

-Smart WeMo-  
Android 取扱説明書

---



ユビキタス・テクノロジーズ株式会社

**Document Revision History**

Version	Date	Details	Comments
2.0.0	04/27/2012	New creation	Jasmin Manalo

## 1. 概要

このドキュメントは、「Smart WeMo」を使った Android 端末向けアプリケーション開発の手順を説明しています。

## 2. 開発環境

「Smart WeMo」を使った Android 端末向けアプリケーションの開発環境には、Eclipse のインストールが必要です。

基本環境::

1. Eclipse For Java Developers 3.5 (Galileo) 以上
2. Android 2.3 (又はそれ以上) SDK
3. Android Developer Tools (ADT) Plugin for Eclipse

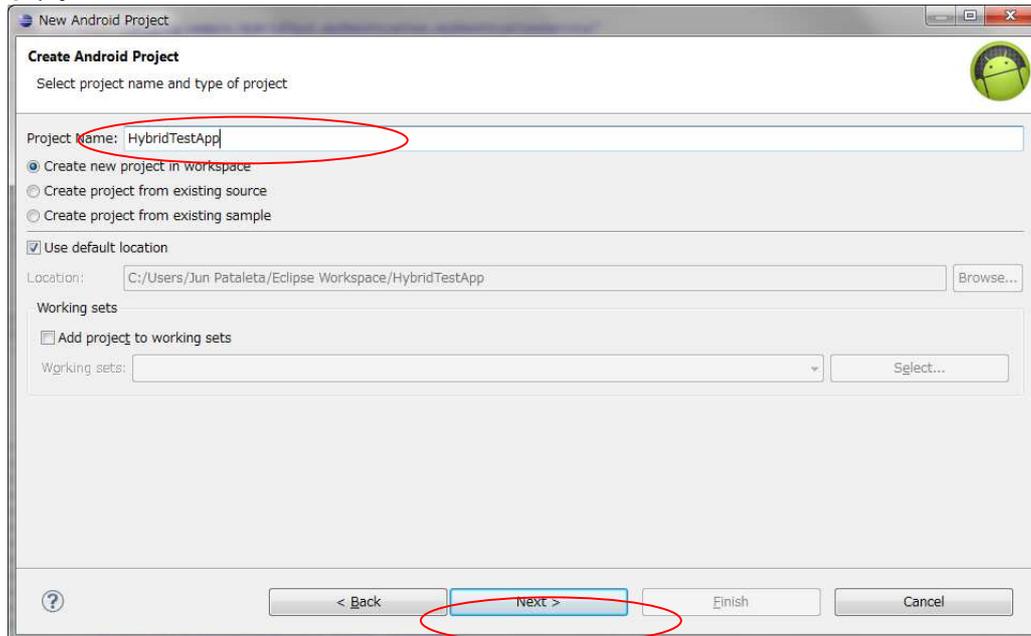
Android アプリケーションの開発に必要なツールは下記の URL に含まれています。

ダウンロード URL <http://developer.android.com/sdk/index.html>

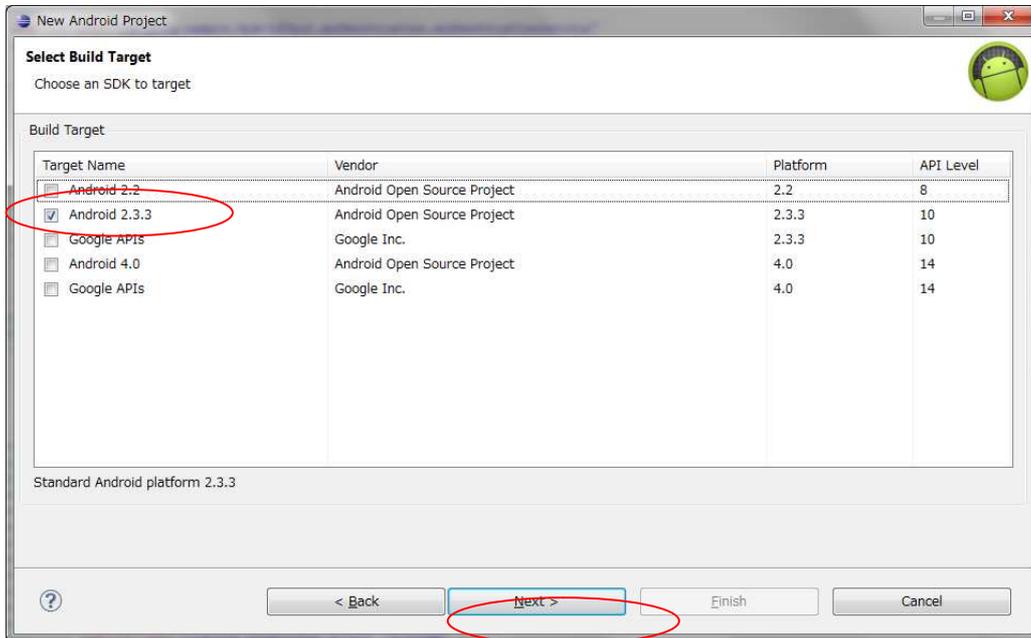
### 3. Android 向け Smart WeMo アプリケーションの開発

#### 1. 新規 Android プロジェクトの作成

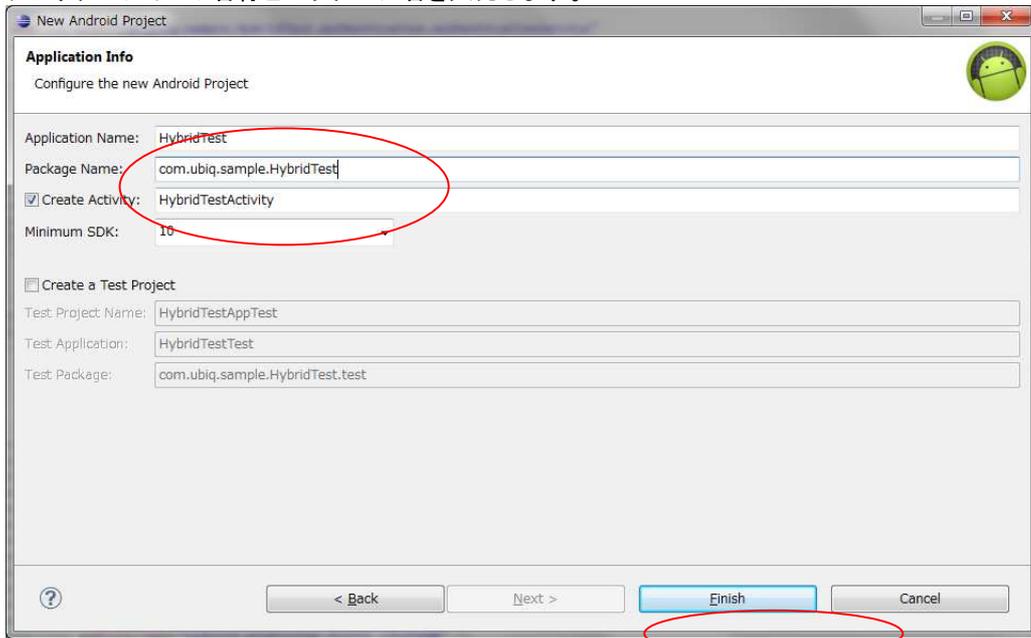
Eclipse を起動したら、「New Android Project」画面を開き、[Project Name:]欄にプロジェクト名を入力します。



「Build Target」でターゲットとなる Android プラットフォームを指定します。  
API レベルが 10 以上 (Android 2.3.3 以上) のターゲットが指定されていることを確認して、「Next」をクリックします。

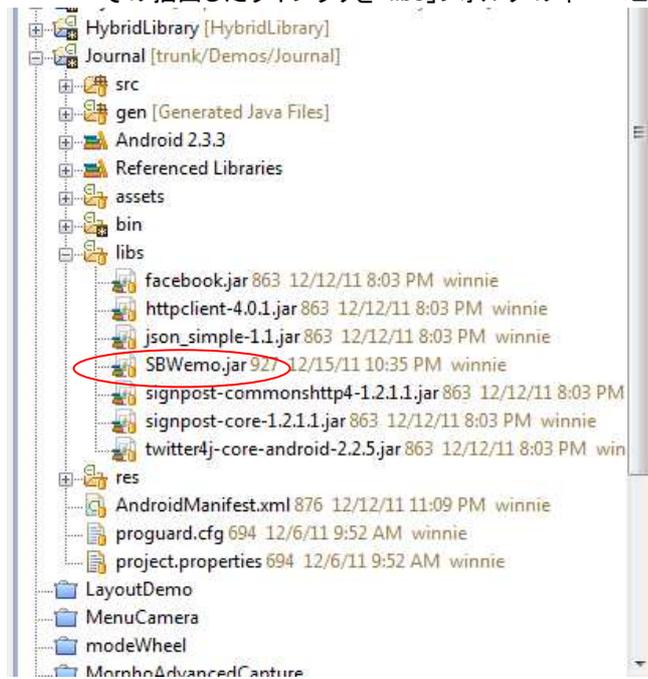


アプリケーションの名称とパッケージ名を入力します。



全て入力が完了したら、「Finish」ボタンをクリックします。

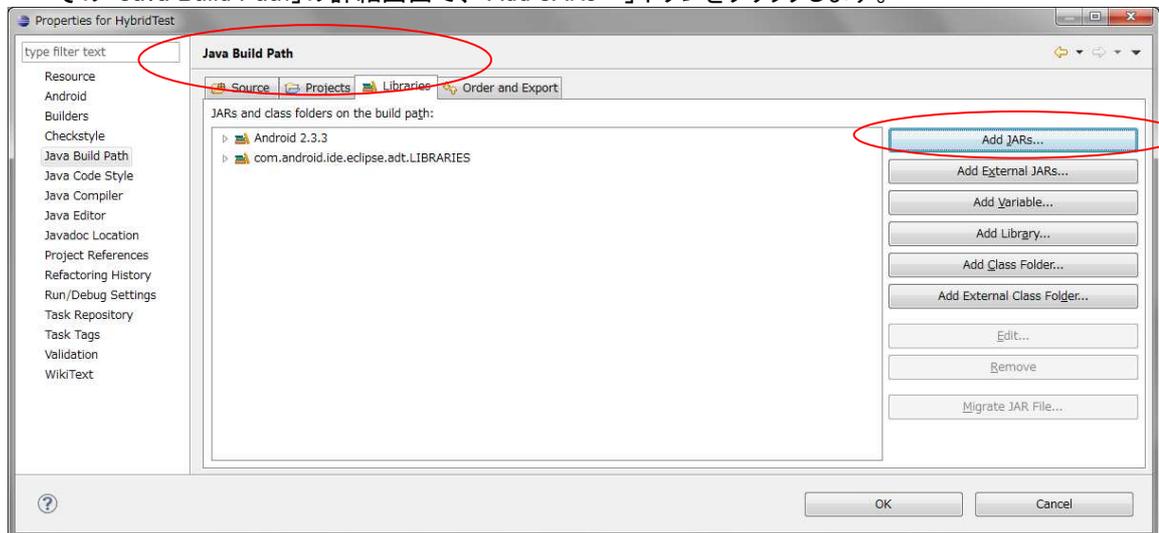
2. 作成したアプリケーションへ Smart WeMo ライブラリを追加します
  - 作成した新規プロジェクトに、「libs」フォルダを作成します。
  - Android ZIP パッケージから Smart WeMo ライブラリ (jar 形式ファイル) を展開します。
  - その抽出したライブラリを「libs」フォルダの中へコピーします。



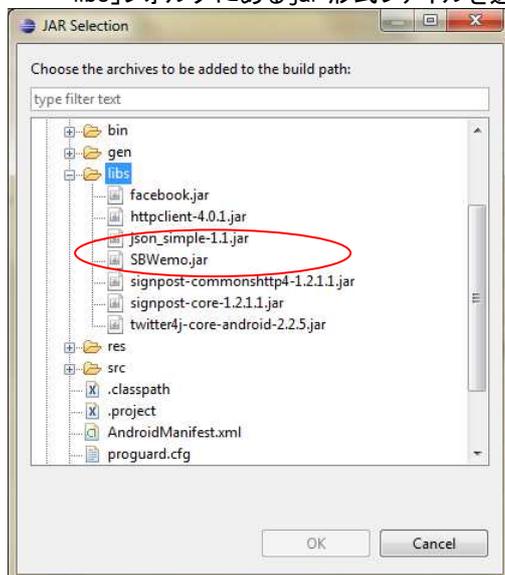
メモ: 他に「libs」フォルダへ追加する jar ファイルは、下記になります。

- facebook.jar
- httpclient-4.0.1.jar
- json\_simple-1.1.jar
- signpost-commonshttp4-1.2.1.1.jar
- signpost-core-1.2.1.1.jar
- twitter4j-core-android-2.2.5.jar

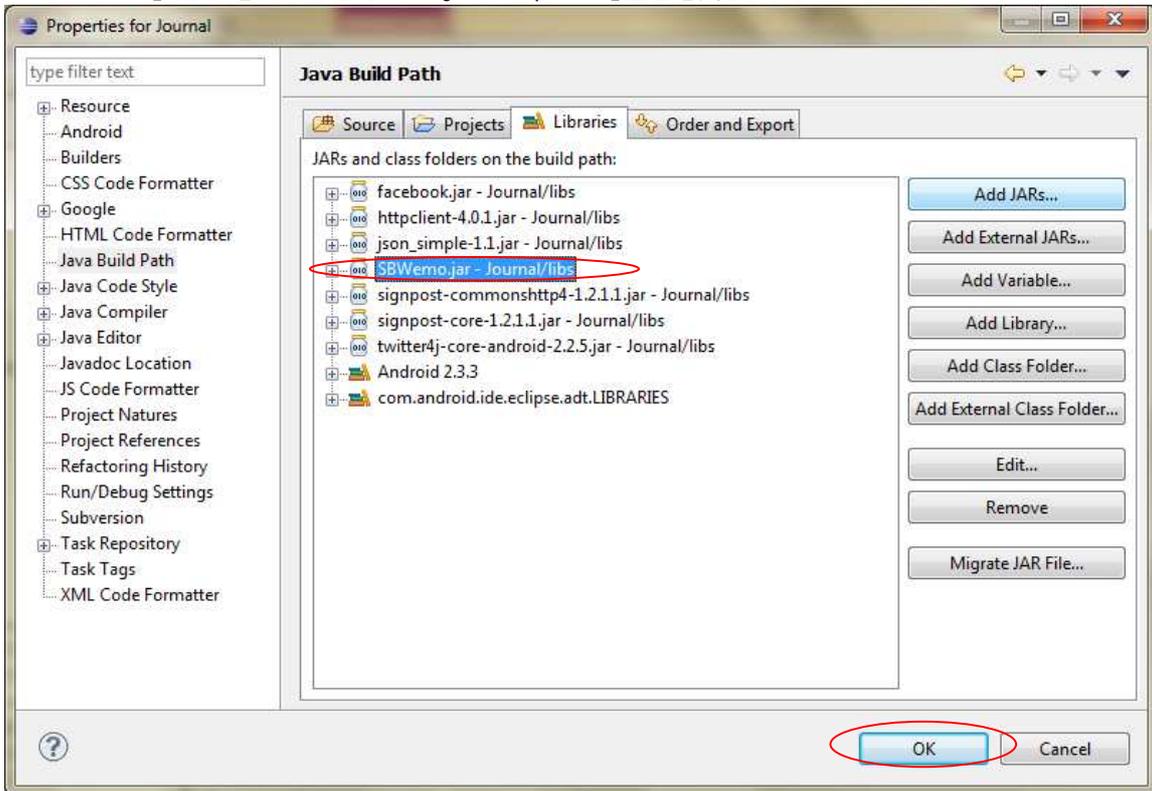
3. 「Properties」画面を開いて、「Java Build Path」をクリックします。  
その「Java Build Path」の詳細画面で、「Add JARs…」ボタンをクリックします。



4. 「JAR Selection」画面が開きます。  
「libs」フォルダにある jar 形式ファイルを選択して「OK」をクリックします。



- 「Java Build Path」画面に戻ると、「SBWemo.jar - Journal/libs」など選択した jar ファイルが表示されていることを確認します。  
「OK」ボタンをクリックして、「Project Properties」画面を閉じます。



- このプロジェクトの「AndroidManifest.xml」を開きます。
  - 次のコードを貼り付けて、全体の<application>タグの内容と入れ替えます。

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.ubiquitous.WemoApp"
    android:versionCode="1"
    android:versionName="1.0">
    <uses-sdk android:minSdkVersion="10" />

    <application android:icon="@drawable/ic_launcher" android:label="@string/app_name">
        <service
            android:name="com.app.package.Name.authentication.AuthenticationService"
            android:exported="true">
            <intent-filter>
                <action
                    android:name="android.accounts.AccountAuthenticator" />
            </intent-filter>
            <meta-data
```

```

        android:name="android.accounts.AccountAuthenticator"
        android:resource="@xml/authenticator" />
    </service>
    <activity
        android:name=".HybridTestActivity"
        android:configChanges="orientation|keyboardHidden"
        android:theme="@android:style/Theme.NoTitleBar">
        <intent-filter>
            <action android:name="android.intent.action.MAIN" />
            <category android:name="android.intent.category.LAUNCHER" />
        </intent-filter>
    </activity>
    <activity android:name="com.ubiquitous_tech.HybridLibrary.PrepareRequestTokenActivity"
        android:launchMode="singleTask">
        <intent-filter>
            <action android:name="android.intent.action.VIEW" />
            <category android:name="android.intent.category.DEFAULT" />
            <category android:name="android.intent.category.BROWSABLE" />
            <data android:scheme="x-oauthflow-twitter" android:host="callback" />
        </intent-filter>
    </activity>
</application>
<uses-permission android:name="android.permission.READ_CONTACTS" />
<uses-permission android:name="android.permission.WRITE_CONTACTS" />
<uses-permission android:name="android.permission.GET_ACCOUNTS" />
<uses-permission android:name="android.permission.AUTHENTICATE_ACCOUNTS" />
<uses-permission android:name="android.permission.CAMERA" />
<uses-permission android:name="android.permission.INTERNET" />
<uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION" />
<uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
<uses-permission android:name="android.permission.ACCESS_GPS" />
<uses-permission android:name="android.permission.ACCESS_ASSISTED_GPS" />
<uses-permission android:name="android.permission.ACCESS_LOCATION" />
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
<uses-permission android:name="android.permission.BLUETOOTH_ADMIN" />
<uses-permission android:name="android.permission.BLUETOOTH" />
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />

```

- Android Developer Tool が 15 以下の場合は下記を使用してください:

```
android:icon="@drawable/ic_launcher" to android:icon="@drawable/icon".
```

これはデフォルト名なので、ランチャのアイコン名に応じて変更 できます。  
 (例: android:icon="@drawable/name\_of\_your\_launcher\_icon")

- 最初のハイライトされている文字が赤の場合、プロジェクトの パッケージ名 にします。

- 二番目のハイライトされている文字が緑の場合、プロジェクトアクティビティの名称にします。

「AndroidManifest.xml」ファイルは下記のようになります：

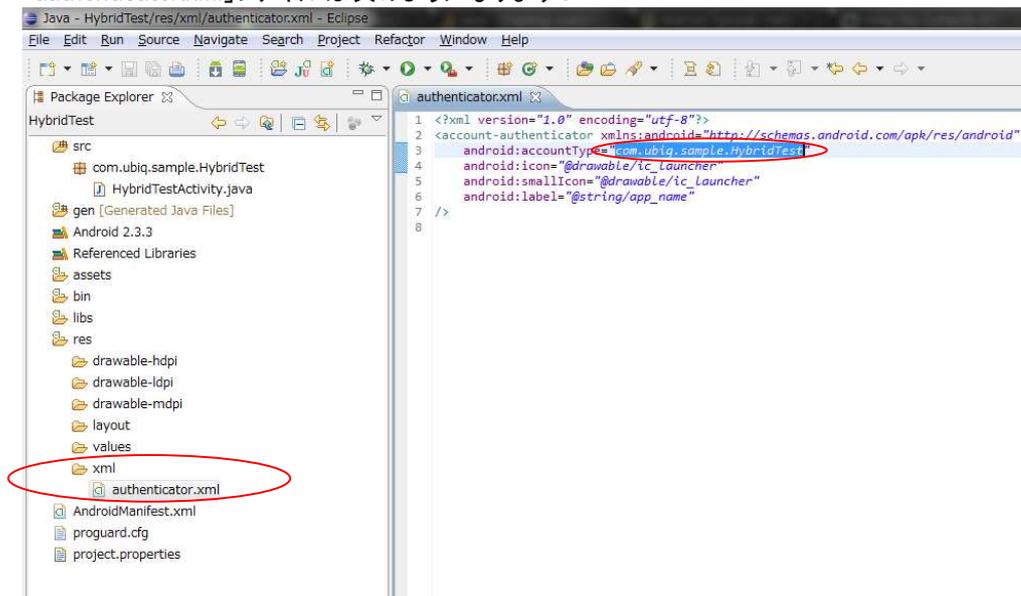
```
HybridLibrary Manifest
1 <?xml version="1.0" encoding="utf-8"?>
2 <manifest xmlns:android="http://schemas.android.com/apk/res/android"
3     package="com.ubiquitous_tech.HybridLibrary.demo"
4     android:versionCode="1"
5     android:versionName="1.0" >
6
7     <uses-sdk android:minSdkVersion="10" />
8
9     <application
10        android:icon="@drawable/icon"
11        android:label="@string/app_name" >
12        <uses-library android:name="android.test.runner" />
13
14        <service
15            android:exported="true"
16            android:name="com.ubiquitous_tech.HybridLibrary.demo.authentication.AuthenticationService" >
17            <intent-filter >
18                <action android:name="android.accounts.AccountAuthenticator" />
19            </intent-filter>
20
21            <meta-data
22                android:name="android.accounts.AccountAuthenticator"
23                android:resource="@xml/authenticator" />
24        </service>
25
26        <activity
27            android:configChanges="orientation|keyboardHidden"
28            android:name=".MainActivity"
29            android:theme="@android:style/Theme.NoTitleBar" >
30            <intent-filter >
31                <action android:name="android.intent.action.MAIN" />
32
33                <category android:name="android.intent.category.LAUNCHER" />
34            </intent-filter>
35        </activity>
36        <activity
37            android:launchMode="singleTask"
38            android:name="com.ubiquitous_tech.HybridLibrary.PrepareRequestTokenActivity" >
39            <intent-filter >
40                <action android:name="android.intent.action.VIEW" />
41
42                <category android:name="android.intent.category.DEFAULT" />
43                <category android:name="android.intent.category.BROWSABLE" />
44
45                <data
46                    android:host="callback"
47                    android:scheme="x-oauthflow-twitter" />
48            </intent-filter>
49        </activity>
50    </application>
```

7. 「res」フォルダ内に「xml」名のサブフォルダを作成します。その「xml」フォルダ内に「authenticator.xml」ファイルを作成して、下記のテキストをファイルの中に貼り付けてください。

```
<?xml version="1.0" encoding="utf-8"?>
<account-authenticator
  xmlns:android="http://schemas.android.com/apk/res/android"
  android:accountType="com.ubiq.sample.HybridTest"
  android:icon="@drawable/ic_launcher"
  android:smallIcon="@drawable/ic_launcher"
  android:label="@string/app_name"
/>
```

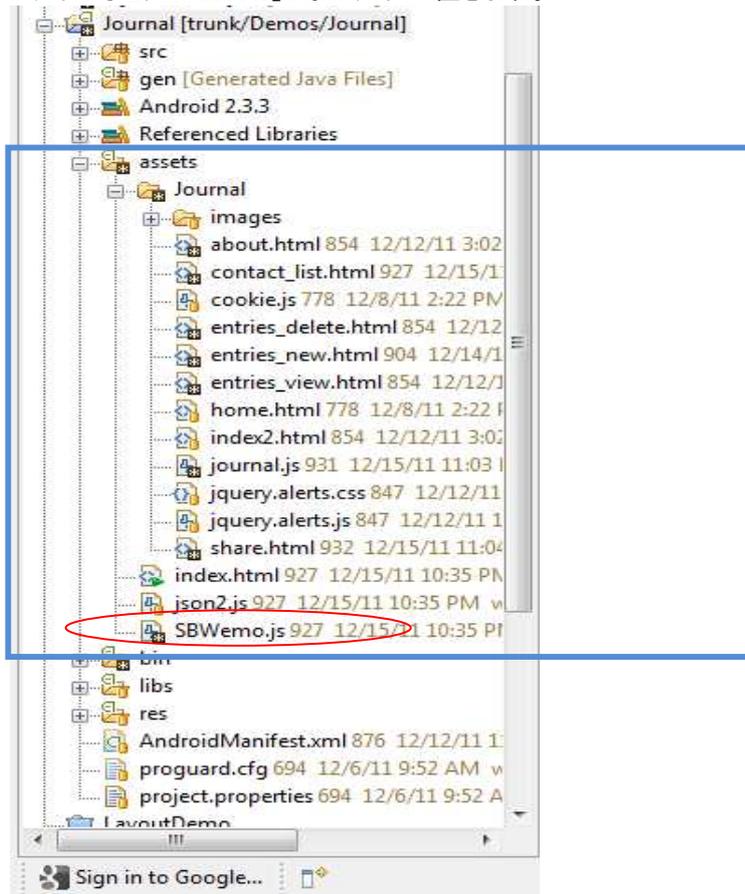
- 上記サンプルで「android:accountType」の値を、アプリケーションのパッケージ名に変更します。
- ADTバージョンが15以下の場合、「@drawable/ic\_launcher」を「@drawable/icon」へ変更します。これはデフォルト名なので、ランチャのアイコン名に応じて変更できます。  
(例:「@drawable/name\_of\_your\_launcher\_icon」)

「authenticator.xml」ファイルは次のようになります：



8. Android ZIP パッケージの JS フォルダから、圧縮させた Smart WeMo JavaScript 用のインターフェイスのライブラリ(SBWemo.js)をコピーして Eclipse プロジェクトの「assets」フォルダの中へ置きます。

アプリケーションに必要な HTML ファイル、JavaScript ファイル、他のリソースファイル (画像、css ファイルなど) は「assets」フォルダ下に 置きます。



9. Smart WeMo JavaScript 用のインターフェイスライブラリ(SBWemo.js)をアプリケーションの HTML ページに正確に引用されて、初期化されていることを確認します。

注:SBWemo.js を初期化しないとライブラリは正常に起動しません。ライブラリを初期化するには、下記の図のように SBWemo.js の関数 `initLib()` を呼びます。

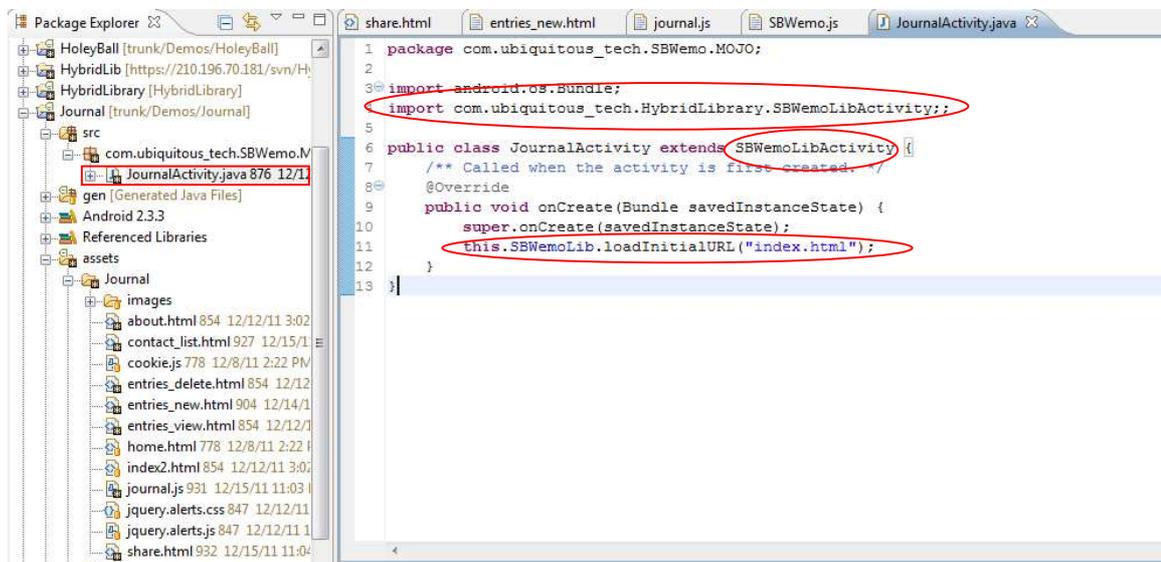
```
<!-- Alerts, Prompts -->
<script src="jquery.alerts.js" type="text/javascript"></script>
<link href="jquery.alerts.css" rel="stylesheet" type="text/css" media="screen" />
<script type="text/javascript" src="SmartWemo.js"></script>
<script>
  function init() {
    initLib();
  }

  $(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">
```

10. 下記の図のようにアプリケーションのメインアクティビティを編集します。  
Smart WeMo パッケージ(com.ubiquitous\_tech.HybridLibrary.SBWeMoLibActivity) をアクティビティにインポートします。

上位クラスのアクティビティを「SBWeMoLibActivity」へ変更して、「this.mHybridNativeLib.loadInitialURL()」メソッドを呼び込んで、最初の html ファイルをアプリケーションにロードします。

Class は次のようになります：



```
1 package com.ubiquitous_tech.SBWeMo.MOJO;
2
3 import android.os.Bundle;
4 import com.ubiquitous_tech.HybridLibrary.SBWeMoLibActivity;
5
6 public class JournalActivity extends SBWeMoLibActivity {
7     /** Called when the activity is first created. */
8     @Override
9     public void onCreate(Bundle savedInstanceState) {
10        super.onCreate(savedInstanceState);
11        this.SBWeMoLib.loadInitialURL("index.html");
12    }
13 }
```

11. Android アプリケーションに必要な HTML ファイル、JavaScript ファイル作成が完了したら、従来通り Android 端末に実装して起動させてください。