BluPrints

Thermal Receipt Printer

Developer Guide

(3inch-80/mm)

Aadharshila Mobility Solutions Pvt. Ltd.

E - 22, Sector – 51, Noida - 201301 UttarPradesh

Website : http://www.bluprints.in/

VERSION CONTROL: 5.0 / Jan 2018

PLEASE NOTE THAT THIS SDK VERSION, NAMELY, AEM_SDK5.1 IS APPLICABLE FOR AEM PRINTER FIRMWARE VERSIONS 5.0 AND ABOVE.

DOCUMENT NAME: ANDROID DEVELOPER GUIDE

RELEASE DATE: 5-Jan-2018

SDK SUPPORTED: AEM_ SDK5.0 FIRMWARE
SUPPORTED: 5.0 AND ABOVE

TABLE OF CONTENTS

SDK (Software Development Kit) for ANDROID

This document describes the use of AEM SCRYBE Thermal Printer SDK for Bluetooth, USB and WiFi Operations. This SDK provide an Interface between an Android Application and the AEM Thermal Printer. The SDK is android based, and requires at least 10 level of android SDK and 1.6 java compiler. It comprises of the following Interfaces, Classes and functions.

com.aem.api consists of Classes – AEMPrinter, AEMScrybeDevice, IAEMCardScanner and

AEMWifiPrinter.

AEMPrinter: AEMPrinter class handles all the Bluetooth related functions. These include:

1. SetLineFeed: public void setLineFeed(int noOfFeeds) throws IOException

In order to print specific number of blank lines or line feeds you need to call this function.

You need to call this function and pass desired number.

LINE_FEED=0X0A

2. print: public void printThreeInch(String text) throws IOException

This function is used to print a specified text (in the form of a string) to the printer.

3. printFormatedText: public void POS_S_TextOutThreeInch(String pszString,String encoding,0,int nWidthTimes,int nHeightTimes,int nFontTypes,int nFontStyle) throws IOException

In order to set the font size to Double width or double height of a particular line in the printer, you need to

call this function and pass the desired parameter of Double Width or Double Height.

the desired parameter DOUBLE_WIDTH or DOUBLE_HEIGHT and UnderLine as defined below.

DOUBLE_WIDTH =nScaleTimesWidth=1

DOUBLE_HEIGHT= nScaleTimesHeight=1

UNDERLINE=nFontStyle=0X100

encoding="US-ASCII"

pszString=String to be Printed

4. sendByteArray: public void sendByteArrayBT(byte[] byteArr) throws IOException

In order to send any of specific byte array (sequence) to the printer via Bluetooth, you need to call this function and pass the desired Byte Array.

5. lineSpace: public void POS_SetLineHeightThreeInch(int nHeight) throws IOException

In order to set Line height you need to use this method.

0<=nHeight>=255

6. Alignment: public void POS_S_AlignThreeInch(int align) throws IOException

In order to set character alignment you need to call this function. align=0 ->Left Alignment

align=1 ->Center Alignment align=2 ->Right Alignment

General Guidelines for Text Printing

1. Send the command to print the various fonts and styles.

2. If multiple commands are given then last command will be effective only.

3. Location to apply – at the start of the line to print

Rules to apply commands:

1. You can change the fonts, as well as Double width, Double height, Negative and underline characters. Command should be sent before the line to print.

2. Should not repeat the command more than one time at the start of line.

Functions for Non-Text Printing

11. printBarcode: public void printBarcodeThreeInch(String barcodeData, BARCODE_TYPE Btype, BARCODE_HEIGHT bHeight) throws IOException

BARCODE_TYPE_CODE39 = 0X45

This function is used to print a 2-Dimensional Barcode on the Printer, generated automatically according to the string passed to this function. The string should be of Capital English Letters or Numerals. The maximum characters in the strings can by up to 11. Under Barcode Type, the Printer supports BARCODE_TYPE_CODE39. The Barcode Height supported is DOUBLEDENSITY_FULLHEIGHT.

The following packet is generated internally within this function and sent to the printer:

barcodePacket[0] = 0x1D; 'GS'

barcodePacket[1] = 0x6B; 'k'

barcodePacket[2] = BARCODE_TYPE_CODE39; 0x45 barcodePacket[3] = (byte) (barcodeBytes.length + 2); //length of barcode data barcodePacket[4] = 0x2A;

barcodePacket[5] to barcodePacket[length of barcode string] = BarcodeBytes;

barcodePacket[Length of barcode string + 1] = 0x2A;

12. printImage: public void printImageThreeInch(Bitmap originalBitmap) throws IOException

This function is used to print an Image in the form of a Raster Image, based on the standard ESC/POS Raster Image command set of GS v protocol. You need to pass the bitmap of the desired image to be printed. Please note that this function can be used to print QR Codes as well, by first generating the QR code from a given string and thereafter, sending its bitmap to the PrintImage Function. The source code for generating the QR Code has been provided at the end of this document.

13. printTextAsImage: public void printTextAsImageThreeInch(String TextToConvert)throws IOException

Thisfunction is used to print any text (multilingual, Unicode type characters, etc) in the form of a Raster Image. You need to pass the String that you need to be printed as an image. The main utility of this function is that the user can easily input any multilingual string, that can be printed as is on the printer, so as to enable printing in any language irrespective of the fonts. (Printing characters of Urdu, Gujrati, Arabic, Oriya, Tamil, Tamil, Telugu, etc.). The benefit of this function is that the print is so clear and fast that there is no perceptible difference to an end user in understanding whether this text has been printed in the form of text or in the form of an image.

14. printBitImage: public void printBitImage(Bitmap originalBitmap, Context context, byte image_alignment)throws IOException

IMAGE_LEFT_ALIGNMENT = 0x6C;
IMAGE_CENTER_ALIGNMENT = 0x63;
IMAGE_RIGHT_ALIGNMENT = 0x72;

This is a deprecated function that has only been kept for backward compatibility. PrintBitImage fiunction uses the standard ESC * algorithm for printing of a bitmap

The maximum image size should be in the following range: - 355 X 500 (WxH) pixels.

For image printing, you need to half the pixel size of height of an image to get the desired width and height. E.g.: - Suppose you need to print a logo of dimensions 355 X 300 (WXH) pixels, then half the size of the height of an image i.e. 150 pixel for one time in your code by scaling function (as explained below). Width will remain same.

Bitmapscaled_bitmap = bitmap.createScaledBitmap(bitmap, 355, 150, false); 355 pixel= Width (Original Width)

150 pixel= Height (Height will get double i.e. 300 pixels as original height of an image while printing). Likewise, you can print the logo of desired dimensions. Note: -

Maximum Width size is 355 pixels Maximum Height

size is 500 pixels

AEMScrybeDevice:

This class is used to instantiate a BLUETOOTH SCRYBE Device, containing the Context, the Bluetooth Adapter, Bluetooth Device and Bluetooth Socket. An object of AEMScrybeDevice once instantiated, is capable of returning the AEMPrinter object that has been described previously. This class has the following functions:

AEMScrybeDevice

1. Constructor: For Creating the Object: public AEMScrybeDevice (IAemScrybeimpl)

2. startDiscover: public void startDiscover(Context iContext)

By calling this method, a list of local Bluetooth devices will be returned.

3. pairDevice: Public String pairDevice(String printerName)

Before pairing any printer by name, first call the method startDiscover(Context iContext) and get the result in the method public void onDiscoveryComplete(ArrayList<String>aemPrinterList) of the Interface IAemScrybe.

Example:

```
AEMScrybeDevice m_AemScrybeDevice = new AEMScrybeDevice (newIAemScrybe()
{
@Override
Public void onDiscoveryComplete(ArrayList<String>aemPrinterList)
{
// TODO Auto-generated method stub
}
});
```

Now call the method public String pairDevice(String printerName) which returns a String which may be:

NOT_SCANNED when you call this method before scanning. DEVICE_NOT_FOUND when the printer is not found. PAIRED when the device is successfully paired

FAILED_TO_PAIRED when the device is failed to paired

4. connectToPrinter: public Boolean connectToPrinter(String printerName) throws IOException

By calling this method, printer gets connected.

5. disConnectPrinter: public Boolean disConnectPrinter() throws IOException

By calling this method, the connected printer gets disconnected.

6. getCardReader: public CardReader getCardReader(IAemCardScannerreaderimpl)

By calling this method, you can create the object of the class CardReader by passing the reference of the class which implements IAemCardScanner Interface.

7. AEMPrinter: public AEMPrinter getAemPrinter()

By calling this method, you can create the object of the class AEMPrinter.

8. ArrayList: public ArrayList<String> getPairedPrinters()

By calling this method, a list of paired printers is returned.

9. getSDKVersion: public String getSDKVersion()

By calling this method, SDK version is obtained.

10. BtConnStatus: public Boolean BtConnStatus()

This method returns true if the Printer is already connected on Bluetooth (Bluetooth connection is already alive). It returns false if the Connection status is False, i.e. if Bluetooth socket is disconnected

IAemCardScanner:
1. onScanMSR: public void onScanMSR(String buffer, CARD_TRACK cardtrack)

This method will provide the MSR card data (buffer) with track type (Track1 or Track 2). You can decode the MSR card data by calling the method:-

publicCardReader.MSRCardDatadecodeCreditCard (String buffer, CARD_TRACK track) of the Class CardReader.

2. onScanDLCard: public void onScanDLCard(String Buffer)

This method will provide the DL card data. You can decode the DL card data (buffer) by calling the method:-

publicCardReader.DLCardDatadecodeDLCardData(String buffer) of the Class CardReader.

3. onScanRCCard: public void onScanRCCard(String Buffer)

This method will provide the RC card data. You can decode the RC card data (buffer) by calling the method:-

publicCardReader.RCCardDatadecodeRCCardData(String buffer) of the Class CardReader.

4. onScanRFD: public void onScanRFD(String Buffer) This

method is for RFID data.

5. onScanPacket: public void onScanPacket(String Buffer)

This method is for getting various responses:-
Responses from Printer:
Packet Structure:

Start	PB*	Separator	Packet	Separator	Response	End delimiter
delimiter			Type			(0x5E)
(0x7E)						
1 Byte	1 Byte	1 Byte	1 Byte	1 Byte	-	1 Byte
			Packet Structure			

***PB: Printer to Android (Host)**

Response table:

Response Name	Type		Packets
No Paper	Error		~PB|PRN|NOPAPER^
Printer Head High Temp:	Error		~PB|PRN|HGHTEMP^
Printer Platen	Error		~PB|PRN|PLTNREL^
Wrong Packet:	Error		~PB|PRN|PKTEROR^
MSR Swiped	Notification		~PB|CRD|MSR_SWP^
Low Battery	Error		~PB|BAT|LOWBATT^
Battery Charging	Notification		~PB|BAT|BATCHG1^
Battery Not Charging	Notification		~PB|BAT|BATCHG0^
Battery Charge Status	Notification		~PB|BAT|BAT%3d%%^
Printer Configuration	Response		~PB|CON|
			Name, Password, FontType, EnableEncryption,
			MSRTimeOut, ICTimeOut^
Printer Power Off	Response		~PB|PRN|PWR_OFF^
		Response Table	

Bit Map Image Print – Note that this method of Printing is only maintained for the purpose of Backward compatibility. The Faster, easier, clearer and more standard way of printing is by using Print Image function and passing it a Bitmap object.

1. **Select the image either from PC or from mobile's SD card.**
2. **Convert the image into monochrome bmp.**
3. **Resize the image to fit into the printer paper area if it is exceeding**

4 . Now read the processed image through file input stream and save it into byte array.

5. Make the packet of image and then send it to printer over output stream.

Bit-Image Packet Structure:

ESC(0x1B)	(0x2A)	Mode m	Alignment Of Image (a)	Width of Image (w)*	Height of Image (h)*	Image Byte Array (D1 to Dn)

Bit-Map Image Structure

Mode M (In Hex)	Mode M Description	Vertical Dot	Horizontal Dot	Max Dot/Line
0x64	Single width single height	1	1	384 (48 bytes)
0x65	Single width double height	2	1	384
0x66	Single width triple height	3	1	384
0x67	Single width Quad. height	4	1	384
0x68	Double width single height	1	2	192 (24 bytes)
0x69	Double width double height	2	2	192
0x6A	Double width triple height	3	2	192
0x6B	Double width Quad. Height	4	2	192
0x6C	Triple width single height	1	3	128 (16 bytes)
0x6D	Triple width double height	2	3	128
0x6E	Triple width triple height	3	3	128
0x6F	Triple width Quad. height	4	3	128

Mode Table

Alignment (a)	Hex Value
Left align	0x6C
Centre align	0x63
Right align	0x72

QR Code Generation Function:

```
public void onPrintQRCode(View v) throws Writer Exception, IO Exception
  {
      Writer writer = new QRCodeWriter(); String text=
      editText.getText().toString(); String finalData =
      Uri.encode(text, "UTF-8"); showAlert("QR " + text);
 try
 {
    BitMatrix bm = writer.encode(finalData,BarcodeFormat.QR_CODE, 300, 300); Bitmap bitmap =
    Bitmap.createBitmap(300, 300, Config.ARGB_8888); for(int i = 0; i < 300; i++) {

        for(int j = 0; j < 300; j++)
        {
                        bitmap.setPixel(i, j, bm.get(i, j) ? Color.BLACK: Color.WHITE);
        }
    }

    Bitmap resizedBitmap = null; int numChars
    = glbPrinterWidth;

            resizedBitmap = Bitmap.createScaledBitmap(bitmap, 384, 384, false);
                        m_AemPrinter.printImage(resizedBitmap);
 }
  catch(WriterException e)
  {
        showAlert("Error WrQR: " + e.toString());
  }
 }
```