SUNMI External Printer

Developer Documentation

Introduction

There are two main SUNMI printer types: inbuilt printer and external printer. This document introduces how to use APIs we provided to get your external printers ready for use quickly. For inbuilt printer documentation, please see **Inbuilt Printer Documentation**.
You can call various print interface libraries in a fast way through simple API integration, and the interface libraries basically have all the functions including printing layout, pattern setting, etc., and will be updated regularly to save your time on learning printer configuration, ESC commands, etc.
Currently, printers supported by SDK for connection include 58mm Cloud Printer, 80mm Kitchen Cloud Printer, 58mm Thermal Receipt Printer and so on.

Quick Start

1. Automatic integration:

Add dependencies and attributes configuration in the program build.gradle file:

```
dependencies{
 compile 'com.sunmi:external-printerlibrary:1.0.9'
}
```

Note: the dependency library is located in the maven central repository, so mavenCentral() support is required.

2. To avoid obfuscation, add the following configuration to proguard-rules.pro obfuscation file:

```
-dontwarn com.sunmi.externalprinteribrary.**
-keep public class com.sunmi.externalprinterlibrary.**{*;}
```

3. Select the corresponding enumeration type for SUNMI printer and specify it as the operation object of SDK. The available types are:

```
//enumeration types for a printer connected via a USB interface
SunmiPrinter.SunmiCloudPrinter -- connect a SUNMI 58mm Cloud Printer
SunmiPrinter.SunmiKitchenPrinter-- connect to SUNMI 80mm Kitchen Cloud Printer
SunmiPrinter.SunmiNTPrinter -- connect to SUNMI 58mm Thermal Receipt Printer

//enumeration type for a printer connected via a LAN
SunmiPrinter.SunmiNetPrinter -- all SUNMI cloud printers connected via a LAN

//enumeration type for a printer connected via Bluetooth
SunmiPrinter.SummiBlueToothPrinter – all SUNMI cloud printers connected via Bluetooth
```

4. Specify a printer to print

```
//1. Specify 58mm Thermal Receipt Printer as the operation object of SDK
SunmiPrinterApi.gtInstance.setPrinter(SunmiPrinter.SunmiNTPrinter);

//2. Connect printer
SunmiPrinterApi.getInstance().connectPrinter(context, new
ConnectCallback{

void onFound(){
//If the printer is found, it will call back this method
}
 void onUnfound(){
//If the printer is not found, it will call back this method
```

```
}
 void onConnect(){
//It will call back this method after successful connection, and the printer is ready
}
 void onDisconnect(){
//It will call back this method if the printer is disconnected during the connection, and the printing task will be interrupted
}
});

//3. Call the printing interface and complete the printing task
SunmiPrinterApi.getInstance().printInit();
SunmiPrinterApi.getInstance().printText("123456789\n");
SunmiPrinterApi.getInstance().printQrCode("123456789, 4, 0);

//4. Complete and disconnect the printer
SunmiPrinterApi.getInstance().disconnectPrinter(context);
```

5. Specify multiple printers connected via a LAN to print (same for the printers connected via Bluetooth)

```
//1. Specify and connect multiple IP printers connected via LAN
SunmiPrinterApi.getInstance().setPrinter(SunmiPrinter.SunmiNetPrinter,
"192.168.1.90");
SunmiPrinterApi.getInstance().connectPrinter(context,new ConnectCallback{

 void onFound(){
//If the printer is found, it will call back this method
}
 void onUnfound(){
//If the printer is not found, it will call back this method
}
 void onConnect(){
//It will call back this method after successful connection, and printing is ready
}
 void onDisconnect(){
//It will call back this method if the printer is disconnected during the connection, and the printing task will be interrupted
}

});
SunmiPrinterApi.getInstance().setPrinter(SunmiPrinter.SunmiNetPrinter,
"192.168.1.91");
SunmiPrinterApi.getInstance().connectPrinter(…);

//2. Specify all the printers connected via LAN to print after successful connection
SunmiPrinterApi.setPrinter(SunmiPrinter.SunmiNetPrinter);

//3. Call the printing interface and complete printing task
SunmiPrinterApi.getInstance().printInit();
SunmiPrinterApi.getInstance().printText("123456789\n");
SunmiPrinterApi.getInstance().printQrCode("123456789, 4, 0);

//4. Disconnect all the printers in turn after completion
SunmiPrinterApi.getInstance().setPrinter(SunmiPrinter.SunmiNetPrinter,
"192.168.1.90");
SunmiPrinterApi.getInstance().disconnect(context);
SunmiPrinterApi.getInstance().setPrinter(SunmiPrinter.SunmiNetPrinter,
```

"192.168.1.91");
SunmiPrinterApi.getInstance().disconnect(context);

Introduction to APIs:

1. Specify a printer for the SDK

Method	void setPrinter(SunmiPrinter sunmiPrinter, String… address)
Description	Specify a printer for the SDK; You can specify the printer connected via the specified LAN or Bluetooth, and if so, a specific LAN or Bluetooth address is needed first.
Input	sunmiPrinter: specify the printer's enumeration type; address: when the enumeration type is SunmiPrinter.SunmiNetPrinter or SunmiPrinter.SunmiBlueTooth: Control a printer when specifying the specific operation address; Control all the printers connected via LAN or Bluetooth when not specifying the specific address.

2. Connect to the printer

Method	void connectPrinter(Context context, ConnectCallback callback)
Description	Connect to the printer specified by the SDK.
Input	Context: dialog context; Callback: printer connection result callback.

3. Disconnect the printer

Method	void disconnectPrinter(Context context)
Description	Complete and disconnect the printer with specified enumeration type, and release resource.
Input	Context: dialog context

4. Get the mac address list of printers connected via Bluetooth

Method	void disconnectPrinter(Context context)
Description	Get the Bluetooth address of SUNMI cloud printer bound to the device.
Input	Context: dialog context
Return	Bluetooth address of the SUNMI cloud printer connected

5. Get the connection status of the printer with specified enumeration type

Method	boolean isConnected()
Description	Whether the printer with the specified enumeration type is being connected; The LAN/Bluetooth printer without a specified address cannot be returned.
Input	void
Return	True connecting False not connected

6. Printer initialization

Method	void printerInit()
Description	Initialize the printer and restore the attributes previously set to default status.
Input	None
Return	None

7. Move the paper according to line height

Method	void lineWrap(int n)
Description	The printer moves the paper for n lines; If there is data in the printer buffer area, the printer will output the data and move the paper for n lines; The moving distance is 0 if the line height is set to 0.
Input	Number of lines (0<n<256)
Return	None

8. Move the paper according to pixels

Method	void pixelWrap(int n)
Description	The printer moves the paper for n lines; If there is data in the printer buffer area, the printer will output the data and move the paper for n lines.
Input	Number of pixels (0<n<256)
Return	None

9. Refresh buffer area

Method	void flush()
Description	Refresh printer buffer area. Output when there's data in buffer area, and feed a line of paper when there's no data.
Input	None
Return	None

10. Horizontal anchor

Method	void tab()
Description	Move the printing position to the next horizontal anchor; If the horizontal anchor exceeds the printing area, move to the end of the line; If it's already at the end of a line, a linefeed will be implemented.
Input	None
Return	None

11. Set the horizontal anchor

Method	void setHorizontalTab(int[] k)
Description	Mark the horizontal anchor, and each anchor is specified by k[n] ascii characters width, 8 ascii characters width by default.
Input	k: horizontal anchors array k[n], with the biggest value for n being 32, and 0<k[n]<256; The array must be in ascending order, or an exception will happen; The default anchor will be restored when k is set to null, and by default there is an 8-characher wide interval between two anchors.
Return	None

12. Print text

Method	void printText(String text)
Description	It will convert the text into a hexadecimal byte stream with corresponding characters encoding; Printing service will convert text into gb18030 code by default.
Input	text: the text to be printed
Return	None

13. Set gb18030 character set encoding

Method	void setGb18030CharSet(boolean set)
Description	The printer can recognize gb18030 character set by default, and it can't recognize it when turning

	off the setting, and will process it as a single byte.
Input	Whether to recognize the gb18030 character set.
Return	None

14. Set printer's mapping page table

Method	void setPageTable(int n)
Description	This method is only valid when setGb18030CharSet（false） when mapping single-byte code 0x80-0xff on different pages.
Input	n: corresponding page table
Return	None

15. Set the international character set encoding

Method	void setInterCharSet(int n)
Description	Some special characters in ascii code vary in different countries, and you select a country to print its corresponding characters.
Input	n: corresponding country
Return	None

16. Set the right sidebearing of characters

Method	void setCharRightSpace(int n)
Description	Set the right spacing of characters, and it is only valid for ascii code, invalid for other characters.
Input	n: number of pixels indicating the sidebearing
Return	None

17. Set the left and right sidebearing between Chinese characters

Method	void setHanziSpace(int m， int n)
Description	Set the left and right spacing between Chinese characters.
Input	m: left sidebearing n: right sidebearing
Return	None

18. Set font size

Method	void setFontZoom(int hori, int veri)
Description	Due to the limitation of the printer's dot matrix fonts, the font size can only be multiplied, so this method can control the font magnification in horizontal and vertical directions.
Input	The range of hori and veri is 1-8, indicating the times the font can be multiplied horizontally and vertically. If it is set outside the range, an error parameter will be returned.
Return	None

19. Set alignment mode

Method	void setAlignMode(int type)
Description	Set the alignment mode of the printing content
Input	0: left (default), 1: center, 2: right
Return	None

20. Set absolute position

Method	void moveAbsolutePos(int pos)
Description	Move the printing position to the specified pixel from the starting position.
Input	Pos: pixel from the starting position

Return	None

21. Set relative position

Method	void moveRelativePos(int pos)
Description	Move the printing position to the specified pixel relative to the current position.
Input	Pos: pixel from the current position
Return	None

22. Enable or disable underline

Method	void enableUnderline(boolean enable)
Description	Enable or disable the underline function of characters.
Input	Is ‘enable’ on or off
Return	None

23. Enable or disable bold

Method	void enableBold(boolean enable)
Description	Enable or disable the bold function of characters.
Input	Is ‘enable’ on or off
Return	None

24. Enable or disable overlap effect (usually the same as bold)

Method	void enableBold(boolean enable)
Description	Enable or disable the bold function of characters.
Input	Is ‘enable’ on or off
Return	None

25. Set the line spacing for the printer

Method	void setLineSpace(int value)
Description	Set the line spacing for the printer; When the line spacing is smaller than the character height, the character height will be adopted.
Input	value: when the value is smaller than 0, it will restore the default line spacing, otherwise it will set the line spacing to the corresponding value.
Return	None

26. Set the left margin for the printer

Method	void setLeftSpace(int value)
Description	Set the number of pixels indicating the left margin If the value is too big, the printing area may be invalid.
Input	value: the size of left margin
Return	None

27. Set print width

Method	void setPrintWidth(int width)
Description	Set the number of pixels indicating the print width.
Input	width: print width
Return	None

28. Print a barcode

Method	void printBarCode(String code, int type, int width, int height, int hriPos)

Description	Print a custom barcode.
Input	code: barcode (it shall be in the format as required by the specified barcode type); type: barcode type 0: UPC-A, 1: UPC-E, 2: EAN13, 3: EAN8, 4: CODE39, 5: ITF, 6: CODABAR, 7: CODE93, 8: CODE128, See barcode type remark; width: barcode width 2-6 pixels (a too wide barcode won't be printed); height: barcode height 1 – 255 pixels; hriPos: HRI position 0: No print, 1: Above the barcode, 2: Below the barcode, 3: Above and below the barcode
Return	None

29. Print a QR code

Method	void printQrCode(String code, int modeSize, int errorlevel)
Description	Print a QR code.
Input	code: QR code to be printed, utf-8 character set by default; modeSize: size of QR code, 1-16 pixels; errorlevel: error correction level of QR code, four levels: 0, 1, 2, 3.
Return	None

30. Method 1 for printing an image

Method	void printBitmap(Bitmap bitmap, int mode)
Description	The bitmap is converted into raster bitmap for printing; This method is suitable for printing images with width within the printing paper; Note that if the data is too large, the sending may fail. In this case, you can use mode to set multiple magnification.
Input	bitmap: the opaque bitmap image to be printed; mode: 0: common, 1: double width, 2: double height, 3: double height double width
Return	None

31. Method 2 for printing an image

Method	void printBitmap2(Bitmap bitmap, int mode)
Description	Convert the image into bitmap for printing; You need to set the line spacing to 0 when using this method; This method differs from printBitmap in that it does not cause printer uninitialized gibberish after data failure; Note: if the data is too large, the sending may fail. In this case, you can use mode to set multiple magnification.
Input	bitmap: the opaque bitmap image to be printed; mode: 0:8 point single density 1:8 point double density 32:24 point single density 33:24 point double density (original size)
Return	None

32. Cutter cutting paper

Method	void cutPaper(int m, int n)
Description	Paper cutting.
Input	m: paper cutting mode, 0: full cut, 1: partial cut, 2: cut when feeding the paper n: paper feeding distance, this parameter will be valid only when m=2. Due to different types of printers, the distance from cutter to print head varies. Only when n=0, automatic paper feeding will take this distance, and extra distance set will be taken when n>0.
Return	None

33. Print a form

Method	void printColumnsText(String[] colsTextArr, int[] colsWidthArr, int[] colsAlign)

Description	Output the printing result in a form format, and each array represents the data and format on this column. You need to call this method for several times to achieve the form format. ! Note: calling this method will initialize the mode set before.
Input	colsTextArr: the content to be printed in each column, Chinese and ascii code supported; colsWidthArr: the maximum number of characters that can be contained in a column, and the characters are expressed in ascii code (one Chinese character is equal to two ascii characters). When the text exceeds the maximum number of characters that can be contained in a column, the text will be moved to the next row of the column. When the maximum number of characters in all columns exceeds the maximum number of characters that can be contained in a row, the text will not be printed; colsAlign: the alignment of the content of each column, and it is only effective when the number of text characters is less than the maximum number of characters.
Return	None

34. Send data commands

Method	void sendRawData(byte[] cmd)
Description	Send raw esc control commands.
Input	cmd: epson commands sent
Return	None

35. Get printer status

Method	int getPrinterStatus()
Description	Get printer's operating status; The operating status of LAN/Bluetooth printers without specified address cannot be obtained.
Input	None
Return	-1 Printer is off-line or the printing service is not connected to the printer; 0 Pinter is under normal operation; 1 Cover is open; 2 Out of paper; 3 Low on paper; 4 Printer is overheated.

36. LAN transaction printing interface of cloud printer

For scenarios requiring LAN printing, we provide the following extension methods. You can enable and submit the information before and after printing to obtain the actual printing results.
You can call by the following method:
Enable transaction - > Print SDK - > Submit transaction - > Wait for The printing result callback
The detailed methods are as follows:

Enable transaction mode

Method	void startTransBuffer(boolean isClear)
Description	Enable transaction mode to control the implementation order and implementation status of printing task. You can only get the printing result accurately under the transaction mode. After enabling transaction mode, all the printing commands will be cached, and will be sent to printer when submitting. The printer will not print when the data is not submitted. Note: currently only LAN printing of cloud printer supports transaction mode.
Input	isClear: whether to clear the cached data not submitted.
Return	None

Submit transaction mode

Method	void commitTransBuffer(TransCallback callback)
Description	Submit this transaction and get the implementation result of this transaction printing; It's suggested to wait for the submission result before submitting the next transaction;

	Note: currently only LAN printing of cloud printer supports transaction mode.
Input	Callback: the transaction submission result
Return	None

Exit transaction mode

Method	void endTransBuffer()
Description	Exit transaction mode; The subsequent printing commands will be directly sent to the printer. Note: currently only LAN printing of cloud printer supports transaction mode.
Input	None
Return	None