



# **BXL SDK for Android\_UPOS Compliant API Reference Guide POS Printer**

---

<b>Rev. 1.11</b>	<b>SPP-R200II</b>
	<b>SPP-R200III</b>
	<b>SPP-R210</b>
	<b>SPP-R220</b>
	<b>SPP-R300</b>
	<b>SPP-R310</b>
	<b>SPP-R400</b>
	<b>SPP-R410</b>
	<b>SRP-350plusIII/352plusIII</b>
	<b>SRP-350III/352III</b>
	<b>SRP-F310II/F312II/F313II</b>
	<b>SRP-380/382</b>
	<b>SRP-330II/332II</b>
	<b>SRP-S300</b>
	<b>SRP-340II/342II</b>
	<b>STP-103III</b>
	<b>SRP-275III</b>
	<b>SRP-Q300/Q302</b>

## ■ Table of Contents

<b>1. About This Manual .....</b>	<b>4</b>
<b>2. Operating Environment .....</b>	<b>5</b>
2-1 Operating System .....	5
2-2 Supported Devices and Interfaces .....	5
<b>3. Development Environment .....</b>	<b>5</b>
3-1 Setting Development Environment .....	5
3-2 Connecting Android Device .....	6
3-2-1 Bluetooth .....	6
3-2-2 Network .....	7
3-2-3 Wi-Fi Direct .....	8
3-2-4 USB .....	9
3-2-5 Setting Android Device Developer Options .....	12
<b>4. Package Contents .....</b>	<b>13</b>
4-1 Manual .....	13
4-2 Library .....	13
4-3 Sample source code .....	13
<b>5. Setting BXL SDK for Android_UPOS compliant.....</b>	<b>14</b>
5-1 BXLConfigLoader .....	14
5-1-1 Procedure to execute the methods .....	14
5-1-2 Constructor .....	14
5-1-3 addEntry .....	15
5-1-4 getAddress .....	16
5-1-5 getDeviceBus .....	16
5-1-6 getDeviceCategory .....	16
5-1-7 getEntries .....	17
5-1-8 getProductName .....	17
5-1-9 modifyEntry .....	17
5-1-10 newFile .....	17
5-1-11 openFile .....	18
5-1-12 removeEntry .....	18
5-1-13 removeAllEntries .....	18
5-1-14 saveFile .....	18
5-2 jpos.xml .....	19
5-2-1 JposEntries .....	19
5-2-2 JposEntry .....	19
5-2-3 creation .....	20
5-2-4 vendor .....	20
5-2-5 jpos .....	20
5-2-6 product .....	20
5-2-7 prop .....	20

<b>6. Sample Program .....</b>	<b>21</b>
6-1 Bitmap Test .....	21
6-2 BXLtest .....	21
6-3 PDF Test .....	21
6-4 PhoneGapSample .....	21
6-4-1 Setting Environment .....	21
6-4-2 Settings to use BXL SDK for Android_UPOS compliant for applications using PhoneGap .....	21
6-5 Text Data Test .....	21
<b>7. User application .....</b>	<b>22</b>
7-1 Library content .....	22
7-2 Using control component on the application .....	22
7-2-1 Constructor .....	22
7-2-2 Connection procedure .....	23
7-2-3 Claim .....	23
7-3 Bitmap printing .....	23
7-4 Printing PDF files .....	24
7-4-1 Inclusion of library .....	24
7-4-2 APIs .....	24
<b>8. Unified POS support list .....</b>	<b>25</b>
8-1 Properties .....	25
8-1-1 POS Printer supported properties .....	25
8-1-2 CashDrawer supported properties .....	26
8-1-3 MSR supported properties .....	27
8-1-4 SmartCardRW supported properties .....	27
8-2 Methods .....	28
8-3 Events .....	28
8-4 Errors .....	28

## **1. About This Manual**

BXL SDK for Android\_UPOS compliant complies with UnifiedPOS.

BXL SDK for Android\_UPOS compliant is for implementing POS Printer service for BIXOLON POS printers and mobile printers. This software provides Java class framework that allows applications' software to access BIXOLON printers.

This manual contains instructions, specifications and limitations of BXL SDK for Android\_UPOS compliant, and it is intended for developers who make application systems using JavaPOS devices.

Before using the BIXOLON printer, the device should be configured using jpos.xml file or BXLConfigLoader class that is included in the BXL SDK for Android\_UPOS Compliant.

[Reference Site]

<http://monroecs.com/unifiedpos.htm>: UnifiedPOS Committee

<http://www.bixolon.com>: SDK Update

<http://www.javapos.com>: JavaPOS Official

<http://developer.android.com/index.html>: Android Developer

BIXOLON is continually improving the functions and quality of products. The specifications of the product and contents of the manual are subject to change without prior notice due to this reason.

## 2. Operating Environment

### 2-1 Operating System

This software supports the following operating systems.

- Android 4.0 (Ice Cream Sandwich) or higher

### 2-2 Supported Devices and Interfaces

Models	Interface
SPP-R200II	Bluetooth / WLAN / USB
SPP-R200III	Bluetooth / WLAN / USB
SPP-R210	Bluetooth / WLAN / USB
SPP-R220	Bluetooth / BLE / WLAN / USB
SPP-R300	Bluetooth / WLAN / USB
SPP-R310	Bluetooth / WLAN / USB
SPP-R400	Bluetooth / WLAN / USB
SPP-R410	Bluetooth / BLE / WLAN / USB
SRP-350plusIII	Bluetooth / WLAN / Ethernet / USB
SRP-352plusIII	Bluetooth / WLAN / Ethernet / USB
SRP-350III	Ethernet / USB
SRP-352III	Ethernet / USB
SRP-F310II	Bluetooth / WLAN / Ethernet / USB
SRP-F312II	Bluetooth / WLAN / Ethernet / USB
SRP-F313II	Bluetooth / WLAN / Ethernet / USB
SRP-380	Bluetooth / WLAN / Ethernet / USB
SRP-382	Bluetooth / WLAN / Ethernet / USB
SRP-330II	Ethernet / USB
SRP-332II	Ethernet / USB
SRP-S300	Bluetooth / WLAN / Ethernet / USB
SRP-340II	Ethernet / USB
SRP-342II	Ethernet / USB
STP-103III	USB
SRP-275III	Ethernet / USB
SRP-Q300	Bluetooth / WLAN / Ethernet / USB
SRP-Q302	Bluetooth / WLAN / Ethernet / USB

※BLE : Bluetooth Low Energy

## 3. Development Environment

### 3-1 Setting Development Environment

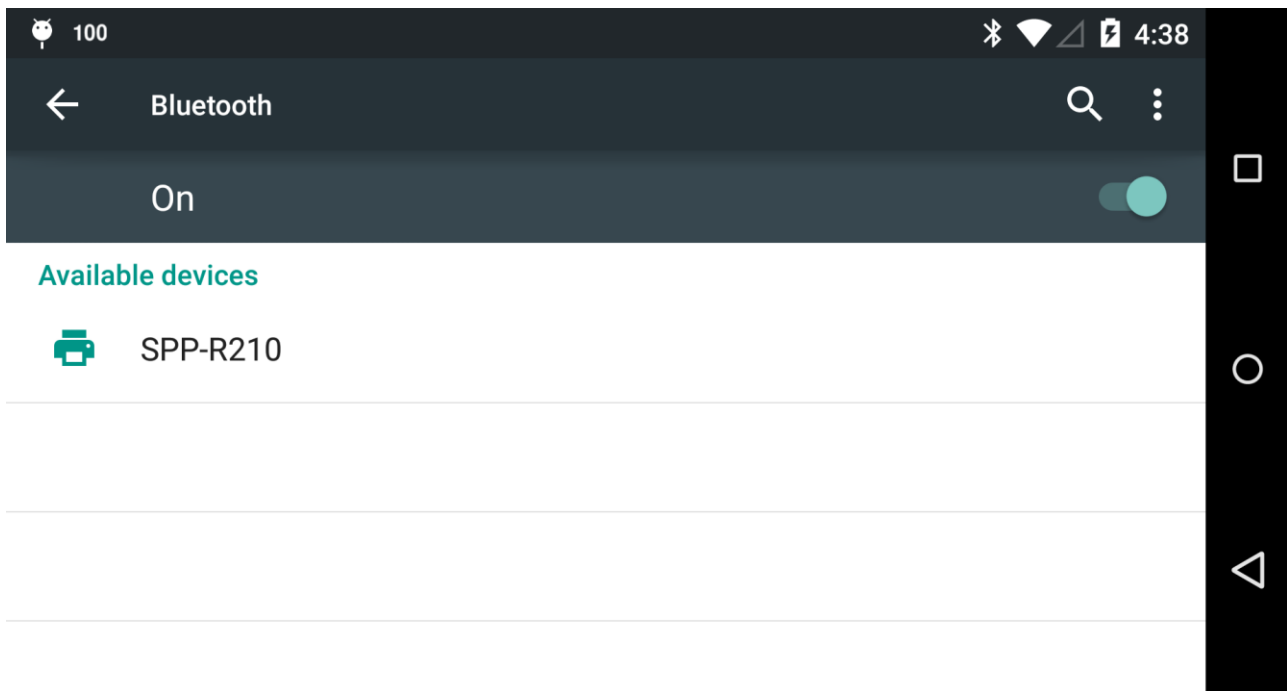
- Java Development Kit (JDK) 7
- Eclipse
- Android SDK Tools
- For Reference: <http://developer.android.com/sdk/index.html>

## 3-2 Connecting Android Device

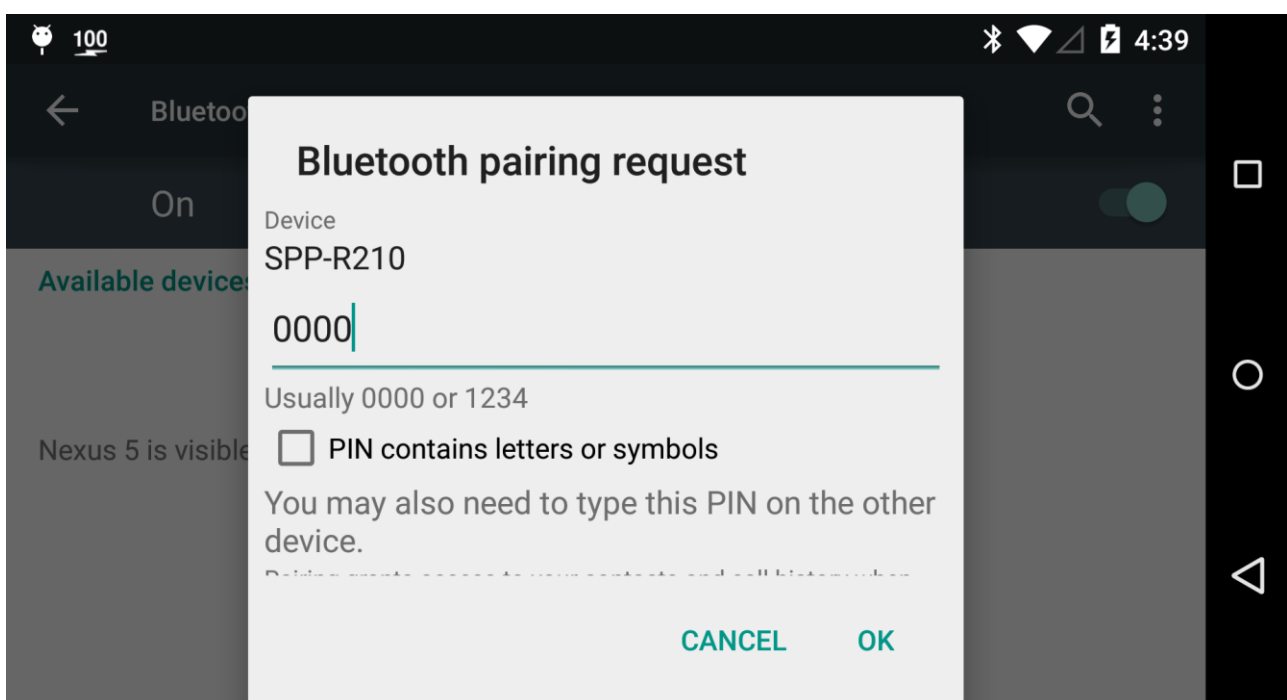
The following screen shot was captured from Nexus 5. Some details and names of specific items could be different depending on the Android version or device.

### 3-2-1 Bluetooth

1. Select [Settings].
2. Bluetooth and printer should be turned on.
3. Select [Bluetooth] for settings.

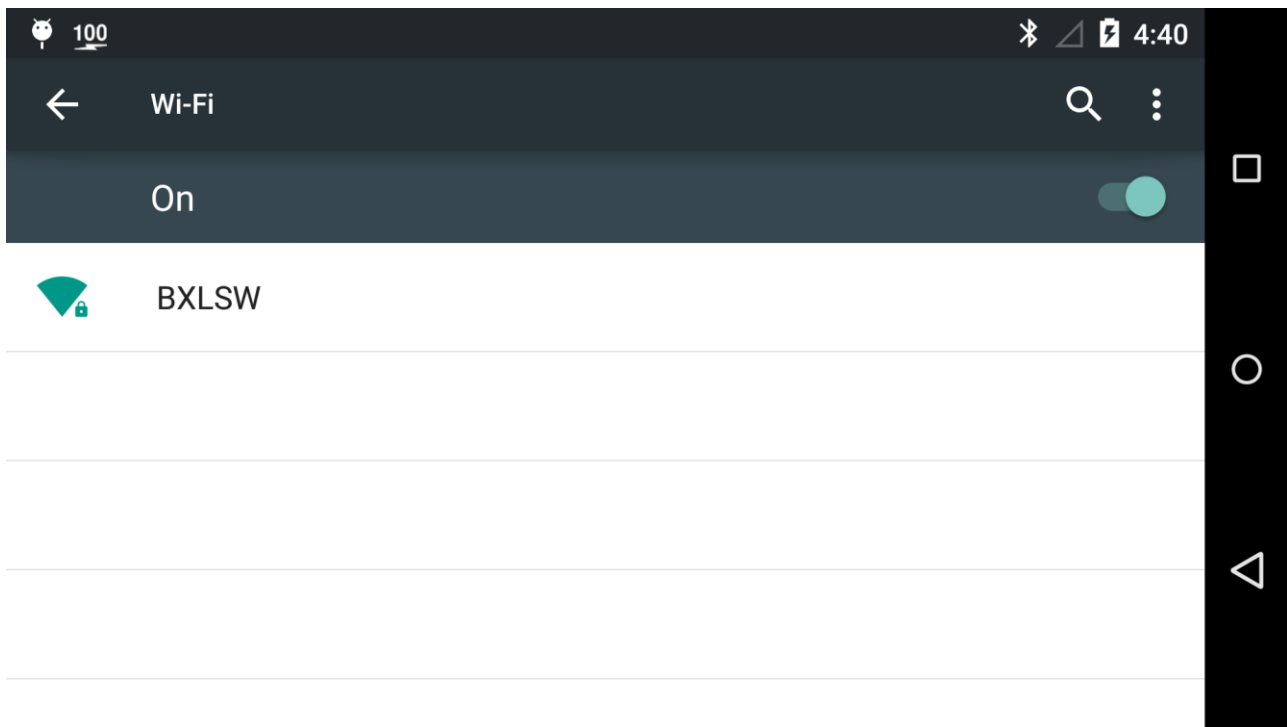


4. Select [Scan] to search the printer to connect and perform pairing.
5. Enter the PIN code. Default PIN code of BIXOLON printers is "0000".



### **3-2-2 Network**

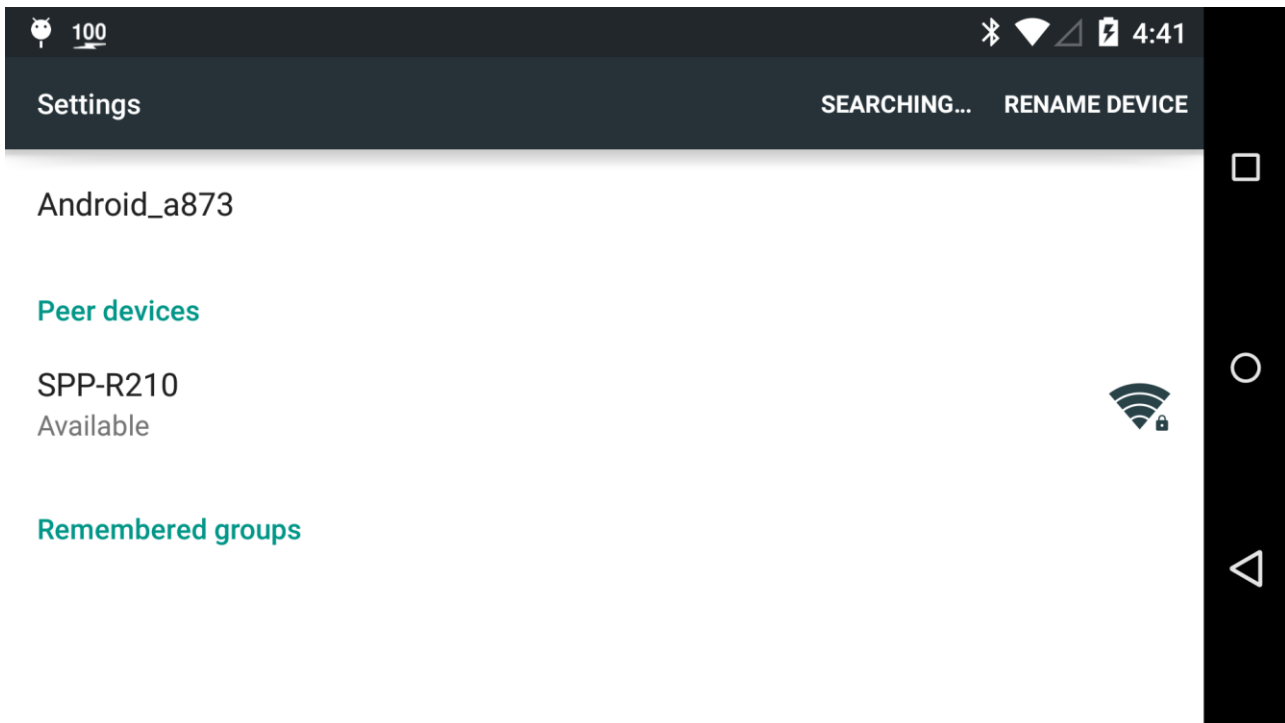
1. Connect the printer to network AP (Access Point) and assign fixed IP address or use DHCP to set the address. Since BIXOLON printers are set to Ad-hoc mode by default, the network configuration should be performed at least once using the Net Configuration Tool that is included in the CD. You can also download the Net Configuration Tool from [BIXOLON Web Site](#).
2. Select [Settings].
3. Wi-Fi should be turned on.
4. Connect the device to the same network that the printer is connected to.



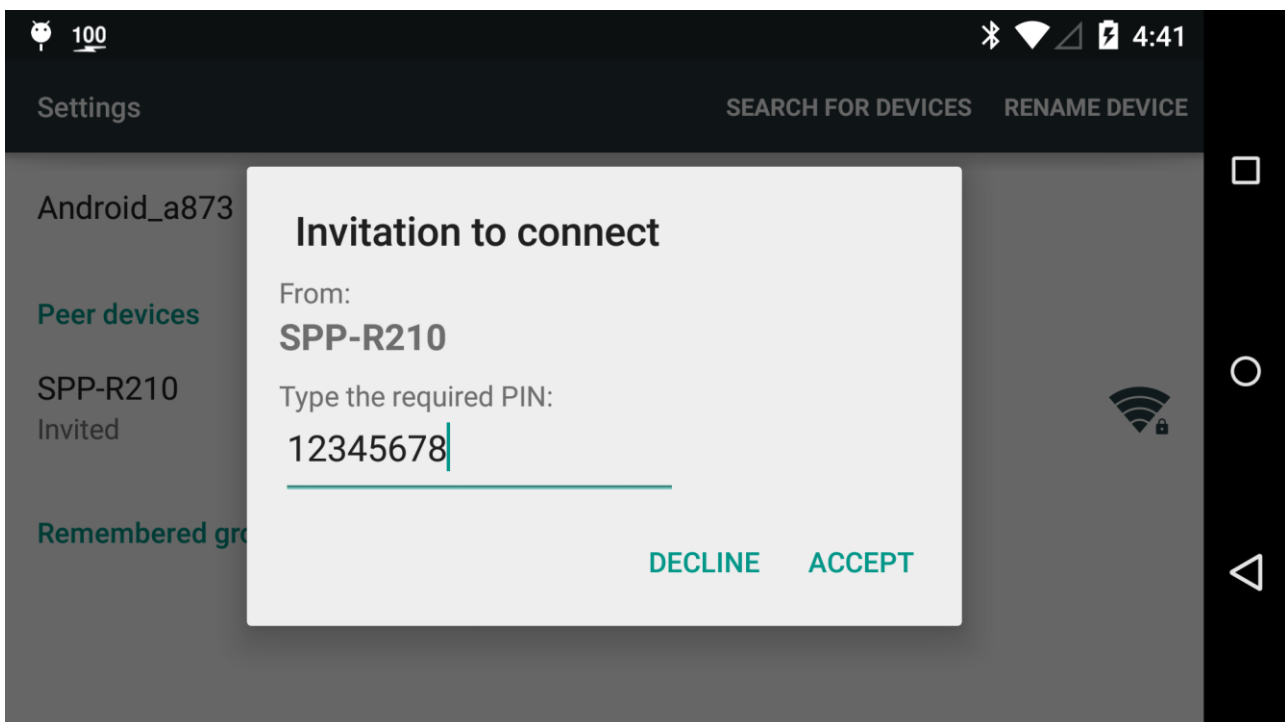
5. No additional setting is required to connect the Android device to TCP/IP port of printer.

### 3-2-3 Wi-Fi Direct

1. Android version should be 4.0 or higher in order to connect with peripheral devices using Wi-Fi Direct.
2. No special driver or printer software is required on Android device.
3. Select [Settings].
4. Wi-Fi should be turned on.
5. Select [Wi-Fi Direct].



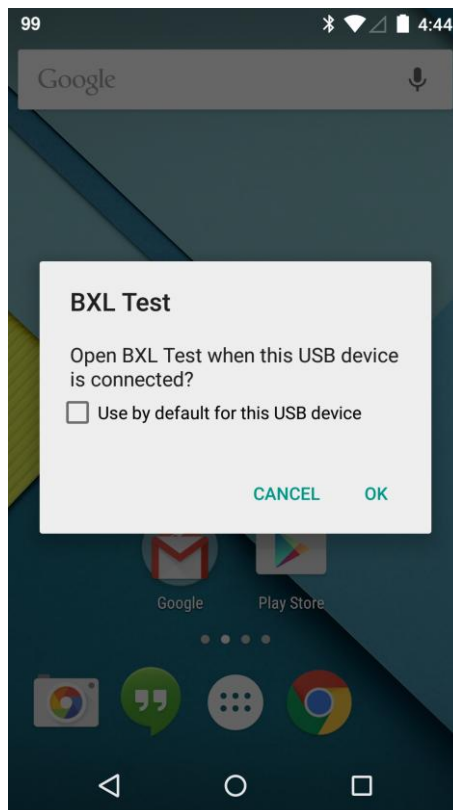
6. Select the printer from search list and connect it.  
Default PIN for Wi-Fi Direct is "12345678".





**3-2-4 USB**

1. Android version should 3.1 or higher to connect USB peripheral devices.
2. No special driver or printer software is required on Android device.
3. Required USB cable depends on specific smartphone or tablet. Most Android devices do not come with A to B USB cable. Mini/micro USB cable or adapter/dock may be required depending on the configuration. Check whether specific type of cable is required for the given Android device.
4. The following message may pop up for some Android devices the first time the printer is connected to the device.



5. Enter the following code in the AndroidManifest.xml and res/xml/device\_filter.xml in order to connect USB peripheral device.

[AndroidManifest.xml]

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

<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" />
```

[device\_filter.xml]

```
<?xml version="1.0" encoding="utf-8"?>
<resources>

    <!-- SPP-R200II -->
    <usb-device product-id="40" vendor-id="5380" />

    <!-- SPP-R200III -->
    <usb-device product-id="91" vendor-id="5380" />

    <!-- SPP-R210 -->
    <usb-device product-id="81" vendor-id="5380" />

    <!-- SPP-R220 -->
    <usb-device product-id="106" vendor-id="5380" />

    <!-- SPP-R300 -->
    <usb-device product-id="33" vendor-id="5380" />

    <!-- SPP-R310 -->
    <usb-device product-id="92" vendor-id="5380" />

    <!-- SPP-R400 -->
    <usb-device product-id="41" vendor-id="5380" />

    <!-- SPP-R410 -->
    <usb-device product-id="75" vendor-id="5380" />

    <!-- SRP-350plusIII -->
    <usb-device product-id="61" vendor-id="5380" />

    <!-- SRP-352plusIII -->
    <usb-device product-id="62" vendor-id="5380" />

    <!-- SRP-350III -->
    <usb-device product-id="43" vendor-id="5380" />

    <!-- SRP-352III -->
    <usb-device product-id="60" vendor-id="5380" />

    <!-- SRP-F310II -->
    <usb-device product-id="87" vendor-id="5380" />

    <!-- SRP-F312II -->
    <usb-device product-id="88" vendor-id="5380" />

    <!-- SRP-F313II -->
    <usb-device product-id="90" vendor-id="5380" />

    <!-- SRP-380 -->
    <usb-device product-id="96" vendor-id="5380" />
```

```
<!-- SRP-382 -->
<usb-device product-id="97" vendor-id="5380" />

<!-- SRP-330II -->
<usb-device product-id="110" vendor-id="5380" />
<!-- SRP-332II-->
<usb-device product-id="111" vendor-id="5380" />

<!-- SRP-S300-->
<usb-device product-id="82" vendor-id="5380" />

<!-- SRP-340II -->
<usb-device product-id="114" vendor-id="5380" />

<!-- SRP-342II -->
  <usb-device product-id="115" vendor-id="5380" />

<!-- STP-103III -->
<usb-device product-id="83" vendor-id="5380" />

<!-- SRP-275III -->
<usb-device product-id="89" vendor-id="5380" />

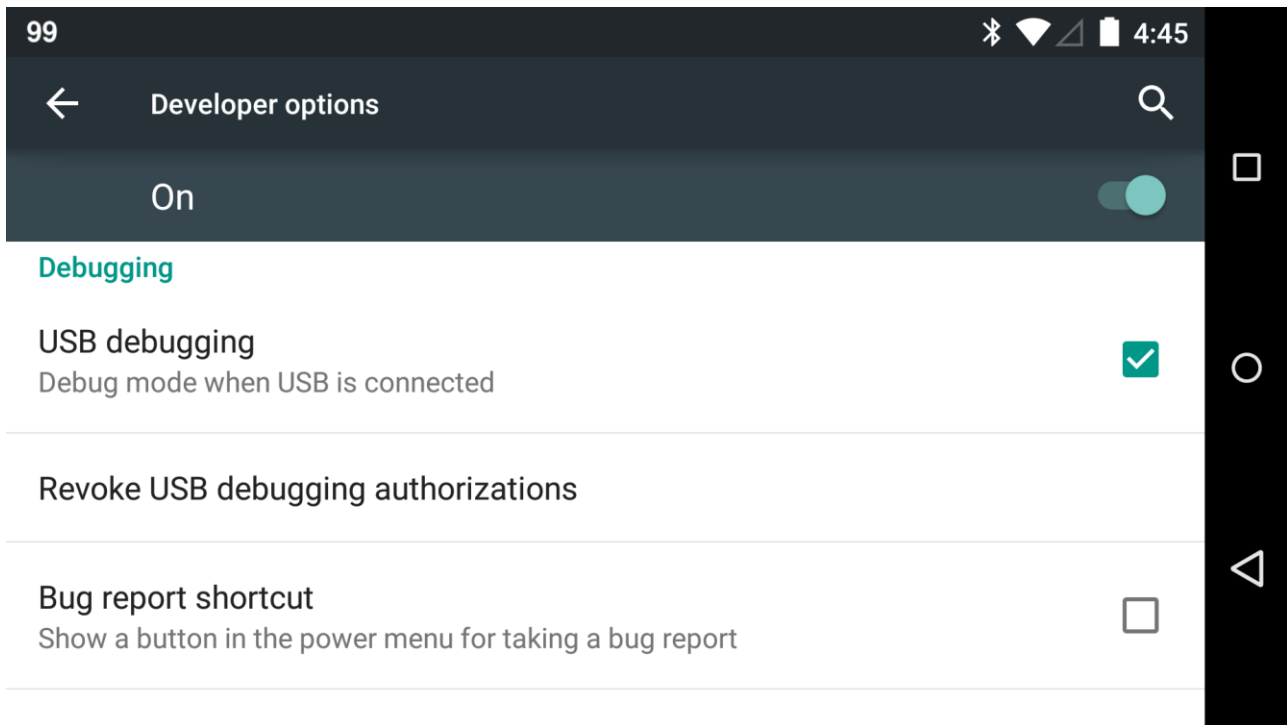
<!-- SRP-Q300 -->
<usb-device product-id="86" vendor-id="5380" />

<!-- SRP-Q302 -->
<usb-device product-id="89" vendor-id="5380" />
```

```
</resources>
```

### **3-2-5 Setting Android Device Developer Options**

1. Select [Settings].
2. Select [Developer options].
3. Enable [USB debugging].



## **4. Package Contents**

### **4-1 Manual**

- docs/Manual\_BXL SDK for Android\_UPOS compliant API Reference Guide\_english\_Rev\_x\_xx: Manual in English
- docs/Manual\_BXL SDK for Android\_UPOS compliant API Reference Guide\_japan\_Rev\_x\_xx: Manual in Japanese
- docs/Manual\_BXL SDK for Android\_UPOS compliant API Reference Guide\_korean\_Rev\_x\_xx: Manual in Korean

### **4-2 Library**

- libs/bixolon\_printer.jar: Implementation of JavaPOS service component layers / printer setting library
- libs/jpos114-controls.jar: Service interface with JavaPOS device control
- libs/xerces.jar: Implementation of apache.org XML service (required for JCL and JavaPOS device control)
- libs/PDF/bixolon\_pdf.jar: Library for printing PDF files
- libs/PDF/armeabi-v7a/libbxmlpdf.so: Native library for printing PDF files

### **4-3 Sample source code**

- samples/android-support-v7-appcompat: v7 Support library project (referred by BXLTest)
- samples/BitmapTest: bitmap printing sample application
- samples/BXLTest: sample applications including printer setting, printer connection, text/barcode/bitmap printing, page mode setting etc.
- samples/PDFTTest: Sample application for printing PDF files
- samples/PhoneGap Plugin: PhoneGap plugin
- samples/PhoneGapSample: Web App sample using PhoneGap
- samples/TextDataTest: text printing option setting sample application

## **5. Setting BXL SDK for Android\_UPOS compliant**

BIXOLON UPOS Printer SDK for Android uses BXLConfigLoader class or jpos.xml to configure the devices.

### **5-1 BXLConfigLoader**

Device settings can be saved by using the library without modifying the jpos.xml file directly and the library manages the jpos.xml internally. Refer to the BXLTest sample source codes for more details.

#### **5-1-1 Procedure to execute the methods**

1. openFile (or newFile if an exception occurs because there is no jpos.xml file to open)
2. addEntry, modifyEntry or removeEntry
3. saveFile

#### **5-1-2 Constructor**

public BXLConfigLoader(Context context)

- Parameters  
*Context:* Context object to pass to POSPrinterService

**5-1-3 addEntry**

public void addEntry (String logicalName, int deviceCategory, String productName, int deviceBus, String address) throws IllegalArgumentException

This method adds a device.

- Parameters

*logicalName*: logical name of the device to register

*deviceCategory*: device category of the device to register

Constant	Value
DEVICE_CATEGORY_CASH_DRAWER	0
DEVICE_CATEGORY_MSR	1
DEVICE_CATEGORY_POS_PRINTER	2
DEVICE_CATEGORY_SMART_CARD_RW	3

*productName*: Product name; it should be set to the same value as the actual name of the model if the deviceCategory is set to DEVICE\_CATEGORY\_POS\_PRINTER.

*deviceBus*: device bus of the device to register

Constant	Value
DEVICE_BUS_BLUETOOTH	0
DEVICE_BUS_ETHERNET	1
DEVICE_BUS_USB	2
DEVICE_BUS_WIFI	3
DEVICE_BUS_WIFI_DIRECT	4

*address*: address of the printer to register

Device bus	Address
DEVICE_BUS_BLUETOOTH	Bluetooth MAC address
DEVICE_BUS_ETHERNET	IP address
DEVICE_BUS_USB	None
DEVICE_BUS_WIFI	IP address
DEVICE_BUS_WIFI_DIRECT	WLAN MAC address

- Throws

*IllegalArgumentException*: This exception occurs if the value of deviceCategory or deviceBus is not the same as one of the defined constants or if productName is null.

**5-1-4 getAddress**

```
public String getProductName(String logicalName)
```

This method searches the logicalName of the registered devices and returns the address of the corresponding device.

- Parameters  
*logicalName:* logical name of the target device to get the address
- Returns  
The address of the corresponding device is returned if a device with logicalName is found, otherwise null is returned.

**5-1-5 getDeviceBus**

```
public int getDeviceBus(String logicalName)
```

This method searches the logicalName of the registered devices and returns the device bus of the corresponding device.

- Parameters  
*logicalName:* logical name of the target device to get the device bus
- Returns  
The device bus of the corresponding device is returned if a device with logicalName is found, otherwise null is returned.

Constant	Value
DEVICE_BUS_BLUETOOTH	0
DEVICE_BUS_ETHERNET	1
DEVICE_BUS_USB	2
DEVICE_BUS_WIFI	3
DEVICE_BUS_WIFI_DIRECT	4

**5-1-6 getDeviceCategory**

```
public int getDeviceCategory(String logicalName)
```

This method searches the logicalName of the registered devices and returns the device category of the corresponding device.

- Parameters  
*logicalName:* logical name of the target device to get the device category
- Returns  
The device category of the corresponding device is returned if a device with logicalName is found, otherwise null is returned

Constant	Value
DEVICE_CATEGORY_CASH_DRAWER	0
DEVICE_CATEGORY_MSR	1
DEVICE_CATEGORY_POS_PRINTER	2
DEVICE_CATEGORY_SMART_CARD_RW	3



**5-1-7 getEntries**

public List<?> getEntries() throws Exception

This method returns the device information saved in the jpos.xml file.

- Returns  
List of registered devices, with JposEntry as the elements of the list

**5-1-8 getProductName**

public String getProductName(String logicalName)

This method searches the logicalName of the registered devices and returns the product name of the corresponding device.

- Parameters  
*logicalName*: logical name of the target device to get the product name
- Returns  
The product name of the corresponding device is returned if a device with logicalName is found, otherwise null is returned.

**5-1-9 modifyEntry**

public boolean modifyEntry(String logicalName, int deviceBus, String address)

This method modifies the device bus or address of the registered device.

- Parameters  
*logicalName*: logical name of the device to modify

*deviceBus*: modified deviceBus

Constant	Value
DEVICE_BUS_BLUETOOTH	0
DEVICE_BUS_ETHERNET	1
DEVICE_BUS_USB	2
DEVICE_BUS_WIFI	3
DEVICE_BUS_WIFI_DIRECT	4

*address*: modified address

- Returns  
The device bus and address of the corresponding device are modified with the given values and true is returned if a device with logicalName is found, otherwise false is returned.

**5-1-10 newFile**

public void newFile()

This method creates a jpos.xml file if it doesn't exist.

### **5-1-11 openFile**

public void openFile() throws Exception

This method opens the jpos.xml file to read or modify the information of the registered device.

- Throws  
*Exception:* if jpos.xml file doesn't exist.

### **5-1-12 removeEntry**

public boolean removeEntry(String logicalName)

This method removes the entry of the registered device.

- Parameters  
*logicalName:* logical name of the device to unregister
- Returns  
The device with the name of logicalName is unregistered and true is returned if a device with logicalName is found, otherwise false is returned.

### **5-1-13 removeAllEntries**

public boolean removeAllEntries()

This method removes all the entries of the registered devices.

### **5-1-14 saveFile**

public void saveFile() throws Exception

This method saves the information of the device to the jpos.xml file.

- Throws  
*Exception:* if there is any error while saving data to the file.

## 5-2 jpos.xml

This method modifies the jpos.xml file directly and saves the printer settings, and the jpos.xml file is saved in a specific format as shown below. Refer to PDFTest or PhoneGapSample sample source codes for more details.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE JposEntries PUBLIC "-//JavaPOS//DTD//EN"
                        "jpos/res/jcl.dtd">
<JposEntries>

  <JposEntry logicalName="SPP-R210" >
    <creation
      factoryClass="com.bxl.loader.ServiceInstanceFactory"
      serviceClass="com.bxl.services.posprinter.POSPrinterService" />

    <vendor
      name="BIXOLON"
      url="http://www.bixolon.com" />

    <jpos
      category="POSPrinter"
      version="1.14" />

    <product
      name="SPP-R210"
      description="Slim yet Rugged 2 inch Thermal Mobile Portable Printer"
      url="http://www.bixolon.com" />

    <prop
      name="deviceBus"
      type="String"
      value="Bluetooth" />
    <prop
      name="address"
      type="String"
      value="74:F0:7D:E2:50:36" />
  </JposEntry>
</JposEntries>
```

### 5-2-1 JposEntries

These are top-level elements.

### 5-2-2 JposEntry

It is an element corresponding to one device. JposEntry element should be added when a new device is added.

**5-2-3 creation**

It is an element to set the service of BIXOLON. The attributes should be fixed as follows.

```
factoryClass="com.bxl.loader.ServiceInstanceFactory"  
serviceClass="com.bxl.services.posprinter.POSPrinterService"
```

**5-2-4 vendor**

It is an element that corresponds to the vendor information of BIXOLON. The attributes should be fixed as follows.

```
name="BIXOLON"  
url="http://www.bixolon.com"
```

**5-2-5 jpos**

It is an element to save the jpos information of device. The category attribute should be set to one of the following values and the version attribute should be fixed as follows.

```
category="CashDrawer" | "MSR" | "POSPrinter" | "SmartCardRW"  
version="1.14"
```

**5-2-6 product**

It is an element to save the detailed information of device. The name attribute should be set to actual name of the model and other attributes can be set to any value.

```
name="SPP-R210"  
description="Slim yet Rugged 2-inch Thermal Mobile Portable Printer"  
url="http://www.bixolon.com" />
```

**5-2-7 prop**

It is an element to save the connectivity information of device. The element with “deviceBus” as name attribute and the element with “address” as name attribute should be configured.

```
name="deviceBus"  
type="String"  
value="Bluetooth" | "Ethernet" | "USB" | "Wi-Fi" | "Wi-Fi Direct"
```

```
name="address"  
type="String"  
value="..."
```

## **6. Sample Program**

### **6-1 Bitmap Test**

This is the reference for developing Android applications using BXL SDK for Android\_UPOS compliant. It can print bitmap files stored in an Android device and those included in the application resource.

### **6-2 BXLtest**

It is an example of an Android application program using BXL SDK for Android\_UPOS compliant. including device connection, text printing, barcode printing, bitmap printing, cash drawer, MSR and SCR functions.

### **6-3 PDF Test**

This is the reference for developing applications that print PDF files using BXL SDK for Android\_UPOS compliant.

### **6-4 PhoneGapSample**

It is an example of an Android web application program using the BXL SDK for Android\_UPOS compliant and it was built with PhoneGap version 3.1.0-0.15.0.

#### **6-4-1 Setting Environment**

- Install Node.js (refer to <http://nodejs.org/>)
- Install PhoneGap (refer to <http://docs.phonegap.com/en/edge/index.html>)
- Add plugins and build the software (refer to [http://docs.phonegap.com/en/edge/guide\\_cli\\_index.md.html#The%20Command-Line%20Interface](http://docs.phonegap.com/en/edge/guide_cli_index.md.html#The%20Command-Line%20Interface))

#### **6-4-2 Settings to use BXL SDK for Android\_UPOS compliant for applications using PhoneGap**

- Add PhoneGap Plugin to the application.
- Add the following code to "res/xml/config.xml" on the PhoneGap application.

```
<feature name="BXLService">
    <param name="android-package"
        value="com.bxl.service.phonegap.BXLService " />
</feature>
```

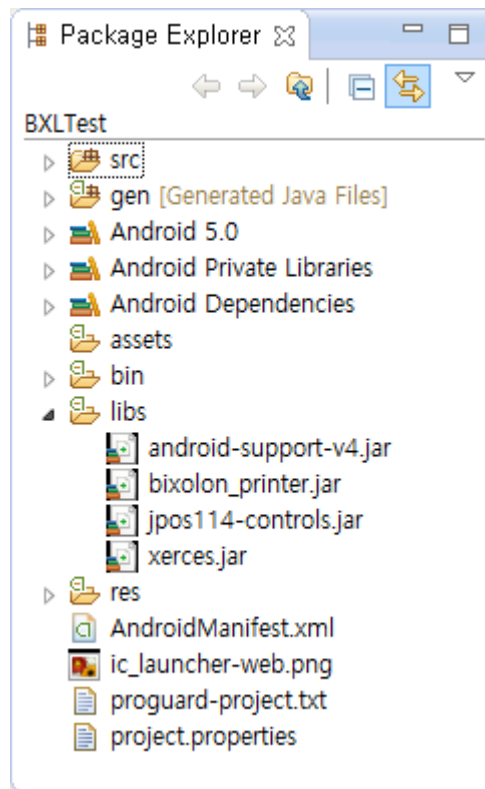
### **6-5 Text Data Test**

This is the reference for developing Android applications using BXL SDK for Android\_UPOS compliant. Text formatting options such as bold and underline can be set.

## 7. User application

### 7-1 Library content

The following files must be present in the application project: bixon\_printer.jar; jpos114-controls.jar; xerces.jar.



### 7-2 Using control component on the application

#### 7-2-1 Constructor

When creating the POSPrinter object, it sends the android.content.Context object to the factor.

CashDrawer, MSR and SmartCardRW use the default constructor.

```
public class MainActivity extends Activity {  
  
    ...  
  
    POSPrinter posPrinter = new POSPrinter(this);  
    CashDrawer cashDrawer = new CashDrawer();  
    MSR msr = new MSR();  
    SmartCardRW smartCardRW = new SmartCardRW();  
    ...  
}
```

### **7-2-2 Connection procedure**

The following methods should be called in the sequence shown below when using the CashDrawer, MSR, POSPrinter and SmartCardRW object to connect to each device.

```
posPrinter.open(logicalName);  
posPrinter.claim(timeout);  
posPrinter.setDeviceEnabled(true);
```

### **7-2-3 Claim**

As CashDrawer, MSR and SmartCardRW are physically built into a printer, they share the connection with POSPrinter. Only if getClaimed() on the POSPrinter is true, claim() on the CashDrawer, MSR and SmartCardRW is allowed.

```
If (posPrinter.getClaimed()) {  
    cashDrawer.claim();  
    msr.claim();  
    smartCardRW.claim();  
}
```

### **7-3 Bitmap printing**

```
void jpos.POSPrinter.printBitmap(int station, String fileName, int width, int alignment)  
    throws JposException
```

Supported bitmap image format: JPEG, GIF, PNG and BMP

Brightness can be set in case of bitmap printing as follows.

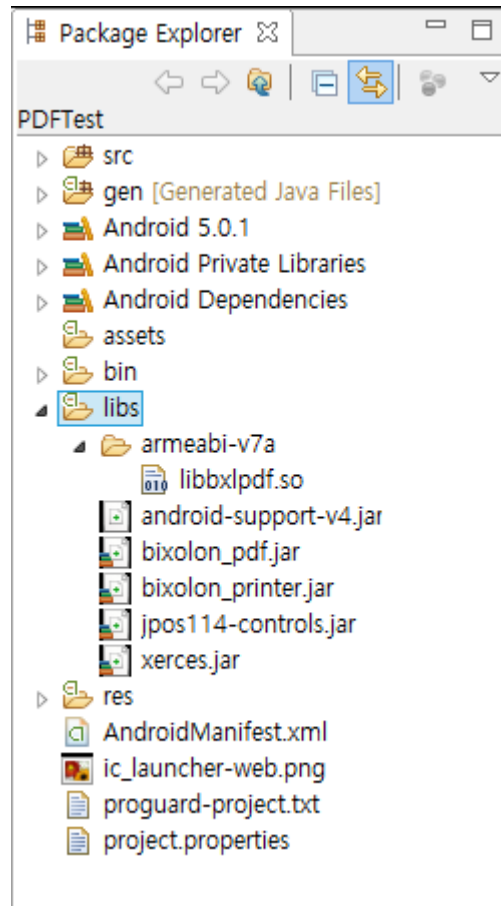
The valid range of the brightness value is from 0 to 100 and the range of effective values is from 13 to 88.

```
ByteBuffer buffer = ByteBuffer.allocate(4);  
buffer.put((byte) POSPrinterConst.PTR_S_RECEIPT);  
buffer.put((byte) brightness); // brightness  
buffer.put((byte) 0x00);        // dummy  
buffer.put((byte) 0x00);        // dummy  
  
posPrinter.printBitmap(buffer.getInt(0), fileName, width, alignment);
```

## 7-4 Printing PDF files

### 7-4-1 Inclusion of library

The library files (armeabi-v7a/libbxmlpdf.so, bixolon\_pdf.jar) located in Software\_BXL SDK for Android\_UPOS compliant\_Vx.x.x/libs/PDF should be included.



### 7-4-2 APIs

Use the following API for printing PDF files:

- Syntax  
void jpos.POSPrinter.printPDF(int station, String fileName, int width, int alignment, int page, int brightness) throws JposException

Parameter	Description
Station	PTR_S_RECEIPT
fileName	Path of the PDF file to be printed
Width	Width of the page to be printed
Alignment	Alignment of the page to be printed
Page	Page to be printed
Brightness	Brightness of the page to be printed (13 - 88)



## 8. Unified POS support list

### 8-1 Properties

#### 8-1-1 POS Printer supported properties

\* List of common property values

Properties	Initial value
CheckHealthText	""
Claimed	False
DeviceEnabled	False
OutputID	0
PowerState	UNKNOWN
State	CLOSED
CapRecNearEndSensor	It depends on the printer model.
CapRecPapercut	It depends on the printer model.
CapRecMarkFeed	It depends on the printer model.
AsyncMode	False
CharacterSet	437
CharacterSetList	"437, 737, 775, 850, 852, 855, 857, 858, 860, 862, 863, 864, 865, 866, 928, 1250, 1251, 1252, 1253, 1254, 1255, 1256, 1257, 1258, 7065(Farsi), 7565(Katakana), 7572(Khmer), 8411(Thai11), 8414(Thai14), 8416(Thai16), 8418(Thai18), 8442(Thai42)"
CoverOpen	False
PageModeArea	It depends on the printer model.
PageModeHorizontalPosition	0
PageModePrintArea	"0,0,0,0"
PageModeVerticalPosition	0
RecLineChars	It depends on the printer model.
RecLineCharsList	It depends on the printer model.
RecLineWidth	It depends on the printer model.
RecEmpty	False

\* List of property values by printer model

Model	PageModeArea	RecLine Chars	RecLine CharsList	RecLine Width
SPP-R200II	"384,2400"	32	"32,42"	384
SPP-R200III	"384,2400"	32	"32,42"	384
SPP-R210	"384,2400"	32	"32,42"	384
SPP-R220	"384,2400"	32	"32,42"	384
SPP-R300	"576,2400"	48	"48,64"	576
SPP-R310	"576,2400"	48	"48,64"	576
SPP-R400	"832,2400"	69	"69,92"	832
SPP-R410	"832,2400"	69	"69,92"	832
SRP-350plusIII	"512,1662"	42	"42,56"	512
SRP-352plusIII	"576,1662"	48	"48,64"	576
SRP-350III	"512,1662"	42	"42,56"	512
SRP-352III	"576,1662"	48	"48,64"	576
SRP-F310II	"512,1662"	42	"42,56"	512
SRP-F312II	"576,1662"	48	"48,64"	576
SRP-F313II	"640,1662"	50	"50,67"	640
SRP-380	"512,1662"	42	"42,56"	512
SRP-382	"576,1662"	48	"48,64"	576
SRP-330II	"512,1662"	42	"42,56"	512
SRP-332II	"576,1662"	48	"48,64"	576
SRP-S300	"576,1662"	48	"48,64"	576
SRP-340II	"512,1662"	42	"42,56"	512
SRP-342II	"576,1662"	48	"48,64"	576
STP-103III	"384,1662"	32	"32,42"	384
SRP-275III	-	33	"33,44"	400
SRP-Q300	"512,1662"	42	"42,56"	512
SRP-Q302	"576,1662"	48	"48,64"	576

## 8-1-2 CashDrawer supported properties

Properties	Initial value
CheckHealthText	" "
Claimed	False
DeviceEnabled	False
PowerState	UNKNOWN
State	CLOSED
DrawerOpened	False

**8-1-3 MSR supported properties**

Properties	Initial value
AutoDisable	False
CheckHealthText	""
Claimed	False
DeviceEnabled	False
PowerState	UNKNOWN
State	CLOSED
CapDataEncryption	False
DataEncryptionAlgorithm	NONE
Track1Data	Byte array with length 0
Track1EncryptedData	Byte array with length 0
Track1EncryptedDataLength	0
Track2Data	Byte array with length 0
Track2EncryptedData	Byte array with length 0
Track2EncryptedDataLength	0
Track3Data	Byte array with length 0
Track3EncryptedData	Byte array with length 0
Track3EncryptedDataLength	0
TracksToRead	NONE

**8-1-4 SmartCardRW supported properties**

Properties	Initial value
CheckHealthText	""
Claimed	False
DeviceEnabled	False
PowerState	UNKNOWN
State	CLOSED
CapInterfaceMode	APDU
CapIsoEmvMode	EMV
CapSCPresentSensor	SLOT1   SLOT2   SLOT3
CapSCSlots	SLOT1   SLOT2   SLOT3
CapTransmissionProtocol	PROTOCOL_T0   PROTOCOL_T1
InterfaceMode	APDU
IsoEmvMode	EMV
SCPresentSensor	0
SCSlot	0

## 8-2 Methods

POSPrinter	CashDrawer	MSR	SmartCardRW
open	open	open	open
close	close	close	close
claim	claim	claim	claim
release	release	release	release
checkHealth	checkHealth	checkHealth	checkHealth
directIO	openDrawer	clearInput clearInputProperties	clearInput clearInputProperties readData
clearOutput			
clearPrintArea			
cutPaper			
markFeed			
pageModePrint			
printBarCode			
printBitmap			
printNormal			
transactionPrint			
updateFirmware			

## 8-3 Events

Events	POSPrinter	CashDrawer	MSR	SmartCardRW
DataEvent	X	X	O	X
OutputCompleteEvent	O	X	X	X
StatusUpdateEvent	O	O	O	O

## 8-4 Errors

Method	Errors
open	NOSERVICE
claim	ILLEGAL TIMEOUT
release checkHealth clearOutput clearPrintArea cutPaper markFeed pageModePrint printBarCode printBitmap printNormal transactionPrint	ILLEGAL