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(5)

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。

前回から少し間が空いてしまいましたが、引き続きD3.jsを使用して横浜市の統計データをビジュアライズします。

今回の題材は、横浜市の緑被率です。
まずは、横浜市統計ポータルサイトで提供されている、行政区別緑被率のExcelファイルから、以下のCSVファイルを作成します。

```
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
港南区,31.9,28.4,24.8,23.3,21.3,22.4,23,22.9
保土ケ谷区,40.2,36.9,35.3,33.8,32.5,32.5,32.2,31.1
旭区,43.9,42,40.3,38.3,36.1,37.8,37.1,36
磯子区,39.2,33.6,29.6,28.2,27.7,26.4,27.8,27.6
金沢区,50.2,38.8,37.4,33.2,33.7,31.5,31.8,31.8
港北区,49.6,42.6,34.2,35.3,31.8,28.2,27.8,26.5
緑区,58.2,50.9,41.5,52.2,50.2,44.6,44.3,42.8
青葉区,58.2,50.9,41.5,38.7,37.8,34.5,34,31.4
都筑区,49.6,42.6,34.2,34.7,38.1,38.1,36.1,33.6
戸塚区,50.9,47.7,45,42.2,40.4,38.5,39,37.8
栄区,44,47.4,43.3,41.6,40.7,41.7,42.1,41.8
泉区,61.8,52.6,50.7,45.9,44.3,41.9,41.1,39
瀬谷区,45.8,42.9,40.3,38.4,35.8,36.6,35.9,35.1
```

行と列を入れ替え、年度を元号から西暦に変更しました。また、1987年までは青葉区と都筑区の値が入っていませんので、青葉区には緑区の値を、都筑区には港北区の値を入れました。

次に、D3.jsで横浜市の地図を描くためのGeoJSONファイルを作成します。元となるSHAPE形式のファイルは、国土交通省の行政区域データのページで入手します。
ダウンロードしたN03-130401_14_GML.zipファイルを解凍し、N03-13_14_130401.shpファイルをogr2ogrコマンドでGeoJSON形式に変換します。

```
ogr2ogr -f GeoJSON -where N03_003=\"横浜市\" yokohama.json
N03-13_14_130401.shp
```

元となるSHAPE形式のファイルには、横浜市以外も含まれているため、「-where N03_003=\"横浜市\"」を付けて横浜市のみ抽出しています。
ogr2ogr2コマンドについては、本連載の第1回を参照して下さい。

以上でデータは揃いました。
HTMLファイルの完全なリストは以下のとおりです。

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8">
  <title>Yokohama Data Visualization</title>
  <link rel="stylesheet" href="bootstrap.min.css">
  <script type="text/javascript" src="d3.js"></script>
</head>
<body style="padding: 10px">
  <div class="btn-group"></div><br />
  <svg width="550px" height="550px"></svg>
  <script type="text/javascript">

    var jsonData;
    var legend = [70,60,50,40,30,20,10];

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

    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", "limegreen")
  .attr("opacity", function(d, i) {
    return d/100 + 0.2;
  });

svg.selectAll("text")
  .data(legend)
  .enter()
  .append("text")
  .attr("x", "510px")
  .attr("y", function(d,i) {
    return ((15 * i)+63) + "px";
  })
  .text(function(d) {
    return d + "%";
  });

var projection = d3.geo.equirectangular()
  .center([139.73, 35.48])
  .scale([90000]);
var path = d3.geo.path().projection(projection);

d3.csv("yokohama02.csv", function(data) {
  var years = [];
  for(var year in data[0]) {
    if(year != "ward") {
      years.push(year);
    }
  }

  var div = d3.select("div");
  var button = div.selectAll("button")
    .data(years)
    .enter()
    .append("button")
    .attr("class", "btn btn-default")
    .attr("onClick", function(d) {
      return "onClick=changeYear(" + d + ")";
    })
    .text( function(d) {
      return d;
    });

});

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(year != "ward") {
              json.features[j].properties[year] = greens[year];
            }
          }
        }
      }
    }

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

    jsonData = json;
  });
});

function changeYear(y) {
  svg.selectAll("path")
    .data(jsonData.features)
    .style("opacity", function(feat) {
      return feat.properties[y]/100 + 0.2;
    });
}
```

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

以下、主なところを解説します。まず、headでは

```
<link rel="stylesheet" href="bootstrap.min.css">
<script type="text/javascript" src="d3.js"></script>
```

BootstrapのCSSと、D3.jsのスクリプトを読み込んでおきます。
bodyの先頭では

```
<div class="btn-group"></div><br />
<svg width="550px" height="550px"></svg>
```

ボタンを表示するためのdivと、D3で描画するためのsvg要素をあらかじめ用意しておきます。

スクリプトでは、まずrect要素とtext要素で凡例を描きます。

```
var legend = [70,60,50,40,30,20,10];

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

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", "limegreen")
  .attr("opacity", function(d, i) {
    return d/100 + 0.2;
  });

svg.selectAll("text")
  .data(legend)
  .enter()
  .append("text")
  .attr("x", "510px")
  .attr("y", function(d,i) {
    return ((15 * i)+63) + "px";
  })
  .text(function(d) {
    return d + "%";
  });
```

配列legendとデータバインドし、rect要素を作成して緑色（LimeGreen）で塗りつぶします。緑被率の値を利用して不透明度（opacity）を指定します。opacityは0で完全に透明、1で完全に不透明です。色が薄すぎないよう、+0.2のゲタを履かせています。
さらに、rect要素に重なるようにtext要素を追加し、legendの値に%を付けて表示します。

次に、GeoJSONのデータを使用して地図を描くための準備として、

```
var projection = d3.geo.equirectangular()
  .center([139.73, 35.48])
  .scale([90000]);
var path = d3.geo.path().projection(projection);
```

投影法（projection）とパスジェネレータ（path）を定義します。

前準備の最後として、緑被率を表示する年度を選択するためのボタンを作成します。

```
d3.csv("yokohama02.csv", function(data) {
  var years = [];
  for(var year in data[0]) {
    if(year != "ward") {
      years.push(year);
    }
```

```
  }

  var div = d3.select("div");
  var button = div.selectAll("button")
    .data(years)
    .enter()
    .append("button")
    .attr("class", "btn btn-default")
    .attr("onClick", function(d) {
      return "onClick=changeYear(" + d + ")";
    })
    .text( function(d) {
      return d;
    });
});
```

はじめにCSVファイルを読み込み、for…inループで反復してプロパティ名を取得し、

```
{ward:"鶴見区",1975:"20.9",1982:"18",1987:"17",1992:"15.5",1997:"15.3",2001:"14.8",2004:"14.7",2009:"13.7"}
```

というオブジェクトから、

```
[1975,1982,1987,1992,1997,2001,2004,2009]
```

という年度の配列が得られます。
button要素と年度の配列をバインドしてボタンを作成します。ボタンのonClickでは、年度を引数としてchangeYear()関数を実行するようにします。

さて、いよいよ横浜市の地図の描画です。CSVファイルとGeoJSONのファイルを読み込み、

```
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(year != "ward") {
              json.features[j].properties[year] = greens[year];
            }
          }
        }
      }
    }
```

JSONのデータに緑被率のデータを追加します。

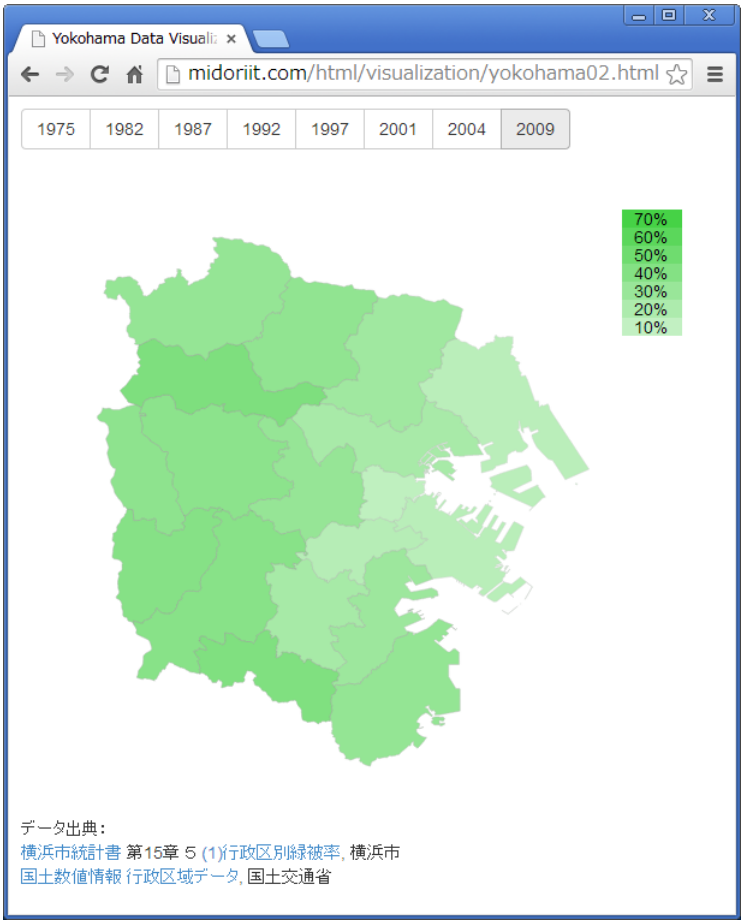
```
svg.selectAll("path")
  .data(json.features)
  .enter()
  .append("path")
  .attr("d", path)
  .style("fill", "limegreen")
  .style("stroke", "gray")
  .style("stroke-width", "0.5px");

jsonData = json;
```

path要素を追加して地図を描画し、緑被率を追加したJSONデータをグローバル変数に格納しておきます。この時点では不透明度を設定していないため、均一な緑色で塗りつぶされています。
最後に、ボタンのonClinkイベントのハンドラです。

```
function changeYear(y) {
  svg.selectAll("path")
    .data(jsonData.features)
    .style("opacity", function(feat) {
      return feat.properties[y]/100 + 0.2;
    });
}
```

すでに作成してあるpath要素と、グローバル変数に格納しておいたJSONデータをバインドし、引数で渡された年度の緑被率を不透明度に設定します。ここでも、+0.2のゲタを履かせています。

実行結果は以下のようになります。



2009年を選択した例です。緑区と栄区に比較的緑が多く残されていることが分かります。

こちらのHTMLファイルでは、緑被率を表示する年度をボタンで選択することができます。

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5月14日 [https://midoriit.com/2014/05/%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%b35.html] | 投稿者: 小池隆