

## Android TSPL SDK Manual

Android TSPL SDK Manual .....	1
1. Initialize .....	2
1.1 initialize .....	2
1.2 PrinterTSPL introduce .....	2
2.1 Bluetooth connection .....	3
2.2 USB connection .....	4
2.3 WIFI connection .....	5
2.4 Disconnect .....	6
3. SDK Command Function .....	7
3.1 Page label starting command: .....	7
3.2 Page label ending command (except for line mode): .....	8
3.3 Print printer information: .....	9
3.4 Set the distance of label paper : .....	10
3.5 Paper feed: .....	11
3.6 Clear the data in the buffer area .....	12
3.7 Text print, including some variant: .....	13
3.8 Print direction .....	15
3.9 Define the reference starting point of label .....	16
3.10 Define international character set code .....	17
3.11 Barcode: .....	18
3.12 Print QR Code .....	19
3.13 Specified length of label paper feed .....	20
3.14 Position the label to the starting point of next label .....	21
3.15 Print line .....	22
3.16 Position the label to the starting point specified by the internal sensor ....	23
3.17 Print image: .....	24
3.18 Print density .....	25
3.19 Print speed .....	26
3.20 Cut paper .....	27
3.21 Print rectangle box .....	28
3.22 Set beep time of buzzer .....	29
3.23 Send data .....	30
3.24 Read data .....	31
3.25 Anti-white box .....	32
3.26 Text Bold .....	33
3.27 Get Printer Status .....	34
3.28 Print Block .....	35
3.29 Print PDF417 .....	36
3.30 Set Black Position .....	38
3.31 Get Printer SN .....	39

# 1. Initialize

## 1.1 initialize

```
public class App extends Application {  
    @Override  
    public void onCreate() {  
        super.onCreate();  
        PrinterHelper.init(this);  
    }  
}
```

## 1.2 PrinterTSPL introduce

This class contains all interfaces related to the printer. You can get the object through the connection interface. If you need multiple connections, save multiple objects, each representing a connection.

You can get the serial number of the printer through the getPrinterSN interface in the object to identify the printer. You can also get the SDK internal identifier through the getPrinterTAG() interface. If it is connected via Bluetooth, the identifier is the Bluetooth address, if it is connected via WiFi, it is the IP address, and if it is connected via USB, it is the device node.

## 2. Communication Port Function

### 2.1 Bluetooth connection

```
public static PrinterTSPL connectBT(String mac)
```

**Parameter:**

mac: "Bluetooth address (uppercase)"

**Return:**

printer == null, the connection fails.

printer != null, the connection succeeds.

**Example:**

```
PrinterTSPL printer = PrinterHelper.connectBT("00:01:02:03:04:05");
```

## 2.2 USB connection

```
public static PrinterTSPL connectUSB(UsbDevice usbdevice)
```

**Parameter:**

usbdevice: The USB device object.

**Return:**

printer == null, the connection fails.

printer != null, the connection succeeds.

**Example:**

```
PrinterTSPL printer = PrinterHelper.connectUSB(device);
```

## 2.3 WIFI connection

```
public static PrinterTSPL connectWifi(String address)
```

**Parameter:**

address: IP address

**Return:**

printer == null, the connection fails.

printer != null, the connection succeeds.

**Example:**

```
PrinterTSPL printer = PrinterHelper.connectWifi("192.168.1.1");
```

## 2.4 Disconnect

```
public boolean portClose()
```

**Parameter:**

null

**Return:**

true: close succeeds, false close fail.

**Example:**

```
PrinterTSPL printer = PrinterHelper.connectWifi( "192.168.1.1" );  
printer.portClose()
```

### 3. SDK Command Function

#### 3.1 Page label starting command:

`int printAreaSize(String width, String height)`

Parameter:

width:width within the print area (unit: Millimeter).

height:height within the print area (unit: Millimeter).

Return:

Greater than 0: normal, otherwise abnormal.

Example:

```
printer.printAreaSize("100", "100")
```

```
printer.printText("0","0","9","0","1","1","TEXT")//Print TEXT
```

```
printer.print("1","1")
```

### 3.2 Page label ending command (except for line mode):

```
int print(String strNum, String strCopies)
```

Parameter:

strNum: times of printing

strCopies: counter (default 1).

Return:

Greater than 0: normal, otherwise abnormal.

Example:

```
printer.printAreaSize("100", "100")
```

```
printer.printText("0","0","9","0","1","1","TEXT");//Print TEXT
```

```
printer.print("1","1")
```



### **3.3 Print printer information:**

```
int selfTest()//Print self test page
```

Return:

Greater than 0: normal, otherwise abnormal.

Example:

```
printer.selfTest()
```

### 3.4 Set the distance of label paper :

`int gap(String distance, String offset)`

Parameter:

Distance: distance between two labels (unit:mm).

Offset: distance between label content and label bottom (unit:mm)

Return:

Greater than 0: normal, otherwise abnormal.

Example:

`printer.gap("5","5")`

### 3.5 Paper feed:

int offset (String distance)

Parameter:

distance: paper feeding distance (unit:mm).

Return:

Greater than 0: normal, otherwise abnormal.

Example:

```
printer.offset("5")
```

### **3.6 Clear the data in the buffer area**

`int cls()`

Return:

Greater than 0: normal, otherwise abnormal.

Example:

`printer.cls()`

### 3.7 Text print, including some variant:

(1) `Int printText(String xPos, String yPos, String font, String rotation, String x_multiplication, String y_multiplication, int alignment, String code_data)`

Parameter:

xPos: starting x-coordinate of text

yPos: starting y-coordinate of text

font:

0: Monotype CG Triumvirate Bold Condensed, font width and height is stretchable

1: 8 x 12 fixed pitch dot font

2: 12 x 20 fixed pitch dot font

3: 16 x 24 fixed pitch dot font

4: 24 x 32 fixed pitch dot font

5: 32 x 48 dot fixed pitch font

6: 14 x 19 dot fixed pitch font OCR-B

7: 21 x 27 dot fixed pitch font OCR-B

8: 14 x 25 dot fixed pitch font OCR-A

9: Only this font can print Chinese.

Rotation: print direction.

0 : No rotation

90 : degrees, in clockwise direction

180 : degrees, in clockwise direction

270 : degrees, in clockwise direction

x\_multiplication: text stretch multiplication in x-coordinate direction.

y\_multiplication: text stretch multiplication in y-coordinate direction.

Alignment: alignment mode (some old models do not support this function, this parameter can be removed)

1: Left aligned

2: Centered

3: Right aligned

code\_data: text data.

Return:

Greater than 0: normal, otherwise abnormal.

Example:

```
printer.printAreaSize("100","200")
```

```
printer.cls();
```

```
printer.printText("0","0","1","0","0","0","TEXT")
```

```
printer.print("1","1")
```

(2) int printText(String x\_pos,String y\_pos,String font,String rotation,int size,  
int alignment, String code\_data)

Parameter:

x\_pos: starting x-coordinate of text

y\_pos: starting x-coordinate of text

font:

Same as the previous interface

Rotation: print direction

0 : No rotation

90 : degrees, in clockwise direction

180 : degrees, in clockwise direction

270 : degrees, in clockwise direction

Size: font size 1~7.

Alignment: alignment mode (some old models do not support this  
function, this parameter can be removed)

1: Left aligned

2: Centered

3: Right aligned

code\_data: text data

Return:

Greater than 0: normal, otherwise abnormal.

Example:

```
printer.printAreaSize("100","200")
```

```
printer.cls();
```

```
printer.printText("0","0","0","0","0","TEXT")
```

```
printer.print("1","1")
```

### 3.8 Print direction

int direction(String direction)

Parameter:

direction:

0: vertical

1: horizontal

Return:

Greater than 0: normal, otherwise abnormal.

Example:

```
printer.printAreaSize("100","200")
```

```
printer.direction("0");
```

```
printer.printText("0","0","0","0","0","TEXT")
```

```
printer.print("1","1")
```

### 3.9 Define the reference starting point of label

```
int reference(String x_pos,String y_pos)
```

Parameter:

x: x-coordinate of starting point

y: y-coordinate of starting point

Return:

Greater than 0: normal, otherwise abnormal.

Example:

```
printer.reference("0","0")
```



### 3.10 Define international character set code

`int codepage(String codepage)`

Parameter:

codepage: code number

USA: USA, BRI: British, GER: German, FRE: French

DAN: Danish, ITA: Italian, SPA: Spanish, SWE: Swedish

SWI: Swiss, 437: United States, 850: Multilingual, 852: Slavic

860: Portuguese, 863: Canadian/French, 865: Nordic,

857: Turkish (TSPL2 printers only)

1250: Central Europe (TSPL2 printers only)

1252: Latin I (TSPL2 printers only), 1253: Greek (TSPL2 printers only)

1254: Turkish (TSPL2 printers only)

Return:

Greater than 0: normal, otherwise abnormal.

Example:

`printer.codepage("437")`

### 3.11 Barcode:

```
int printBarcode(String x_pos, String y_pos, String code_type,String
height,String readable,String rotation,String narrow,String wide,String
code_data)
```

#### Parameter:

x\_pos: starting x-coordinate of bar code

y\_pos: starting y-coordinate of bar code

code\_type: bar code type

128,128M,EAN128 ,39 ,93,UPCA ,MSI ,ITF14 ,EAN13

Height: height of bar code(Unit: pixel)

Readable: whether the bar code data is readable

0: not readable

1: human readable

rotation: bar code direction

0 : No rotation

90 : degrees, in clockwise direction

180 : degrees, in clockwise direction

270 : degrees, in clockwise direction

Narrow: unit width of narrow bar code(Default: 1).

Wide: unit width of wide bar code(Default: 1).

code\_data: bar code data

#### Return:

Greater than 0: normal, otherwise abnormal.

#### Example:

```
printer.printAreaSize("100","200")
printer.cls();
printer.printBarcode("0","0","128","100","1","0","1","1","1234567890
")
printer.print("1","1")
```

### 3.12 Print QR Code

```
printQRCode(String x_pos,String y_pos,String ecc_level,String  
width,String mode,String rotation,String code_data)
```

Parameter:

rotation: bar code direction

0 : No rotation

90 : degrees, in clockwise direction

180 : degrees, in clockwise direction

270 : degrees, in clockwise direction

X: starting x-coordinate of QR code

Y: starting y-coordinate of QR code

ecc\_level: correction level

L : 7%

M : 15%

Q : 25%

H : 30%

width: 1~10

mode :Auto/manual encode

A: Auto

M: Manual

Data: QR code data

Return:

Greater than 0: normal, otherwise abnormal.

Example:

```
printer.printAreaSize("100","200")  
printer.cls();  
printer.printQRCode("0","0","M","6","A","0","1234567890")  
printer.print("1","1")
```

### 3.13 Specified length of label paper feed

`int feed(String len)`

Parameter:

len: length of paper feed (unit: mm)

Return:

Greater than 0: normal, otherwise abnormal.

Example:

`printer.feed("5")`

### **3.14 Position the label to the starting point of next label**

`int formFeed()`

Return:

Greater than 0: normal, otherwise abnormal.

Example:

`printer.formFeed()`

### 3.15 Print line

```
int bar(String x_pos,String y_pos,String width, String height)
```

Parameter:

x\_pos: starting x-coordinate

y\_pos: starting y-coordinate

width: width of line (pixel)

height: height of line (pixel)

Return:

Greater than 0: normal, otherwise abnormal.

Example:

```
printer.printAreaSize("100","200")
```

```
printer.bar("10","10","100","2")//horizontal line
```

```
printer.bar("10","10","2","100")//vertical line
```

```
printer.print("1","1")
```

### **3.16 Position the label to the starting point specified by the internal sensor**

`int home()`

Return:

Greater than 0: normal, otherwise abnormal.

Example:

`printer.home()`

### 3.17 Print image:

`int printImage(String xPos, String yPos, Bitmap bmp ,boolean isNegate, boolean isLZO, int type)`

**Parameter:**

xPos: starting x-coordinate of image

yPos: starting y-coordinate of image

bitmap: the object of image Bitmap

isNegate: image reverse

True: normal display

False: negate display

isLZO:Whether compression (the printer must support compression)

Type:Picture algorithm

0:black white algorithm

1:halftone algorithm

**Return:**

Greater than 0: normal,Equal to -1: width or height of image is over the printer area

**Example:**

`printer.printAreaSize("100","200")//The height of label should be greater than that of image`

`printer.cls()`

`printer.printImage("10","10",bitmap,true,false,1)`

`printer.print("1","1")`



### 3.18 Print density

`int density(String contrast )`

Parameter:

Contrast: 0~15

0: specifies the lightest level

15: specifies the darkest level

Return:

Greater than 0: normal, otherwise abnormal.

Example:

`printer.density("5")`

### 3.19 Print speed

int speed(String speed )

Parameter:

speed

1	1.	2	2.	3	3.	4	5	6	8	10	12
	5		5		5						

Return:

Greater than 0:normal, otherwise abnormal.

Example:

printer.speed("2")

### 3.20 Cut paper

`int cut()`

Return:

Greater than 0: normal, otherwise abnormal.

Example:

`printer.cut()`

### 3.21 Print rectangle box

```
int box(String x_start,String y_start,String x_end, String y_end,String  
thickness)
```

Parameter:

x\_start: x-coordinate of top left corner  
y\_start: y-coordinate of top left corner  
x\_end: x-coordinate of lower right corner  
y\_end: y-coordinate of lower right corner  
Thickness: width of the line

Return:

Greater than 0: normal, otherwise abnormal.

Example:

```
printer.printAreaSize("100","200")  
printer.cls()  
printer.box("10","10","100","100","2")  
printer.print("1","1")
```

### 3.22 Set beep time of buzzer

`int sound(String level, String interval)`

Parameter:

level: the lasting time of beep, (1/8) second is specified unit

Interval: the interval time

Return:

Greater than 0: normal, otherwise abnormal.

Example:

`printer.sound("8","8")`

### 3.23 Send data

```
int writeData(byte[] bData)
```

Parameter:

bData: data that needs to be sent to the printer

Return:

Greater than 0: normal, otherwise abnormal.

Example:

```
printer.writeData(bData)//Send data to the printer
```

### 3.24 Read data

Byte[] readData(int timeout)

Parameter:

timeout: time out(unit:millisecond)

Return:

Data:the data read

Example:

bData=printer.readData(2000)//bData is the data read

### 3.25 Anti-white box

```
int reverse(String x_start,String y_start,String x_width,String y_height)
```

Parameter:

x\_start: X coordinate.

y\_start: Y coordinate.

x\_width: width.

y\_height: height.

Note:

*200 DPI: 1 mm = 8 dots*

*300 DPI: 1 mm = 12 dots*

Return:

Greater than 0: normal, otherwise abnormal.

Example:

```
printer.printAreaSize("100","200")
```

```
printer.cls()
```

```
printer.printText("100","100","9","0","32","32","REVERSE" )
```

```
printer.reverse("90","90","128","40")
```

```
printer.print("1","1")
```





### 3.26 Text Bold

`int bold(int bold)`

Parameter:

bold: 0: Not bold.

1: bold.

Return:

Greater than 0: normal, otherwise abnormal.

Example:

`printer.bold(1)`

### 3.27 Get Printer Status

int getPrinterStatus()

Parameter:

NULL

return:

printer.STATUS_DISCONNECT:	Disconnection
printer.STATUS_TIMEOUT:	Time Out
printer.STATUS_OK:	Ready
printer.STATUS_COVER_OPENED:	Cover Opened
printer.STATUS_NOPAPER:	Not Paper
printer.STATUS_OVER_HEATING:	Over Heating
printer.STATUS_PRINTING:	Printing

Example:

printer.getPrinterStatus()

### 3.28 Print Block

```
int printBlock(int startX,int startY,int width,int height,int font,int rotation,  
int multiplicationX,int multiplicationY,int space,int alignment,String content)
```

#### Parameter:

StartX: x coordinate of upper left corner of text.

StartY: Y coordinate at upper left corner of the text.

Width: The Width of the text.

Height: The Height of the text.

FONT: Font size (0:16 \*16, 1:24 \*24)

Rotate to: (0,90,180,270)

MultiplicationX: Multiplication X in the X-axis direction of the font.

MultiplicationY: Multiplication Y in the Y-axis direction of the font.

Space: Line spacing.

Alignment: Alignment. 1: Left Align, 2: Center, 3: Right Align

Content: Print the Content.

#### Return:

Greater than 0: normal, otherwise abnormal.

#### Example:

```
printer.printAreaSize("100","200")
```

```
printer.cls()
```

```
printer.printBlock(0,0,100,100,0,0,2,2,16,2,"Test,Test,Test,Test")
```

```
printer.print("1","1")
```

### 3.29 Print PDF417

```
int printPDF417(int x_pos, int y_pos, int width, int height, int rotate, ArrayList<String>  
    option, String expression)
```

Parameter:

x\_pos: x coordinate.

y\_pos: y coordinate.

width: Barcode width.

height: Barcode height.

rotate: Rotate . (0, 90, 180, 270)

option: Barcode parameter. (enable null)

P Data compression method

0: Auto encoding

1: Binary mode

E Error correction level. Range: 0~8

M Center pattern in barcode area

0: The pattern will print upper left justified the area

1: The pattern is printed middle of area

Ux, y, c Human readable.

x: Human readable characters in the specified x-coordinate

y: Human readable characters in the specified y-coordinate

c: Maximum characters of human readable character per line

W Module width in dot. Range: 2~9

H Bar height in dot. Range: 4~99

R Maximum number of rows

C Maximum number of columns

T Truncation.

0: Not truncated

1: Truncated

Lm Expression length (without double quote), 1<m<2048

expression: data.

Return:

Greater than 0: normal, otherwise abnormal.

Example:

```
printer.printAreaSize("100", "100");  
printer.cls();  
ArrayList<String> option = new ArrayList<>();  
option.add("P1");  
option.add("E4");  
option.add("M1");  
option.add("U100,500,10");
```

```

option.add("W6");
option.add("H6");
option.add("R60");
option.add("C4");
option.add("T1");
option.add("L297");
String data = "Data" +
    "compression method: P1" +
    "Error correction level: E4" +
    "Center pattern in barcode area: M1" +
    "Human Readable: Yes: U100,500,10" +
    "Module Width 6 dots: W6" +
    "Bar Height 6 dots: H6" +
    "Maximum Number of Rows: 60 Rows: R60" +
    "Maximum number of columns: 4 Cols: C4" +
    "Truncation:1: T1" +
    "Expression length:297: L297";
printer.printPDF417(50, 50, 900, 600, 0, option, data);
printer.print("1", "1");

```

### 3.30 Set Black Position

`boolean setBlackPosition(int position)`

Parameter:

position:

- 1: 2-inch front-right black mark.
- 2: 3-inch front-right black mark.
- 3: 2-inch rear-right black mark.
- 4: 3-inch rear-right black mark.

Return:

true: Sent successfully, false: Sending failed.

Example:

```
printer.setBlackPosition(1);
```

### 3.31 Get Printer SN

String getPrinterSN()

Parameter:

    null

Return:

    The serial number of the printer, (empty string means failure)

Example:

    printer.getPrinterSN();