

## Lecture 12: Toolkits

### Information Visualization CPS3 533C, Fall 2011

Tamara Munzner  
UBC Computer Science

Wed, 19 October 2011

## Required Readings

Prefuse: A Toolkit for Interactive Information Visualization. Jeffrey Heer, Stuart K. Card, James Landay. Proc ACM CHI, 421-430, 2005.

Protovis: A Graphical Toolkit for Visualization. Michael Bostock and Jeffrey Heer. IEEE Trans. Visualization & Computer Graphics (Proc. InfoVis), 2009.

D3: Data-Driven Documents. Michael Bostock, Vadim Ogievetsky, Jeffrey Heer. IEEE Trans. Visualization & Computer Graphics (Proc. InfoVis), 2011.

## Further Reading

Readings in Information Visualization: Using Vision To Think, Chapter 1. Stuart K. Card, Jock Mackinlay, and Ben Shneiderman. Morgan Kaufmann, 1999.

A Taxonomy of Visualization Techniques using the Data State Reference Model. Ed H. Chi. Proc. InfoVis 2000.

Wrangler: Interactive Visual Specification of Data Transformation Scripts. Sean Kandel, Andreas Paepcke, Joseph Hellerstein, Jeffrey Heer. Proc. CHI 2011.

## Toolkits

- imperative: how
  - low-level rendering: Processing, OpenGL
  - parameterized visual objects: prefuse
    - also rare: prefuse for Flash
- declarative: what
  - Protovis, D3, ggplot2
  - separation of specification from execution
- considerations
  - expressiveness
    - can I build it?
  - efficiency
    - how long will it take?
  - accessibility
    - do I know how?

## OpenGL

- graphics library
  - pros
    - power and flexibility: complete control for graphics
    - hardware acceleration
    - many language bindings: C, C++, Java (w/ JOGL)
  - cons
    - big learning curve if you don't know already
    - no vs support, must roll your own everything
  - example app: TreeKantoposter



[Fig 6. Munzner et al. TreeKantoposter: Scalable Tree Visualization using OpenGL in Concert with Guaranteed Visibility. Proc SIGGRAPH 2011, pp. 453-462.]

## Processing

- layer on top of Java/OpenGL
- visualization esp. for artists/designers
- pros
  - great sandbox for rapid prototyping
  - huge user community, great documentation
- cons
  - poor widget library support
- example app: MixBee



[Fig 1. Meyer et al. MixBee: A Multiscale Synthesis Browser. Proc. InfoVis 2008.]

## prefuse

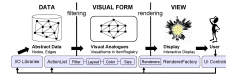
- infovis toolkit, in Java
- fine-grained building blocks for tailored visualizations
- pros
  - heavily used
  - very powerful abstractions
  - quickly implement most techniques covered so far
- cons
  - no longer under active development
  - nontrivial learning curve
- example app: DOITrees Revisited



[Fig 3. Heer, Card, and Landay. Prefuse: A Toolkit for Interactive Information Visualization. Proc. CHI 2005, 472-483.]

## Prefuse

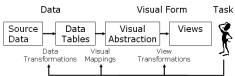
- separation: abstract data, visual form, view
  - data: tables, networks (nodes, edges)
  - visual form: layout, color, size, ...
  - view: multiple renderers



[Fig 2. Heer, Card, and Landay. Prefuse: A Toolkit for Interactive Information Visualization. Proc. CHI 2005, 473-481.]

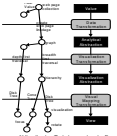
## InfoVis Reference Model

- conceptual model underneath prefuse design
- heavily influenced much of infovis (incl nested model)
  - aka infovis pipeline, data state model [Chi99]



[Replaces Fig 1.23. Card, Mackinlay, and Shneiderman. Readings in Information Visualization: Using Vision To Think, Chapter 1. Morgan Kaufmann, 1999.]

## Data State Model



[Fig 2. Chi: A Taxonomy of Visualization Techniques using the Data State Reference Model. Proc. InfoVis 2000.]

## Prefuse Design Implications

- separating abstraction, visual form, view
  - supports linked multiple views
  - supports novel visual encoding design
- actions: operator composition
  - supports distortion: layout modification
  - supports animated transitions
- multiple renderers
  - supports semantic zooming
- many/most common methods well supported
  - abstractions map well to infovis concerns
  - nevertheless takes time to wrap head around it
  - good choice for local app

## Prefuse Validation

- wide set of old/new app examples
  - expressiveness, effectiveness, scalability
- qualitative usability for system API
  - nice methodology?
  - for specific application

## Declarative Toolkits

- imperative: toolkits/libraries discussed so far
  - say exactly how to do it
  - familiar programming model
- declarative: other possibility
  - just say what to do
  - Protovis, D3

## Protovis

- declarative infovis toolkit, in JavaScript
  - (also later Java version)
  - marks with inherited properties
- pros
  - runs in browser
  - matches mark/channel mental model
  - also much more: interaction, geospatial, trees, ...
- cons
  - not all kinds of operations supported
- example app: NapkinVis (2009 course project)



[Fig 1. 3. Chao. NapkinVis. http://www.cs.ubc.ca/~chao/visover/533-09/projects.html#w/01]

## Protovis Validation


- wide set of old/new app examples
  - expressiveness, effectiveness, scalability
  - accessibility
- analysis with cognitive dimensions of notation
  - closeness of mapping, hidden dependencies,
  - role-expressiveness, visibility, consistency,
  - viscosity, diffuseness, abstraction,
  - hard mental operations

## D3

- declarative infovis toolkit, in JavaScript
- Protovis meets Document Object Model / CSS
  - seamless interoperability with Web
  - explicit transforms of scenes with dependency info
- cons
  - even more different from traditional programming model
- example app: calendar (gallery of many)



[Fig 3. Bostock, Ogievetsky, and Heer. D3: Data-Driven Documents. Proc. InfoVis 2011.]

<h3>D3 Validation</h3> <ul style="list-style-type: none"> <li>■ wide set of old/new app examples</li> <li>■ software performance           <ul style="list-style-type: none"> <li>■ initialization, frame rates</li> </ul> </li> </ul>	<h3>Paper Types</h3> <ul style="list-style-type: none"> <li>■ design studies</li> <li>■ technique/algorithm</li> <li>■ evaluation</li> <li>■ model/taxonomy</li> <li>■ <b>system</b> <ul style="list-style-type: none"> <li>■ today's emphasis</li> </ul> </li> </ul>	<h3>ggplot2</h3> <ul style="list-style-type: none"> <li>■ declarative statistical graphics in R</li> <li>■ implementation of Wilkinson's Grammar of Graphics</li> </ul>  <p>[Slide 17, Wickham: ggplot2 past, present, and future. <a href="http://had.co.nz/ggplot2/resources/2007-past-present-future.pdf">http://had.co.nz/ggplot2/resources/2007-past-present-future.pdf</a>]</p>	<h3>Wrangler</h3> <ul style="list-style-type: none"> <li>■ interactive data cleaning</li> <li>■ inference engine: system suggests applicable transforms</li> <li>■ programming by demonstration (vs complex regexps)</li> <li>■ declarative transformation language underlying</li> <li>■ <a href="http://vis.stanford.edu/wrangler/">http://vis.stanford.edu/wrangler/</a></li> </ul>
<h3>Systems</h3> <ul style="list-style-type: none"> <li>■ Tableau: general/powerful database vis</li> <li>■ Mondrian: statistical graphics</li> <li>■ ggobi: high-dimensional analysis</li> <li>■ load data directly, as opposed to build with toolkit</li> </ul>	<h3>Resource Page: Software</h3> <p><a href="http://www.cs.ubc.ca/tmm/courses/533-11/resources.html">http://www.cs.ubc.ca/tmm/courses/533-11/resources.html</a></p>	<h3>Reading For Next Time</h3> <ul style="list-style-type: none"> <li>■ <b>Mon Oct 31:</b> no class next week!</li> </ul> <p>Graph Visualization in Information Visualization: a Survey. Ivan Herman, Guy Melançon, M. Scott Marshall. IEEE Transactions on Visualization and Computer Graphics, 6(1), pp. 24-44, 2000</p> <p>Topological Fish-eye Views for Visualizing Large Graphs. Emdin Gansner, Yehuda Koren and Stephen North. IEEE TVCG 11(4):457-468 (Proc. InfoVis 2005), 2005.</p> <p>Online Dynamic Graph Drawing. Yaniv Fishman and Ayellet Tal. Proc EuroVis 2007, 75-82.</p>	<h3>Reminders</h3> <ul style="list-style-type: none"> <li>■ this Friday: presentation topics due</li> <li>■ next Friday: written project proposals due</li> <li>■ next week: no class</li> </ul>