

# Android & Bluetooth

Juan Antonio Breña Moral

## Índice de contenido

Infraestructura.....	1
Creando el primer proyecto en Android.....	1
Ejemplo1.....	1
Estructura de ficheros de un proyecto en Android.....	1
Codigo fuente.....	2
Bluetooth.....	3
FAQ.....	3
Debug certificate expired.....	3

## Infraestructura

<http://www.eclipse.org/downloads/>

<http://developer.android.com/sdk/index.html>

<http://developer.android.com/sdk/eclipse-adt.html>

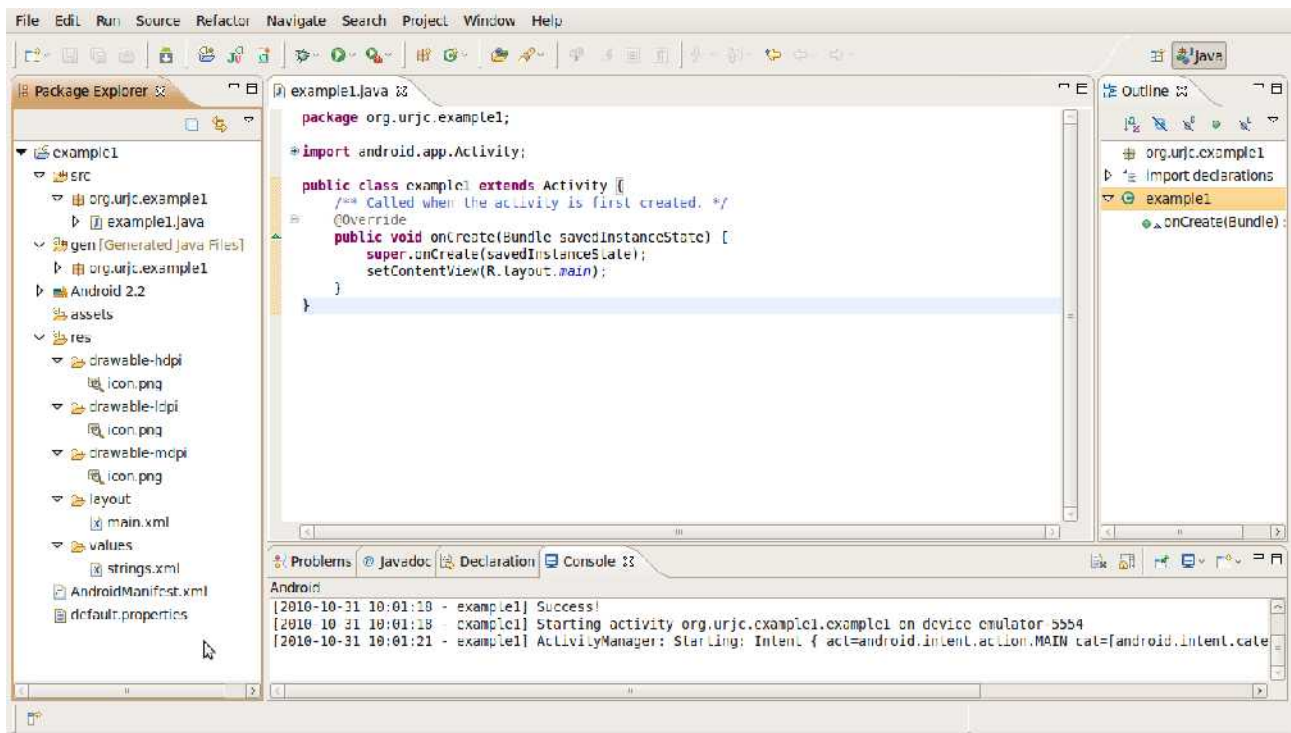
<https://dl-ssl.google.com/android/eclipse/>

## Creando el primer proyecto en Android

### *Ejemplo1*

En el primer ejemplo, se creara un proyecto desde 0 y se aprendera los siguientes conceptos:

## Estructura de ficheros de un proyecto en Android



## Codigo fuente

### Clase

```
package org.urjc.example1;
```

```
import android.app.Activity;
```

```
import android.os.Bundle;
```

```
public class example1 extends Activity {
    /** Called when the activity is first created. */
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);
    }
}
```

### UI

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
```

```

        android:orientation="vertical"
        android:layout_width="fill_parent"
        android:layout_height="fill_parent"
    >
<TextView
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:text="@string/hello"
/>
</LinearLayout>

```

## Bluetooth

<http://developer.android.com/guide/topics/wireless/bluetooth.html>

```

package jab.android.essentials;

import jab.lejos.gps.*;
import android.app.Activity;
import android.app.Dialog;
import android.bluetooth.BluetoothAdapter;
import android.bluetooth.BluetoothDevice;
import android.bluetooth.BluetoothSocket;
import android.content.Context;
import android.hardware.Sensor;
import android.hardware.SensorEvent;
import android.hardware.SensorEventListener;
import android.hardware.SensorManager;
import android.os.Bundle;
import android.view.Menu;
import android.view.MenuItem;
import android.view.View;
import android.content.Intent;
import android.content.res.Resources;
import android.net.Uri;
import android.util.Log;
import android.view.View.OnClickListener;
import android.view.Window;
import android.widget.Button;
import android.widget.EditText;
import android.widget.SeekBar;
import android.widget.SeekBar.OnSeekBarChangeListener;
import android.widget.TextView;
import android.widget.Toast;

import java.io.DataInputStream;
import java.io.IOException;
import java.io.PrintWriter;
import java.io.DataOutputStream;
import java.net.*;

```

```

import java.util.UUID;
import java.util.List;
import java.util.Enumeration;
import java.util.Set;

public class RFCommConnection extends Activity
{
    public static final String DEFAULT_NXT_NAME = "GSYC";

    public static final int ACTION=10;
    public static final int DISCONNECT = 99;

    public static final int MENU_ABOUT = Menu.FIRST;
    public static final int MENU_QUIT = Menu.FIRST + 1;

    private SensorManager sensorManager;
    private boolean runWithEmulator = false;
    private SeekBar pitchSeekBar;
    private SeekBar rollSeekBar;
    private Button connectButton;
    private Button actionButton;
    private TextView myNXT;
    private BluetoothSocket nxtBTsocket = null;
    private DataOutputStream nxtDos = null;
    private DataInputStream nxtDis = null;

    private GPS gps;

    private long timeDataSent = 0;

    /** Called when the activity is first created. */
    @Override
    public void onCreate(Bundle savedInstanceState)
    {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);
        sensorManager = (SensorManager) getSystemService(Context.SENSOR_SERVICE);
        myNXT = (TextView) findViewById(R.id.btName);
        myNXT.setText(DEFAULT_NXT_NAME);
        initSeekBars();
        initConnectButton();
        initActionButton();
    }

    private void initConnectButton() {
        connectButton = (Button) findViewById(R.id.connect_button);
        connectButton.setOnClickListener(new OnClickListener() {
            public void onClick(View v) {
                if (nxtBTsocket == null)
                    createBTConnection();
                else
                    destroyNXTConnection();
            }
        });
    }
}

```

```

        if (nxtBTsocket == null) {
            connectButton.setText(getResources().getString(R.string.connect));
            actionButton.setEnabled(false);
        }
        else {
            connectButton.setText(getResources().getString(R.string.disconnect));
            actionButton.setEnabled(true);
        }
    }
});
}

private void initActionButton() {
    actionButton = (Button) findViewById(R.id.action_button);
    actionButton.setOnClickListener(new OnClickListener() {
        public void onClick(View v) {

            Log.e(ACTIVITY_SERVICE, "NSat: " + gps.getSatellitesTracked());
            Log.e(ACTIVITY_SERVICE, "Altitude: " + gps.getAltitude());
            Log.e(ACTIVITY_SERVICE, "Latitude: " + gps.getLatitude());
            Log.e(ACTIVITY_SERVICE, "Longitude: " + gps.getLongitude());
            Date d = gps.getDate();
            Log.e(ACTIVITY_SERVICE, "Date: " + d.getYear() + "/" + d.getMonth() + "/" +
d.getDay() + " - " + d.getHours() + ":" + d.getMinutes() + ":" + d.getSeconds());
        }
    });
}

private void initSeekBar() {
    pitchSeekBar = (SeekBar) findViewById(R.id.seekbar1);
    rollSeekBar = (SeekBar) findViewById(R.id.seekbar2);
    pitchSeekBar.setOnSeekBarChangeListener(new OnSeekBarChangeListener() {

        public void onProgressChanged(SeekBar s, int progress, boolean fromUser) {
            if (runWithEmulator)
                updateOrientation((float) 0.0, (float)((progress-50.0)*30.0/50.0), (float)
((rollSeekBar.getProgress()-50.0)*30.0/50.0), false);
        }

        public void onStartTrackingTouch(SeekBar s) {
        }

        public void onStopTrackingTouch(SeekBar s) {
        }

    });
    rollSeekBar.setOnSeekBarChangeListener(new OnSeekBarChangeListener() {

        public void onProgressChanged(SeekBar s, int progress, boolean fromUser) {
            if (runWithEmulator)
                updateOrientation((float) 0.0, (float)((pitchSeekBar.getProgress()-50.0)*30.0/50.0),

```

```
(float)((progress-50.0)*30.0/50.0), false);
}
```

```
public void onStartTrackingTouch(SeekBar s) {
}
```

```
public void onStopTrackingTouch(SeekBar s) {
}
```

```
});
}
```

```
private final SensorEventListener orientationSensorEventListener = new SensorEventListener() {
```

```
public void onSensorChanged(SensorEvent event) {
    updateOrientation(event.values[0],
        event.values[1],
        event.values[2],
        true);
}
```

```
public void onAccuracyChanged(Sensor sensor, int accuracy) {
}
};
```

```
private void updateOrientation(float heading, float pitch, float roll, boolean fromSensor) {
```

```
    long currentTime;
```

```
    // show position at the seekbars
```

```
    if (fromSensor) {
        pitchSeekBar.setProgress((int) (pitch*50.0/30.0 + 50.5));
        rollSeekBar.setProgress((int) (roll*50.0/30.0 + 50.5));
    }
```

```
    // send values to every 100 msecs
```

```
    if (nxtDos != null) {
        currentTime = System.currentTimeMillis();
        if ((currentTime - timeDataSent) > 100) {
            timeDataSent = currentTime;
            if (Math.abs(pitch) < 10) {
                return;
            }
        }
    }
```

```
    // calculate motor values
```

```
    int left = (int) Math.round(20.0 * pitch * (1.0 + roll / 90.0));
    int right = (int) Math.round(20.0 * pitch * (1.0 - roll / 90.0));
```

```
    int reverseLeft = left < 0 ? 1 : 0;
    left = Math.abs(left);
    if (left > 700) {
```

```

        left = 700;
    }
    int reverseRight = right < 0 ? 1 : 0;
    right = Math.abs(right);
    if (right > 700) {
        right = 700;
    }
}
}
}

```

```

@Override
public void onResume() {
    List<Sensor> sensorList;

    super.onResume();
    connectButton.setText(getResources().getString(R.string.connect));
    actionButton.setEnabled(false);

    // register orientation sensor
    sensorList = sensorManager.getSensorList(Sensor.TYPE_ORIENTATION);
    runWithEmulator = (sensorList.size() == 0);
    if (!runWithEmulator)
        sensorManager.registerListener(orientationSensorEventListener, sensorList.get(0),
SensorManager.SENSOR_DELAY_GAME);
}

```

```

@Override
public void onPause() {
    destroyNXTConnection();
    sensorManager.unregisterListener(orientationSensorEventListener);
    super.onStop();
}

```

```

/* Creates the menu items */
public boolean onCreateOptionsMenu(Menu menu) {
    menu.add(0, MENU_ABOUT, 0, "About").setIcon(R.drawable.menu_info_icon);
    menu.add(0, MENU_QUIT, 0, "Quit").setIcon(R.drawable.menu_quit_icon);
    return true;
}

```

```

/* Handles item selections */
public boolean onOptionsItemSelected(MenuItem item) {
    switch (item.getItemId()) {
        case MENU_ABOUT:
            showAboutDialog();
            return true;
        case MENU_QUIT:
            finish();
            return true;
    }
    return false;
}

```

```

    }

    private void showAboutDialog()
    {
        final Dialog dialog = new Dialog(this);
        dialog.getWindow().requestFeature(Window.FEATURE_NO_TITLE);
        dialog setContentView(R.layout.aboutbox);
        dialog.show();
    }

    private void createBTConnection() {
        try {
            BluetoothAdapter btAdapter = BluetoothAdapter.getDefaultAdapter();
            Set<BluetoothDevice> bondedDevices = btAdapter.getBondedDevices();
            BluetoothDevice nxtDevice = null;

            for (BluetoothDevice bluetoothDevice : bondedDevices)
            {
                Log.i(TELEPHONY_SERVICE,bluetoothDevice.getName());
                Log.i(TELEPHONY_SERVICE,bluetoothDevice.getAddress());

                String GPSDevice = "SJA GPS";

                if (bluetoothDevice.getName().equals(GPSDevice)) {
                    //if (bluetoothDevice.getName().equals(myNXT.getText().toString())) {
                    nxtDevice = bluetoothDevice;
                    break;
                }
            }

            if (nxtDevice == null)
            {
                Toast toast = Toast.makeText(this, "No paired NXT device found",
                Toast.LENGTH_SHORT);
                toast.show();
                return;
            }

            nxtBTsocket =
            nxtDevice.createRfcommSocketToServiceRecord(UUID.fromString("00001101-0000-1000-8000-
            00805F9B34FB"));
            nxtBTsocket.connect();
            nxtDos = new DataOutputStream(nxtBTsocket.getOutputStream());
            //nxtDis = new DataInputStream();

            try{
                //in = btGPS.openInputStream();
                gps = new GPS(nxtBTsocket.getInputStream());
                gps.updateValues(true);//Update values always

                Log.e(CONNECTIVITY_SERVICE, "Processing");
            }catch(Exception e){

```



```

        Log.e(CONNECTIVITY_SERVICE, "Problems");
    }

    } catch (IOException e) {
        Toast toast = Toast.makeText(this, "Problem at creating a connection",
Toast.LENGTH_SHORT);
        toast.show();
    }
}

private void destroyNXTConnection() {
    try {
        if (nxtBTsocket != null) {
            // send one close message
            nxtBTsocket.close();
            nxtBTsocket = null;
        }
        nxtDis = null;
        nxtDos = null;
    } catch (IOException e) {
        Toast toast = Toast.makeText(this, "Problem at closing the connection",
Toast.LENGTH_SHORT);
        toast.show();
    }
}

public void sendBTcommand(int command, int value) {
    if (nxtDos == null) {
        return;
    }

    try {
        nxtDos.writeInt(command);
        nxtDos.writeInt(value);
        nxtDos.flush();
    } catch (IOException ioe) {
        Toast toast = Toast.makeText(this, "Problem at writing command",
Toast.LENGTH_SHORT);
        toast.show();
    }
}
}

```

## FAQ

### ***Debug certificate expired***

<http://developer.android.com/resources/faq/troubleshooting.html#signingcalendar>

<http://stackoverflow.com/questions/2194808/debug-certificate-expired-error-in-eclipse-android-plugins>

## ***GIT***

<http://www.eclipse.org/egit/>