合同会社 緑ＩＴ事務所

Midori IT Office, LLC

データビジュアライゼーション(4)

この記事は1年以上前に書かれました。
内容が古くなっている可能性がありますのでご注意下さい。

今回は、ビジュアライゼーション(1)で紹介したD3.jsを使用して、動きのあるビジュアライゼーションにチャレンジしようと思います。

まずは、横浜市統計ポータルサイトで提供されている、昼夜間人口、流出入人口及び人口密度のExcelファイルから、以下のCSVファイルを作成します。

```
ward,nighttime,daytime
鶴見区,272178,250323
神奈川区,233429,233168
西区,94867,170450
中区,146033,243277
南区,196153,154387
港南区,221411,173691
保土ケ谷区,206634,173514
旭区,251086,197891
磯子区,163237,136711
金沢区,209274,195740
港北区,329471,309610
緑区,177631,146647
青葉区,304297,234794
都筑区,201271,193939
戸塚区,274324,238630
栄区,124866,97103
泉区,155698,121197
瀬谷区,126913,104258
```

横浜市の各行政区ごとの、昼夜の人口変化をビジュアライズしてみます。

HTMLファイルの完全なコードは以下のとおりです。

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8">
  <title>Yokohama Data Visualization</title>
  <script type="text/javascript" src="d3.js"></script>
  <style type="text/css">
    .axis path,
    .axis line {
      fill: none;
      stroke: black;
      shape-rendering: crispEdges;
    }
    .axis text {
      font-size: 12px;
    }
</style>
</head>
<body>
  <div></div><br />
  <script type="text/javascript">

    var dataset;
    var w = 500;
    var h = 500;
    var hour = 0;

    var div = d3.select("div");
    div.text(hour + "時");

    var svg = d3.select("body")
      .append("svg")
      .attr("width", w)
      .attr("height", h);

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

```
      .range([0,420]);

    d3.csv("yokohama01.csv", function(data) {

      dataset = data;

      // ラベル
      svg.selectAll("text")
        .data(dataset)
        .enter()
        .append("text")
        .text(function(d) {
          return d.ward;
        })
        .attr("x", 0)
        .attr("y", function(d, i) {
          return (i * 25) + 16;
        })
        .attr("font-size", "14px");

      // 棒グラフの準備
      svg.selectAll("rect")
        .data(dataset)
        .enter()
        .append("rect");

      var axis = d3.svg.axis()
        .scale(scale)
        .orient("bottom");

      svg.append("g")
        .attr("class", "axis")
        .attr("transform", "translate(80,450)")
        .call(axis);

    });

    setInterval( function() {

      // 0時(24時)＝1、12時＝0、6時と18時＝0.5となる係数
      fct = (Math.cos(Math.PI * (hour/12))+1) / 2;
      div.text(hour + "時");

      // グラフ描画更新
      svg.selectAll("rect")
        .data(dataset)
        .attr("x", 80)
        .attr("y", function(d, i) {
          return (i * 25);
        })
        .attr("fill", function() {
          return "rgba(" + Math.round((1-fct)*255) +",0," + Math.round(fct*255) + ",1.0)";
        })
        .attr("height", "20px")
        .attr("width", function(d) {
          var pop = d.daytime - (fct * (d.daytime - d.nighttime));
          return scale(pop) + "px";
        });

      hour ++;
      if(hour == 24) {
        hour = 0
      }

    }, 1000);

  </script>
</body>
</html>
```

まず、headタグの中でD3.jsを読み込み、bodyの先頭で時刻を表示するためのdiv要素を用意しています。
JavaScriptでは、まず

```
var div = d3.select("div");
div.text(hour + "時");

var svg = d3.select("body")
  .append("svg")
  .attr("width", w)
  .attr("height", h);

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

divに時刻の初期値を表示し、幅と高さが500ピクセルのSVG（Scalable Vector Graphics）要素を作成し、スケールを定義しています。スケールは、0から38万までの人口の値を、0から420までのピクセル数に変換するためのものです。

次に、

```
d3.csv("yokohama01.csv", function(data) {
```

先ほどのCSVファイルを読み込んで処理します。function(data) { 以降が処理の内容です。
まず、

```
dataset = data;
```

読み込んだデータを後で利用できるよう、グローバル変数に代入しておきます。

次に、行政区名のラベルを表示します。

```
svg.selectAll("text")
  .data(dataset)
  .enter()
  .append("text")
  .text(function(d) {
    return d.ward;
  })
  .attr("x", 0)
  .attr("y", function(d, i) {
    return (i * 25) + 16;
  })
  .attr("font-size", "14px");
```

この時点ではまだ存在しないtext要素とデータセットをバインドし、enter()でデータに対応するプレースホルダを取得し、append(“text”)でtext要素を追加します。
CSVファイルのward列の値をtext要素のテキストとします。
ラベルのx座標は0固定、y座標は25ずつ増加させます。

次に、棒グラフの棒を表すrect要素の準備をします。

```
svg.selectAll("rect")
  .data(dataset)
  .enter()
  .append("rect");
```

この時点ではまだ存在しないrect要素とデータセットをバインドし、enter()でデータに対応するプレースホルダを取得し、append(“rect”)でrect要素を追加します。棒グラフは後で描画します。
さらに、軸と目盛を追加します。

```
var axis = d3.svg.axis()
  .scale(scale)
  .orient("bottom");

svg.append("g")
  .attr("class", "axis")
  .attr("transform", "translate(80,450)")
  .call(axis);
```

作成してあるスケールを用いて軸を作り、svg要素の[80, 450]の位置に表示します。
以上で、CSVファイル読み込みのコールバック処理は終わりです。

動きのあるビジュアライゼーションのために、JavaScriptのsetIntervalを使用し、

```
setInterval( function() { }, 1000);
```

1秒毎に描画を行い、24秒で24時間分の変化を表すようにします。とは言っても、元となるデータは「昼間人口」と「夜間人口」であり、1時間毎の人口データはありません。そこで、深夜0時の人口を「夜間人口」、12時の人口を「昼間人口」とし、その間はサインカーブを描いて変化するという仮定で、毎時の推定人口を計算します。

```
fct = (Math.cos(Math.PI * (hour/12))+1) / 2;
```

が、そのための係数の計算です。0時には1、12時には0、6時と18時には0.5になります。
0時や12時の前後では係数の値の変化は少なく、6時や18時の前後では変化が大きくなります。

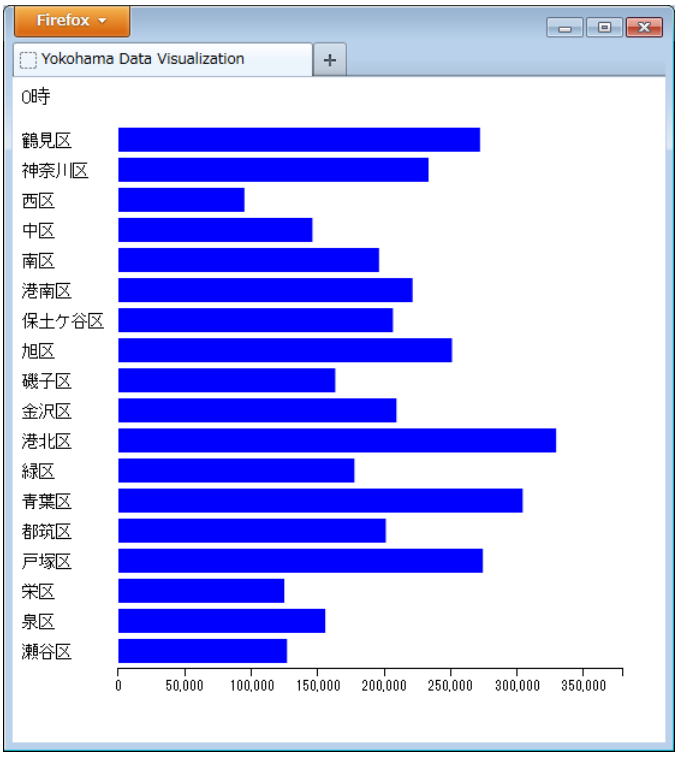
```
svg.selectAll("rect")
  .data(dataset)
  .attr("x", 80)
  .attr("y", function(d, i) {
    return (i * 25);
  })
  .attr("fill", function() {
    return "rgba(" + Math.round((1-fct)*255) +",0," + Math.round(fct*255) + ",1.0)";
  })
  .attr("height", "20px")
  .attr("width", function(d) {
    var pop = d.daytime - (fct * (d.daytime - d.nighttime));
    return scale(pop) + "px";
  });
```

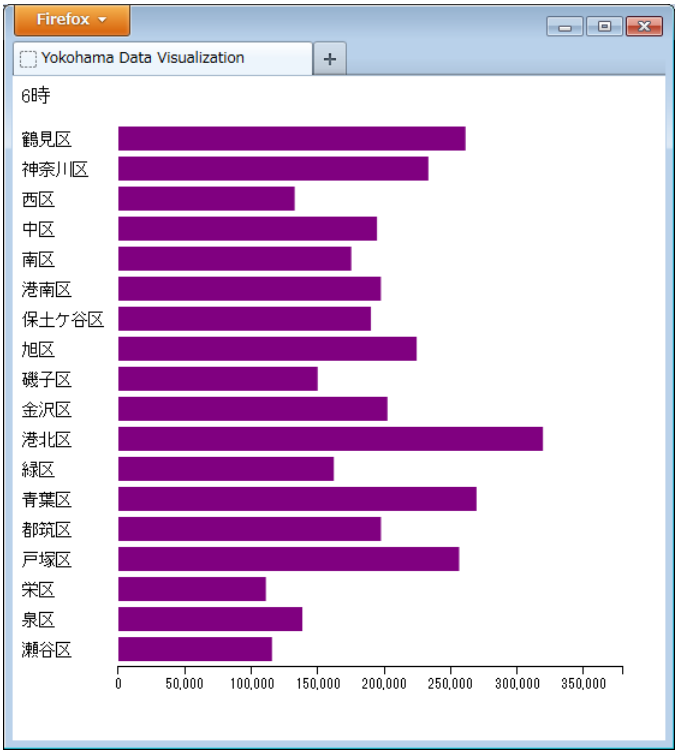
棒グラフを表すrect要素のx座標は80固定、y座標は25ずつ増加させます。
棒の色は、深夜0時には青（rgba(0,0,255,1.0)）、12時には赤（rgba(255,0,0,1.0)）、その間では青と赤が混じりあうようにします。
棒の長さは、係数を用いて求めた推定人口からスケールを用いてピクセル数に変換します。
最後に、24時間でループするよう、

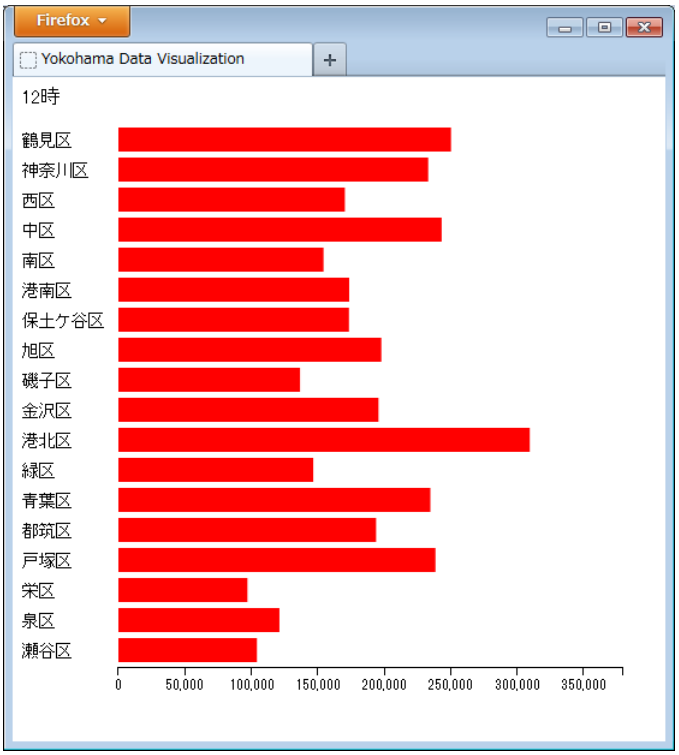
```
hour++;
if(hour == 24) {
  hour = 0
}
```

時刻のカウントアップとリセットをします。

0時、6時、12時の表示は以下のようになります。







実際の動きは、こちらのHTMLファイルで見て下さい。

動きがあると、西区と中区だけは他の区と逆に動く（夜間人口より昼間人口が多い）こと、神奈川区は昼夜でほとんど変化がないことが良く分かります。

カテゴリー: オープンデータ, ビジュアライゼーション | タグ: D3.js | 投稿日: 2014年4月12日
[https://midoriit.com/2014/04/%e3%83%87%e3%83%bc%e3%82%bf%e3%83%93%e3%82%b8%e3%83%a5%e3%82%a2%e3%83%a9%e3%82%a4%e3%82%bc%e3%83%bc%e3%82%b7%e3%83%a7%e3%83%b34.html] | 投稿者: 小池隆