ソフトウェア開発
第13回目授業

平野 照比古

2017/1/6

W3C における XML(eXtensible Markup Language) の説明 (1)

> *The Extensible Markup Language (XML) is a simple text-based format for representing structured information: documents, data, configuration, books, transactions, invoices, and much more. It was derived from an older standard format called SGML (ISO 8879), in order to be more suitable for Web use.*

XML は構造化されたデータを表現する単純なテキストベースのフォーマットである。構造化された文書とは文書、データ、構成、本、送付状やそのほかもろもろのものである。

W3C における XML(eXtensible Markup Language) の説明 (2)

さらに続けて次のように記されている。

> *If you are already familiar with HTML, you can see that XML is very similar. However, the syntax rules of XML are strict: XML tools will not process files that contain errors, but instead will give you error messages so that you fix them. This means that almost all XML documents can be processed reliably by computer software.*

ここにも書いてあるように XML の形式は HTML の形式に似ているが、XML の文法上に規則は厳格である。

HTML との違い

- すべての要素は閉じられているか、空の要素としてマーク
- 空の要素は通常のように閉じられている (<happiness></happiness>) か、特別な短い形式 (<happiness />)
- XML においては属性値は常に"で囲む必要がある (<happiness type="joy">)
- HTML では組み込まれている要素名 (とその属性) の集まりがあるが、 XML ではそのようなものは存在しない (例外は xml で始まるものがある)。
- HTML ではいくつかの組み込まれた文字名 (エンティティ) があるが、XML には次の 5 つのものしかない。
  <(<), >(>), &(&), "(&), &apos(')
  XML では独自にエンティティを定義できる。

GPX ファイルとは (1)

- GPS のログデータを保存する形式の一つとして普及
- http://www.topografix.com/gpx.asp で規格を見ることが可能

ルート要素は gpx で、その下の子要素としては次のようなものがある。

- wpt(ウェイポイント)
  ある地点の情報
- rte(ルート)
  子要素として地点を表す rtept を持つ
- trk(トラック)
  順序付けられた地点のリスト。子要素として trkseg を持つ
- trkseg は順序付けられた地点 (trkpt) のリストを持つ

これらの要素はすべてある必要はない。

GPX ファイルとは (2)

wpt と trkpt のおもな構成要素は次の通り

名称	型	説明
lat	属性	地点の緯度
lon	属性	地点の経度
ele	要素	標高
time	要素	地点での時刻。世界標準時

GPX ファイルの例

```
 1 <?xml version="1.0" encoding="UTF-8"?>
 2 <gpx version="1.1"
 3 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 4 xmlns="http://www.topografix.com/GPX/1/1"
 5 xsi:schemaLocation="http://www.topografix.com/GPX/1/1
 6 http://www.topografix.com/GPX/1/1/gpx.xsd>
 7 <trk>
 8  <name>20160324</name>
 9  <number>1</number>
10 <trkseg>
11 <trkpt lat="35.485802" lon="139.340909">
12  <ele>70.794189</ele>
13  <time>2016-03-24T08:33:23Z</time>
14 </trkpt>
15 <trkpt lat="35.485799" lon="139.340913">
16  <ele>70.794189</ele>
17  <time>2016-03-24T08:33:28Z</time>
18 </trkpt>
19 <trkpt lat="35.485793" lon="139.340899">
20  <ele>70.794189</ele>
21  <time>2016-03-24T08:33:33Z</time>
22 </trkpt>
23 ... 略 ...
24 </trkseg>
25 </trk>
26 </gpx>
```

内容の解説

- 2 行目から 7 行目が gpx のルート要素
- 8 行目に trk
- 9 行目と 10 行目にはこのルートの名前、通し番号などが存在
- さらに trkseg が一つだけ存在
- trkseg 内には trkpt が存在
  初めの位置の情報は次の通り
  - 位置は北緯 $35.4858026°$(lat)、東経 $139.340909°$(lon)
  - 標高 70.794189m(ele)
  - 時刻 $2016-03-24T08:33:28Z$ (2016 年 3 月 24 日 8:33:28 世界標準時)

地球の形

地球の形は測量法や測量法施行令で定められている。測量法施行令第三条では地球の長半径と扁平率が定められている。

一　長半径　六百三十七万八千百三十七メートル
二　扁平率　二百九十八・二五七二二二一〇一分の一

地球上の 3 次元の位置

- 緯度と経度に対して扁平率を考慮して空間の位置を求める必要がある
- GPX ファイルのデータでは 2 点間の距離が小さいのと、地球の扁平率と高さも地球の半径 $R = 6378137\mathrm{m}$ に対して小さい

これらの理由で地球の形を球として考えて計算
経度 λ、緯度 ϕ の地点の空間位置は次の通り

$$\begin{aligned} x &= R\cos\phi\cos\lambda \\ y &= R\cos\phi\sin\lambda \\ z &= R\sin\phi \end{aligned}$$

Polyline と Polygon オブジェクト

- Polyline は指定した地点を折れ線でつなぐ
  new google.maps.Polyline(<option>) で作成
- Polygon は指定した地点を折れ線でつないだ多角形
  new google.maps.Polygn(<option>) で作成

オプションで指定する代表的なものは次の通り

プロパティ	説明
map	表示する地図
path	LatLng を要素とする (MVC) 配列
strokeColor	(縁取りの) 線の色
strokeOpacity	(縁取りの) 線の色の不透明度
strokeWeight	(縁取りの) 線の幅
fillColor	塗りつぶしの色 (Polygon のみ)
fillOpacity	塗りつぶしの色の不透明度 (Polygon のみ)

ブラウザでの処理の制限

- 通常、セキュリティのため、ブラウザのプログラムではローカルに配置されたフォルダを参照できない
- この例外がファイルのアップロード
  ユーザの責任において指定したファイルをサーバーにアップできる

FileReader オブジェクト

最近の JavaScript では FileReader オブジェクトを用いることで、サーバーに転送する代わりにそのファイルをブラウザ内で処理できる。

GPX ファイル内の道のりを Google Maps 上に表示

```
 1 <!DOCTYPE html>
 2 <html>
 3 <head>
 4 <title>Check Trace</title>
 5 <meta http-equiv="Content-Type" content="text/html; charset=UTF-8"/>
 6 <link rel="stylesheet" type="text/css" href="map.css"/>
 7 <script src="http://maps.google.com/maps/api/js?"
 8         type="text/ecmascript" charset="UTF-8"></script>
 9 <script src="map-new2.js" type="text/ecmascript"></script>
10 </head>
11 <body>
12 <div id="map_canvas" style="width:800px; height:800px"></div>
13 <div id="form">
14   <div>
15     <form id="params">
16       <div><input type="file" id="file" value="ログデータファイルの選択"/></div>
17     </form>
18   </div>
19   <div id="info">
20   </div>
21 </div>
22 </body>
23 </html>
```

GPX ファイル内の道のりを Google Maps 上に表示 (解説)

- 6 行目はこのページの外部スタイルシートを読み込むことを指定
- ７行目から８行目で Google Maps API のファイルを読み込む。ここでは API キーが表示されていないので、コンソール画面に警告が現れる。
- ９行目は GPX ファイルのデータを処理する JavaScript の読み込みを指定
- 12 行目は Google Maps の表示域
- 13 行目から 17 行目にはファイルのアップロードの要素が定義
- 19 行目は表示された GPX ファイルに関する情報を示す

HTML で読み込まれるスタイルシート

```
 1 #map_canvas{
 2     display: inline-block;
 3 }
 4 #form{
 5     display: inline-block;
 6     vertical-align:top;
 7 }
 8 table tr td {
 9     text-align: right;
10 }
11 table > tr:nth-child(1) td:nth-child(2) {
12     text-align:center;
13 }
```

- 地図とフォームを横並び (2 行目と 5 行目の display:inline-block)
- 表の数字を右寄せ (9 行目)
- 文字のところを中央ぞろえ (12 行目)。

外部の JavaScript ファイル

```
1  var PARAMS = {w:800, h:800};
2  window.onload = function (){
3    var param = location.href.split("?");
4    if(param.length>1) {
5      param = param[1].split("&");
6      param.forEach(function(v) {
7        vv = v.split("=");
8        if(vv.length>1) PARAMS[vv[0]] = vv[1];
9      });
10   }
```

2 行目から 53 行目で読み込み終了後に実行される関数を定義している。

- 3 行目から 10 行目で URL の後ろにパラメータ (?) があるかをチェック
- パラメータは&で区切られている
  - &で分解 (5 行目)、
  - れぞれに対して (6 行目の forEach)=で名前と値に分割
  - パラメタ (変数 PARAM) に代入

```
11   var Clrs = [[255,0,0],[63,63,63],[0,255,0],
12         [0,0,255],[127,127,0],[127,0,127],[0,127,127]];
13   var Colors = Clrs.map(function(C) { return "rgb("+C.join(",")+")";});
14   var C2 = Clrs.map(function(C) {
15     return "rgb("+ C.map(function(C1){
16         return Math.floor((C1+255*2)/3,1)}).join(",")+")";});
```

- GPX のログ内に複数の<trkseg>があった場合に、それらを色分けするための色を指定
- 道のりの色は後で見るように下が透けるように指定して (88 行目) いるので、地図上と情報用の背景色が同じように見えるように色を調整

```
17   var GMap = google.maps;
18   var canvas = document.getElementById("map_canvas");
19   canvas.setAttribute("style","width:"+PARAMS.w+"px; height:"+PARAMS.h+"px;");
20   var map = new GMap.Map(canvas,
21     {center:new GMap.LatLng(35.486210,139.341443),
22      mapTypeId: GMap.MapTypeId.ROADMAP,
23      scaleControl:true,
24      scaleControlOptions:GMap.ControlPosition.BOTTOM_LEFT,
25      zoom:10
26     });
```

ここでは初期の地図の表示を行っている。

```
27  var Points=[];
28  var table = document.getElementById("info");
29  document.getElementById("file").onchange =function(E) {
30    var R = 6378137;
31    var reader = new FileReader();
32    reader.onload = function(){
33      var log = new window.DOMParser().parseFromString(reader.result,"text/xml");
34      var trks = log.getElementsByTagName("trk");
35      Array.prototype.forEach.call(trks,function(trk,i){
36        var trksegs = trk.getElementsByTagName("trkpt");
37        Points[i] = Array.prototype.map.call(trksegs,function(V) {
38          var lat = V.getAttribute("lat");
39          var lon = V.getAttribute("lon");
40          var latRad = lat * Math.PI/180;
41          var lonRad = lon * Math.PI/180;
42          var latCos = Math.cos(latRad);
43          return [lat, lon,
44            new Date(V.getElementsByTagName("time")[0].firstChild.nodeValue).getTime()].concat(
45              [R*latCos*Math.cos(lonRad),R*latCos*Math.sin(lonRad),R*Math.sin(latRad)]);
46        });
47        console.log(i);
48      });
49      ShowPath(8, 0.5, 0);
50    }
51    reader.onerror= function(){alert("Error")};
52    reader.readAsText(E.target.files[0]);
53  };
```

アップロードするファイルが変化したときのイベント処理 (1)

- 31 行目で FileReader オブジェクトを作成
- 32 行目から 50 行目で、このオブジェクトが利用可能になった時のイベント処理関数を登録 (実際の起動は 52 行目で発生)
- 33 行目で読み込まれたテキストを"text/xml"で処理
  処理をするオブジェクトが window.DOMParser() の parseFromString() メソッド
- この処理結果が DOM の構造になるので、今までの DOM の処理が適用できる
- 34 行目で trk 要素に分解
- それぞれの trk 要素に対して処理を行うために、Array オブジェクトのメソッド forEach を用いる。
- 34 行目で得られたものは HTML コレクションなので、このメソッドが直接適用できない。call メソッドを利用することで解決

アップロードするファイルが変化したときのイベント処理 (2)

- 36 行目で各 trk 要素内の trkpt 要素を求め、そこから各 trkpt 要素の緯度、経度、時間を求めて、空間の位置を計算し (38 行目から 42 行目)、結果を配列で返している (map メソッドを利用)。
- 空間の位置を計算しているのは道のりの長さを求めるためであるが、今回のプログラムではそれは利用していない。
- 49 行目でログを表示する関数を呼び出し
- 51 行目は読み込みが失敗したときの関数を登録
- 52 行目では指定されたファイルのデータをテキストとして読み込んでいる。

ログの情報を表示

```
54   function makeTR(data, tbl, color){
55     var tr = document.createElement("tr");
56     tbl.appendChild(tr);
57     if(color>=0) tr.style.background = C2[color%C2.length];
58     data.forEach(function(val) {
59       var td = document.createElement("td");
60       tr.appendChild(td);
61       td.appendChild(document.createTextNode(val));
62     });
63   }
```

- この関数の引数は、表に表示するデータ、表示するオブジェクト、(14 行目で定義した) 背景色の指定の 3 つ
- 55 行目で作成した tr 要素の中に、与えられたデータを表示 (58 行目から 62 行目)

道のりを表示

```
64  function ShowPath(w, o, s) {
65    var tbl = document.createElement("table");
66    table.appendChild(tbl);
67    var latMin=180, latMax=-180, lonMin=180, lonMax=-180;
68    makeTR(["番号","日付","開始時間","終了時間","地点数"], tbl, -1);
69    Points.forEach(function(Ps, i) {
70      var Routes = Ps.map(function(P){
71          latMax = Math.max(latMax, P[0]);
72          latMin = Math.min(latMin, P[0]);
73          lonMax = Math.max(lonMax, P[1]);
74          lonMin = Math.min(lonMin, P[1]);
75          return new GMap.LatLng(P[0],P[1]);
76        });
77      var d1 = new Date(Ps[0][2]);
78      var d2 = new Date(Ps[Ps.length-1][2]);
79      makeTR([i+1,
80          d1.toLocaleDateString(),
81          d1.toLocaleTimeString(),
82          d2.toLocaleTimeString(),
83          Routes.length], tbl, i);
84      new GMap.Polyline({
85        path: Ps.map(function(P){return new GMap.LatLng(P[0],P[1]);}),
86        strokeColor: Colors[(i+s)%Colors.length],
87        strokeWeight:w,
88        strokeOpacity:o,
89        map: map});
90      }
91    );
92    var latLngB = new GMap.LatLngBounds(new GMap.LatLng(latMax, lonMax));
93    latLngB.extend(new GMap.LatLng(latMin, lonMin));
94    map.fitBounds(latLngB);
95  }
```

- この関数の引数は道のりの幅、不透明度である。
- 69 行目から 76 行目で道のりを構成する地点を Google Maps の地点オブジェクトに変え、かつ、道のりの緯度、経度の最大値と最小値を求めている。
- 79 行目から 83 行目で道のりの情報を表示する。
- 84 行目から 90 行目で与えられたデータをもとに、道のりを表示している。道のりのデータはオブジェクトリテラルの形で与えられている。
- 92 行目から 94 行目では表示された道のりをすべて含む最大のズームレベルを設定 (`fitBounds` メソッド) している。

GPX アイルから道のりに必要な JSON オブジェクトを作成

```
1 <?php
2 function getPos($lat, $lon) {
3     $R = 6378137;
4     $latRad = $lat * M_PI/180;
5     $lonRad = $lon * M_PI/180;
6     $latCos = cos($latRad);
7     return array(
8         $R*$latCos*cos($lonRad),$R*$latCos*sin($lonRad),$R*sin($latRad));
9 }
```

与えられた緯度経度から空間の位置を求める関数である。戻り値は空間の位置を配列として返す。

与えられたファイルから GPX ファイルを処理

```
10 function SetDatafromFile($fn,$mode) {
11   $data = new DOMDocument();
12   $data->load($fn);
13   $trks = $data->getElementsByTagName("trk");
14   $len = $trks->length;
15   $trackdata = array();
16   $lengthdata = array();
17   $cnt=0;
```

- XML ファイルを読み込むために DOMDocument のインスタンスを作成 (11 行目)
- load メソッドにファイル名を指定することで、DOM オブジェクトに変換される (12 行目)。
- PHP ではピリオド . は文字列の連接演算子なのでメソッドは->を用いる。
- getElementsByTagName で trk を取得 (13 行目) する。
- 14 行目でその数を変数$len に格納
- トラックごとの情報をしまうための変数を初期化 (15 行目と 16 行目) いる。

```
18   for($i=0;$i<$len;$i++){
19     $trk = $trks->item($i);
20     $trksegs = $trk->getElementsByTagName("trkpt");
21     $len2 = $trksegs->length;
22     if($len2 <10) continue;
23     $trkseg = $trksegs->item(0);
24     $latmin = $latMax = $lat = $trkseg->getAttribute("lat");
25     $lonmin = $lonMax = $lon = $trkseg->getAttribute("lon");
26     $newtrk = array("[$lat,$lon]");
27     $ppos = getPos($lat, $lon);
28     $length = 0;
```

各 trk に対して距離などを求める。

- trk の一つを変数$trk に格納している。呼び出しに注意 (19 行目)
- $trk に含まれる trkpt のリストを得る (20 行目)。
- 極端に短いログは省く (22 行目)。
- ログ内の地点の範囲を示す変数を初期化 (24、25 行目)
- 地点の情報をしまう配列を初期化 (26 行目)
- 初めの位置の空間座標を変数$ppos に格納 (27 行目)
- 積算距離の変数を初期化 (28 行目)

```
29      for($j = 1; $j < $len2; $j++) {
30        $trkseg = $trksegs->item($j);
31        $lat = $trkseg->getAttribute("lat");
32        $lon = $trkseg->getAttribute("lon");
33        $latMax = max($lat-0,$latMax);
34        $latmin = min($lat-0,$latmin);
35        $lonMax = max($lon-0,$lonMax);
36        $lonmin = min($lon-0, $lonmin);
37        array_push($newtrk,"[$lat,$lon]");
38        $cpos = getPos($lat, $lon);
39        $xd = $ppos[0]-$cpos[0];
40        $yd = $ppos[1]-$cpos[1];
41        $zd = $ppos[2]-$cpos[2];
42        $length += sqrt($xd*$xd+$yd*$yd+$zd*$zd);
43        $ppos = $cpos;
44      }
```

- 各 trkpt から緯度と経度を取り出し (31、32 行目)、今までの範囲外ならば範囲を更新 (33 行目から 36 行目)
- 緯度と経度を配列に追加 (37 行目) し、空間の位置を計算 (38 行目)
- ひとつ前の点との差を求め (39 行目から 41 行目)、距離を計算 (42 行目)
- ひとつ前の点を更新 (43 行目)

```
45      if($lonMax - $lonmin >0.001 ||$latMax - $latmin > 0.001) {
46        $cnt++;
47        array_push($trackdata,
48          '{"range":['."$latMax,$latmin,$lonMax,$lonmin".'],"route":[' .
49        implode(",",$newtrk).'],"length":'.($length/1000).'}');
50      }
51    }
52    return $trackdata;
53  }
54  ?>
```

- 移動がほとんどないトラックは登録しない (45 行目)。
- 登録するトラックを JSON 形式で作成 (48 行目から 49 行目)
- 関数 implode は配列の要素を与えられた文字列を挟んで出力