Sunmi-K2 Print Service Development Documentation

Introduction:

Sunmi printers mainly include built-in printers and external independent printers. This document mainly introduces how to quickly use the external independent printers on the personal computers through the API provided by Sunmi.

This document are adapted to use the AIDL interface to call the printer function to connect.

Connect print service via AIDL method:

AIDL is the abbreviation of Android Interface Definition language, which is a type of descriptive language of the communication interface of Android internal process. Through it we can define the communication interface between processes. Sunmi AIDL provides encapsulated common printing commands to make it easy for the developers to quickly access Sunmi printer. Meanwhile, Sunmi also supports all the common "ESC/POS" command set, capable of directly sending command set via the interface to control the printer.

The establishment of connection can be divided into the following 5 steps:

1. Add AIDL document (also including java document as for partial models) attached to the resource file in the project.
2. Realize ServiceConnection in the code class controlling the print.
3. Invoke ApplicationContext.bindService() and transfer in the realization of ServiceConnection. Note: bindservice is non-blocking invoke, which means the immediate binding after the completion of invoke is not successful, and it must be subject to serviceConnected.
4. In the realization of ServiceConnection.onServiceConnected(), you will receive an IBinder example (Service to be invoked). Invoke ExtPrinterService.Stub.asInterface(service) and transform the parameters into the corresponding print service type in Aidl document.
5. Now it's OK to invoke various methods defined in Aidl interface to carry out printing.

Pseudocode example:

Bound Service

```
Intent intent = new Intent();
intent.setPackage("com.sunmi.extprinterservice");
intent.setAction("com.sunmi.extprinterservice.PrinterService");
bindService(intent, serviceConnection, Context.BIND_AUTO_CREATE);
```

It is necessary to establish a new ServiceConnection service to bind the callback

```
ServiceConnection serviceConnection = new ServiceConnection() {
@Override public void onServiceConnected(ComponentName name, IBinder service) {
ExtPrinterService ext = ExtPrinterService.Stub.asInterface(service);
}
@Override public void onServiceDisconnected(ComponentName name) { }
};
```

Use ext object to realize one's own printing task

```
ext.printText("123456\n");
```

Unbind the service after the completion of the usage

```
unbindService(serviceConnection);
```

Interface API:

1、Initialization of the printer

Method	Int printerInit()
Description	As for the initialization of the printer, the data in the print cache will be cleared, yet it will not affect the receive cache. And get the properties of all the previous settings back to default state.
Input	None
Return	See return value of the interface

2、Obtain the printer status

Method	int getPrinterStatus()
Description	Obtain the operating status of the printer
Input	None
Return	–1 The printer is offline or the print service has not been connected to the printer 0 Normal operation of the printer 1 The printer is uncapped 2 Printer out of paper 3 The printer is going to be out of paper 4 Printer is overheating

3、Send data command

Method	Int sendRawData(byte[] cmd)
Description	Support sending original esc control commands
Input	cmd sent epson command

Return	See return value of the interface

4、Refresh the cache

Method	Int flush()
Description	Refresh the print buffer, when there is data in the buffer, it will output, if there is no data, it will feed one line of paper
Input	None
Return	See return value of the interface

5、Print text content

Method	Int printText(String text)
Description	This method will transform the input text into hexadecimal byte stream corresponding to character set coding The print service transforms the text content into gb18030 coding by default
Input	text need to be printed
Return	See return value of the interface

6、Print bar code

Method	Int printBarCode(String code, int type, int width, int height, int hriPos)
Description	Print the bar code of the custom content
Input	code Content of bar code （bar code content need to meet its format according to different bar code types) type Bar code type 0：UPC-A 1：UPC-E 2：EAN13 3：EAN8 4:CODE39 5:ITF 6:CODABAR 7:CODE93 8:CODE128 See remarks on bar code type width Bar code width 2-6 pixels （If the set bar code width is too large, the exceeded part of the whole bar code to the paper width will not output bar code content) height Bar code height 1-255 pixels HRI Pos HRI position 0：Not to print 1：Upward side of bar code 2：Downward side of bar code 3：Upward & downward side of bar code
Return	See return value of the interface

7、Print QR code

Method	Int printQrCode(String code, int modeSize, int errorlevel)
Description	Print QR code
Input	code QR code content to be printed, utf-8 character set by default mode Size Size of QR code block 1-16 pixels error level Error correction level of QR code 0-3 four levels
Return	See return value of the interface

8、Print pictures

Method	Int printBitmap(Bitmap bitmap, int mode)
Description	Transform bitmap picture into raster bitmap picture for printing This method is applicable for pictures with a print width within the print paper The serial port flow control mode will be blocked when the data are too large, and the data will be lost when there is no flow control
Input	Bitmap non-transparent image to be printed mode0：Common 1：Double width 2：Double height 3：Double height & double width
Return	See return value of the interface

9、Print tables

Method	Int printColumnsText(String[] colsTextArr, int[] colsWidthArr, int[] colsAlign)
Description	Print content is outputted via table. Each array indicates the data & format on this column, and you need toinvoke this method several times to reach the style effect of table output
Input	**colsTextArr:** content of each column to be printed, supporting Chinese & ascii code **colsWidthArr：** maximum character amount to be contained in each column.Take ascii code number as unit as for character amount (one Chinese character equals 2 ascii code amount) When the text content exceeds the max amount to be contained, it will move to the next row of this column. The print will not be carried out if max

	character amount of all the columns exceeds that to be contained by one row **colsAlign：** alignment of content of each column. It will be effective only when the character amount
Return	See return value of the interface

10、Paper feed of the printer according to row height

Method	Int lineWrap(int n)
Description	Paper feed of the printer for n If there are data in the printer's print cache, the data will be outputted and the paper will be fed If the row height is set as 0, the paper feed distance will be 0
Input	Row number （0<n<256）
Return	See return value of the interface

11、Paper feed of the printer according to pixel

Method	Int pixelWrap(int n)
Description	Paper feed of the printer for n rows of dots If there are data in the printer's print cache, data will be outputted and the paper will be fed for n rows of dots
Input	Pixel （0<n<256）
Return	See return value of the interface

12、Horizontal positioning

Method	Int tab()
Description	Move the print position to the next horizontal positioning If the horizontal positioning point exceeds the print area, move to the end of row If it is already at the end of the line, a newline operation will be performed
Input	None
Return	See return value of the interface

13、Set lateral horizontal positioning point

Method	Int setHorizontalTab(int[] k)
Description	The position of the horizontal positioning will be marked. The position of each mark point will be specified by the character width of k[n] ascii. The default horizontal positioning point takes the character width of 8 ascii as a positioning point
Input	k lateral positioning point array k[n], the maximum length of the array (n) can reach 32, 0<k[n]<256 Within the array, there must be an ascending order, otherwise exception will be thrown out When k is set as null, default positioning point will be recovered, the default position is an interval of 8 characters
Return	See return value of the interface

14、Set alignment

Method	Int setAlignMode(int type)
Description	Set the alignment of the print content
Input	0 Left aligned (default)、1 Centered、2 Right aligned
Return	See return value of the interface

15、Set font size

Method	Int setFontZoom(int hori, int veri)
Description	Due to limitation by the printer hard font library, the font size can only be amplified by multiples. This method can be applied to control the amplification of font in lateral & longitudinal direction
Input	The range of hori 、veri is 1–8, indicating the amplification of font in lateral & longitudinal direction. Error parameter will be returned if the setting is out of the range
Return	See return value of the interface

16、Cutter's paper cutting

Method	Int cutPaper(int m, int n)
Description	Paper cutting
Input	m paper cutting mode 0：full cutting 1：half cutting 2：cutting paper feed n paper feed This parameter is valid only when the setting m=2. Due to different printer models, the distances from the cutter to the printing head are different. When n=0, the paper will be automatically fed for this distance. When n>0, it will be fed for additionally set distance
Return	See return value of the interface

17、Cash box control

Method	Void openDrawer()
Description	Cash box control，the external cash box will be open after calling this interface
Input	None
Return	None

18、Enable transaction mode（page mode)

Method	Void startTransBuffer()
Description	Similar to the page mode in the esc command, the subsequent printing command will be stored and printed after the transaction is submitted Please refer to Transaction printing instructions
Input	None
Return	None

19、End and commit the transaction（page mode)

Method	Boolean endTransBuffer()
Description	Similar to the page mode in the esc command, When the transaction mode is

	enabled, the print instructions sent later will be stored until the transaction is submitted by this interface before being sent to the printer, and synchronously waiting for the result of the printer processing in this transaction, and returning the result Please refer to Transaction printing instructions Note: This interface will block the process, so it cannot be called in the main thread
Input	None
Return	Success or failure of transaction printing

Appendix

Descriptions for the interface return value:

Since the print task carried out by the printer is asynchronous with sending data, the interface return does not represent whether the actual printing is successful or not. Except special instructions, the interface return values all indicate the reception conditions of the print service in terms of this command. Thus, it is possible that the printer can still receive the print data even in abnormal state such as lack of paper, etc. When the exception is recovered, the cache will be continuously carried out to print data;

When the return value >=0, it indicates that this command has been sent successfully, the printer will process it; As for the successfully returned concrete value, refer to the printer status return value;
When the return value <0, it indicates that sending of this command has failed, the print task will not be carried out. The concrete errors are as follows:

-1 The printer is off line or it is not ready
-2 The cache is full, unable to receive print data
-3 Exception of the data sending
-4 Errors of sending command or parameters

Descriptions of the bar code type:

Code	Description
code39	maximum 13 numbers in length to be printed
code93	maximum 17numbers in length to be printed

ean8	requires 8 digit number (the last one is check digit) with an effective length of 8 numbers
ean13	requires 13 digit number (the last one is check digit)
ITF	requires the inputmust be digits, with an effective length less than 14 digits, which must be of even bit
code 128	code128 are divided into three types: {A,{B,{C; A type: including capital letters, numbers, punctuation, etc. B type: capital & lower case letters, numbers; C type: pure numbers; As for default B type code, if you are going to use this code, it is necessary to add “{A”、“{B” 、“{C” before the content to declare the type to be used, and you can use a mixture of it, for example: “{A2344A”，”{C123123”，”{A1A{B13Bxc{C12”.

Status feedback

The printer status can be actively obtained by invoking the interface getPrinterStatus(); it can also be obtained asynchronously via registering broadcast:

Functions	action
Printable	com.sunmi.extprinterservice.NORMAL_ACTION
The printer is offline	com.sunmi.extprinterservice.OFLLINE_ACTION
The printer cover is not closed	com.sunmi.extprinterservice.COVER_OPEN_ACTION
Lack of the printer paper	com.sunmi.extprinterservice.OUT_OF_PAPER_ACTION
The printer paper is going to run out	com.sunmi.extprinterservice.LESS_OF_PAPER_ACTION
The printer is overheating	com.sunmi.extprinterservice.HOT_ACTION

Transaction printing instructions:

The transaction printing mode is suitable for the needs to control the printing content and get the printing result feedback (whether to print a small ticket). This mode is equivalent to establishing a transaction queue buffer. When the developer enters the transaction printing mode, a transaction queue will be opened. Add a printing method to it. At this time, the printer will not print the content immediately. After the transaction is submitted, the printer will execute the tasks in the queue in sequence, and the result of the transaction will be feedback when the execution is completed;

Pseudo code example:

```
startTransBuffer();//Enable transaction mode
printText();//Print text
printText();
…
printText();
endTransBuffer();//End and submit the transaction, the printer processes and prints the data
```

Note for transaction printing:

1. When the transaction mode is enabled, it is necessary to cooperate with the execution to end the commit transaction, otherwise all data will be cached and neither return nor print will be executed;
2. The transaction mode commit uses synchronous return, which will block the current thread until a result is returned, and will also affect subsequent connections
3. Called, so using transaction mode cannot be done in the main thread! ! !
4. A successful result will be returned if the printing is submitted successfully, but if the printer is abnormal, such as out of paper, overheating, etc., all command tasks in the submitted transaction will be lost, and the printing failed will be returned at the same time;