

# Data Science Tools: Interactive visualization

10-March-2020

Sepi Chakaveh & Assel Altayeva

**Overview Data Science**

Overview of Data Science

# D3 –Data-Driven Document

- D3 Show Reel:

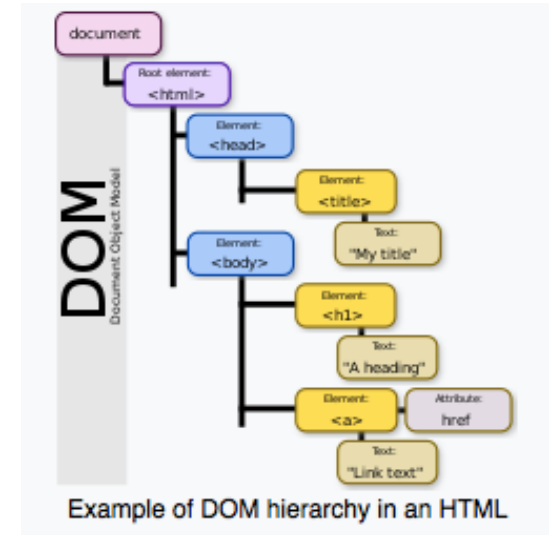
<https://bl.ocks.org/mbostock/1256572>

## D3 is Data $\mapsto$ Elements

- Visualizing Data with Web Standards (HTML/SVG)
- Data  $\mapsto$  Elements – constructing the DOM from Data
- In visualization, each data point has a corresponding element (graphical marks). D3 helps you maintain this mapping!

SVG – Scalable Vector Graphics

DOM – Document Object Model



# D3- Data-Driven Documents

- D3.js is a JavaScript library for manipulating documents based on data.
- [d3js.org](https://d3js.org)
- [github.com/mbostock/d3](https://github.com/mbostock/d3)
- [github.com/mbostock/d3/wiki/API-Reference](https://github.com/mbostock/d3/wiki/API-Reference)

# D3 is a little like jQuery (jQuery)

```
// Find element
var node = $("#elementId");

// Style element
node.css("color", "#000");

// Set attribute
node.attr("data-lc", "data-value");

// Handle event
node.click(function(ev) {
    alert("Hello, world!");
});
```

## Pre-conditions

# Index.html

```
<!DOCTYPE html>
<html>
<head>
  <title>TernopilJS > Example 1</title>
  <meta charset="utf-8">
  <link rel="stylesheet" href="style.css"/>
</head>
<body>
  <svg width="960" height="500"></svg>

  <script src="//d3js.org/d3.v4.0.0-alpha.9.min.js"></script>
  <script src="script.js"></script>
</body>
</html>
```

# Selections

- A selection is an array of elements pulled from the current document. D3 uses CSS3 to select elements.
- After selecting elements, you apply operators to them to do stuff. These operators can get or set attributes, styles, properties, HTML and text content.
- `d3.select(selector), d3.select(node)`
- `d3.selectAll(selector), d3.selectAll(nodes)`
- See more: [github.com/mbostock/d3/wiki/Selections](https://github.com/mbostock/d3/wiki/Selections)

# Select element and style it a bit

```
var svg = d3.select("svg");

var margin = {top: 30, right: 50, bottom: 30, left: 30},
    width = +svg.attr("width") - margin.left - margin.right,
    height = +svg.attr("height") - margin.top - margin.bottom,
    labelPadding = 3;

var g = svg.append("g")
    .attr("transform", "translate(" + margin.left + "," + margin.top + ")");
```

# Selections

## D3 uses CSS Selectors

Single selector:

```
#foo      // <any id="foo"> </any>  
foo       // <foo> </foo>  
.foo      // <any class="foo"> </any>  
[foo=bar] // <any foo="bar"> </any>  
foo bar   // <foo><bar> </bar></foo>
```

Multiple selectors:

```
foo.bar    // <foo class="bar"> </foo>  
foo#bar    // <foo id="bar"> </foo>
```



# .select()

HTML

CSS

JS

Result

```
var svg = d3.select("svg")

var run = function() {
  var myRect = svg.select("rect")
  myRect.attr("width", 100)
  myRect.attr("height", 100)
  myRect.style("fill", "steelblue")
}
```

Resources


HTML

CSS

JS

Result

Run A



A single vertical bar chart is displayed. The bar has a steelblue fill and a gray border. It is centered on a white background.

Resources

# External data files

- **d3.csv** - request a comma-separated values (CSV) file.
- **d3.html** - request an HTML document fragment.
- **d3.json** - request a JSON blob.
- **d3.text** - request a text file.
- **d3.tsv** - request a tab-separated values (TSV) file.
- **d3.xhr** - request a resource using XMLHttpRequest.
- **d3.xml** - request an XML document fragment.
- See more: [github.com/mbostock/d3/wiki/Requests](https://github.com/mbostock/d3/wiki/Requests)

# Data joints

The act of creating a mapping between data points and the objects representing them.

- `[1,2,3] -> μ £ ¥`
- `[1,2,3] -> <div>1</div>`  
`<div>2</div>`  
`<div>3</div>`
- See more: [github.com/mbostock/d3/wiki/Requests](https://github.com/mbostock/d3/wiki/Requests)

# Fetch data from file

```
d3.requestTsv("data.tsv", function (d) {  
  return d;  
}, function (error, data) {  
  if (error) throw error;  
  
  console.log(data);  
});
```

# Layouts

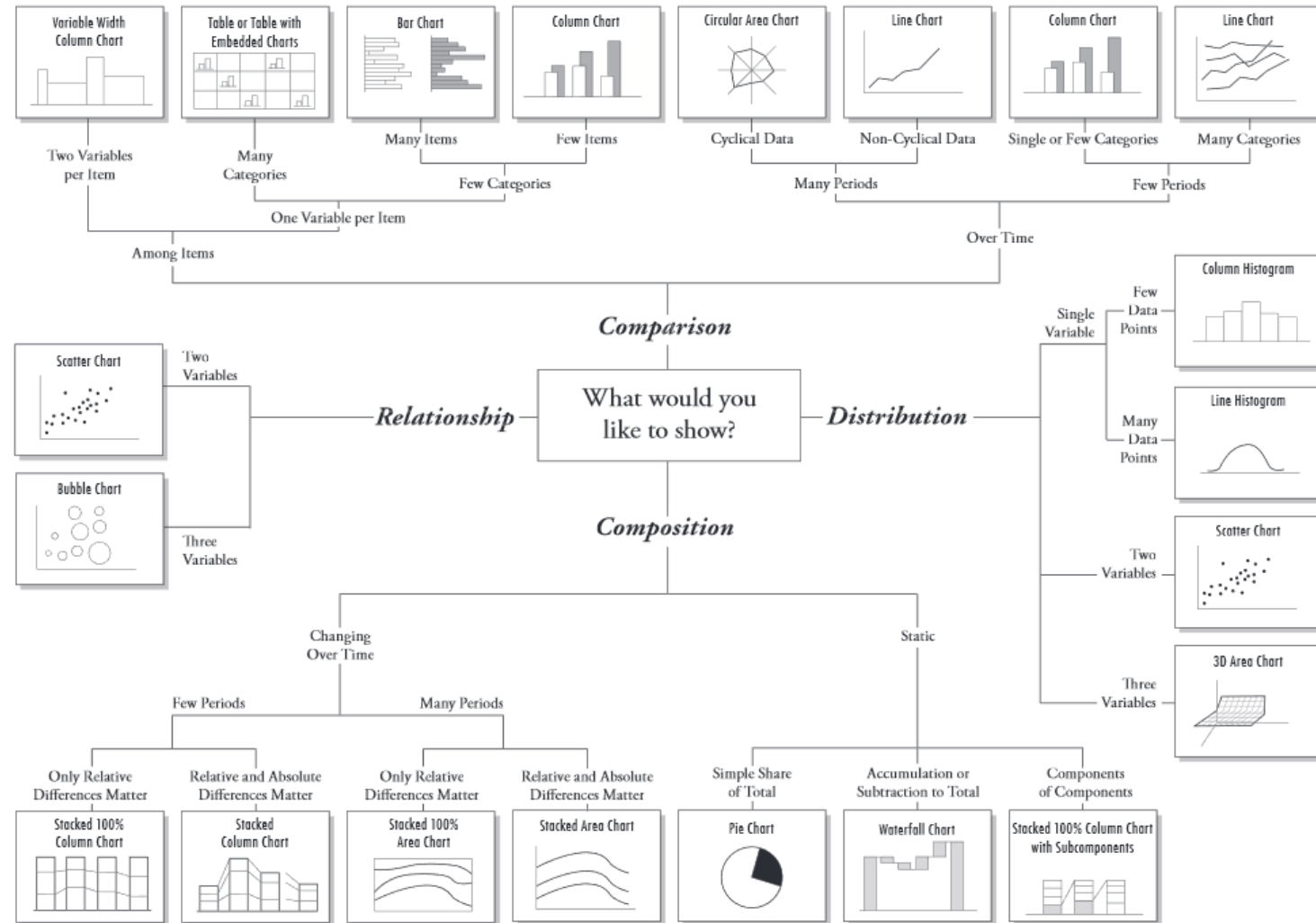
- A layout encapsulates a strategy for laying out data elements visually, relative to each other. Many layouts are built in to D3 itself:
  - **Chord** - produce a chord diagram from a matrix of relationships.
  - **Partition** - recursively partition a node tree into a sunburst or icicle.
  - **Pie** - compute the start and end angles for arcs in a pie or donut chart.
  - **Tree** - position a tree of nodes tidily.
  - etc.
- See more: [github.com/mbostock/d3/wiki/Layouts](https://github.com/mbostock/d3/wiki/Layouts)

# How to select a chart type?

Main purpose of data visualization:

- Comparison
  - Composition
  - Distribution
  - Relationship
- 
- See more: [goo.gl/1jkVNk](https://goo.gl/1jkVNk)
  - Try by yourself: [goo.gl/gl6Q4R](https://goo.gl/gl6Q4R)

# Chart Suggestions—A Thought-Starter



Source: [storytellingwithdata.com](http://storytellingwithdata.com)

# Plotly

## What is Plotly?

- Plotly is one of the finest data visualization tools available built on top of visualization library D3.js, HTML and CSS.
- It is created using Python and the Django framework.
- One can choose to create interactive data visualizations online or use the libraries that plotly offers to create these visualizations in the language/ tool of choice. It is compatible with a number of languages/ tools: R, Python, MATLAB, Perl, Julia, Arduino