May 15, 2011

Identifying Your Android USB Accessory (http://jeffreysambells.com/posts/2011/05/15/identifying-your-android-usb-accessory/)

When you create your first Android USB Accessory with the Android ADK, one of the first things you'll need to do is identify your accessory when you connect it to your Android device. Within the Arduino sketch for your accessory, you identify your accessory in the **`AndroidAccessory`** object.

```
AndroidAccessory(const char *manufacturer,
                 const char *model,
                 const char *description,
                 const char *version,
                 const char *uri,
                 const char *serial);
```

For example, the DemoKit app that comes with the ADK identifies itself as version 1.0 of the Google DemoKit accessory:

```
AndroidAccessory acc("Google, Inc.",
                 "DemoKit",
                 "DemoKit Arduino Board",
                 "1.0",
                 "http://www.android.com",
                 "0000000012345678");
```

You're own app might looks something like this:

```
AndroidAccessory acc("My App Company",
                 "CoolAccessory",
                 "CoolAccessory Arduino Board",
                 "1.0",
                 "http://www.example.com/CoolAccessory",
                 "0000000012345678");
```

The three key one's you'll need to remember are **manufacturer**, **model** and **version** as you'll need to define these in you're Android app so that the device can notify your app when your accessory is connected.

When you create your Android app, there's a few steps you need to take. First, not every Android-powered device supports the USB accessory APIs. In fact, the Android Open Accessory ADK is only supported by Android OS 3.1 and 2.3.4 platforms. To identify that your app has USB features, you need to include a **<uses-feature>** element in your manifest to declare that your application uses the **android.hardware.usb.accessory** feature.

```
<uses-feature android:name="android.hardware.usb.accessory" />
```

Next, if you're using the add-on library, add a **<uses-library>** element with **com.android.future.usb.accessory** for the USB accessory library.

```
<uses-library android:name="com.android.future.usb.accessory" />
```

and set the minimum SDK of the application to API Level 10. If you're using **theandroid.hardware.usb package** then the **<uses-library>** isn't necessary and you only need to set the minimum SDK to 12 .

Lastly, you'll want your application to be notified when your accessory is attached. Specify an **<intent-filter>** and **<meta-data>** element pair for the **android.hardware.usb.action.USB_ACCESSORY_ATTACHED** intent in your main activity. Overall your manifest should look something like this:

```
<manifest ...>
Â  Â  <uses-feature android:name="android.hardware.usb.accessory" />
Â  Â  <uses-sdk android:minSdkVersion="<version>" />
Â  Â  ...
Â  Â  <application>
Â  Â  Â    <uses-library android:name="com.android.future.usb.accessory" />
Â  Â  Â  Â  <activity ...>
Â  Â  Â  Â  Â  Â  ...
Â  Â  Â  Â  Â  Â  <intent-filter>
Â  Â  Â  Â  Â  Â  Â  Â  <action android:name="android.hardware.usb.action.USB_ACCESSORY_ATTACHED" />
Â  Â  Â  Â  Â  Â  </intent-filter>
Â  Â  Â  Â  Â  Â  <meta-data android:name="android.hardware.usb.action.USB_ACCESSORY_ATTACHED"
Â  Â  Â  Â  Â  Â  Â  Â  android:resource="@xml/accessory_filter" />
Â  Â  Â  Â  </activity>
Â  Â  </application>
</manifest>
```

The **<meta-data>** element references an XML resource file that includes the **manufacturer**, **model** and **version** you defined earlier in your Arduino sketch. Create an **accessory_filter.xml** resource

file in the **res/xml/** directory with a **<usb-accessory>** element to identify your accessory.

```
<resources>
    <usb-accessory model="CoolAccessory" manufacturer="My App Company" version="1.0"/>
</resources>
```

Now, when you connect your accessory to the device android will send an intent to open an appropriate application. The best part is that more than one application can respond to a given intent so multiple apps could optionally respond for the same accessory. ☆

Jeffrey Sambells does what he loves. He is a father, designer, developer, author and entrepreneur among many other things. He started dabbling in the web more than a decade ago and has turned it into a passion, pushing the limits of what's possible. With an expertise in creating slick end-to-end user experiences, Jeffrey is always on top of the latest technologies, especially when it comes to mobile devices. ☆

★ **@iamamused (http://twitter.com/iamamused)**
★ **About.me (http://about.me/iamamused)**
★ **Published Writing (http://jeffreysambells.com/books/)**
★ **GitHub (http://github.com/iamamused)**
★ **Contact (http://jeffreysambells.com/contact/)**
★ **Archives (http://jeffreysambells.com/archives/)**
★ **Subscribe (http://jeffreysambells.com/feed)**

Proudly powered by **WordPress (http://www.wordpress.org)** and **html5boilerplate (https://github.com/iamamused/wp-theme-html5-boilerplate/)**

Theme by **Jeffrey Sambells (http://www.jeffreysambells.com)**. **RSS Feed (http://jeffreysambells.com/feed/)**.