CUSTOMANDROIDAPI

USER MANUAL FOR
ANDROID LIBRARY CustomAndroidAPI

IMPORTANT NOTE

© CUSTOM S.p.A. All rights reserved

CUSTOM S.p.A.
Str. Berettine 2 - 43010 Fontevivo (PARMA) - Italy
Tel. : +39 0521-680111 -   Fax : +39 0521-610701
http: www.custom.biz   -   info@custom.biz

Customer Technical Support:
Email : support@custom.it

Rev. History	Description :	Date	Revision
	First Revision	11/04/2012	1.00
	Added support for new printers	15/10/2012	1.01
	Added support for new printers and for new communications ports (Ethernet and USB)	14/11/2012	1.02
	Add new features	03/10/2014	1.03
	Add new features	02/02/2015	1.04
	Add new features	25/09/2020	1.05
	Add new communication port (BLE)	01/09/2021	1.06

INDEX

Document Overview

This document details about CUSTOM android API for printers. And also explains how to use this API for android application development for CUSTOM printers.

About CUSTOM Android API

This API provides functions to access the CUSTOM printers via Bluetooth (using serial port profile - SPP), BLE (Bluetooth low energy), Ethernet / Wifi and USB.

It includes below functions to access the printer
- *Get the printer class for printer*
- *Connect to printer*
- *Close connection*
- *Set timeout for read/write*
- *Write / Read bytes to/from printer*
- *Print text with various text attributes*
- *Print image with image scaling options*
- *Cut, eject / retract and present the ticket*
- *Get picture, card and barcode from scanner*

Android Platform Compatibility

Android 2.2 or latest (For Ethernet and Bluetooth)
Android 3.1 or latest (For USB)
Android 6 or latest (For Bluetooth low energy)

API reference guide

Attached separately (HTML files format)

Prerequisite
- *You must be familiar with Android development environment using Eclipse IDE or Android Studio*
- *Android Platform version 3.1 or above (for USB), 2.2 or above (for Ethernet and Bluetooth) or 6 or above (for Bluetooth low energy)*
- *API Jar file: customandroidapi.jar*
- *Bluetooth or Ethernet / Wifi or USB support on Android device*
- *Bluetooth or Ethernet / Wifi or USB Permission for Android project*
- *Position support / permission for BLE (Bluetooth low energy)*

How to use API

Developer needs to add the API in Android project library and then developer can access the various API functions.

**Add the API to android project (Eclipse)**

Create, if it doesn't exist, the folder "libs", into the project
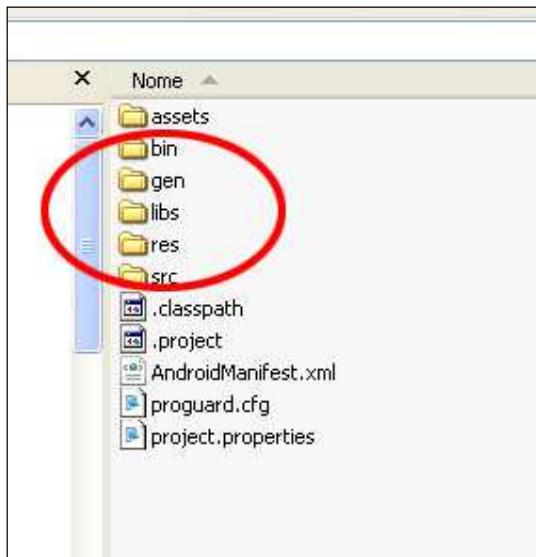
Copy into this folder, the API file "customandroidapi.jar"
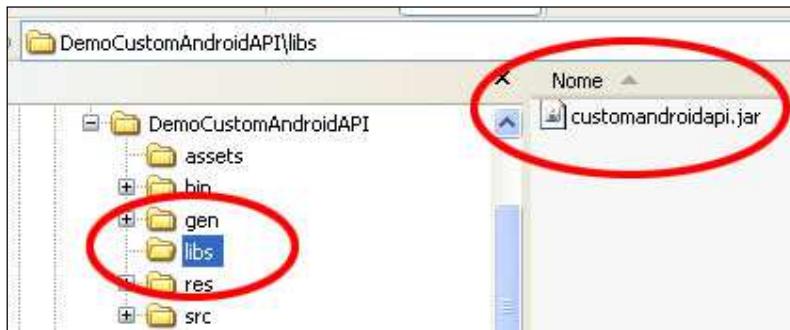
Load Project into Eclipse and Click "Clean…" into "Project" menu
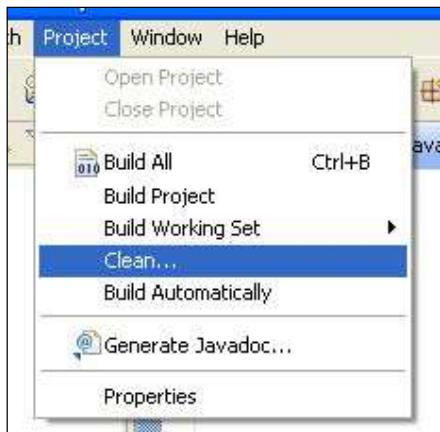
Now you will see the library into your Eclipse project
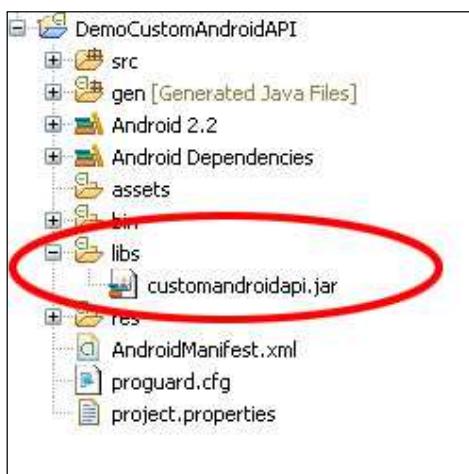
Android API is correctly added in project library. Now you can use this API.

Add the API to android project (Android Studio)

Create, if it doesn't exist, the folder "libs", into the "app" folder of the project project
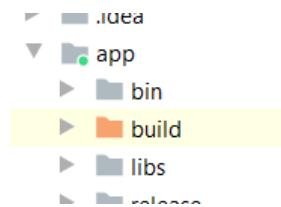
And copy, into this folder, the API file "customandroidapi.jar"

Add the library dependency to the module-level build.gradle file. For example:

```
dependencies {
    implementation files('libs/customandroidapi.jar')
}
```

If is impossible to run the app for a "Please select Android SDK" Error, you need to go to build.gradle and click sync now. Then it worked.
To do it:

File -> Sync Project with Gradle Files (Android Studio 3.1.1)
Tools -> Android -> Sync Project with Gradle Files (Android Studio 3.0.1)

Sample Code

Enable Bluetooth Permissions

Modify the "*AndroidManifest.xml*" and add

```xml
<!-- BLUETOOTH PERMISSIONS -->
<uses-permission android:name="android.permission.BLUETOOTH" />
<uses-permission android:name="android.permission.BLUETOOTH_ADMIN" />
```

to enable the permissions to use Bluetooth.

Enable BLE (Bluetooth low energy) Permissions

Modify the "*AndroidManifest.xml*" and add

```xml
<!-- BLE PERMISSIONS -->
<uses-permission android:name="android.permission.BLUETOOTH" />
<uses-permission android:name="android.permission.BLUETOOTH_ADMIN" />
<uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
<uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION" />
<uses-feature android:name="android.hardware.bluetooth_le" android:required="true"/>
```

Add into the application the request of these permissions too :

```
Manifest.permission.ACCESS_FINE_LOCATION
Manifest.permission.ACCESS_COARSE_LOCATION
```

to enable the permissions to use BLE (Bluetooth low energy).

Enable Ethernet/Wifi Permissions

Modify the "*AndroidManifest.xml*" and add

```xml
<!-- WIFI / ETHERNET PERMISSIONS -->
<uses-permission android:name="android.permission.CHANGE_WIFI_MULTICAST_STATE"/>
<uses-permission android:name="android.permission.INTERNET" />
<uses-permission android:name="android.permission.ACCESS_WIFI_STATE" />
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
```

to enable the permissions to use Ethernet / Wifi.

Enable USB Support

Modify the "*AndroidManifest.xml*" and add

```xml
<!-- USB -->
<intent-filter>
  <action android:name="android.hardware.usb.action.USB_DEVICE_ATTACHED" />
```

```
</intent-filter>

<meta-data android:name="android.hardware.usb.action.USB_DEVICE_ATTACHED"
android:resource="@xml/device_filter" />
<!-- USB END -->
```

Create a Folder "*xml*" into "*res*" folder
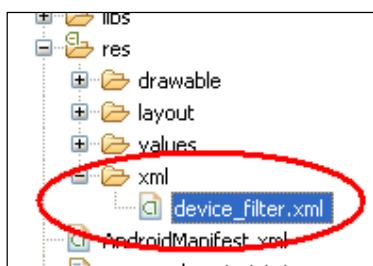
and create a file "*device_filter.xml*" with the code

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <usb-device vendor-id="3540"/>
</resources>
```

to enable the support of CUSTOM printer with USB port. If you don't create this file, the O.S. asks to you the permission to use this device every time.

The first time you plug the USB printer, and the application is already installed, you will see this message:
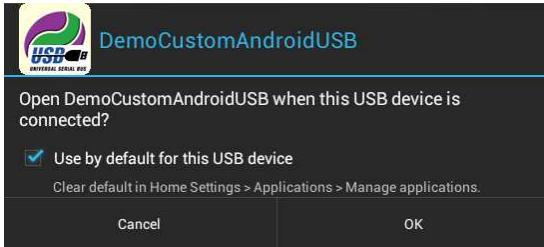
Is important to set the checkbox and press "OK" button to enable the application to use this USB device; Otherwise the USB communication fails.

Connect to Printer (Bluetooth)

//Open it
prnDevice = new CustomAndroidAPI().getPrinterDriverBT(btDevice);

where "*btDevice*" is a BluetoothDevice object.

Connect to Printer (Bluetooth low energy)

```
//Open it
prnDevice = new CustomAndroidAPI().getPrinterDriverBLE(bleDevice, this);
```

where "bleDevice" is a BluetoothDevice object (that supports BLE).

Connect to Printer (Ethernet / Wifi)

```
//Open it
prnDevice = new CustomAndroidAPI().getPrinterDriverETH(strDeviceIP, 9100);
```

where "*strDeviceIP*" is a String with the printer IP (example "192.168.0.1") and "9100" is the default IP port for Raw print.

NOTE: From Android 3.0 Network operations are not allowed on the UI Thread(main thread). You have to do the network communication in a separate thread.

Connect to Printer (USB)

```
//Open it
prnDevice = new CustomAndroidAPI().getPrinterDriverUSB(usbDevice, this);
```

where "*usbDevice*" is a UsbDevice object.

Connect to Printer (COM/RS232)

```
//Open it
prnDevice = new CustomAndroidAPI().getPrinterDriverCOM(strDevPort);
```

where "*strDevPort*" is a string with the hardware device serial port (example "/dev/ttys0").

NOTE(1): the function getPrinterDriverCOM, opens the device port in R/W mode and with the current serial configuration, so must be set these paramethers before to open it.

Example: how to set the device serial port to R/W

```
//Init COM port running linux commands (set the port in R/W mode)
try
{
        Runtime.getRuntime().exec("chmod 666 " + strDevPort);
}
catch (Exception e) {}
```

NOTE(2): if the device to connect is a "CUSTOM Fusion" the port must be "/dev/ttymxc4" and the port must be set with the object "SerialPort".

Example: how to set the serial parameters with "SerialPort"

```
SerialPort sp = null;
//Open and set the configuration object
try
{
        //Open the port "/dev/ttymxc4" with 115200,n,8,1,HW parameters
        sp = new SerialPort(new File("/dev/ttymxc4"), 115200, SerialPort.NON_BLOCKING);
        sp.setParameters(SerialPort.DATABITS_8, SerialPort.PARITY_NONE, SerialPort.STOPBITS_1,
        SerialPort.FLOWCONTROL_RTSCTS);
}
catch (Exception ex )
{
        sp = null;
}
//Close the configuration port object
if (sp != null)
{
        sp.Close();
}
```

Print text

```
PrinterFont font = new PrinterFont();      //create font object and required
font.setEmphasized(true);                  //Set required attributes
font.setJustification(PrinterFont.FONT_JUSTIFICATION_CENTER);
String text = "Hello World !!!";
printer.printText(text, 104,510, font);    //104 is left margin in pixel and 510 is text width in pixel
```

Print image

```
Bitmap image = BitmapFactory.decodeResource(getResources(), R.drawable.sample3);
//Bitmap can be also created from image file as mentioned below
//Bitmap image = BitmapFactory.decodeFile(String pathName);

api.printImage(image, 100, CustomPrinter.IMAGE_SCALE_TO_WIDTH, 376);
//100 is left margin in pixel and 376 is image width in pixel to print
//Above method will scale the image to 376 pixel width and then prints the image with left margin of 100
//pixel
```