

2014年5月14日

Geekカフェ関内 ～もくもく会

@さくらWORKS <関内>

横浜市統計データの ビジュアライゼーション

合同会社 緑 I T 事務所

代表 小池 隆

統計データを地域課題の解決に

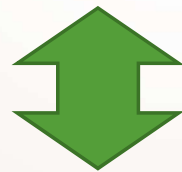
- ▶ 地域課題の見える化・解決の糸口の探求
- ▶ オープンデータ化による後押し
(再頒布・二次利用等の許諾)

<http://www.city.yokohama.lg.jp/ex/stat/>

The screenshot displays the Yokohama City Statistical Portal website. The browser address bar shows the URL www.city.yokohama.lg.jp/ex/stat/. The page header includes the city name '横浜市' and a search bar. The main content area is titled '統計情報の総合窓口 横浜市統計ポータルサイト'. Below the header, there are navigation tabs for 'トップ', '主な統計調査結果', '横浜市統計書', '大都市比較統計年表', '市民経済計算', and 'その他'. A sidebar on the left provides a 'カテゴリ' (Category) menu with links to '主な統計調査結果', '横浜市統計書', '大都市比較統計年表', '市民経済計算', and 'その他'. The main content area features a section for '横浜の人口' (Yokohama Population) with sub-sections for '推計人口' (Estimated Population), '登録者数' (Registered Users), 'むかしの人口' (Past Population), and 'お知らせ' (Notice). The '推計人口' section shows data for '平成26年5月1日現在の推計' (Estimated as of May 1, 2014): 人口: 3,708,122 人 (Population: 3,708,122 people) and 世帯数: 1,628,880 世帯 (Number of households: 1,628,880 households). Below this, there are three main data sections: 1. 人口ニュース (Population News) with a '月次更新' (Monthly Update) button, 2. 人口動態と年齢別人口 (Population Dynamics and Age-Specific Population) with a '年次更新' (Annual Update) button, and 3. 年齢(各歳・5歳階級)別男女別人口 (Age-Specific Male and Female Population by 5-Year Age Group) with a '年次更新' (Annual Update) button. A list of districts is provided at the bottom of the page.

データビジュアライゼーション

- プログラムでデータを視覚的に表現
 - ダイナミック（動的）
 - データの変化に即対応
 - インタラクティブ（対話的）
 - ユーザがデータを選択・など



インフォグラフィックス

- スタティック（静的）

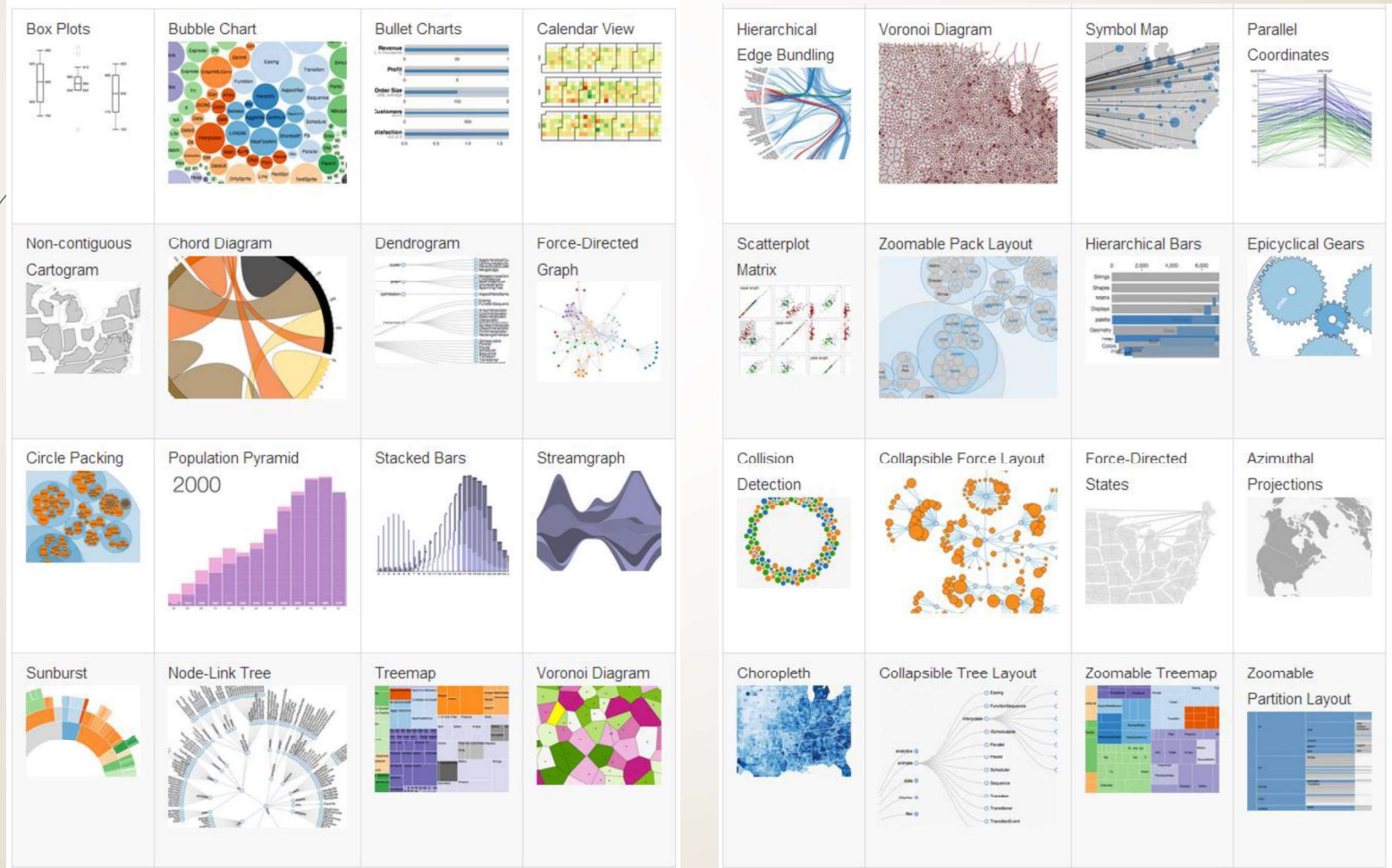
D3.js

- Data-Driven Documentsの略
- データビジュアライゼーションのためのJavaScriptライブラリ
 - Webブラウザ上で動作
 - データ(CSV,JSON,..)バインド
 - SVG(Scalable Vector Graphics)描画
 - ジオマッピング
 - チェーン構文
 - ループなしで記述できる

D3 gallery

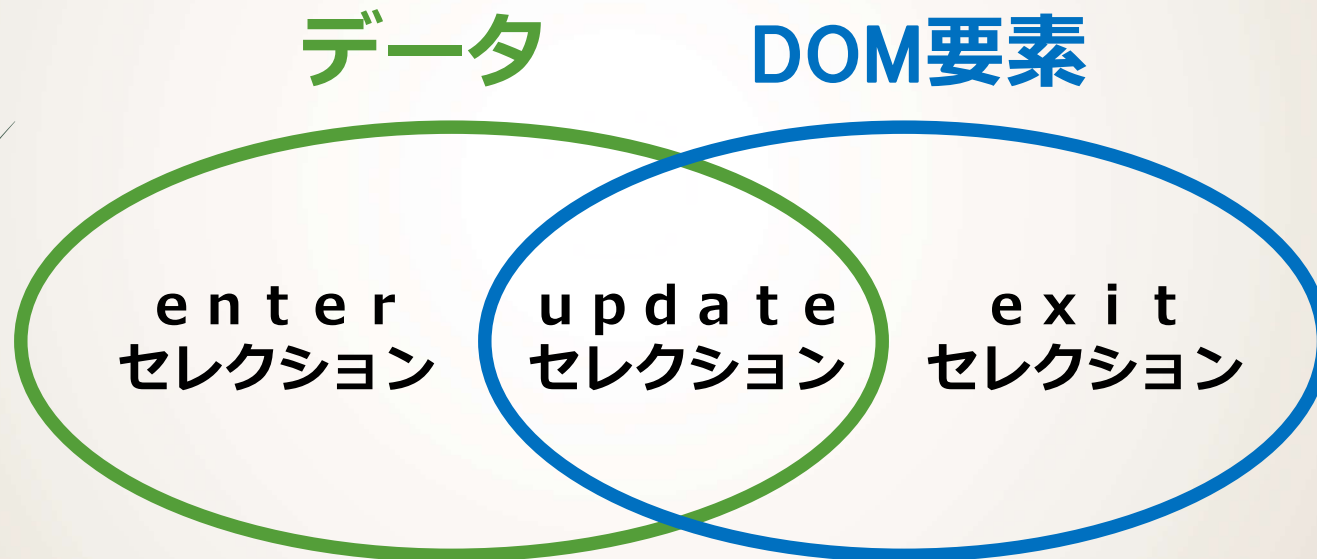
<https://github.com/mbostock/d3/wiki/Gallery>

Visual Index



データバインド

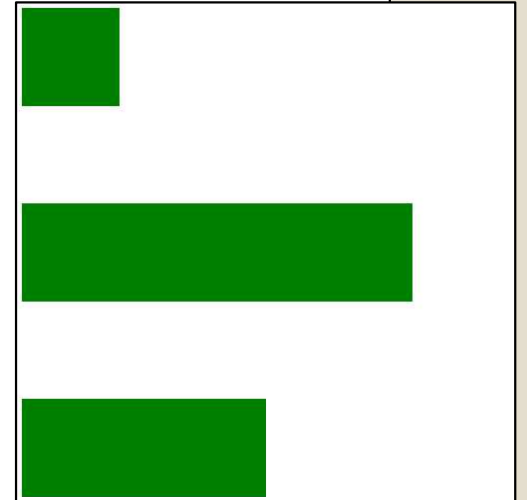
- データをDOM要素とバインド
 - バインドしてセレクションを得る



- グラフ作成の基本はenterセレクションに対応するDOM要素の追加

棒グラフのサンプルコード

```
<svg width="500" height="500"></svg>
<script type="text/javascript">
  var svg = d3.select("svg")
  svg.selectAll("rect")
    .data([100, 400, 250])
    .enter()
    .append("rect")
    .attr("x", 0)
    .attr("y", function(d,i){return i*200})
    .attr("width", function(d,i){return d+"px"})
    .attr("height", "100px")
    .attr("fill", "green");
</script>
```



サンプル①昼夜間人口の可視化

- 横浜市各区の昼夜間人口を棒グラフ化
- 1時間毎の人口変化をアニメーション
 - 実際のデータは昼間人口と夜間人口
 - 正弦曲線を描いて変化するよう計算
- 技術概要
 - CSVファイルのデータバインド
 - rect要素による棒グラフの描画
 - スケール

サンプル①データ

- ▶ 横浜市が公開するExcelファイルからCSVファイルを作成

```
ward, nighttime, daytime  
鶴見区, 272178, 250323  
神奈川区, 233429, 233168  
西区, 94867, 170450  
:
```

JavaScriptで読み込むと
オブジェクトの配列に

```
[{ward:"鶴見区", nighttime:"272178", daytime:"250323"},  
{ward:"神奈川区", nighttime:"233429", daytime:"233168"},  
{ward:"西区", nighttime:"94867", daytime:"170450"},  
:  
]
```

サンプル①プログラム解説

棒グラフのrect要素追加

CSVファイル読み込み

無名のコールバック関数

```
d3.csv("yokohama01.csv", function(data) {  
  dataset = data; //保存しておく  
  svg.selectAll("rect")  
    .data(dataset) } rect要素とデータをバインド  
  .enter()  
  .append("rect"); } 各区のrect要素が作られる  
});
```

サンプル①プログラム解説 つづき

■ スケール

- 入力ドメイン(domain) – 人口
- 出力レンジ(range) – ピクセル

```
var scale = d3.scale.linear()  
  .domain([0, 380000])  
  .range([0, 420]);
```

0～38万人を0～420ピクセルに変換する関数

サンプル①プログラム解説 つづき

■ アニメーション

```
setInterval( function() {  
  svg.selectAll("rect") } rect要素とデータをバインド  
  .data(dataset)           (updateセレクション取得)  
  .attr("x", 80)  
  .attr("y", function(d, i) {return (i * 25);})  
  .attr("fill", function() {return "rgba(...)";}) 色を変える  
  .attr("height", "20px")  
  .attr("width", function(d) { } 棒グラフの  
    var pop = ...d.daytime...; } 長さを  
    return scale(pop) + "px"; } 変える  
  });           (スケールを使用)  
}, 1000);
```

サンプル②緑被率の可視化

- 横浜市各区の緑被率を地図表示
 - ボタンで年度を選択
 - 緑被率によって不透明度を変化
 - 緑の多さを色の濃さで表現
- 技術概要
 - CSVファイルのデータバインド
 - GeoJSONによるベクトル地図描画

サンプル②データ

- ▶ 横浜市が公開するExcelファイルからCSVファイルを作成

ward,	1975,	1982,	1987,	1992,	1997,	2001,	2004,	2009
鶴見区,	20.9,	18,	17,	15.5,	15.3,	14.8,	14.7,	13.7
神奈川区,	27.4,	26.2,	25.9,	24.3,	23,	24.1,	23.5,	22.6
西区,	11.7,	11.6,	11.2,	10.9,	11.4,	12.3,	13.1,	11.2
中区,	19.6,	16.6,	17.1,	15.8,	15.2,	14.8,	15.2,	14.3
南区,	34.4,	23.9,	20.4,	17.8,	17.2,	15.6,	16,	15.4
:								

行と列を転置

年度を元号から西暦に変更

1987年までの青葉区は緑区の値を使用

1987年までの都築区は港北区の値を使用

サンプル②GeoJSONファイル

- 国土交通省提供のSHAPE形式データをGeoJSON形式へ変換（横浜市のみ抽出）

```
{ "type": "FeatureCollection",  
  "features": [  
    { "type": "Feature",  
      "properties": { "N03_001": "神奈川県",  
                    "N03_002": null,  
                    "N03_003": "横浜市",  
                    "N03_004": "青葉区",  
                    "N03_007": "14117" },  
      "geometry": { "type": "Polygon",  
                   "coordinates": [ [ [139.567873, 35.571416],  
                                     [139.567766, 35.571269],  
                                     [139.567678, 35.571164],  
                                     :  
                                     ] ] ]
```

サンプル②プログラム解説

▶ 凡例の表示

```
var legend = [70,60,50,40,30,20,10];  
svg.selectAll("rect")  
  .data(legend)  
  .enter()  
  .append("rect")  
  .attr("x", "500px")  
  .attr("y", function(d, i) { return ((15*i)+50) + "px"; })  
  .attr("width", "50px")  
  .attr("height", "15px")  
  .attr("fill", "green")  
  .attr("opacity", function(d, i) { return d/100; });
```

7つのrectが作られる

iにはインデックスが入る
0,1,2,...

dにはデータが入る
70,60,50,...

サンプル②プログラム解説 つづき

- ➡ CSVのデータとJSONのデータを突き合せ
JSONデータに緑被率の値を追加

```
d3.csv("yokohama02.csv", function(data) {  
  d3.json("yokohama.json", function(json) {  
    for(var i=0; i<data.length; i++) {  
      for(var j=0; j<json.features.length; j++) {  
        if( data[i].ward ==  
          json.features[j].properties.N03_004 ) {  
          greens = data[i];  
          for(var year in greens) {  
            if(parseInt(year)) {  
              json.features[j].properties[year]=greens[year];  
            }  
          }  
        }  
      }  
    }  
  }  
})
```

サンプル②プログラム解説 つづき

- ▶ path要素を追加し、地図を描画

```
svg.selectAll("path")  
  .data(json.features)  
  .enter()  
  .append("path")  
  .attr("d", path)  
  .style("fill", "green")  
  .style("stroke", "gray")  
  .style("stroke-width", "0.5px");
```

サンプル②プログラム解説 つづき

- ▶ ボタンのonClickのハンドラで pathの不透明度を変化させる

```
function changeYear(y) {  
  svg.selectAll("path")  
    .data(jsonData.features)  
    .style("opacity", function(feat) {  
      return feat.properties[y]/100;  
    });  
}
```

CSVから取得してJSONに
格納しておいた緑被率

ご清聴ありがとうございました

midoriit.com