D3.js

データビジュアライゼーション

一週間で D3.js を覚えられるか？

D3.js の D3 は

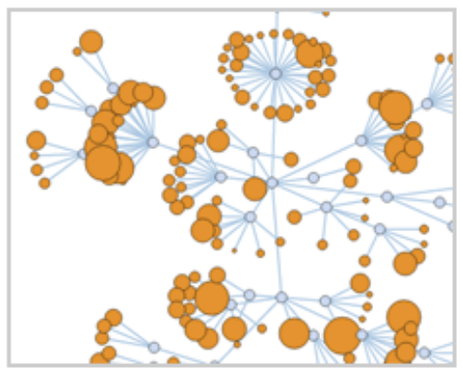
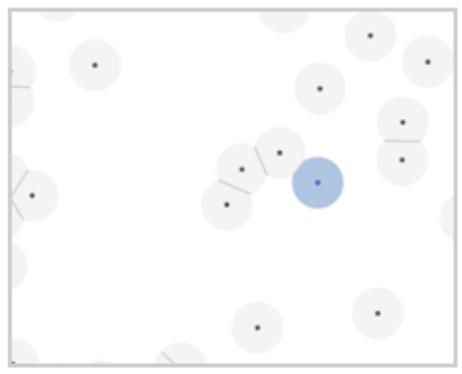
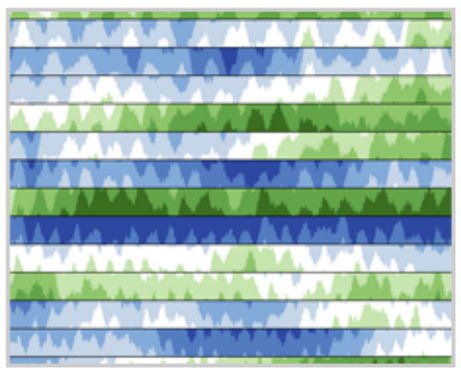
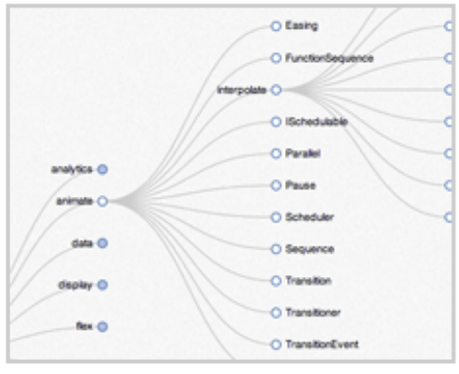
Data Driven Document

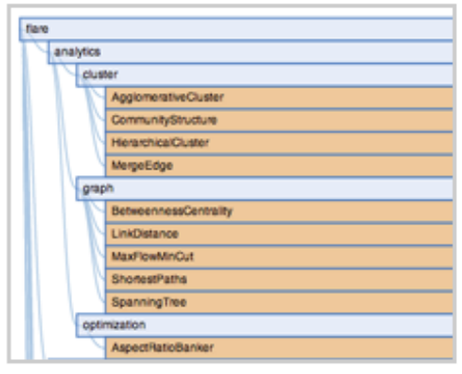
の頭文字をとったもの。

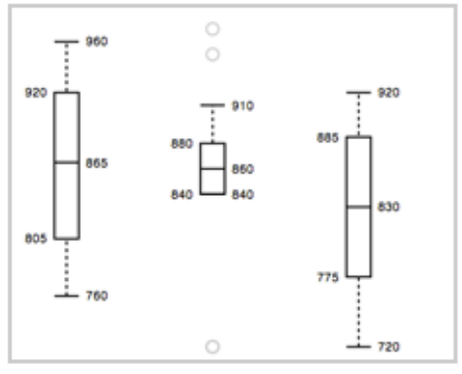
つまり、データをもとにしてドキュメントを操作する JavaScript ライブラリ

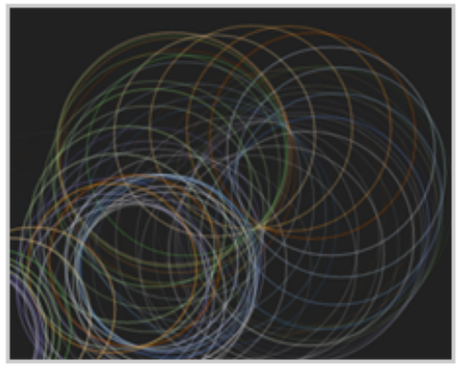
D3.js で何ができるのか？

・データから各種グラフを作成
・インタラクティブなグラフ表示







D3.js のメリット

・データと計算、出力が分離されている
・ブラウザのバージョンに依存しない
・DOM によるグラフ表示だけでなく
　SVG (スケーラビリティベクターグラフィックス) も使える
・実際はブラウザの最新機能は何でも可能
　（Canvas や WebGL も使える）

D3.js のオチ

・一般的なグラフ表示ライブラリや API サービスとは異なり D3.js には、豪華な表示を行う機能がない。＞自前でよろしく

・ビジュアル表現が意外と難しい

・日本語のサイトや資料はあるが、初心者向けか、超上級者向けのどちらか。

・予想以上に学習コストが高い

D3.js 利用時に求められるスキル

- DOM の処理に関する知識
- SVG に関する知識（別途覚える必要がある）
- データ可視化に関する知識（どのように
  ビジュアル化するか）
- JavaScript の知識（基本的なので可）

D3.js に欠かせない SVG

・SVG でのビジュアル表現がほとんど
・SVG はアニメーションも可能
　実際にはアニメーション機能をサポートしていないブラウザもあるので D3.js が計算して処理をしている
・SVG はベクトルベースなので拡大縮小しても文字やグラフは綺麗なまま

一週間で D3.js を学習できるか？

・毎度、時間がないので短時間に習得するしかない。そこで作ったのが D3.js 例文辞典。

とにかくたくさん作って覚える。

http://www.openspc2.org/reibun/D3.js/

まずは D3.js ライブラリの読み込み

・script タグを使って CDN 読み込み

```
<script src="http://d3js.org/d3.v3.min.js" charset="utf-8"></script>
```

ただし、結構遅いのでダウンロードして
自前サーバーの方がよいかも。

まずは D3.js の基本と DOM 操作

- DOM へのアクセスは select() か selectAll()
- CSS の操作は style() メソッド
- 要素の属性は attr() メソッド

```
d3.select("#myText")
        .style("color", "red")
        .attr(”  align”  , “right”  )
```

横棒グラフを表示する

・まずデータを用意
・出力先の要素を select() で指定
・selectAll() で棒グラフとなる div 要素を指定
・data() にデータを設定
・enter() でデータの数だけ処理
・append() で div 要素を追加
・style() でスタイルを指定

```
var list = [10, 30, 5, 60, 40, 78, 56, 30, 24, 80];
d3.select("#myGraph").selectAll("div")
    .data(list)
    .enter()
    .append("div")
    .style("width", function(d){
        return (d*2) +"px";
});
```

メソッドのパラメーターに関数を指定すると
1つのデータが渡される

・例えばデータをそのまま出力する場合、
　text() メソッドに関数を指定する

```
function (d, i){
        return d;
})
```

D3.js ではデータの内容は DOM 要素の __data__ プロパティに格納される

・D3.js 本体がデータを抱えて処理するのではなく、与えられたデータを DOM 要素または SVG 要素と結びつけている。

完成した横棒グラフ

http://www.openspc2.org/reibun/D3.js/code/graph/horizontal-bar/0001/index.html

横棒グラフを SVG で表示する

・D3.js でよく使われる表現手段の 1 つが SVG
・SVG にするとこんな感じになる

```
var list = [10, 30, 5, 60, 40, 78, 56, 30, 24, 80];
var svgWidth = 640; // SVG 領域の横幅
var svgHeight = 480;   // SVG 領域の縦幅
d3.select("#myGraph").append("svg")
    .attr("width", svgWidth).attr("height", svgHeight)
    .selectAll("rect")  // SVG での四角形を示す要素を指定
    .data(list) // データを設定
    .enter()
    .append("rect") // SVG での四角形を示す要素を生成
.attr("x", 0)   // 横棒グラフなので X 座標は 0。これは SVG 上での座標
.attr("y", function(d,i){   // Y 座標を配列の順序に応じて計算
    return i * 18;
})
.attr("width", function(d){ // 横幅を配列の内容に応じて計算
    return (d*2) +"px";
})
.attr("height", "16")   // 棒グラフの高さを指定
.attr("style", "fill:rgb(255,0,0)") // 棒グラフの色を赤色に設定
```

この程度なら DOM 使っても同じ

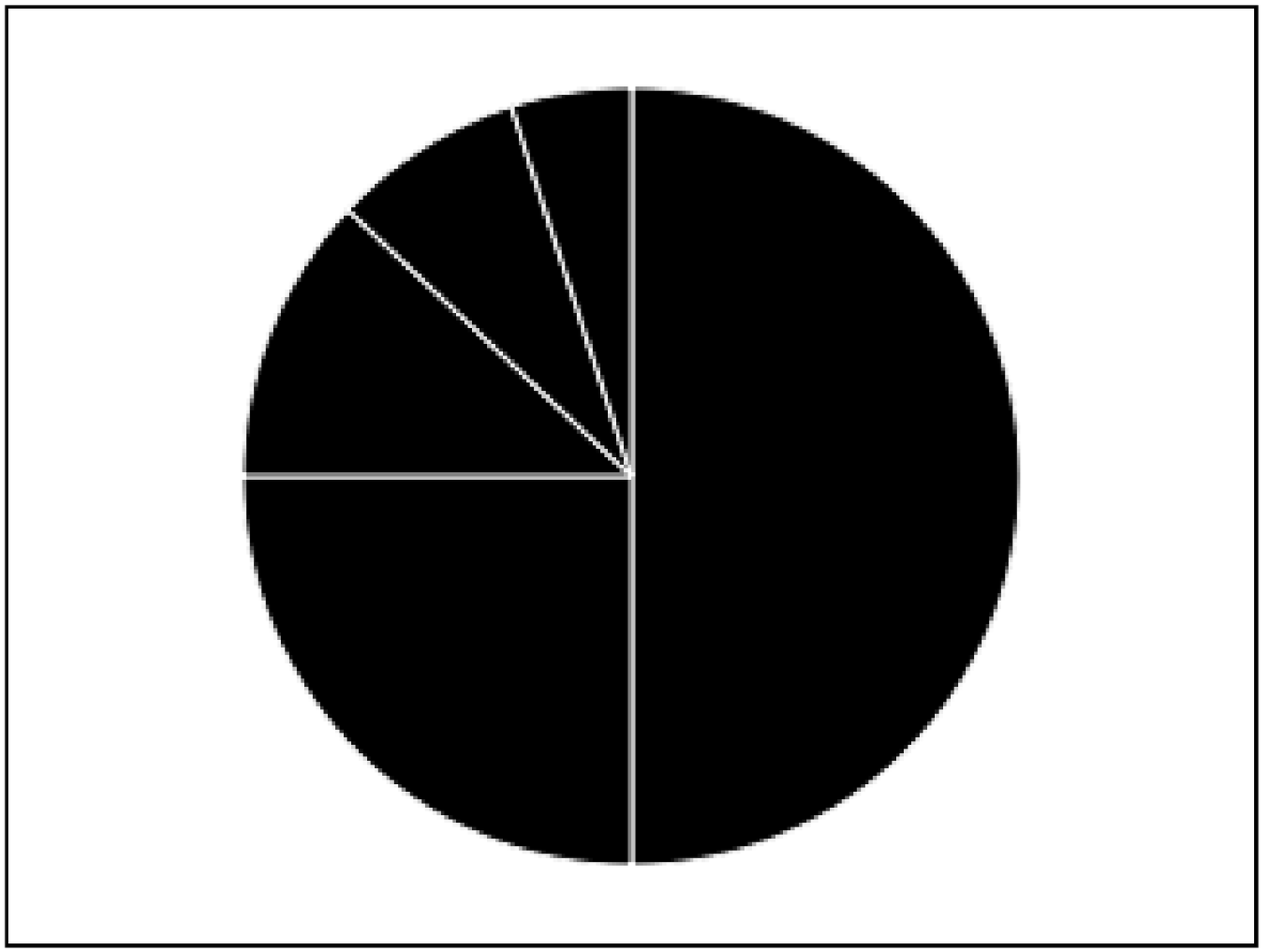
円グラフを SVG で表示する

・DOM では難しいのが円グラフ
・ここは SVG の出番
・しかし、SVG には円弧を描く要素がない
・どうするか？
・D3.js に計算してもらう
・レイアウト計算をする機能があり
　円グラフは d3.layout.pie() を使う

```
var list = [50, 25, 12, 8, 5];
var svgWidth = 320; // SVG 領域の横幅
var svgHeight = 240;    // SVG 領域の縦幅
// SVG の表示領域を生成
var svg = d3.select("#myGraph").append("svg")
    .attr("width", svgWidth).attr("height", svgHeight)
// 円グラフを生成
var pie = d3.layout.pie()
    .value(function(d){ return d; })
// 円グラフのサイズを指定
var arc = d3.svg.arc().innerRadius(0).outerRadius(100);
// 円グラフを描画
svg.selectAll("path")
    .data(pie(list))
    .enter()
    .append("path") // 円弧はパスで指定する
    .attr("d", arc) // 円弧を設定
    .attr("stroke", "white")    // 円グラフの区切り線を白色にする
    .attr("transform", "translate("+svgWidth/2+", "+svgHeight/2+")")
```

実行すると、こうなる

D3.jsサンプル



円グラフを色分けする

・どうするか？

・あらかじめ配列に表示したい色を用意し

　fill 属性値にカラーを設定

```
var color = ["red", "blue", "#0a0", "#ffef80", "rgba(255, 0, 255, 0.5)"];

    .attr("fill", function(d, i){
        return color[i];
    })
```

D3.js が用意した色を使う

・d3.scale.category10() を使う
　これは戻り値がメソッドになる
・先ほどのコードは以下のようになる

```
var color = d3.scale.category10();  // 10 色を指定

    .attr("fill", function(d, i){
        return color(i);
    })
```

グラフをアニメーションさせる

・マウスオーバーで棒グラフが伸びる
・イベントの検知は on() メソッドを使う
　これは最新の jQuery と同じメソッド
　使い方もパラメーターも同じ

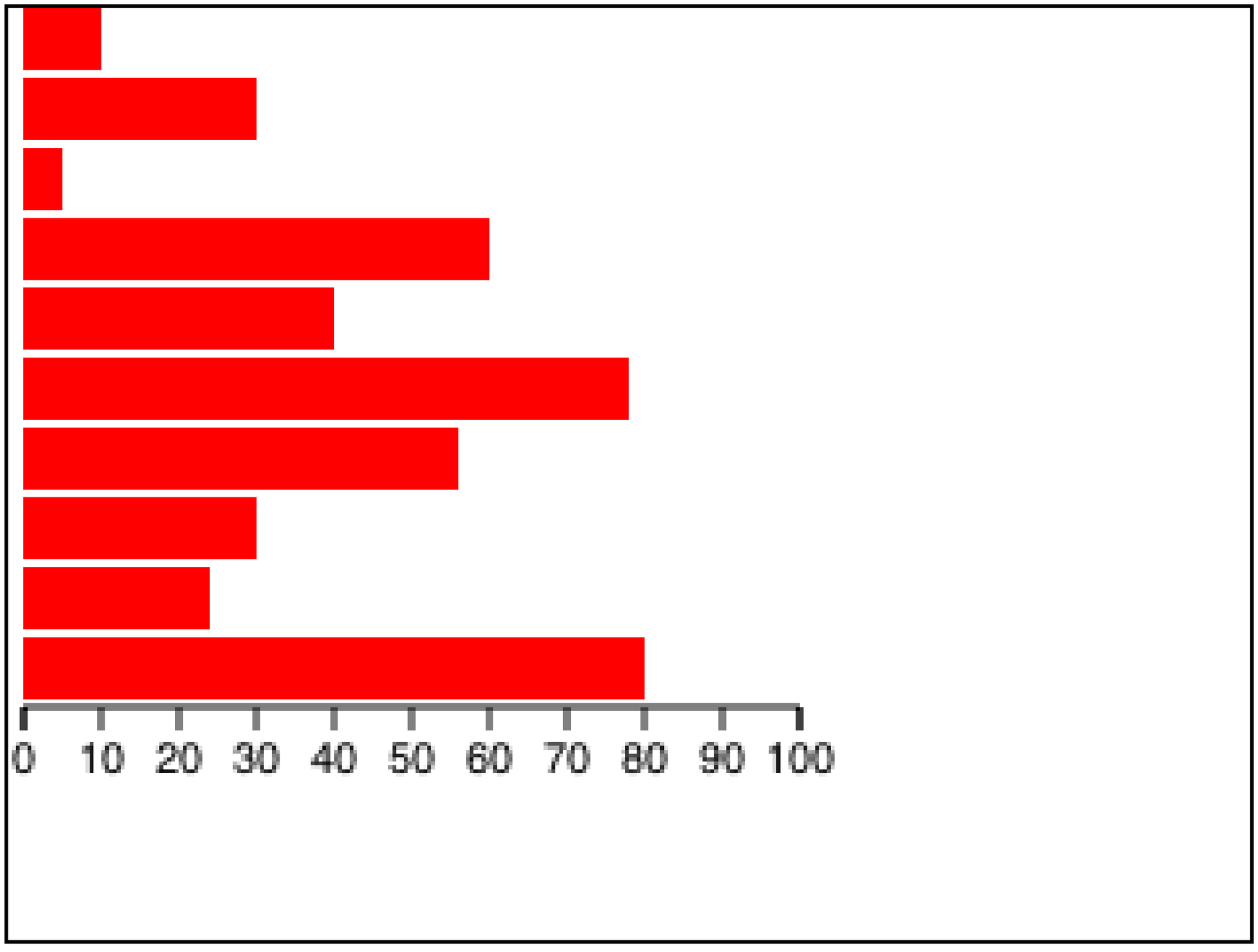
on(イベント名 , イベントハンドラ)

どうやってアニメーションさせるのか？

- 座標や幅はアニメーションの初期値にする
  棒グラフを伸びるようにするには幅を 0 に
- 初期値を設定したらイベントを検知
- イベントハンドラ内で transition() を
  記述した後に最後の棒グラフの横幅を
  指定すれば自動的にアニメーションする

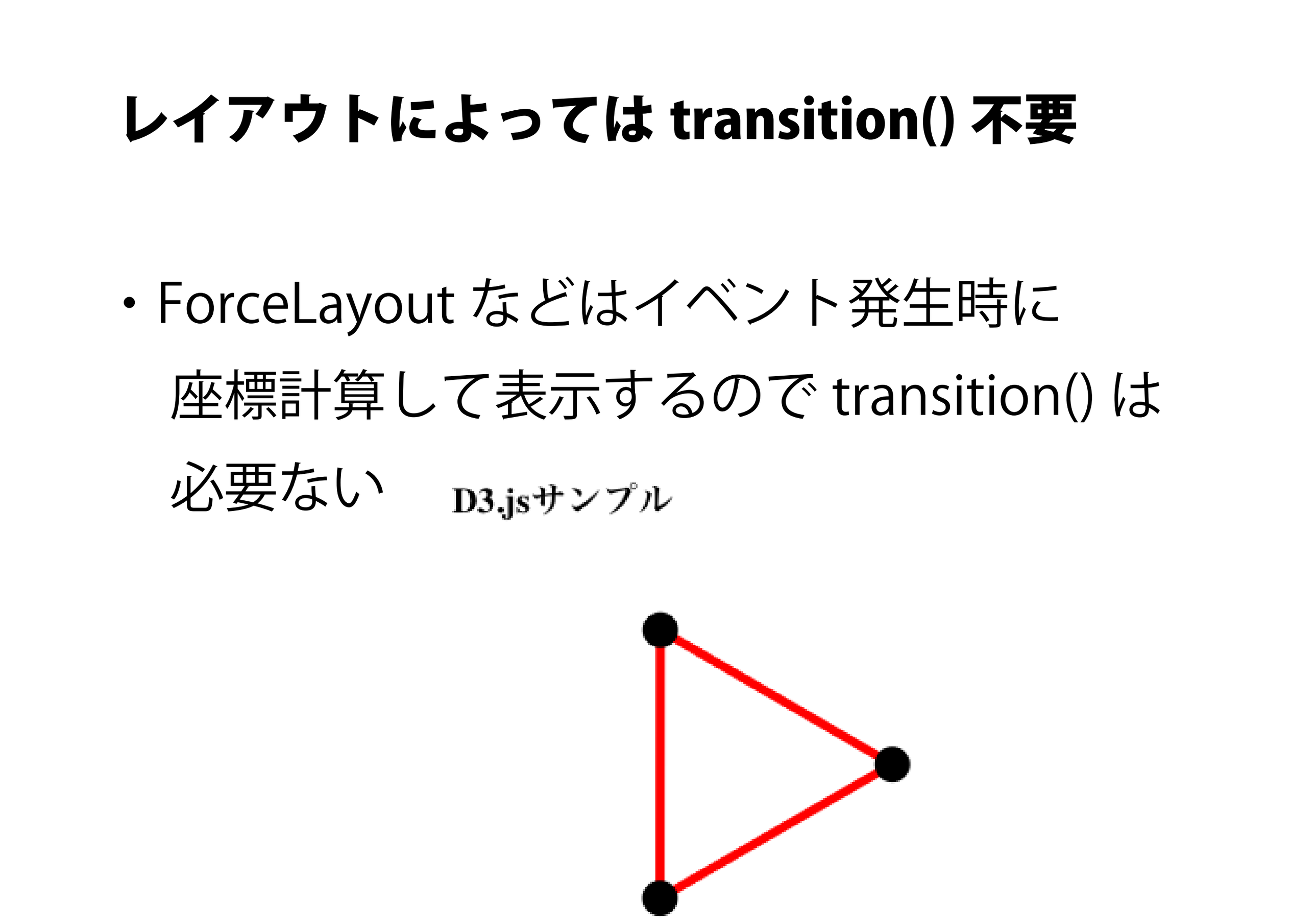
```
    .attr("width", function(d){ // 横幅を配列の内容に応じて計算
        return "0px";
    })

d3.select("svg").on("mouseover", function(){
    bar
        .transition()
        .attr("width", function(d){ // 横幅を配列の内容に応じて計算
            return (d*barScale) +"px";
        })
})
```

棒グラフが左から右に伸びる



レイアウトによっては transition() 不要

・ForceLayout などはイベント発生時に
座標計算して表示するので transition() は
必要ない



実際のデータを使ってみる

・長野県塩尻市の地区別世帯数
　塩尻市が公開しているデータを使用。

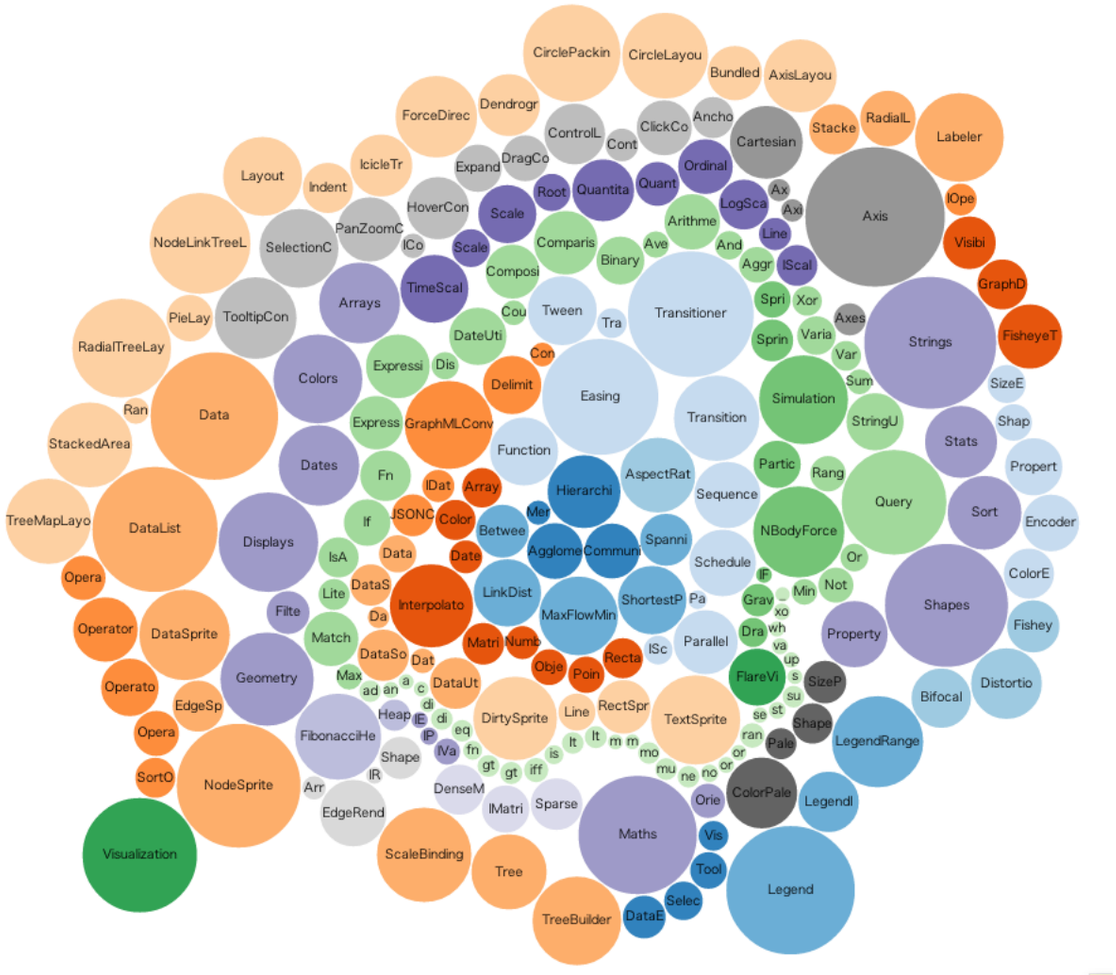
http://www.city.shiojiri.nagano.jp/gyosei/tokei/setaisu/jinkosetaisuu.html

・地区別に見た目でわかるようにバブルチャートを使って表示する

http://bl.ocks.org/mbostock/4063269

バブルチャートは、これ

・円の大きさで量やサイズを示す



D3.js は CSV ファイルを読み込める

・CSV データの読み込みは d3.csv()
・タブ区切りだと d3.tsv()
・書式は以下のようになっている

d3.csv(URL, 読み込んだ際に実行する関数)

バブルチャートを生成。

・バブルチャートは d3.layout.pack() で生成
・横幅と縦幅を指定する
・他にも円の半径を指定したり円と円の
　間隔を指定できる

```
var bubble = d3.layout.pack()
    .size([ 横幅 , 縦幅 ])
```

SVG の円を生成。

・文字を表示するため円と文字をグループ化
・.append("g") でグループが作れる
・グループ内に円と文字を生成する

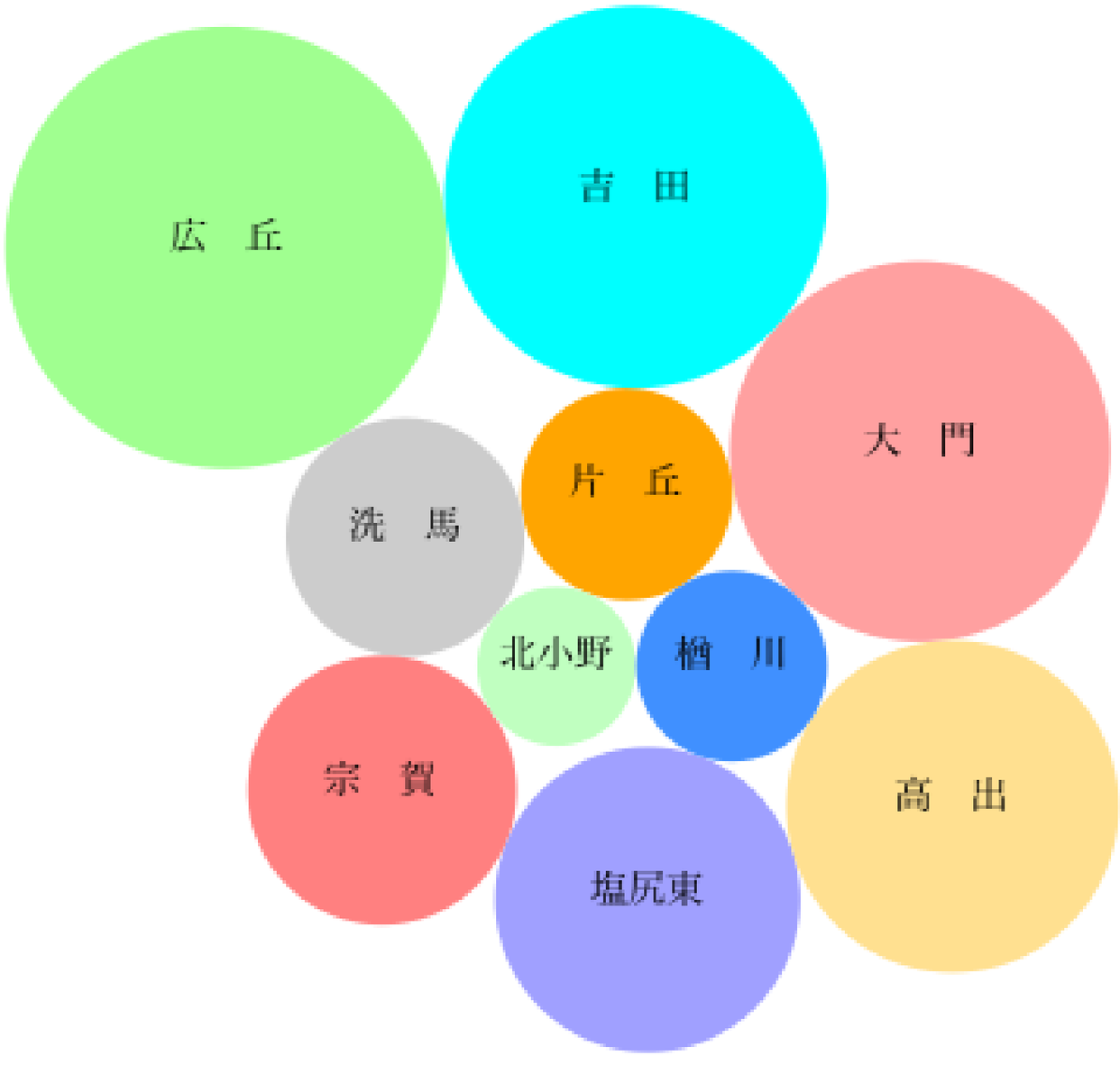
```
grp.append("circle")    // 円を生成する
    .attr("r", function(d){ // 円を指定した半径にする
        return d.r;
    })
```

文字を生成。

・append(”text”) で text 要素を生成
・text() メソッドで文字（地区名）を表示

```
grp.append("text")
    .attr("font-size", "9pt")
    .attr("fill", "black")
    .style("text-anchor", "middle")
 // データの中の className が地区名なので、それを出力
    .text(function(d) { return d.className; } )
```

塩尻市の地区別世帯数のバブルチャート



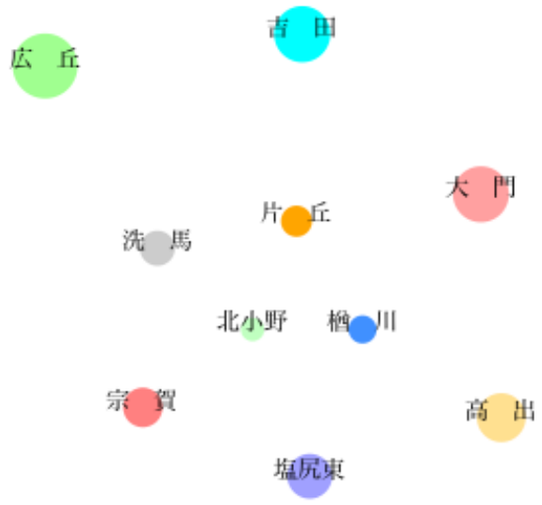
バブルチャートをアニメーションさせる

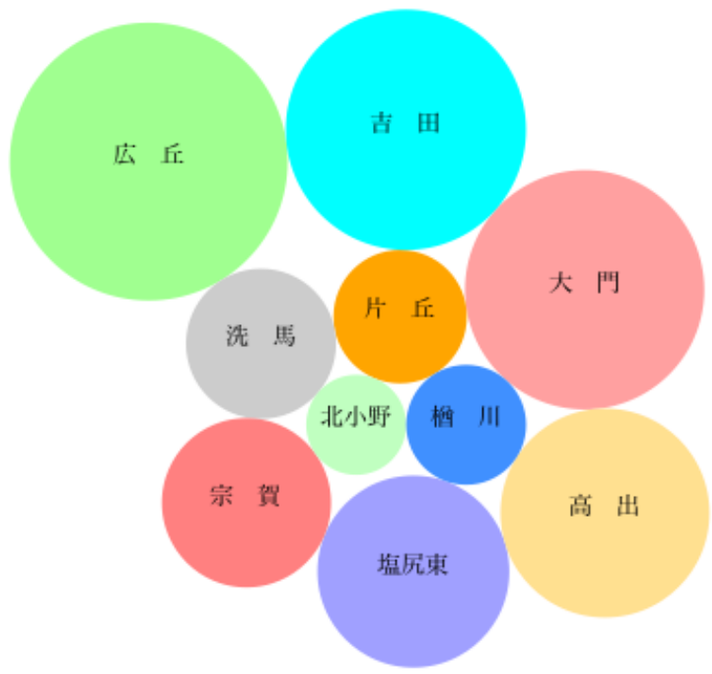
・transition() を使えば簡単にできる

```
d3.select("body").on("click", function(){
    circle
    .transition()   // トランジション指定
    .duration(2000) // 2 秒かけてアニメーション
    .attr("r", function(d){ // 円を指定した半径にする
        return d.r;
    })
});
```

マウスクリックでアニメーション

・これで完成
・年度別にアニメーションさせるようにすると
　わかりやすくなる





ここまでで D3.js の機能の 1/100 くらい

・D3.js は予想以上に機能が多い
・ライブラリなどもあって、組み合わせると
　さらに、いろいろできる

汎用性を持たせるようにすると大変だが必須

- データと機能が分離しているので上図に使う必要がある。CSS の属性をスクリプトで設定するのは、よくなさそう。
- データは生データは駄目で D3.js で使えるようにサーバー等で加工してから渡す
- インタラクティブにしないなら jqPlot など一般的なライブラリを使う方がよい

ここまでが１日で覚えた範囲

１週間分だと、かなりの量

頑張れば１ヶ月で全容把握できるのでは？

完