Device Connect 1.0

Android アプリケーション開発チュートリアル

1.0.0 版

2014 年 9 月 24 日

変更履歴

変更日付	変更内容	変更担当者
2014/09/24	初版作成。	畠山

目次

1. はじめに

本ドキュメントは、Device Connect API 1.0 を用いて Android アプリケーションを開発する手法について解説する。Android に関する一般的な知識と、Java に関する一般的な知識を保持している事を提に解説していく。

2. Device Connect について

Device Connect では、RESTful のやり取りで、命令やデータの送受信が可能である。各処理は、Profile という形で URI が割り振られ、その URI を JavaScript で呼び出す事で、処理を行う。

2.1 Device Connect 対応の Profile

Device Connect で、サポートしている標準 Profile を以下に示す。

System API	Device Connect 本体の情報や、プラグインの情報、プラグインの設定画面の起動などができる。
Battery Status API	バッテリーの状態を保持できる。
Proximity API	デバイスと物体との近接状態を取得する。
Media Player API	メディアの再生(音声, 音楽, 動画)を行う。
Network Service Discovery API	デバイスの対応 Profile 一覧を取得する。
Vibration API	デバイスをバイブレートする命令を送る。
Notification API	デバイスに通知(Notification)を行う。
Setting API	デバイスの設定(音量, 画面輝度, 画面スリープ時間等)を行う。
DeviceOrientation API	デバイスの加速度等を取得する。
File API	デバイスと File の送信、受信等を行う。
File Descriptor API	デバイスに File の生成、書き込み/読み込み等を行う。
MediaStream Recording API	メディアの録音(音声,音楽,動画)を行う。
Phone API	デバイスに電話発信の命令を送る。

2.2 Device Connect への接続

Device Connect では、Android で開発されるアプリケーションは、スマートデバイス内のローカルに起動している httpd サーバに問い合わせを行う事で、ハードウェアの操作や、情報取得を行う事が可能である。

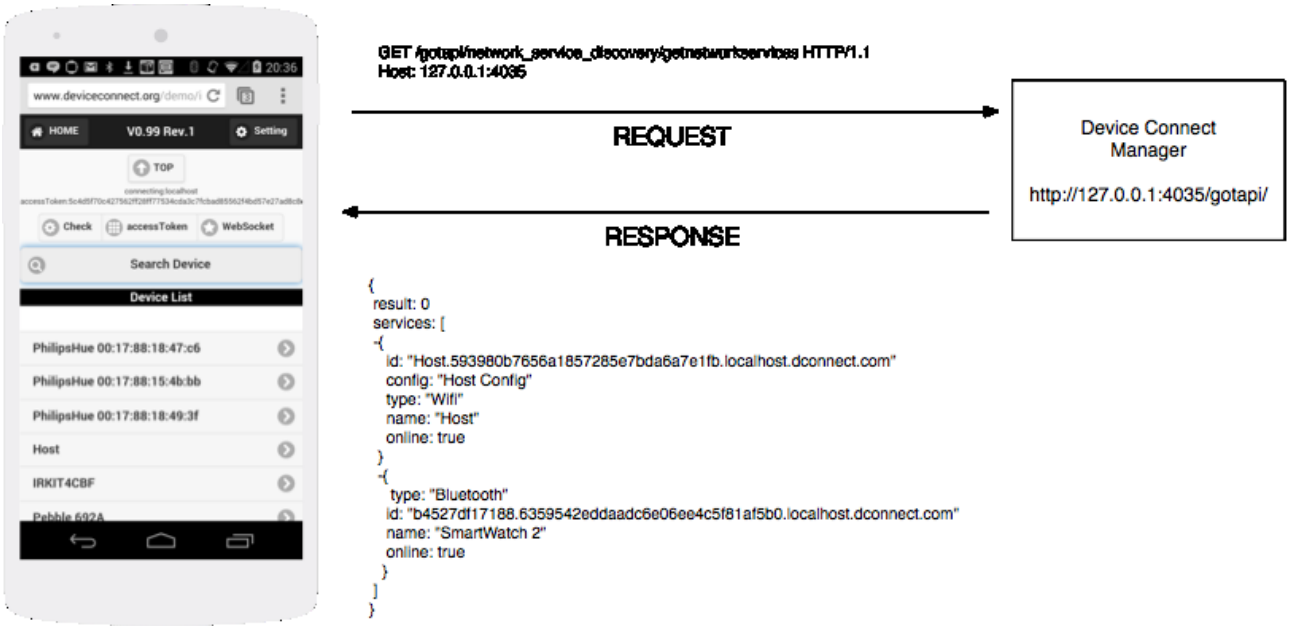
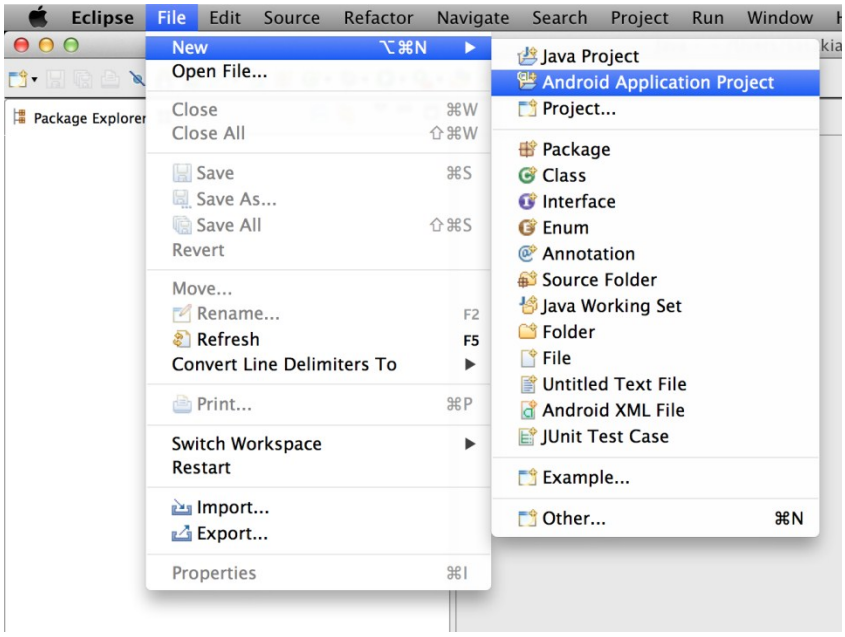


図 1. Device Connect Manager への問い合わせ

2.3 サンプルの作成

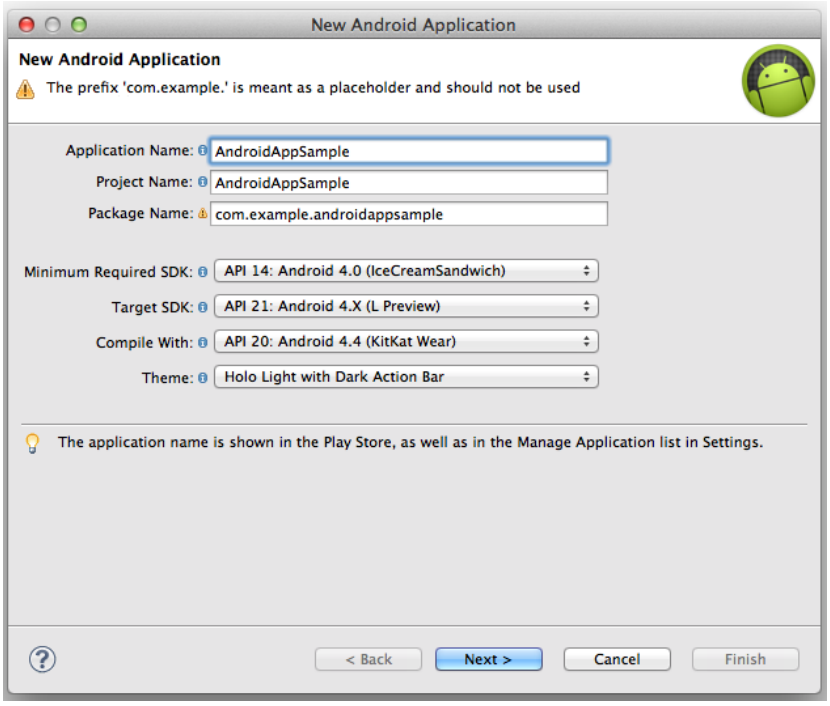
Android アプリケーションにて、Device Connect Manager に問い合わせを行い、結果の JSON を取得するサンプルを作成しながら解説をする。

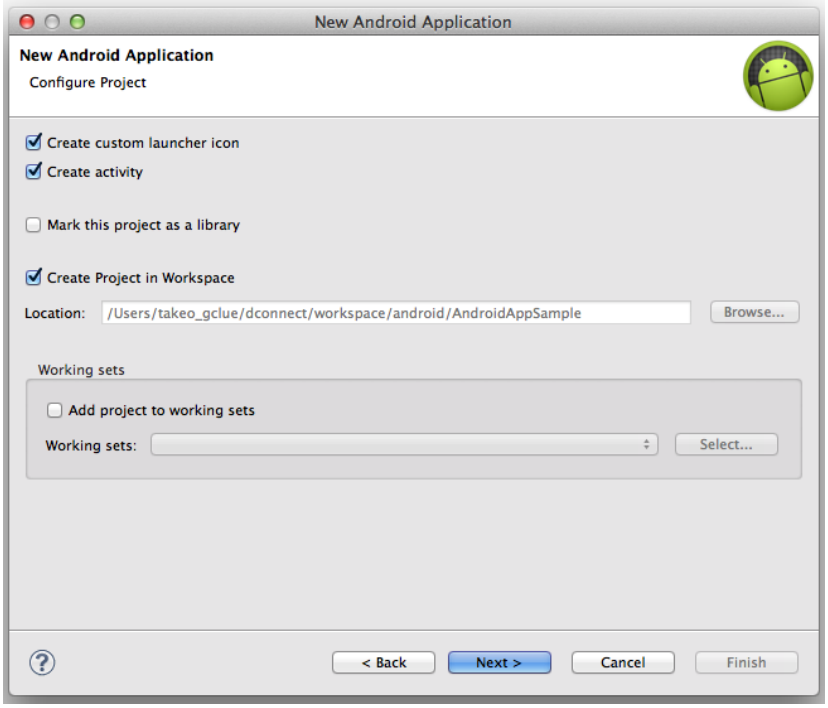
Eclipse の[File]-[New]-[Android Application Project]を選択する。



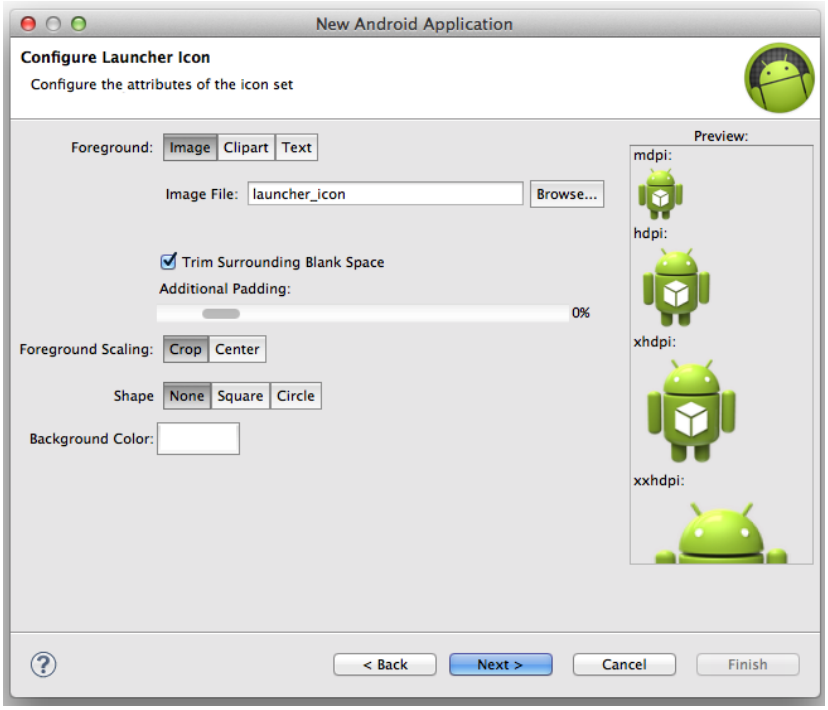
Application Name, Project Name, Package Name を入力する。

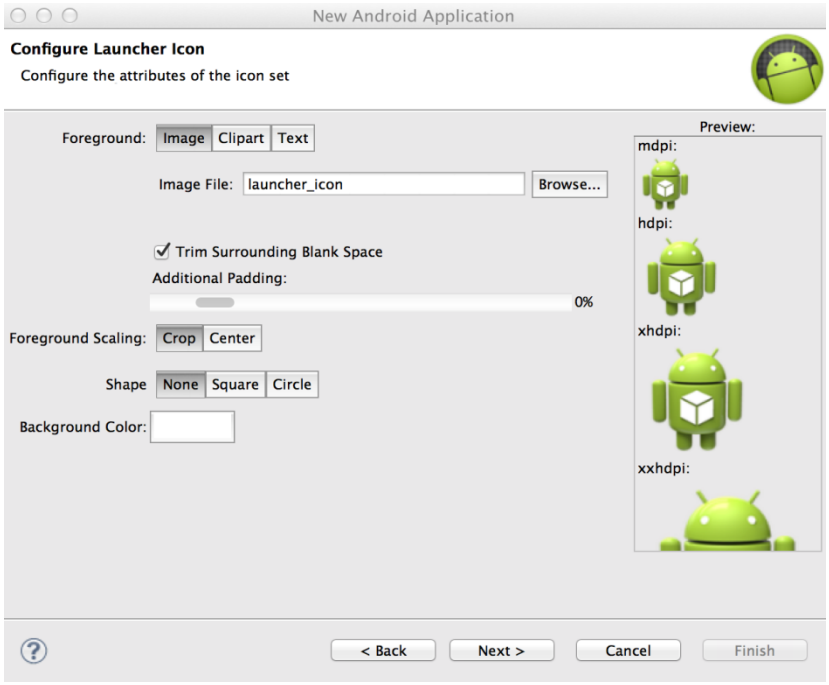
Application Name	AndroidAppSample
Project Name	AndroidAppSample
Package Name	com.example.androidappsample
Minimum Required SDK	Android 4.0



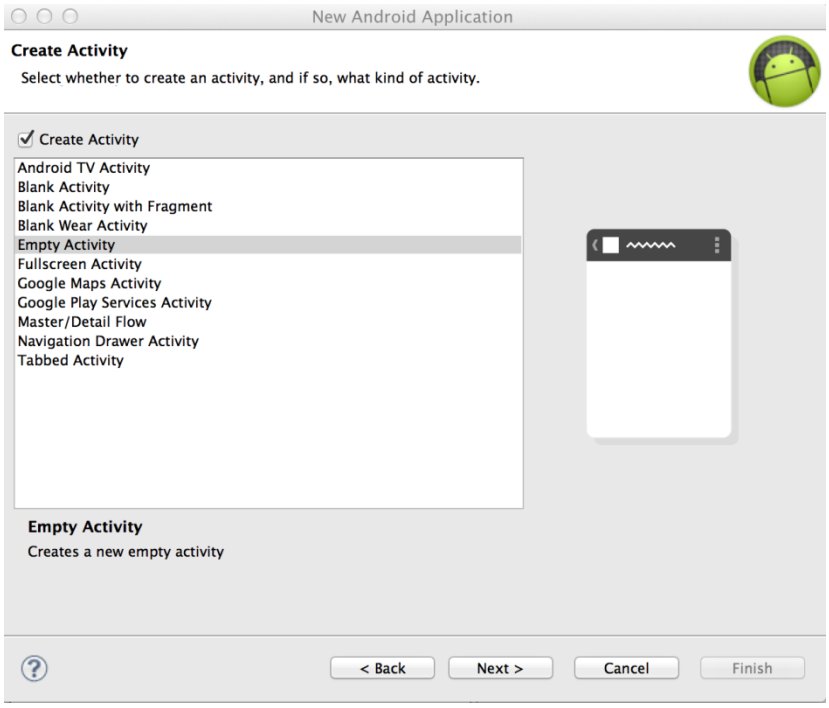

New Android Application
New Android Application
Configure Project
Create custom launcher icon
Create activity
Mark this project as a library
Create Project in Workspace
Location:
/Users/takeo_gclue/dconnect/workspace/android/AndroidAppSample
Browse...
Working sets
Add project to working sets
Working sets:
Select...
< Back
Next >
Cancel
Finish

アイコンを選択する。

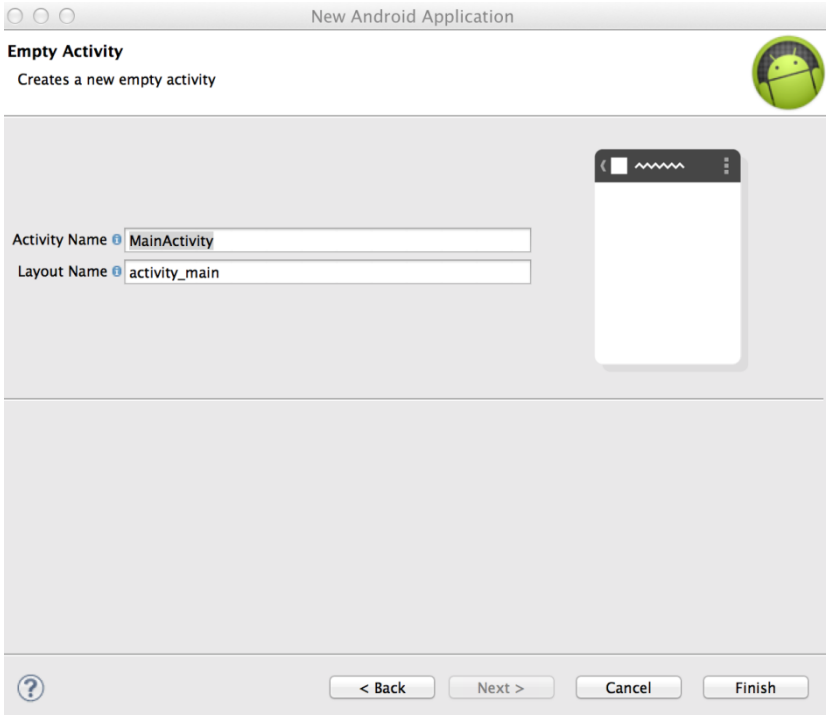

New Android Application
Configure Launcher Icon
Configure the attributes of the icon set
Foreground:
Image
Clipart
Text
Image File:
launcher_icon
Browse...
Trim Surrounding Blank Space
Additional Padding:
0%
Foreground Scaling:
Crop
Center
Shape
None
Square
Circle
Background Color:
Preview:
mdpi:
hdpi:
xhdpi:
xxhdpi:
< Back
Next >
Cancel
Finish



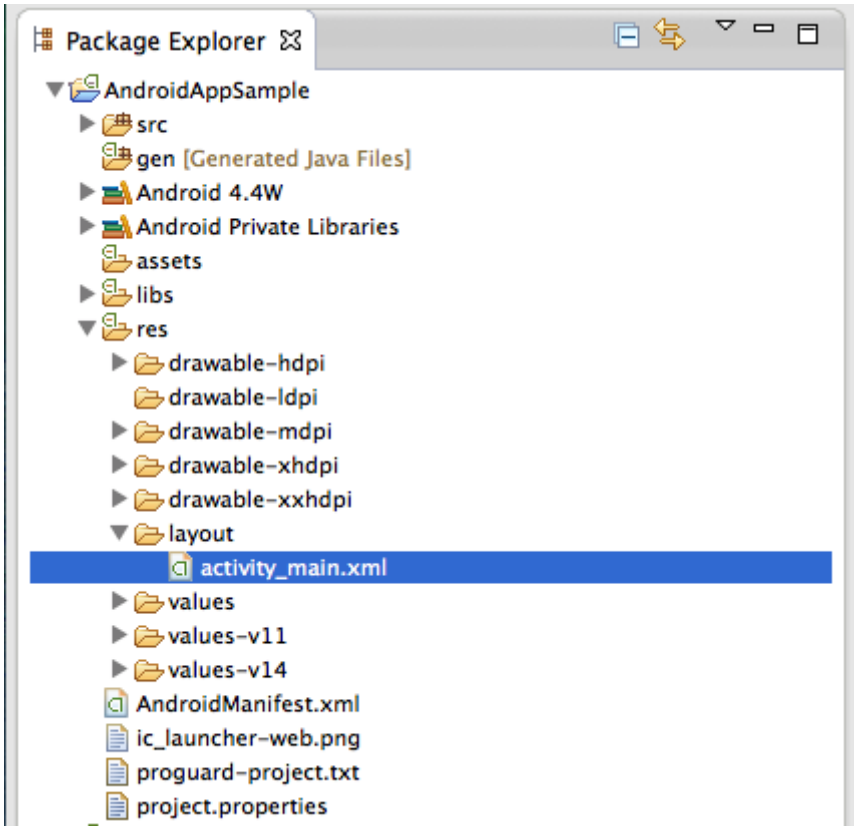
Activity を選択する。



Activity 名を入力する。



ここまで Android Application Wizard に従ってプロジェクトを作成すると、以下のように Package Explorer にプロジェクトが追加される。



2.4 MainActivity の編集

表示画面となる res/layout/main_activity.xml と、MainActivity.java を編集する。

res/layout/main_activity.xml

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context="${relativePackage}.${activityClass}" >

    <Button android:id="@+id/button"
        android:layout width="fill parent"
        android:layout height="wrap content"
        android:text="Button" />

</RelativeLayout>
```

MainActivity.java

```
package com.example.androidappsample;


import org.apache.http.HttpResponse;
import org.apache.http.client.methods.HttpGet;
import org.apache.http.impl.client.DefaultHttpClient;
import org.apache.http.util.EntityUtils;


import android.app.Activity;
import android.content.Context;
import android.os.Bundle;
import android.os.Looper;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;
import android.widget.Toast;


public class MainActivity extends Activity implements OnClickListener {


        private Button mButton;
        private Context mContext;


        @Override
        protected void onCreate(Bundle savedInstanceState) {
                super.onCreate(savedInstanceState);
                setContentView(R.layout.activity_main);
                mContext = this.getBaseContext();


                mButton = (Button) findViewById(R.id.button);
                mButton.setOnClickListener(this);
        }


        @Override
        public void onClick(View v) {
                (new Thread(new Runnable() {
                        @Override
                        public void run() {
                                try {
                                        String url = "http://localhost:4035/gotapi/system";
                                        HttpGet mMethod = new HttpGet(url);
                                        DefaultHttpClient mClient = new DefaultHttpClient();
                                        HttpResponse response = mClient.execute(mMethod);


                                        String mResult = EntityUtils.toString(response.getEntity(),
                                                        "UTF-8");


                                        Looper.prepare();
                                        Toast.makeText(mContext, mResult, Toast.LENGTH_LONG).show();
                                        Looper.loop();


                                } catch (Exception e) {
                                        e.printStackTrace();
                                }
                        }
                })).start();
        }
```

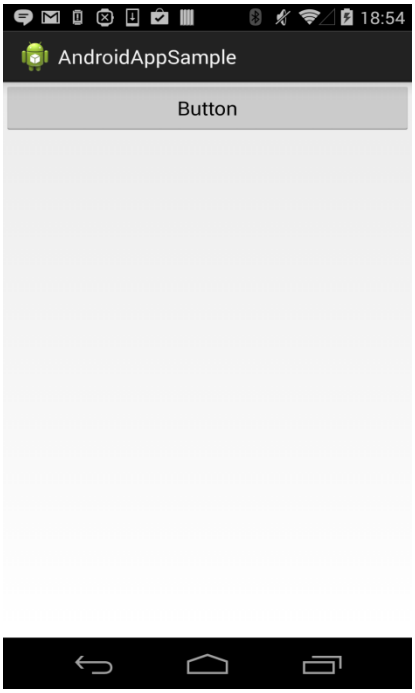
```
}
```

http 通信を行うため、AndroidManifest.xml に、INTERNET の Permission を設定する。

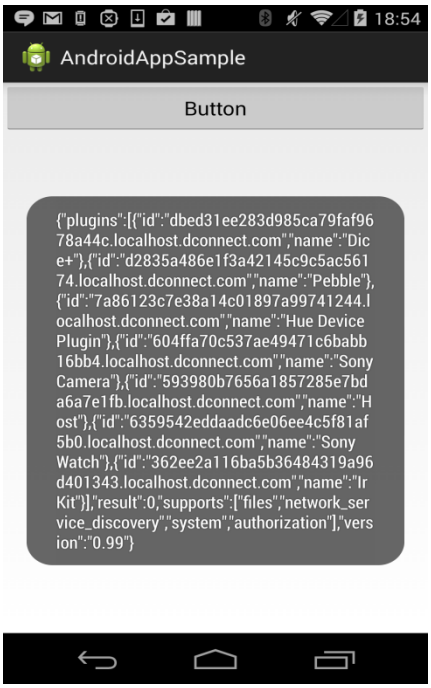
AndroidManifest.xml

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.androidappsample"
    android:versionCode="1"
    android:versionName="1.0" >


    <uses-sdk
        android:minSdkVersion="14"
        android:targetSdkVersion="21" />
    <uses-permission android:name="android.permission.INTERNET"/>


    <application
        android:allowBackup="true"
        android:icon="@drawable/ic_launcher"
        android:label="@string/app_name"
        android:theme="@style/AppTheme" >
        <activity
            android:name=".MainActivity"
            android:label="@string/app_name" >
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />


                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>


</manifest>
```

ここまでの作成を終えた後、このサンプルをインストールして起動すると、以下のような画面が表示される。

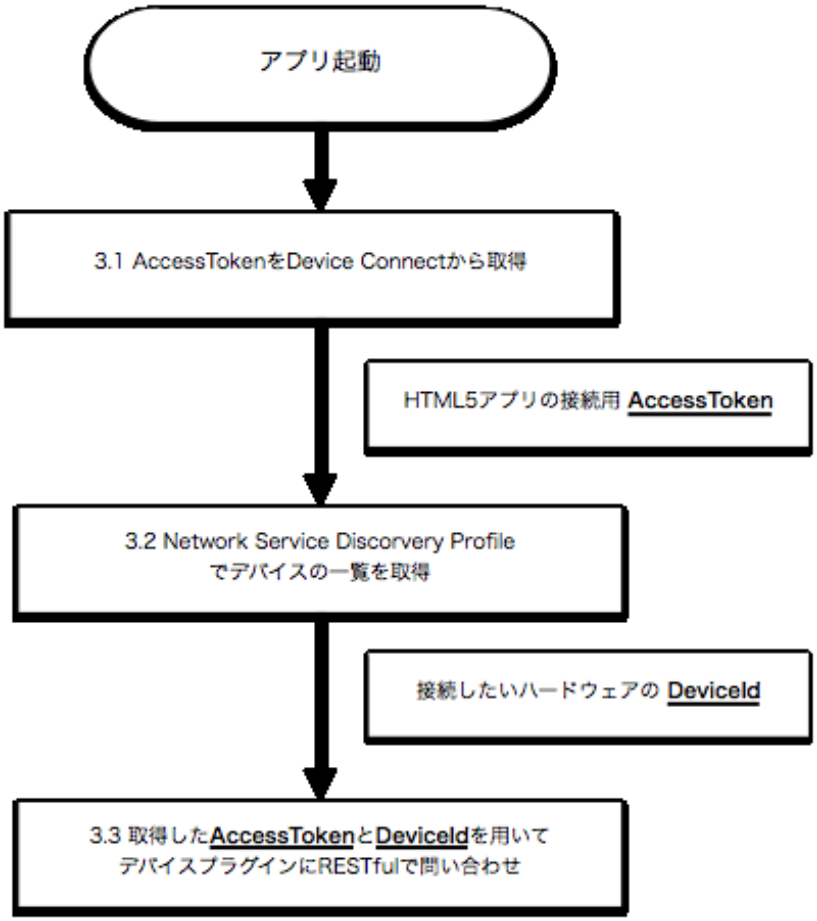


Button を押す事で、端末にインストールされている Device Connect 1.0 デバイスプラグインの情報を取得して、Toast で表示する事が出来る。



3. SDK を用いた Android アプリケーションの開発

Device Connect との通信には、いくつかの手順を行う必要がある。Android でアプリケーションを開発する際に、最初に AccessToken を取得する。また、接続したハードウェアの DeviceId を Network Service Discovery Profile にアクセスして取得する。AccessToken と DeviceId を用いて、操作したいハードウェアのプラグインに対して処理の送信が可能となる。

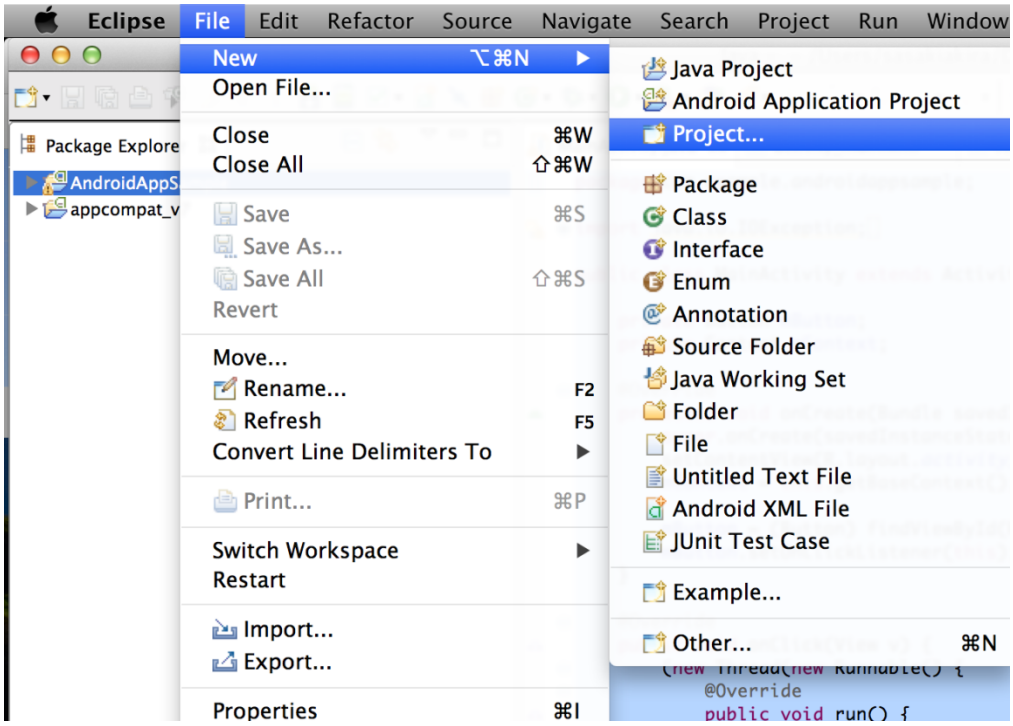


3.1 AccessToken を Device Connect から取得

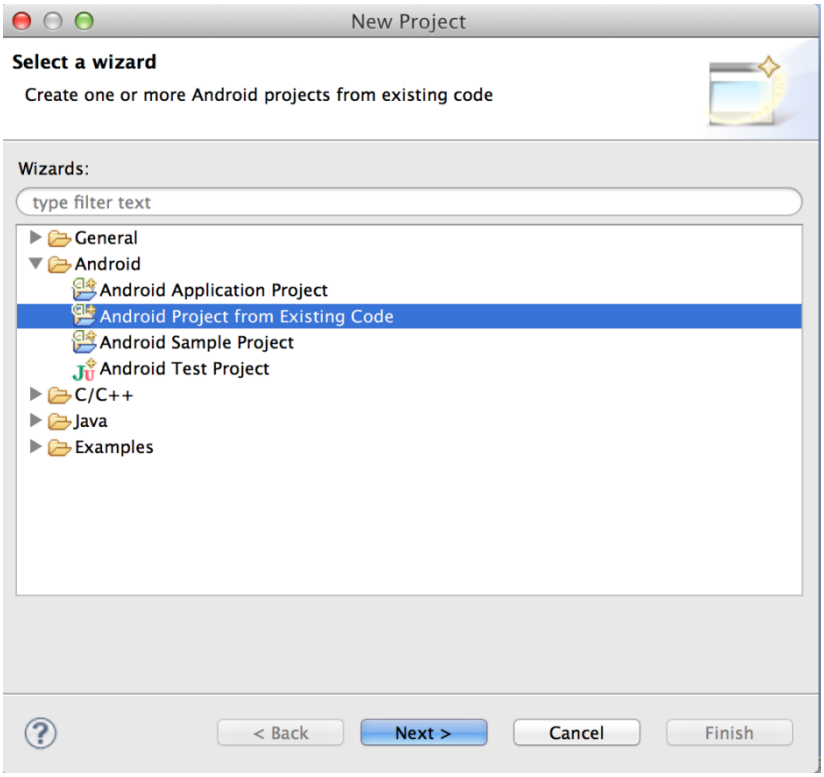
Device Connect にアクセスするアプリは、OAuth(Local)認証で用いる AccessToken を、Device Connect Manager より発行してもらう必要がある。本サンプルから、Device Connect SDK Android 版を用いて解説する。

3.1.1 dConnectSDKAndroid のインポート

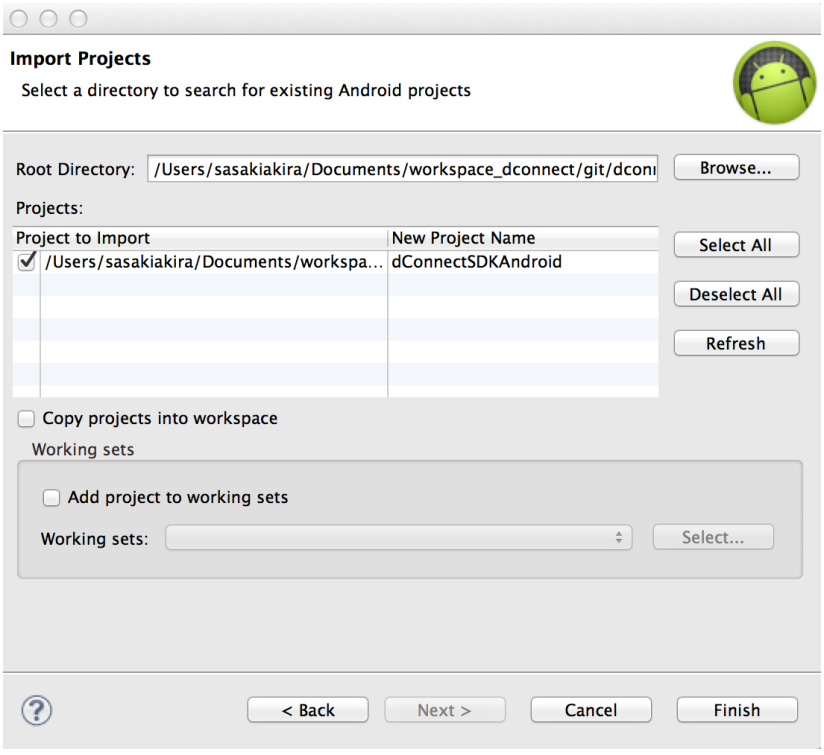
Eclipse の import 機能で、プロジェクト「dConnectSDKAndroid」をインポートする。
Eclipse のメニューから[File]-[New]-[Project..]を選択する。



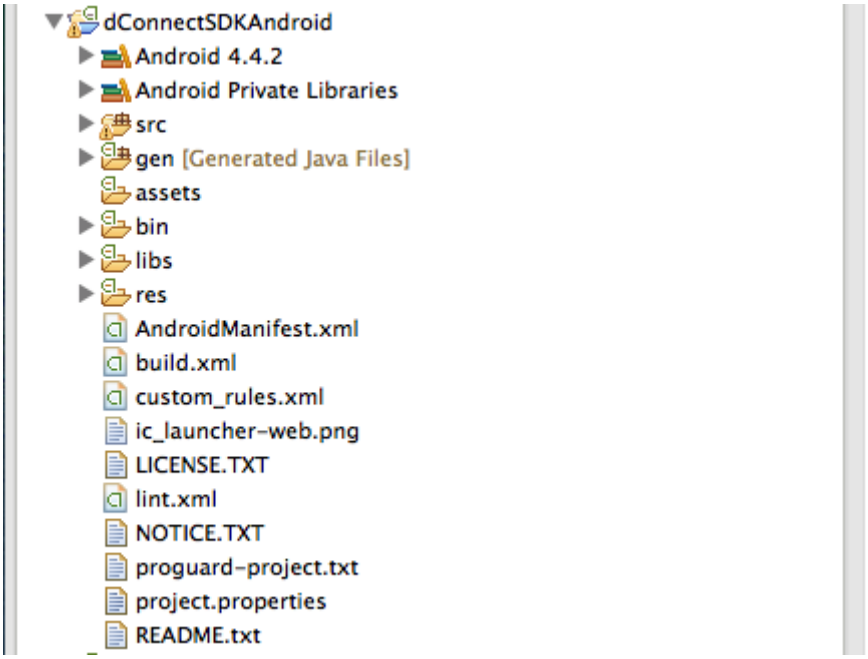
Android の下にある Android Project from Existing Code を選択する。



Browse...ボタンを押下して、dConnectSDKAndroid を選択すると、Project to import に dConnectSDKAndroid が表示されるので、チェックボックスにチェックをして Finish ボタンを押下する。

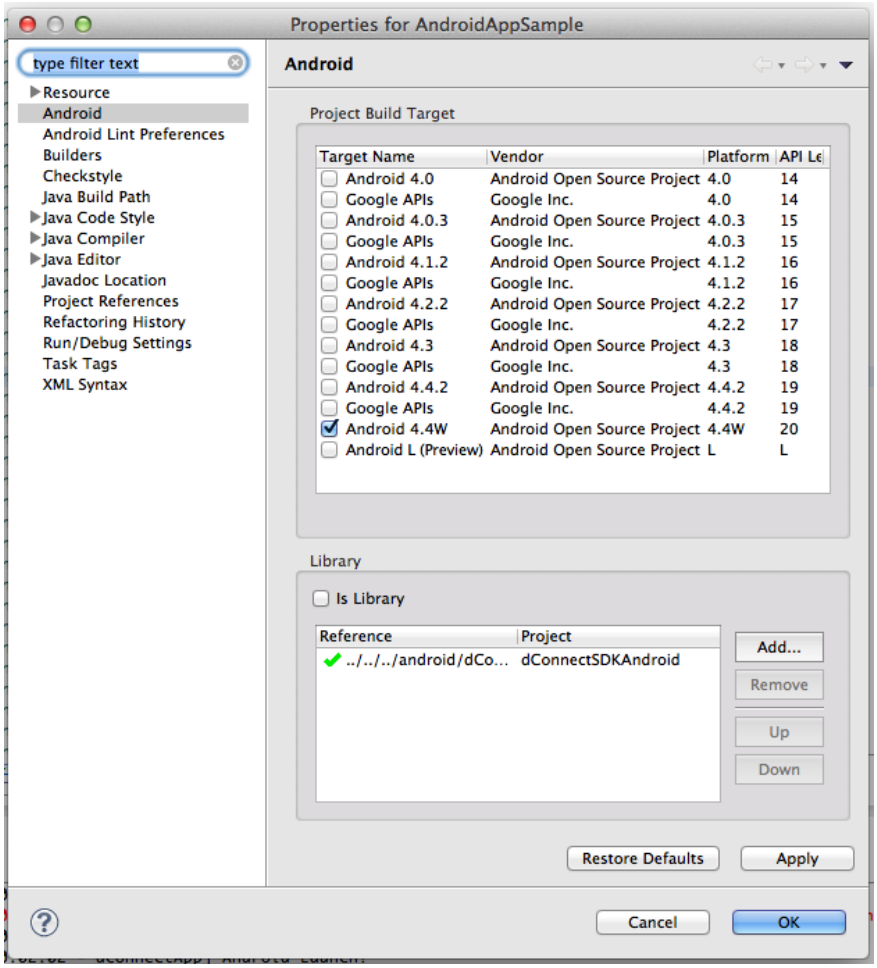


これで、dConnetSDKAndroid が Eclipse のプロジェクトとしてインポートされる。
正常にインポートされた場合に以下のような Android ライブラリプロジェクトが Package Explorer に追加される。



3.1.2　dConnectSDKAndroid ライブラリプロジェクトの追加

インポートした dConnectSDKAndroid ライブラリを AndroidAppSample で使用できるように設定を行う。
Package Explorer で AndroidAppSample のプロパティを開き、左の項目から Android を選択し、右下の参照ライブラリ一覧の脇にある Add ボタンを押下して dConnectSDKAndroid を追加する。



プロジェクトによっては、android-support-v4.jar で警告が発生する場合がある。その際には、android-support-v4.jar を削除する*1。

*1: オフィシャル Android SDK によって提供される Support Library は随時アップデートされるため、様々なバージョンが存在する。Device Connect のプロジェクト群の内にバージョンの異なる Support Library が共存してしまう際は、バージョン競合によってプロジェクトのビルドが失敗するので、いずれか 1 つのバージョンの Support Library のみ残す様に、参照ライブラリを含めプロジェクトで使われる Support Library を整理する必要ある。

3.1.3 AccessToken 取得部の実装

画面にボタンを追加して、ボタン押下で AccessToken 取得処理を行わせるように実装する。

res/layout/main_activity.xml

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context="${relativePackage}.${activityClass}" >


    <Button android:id="@+id/button1"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:text="System" />


    <Button android:id="@+id/button2"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:layout_alignParentLeft="true"
        android:layout_below="@+id/button1"
        android:text="Get AccessToken" />


</RelativeLayout>
```

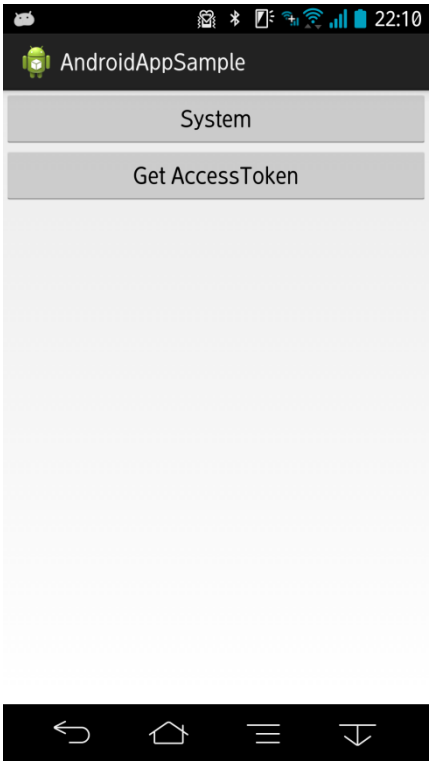
MainActivity.java

```
package com.example.androidappsample;


import org.apache.http.HttpResponse;
import org.apache.http.client.methods.HttpGet;
import org.apache.http.impl.client.DefaultHttpClient;
import org.apache.http.util.EntityUtils;


import com.nttdocomo.dconnect.message.DConnectMessage.ErrorCode;
import com.nttdocomo.dconnect.profile.AuthorizationProfileConstants;
import com.nttdocomo.dconnect.profile.BatteryProfileConstants;
import com.nttdocomo.dconnect.profile.ConnectProfileConstants;
import com.nttdocomo.dconnect.profile.DeviceOrientationProfileConstants;
import com.nttdocomo.dconnect.profile.FileDescriptorProfileConstants;
import com.nttdocomo.dconnect.profile.FileProfileConstants;
import com.nttdocomo.dconnect.profile.MediaPlayerProfileConstants;
import com.nttdocomo.dconnect.profile.MediaStreamRecordingProfileConstants;
import com.nttdocomo.dconnect.profile.NetworkServiceDiscoveryProfileConstants;
import com.nttdocomo.dconnect.profile.NotificationProfileConstants;
import com.nttdocomo.dconnect.profile.PhoneProfileConstants;
import com.nttdocomo.dconnect.profile.ProximityProfileConstants;
import com.nttdocomo.dconnect.profile.SettingsProfileConstants;
import com.nttdocomo.dconnect.profile.SystemProfileConstants;
import com.nttdocomo.dconnect.profile.VibrationProfileConstants;
import com.nttdocomo.dconnect.utils.AuthProcesser;
import com.nttdocomo.dconnect.utils.AuthProcesser.AuthorizationHandler;


import android.app.Activity;
import android.content.Context;
import android.os.Bundle;
import android.os.Looper;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;
import android.widget.Toast;


public class MainActivity extends Activity implements OnClickListener {
    private Button mButton1;
    private Button mButton2;
    private Context mContext;


    private String mClientId;
    private String mClientSecret;
    private String mAccessToken;
    private ErrorCode mError;
    /**
     * Local OAuth に使用するスコープ一覧.
     */
    private String[] scopes = {
        AuthorizationProfileConstants.PROFILE_NAME,
        BatteryProfileConstants.PROFILE_NAME,
```

```
        ConnectProfileConstants.PROFILE_NAME,
        DeviceOrientationProfileConstants.PROFILE_NAME,
        FileDescriptorProfileConstants.PROFILE_NAME,
        FileProfileConstants.PROFILE_NAME,
        MediaPlayerProfileConstants.PROFILE_NAME,
        MediaStreamRecordingProfileConstants.PROFILE_NAME,
        NetworkServiceDiscoveryProfileConstants.PROFILE_NAME,
        NotificationProfileConstants.PROFILE_NAME,
        PhoneProfileConstants.PROFILE_NAME,
        ProximityProfileConstants.PROFILE_NAME,
        SettingsProfileConstants.PROFILE_NAME,
        SystemProfileConstants.PROFILE_NAME,
        VibrationProfileConstants.PROFILE_NAME,


        // 独自プロファイル
        "light",
        "camera",
        "temperature",
        "dice",
        "sphero",
        "drive_controller",
        "remote_controller",
        "mhealth",


        // テスト用
        "*"
    };

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        mContext = this.getBaseContext();


        mButton1 = (Button) findViewById(R.id.button1);
        mButton2 = (Button) findViewById(R.id.button2);
        mButton1.setOnClickListener(this);
        mButton2.setOnClickListener(this);
    }


    @Override
    public void onClick(View v) {
        switch(v.getId()) {
        case R.id.button1:
            (new Thread(new Runnable() {
                @Override
                public void run() {
                    try {
                        String url = "http://localhost:4035/gotapi/system";
                        HttpGet mMethod = new HttpGet(url);
                        DefaultHttpClient mClient = new DefaultHttpClient();
                        HttpResponse response = mClient.execute(mMethod);


                        String mResult = EntityUtils.toString(response.getEntity(),
                                     "UTF-8");


                        Looper.prepare();
                        Toast.makeText(mContext, mResult, Toast.LENGTH_LONG).show();
                        Looper.loop();


                    } catch (Exception e) {
                        e.printStackTrace();
                    }
                }
            })).start();
            break;
        case R.id.button2:
            String host = "localhost";
            int port = 4035;
            String appName = getResources().getString(R.string.app_name);
            AuthProcesser.asyncAuthorize(host, port, false, getPackageName(),
              appName, scopes, mAuthHandler);
            break;
        }
    }


    /**
     * Local OAuth のリスナー.
     */
    private AuthorizationHandler mAuthHandler =  new AuthorizationHandler() {
        @Override
        public void onAuthorized(final String clientId, final String clientSecret,
```

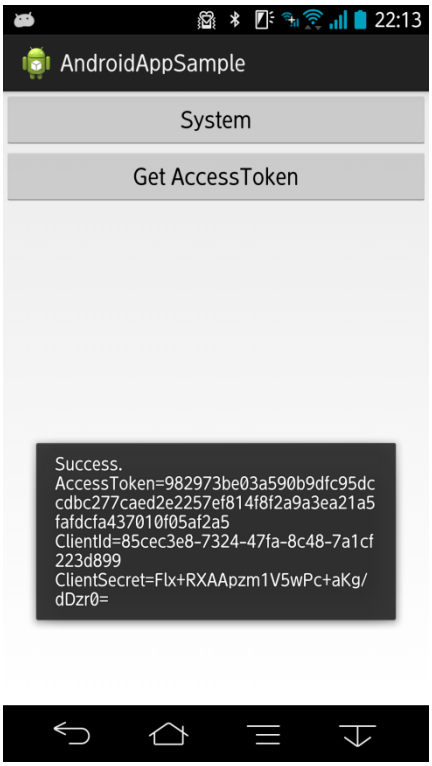
```
                    final String accessToken) {
            mClientId = clientId;
            mClientSecret = clientSecret;
            mAccessToken = accessToken;
            mError = null;


            MainActivity.this.runOnUiThread(new Runnable() {
                @Override
                public void run() {
                    Toast.makeText(MainActivity.this, "Success. AccessToken="
                                   + mAccessToken + " ClientId=" + clientId
                                   + " ClientSecret=" + clientSecret,
                                   Toast.LENGTH_LONG).show();
                }
            });
        }
        @Override
        public void onAuthFailed(final ErrorCode error) {
            mError = error;


            mClientId = null;
            mClientSecret = null;
            mAccessToken = null;


            MainActivity.this.runOnUiThread(new Runnable() {
                @Override
                public void run() {
                    Toast.makeText(MainActivity.this, error.toString(),
                       Toast.LENGTH_LONG).show();
                }
            });
        }
    };

}
```

ここまでの作成を終えた後、このサンプルをインストールして起動すると、以下のような画面が表示される。



「Get AccessToken」Button を押す事で、AccessToken 取得処理が実行され、認証画面が表示される。

認証を行うと、Toast で AccessToken が表示される。



3.2 NetworkServiceDiscovery の実装

デバイス一覧を取得するための NetworkServiceDiscovery の実装を行う。
画面にボタンを追加して、ボタン押下で NetworkServiceDiscovery 処理を行わせるように実装する。
また、Toast で表示していた応答を、EditText にて表示できるように変更も行う。

res/layout/main_activity.xml

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context="${relativePackage}.${activityClass}" >


    <Button android:id="@+id/button1"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:text="System" />


    <Button android:id="@+id/button2"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:layout_alignParentLeft="true"
        android:layout_below="@+id/button1"
        android:text="Get AccessToken" />


    <Button android:id="@+id/button3"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:layout_alignParentLeft="true"
        android:layout_below="@+id/button2"
        android:text="Network Service Discovery" />


    <EditText
        android:id="@+id/editText1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentBottom="true"
        android:layout_alignParentLeft="true"
        android:layout_alignParentRight="true"
        android:ems="10"
        android:inputType="textMultiLine" >


        <requestFocus />
    </EditText>


</RelativeLayout>
```

MainActivity.java

```
package com.example.androidappsample;


import java.io.IOException;
import java.net.URISyntaxException;
import java.util.ArrayList;
import java.util.List;
import java.util.Map;


import org.apache.http.HttpResponse;
import org.apache.http.client.methods.HttpGet;
import org.apache.http.client.methods.HttpUriRequest;
import org.apache.http.impl.client.DefaultHttpClient;
import org.apache.http.util.EntityUtils;


import com.nttdocomo.dconnect.message.DConnectMessage;
import com.nttdocomo.dconnect.message.DConnectMessage.ErrorCode;
import com.nttdocomo.dconnect.message.basic.message.DConnectResponseMessage;
import com.nttdocomo.dconnect.message.http.impl.factory.HttpMessageFactory;
import com.nttdocomo.dconnect.profile.AuthorizationProfileConstants;
import com.nttdocomo.dconnect.profile.BatteryProfileConstants;
import com.nttdocomo.dconnect.profile.ConnectProfileConstants;
import com.nttdocomo.dconnect.profile.DeviceOrientationProfileConstants;
import com.nttdocomo.dconnect.profile.FileDescriptorProfileConstants;
import com.nttdocomo.dconnect.profile.FileProfileConstants;
import com.nttdocomo.dconnect.profile.MediaPlayerProfileConstants;
import com.nttdocomo.dconnect.profile.MediaStreamRecordingProfileConstants;
```

```
import com.nttdocomo.dconnect.profile.NetworkServiceDiscoveryProfileConstants;
import com.nttdocomo.dconnect.profile.NotificationProfileConstants;
import com.nttdocomo.dconnect.profile.PhoneProfileConstants;
import com.nttdocomo.dconnect.profile.ProximityProfileConstants;
import com.nttdocomo.dconnect.profile.SettingsProfileConstants;
import com.nttdocomo.dconnect.profile.SystemProfileConstants;
import com.nttdocomo.dconnect.profile.VibrationProfileConstants;
import com.nttdocomo.dconnect.utils.AuthProcesser;
import com.nttdocomo.dconnect.utils.URIBuilder;
import com.nttdocomo.dconnect.utils.AuthProcesser.AuthorizationHandler;


import android.app.Activity;
import android.content.Context;
import android.os.Bundle;
import android.os.Looper;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;


public class MainActivity extends Activity implements OnClickListener {
    private Button mButton1;
    private Button mButton2;
    private Button mButton3;
    private Context mContext;
    private EditText mEditText1;


    private String mClientId;
    private String mClientSecret;
    private String mAccessToken;
    private ErrorCode mError;


    private List<SmartDevice> devices = null;


    /**
     * Local OAuth に使用するスコープ一覧.
     */
    private String[] scopes = {
        AuthorizationProfileConstants.PROFILE_NAME,
        BatteryProfileConstants.PROFILE_NAME,
        ConnectProfileConstants.PROFILE_NAME,
        DeviceOrientationProfileConstants.PROFILE_NAME,
        FileDescriptorProfileConstants.PROFILE_NAME,
        FileProfileConstants.PROFILE_NAME,
        MediaPlayerProfileConstants.PROFILE_NAME,
        MediaStreamRecordingProfileConstants.PROFILE_NAME,
        NetworkServiceDiscoveryProfileConstants.PROFILE_NAME,
        NotificationProfileConstants.PROFILE_NAME,
        PhoneProfileConstants.PROFILE_NAME,
        ProximityProfileConstants.PROFILE_NAME,
        SettingsProfileConstants.PROFILE_NAME,
        SystemProfileConstants.PROFILE_NAME,
        VibrationProfileConstants.PROFILE_NAME,


        // 独自プロファイル
        "light",
        "camera",
        "temperature",
        "dice",
        "sphero",
        "drive_controller",
        "remote_controller",
        "mhealth",


        // テスト用
        "*"
    };


    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        mContext = this.getBaseContext();
        mEditText1 = (EditText) findViewById(R.id.editText1);


        mButton1 = (Button) findViewById(R.id.button1);
        mButton2 = (Button) findViewById(R.id.button2);
        mButton3 = (Button) findViewById(R.id.button3);
        mButton1.setOnClickListener(this);
        mButton2.setOnClickListener(this);
        mButton3.setOnClickListener(this);
```

```
    }

    public void setTextField(String text) {
        mEditText1.setText(text);
    }

    @Override
    public void onClick(View v) {
        switch(v.getId()) {
        case R.id.button1:
            (new Thread(new Runnable() {
                @Override
                public void run() {
                    try {
                        String url = "http://localhost:4035/gotapi/system";
                        HttpGet mMethod = new HttpGet(url);
                        DefaultHttpClient mClient = new DefaultHttpClient();
                        HttpResponse response = mClient.execute(mMethod);

                        final String mResult = EntityUtils.toString(response.getEntity(), "UTF-8");

                        Looper.prepare();
                        Toast.makeText(mContext, mResult, Toast.LENGTH_LONG).show();
                        runOnUiThread(new Runnable() {
                            public void run() {
                                mEditText1.setText(mResult);
                            }
                        });
                        Looper.loop();

                    } catch (Exception e) {
                        e.printStackTrace();
                    }
                }
            })).start();
            break;
        case R.id.button2:
            String host = "localhost";
            int port = 4035;
            String appName = getResources().getString(R.string.app_name);
            AuthProcesser.asyncAuthorize(host, port, false, getPackageName(), appName, scopes, mAuthHandler);
            break;
        case R.id.button3:
            (new Thread(new Runnable() {
                @Override
                public void run() {
                    DConnectMessage message = new DConnectResponseMessage(DConnectMessage.RESULT_ERROR);
                    if (devices != null) {
                        devices = null;
                    }
                    devices = new ArrayList<SmartDevice>();

                    try {
                        URIBuilder builder = new URIBuilder();
                        builder.setProfile(NetworkServiceDiscoveryProfileConstants.PROFILE_NAME);

builder.setAttribute(NetworkServiceDiscoveryProfileConstants.ATTRIBUTE_GET_NETWORK_SERVICES);
                        builder.setScheme("http");
                        builder.setHost("localhost");
                        builder.setPort(4035);
                        builder.addParameter(DConnectMessage.EXTRA_ACCESS_TOKEN, mAccessToken);

                        HttpUriRequest request = new HttpGet(builder.build());
                        DefaultHttpClient mClient = new DefaultHttpClient();
                        HttpResponse response = mClient.execute(request);
                        message = (new HttpMessageFactory()).newDConnectMessage(response);
                    } catch (URISyntaxException e) {
                        e.printStackTrace();
                        return;
                    } catch (IOException e) {
                        e.printStackTrace();
                        return;
                    }

                    if (message == null) {
                        return;
                    }

                    int result = message.getInt(DConnectMessage.EXTRA_RESULT);
                    if (result == DConnectMessage.RESULT_ERROR) {
                        return;
                    }
```

```
                    List<Object> services = message.getList(
                            NetworkServiceDiscoveryProfileConstants.PARAM_SERVICES);
                   final StringBuffer sb = new StringBuffer();
                    if (services != null) {
                        for (Object object: services) {
                            @SuppressWarnings("unchecked")
                            Map<String, Object> service = (Map<String, Object>) object;
                            SmartDevice device = new SmartDevice(
                                service.get(NetworkServiceDiscoveryProfileConstants.PARAM_ID).toString(),
                                service.get(NetworkServiceDiscoveryProfileConstants.PARAM_NAME).toString());
                            devices.add(device);
                            sb.append("id:" +
service.get(NetworkServiceDiscoveryProfileConstants.PARAM_ID).toString() + ", ");
                            sb.append("name:" +
service.get(NetworkServiceDiscoveryProfileConstants.PARAM_NAME).toString() + ", ");
                        }
                    }
                    runOnUiThread(new Runnable() {
                        public void run() {
                            mEditText1.setText(sb);
                        }
                    });
                }
            })).start();
            break;
        }
    }

    /**
     * Local OAuth のリスナー.
     */
    private AuthorizationHandler mAuthHandler =  new AuthorizationHandler() {
        @Override
        public void onAuthorized(final String clientId, final String clientSecret, final String accessToken) {
            mClientId = clientId;
            mClientSecret = clientSecret;
            mAccessToken = accessToken;
            mError = null;


            MainActivity.this.runOnUiThread(new Runnable() {
                @Override
                public void run() {
                    Toast.makeText(MainActivity.this, "Success. AccessToken=" + mAccessToken + " ClientId=" +
clientId + " ClientSecret=" + clientSecret, Toast.LENGTH_LONG).show();
                    mEditText1.setText("Success. AccessToken=" + mAccessToken + " ClientId=" + clientId + "
ClientSecret=" + clientSecret);


                }
            });
        }
        @Override
        public void onAuthFailed(final ErrorCode error) {
            mError = error;


            mClientId = null;
            mClientSecret = null;
            mAccessToken = null;


            MainActivity.this.runOnUiThread(new Runnable() {
                @Override
                public void run() {
                    Toast.makeText(MainActivity.this, error.toString(), Toast.LENGTH_LONG).show();
                }
            });
        }
    };


}
```

SmartDevice.java

```
package com.example.androidappsample;


import java.util.ArrayList;
import java.util.List;


import android.os.Parcel;
import android.os.Parcelable;


/**
 * d-Connect スマートデバイス.
 */
public class SmartDevice implements Parcelable {


    /**
     * Parcelable クリエイター.
     */
    public static final Parcelable.Creator<SmartDevice> CREATOR =
            new Parcelable.Creator<SmartDevice>() {
        public SmartDevice createFromParcel(final Parcel in) {
            return new SmartDevice(in);
        }
        public SmartDevice[] newArray(final int size) {
            return new SmartDevice[size];
        }
    };


    /**
     * デバイス名.
     */
    private String mName;


    /**
     * デバイス種別.
     */
    private String mType;


    /**
     * デバイス ID.
     */
    private String mId;


    /**
     * サービスリスト.
     */
    private List<SmartService> mServiceList = new ArrayList<SmartService>();


    /**
     * コンストラクタ.
     * @param id デバイス ID
     * @param name デバイス名
     */
    public SmartDevice(final String id, final String name) {
        setId(id);
        setName(name);
    }


    /**
     * Parcelable コンストラクタ.
     * @param in 入力
     */
    private SmartDevice(final Parcel in) {
        setName(in.readString());
        setType(in.readString());
        setId(in.readString());
    }


    @Override
    public int describeContents() {
        return 0;
    }


    @Override
    public void writeToParcel(final Parcel dest, final int flags) {
        dest.writeString(mName);
        dest.writeString(mType);
```

```
        dest.writeString(mId);
    }

    @Override
    public String toString() {
        return mName;
    }

    /**
     * デバイス ID を設定する.
     * @param id デバイス ID
     */
    public void setId(final String id) {
        mId = id;
    }

    /**
     * デバイス名を設定する.
     * @param name デバイス名
     */
    public void setName(final String name) {
        mName = name;
    }

    /**
     * デバイス種別を設定する.
     * @param type デバイス種別
     */
    public void setType(final String type) {
        mType = type;
    }

    /**
     * デバイス名.
     * @return デバイス名
     */
    public String getName() {
        return mName;
    }

    /**
     * デバイス種別を取得する.
     * @return デバイス種別
     */
    public String getType() {
        return mType;
    }

    /**
     * デバイス ID を取得する.
     * @return デバイス ID
     */
    public String getId() {
        return mId;
    }

    /**
     * サービスリストを取得する.
     * @return サービスリスト
     */
    public List<SmartService> getServiceList() {
        return mServiceList;
    }

    /**
     * サービスを追加する.
     * @param service サービス
     */
    public void addService(final SmartService service) {
        mServiceList.add(service);
    }

    /**
     * サービスを削除する.
     * @param service サービス
     */
    public void removeService(final SmartService service) {
        mServiceList.remove(service);
    }
```

```
}
```

SmartService.java

```
package com.example.androidappsample;


/**
 * スマートサービス(プロファイル).
 */
public class SmartService {


    /**
     * プロファイル名.
     */
    private String mName;


    /**
     * コンストラクタ.
     * @param name プロファイル名
     */
    public SmartService(final String name) {
        mName = name;
    }


    @Override
    public String toString() {
        return mName;
    }


    /**
     * プロファイル名を設定する.
     * @param name プロファイル名
     */
    public void setName(final String name) {
        mName = name;
    }


    /**
     * プロファイル名を取得する.
     * @return プロファイル名
     */
    public String getName() {
        return mName;
    }


    /**
     * アイコンを取得する.
     * @return アイコンID
     */
    public int getIconId() {
        return android.R.drawable.ic_menu_info_details;
    }


}
```

ここまでの作成を終えた後、このサンプルをインストールして起動すると、以下のような画面が表示される。

「Network Service Discovery」Button を押す事で、Network Service Discovery 処理が実行され、デバイスの情報が表示される。

3.3 デバイスプラグインの RESTful での問い合わせ処理実装

ここまでの実装で取得した「AccessToken」「DeviceId」を使用して、デバイスプラグインへ RESTful での問い合わせ処理の実装を行う。
ここでは、画面にボタンを追加して、ボタン押下で Network Service Discovery で最初に発見されたデバイスプラグインに対して、「/System/device」を実行し、その応答を得る実装を行う。

res/layout/main_activity.xml

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context="${relativePackage}.${activityClass}" >


    <Button android:id="@+id/button1"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:text="System" />


    <Button android:id="@+id/button2"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:layout_alignParentLeft="true"
        android:layout_below="@+id/button1"
        android:text="Get AccessToken" />


    <Button android:id="@+id/button3"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:layout_alignParentLeft="true"
        android:layout_below="@+id/button2"
        android:text="Network Service Discovery" />


    <Button android:id="@+id/button4"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:layout_alignParentLeft="true"
        android:layout_below="@+id/button3"
        android:text="Device" />


    <EditText
        android:id="@+id/editText1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentBottom="true"
        android:layout_alignParentLeft="true"
        android:layout_alignParentRight="true"
        android:ems="10"
        android:inputType="textMultiLine" >


        <requestFocus />
    </EditText>


</RelativeLayout>
```

MainActivity.java

```
 package com.example.androidappsample;


import java.io.IOException;
import java.net.URISyntaxException;
import java.util.ArrayList;
import java.util.List;
import java.util.Map;


import org.apache.http.HttpResponse;
import org.apache.http.client.methods.HttpGet;
import org.apache.http.client.methods.HttpUriRequest;
import org.apache.http.impl.client.DefaultHttpClient;
import org.apache.http.util.EntityUtils;


import com.nttdocomo.dconnect.message.DConnectMessage;
import com.nttdocomo.dconnect.message.DConnectMessage.ErrorCode;
import com.nttdocomo.dconnect.message.basic.message.DConnectResponseMessage;
import com.nttdocomo.dconnect.message.http.impl.factory.HttpMessageFactory;
```

```
import com.nttdocomo.dconnect.profile.AuthorizationProfileConstants;
import com.nttdocomo.dconnect.profile.BatteryProfileConstants;
import com.nttdocomo.dconnect.profile.ConnectProfileConstants;
import com.nttdocomo.dconnect.profile.DeviceOrientationProfileConstants;
import com.nttdocomo.dconnect.profile.FileDescriptorProfileConstants;
import com.nttdocomo.dconnect.profile.FileProfileConstants;
import com.nttdocomo.dconnect.profile.MediaPlayerProfileConstants;
import com.nttdocomo.dconnect.profile.MediaStreamRecordingProfileConstants;
import com.nttdocomo.dconnect.profile.NetworkServiceDiscoveryProfileConstants;
import com.nttdocomo.dconnect.profile.NotificationProfileConstants;
import com.nttdocomo.dconnect.profile.PhoneProfileConstants;
import com.nttdocomo.dconnect.profile.ProximityProfileConstants;
import com.nttdocomo.dconnect.profile.SettingsProfileConstants;
import com.nttdocomo.dconnect.profile.SystemProfileConstants;
import com.nttdocomo.dconnect.profile.VibrationProfileConstants;
import com.nttdocomo.dconnect.utils.AuthProcesser;
import com.nttdocomo.dconnect.utils.URIBuilder;
import com.nttdocomo.dconnect.utils.AuthProcesser.AuthorizationHandler;


import android.app.Activity;
import android.content.Context;
import android.os.Bundle;
import android.os.Looper;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;


public class MainActivity extends Activity implements OnClickListener {
    private Button mButton1;
    private Button mButton2;
    private Button mButton3;
    private Button mButton4;
    private Context mContext;
    private EditText mEditText1;


    private String mClientId;
    private String mClientSecret;
    private String mAccessToken;
    private ErrorCode mError;


    private List<SmartDevice> devices = null;


    /**
     * Local OAuth に使用するスコープ一覧.
     */
    private String[] scopes = {
        AuthorizationProfileConstants.PROFILE_NAME,
        BatteryProfileConstants.PROFILE_NAME,
        ConnectProfileConstants.PROFILE_NAME,
        DeviceOrientationProfileConstants.PROFILE_NAME,
        FileDescriptorProfileConstants.PROFILE_NAME,
        FileProfileConstants.PROFILE_NAME,
        MediaPlayerProfileConstants.PROFILE_NAME,
        MediaStreamRecordingProfileConstants.PROFILE_NAME,
        NetworkServiceDiscoveryProfileConstants.PROFILE_NAME,
        NotificationProfileConstants.PROFILE_NAME,
        PhoneProfileConstants.PROFILE_NAME,
        ProximityProfileConstants.PROFILE_NAME,
        SettingsProfileConstants.PROFILE_NAME,
        SystemProfileConstants.PROFILE_NAME,
        VibrationProfileConstants.PROFILE_NAME,


        // 独自プロファイル
        "light",
        "camera",
        "temperature",
        "dice",
        "sphero",
        "drive_controller",
        "remote_controller",
        "mhealth",


        // テスト用
        "*"
    };


    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        mContext = this.getBaseContext();
```

```
        mEditText1 = (EditText) findViewById(R.id.editText1);


        mButton1 = (Button) findViewById(R.id.button1);
        mButton2 = (Button) findViewById(R.id.button2);
        mButton3 = (Button) findViewById(R.id.button3);
        mButton4 = (Button) findViewById(R.id.button4);
        mButton1.setOnClickListener(this);
        mButton2.setOnClickListener(this);
        mButton3.setOnClickListener(this);
        mButton4.setOnClickListener(this);
    }


    public void setTextField(String text) {
        mEditText1.setText(text);
    }


    @Override
    public void onClick(View v) {
        switch(v.getId()) {
        case R.id.button1:
            (new Thread(new Runnable() {
                @Override
                public void run() {
                    try {
                        String url = "http://localhost:4035/gotapi/system";
                        HttpGet mMethod = new HttpGet(url);
                        DefaultHttpClient mClient = new DefaultHttpClient();
                        HttpResponse response = mClient.execute(mMethod);


                        final String mResult = EntityUtils.toString(response.getEntity(), "UTF-8");


                        Looper.prepare();
                        Toast.makeText(mContext, mResult, Toast.LENGTH_LONG).show();
                        runOnUiThread(new Runnable() {
                            public void run() {
                                mEditText1.setText(mResult);
                            }
                        });
                        Looper.loop();


                    } catch (Exception e) {
                        e.printStackTrace();
                    }
                }
            })).start();
            break;
        case R.id.button2:
            String host = "localhost";
            int port = 4035;
            String appName = getResources().getString(R.string.app_name);
            AuthProcesser.asyncAuthorize(host, port, false, getPackageName(), appName, scopes, mAuthHandler);
            break;
        case R.id.button3:
            (new Thread(new Runnable() {
                @Override
                public void run() {
                    DConnectMessage message = new DConnectResponseMessage(DConnectMessage.RESULT_ERROR);
                    if (devices != null) {
                        devices = null;
                    }
                    devices = new ArrayList<SmartDevice>();


                    try {
                        URIBuilder builder = new URIBuilder();
                        builder.setProfile(NetworkServiceDiscoveryProfileConstants.PROFILE_NAME);

builder.setAttribute(NetworkServiceDiscoveryProfileConstants.ATTRIBUTE_GET_NETWORK_SERVICES);
                        builder.setScheme("http");
                        builder.setHost("localhost");
                        builder.setPort(4035);
                        builder.addParameter(DConnectMessage.EXTRA_ACCESS_TOKEN, mAccessToken);


                        HttpUriRequest request = new HttpGet(builder.build());
                        DefaultHttpClient mClient = new DefaultHttpClient();
                        HttpResponse response = mClient.execute(request);
                        message = (new HttpMessageFactory()).newDConnectMessage(response);
                    } catch (URISyntaxException e) {
                        e.printStackTrace();
                        return;
                    } catch (IOException e) {
                        e.printStackTrace();
                        return;
                    }
```

```
                    if (message == null) {
                        return;
                    }

                    int result = message.getInt(DConnectMessage.EXTRA_RESULT);
                    if (result == DConnectMessage.RESULT_ERROR) {
                        return;
                    }

                    List<Object> services = message.getList(
                            NetworkServiceDiscoveryProfileConstants.PARAM_SERVICES);
                   final StringBuffer sb = new StringBuffer();
                    if (services != null) {
                        for (Object object: services) {
                            @SuppressWarnings("unchecked")
                            Map<String, Object> service = (Map<String, Object>) object;
                            SmartDevice device = new SmartDevice(
                                service.get(NetworkServiceDiscoveryProfileConstants.PARAM_ID).toString(),
                                service.get(NetworkServiceDiscoveryProfileConstants.PARAM_NAME).toString());
                            devices.add(device);
                            sb.append("id:" +
service.get(NetworkServiceDiscoveryProfileConstants.PARAM_ID).toString() + ", ");
                            sb.append("name:" +
service.get(NetworkServiceDiscoveryProfileConstants.PARAM_NAME).toString() + ", ");
                        }
                    }
                    runOnUiThread(new Runnable() {
                        public void run() {
                            mEditText1.setText(sb);
                        }
                    });
                }
            })).start();
            break;
        case R.id.button4:
            (new Thread(new Runnable() {
                @Override
                public void run() {
                    DConnectMessage message = new DConnectResponseMessage(DConnectMessage.RESULT_ERROR);

                    URIBuilder uriBuilder = new URIBuilder();
                    uriBuilder.setProfile(SystemProfileConstants.PROFILE_NAME);
                    uriBuilder.setAttribute(SystemProfileConstants.ATTRIBUTE_DEVICE);
                    uriBuilder.setScheme("http");
                    uriBuilder.setHost("localhost");
                    uriBuilder.setPort(4035);
                    uriBuilder.addParameter(DConnectMessage.EXTRA_DEVICE_ID, devices.get(0).getId());
                    uriBuilder.addParameter(DConnectMessage.EXTRA_ACCESS_TOKEN, mAccessToken);

                    try {
                        HttpUriRequest req = new HttpGet(uriBuilder.build());
                        DefaultHttpClient mCli = new DefaultHttpClient();
                        HttpResponse res = mCli.execute(req);
                        final String mResult = EntityUtils.toString(res.getEntity(), "UTF-8");
                        runOnUiThread(new Runnable() {
                            public void run() {
                                mEditText1.setText(mResult);
                            }
                        });
                    } catch (IOException e) {
                        e.printStackTrace();
                    } catch (URISyntaxException e) {
                        e.printStackTrace();
                    }

                    int result = message.getInt(DConnectMessage.EXTRA_RESULT);
                    if (result == DConnectMessage.RESULT_ERROR) {
                        return;
                    }
                }
            })).start();
            break;
        }
    }

    /**
     * Local OAuth のリスナー.
     */
    private AuthorizationHandler mAuthHandler =  new AuthorizationHandler() {
        @Override
        public void onAuthorized(final String clientId, final String clientSecret, final String accessToken) {
            mClientId = clientId;
            mClientSecret = clientSecret;
```

```
            mAccessToken = accessToken;
            mError = null;


            MainActivity.this.runOnUiThread(new Runnable() {
                @Override
                public void run() {
                    Toast.makeText(MainActivity.this, "Success. AccessToken=" + mAccessToken + " ClientId=" +
clientId + " ClientSecret=" + clientSecret, Toast.LENGTH_LONG).show();
                    mEditText1.setText("Success. AccessToken=" + mAccessToken + " ClientId=" + clientId + "
ClientSecret=" + clientSecret);


                }
            });
        }
        @Override
        public void onAuthFailed(final ErrorCode error) {
            mError = error;


            mClientId = null;
            mClientSecret = null;
            mAccessToken = null;


            MainActivity.this.runOnUiThread(new Runnable() {
                @Override
                public void run() {
                    Toast.makeText(MainActivity.this, error.toString(), Toast.LENGTH_LONG).show();
                }
            });
        }
    };


}
```

ここまでの作成を終えた後、このサンプルをインストールして起動すると、以下のような画面が表示される。

「Get AccessToken」Button を押下し、認証を行い、AccessToken を取得する。

「Network Service Discovery」Button を押下し、デバイス一覧を取得する。

「Device」Button を押下し、デバイスの情報を取得する。この例では、サンプルのデバイスプラグインのデバイス情報が表示される。

—以上—