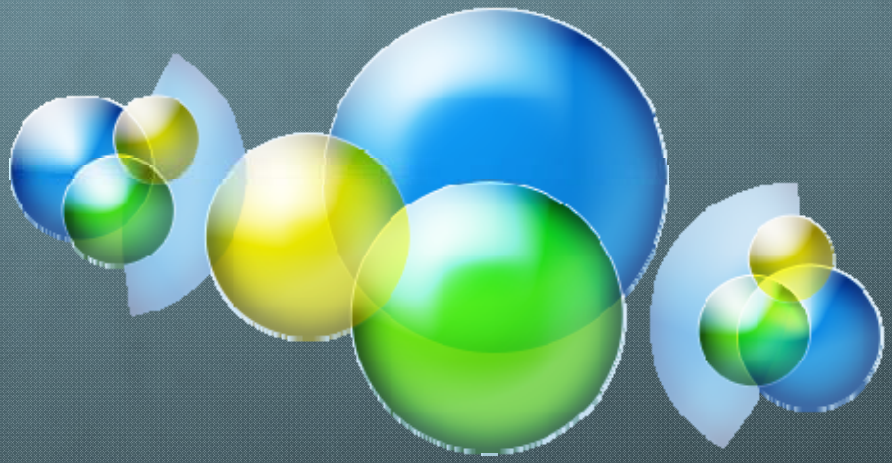
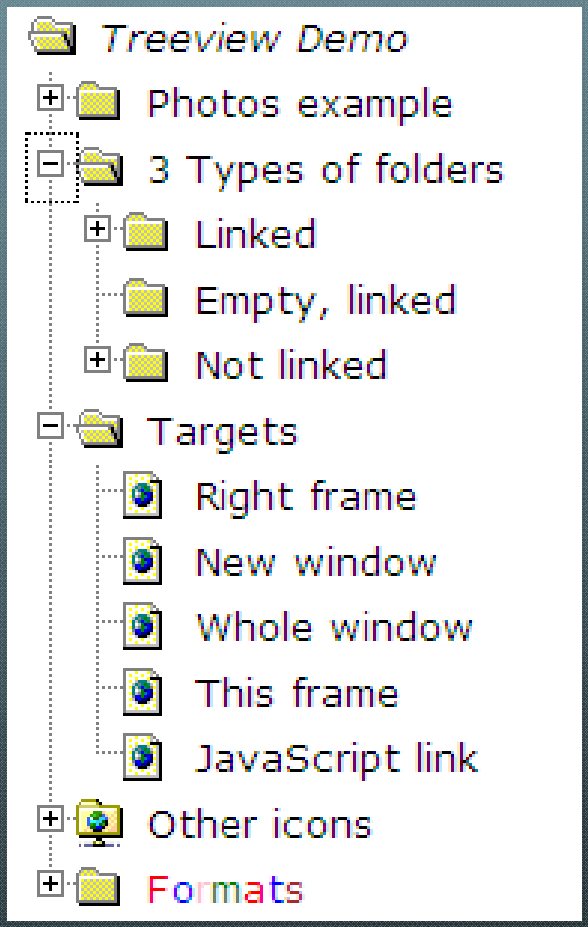


Highly Interactive Tree-view

Johnny Zhou & Juho Kim

CS349W – Autumn 2008

Motivation



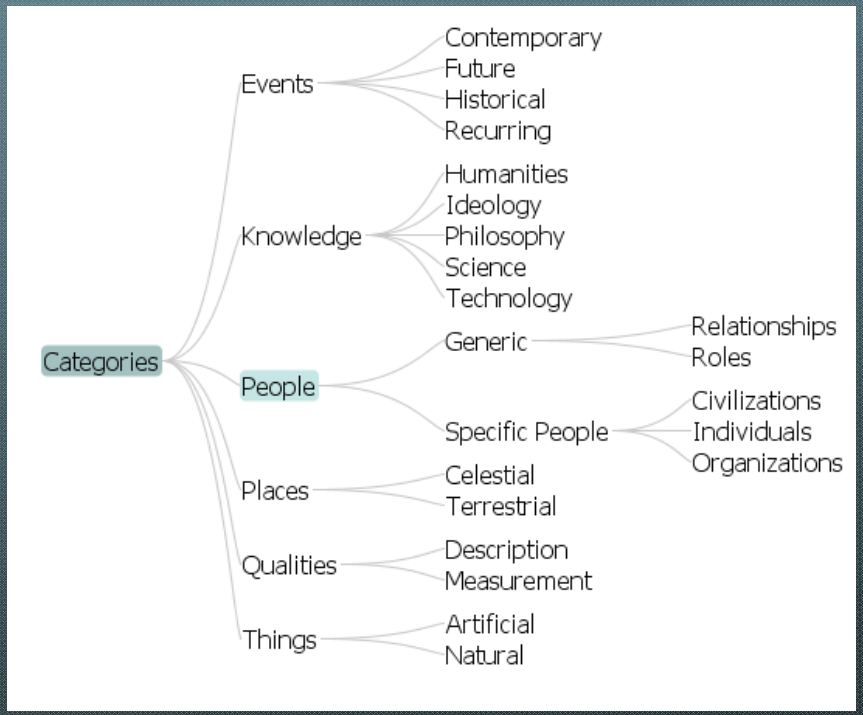
Traditional treeview

- Easy navigation
- Intuitive
- Not fun or interactive enough?

http://www.treeview.net/tv/demo/demoFrameset.html

Motivation

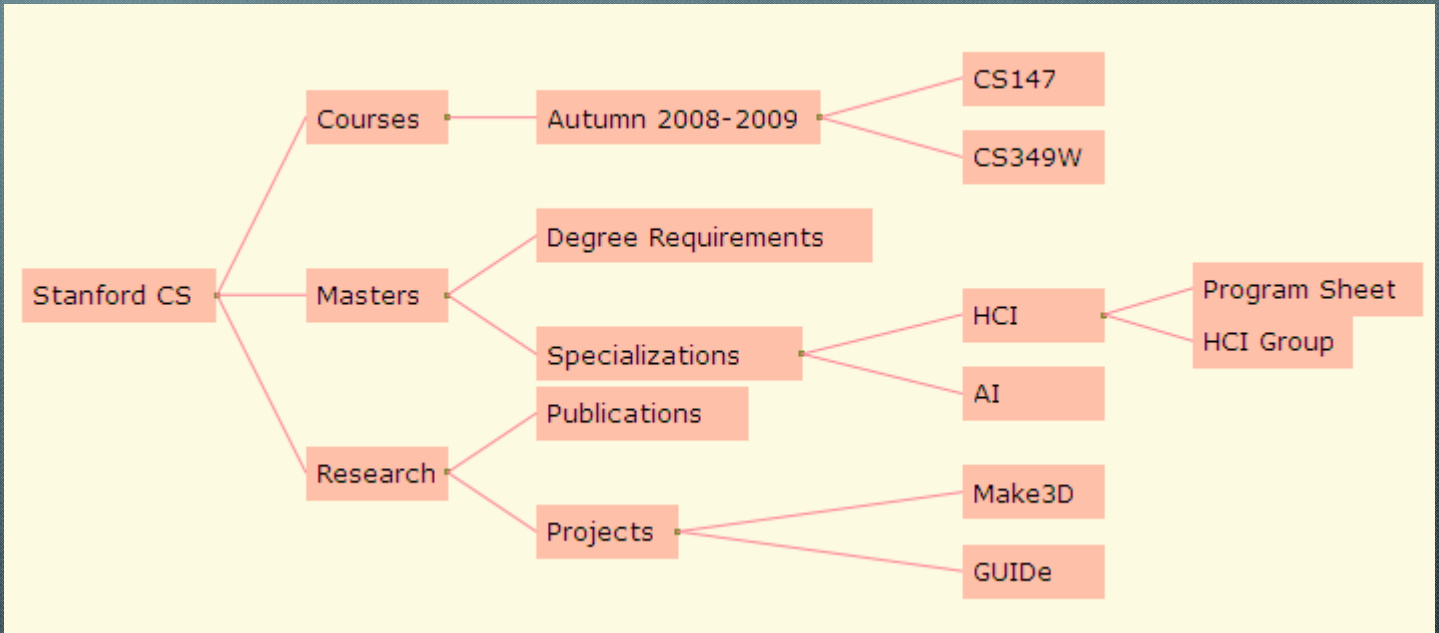
Prefuse Treeview



- Cool interaction
- Dynamic expand/shrink
- No editing possible
- Only in Java / Flash

http://prefuse.org/gallery/treeview/

Interactive Treeview



- Direct manipulation
- Node editing
- No Java/Flash/SilverLight

Features

- XML-represented tree
- Link-associated
- Expanding / Collapsing
- Drag-and-Drop
- Zoom and Panning
- Zoom to Fit
- Dynamically adjusting node sizes

Implementation

- Processing -> Javascript conversion
- Processing API ported
- Using Canvas element

Implementation (Base)

HTML5 Canvas Element

- Pixel-level access, drawing
- Standard, supported by latest Gecko and Webkit engines

Processing

- Open-source programming language
- Used for visualization, sketching, animation
- API for drawing shapes, text, images, colors, callbacks

Implementation (Tree)

Tree Data

- Parse XML file from arbitrary location
- Build JavaScript objects corresponding to each tree node

Display

- Each node drawn as a rectangle
- Position & size of nodes dynamically updated upon user interaction
- Javascript object manipulation for dynamic node data update

Reusability

Universal Usage

- Compatible with any existing JavaScript library or full web application frameworks
- Customization of tree data via input XML

Requirements

- Include processing.js, treeview.js
- Create empty canvas element with id="canvas"
- Write your XML file representing your tree
  - Need to have certain elements in XML schema
- onLoad="displayTree(XMLFile)"

Reusability (cont.)

Distribution

- Project Website: http://code.google.com/p/interactive-treeview/
- Javascript files, documentations, examples

API functions

- Tree customization
- Various colors
- Font
- Size

Reusability (Cont.)

Data Freedom

- Input XML file completely defines the tree
- Each node is free to have any data
- Sample XML:

```
<node>
        <label>… </label>
        <url>… </url>
        <someData>… </someData>
        <children>
                <node>… </node>
        </children>
</node>
```

Strengths

Very well suited for navigation

- Also can be used as a generic tree-view
- Nodes can contain arbitrary data
- Potential to serve as foundation for unforeseen features

Reusability

- XML data -> create canvas element -> function call
- Wide array of customization options
- HTML5 Standard

Desktop-like, direct manipulative interface

- Brings the traditional DHTML tree-view to the Web 2.0 world
- Instant user feedback and response

Weaknesses

Not universally supported

- Firefox 3.0+ (Gecko 1.9+), Webkit Nightly, Opera 9.5+ officially work

Too “paint” oriented

- Text must be drawn
- No DOM structure

Scalability issues

- Too many nodes results in clutter
- Potentially significant slowdown in draw times

Real Estate requirements

- Tree takes up a lot of space
- Could benefit from a smart node spacing algorithm

Conclusions

Very useful for navigation interfaces and site maps

- Also can be used as a generic tree-view

Built with JavaScript and the canvas element

- Processing.js port
- Standards supported

Very easily reused

- Encapsulated in its own JavaScript file, easy to hook into
- Data freedom and many appearance customizations

New age tree-view

- Highly interactive and intuitive
- Potential for many applications