

気象庁REST APIで取得したデータを D3.js で地図上にプロットする ハンズオン

2014年9月27日

先端IT活用推進コンソーシアム
クラウド・テクノロジー活用部会
リーダー 荒本道隆

タイムテーブル

- 14:30～16:00
 - 気象庁REST API をアクセスするソースコードの解説
 - D3.js & JavaScript
- 16:00～17:00
 - 各自で機能拡張する時間
- 17:00～17:30
 - 機能拡張したもの & その他の作ったものを発表

最初に

- APIになっているオープンデータはまだ少ない
 - 多くのオープンデータは、CSVやExcelで公開
 - CSVやExcelを地図上にプロットする手順
 - 前回のハンズオン資料を参照
 - http://cloud.aitc.jp/20140717_D3js/
- 地図上にプロットすると、見えてくることもある
 - 慣れれば、半日程度でプロットできる
 - 他のオープンデータと組み合わせるには、コードが書けた方が絶対良い

本日の目標

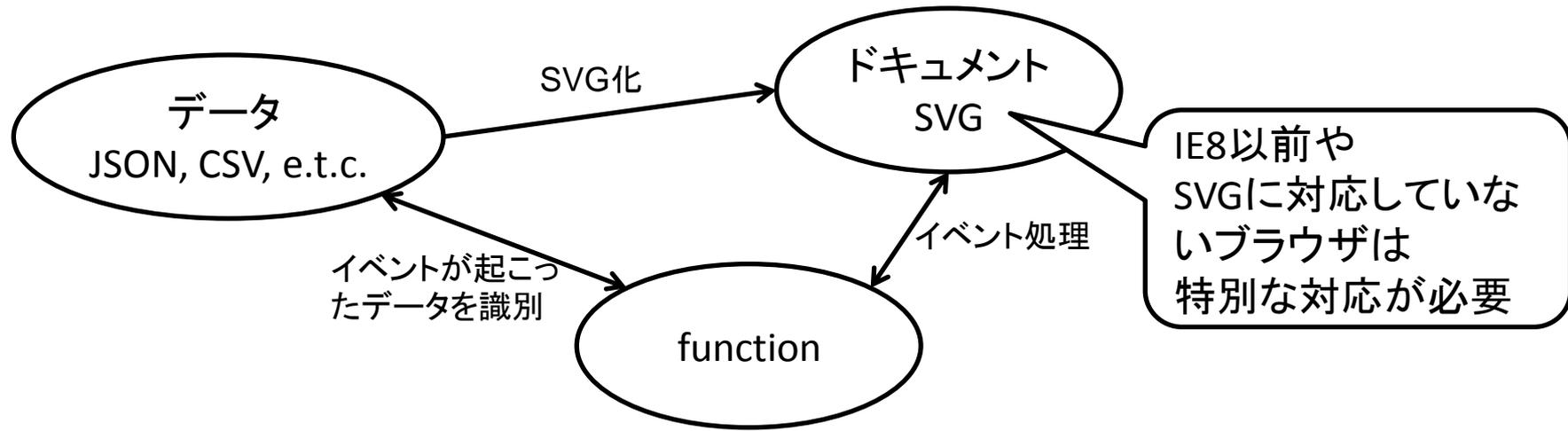
- 気象庁REST APIの使い方をマスターしてもらう
 - 実際にAPIを叩きまくってみる
- D3.jsでデータを可視化する手順を理解
 - 「地図上にプロットする」は、他でも使える
 - JavaScript、SVG(Scalable Vector Graphics)も身に付く

D3.js とは

D3.js (またはD3:Data-Driven Documents、旧:Protovis^[1])は、2011年に開発が始まった^[2][ウェブブラウザ](#)で動的コンテンツを描画する[JavaScriptライブラリ](#)である。[World Wide Web Consortium](#)準拠の[データ可視化](#)ツールとして、[Scalable Vector Graphics](#) (SVG)、JavaScript、[HTML5](#)、[Cascading Style Sheets](#)を最大限に活用している。その他多くのライブラリとは対照的に、最終的に出力された結果に視覚的な調整ができる。^[3]

ウィキペディアより

- データをドキュメント化して、関係も保持
– SVGを効率良く生成するためのライブラリです



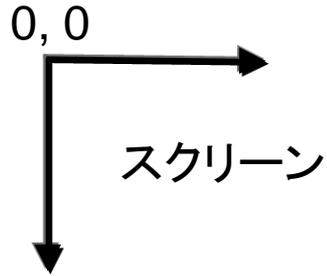
D3.jsを使ったビジュアライズ

- 今回は地図を軸に可視化してみる
 - D3.js は、地図描画の機能が充実
- 注意：地図は楽だけど、グラフを描くのは意外に面倒
 - 必要な部品がそろっているだけ
 - D3.js Examples → 各グラフを描くためのソースを参照
 - <https://github.com/mbostock/d3/wiki/Gallery>
 - 参考URL <http://postd.cc/what-d3js-is-not/>
 - D3.jsを使って簡単にグラフを書くためのライブラリ
 - <http://nvd3.org/>
- 参考書籍
エンジニアのための データ可視化[実践]入門
~D3.jsによるWebの可視化
 - <http://www.amazon.co.jp/dp/4774163260>
 - D3.js はほとんど出てこないけど、「可視化」の理解が深まります

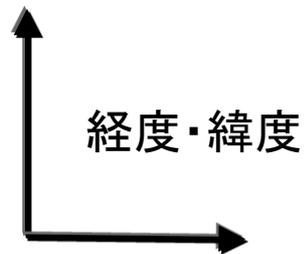
緯度・経度を扱う上での注意

0, 0
136.09, 36.77

900, 0
140.18, 36.77



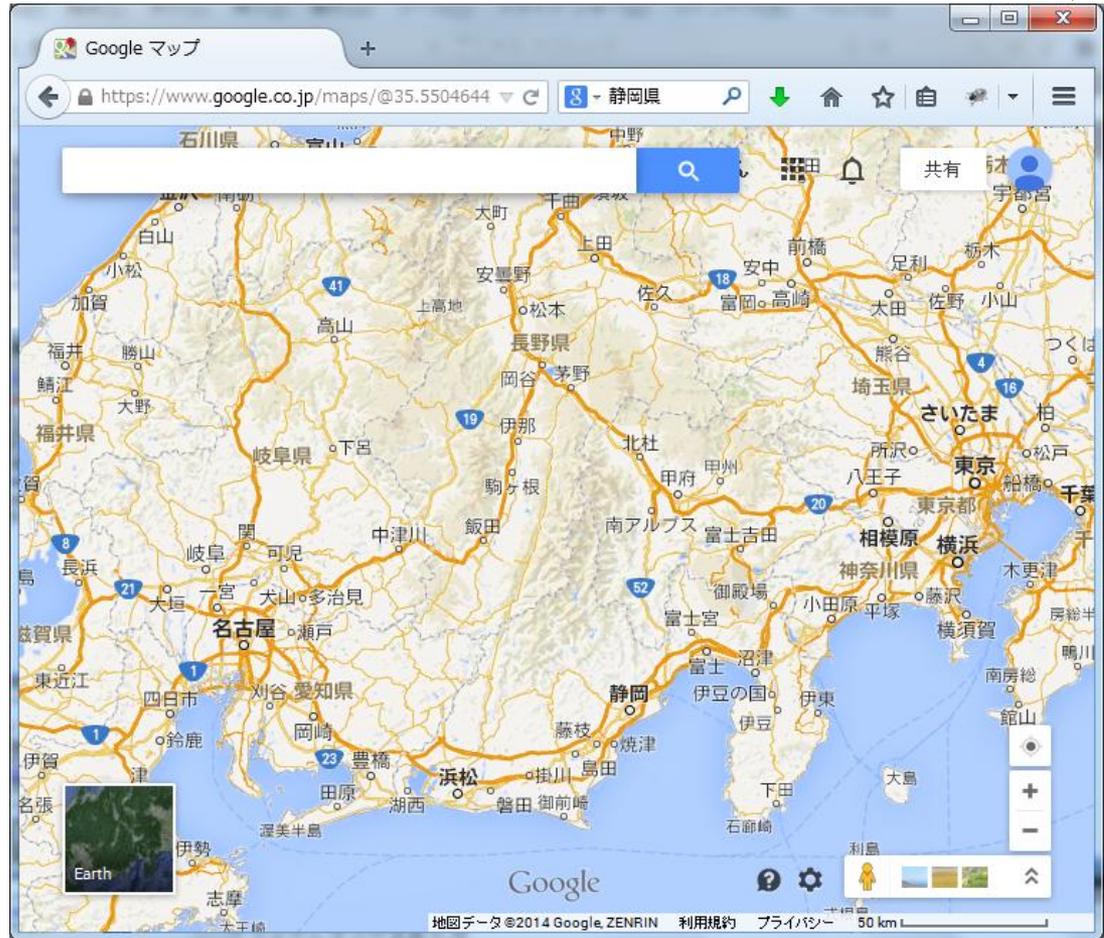
- 画面内に描画するため
- ・スクリーンの原点と、緯度・経度の原点が違う
 - ・表示スケールの調整



136.09, 34.36

0, 600
136.09, 34.36

900, 600
140.18, 34.36



開発環境の準備

- 環境一式をダウンロード & 解凍
 - http://cloud.aitc.jp/20140927_D3js/
 - ALL.zip をダウンロードして、解凍
- 編集用テキストエディタ
 - お気に入りのエディタでOK
- 動作確認用ブラウザ
 - [F-12]を押せば、開発ツールが起動する
 - Chrome: 「ツール」→「デベロッパーツール」
 - Firefox + firebug: 「ツール」→「Web開発」→「Firebug」→「Firebugを開く」
 - Safari: 「環境設定」→「詳細」→「メニューバーに”開発”メニューを表示」
 - IE(10以降): 「F12 開発者ツール」
- ブラウザによっては、ローカルがうまく読み込めない

```
d3.json("http://cloud.aitc.jp/20140927_D3js/gis/nihon.json", function(json) {  
// d3.json("./gis/nihon.json", function(json) {
```

地図データの準備

- 地図データ
 - D3.js の geo パッケージで読める形式が必要
 - GeoJSON: 地理情報用に規格されたJSONデータ形式
 - TopoJSON: GeoJSONの拡張形式。D3.jsではプラグインが必要
 - 冗長性を排するので、データサイズが20%程度になる
 - Shape形式の地図を入手し、GeoJSON形式に変換
 - 入手元: 国土交通省、国土数値情報
 - <http://nlftp.mlit.go.jp/ksj/gml/datalist/KsjTmplt-N03.html>
 - 全国: N03-140401_GML.zip
 - 別のサイトで地図を圧縮 + GeoJSONに変換
 - <http://www.mapshaper.org/>
 - 2回圧縮することで、153MByte → 0.5MByte
 - 元のままで使うと、詳細すぎて重い
 - 変換結果 (GeoJSON形式、539KB)
 - http://cloud.aitc.jp/20140927_D3js/gis/nihon.json

- セレクタ (W3C Selectorsを参照)
 - `d3.select("#hoge")` → `<xxx id="hoge">` を対象
 - `d3.select(".hoge")` → `<xxx class="hoge">` を対象
 - `d3.select("hoge")` → `<hoge>` を対象

```
signal = [
  { "cx": 100, "cy": 100, "color": "#0000ff", "title": "青" },
  { "cx": 200, "cy": 100, "color": "#ffff00", "title": "黄" },
  { "cx": 300, "cy": 100, "color": "#ff0000", "title": "赤" },
];
```

- セレクション
 - `selectAll()`, `enter()`, `exit()`
 - 繰り返し処理が楽に書ける
 - `d3.select("#TEXT1").selectAll("p").style("color", "#000000");`
- 動的プロパティ
 - `svg.selectAll(".node").data(signal).text(function(d) { return d.title; });`
 - `svg.selectAll(".node").data(signal).text(function(d, i) { return i; });`
- データの結合
 - 追加: `svg.selectAll(".node").data(signal).enter().append("text").text("piyopiyo");`
 - 更新: `svg.selectAll(".node").data(signal).text("hogehoge");`
 - 削除: `svg.selectAll(".node").data(signal).exit().remove();`
- アニメーション
 - `d3.select("#TEXT4").transition().style("background-color", "black");`

- ファイルの読み込み
 - `d3.json()`, `d3.text()`, `d3.csv()`
- 地図描画
 - `d3.geo.mercator()`
 - `d3.geo.path().projection(projection);`

- Wikipedia

- http://ja.wikipedia.org/wiki/Scalable_Vector_Graphics

Scalable Vector Graphics (スケーラブル・ベクター・グラフィックス、**SVG**) は、[XML](#)をベースとした、2次元**ベクターイメージ**用の**画像形式**の1つである。アニメーションやユーザインタラクションもサポートしている。SVGの仕様は[W3C](#)によって開発され、**オープン標準**として勧告されている。

- SVG仕様

- http://www.hcn.zaq.ne.jp/___/SVG11-2nd/

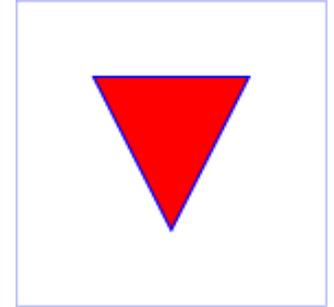
- 使用上の注意

- 順番通りに上に重ねて描画
- 対応していないブラウザだと、何も表示されない
- ブラウザによっては、微妙に見え方が違う

- 代表的な図形

- パス

```
<path d="M 100 100 L 300 100 L 200 300 z" fill="red"
      stroke="blue" stroke-width="3" />
```



- 円

```
<circle cx="600" cy="200" r="100" fill="red" stroke="blue"
        stroke-width="10" />
```

- 線

```
<line x1="100" y1="300" x2="300" y2="100" stroke="green"
      stroke-width="5" />
```

- テキスト

```
<text x="250" y="150" font-family="Verdana" font-size="55"
      fill="blue" >
```

Hello, out there

```
</text>
```

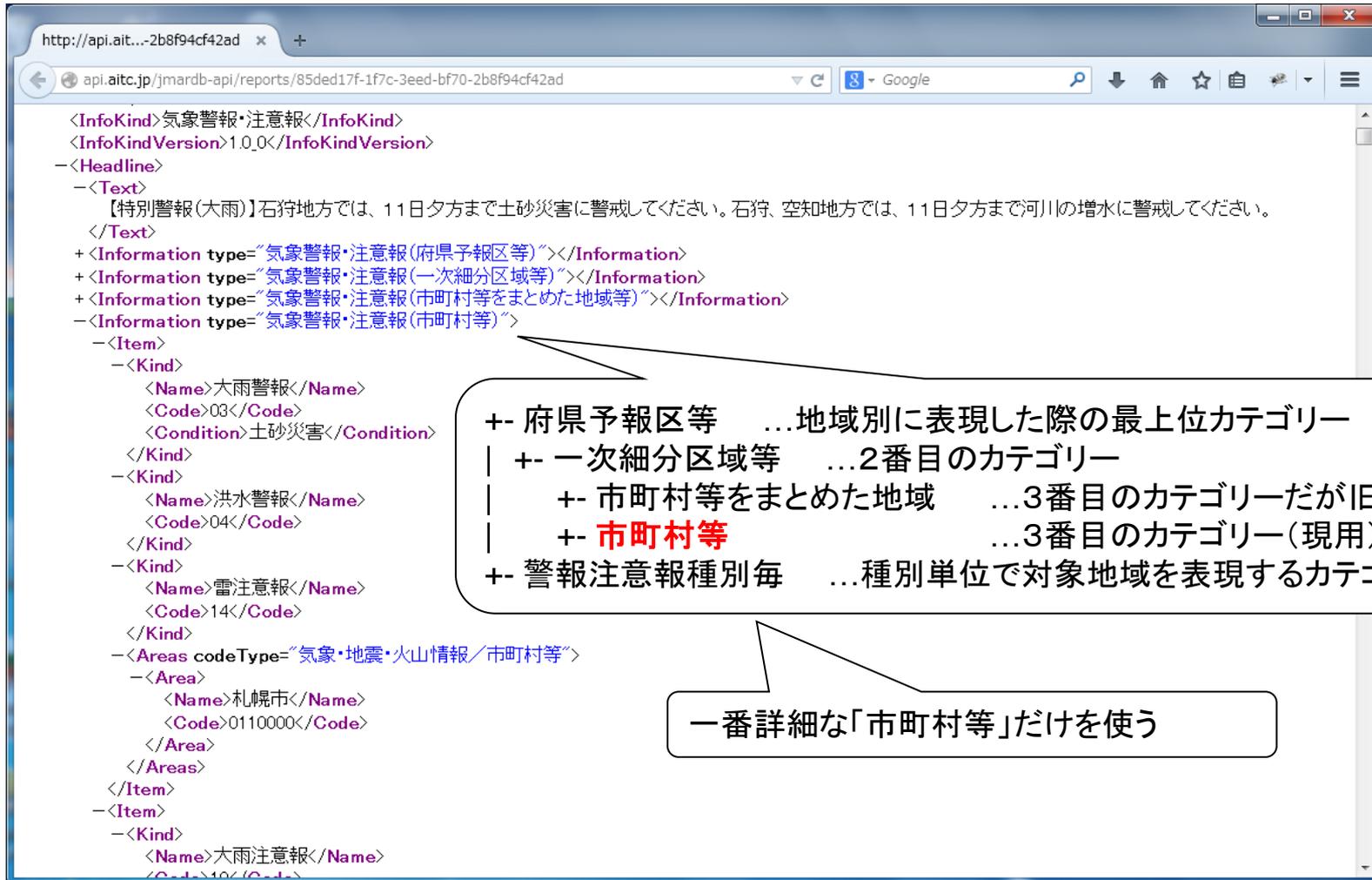
以降は、サンプルコードを見ながら解説

- ブラウザでソースコードが参照できます
 - http://cloud.aitc.jp/20140927_D3js/
- D3.jsでデータとSVGが連動するサンプル
 - step1_1.html + js, step1_2.html + js
- D3.jsで地図を描画する
 - step2.html + js
- D3.jsでデータを取得する
 - step3_1.html + js
 - step3_2.html + js
- 取得したデータをsvgで可視化する
 - step4.html + js
 - 十分な量のデータがあれば、地図無しでもいいけるかも
- 地図に重ねてデータを描画する
 - step5.html + js 「気象警報・注意報」を描画する

「気象警報・注意報」使用上の注意

- 気象庁XML

- <http://api.aitc.jp/jmardb-api/reports/85ded17f-1f7c-3eed-bf70-2b8f94cf42ad>



```

<InfoKind>気象警報・注意報</InfoKind>
<InfoKindVersion>1.0.0</InfoKindVersion>
-<Headline>
-<Text>
【特別警報(大雨)】石狩地方では、11日夕方まで土砂災害に警戒してください。石狩、空知地方では、11日夕方まで河川の増水に警戒してください。
</Text>
+<Information type="気象警報・注意報(府県予報区等)"></Information>
+<Information type="気象警報・注意報(一次細分区域等)"></Information>
+<Information type="気象警報・注意報(市町村等をまとめた地域等)"></Information>
-<Information type="気象警報・注意報(市町村等)">
-<Item>
-<Kind>
<Name>大雨警報</Name>
<Code>03</Code>
<Condition>土砂災害</Condition>
</Kind>
-<Kind>
<Name>洪水警報</Name>
<Code>04</Code>
</Kind>
-<Kind>
<Name>雷注意報</Name>
<Code>14</Code>
</Kind>
-<Areas codeType="気象・地震・火山情報/市町村等">
-<Area>
<Name>札幌市</Name>
<Code>0110000</Code>
</Area>
</Areas>
</Item>
-<Item>
-<Kind>
<Name>大雨注意報</Name>
<Code>10</Code>

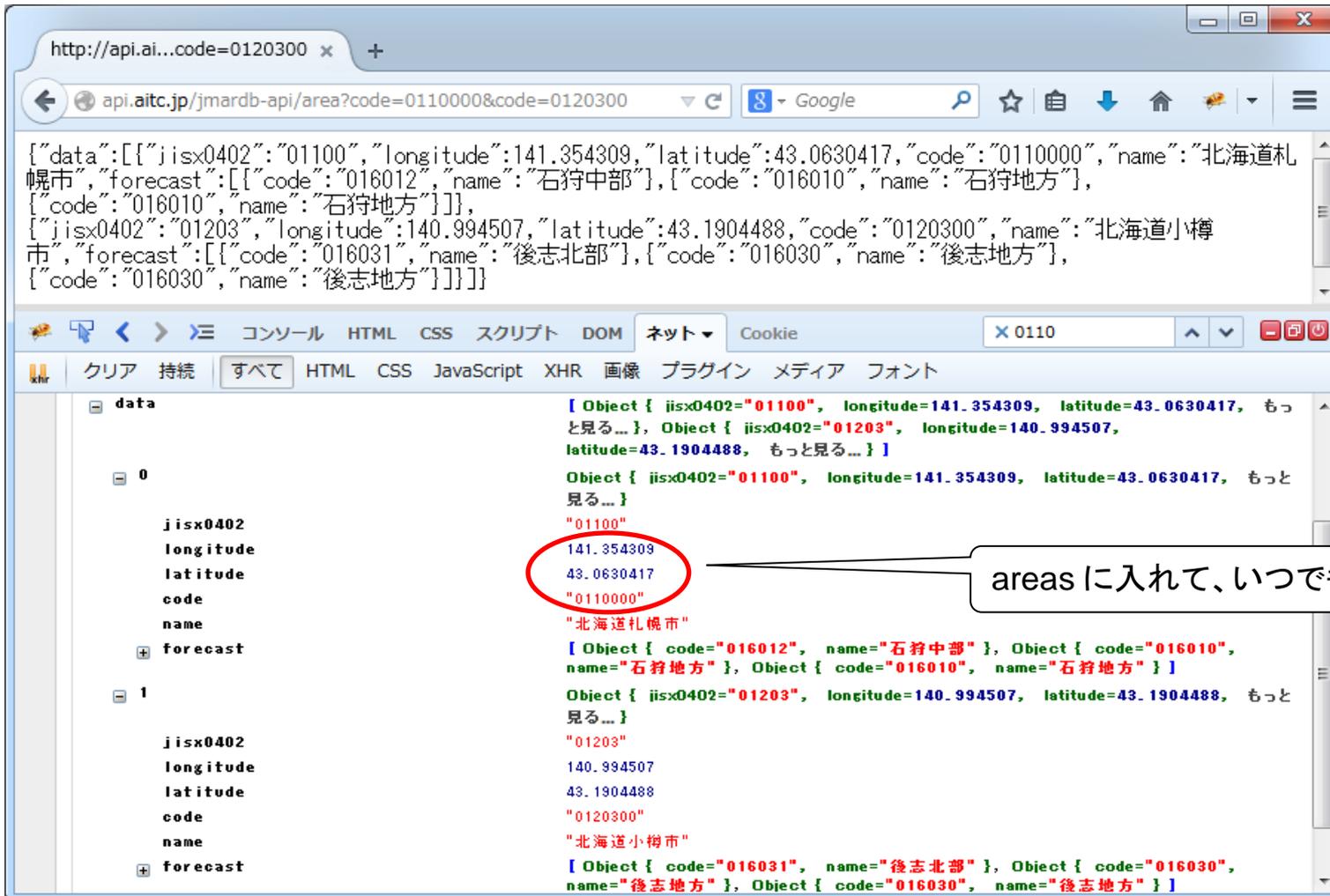
```

- + 府県予報区等 ... 地域別に表現した際の最上位カテゴリー
- | +- 一次細分区域等 ... 2番目のカテゴリー
- | +- 市町村等をまとめた地域 ... 3番目のカテゴリーだが旧方式
- | +- **市町村等** ... 3番目のカテゴリー(現用)
- + 警報注意報種別毎 ... 種別単位で対象地域を表現するカテゴリー

一番詳細な「市町村等」だけを使う

- 気象庁API

- <http://api.aipc.jp/jmardb-api/area?code=0110000&code=0120300>



```

{"data": [{"jisx0402": "01100", "longitude": 141.354309, "latitude": 43.0630417, "code": "0110000", "name": "北海道札幌市", "forecast": [{"code": "016012", "name": "石狩中部"}, {"code": "016010", "name": "石狩地方"}, {"code": "016010", "name": "石狩地方"}]}, {"jisx0402": "01203", "longitude": 140.994507, "latitude": 43.1904488, "code": "0120300", "name": "北海道小樽市", "forecast": [{"code": "016031", "name": "後志北部"}, {"code": "016030", "name": "後志地方"}, {"code": "016030", "name": "後志地方"}]}]}

```

areasに入れて、いつでも参照可能にする

残った時間で機能拡張してください

- 初級:「気象特別警報・警報・注意報」に対応
 - URLのtitle=を変更
 - title=「気象警報・注意報」→「気象特別警報・警報・注意報」
 - 取得できるJSONの構造はほとんど同じ
 - Kindname=「特別警報」になっているものがある
 - 「特別警報」をもっと目立つように描画する
 - 同じ警報・注意報の気象庁XML
 - <http://api.aitc.jp/jmardb-api/reports/85ded17f-1f7c-3eed-bf70-2b8f94cf42ad>
 - <http://api.aitc.jp/jmardb-api/reports/328e143d-b881-34e8-8b37-596d1c9f9ebb>
- 応用:「震源・震度に関する情報」に対応
 - URLのtitle=とpath=を変更
 - title=「気象警報・注意報」→「震源・震度に関する情報」
 - path=/report/head/headline/information
 - ↓ ↓ ↓ ↓ ↓
 - path=/report/body/intensity/observation/pref
 - 取得できるJSONの構造は大きく違う
 - 円の半径=震度
- その他のオープンデータを使ってみる
- 気象庁APIを別の方法で使ってみる

気象警報・注意報

Firebug -

DOM

特別警報

Window > Object

datetime	"2014-09-11T13:10:18.000+0900"
fragment	[Object { item=[52], type="気象警報・注意報（市町村等）" }]
0	Object { item=[52], type="気象警報・注意報（市町村等）" }
item	[Object { any=[0], areas=[...], kind=[3], もっと見る... }, Object { any=[0], areas=[...], kind=[2], もっと見る... }, Object { any=[0], areas=[...], kind=[3], もっと見る... }, 49 もっと見る...]
0	Object { any=[0], areas=[...], kind=[3], もっと見る... }
any	[]
areas	Object { area=[1], codeType="気象・地震・火山情報／市町村等" }
kind	[Object { code="03", condition="土砂災害", name="大雨警報" }, Object { code="04", name="洪水警報" }, Object { code="14", name="雷注意報" }]
0	Object { code="03", condition="土砂災害", name="大雨警報" }
1	Object { code="04", name="洪水警報" }
2	Object { code="14", name="雷注意報" }
lastKind	[]
1	Object { any=[0], areas=[...], kind=[2], もっと見る... }
any	[]
areas	Object { area=[1], codeType="気象・地震・火山情報／市町村等" }
kind	[Object { code="10", name="大雨注意報" }, Object { code="14", name="雷注意報" }]
0	Object { code="10", name="大雨注意報" }
1	Object { code="14", name="雷注意報" }
lastKind	[]
2	Object { any=[0], areas=[...], kind=[3], もっと見る... }
3	Object { any=[0], areas=[...], kind=[2], もっと見る... }
4	Object { any=[0], areas=[...], kind=[2], もっと見る... }
5	Object { any=[0], areas=[...], kind=[2], もっと見る... }
6	Object { any=[0], areas=[...], kind=[3], もっと見る... }

気象特別警報・警報・注意報

Firebug -

DOM

特別警報

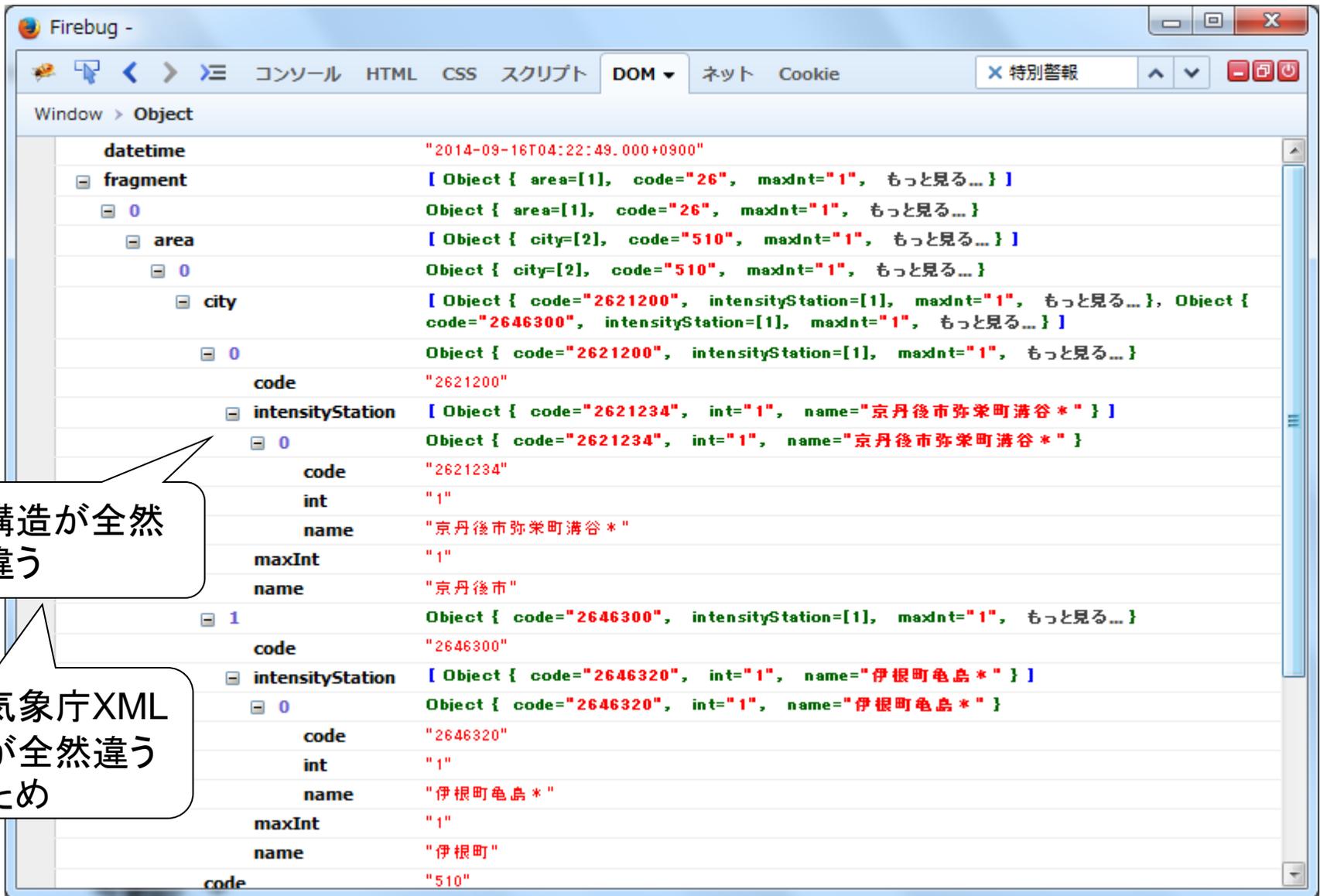
Window > Object

```

datetime      "2014-09-11T13:10:15.000+0900"
fragment      [ Object { item=[52], type="気象警報・注意報(市町村等)" } ]
  0           Object { item=[52], type="気象警報・注意報(市町村等)" }
    item      [ Object { any=[0], areas=[...], kind=[3], もっと見る... }, Object { any=[0], areas=[...],
kind=[2], もっと見る... }, Object { any=[0], areas=[...], kind=[3], もっと見る... }, 49もっと見る... ]
      0       Object { any=[0], areas=[...], kind=[3], もっと見る... }
        any   [ ]
          areas Object { area=[1], codeType="気象・地震・火山情報/市町村等" }
            kind [ Object { code="33", condition="土砂災害", name="大雨特別警報" }, Object { code="04",
name="洪水警報" }, Object { code="14", name="雷注意報" } ]
              0 Object { code="33", condition="土砂災害", name="大雨特別警報" }
              1 Object { code="04", name="洪水警報" }
              2 Object { code="14", name="雷注意報" }
            lastKind [ ]
          1     Object { any=[0], areas=[...], kind=[2], もっと見る... }
            any [ ]
              areas Object { area=[1], codeType="気象・地震・火山情報/市町村等" }
                kind [ Object { code="10", name="大雨注意報" }, Object { code="14", name="雷注意報" } ]
                  0 Object { code="10", name="大雨注意報" }
                  1 Object { code="14", name="雷注意報" }
                lastKind [ ]
              2 Object { any=[0], areas=[...], kind=[3], もっと見る... }
              3 Object { any=[0], areas=[...], kind=[2], もっと見る... }
              4 Object { any=[0], areas=[...], kind=[2], もっと見る... }
              5 Object { any=[0], areas=[...], kind=[2], もっと見る... }
              6 Object { any=[0], areas=[...], kind=[3], もっと見る... }
  
```

ここだけ違う

震源・震度に関する情報



Window > Object

```

datetime "2014-09-16T04:22:49.000+0900"
fragment [ Object { area=[1], code="26", maxInt="1", もっと見る... } ]
  0 Object { area=[1], code="26", maxInt="1", もっと見る... }
    area [ Object { city=[2], code="510", maxInt="1", もっと見る... } ]
      0 Object { city=[2], code="510", maxInt="1", もっと見る... }
        city [ Object { code="2621200", intensityStation=[1], maxInt="1", もっと見る... }, Object { code="2646300", intensityStation=[1], maxInt="1", もっと見る... } ]
          0 Object { code="2621200", intensityStation=[1], maxInt="1", もっと見る... }
            code "2621200"
            intensityStation [ Object { code="2621234", int="1", name="京丹後市弥栄町溝谷*" } ]
              0 Object { code="2621234", int="1", name="京丹後市弥栄町溝谷*" }
                code "2621234"
                int "1"
                name "京丹後市弥栄町溝谷*"
                maxInt "1"
                name "京丹後市"
              1 Object { code="2646300", intensityStation=[1], maxInt="1", もっと見る... }
                code "2646300"
                intensityStation [ Object { code="2646320", int="1", name="伊根町亀島*" } ]
                  0 Object { code="2646320", int="1", name="伊根町亀島*" }
                    code "2646320"
                    int "1"
                    name "伊根町亀島*"
                    maxInt "1"
                    name "伊根町"
                  code "510"

```

構造が全然違う

気象庁XMLが全然違うため

その他の拡張案

- 地名をAPIの分かり易いものにする
- 日付の指定をもっと親切にする
- 時刻も指定可能にする
- D3.jsのアニメーション機能を使用
- 特定の県だけを拡大表示する
- 他のデータとマッシュアップする

発表タイム

- 機能拡張したもの
- その他のオープンデータを使ったもの
- 気象庁APIを他の方法で使ってみる

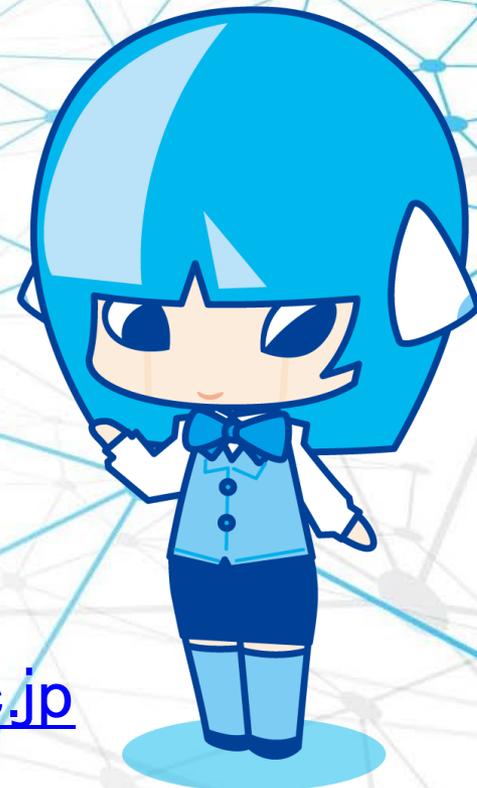
次回部会は、
10/21(火)13:30～です。



<http://aitc.jp>



<https://www.facebook.com/aitc.jp>



ハルミン

AITC非公式イメージキャラクター