-Smart WeMo-

API ライブラリ仕様

ユビキタス・テクノロジーズ株式会社

序文

このドキュメントは、Smart WeMo の使用方法、表示出力、動作などを説明しています。

改訂履歴

版	日付	内容	作成者
2.0.0	04/27/2012	New creation	Jasmin Manalo

目次

1. Smart WeMo

1.1. 概要

Smart WeMo API は、HTML、CSS 、JavaScript の Web プログラミング言語使って、スマートフォン向けアプリケーションの開発をより簡単ににすることができます。

1.2. 目的

このドキュメントは、 Smart WeMo API ライブラリの使用方法を説明しています。

1.3 オープンソースライブラリ

Smart WeMo API 開発で使用したオープンソースライブラリのライセンスは下記になります：

Library Name	License Terms
Facebook	*https://developers.facebook.com/licensing/* Creative Commons Attribution-ShareAlike 3.0 license –規定内であればフリーライセンス。商用利用も可能。主な内容は次の通り： - 開発者は、開発者向け原則とポリシーに定められたすべての制約事項を遵守すること。 - 開発した製品を流通また配信する際は、すべての規約事項を遵守すること。
Twitter	*https://dev.twitter.com/terms/api-terms*
httpclient-4.0.1.jar	Apache License 2.0 -規定内であればフリーライセンス。商用利用も可能。
json_simple-1.1.jar	*http://code.google.com/p/json-simple/* Apache License 2.0 -規定内であればフリーライセンス。商用利用も可能。
signpost-commonshttp4-1.2.1.1.jar	*http://code.google.com/p/oauth-signpost/* Apache License 2.0 -規定内であればフリーライセンス。商用利用も可能。
signpost-core-1.2.1.1.jar	*http://code.google.com/p/oauth-signpost/downloads/list* Apache License 2.0 -規定内であればフリーライセンス。商用利用も可能。
twitter4j-core-android-2.2.5.jar	*http://code.google.com/p/posit-mobile/* GNU Lesser GPL – Lessor GPL を使用すると、独自のプログラムでライブラリの使用可能
Base64 encoding/decoding	*http://cocoawithlove.com/2009/06/base64-encoding-options-on-mac-and.html* 規定内であればフリーライセンス。商用利用も可能。

2. SBWemo.js

説明： SBWemo.js には、HTML/Javascript を Android,又は iOS 向け SmartWemo のコアライブラリへ接続するための javascript API が含まれています。これはリリースパッケージや、Android 又は iOS 向けアプリケーションのプロジェクトディレクトリ内にも含まれます。このファイルの実装方法は別ファイル「SmartWemo Android 向け取扱説明書 2.0 版」、又は「Smart Wemo iOS 向け取扱説明書 2.0 版」で確認してください。全ての API を起動させるには SBWemo.js を初期化する必要があります。

メソッド: initLib()

2.1 initLib()

initLib()

initLib()は、ライブラリを初期化する関数です。ライブラリ API を使用するにはライブラリを初期化しなければならないため、とても重要です。この関数は HTML ページが作成される度に呼ばれます。

例:

```
<!-- Alerts, Prompts -->
<script src="jquery.alerts.js" type="text/javascript"></script>
<link href="jquery.alerts.css" rel="stylesheet" type="text/css" media="screen" />

<script type="text/javascript" src="SBWemo.js"></script>
<script>
        function init() {
        initLib();
        }

        $(document).ready(function() {
            $("#btnCheckInternet").click(function() {
                Connectivity.hasInternet(
                    function(okArgs) {
                        if (okArgs) {
                            jAlert("Device is connected to the internet!","Connectivity");
                        } else {
                            jAlert("Device is not connected to the internet!","Connectivity");
                        }
                    },
                    function(ngArgs) {
                        //alert(ngArgs);
                        jAlert(ngArgs,"Error Message");
                    }
                );
            });
        });
    </script>
</head>
<body onload="init()">
    <div data-role="header" data-position="inline">
```

3. Accelerometer

説明： 端末の加速度センサーから値を取得するオブジェクトを定義します。
メソッド: startListener(), stopListener()

3.1. startListener()

startListener(*successCallback(accelerometerObject), errorCallback(errorString) sampleRateObject*)

successCallback(accelerometerObject) - センサー情報取得の開始が成功したときに呼ばれる関数
accelerometerObject - センサーの値を保持するオブジェクト
errorCallback(errorString) – startListener ()メソッドエラーのときに呼ばれる関数
errorString (オプション) - 返されたエラーの文字列
sampleRateObject- １秒毎に取得できる数を含んでいるオブジェクト

*startListener()*メソッドは、加速度センサーを開始して、その値をオブジェクトとして取得するのに使用します。

sampleRateObject の内容:

sampleRateObject のメンバー	説明
samplingRate	1～60 の値は、1 秒毎に加速度センサー値が取得するインスタンスの数を表しています。

accelerometerObject から返されるデータ：

accelerometerObject のメンバー	説明
X	X 軸方向への加速度センサー値
Y	Y 軸方向への加速度センサー値
Z	Z 軸方向への加速度センサー値

例: 端末の加速度センサーの値を印刷します。

```
<!DOCTYPE html>
<html>
<head>
    <link href="http://code.jquery.com/mobile/latest/jquery.mobile.min.css" rel="stylesheet" type="text/css" />
    <script src="http://code.jquery.com/jquery-1.6.4.min.js"> </script>
    <script src="http://code.jquery.com/mobile/latest/jquery.mobile.min.js">    </script>
    <script type="text/javascript" src="HybridLib.js"> </script>
  <script>
        $(document).ready(function() {
            Accelerometer.startListener(onSuccess);
            function onSuccess(acc) {
                if (typeof acc !== "undefined") {
                    $("#txtAccX").text("" + acc.aX);
                    $("#txtAccY").text("" + acc.aY);
                    $("#txtAccZ").text("" + acc.aZ);
                }
            }
        });
  </script>
</head>
<body>
    <p>
        X: <span id="txtAccX">0.0</span>
        Y: <span id="txtAccY">0.0</span>
        Z: <span id="txtAccZ">0.0</span>
    </p>
</body>
</html>
```

3.2. stopListener()

stopListener(*successCallback*)
successCallback – センサー情報取得の開始が成功したときに呼ばれる関数。関数には、センサーの情報にアクセスするためのオブジェクトパラメータが必要です。
stopListener(*successCallback*, *errorCallback(errorString)*)
successCallback - センサー情報取得の開始が成功した時に呼ばれる関数。関数には、センサーの情報にアクセスするためのオブジェクトパラメータが必要です。 *errorCallback* - stopListener()メソッドエラー時に呼ばれる関数 *errorString* (オプション) - 返されたエラーの文字列

stopListener() メソッドは、 startListener() メソッドを停止させて、センサーからのデータ取得を停止させるのに使用します。

例: 加速度センサーから取得した値を印刷して、センサーを停止させます。

```
<!DOCTYPE html>
<html>
<head>
    <link href="http://code.jquery.com/mobile/latest/jquery.mobile.min.css" rel="stylesheet" type="text/css" />
    <script src="http://code.jquery.com/jquery-1.6.4.min.js"> </script>
    <script src="http://code.jquery.com/mobile/latest/jquery.mobile.min.js">    </script>
    <script type="text/javascript" src="HybridLib.js"> </script>
  <script>
      $(document).ready(function() {
          Accelerometer.startListener(onSuccess);
          function onSuccess(acc) {
              if (typeof acc !== "undefined") {
                  $("#txtAccX").text("" + acc.aX);
                  $("#txtAccY").text("" + acc.aY);
                  $("#txtAccZ").text("" + acc.aZ);
```

```
                        Accelerometer.stopListener(function(){alert("stop");});
                    }
                }
            });
    </script>
</head>
<body>
    <p>
        X: <span id="txtAccX">0.0</span>
        Y: <span id="txtAccY">0.0</span>
        Z: <span id="txtAccZ">0.0</span>
    </p>
</body>
</html>
```

4. AudioSettings

説明: 端末の音声設定を取得するためのオブジェクトを定義します。

メソッド: disableVibrate(), enableVibrate(), getMediaVolume(), getRingtoneVolume(), isVibrateEnabled(), setMediaVolume(), setRingtone(), setRingtoneVolume()

4.1. disableVibrate()

disableVibrate(*successCallback()*)
successCallback - バイブレーションの有効化が成功したときに呼ばれる関数
disableVibrate(*successCallback(), errorCallback(errorString)*)
successCallback - バイブレーションの有効化が成功したときに呼ばれる関数 *errorCallback* - バイブレーションの有効化が失敗したときに呼ばれる関数 *errorString* (オプション) - 返されたエラーの文字列

disableVibrate() メソッドは、端末のバイブレーション機能を無効にするときに使用します。

例：端末で設定したバイブレーションを無効にします。

```
<!DOCTYPE html>
<html>
<head>
    <link href="http://code.jquery.com/mobile/latest/jquery.mobile.min.css" rel="stylesheet" type="text/css" />
    <script src="http://code.jquery.com/jquery-1.6.4.min.js"> </script>
    <script src="http://code.jquery.com/mobile/latest/jquery.mobile.min.js">    </script>
    <script type="text/javascript" src="HybridLib.js"> </script>
  <script>
      $(document).ready(function() {
          AudioSettings.disableVibrate(onSuccess, onError);
       });
       function onSuccess() {
```

```
                alert("Success!");
            }
            function onError() {
                alert("Error!");
            }
    </script>
</head>
<body>
</body>
</html>
```

4.2. enableVibrate()

enableVibrate(*successCallback()*)
successCallback - バイブレーションの有効化が成功したときに呼ばれる関数
enableVibrate(*successCallback(), errorCallback(errorString)*)
successCallback - バイブレーションの有効化が成功したときに呼ばれる関数 *errorCallback* - バイブレーションの有効化が失敗したときに呼ばれる関数 *errorString* (オプション) - 返されたエラーの文字列

enableVibrate() メソッドは、端末のバイブレーションを有効にするときに使用します。

例：端末のバイブレーションを有効にします。

```
<!DOCTYPE html>
<html>
<head>
    <link href="http://code.jquery.com/mobile/latest/jquery.mobile.min.css" rel="stylesheet" type="text/css" />
    <script src="http://code.jquery.com/jquery-1.6.4.min.js"> </script>
    <script src="http://code.jquery.com/mobile/latest/jquery.mobile.min.js">  </script>
    <script type="text/javascript" src="HybridLib.js"> </script>
  <script>
      $(document).ready(function() {
          AudioSettings.enableVibrate(onSuccess, onError);
       });
       function onSuccess() {
           alert("Success!");
       }
       function onError() {
            alert("Error!");
       }
  </script>
</head>
<body>
```

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

4.3. getMediaVolume()

getMediaVolume(*successCallback(volumeInt)*)
successCallback – メディア音量の取得が成功したときに呼ばれる関数 *volumeInt* - 返されたメディア音量
getMediaVolume(*successCallback(volumeInt)*, *errorCallback(errorString)*)
successCallback - メディア音量の取得が成功したときに呼ばれる関数 *volumeInt* - 返されたメディア音量 *errorCallback* - メディア音量の取得が失敗したときに呼ばれる関数 *errorString* (オプション)- 返されたエラーの文字列

getMediaVolume()メソッドは、端末のメディア音量を取得するのに使用します。

例：端末のメディア音量を表示します。

```
<!DOCTYPE html>
<html>
<head>
    <link href="http://code.jquery.com/mobile/latest/jquery.mobile.min.css" rel="stylesheet" type="text/css" />
    <script src="http://code.jquery.com/jquery-1.6.4.min.js"> </script>
    <script src="http://code.jquery.com/mobile/latest/jquery.mobile.min.js">    </script>
    <script type="text/javascript" src="HybridLib.js"> </script>
  <script>
      $(document).ready(function() {
          AudioSettings.getMediaVolume(onSuccess, onError);
       });
       function onSuccess(value) {
          alert("Success! " + value);
       }
       function onError() {
          alert("Error!");
       }
  </script>
</head>
```

```
<body>

</body>

</html>
```

4.4. getRingtoneVolume()

getRingtoneVolume(*successCallback(volumeInt)*)
successCallback - メディア音量の取得が成功したときに呼ばれる関数 *volumeInt*- 返されたメディア音量
getRingtoneVolume(*successCallback(volumeInt), errorCallback(errorString)*)
successCallback - メディア音量の取得が成功したときに呼ばれる関数 *volumeInt*- 返されたメディア音量 *errorCallback* - メディア音量の取得が失敗したときに呼ばれる関数 *errorString* (オプション) - 返されたエラーの文字列

getRingtoneVolume() メソッドは、端末のメディア音量を取得するのに使用します。

例：端末のメディア音量を表示します。

```
<!DOCTYPE html>
<html>
<head>
    <link href="http://code.jquery.com/mobile/latest/jquery.mobile.min.css" rel="stylesheet" type="text/css" />
    <script src="http://code.jquery.com/jquery-1.6.4.min.js"> </script>
    <script src="http://code.jquery.com/mobile/latest/jquery.mobile.min.js">    </script>
    <script type="text/javascript" src="HybridLib.js"> </script>
  <script>
      $(document).ready(function() {
          AudioSettings.getRingtoneVolume(onSuccess, onError);
       });
       function onSuccess(value) {
           alert("Success! " + value);
       }
       function onError() {
           alert("Error!");
       }
  </script>
```

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

4.5. isVibrateEnabled()

isVibrateEnabled(*successCallback(isVibrateBoolean)*)

successCallback - 端末のバイブレーション設定の確認が成功したときに呼ばれる関数
isVibrateBoolean - 端末のバイブレーション設定を確認すると返されるブール値

isVibrateEnabled(*successCallback(isVibrateBoolean)*, *errorCallback(errorString)*)

successCallback - 端末のバイブレーション設定の確認が成功したときに呼ばれる関数
isVibrateBoolean - 端末のバイブレーション設定を確認すると返されるブール値
errorCallback - 端末のバイブレーション設定の確認が失敗したときに呼ばれる関数
errorString (オプション) - 返されたエラーの文字列

isVibrateEnabled()メソッドは、端末のバイブレーション設定を確認するために使用されます。

例：端末のバイブレーション設定を確認します。

```
<!DOCTYPE html>
<html>
<head>
    <link href="http://code.jquery.com/mobile/latest/jquery.mobile.min.css" rel="stylesheet" type="text/css" />
    <script src="http://code.jquery.com/jquery-1.6.4.min.js"> </script>
    <script src="http://code.jquery.com/mobile/latest/jquery.mobile.min.js">    </script>
    <script type="text/javascript" src="HybridLib.js"> </script>
  <script>
      $(document).ready(function() {
          AudioSettings.isVibrate(onSuccess, onError);
       });
        function onSuccess(value) {
            alert("Success! " + value);
        }
        function onError() {
```

```
                    alert("Error!");
                }
    </script>
</head>
<body>
</body>
</html>
```

4.6. setMediaVolume()

setMediaVolume(*successCallback(), errorCallback(errorString), volumeRaiseObject*)

successCallback - 端末のメディア音量の設定が成功したときに呼ばれる関数
errorCallback - 端末のメディア音量の設定が失敗したときに呼ばれる関数
errorString (オプション) - 返されたエラーの文字列

setMediaVolume()メソッドは、volumeRaiseObject のメンバー「toRaise」 の boolean 値を TRUE または FALSE に指定して、端末のメディア音量を調整します。

例：端末のメディア音量を上げます。

```
<!DOCTYPE html>
<html>
<head>
    <link href="http://code.jquery.com/mobile/latest/jquery.mobile.min.css" rel="stylesheet" type="text/css" />
    <script src="http://code.jquery.com/jquery-1.6.4.min.js"> </script>
    <script src="http://code.jquery.com/mobile/latest/jquery.mobile.min.js">    </script>
    <script type="text/javascript" src="HybridLib.js"> </script>
  <script>
        $(document).ready(function() {
            params = {toRaise : false};
            AudioSettings.setMediaVolume(onSuccess, onError, params);
        });
        function onSuccess(value) {
            alert("Success!");
        }
        function onError() {
            alert("Error!");
        }
  </script>
</head>
<body>
```

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

4.7. setRingtone()

setRingtone()

setRingtone()メソッドは、端末の着信音を設定するのに使用されます。着信音の選択肢を表示します。

例：端末の着信音を設定することができます。

```
<!DOCTYPE html>
<html>
<head>
    <link href="http://code.jquery.com/mobile/latest/jquery.mobile.min.css" rel="stylesheet" type="text/css" />
    <script src="http://code.jquery.com/jquery-1.6.4.min.js"> </script>
    <script src="http://code.jquery.com/mobile/latest/jquery.mobile.min.js">    </script>
    <script type="text/javascript" src="HybridLib.js"> </script>
  <script>
        $(document).ready(function() {
            AudioSettings.setRingtone(onSuccess, onError, params);
        });
        function onSuccess(value) {
            alert("Success! " + value);
        }
        function onError() {
            alert("Error!");
        }
  </script>
</head>
<body>
</body>
</html>
```

4.8. setRingtoneVolume()

setRingtoneVolume(*successCallback(), errorCallback(errorString), volumeRaiseObject*)

successCallback - 端末の着信音の音量設定が成功したときに呼ばれる関数
errorCallback - 端末の着信音の音量設定が失敗したときに呼ばれる関数
errorString (オプション) - 返されたエラーの文字列

setMediaVolume()メソッドは、volumeRaiseObject のメンバー「toRaise」のブール値を TRUE もしくは FALSE に指定することで着信音の音量を調整します。

例：端末のメディア音量を上げます。

```
<!DOCTYPE html>
<html>
<head>
    <link href="http://code.jquery.com/mobile/latest/jquery.mobile.min.css" rel="stylesheet" type="text/css" />
    <script src="http://code.jquery.com/jquery-1.6.4.min.js"> </script>
    <script src="http://code.jquery.com/mobile/latest/jquery.mobile.min.js">    </script>
    <script type="text/javascript" src="HybridLib.js"> </script>
  <script>
        $(document).ready(function() {
            params = {toRaise : false};
            AudioSettings.setRingtoneVolume(onSuccess, onError, params);
        });
        function onSuccess(value) {
            alert("Success!");
        }
        function onError() {
            alert("Error!");
        }
  </script>
</head>
<body>
</body>
</html>
```

5. Camera

説明: 端末のカメラ機能にアクセスするオブジェクトを定義します。
メソッド: captureImage()

5.1. captureImage()

captureImage(*successCallback(imageFilePath)*)
successCallback - 端末のカメラ起動が成功したときに呼ばれる関数 端末のカメラから返された画像ファイルのパスにアクセスするために文字列のパラメーターが必要です。
captureImage (*successCallback(imageFilePath)*), *errorCallback(errorString)*)
successCallback - 端末のカメラ起動が成功したときに呼ばれる関数 端末のカメラから返された画像ファイルのパスにアクセスするために文字列のパラメーターが必要です。 *errorCallback* - captureImage()メソッド がエラーのときに呼ばれる関数 *errorString* (オプション) - 返されたエラー文字列

captureImage() メソッドは、端末のカメラを起動させるときに使用します。このメソッドで、撮影した画像ファイルのパスも取得することもできます。

例：端末のカメラを取得して画像を表示します。

```
<!DOCTYPE html>
<html>
<head>
    <link href="http://code.jquery.com/mobile/latest/jquery.mobile.min.css" rel="stylesheet" type="text/css" />
    <script src="http://code.jquery.com/jquery-1.6.4.min.js"> </script>
    <script src="http://code.jquery.com/mobile/latest/jquery.mobile.min.js">  </script>
    <script type="text/javascript" src="HybridLib.js"> </script>
  <script>
        $(document).ready(function() {
            $("#btnCamera").click(function() {
```

```
                        var params = new CameraParams();
                        Camera.captureImage(captureSuccess, captureError);
                });
});

function captureSuccess(imageFilePath) {
                alert("Image Capture Success!");
                $("#imgSampleFilePath ").text("" + imageFilePath);
}

function captureError(strError) {
                alert(strError);
}    </script>
</head>
<body>
      Image File Path: <span id="imgSampleFilePath">n/a</span>
      <button id="btnCamera">Open Device Camera</button>
</body>
</html>
```

6. Connectivity

説明：端末のインターネット接続情報を取得するオブジェクトを定義します。
メソッド: getInternetSource(), hasInternet()

6.1. getInternetSource()

getInternetSource(*successCallback(typeString)*)
successCallback - 端末のインターネット接続情報の取得が成功したときに呼ばれる関数 *typeString* - 端末から返された文字列の値
getInternetSource(*successCallback(typeString), errorCallback(errorString)*)
successCallback - 端末のインターネット接続情報の取得が成功したときに呼ばれる関数 *typeString*- 端末から返された文字列の値 *errorCallback*- 端末のインターネット接続情報の取得が失敗したときに呼ばれる関数 *errorString* (オプション) - 返されたエラーの文字列

getInternetSource() メソッドは、端末のインターネットのリソースにある情報を取得するのに使用されます。

例：端末のインターネットリソースを取得します。

```
<!DOCTYPE html>
<html>
<head>
    <link href="http://code.jquery.com/mobile/latest/jquery.mobile.min.css" rel="stylesheet" type="text/css" />
    <script src="http://code.jquery.com/jquery-1.6.4.min.js"> </script>
    <script src="http://code.jquery.com/mobile/latest/jquery.mobile.min.js">    </script>
    <script type="text/javascript" src="HybridLib.js"> </script>
  <script>
      $(document).ready(function() {
          $("#btnGetInternetSource").click(function() {
```

```
                    Connectivity.getInternetSource(
                        function(okArgs) {
                            alert("Internet connectivity via " + okArgs);
                        },
                        function(ngArgs) {
                            alert(ngArgs);
                        }
                    );
                });
            });
    </script>
</head>
<body>
    <div>
        <input type="button" id="btnGetInternetSource" value="Get Internet Source" />
    </div>
</body>
</html>
```

6.2. hasInternet()

hasInternet(*successCallback(isConnectedBoolean)*)
successCallback -端末のネットワーク接続状態の取得が成功したときに呼ばれる関数 *isConnectedBoolean* - 端末から返されたブール値
hasInternet (*successCallback(isConnectedBoolean)*, *errorCallback(errorString)*)
successCallback -端末のネットワーク接続状態の取得が成功したときに呼ばれる関数 *isConnectedBoolean* - 端末から返された boolean 値 *errorCallback*- 端末のインターネット接続状態の取得が失敗したときに呼ばれる関数 *errorString* (オプション) - 返された文字列のエラー

hasInternet() メソッドは、 TRUE または FALSE の返し値で端末のインターネット接続状態を取得するのに使用されます。

例：端末のネットワーク接続状態を確認します。

```
<!DOCTYPE html>
<html>
<head>
    <link href="http://code.jquery.com/mobile/latest/jquery.mobile.min.css" rel="stylesheet" type="text/css" />
    <script src="http://code.jquery.com/jquery-1.6.4.min.js"> </script>
    <script src="http://code.jquery.com/mobile/latest/jquery.mobile.min.js">    </script>
    <script type="text/javascript" src="HybridLib.js"> </script>
  <script>
      $(document).ready(function() {
          $("#btnCheckInternet").click(function() {
              Connectivity.hasInternet (
                  function(okArgs) {
                      if (okArgs) {
                          alert("Device is connected to the internet!");
                      } else {
```

```
                                        alert("Device is not connected to the internet!");
                                    }
                                },
                                function(ngArgs) {
                                    alert(ngArgs);
                                }
                            );
                        });
                });
    </script>
</head>
<body>
    <div>
        <input type="button" id=" btnCheckInternet " value="HasInternet" />
    </div>
</body>
</html>
```

7. Contacts

説明: 端末に保存してある連絡先リストと連絡先の詳細にアクセスするオブジェクトを定義します。

メソッド: addContact(), deleteContact(), editContact(), getContactDetails(), getContacts()

7.1. addContact()

addContact(*successCallback(sucessString), errorCallback(errorString), contactObject*)
successCallback – 操作が成功したときに呼ばれる関数 *sucessString*(オプション) - 返された成功メッセージ *errorCallback* – 操作失敗したときに呼ばれる関数 *errorString* (オプション) - 返されたエラーのメッセージ *contactObject* - 連絡先の詳細を含むオブジェクト

addContact()メソッドは、連絡先を端末へ追加するときに使用します。その追加する連絡先の詳細は「contactObject」に含まれます。
「contactObject」が保持する詳細内容については、7.6 章を参照してください。その中の「firstName」は必須になっています。

例: 新しい連絡先を追加します。

```
<!DOCTYPE html>
<html>
<head>
    <link href="http://code.jquery.com/mobile/latest/jquery.mobile.min.css" rel="stylesheet" type="text/css" />
    <script src="http://code.jquery.com/jquery-1.6.4.min.js"> </script>
    <script src="http://code.jquery.com/mobile/latest/jquery.mobile.min.js">  </script>
    <script type="text/javascript" src="HybridLib.js"> </script>
  <script>
        $(document).ready(function() {
            Contacts.addContact(null, null,
                {
```

```
                    firstName: “Albert”,
                    lastName: “Einstein”
                });
        });
    </script>
</head>
<body>
</body>
</html>
```

7.2. deleteContact()

deleteContact(*successCallback(), errorCallback(errorString), paramObject*)

successCallback – 操作が成功したときに呼ばれる関数
errorCallback - 操作が失敗したときに呼ばれる関数
errorString（オプション）- 返されたエラーの文字列
paramObject - 削除されるコンタクト ID を含むオブジェクト

deleteContact() メソッドは、「paramObject」の ID で示された連絡先の削除に使用します。

例：５の ID を持つ連絡先を削除します。

```
<!DOCTYPE html>
<html>
<head>
    <link href="http://code.jquery.com/mobile/latest/jquery.mobile.min.css" rel="stylesheet" type="text/css" />
    <script src="http://code.jquery.com/jquery-1.6.4.min.js"> </script>
    <script src="http://code.jquery.com/mobile/latest/jquery.mobile.min.js">  </script>
    <script type="text/javascript" src="HybridLib.js"> </script>
  <script>
      $(document).ready(function() {
          Contacts.deleteContact(null, null, {'id':5});
      });
  </script>
</head>
<body>
</body>
</html>
```

7.3. editContact()

editContact(*successCallback(), errorCallback(errorString), contactObject*)

successCallback - 操作が成功したときに呼ばれる関数
errorCallback - 操作が失敗したときに呼ばれる関数
errorString(オプション)- 返されたエラーの文字列
contactObject - 削除されるコンタクト ID を含むオブジェクト

editContact()メソッドは、連絡先の編集に使用されます。
追加する連絡先の詳細は「contactObject」に含まれます。
「contactObject」が保持する内容については、 7.6 章を参照してください。

例: 7 の ID を持つ連絡先を編集します。

```
<!DOCTYPE html>
<html>
<head>
    <link href="http://code.jquery.com/mobile/latest/jquery.mobile.min.css" rel="stylesheet" type="text/css" />
    <script src="http://code.jquery.com/jquery-1.6.4.min.js"> </script>
    <script src="http://code.jquery.com/mobile/latest/jquery.mobile.min.js">  </script>
    <script type="text/javascript" src="HybridLib.js"> </script>
  <script>
        $(document).ready(function() {
            Contacts.editContact(null, null,
                {
                    id: 7,
                    firstName: “Albert”,
                    lastName: “Einstein”
                });
        });
  </script>
</head>
<body></body></html>
```

7.4. getContactDetails()

getContactDetails(*successCallback(detailArray), errorCallback(errorString), paramObject*)

successCallback - 操作が成功したときに呼ばれる関数
detailArray- 端末から返された連絡先の詳細の配列。
errorCallback - 操作が成功したときに呼ばれる関数
errorString(オプション)- 返されたエラーの文字列
paramObject - 取得するコンタクト ID を含むオブジェクト

getContactDetails() メソッドは paramObject の ID で示された連絡先の情報を取得します。

例: ８の ID を持つ連絡先の詳細を表示します。

```
<!DOCTYPE html>
<html>
<head>
    <link href="http://code.jquery.com/mobile/latest/jquery.mobile.min.css" rel="stylesheet" type="text/css" />
    <script src="http://code.jquery.com/jquery-1.6.4.min.js"> </script>
    <script src="http://code.jquery.com/mobile/latest/jquery.mobile.min.js">    </script>
    <script type="text/javascript" src="HybridLib.js"> </script>
  <script>
      $(document).ready(function() {
          var contacts = Contacts.getContactDetails(null, null, {'id':8});
          $("input#id").val(id);
          $("input#first").val(contacts['first_name']);
          $("input#last").val(contacts['last_name']);
          $("input#phone").val(contacts['phone_number']);
          $("input#email").val(contacts['email_value']);
          $("input#street").val(contacts['address_street']);
          $("input#city").val(contacts['address_city']);
          $("input#zip").val(contacts['address_zip']);
      });
  </script>
```

```
</head>
<body>
    <table border='0'>
        <tr>
            <td>First Name</td>
            <td><input type="text" name="first" id="first" /></td>
        </tr>
        <tr>
            <td>Last Name</td>
            <td><input type="text" name="last" id="last" /></td>
        </tr>
        <tr>
            <td>Phone</td>
            <td><input type="text" name="phone" id="phone" /></td>
        </tr>
        <tr>
            <td>Email</td>
            <td><input type="text" name="email" id="email" /></td>
        </tr>
        <tr>
            <td>Street</td>
            <td><input type="text" name="street" id="street" /></td>
        </tr>
        <tr>
            <td>City</td>
            <td><input type="text" name="city" id="city" /></td>
        </tr>
        <tr>
            <td>Zip</td>
            <td><input type="text" name="zip" id="zip" /></td>
        </tr>
    </table>
</body>
</html>
```

7.5. getContacts()

getContacts(*successCallback(detailArray), errorCallback(errorString)*)

successCallback - 操作が成功したときに呼ばれる関数
errorCallback - 操作が失敗したときに呼ばれる関数
errorString(オプション)- 返されたエラーの文字列

getContacts() メソッドは、端末から連絡先リストを取得します。

例：連絡先リストを取得します。

```
<!DOCTYPE html>
<html>
<head>
    <link href="http://code.jquery.com/mobile/latest/jquery.mobile.min.css" rel="stylesheet" type="text/css" />
    <script src="http://code.jquery.com/jquery-1.6.4.min.js"> </script>
    <script src="http://code.jquery.com/mobile/latest/jquery.mobile.min.js">    </script>
    <script type="text/javascript" src="HybridLib.js"> </script>
  <script>
      $(document).ready(function() {
          $('ul#contactList').html("");
          var contacts = Contacts.getContacts();
          var char = '';

          if("undefined" !== typeof contacts) {
              $.each(contacts, function(index, value)
              {
                  if(char != value['name'].charAt(0).toUpperCase())
                  {
                      char = value['name'].charAt(0).toUpperCase();
                      $('ul#contactList').append("<li data-role='list-divider'>"+char+"</li>");
                   }
                   $('ul#contactList').append("<li>"+value['name']+"</li>");
              });
          }
```

```
                $('ul').listview('refresh');
            });
        </script>
</head>
<body>
        <div>
            <ul id="contactList" data-role="listview" data-inset="true"></ul>
        </div>
</body>
</html>
```

7.6. contactObject

paramObject に含まれるメンバーのリスト：

メンバー	説明
id	連絡先の ID
firstName	その連絡先の名前・名
lastName	その連絡先の名前・性
phone	その連絡先の電話番号.
email	その連絡先のメールアドレス.
addressStreet	その連絡先の住所・番地
addressCity	その連絡先の住所・市町村
addressZip	その連絡先の郵便番号

8. Database

説明：端末のデータベースに接続するオブジェクトを定義します。

メソッド: createDatabase(), createTable(), deleteRecord(), dropTable(), getColumnDetails(), getDatabases(), getTables(), insertRecord(), queryRecord(), updateRecord()

8.1. createDatabase()

createDatabase(*successCallback(), errorCallback(), DBNameObject*)

successCallback - データベース作成が成功したときに呼ばれる関数
errorCallback - データベース作成が失敗したときに呼ばれる関数
DBNameObject - 作成するデータベースの名前を含んでいるオブジェクト

createDatabase() メソッドは、データベースを作成します。
そのデータベース名は「DBNameObject」の dbname で返された名前になります。

例：データベースを作成します。

```
<!DOCTYPE html>
<html>
<head>
    <link href="http://code.jquery.com/mobile/latest/jquery.mobile.min.css" rel="stylesheet" type="text/css" />
    <script src="http://code.jquery.com/jquery-1.6.4.min.js"> </script>
    <script src="http://code.jquery.com/mobile/latest/jquery.mobile.min.js">    </script>
    <script type="text/javascript" src="HybridLib.js"> </script>
  <script>
        function create() {
            var params = {'dbname':$("input#name").val()};
            Database.createDatabase(null, null, params);
        }
  </script>
</head>
```

```
<body>
	<div>
		<table border='0'>
			<tr>
				<td>Database Name</td>
			</tr>
			<tr>
				<td><input type="text" name="name" id="name" onFocus="setEnalbed(false)"
onBlur="isBlank(this)" /></td>
			</tr>
		</table>
		<button id="btnCreate" data-inline="true" disabled ="true" onClick= "create()"> Create </button>
	</div>
</body>
</html>
```

8.2. createTable()

createTable(*successCallback(), errorCallback(), DBDataObject*)

successCallback – テーブルの作成が成功したときに呼ばれる関数
errorCallback – テーブルの作成が失敗したときに呼ばれる関数
DBDataObject – テーブルを作成するのに必要なデータを含んでいるオブジェクト

createTable() メソッドは、指定したデータベースにテーブルを作成します。必要なデータは「DBDataObject」に含まれています。
必須なテーブルのデータは次の通りです：

DBDataObject メンバー	説明
dbname	テーブルを作成したデータベースの名称。データベースは存在します。
tablename	作成するテーブルの名称。
columns	columns オブジェクトの配列。Columns に含まれるメンバーは次の通り： name - カラム名を示します。 type - カラム型の値を示します。

例：テーブルを作成します。

```
<!DOCTYPE html>
<html>
<head>
    <link href="http://code.jquery.com/mobile/latest/jquery.mobile.min.css" rel="stylesheet" type="text/css" />
    <script src="http://code.jquery.com/jquery-1.6.4.min.js"> </script>
    <script src="http://code.jquery.com/mobile/latest/jquery.mobile.min.js">    </script>
    <script type="text/javascript" src="HybridLib.js"> </script>
  <script>
      $(document).ready(function() {
          var columns = new Array();
          columns[0] = {name: “col_id”, type: “INTEGER”};
          columns[1] = {name: “col_val”, type: “TEXT”}
```

```
                var params = {
                    dbname: “sample”,
                    tablename: “demo”,
                    columns:columns
                };
                Database.createTable(function(){alert(“table created!”);}, null, params);
            });
    </script>
</head>
<body>
    <div>
    </div>
</body>
</html>
```

8.3. deleteRecord()

deleteRecord(*successCallback(), errorCallback(), DBDataObject*)

successCallback - 記録の削除が成功したときに呼ばれる関数
errorCallback -記録の削除が失敗したときに呼ばれる関数
DBDataObject - 記録の削除に必要なデータを含むオブジェクト

deleteRecord()メソッドは、指定したデータベースのテーブルから記録を削除します。必要なデータは DBDataObject に含まれます。
必要な記録データは次の通りです：

DBDataObject Member	説明
dbname	データベースの名前
tablename	テーブルの名前
where	(オプション) 基本的な SQL の WHERE 句

例：（サンプル）テーブルで 1 の col_ID を持つレコードを削除します。

```
<!DOCTYPE html>
<html>
<head>
    <link href="http://code.jquery.com/mobile/latest/jquery.mobile.min.css" rel="stylesheet" type="text/css" />
    <script src="http://code.jquery.com/jquery-1.6.4.min.js"> </script>
    <script src="http://code.jquery.com/mobile/latest/jquery.mobile.min.js">    </script>
    <script type="text/javascript" src="HybridLib.js"> </script>
  <script>
        $(document).ready(function() {
            var params = {
                dbname: “sample”,
                tablename: “demo”,
                where:columns
            };
            Database.deleteRecord(function(){alert(“record Deleted!”);}, null, params);
        });
```

```
}    </script>
</head>
<body>
      <div>
      </div>
</body>
</html>
```

8.4. dropTable()

dropTable(*successCallback(), errorCallback(), DBDataObject*)

successCallback – テーブルの削除が成功したときに呼ばれる関数
errorCallback- テーブルの削除が成功したときに呼ばれる関数
DBDataObject- データベースを含むオブジェクト。テーブル名は削除するテーブルを識別するために使用します

dropTable() メソッドは、指定したデータベースのテーブルを削除します。 必要なテーブルデータは、「DBDataObjec」に含まれます。
必須なデータは次の通りです：

DBDataObject メンバー	説明
dbname	テーブルがあるデータベースの名前
tablename	削除するテーブルの名前

例：（サンプル）テーブルを削除します。

```
<!DOCTYPE html>
<html>
<head>
    <link href="http://code.jquery.com/mobile/latest/jquery.mobile.min.css" rel="stylesheet" type="text/css" />
    <script src="http://code.jquery.com/jquery-1.6.4.min.js"> </script>
    <script src="http://code.jquery.com/mobile/latest/jquery.mobile.min.js">    </script>
    <script type="text/javascript" src="HybridLib.js"> </script>
  <script>
        $(document).ready(function() {
            var params = {
                dbname: “sample”,
                tablename: “demo”
            };
            Database.dropTable(function(){alert(“record Deleted!”);}, null, params);
        });
} </script>
</head>
<body>
```

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

8.5. getColumnDetails()

getColumnDetails(*successCallback(columnListArray), errorCallback(), DBDataObject*)

successCallback- テーブルのカラム情報の取得が成功したときに呼ばれる関数
columnListArray - 指定したテーブルのカラムリスト
errorCallback - テーブルのカラム情報の取得が失敗したときに呼ばれる関数
DBDataObject - データベースや表の名前を含むオブジェクトは、削除するテーブルを識別するために使用します

getColumnDetails() メソッドは、DBDataObject で特定したカラム名とテーブル型を取得するのに使用します。
必要なデータは DBDataObject に含まれています。
必要な表データは次の通りです：

DBDataObject メンバー	説明
dbname	テーブルがあるデータベースの名前
tablename	削除するテーブルの名前

例:（サンプル）テーブルのカラム情報を取得します。

```
<!DOCTYPE html>
<html>
<head>
    <link href="http://code.jquery.com/mobile/latest/jquery.mobile.min.css" rel="stylesheet" type="text/css" />
    <script src="http://code.jquery.com/jquery-1.6.4.min.js"> </script>
    <script src="http://code.jquery.com/mobile/latest/jquery.mobile.min.js">    </script>
    <script type="text/javascript" src="HybridLib.js"> </script>
  <script>
        var items;
        $(document).ready(function() {
            var params = {
                dbname: “sample”,
                tablename: “demo”
```

```
                };
                Database.getColumnDetails(
                        function(values) { items = values;},
                        function(){ alert("Error!");},
                        {}
                );
                $('ul#databaseList').html("");
                for (var i = 0; i < items.length; i++) {
                        $('ul#databaseList').append("<li>"+items[i].name+" : "+items[i].type+"</li>");
                }
                $('ul').listview('refresh');
            });
}}    </script>
</head>
<body>
    <div>
        <ul id="databaseList" data-role="listview" data-inset="true"></ul>
    </div>
</body>
</html>
```

8.6. getDatabases()

getDatabases(*successCallback(databaseListArray)*)

successCallback - データベースリストの取得が成功したときに呼ばれる関数
databaseListArray- データベースから返されたデータベースのリスト。配列で返されます。

getDatabases(*successCallback(), errorCallback()*)

successCallback - データベースリストの取得が成功したときに呼ばれる関数
databaseListArray- データベースから返されたデータベースのリスト。配列で返されます。
errorCallback - データベースリストの取得が失敗したときに呼ばれる関数

getDatabases() メソッドは、端末にあるデータベースのリストを取得します。

例：（サンプル）テーブルを削除します。

```
<!DOCTYPE html>
<html>
<head>
    <link href="http://code.jquery.com/mobile/latest/jquery.mobile.min.css" rel="stylesheet" type="text/css" />
    <script src="http://code.jquery.com/jquery-1.6.4.min.js"> </script>
    <script src="http://code.jquery.com/mobile/latest/jquery.mobile.min.js">    </script>
    <script type="text/javascript" src="HybridLib.js"> </script>
  <script>
        var items;
        $(document).ready(function() {
            Database.getDatabases(
                function(values) { items = values;},
                function(){ alert("Error!");
            });
            $('ul#databaseList').html("");
          for (var i = 0; i < items.length; i++) {
              $('ul#databaseList').append("<li>" + items[i] + "</li>");
```

```
                }
                $('ul').listview('refresh');
            });
}}    </script>
</head>
<body>
    <div>
        <ul id="databaseList" data-role="listview" data-inset="true"></ul>
    </div>
</body>
</html>
```

8.7. getTables()

getTables(*successCallback(tableListArray), errorCallback(), DBNameObject*)
successCallback - データベースのテーブルリストの取得が成功したときに呼ばれる関数
tableListArray - データベースから返されたデータベースのリスト。配列で返される
errorCallback - 取得に失敗したデータベースのリストを処理する関数
DBNameObject - データベースの名前を含むオブジェクト

getTables() は、指定したデータベースにあるテーブルリストを取得します。そのデータベース名は、「DBNameObject」 の dbname で指定されます。

例:（サンプル）データベースのテーブルリストを取得します。

```
<!DOCTYPE html>
<html>
<head>
      <link href="http://code.jquery.com/mobile/latest/jquery.mobile.min.css" rel="stylesheet" type="text/css" />
      <script src="http://code.jquery.com/jquery-1.6.4.min.js"> </script>
      <script src="http://code.jquery.com/mobile/latest/jquery.mobile.min.js">     </script>
      <script type="text/javascript" src="HybridLib.js"> </script>
    <script>
          $(document).ready(function() {
                Database.getTables(
                      function(values) { items = values;},
                      function(){ alert("Error!");},
                      {dbname : “sample”}
);
                 $('ul#tableList').html("");
                for (var i = 0; i < items.length; i++) {
                      $('ul#tableList').append("<li>" + items[i] + “</li>");
                }
                $('ul').listview('refresh');
          });
```

```
}</script>
</head>
<body>
	<div>
		<ul id="tableList" data-role="listview" data-inset="true"></ul>
	</div>
</body>
</html>
```

8.8. insertRecord()

insertRecord(*successCallback(resultString), errorCallback(), DBDataObject*)
successCallback - 記録の挿入を処理する関数
resultString - 記録を挿入した結果としての文字列
errorCallback - 失敗した記録の挿入を処理する関数
DBDataObject - 挿入する記録の為のデータベース情報を含むオブジェクト

insertRecord() メソッドは、指定したテーブルにデータを挿入します。 必要なレコードのデータは次の通りです：

DBDataObject Member	説明
Dbname	データベースの名前。
tablename	表の名前。
Values	Value オブジェクトの配列。value オブジェクトのメンバーは次の通りです： name - カラム名を示します。 type - カラムの型を示します。 value - カラムの値を示します。

例:（サンプル）テーブルにレコードを挿入します。

```
<!DOCTYPE html>
<html>
<head>
    <link href="http://code.jquery.com/mobile/latest/jquery.mobile.min.css" rel="stylesheet" type="text/css" />
    <script src="http://code.jquery.com/jquery-1.6.4.min.js"> </script>
    <script src="http://code.jquery.com/mobile/latest/jquery.mobile.min.js">  </script>
    <script type="text/javascript" src="HybridLib.js"> </script>
  <script>
        $(document).ready(function() {
            var value = new Array();
            value[0] = {name: “col_id”, type: “INTEGER”,  value: “2”};
            value[1] = {name: “col_val”, type: “TEXT”, value: “data1”}
```

```
                var params = {
                    dbname: “sample”,
                    tablename: “demo”,
                    values: value
                };
                Database.insertRecord(function(){alert(“record inserted!”);}, null, params);
            });
}}    </script>
</head>
<body>
    <div> </div>
</body>
</html>
```

8.9. queryRecord()

insertRecord(*successCallback(resultRecordArray), errorCallback(), DBDataObject*)
successCallback - テーブルレコードの取得が成功したときに呼ばれる関数
resultRecordArray- クエリから返されたレコードの配列
errorCallback - テーブルレコードの取得が失敗したときに呼ばれる関数
DBDataObject – レコードを検索するためのデータベースの情報を含むオブジェクト

insertRecord()メソッドは、特定のデータベース表にデータを追加します 。
必要なデータは次の通りです：

DBDataObject Member	説明
dbname	データベースの名前.
tablename	テーブルの名前
columns	SQL 文　カラム型文字列
where	(オプション) SQL 文の where 句
groupby	(オプション) SQL 文の groupby 句
orderby	(オプション) SQL 文の orderby 句

例：（サンプル）データベースのテーブルに、col_id: 4 を持つレコードを問合わせます。

```
<!DOCTYPE html>
<html>
<head>
    <link href="http://code.jquery.com/mobile/latest/jquery.mobile.min.css" rel="stylesheet" type="text/css" />
    <script src="http://code.jquery.com/jquery-1.6.4.min.js"> </script>
    <script src="http://code.jquery.com/mobile/latest/jquery.mobile.min.js">    </script>
    <script type="text/javascript" src="HybridLib.js"> </script>
  <script>
        $(document).ready(function() {
            var params = {
                dbname: “sample”,
```

```
                tablename: “demo”,
                columns: “*”,
                where: “col_id = 4”
            };
            Database.queryRecord(
                function(ret) {
                    var str = "";
                    for (var i = 0; i < ret.length; i++) {
                        if (i > 0)
                            str += ",";
                        var count = ret[i].length;
                        for (var itemCnt = 0; itemCnt < count; itemCnt++) {
                            if (itemCnt > 0)
                                str += " | ";
                            }
                            str += ret[i][itemCnt];
                         }
                     }
                     alert("result : " + str);
                }, null, params
            );
        });
}}  </script>
</head>
<body>
    <div> </div>
</body>
</html>
```

8.10. updateRecord()

updateRecord(*successCallback(resultString), errorCallback(), DBDataObject*)
successCallback - レコード更新が成功したときに呼ばれる関数
resultString - 更新したレコードから返された文字列
errorCallback - レコード更新が成功したときに呼ばれる関数
DBDataObject - レコードを挿入するデータベースの情報を含むオブジェクト

updateRecord() メソッドは、既存のレコードの値を更新します。
レコードデータは次の通りです：

DBDataObject Member	説明
dbname	データベースの名前
tablename	テーブルの名前
values	values オブジェクトの配列。Value オブジェクトに含まれるメンバーは次の通り： name - カラム名を示します。 Type - カラム型を示します。 Value – カラム値を示します。
where	（オプション） SQL 文の where 句

例：データベースのサンプル表に、col_id 番号４を持つデモ記録の問合わせをします。

```
<!DOCTYPE html>
<html>
<head>
    <link href="http://code.jquery.com/mobile/latest/jquery.mobile.min.css" rel="stylesheet" type="text/css" />
    <script src="http://code.jquery.com/jquery-1.6.4.min.js"> </script>
    <script src="http://code.jquery.com/mobile/latest/jquery.mobile.min.js">    </script>
    <script type="text/javascript" src="HybridLib.js"> </script>
  <script>
        $(document).ready(function() {
```

```
                var value = new Array();
                value[0] = {name: “col_id”, type: “INTEGER”,    value: “2”};
                value[1] = {name: “col_val”, type: “TEXT”, value: “data1”}
                var params = {
                        dbname: “sample”,
                        tablename: “demo”,
                        values: value,
                        where: “col_id = 4”
                };
                Database.insertRecord(function(){alert(“record inserted!”);}, null, params);
        });
}}    </script>
</head>
<body>
    <div> </div>
</body>
</html>
```

9. DeviceInformation

説明: 端末とソフトウェアの情報にアクセスするオブジェクトを定義します。
メソッド: getDeviceInfo()

9.1. getDeviceInfo()

getDeviceInfo(*successCallback(deviceInfoObject)*)
successCallback – 連絡先の情報の取得に成功したときに呼ばれる関数 *deviceInfoObject* - 端末情報を含んで返されたオブジェクト
getDeviceInfo(*successCallback(deviceInfoArray), errorCallback(errorString)*) *successCallback* -連絡先の情報の取得に成功したときに呼ばれる関数 *deviceInfoObject*- 端末情報を含んで返されたオブジェクト *errorCallback* -連絡先の情報の取得に失敗したときに呼ばれる関数

getDeviceInfo()メソッドは、deviceInfoObject を使って端末情報を取得するのに使用します。deviceInfoObject は下記のデータを含みます：

EventStringValue	説明
name	端末の名前
uuid	端末の UUID
os	端末にインストールされている OS
version	インストールされている OS のバージョン

例：アラートダイアログを表示して、指定したトリガーを表示します。

```
<!DOCTYPE html>
<html>
<head>
    <link href="http://code.jquery.com/mobile/latest/jquery.mobile.min.css" rel="stylesheet" type="text/css" />
    <script src="http://code.jquery.com/jquery-1.6.4.min.js"> </script>
    <script src="http://code.jquery.com/mobile/latest/jquery.mobile.min.js">    </script>
    <script type="text/javascript" src="HybridLib.js"> </script>
  <script>
        $(document).ready(function() {
            DeviceInformation.getDeviceInfo(onSuccess, onError);
        });
        function onSuccess(deviceInfo) {
            $('#devName').text(deviceInfo.name);
            $('#devUUID').text(deviceInfo.uuid);
            $('#sysName').text(deviceInfo.os);
            $('#sysVer').text(deviceInfo.version);
        };
        function onError() {
            alert('onError!');
        };
  </script>
</head>
<body>
    <p>Device Name: <span id="devName">N/A</span></p>
    <p>Device UUID: <span id="devUUID">N/A</span></p>
    <p>System OS: <span id="sysName">N/A</span></p>
    <p>System Version: <span id="sysVer">N/A</span></p>
</body>
</html>
```

10. Events

説明: 端末のエベントトリガーにアクセスするオブジェクトを定義します。
メソッド: registerCallback()

10.1. registerCallback()

registerCallback(*successCallback(eventString)*)

successCallback - 端末のイベントを引き起こす時に呼ばれる関数
eventString - 引き起こされたイベントの文字列の値

registerCallback()メソッドは、端末のイベントが起きた時に呼ばれる関数を定義します。

可能性のある eventString 値は次の通りです：

EventStringValue	説明
onResume	アプリケーションが再開すると呼ばれる
onPause	アプリケーションが一時中断すると呼ばれる
onAppLaunched	アプリケーションが起動すると呼ばれる
onAppExit	アプリケーションが終了すると呼ばれる
onBackPressed	端末の「Back」キーを押下した時に呼ばれる

例：アラートダイアログを表示して、指定したトリガーを表示します。

```
<!DOCTYPE html>
<html>
<head>
    <link href="http://code.jquery.com/mobile/latest/jquery.mobile.min.css" rel="stylesheet" type="text/css" />
    <script src="http://code.jquery.com/jquery-1.6.4.min.js"> </script>
    <script src="http://code.jquery.com/mobile/latest/jquery.mobile.min.js">    </script>
    <script type="text/javascript" src="HybridLib.js"> </script>
  <script>
        $(document).ready(function() {
            Events.registerCallback(function(event) {
                alert("callback, event : " + event);
              });
       });
  </script>
</head>
<body>
</body>
</html>
```

11. Facebook

説明: Facebookに接続して、その機能を使用するためのオブジェクトを定義します。
メソッド: initialize(), logout(), postStatus(), queryGraph()

11.1. initialize()

initialize(*successCallback(), errorCallback(errorString), paramObject*)
successCallback() - Facebookの初期化が成功したときに呼ばれる関数 *errorCallback(errorString)* - Facebookの初期化が失敗したときに呼ばれる関数 *errorString*(オプション)- 返されたエラーの文字列 *paramObject* - FacebookのアプリケーションIDと使用許可の配列を含むオブジェクト

initialize() メソッドは、Facebookに接続を開始します。ただし、ユーザーはログインして、使用許可の受け入れをする必要があります。 ユーザーは、Facebookアプリケーション IDと使用許可のための app_id が必要です。

例: Facebook を開始します

```
<!DOCTYPE html>
<html>
<head>
    <link href="http://code.jquery.com/mobile/latest/jquery.mobile.min.css" rel="stylesheet" type="text/css" />
    <script src="http://code.jquery.com/jquery-1.6.4.min.js"> </script>
    <script src="http://code.jquery.com/mobile/latest/jquery.mobile.min.js">    </script>
    <script type="text/javascript" src="HybridLib.js"> </script>
  <script>
      $(document).ready(function() {
          var fbParams = {
                  app_id : "285508321489436",
                  permissions : {
                      0 : "read_stream" ,
                      1 : "publish_stream",
```

```
                                2 : "offline_access" }
                }
                Facebook.initialize(
                        function(result) {
                                alert(result);
                                $("#main").removeAttr("hidden");
                        },
                        function(error) {
                                alert(error);
                                $("#login").removeAttr("hidden");
                        },
                        fbParams
                );
        });
    </script>
</head>
<body>
</body>
</html>
```

11.2. logout()

logout(*successCallback(), errorCallback(errorString)*)
successCallback() - Facebook のログアウトが成功したときに呼ばれる関数 *errorCallback(errorString)* - Facebook のログアウトが失敗したときに呼ばれる関数 *errorString*(オプション)- 返されたエラーの文字列

logout() メソッドは、initialize()を使って接続した Facebook を停止します。

例: Facebook の開始/停止

```
<!DOCTYPE html>
<html>
<head>
    <link href="http://code.jquery.com/mobile/latest/jquery.mobile.min.css" rel="stylesheet" type="text/css" />
    <script src="http://code.jquery.com/jquery-1.6.4.min.js"> </script>
    <script src="http://code.jquery.com/mobile/latest/jquery.mobile.min.js">    </script>
    <script type="text/javascript" src="HybridLib.js"> </script>
  <script>
      $(document).ready(function() {
          var fbParams = {
                  app_id : "285508321489436",
                  permissions : {
                      0 : "read_stream" ,
                      1 : "publish_stream",
                      2 : "offline_access" }
           }
           Facebook.initialize(
                function(result) {
                    alert(result);
                    Facebook.logout(function(){alert("logout")});
                },
                function(error) {
                    alert(error);
```

```
                    },
                    fbParams
                );
            });
    </script>
</head>
<body>
</body>
</html>
```

11.3. postStatus()

postStatus(*successCallback(), errorCallback(errorString), messageObject*)

successCallback() - Facebook で近況の投稿が成功したときに呼ばれる関数
errorCallback(errorString) - Facebook で近況の投稿が失敗したときに呼ばれる関数
errorString (オプション)- 返されたエラーの文字列
messageObject - Facebook の近況に書き込む内容を含むオブジェクト

postStatus()メソッドは 、Facebook にログインしているユーザーの近況に messageObject にある内容を書き込みます。

例: Facebook を開始して近況に投稿します。

```
<!DOCTYPE html>
<html>
<head>
    <link href="http://code.jquery.com/mobile/latest/jquery.mobile.min.css" rel="stylesheet" type="text/css" />
    <script src="http://code.jquery.com/jquery-1.6.4.min.js"> </script>
    <script src="http://code.jquery.com/mobile/latest/jquery.mobile.min.js">    </script>
    <script type="text/javascript" src="HybridLib.js"> </script>
  <script>
      $(document).ready(function() {
          var fbParams = {
                  app_id : "285508321489436",
                  permissions : {
                      0 : "read_stream" ,
                      1 : "publish_stream",
                      2 : "offline_access" }
           }
           Facebook.initialize(
               function(result) {
                  alert(result);
                  Facebook.postStatus(
                      function() {
```

```
                                    alert(“success”);
                                },
                                function(ngArgs) {
                                    alert(ngArgs);
                                },
                                {message : “hello world”}
                            );
                        },
                        function(error) {
                            alert(error);
                        },
                        fbParams
                    );
            });
        </script>
</head>
<body>
</body>
</html>
```

11.4. queryGraph()

queryGraph(*successCallback(JSONString), errorCallback(errorString), pathObject*)

successCallback() - Facebook Graph API の使用が成功したときに呼ばれる関数
JSONString- Facebook から返された JSON の文字列
errorCallback(errorString) - Facebook Graph API の使用が失敗したときに呼ばれる関数
errorString (オプション) - 返されたエラーの文字列
pathObject - Facebook の Graph API パスを含むオブジェクト

queryGraph()メソッドは 、Facebook からデータを取得するために Facebook Graph API を使用します。 ユーザーは、pathObject のパスメンバーを使って Graph API のパスを提供する必要があります。

例: Facebook を開始してユーザーの Wall Post を取得します。

```
<!DOCTYPE html>
<html>
<head>
    <link href="http://code.jquery.com/mobile/latest/jquery.mobile.min.css" rel="stylesheet" type="text/css" />
    <script src="http://code.jquery.com/jquery-1.6.4.min.js"> </script>
    <script src="http://code.jquery.com/mobile/latest/jquery.mobile.min.js">    </script>
    <script type="text/javascript" src="HybridLib.js"> </script>
  <script>
      $(document).ready(function() {
          var fbParams = {
                  app_id : "285508321489436",
                  permissions : {
                      0 : "read_stream" ,
                      1 : "publish_stream",
                      2 : "offline_access" }
           }
           Facebook.initialize(
                function(result) {
                    alert(result);
```

```
                        Facebook.queryGraph (
                            function(okArgs) {
                                alert(okArgs);
                            },
                            function(ngArgs) {
                                alert(ngArgs);
                            },
                            {path : "me/feed"}
                        );
                    },
                    function(error) {
                        alert(error);
                    },
                    fbParams
                );
            });
        </script>
    </head>
    <body>
    </body>
    </html>
```

12. FileAccess

説明: 端末の SD カードにアクセスするオブジェクトを定義します。

メソッド: copyFile(), createDirectory(), deleteFile(), fileExists(), isDirectory(), listPathContents(), moveFile(), readFile(), writeToFile(), getImage()

12.1. copyFile()

copyFile(*successCallback(), errorCallback(errorString), pathObject*)

successCallback() - ファイルのコピーが成功したときに呼ばれる関数
errorCallback(errorString) - ファイルのコピーが失敗したときに呼ばれる関数
errorString (オプション) - 返されたエラーの文字列
pathObject - ファイルパスを含むオブジェクト

copyFile()メソッドは 、ファイルをコピーして新しくファイルを作成すします。
pathObject は、 fromPath(ソースパス)、toPath(移動先パス)を含みます。

例: Temp.txt を to Temp2.txt へコピーします。

```
<!DOCTYPE html>
<html>
<head>
    <link href="http://code.jquery.com/mobile/latest/jquery.mobile.min.css" rel="stylesheet" type="text/css" />
    <script src="http://code.jquery.com/jquery-1.6.4.min.js"> </script>
    <script src="http://code.jquery.com/mobile/latest/jquery.mobile.min.js">    </script>
    <script type="text/javascript" src="HybridLib.js"> </script>
  <script>
      $(document).ready(function() {
          FileAccess.copyFile(
              function() {
                  alert("success");
               },
               function(msg) {
                  alert(msg);
```

```
                    },
                    {"fromPath" : “Temp.txt”, "toPath" : “Temp2.txt”}
                );
        });
    </script>
</head>
<body>
</body>
</html>
```

12.2. createDirectory()

createDirectory(*successCallback(), errorCallback(errorString), pathObject*)

successCallback() - ディレクトリの作成が成功したときに呼ばれる関数
errorCallback(errorString) - ディレクトリの作成が失敗したときに呼ばれる関数
errorString (オプション) - 返されたエラーの文字列
pathObject - ファイルパスを含むオブジェクト

createDirectory()メソッドは 、ディレクトリを新規作成します。
pathObject は、新しいディレクトリのパスを保持する filePath を含みます。

例:「Temp」ディレクトリを作成します

```
<!DOCTYPE html>
<html>
<head>
    <link href="http://code.jquery.com/mobile/latest/jquery.mobile.min.css" rel="stylesheet" type="text/css" />
    <script src="http://code.jquery.com/jquery-1.6.4.min.js"> </script>
    <script src="http://code.jquery.com/mobile/latest/jquery.mobile.min.js">    </script>
    <script type="text/javascript" src="HybridLib.js"> </script>
  <script>
      $(document).ready(function() {
          FileAccess.createDirectory(
              function() {
                  alert(“Success”);
               },
               function(msg) {
                  alert(msg);
               },
               {"filePath" : “Temp”}
           );
      });
  </script>
</head>
```

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

12.3. deleteFile()

deleteFile(*successCallback(), errorCallback(errorString), pathObject*)

successCallback() - ファイルの削除が成功したときに呼ばれる関数
errorCallback(errorString) - ファイルの削除が失敗したときに呼ばれる関数
errorString (オプション) - 返されたエラーの文字列
pathObject - ファイルパスを含むオブジェクト

deleteFile()メソッドは 、ファイルを削除します。
pathObject は、削除するファイルのパスを保持する filePath を含みます。

例:「Temp.txt」を削除します。

```
<!DOCTYPE html>
<html>
<head>
    <link href="http://code.jquery.com/mobile/latest/jquery.mobile.min.css" rel="stylesheet" type="text/css" />
    <script src="http://code.jquery.com/jquery-1.6.4.min.js"> </script>
    <script src="http://code.jquery.com/mobile/latest/jquery.mobile.min.js">    </script>
    <script type="text/javascript" src="HybridLib.js"> </script>
  <script>
      $(document).ready(function() {
          FileAccess.deleteFile(
              function() {
                  alert(“success”);
               },
               function(msg) {
                  alert(msg);
               },
               {"filePath" : “Temp.txt”}
            );
      });
  </script>
</head>
<body>
```

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

12.4. fileExists()

fileExists(*successCallback(isFileExistBoolean), errorCallback(errorString), pathObject*)

successCallback() - ファイルの確認が成功したときに呼ばれる関数
isFileExistBoolean- この関数によって返されたブール値
errorCallback(errorString) - ファイルの確認が失敗したときに呼ばれる関数
errorString (オプション) - 返されたエラーの文字列
pathObject - ファイルパスを含むオブジェクト

fileExists() メソッドは、ファイルが存在するかどうか確認します。
pathObject は､確認するファイルのパスを保持するために filePath を含みます。

例:「Temp.txt 」が存在するかどうか確認します。

```
<!DOCTYPE html>
<html>
<head>
    <link href="http://code.jquery.com/mobile/latest/jquery.mobile.min.css" rel="stylesheet" type="text/css" />
    <script src="http://code.jquery.com/jquery-1.6.4.min.js"> </script>
    <script src="http://code.jquery.com/mobile/latest/jquery.mobile.min.js">    </script>
    <script type="text/javascript" src="HybridLib.js"> </script>
  <script>
      $(document).ready(function() {
          FileAccess.fileExists(
              function(msg) {
                  alert("is File Exist: "+msg);
              },
              function(msg) {
                  alert(msg);
              },
              {"filePath" : "Temp.txt"}
          );
      });
  </script>
```

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

12.5. isDirectory()

isDirectory(*successCallback(isDirectoryBoolean), errorCallback(errorString), pathObject*)

successCallback()- 指示しているパスがディレクトリかどうかの確認が成功したときに呼ばれる関数
isDirectoryBoolean- この関数によって返されたブール値。
errorCallback(errorString) – 指示しているパスがディレクトリかどうかの確認が失敗したときに呼ばれる関数
errorString (オプション)- 返されたエラーの文字列
pathObject- ファイルパスを含むオブジェクト

isDirectory()メソッドは、ファイルがディレクトリかどうか確認します。
pathObject は､確認するファイルのパスを保持するために filePath を含みます。

例:「Temp」がディレクトリかどうか確認します。

```
<!DOCTYPE html>
<html>
<head>
    <link href="http://code.jquery.com/mobile/latest/jquery.mobile.min.css" rel="stylesheet" type="text/css" />
    <script src="http://code.jquery.com/jquery-1.6.4.min.js"> </script>
    <script src="http://code.jquery.com/mobile/latest/jquery.mobile.min.js">    </script>
    <script type="text/javascript" src="HybridLib.js"> </script>
  <script>
      $(document).ready(function() {
          FileAccess.isDirectory(
              function(msg) {
                  alert(""is a directory: "+msg);
               },
               function(msg) {
                  alert(msg);
               },
               {"filePath" : "Temp"}
            );
```

```
        });
    </script>
</head>
<body>
</body>
</html>
```

12.6. listPathContents()

listPathContents(*successCallback(fileListObject)*, *errorCallback(errorString)*, *pathObject*)

successCallback() - ファイルリストの取得が成功したときに呼ばれる関数
fileListObject - ファイルリストを含むオブジェクト
errorCallback(errorString) - ファイルリストの取得が成功したときに呼ばれる関数
errorString (オプション)- 返されたエラーの文字列
pathObject - ファイルのリスト含むオブジェクト

listPathContents()メソッドは 、ファイルリストを取得します。
pathObject は､ファイルリストを取得するのに必要なディレクトリのパスを保持するために filePath を含みます。

例:「Temp」ディレクトリでファイルのリストを表示します。

```
<!DOCTYPE html>
<html>
<head>
    <link href="http://code.jquery.com/mobile/latest/jquery.mobile.min.css" rel="stylesheet" type="text/css" />
    <script src="http://code.jquery.com/jquery-1.6.4.min.js"> </script>
    <script src="http://code.jquery.com/mobile/latest/jquery.mobile.min.js">    </script>
    <script type="text/javascript" src="HybridLib.js"> </script>
  <script>
      $(document).ready(function() {
          FileAccess.listPathContents(
              function(msg) {
                  var jsonObj = $.parseJSON(msg);
                  var txtHtml = "<ul>";
                  $.each(jsonObj, function(key, val) {
                      txtHtml = txtHtml + '<li id="' + key + '">' + val + '</li>';
                  });
                  txtHtml = txtHtml + "</ul>";
                  $("#txtDirContents").html(txtHtml);
```

```
                },
                function(msg) {
                        alert(msg);
                },
                {'filePath': “Temp”}
            );
        });
    </script>
</head>
<body>
      <div id="txtDirContents"></div>
</body>
</html>
```

12.7. moveFile()

moveFile(*successCallback(), errorCallback(errorString), pathObject*)

successCallback() - ファイル移動が成功したときに呼ばれる関数
errorCallback(errorString) - ファイル移動が失敗したときに呼ばれる関数
errorString (オプション) - 返されたエラーの文字列
pathObject - ファイルパスを含むオブジェクト

moveFile()メソッドは、指定した場所へファイルを移動します。
pathObject は　fromPath(ソースパス)と toPath(移動先のパス)を含みます。

例:「Temp.txt」を「Temp2.txt」へ移動します。

```
<!DOCTYPE html>
<html>
<head>
    <link href="http://code.jquery.com/mobile/latest/jquery.mobile.min.css" rel="stylesheet" type="text/css" />
    <script src="http://code.jquery.com/jquery-1.6.4.min.js"> </script>
    <script src="http://code.jquery.com/mobile/latest/jquery.mobile.min.js">    </script>
    <script type="text/javascript" src="HybridLib.js"> </script>
  <script>
      $(document).ready(function() {
          FileAccess.moveFile(
              function() {
                  alert(“success”);
               },
               function(msg) {
                  alert(msg);
               },
               {"fromPath" : “Temp.txt”, "toPath" : “Temp2.txt”}
           );
      });
  </script>
</head>
<body>
```

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

12.8. readFile()

readFile(*successCallback(fileString), errorCallback(errorString), pathObject*)

successCallback() - ファイルの読み込みが成功したときに呼ばれる関数
fileString- 返されたファイルの内容
errorCallback(errorString) - ファイルの読み込みが失敗したときに呼ばれる関数
errorString (オプション)- 返されたエラーの文字列
pathObject - ファイルパスを含むオブジェクト

readFile() メソッドは、ファイルの内容を文字列として読み込みます。
pathObject は、読み込むファイルのパスを保持するために、filePath を含みます。

例:「Temp.txt」の内容を読み込みます。

```
<!DOCTYPE html>
<html>
<head>
    <link href="http://code.jquery.com/mobile/latest/jquery.mobile.min.css" rel="stylesheet" type="text/css" />
    <script src="http://code.jquery.com/jquery-1.6.4.min.js"> </script>
    <script src="http://code.jquery.com/mobile/latest/jquery.mobile.min.js">    </script>
    <script type="text/javascript" src="HybridLib.js"> </script>
  <script>
      $(document).ready(function() {
          FileAccess.readFile(
              function(msg) {
                  alert(msg);
              },
              function(msg) {
                  alert(msg);
              },
              {"filePath" : “Temp.txt”}
          );
      });
```

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

12.9. writeFile()

writeFile(*successCallback(), errorCallback(errorString), pathObject*)

successCallback() - ファイルの書き込みが成功したときに呼ばれる関数
errorCallback(errorString) - ファイルの書き込みが失敗したときに呼ばれる関数
errorString (オプション) - 返されたファイルの文字列の内容
pathObject - ファイルパス含むオブジェクト

writeFile()メソッドは、指定したファイルの中に書き込みます。
pathObject は、filePath (書き込むためのファイルのパス)と、fileContent(ファイルに書き込む文字列)を含みます。
writeFile() メソッドは、ファイルがまた無い場合は新規作成して、存在する場合はファイルを上書きします。

例:「Temp.txt」の中に「Hello World!」を書き込みます。

```
<!DOCTYPE html>
<html>
<head>
    <link href="http://code.jquery.com/mobile/latest/jquery.mobile.min.css" rel="stylesheet" type="text/css" />
    <script src="http://code.jquery.com/jquery-1.6.4.min.js"> </script>
    <script src="http://code.jquery.com/mobile/latest/jquery.mobile.min.js">    </script>
    <script type="text/javascript" src="HybridLib.js"> </script>
  <script>
      $(document).ready(function() {
          FileAccess.writeFile(
              function() {
                  alert(“success”);
              },
              function(msg) {
                  alert(msg);
              },
              {"filePath" : “Temp.txt”, “fileContents” : “Hello World!”}
          );
```

```
        });
    </script>
</head>
<body>
</body>
</html>
```

12.10. getImage()

getImage (*successCallback(base64imageString), errorCallback(errorString), pathObject*)
successCallback(base64imageString) - 画像の取得が成功したとき呼ばれる関数。関数は画像にアクセスするためのオブジェクトパラメータが必要です。 *errorCallback(errorString)* -画像の取得が失敗したとき呼ばれる関数。 *errorString* (オプション) - 返されたエラーの文字列 *pathObject* - 画像ファイルのパスを含むオブジェクト

getImage()メソッドは、端末の指定した場所から画像を取得するために使用します。 またこのメソッドは、base64imageString によって端末から画像を取得します。画像は、base64 形式の文字列で返されます。

例: 端末の特定の場所からアクセスして、画像を表示します。

```
<!DOCTYPE html>
<html>
<head>
      <link href="http://code.jquery.com/mobile/latest/jquery.mobile.min.css" rel="stylesheet" type="text/css" />
      <script src="http://code.jquery.com/jquery-1.6.4.min.js"> </script>
      <script src="http://code.jquery.com/mobile/latest/jquery.mobile.min.js">     </script>
      <script type="text/javascript" src="HybridLib.js"> </script>
    <script>
            $(document).ready(function() {
               $("#btnCamera").click(function() {
                              var params = new CameraParams();
                              Camera.captureImage(captureSuccess, captureError);
               });
});

function captureSuccess(strImage) {
           alert (“Image capture success!!!”);
             var capture_image = document.getElementById(' imgSample’);
```

```
            var fromPath = new String();
            var params = {fromPath:imagePath};
            FileAccess.getImage(function(image) {
                $("#capture_img").attr("src", "" + image);
            }, captureError, params);
                strImage =    imagePath;}

function captureError(strError) {
                alert(strError);
}    </script>
</head>
<body>
      <img id="imgSample"></img>
      <button id="btnCamera">Open Device Camera</button>
</body>
</html>
```

13. Geolocation

説明: 端末のＧＰＳにアクセスするオブジェクトを定義します。
メソッド: getLatitude(), getLongitude(), startUpdatingLocation(), stopUpdatingLocation()

13.1. getLatitude()

getLatitude()

getLatitude()メソッドは、端末の現在位置の緯度を取得します。この関数は、startUpdatingLocation が呼ばれたときに使用されます

例: 現在の緯度を表示します。

```
<!DOCTYPE html>
<html>
<head>
    <link href="http://code.jquery.com/mobile/latest/jquery.mobile.min.css" rel="stylesheet" type="text/css" />
    <script src="http://code.jquery.com/jquery-1.6.4.min.js"> </script>
    <script src="http://code.jquery.com/mobile/latest/jquery.mobile.min.js">    </script>
    <script type="text/javascript" src="HybridLib.js"> </script>
  <script>
      $(document).ready(function() {
          Geolocation.startUpdatingLocation();
          $('#latDisplay').text("latitude: "+ Geolocation.getLatitude());
      });
  </script>
</head>
<body><div id="latDisplay"></div></body>
</html>
```

13.2. getLongitude()

getLongitude()

getLongitude()メソッドは、端末の現在位置の経度を取得します。この関数は、startUpdatingLocation が呼ばれたときに使用されます。

例: 現在の経度を表示します。

```
<!DOCTYPE html>
<html>
<head>
    <link href="http://code.jquery.com/mobile/latest/jquery.mobile.min.css" rel="stylesheet" type="text/css" />
    <script src="http://code.jquery.com/jquery-1.6.4.min.js"> </script>
    <script src="http://code.jquery.com/mobile/latest/jquery.mobile.min.js">    </script>
    <script type="text/javascript" src="HybridLib.js"> </script>
  <script>
        $(document).ready(function() {
            Geolocation.startUpdatingLocation();
            $('#longDisplay').text("longitude: "+ Geolocation.getLongitude());
        });
  </script>
</head>
<body><div id="longDisplay"></div></body>
</html>
```

13.3. startUpdatingLocation()

startUpdatingLocation()

startUpdatingLocation() メソッドは、端末の位置情報を取得する関数を開始します。

例: 現在の緯度を表示します。

```
<!DOCTYPE html>
<html>
<head>
    <link href="http://code.jquery.com/mobile/latest/jquery.mobile.min.css" rel="stylesheet" type="text/css" />
    <script src="http://code.jquery.com/jquery-1.6.4.min.js"> </script>
    <script src="http://code.jquery.com/mobile/latest/jquery.mobile.min.js">    </script>
    <script type="text/javascript" src="HybridLib.js"> </script>
  <script>
        $(document).ready(function() {
            Geolocation.startUpdatingLocation();
            $('#latDisplay').text("latitude: "+ Geolocation.getLatitude());
        });
  </script>
</head>
<body><div id="latDisplay"></div></body>
</html>
```

13.4. stopUpdatingLocation()

stopUpdatingLocation()

stopUpdatingLocation()メソッドは、端末の位置情報を取得する関数を停止します。

例:　現在の緯度を表示します。

```
<!DOCTYPE html>
<html>
<head>
    <link href="http://code.jquery.com/mobile/latest/jquery.mobile.min.css" rel="stylesheet" type="text/css" />
    <script src="http://code.jquery.com/jquery-1.6.4.min.js"> </script>
    <script src="http://code.jquery.com/mobile/latest/jquery.mobile.min.js">    </script>
    <script type="text/javascript" src="HybridLib.js"> </script>
  <script>
      $(document).ready(function() {
          Geolocation.startUpdatingLocation();
          $('#latDisplay').text("latitude: "+ Geolocation.getLatitude());
          Geolocation.stopUpdatingLocation();
      });
  </script>
</head>
<body><div id="latDisplay"></div></body>
</html>
```

14. MediaPlayer

説明: 端末の Media Player にアクセスするオブジェクトを定義します。
メソッド: playAudio(), pauseAudio(), selectAudio()

14.1. playAudio()

playAudio(*successCallback(isPlayingBoolean)*)
successCallback -選択した音声ファイルの再生が成功したときに呼ばれる関数 isPlayingBoolean- 音声ファイルが再生しているかどうかを示すブール値。
playAudio(*successCallback(isPlayingBoolean), errorCallback(errorString)*)
successCallback -選択した音声ファイルの再生が成功したときに呼ばれる関数 isPlayingBoolean- 音声ファイルが再生しているかどうかを示すブール値。 *errorCallback* -選択した音声ファイルの再生が失敗したときに呼ばれる関数 *errorString* (オプション) - 返されたエラーの文字列

playAudio()メソッドは selectAudio()メソッドを使って、選択した音声ファイルを再生します。

例: ファイルを再生します。

```
<!DOCTYPE html>
<html>
<head>
    <link href="http://code.jquery.com/mobile/latest/jquery.mobile.min.css" rel="stylesheet" type="text/css" />
    <script src="http://code.jquery.com/jquery-1.6.4.min.js"> </script>
    <script src="http://code.jquery.com/mobile/latest/jquery.mobile.min.js">    </script>
    <script type="text/javascript" src="HybridLib.js"> </script>
  <script>
      $(document).ready(function() {
          MediaPlayer.playAudio(function(isPlay) {
              alert("Audio is playing? "+isPlay);
          },
```

```
                function() {
                        alert("Error");
                });
        });
    </script>
</head>
<body>
</body>
</html>
```

14.2. pauseAudio()

pauseAudio(*successCallback(isPlayingBoolean)*)
successCallback - 音声ファイルの停止が成功したときに呼ばれる関数 isPlayingBoolean - 音声ファイルが再生しているかどうかを示すブール値。
pauseAudio(*successCallback(isPlayingBoolean), errorCallback(errorString)*)
successCallback -音声ファイルの停止が成功したときに呼ばれる関数 isPlayingBoolean- 音声ファイルが再生しているかどうかを示すブール値。 *errorCallback* -音声ファイルの停止が失敗したときに呼ばれる関数 *errorString* (オプション) - 返されたエラーの文字列

pauseAudio() メソッドは、playAudio() メソッドによって再生された音声ファイルを一時停止します。

例: ファイルを一時停止します。

```
<!DOCTYPE html>
<html>
<head>
    <link href="http://code.jquery.com/mobile/latest/jquery.mobile.min.css" rel="stylesheet" type="text/css" />
    <script src="http://code.jquery.com/jquery-1.6.4.min.js"> </script>
    <script src="http://code.jquery.com/mobile/latest/jquery.mobile.min.js">    </script>
    <script type="text/javascript" src="HybridLib.js"> </script>
  <script>
      $(document).ready(function() {
          MediaPlayer.pauseAudio(function(isPlay) {
              alert(“Audio is playing? ”+isPlay);
          },
          function() {
              alert("Error");
          });
      });
  </script>
```

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

14.3. selectAudio()

selectAudio()

selectAudio()メソッドは、再生する音声ファイルを選択するために、端末のファイルエクスプローラーを開始します。

例: 音声ファイル選択の為に、File Picker を開きます。

```
<!DOCTYPE html>
<html>
<head>
    <link href="http://code.jquery.com/mobile/latest/jquery.mobile.min.css" rel="stylesheet" type="text/css" />
    <script src="http://code.jquery.com/jquery-1.6.4.min.js"> </script>
    <script src="http://code.jquery.com/mobile/latest/jquery.mobile.min.js">    </script>
    <script type="text/javascript" src="HybridLib.js"> </script>
  <script>
      $(document).ready(function() {
          MediaPlayer.selectAudio();
      });
  </script>
</head>
<body>
</body>
</html>
```

15. Twitter

説明: Twitter に接続して、その機能を使うためのオブジェクトを定義します。

メソッド: deleteTweet(), getHomeTimeline(), getPublicTimeline(), getUserTimeline(), postTweet()

15.1. deleteTweet()

deleteTweet(*successCallback(), errorCallback(errorString), tweetIdObject*)

successCallback - Tweet の削除が成功したときに呼ばれる関数
errorCallback - Tweet の削除が失敗したときに呼ばれる関数
errorString (オプション) - 返されたエラーの文字列
tweetIdObject - 削除する Tweet ID を含むオブジェクト

deleteTweet()メソッドは、tweetIdObject の ID に保存されている Tweet ID を削除します。

例: Tweet ID を削除します。

```
<!DOCTYPE html>
<html>
<head>
    <link href="http://code.jquery.com/mobile/latest/jquery.mobile.min.css" rel="stylesheet" type="text/css" />
    <script src="http://code.jquery.com/jquery-1.6.4.min.js"> </script>
    <script src="http://code.jquery.com/mobile/latest/jquery.mobile.min.js">  </script>
    <script type="text/javascript" src="HybridLib.js"> </script>
  <script>
      $(document).ready(function() {
          Twitter.deleteTweet(
              function() {
                  alert("Tweet deleted.");
              },
              function(error) {
                  alert("Delete error: " + error);
```

```
                    },
                    {tweetId : 123}
                );
            });
    </script>
</head>
<body>
</body>
</html>
```

15.2. getHomeTimeline()

getHomeTimeline(*successCallback(timelineObject), errorCallback(errorString)*)

successCallback - ホームタイムラインの取得が成功したときに呼ばれる関数
timelineObject - 取得したタイムラインを含むオブジェクト
errorCallback - ホームタイムラインの取得が失敗したときに呼ばれる関数
errorString (オプション) - 返されたエラーの文字列

getHomeTimeline()メソッドは、twitter からホームタイムラインを取得して、timelineObject を返します。
timelineObject は、Twitter JSON から返されたデータの twitter オブジェクトの配列を含み、twitter で設定したものと同じメンバーの名称を持ちます。

例: ホームタイムラインを表示します。

```
<!DOCTYPE html>
<html>
<head>
    <link href="http://code.jquery.com/mobile/latest/jquery.mobile.min.css" rel="stylesheet" type="text/css" />
    <script src="http://code.jquery.com/jquery-1.6.4.min.js"> </script>
    <script src="http://code.jquery.com/mobile/latest/jquery.mobile.min.js">    </script>
    <script type="text/javascript" src="HybridLib.js"> </script>
  <script>
      $(document).ready(function() {
                Twitter.getHomeTimeline(processJSONData, function(err)alert("error: ”+err));
      });
       function processJSONData(jsonResponse) {
            var htmlTxt = "";
            $.each(jsonResponse, function(index, value) {
                htmlTxt += "<li id='" + jsonResponse[index].id_str + "'>";
                htmlTxt += '<a href="javascript:deleteTweet(¥'' + jsonResponse[index].id_str + '¥');">';
                htmlTxt += "<img src='" + jsonResponse[index].user.profile_image_url +"'/>";
                htmlTxt += "<h3>" + jsonResponse[index].user.name +"</h3>";
                htmlTxt += "<p>" + jsonResponse[index].text + "</p>";
                htmlTxt += "</a>";
```

```
                htmlTxt += "</li>";
            });
            $("#timeline").html(htmlTxt);
            $("#timeline").listview('refresh');
        }
    </script>
</head>
<body>
    <ul id="timeline" data-role='listview'></ul>
</body>
</html>
```

15.3. getPublicTimeline()

getPublicTimeline(*successCallback(timelineObject), errorCallback(errorString)*)

successCallback- パブリックタイムラインの取得が成功したときに呼ばれる関数
timelineObject- 取得したタイムラインを含むオブジェクト
errorCallback -パブリックタイムラインの取得が成功したときに呼ばれる関数
errorString (オプション) - 返されたエラーの文字列

getPublicTimeline()メソッドは、twitter からパブリックタイムラインを取得して、timelineObject を返します。
timelineObject は、Twitter JSON から返されたデータの twitter オブジェクトの配列を含み、twitter で設定したものと同じメンバー名を持ちます。

例: パブリックタイムラインを表示します。

```
<!DOCTYPE html>
<html>
<head>
    <link href="http://code.jquery.com/mobile/latest/jquery.mobile.min.css" rel="stylesheet" type="text/css" />
    <script src="http://code.jquery.com/jquery-1.6.4.min.js"> </script>
    <script src="http://code.jquery.com/mobile/latest/jquery.mobile.min.js">    </script>
    <script type="text/javascript" src="HybridLib.js"> </script>
  <script>
      $(document).ready(function() {
                Twitter.getPublicTimeline(processJSONData, function(err)alert("error: ”+err));
      });
      function processJSONData(jsonResponse) {
          var htmlTxt = "";
          $.each(jsonResponse, function(index, value) {
              htmlTxt += "<li id='" + jsonResponse[index].id_str + "'>";
              htmlTxt += '<a href="javascript:deleteTweet(¥'' + jsonResponse[index].id_str + '¥');">';
              htmlTxt += "<img src='" + jsonResponse[index].user.profile_image_url +"'/>";
              htmlTxt += "<h3>" + jsonResponse[index].user.name +"</h3>";
```

```
                htmlTxt += "<p>" + jsonResponse[index].text + "</p>";
                htmlTxt += "</a>";
                htmlTxt += "</li>";
            });
            $("#timeline").html(htmlTxt);
            $("#timeline").listview('refresh');
        }
    </script>
</head>
<body>
    <ul id="timeline" data-role='listview'></ul>
</body>
</html>
```

15.4. getUserTimeline()

getUserTimeline(*successCallback(timelineObject), errorCallback(errorString)*)

successCallback - ユーザータイムラインの取得が成功したときに呼ばれる関数
timelineObject- 取得したタイムラインを含むオブジェクト
errorCallback - ユーザータイムラインの取得が失敗したときに呼ばれる関数
errorString (オプション)- 返されたエラーの文字列

getUserTimeline()メソッドはツイッターからユーザータイムラインを取得します。 timelineObject は、Twitter JSON から反応されたデータを含むオブジェクトの配列と、ツイッターで設定している同じメンバーの名称を持ちます。

例: ユーザータイムラインを表示します。

```
<!DOCTYPE html>
<html>
<head>
    <link href="http://code.jquery.com/mobile/latest/jquery.mobile.min.css" rel="stylesheet" type="text/css" />
    <script src="http://code.jquery.com/jquery-1.6.4.min.js"> </script>
    <script src="http://code.jquery.com/mobile/latest/jquery.mobile.min.js">    </script>
    <script type="text/javascript" src="HybridLib.js"> </script>
  <script>
      $(document).ready(function() {
                Twitter.getUserTimeline(processJSONData, function(err)alert("error: ”+err));
      });
       function processJSONData(jsonResponse) {
            var htmlTxt = "";
            $.each(jsonResponse, function(index, value) {
                htmlTxt += "<li id='" + jsonResponse[index].id_str + "'>";
                htmlTxt += '<a href="javascript:deleteTweet(¥'' + jsonResponse[index].id_str + '¥');">';
                htmlTxt += "<img src='" + jsonResponse[index].user.profile_image_url +"'/>";
                htmlTxt += "<h3>" + jsonResponse[index].user.name +"</h3>";
                htmlTxt += "<p>" + jsonResponse[index].text + "</p>";
                htmlTxt += "</a>";
```

```
                htmlTxt += "</li>";
            });
            $("#timeline").html(htmlTxt);
            $("#timeline").listview('refresh');
        }
    </script>
</head>
<body>
    <ul id="timeline" data-role='listview'></ul>
</body>
</html>
```

15.5. postTweet()

postTweet(*successCallback(), errorCallback(errorString), tweetObject*)

successCallback - Tweet 投稿が成功したときに呼ばれる関数
errorCallback - Tweet 投稿が失敗したときに呼ばれる関数
errorString (オプション) - 返されたエラーの文字列
tweetObject - 投稿する Tweet を含むオブジェクト

postTweet()メソッドは、twitter に投稿するのに使用されます。tweetObject を使って投稿される Tweet は送信され、tweetObject の tweet メンバーに保存されます。

例：Tweet を投稿します。

```
<!DOCTYPE html>
<html>
<head>
    <link href="http://code.jquery.com/mobile/latest/jquery.mobile.min.css" rel="stylesheet" type="text/css" />
    <script src="http://code.jquery.com/jquery-1.6.4.min.js"> </script>
    <script src="http://code.jquery.com/mobile/latest/jquery.mobile.min.js">    </script>
    <script type="text/javascript" src="HybridLib.js"> </script>
  <script>
      $(document).ready(function() {
          Twitter.postTweet(
              function() {
                  alert("Tweet posted.");
              },
              function(error) {
                  alert("Post error: " + error);
              },
              {tweet : "hello tweet"}
          );
      });
  </script>
</head>
```

```
<body>

</body>

</html>
```