Information Visualization

A Tour through the Visualization Zoo

&

html5

Ming-Te Chi

Reading

- **A Tour through the Visualization Zoo**

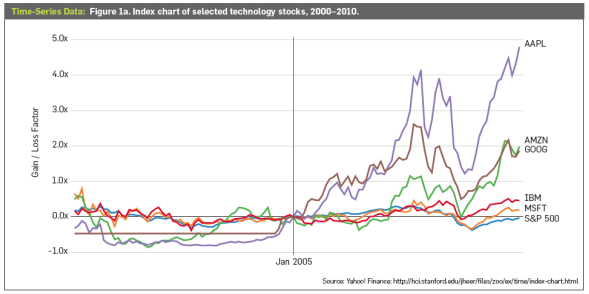
Jeffrey Heer, Michael Bostock, and Vadim Ogievetsky, Stanford University

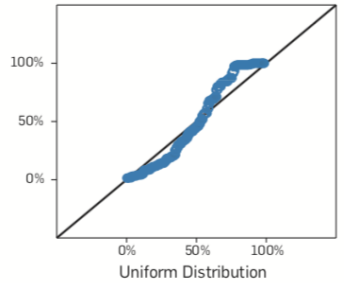
Communications of the ACM, 53(6), pp. 59-67, Jun 2010

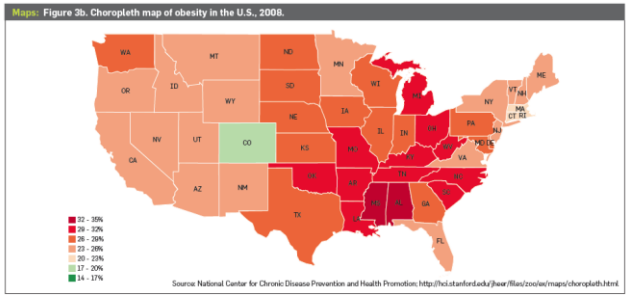
https://queue.acm.org/detail.cfm?id=1805128

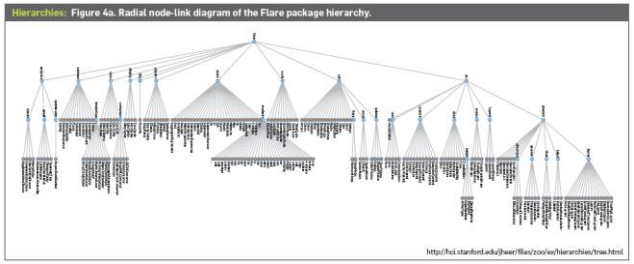
- The goal of visualization is to aid our understanding of data by leveraging the human visual system's highly tuned ability to see patterns, spot trends, and identify outliers.
- By making data more accessible and appealing, visual representations may also help engage more diverse audiences in exploration and analysis.
- The challenge is to create effective and engaging visualizations that are appropriate to the data.

- time-series data
- statistical data
- maps
- hierarchies and networks



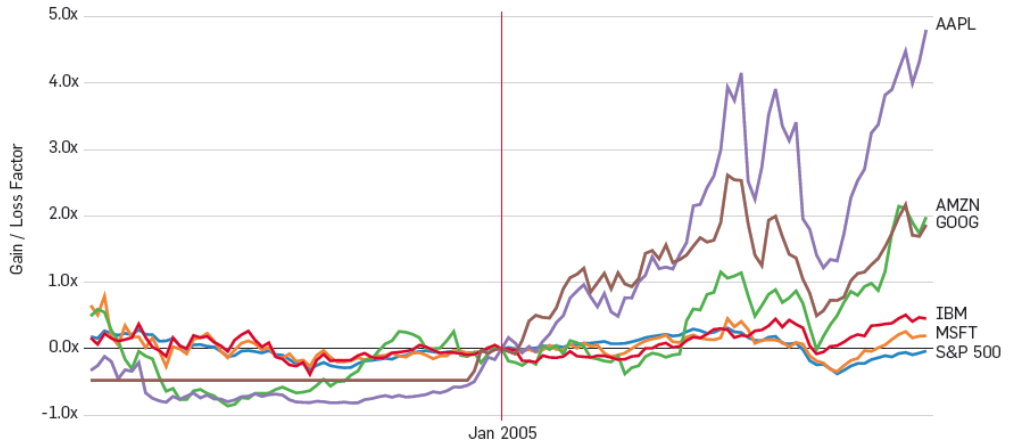






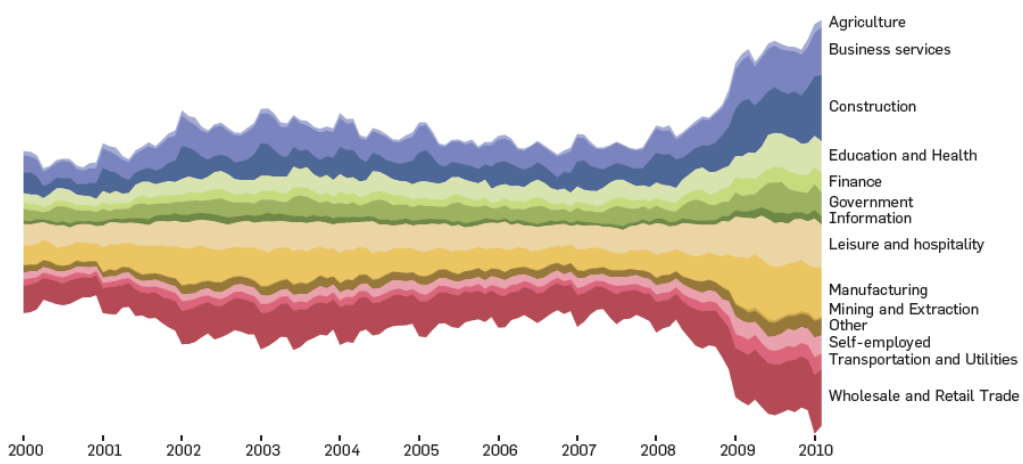
time-series data (1)

Time-Series Data: Figure 1a. Index chart of selected technology stocks, 2000–2010.



Source: Yahoo! Finance; http://hci.stanford.edu/jheer/files/zoo/ex/time/index-chart.html

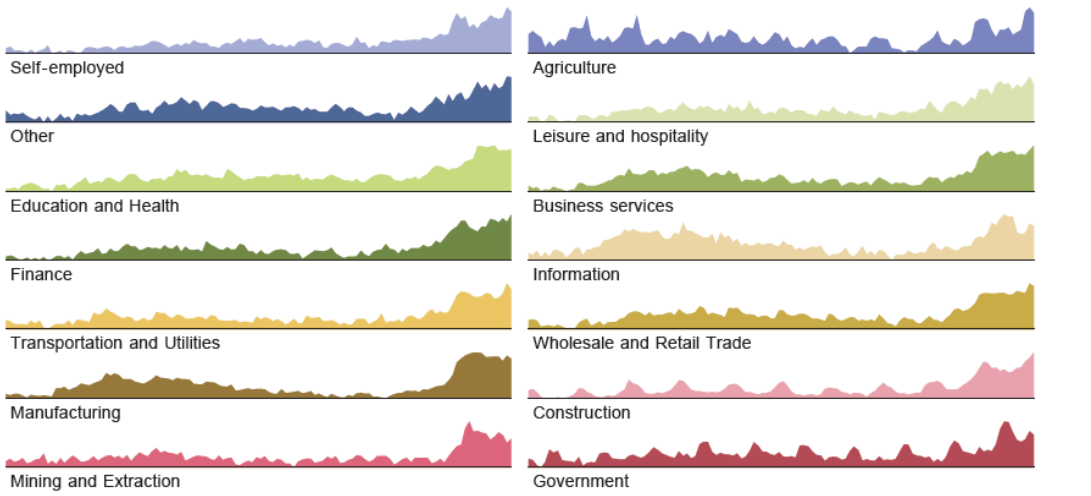
Time-Series Data: Figure 1b. Stacked graph of unemployed U.S. workers by industry, 2000–2010.



Source: U.S. Bureau of Labor Statistics; http://hci.stanford.edu/jheer/files/zoo/ex/time/stack.html

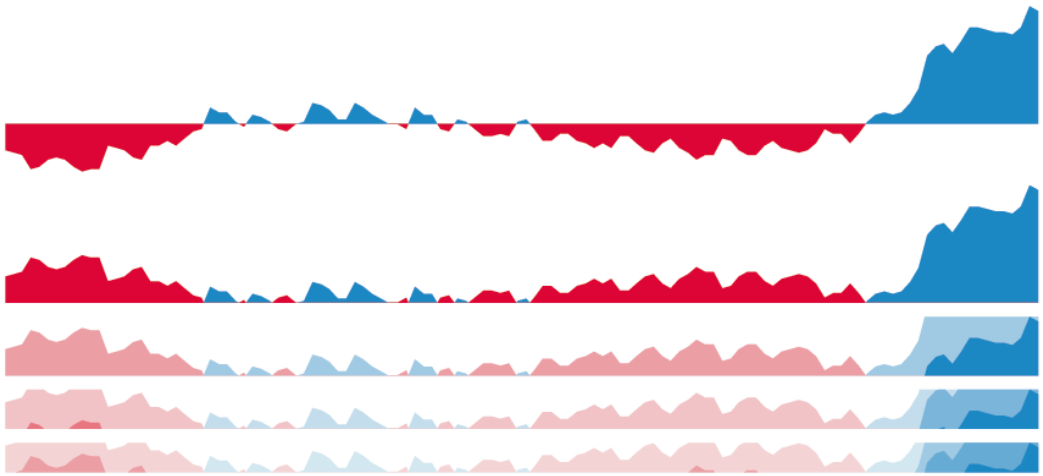
time-series data (2)

Time-Series Data: Figure 1c. Small multiples of unemployed U.S. workers, normalized by industry, 2000–2010.



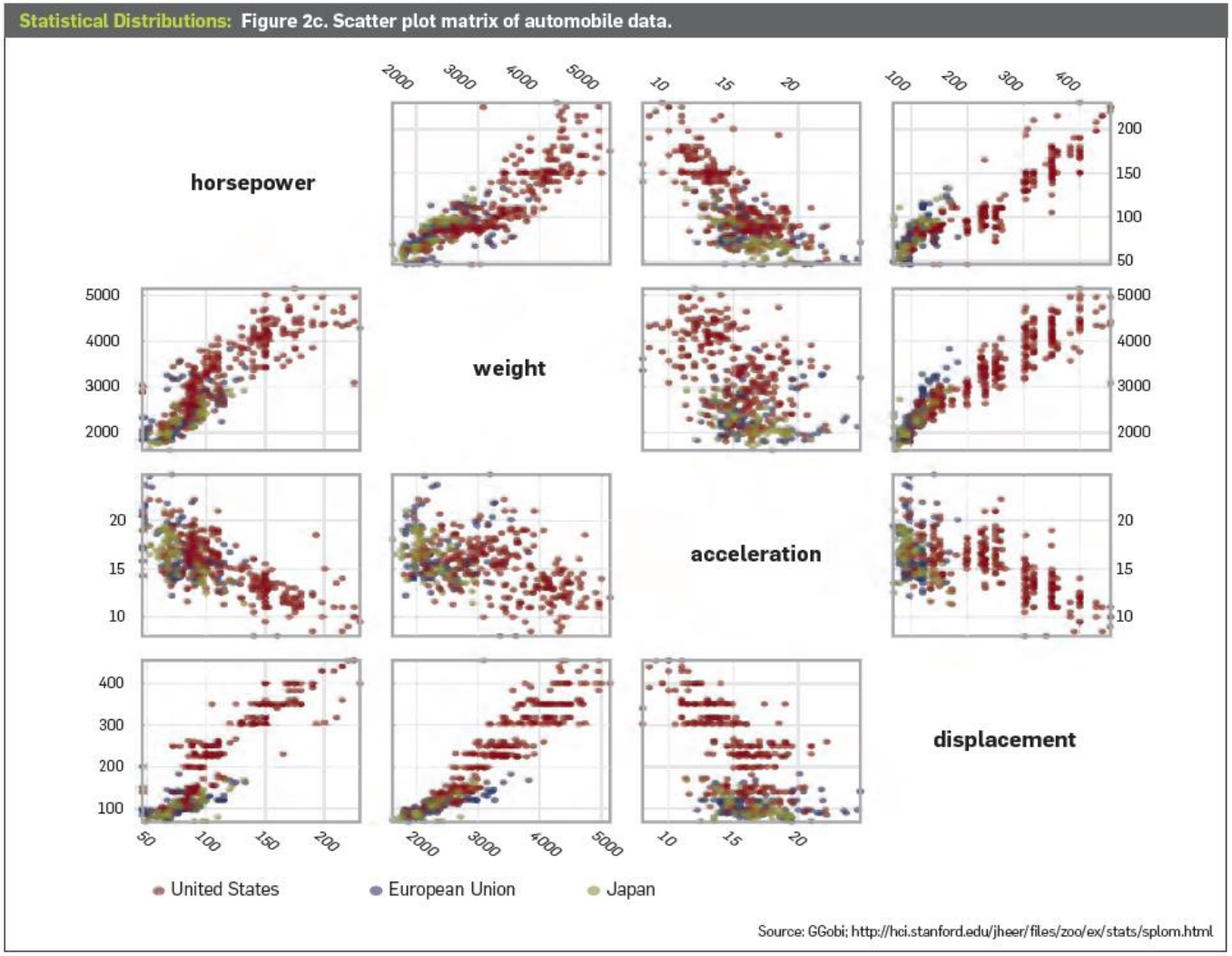
Source: U.S. Bureau of Labor Statistics; http://hci.stanford.edu/jheer/files/zoo/ex/time/multiples.html

Time-Series Data: Figure 1d. Horizon graphs of U.S. unemployment rate, 2000–2010.

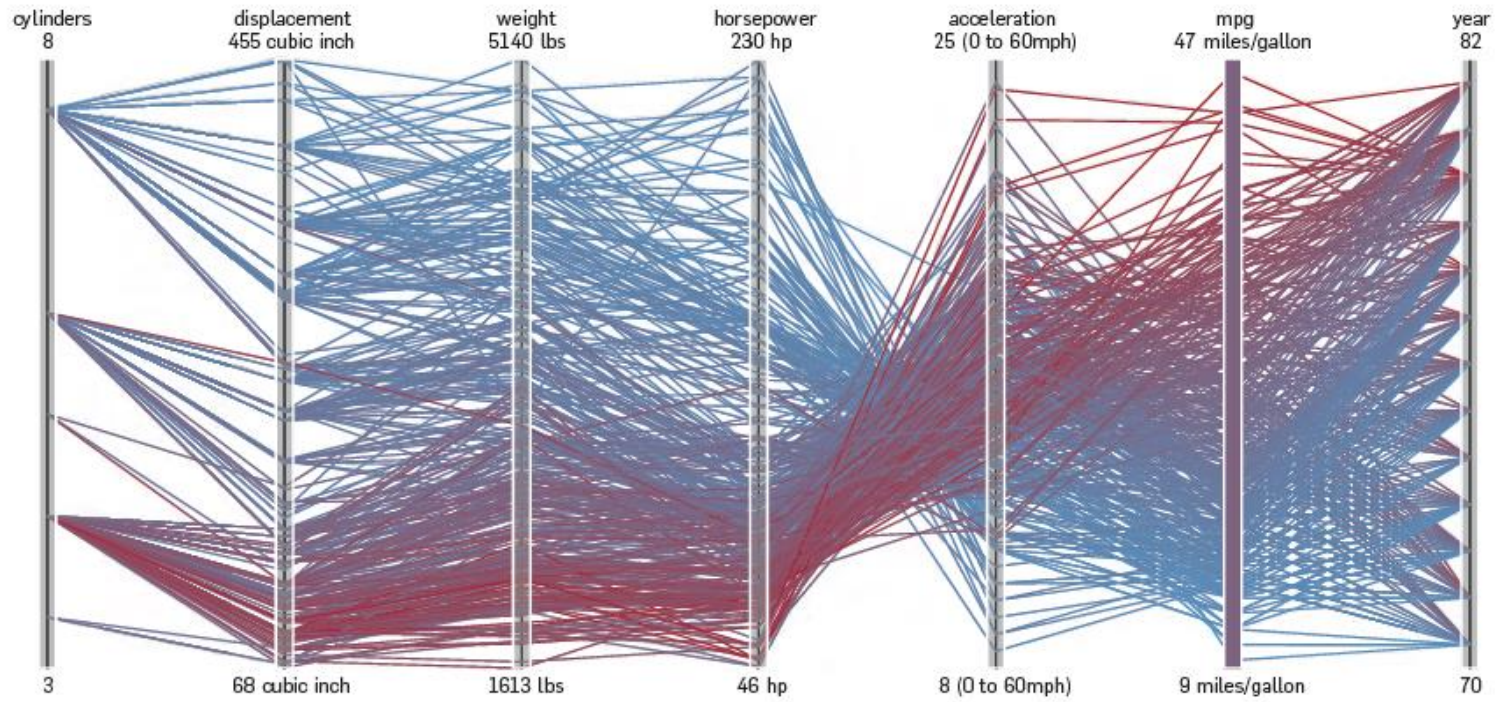


Source: U.S. Bureau of Labor Statistics; http://hci.stanford.edu/jheer/files/zoo/ex/time/horizon.html

statistical data



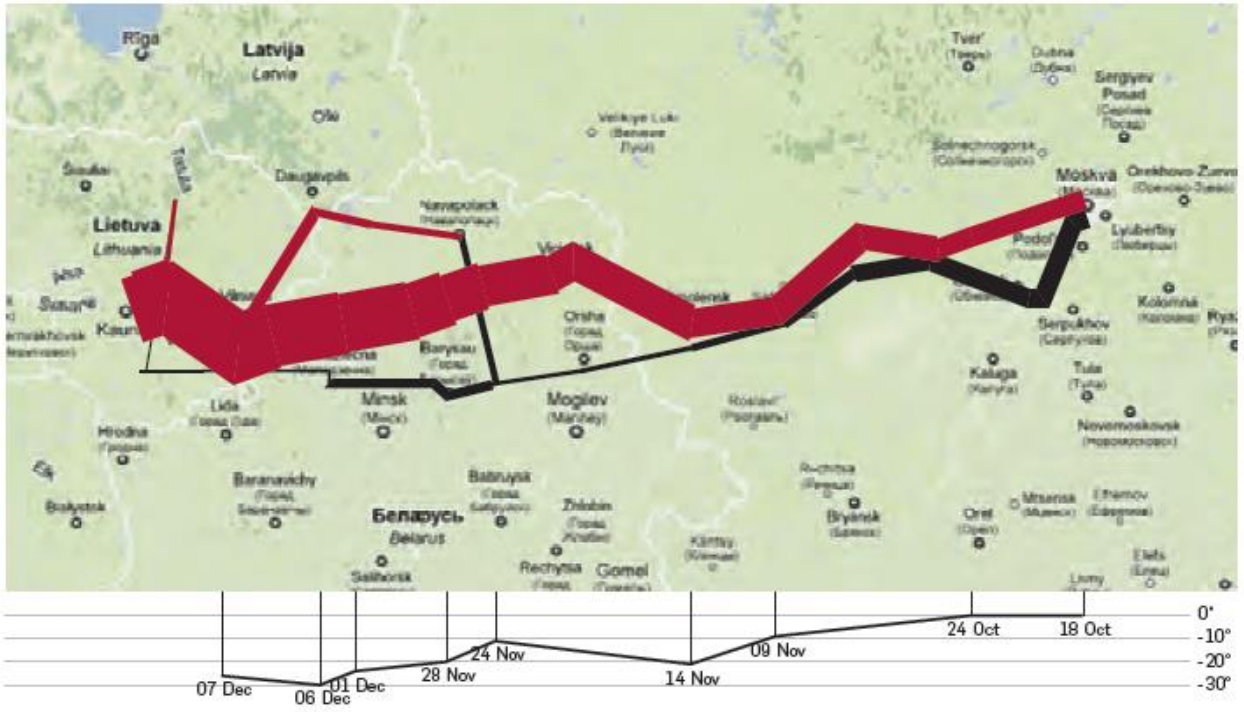
Statistical Distributions: **Figure 2d. Parallel coordinates of automobile data.**



Source: GGobi; http://hci.stanford.edu/jheer/files/zoo/ex/stats/parallel.html

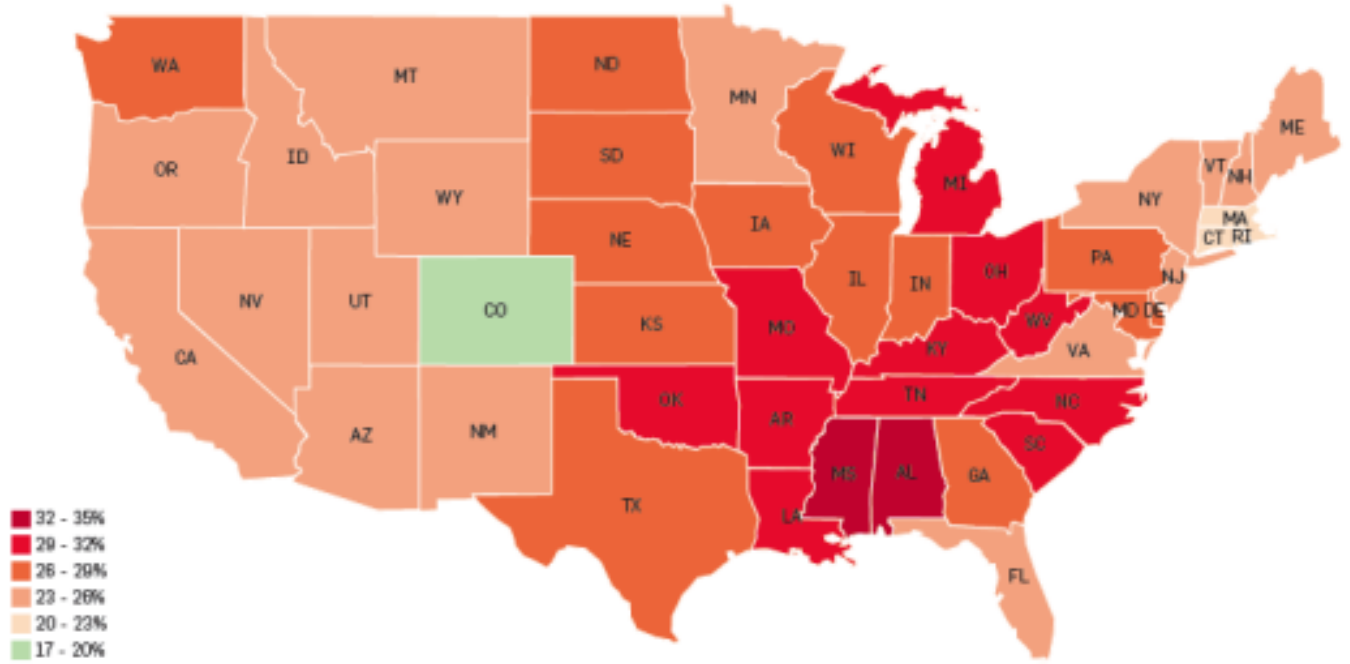
maps

Maps: **Figure 3a. Flow map of Napoleon's March on Moscow, based on the work of Charles Minard.**



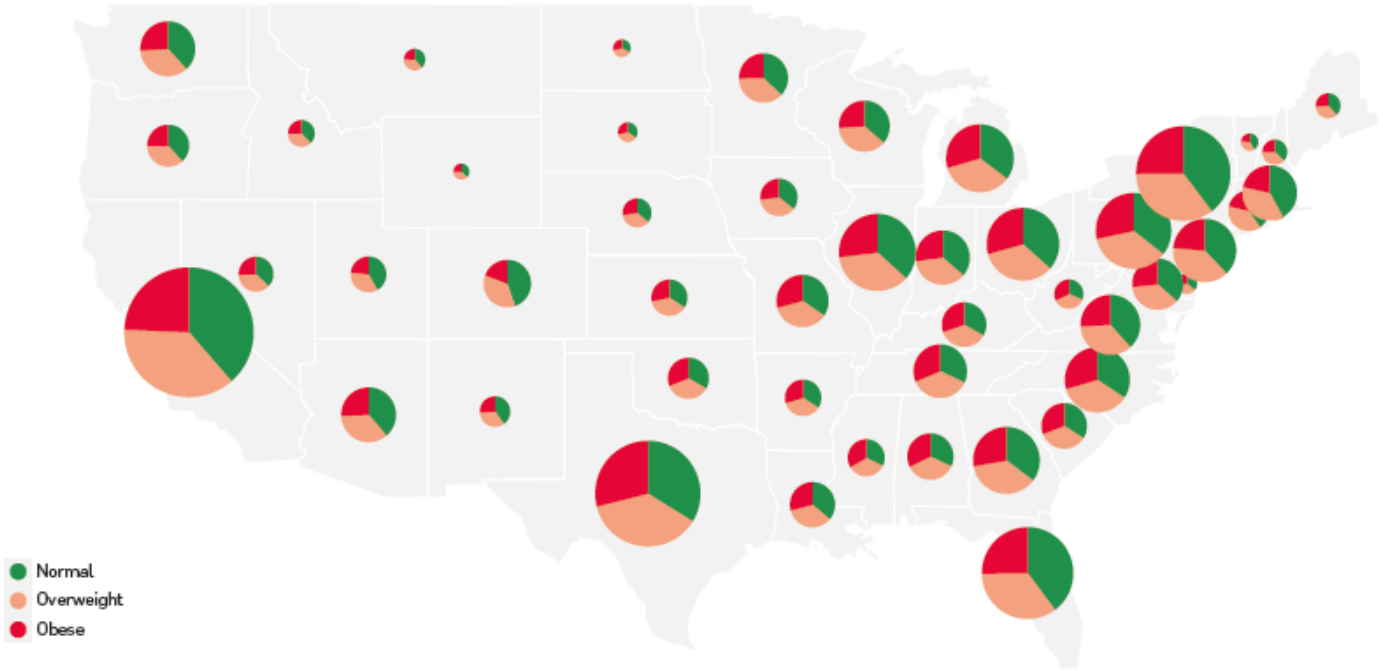
http://hci.stanford.edu/jheer/files/zoo/ex/maps/napoleon.html

Maps: Figure 3b. Choropleth map of obesity in the U.S., 2008.



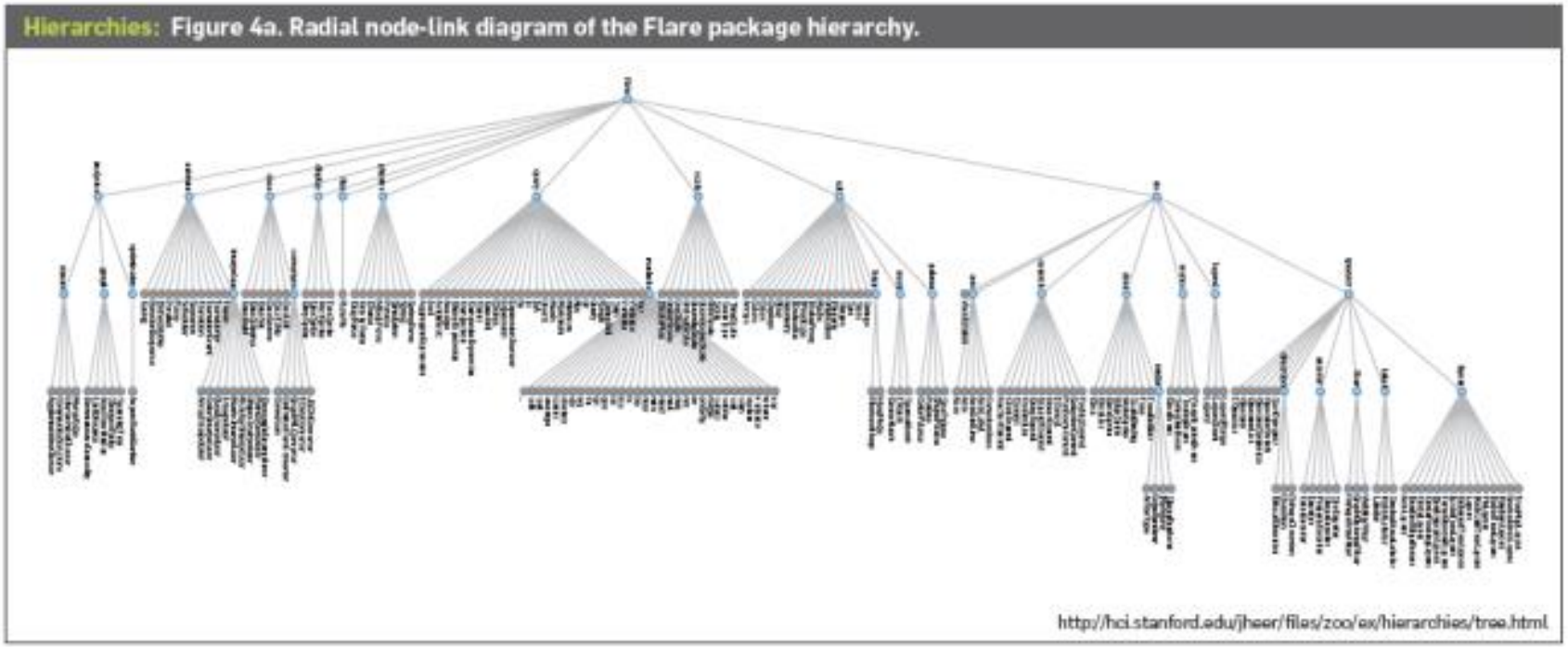
Source: National Center for Chronic Disease Prevention and Health Promotion; http://hci.stanford.edu/jheer/files/zoo/ex/maps/choropleth.html

Maps: **Figure 3c. Graduated symbol map of obesity in the U.S., 2008.**

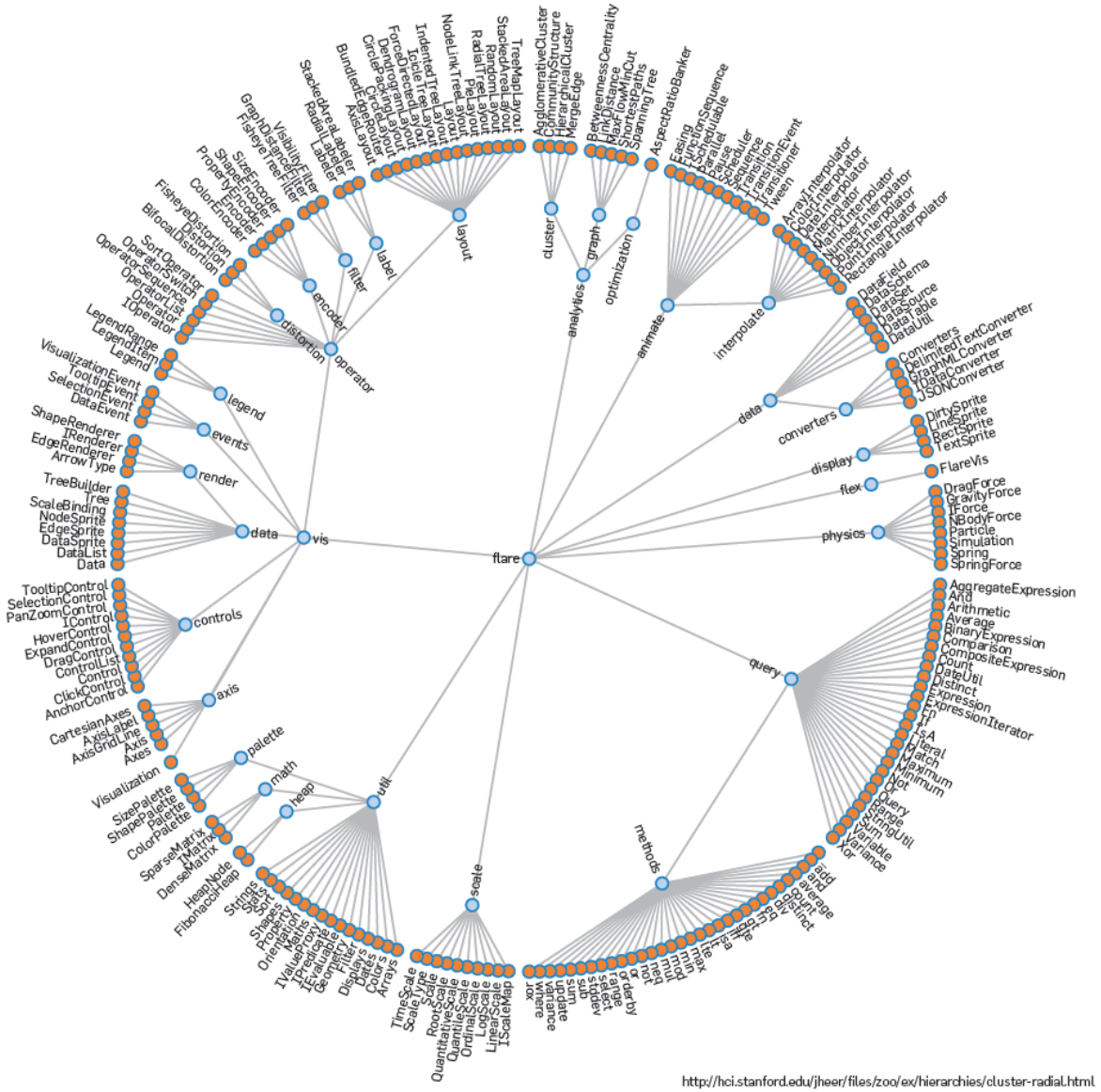


Source: National Center for Chronic Disease Prevention and Health Promotion; http://hci.stanford.edu/jheer/files/zoo/ex/maps/symbol.html

hierarchies

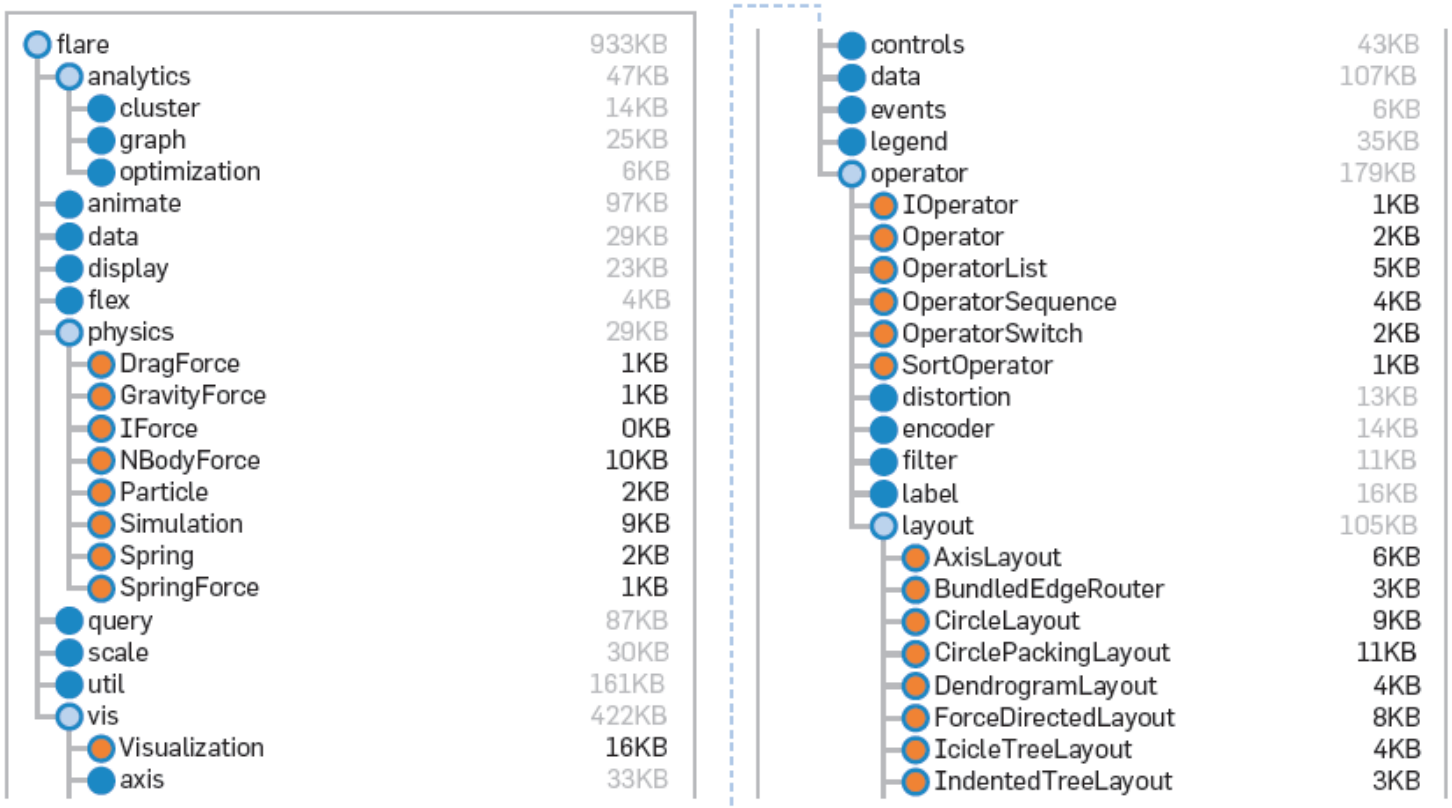


Hierarchies: **Figure 4b. Cartesian node-link diagram of the Flare package hierarchy.**



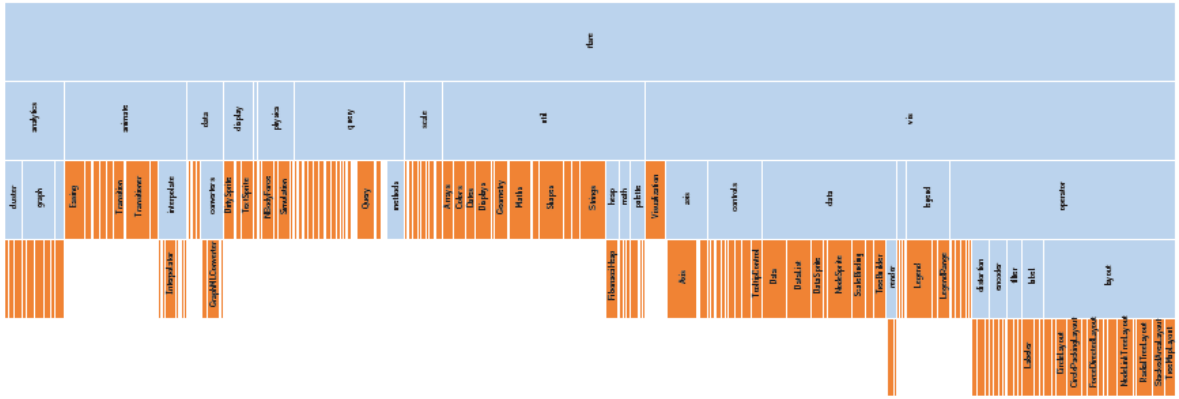
http://hci.stanford.edu/jheer/files/zoo/ex/hierarchies/cluster-radial.html

Hierarchies: **Figure 4c. Indented tree layout of the Flare package hierarchy.**



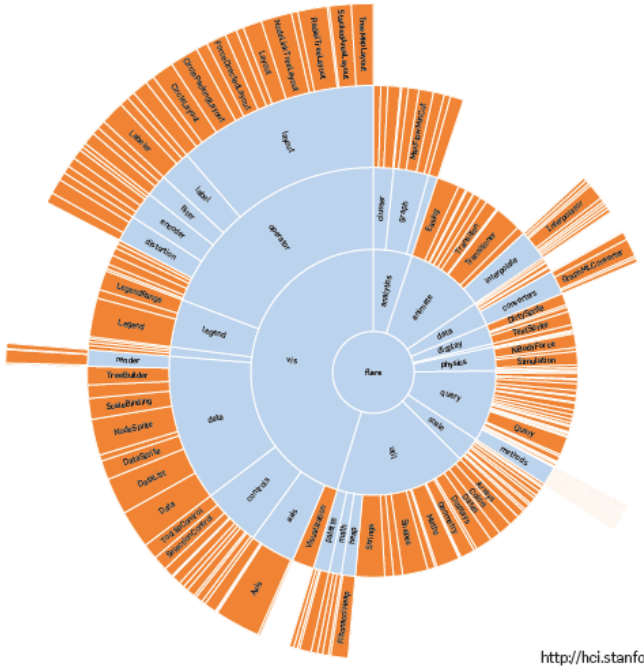
http://hci.stanford.edu/jheer/files/zoo/ex/hierarchies/indent.html

Hierarchies: **Figure 4d. Icicle tree layout of the Flare package hierarchy.**



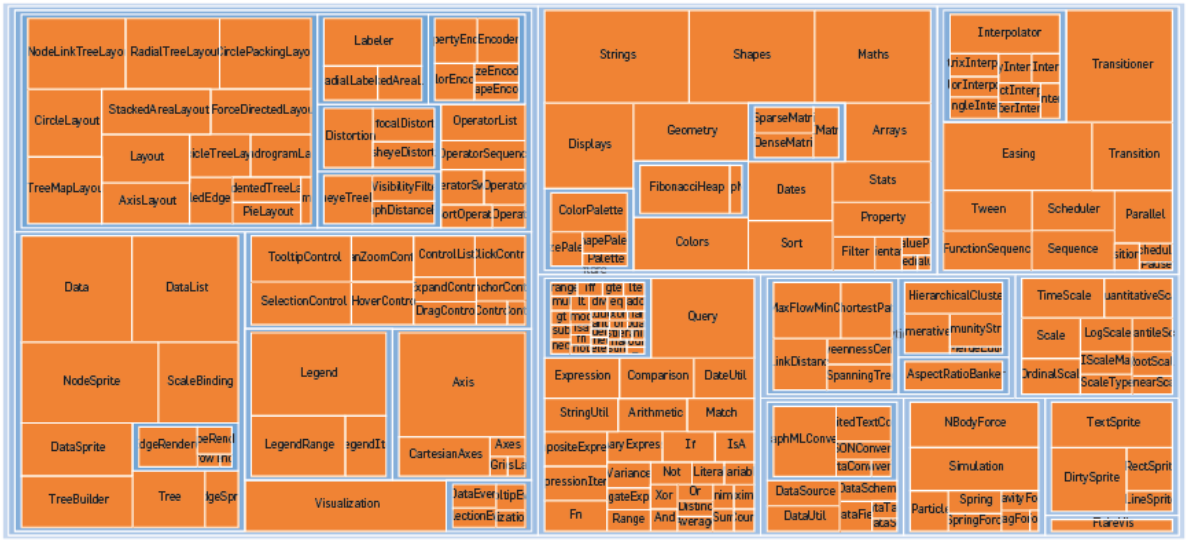
http://hci.stanford.edu/jheer/files/zoo/ex/hierarchies/icicle.html

Hierarchies: **Figure 4e. Sunburst (radial space-filling) layout of the Flare package hierarchy.**



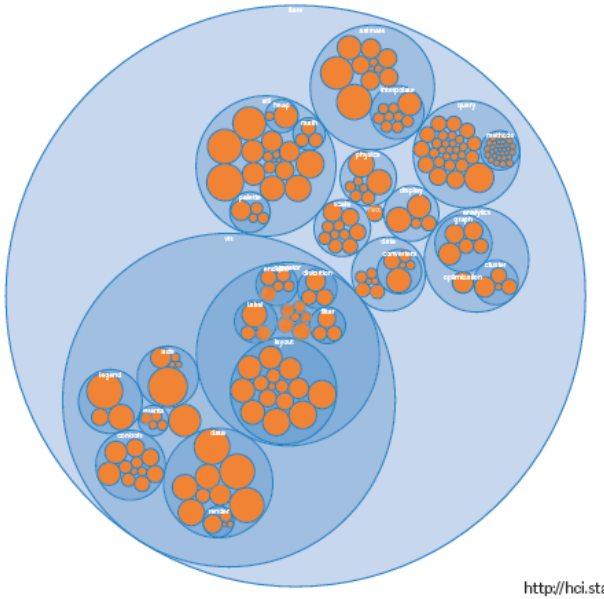
http://hci.stanford.edu/jheer/files/zoo/ex/hierarchies/sunburst.html

Hierarchies: **Figure 4f. Treemap layout of the Flare package hierarchy.**



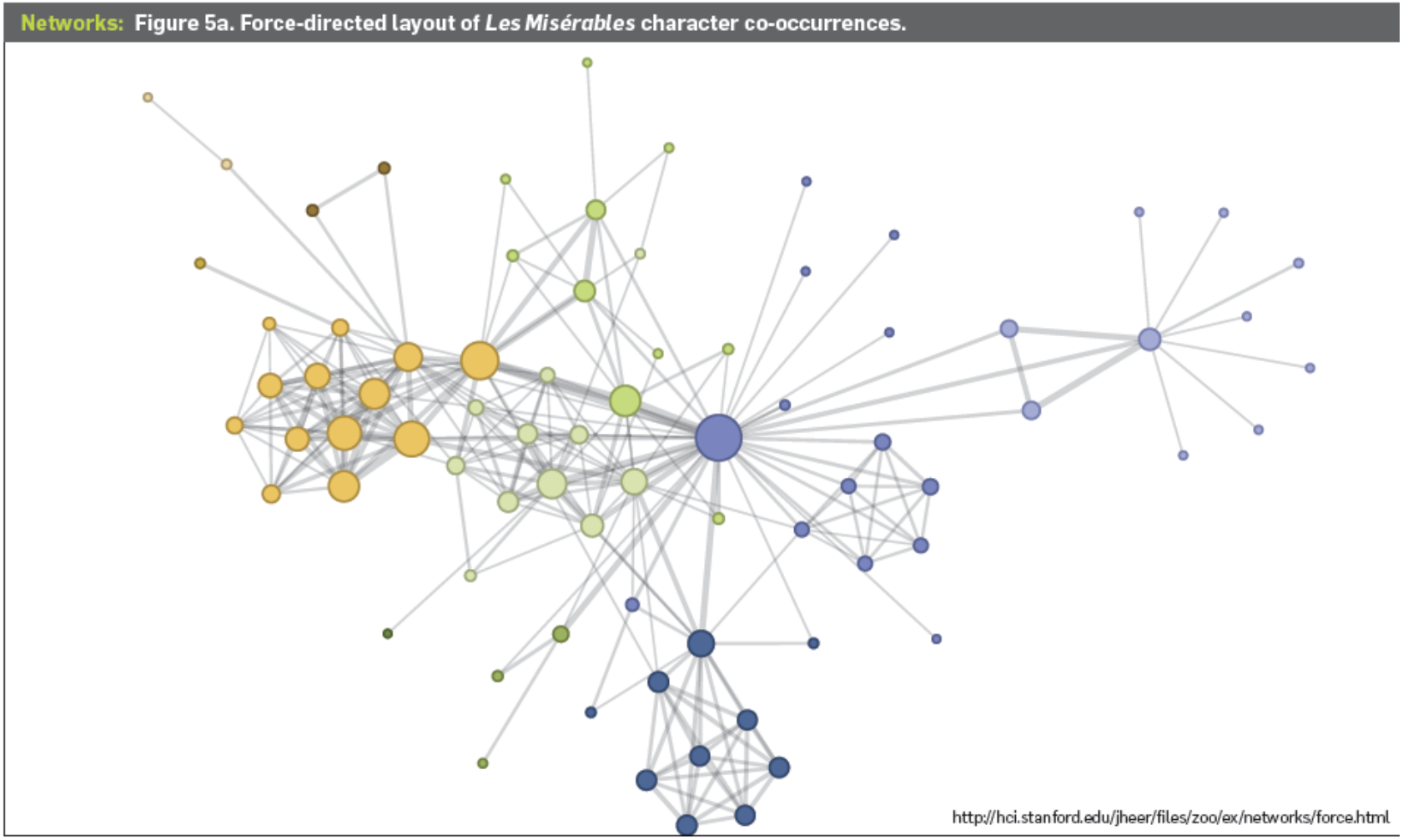
http://hci.stanford.edu/jheer/files/zoo/ex/hierarchies/treemap.html

Hierarchies: **Figure 4g. Nested circles layout of the Flare package hierarchy.**

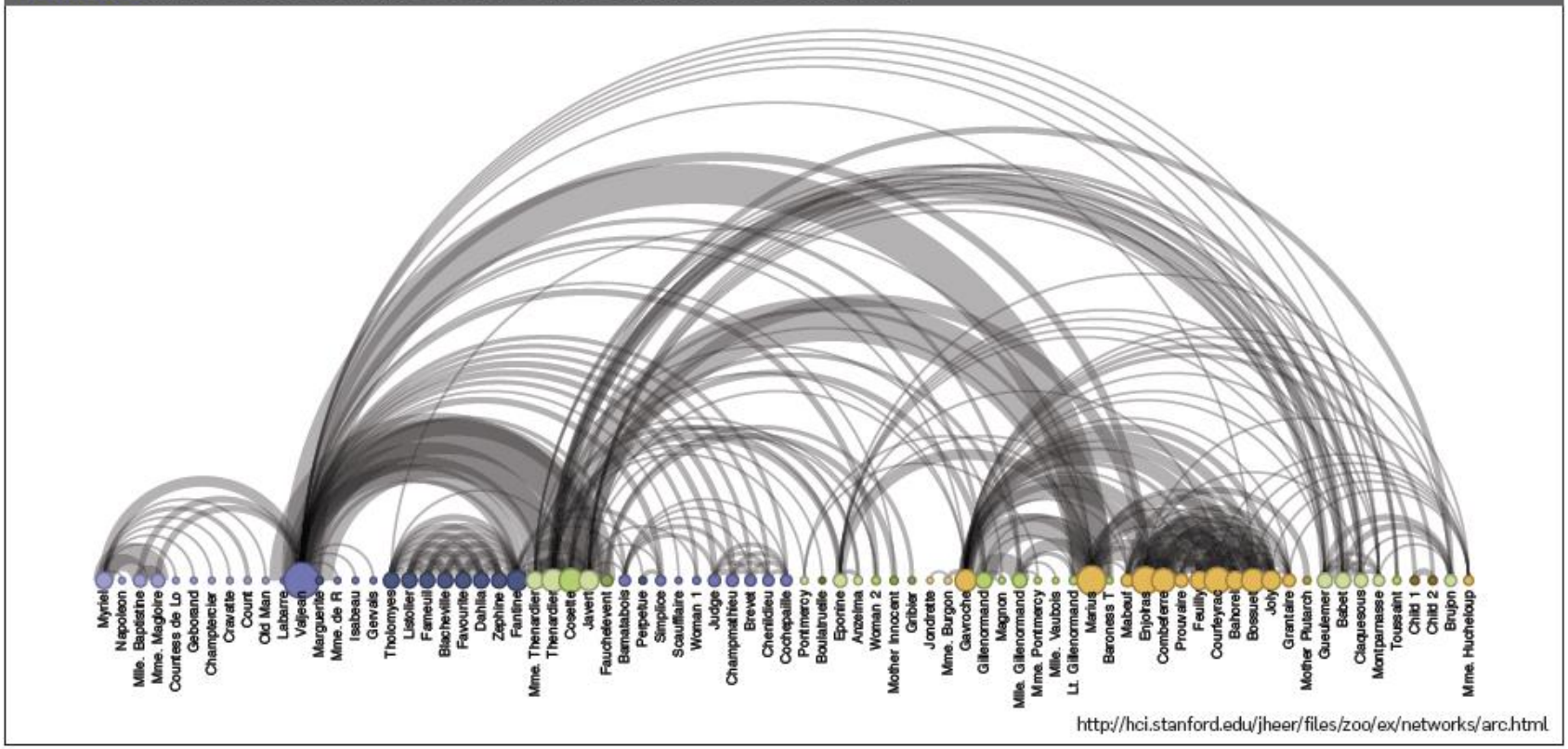


http://hci.stanford.edu/jheer/files/zoo/ex/hierarchies/pack.html
Source: The Flare Toolkit http://flare.prefuse.org

Networks

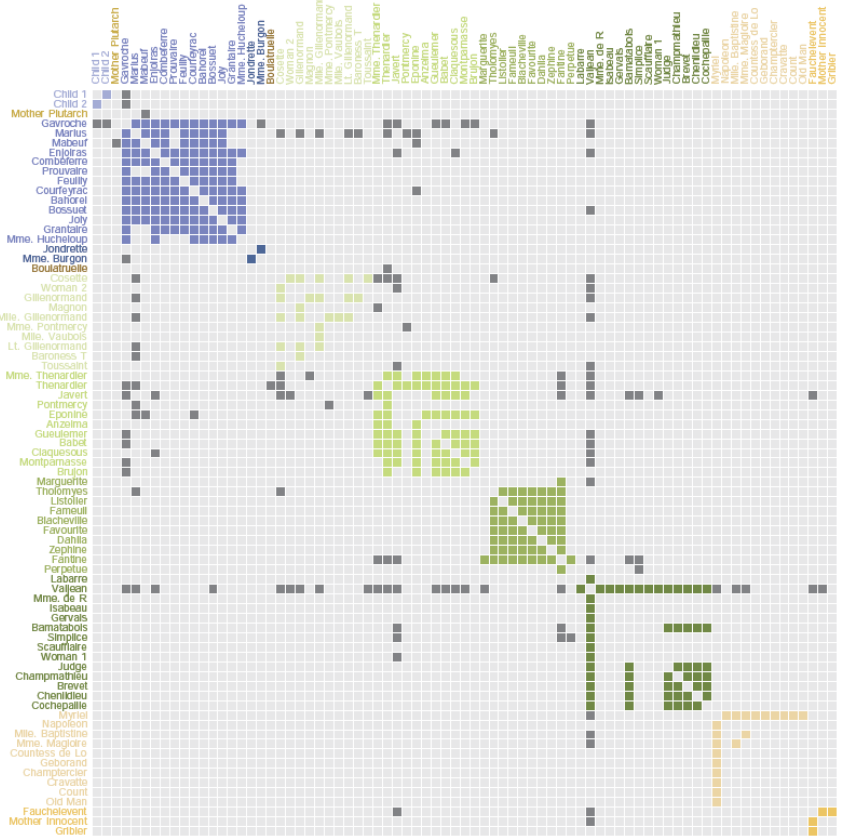

Networks: Figure 5a. Force-directed layout of *Les Misérables* character co-occurrences.
http://hci.stanford.edu/jheer/files/zoo/ex/networks/force.html

Networks: **Figure 5b. Arc diagram of *Les Misérables* character co-occurrences.**



http://hci.stanford.edu/jheer/files/zoo/ex/networks/arc.html

Networks: Figure 5c. Matrix view of *Les Misérables* character co-occurrences.



http://hci.stanford.edu/jheer/files/zoo/ex/networks/matrix.html
Source: http://www-personal.umich.edu/~mejn/netdata

D3.JS AND HTML5

DNA

- a set of mappings between data properties and visual attributes such as position, size, shape, and color—and that customized species of visualization might always be constructed by varying these encodings.

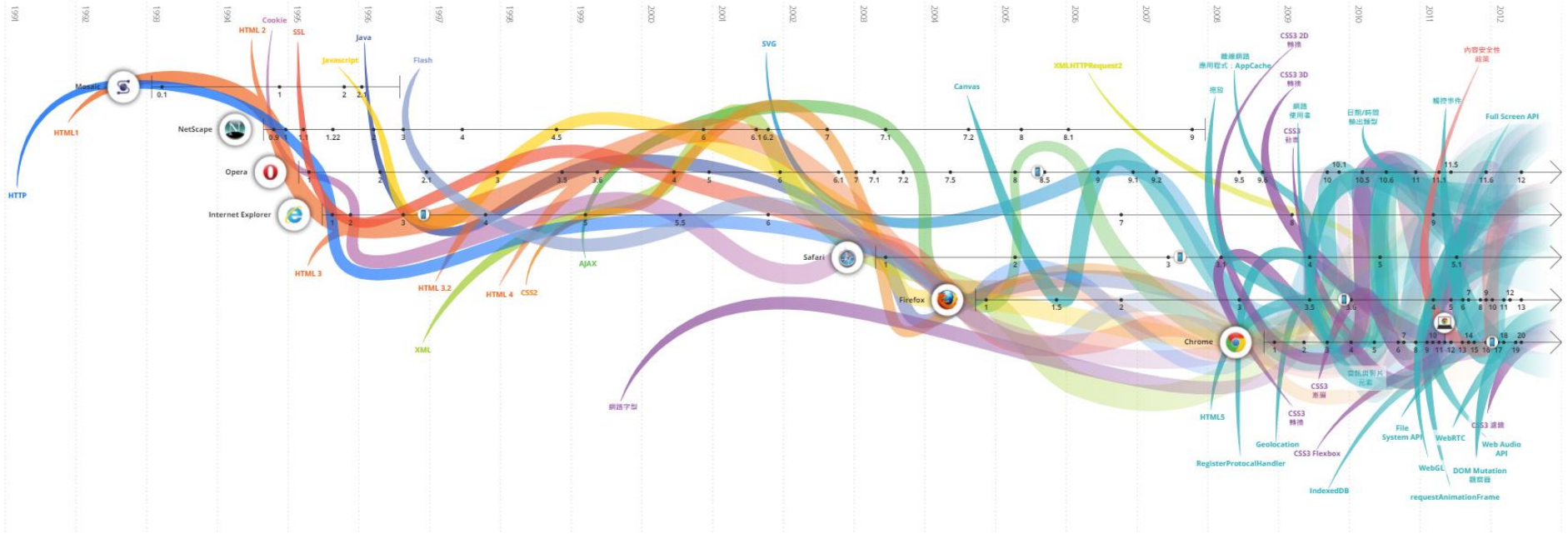
Protovis

- a graphical approach to visualization
  - 2009~2011
  - http://mbostock.github.io/protovis/
- D3.js (2011~)
  - https://d3js.org/

Outline

- Web history
- HTML5 = markup + CSS + javascript
- Learning Resource
- JavaScript Library
- Node JS

Web history

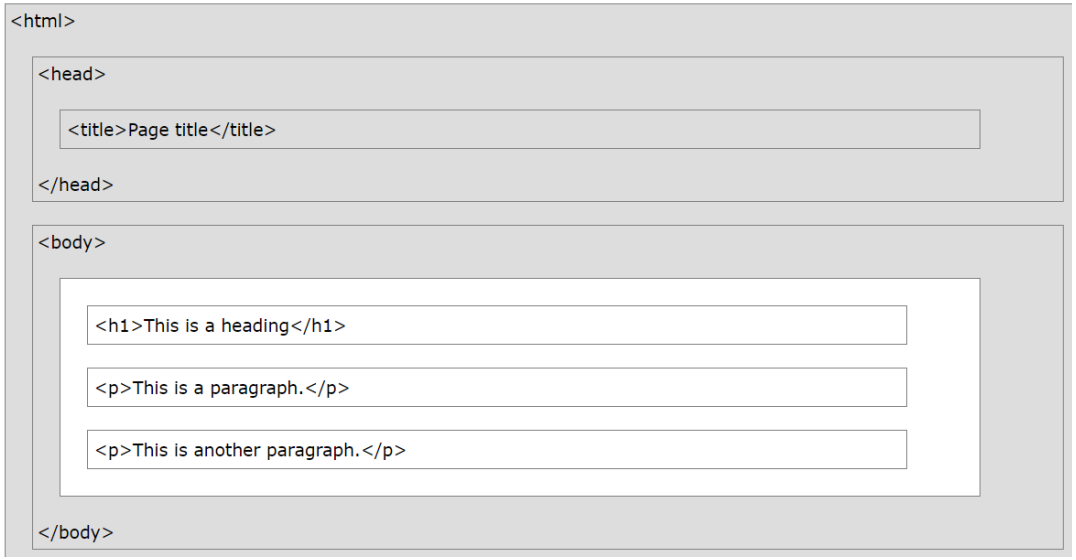


- The color bands in this visualization represent the interaction between web technologies and browsers

http://www.evolutionoftheweb.com/?hl=zh-tw

HTML5 = markup + CSS + javascript

- HyperText Markup Language is the standard **markup language** used to create web pages.

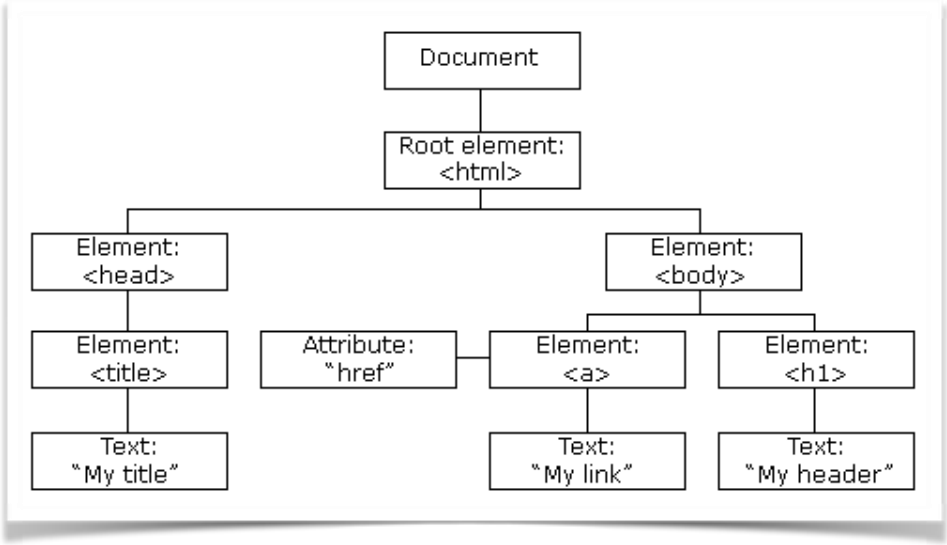



```
<!DOCTYPE html>
<html>
  <head>
    <title>This is a title</title>
  </head>
  <body>
    <p>Hello world!</p>
  </body>
</html>
```

The content inside the <body> section will be displayed in the browser. The content inside the <title> element will be shown in the browser's title bar.

HTML DOM (Document Object Model)

```
<!DOCTYPE html>
<html>
  <head>
    <title>My title</title>
  </head>
  <body>
    <a href="…">My link</a>
    <h1>Hello world!</h1>
  </body>
</html>
```

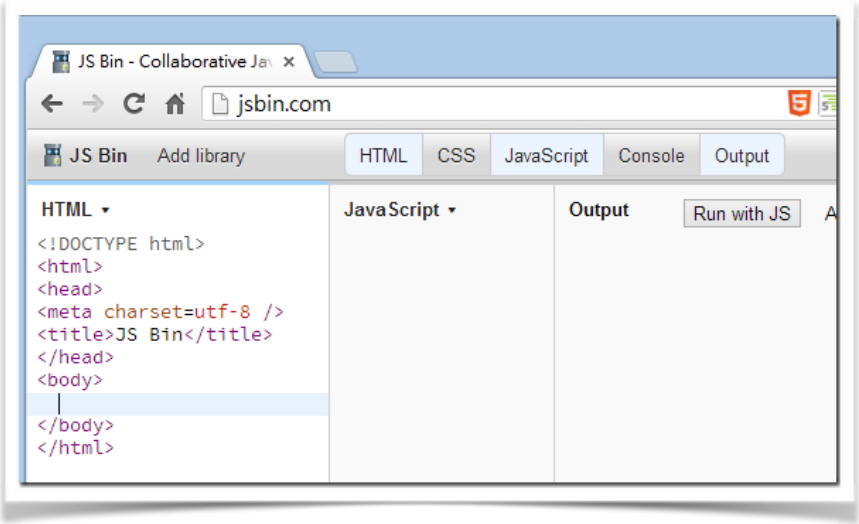


<h1>My First JavaScript</h1>

```
<p>JavaScript can change the content of an HTML element:</p>

<button type="button" onclick="myFunction()">Click Me!</button>

<p id="demo">This is a demonstration.</p>

<script>

function myFunction() {

    document.getElementById("demo").innerHTML = "Hello JavaScript!";

}

</script>
```

JS Bin

- JS Bin is a JavaScript, HTML and CSS playground.



SVG - Scalable Vector Graphics

```
<svg width="400" height="300">
  <rect width="300" height="100" style="fill:rgb(0,0,255)" />
  <rect x="50" y ="50" width="300" height="100"
style="fill:rgb(0,0,255);stroke-width:3;stroke:rgb(0,0,0)" />
</svg>
```

SVG - Scalable Vector Graphics

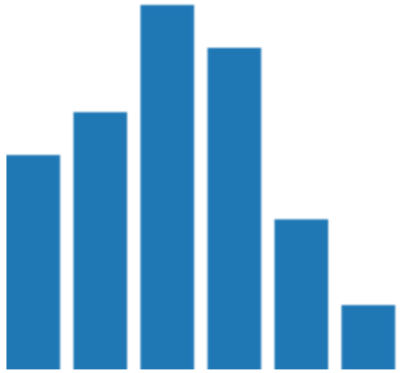
```
<svg width="400" height="300">
   <rect width="300" height="100" style="fill:rgb(0,0,255)" />
   <rect id="myrect"
         x="50" y ="50" width="300" height="100"
         style="fill:rgb(0,0,255);stroke-width:3;stroke:rgb(0,0,0)"
         onclick="doRectClick();"/>
 </svg>
```

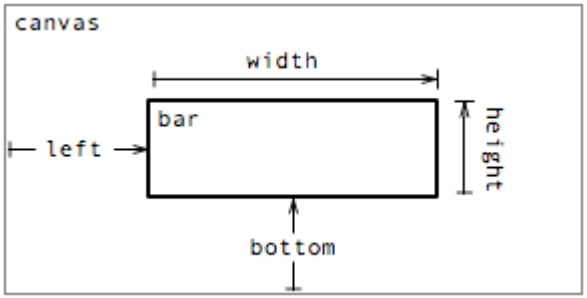
SVG with JavaScript

```
function doRectClick(){
  var myrect = document.getElementById('myrect');
  var r = Math.floor(Math.random() * 255);
  var g = Math.floor(Math.random() * 255);
  var b = Math.floor(Math.random() * 255);
  var x = Math.floor(Math.random() * 255);

  myrect.style.fill = 'rgb(' + r + ', ' + g + ' , ' + b + ')';
  myrect.setAttribute('x', x);
  myrect.setAttribute('y', Math.floor(Math.random() * 250));
}
```

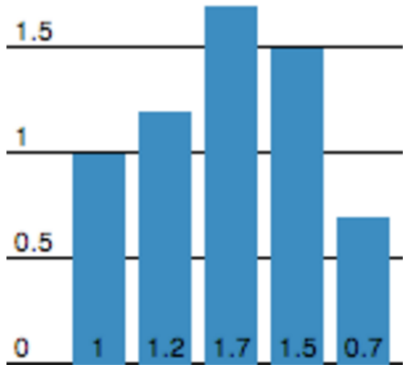
A preview for D3.js

```
var vis = new pv.Panel()
    .width(150)
    .height(150);

vis.add(pv.Bar)
    .data([1, 1.2, 1.7, 1.5, .7, .3])
    .width(20)
    .height(function(d) d * 80)
    .bottom(0)
    .left(function() this.index * 25);

vis.render();
```





```
var vis = new pv.Panel()
    .width(150)
    .height(150);

vis.add(pv.Rule)
    .data(pv.range(0, 2, .5))
    .bottom(function(d) d * 80 + .5)
  .add(pv.Label);

vis.add(pv.Bar)
    .data([1, 1.2, 1.7, 1.5, .7])
    .width(20)
    .height(function(d) d * 80)
    .bottom(0)
    .left(function() this.index * 25 + 25)
  .anchor("bottom").add(pv.Label);

vis.render();
```



http server using Python

- Python3指令:
  - python -m http.server

- Python2指令:
  - python -m SimpleHTTPServer

Node.js

- Run JavaScript outside a web browser
- Open source
- Cross platform
- Back-end runtime environment

Node.js Example

- A simple http server made in Node.js

```
var http = require('http');

//create a server object:
http.createServer(function (req, res) {
  res.write('Hello World!'); //write a response to the client
  res.end(); //end the response
}).listen(8080); //the server object listens on port 8080
```