Lecture 19:
Toolkits for building speech/conversational/chatbot User Interfaces, and Visualizations

05-431/631 Software Structures for User Interfaces (SSUI)

Fall, 2022

Logistics

- Changed deadlines for HWs 4, 5, 6
  - HW5 due today
  - HW6 due Thursday, 11/17
- Will still start on projects next Tuesday
  - More information soon!

Based on:
Toolkits for Creating Conversational Interfaces

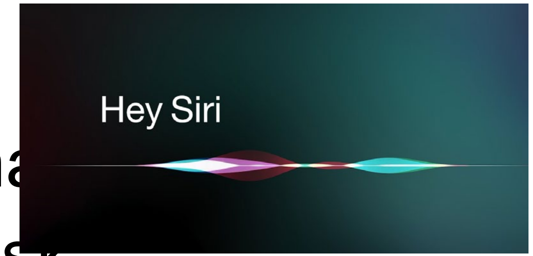
by Toby Jia-Jun Li http://toby.li/

04/20/2020

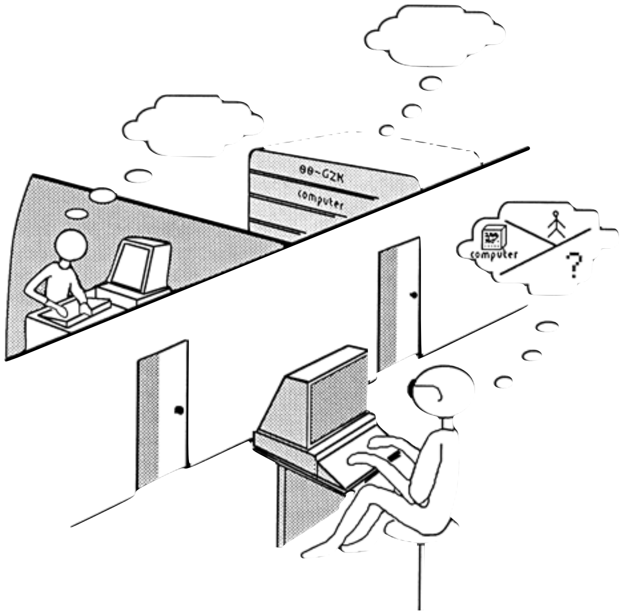
Conversational Interfaces

- Intelligent personal assistants
  Alexa, Siri, Google Assistant, Cortana
- Voice command support for specific task domains
  e.g., Talking to your car
- Automated phone systems for customer service
- Chatbots for tech support or fun

- Sometimes students like to use speech interfaces for their final projects



History



Turing Test (1950)

History

- Let computers facilitate formulative thinking as they now facilitate the solution of formulated problems
- Enable men and computers to cooperate in making decisions and controlling complex situations without inflexible dependence on predetermined programs.
- "Man-Computer Symbiosis (1960): Cooperative interaction between men and electronic computers"

J. C. R. Licklider

Lots of research and commercial attempts

- Influential early *multi-modal* system: Put That There (1980)
  - Bolt, Richard A. "Put-that-there": Voice and gesture at the graphics interface. *SIGGRAPH Computer Graphics*. Vol. 14. No. 3. ACM, 1980.
  - https://youtu.be/sC5Zg0fU2e8 (5:30)

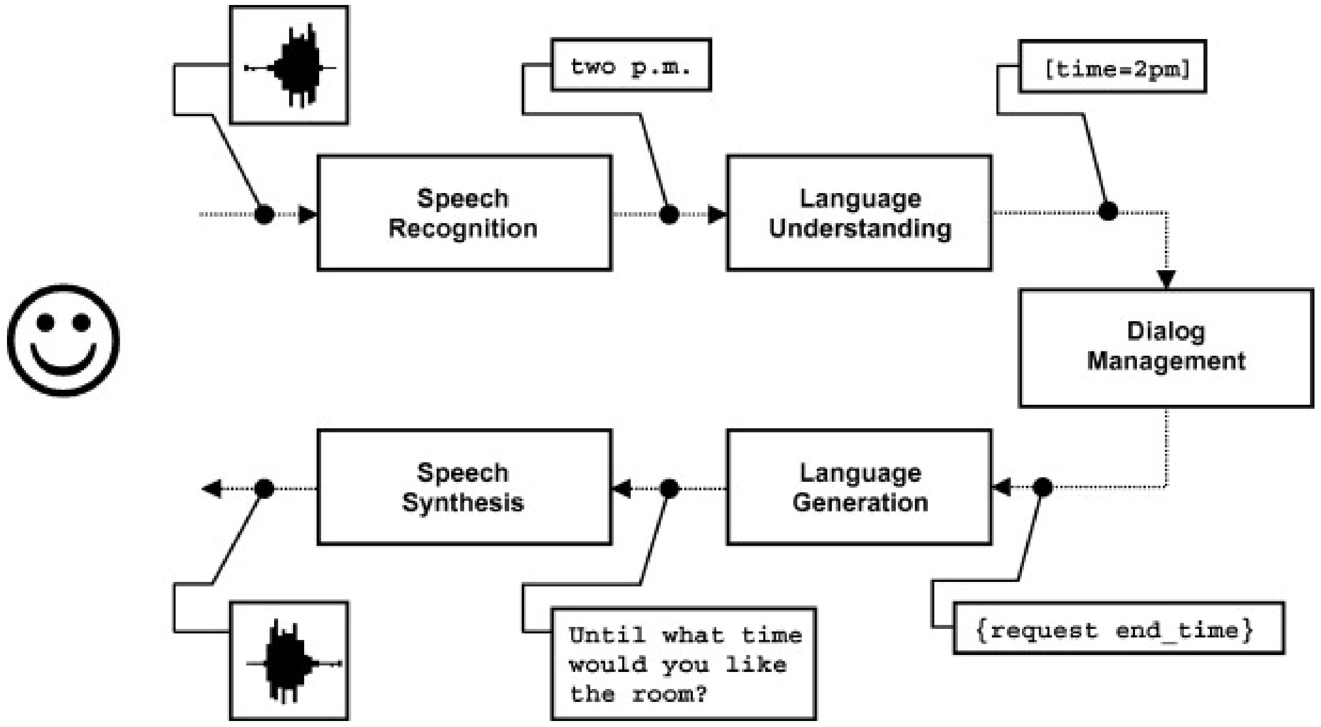
Advantages of conversational interfaces

- **Hands-free**: can handle situations where direct manipulation is not possible or convenient (e.g., far away, driving, users with accessibility needs)
- **Screen size independence**: can operate on devices with small screens (e.g., wearable) and no screen.
- **Intuitive to use**: well-designed conversational interfaces should have low learning barriers to users.
- **Efficient**: takes less time and effort for ***some tasks*** that require a lot of text entry, or navigating complex menus.
  - Can be inefficient and hard-to-use in some situations too! E.g., when the prompts are too verbose, when the affordances are unclear (discoverability), or when the error handling mechanism is lacking.

Two classes of conversational systems

1. Task-oriented conversational agents
   - Purpose: help the user perform some specific tasks
2. Social chatbots (“chit-chat” bots)
   - Purpose: maintain realistic conversations with humans

Practical architectures for task-oriented dialog systems



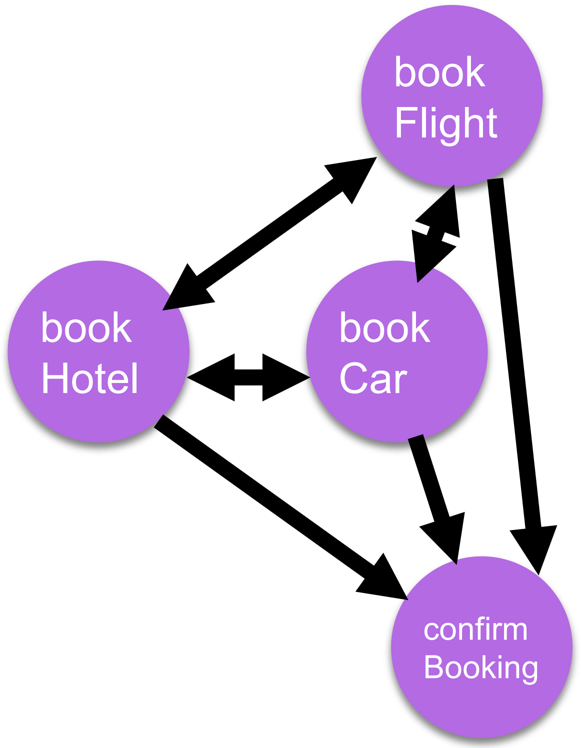
RavenClaw (Bohus and Rudnicky, 2003)

Bohus, Dan, and Alexander I. Rudnicky. "RavenClaw: Dialog management using hierarchical task decomposition and an expectation agenda." Eighth European Conference on Speech Communication and Technology. 2003.

Practical architectures for task-oriented dialog systems

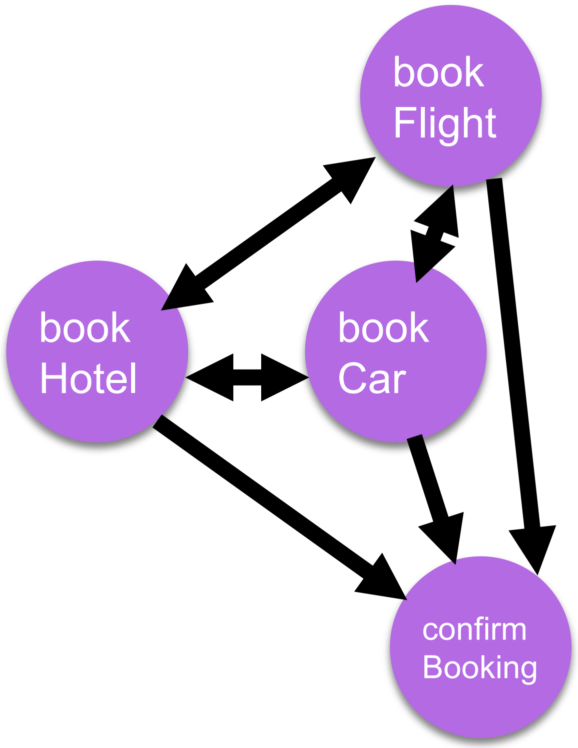
- Finite-state
  - The developer manually defines all the conversation states in the system, and the transitions between the states.
- Frame-based
  - **frame** ("intent"): the user's intention for one conversation turn (e.g., book_flight)
  - **slot**: the information that the system needs to know to fulfill an intent (e.g., departure_date, destination_city)
  - **slot values**: the values that each slot can take

User: I want to book a flight for 2 to Munich.



User: I want to **book a flight** for 2 to Munich.

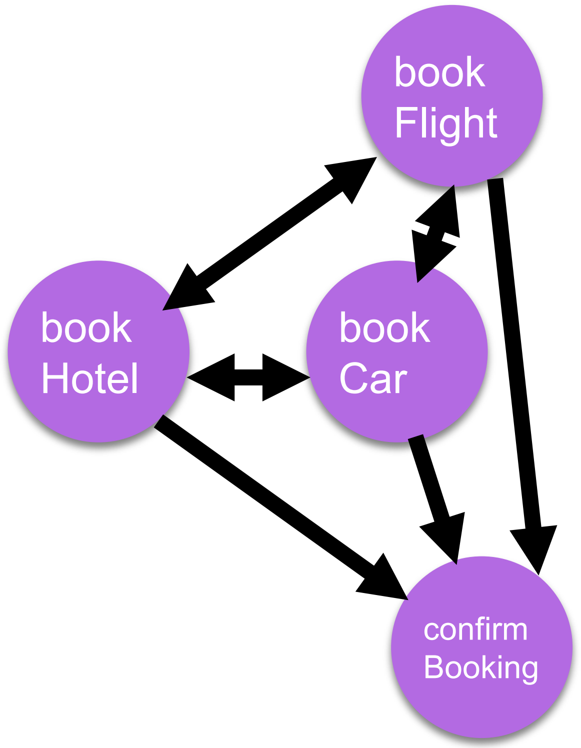
Intent recognition



Intent: bookFlight **Slots:** departureCity, arrivalCity, personCount, date

User: I want to book a flight for **2** to **Munich**.

Entity extraction / slot filling

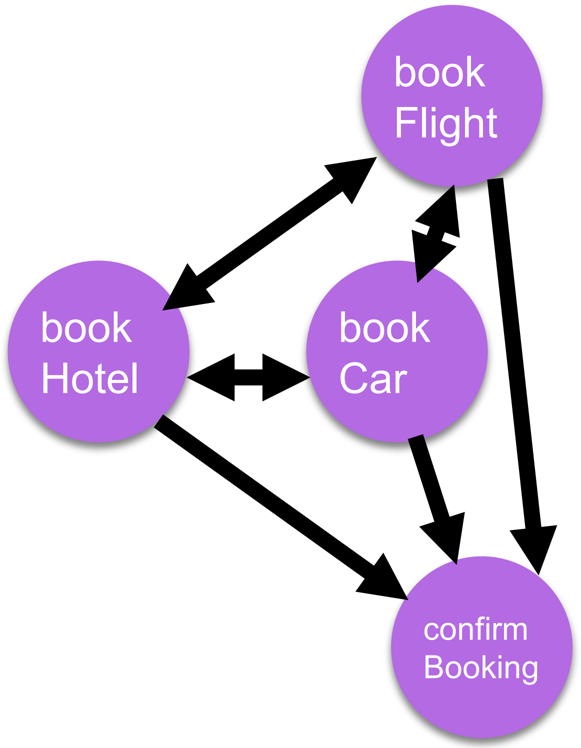


Intent: bookFlight **Slots:** departureCity, ~~**arrivalCity**~~, ~~**personCount**~~, date

User: I want to **book a flight** for **2** to **Munich**.

Bot: What city are you flying from?

User: Pittsburgh.

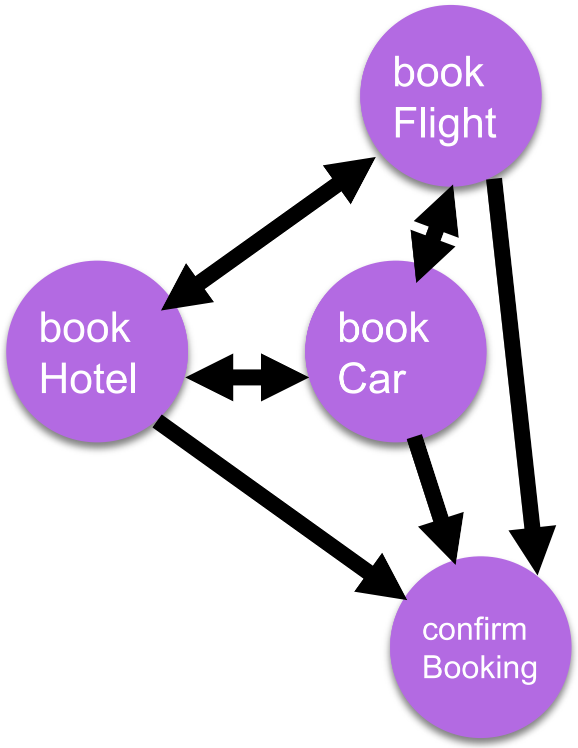


Intent: bookFlight **Slots:** departureCity, ~~**arrivalCity**~~, ~~**personCount**~~, date

User: I want to **book a flight** for **2** to **Munich**.

Bot: What city are you flying from?

User: **Pittsburgh**.



Intent: bookFlight **Slots:** ~~**departureCity**~~, ~~**arrivalCity**~~, ~~**personCount**~~, date

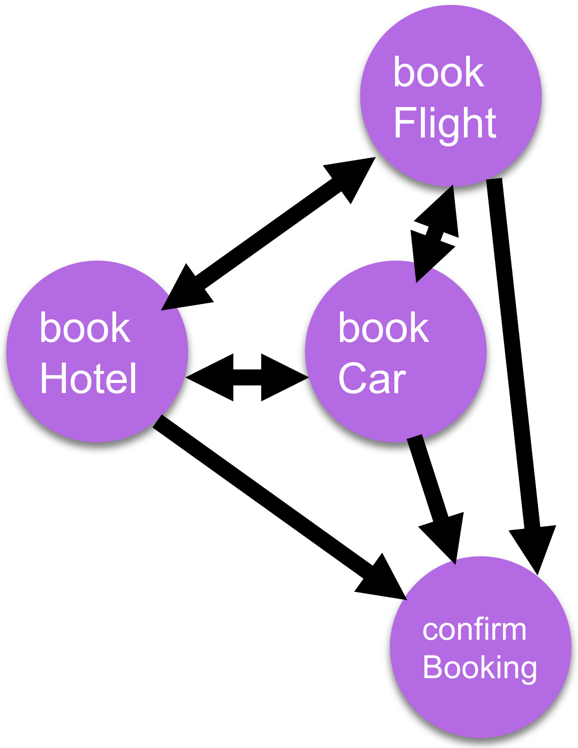
User: I want to **book a flight** for **2** to **Munich**.

Bot: What city are you flying from?

User: **Pittsburgh**.

Bot: What's the departure date for the flight?

User: **Tomorrow**.



Intent: bookFlight **Slots: ~~departureCity~~, ~~arrivalCity~~, ~~personCount~~, ~~date~~**

User: I want to **book a flight** for **2** to **Munich**.

Bot: What city are you flying from?

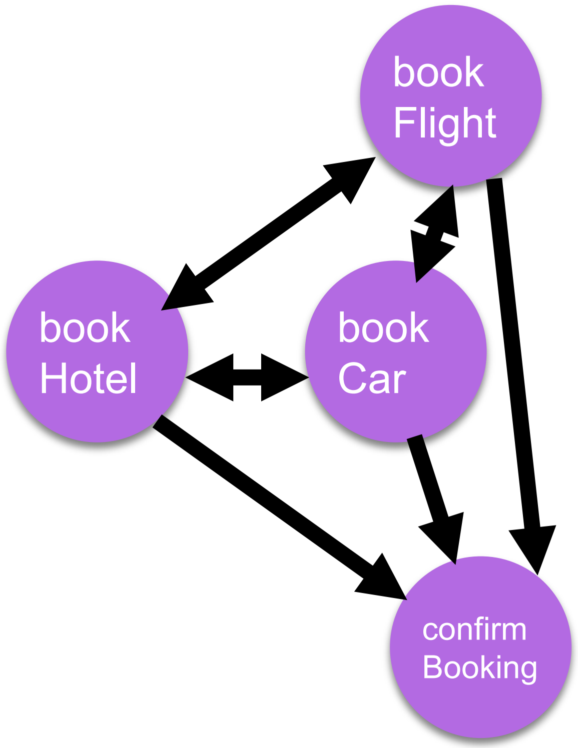
User: **Pittsburgh**.

Bot: What's the departure date for the flight?

User: **Tomorrow**.

Bot: Do you want to also book a hotel or a car?

User: I'd like to **get a place to stay** too.



Intent: bookHotel **Slots:** …..

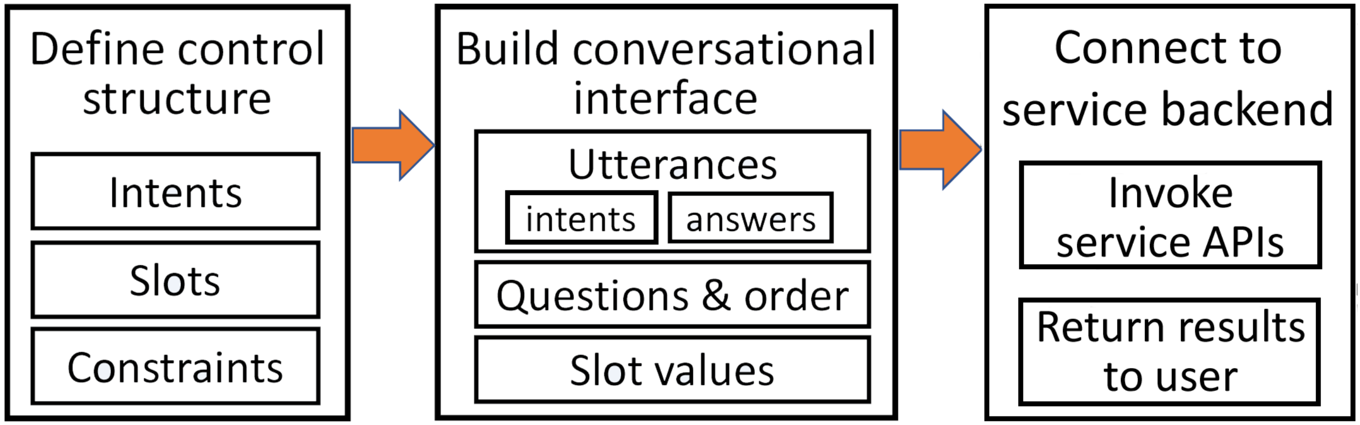
Existing tools for building slot-filling bots

Dialogflow

wit.ai

Microsoft Bot Framework

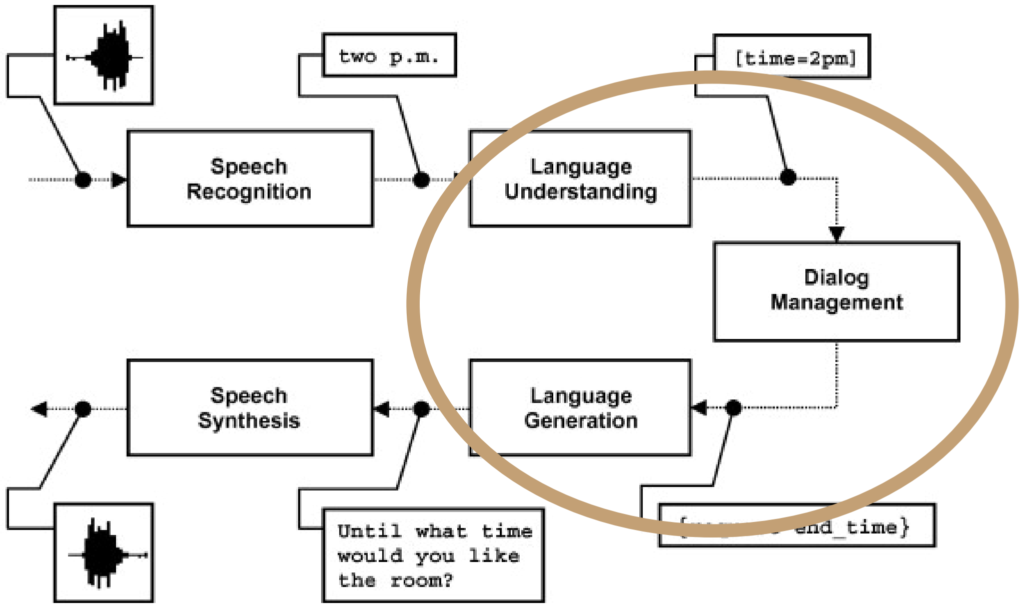
Amazon Lex



Dialogflow

Dialogflow

- One of the more popular toolkits
  - https://cloud.google.com/dialogflow/docs
- Can easily connect to other Google components (e.g., speech recognition, speech synthesis, knowledge graph…) ☺



Other architectures for dialog systems

1. **Rule-based**

```
(if (contains (or “hi” “hello”)) (output “hello”))
(if (and (= detect_comm_type SELF_DISCLOSURE) (= detect_emotion SAD))
    (output “I'm sorry to hear [$USER_DISCLOSURE]”))
```

2. **Corpus-based:** use a very large corpus of human-human or human-machine conversations
   - **Information retrieval (IR) based approach:** find the best-matched prior utterance for the user’s input in the corpus, and use the prior response for that utterance
   - **Sequence-to-sequence dialog generation:** model conversation as a sequence transduction problem -> generate a response from a user input (and probably with some other contexts encoded in)

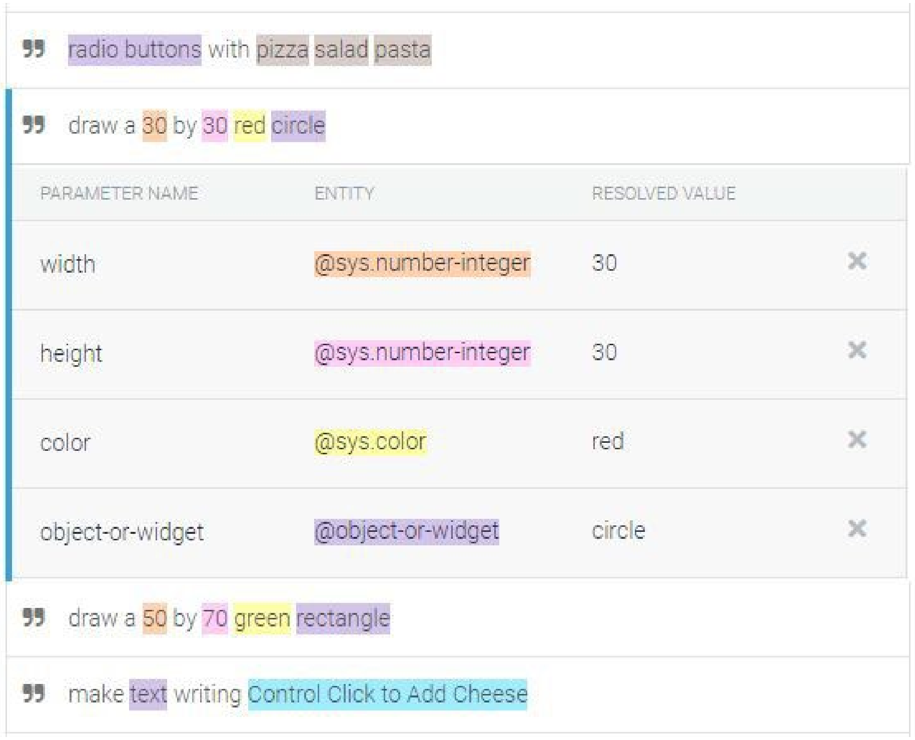
Example: 05-830 project (Spring'20)

- Use DialogFlow to create a GUI Builder
- Thanks to Hongyi Zhang, Mengxin Cao, Ron Chew
- 1-month project

Conversation Design in DialogFlow

Two intents: **Initialization, Interaction**

Capability vs Complexity: What things do we need to specify via voice, or could we use a **demonstration**?



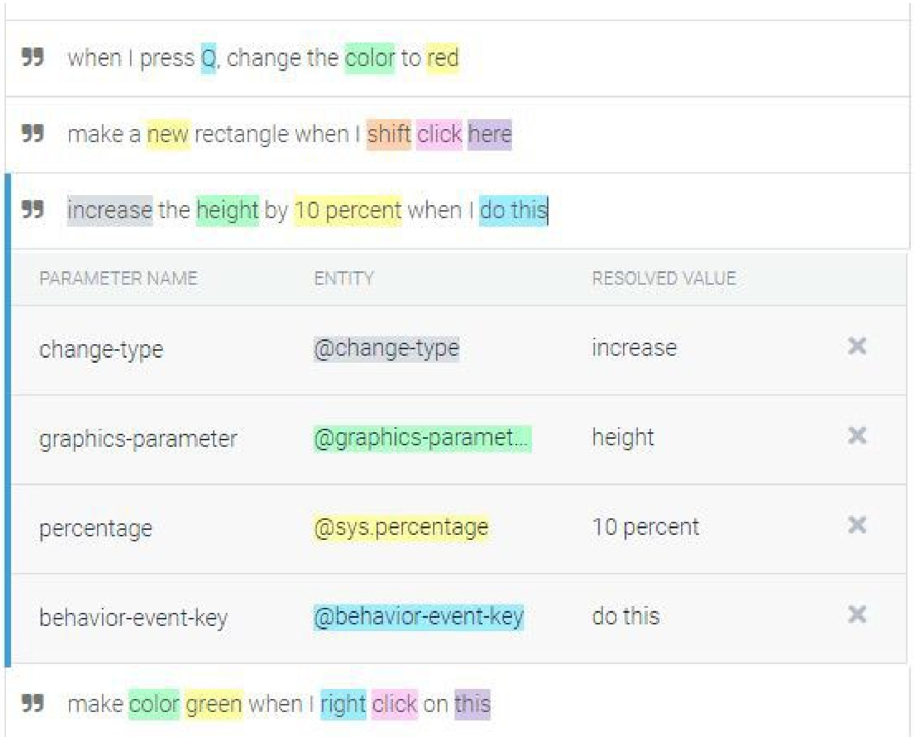
radio buttons with pizza salad pasta

draw a 30 by 30 red circle

PARAMETER NAME	ENTITY	RESOLVED VALUE
width	@sys.number-integer	30
height	@sys.number-integer	30
color	@sys.color	red
object-or-widget	@object-or-widget	circle

draw a 50 by 70 green rectangle

make text writing Control Click to Add Cheese



when I press Q, change the color to red

make a new rectangle when I shift click here

increase the height by 10 percent when I do this

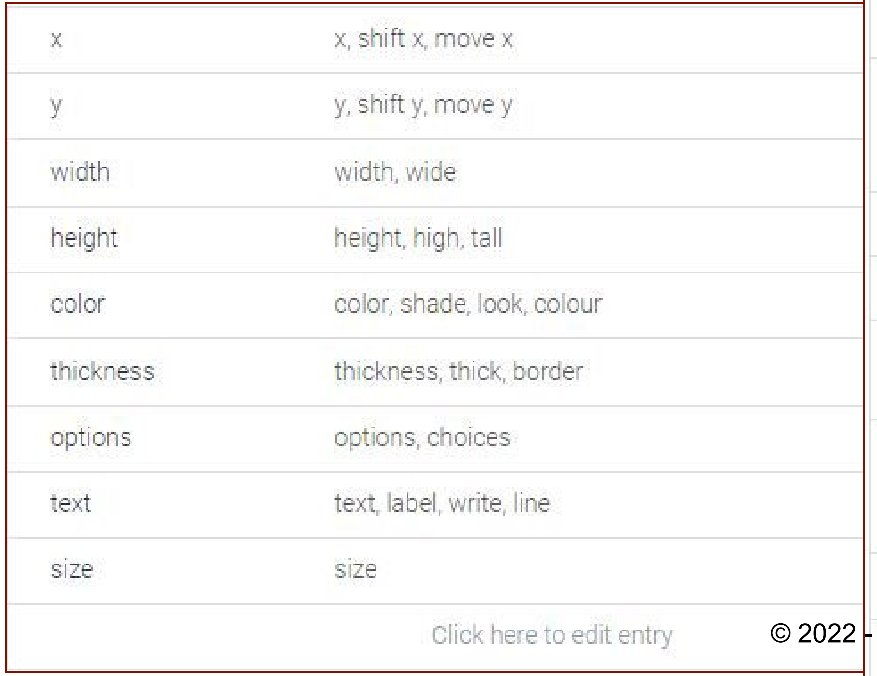
PARAMETER NAME	ENTITY	RESOLVED VALUE
change-type	@change-type	increase
graphics-parameter	@graphics-paramet...	height
percentage	@sys.percentage	10 percent
behavior-event-key	@behavior-event-key	do this

make color green when I right click on this

Conversation Design in DialogFlow

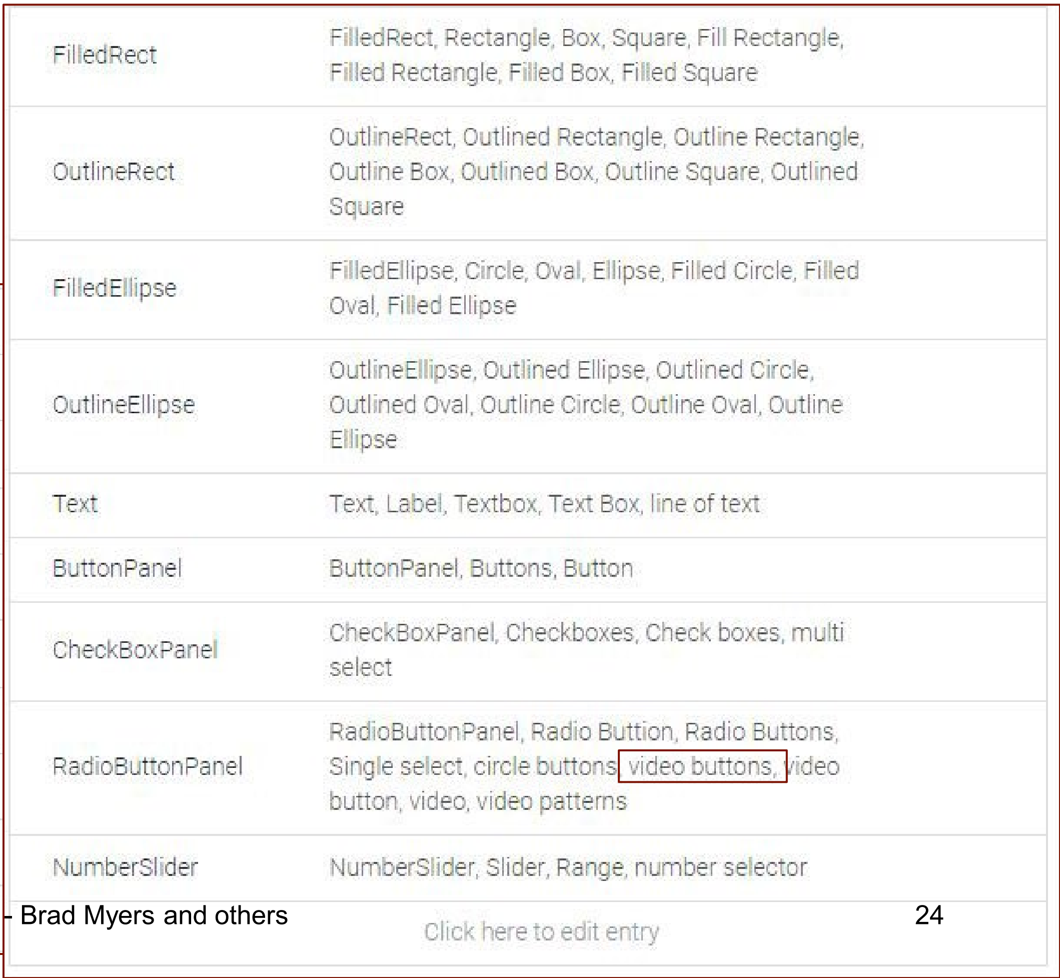
Human-Computer Interaction Institute

- Everyone has a different word for everything…
- Provide synonyms



x	x, shift x, move x
y	y, shift y, move y
width	width, wide
height	height, high, tall
color	color, shade, look, colour
thickness	thickness, thick, border
options	options, choices
text	text, label, write, line
size	size

Click here to edit entry



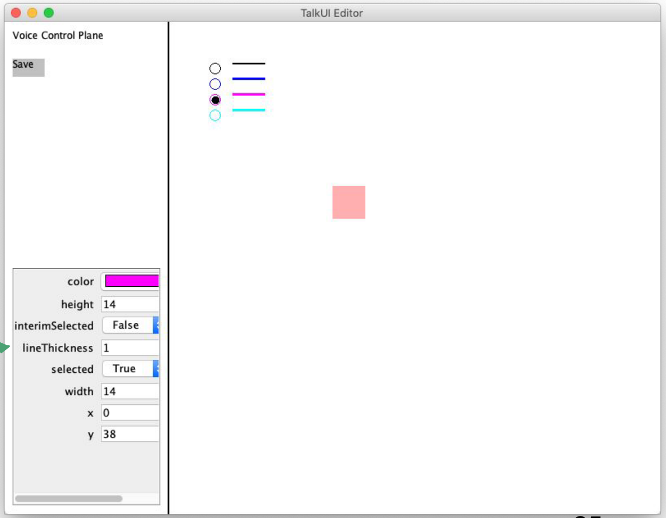
FilledRect	FilledRect, Rectangle, Box, Square, Fill Rectangle, Filled Rectangle, Filled Box, Filled Square
OutlineRect	OutlineRect, Outlined Rectangle, Outline Rectangle, Outline Box, Outlined Box, Outline Square, Outlined Square
FilledEllipse	FilledEllipse, Circle, Oval, Ellipse, Filled Circle, Filled Oval, Filled Ellipse
OutlineEllipse	OutlineEllipse, Outlined Ellipse, Outlined Circle, Outlined Oval, Outline Circle, Outline Oval, Outline Ellipse
Text	Text, Label, Textbox, Text Box, line of text
ButtonPanel	ButtonPanel, Buttons, Button
CheckBoxPanel	CheckBoxPanel, Checkboxes, Check boxes, multi select
RadioButtonPanel	RadioButtonPanel, Radio Buttion, Radio Buttons, Single select, circle buttons, video buttons, video button, video, video patterns
NumberSlider	NumberSlider, Slider, Range, number selector

Click here to edit entry

Interface Design

- Features
  - Continuous voice monitoring
  - Voice control to interact with graphical objects
  - Dialog feedback in both audio and text
  - Property sheet that supports direct manipulation
  - Export existing canvas as a static picture

Property sheet of active object



color	🟪
height	14
interimSelected	False
lineThickness	1
selected	True
width	14
x	0
y	38

Issues Encountered

DialogFlow Issues

- Speech-to-text is pretty crappy
  - Generic speech recognition service vs Google Assistant
  - Compounded by audio recording quality in Java
- Cannot have too many parameters in one intent, but graphics need many
  - Possible Solution: multiple intents, but difficult to manage

Interface Issues

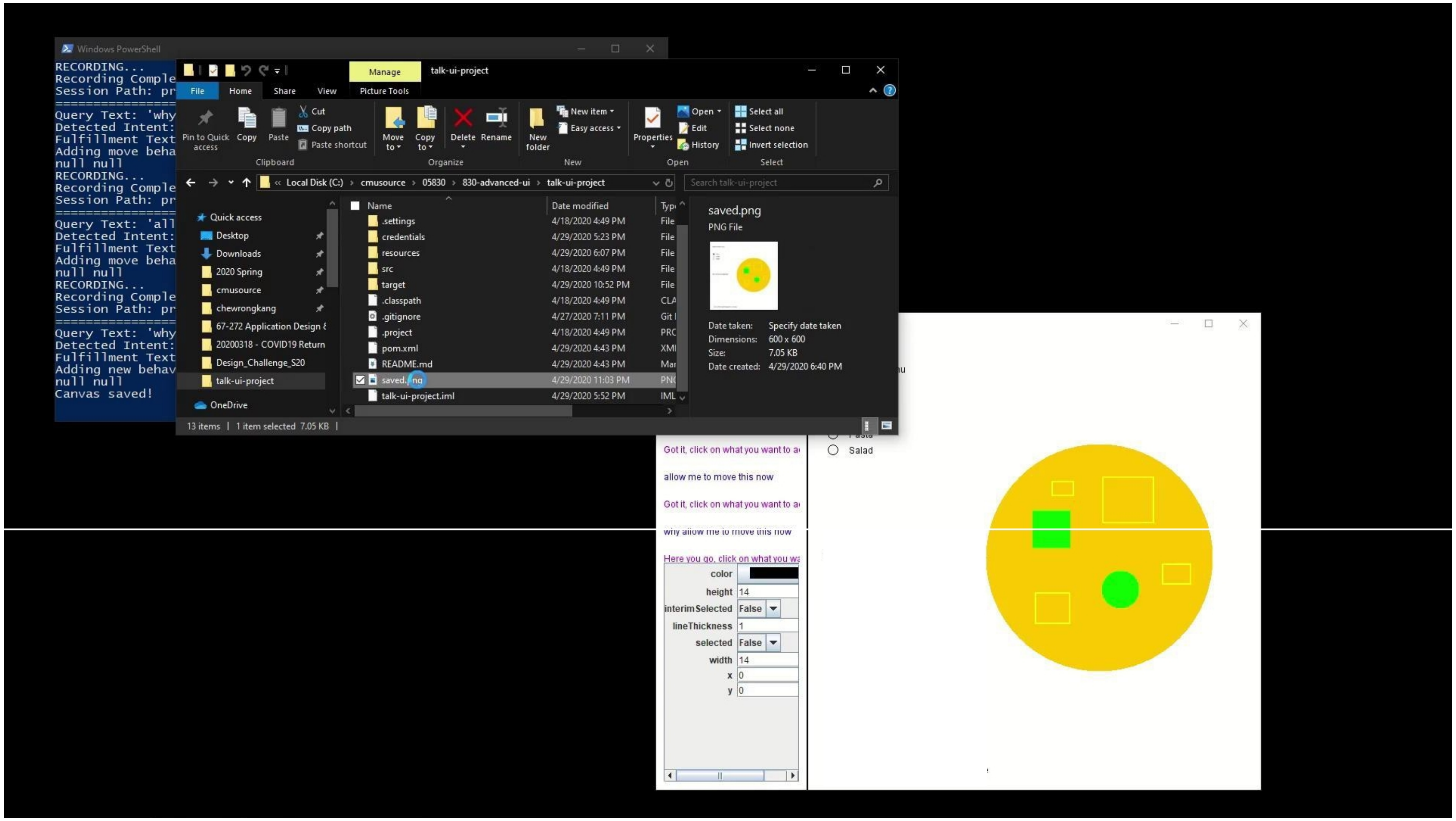
- Hardware heterogeneity
- Background noise interference
- Errors in text recognized from audio
- Timeout for slow interaction
- Property sheet not updated properly when integrated with our toolkit

Current limitations

- Doesn't give response in ideally real time
- Doesn't properly deal with errors from user input and system internals
- One way conversation, doesn't support constraints and "natural" placement

Video demo of result

- *Local video (4:28)*



```
RECORDING...
Recording Comple
Session Path: pr
================
Query Text: 'why
Detected Intent:
Fulfillment Text
Adding move beha
null null
RECORDING...
Recording Comple
Session Path: pr
================
Query Text: 'all
Detected Intent:
Fulfillment Text
Adding move beha
null null
RECORDING...
Recording Comple
Session Path: pr
================
Query Text: 'why
Detected Intent:
Fulfillment Text
Adding new behav
null null
Canvas saved!
```

Local Disk (C:) › cmusource › 05830 › 830-advanced-ui › talk-ui-project

Name	Date modified	Type
.settings	4/18/2020 4:49 PM	File
credentials	4/29/2020 5:23 PM	File
resources	4/29/2020 6:07 PM	File
src	4/18/2020 4:49 PM	File
target	4/29/2020 10:52 PM	File
.classpath	4/18/2020 4:49 PM	CLA
.gitignore	4/27/2020 7:11 PM	Git I
.project	4/18/2020 4:49 PM	PRC
pom.xml	4/29/2020 4:43 PM	XMl
README.md	4/29/2020 4:43 PM	Mar
saved.png	4/29/2020 11:03 PM	PNG
talk-ui-project.iml	4/29/2020 5:52 PM	IML

13 items | 1 item selected 7.05 KB

saved.png
PNG File

Date taken:	Specify date taken
Dimensions:	600 x 600
Size:	7.05 KB
Date created:	4/29/2020 6:40 PM

Got it, click on what you want to a
allow me to move this now
Got it, click on what you want to a
why allow me to move this now
Here you go, click on what you wa

color	
height	14
interimSelected	False
lineThickness	1
selected	False
width	14
x	0
y	0

Salad

Based on:
Toolkits for Visualization and UIs in Data Science

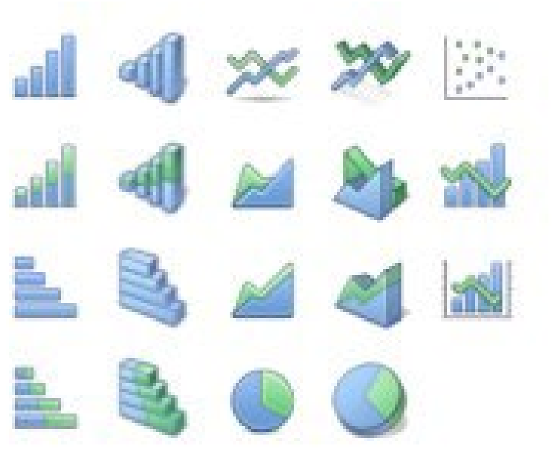
by Dominik Moritz, April 8, 2020

https://dig.cmu.edu

Origins

- Four major influences act on data analysis today:
  1. The formal theories of statistics.
  2. Accelerating developments in computers and display devices.
  3. The challenge, in many fields, of more and ever larger bodies of data.
  4. The emphasis on quantification in an ever wider variety of disciplines.
  - Data Analysis & Statistics. Turkey and Wilk. 1965.
  - Effective Data Visualization. Heer. 2015.

How do people create visualizations?



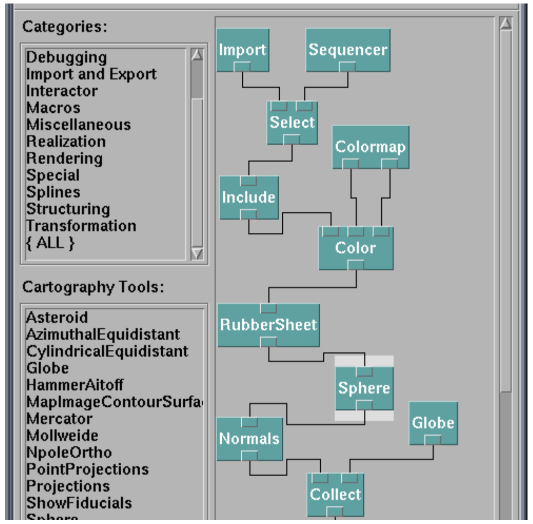


Chart Typology

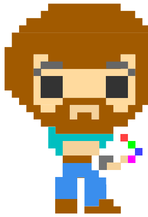
Pick from a stock of templates
Easy-to-use but limited expressiveness
Prohibits novel designs, new data types

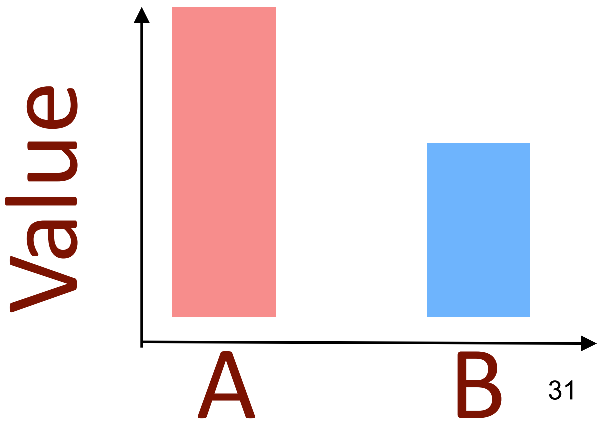
Component Architecture

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

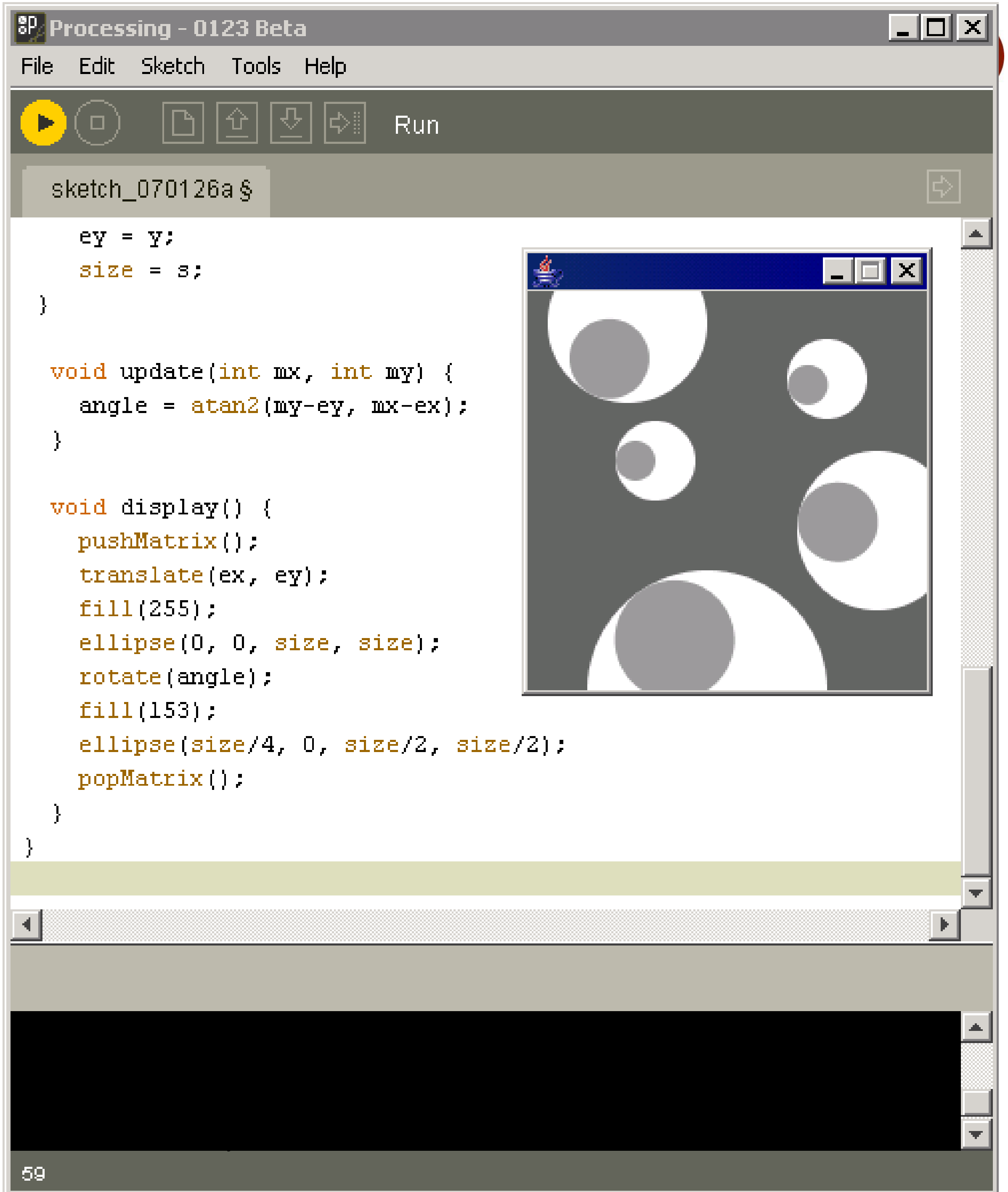
Drawing Visualizations with Imperative Programs

- Graphics APIs: Processing, OpenGL, Java2D, JavaScript/html SVG and Canvas
- Program by giving explicit steps. e.g.:
  - "Put a red bar here and a blue bar there."
  - "Draw a line and some text."
- Specification and execution are intertwined.
- *"You have unlimited power on this canvas. You can literally move mountains."* — Bob Ross





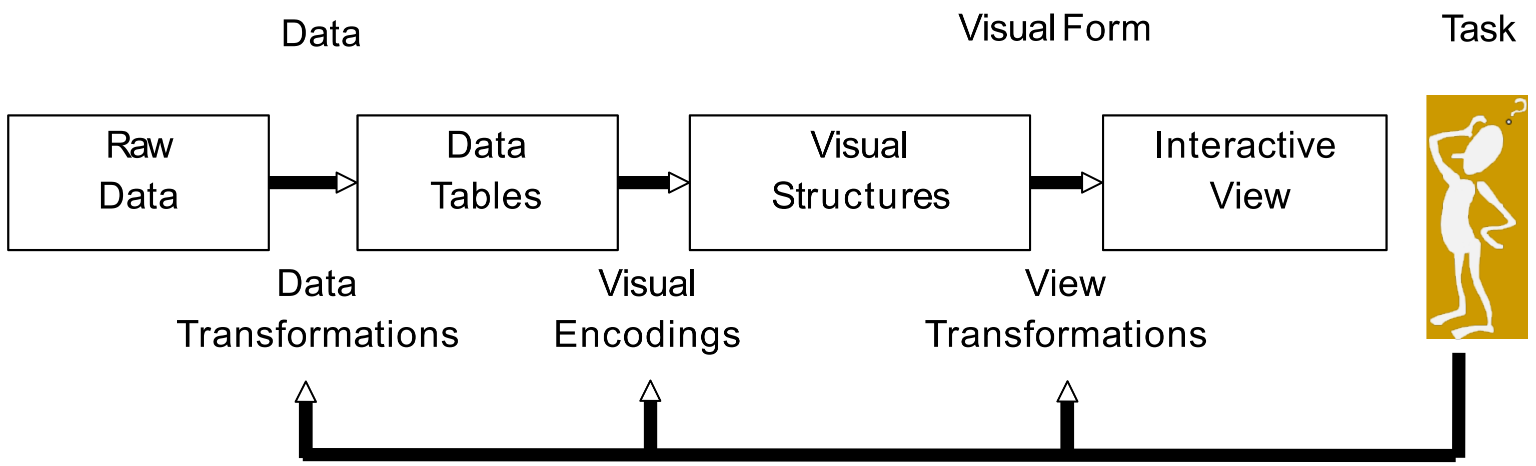
Example: processing.org

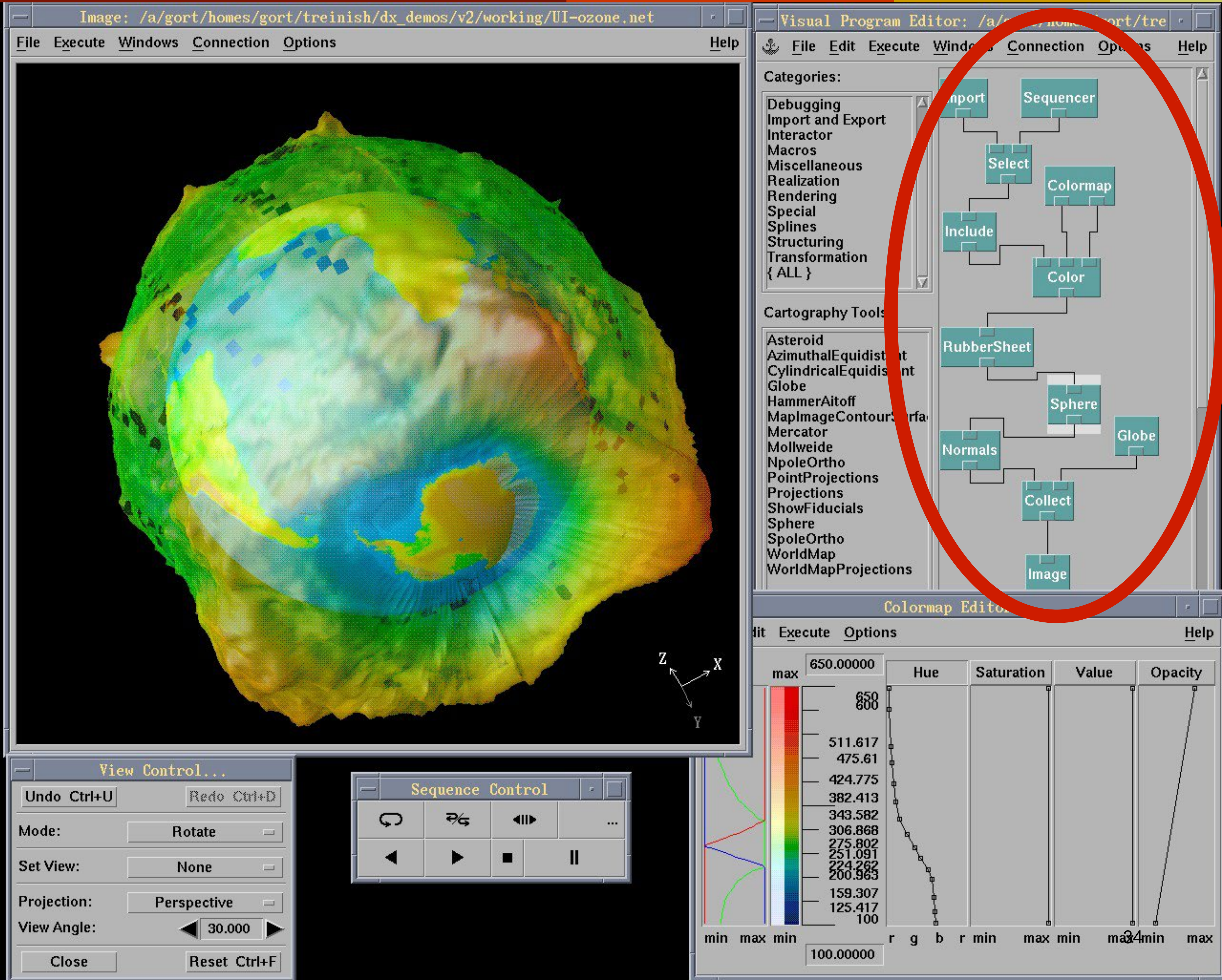


```
    ey = y;
    size = s;
  }

  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();
  }
}
```

Component Architectures

- Component Architectures on top of the graphics APIs
  - Examples: Prefuse, Flare, Improvise, VTK
- Dataflow architecture – wire together nodes

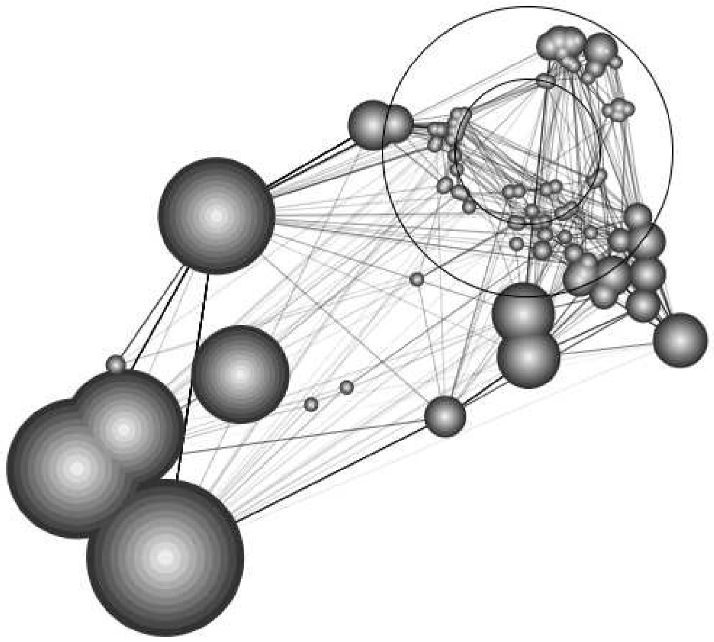


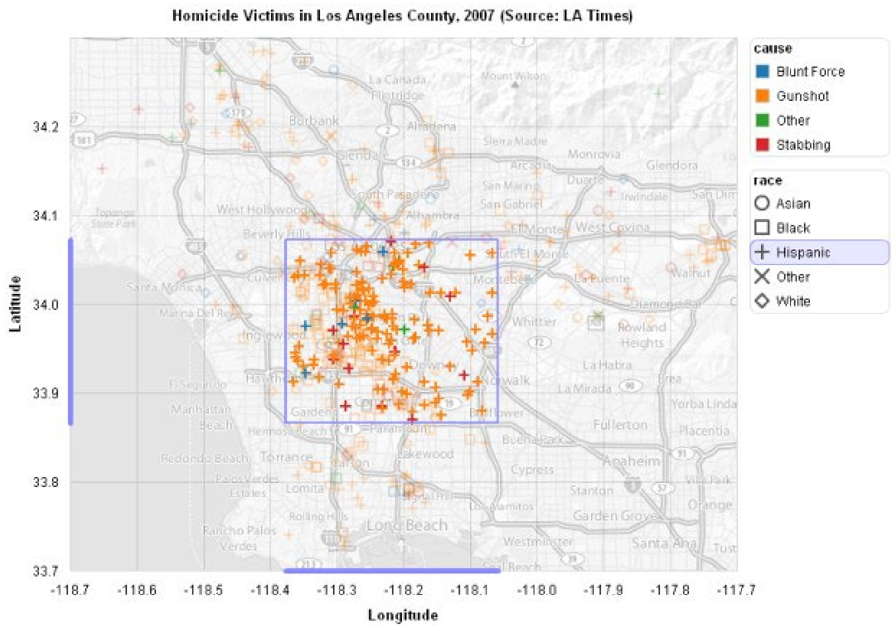


Mode: Rotate
Set View: None
Projection: Perspective
View Angle: 30.000

Prefuse & Flare

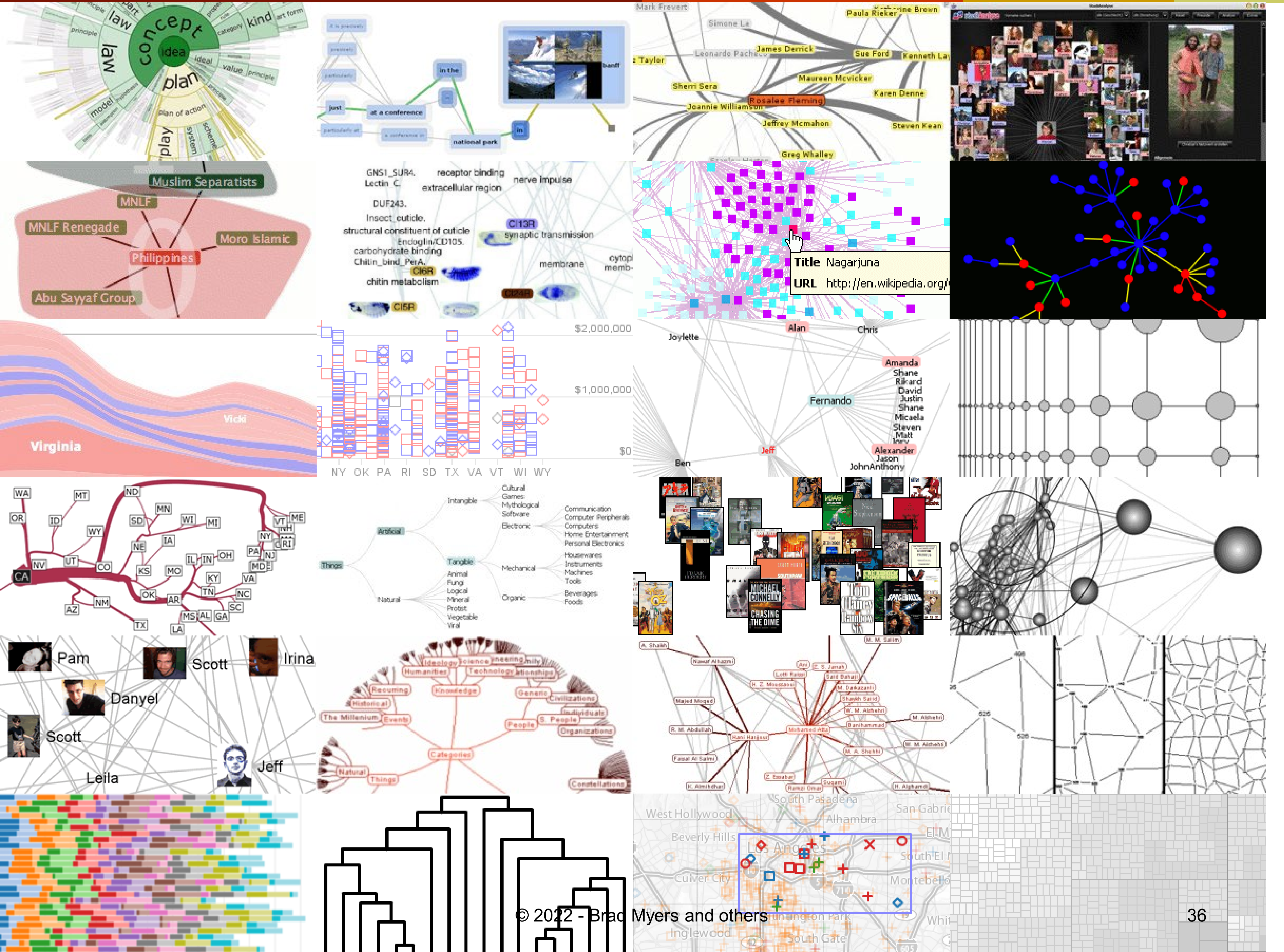
- Operator-based toolkits for visualization design
- Vis = (Input Data -> Visual Objects) + Operators





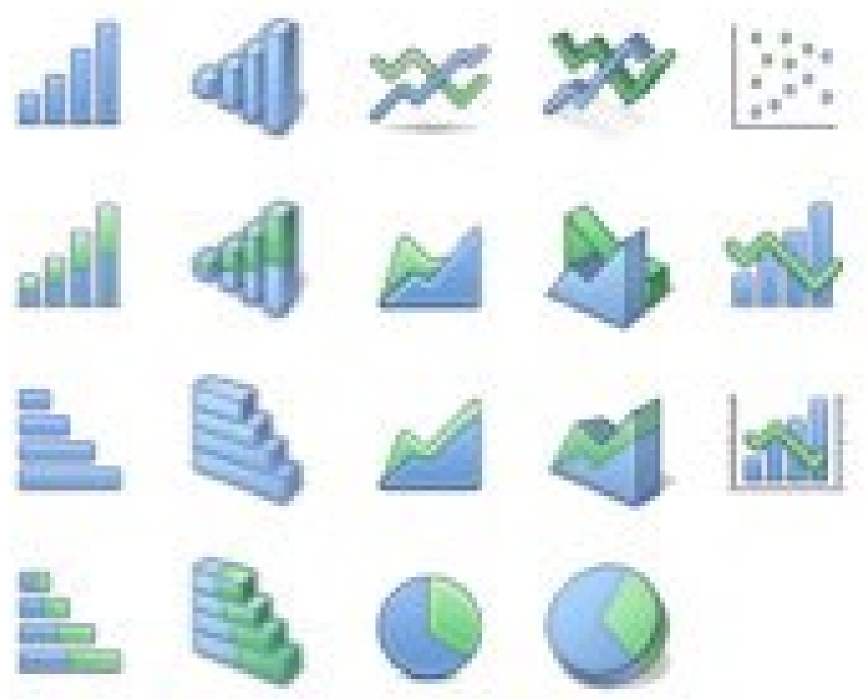
Prefuse (http://prefuse.org)

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

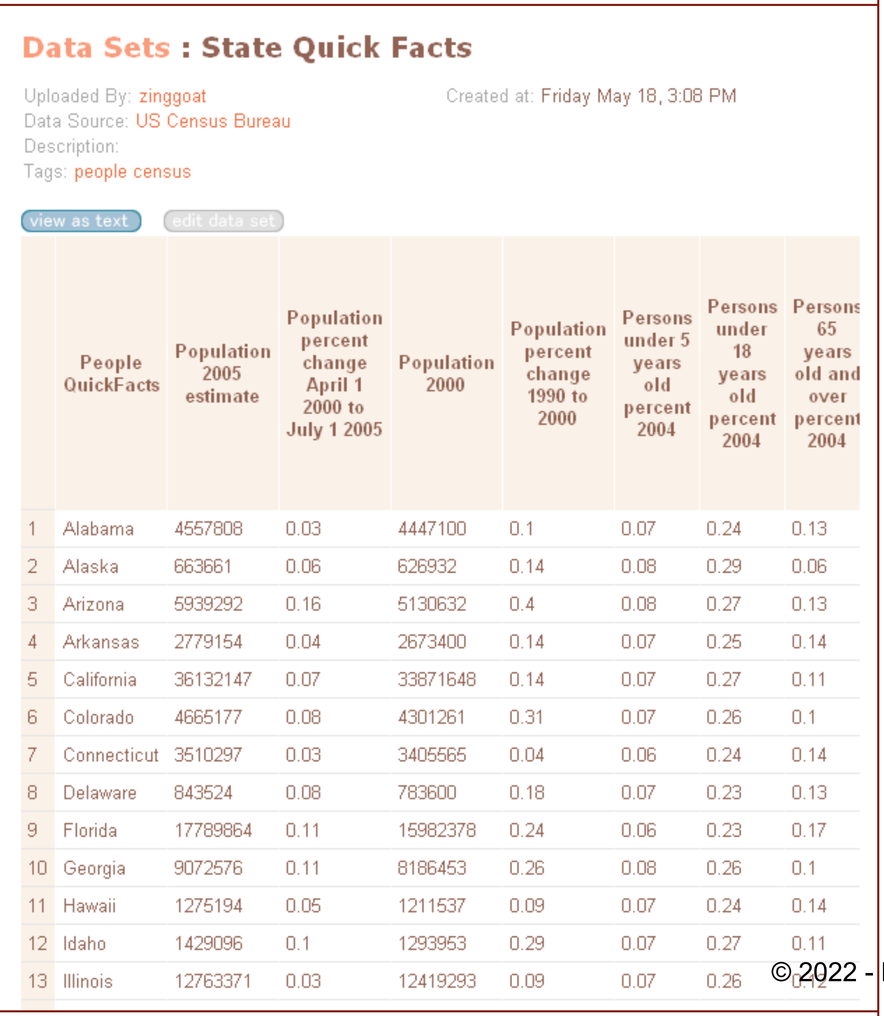

concept
idea
plan
law
kind
model
play
at a conference
national park
Muslim Separatists
MNLF
MNLF Renegade
Moro Islamic
Philippines
Abu Sayyaf Group
synaptic transmission
chitin metabolism
membrane
Title Nagarjuna
URL http://en.wikipedia.org/
Virginia
Vicki
$2,000,000
$1,000,000
$0
NY OK PA RI SD TX VA VT WI WY
Joylette
Alan
Chris
Amanda
Fernando
Jeff
Ben
Alexander
Things
Artificial
Natural
Intangible
Tangible
Mechanical
Organic
Pam
Scott
Irina
Danyel
Leila
Categories
Knowledge
Humanities
People
Mohamed Atta
West Hollywood
Beverly Hills
Culver City
Inglewood
South Pasadena
Alhambra
Montebello
South Gate

Other extreme: Chart Typologies

- Excel, Many Eyes, Google Charts, Tableau



Select from Menus…

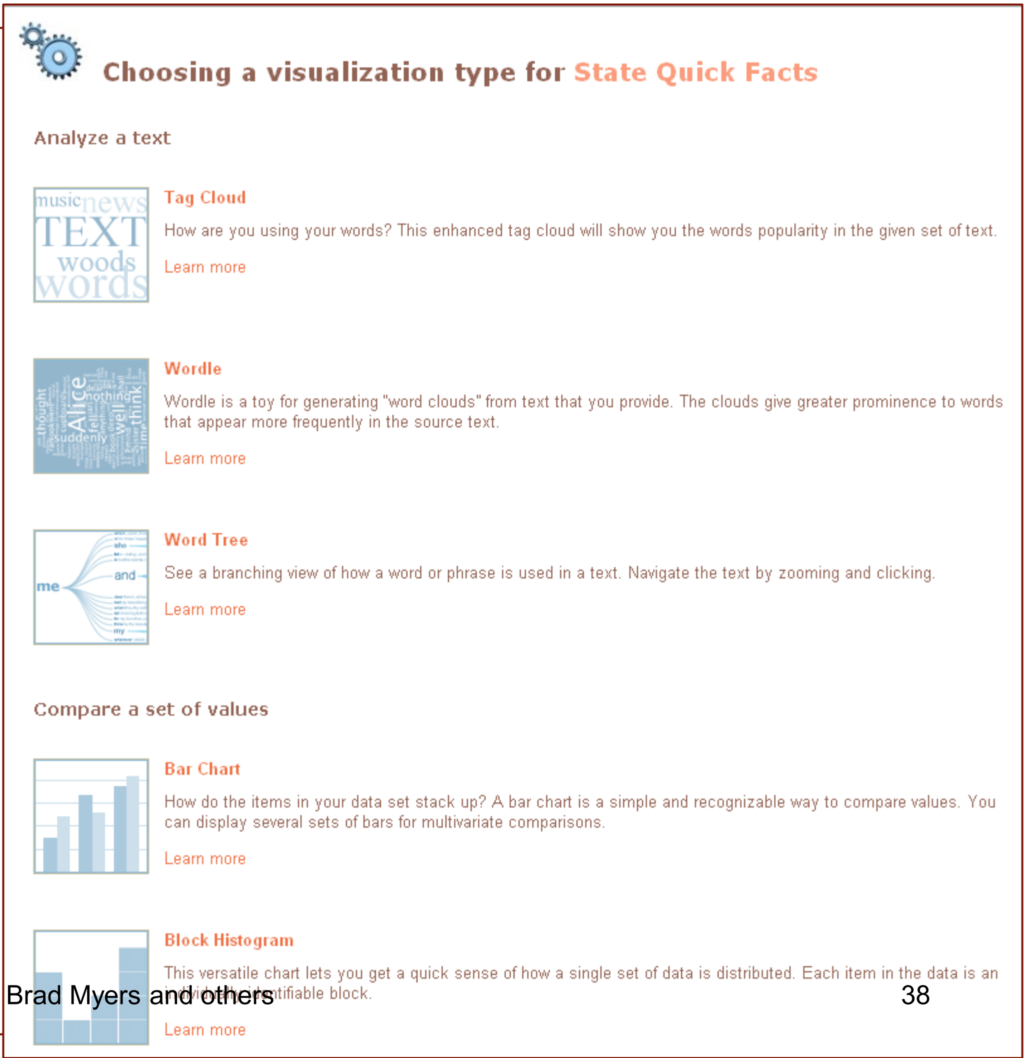


Data Sets : State Quick Facts

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

view as text | edit data set

	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

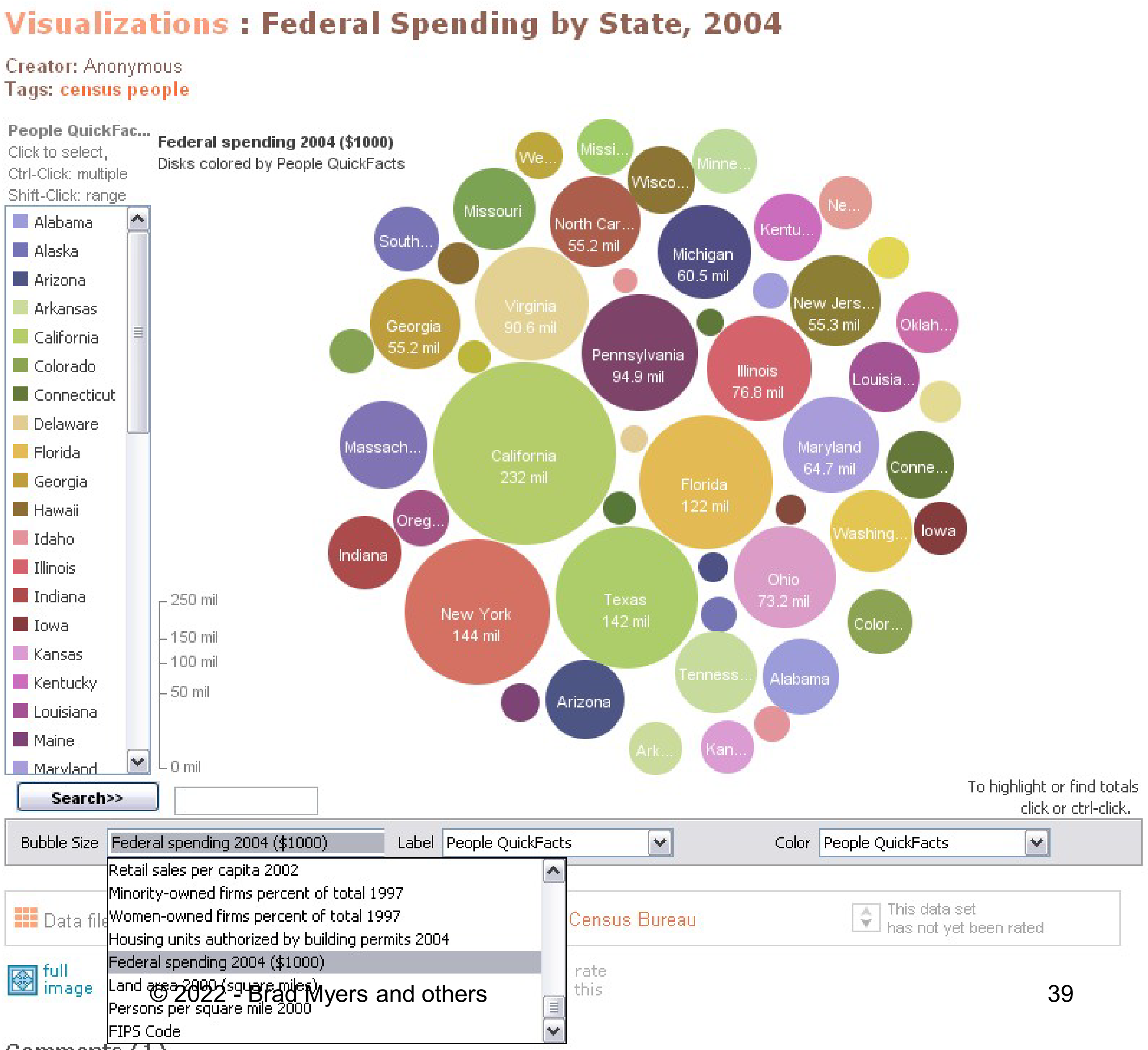
Compare a set of values

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

Results:

- Inflexible
- Can tinker after generated



Visualizations : Federal Spending by State, 2004

Creator: Anonymous
Tags: census people

People QuickFac...
Click to select,
Ctrl-Click: multiple
Shift-Click: range

Federal spending 2004 ($1000)
Disks colored by People QuickFacts

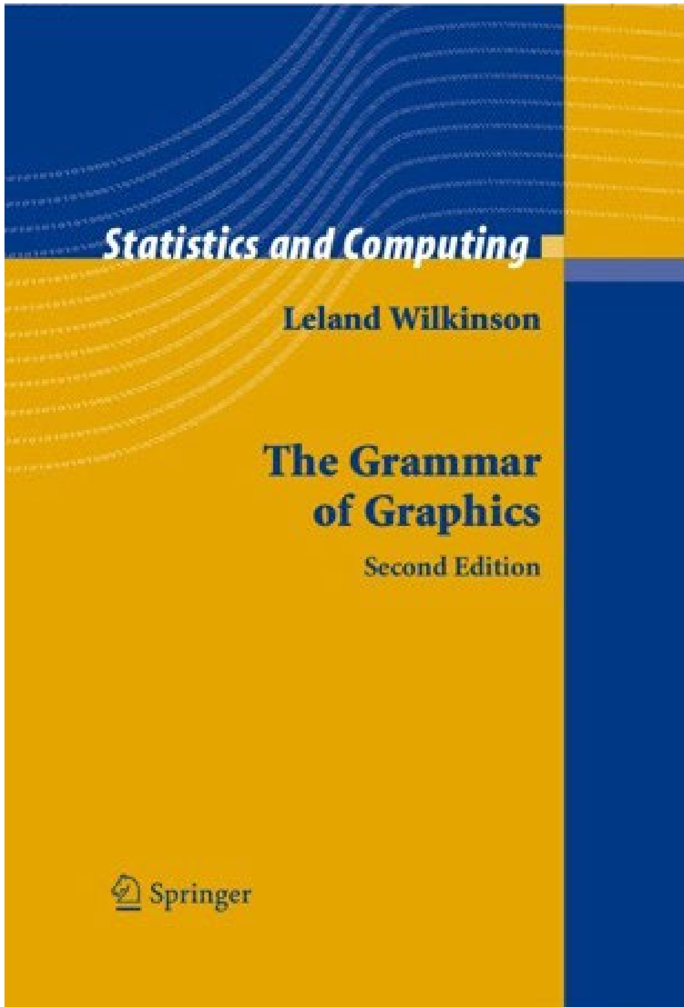
To highlight or find totals click or ctrl-click.

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

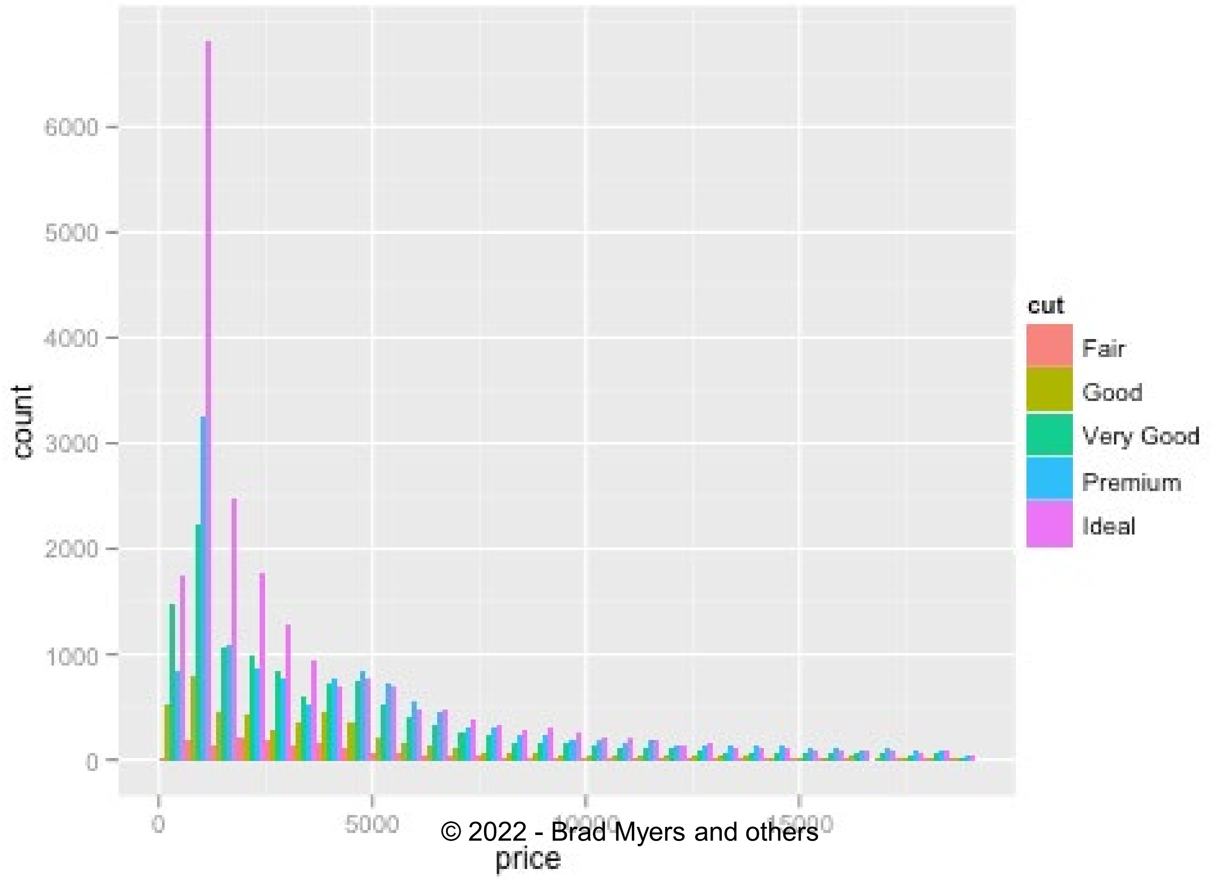
Visual Analysis Grammars

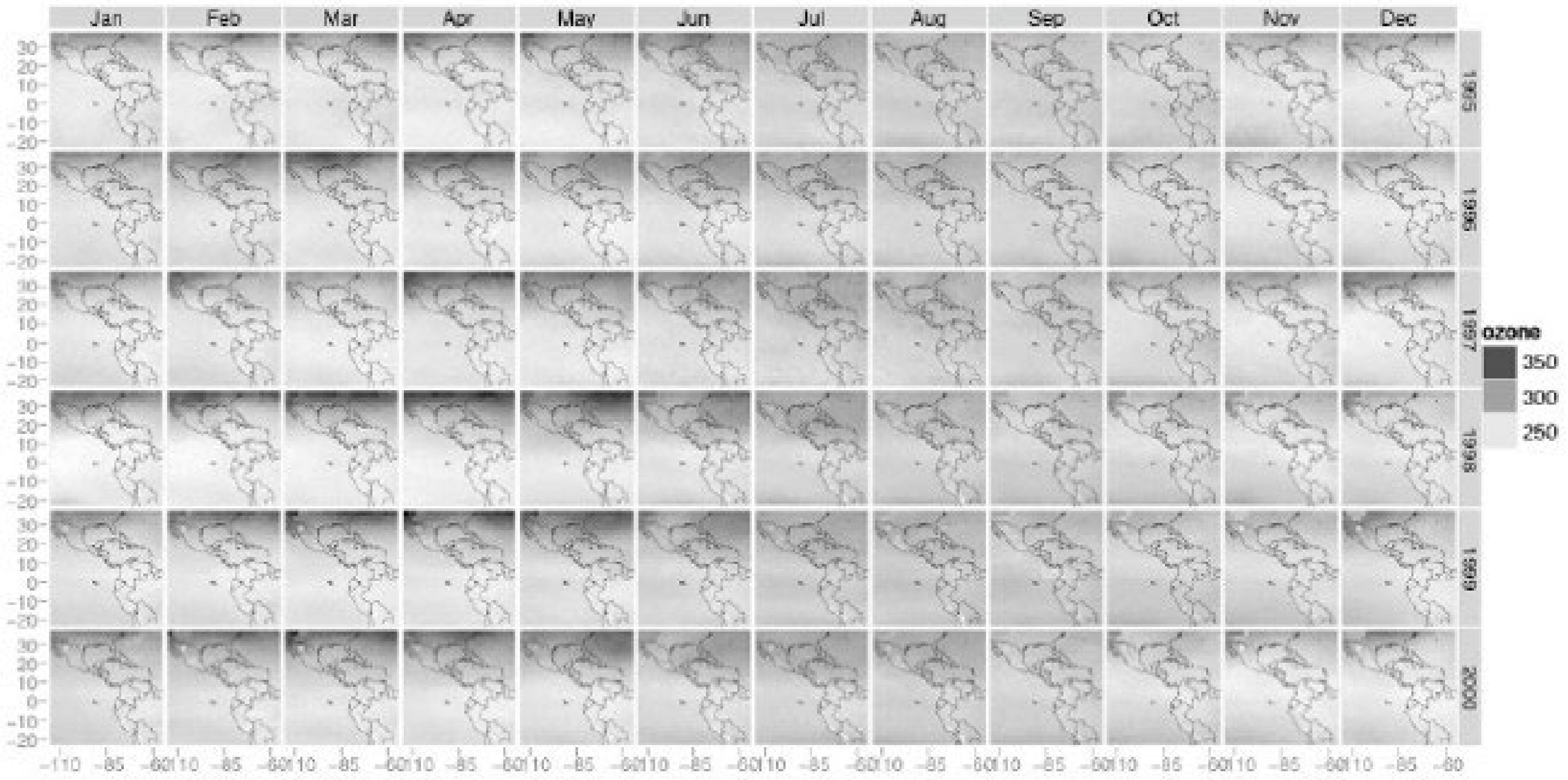
- Examples: VizQL, ggplot2
- Specialized programming language
  - Declarative – what to produce, not how (like html)



Grammar examples

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





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

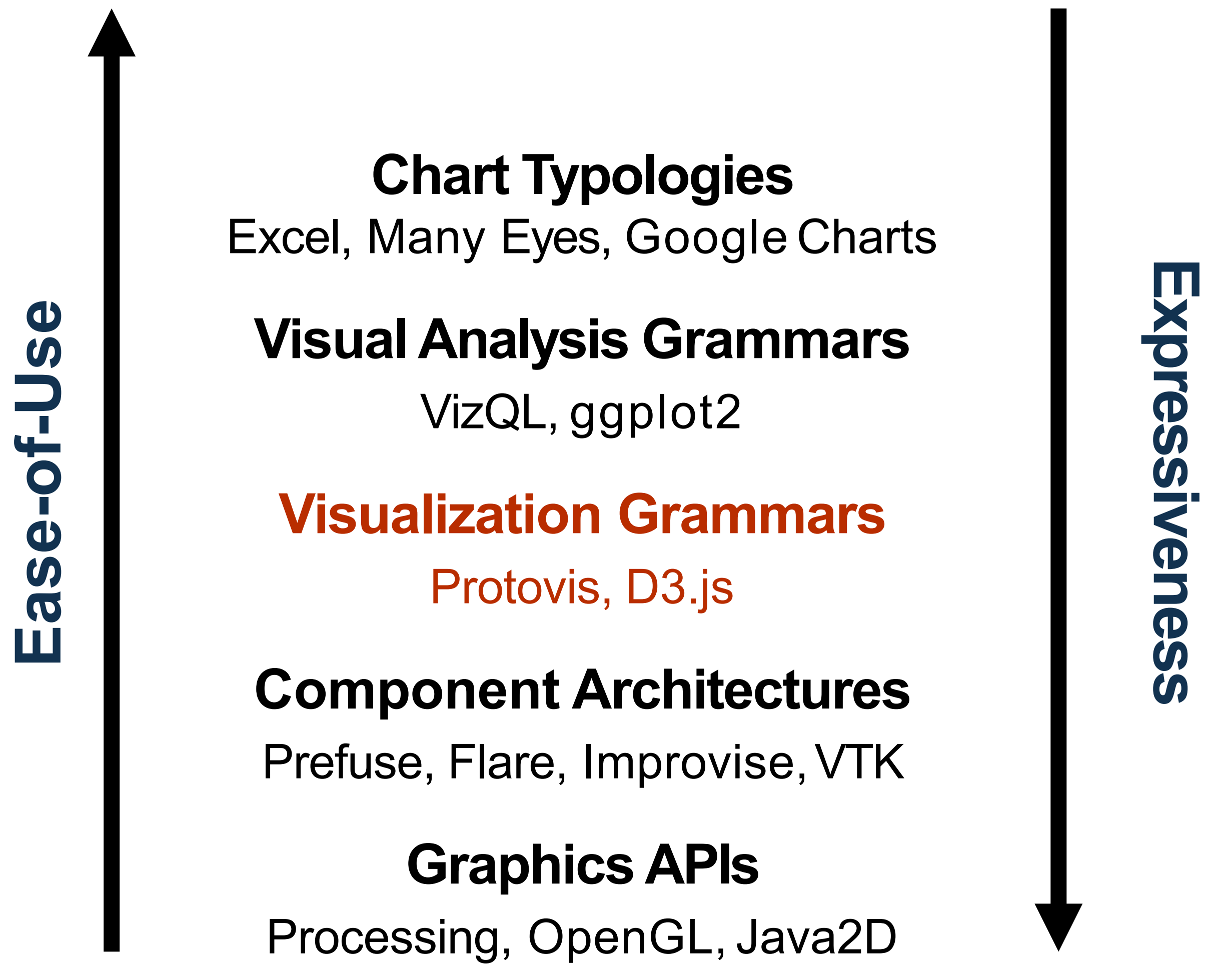


Chart Typologies
Excel, Many Eyes, Google Charts

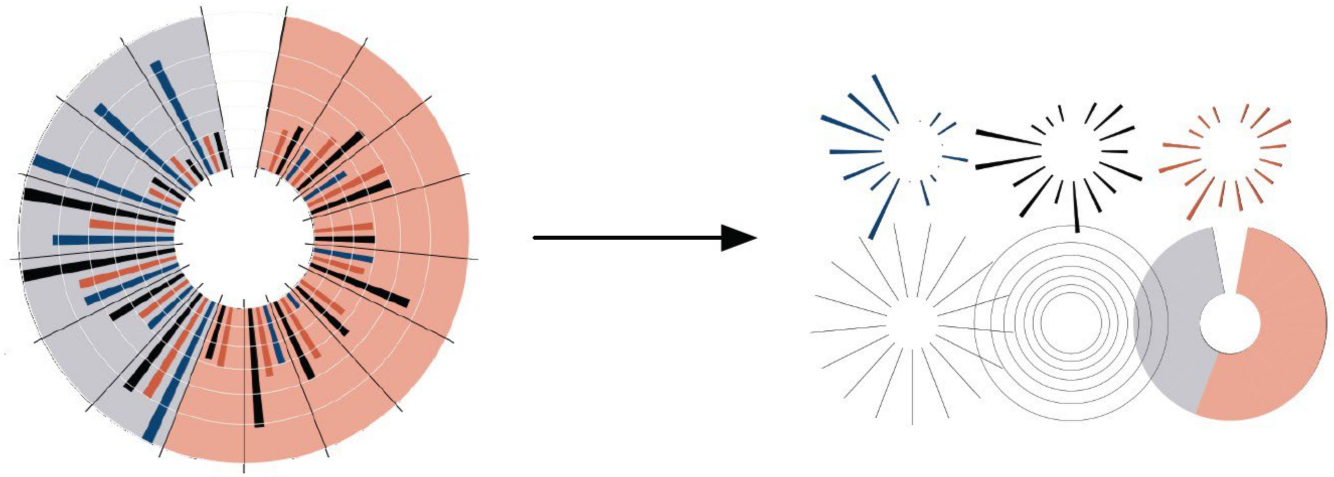
Visual Analysis Grammars
VizQL, ggplot2

Visualization Grammars
Protovis, D3.js

Component Architectures
Prefuse, Flare, Improvise, VTK

Graphics APIs
Processing, OpenGL, Java2D

Protovis: A Grammar for Visualization



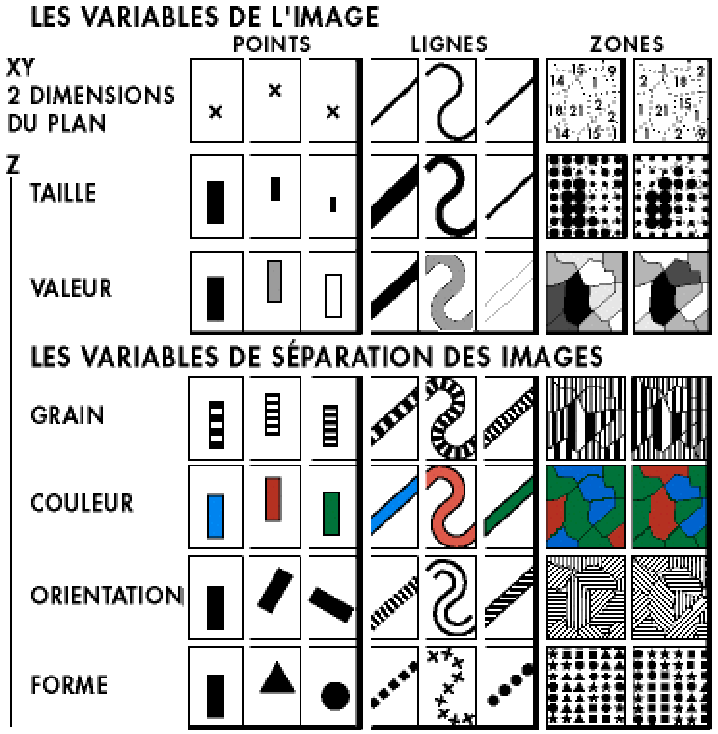
A graphic is a composition of data-representative marks.

Jeffrey Heer, **Mike Bostock** & **Vadim Ogievetsky**

Visualization Grammar

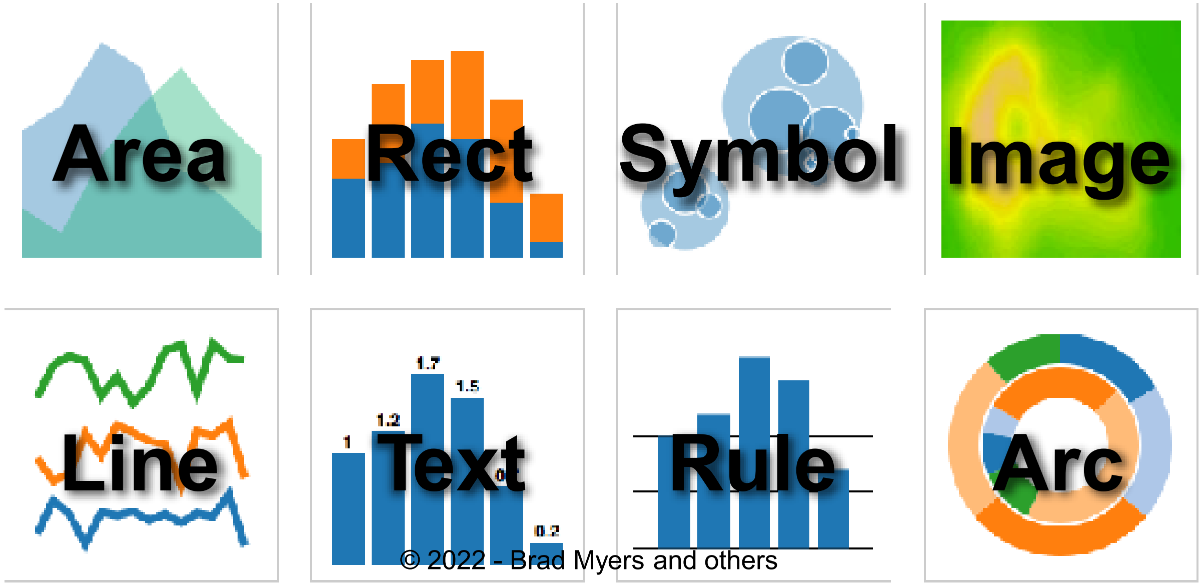
Data	Input data to visualize
Transforms	Grouping, stats, projection, layout
Scales	Map data values to visual values

Jacques Bertin
Sémiologie Graphique, 1967



Visualization Grammar

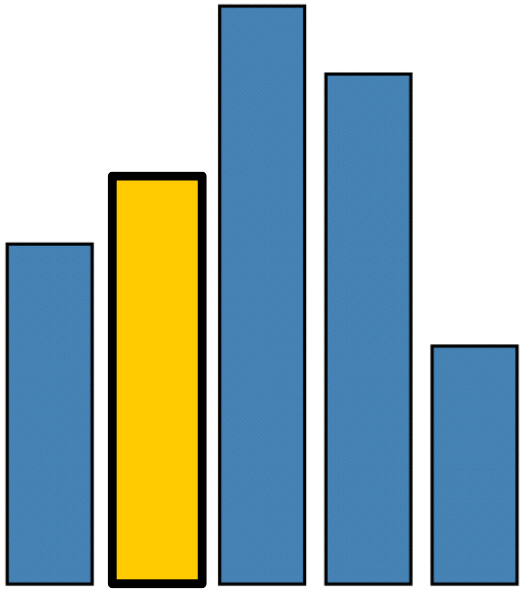
Data	Input data to visualize
Transforms	Grouping, stats, projection, layout
Scales	Map data values to visual values
Guides	Axes & legends visualize scales
Marks	Data-representative graphics



Properties of a "Mark"

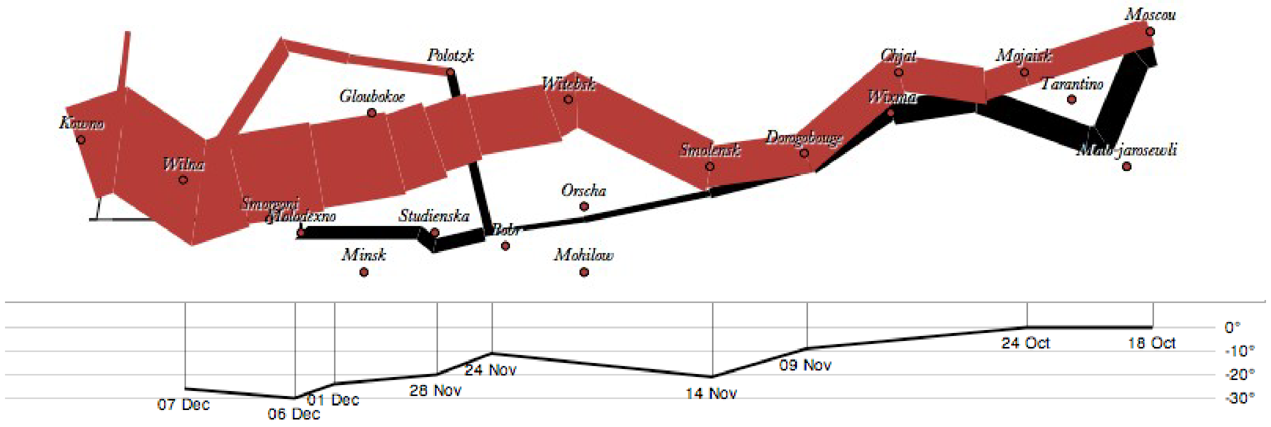
RECT **λ : D → R**

data	1	1.2	1.7	1.5	0.7
visible	true				
left	1 * 25				
bottom	0				
width	20				
height	1.2 * 80				
fillStyle	blue				
strokeStyle	black				
lineWidth	1.5				
…	…				



```
var vis =   newpv.Panel();
vis.add(pv.Bar)
   .data([1, 1.2, 1.7, 1.5, 0.7])
   .visible(true)
   .left((d) = > this.index *  25)
   .bottom(0)
   .width(20)
   .height((d) = >  d  * 80)
   .fillStyle("blue")
   .strokeStyle("black")
   .lineWidth(1.5);
vis.render();
```

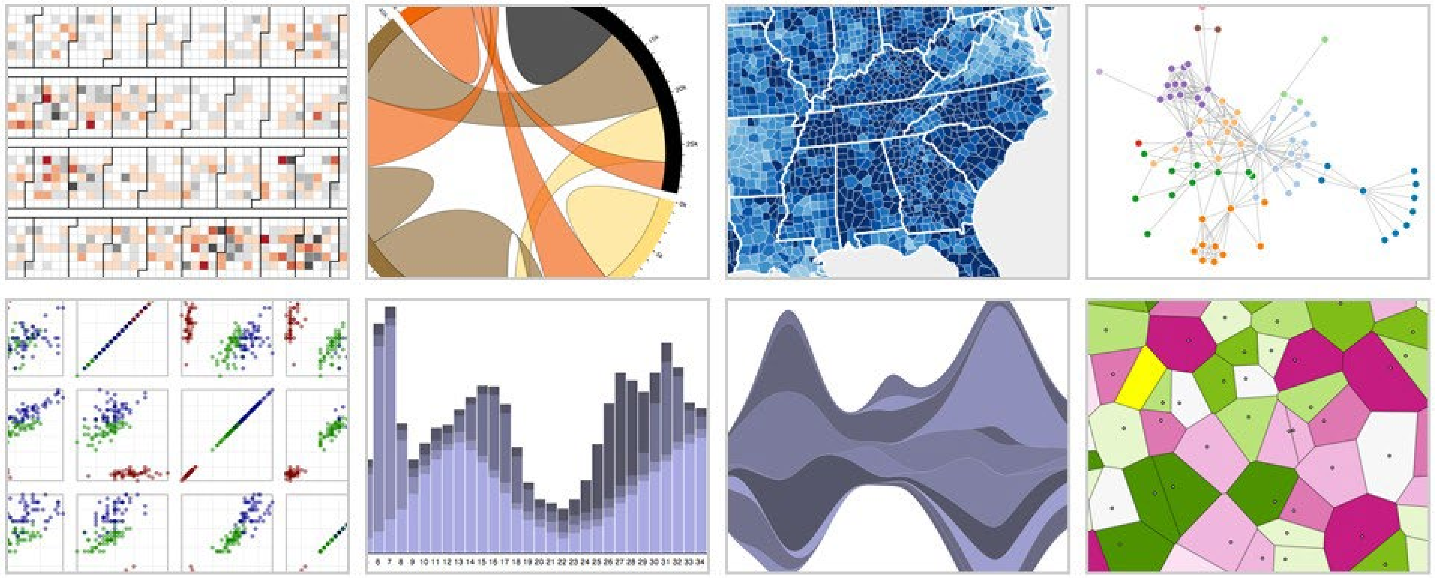
Minard 1869: Napoleon's March, in ProtoViz



```
var army = pv.nest(napoleon.army, "dir", "group“);
var vis =  newpv.Panel();

var lines = vis.add(pv.Panel).data(army);
lines.add(pv.Line)
  .data(() => army[this.idx])
  .left(lon).top(lat).size((d) => d.size/8000)
  .strokeStyle(() => color[army[panelIndex][0].dir]);

vis.add(pv.Label).data(napoleon.cities)
  .left(lon).top(lat)
  .text((d) => d.city).font("italic 10pxGeorgia")
  .textAlign("center").textBaseline("middle");
```

```
vis.add(pv.Rule).data([0,-10,-20,-30])
  .top((d) => 300 - 2*d - 0.5).left(200).right(150)
  .lineWidth(1).strokeStyle("#ccc")
  .anchor("right").add(pv.Label)
   .font("italic 10px Georgia")
   .text((d) => d+"°").textBaseline("center");

vis.add(pv.Line).data(napoleon.temp)
  .left(lon).top(tmp) .strokeStyle("#0")
 .add(pv.Label)
   .top((d) => 5 + tmp(d))
   .text((d) => d.temp+"° "+d.date.substr(0,6))
```

d3.js: Data-Driven Documents



Mike Bostock, Dominik Moritz, Vadim Ogievetsky, Jeff Heer, etc.

Protovis vs. D3

Protovis

Specialized mark types

+ Streamlined design
- Limits expressiveness
- More overhead (slower)
- Harder to debug
- Self-contained model

Specify a scene (nouns)

+ Quick for static vis
- Delayed evaluation
- Animation, interaction are more cumbersome

D3

Bind data to DOM

- Exposes SVG/CSS/…
+ Exposes SVG/CSS/…
+ Less overhead (faster)
+ Debug in browser
+ Use with other tools

Transform a scene (verbs)

- More complex model
+ Immediate evaluation
+ Dynamic data, anim, and interaction natural

D3 Selections

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

```
// Add and configure an SVG element
var 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 Binding

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

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

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

// 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();
```

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, …)