

# SmartPos SDK User Guide

Version 2.6

2024-08-05

Version	Date	Description
1.0	2015-06-28	First release
2.0	2015-10-17	Added magnetic card, smartcard
2.1	2019-02-16	Printer support new feature
2.2	2019-04-10	Added SAM card
2.3	2020-03-21	Revision
2.4	2020-08-17	SmartCard update API
2.5	2024-02-02	Added LED API
2.6	2024-08-05	Update API for SAM card, not support old module

No part of this manual may be reproduced or transmitted in any form or by any means without prior written consent of ShenZhen Pay Device Technology Co., Ltd.

## Contents

1.Overview.....	4
2.Deliverables.....	4
3.Integration.....	5
4.Thermal printing.....	6
4.1 Overview.....	6
4.2 Features.....	6
4.3 Workflow.....	7
4.4 API.....	8
4.5 FAQ.....	23
5. Magnetic Card Reader.....	25
5.1 Overview.....	25
5.2 Workflow.....	25
5.3 API.....	26
5.4 FAQ.....	28
6.Smart Card Reader.....	29
6.1 Overview.....	29
6.2 Workflow.....	29
6.3 API.....	30
6.4 FAQ.....	33
7.NFC Reader.....	34
7.1 Overview.....	34
7.2 FAQ.....	34
8.Camera.....	34
8.1 Overview.....	34
8.2 FAQ.....	34
9.FingerPrint.....	35
9.1 Overview.....	35
9.2 Workflow.....	35
9.3 API.....	36
9.4 FAQ.....	37
10.Secondary Display.....	38
10.1 Overview.....	38
10.2 FAQ.....	38

11.SerialPort.....	39
11.1 Overview.....	39
11.2 Workflow.....	39
11.3 API.....	40
12.Secure Access Module(SAM/PSAM).....	41
12.1 Overview.....	41
12.2 Workflow.....	41
12.3 API.....	42
12.4 FAQ.....	45
13.Cash Drawer.....	46
13.1 Overview.....	46
13.2 API.....	46
13.3 FAQ.....	46
14.Scanner.....	47
14.1 Overview.....	47
14.2 FAQ.....	47
15.LED.....	48
15.1 Overview.....	48
15.2 API.....	48
16.Appendix.....	49
16.1 Restore the printer firmware.....	49
16.2 Full screen mode(Kiosk mode).....	49

# 1.Overview

SmartPos SDK provides java API to control modules of pos terminal, such as Thermal printing, Magnetic Card Reader, Smart Card Reader, Camera etc. User can develop application quickly based on the SDK.

## 2.Deliverables

SDK includes 2 parts, you just need import these files to your project:

### 1. [paydevice-smartpos-sdk.jar](#)

a java archive that includes all core api for all modules.

### 2. [libpaydevice-smartpos.so](#) [libAIUSB.so](#)

dynamic-link libraries that provide low-level interface of java core api.

### Demo

this demo shows how to use the API to control modules.

### [zbar.jar](#)

### [libiconv.so](#)

### [libzbarjni.so](#)

**Note:** Demo based on ZBar to implement QR code and barcode scanning, if you need not scanning just ignore.

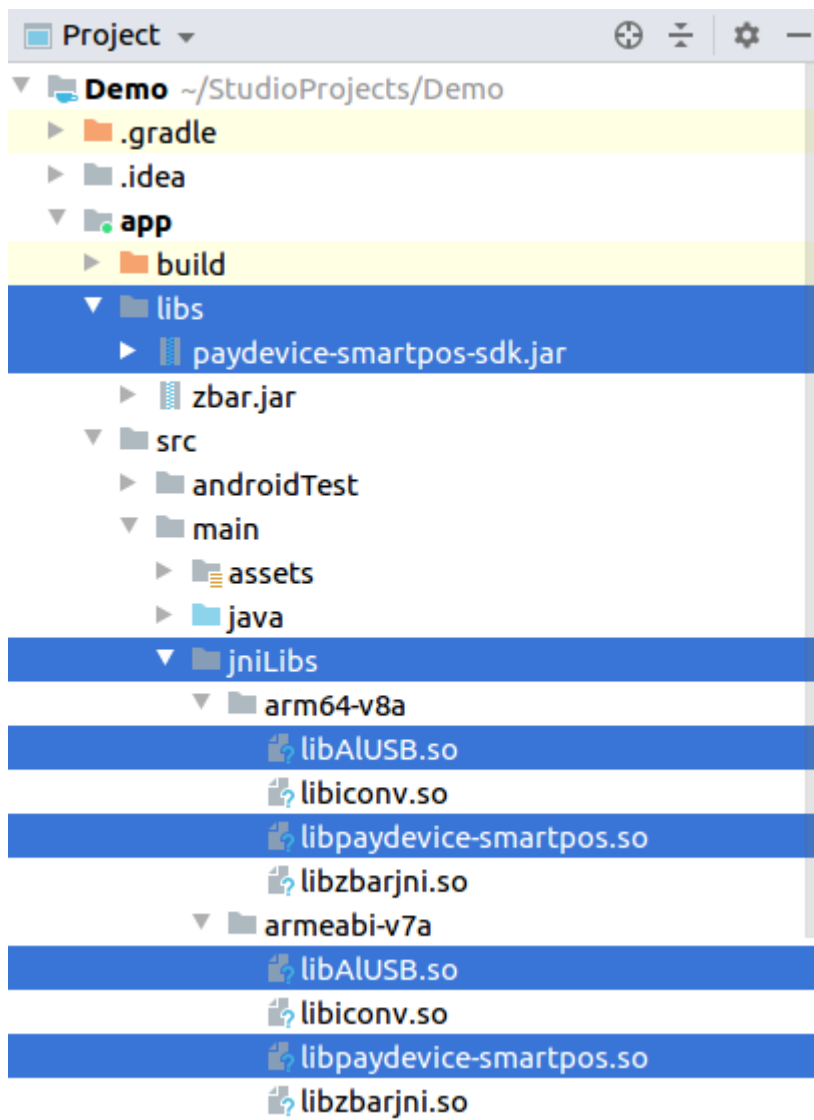
<https://github.com/ZBar/ZBar>

## 3.Integration

This chapter describes the integration of SmartPos SDK in Android Studio development tool. You just need import 2 parts:

1. Copy the [paydevice-smartpos-sdk.jar](#) to the [libs](#) folder in your project
2. Copy the [libpaydevice-smartpos.so](#) [libAlUSB.so](#) to the folder [armeabi-v7a](#) and [arm64-v8a](#) in your project

Folder structure like as follow:



## 4. Thermal printing

### 4.1 Overview

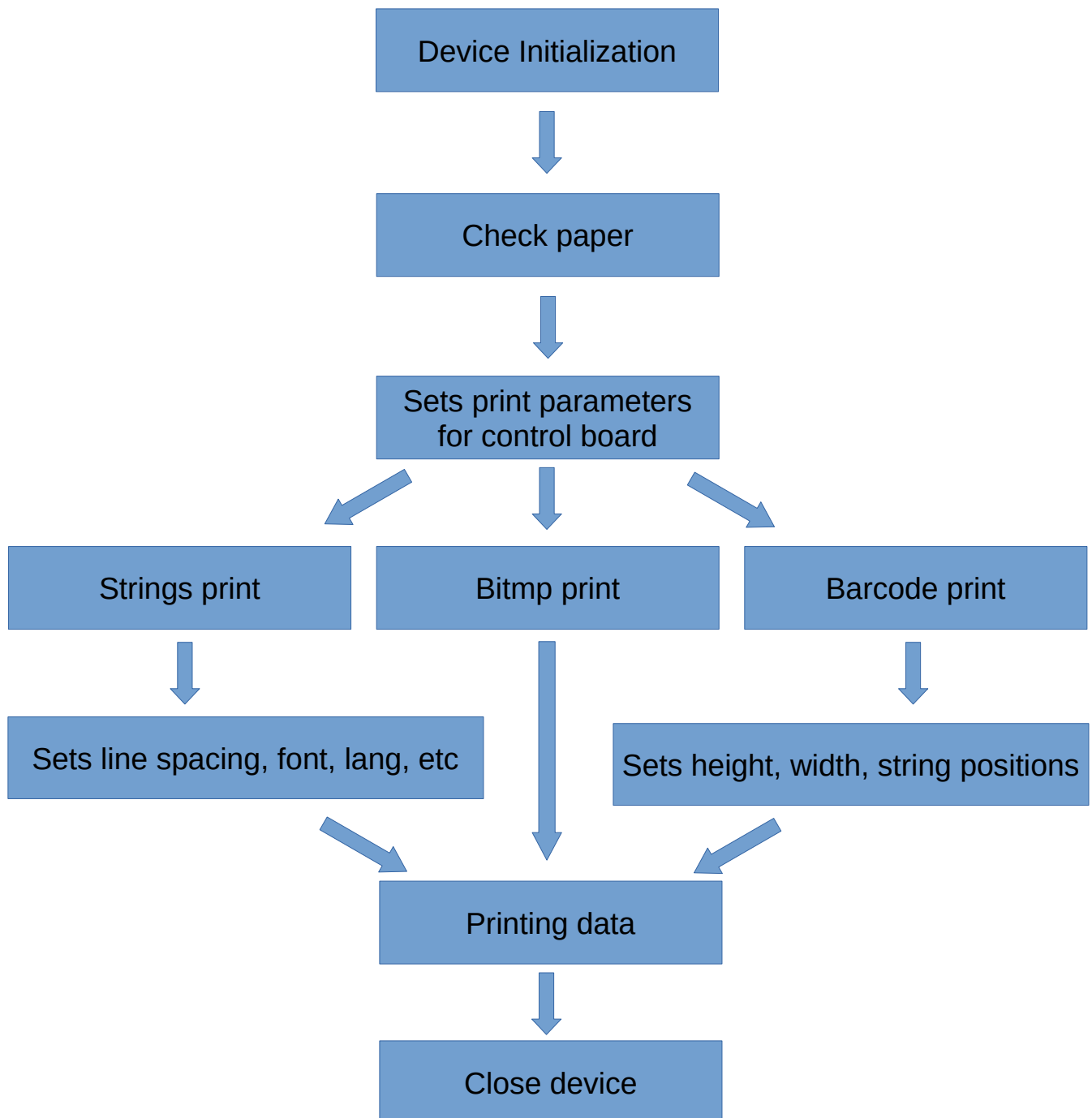
Thermal printing module provides a rich API to implement printing features

### 4.2 Features

Thermal printing module supports features:

- \* Allows multi-language, print parameters, character format
- \* Support strings, QRcode, barcodes, bitmap printing

## 4.3 Workflow



## 4.4 API

### Glossary

**dots:** The number of print dots that is the smallest unit of print. 1 dots = 0.125 mm

**max printing width:**

58mm printer: 48mm(384dots \* 0.125mm), maximum dots is 384

80mm printer: 72mm(576dots \* 0.125mm), maximum dots is 576

**HRI:** Barcode human readable interpretation

Throws error code:

public static final int PRINTER_ERR_OPEN	//open error
public static final int PRINTER_ERR_CLOSE	//close error
public static final int PRINTER_ERR_READ	//read error
public static final int PRINTER_ERR_WRITE	//write error
public static final int PRINTER_ERR_PARAM	//parameter error
public static final int PRINTER_ERR_NO_PAPER	//no paper
public static final int PRINTER_ERR_UPDATE	//fw update error

Printer model:

public static final int PRINTER_MODEL_UNKNOWN	//unknown model
public static final int PRINTER_MODEL_PRN2103	//model PRN2103



Interface type:

```
public static final int PRINTER_TYPE_USB    //USB printer
public static final int PRINTER_TYPE_SERIAL //Serial port printer
```

Paper type:

```
public static final int TYPE_PAPER_WIDTH_58MM //58mm
public static final int TYPE_PAPER_WIDTH_80MM //80mm
```

Cut mode:

```
public static final int FULL_CUT    //full cut
public static final int HALF_CUT    //half cut
```

Align mode:

```
public static final int ALIGN_LEFT    //align left
public static final int ALIGN_MIDDLE  //align middle
public static final int ALIGN_RIGHT   //align right
```

Print mode:

```
public static final int FONT_DEFAULT    =0           //default
public static final int FONT_SMALL      =1<<0       //9x17 font
public static final int FONT_INVERSE    =1<<1        //white/black reverse
public static final int FONT_UPSIDE_DOWN =1<<2       //upside down
public static final int FONT_EMPHASIZED  =1<<3       //emphasized
public static final int FONT_DOUBLE_HEIGHT =1<<4     //double height
```

```
public static final int FONT_DOUBLE_WIDTH  =1<<5 //double width
public static final int FONT_ROTATE        =1<<6      //rotate 90° clockwise
public static final int FONT_UNDERLINE     =1<<7      //underline
```

Underline height:

```
public static final int UNDERLINE_ZERO      //no underline
public static final int UNDERLINE_HIGH_1    //hight 1 dots
public static final int UNDERLINE_HIGH_2    //hight 2 dots
```

Barcode HRI positions:

```
public static final int CODEBAR_STRING_MODE_NONE      //no string
public static final int CODEBAR_STRING_MODE_ABOVE     //above
public static final int CODEBAR_STRING_MODE_BELOW     //below
public static final int CODEBAR_STRING_MODE_BOTH      //both
```

Encoding format:

```
public static final int UPC_A
public static final int UPC_E
public static final int EAN13
public static final int EAN8
public static final int CODE39
public static final int I25
public static final int CODEBAR
public static final int CODE93
```

public static final int CODE128

public static final int CODE11

public static final int MSI

QR correction level

public static final int QR\_ECC\_LEVEL\_L = 1 //Low 7%

public static final int QR\_ECC\_LEVEL\_M = 2 //Medium 15%

public static final int QR\_ECC\_LEVEL\_Q = 3 //Quartile 25%

public static final int QR\_ECC\_LEVEL\_H = 4 //High 30%

Code page(totals 45):

Details refer: [https://en.wikipedia.org/wiki/Code\\_page](https://en.wikipedia.org/wiki/Code_page)

public static final int CODE\_PAGE\_CP437 //English

public static final int CODE\_PAGE\_CP720 //Arabic

public static final int CODE\_PAGE\_CP737 //Greek

public static final int CODE\_PAGE\_CP755 //East Europe, Latvian2

public static final int CODE\_PAGE\_CP775 //Estonian, Lithuanian, Latvian

public static final int CODE\_PAGE\_CP850 //Western Europe

public static final int CODE\_PAGE\_CP852 //Latin2

public static final int CODE\_PAGE\_CP855 //Cyrillic script

public static final int CODE\_PAGE\_CP856 //Hebrew

public static final int CODE\_PAGE\_CP857 //Turkish

public static final int CODE\_PAGE\_CP858 //West Europe

public static final int CODE\_PAGE\_CP860 //Portuguese

```
public static final int CODE_PAGE_CP862      //Hebrew
public static final int CODE_PAGE_CP863      //Canadian-French
public static final int CODE_PAGE_CP864      //Arabic
public static final int CODE_PAGE_CP865      //Nordic
public static final int CODE_PAGE_CP866      //Cyrillic2
public static final int CODE_PAGE_CP874      //Thai
public static final int CODE_PAGE_CP1250     //Central Europe
public static final int CODE_PAGE_CP1251     //Cyrillic
public static final int CODE_PAGE_CP1252     //Latin1
public static final int CODE_PAGE_CP1253     //Greek

public static final int CODE_PAGE_CP1254     //Turkish
public static final int CODE_PAGE_CP1255     //Hebrew
public static final int CODE_PAGE_CP1256     //Arabic
public static final int CODE_PAGE_CP1257     //Baltic
public static final int CODE_PAGE_CP1258     //Vietnamese
public static final int CODE_PAGE_ISO_8859_1 //West Europe
public static final int CODE_PAGE_ISO_8859_2 //Latin2
public static final int CODE_PAGE_ISO_8859_3 //Latin3
public static final int CODE_PAGE_ISO_8859_4 //Baltic
public static final int CODE_PAGE_ISO_8859_5 //Cyrillic
public static final int CODE_PAGE_ISO_8859_6 //Arabic
public static final int CODE_PAGE_ISO_8859_7 //Greek
public static final int CODE_PAGE_ISO_8859_8 //Hebrew
```

```
public static final int CODE_PAGE_ISO_8859_9 //Turkish
public static final int CODE_PAGE_ISO_8859_15 //Latin9
public static final int CODE_PAGE_BIG5 //Traditional Chinese
public static final int CODE_PAGE_GB18030 //Simplified Chinese
public static final int CODE_PAGE_LATVIAN //Latvian
public static final int CODE_PAGE_IRAN //Iran System encoding
standard
public static final int CODE_PAGE_IRAN2 //Iran2
public static final int CODE_PAGE_KATAKANA //Japanese
public static final int CODE_PAGE_MIK //Cyrillic,Bulgarian
public static final int CODE_PAGE_THAI //Thai1
public static final int CODE_PAGE_THAI2 //Thai2
```

API list:

\* Initialize the printer

`void connect()`

**Note:** Serialport printer takes about 100ms during power on

\* Turn off the printer

`void disconnect()`

**Note:** Serialport printer need wait print finish before power down

\* Check the paper

`boolean checkPaper()`

throw `PRINTER_ERR_NO_PAPER` if no paper

\* Get printer type

`int getPrinterType()`

return: PRINTER\_TYPE\_USB,PRINTER\_TYPE\_SERIAL

\* Send string

`void sendData(String data)`

parameters: `data` The characters to be printed

`void sendData(String data, String encoding)`

parameters: `data` The characters to be printed

`encoding` character encoding(such as CP437,ISO-8859-1,etc.)

**Note:** As default, Java used Unicode/UCS-2 encoding for type String. All string need to be converted into the corresponding character encoding then the printer could print correctly.

Typically we only need print in one language, you just need set the global string encoding via function `void setStringEncoding(String encoding)` and then invoke function `void sendData(String data)` to do print.

Also If you want to simultaneously print in multiple languages, please use the function `void sendData(String data, String encoding)`

\* Set global character encoding

`void setStringEncoding(String encoding)`

parameters: `encoding` character encoding

\* To feed one line

`void cmdLineFeed()`

\* To feed n line

`void cmdLineFeed(int n)`

parameters: `n` the number of the lines

**Note:** The line spacing determined by `void cmdSetDefaultLineSpacing()` and `void cmdSetLineSpacing(int dots)`

\* Jump next tab position

`public void cmdJumpTab()`

**Note:** tab position decide by `void cmdSetTable(byte[] offset)`

\*Sets horizontal table position

`void cmdSetTable(byte[] offset)`

parameters: `offset` position array,array length 1-16. unit is 9dots for small font or 12dots for normal font

**Note:** refer Demo

\*Unset horizontal table position

`void cmdUnSetTable()`

\* Set the line spacing to default(32 dots)

`void cmdSetDefaultLineSpacing()`

\* Set the line spacing to n dots

`void cmdSetLineSpacing(int dots)`

parameters: `dots` number of the dots

\* Set the align mode(default is align left)

`void cmdSetAlignMode(int mode)`

parameters: `mode` align mode

`mode=ALIGN_LEFT` align left

`mode=ALIGN_MIDDLE` align middle

`mode=ALIGN_RIGHT` align right

\* Set print offset(Only valid in current line)

`void cmdSetPrintOffset(int offset)`

parameters: `offset` x offset

\* Set print mode

`void cmdSetPrintMode(int mode)`

parameters: `mode`

bit 0: default(set 1 used small font)

bit 1: character inverse

bit 2: character upside down

bit 3: character emphasized

bit 4: double height

bit 5: double width



bit 6: rotate

bit 7: underline

**Note:** Set the corresponding bit to 1 takes effect. The underline no effect if set rotated 90° clockwise or inverse

\* Set the height of underline

`void cmdSetUnderlineHeight(int n)`

parameters: `n` dots

`n=UNDERLINE_ZERO`      no underline

`n=UNDERLINE_HIGH_1`    height is 1 dots

`n=UNDERLINE_HIGH_2`    height is 2 dots

\* Set the position of the barcode string

`void cmdSetBarCodeStringPosition(int mode)`

parameters: `mode` position of the barcode string

`mode=CODEBAR_STRING_MODE_NONE`    no string

`mode=CODEBAR_STRING_MODE_ABOVE`   above the barcode

`mode=CODEBAR_STRING_MODE_BELOW`   below the barcode

`mode=CODEBAR_STRING_MODE_BOTH`    both above and below  
the barcode

\* Set the n multiple scale for font's width and height

`void cmdSetFontScaleSize(int scaleWidth, int scaleHeight)`

parameters: `scaleWidth`      width multiple 0~7

`scaleHeight`      height multiple 0~7

\* Set the height of the barcode

`void cmdSetBarCodeHeight(int n)`

parameters: `n` height(unit: dots)

$1 \leq n \leq 255$  (default is: 50 dots)

\* Set the width of the basic line

`void cmdSetBarCodeWidth(int n)`

parameters: `n` width(unit: dots)

$n = 2$  or  $n = 3$  (default is: 3 dots)

\* Set the left spacing of the barcode

`void cmdSetBarCodeLeftSpacing(int n)`

parameters: `n` left spacing(unit: dots)

\* Printing barcode

`void cmdBarCodePrint(int type, String string)`

parameters: `type` the encoding of the barcode

`string` corresponding string of the barcode

`void cmdBarCodePrint(int type, byte[] stringBytes)`

parameters: `type` the encoding of the barcode

`stringBytes` corresponding string of the barcode

\* Print bitmap

`void cmdBitmapPrint(Bitmap bitmap, int left, int top)`

parameters: `bitmap` android bitmap

`left` horizontal offset dots

`top` vertical offset dots

**Note:** required:  $\text{left} + w < 384(576 \text{ for } 80\text{mm printer})$

\* Print bitmap(Split the bitmap into multiple parts with height 24 dots and print them one by one)

`void cmdBitmapPrintEx(Bitmap bitmap, int left, int top)`

parameters: `bitmap` android bitmap

`left` horizontal offset dots

`top` vertical offset dots

**Note:** bitmap height must be an integer multiple of 8

\* Set the parameter of the control board(some control board do not supported)

`void cmdSetHeatingParam(int dots, int time, int interval)`

parameters: `dots` maximum heating dots, range 0-255, unit 8dots. default is 7

`time` heating time, range 0-255, unit 10us. default is 80

`interval` heating time interval, range 0-255, unit 10us. default is 2

**Note:** This function is normally used to adjust the print sharpness. The more max heating dots, the more peak current will cost when printing, the faster printing speed. The max heating dots is  $8 \times (n1 + 1)$

The more heating time, the more density , but the slower printing speed. If heating time is too short, blank page may occur. The more heating interval, the more clear, but the slower printing speed.

\* Set the print density

`void cmdSetPrintDensity(int density, int delay)`

parameters: `density` print density, range 0-31(real density: 50%+5%\*density)

`delay` delay time, range 0-7 (delay time: delay\*250us)

**Note:** This function is used to set the print density. maybe different brands of paper needs different value.

\* Set the code page

`void cmdSetPrinterLanguage(int code)`

parameters: `code` code page(such as CODE\_PAGE\_CP437,etc)

\* Print QR code(after printer control board firmware v1.04)

`void cmdQrCodePrint(int version, int ecc, String data)`

parameters: `version` QR version 0~16(n dots x n dots square)

`ecc` QR correction level 1~4

`data` QR string data

\* Print line segments(after printer control board firmware v1.04)

`void cmdPrintMultipleLines(int lineCount, int[] lineStartPos, int[] lineEndPos)`

parameters: `lineCount` count of the line segments

`lineStartPos` start position of the line segments array

`lineEndPos` end position of the line segments array

Note: line segments can make lines or curves.you can used this function to print horizontal line or vertical line and even curve

\* Save bitmaps into NVRAM(control board's NVRAM, bitmaps always hold even after power off)

`void cmdSaveBitmapToNVRAM(Bitmap[] bmpArray)`

parameters: `bmpArray` bitmaps array

**Note:** Total size of NVRAM is 64KB.Bitmap width and height must be an integer multiple of 8.

**Warning:** You should not write NVRAM often, it maybe cause NVRAM broken.

Recommended less than 10 times a day.You should save bitmaps only once and then just print it every times.

Usually, NVRAM used to save large bitmaps(or large area black area) which cannot print by `cmdBitmapPrint()` or `cmdBitmapPrintEx()`.

\* Print bitmap form NVRAM by index

`void cmdPrintBitmapFromNVRAM(int index, int zoom)`

parameters: `index` bitmap index (1~255)

`zoom` zoom mode

BITMAP\_ZOOM\_NONE: original Size

BITMAP\_ZOOM\_WIDTH: double width

BITMAP\_ZOOM\_HEIGHT: double height

BITMAP\_ZOOM\_BOTH: double width and double height

\* Delete all bitmaps form NVRAM

`void cmdDeleteBitmapFromNVRAM()`

\* Print a test page

`void cmdPrintTest()`

\* Get printer model(Only valid for serialport printer)

`int cmdGetPrinterModel()`

\* Cut paper(Only valid for usb printer)

`void cmdCutPaper(int mode)`

parameters: `mode` FULL\_CUT,HALF\_CUT

## 4.5 FAQ

Question:

Print bitmap error or blank

Solution:

1. Adjust heating parameter [cmdSetHeatingParam\(\)](#) for bitmap or try [cmdBitmapPrintEx\(\)](#) to do split printing. You could refer to the Demo of the function [printLogo\(\)](#) in the file [PosSalesSlip.java](#). We recommend all bitmap print from NVRAM. Generally, the more black areas in the bitmap, the heating time needs to be reduced. You can use the ImageUtils class to convert color bitmap to black and white bitmap.

2. The width of the bitmap should not exceed the valid printing width, the valid width of 58mm printer is 48mm corresponds to bitmap width is 384px and 80mm printer valid width is 72mm corresponds to bitmap width is 576px.

**Note:** More print density (more slow print speed) more delay times. Also don't invoke printing command in multiple thread. When the battery is powered, the lower the voltage, the slower the print speed, and the low voltage may cause the bitmap printing failed.

Question:

Print language error

Solution:

The common reason was the code page mismatch, the code page must be match with string encoding. It means you need to use the matching parameters to function [setStringEncoding\(\)](#) and [cmdSetPrinterLanguage\(\)](#).

More details to see the Demo function [languageTest\(\)](#) in the file [PosSalesSlip.java](#).

You could try print string by bitmap if your language could not work correctly. You could see Demo function [printUnsupportLanguage\(\)](#) in the file [PosSalesSlip.java](#).

**In addition, pay attention to the printing process. Some users send other commands or invoke connect() repeatedly during the printing process, which will cause garbled printing or printer abnormal. When printers are abnormal, you can invoke fwUpdate() to repair, and do not operate the printer in multiple threads, used queue for multiple task. The correct process of a printing task is connect->check paper->printing->disconnect**

Question:

How to printing via external printer

Solution:

For external serialport printer invoke [selectPrinter\(String path, int speed\)](#), for external usb printer invoke [Usbprinter.getPrinterList\(\)](#) to list all printer then [selectPrinter\(UsbDevice device\)](#). Note, maybe different brand printer have difference on ESC commands, you could invoke [sendCmd\(byte\[\] cmd\)](#) to send special commands.



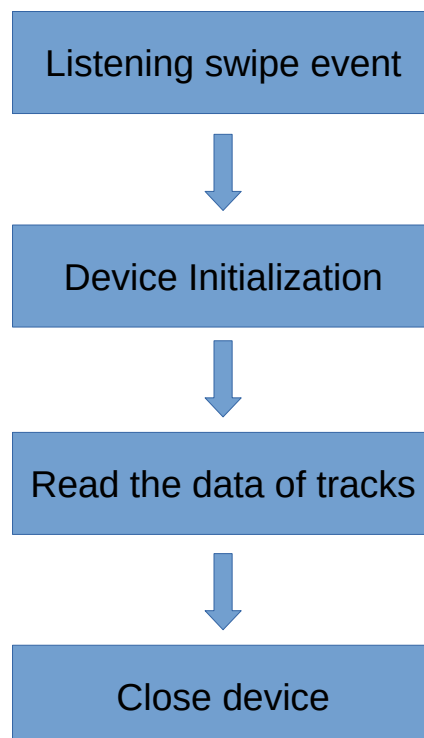
## 5. Magnetic Card Reader

### 5.1 Overview

Magnetic card(also known as magnetic stripe card), related standards: ISO7810, ISO7811(parts 1-6), ISO7812, ISO7813, ISO15457.

Magnetic card reader can read many magnetic stripe card information, such as credit cards, bank cards, membership cards, shopping cards, etc.

### 5.2 Workflow



## 5.3 API

Throws error code:

<code>public static final int MCR_ERR_INIT</code>	<code>//initialization error</code>
<code>public static final int MCR_ERR_DEINIT</code>	<code>//close error</code>
<code>public static final int MCR_ERR_NO_INIT</code>	<code>//not initialized</code>
<code>public static final int MCR_ERR_ALREADY_INIT</code>	<code>//already initialized</code>
<code>public static final int MCR_ERR_PARAM</code>	<code>//parameter error</code>

Internal error code:

```
//Preamble error in card read data
public static final int MCR_ERR_CODE_PREAMBLE    = 0x51;

//Postamble error in card read data
public static final int MCR_ERR_CODE_POSTAMBLE    = 0x52;

//LRC error in card read data
public static final int MCR_ERR_CODE_LRC          = 0x53;

//Parity error in card read data
public static final int MCR_ERR_CODE_PARITY        = 0x54;

//Blank track
public static final int MCR_ERR_CODE_BLANK_TRACK  = 0x55;

//STX/ETX error in command communication
public static final int MCR_ERR_CODE_STX_ETX      = 0x61;

//Class/Function un-recognizable in command
public static final int MCR_ERR_CODE_CLASS_FUNCTION = 0x62;
```

```
//BCC error in command communication
public static final int MCR_ERR_CODE_BBC = 0x63;

//Length error in command communication
public static final int MCR_ERR_CODE_LENGTH = 0x64;

//No data available to re-read
public static final int MCR_ERR_CODE_NO_DATA = 0x65;

//No more space available for OTP write
public static final int MCR_ERR_CODE_OTP_WRITE_FULL = 0x71;

//OTP write try without data
public static final int MCR_ERR_CODE_OTP_WRITE = 0x72;

//CRC error in read data from OTP
public static final int MCR_ERR_CODE_OTP_CRC = 0x73;

//No data stored in OTP
public static final int MCR_ERR_CODE_OTP_EMPTY = 0x74;
```

## API list:

\* Initialize the magnetic card reader

`void init()`

\* close the magnetic card reader

`void deinit()`

\* To query the status of the data

`int query()`

return: 0 have data

-1 no data

\* Reading data from specified magnetic track

`byte[] getTrack(int trackId)`

return: track data(hex byte arrays)

parameters: `trackId` track index(1, 2, 3)

## 5.4 FAQ

## 6.Smart Card Reader

### 6.1 Overview

Smart card reader supported card mode:

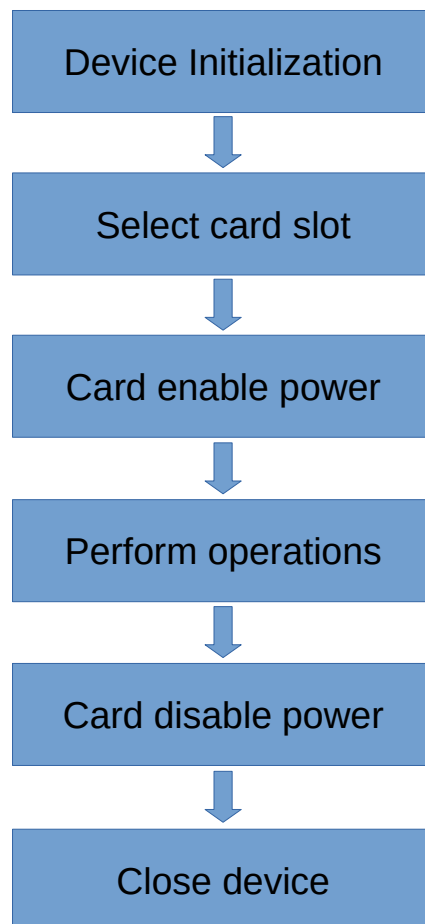
ISO7816/AT24C/AT45D/AT88SC160X/AT88SC10X/SLE4428/SLE4418/SLE4442/SLE6636. related standards: ISO7816(parts 1-5).

In ISO7816-4 defines two kinds of transport protocols:

T0 character transmission protocol

T1 block transfer protocol

### 6.2 Workflow



## 6.3 API

Throws error code:

```
public static final int SCR_ERR_INIT                //initialization error
public static final int SCR_ERR_DEINIT             //close error
public static final int SCR_ERR_NO_INIT            //not initialized
public static final int SCR_ERR_ALREADY_INIT       //already initialized
public static final int SCR_ERR_PARAM              //parameter error
public static final int SCR_ERR_CODE_SUCCESSFUL    //communication
successful
public static final int SCR_ERR_CODE_TRANSMIT_ERROR //transmit error
public static final int SCR_ERR_CODE_CMD_FAIL      //send command fails
public static final int SCR_ERR_CODE_CMD_BUSY      //device busy
public static final int SCR_ERR_CODE_NOT_SUPPORT   //command not
supported
public static final int SCR_ERR_CODE_NO_CARD       //no card
public static final int SCR_ERR_CODE_MEM_ERROR     //internal memory
error
public static final int SCR_ERR_CODE_TIMEOUT       //communication timeout
public static final int SCR_ERR_CODE_INVALID_PARAMETER //Invalid
parameter
public static final int SCR_ERR_CODE_NOT_INITIALIZED //card not
initialized
public static final int SCR_ERR_CODE_PBOC_FAIL     //PBOC fail
public static final int SCR_ERR_CODE_CARD_INACTIVE //card
inactive
```

```
public static final int SCR_ERR_CODE_CARD_LOCKED    //card locked
public static final int SCR_ERR_CODE_DEV_ERR
```

Card mode:

```
public static final int SCR_MODE_ISO7816
public static final int SCR_MODE_AT24C
public static final int SCR_MODE_SLE4428
public static final int SCR_MODE_SLE4442
public static final int SCR_MODE_AT88SC1608
public static final int SCR_MODE_AT45D041
public static final int SCR_MODE_SLE6636
public static final int SCR_MODE_AT88SC102
```

CCID protocol:

```
public static final int PROTOCOL_T0    //character transmission protocol
public static final int PROTOCOL_T1    //block transfer protocol
```

API list:

\* Initialize the smart card reader

`void init()`

\* Close the smart card reader

`void deinit()`

\* Select the slot

`void selectSlot(int slot)`

parameters: `slot` Slot number

**Note:** Double reader have slot 0 and 1, The single reader has only slot 0

\* Card enable power

`void powerOn()`

\* Card disable power

`void powerOff()`

\* Get the ATR (Answer To Reset)

`public byte[] getATR()`

return: ATR(Hex byte arrays)

\* Get the transport protocol of card

`public int getProtocol()`

return: `PROTOCOL_T0`

`PROTOCOL_T1`

\* Get the serial number of card

`public byte[] getSN()`

return: serial number(Hex byte arrays)

\* Send APDU command



`public byte[] sendAPDU(byte[] apdu)`

return: APDU ack(Hex byte arrays)

parameters: `apdu` APDU command(Hex byte arrays)

\* Send APDU command, for long response data

`public void sendAPDU(byte[] apdu, byte[] response, int[] responseLen)`

parameters : `apdu` format is CLA+INS+P1+P2+Lc+cmd+Le

`response` response bytes

`responseLen` array of response bytes len

Non ISO7816 card mode API reference doc and Demo.

## 6.4 FAQ

## 7.NFC Reader

### 7.1 Overview

Android standard API, support read-write mode. Compatible with ISO / IEC 14443 A / B, ISO / IEC 15693, MIFARE, FeliCa multiple standard cards, Demo shows how to read UID and read/write NFC tag.

### 7.2 FAQ

Refer:<https://developer.android.com/guide/topics/connectivity/nfc/nfc>

## 8.Camera

### 8.1 Overview

Camera Parameters: 5 megapixels / 13 megapixels , 720P@30fps preview, AF, AWB / AEC / AGC, Flashligh. The parameters of different models may be different.

### 8.2 FAQ

Camera is mainly used for bar code recognition and QR code recognition, user can reference these open source project.

ZXing:

<https://github.com/zxing/zxing/wiki/Getting-Started-Developing>

ZBar:

<https://github.com/ZBar/ZBar>

## 9.FingerPrint

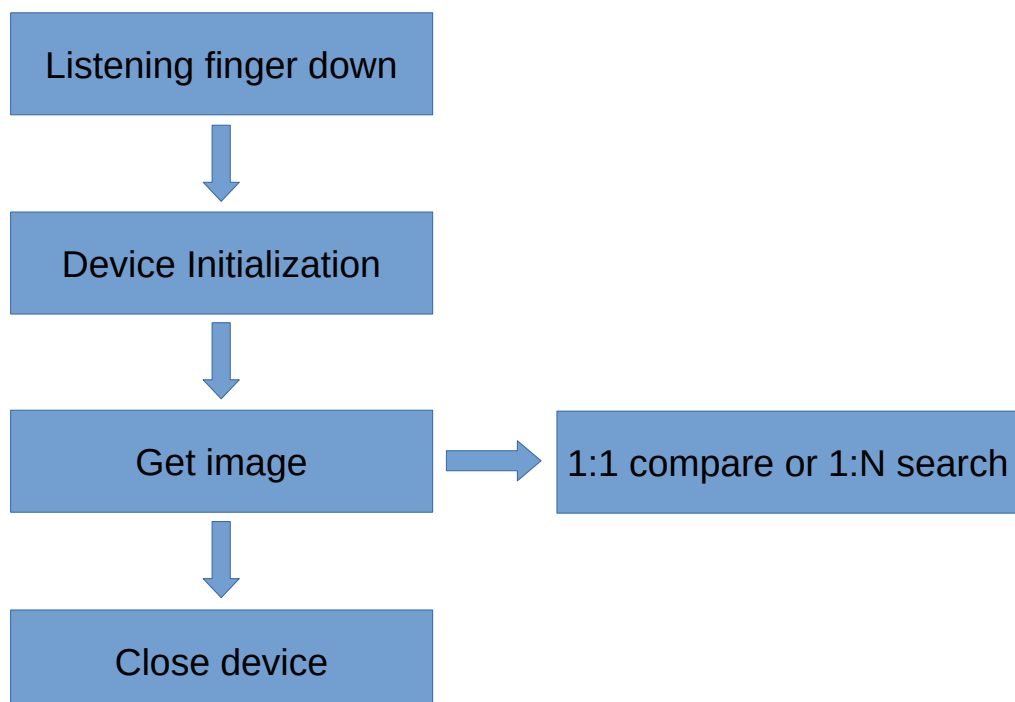
### 9.1 Overview

Fingerprint module provides the interface: read the fingerprint image, fingerprint minutia extraction, 1:1 comparison. Also you can implement fingerprint minutia extraction, 1:1 comparison by yourself.

Reference data:

- Image depth: 8bit
- Image size: 192x192
- DPI: 508

### 9.2 Workflow



## 9.3 API

public static final int FINGERPRINT\_FINGER\_PRESENT       //the keycode  
of the finger down

Throws error code:

```
public static final int FINGERPRINT_ERR_NO_INIT    //not initialized
public static final int FINGERPRINT_ERR_INIT       //initialization error
public static final int FINGERPRINT_ERR_DEINIT     //close error
public static final int FINGERPRINT_ERR_ALREADY_INIT //already
initialized
public static final int FINGERPRINT_ERR_NO_FINGER  //no finger
public static final int FINGERPRINT_ERR_IMAGE_UNCLEAR //image
unclear
public static final int FINGERPRINT_ERR_IO         //IO error
```

API list:

\* Initialize fingerprint sensor

[void init\(\)](#)

\* Close fingerprint sensor

[void deinit\(\)](#)

\* To get image of the finger

`byte[] getBitmapBytes()`

return: byte array of finger image(include 1078 byte that is bmp header info)

\* To get dimension of the image

`int getBitmapWidth()`

`int getBitmapHeight()`

\* To get the minutiae of current finger image(developing)

`byte[] getMinutiae()`

return: finger minutiae record(refer to ISO/IEC 19794-2:2005)

\* 1:1 comparison (developing)

`int verify(byte[] probe, byte[] candidate)`

## 9.4 FAQ

## 10.Secondary Display

### 10.1 Overview

The secondary display used to display a wide variety of content(such as: Video, Photo, provider password and account input for customers). Demo include a sample for secondary display.

Note: The single display device has only one LCD, you can implement secondary display via HDMI. The dual display device have double LCD.

Now we have two ways to show something on secondary display:

1. Presentation (Added in API level 17, Android 4.2)
2. Launching activity on secondary display (Added in API level 26, Android 8.0)

### 10.2 FAQ

Presentation:

<https://developer.android.com/reference/android/app/Presentation.html>

Launching activity on secondary display:

<https://developer.android.com/about/versions/oreo/android-8.0?#mds>

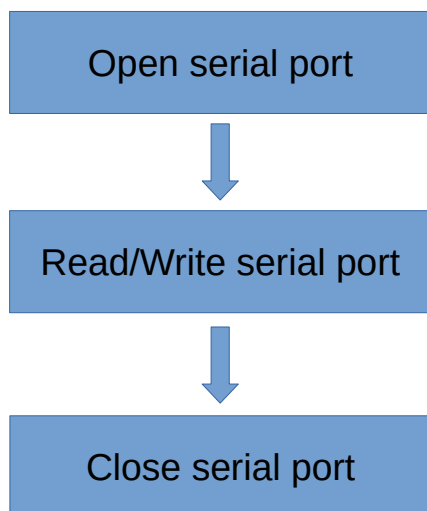
## 11.SerialPort

### 11.1 Overview

SmartPos device have one DB9 port (RS232 protocol), user could connect electronic weighing, extenal printer and others devices via serial port.

**Note:** Port node is `"/dev/db9"`. If you connect external device via USB serial then port node should be like these `/dev/ttyUSBx`(`ttyUSB0`, `ttyUSB1`, ...).

### 11.2 Workflow



## 11.3 API

API list:

\* Initialization

[SerialPort \(File device, int baudrate\)](#)

parameters: [device](#) serial port

[baudrate](#) baudrate

[SerialPort \(File device, int baudrate, int dataBits, int stopBits, char parity, char flowCtrl\)](#)

parameters: [device](#) serial port

[baudrate](#) baudrate 9600~4000000

[dataBits](#) data bits 7,8

[stopBits](#) stop bits 1,2

[parity](#) N/n,E/e,O/o (N-none, E-Even, O-Odd)

[flowCtrl](#) N/n,S/s,H/h (N-none, S-XON/XOFF, H-RTS/CTS)

\* To read data stream from serial port

[InputStream getInputStream\(\)](#)

\* Write data stream to serial port

[OutputStream getOutputStream\(\)](#)

\* Close serial port

[void close\(\)](#)

## 11.4 FAQ

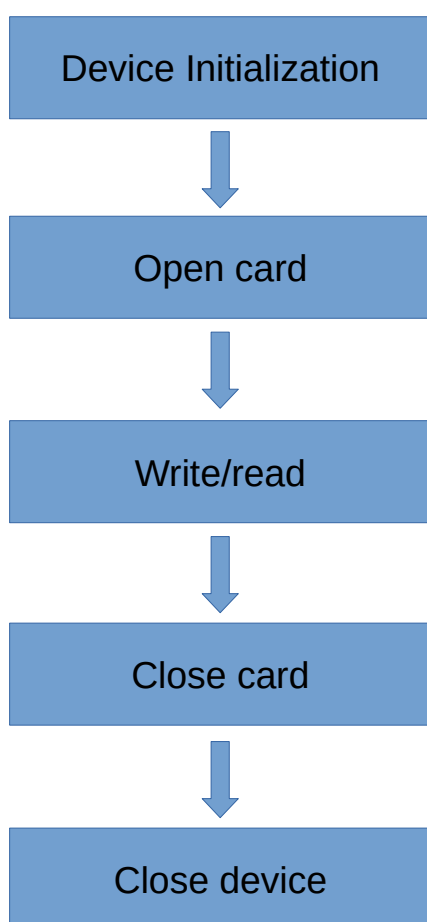


## 12. Secure Access Module(SAM/PSAM)

### 12.1 Overview

SmartPos support up to 2 card slots and support card baud rate 9600, 19200, 38400.

### 12.2 Workflow



## 12.3 API

Card protocol type:

```
public static final int CARD_TYPE_T0;  
public static final int CARD_TYPE_T1;
```

Card slot value:

```
public static final int CARD_SLOT_1;  
public static final int CARD_SLOT_2;
```

Card baud rate:

```
public static final int CARD_BPS_9600;  
public static final int CARD_BPS_19200;  
public static final int CARD_BPS_38400;
```

Card params:

```
public static final int CARD_VOLTAGE_1V8; //Class A card  
public static final int CARD_VOLTAGE_3V0; //Class B card  
public static final int CARD_VOLTAGE_5V0; //Class C card  
public static final int CARD_PPS_NOSUPPORT; //card not support PPS  
public static final int CARD_PPS_SUPPORT; //card support PPS  
public static final int CARD_PPS_ENFORCE; //enforce PPS, maybe no effect  
public static final int CARD_BPS_9600; //ATR speed 9600  
public static final int CARD_BPS_19200; //ATR speed 19200
```

```
public static final int CARD_BPS_38400; //ATR speed 38400
public static final int CARD_EMV_SPEC; //EMV card, such as bank card
public static final int CARD_ISO_SPEC; //normal ISO7816 card
public static final int CARD_NOCHECK_SPEC; //some special card, such as
SIM card
```

API error code:

```
public static final int ERR_NODEV;           //SAM reader no found
public static final int ERR_PARAM ;          //param error
public static final int ERR_NOINIT;         //reader no init
public static final int ERR_SLOT;           //slot value invalid
public static final int ERR_WRITE;          //write error
public static final int ERR_READ;           //read error
public static final int ERR_OOM;            //Out of memory
public static final int ERR_RESPONSE;       //card response error
public static final int ERR_RESPONSE_SUM;   //response checksum error
public static final int ERR_CARD_DETECT;    //card no found
public static final int ERR_CARD_OPEN;      //card open error
public static final int ERR_CARD_EXCHANGE;  //APDU exchange error
```

## API list:

### \* Initialization SAM reader

`int init ()`

return: 0 is success, otherwise is error code

### \* Deinitialization SAM reader

`void deinit()`

### \* Select card

`int cardOpen(int slot, int config, byte[] atr)`

parameters: `slot` card slot index value

`config` card param

bit0-1: card voltage

bit2-3: PPS param

bit4-5: card baudrate

bit6-7: card spec

`atr` card ATR, format is 2byte lenght(MSB) + ATR bytes

return: 0 is T0 card, 1 is T1 card. otherwise error code

### \* Close card

`void cardClose(int slot)`

parameters: `slot` card slot index value

### \* Check card

### `int cardDetect(int slot)`

parameters: `slot` card slot index value

return: 0 card exist, otherwise error code

#### \* Transmit

### `int cardExchange(int slot, byte[] command, int cmdLen, byte[] response, int respLen, int timeoutMs)`

parameters: `slot` card slot index value

`command` APDU command

`cmdLen` APDU command length

`response` APDU response buffer

`respLen` APDU response buffer length

`timeoutMs` APDU response read timeout in millisecond

return : the real lenght of APDU response

## 12.4 FAQ

Since SDK 1.3.7 we start use new SAM reader, it support extended APDU. Because of some card respond speed, you can try increase `timeoutMs` to fix response read timeout problem.

## 13.Cash Drawer

### 13.1 Overview

SmartPos device has two ways to connect the cash drawer:

- 1.Built-in RJ11 interface
- 2.USB to RJ11 interface

### 13.2 API

API list:

\* Open cash drawer via built-in RJ11 port

`static void open()`

\* Open cash drawer via USB-RJ11 converter

`static void openEx()`

### 13.3 FAQ

## 14.Scanner

### 14.1 Overview

Scanner is a module used to identify barcode or QR code. Normally is USB port or Serial Port.

For USB interface, the scanner is HID mod and scanning result ends with the character "\n". For serial port interface, scanning result output via serial port.

### 14.2 FAQ

Some USB scanning modules can be switched to virtual serial port mode and read results via node `/dev/ttyACMx`.

Demo code scanning only shows how to read the scanning results of HID mode, The serial port mode is read via the serial port. Please refer to the document of the scanning module for details.

## 15.LED

### 15.1 Overview

LED API could control LEDs on device. Now only FH101A1-D model (cpu version RK3568) have LEDs.

### 15.2 API

API list:

\* Sets Red LED

`void setRedLed(boolean on)`

parameters: `on` true, turn on LED  
false, turn off LED

\* Sets Green LED

`void setGreenLed(boolean on)`

parameters: `on` true, turn on LED  
false, turn off LED

\* Sets Blue LED

`void setBlueLed(boolean on)`

parameters: `on` true, turn on LED  
false, turn off LED



## 16.Appendix

### 16.1 Restore the printer firmware

Refer to `fwUpdate(getAssets(), "fw.bin")` in Demo source. You can try this function to repair while printer can't work correctly. Update duration time about 12 second, don't interrupt!

**Note:** only built-in serial port printer supported.

### 16.2 Full screen mode(Kiosk mode)

In full screen mode, navigation bar and status bar always hidden.

```
Intent intent = new Intent();  
intent.setAction("com.paydevice.action.CONTROL_STATUSBAR");  
intent.putExtra("cmd", "hide");  
sendBroadcast(intent);
```

exit full screen mode

```
Intent intent = new Intent();  
intent.setAction("com.paydevice.action.CONTROL_STATUSBAR");  
intent.putExtra("cmd", "show");  
sendBroadcast(intent);
```