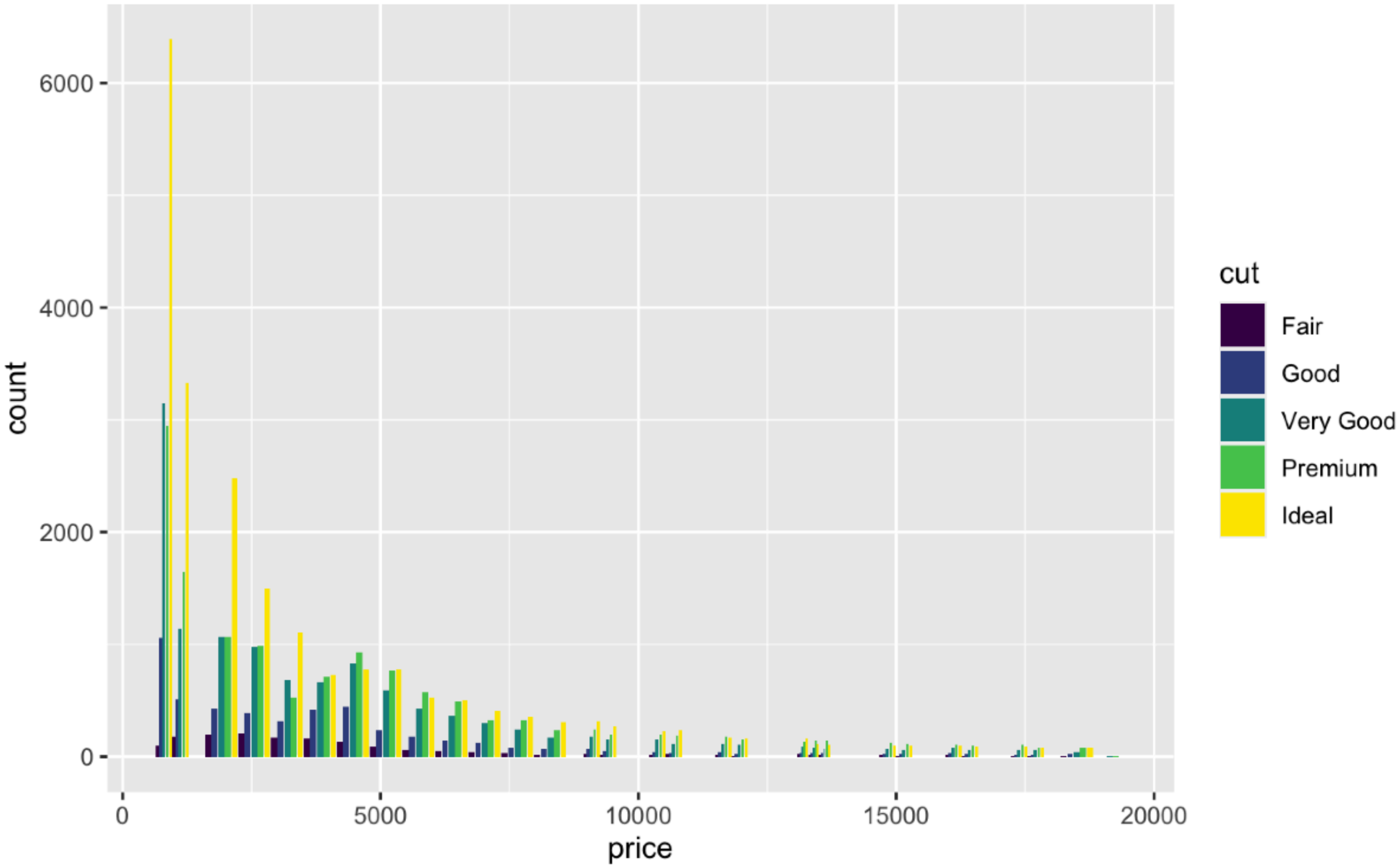
The Grammar of Graphics

Second Edition

Springer

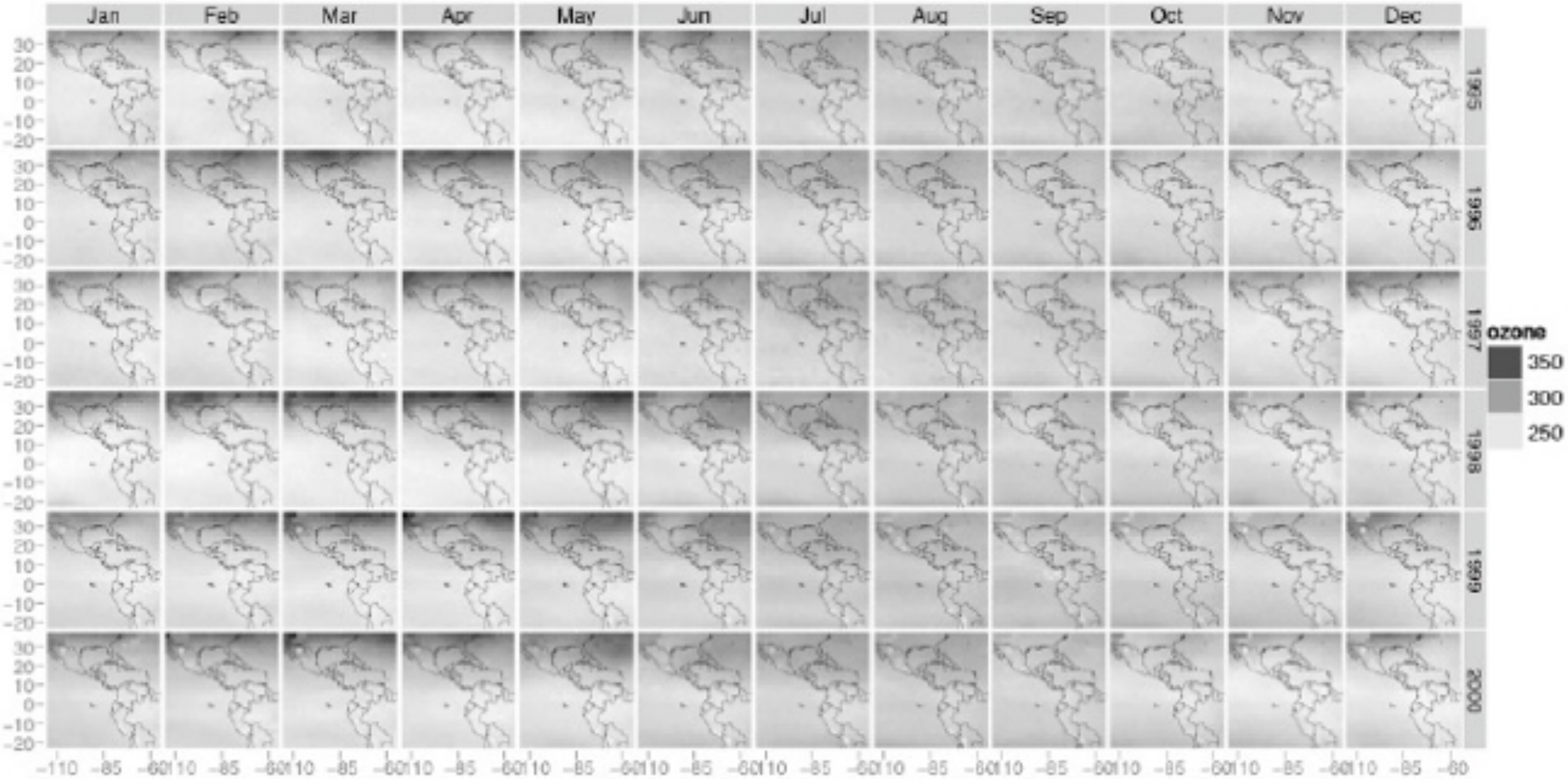
```
ggplot(diamonds, aes(x=price, fill=cut))
+ geom_bar(position="dodge")
```

ggplot2



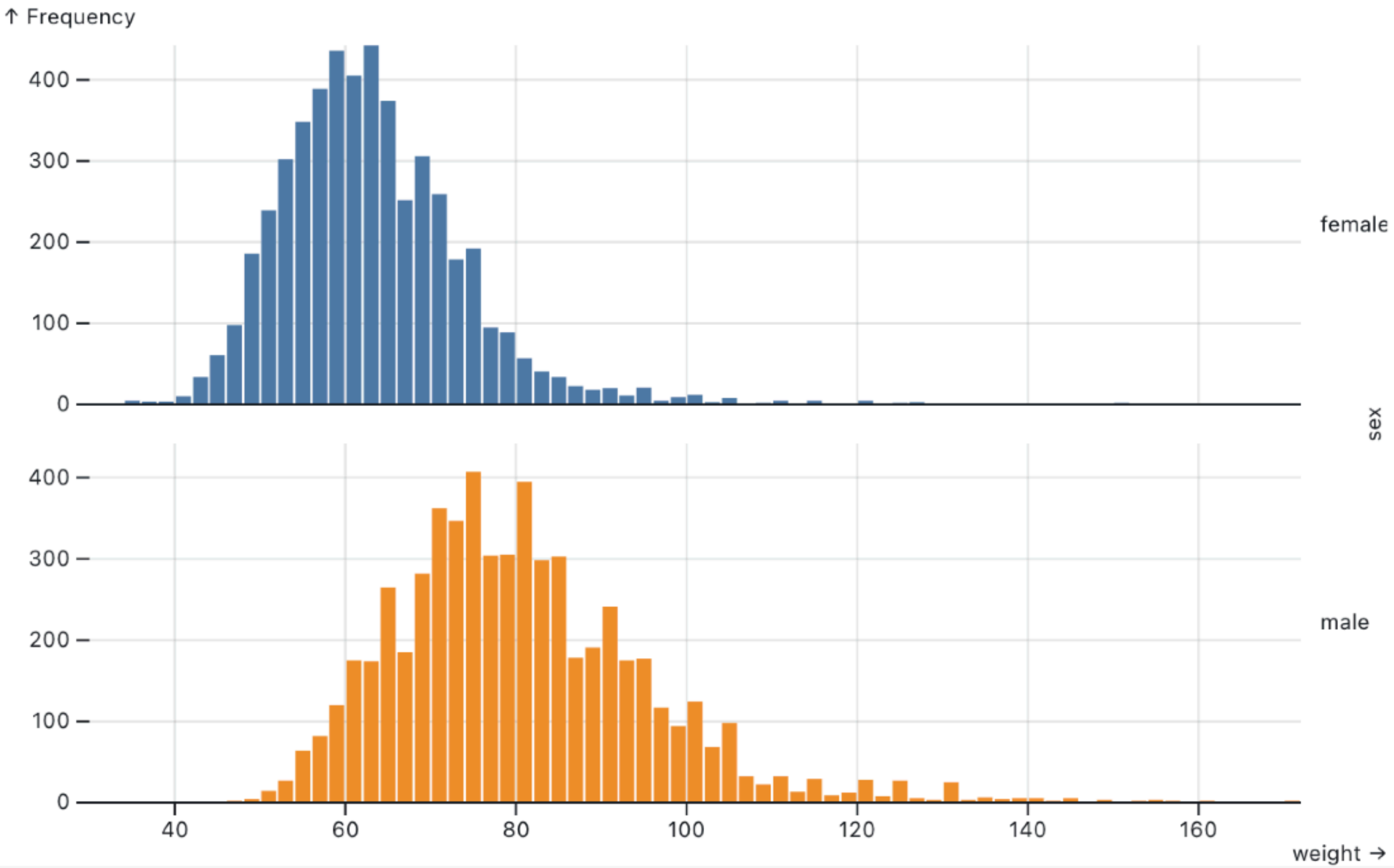
```
ggplot(diamonds, aes(x=price, fill=cut))
+ geom_bar(position="dodge")
```

ggplot2



```
qplot(long, lat, data = expo, geom = "tile", fill = ozone,
  facets = year ~ month) +
scale_fill_gradient(low = "white", high = "black") + map
```

ggplot2



```
Plot.plot({
  grid: true,
  facet: {
    data: athletes,
    y: "sex"
  },
  marks: [
    Plot.rectY(athletes, Plot.binX({y: "count"}, {x: "weight", fill: "sex"})),
    Plot.ruleY([0])
  ]
})
```

Observable Plot

Chart Typologies

Excel, Many Eyes, Google Charts

Visual Analysis Grammars

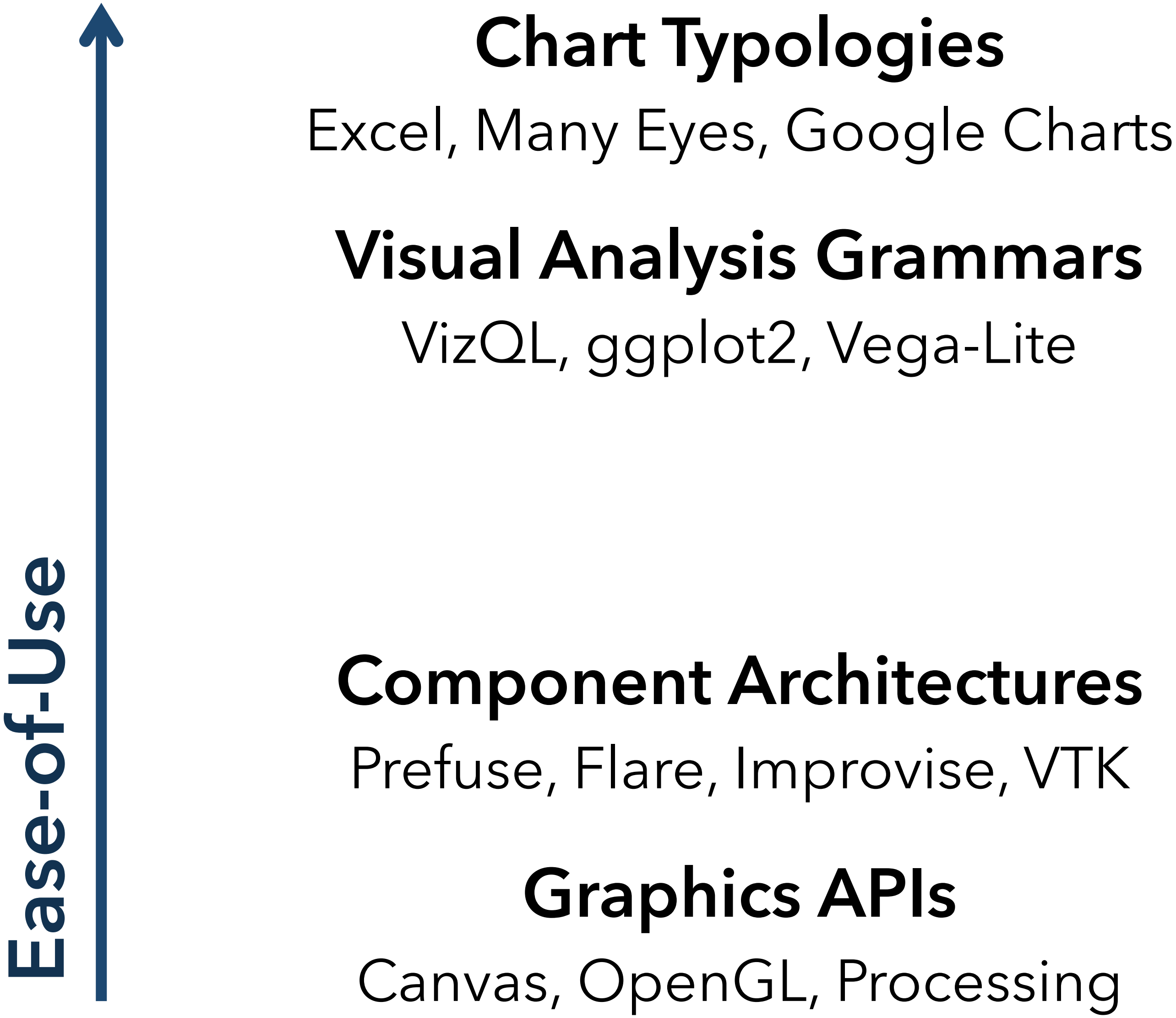
VizQL, ggplot2, Vega-Lite

Component Architectures

Prefuse, Flare, Improvise, VTK

Graphics APIs

Canvas, OpenGL, Processing



Ease-of-Use

Chart Typologies

Excel, Many Eyes, Google Charts

Visual Analysis Grammars

VizQL, ggplot2, Vega-Lite

Component Architectures

Prefuse, Flare, Improvise, VTK

Graphics APIs

Canvas, OpenGL, Processing

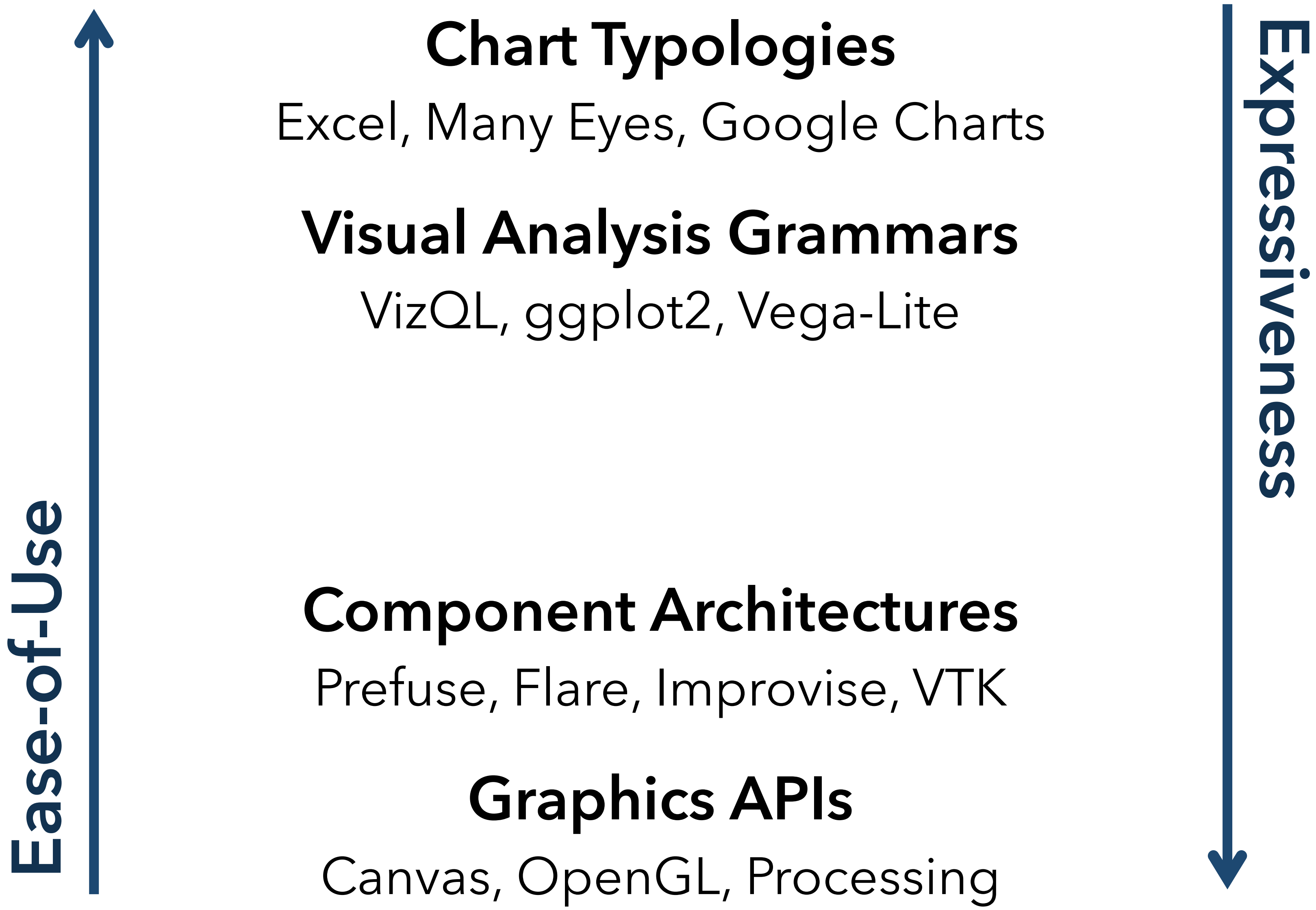


Chart Typologies

Excel, Many Eyes, Google Charts

Visual Analysis Grammars

VizQL, ggplot2, Vega-Lite

Component Architectures

Prefuse, Flare, Improvise, VTK

Graphics APIs

Canvas, OpenGL, Processing

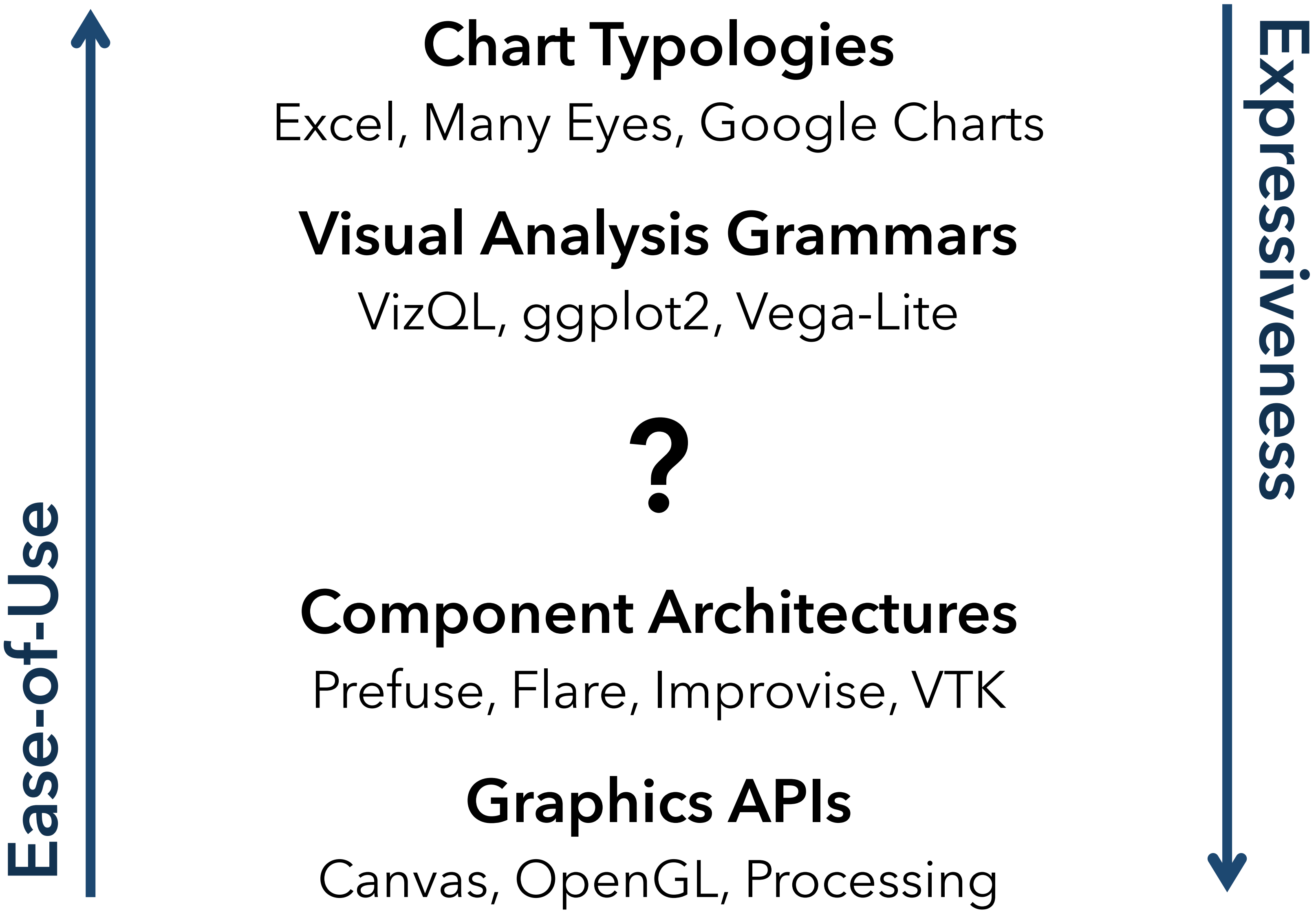


Chart Typologies

Excel, Many Eyes, Google Charts

Visual Analysis Grammars

VizQL, ggplot2, Vega-Lite

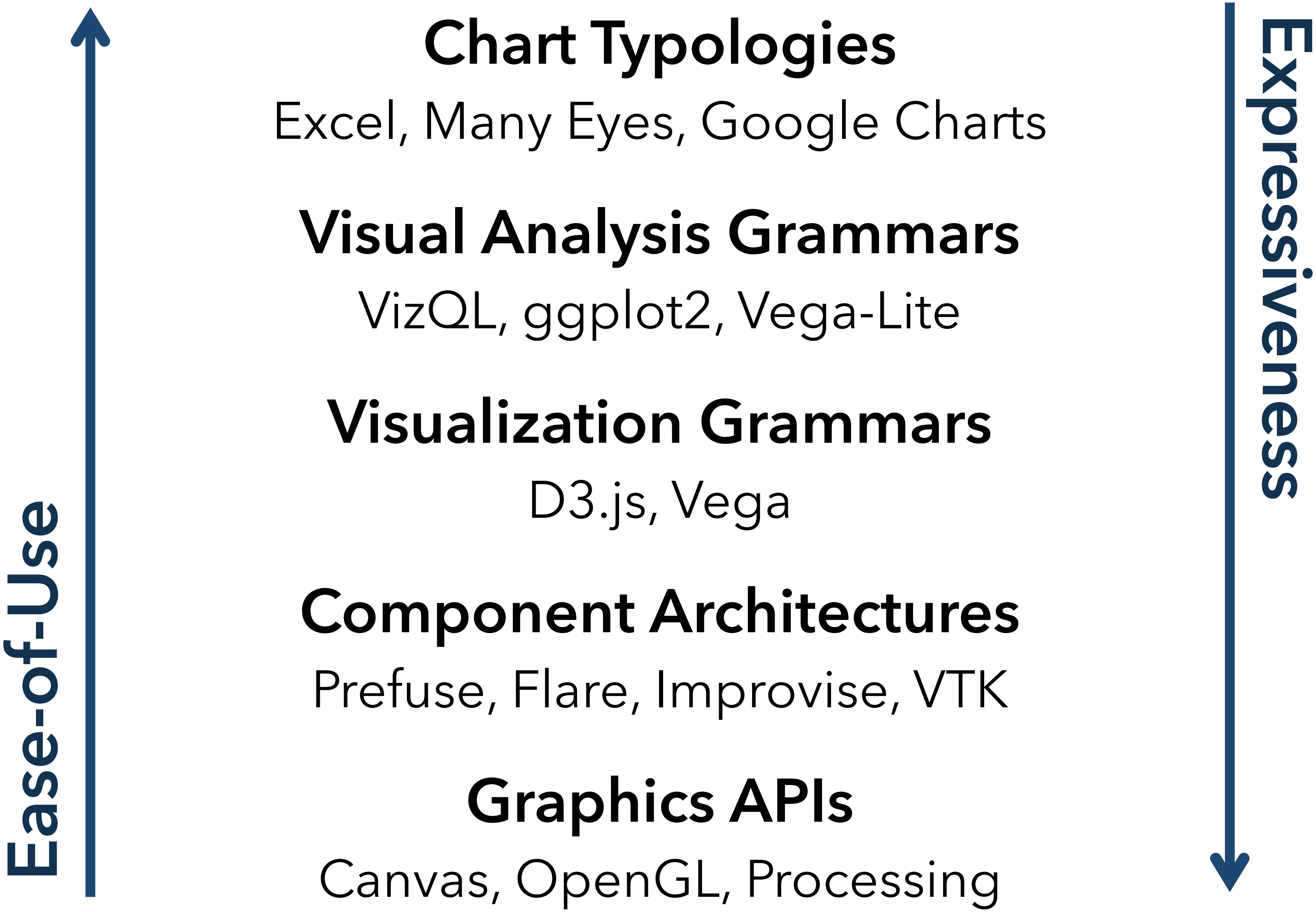
?

Component Architectures

Prefuse, Flare, Improvise, VTK

Graphics APIs

Canvas, OpenGL, Processing



Ease-of-Use

Chart Typologies

Excel, Many Eyes, Google Charts

Visual Analysis Grammars

VizQL, ggplot2, Vega-Lite

Visualization Grammars

D3.js, Vega

Component Architectures

Prefuse, Flare, Improvise, VTK

Graphics APIs

Canvas, OpenGL, Processing

Expressiveness

Visualization Building Blocks

Visualization Building Blocks

Data	Input data to visualize

Visualization Building Blocks

Data	Input data to visualize
Transforms	Group, aggregate, stats, layout

Visualization Building Blocks

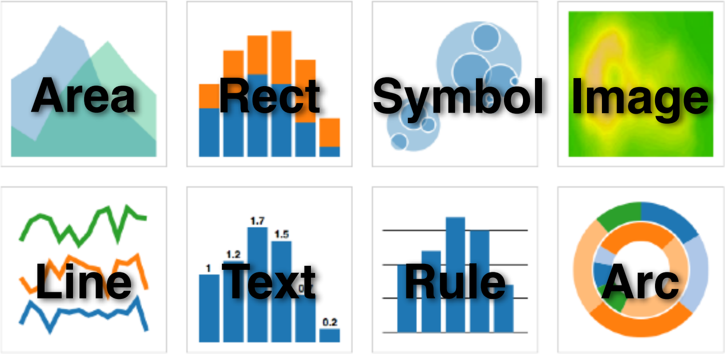
Data	Input data to visualize
Transforms	Group, aggregate, stats, layout
Scales	Map data values to visual values

Visualization Building Blocks

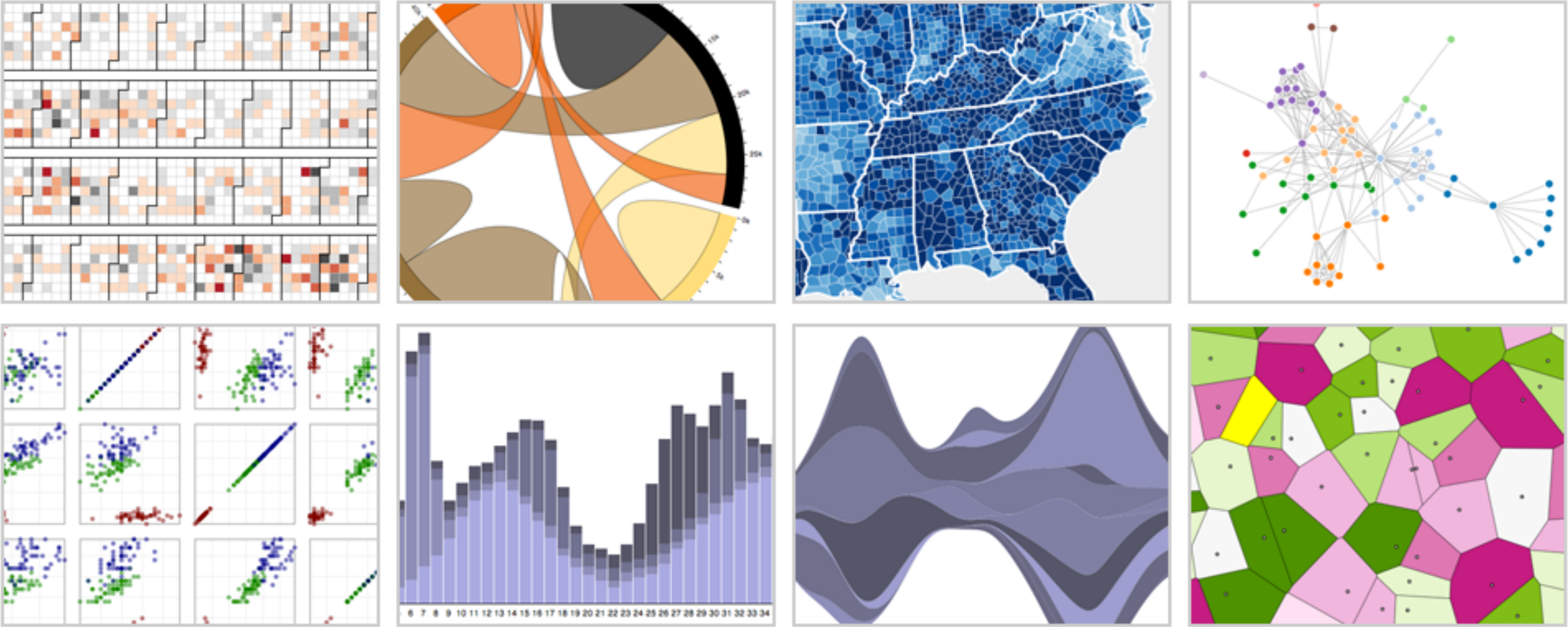
Data	Input data to visualize
Transforms	Group, aggregate, stats, layout
Scales	Map data values to visual values
Guides	Axes & legends visualize scales

Visualization Building Blocks

Data	Input data to visualize
Transforms	Group, aggregate, stats, layout
Scales	Map data values to visual values
Guides	Axes & legends visualize scales
Marks	Data-representative graphics



d3.js Data-Driven Documents



Mike Bostock, Vadim Ogievetsky, Jeffrey Heer [TVCG 2011]

+ Jason Davies (geo, 2011–13) & Philippe Rivière (2016–)

What is D3?

1. A collection of reusable visualization utilities

2. A tool for updating the browser’s Document Object Model (DOM) in response to input data

What is D3?

1. A collection of reusable visualization utilities

    Data: d3.csv, d3.json, …

    Scales: d3.scaleLinear, d3.scaleLog, …

    Projections: d3.geoPath, d3.geoMercator, …

    Layout: d3.tree, d3.treemap, d3.force, …

    Interaction: d3.brush, d3.zoom, …

2. A tool for updating the browser's Document Object Model (DOM) in response to input data

What is D3?

1. A collection of reusable visualization utilities

2. A tool for updating the browser's Document Object Model (DOM) in response to input data

    Select: query DOM content

    Join: bind input data to DOM elements

    Update: set DOM element properties

    Transition: animate changes over time

Why D3?

Enable highly custom visualization design

Support animation and dynamic displays

Support rich and varied interactions

Interoperate via web standards (HTML, SVG, CSS)

Avoid artificial limits. If a browser can do it, D3 should be able to take advantage of it.

Why D3?

"the authors have undeniably helped to bring data visualization to the mainstream. [D3] is a cornerstone contribution to this conference specifically and more generally to the success of our field as a whole"

IEEE VIS 2021 Test of Time Award

Why D3?

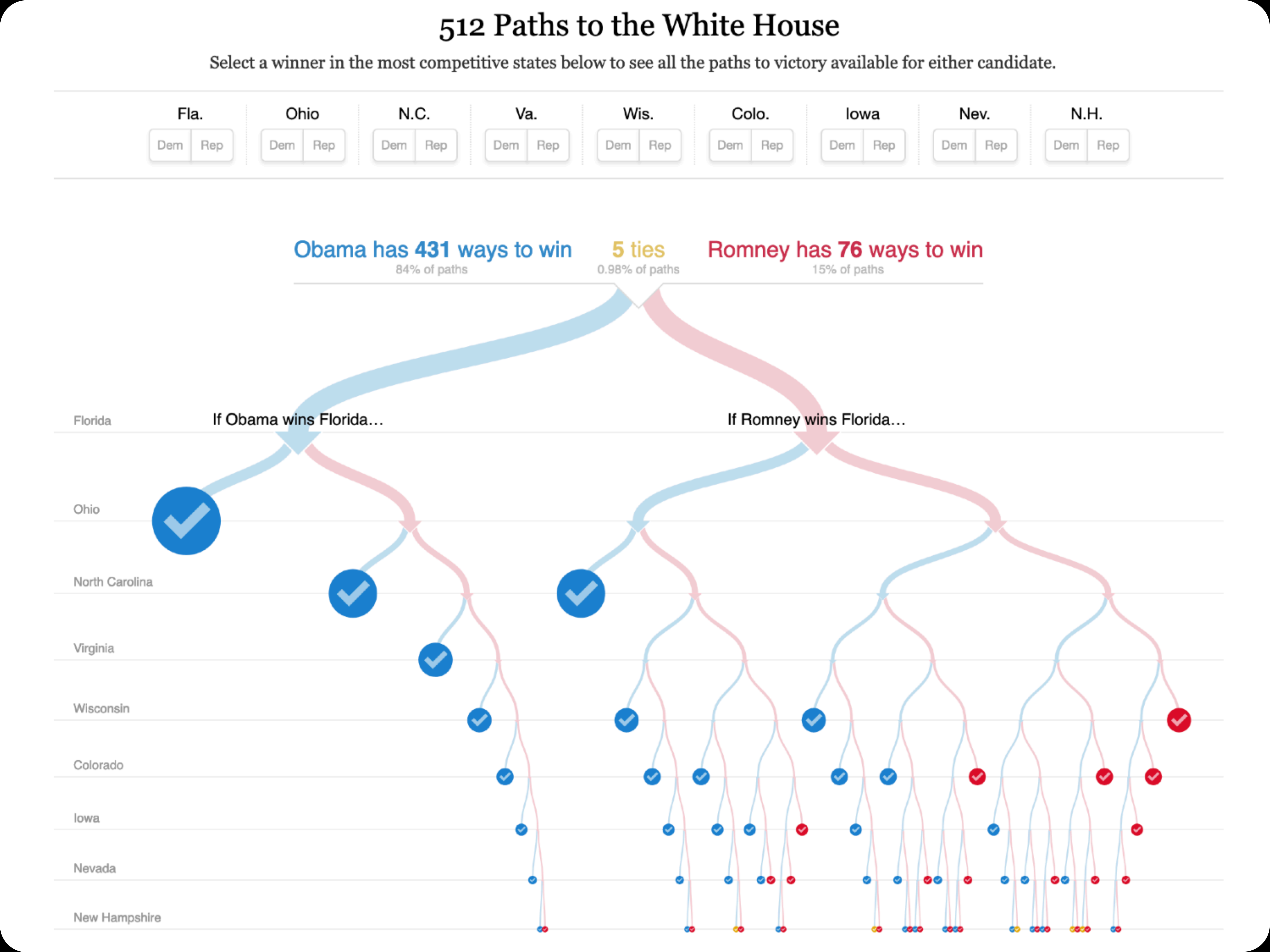
D3 “slingshotted the field into growth, diversification and creativity that has been unprecedented” and “changed how millions of data visualizations are created across newsrooms, websites, and personal portfolios”

Information is Beautiful 2022 Test of Time Award

Why D3?

“Use D3 if you think it’s perfectly normal to write a hundred lines of code for a bar chart.”

Amanda Cox, Former Graphics Editor, NY Times



512 Paths to the White House

Select a winner in the most competitive states below to see all the paths to victory available for either candidate.

Obama has **431** ways to win
84% of paths

5 ties
0.98% of paths

Romney has **76** ways to win
15% of paths

If Obama wins Florida…

If Romney wins Florida…

D3 Selections

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

D3 Selections

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

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

Data

DOM

Data

DOM

<svg width="500" …>

```
svg = d3.append("svg")
   .attr("width", 500)
   .attr("height", 300);
```

</svg>

D3 Selections

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

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

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

Data

DOM

<svg width=“500” …>

</svg>

Data

DOM

```
<svg width="500" ...>
```

```
svg.selectAll("rect")
```

???

```
</svg>
```

Data

DOM

```
<svg width=“500” …>
  <rect ..></rect>
  <rect ..></rect>
  <rect ..></rect>
  <rect ..></rect>
  <rect ..></rect>
</svg>
```

Data

svg.selectAll(“rect”)

DOM

```
<svg width=“500” ...>
  <rect ... />
  <rect ... />
  <rect ... />
  <rect ... />
  <rect ... />
</svg>
```

Data

```
svg.selectAll(“rect”)
 .attr(“width”, 100)
 .style(“fill”, “steelblue”)
```

DOM

```
<svg width=“500” ...>
 <rect width=“100”
  style=“fill: steelblue;”
 />
 <rect width=“100
  style=“fill: steelblue;”
 />
 <rect width=“100
  style=“fill: steelblue;”
```

Data Binding

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

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

Data Binding

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

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

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

Data

```
values = [
  { cat: “a”, value: 5  },
  { cat: “b”, value: 7 },
  { cat: “c”, value: 3 },
  { cat: “d”, value: 4 },
  { cat: “e”, value: 6 }
];
```

DOM

```
<svg width=500 …>






</svg>
```

Data

```
values = [
  { cat: “a”, value: 5  },
  { cat: “b”, value: 7 },
  { cat: “c”, value: 3 },
  { cat: “d”, value: 4 },
  { cat: “e”, value: 6 }
];
```

DOM

```
<svg width=500 ...>

   ? ? ? ? ?

</svg>
```

```
bars = svg.selectAll(“rect”) .data(values)
```

Data

```
values = [
  { cat: “a”, value: 5  },    ?
  { cat: “b”, value: 7 },     ?
  { cat: “c”, value: 3 },     ?
  { cat: “d”, value: 4 },     ?
  { cat: “e”, value: 6 }      ?
];
```

DOM

```
<svg width=500 ...>





</svg>
```

```
bars = svg.selectAll(“rect”) .data(values)
```

Data Binding

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

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

// Select SVG rectangles and bind them to data values.
bars = svg.selectAll(“rect.bars”).data(values);

// What if the DOM elements don’t exist yet? The enter set represents data
// values that do not yet have matching DOM elements.
bars.enter().append(“rect”).attr(“class”, “bars”);
```

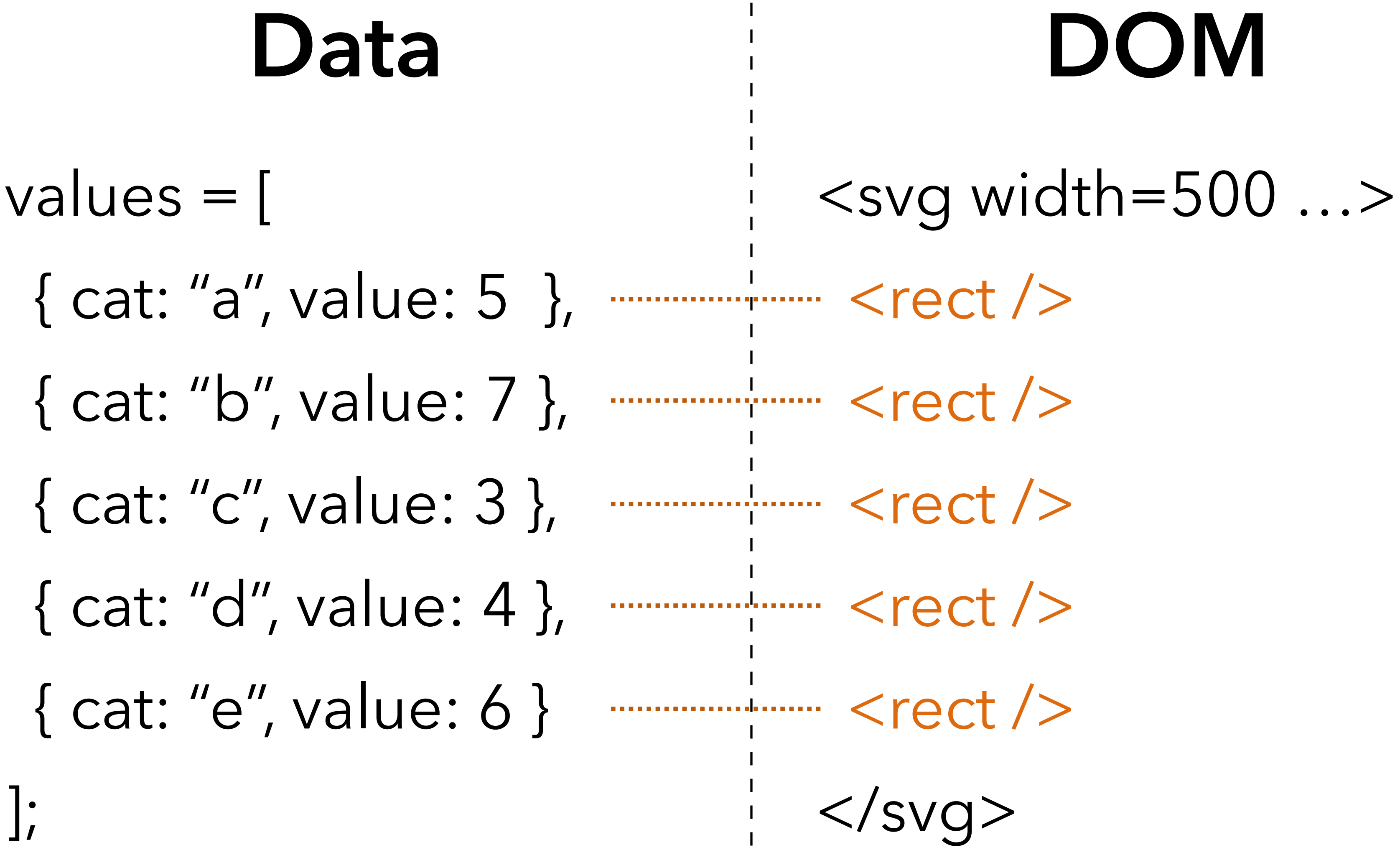
Data

```
values = [
  { cat: “a”, value: 5  },    ?
  { cat: “b”, value: 7 },     ?
  { cat: “c”, value: 3 },     ?
  { cat: “d”, value: 4 },     ?
  { cat: “e”, value: 6 }      ?
];
```

DOM

```
<svg width=500 ...>





</svg>
```

```
bars = svg.selectAll(“rect”) .data(values)
```



Data

DOM

values = [
  { cat: “a”, value: 5 },
  { cat: “b”, value: 7 },
  { cat: “c”, value: 3 },
  { cat: “d”, value: 4 },
  { cat: “e”, value: 6 }
];

<svg width=500 …>
  <rect />
  <rect />
  <rect />
  <rect />
  <rect />
</svg>

```
bars.enter().append(“rect”)
```

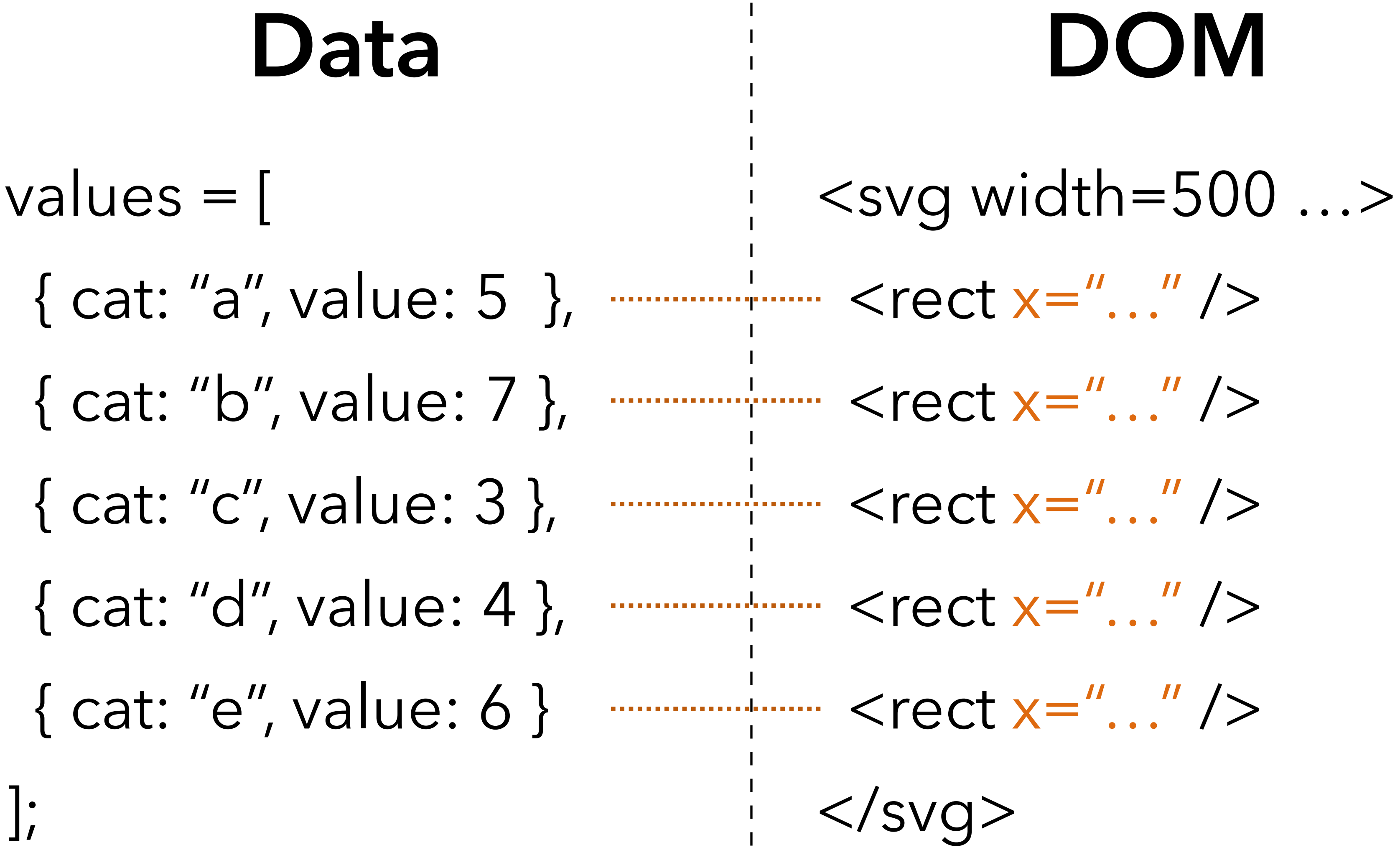
Data

```
values = [
  { cat: “a”, value: 5  },
  { cat: “b”, value: 7 },
  { cat: “c”, value: 3 },
  { cat: “d”, value: 4 },
  { cat: “e”, value: 6 }
];
```

DOM

```
<svg width=500 ...>
  <rect class=“bars” />
  <rect class=“bars” />
  <rect class=“bars” />
  <rect class=“bars” />
  <rect class=“bars” />
</svg>
```

```
bars.enter().append(“rect”).attr(“class”, “bars”)
```



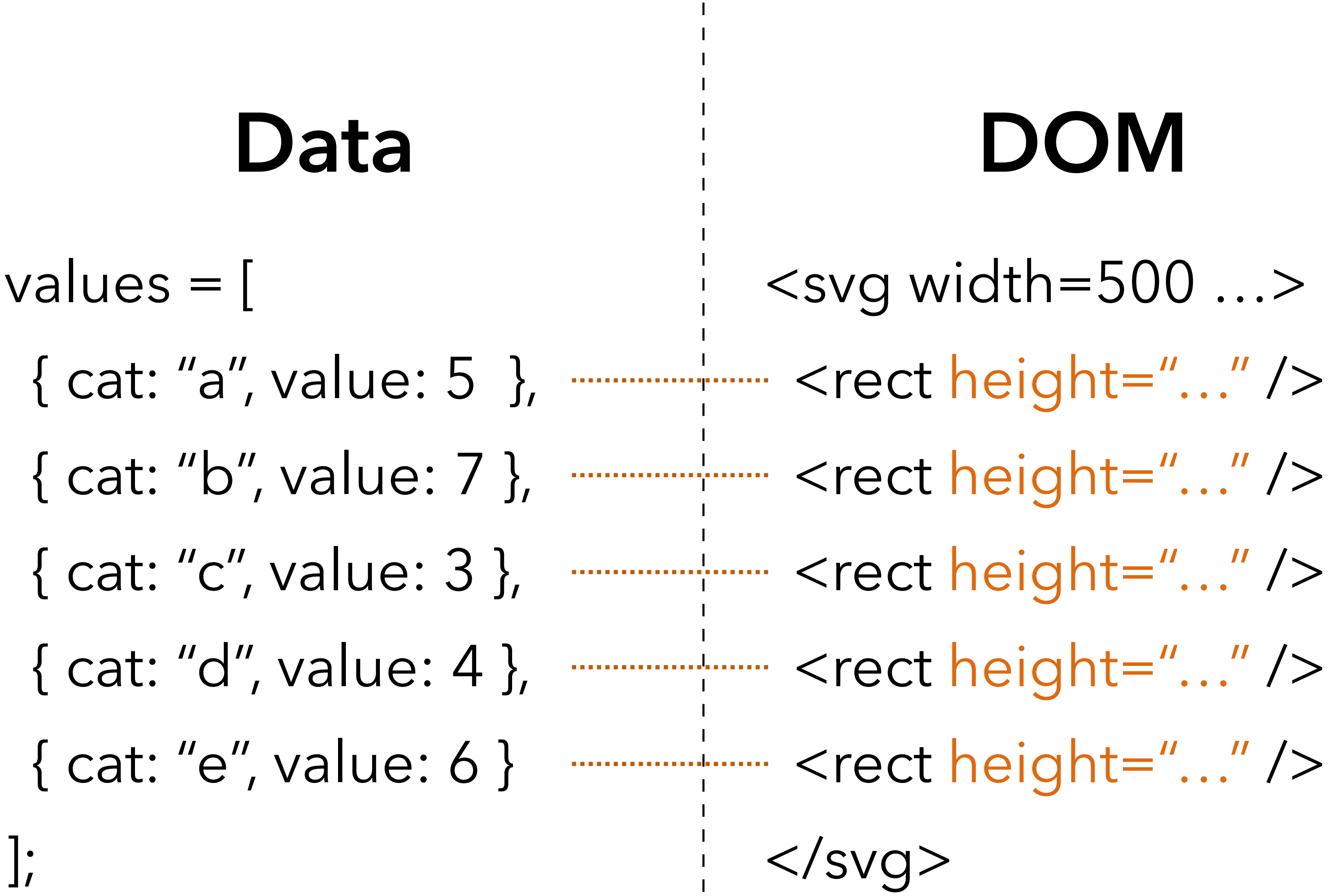
Data

DOM

```
values = [
  { cat: “a”, value: 5  },
  { cat: “b”, value: 7 },
  { cat: “c”, value: 3 },
  { cat: “d”, value: 4 },
  { cat: “e”, value: 6 }
];
```

```
<svg width=500 ...>
  <rect x=“...” />
  <rect x=“...” />
  <rect x=“...” />
  <rect x=“...” />
  <rect x=“...” />
</svg>
```

```
bars.enter().append(“rect”)
.attr(“x”, d => xscale(d.cat))
```



Data

DOM

```
values = [
  { cat: “a”, value: 5  },
  { cat: “b”, value: 7 },
  { cat: “c”, value: 3 },
  { cat: “d”, value: 4 },
  { cat: “e”, value: 6 }
];
```

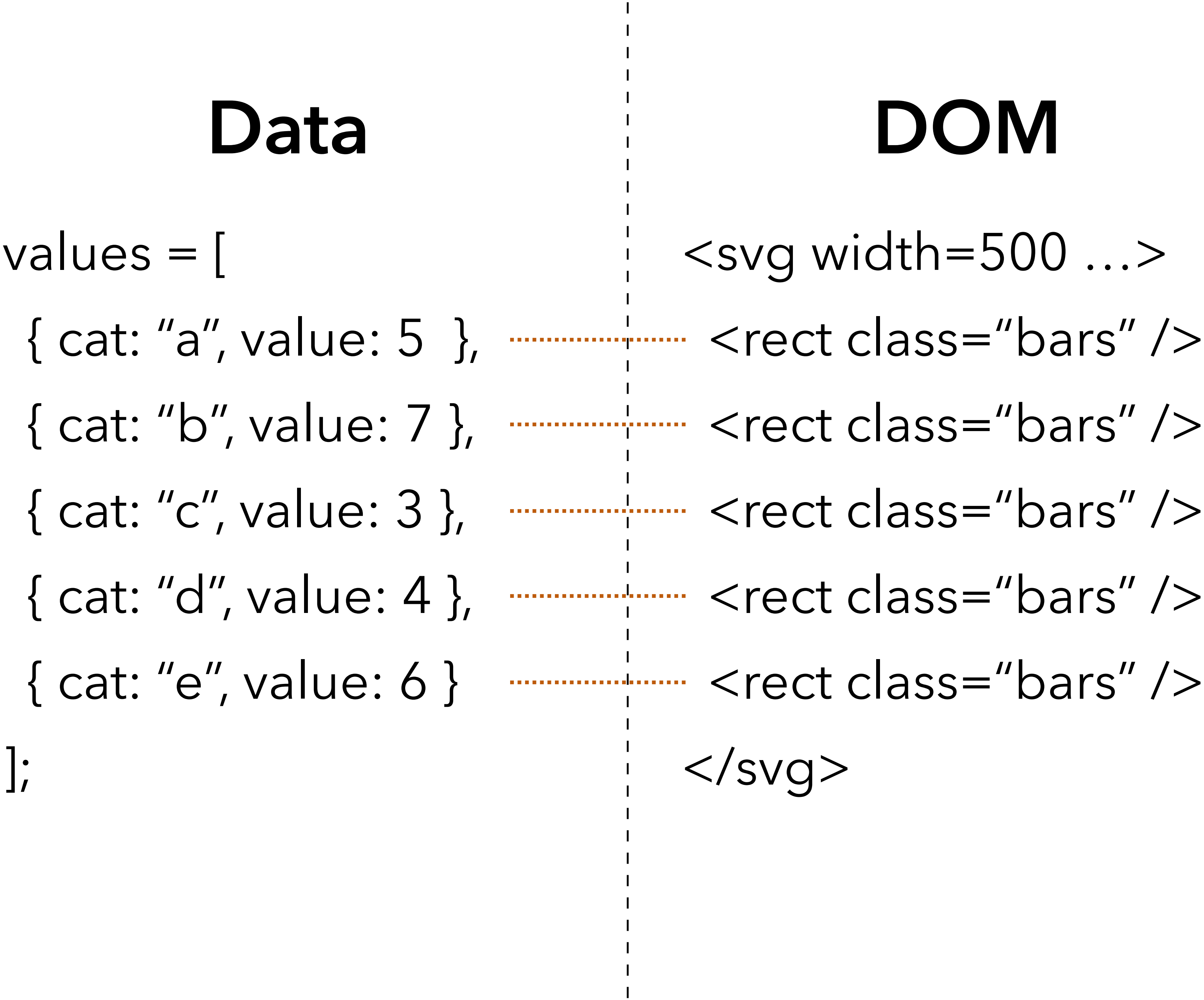
```
<svg width=500 ...>
  <rect height=“...” />
  <rect height=“...” />
  <rect height=“...” />
  <rect height=“...” />
  <rect height=“...” />
</svg>
```

```
bars.enter().append(“rect”)
.attr(“height”, d => yscale(d.value))
```

Data Binding

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

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

// Select SVG rectangles and bind them to data values.
bars = svg.selectAll(“rect.bars”).data(values);

// What if the DOM elements don’t exist yet? The enter set represents data
// values that do not yet have matching DOM elements.
bars.enter().append(“rect”).attr(“class”, “bars”);

// What if data values are removed? The exit set is a selection of existing
// DOM elements who no longer have matching data values.
bars.exit().remove();
```

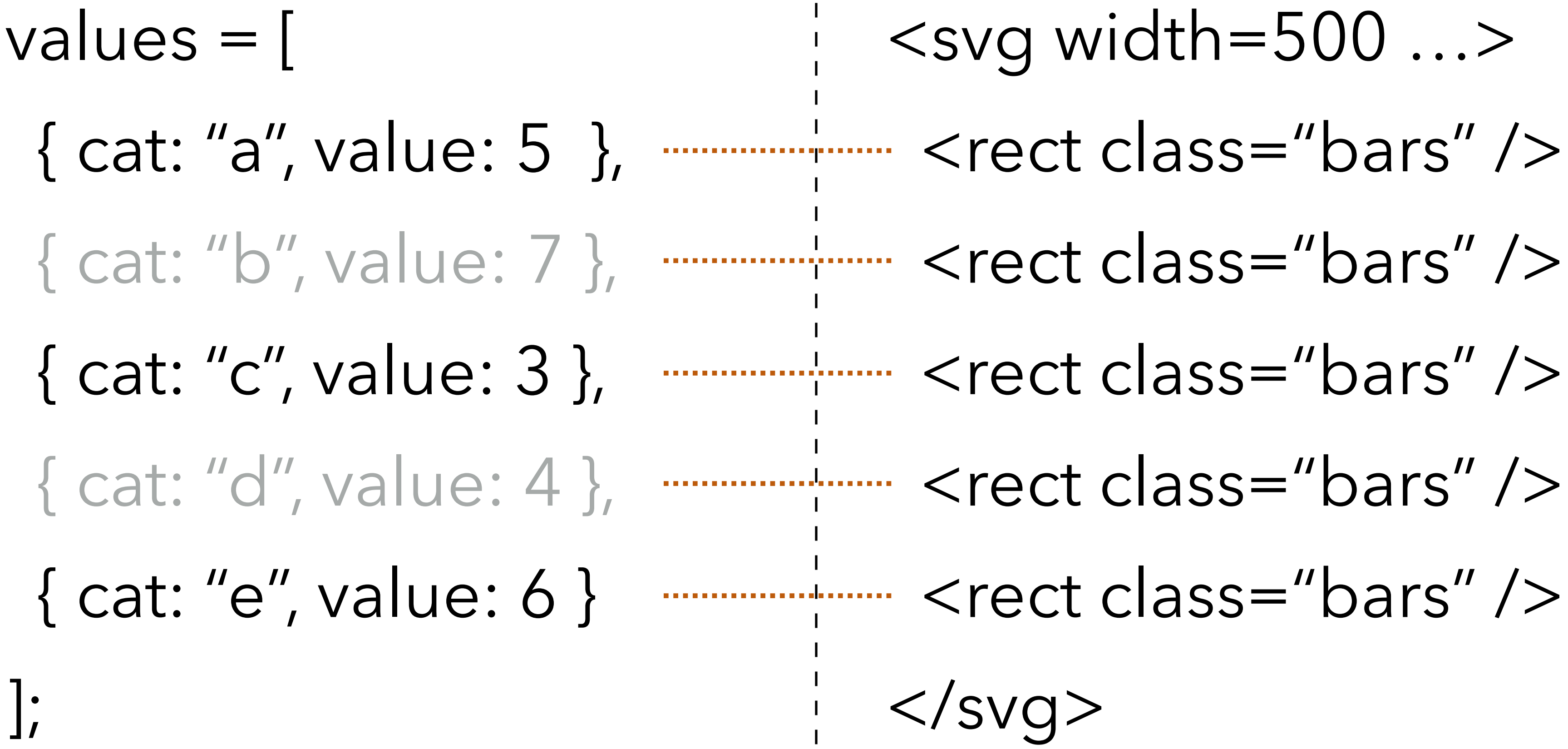


Data

DOM

```
values = [
  { cat: “a”, value: 5  },
  { cat: “b”, value: 7 },
  { cat: “c”, value: 3 },
  { cat: “d”, value: 4 },
  { cat: “e”, value: 6 }
];
```

```
<svg width=500 ...>
  <rect class=“bars” />
  <rect class=“bars” />
  <rect class=“bars” />
  <rect class=“bars” />
  <rect class=“bars” />
</svg>
```



Data

```
values = [
  { cat: “a”, value: 5  },
  { cat: “b”, value: 7 },
  { cat: “c”, value: 3 },
  { cat: “d”, value: 4 },
  { cat: “e”, value: 6 }
];
```

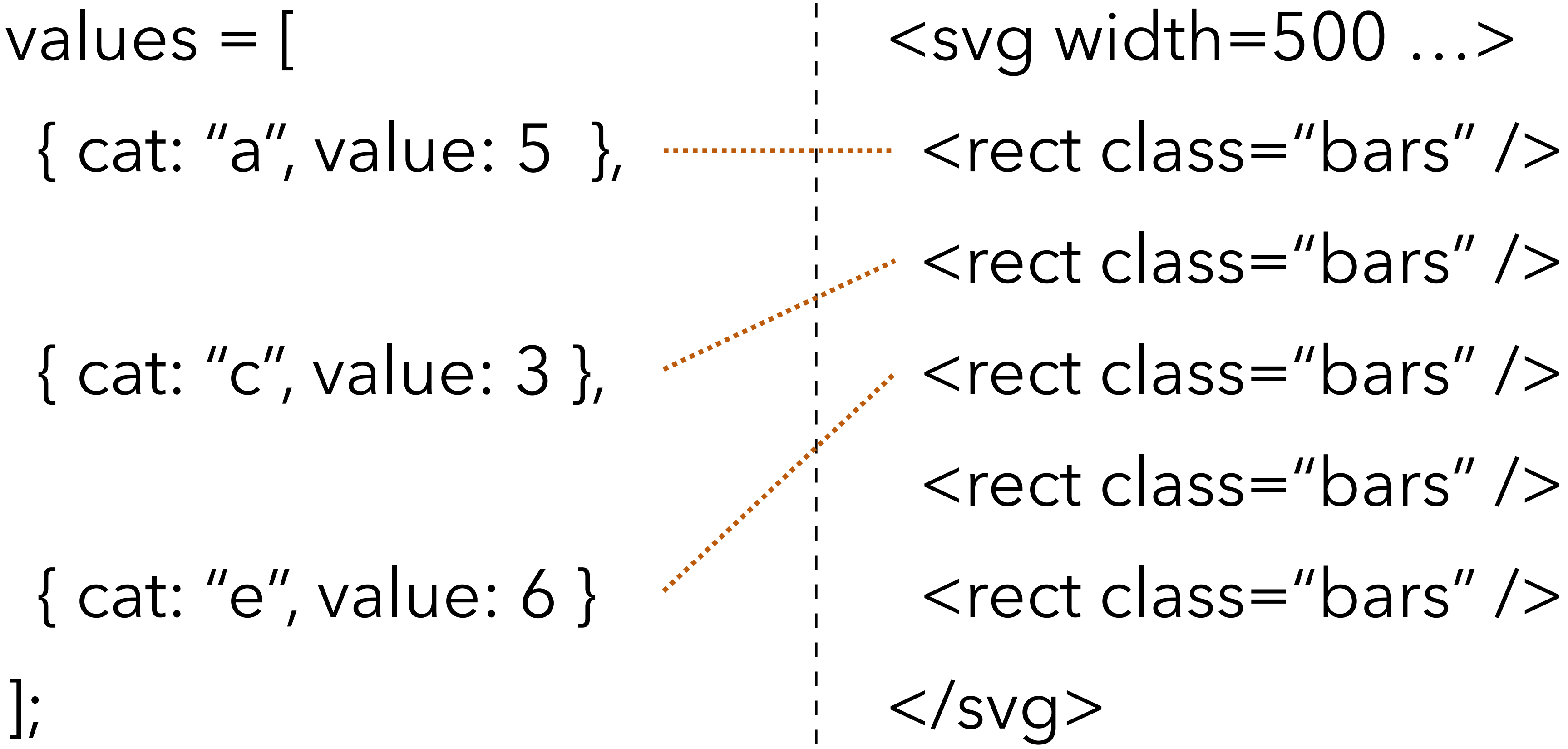
DOM

```
<svg width=500 ...>
  <rect class=“bars” />
  <rect class=“bars” />
  <rect class=“bars” />
  <rect class=“bars” />
  <rect class=“bars” />
</svg>
```

```
values.filter(d => ![‘b’, ‘d’].includes(d.cat))
```

Data

DOM



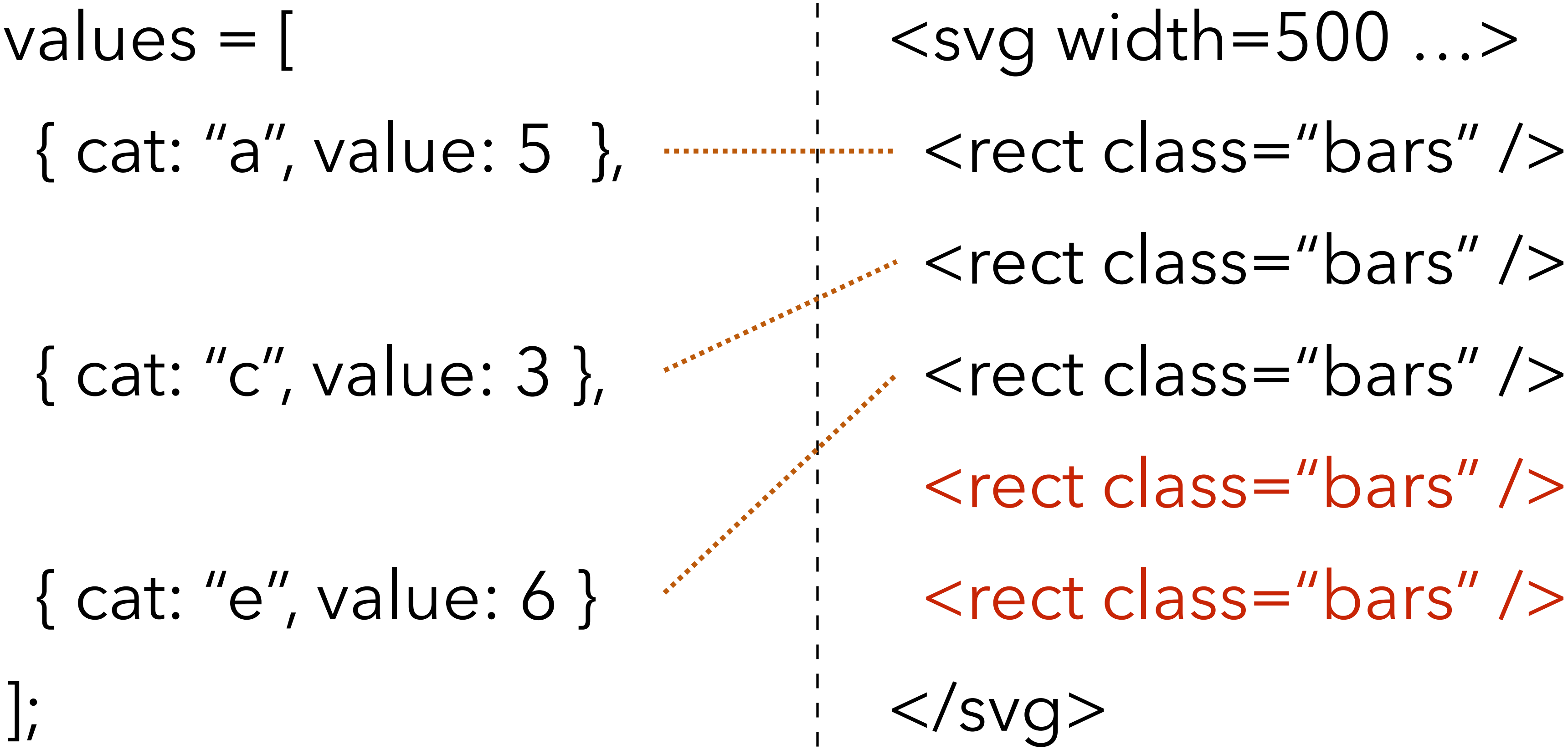
```
values = [
  { cat: “a”, value: 5  },

  { cat: “c”, value: 3 },

  { cat: “e”, value: 6 }
];
```

```
<svg width=500 ...>
  <rect class=“bars” />
  <rect class=“bars” />
  <rect class=“bars” />
  <rect class=“bars” />
  <rect class=“bars” />
</svg>
```

```
bars = svg.selectAll(“rect.bars”).data(values)
```

Data

DOM



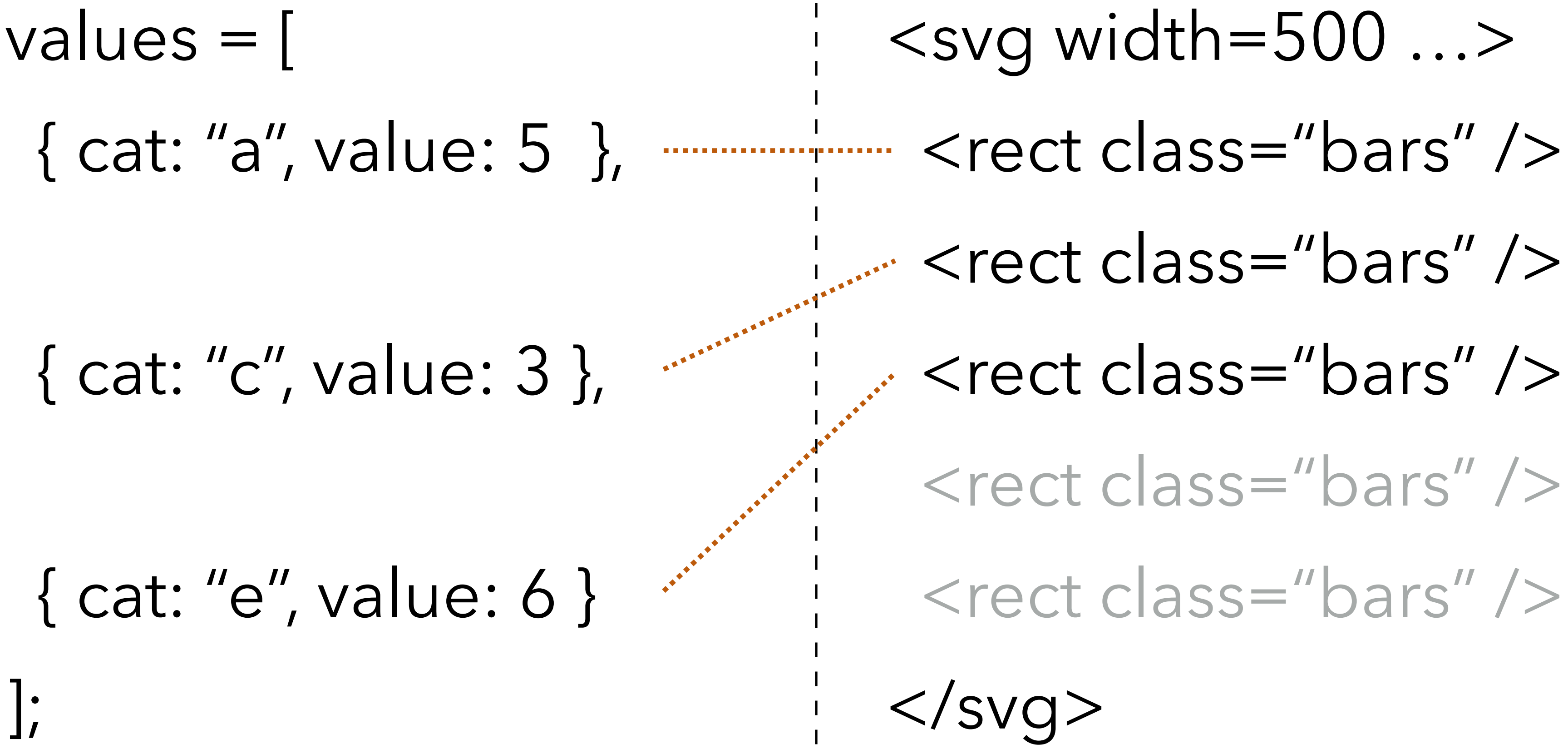
```
values = [
  { cat: “a”, value: 5  },


  { cat: “c”, value: 3 },


  { cat: “e”, value: 6 }
];
```

```
<svg width=500 ...>
  <rect class=“bars” />
  <rect class=“bars” />
  <rect class=“bars” />
  <rect class=“bars” />
  <rect class=“bars” />
</svg>
```

bars.exit()

Data

DOM



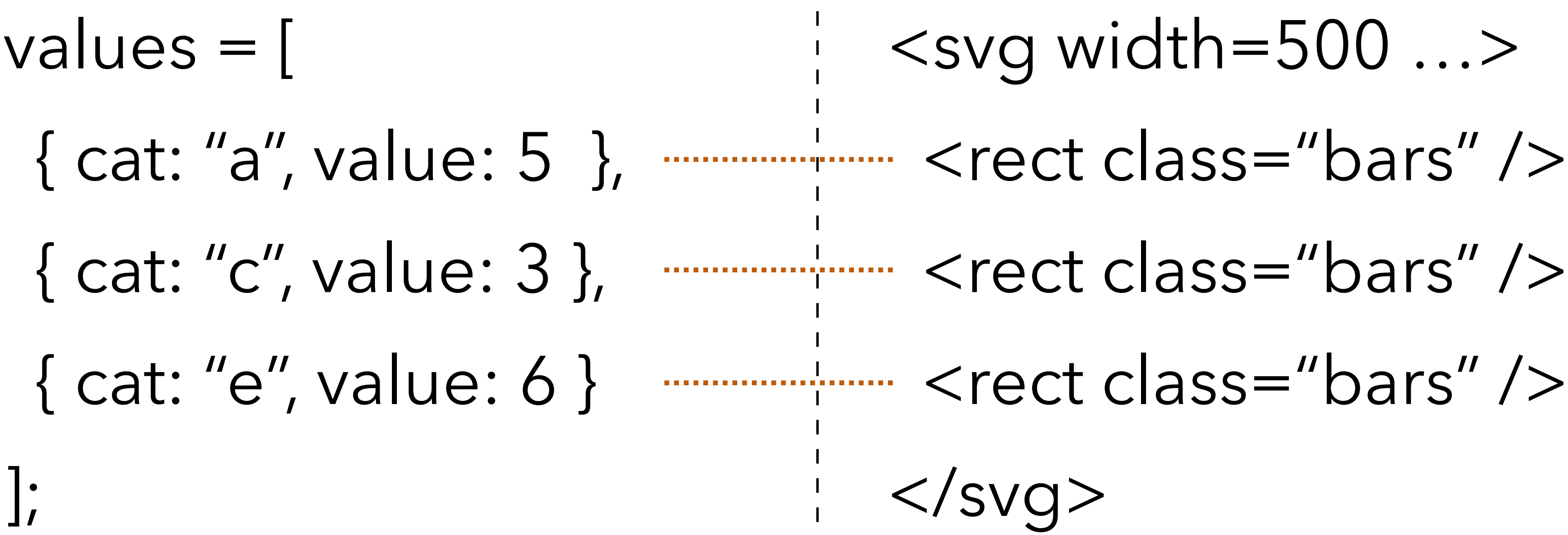
```
values = [
  { cat: “a”, value: 5  },
  { cat: “c”, value: 3 },
  { cat: “e”, value: 6 }
];
```

```
<svg width=500 ...>
  <rect class=“bars” />
  <rect class=“bars” />
  <rect class=“bars” />
  <rect class=“bars” />
  <rect class=“bars” />
</svg>
```

```
bars.exit().remove()
```

Data

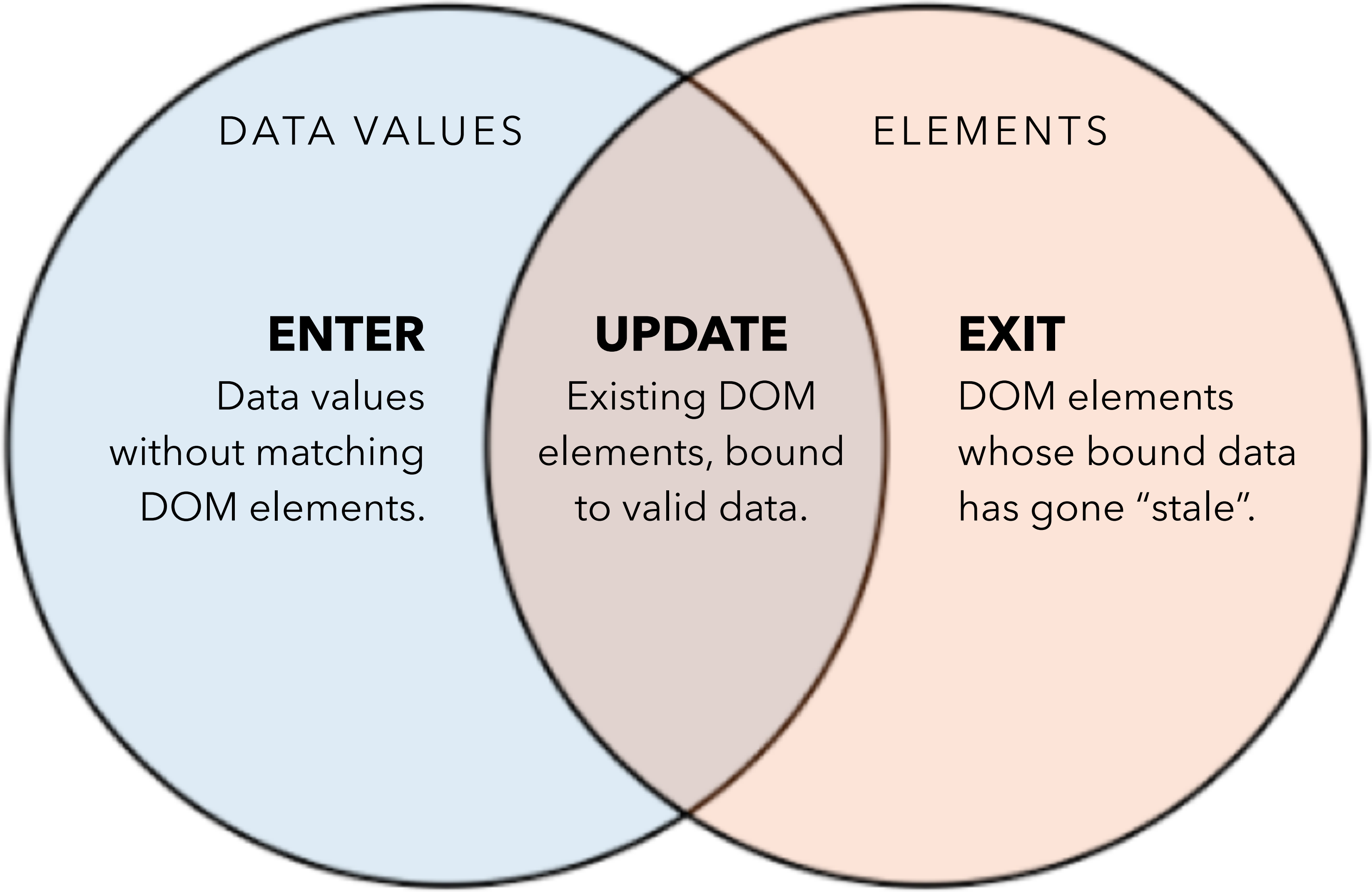
DOM



```
values = [
  { cat: “a”, value: 5  },
  { cat: “c”, value: 3 },
  { cat: “e”, value: 6 }
];
```

```
<svg width=500 ...>
 <rect class=“bars” />
 <rect class=“bars” />
 <rect class=“bars” />
</svg>
```

The Data Join



DATA VALUES

ELEMENTS

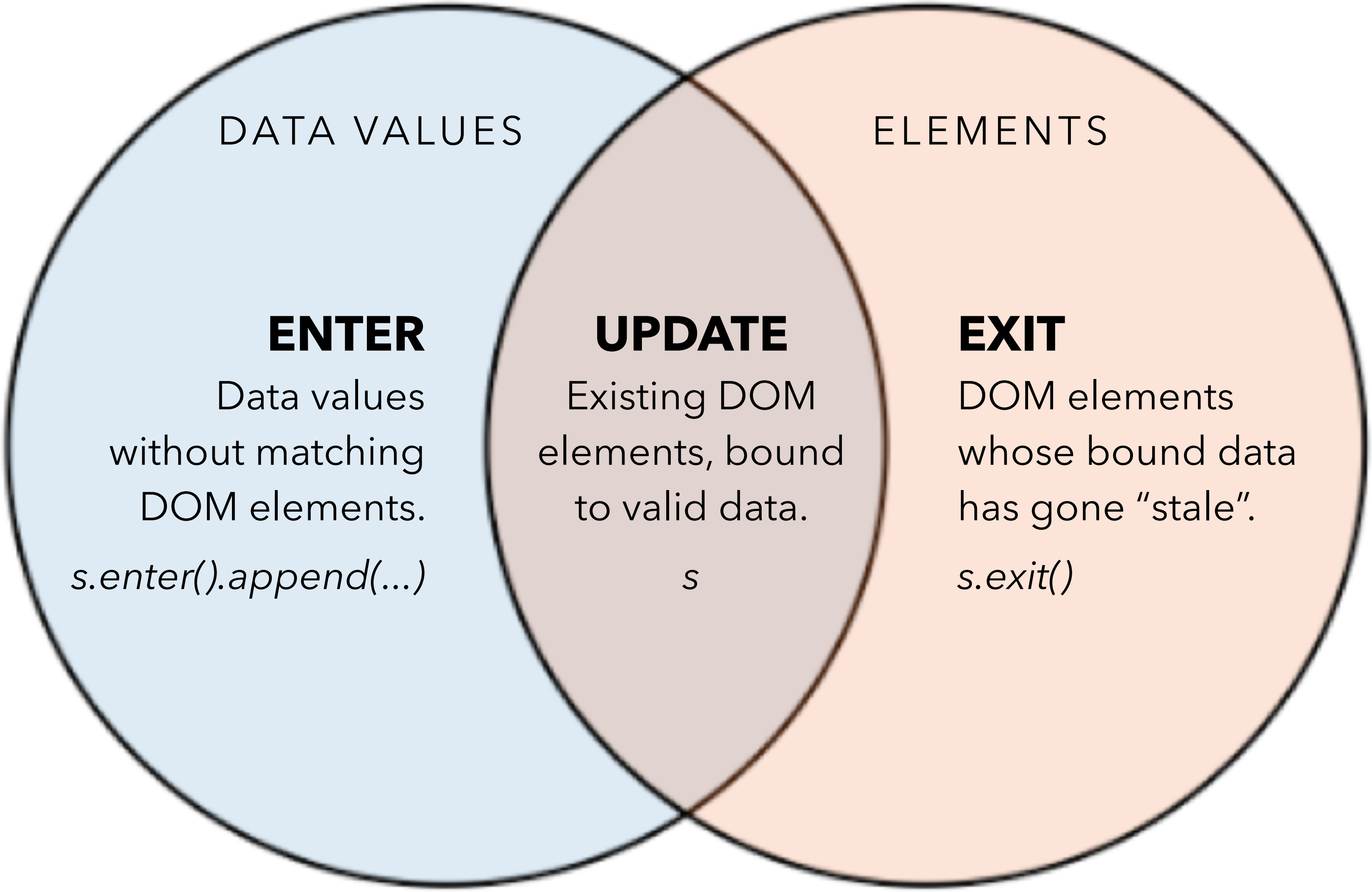
ENTER
Data values without matching DOM elements.

UPDATE
Existing DOM elements, bound to valid data.

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

The Data Join

var s = d3.selectAll(...).data(...)



DATA VALUES

ELEMENTS

ENTER

Data values without matching DOM elements.

s.enter().append(...)

UPDATE

Existing DOM elements, bound to valid data.

s

EXIT

DOM elements whose bound data has gone “stale”.

s.exit()

Data Binding

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

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

// Select SVG rectangles and bind them to data values.
bars = svg.selectAll(“rect.bars”).data(values)
    .join(
        enter => enter.append("rect"), // create new
        update => update,              // update current
        exit => exit.remove()          // remove outdated
    )
```

D3 Modules

Data Parsing / Formatting (JSON, CSV, …)

Shape Helpers (arcs, curves, areas, symbols, …)

Scale Transforms (linear, log, ordinal, …)

Color Spaces (RGB, HSL, LAB, …)

Animated Transitions (tweening, easing, …)

Geographic Mapping (projections, clipping, …)

Layout Algorithms (stack, pie, force, trees, …)

Interactive Behaviors (brush, zoom, drag, …)

Many of these correspond to future lecture topics!

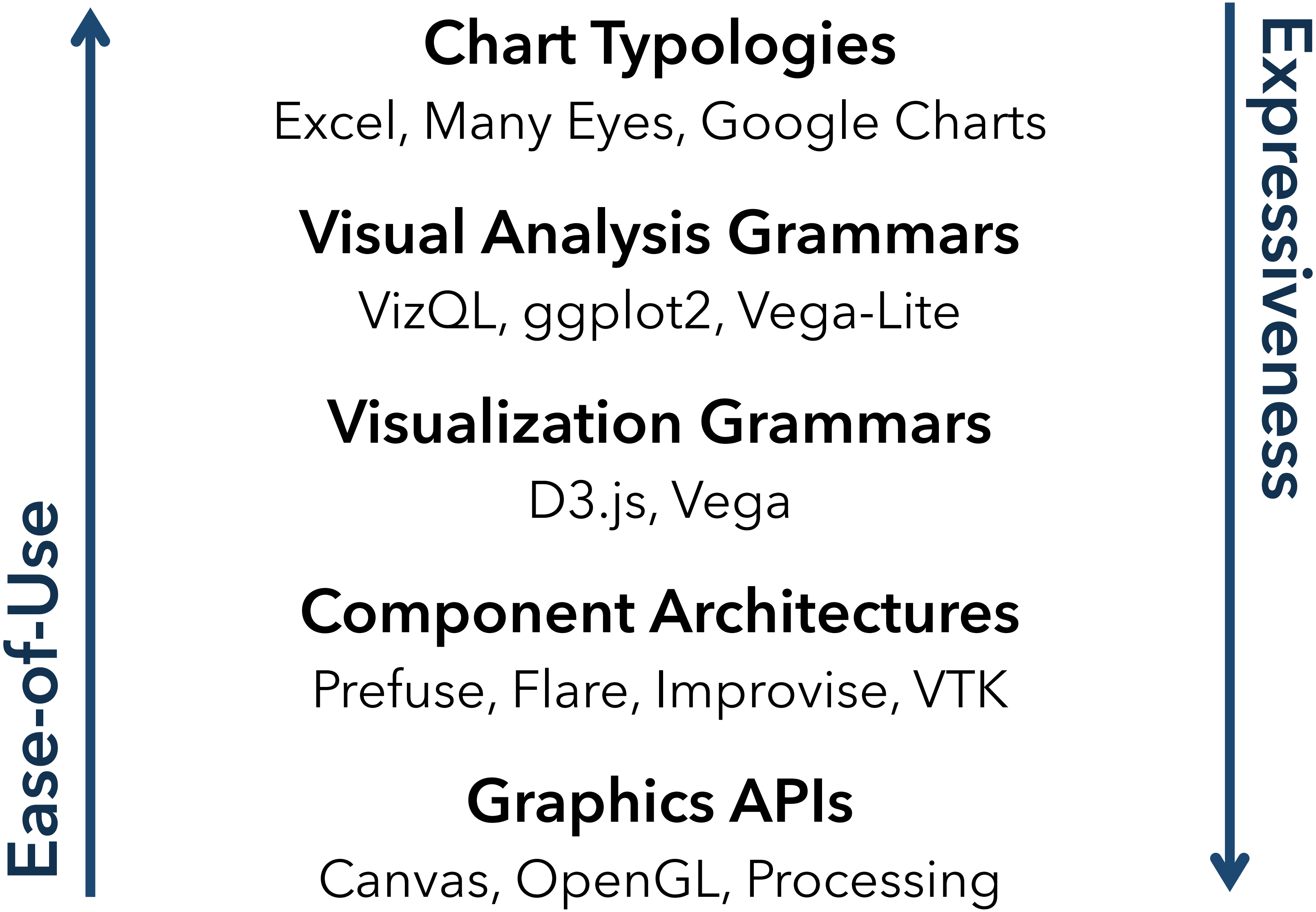


Chart Typologies

Excel, Many Eyes, Google Charts

Visual Analysis Grammars

VizQL, ggplot2, Vega-Lite

Visualization Grammars

D3.js, Vega

Component Architectures

Prefuse, Flare, Improvise, VTK

Graphics APIs

Canvas, OpenGL, Processing

Administrivia

A2 Peer Reviews

You have been assigned two peer A2 submissions to review. For each:

- Try to determine which is earnest and which is deceptive
- Share a rationale for how you made this determination
- Share feedback using the “I Like / I Wish / What If” rubric

Assigned reviews will be posted on the A2 Peer Review page on Canvas, along with a link to a Google Form. You should submit two forms: one for each A2 peer review.

Due by **Tue 10/22 11:59pm**.

I Like… / I Wish… / What If?

I LIKE…
Praise for design ideas and/or well-executed implementation details. *Example: "I like the navigation through time via the slider; the patterns observed as one moves forward are compelling!”*

I WISH…
Constructive statements on how the design might be improved or further refined. *Example: "I wish moving the slider caused the visualization to update immediately, rather than the current lag."*

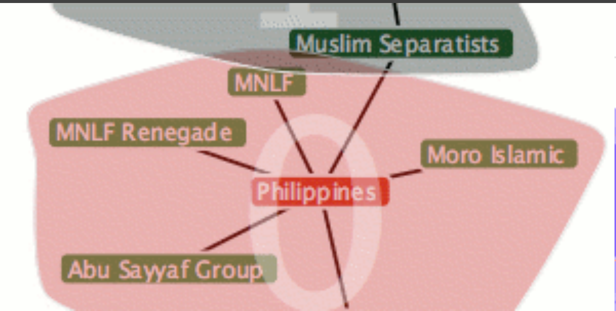
WHAT IF?
Suggest alternative design directions, or even wacky half-baked ideas. *Example: "What if we got rid of the slider and enabled direct manipulation navigation by dragging data points directly?"*

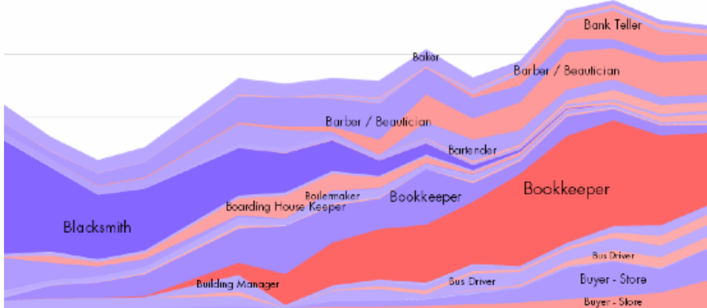
A3: Interactive Prototype

Create an interactive visualization. Choose a driving question for a dataset and develop an appropriate visualization + interaction techniques, then deploy your visualization on the web.

Due by *11:59pm* on **Monday, November 4**.

Work in project teams of 3-4 people.





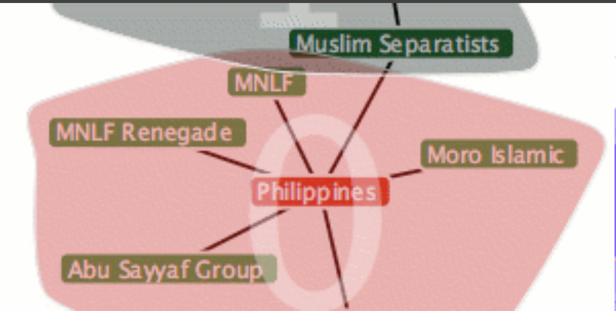
Form A3 + Final Project Team

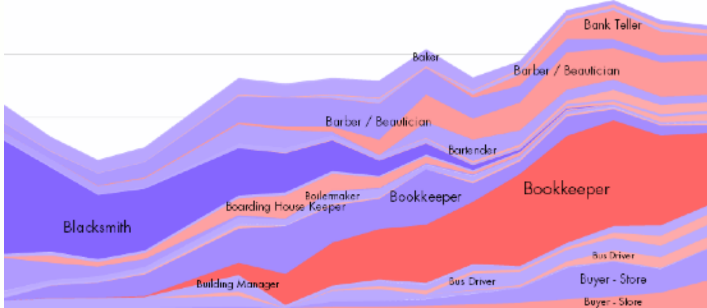
Form a **team of 3-4** for A3 and the Final Project.

Submit signup form by **Wed 10/23, 11:59pm**.

If you do not have team mates, post on Ed about your interests/skills/project ideas!

We will send out a reminder early next week.



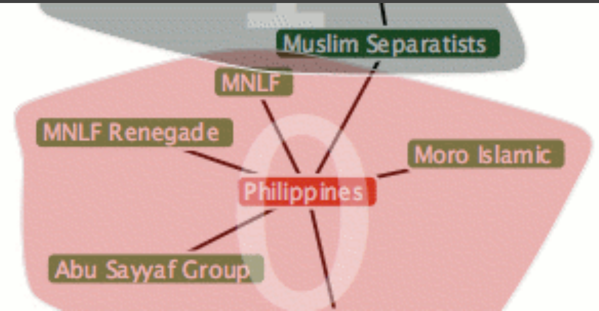


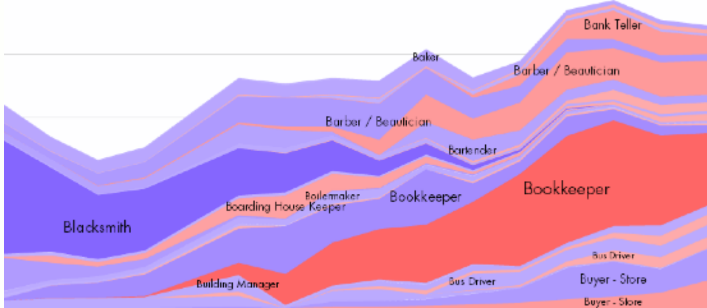
Requirements

Interactive. You must implement interaction methods! However, this is not only selection / filtering / tooltips. Also consider annotations or other narrative features to draw attention and provide additional context

Web-based. D3/Vega-Lite are encouraged, but not required. Deploy to web using GitHub pages.

Write-up. Provide design rationale.



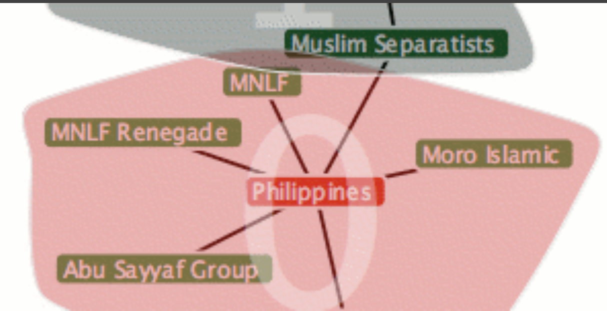


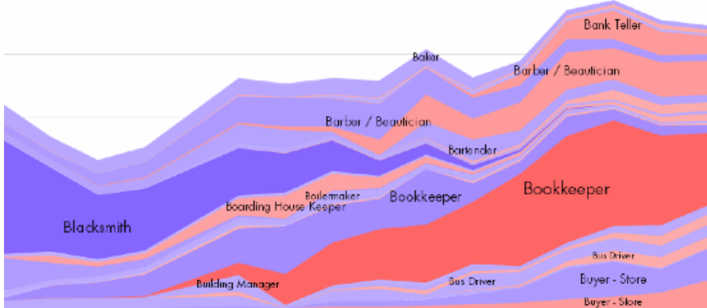
Interactive Prototype Tips

Start now. It will take longer than you think.

Keep it simple. Choose a *minimal* set of interactions that enables users to explore and generate interesting insights. Do not feel obligated to convey *everything* about the data: focus on a compelling subset.

Promote engagement. How do your chosen interactions reveal interesting observations?





D3 Tutorial - In Class Thu Oct 24

D3.js Deep Dive led by Madeleine and Lisa

Be sure to read the D3, Part 1 notebook ahead of time. We’ll work through Part 2 in class.

Bring your laptops and follow along in real-time.

Web Publishing Tutorial

On Zoom, led by Heer and Tae - Monday 10/28

Gain skills publishing projects to the web:

- Publish sites using GitLab pages

- Export Vega-Lite or Observable cells to HTML

- Learn dashboard publishing tools

A Visualization Tool Stack

Chart Typologies

Excel, Many Eyes, Google Charts

Visual Analysis Grammars

VizQL, ggplot2, Vega-Lite

Visualization Grammars

D3.js, Vega

Component Architectures

Prefuse, Flare, Improvise, VTK

Graphics APIs

Canvas, OpenGL, Processing

Chart Typologies

Excel, Many Eyes, Google Charts

Charting
Tools

Visual Analysis Grammars

VizQL, ggplot2, Vega-Lite

Visualization Grammars

D3.js, Vega

Declarative
Languages

Component Architectures

Prefuse, Flare, Improvise, VTK

Graphics APIs

Canvas, OpenGL, Processing

Programming
Toolkits

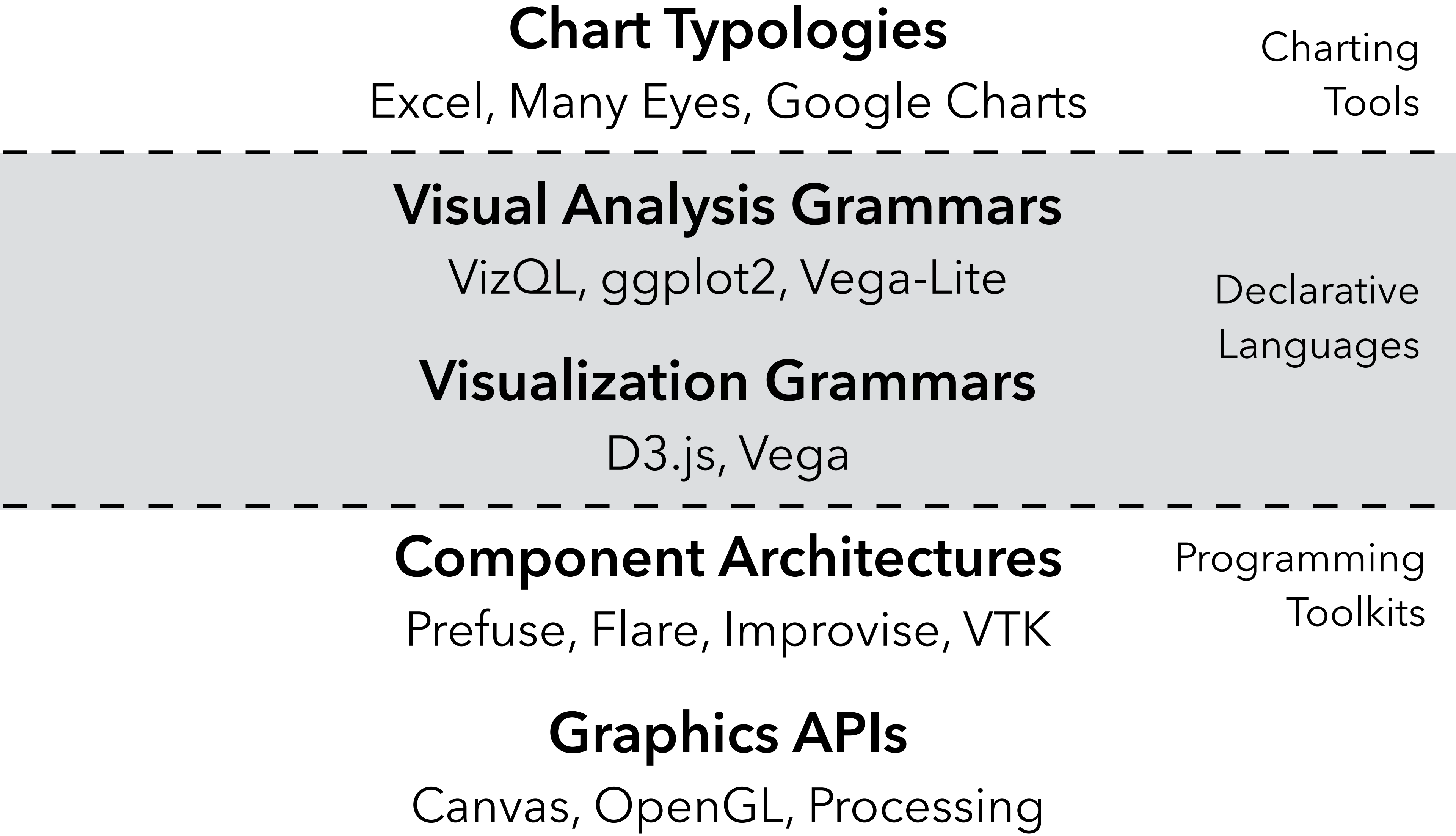


Chart Typologies

Excel, Many Eyes, Google Charts

Charting
Tools

Visual Analysis Grammars

VizQL, ggplot2, Vega-Lite

Visualization Grammars

D3.js, Vega

Declarative
Languages

Component Architectures

Prefuse, Flare, Improvise, VTK

Graphics APIs

Canvas, OpenGL, Processing

Programming
Toolkits

What is a Declarative Language?

Programming by describing *what*, not *how*

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

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

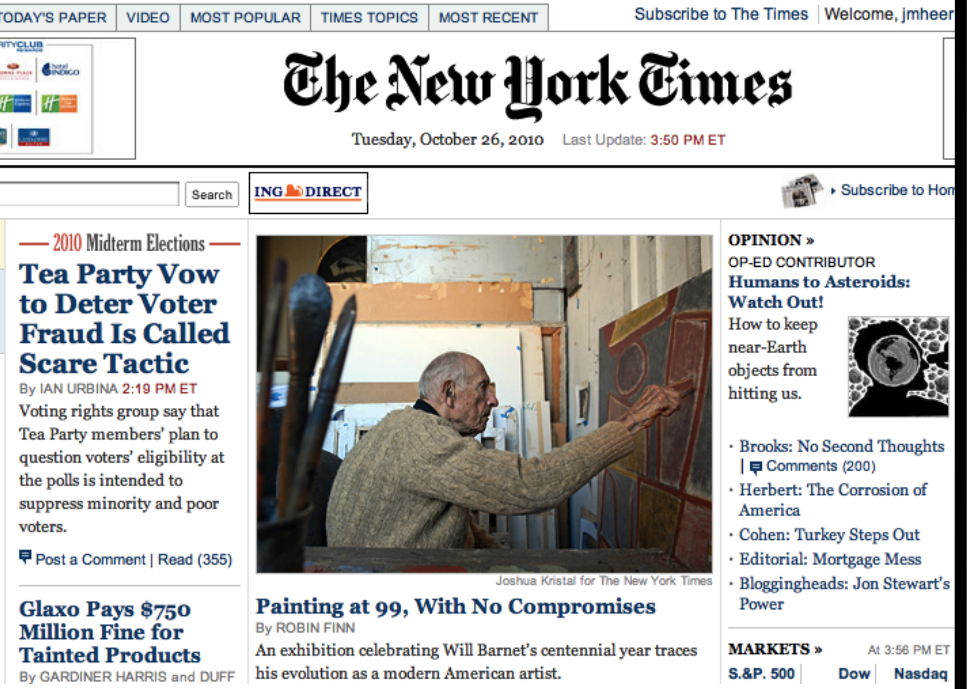
What is a Declarative Language?

Programming by describing *what*, not *how*

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

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

```
d3.selectAll("rect")
  .data(my_data)
 .join("rect")
  .attr("x", d => xscale(d.foo))
  .attr("y", d => yscale(d.bar))
```



TODAY'S PAPER | VIDEO | MOST POPULAR | TIMES TOPICS | MOST RECENT

Subscribe to The Times | Welcome, jmheer

The New York Times

Tuesday, October 26, 2010 Last Update: 3:50 PM ET

Search

ING DIRECT

Subscribe to Hom

— 2010 Midterm Elections —

Tea Party Vow to Deter Voter Fraud Is Called Scare Tactic

By IAN URBINA 2:19 PM ET

Voting rights group say that Tea Party members' plan to question voters' eligibility at the polls is intended to suppress minority and poor voters.

Post a Comment | Read (355)

Glaxo Pays $750 Million Fine for Tainted Products

By GARDINER HARRIS and DUFF

Joshua Kristal for The New York Times

Painting at 99, With No Compromises

By ROBIN FINN

An exhibition celebrating Will Barnet's centennial year traces his evolution as a modern American artist.

OPINION »

OP-ED CONTRIBUTOR

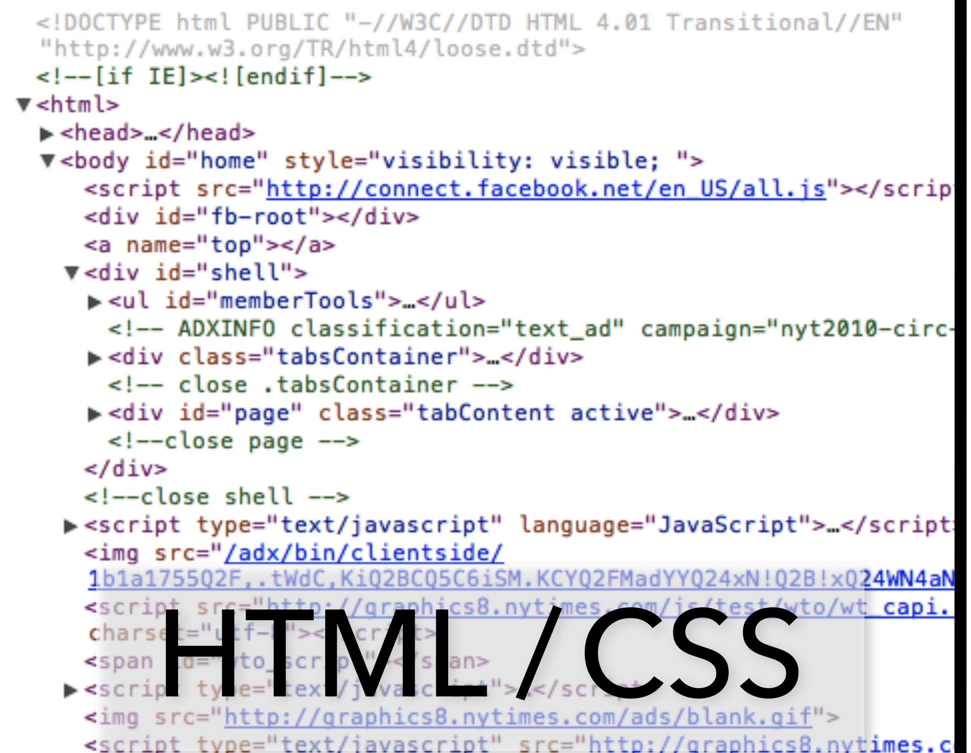
Humans to Asteroids: Watch Out!

How to keep near-Earth objects from hitting us.

- Brooks: No Second Thoughts | Comments (200)
- Herbert: The Corrosion of America
- Cohen: Turkey Steps Out
- Editorial: Mortgage Mess
- Bloggingheads: Jon Stewart's Power

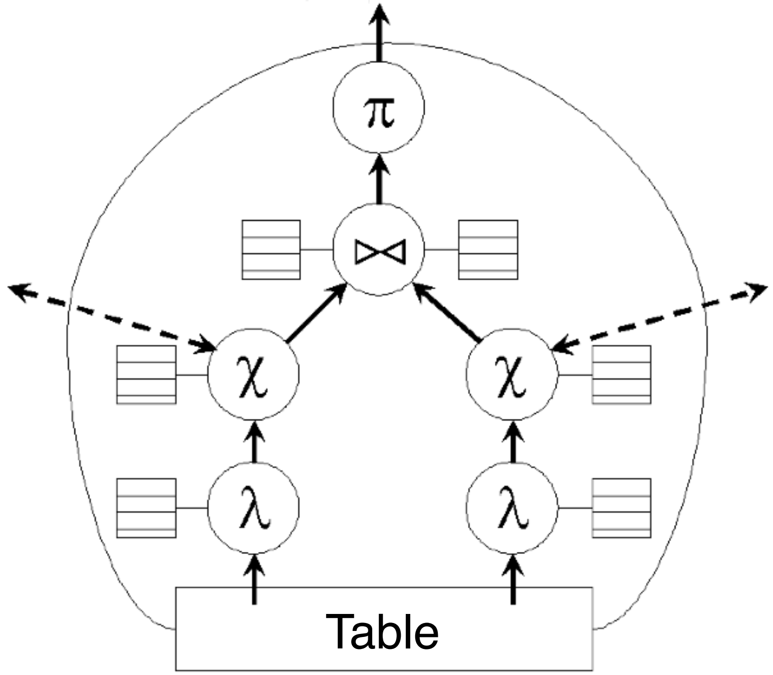
MARKETS » At 3:56 PM ET

S.&P. 500 | Dow | Nasdaq



```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<!--[if IE]><![endif]-->
▼<html>
 ▶<head>…</head>
 ▼<body id="home" style="visibility: visible; ">
   <script src="http://connect.facebook.net/en_US/all.js"></scrip
   <div id="fb-root"></div>
   <a name="top"></a>
  ▼<div id="shell">
   ▶<ul id="memberTools">…</ul>
     <!-- ADXINFO classification="text_ad" campaign="nyt2010-circ-
   ▶<div class="tabsContainer">…</div>
     <!-- close .tabsContainer -->
   ▶<div id="page" class="tabContent active">…</div>
     <!--close page -->
   </div>
   <!--close shell -->
  ▶<script type="text/javascript" language="JavaScript">…</script
   <img src="/adx/bin/clientside/
   1b1a1755Q2F,.tWdC,KiQ2BCQ5C6iSM.KCYQ2FMadYYQ24xN!Q2B!xQ24WN4aN
   <script src="http://graphics8.nytimes.com/js/test/wto/wt_capi.
   charset="utf-8"></script>
   <span id="wto_script"></span>
  ▶<script type="text/javascript">…</script>
   <img src="http://graphics8.nytimes.com/ads/blank.gif">
   <script type="text/javascript" src="http://graphics8.nytimes.c
```

HTML / CSS



```
SELECT customer_id, customer_name,
    COUNT(order_id) as total
FROM customers
INNER JOIN orders ON
    customers.customer_id
    = orders.customer_id
GROUP BY customer_id, customer_nam
HAVING COUNT(order_id) > 5
ORDER BY COUNT(order_id) DESC
```

SQL

Why Declarative Languages?

Faster iteration, less code, larger user base?

Better visualization. *Smart defaults.*

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

Performance. *Optimization, scalability.*

Portability. *Multiple devices, renderers, inputs.*

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

Chart Typologies

Excel, Many Eyes, Google Charts

Charting
Tools

Visual Analysis Grammars

VizQL, ggplot2, ***Vega-Lite***

Visualization Grammars

D3.js, ***Vega***

Declarative
Languages

Component Architectures

Prefuse, Flare, Improvise, VTK

Programming
Toolkits

Graphics APIs

Processing, OpenGL, Java2D

Chart Typologies

Excel, Many Eyes, Google Charts

?! Charting Tools

Visual Analysis Grammars

VizQL, ggplot2, ***Vega-Lite***

Visualization Grammars

D3.js, ***Vega***

Declarative Languages

Component Architectures

Prefuse, Flare, Improvise, VTK

Programming Toolkits

Graphics APIs

Processing, OpenGL, Java2D

Visual Analysis Grammars

VizQL, ggplot2, ***Vega-Lite***

Visualization Grammars

D3.js, ***Vega***

Declarative Languages

Component Architectures

Prefuse, Flare, Improvise, VTK

Graphics APIs

Processing, OpenGL, Java2D

Programming Toolkits

Interactive Data Exploration

Tableau, ***Lyra, Voyager***

Graphical Interfaces

Visual Analysis Grammars

VizQL, ggplot2, ***Vega-Lite***

Visualization Grammars

D3.js, ***Vega***

Declarative Languages

Component Architectures

Prefuse, Flare, Improvise, VTK

Programming Toolkits

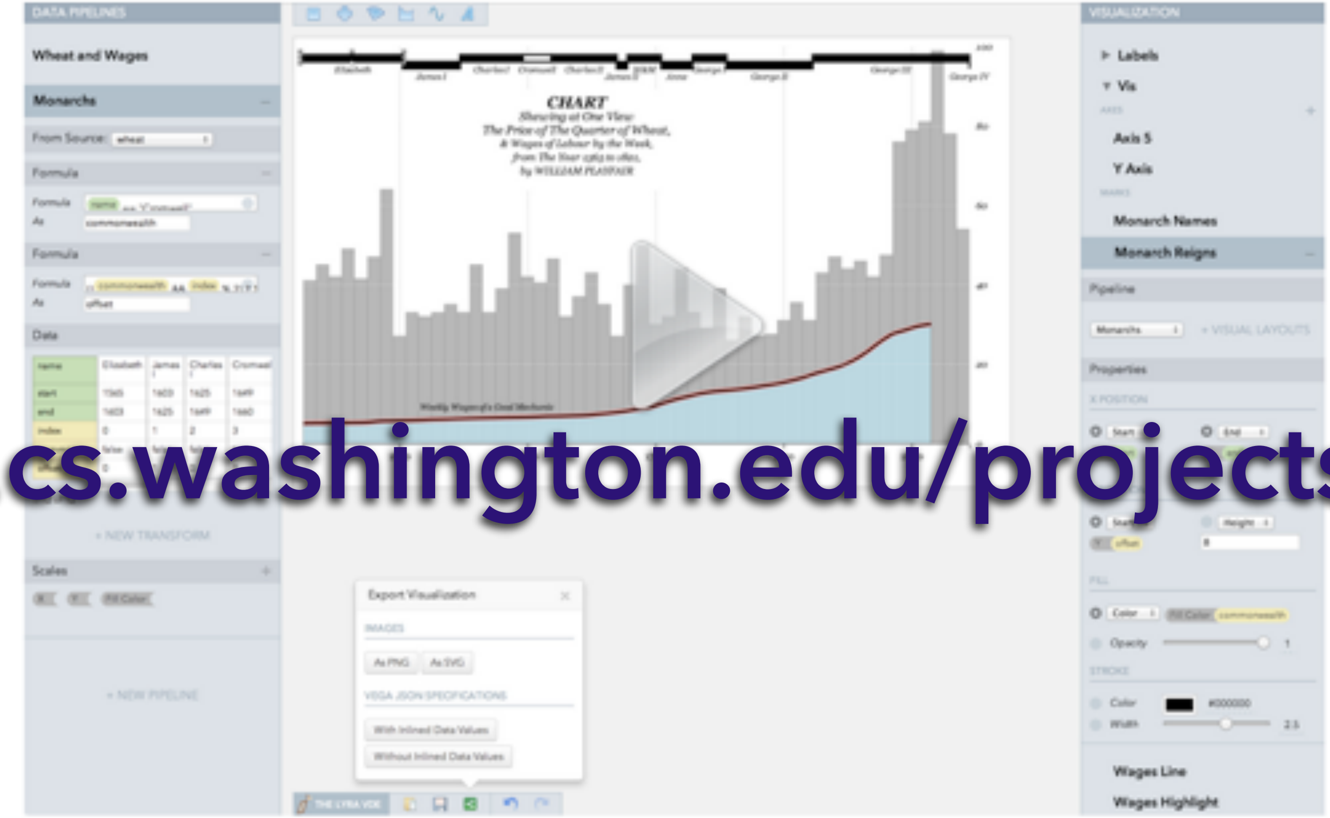
Graphics APIs

Processing, OpenGL, Java2D

UW Interactive Data Lab PROJECTS

The Lyra Visualization Design Environment (VDE) alpha

Arvind Satyanarayan, Kanit "Ham" Wongsuphasawat, Jeffrey Heer

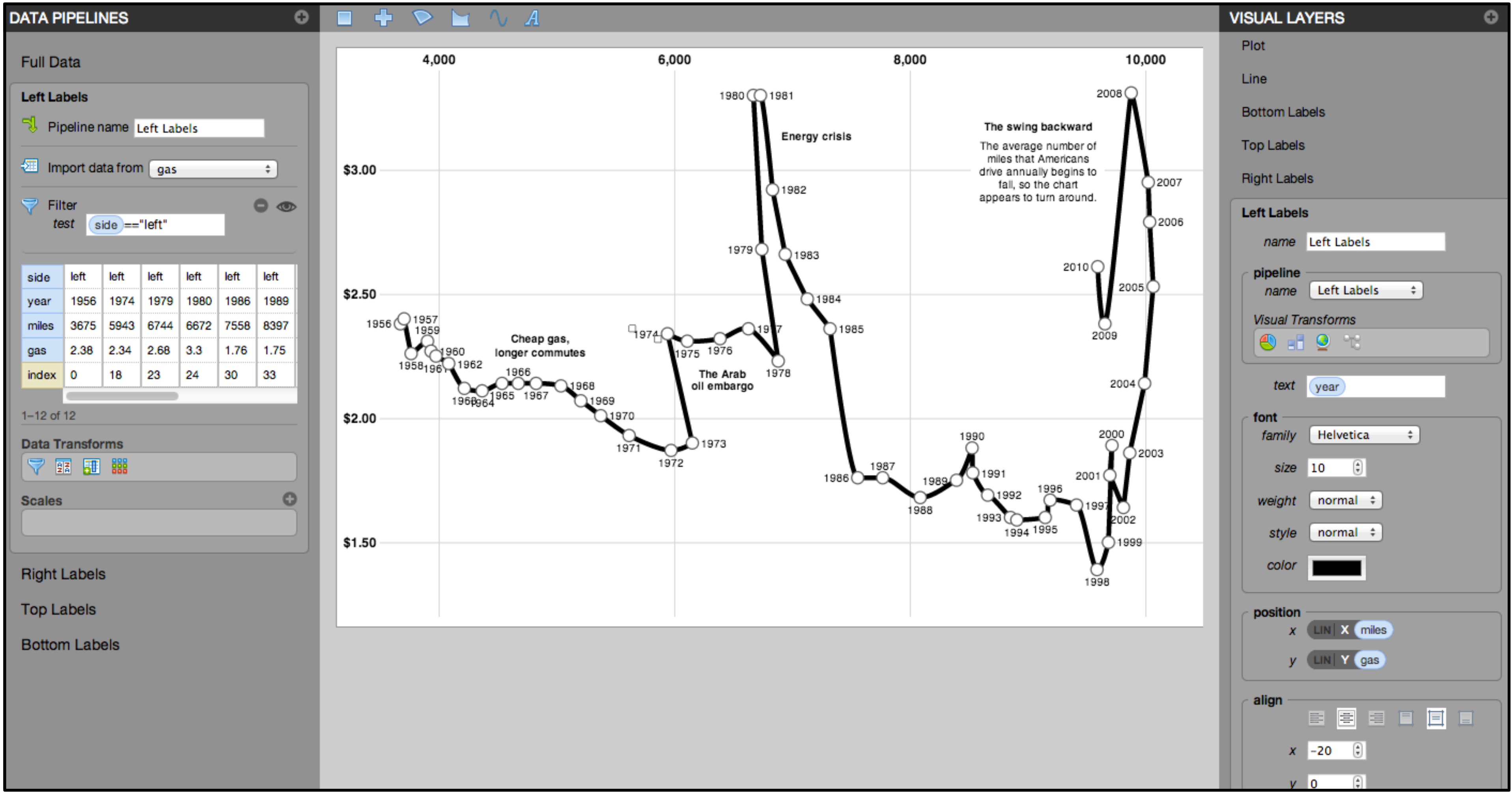


William Playfair's classic chart comparing the price of wheat and wages in England recreated in the Lyra VDE.

idl.cs.washington.edu/projects/lyra

See *also*: Charticulator, Data Illustrator

Lyra A Visualization Design Environment



DATA PIPELINES

Full Data

Left Labels

Pipeline name: Left Labels

Import data from: gas

Filter
test side=="left"

side	left	left	left	left	left	left
year	1956	1974	1979	1980	1986	1989
miles	3675	5943	6744	6672	7558	8397
gas	2.38	2.34	2.68	3.3	1.76	1.75
index	0	18	23	24	30	33

1–12 of 12

Data Transforms

Scales

Right Labels

Top Labels

Bottom Labels

VISUAL LAYERS

Plot

Line

Bottom Labels

Top Labels

Right Labels

Left Labels

name Left Labels

pipeline
name Left Labels

Visual Transforms

text year

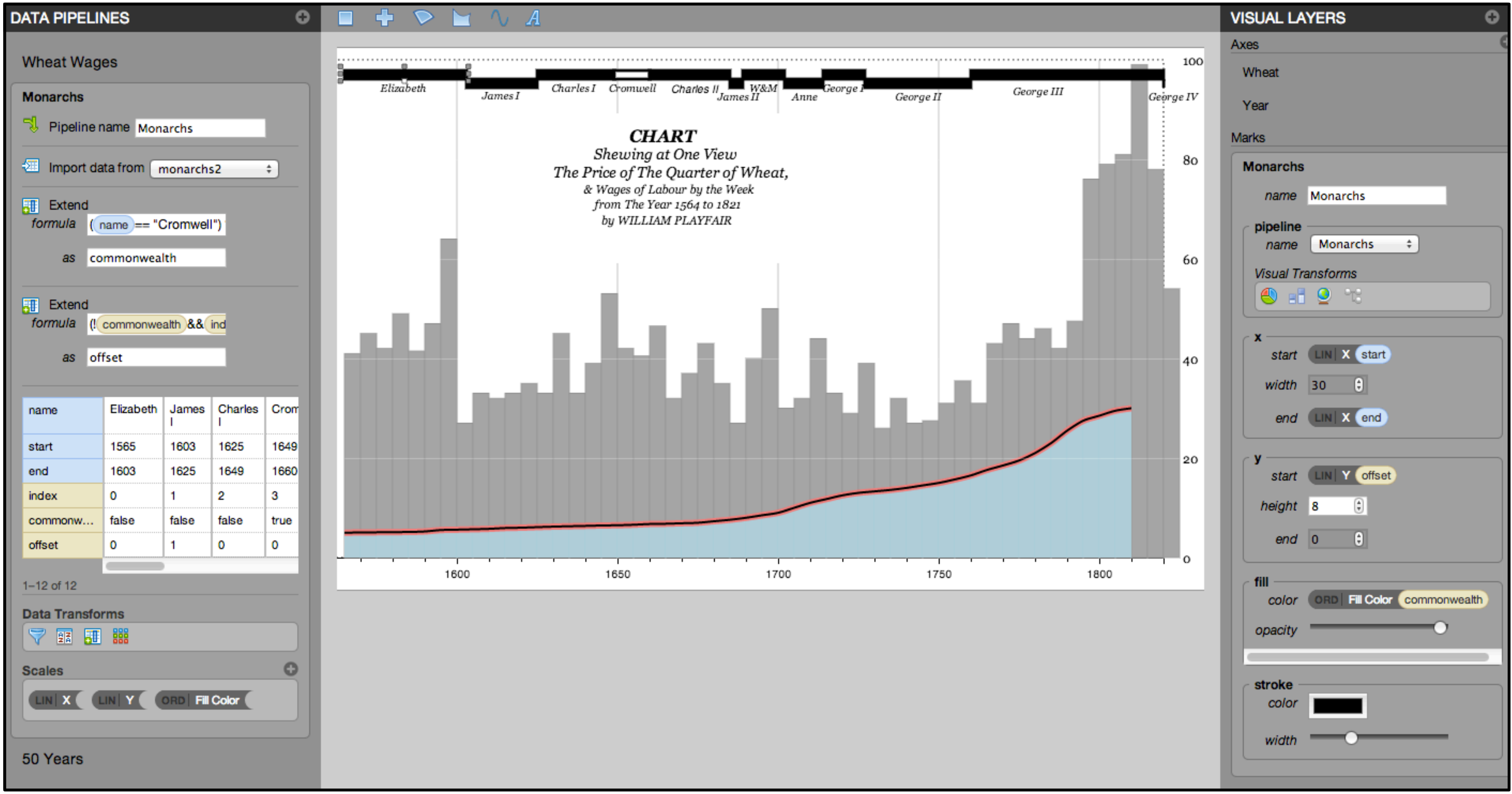
font
family Helvetica
size 10
weight normal
style normal
color ■

position
x LIN X miles
y LIN Y gas

align
x -20
y 0

Driving Shifts into Reverse by Hannah Fairfield, NYTimes

Lyra A Visualization Design Environment



DATA PIPELINES

Wheat Wages

Monarchs

Pipeline name: Monarchs

Import data from: monarchs2

Extend
formula (name == "Cromwell")
as commonwealth

Extend
formula (! commonwealth && ind
as offset

name	Elizabeth	James I	Charles I	Crom
start	1565	1603	1625	1649
end	1603	1625	1649	1660
index	0	1	2	3
commonw...	false	false	false	true
offset	0	1	0	0

1–12 of 12

Data Transforms

Scales

LIN X · LIN Y · ORD Fill Color

50 Years

VISUAL LAYERS

Axes

Wheat

Year

Marks

Monarchs

name Monarchs

pipeline
name Monarchs

Visual Transforms

x
start LIN X start
width 30
end LIN X end

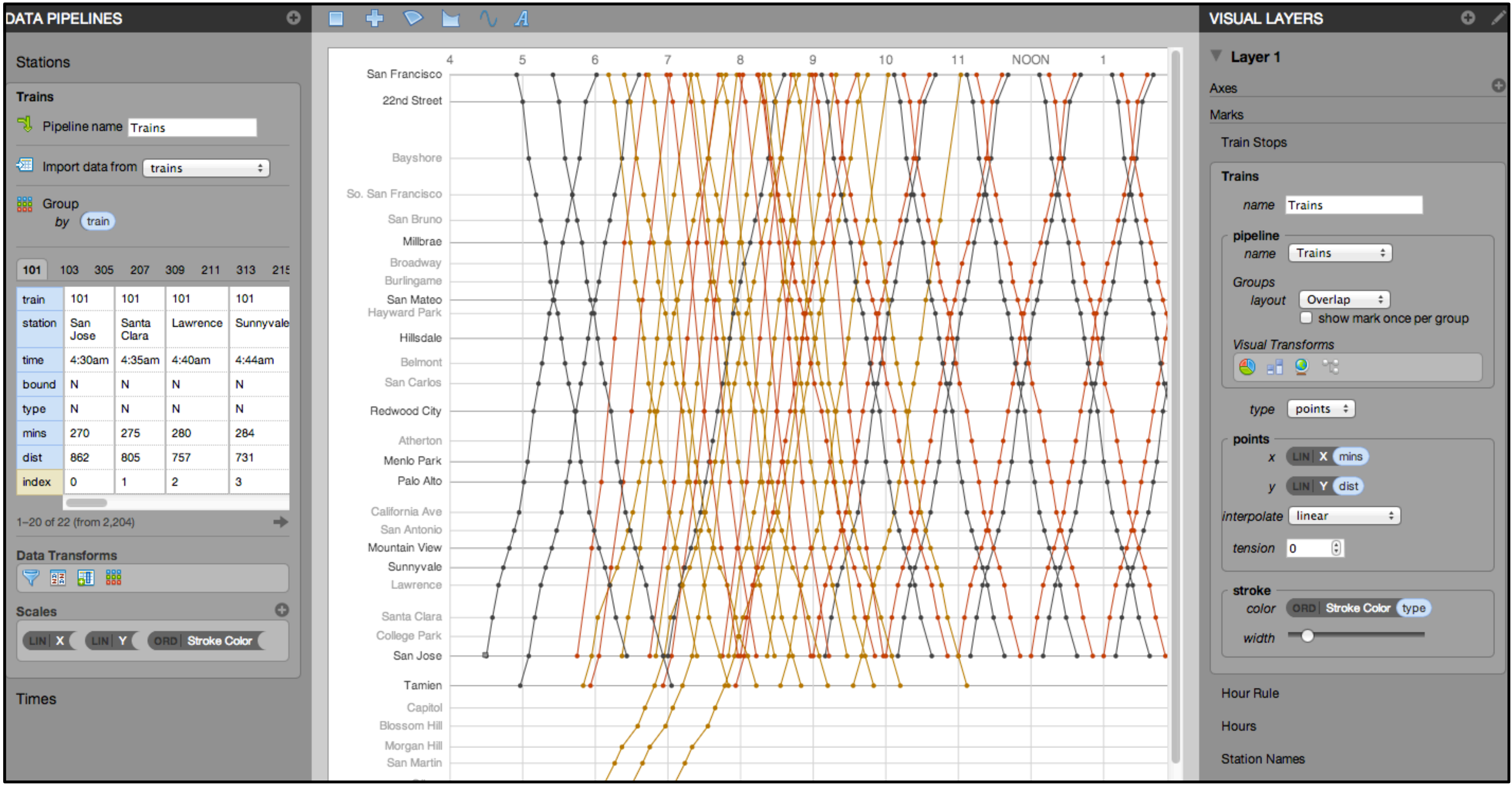
y
start LIN Y offset
height 8
end 0

fill
color ORD Fill Color commonwealth
opacity

stroke
color
width

by William Playfair

Lyra A Visualization Design Environment



DATA PIPELINES

Stations

Trains

Pipeline name: Trains

Import data from: trains

Group by: train

101	103	305	207	309	211	313	215

train	101	101	101	101
station	San Jose	Santa Clara	Lawrence	Sunnyvale
time	4:30am	4:35am	4:40am	4:44am
bound	N	N	N	N
type	N	N	N	N
mins	270	275	280	284
dist	862	805	757	731
index	0	1	2	3

1–20 of 22 (from 2,204)

Data Transforms

Scales: LIN X · LIN Y · ORD Stroke Color

Times

VISUAL LAYERS

Layer 1

Axes

Marks

Train Stops

Trains

name: Trains

pipeline

name: Trains

Groups

layout: Overlap

□ show mark once per group

Visual Transforms

type: points

points

x: LIN X mins

y: LIN Y dist

interpolate: linear

tension: 0

stroke

color: ORD Stroke Color type

width

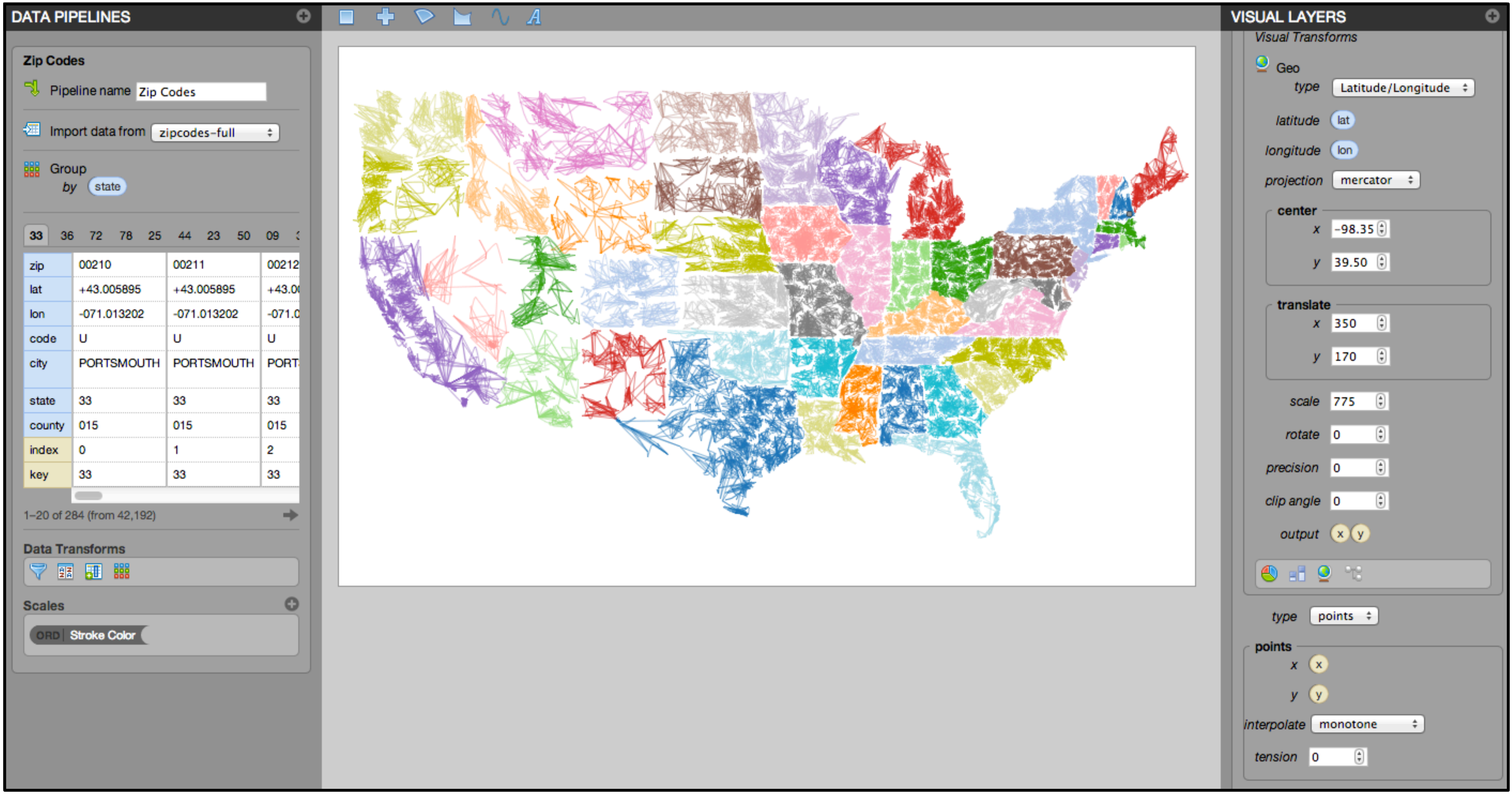
Hour Rule

Hours

Station Names

based on the **Railway Timetable** by E. J. Marey

Lyra A Visualization Design Environment



DATA PIPELINES

Zip Codes

Pipeline name Zip Codes

Import data from zipcodes-full

Group
by state

33	36	72	78	25	44	23	50	09

zip	00210	00211	00212
lat	+43.005895	+43.005895	+43.0
lon	-071.013202	-071.013202	-071.0
code	U	U	U
city	PORTSMOUTH	PORTSMOUTH	PORT
state	33	33	33
county	015	015	015
index	0	1	2
key	33	33	33

1–20 of 284 (from 42,192)

Data Transforms

Scales

ORD Stroke Color

VISUAL LAYERS

Visual Transforms

Geo

type Latitude/Longitude

latitude lat

longitude lon

projection mercator

center
x -98.35
y 39.50

translate
x 350
y 170

scale 775

rotate 0

precision 0

clip angle 0

output x y

type points

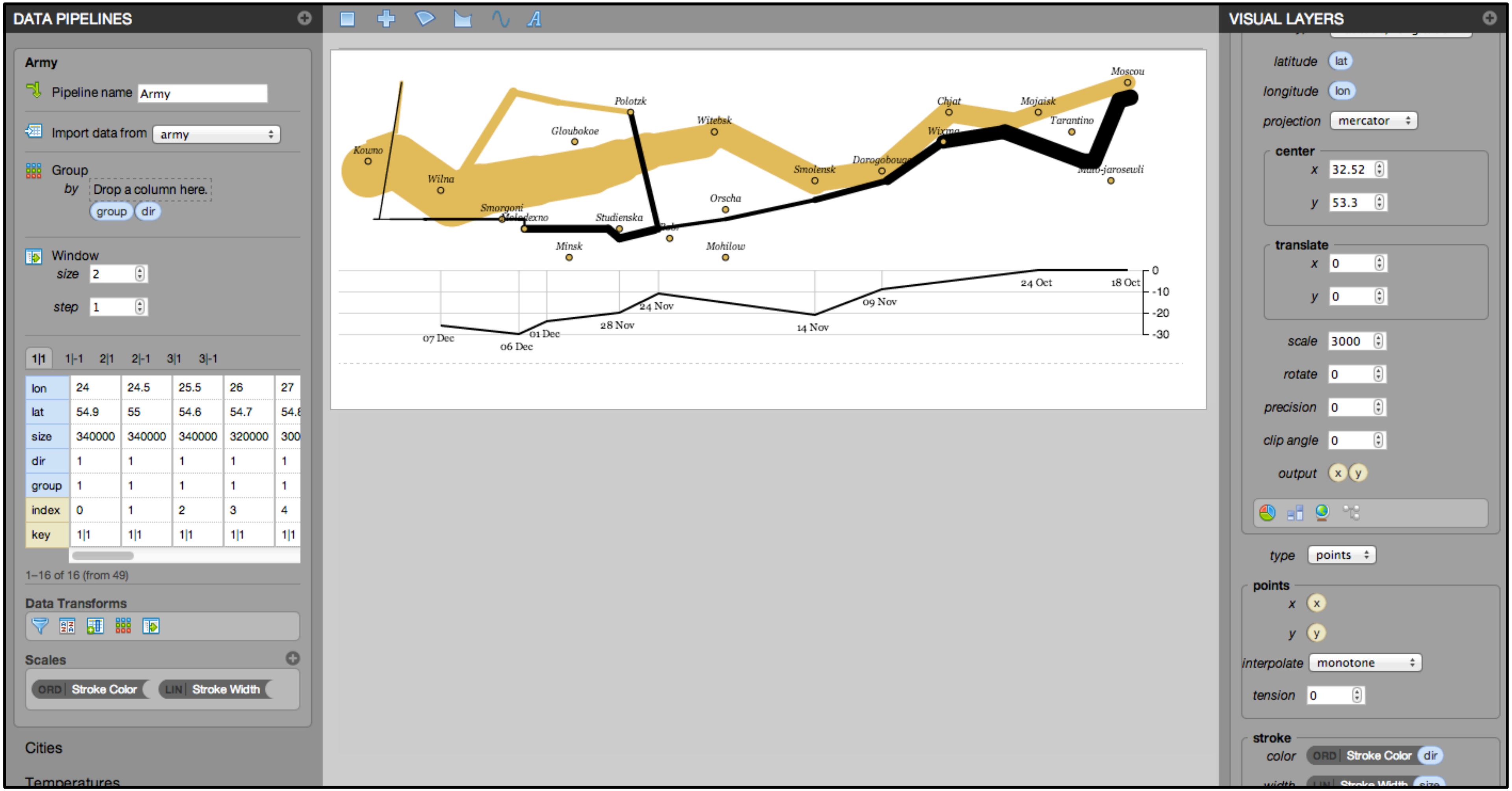
points
x x
y y

interpolate monotone

tension 0

ZipScribble by Robert Kosara

Lyra A Visualization Design Environment



Army

Pipeline name: Army

Import data from: army

Group by: Drop a column here. (group, dir)

Window size: 2

step: 1

1|1	1|-1	2|1	2|-1	3|1	3|-1

lon	24	24.5	25.5	26	27
lat	54.9	55	54.6	54.7	54.8
size	340000	340000	340000	320000	300
dir	1	1	1	1	1
group	1	1	1	1	1
index	0	1	2	3	4
key	1|1	1|1	1|1	1|1	1|1

1–16 of 16 (from 49)

Data Transforms

Scales: ORD Stroke Color, LIN Stroke Width

Cities

Temperatures

latitude: lat

longitude: lon

projection: mercator

center x: 32.52

center y: 53.3

translate x: 0

translate y: 0

scale: 3000

rotate: 0

precision: 0

clip angle: 0

output: x, y

type: points

points x: x

points y: y

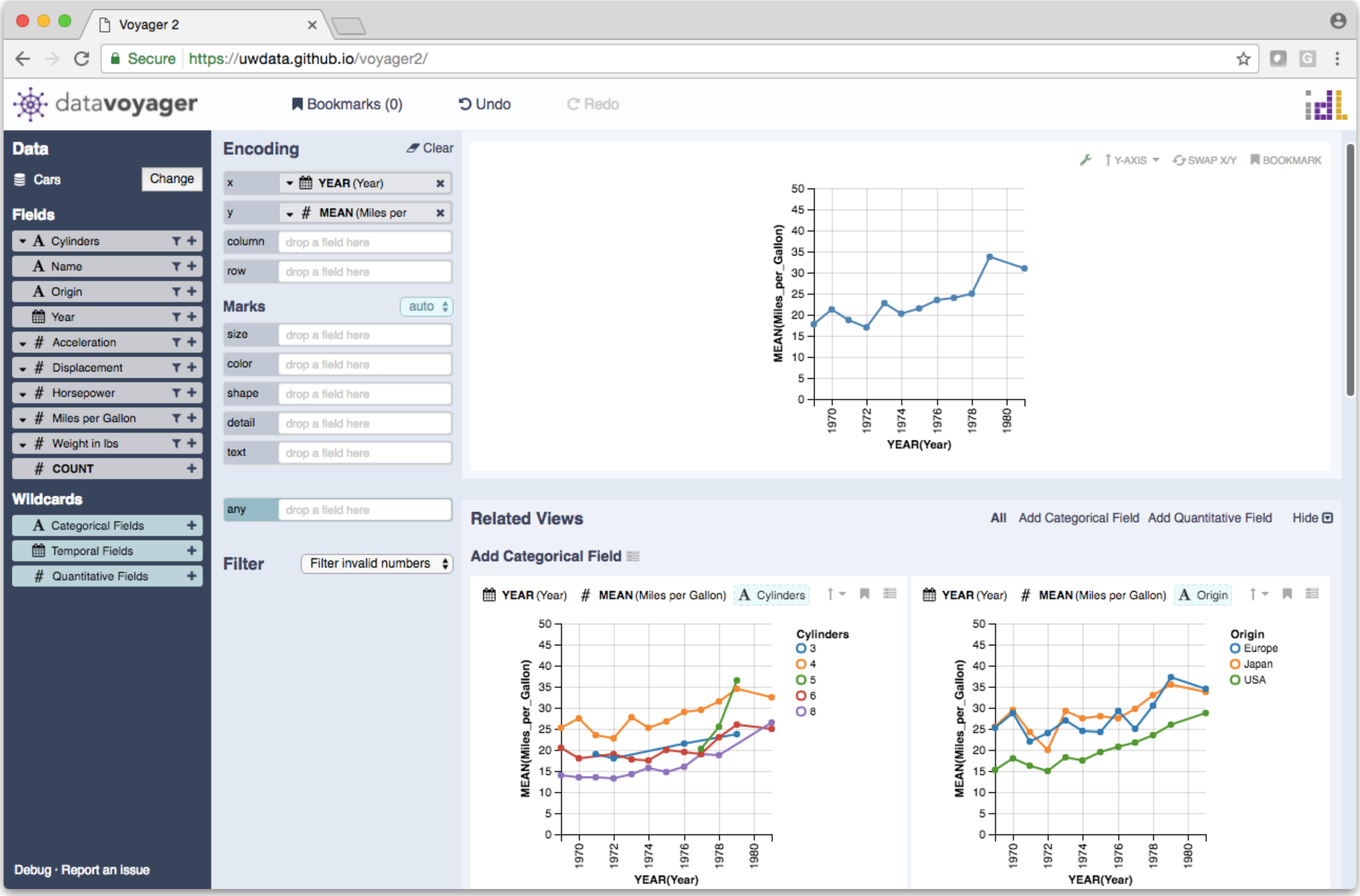
interpolate: monotone

tension: 0

stroke color: ORD Stroke Color, dir

width: LIN Stroke Width, size

Napoleon's March by Charles Minard



x: YEAR (Year)

y: MEAN (Miles per

column: drop a field here

row: drop a field here

Marks: auto

size: drop a field here

color: drop a field here

shape: drop a field here

detail: drop a field here

text: drop a field here

any: drop a field here

Filter: Filter invalid numbers

Voyager. Wongsuphasawat et al. *InfoVis’15, CHI’17*

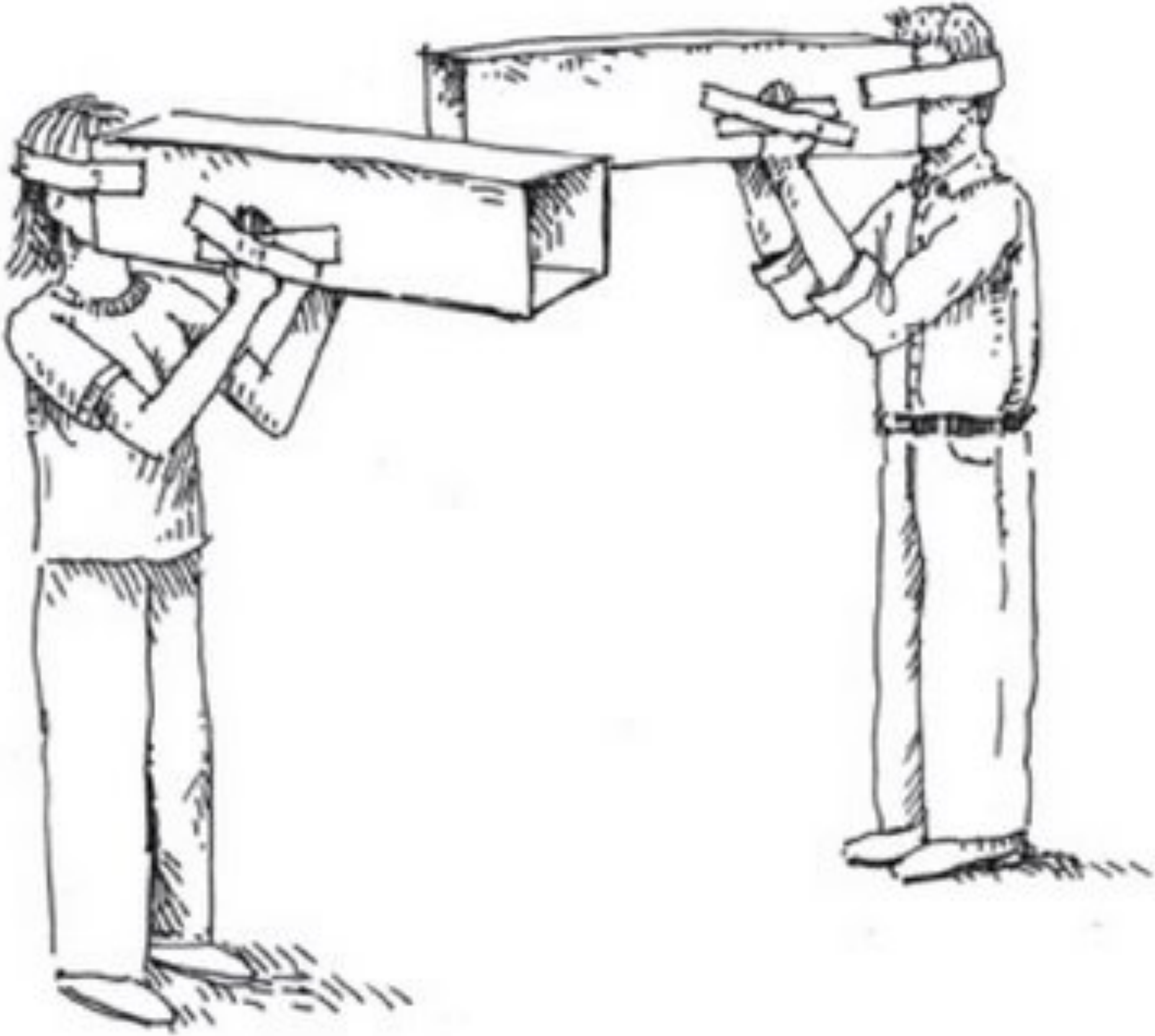
Common exploration pitfalls:

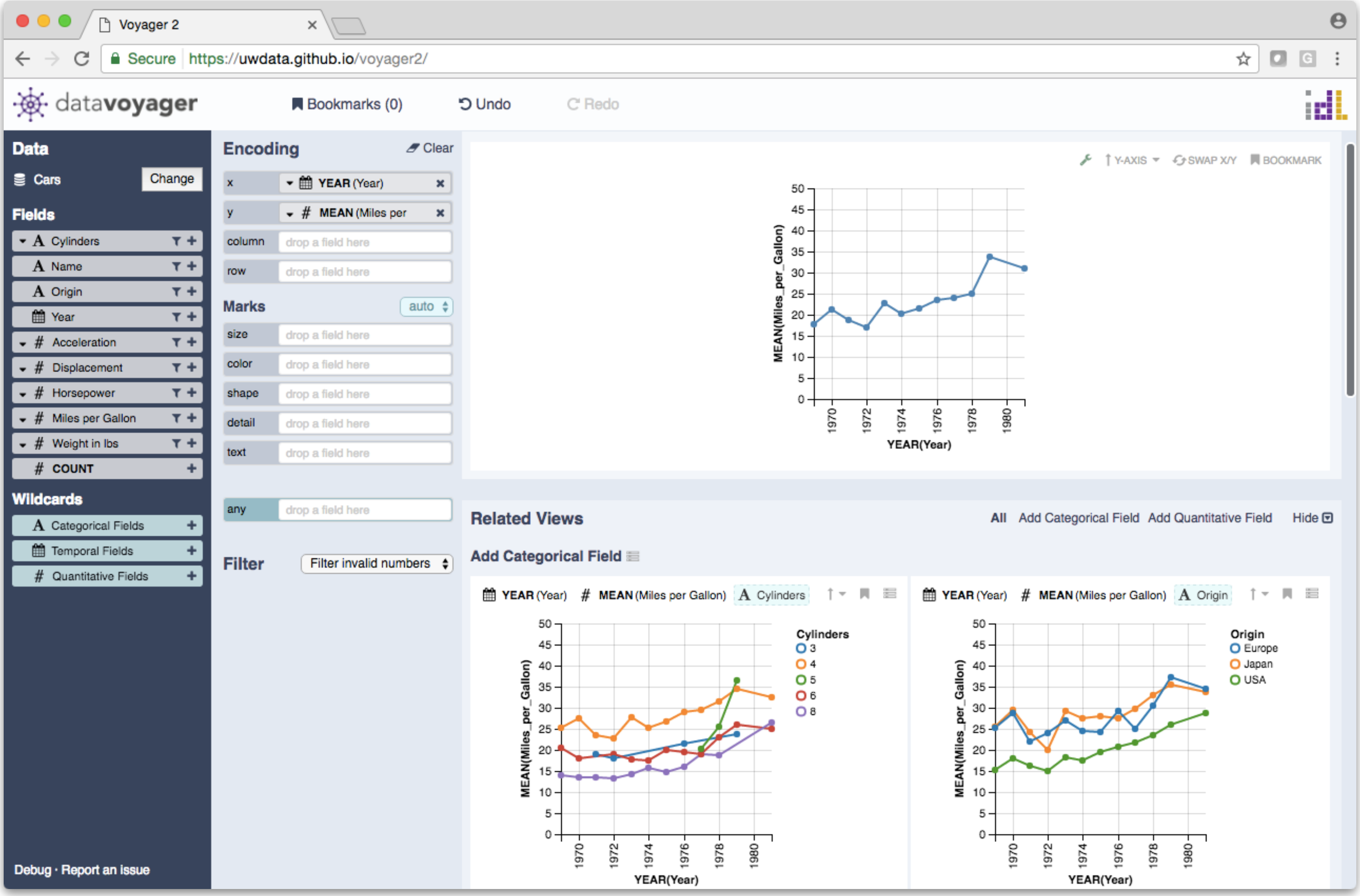
Overlook data quality issues

Fixate on specific relationships

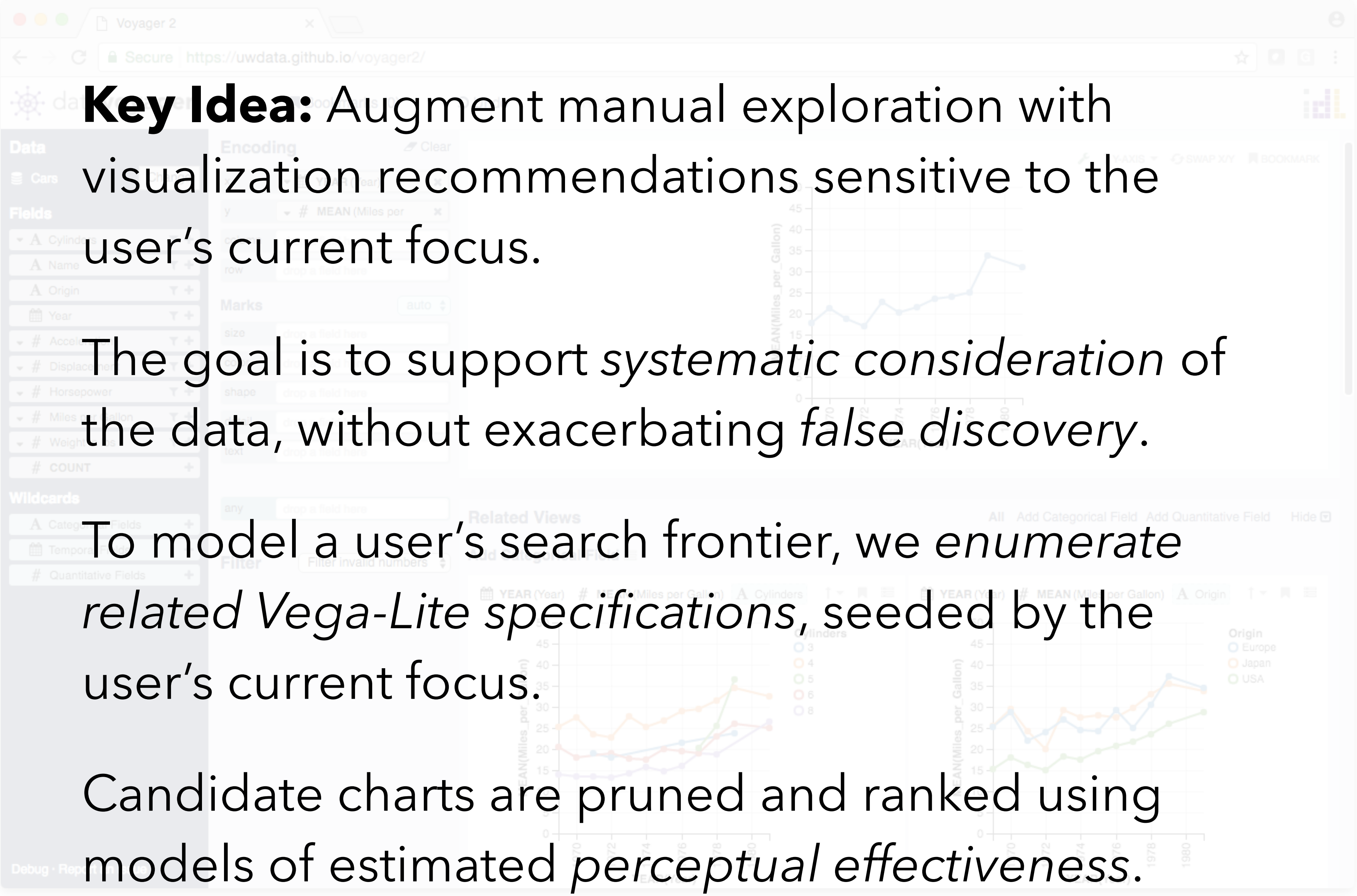
Plus many other biases…

[Heuer 1999, Kahneman 2011, …]





Voyager. Wongsuphasawat et al. *InfoVis'15, CHI'17*



Key Idea: Augment manual exploration with visualization recommendations sensitive to the user's current focus.

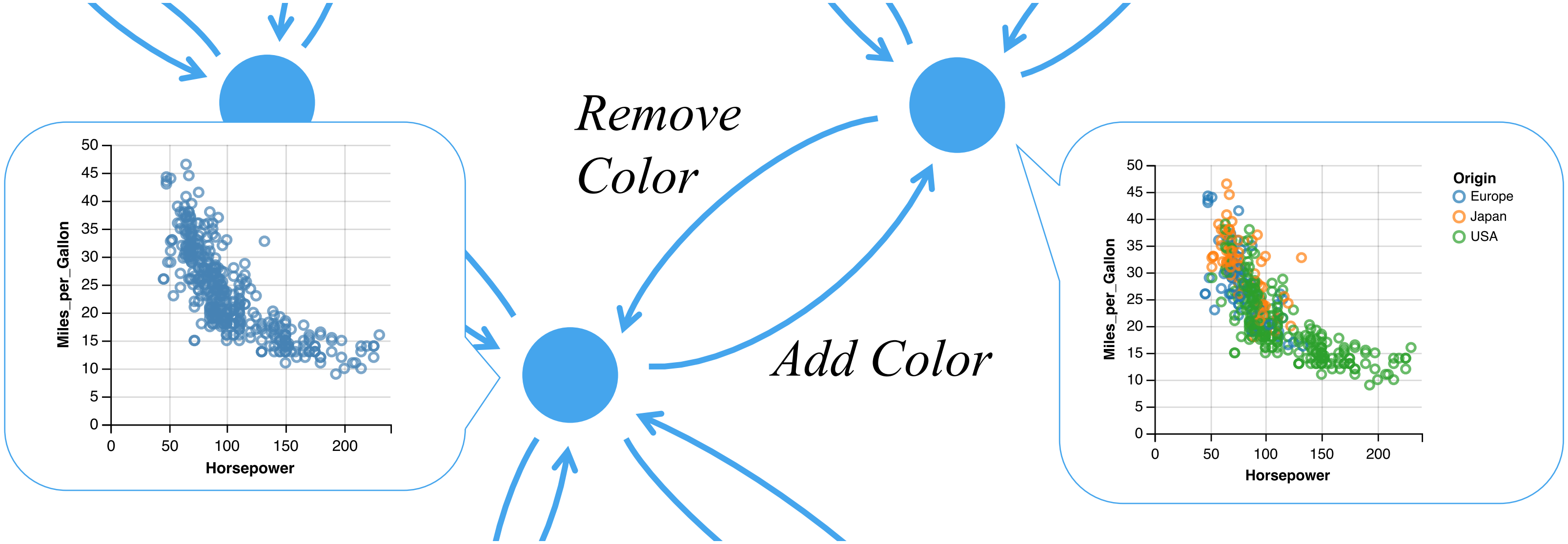
The goal is to support *systematic consideration* of the data, without exacerbating *false discovery*.

To model a user's search frontier, we *enumerate related Vega-Lite specifications*, seeded by the user's current focus.

Candidate charts are pruned and ranked using models of estimated *perceptual effectiveness*.

Voyager. Wongsuphasawat et al. *InfoVis'15, CHI'17*

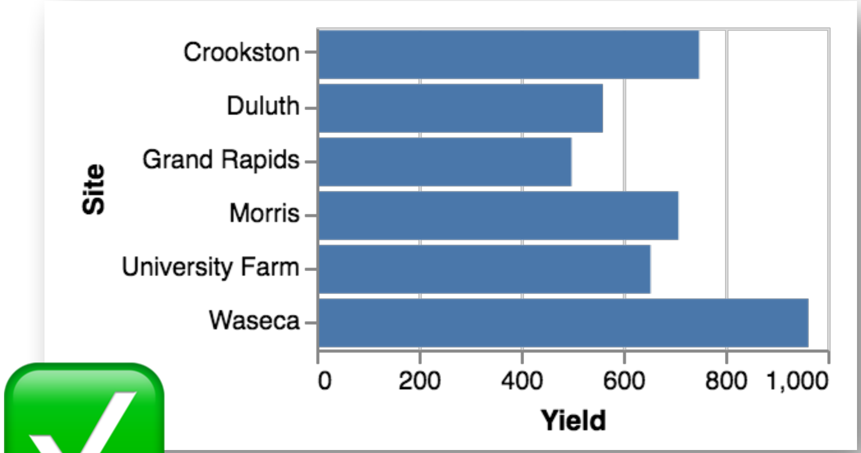
A Formal Design Space of Visualizations



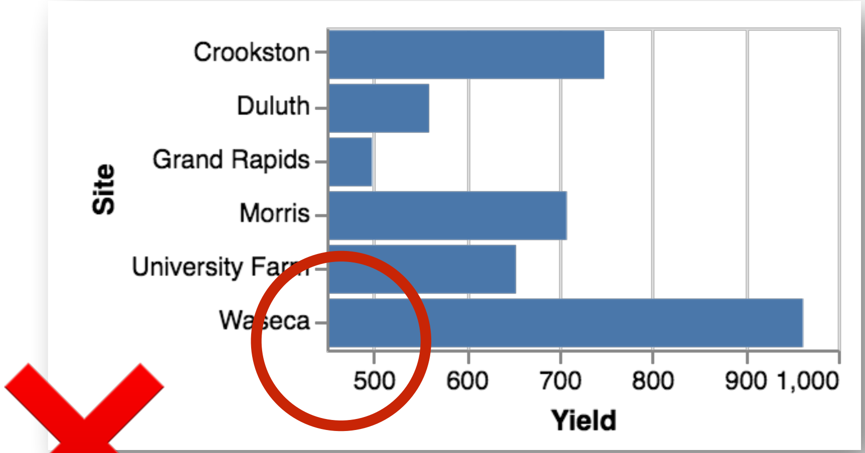
Enumerate **Vega-Lite specifications** and transformations among them. Search the space using logic programming methods.

[Kim et al. 2017]

Articulate Design Constraints



✓



✗

"Quantitative axes should include a zero baseline"

When and how strongly should we apply this?

How to balance with other such constraints?

[Moritz et al. 2019]

Learn Design Trade-Offs from Data

Training Data

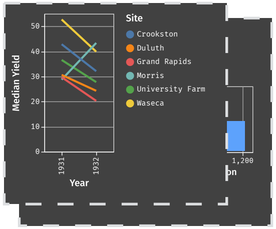
Pairs of Ranked Visualizations

Features

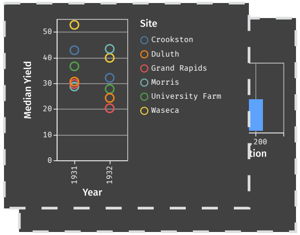
Violations of Design Constraints

Learning Algorithm

Learning to Rank with Linear SVM



👍 Positive example $[u_1, u_2, \ldots, u_k]$



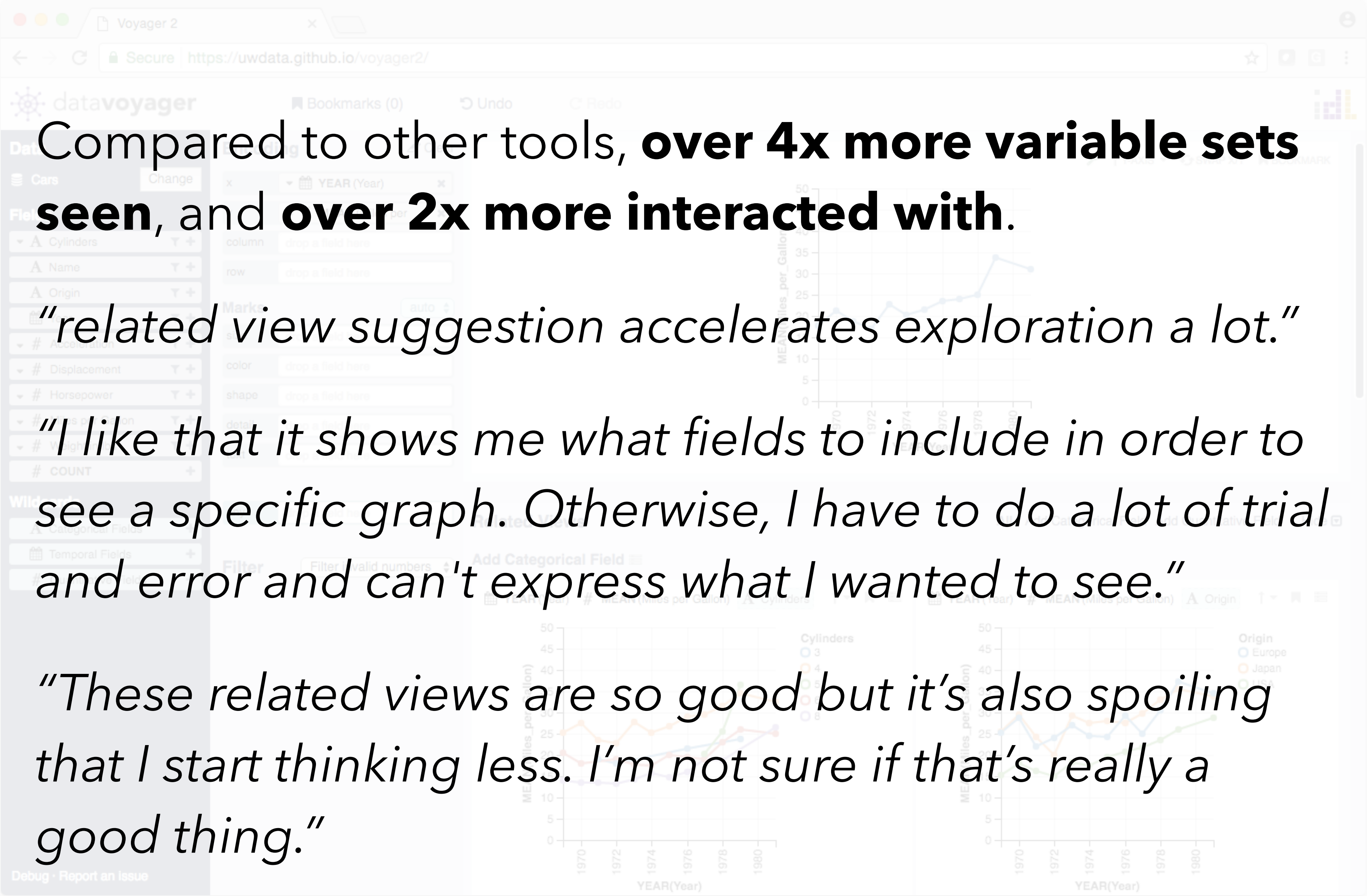
👎 Negative example $[v_1, v_2, \ldots, v_k]$

v_i: the number of violations of constraint i.

$$\arg\max_w \sum_{i \in 0 \ldots k} w_i (u_i - v_i)$$

w is the weight vector of the soft constraints

[Moritz et al. 2019]



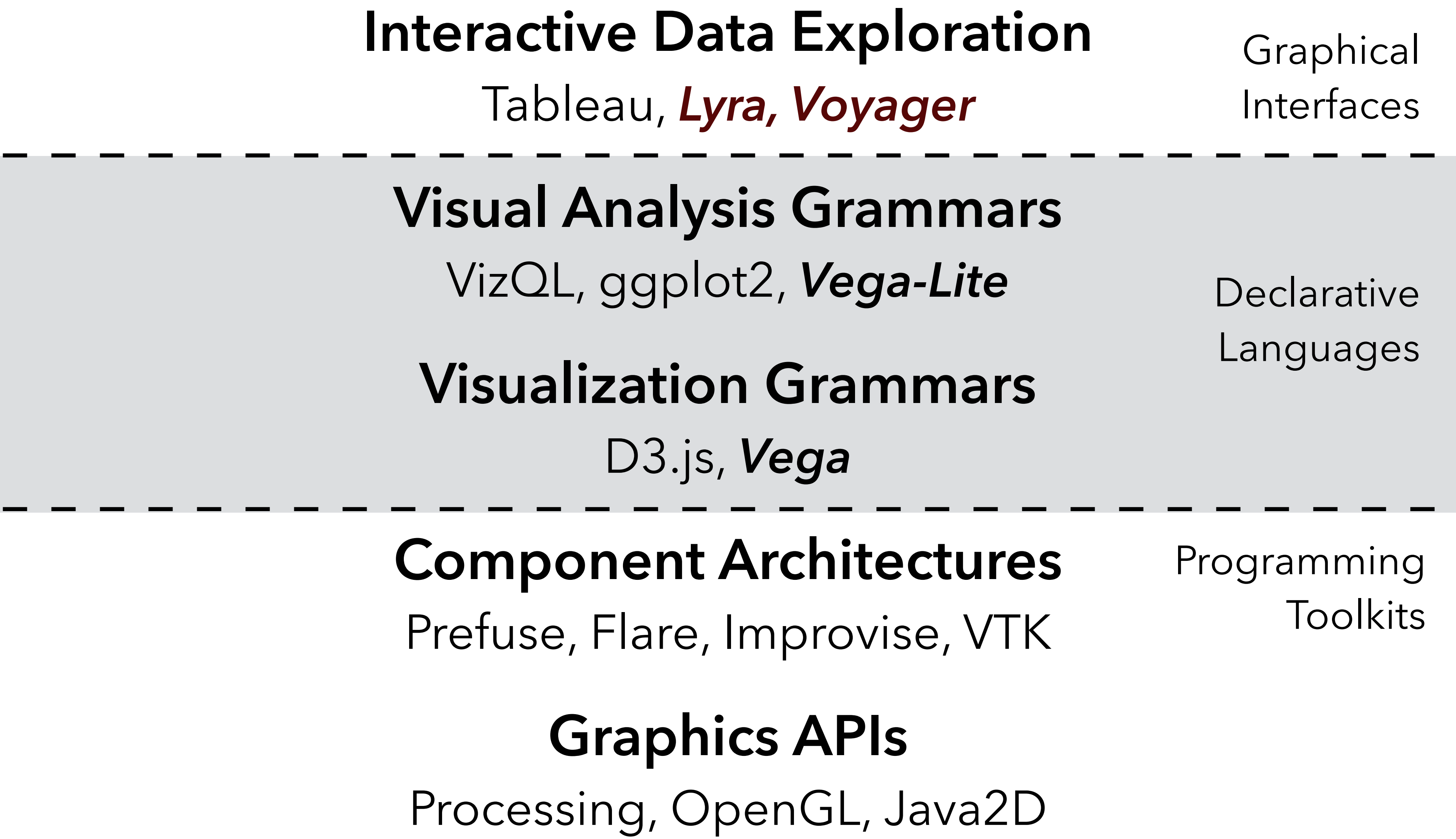
Compared to other tools, **over 4x more variable sets seen**, and **over 2x more interacted with**.

"related view suggestion accelerates exploration a lot."

"I like that it shows me what fields to include in order to see a specific graph. Otherwise, I have to do a lot of trial and error and can't express what I wanted to see."

"These related views are so good but it's also spoiling that I start thinking less. I'm not sure if that's really a good thing."

Voyager. Wongsuphasawat et al. *InfoVis'15, CHI'17*



Interactive Data Exploration

Tableau, ***Lyra, Voyager***

Graphical Interfaces

Visual Analysis Grammars

VizQL, ggplot2, ***Vega-Lite***

Declarative Languages

Visualization Grammars

D3.js, ***Vega***

Component Architectures

Prefuse, Flare, Improvise, VTK

Programming Toolkits

Graphics APIs

Processing, OpenGL, Java2D