HTML5 と JavaScript を利用した Android ネイティブアプリの作成

東京理科大学　理学部　物理学科　山本研究室

1210052 五木田　隼人

目次

０．　はじめに

０－１．はじめに

Android [1] や iPhone [2] に代表される“スマートフォン”はここ数年で急速に普及し、普及率は、50%近くにまで増えています。（2013 年 6 月・総務省データより）

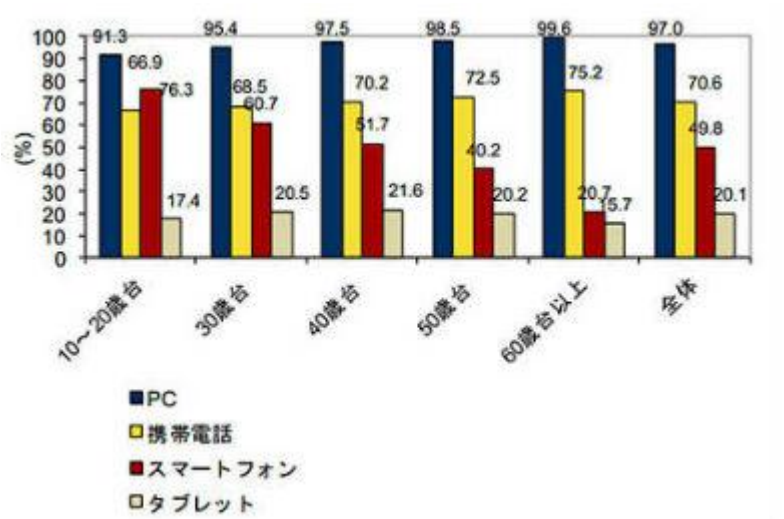

図　スマートフォン・タブレット・PC の普及率の年代別グラフ（総務省データ）
（http://www.itmedia.co.jp/news/articles/1310/04/news128.html より）

依然 PC が 97%という高い普及率となっていますが、タブレットの普及率も 20%近くに達し、若年層を中心に“スマーフォン”や“タブレット”は急速に普及しています。私自身も「何か簡単な調べ物をしたい」「暇つぶしの利用したい」と考えたとき、PC に代わって“スマートフォン”や“タブレット”などの端末を使用する頻度が増えていることを実感しています。

これらの端末は Android や iOS [3] などのオペレーティングシステム（以下 OS とします。）が搭載されており、マウスを使用しない直感的な動作で使用できるだけでなく、使用用途に合わせて“アプリ（アプリケーション）”を導入することによって、手軽にゲームを楽しんだり仕事を効率化したりすることができます。アプリケーションの導入は、従来の PC [4] でも勿論できますが、“スマーフォン”や“タブレット”においては各社が OS に合わせてアプリを公開している“ストア”というアプリケーションを利

[1] Android は Google.Inc の商標です。
[2] iPhone は Apple.Inc の商標です。
[3] iOS は Apple.Inc の商標です。
[4] ここでは、Windows 7 以前の PC を指しています。（Windows 8 からストアがインストールされています。）

用して簡単に入手することができるため、「アプリを入れる」ということが一般的になっています。「アプリを入れる」ということが一般的になれば、当然アプリケーションを提供する側も増えます。

かくして私も卒業作品にAndroidとiPhone両対応のアプリケーションを作りたいと考えました。しかし、スマートフォン向けのアプリを開発する場合、Androidアプリの場合はJava、iPhoneアプリの場合はObjective-Cという言語で開発するのが一般的で、プログラミング初心者である私が複数言語を取得し、アプリを開発するまでのレベルになるには、期間的に無理があると考えました。そこで、HTMLやJavaScriptなどのWeb技術を使って、複数OSに対応した**クロスプラットフォーム**[5]のアプリを作成することにしました。

アプリと一口にいっても様々なものがあります。メール、マップ、ゲーム、仕事効率化…etc、その中でも、私はスマートフォンの特徴でもある**GPSを使ったマップアプリ**を作成したいと考えました。

今回 Web アプリではなく、Web 技術でネイティブアプリを作るために“**Phone Gap(Cordova)**[6]”というフレームワークを利用して開発を行います。参考にできる書籍も少ない上に、私自身がプログラミングに不慣れであることで、薄い内容になっていますが、どうぞご了承下さい。

０－２．各種 OS について

現在スマートフォンに搭載されている OS は Android と iOS が有名ですが、開発中で市場に投入されていないものも含めると多くの OS が存在します。簡単に有名な OS を紹介します。

- **Android**

  Google によってスマートフォンやタブレットなどの携帯端末を主なターゲットとして開発されたプラットフォーム・OS であり、2013 年 9 月時、世界、日本においてスマートフォン OS のシェアは 1 位。オープンソースであり、Apache v2 ライセンス[7]で配布されています。2014 年 1 月現在、Android 4.4 KitKat が最新版。アプリは Java と Google の提供するライブラリィを用いて開発されます。

  Android Developer: http://developer.android.com/index.html

---

[5] 異なるプラットフォーム（OS 等）で同じ仕様のものを動かすことのできるプログラムのことです。

[6] ０－３．クロスプラットフォームツールの紹介を参照してください。

[7] Apache ソフトウェア財団によるソフトウェア向けライセンス規定。ソースコードはオープンソースだけでなくクローズドソースの開発にも使用できます。

- **iOS**

  アップル社が開発・提供する OS。アプリの開発に基本的には Objective-C を使用します。一部はオープンソースですが、基本的にはプロプライエタリソフトウェア [8] となっています。

- **Windows Phone**

  マイクロソフト社が開発・提供しているモバイル OS。2014 年 1 月現在の最新バージョンが Windows Phone 8。日本では 1 機種だけ発売されています。クローズドソースでアプリ開発は、C, C#, C++で行われます。

- **Firefox OS**

  ブラウザで有名な Firefox の Mozilla が提供する OS。一部は Android を母体としていますが、アプリの開発言語は C++, HTML, JavaScript, CSS となっています。日本ではまだ市場に投入されていませんが、ブラジル、スペインなどでは既に発売されています。オープンソースであり、Web 技術を用いたアプリ開発が可能な点が特徴的です。

  Firefox OS：http://www.mozilla.jp/firefoxos/

- **BrackBerry OS**

  ブラックベリー社が開発している BrackBerry シリーズのスマートフォン向けの OS。2014 年 1 月現在、最新バージョンは 7.1。クローズドソースで、アプリ開発には主に Java が使用されています。

- **Tizen（タイゼン）**

  LiMo Foundation, Linux Foundation, インテル社が主導する Tizen プロジェクトの OS。日本企業としては、NEC、NTT ドコモなどが参加しています。OS 自体はオープンソースで配布され、アプリの開発には HTML, JavaScript, C, C++などを使用します。2014 年 1 月現在、商用として発売されていません。

  Tizen: https://www.tizen.org/ja

---

[8] ソフトウェアの配布者が、利用者の持つ権利を制限的にすることで自身の利益を保持しようとするソフトウェアを指します。

- **Fire OS**

  Amazon.com が販売するタブレット Kindle Fire , Kindle Fire HD, Kindle Fire HDX などに搭載されている OS。Android がベースになっており、Kindle アプリの開発環境、言語は基本的に Android と同様です。

紹介しきれないほど数多くの OS が存在しており、１つ１つに対応するアプリを開発することは個人では困難になります。さらには HTML, JavaScript などで開発したいと考えた場合、OS 自体がこれらの言語でのアプリ開発に対応していなければ開発することはできず、対応する言語の取得は必須となります。それを解決するために登場したのが、**クロスプラットフォームツール**です。

０－３．クロスプラットフォームツールの紹介

それぞれのツールがどのようなものか簡単に説明します。

- **Phone Gap(Cordova)**

  まず名前についてですが、”Cordova”とはオープンソースの製品名で、”Phone Gap”とは Adobe 社が Cordova を配布するときの名前です。（ここからは Phone Gap で統一します。）Phone Gap を利用すれば、アプリ内にブラウザの機能を埋め込むことができるため HTML, JavaScript, CSS などをそのまま使用してネイティブアプリの開発することができます。Phone Gap で開発する問題点として、アプリの動作が Java（Android の場合）で開発した場合より遅くなる点があります。この辺りは Web アプリの動作がネイティブアプリに比べて遅くなることと同じ理由です。今回はこちらを利用して開発しました。

  配布先：http://phonegap.com/install/

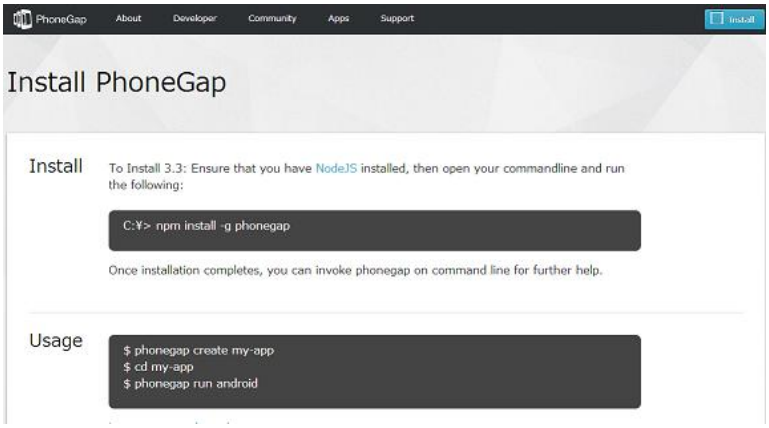

図　Phone Gap

- **Unity**

　３Ｄ コンテンツも作成できるゲーム開発プラットフォームで、Web や iOS, Android だけでなく、Windows、Mac、Wii、PlayStation、Xbox 360 に対応して書き出しすることができます。JavaScript でコーディングして、プラットフォームに対応してコンパイルしてくれるツールです。基本的にはゲームを作るためのツールです。

配布先：http://japan.unity3d.com/unity/

図　Unity

- **Titanium**

　Appcelerator 社が開発・提供しているスマートフォンアプリの開発環境です。JavaScript でコーディングして、OS に合わせて Titanium がコンパイルします。コンパイルするという点が PhoneGap との大きな違いです。

配布先：http://www.appcelerator.com/platform/appcelerator-platform/

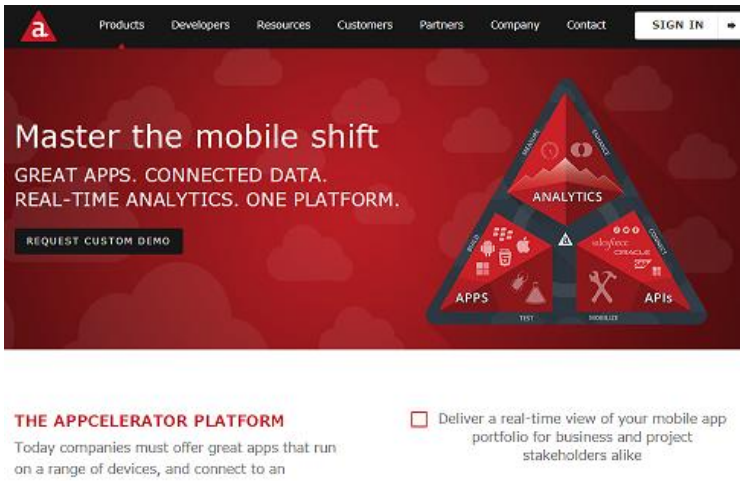

図　Titanium

1．　開発環境の準備

通常 Phone Gap を使用する場合、以下のツールやソフトウェアをインストールしなければなりません。

- **Android の場合**
  - Android SDK
  - Eclipse Classic
  - ADT Plugin
  - Phone Gap

  設定方法：
  http://docs.phonegap.com/jp/2.2.0/guide_getting-started_android_index.md.html#Getting%20Started%20with%20Android

- **iOS の場合**
  - XCode
  - Phone Gap

当然インストールするだけでなく、PATH などの様々な設定が必要になります。通常これだけの準備が必要ですが、”Monaca”という PC のブラウザ上で Phone Gap を使ったアプリを作成できる環境がありますので、そちらを利用します。

- **Monaca について**

  HTML5 と JavaScript を使って Android, iOS, Windows　8 のネイティブアプリを同時開発することができるクラウドサービスで、統合開発環境とデバッカーなど開発に必要なものが全て揃っています。アプリを利用して、実機上（Android などの端末）でデバックすることもできます。会員登録が必要でプロジェクトの数などで異なりますが、基本的に無料です。

  使用方法については割愛しますが、Eclipse などの一般的な統合開発環境とほぼ同様です。日本語に対応しており、モバイルアプリ開発に特化しているため、非常に扱いやすくクラウドであることを感じさせない程、利用し易いと感じました。

  Monaca：http://monaca.mobi/ja/

図　Monaca

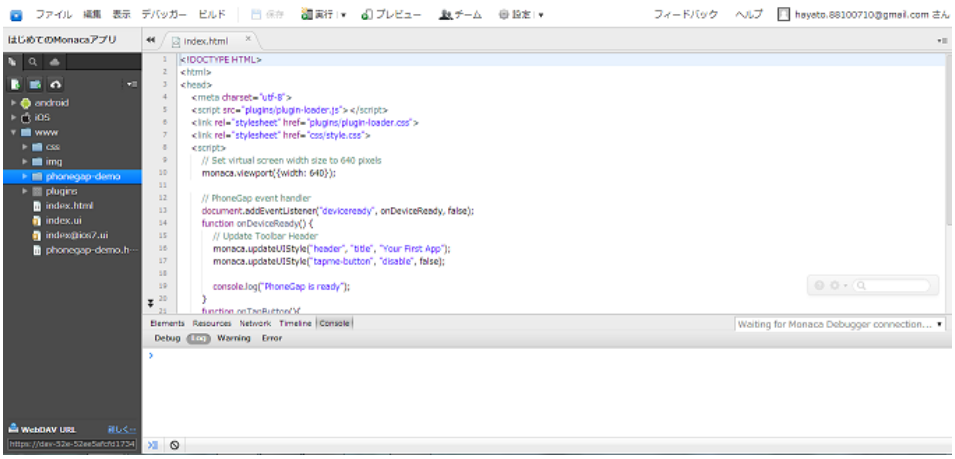

図　Monaca の使用画面

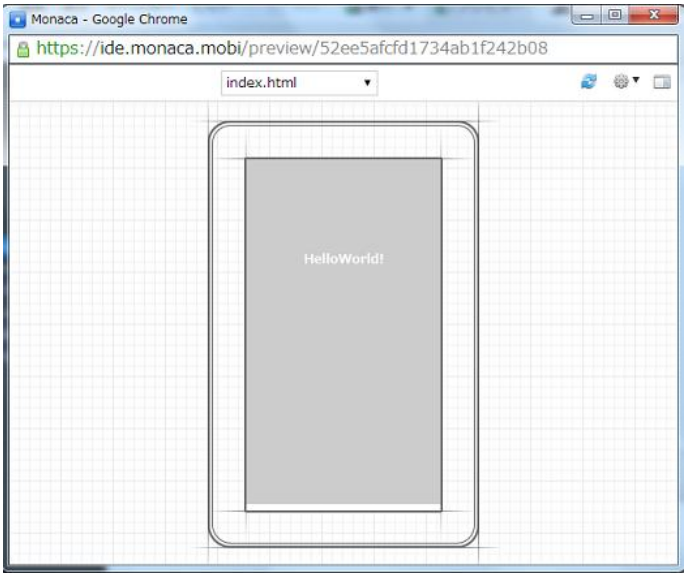

図　ライブプレビュー画面（搭載されているエミュレータ）

図　実機でデバックを行うためのアプリ・実際の画面

２．　アプリの概要

- **概要**

HTML5, JavaScript, CSS, jQuery[9], Phone Gap を用い、端末の GPS を利用して位置情報を追跡するアプリです。【追跡開始・停止】、【軌跡の削除】、【軌跡の保存】、【保存した軌跡の呼び出し】のボタンを備え、自分が移動した軌跡を表示するだけでなく軌跡の情報を LocalStorage に保存し、復元することもできます。また、Monaca に標準装備されている Phone Gap のテンプレートにもリンクしています。Phone Gap を利用してアクセスできる端末内の機能（GPS・バイブレーション・加速器など）を体感できるものになっています。

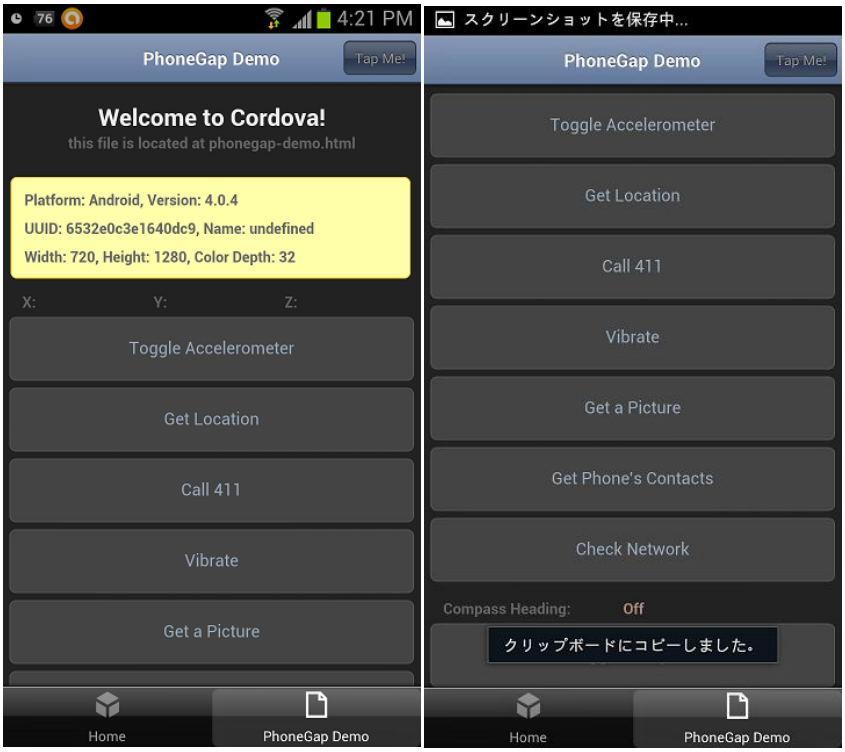

図　Phone Gap のテンプレート

[9] JavaScript をより容易に記述するために設計されたウェブブラウザ用の軽量な JavaScript ライブラリィです。

- スクリーンショット

図　アイコン画面

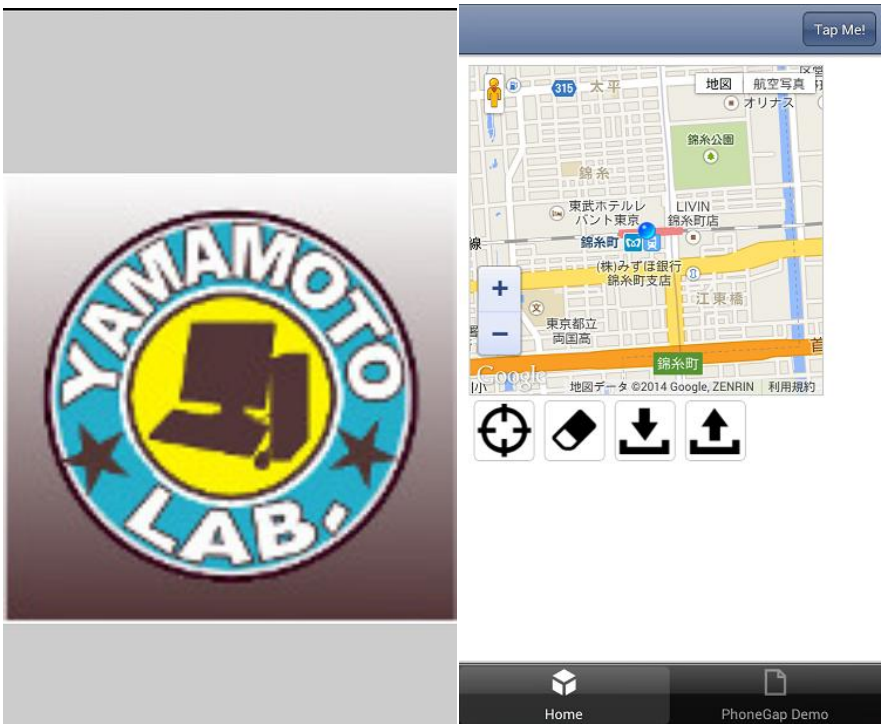

図　起動中画面・起動後画面

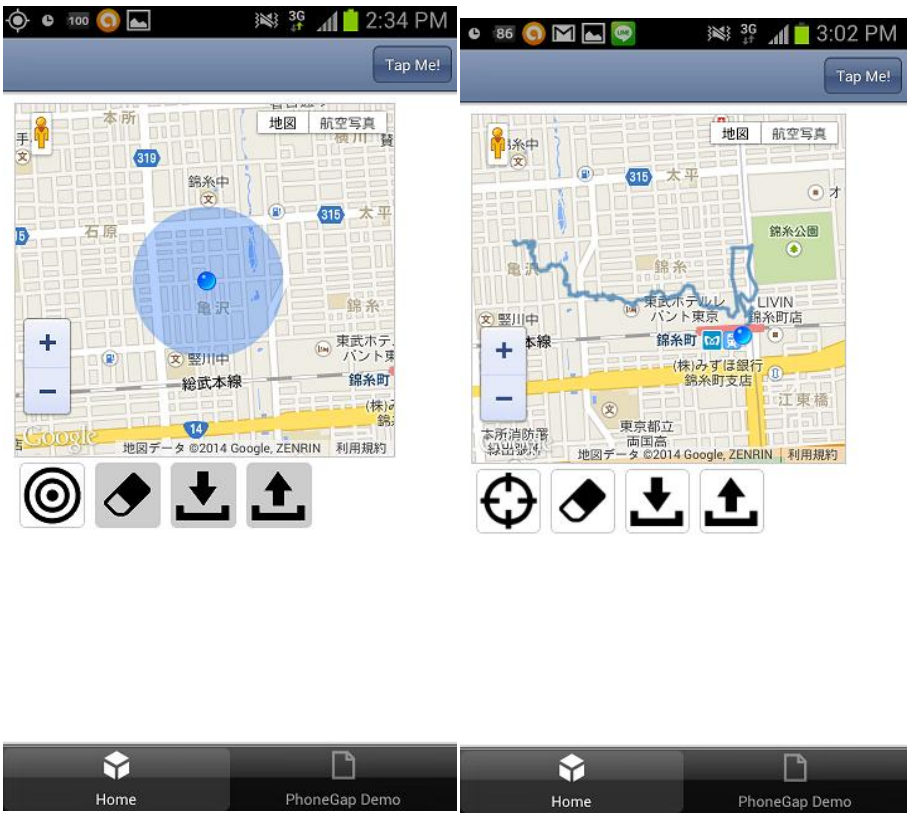

図　追跡開始・追跡停止画面

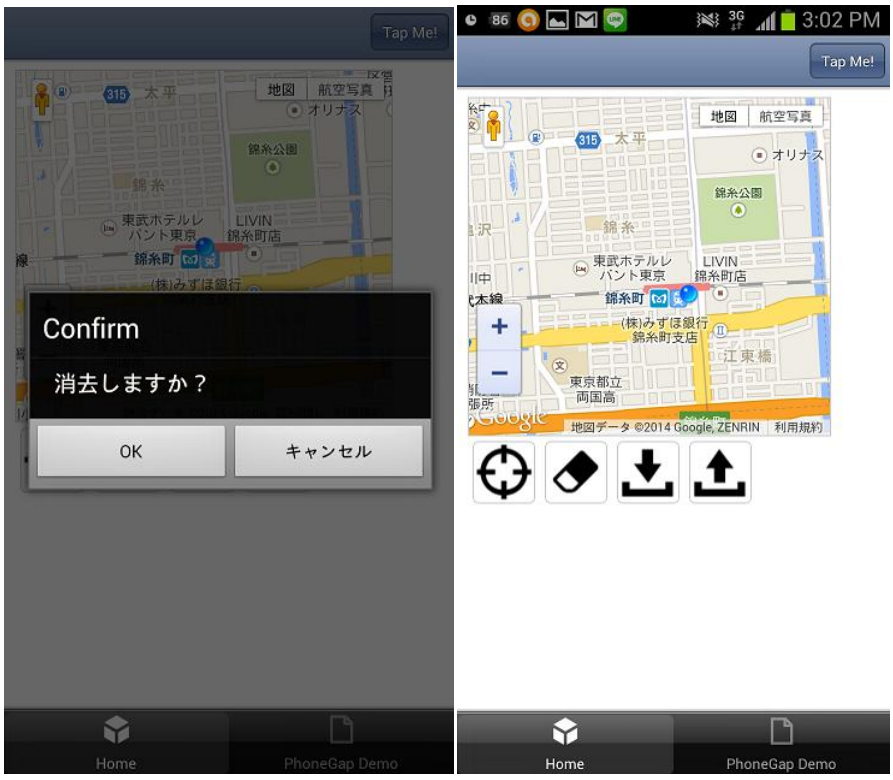

図　削除画面・削除後

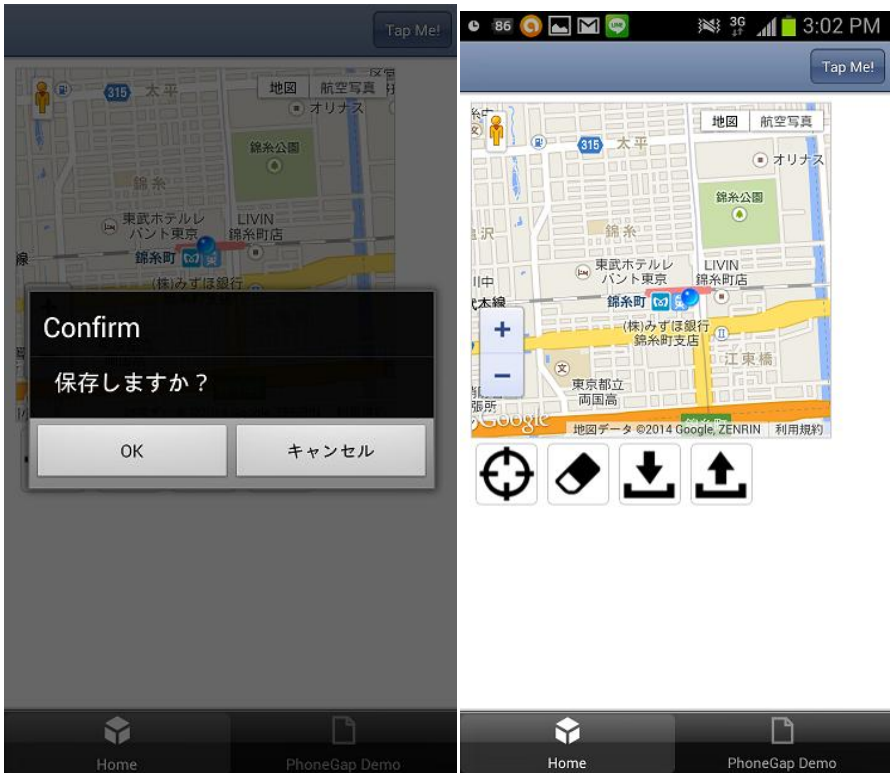

図　保存画面・保存後

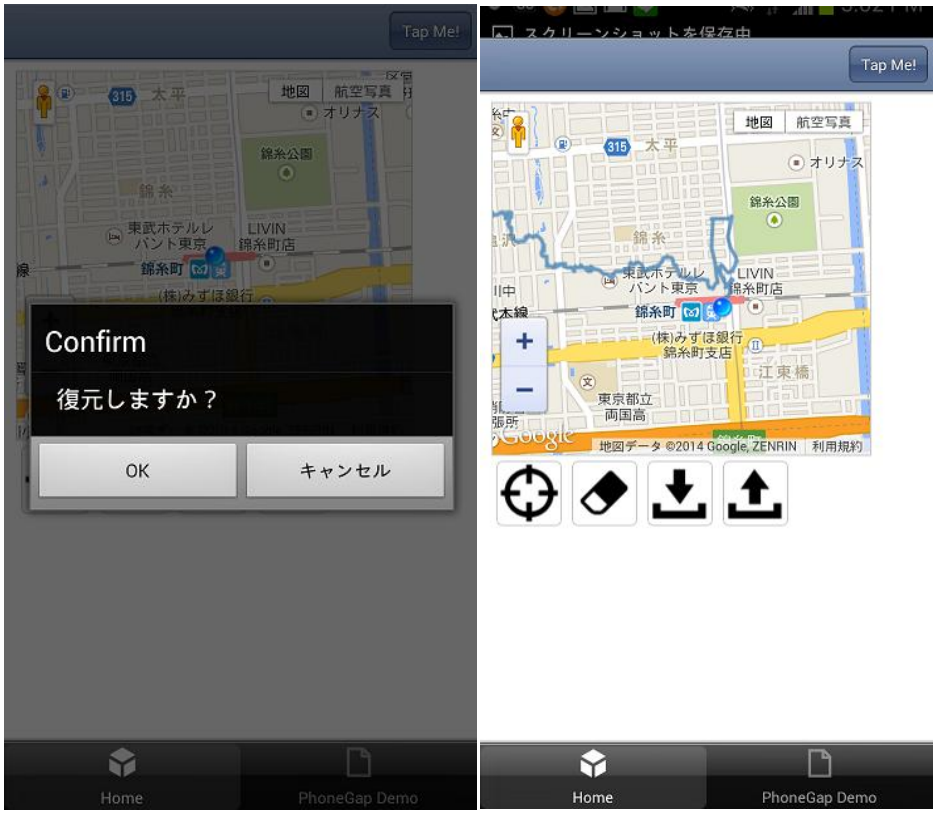

図　呼び出し・呼び出し後

３．　ソースコードとその解説

３－１．HTML・JavaScript

```
1  <!DOCTYPE HTML>
2  <html>
3  <head>
4      <meta charset="utf-8">
5      <meta name="viewport"
       content="width=device-width,initial-scale=1,maximum-scale=1,user-scalable=no" />
6      <script src="jquery-1.8.2.min.js" type="text/javascript" ></script>
7      <script src="http://maps.google.com/maps/api/js?libraries=geometry&sensor=true"
       type="text/javascript" ></script>
8      <link rel="stylesheet" type="text/css" href="Css.css">
9      <title>Trace Your Position</title>
10 </head>
11
12 <body>
13     <div id="map" ></div>
14     <ul>
15         <li id="current" class="tool"></li>
16         <li id="tracking" class="tool"></li>
17         <li id="delete" class="tool"></li>
18         <li id="save" class="tool"></li>
19         <li id="load" class="tool"></li>
20     </ul>
21
22     <script type="text/javascript">
23     $(function() {
24         var map, marker, poly, watchId;
25         var isTracking = false;
26
27         //地図の作成
28         createMap();
29         getCurrentPosition();
30
```

```
31          //GPS の設定
32          var geoOptions = {
33              enableHighAccuracy: true,
34              timeout: 60000,
35              maximumAge: 0
36          };
37
38          //継続的な位置の取得の開始
39          function startPositionTracking(){
40              watchId = navigator.geolocation.watchPosition(updateMap, null, geoOptions);
41              isTracking = true;
42          }
43
44          //継続的な位置の取得の停止
45          function stopPositionTracking(){
46              navigator.geolocation.clearWatch(watchId);
47              isTracking = false;
48          }
49
50          //位置の取得
51          function getCurrentPosition(){
52              navigator.geolocation.getCurrentPosition(updateMap, null, geoOptions);
53          }
54
55          //地図の更新
56          function updateMap(position){
57              var latlng = new google.maps.LatLng(position.coords.latitude,
    position.coords.longitude);
58              map.setCenter(latlng);
59              marker.setPosition(latlng);
60              poly.getPath().push(latlng);
61          }
62
63          //地図の作成
64          function createMap(){
65              var mapOptions = {
```

```
66                  zoom: 15,
67                  center: new google.maps.LatLng(35.696802, 139.814136),
68                  mapTypeId: google.maps.MapTypeId.ROADMAP
69              };
70              map = new google.maps.Map($('#map')[0], mapOptions);
71
72              var image = new google.maps.MarkerImage(
73                  'img/marker.png', null, null,
74                  new google.maps.Point( 8, 8 ),
75                  new google.maps.Size( 17, 17 )
76              );
77              marker = new google.maps.Marker({
78                  position: new google.maps.LatLng(35.696802, 139.814136),
79                  map: map,
80                  icon: image,
81
82                  optimized: false,
83                  title: 'marker'
84              });
85
86              var polyLineOptions = {
87                  strokeWeight: 3,
88                  strokeColor: '#4682B4',
89                  strokeOpacity: 0.8
90              };
91              poly = new google.maps.Polyline(polyLineOptions);
92              poly.setMap(map);
93          }
94
95          //軌跡の削除
96          function deletePath() {
97              var path = poly.getPath();
98              while (path.getLength()) {
99                  path.pop();
100             }
101         }
```

```
102
103         //軌跡の保存
104         function savePath() {
105             var path = poly.getPath();
106             for (var i=0; i<path.getLength(); i++) {
107                 localStorage[i] = path.getAt(i).toString();
108             }
109         }
110
111         //軌跡の復元
112         function loadPath() {
113             for(var i=0; i<localStorage.length; i++){
114                 var latlng = localStorage[i].replace(/¥(|¥)/g,'').split(',');
115                 poly.getPath().push(new google.maps.LatLng(latlng[0], latlng[1]));
116             }
117         }
118
119         //監視開始ボタンのイベント
120         $('#current').bind('click', function() {
121             isTracking = true;
122             startPositionTracking();
123             $('#current').hide();
124             $('#tracking').show();
125             $('#delete').addClass('disabled');
126             $('#save').addClass('disabled');
127             $('#load').addClass('disabled');
128             $('div.gmnoprint[title="marker"]').addClass('pulse');
129         });
130
131         //監視停止ボタンのイベント
132         $('#tracking').bind('click', function() {
133             isTracking = false;
134             stopPositionTracking();
135             $('#current').show();
136             $('#tracking').hide();
137             $('#delete').removeClass('disabled');
```

```
138             $('#save').removeClass('disabled');
139             $('#load').removeClass('disabled');
140             $('div.gmnoprint[title="marker"]').removeClass('pulse');
141         });
142 
143         //消去ボタンのイベント
144         $('#delete').bind('click', function() {
145             if(!isTracking){
146                 if(confirm('消去しますか？')){
147                     deletePath();
148                 }
149             }
150         });
151 
152         //保存ボタンのイベント
153         $('#save').bind('click', function() {
154             if(!isTracking){
155                 if(confirm('保存しますか？')){
156                     localStorage.clear();
157                     savePath();
158                 }
159             }
160         });
161 
162         //復元ボタンのイベント
163         $('#load').bind('click', function() {
164             if(!isTracking){
165                 if(confirm('復元しますか？')){
166                     deletePath();
167                     loadPath();
168                 }
169             }
170         });
171     });
172     </script>
173 
```

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

3－2．CSS

```
@charset "utf-8";

* {
    margin:          0px;
    padding:     0px;
}

body {
    padding:     9px;
}

#map {
    width:300px;
    height:280px;
    border: solid #CCC 1px;
}

ul{
    height:          60px;
}

li {
    float:           left;
    margin:          4px;
    width:           50px;
    height:          50px;
    border:          1px solid #ccc;
    border-radius:     5px;
    list-style-type: none;
}

```

```
#current {
    background-image:url("img/icon_current.png");
}

#tracking {
    background-image:url("img/icon_tracking.png");
    display: none;
}

#delete {
    background-image:url("img/icon_eraser.png");
}

#save {
    background-image:url("img/icon_save.png");
}

#load {
    background-image:url("img/icon_load.png");
}

.disabled {
    background-color:    #ccc;
}

@-webkit-keyframes pulsate {
    from {
        -webkit-transform: scale(2.0);
        opacity: 1.0;
    }
    to {
        -webkit-transform: scale(4.0);
        opacity: 0;
    }
}

```

```
68 div.gmnoprint[title="marker"] img {
69     display:none;
70 }
71
72 .pulse {
73     -webkit-animation:      pulsate 1.5s ease-in-out infinite;
74     border:                 1pt solid #4682B4;
75     border-radius:          100px;
76     background-color:       rgba(0,100,255, 0.1);
77 }
```

３－３．ソースコードの解説

```
1 <!DOCTYPE HTML>
```

HTML5 で記述することを宣言します。

- head 部分

```
3  <head>
4      <meta charset="utf-8">
5      <meta name="viewport"
 content="width=device-width,initial-scale=1,maximum-scale=1,user-scalable=no" />
6      <script src="jquery-1.8.2.min.js" type="text/javascript" ></script>
7      <script src="http://maps.google.com/maps/api/js?libraries=geometry&sensor=true"
type="text/javascript" ></script>
8      <link rel="stylesheet" type="text/css" href="Css.css">
9      <title>Trace Your Position</title>
10 </head>
```

Google Maps API JavaScript,　関連ファイルとのリンクを行います。

- **body 部分**

```
13      <div id="map" ></div>
14      <ul>
15          <li id="current" class="tool"></li>
16          <li id="tracking" class="tool"></li>
17          <li id="delete" class="tool"></li>
18          <li id="save" class="tool"></li>
19          <li id="load" class="tool"></li>
20      </ul>
```

ユーザーインターフェース部分を HTML で記述しています。13 行目の div タグ内がマップの部分に相当します。ul 要素は、上（アプリ内では左）から順に【追跡開始・停止】、【軌跡の削除】、【軌跡の保存】、【保存した軌跡の呼び出し】のボタンに相当します。

図　アプリの画面とボタン配置

```
12  #map {
13      width:300px;
14      height:280px;
15      border: solid #CCC 1px;
16  }
```

CSS で表示するマップの大きさを指定します。

```
18 ul{
19     height:          60px;
20 }
21
22 li {
23     float:          left;
24     margin:          4px;
25     width:          50px;
26     height:          50px;
27     border:          1px solid #ccc;
28     border-radius:     5px;
29     list-style-type: none;
30 }
```

CSS で ul 要素が横に並ぶように設定しておきます。

```
32 #current {
33     background-image:url("img/icon_current.png");
34 }
35
36 #tracking {
37     background-image:url("img/icon_tracking.png");
38     display: none;
39 }
40
41 #delete {
42     background-image:url("img/icon_eraser.png");
43 }
44
45 #save {
46     background-image:url("img/icon_save.png");
47 }
48
49 #load {
50     background-image:url("img/icon_load.png");
51 }
52
```

```
53 .disabled {
54     background-color:     #ccc;
55 }
```

【追跡開始・停止】、【軌跡の削除】、【軌跡の保存】、【保存した軌跡の呼び出し】のボタンを CSS で設定します。最後の disabled によって、現在位置取得中に【軌跡の削除】、【軌跡の保存】、【保存した軌跡の呼び出し】のボタンが使用できないことを表すため、背景色を変更しています。

図　ボタンの色の変更

```
24         var map, marker, poly, watchId;
```

使う変数を宣言します。

```
25         var isTracking = false;
```

isTracking は現在位置情報を取得しているかを判定するものです。初期値は【停止】状態にするため、false としています。

```
27         //地図の作成
28         createMap();
29         getCurrentPosition();
```

createMap は実際に地図を作成する関数です。getCurrentPosition は現在位置を取得して地図を更新するための関数です。これらにより地図を初期化します。

```
31          //GPS の設定
32          var geoOptions = {
33              enableHighAccuracy: true,
34              timeout: 60000,
35              maximumAge: 0
36          };
```

geoOption は現在位置を取得するオプションです。60 秒でタイムアウトするように設定しています。

```
38          //継続的な位置の取得の開始
39          function startPositionTracking(){
40              watchId = navigator.geolocation.watchPosition(updateMap, null, geoOptions);
41              isTracking = true;
42          }
```

現在位置の取得するための関数です。この関数が実行されると、watchId に取得開始時に返される ID を格納し、isTracking を ture で更新します。また、コールバック関数として updateMap が実行されて地図が更新されます。

```
44          //継続的な位置の取得の停止
45          function stopPositionTracking(){
46              navigator.geolocation.clearWatch(watchId);
47              isTracking = false;
48          }
```

現在位置の取得を停止する関数です。最後に取得した watchId をもとに停止し、isTracking 関数を false で更新します。

```
50          //位置の取得
51          function getCurrentPosition(){
52              navigator.geolocation.getCurrentPosition(updateMap, null, geoOptions);
53          }
```

この関数によって現在位置を一度だけ取得し、コールバック関数として updateMap を実行し現在位置に地図を更新します。

```
55          //地図の更新
56          function updateMap(position){
57              var latlng = new google.maps.LatLng(position.coords.latitude,
    position.coords.longitude);
58              map.setCenter(latlng);
59              marker.setPosition(latlng);
60              poly.getPath().push(latlng);
61          }
```

地図の更新を実行する部分です。引数として渡された位置情報をもとに画面に表示されている地図の中心とマーカーの位置を更新します。poly の path に GPS で取得した latlng（緯度・軽度の値）を配列として格納して軌跡を描きます。

```
63          //地図の作成
64          function createMap(){
65              var mapOptions = {
66                  zoom: 15,
67                  center: new google.maps.LatLng(35.696802, 139.814136),
68                  mapTypeId: google.maps.MapTypeId.ROADMAP
69              };
70              map = new google.maps.Map($('#map')[0], mapOptions);
71
72              var image = new google.maps.MarkerImage(
73                  'img/marker.png', null, null,
74                  new google.maps.Point( 8, 8 ),
75                  new google.maps.Size( 17, 17 )
76              );
77              marker = new google.maps.Marker({
78                  position: new google.maps.LatLng(35.696802, 139.814136),
79                  map: map,
80                  icon: image,
81
82                  optimized: false,
83                  title: 'marker'
84              });
85
86              var polyLineOptions = {
```

```
87                  strokeWeight: 3,
88                  strokeColor: '#4682B4',
89                  strokeOpacity: 0.8
90              };
91              poly = new google.maps.Polyline(polyLineOptions);
92              poly.setMap(map);
93          }
```

実際に地図を作成する部分です。軌跡を初期化するために各種プロパティを持った polyLineOptions を作成しています。strokeWeight は軌跡の線幅、strokeColor は軌跡の色、strokeOpacity は軌跡の透明度を指定しています。
こちらの詳しい説明は Web サイトを参照してください。

URL：http://www.rs.kagu.tus.ac.jp/yamalab/2013/gokita/map1.html

```
57  @-webkit-keyframes pulsate {
58      from {
59          -webkit-transform: scale(2.0);
60          opacity: 1.0;
61      }
62      to {
63          -webkit-transform: scale(4.0);
64          opacity: 0;
65      }
66  }
67
68  div.gmnoprint[title="marker"] img {
69      display:none;
70  }
71
72  .pulse {
73      -webkit-animation:      pulsate 1.5s ease-in-out infinite;
74      border:                 1pt solid #4682B4;
75      border-radius:          100px;
76      background-color:       rgba(0,100,255, 0.1);
77  }
```

marker.png をただ表示するだけでなく、CSS でマーカー周りを波紋が広がるようにアニメーションを付けています。

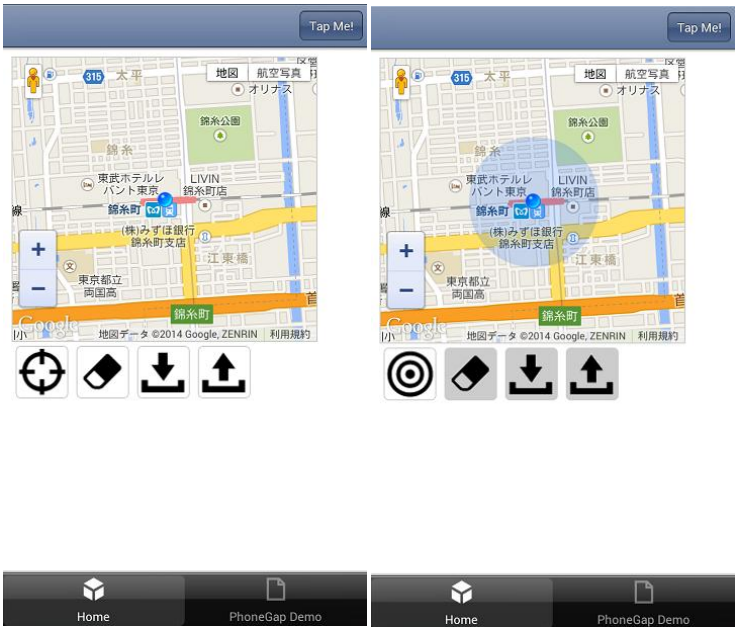

図　波紋が広がるアニメーション

```
95          //軌跡の削除
96          function deletePath() {
97              var path = poly.getPath();
98              while (path.getLength()) {
99                  path.pop();
100             }
101         }
```

軌跡を削除するための関数です。消しゴムボタンを押すと実行され、poly に格納されている位置情報の要素を削除します。

```
103         //軌跡の保存
104         function savePath() {
105             var path = poly.getPath();
106             for (var i=0; i<path.getLength(); i++) {
107                 localStorage[i] = path.getAt(i).toString();
108             }
109         }
```

位置情報の軌跡を LocalStorage に保存する関数です。保存ボタンを押すと実行され、poly に格納されている位置情報を取り出し文字列として保存します。

```
111         //軌跡の復元
112         function loadPath() {
113             for(var i=0; i<localStorage.length; i++){
114                 var latlng = localStorage[i].replace(/¥(|¥)/g,'').split(',');
115                 poly.getPath().push(new google.maps.LatLng(latlng[0], latlng[1]));
116             }
117         }
```

LocalStorage に保存した位置情報の軌跡を復元するための関数です。復元ボタンを押すと実行され、ストレージのデータを poly に格納し、保存した軌跡を表示します。

```
119         //監視開始ボタンのイベント
120         $('#current').bind('click', function() {
121             isTracking = true;
122             startPositionTracking();
123             $('#current').hide();
124             $('#tracking').show();
125             $('#delete').addClass('disabled');
126             $('#save').addClass('disabled');
127             $('#load').addClass('disabled');
128             $('div.gmnoprint[title="marker"]').addClass('pulse');
129         });
130
131         //監視停止ボタンのイベント
132         $('#tracking').bind('click', function() {
133             isTracking = false;
134             stopPositionTracking();
135             $('#current').show();
136             $('#tracking').hide();
137             $('#delete').removeClass('disabled');
138             $('#save').removeClass('disabled');
139             $('#load').removeClass('disabled');
140             $('div.gmnoprint[title="marker"]').removeClass('pulse');
141         });
```

【追跡開始・停止】ボタンのイベント動作を記述しています。前述の関数が実行されるように設定されています。

```
143         //消去ボタンのイベント
144         $('#delete').bind('click', function() {
145             if(!isTracking){
146                 if(confirm('消去しますか？')){
147                     deletePath();
148                 }
149             }
150         });
```

【軌跡の削除】ボタンを押すと delete 関数が実行されるように設定されています。

図　消しゴムボタンを押した際の画面

```
152         //保存ボタンのイベント
153         $('#save').bind('click', function() {
154             if(!isTracking){
155                 if(confirm('保存しますか？')){
156                     localStorage.clear();
157                     savePath();
158                 }
159             }
160         });
161
```

```
162         //復元ボタンのイベント
163         $('#load').bind('click', function() {
164             if(!isTracking){
165                 if(confirm('復元しますか？')){
166                     deletePath();
167                     loadPath();
168                 }
169             }
170         });
171     });
```

【軌跡の保存】、【保存した軌跡の呼び出し】ボタンを押した際のイベントを記述しています。それぞれ前述した関数を実行するように設定しています。

図　それぞれのボタンを押した際の画面

４．　おわりに

HTML5 と JavaScript を使ってネイティブアプリを作成することは一般的になりつつありますが、動作が遅いことやバックグラウンドアプリの作成に適さないことなど様々な問題が残っています。2014 年 1 月現在、スマートフォンを始めとする携帯端末は一昔前のパソコン程のスペックを持つ程までになっており、動作が遅い点は解決されていくと思われます。また、バックグラウンドアプリに適さないという問題は、今回用いた Phone Gap というツールがブラウザの様な機能を利用して HTML5 と JavaScript を動かしている点に依る部分が大きいため、０－３項で紹介した Unity や Titanium のように JavaScript をコンパイルしてネイティブアプリ化するツールを用いることで、解決することは可能です。以上のことからも、これから益々HTML5 や JavaScript などの Web 技術を使ったネイティブアプリが増えていくと思います。

プログラミングの知識は全くなく、結果としてお粗末なアプリを作り上げて終わってしまいましたが、まだ殆ど書籍化もされていない最新の技術に触れることができたこと、何よりプログラミングだけでなく PC 関連のものに対してアレルギーが全くなくなったことが非常に有益でした。一年間という短い期間でしたが、山本教授、研究室のメンバー（海江田さん、小池君、佐藤さん、鈴木君）にこの場を借りて御礼させて頂きます。ありがとうございました。

※作成したアプリはサイトに貼ってありますので、自由にダウロードしてください。使用する際は、GPS へのアクセスを許可してください。ダウンロード後、「はじめての Monaca アプリ」というアプリでインストールされてしまう場合がありますが、起動するともとに戻るはずです。（初期値がそうなっていたため）

図　GPS の許可（タップして許可）

ダウンロード：http://www.rs.kagu.tus.ac.jp/yamalab/2013/gokita/map2.html

５．　参考文献・Web

- 文献

1. HTML5 と JavaScript による iPhone / Android 両対応アプリ開発ガイド（翔泳社）
2. ゼロからわかる JavaScript 超入門 楽しいプログラミング（技術評論社）
3. はじめての Android アプリ作成 HTML5 入門（日経 BP 社）

- Web

1. スマートフォン・タブレット・PC の普及率の年代別グラフ
   http://www.itmedia.co.jp/news/articles/1310/04/news128.html
   端末の普及のデータを拝借しました。
2. Android Developer
   http://developer.android.com/index.html
   アプリ開発全般で参考にさせて頂きました。
3. Phone Gap Fan
   http://phonegap-fan.com/
   PhoneGap のインストールを始めとして、使用方法等で参考にさせて頂きました。
4. Unity 関連の記事
   http://www.atmarkit.co.jp/ait/articles/1204/04/news112.html
5. Titanium について
   http://gihyo.jp/dev/serial/01/titanium/0001
6. Phone Gap の API リファレンス
   http://docs.phonegap.com/jp/2.2.0/index.html
7. Monaca
   http://monaca.mobi/ja/