POS SDK For Android
用户手册

目录

关于本用户手册

目的

本手册的目的是为客户介绍 POS SDK For Android 的例程和 API 的使用。

内容

本手册主要由以下几部分组成：

1. 综述

本章主要是关于 POS SDK For Android 的 WIFI 搜索连接、蓝牙搜索连接、USB 和串口连接，API 函数功能、操作应用环境、发布包中文件及版本的说明。

1.1 功能

- **WIFI**

WIFI 搜索功能、打印机连接功能、数据读写功能及读写超时设置、将发送到打印机的数据保存到文件功能、关闭 WIFI 连接功能。

- **蓝牙**

蓝牙搜索功能、打印机连接功能、数据读写功能及读写超时设置、将发送到打印机的数据保存到文件功能、关闭蓝牙连接功能。

- USB

打印机连接功能、数据读写功能及读写超时设置、将发送到打印机的数据保存到文件功能、关闭连接功能。

- **串口**

打印机连接功能、数据读写功能及读写超时设置、将发送到打印机的数据保存到文件功能、关闭连接功能。

- **API**

1) 系统设置功能(打印机初始化、选择打印模式、选择纸张类型、设置横纵向移动单位、查询打印机状态、走纸、切纸、下载文件)钱箱打开功能。

2) 打印字符（设置字符集和代码页、设置文本行高、设置字符间距、文本对齐方式、字体选择、反显、粗体、下划线、旋转、放大、双色打印、英文用户自定义字符打印、光栅化字符打印）。

3) 图像打印功能（8 点/24 点单双精度位图打印、RAM 位图下载及打印、Flash 位图下载及打印、光栅化位图打印）。

4) 一维条码打印功能（UPC-A、UPC-E、EAN-8、EAN-13、Code39、Code93、ITF、Codabar、Code128）。

5) 二维条码打印功能（PDF417、QR、Maxicode、GS1 DataBar 和 GS1 复合码）。

6) 标准模式设置（打印区域宽度、左边距、打印横向起始的位置）。

7) 页模式设置（打印区域设置、打印方向、打印横纵向起始位置、页模式打印及清空。

1.2 操作环境

- **Android 版本**

  WIFI、蓝牙和串口：Android 2.1 及以上版本

  USB：Android 3.1 及以上版本

- **Android 设备**

  Android 系统手机

  Android 系统平板

  Android 系统开发板

- **打印机**

  POS 系列打印机

- **支持端口**

  WIFI、蓝牙、USB 和串口

- **应用环境**

  JDK：版本 1.6 及以上

  eclipse：版本 3.5 及以上

  Android SDK ：版本 1.6 及以上

  ADT：20.0.3 及以上

1.3 发布包中的文件

文件夹	文件
二次开发包	POSSDKForAndroid.jar libserial_port.so、 android.hardware.usb.host.xml、
例程工程及源码	POSSDKForAndroid.apk、 POSSDKForAndroidDemo
帮助文档	POS SDK For Android Chinese User’s Manual.pdf 、

	POS SDK For Android English User's Manual.pdf

1.4 版本记录

版本	日期	版本记录
V1.00	6/3/2016	首版本

2. 示例程序

本章主要描述如何使用示例程序。

例程有如下功能：

- WIFI 搜索打印机
- 蓝牙搜索打印机
- 输入 IP 地址
- 输入串口端口号及波特率
- 打开端口
- 关闭端口
- 设置打印模式
- 文本打印
- RAM/Flash 位图下载并打印
- 一维条码打印
- PDF417 条码打印
- QR 码打印
- GS1 Databar 条码打印
- 光栅化文本打印
- 用户自定义字符打印
- 打印机状态查询
- 走纸
- 切纸

2.1 例程启用

1)从压缩包中直接解压出例程 POSSDKForAndroidDemo。

2)运行 eclipse，选择“File->Import”，如图 1 所示。

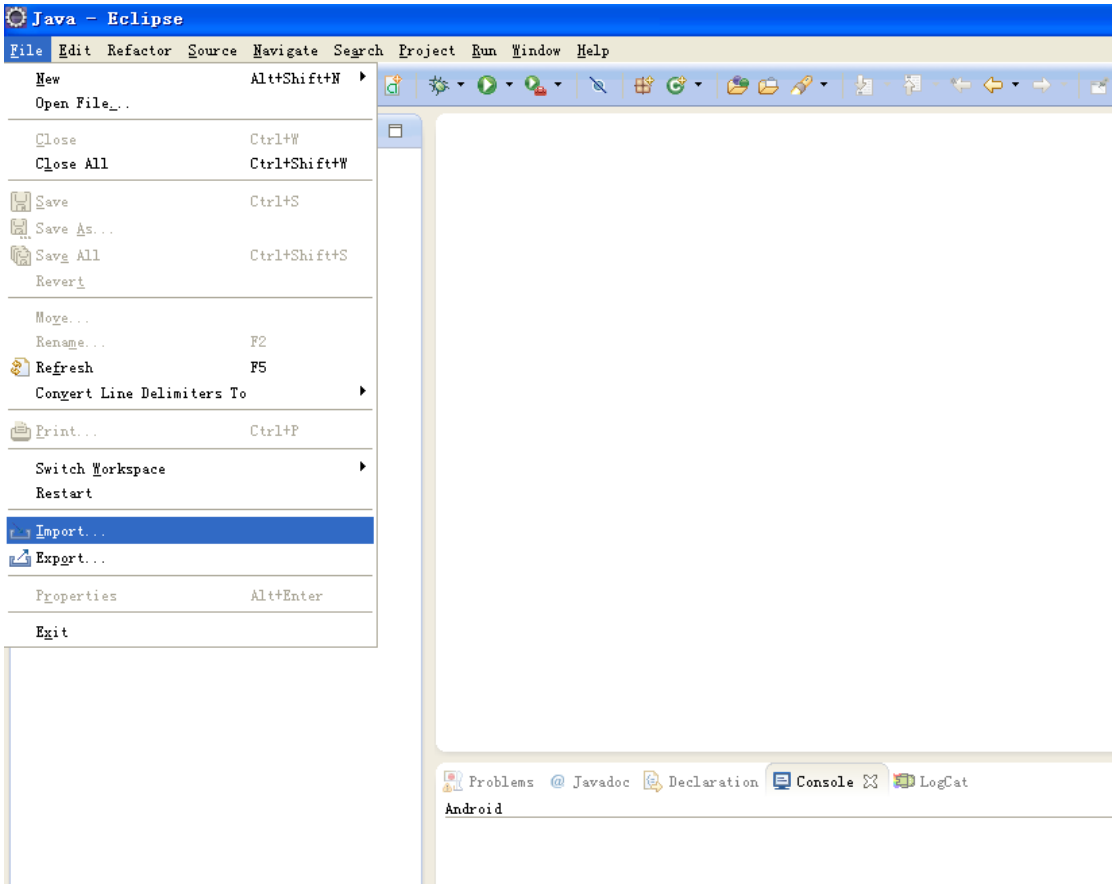
Java - Eclipse
File Edit Refactor Source Navigate Search Project Run Window Help
New Alt+Shift+N
Open File...
Close Ctrl+W
Close All Ctrl+Shift+W
Save Ctrl+S
Save As...
Save All Ctrl+Shift+S
Revert
Move...
Rename... F2
Refresh F5
Convert Line Delimiters To
Print... Ctrl+P
Switch Workspace
Restart
Import...
Export...
Properties Alt+Enter
Exit
Problems Javadoc Declaration Console LogCat
Android

图 1 工程导入选项

3)选择 General 中的 Existing Projects into Workspace 选项，如图 2 所示。

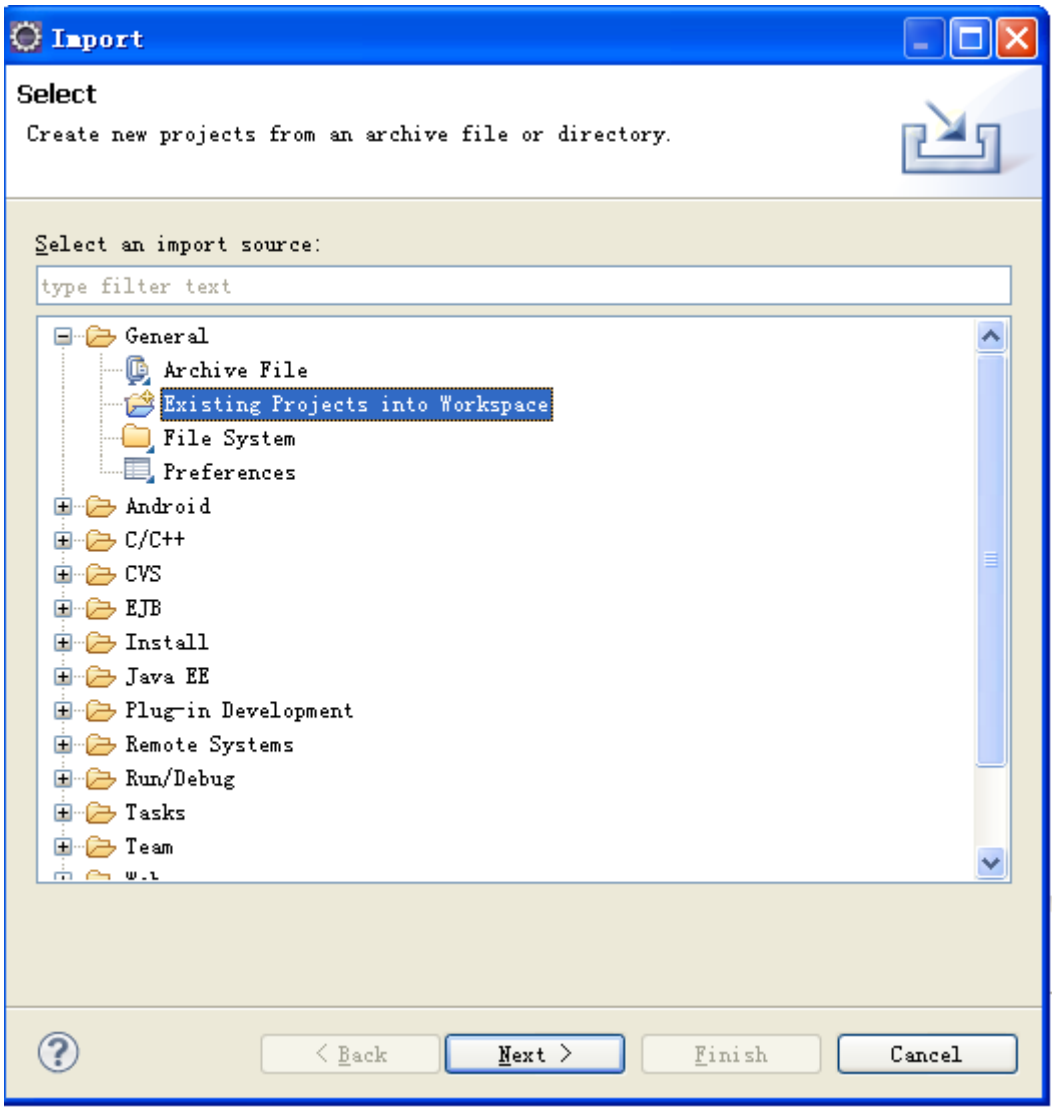

图 2 加载工程

4)选择解压的例程 POSSDKForAndroidDemo，如图 3 和图 4 所示。

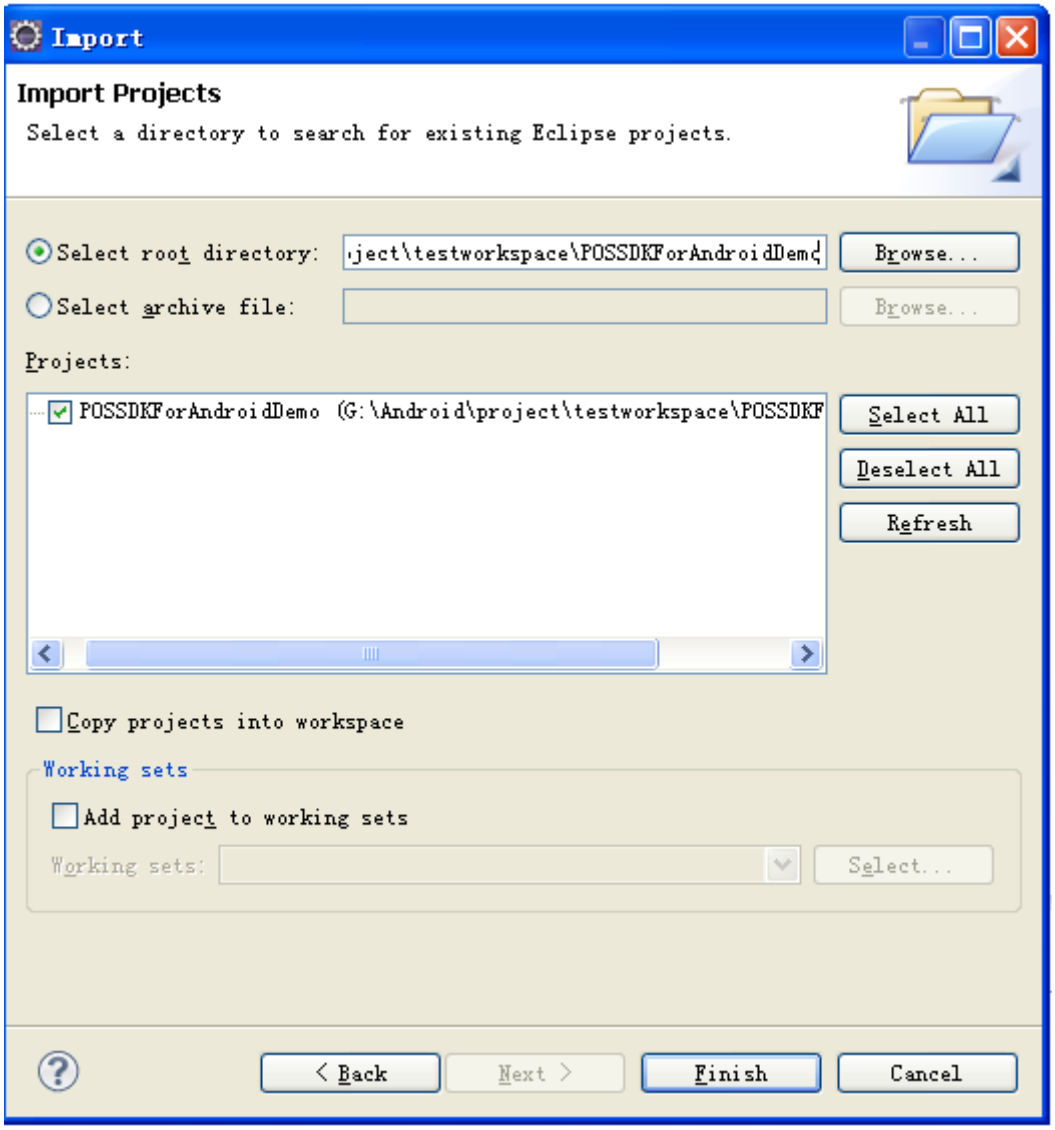

图 3 导入例程

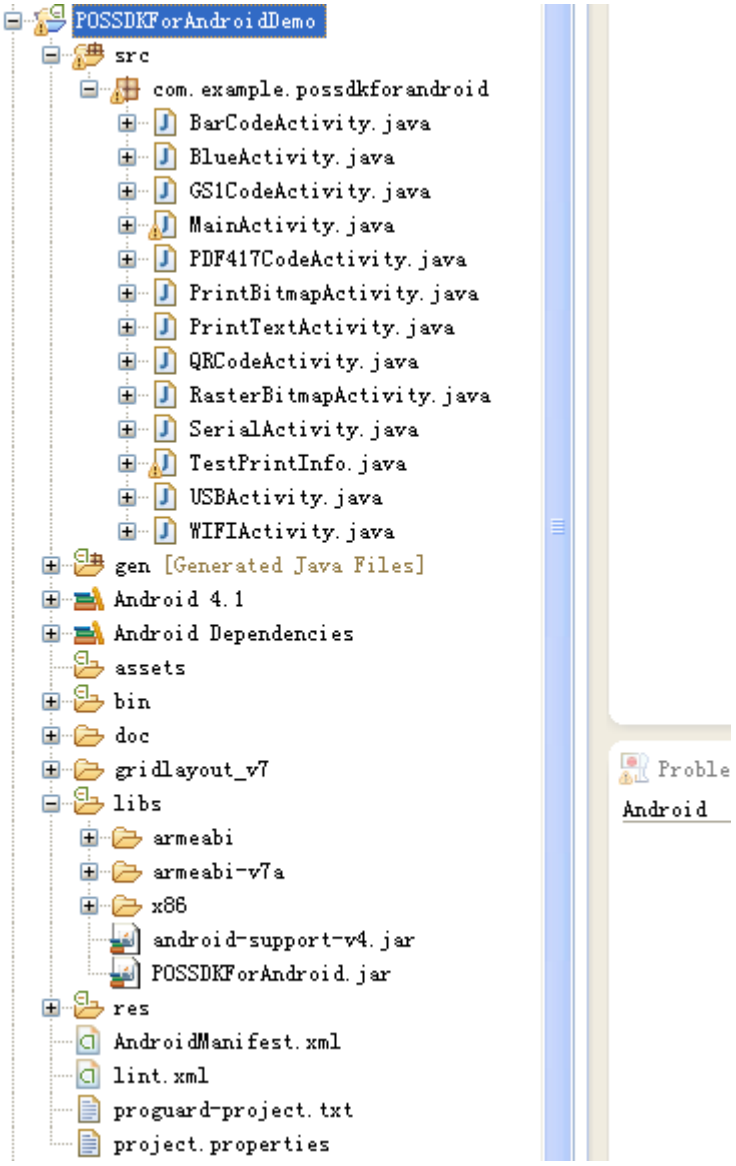

图 4 导入例程后运行界面

5)将发布包中的“android.hardware.usb.host.xml”拷贝到 Android 系统终端设备的“/system/etc/permissions”文件夹下，将终端设备赋予 USB HOST 权限，实现终端设备通过 USB 接口控制打印机操作。

6)将发布包中的安装文件“POSSDKForAndroid.apk”拷贝到 Android 系统终端设备中，直接安装也可以生成可执行的例程。

注：通过 adb 命令实现 Windows 系统终端与 Android 系统终端的文件传递，例如：文件导入命令格式：adb push 文件所在 Windows 系统路径及文件名称 文件需要存放在 Android 系统路径；

文件导出命令格式：adb pull 文件所在 Android 系统路径及文件名称 文件导出存放在 Windows 系统路径。

2.2 如何使用例程

● **主界面操作**

主界面的功能图，如下图 5 所示：

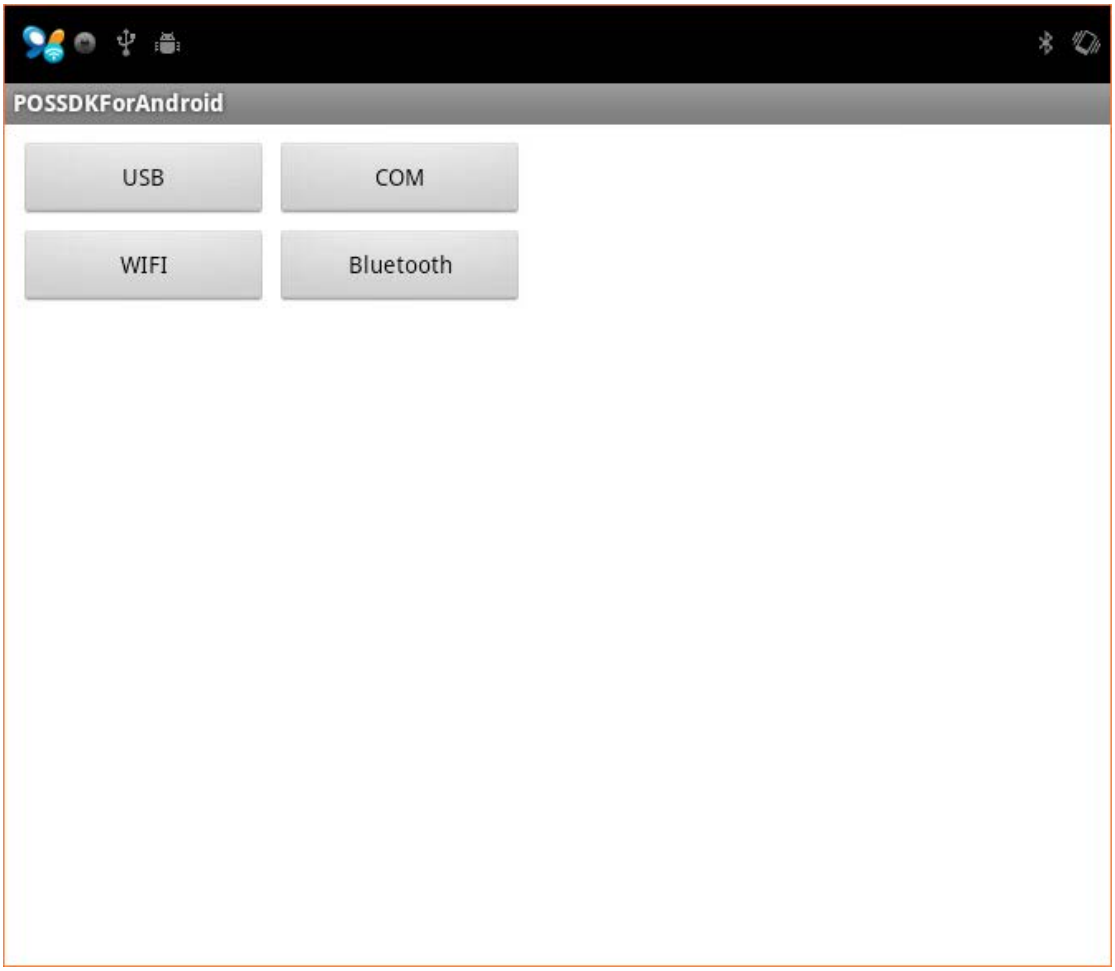

图 5 主界面

主界面的功能说明：

控件	描述
“USB”按钮	单击进入 USB 端口通讯操作界面。
“COM”按钮	单击进入串口通讯操作界面。
“WIFI”按钮	单击进入 WIFI 接口通讯操作界面。
“Bluetooth”按钮	单击进入蓝牙接口通讯操作界面。

● **USB 通讯**

USB 端口通讯操作界面如下图 6 所示。

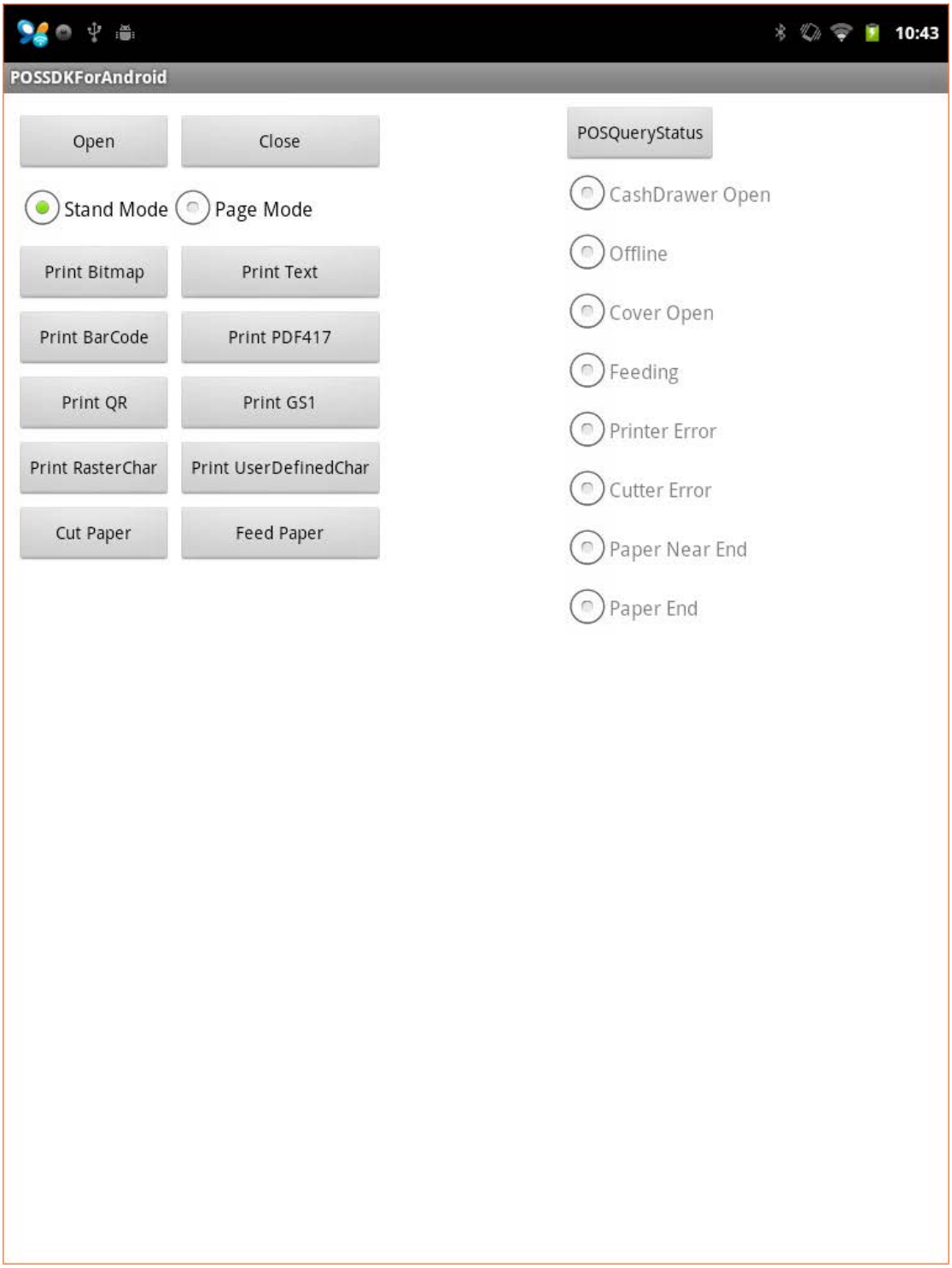

图 6 USB 端口通讯操作界面

界面的功能说明如下：

控件	描述
“Open”按钮	单击连接打印机设备。
“Close”按钮	单击断开打印机设备。
“Stand Mode”和“Page	选择打印模式，Stand Mode--表示标准打印模式，

Mode”单选框	Page Mode--表示页打印模式。
“Print Bitmap”按钮	单击进入位图打印，具体细节请参见位图打印。
“Print Text”按钮	单击进入文本打印，具体细节请参见文本打印。
“Print BarCode”按钮	单击进入一维条码打印，具体细节请参见一维条码打印。
“Print PDF417”按钮	单击进入 PDF417 码打印，具体细节请参见 PDF417 条码打印。
“Print QR”按钮	单击进入 QR 码打印，具体细节请参见 QR 码打印。
“Print GS1”按钮	单击进入 GS1 条码打印，具体细节请参见 GS1 条码打印。
“Print RasterChar”按钮	单击进入光栅字符打印，具体细节请参见光栅字符打印。
“Print UserDefinedChar”按钮	单击进行用户自定义字符打印。
“Cut Paper”按钮	单击进行切纸操作。
“Feed Paper”按钮	单击进行走纸操作。
“POSQueryStatus”按钮	单击进行打印机设备状态查询。

- **串口通讯**

串口通讯操作界面如下图 7 所示。

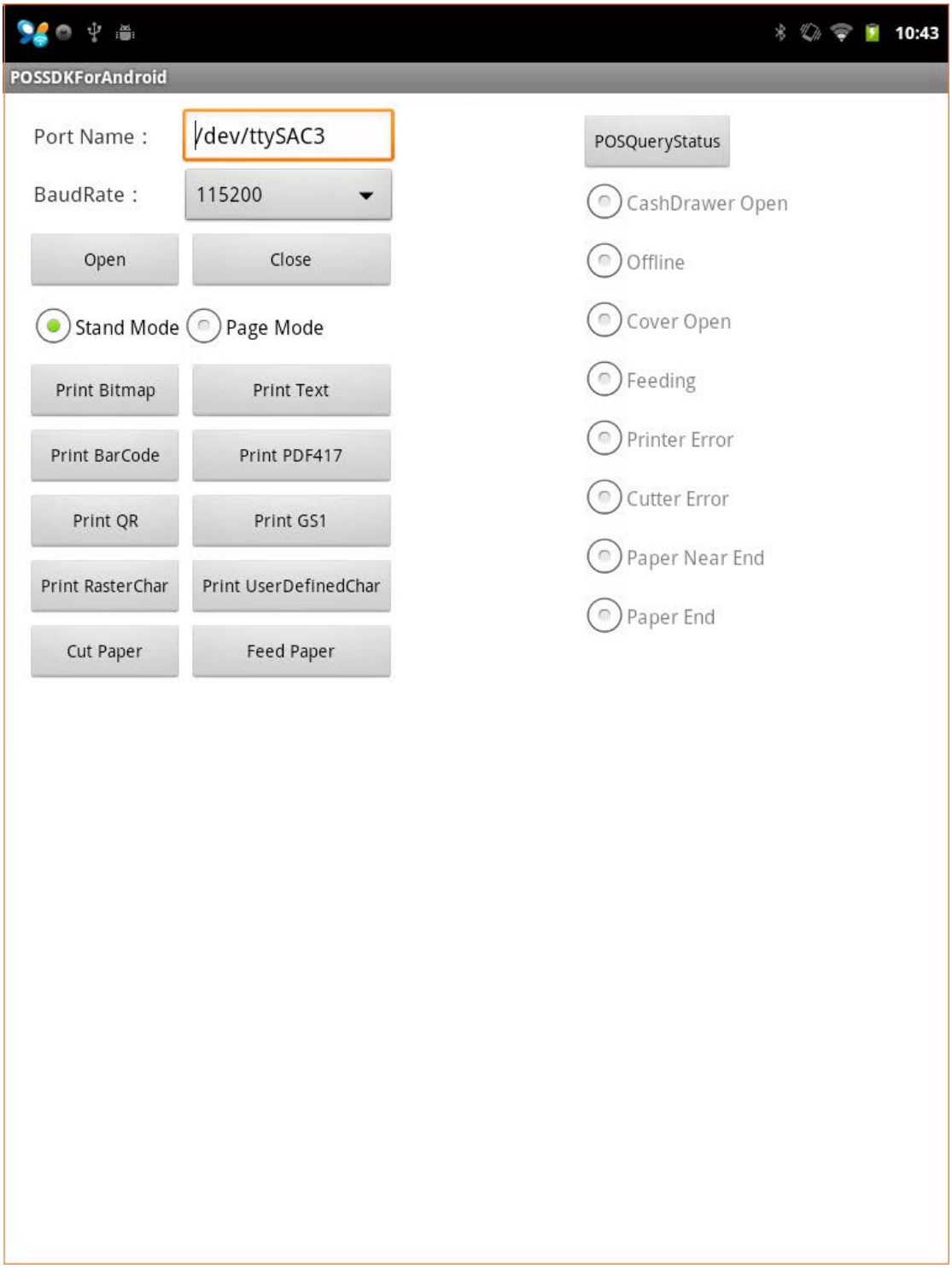

图 7 串口通讯操作界面

以 USB 界面功能为例，不同之处如下：

控件	描述
“Port Name”编辑框	输入设备串口端口号。
“BaudRate”下拉列表	选择通讯的波特率。

其他控件操作与上述 USB 端口操作界面控件功能相同。

- **WIFI 通讯**

WIFI 通讯操作界面如下图 8 所示。

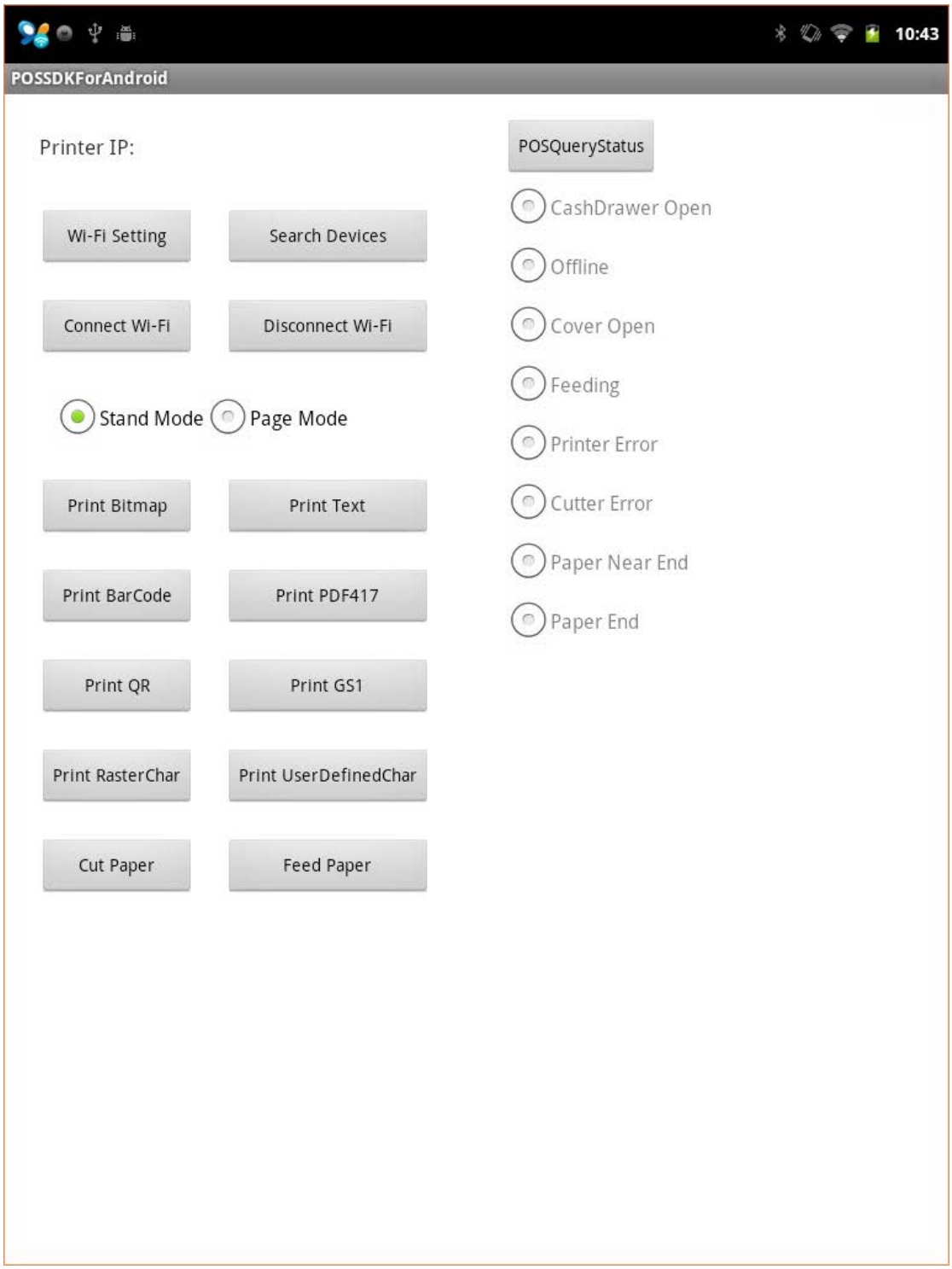

图 8 WIFI 通讯操作界面

以 USB 界面功能为例，不同之处如下：

控件	描述
“Printer IP”编辑框	输入要连接打印机设备的 IP 地址。
“WI-FI Serring”按钮	系统 WIFI 管理界面设置，连接打印机所在的基站地址。
“Search Devices”按钮	搜索打印机设备。
“Connect WI-FI”按钮	连接指定的打印机设备。
“Disconnect WI-FI”按钮	断开打印机设备连接。

其他控件操作与上述 USB 端口操作界面控件功能相同。

- **蓝牙通讯**

蓝牙通讯操作界面如下图 9 所示。

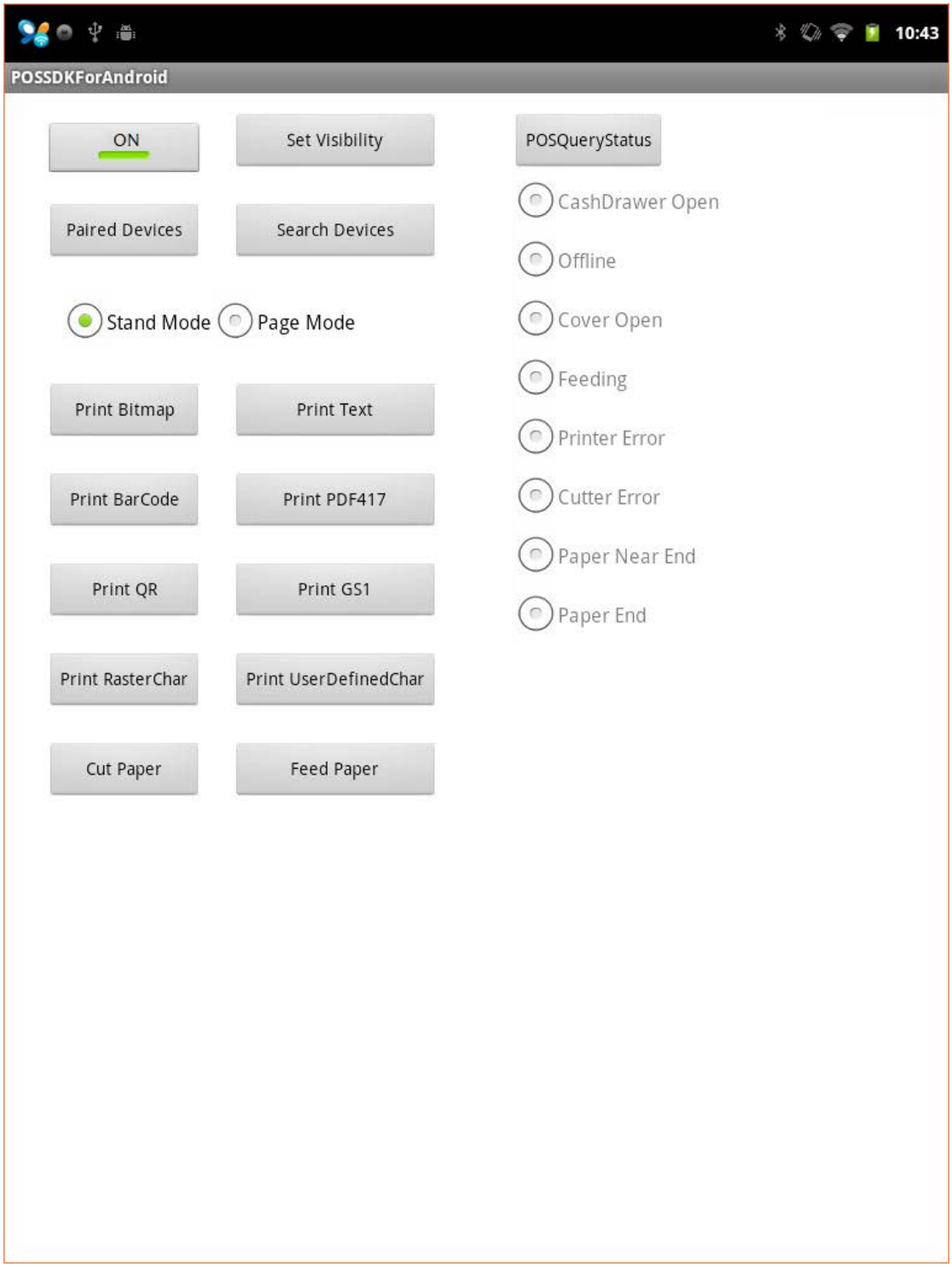

图 9 蓝牙通讯操作界面

以 USB 界面功能为例，不同之处如下：

控件	描述
“蓝牙开关”双状态按钮	进行 Android 终端设备的蓝牙开启和关闭操作。
“Set Visibility”按钮	设置 Android 终端设备的蓝牙为可见设备。
“Paired Devices”按钮	查询已经与 Android 终端设备配对的蓝牙打印机设

	备。
“Search Devices”按钮	搜索 Android 终端设备附近的蓝牙设备。

其他控件操作与上述 USB 端口操作界面控件功能相同。

● **位图打印**

单击通讯界面上的“Print Bitmap”按钮，进入位图打印的界面，如下图 10 所示。

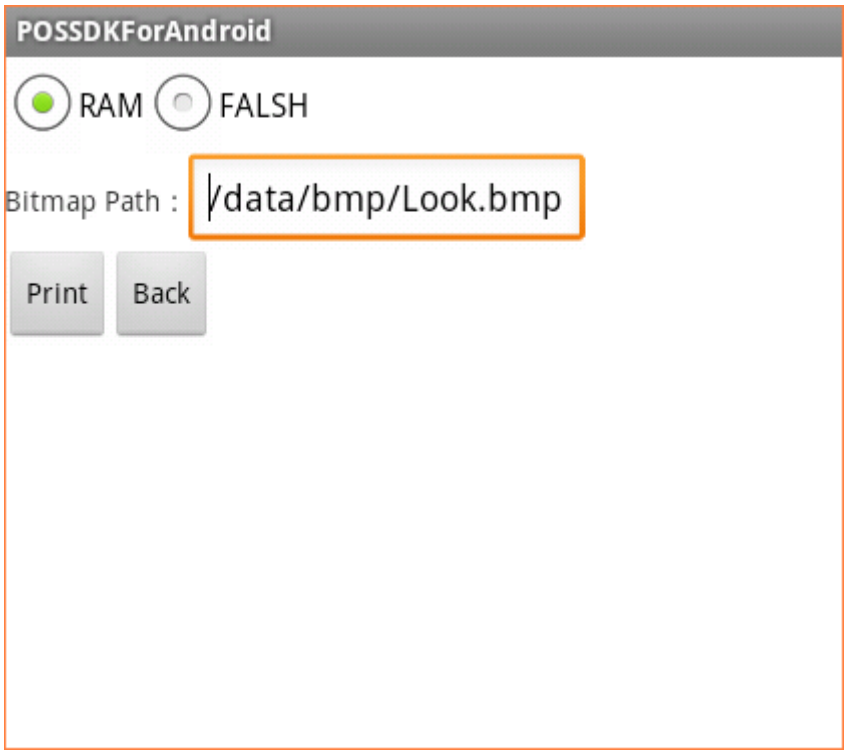

图 10 位图打印界面

界面控件功能如下：

控件	描述
“RAM”和“FALSH”单选框	选择下载位图的位置。
“Bitmap path”编辑框	输入要打印位图的路径及名称。 注：选择 RAM 下载打印模式时，位图只能一个文件，选择 FALSH 下载打印模式时，可以输入多个位图名称，中间以"@"作为分割标志，如：输入图像为"/data/bmp/1.bmp"+"@"+"/data/bmp/2.bmp"。
“Print”按钮	进行打印操作。
“Back”按钮	返回上一层操作界面。

● 文本打印

单击通讯界面上的“Print Text”按钮，进入文本打印界面，如下图 11 所示。

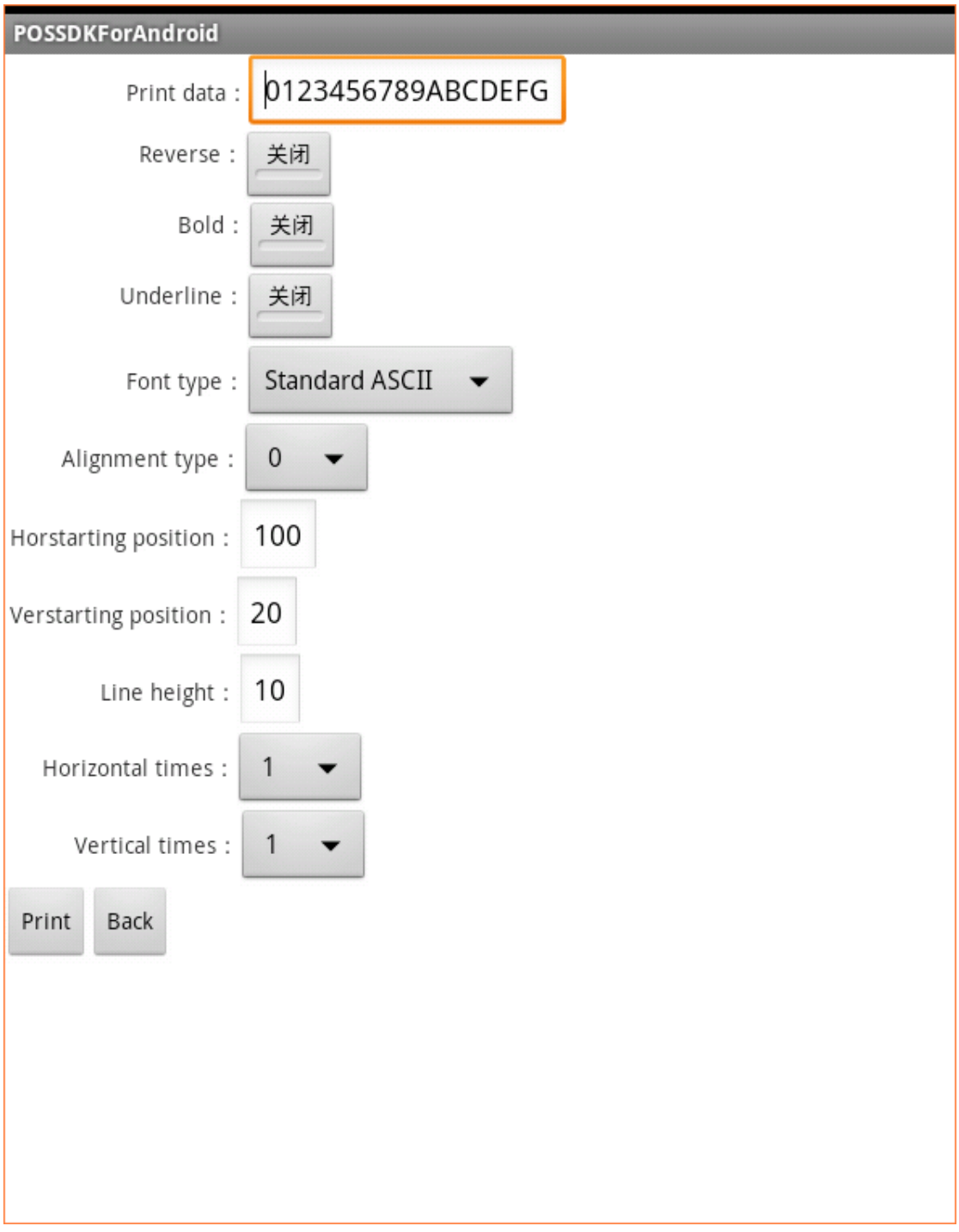

图 11 文本打印界面

界面控件功能如下：

控件	描述
“Print data”编辑框	输入要打印的文本内容。

“Reverse”双状态按钮	设置反显打印。
“Bold”双状态按钮	设置加粗打印。
“Underline”双状态按钮	设置打印下划线。
“Font type”下拉列表	设置打印字体类型。
“Alignment type”下拉列表	设置打印内容对齐方式。
“Horstarting position”编辑框	设置打印的初始横向坐标。
“Verstarting position”编辑框	设置打印的初始纵向坐标。
“Line height”编辑框	设置打印内容行高。
“Horizontal times”下拉列表	设置打印内容横向放大倍数。
“Vertical times”下拉列表	设置打印内容纵向放大倍数。
“Print”按钮	进行打印操作。
“Back”按钮	返回上一层操作界面。

● **一维条码打印**

单击通讯界面上的“Print BarCode”按钮，进入一维条码打印界面，如下图12 所示。

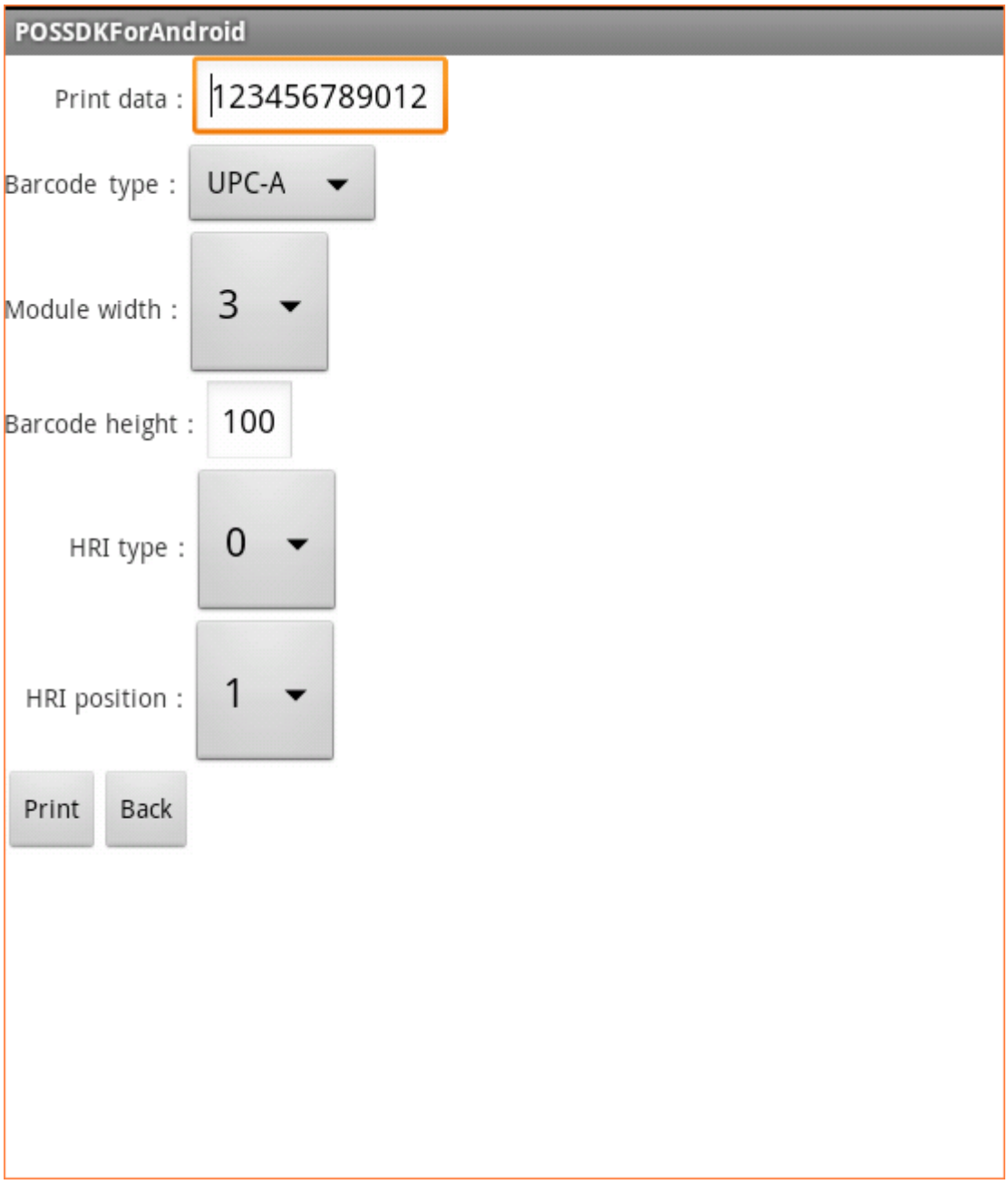

图 12 一维条码打印界面

界面控件功能如下：

控件	描述
“Print data”编辑框	输入要打印的条码内容。
“BarCode type”下拉列表	选择一维条码类型。
“Module width”下拉列表	设置打印条码模块宽度。
“Barcode height”编辑框	输入条码高度。
“HRI type”下拉列表	设置 HRI 字体类型。
“HRI position”下拉列表	设置 HRI 字体位置。

“Print”按钮	进行打印操作。
“Back”按钮	返回上一层操作界面。

- **PDF417 条码打印**

单击通讯界面上的“Print PDF417”按钮，进入 PDF417 条码打印界面，如下图 13 所示。

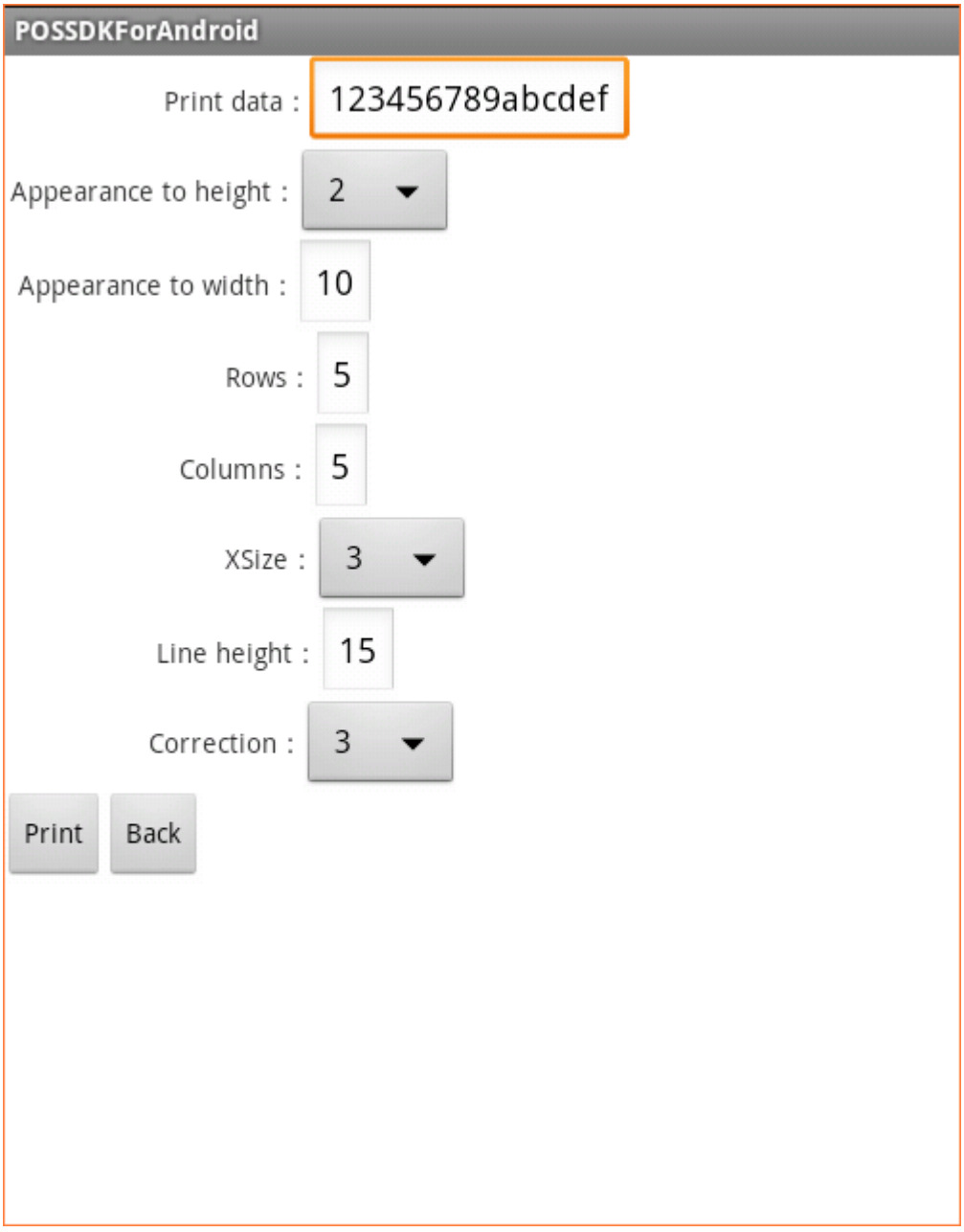

图 13 PDF417 条码打印界面

界面控件功能如下：

控件	描述
“Print data”编辑框	输入要打印的条码内容。

“Appearance to height”下拉列表	设置外观比高度比例因子。
“Appearance to width”下拉列表	设置外观比宽度比例因子。
“Rows”编辑框	设置条码行数。
“Columns”编辑框	设置条码列数。
“XSize”下拉列表	设置条码 X 尺寸。
“Line height”编辑框	设置条码行高。
“Correction”下拉列表	设置条码纠错等级。
“Print”按钮	进行打印操作。
“Back”按钮	返回上一层操作界面。

- **QR 码打印**

单击通讯界面上的“Print QR”按钮，进入 QR 条码打印界面，如下图 14 所示。

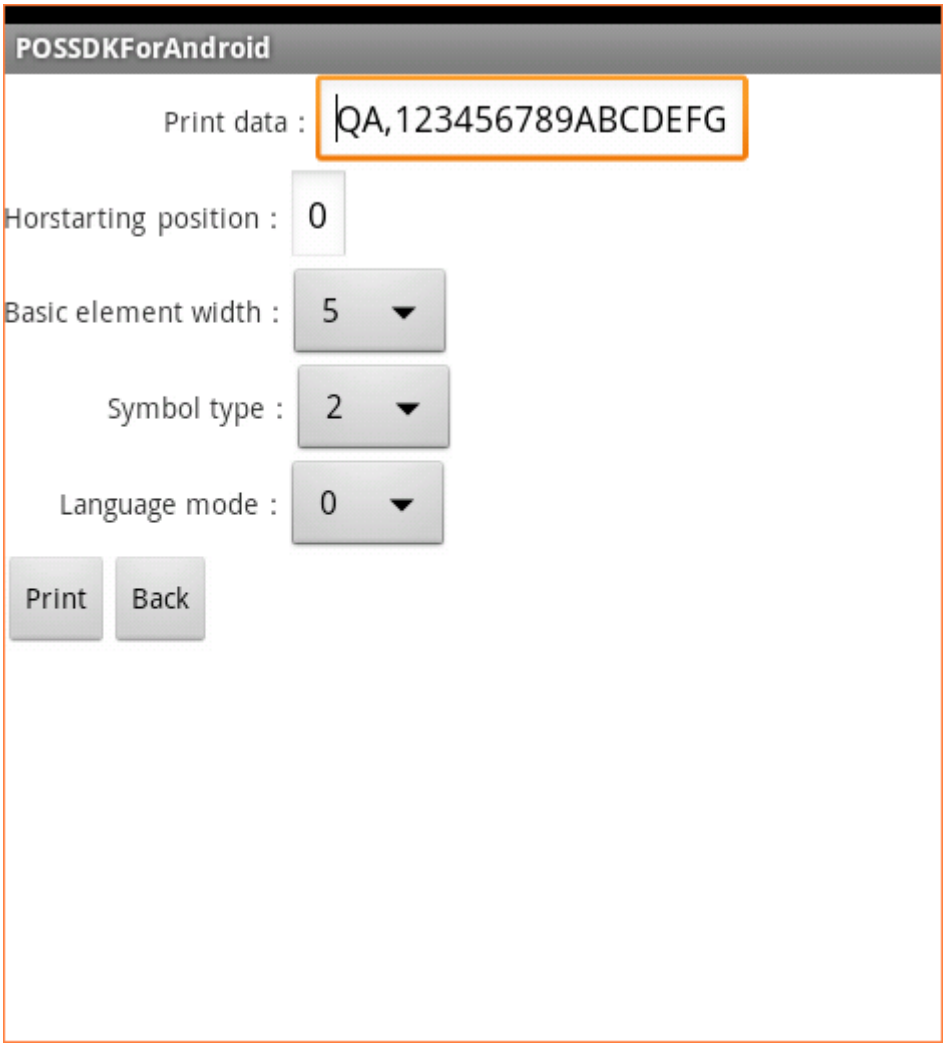

图 14 QR 码打印界面

界面控件功能如下：

控件	描述
“Print data”编辑框	输入要打印的条码内容。
“Horstarting position”编辑框	设置条码横向打印位置。
“Basic element width”下拉列表	设置打印条码模块宽度。
“Symbol type”下拉列表	设置打印符号类型。
“Language mode”下拉列表	设置语言模式。
“Print”按钮	进行打印操作。
“Back”按钮	返回上一层操作界面。

- **GS1 条码打印**

单击通讯界面上的“Print GS1”按钮，进入 GS1 条码打印界面，如下图 15 所示。

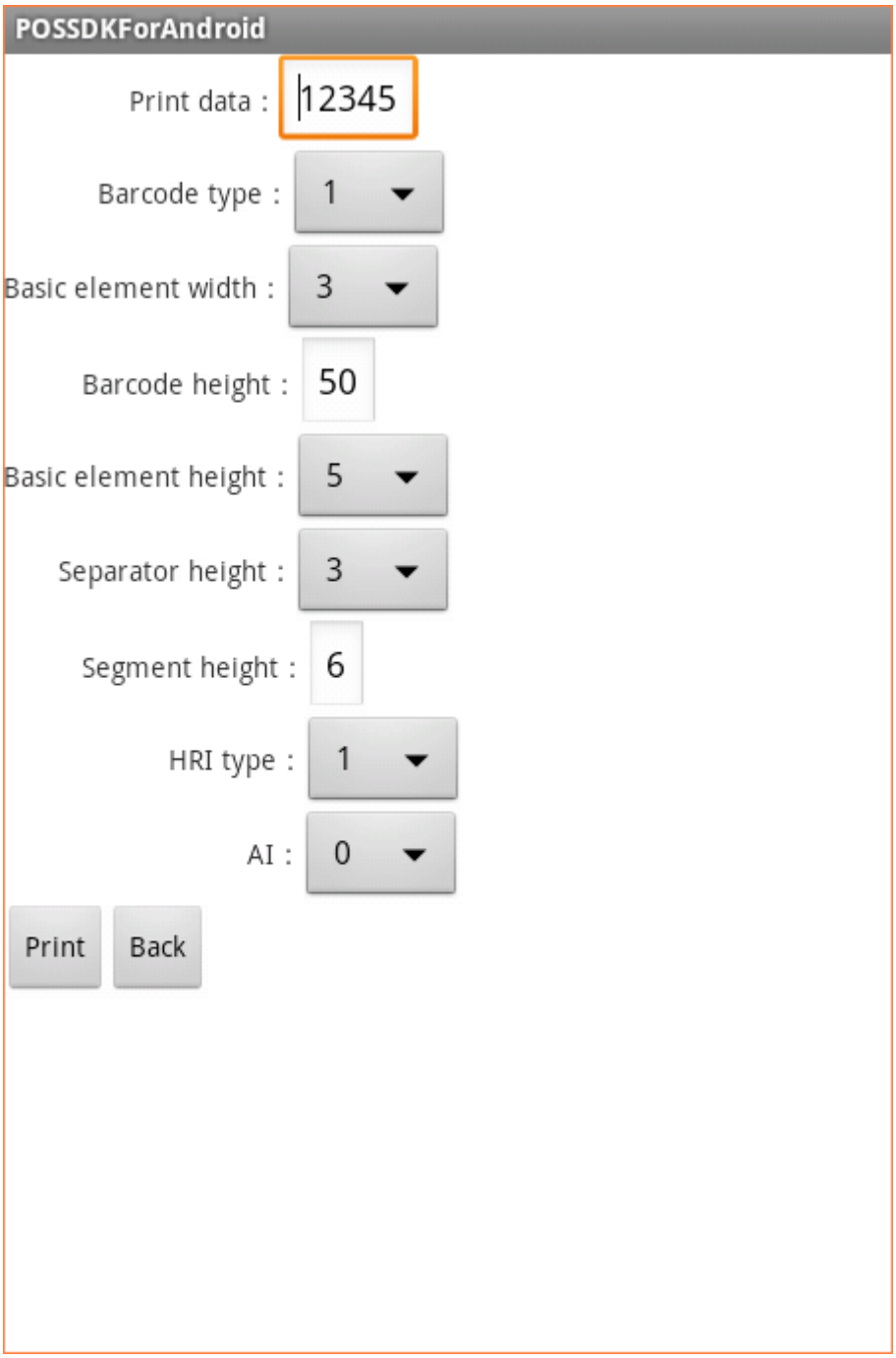

图 15 GS1 条码打印界面

界面控件功能如下：

控件	描述
“Print data”编辑框	输入要打印的条码内容。
“Barcode type”下拉列表	设置 GS1 条码类型。
“Basic element width”下拉列表	设置打印条码基本元素宽度。
“Barcode height”编辑框	设置 DataBar 的高度。

“Basic element height”下拉列表	设置复合码中 2D 条码符号基本元素高度。
“Separator height”下拉列表	设置分隔符的高度。
“Segment height”编辑框	设置每行条码符号的段数。
“HRI type”下拉列表	设置条码注释内容。
“AI”下拉列表	设置是否应用 AI 标识符。
“Print”按钮	进行打印操作。
“Back”按钮	返回上一层操作界面。

● **光栅字符打印**

单击通讯界面上的“Print RasterChar”按钮，进入光栅化字符打印界面，如下图 16 所示。

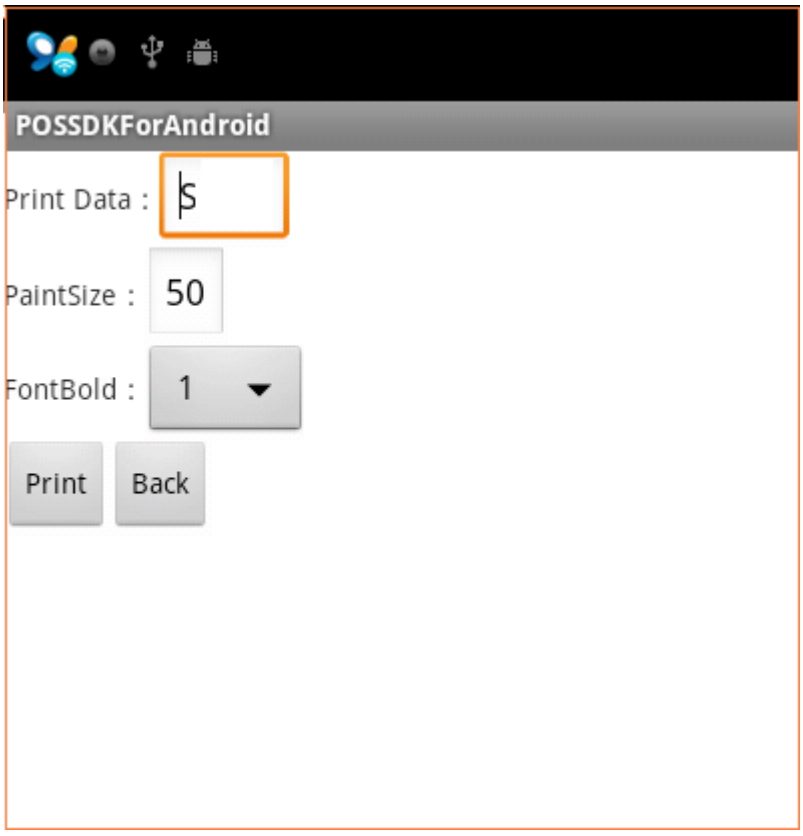

图 16 光栅化字符打印界面

界面控件功能如下：

控件	描述
“Print Data”编辑框	输入要打印的条码内容。
“PaintSize”编辑框	设置打印内容画布大小。
“FontBold”下拉列表	设置打印内容字体的粗细程度。

“Print”按钮	进行打印操作。
“Back”按钮	返回上一层操作界面。

3. 编程指南

本章介绍了在应用程序开发时如何使用 POS SDK For Android 编写程序

3.1 连接打印机

● **USB 连接打印机**

将发布包中的“android.hardware.usb.host.xml”拷贝到 Android 设备的“/system/etc/permissions”文件夹下，确保终端设备具有 USB HOST 权限，实现终端设备通过 USB 端口控制打印机操作。

● **WIFI 连接打印机**

确保 Android 设备和打印机均有正确的 WIFI 参数（SSID、encryption、password 等）设置，并且正常连接网络。使用点对点模式时，Android 设备通过 WIFI 正常连接打印机设备，输入正确的 IP 地址建立连接。如果 Android 设备连接打印机时间超过 3S，请使用 Android 设备静态地址。设置与打印机同网段的 IP 地址和网关地址，并将域名 1 和域名 2 设置为 0.0.0.0。

● **蓝牙连接打印机**

确保 Android 设备与打印机的蓝牙端口已经配对，保证 Android 设备的蓝牙正常开启。

● **串口连接打印机**

确保 Android 设备要进行串口通讯的端口未被系统控制台占用，输入正确的端口号与波特率，保证 Android 设备与打印机设备的波特率相同，另外保证打印机设备 EEPROM 中的串口参数为默认设置，其中握手协议为硬握手，数据位为 8，校验位为 0，停止位为 1。

3.2 使用二次开发包

这部分介绍如何使用提供的 Android 系统下的 POS 类打印机 JAR 包。

● **如何导入 JAR 包**

JAR 包的导入很简单，将 JAR 包放入项目下的 libs 文件夹内即可（若没有该目录，则自己新建一个），选中该项目单击右键，选择“Refresh”就可以了，就看到该 JAR 包被导入在 libs 目录下（如果 Refresh 之后没有看到的 JAR 包，

可以重新导入项目）。

● **如何导入串口 JNI 动态库**

对于 USB、WIFI 和蓝牙接口通讯项目，只需要使用提供 JAR 包即可；而串口通讯项目除了 JAR 包，还需要提供的 libserialport.so 动态库（JNI 串口连接动态库）。

动态库的导入和 jar 包是一样的，将.so 文件放入 libs 目录即可。

● **如何使用 JAR 包中的接口类 API 函数**

如使用 JAR 包的接口类 API 函数，需要导入接口类，并创建对象，其中接口父类 POSInterfaceAPI，四种接口子类为 POSUSBAPI（USB）、POSWIFIAPI（WIFI）、POSBluetoothAPI（蓝牙）、POSSerialAPI（串口），例如：

```
import POSAPI.POSInterfaceAPI;
import POSAPI.POSWIFIAPI;
import POSAPI.POSUSBAPI;
import POSAPI.POSBluetoothAPI;
import POSAPI.POSSerialAPI;

//创建USB对象
POSInterfaceAPI  interface_usb = new POSUSBAPI(this);
//创建WIFI对象
POSInterfaceAPI interface_wifi = new POSWIFIAPI();
//创建蓝牙对象
POSInterfaceAPI  interface_blue=
POSBluetoothAPI.getInstance(Activity.this);
//创建串口对象
POSInterfaceAPI  interface_com = new POSSerialAPI();
```

创建对象后，就可以调用对应接口类中的 API 函数，详细请参加例程 POSSDKForAndroidDemo。

● **如何使用 JAR 包中的 POSSDK 类 API 函数**

如使用 JAR 包的 POSSDK 类 API 函数，需要导入 POSSDK 类，并创建对象，其中类名为 POSSDK，例如：

```
import POSSDK.POSSDK;
//创建USB口POSSDK类对象
POSSDK pos_sdk = new POSSDK(interface_usb);
```

● 方法举例：

WIFI 搜索打印机

```
private static final int SearchPortMAX = 10;
private SearchPortInfo port_info[] = new SearchPortInfo[SearchPortMAX];
private int sch_prt_num = 0;
for(i = 0; i < SearchPortMAX; i++){
    port_info[i] = new SearchPortInfo();
}
sch_prt_num = interface_wifi.WIFISearchPort(port_info, SearchPortMAX);
```

USB 连接打印机

```
error_code = interface_usb.OpenDevice();//正常连接打印机设备
error_code = interface_usb.OpenDevice(5455,5455);//连接指定的Vid和Pid设备
```

WIFI 连接打印机

```
private static final int POSPORT = 9100;
private static final int STATEPORT = 4000;
private static String POSIP = "192.168.1.210";
error_code = interface_wifi.OpenDevice(POSIP, POSPORT);
```

蓝牙连接打印机

```
String address = "00:1B:35:07:16:AC";
error_code = interface_blue.OpenDevice(address);
```

串口连接打印机

```
String port_name = "/dev/ttySAC3";
int baud_rate = 115200;
error_code = interface_com.OpenDevice(new File(port_name),baud_rate);
```

关闭连接

```
error_code = interface_wifi.CloseDevice();
interface_wifi = null;
```

设置标准模式参数

```
private static final int PRINT_MODE_STANDARD = 0;
private static final int PRINT_MODE_PAGE = 1;
error_code = pos_wifi.systemSelectPrintMode(PRINT_MODE_STANDARD);
```

页模式下设置参数举例

```
error_code = pos_wifi.systemSelectPrintMode(PRINT_MODE_PAGE);
error_code = pos_sdk.pageModeSetPrintArea(0,0,640,500,0);
error_code = pos_sdk.pageModeSetStartingPosition(20,200);
```

文本打印

```
String txtbuf = "123456789山东";
byte []send_buf = txtbuf.getBytes("GB18030");
```

```
error_code = pos_sdk.textPrint(send_buf, send_buf.length);
send_buf = null;
```

文本光栅化打印

```
String txt = " ";
c_image = pos_sdk.imageCreateRasterBitmap(txt,50,1);
error_code = pos_sdk.imageStandardModeRasterPrint(c_image,640);
```

用户自定义字符打印

```
String path =  "/data/bmp/u1.bmp" + "@" + "/data/bmp/u2.bmp" + "@" +
               "/data/bmp/u3.bmp";
FileInputStream temp_stream = null;
String sigPaths[] = path.split("@");
int image_num = sigPaths.length;
Bitmap cg_image[] = new Bitmap[image_num];
int i = 0;
for(i = 0; i < image_num; i++){
    try {
             temp_stream = new FileInputStream(sigPaths[i]);
       } catch (FileNotFoundException e) {
                         e.printStackTrace();
       }
   if(temp_stream == null){
                     return;
                  }
    cg_image[i] = BitmapFactory.decodeStream(temp_stream);
    temp_stream = null;
 }
 error_code = pos_sdk.textUserDefinedCharacterEnable(1);
 if(error_code != POS_SUCCESS){
                     return;
              }
error_code = pos_sdk.textUserDefinedCharacterDefine(3, 12, 48, 50,
cg_image);
if(error_code != POS_SUCCESS){
                     return;
              }
error_code = pos_sdk.textSelectFontMagnifyTimes(2,2);
error_code = pos_sdk.textUserDefinedCharacterCancel(48);
error_code = pos_sdk.textUserDefinedCharacterCancel(49);
error_code = pos_sdk.textUserDefinedCharacterCancel(50);
```

一维条码打印

```
String pszBuffer = "012345678912";
error_code =
pos_sdk.barcodePrint1Dimension(pszBuffer,pszBuffer.length(),BarcodeUP
```

```
C_A,4, 100,0,1);
```

PDF417 条码打印

```
String str = "123456789山东";
int data_size = 0;
try {
        data_size=str.getBytes("GB18030").length;
    } catch (UnsupportedEncodingException e) {
        e.printStackTrace();
    }
error_code = pos_sdk.barcodePrintPDF417(str,
data_size,2,10,5,5,3,10,3);
```

QR 码打印

```
String str = "123456789山东";
int data_size = 0;
try {
        data_size=str.getBytes("GB18030").length;
    } catch (UnsupportedEncodingException e) {
            e.printStackTrace();
    }
error_code = pos_sdk.barcodePrintQR(str,data_size, 0, 5, 1, 0);
```

Maxicode 条码打印

```
String str = "123456789山东";
int data_size = 0;
try {
        data_size=str.getBytes("GB18030").length;
    } catch (UnsupportedEncodingException e) {
            e.printStackTrace();
    }
error_code = pos_sdk.barcodePrintMaxicode (str,data_size);
```

GS1 DataBar 和 GS1 复合条码打印

```
String str = "123456789";
int data_size = 0;
try {
        data_size=str.getBytes("GB18030").length;
    } catch (UnsupportedEncodingException e) {
        e.printStackTrace();
    }
error_code = pos_sdk.barcodePrintGS1DataBar(str, data_size, 1,3,50, 5,
3, 6, 1, 0);
```

位图打印

```
String str = "/data/bmp/Look.bmp";
 try {
```

```
        temp_stream = new FileInputStream(str);
    } catch (FileNotFoundException e) {
                e.printStackTrace();
    }
    if(temp_stream == null){
          return;
        }
image = BitmapFactory.decodeStream(temp_stream);
error_code = pos_sdk.imageStandardModePrint (image, 33, 0, 640);
```

RAM/Flash 位图下载及打印

RAM

```
error_code = pos_sdk.imageDownloadToPrinterRAM(2, image, 640);
error_code = pos_sdk.imageRAMPrint(2,0);
```

Flash

```
String path =  "/data/bmp/Look.bmp" + "@" + "/data/bmp/s.bmp" + "@" +
               "/data/bmp/1.bmp";
String sigPaths[] = path.split("@");
int image_num = sigPaths.length;
Bitmap cg_image[] = new Bitmap[image_num];
int i = 0;
for(i = 0; i < image_num; i++){
    try {
            temp_stream = new FileInputStream(sigPaths[i]);
        } catch (FileNotFoundException e) {
            e.printStackTrace();
        }
    if(temp_stream == null) {
            return;
    }
    cg_image[i] = BitmapFactory.decodeStream(temp_stream);
    temp_stream = null;
}
error_code = pos_sdk.imageDownloadToPrinterFlash(image_num, cg_image,
640);
 if(error_code != POS_SUCCESS){
                  return;
              }
error_code = pos_sdk.imageFlashPrint(1, 0);
error_code = pos_sdk.imageFlashPrint(2, 0);
error_code = pos_sdk.imageFlashPrint(3, 0);
for(i = 0; i < image_num; i++){
       cg_image[i].recycle();
 }
```

光栅化位图打印

```
error_code = pos_sdk.imageStandardModeRasterPrint(image,640);
```

文件下载

```
String str = "/data/bmp/Printer.JK";
error_code = pos_sdk.systemDownloadFile(str,5000);
if(error_code == POS_SUCCESS){
    System.out.println("Download file success!");
    return error_code;
}else{
    System.out.println("Download file fail!");
}
```

打印机状态查询

```
final int QueryStatusSize=4;
byte StatusBuffer[] = new byte[QueryStatusSize];
error_code =
pos_sdk.systemQueryStatus(StatusBuffer,QueryStatusSize,1);
```

走纸

```
error_code = pos_sdk.systemFeedLine(3);
```

切纸

```
error_code = pos_sdk.systemCutPaper(65, 0);
```

4. API 相关

本章描述在 POS SDK for Android 中提供的 API 接口函数。

4.1 接口相关

API	描述
OpenDevice	建立与打印机的 USB 连接
OpenDevice：Vid, Pid	建立与指定的 Vid 和 Pid 打印机的 USB 连接
OpenDevice：device, baudrate	建立与打印机的串口连接
OpenDevice：dstName, dstPort	建立与打印机的 WIFI 连接
OpenDevice：macAddress	建立与打印机的蓝牙连接
CloseDevice	关闭与打印机的连接
WIFISearchPort	通过 WIFI 搜索打印机
WriteBuffer	发送数据
ReadBuffer	读取数据
recordCommunicationData	记录数据
LogTrace	记录日志

- **OpenDevice**

  建立与打印机的 USB 连接。

函数

- public int **OpenDevice**()

返回值

返回值	情况
POS_SUCCESS	连接成功
ERR_PROCESSING	连接失败

示例代码

见方法举例中的建立与打印机 USB 连接。

● **OpenDevice：Vid, Pid**

建立与指定的 Vid 和 Pid 打印机的 USB 连接。

函数

- public int **OpenDevice** (int Vid,int Pid)

参数

- **Vid** 打印机的 VID。
- **Pid** 打印机的 PID。

返回值

返回值	情况
POS_SUCCESS	连接成功
ERR_PROCESSING	连接失败

示例代码

见方法举例中的建立与打印机 USB 连接。

● **OpenDevice：device, baudrate**

建立与打印机的串口连接。

函数

- public int **OpenDevice**(File device,int baudrate)

参数

- **device** 串口号，如：/dev/ttySAC3
- **baudrate** 波特率，打印机波特率必须与开发板串口波特率相同。

返回值

返回值	情况
POS_SUCCESS	连接成功
ERR_PROCESSING	连接失败

示例代码

见方法举例中的建立与打印机串口连接。

● **OpenDevice：dstName, dstPort**

建立与打印机的 WIFI 连接。

函数

- public int **OpenDevice**(String dstName, int dstPort)

参数

- **dstName** 打印机的 IP 地址，合法值如 192.168.1.200
- **dstPort** 端口号，9100 或 4000。

返回值

返回值	情况
POS_SUCCESS	连接成功
ERR_PROCESSING	连接失败

示例代码

见方法举例中的建立与打印机的 WIFI 连接。

● **OpenDevice：macAddress**

建立与打印机的蓝牙连接。

函数

- public int **OpenDevice**(String macAddress)

参数

- macAddress 打印机的 MAC 地址，合法值如 00:1B:35:07:16:AC

返回值

返回值	情况
POS_SUCCESS	连接成功
ERR_PROCESSING	连接失败

示例代码

见方法举例中的建立与打印机的蓝牙连接。

● **CloseDevice**

关闭与打印机连接。

函数

- public int **CloseDevice**()

返回值

返回值	情况
POS_SUCCESS	成功
ERR_PROCESSING	失败

示例代码

见方法举例中的关闭连接。

● **WIFISearchPort**

WIFI 搜索打印机。

函数

- public int **WIFISearchPort**(SearchPortInfo port_info[],int printer_num_max)

参数

- SearchPortInfo　　搜索到的打印机设备信息。
- printer_num_max　　要求搜索到设备信息最大值。

返回值

返回值	情况
搜索到的打印机设备个数	搜索到打印机
0	搜索不到打印机

打印机的信息列表如下：

```
@ public class SearchPortInfo
{
    private String Allinfo;      //所有信息
    private String Prt_name;     //设备名称
    private String IP_address;   //设备IP地址
    private String MAC_address; //设备 MAC 地址
}
```

示例代码

见方法举例中的 WIFI 搜索打印机。

● **WriteBuffer**

向打印机设备发送数据，数据包大于 4096 字节时拆包为 4096 字节。

函数

- public int **WriteBuffer** (byte[] WriteBuffer,int OffsetSize,int nBytesToWrite, int WriteTimeOut)

参数

- **WriteBuffer** 待发送数据的存储空间地址，非空
- **OffsetSize** 首个发送字节在 WriteBuffer 中的偏移量
- **nBytesToWrite** 发送字节数
- **WriteTimeOut** 超时时间，单位毫秒，正数

返回值

返回值	情况
发送数据的字节数	发送数据成功
实际发送数据字节数	发送部分数据
0	未发送成功

示例代码

```
pszCommand [2] = {0x1b,0x40};
nReturn = WriteBuffer(pszCommand,0,2,WRITETIMEOUT);
```

● **ReadBuffer**

从打印机设备读取数据。

函数

- public int **ReadBuffer** (byte[] ReadBuffer,int OffsetSize,int nBytesToRead,int ReadTimeOut)

参数

- **ReadBuffer** 接收数据缓冲区，非空
- **OffsetSize** 首个接收字节在 ReadBuffer 中的偏移量
- **nBytesToRead** 需要读取的字节数
- **ReadTimeOut** 读超时时间，单位毫秒

返回值

返回值	情况
读取的字节数	成功
0	读取数据失败

示例代码

```
byte pointBuffer[4];
nReturn = ReadBuffer(pointBuffer,0, pointBuffer.length,10000);
```

● **recordCommunicationDataEnable**

记录数据，记录的文件大小不超过 5M。

函数

- public int **recordCommunicationData**(Context contexts,int IsRecord,String FileName)

参数

- contexts 应用程序目录
- IsRecord 是否记录通讯数据
- FileName 记录数据文件的名称

返回值

返回值	情况
POS_SUCCESS	成功
ERR_PROCESSING	失败

示例代码

```
recordCommunicationData(MainActivity.this,0x01,data_file_name);
```

● **LogTrace**

记录日志，记录的文件大小不超过 5M，另外日志中记录的时间为系统的标准 GMT 时间，如果 Android 系统显示的时间对标准的 GMT 时间不一致（程序进行了修改），则记录的时间与 Android 设备显示的时间不一致。

函数

-public int **LogTrace**(Context contexts,int IsRecord,String FileName)

参数

- contexts　应用程序目录
- IsRecord　是否记录日志
- FileName 记录日志文件的名称

返回值

返回值	情况
POS_SUCCESS	成功
ERR_PROCESSING	失败

示例代码

```
LogTrace(MainActivity.this,0x01,log_file_name);
```

4.2 POS SDK API 相关

前缀	API	Description
前缀 system 系统相关函数	systemDownloadFile	下载文件
	systemReset	清除打印缓冲区数据，打印模式被设为上电时的默认值模式
	systemSelectPrintMode	选择打印模式
	systemSelectPaperType	选择纸张类型
	systemSetMotionUnit	设置横纵向可移动单位
	systemQueryStatus	查询打印机状态
	systemFeedLine	打印机进纸
	systemCutPaper	切纸
前缀 cashdrawer 钱箱	cashdrawerOpen	产生钱箱控制脉冲，输出到指定引脚
前缀 text 文本相关函数	textSelectCharSetAndCodePage	选择字符集和代码页
	textSetLineHeight	设置文本行高
	textSetCharacterSpace	设置字符间距
	textStandardModeAlignment	设置文本对齐方式(只在标准模式下的行首有效）
	textStandardModeUpsideDown	选择是否倒置打印（只在标准模式的行首有效）
	textPrint	打印字符
	textSelectFontMagnifyTimes	选择横纵向放大倍数

	textStandardModeRotate	标准模式下文本旋转打印
	textSelectFont	选择打印字符的字体及字体风格
	textEnterOrQuitColorPrint	选择 / 取消双色打印模式
	textSetColorPrint	设置打印字符颜色
	textUserDefinedCharacterEnable	用户自定义字符功能使能
	textUserDefinedCharacterDefine	用户自定义字符下载
	textUserDefinedCharacterCancel	取消用户自定义的字符
前缀 image 图像相关函数	imageCreateRasterBitmap	创建光栅化图像
	imageStandardModePrint	标准模式下打印图像
	imageDownloadToPrinterRAM	RAM 图像下载
	imageRAMPrint	RAM 图像打印
	imageDownloadToPrinterFlash	Flash 图像下载
	imageFlashPrint	Flash 图像打印
	imageCreateRasterBitmap	创建光栅化图像
	imageStandardModeRasterPrint	标准模式下光栅位图打印
前缀 barcode 条码相关函数	barcodePrint1Dimension	选择一维条码类型
	barcodePrintQR	打印 QR 码
	barcodePrintPDF417	打印 PDF417 条码
	barcodePrintMaxicode	打印 Maxicode 条码
	barcodePrintGS1DataBar	打印 GS1 DataBar 条码和 GS1 复合码
前缀 standardMode 标准模式	standardModeSetPrintAreaWidth	设置标准模式的打印左边距及打印区域宽度
	standardModeSetStartingPosition	设置标准模式的横向起始坐标
前缀 pageMode 页模式	pageModeSetStartingPosition	设置页模式的纵向起始坐标
	pageModeSetPrintArea	设置页模式的打印区域
	pageModePrint	页模式打印
	pageModeClearBuffer	页模式清空 Buffer

- **systemDownloadFile**

  下载文件。

函数

- public int **systemDownloadFile**(String FileName,int TimeOut)

参数

- **FileName** 下载文件的文件名
- **TimeOut** 下载文件超时时间，以毫秒为单位

返回值

返回值	情况
POS_SUCCESS	成功
ERR_COMMUNICATE	失败
ERR_PARAM	文件名错误或文件数据为空

示例代码

```
String str = "/data/bmp/Printer.JK";
error_code = pos_sdk.systemDownloadFile(str,5000);
```

● **systemReset**

清除打印缓冲区数据，打印模式被设为上电时的默认值模式。

函数

- public int **systemReset**()

返回值

返回值	情况
POS_SUCCESS	成功
ERR_PROCESSING	失败

示例代码

```
pos_sdk.systemReset();
```

● **systemSelectPrintMode**

选择打印模式 （该指令只在行首有效）。

函数

- public int **systemSelectPrintMode**(int Mode)

参数

- **Mode** 打印模式

PrintMode 取值	描述
PRINT_MODE_STANDARD	标准模式
PRINT_MODE_PAGE	页模式
其他	参数错误

返回值

返回值	情况
POS_SUCCESS	成功
ERR_SYSTEM_SELECT_PRINT_MODE	失败
ERR_PARAM	参数错误

示例代码

```
error_code = pos_sdk.systemSelectPrintMode(PRINT_MODE_STANDARD); //标准模式
error_code = pos_sdk.systemSelectPrintMode(PRINT_MODE_PAGE); //页模式
```

- **systemSelectPaperType**

选择纸张类型。

函数

- public int **systemSelectPaperType**(int PaperType)

参数

- **PaperType** 纸张类型

PaperType 取值	描述
PaperTypeCoutinuous	连续纸
PaperTypeMarked	标记纸
其他	参数错误

返回值

返回值	情况
POS_SUCCESS	成功
ERR_SYSTEM_SELECT_PAPER_TYPE	失败
ERR_PARAM	参数错误

示例代码

```
error_code = pos_wifi. systemSelectPaperType (PaperTypeCoutinuous); //连续纸
error_code = pos_wifi. systemSelectPaperType (PaperTypeMarked); //标记纸
```

● **systemSetMotionUnit**

设置横纵向移动单位。

函数

- public int **systemSetMotionUnit**(int HorizontalUnit,int VerticalUnit)

参数

- **HorizontalUnit** 横向移动单位

HorizontalUnit 取值	描述
0-255	合法值
其他	参数错误

- **VerticalUnit** 纵向移动单位

VerticalUnit 取值	描述
0-255	合法值
其他	参数错误

返回值

返回值	情况
POS_SUCCESS	成功
ERR_SYSTEM_SET_MOTION_UNIT	失败
ERR_PARAM	参数错误

示例代码

```
error_code = pos_sdk.systemSetMotionUnit(203,203);
```

● **systemQueryStatus**

查询打印机状态。

函数

- public int **systemQueryStatus**(byte[] QueryStatusBuffer,int ReadSize,int port_type)

参数

- **QueryStatusBuffer** 查询状态的缓冲区
- **ReadSize** 读入数据的字节数
- **Port_type** 接口类型

Port_type 取值	描述
1	USB
2	串口
3	WIFI
4	蓝牙
其他	参数错误

返回值

返回值	情况
POS_SUCCESS	成功
ERR_SYSTEM_QUERY_STATUS	失败
ERR_PARAM	参数错误

示例代码

```
final int QueryStatusSize=4;
byte StatusBuffer[] = new byte[QueryStatusSize];
error_code = pos_sdk.systemQueryStatus(StatusBuffer,QueryStatusSize,1);
```

- **systemFeedLine**

  打印机进纸。

函数

- public int **systemFeedLine**(int LineNum)

参数

- **LineNum** 走纸行数

LineNum 取值	描述
0-255	合法值
其他	参数错误

返回值

返回值	情况
POS_SUCCESS	成功
ERR_SYSTEM_FEED_LINE	失败
ERR_PARAM	参数错误

示例代码

```
error_code = pos_sdk.systemFeedLine(5);
```

- **systemCutPaper**

选择切纸模式，打印机进纸并切纸。

函数

- public int **systemCutPaper**(int CutMode, int FeedDistance)

参数

- **CutMode** 切纸方式

LineNum 取值	描述
CutFullImmdediately	全切
CutPartImmdediately	半切
CutPartAfterFeed	走纸并半切
其他	参数错误

- **FeedDistance** 走纸距离

FeedDistance 取值	描述
0-255	走纸距离合法值
其他	参数错误

【说明】

1)如果参数为 CutFullImmdediately 和 CutPartImmdediately 时，走纸距离参数 FeedDistance 被忽略。

2)如果参数为 CutPartAfterFeed 时，打印机走纸 FeedDistance 并切纸。

3)标记纸模式时，走纸距离被忽略，打印机搜索标记并切纸。

返回值

返回值	情况
POS_SUCCESS	成功
ERR_SYSTEM_CUT_PAPER	失败
ERR_PARAM	参数错误

示例代码

```
error_code = pos_sdk.systemCutPaper(CutPartAfterFeed,80);
```

● **cashdrawerOpen**

产生钱箱控制脉冲，输出到指定引脚。

函数

- public int **cashdrawerOpen**(int CashdrawerID, int PulseOnTimes, int PulseOffTimes)

参数

- **CashdrawerID** 钱箱引脚

CashdrawerID 取值	描述
0	钱箱插座的引脚 2
1	钱箱插座的引脚 5
其他	参数错误

- **PulseOnTimes** 钱箱开启脉冲高电平时间

PulseOnTimes 取值	描述
0-255	合法值
其他	参数错误

- **PulseOffTimes** 钱箱开启脉冲低电平时间

PulseOffTimes 取值	描述
0-255	合法值
其他	参数错误

返回值

返回值	情况
POS_SUCCESS	成功
ERR_CASH_DRAWER_OPEN	失败
ERR_PARAM	参数错误

示例代码

```
error_code = pos_sdk.cashdrawerOpen(0,100,100);
```

● **textSelectCharSetAndCodePage**

选择国际字符集和代码页。

函数

- public int **textSelectCharSetAndCodePage**(int CharSet,int CodePage)

参数

- **CharSet** 国际字符集

CharSet 取值	描述
CharacterSetUSA	美国（U.S.A）
CharacterSetFrance	法国（France）
CharacterSetGermany	德国（Germany）
CharacterSetUK	英国（U.K）
CharacterSetDenmark_I	丹麦 I（Denmark I）
CharacterSetSweden	瑞典（Sweden）
CharacterSetItaly	意大利（Italy）
CharacterSetSpain_I	西班牙 I（Spain I）
CharacterSetJapan	日本（Japan）
CharacterSetNorway	挪威（Norway）
CharacterSetDenmark_II	丹麦 II （Denmark II）
CharacterSetSpain_II	西班牙 II Spain II
CharacterSetLatin_America	拉丁美洲（Latin America）
CharacterSetKorea	韩国（Korea）
其他	参数错误

- **CodePage** 国际代码页

CodePage 取值	描述
0	PC437
1	Katakana
2	PC850
3	PC860
4	PC863
5	PC865
16	WPC1252
17	PC866
18	PC852
19	PC858
12	PC857
13	771
14	Hebrew1

15	Hebrew2
21	Thai1
22	Thai2
23	Thai3
24	Thai4
25	Thai5
26	Thai6
27	Frasi
28	864[Arabic]
29	737[Greek]
32	1254[Turkish]
33	862[hebrew]
34	1251[Cyrillic]
35	1253[Greek]
36	855[Cyrillic]
37	774[Lithuanian]
38	928[Greek]
39	775[Baltic]
40	772[Lithuanian]
41	Hebrew3
42	851[Greek]
43	869[Greek]
44	1257[Baltic]
45	1250[Latin-2]
46	1255
47	1256[Arabic]
64	3840 (IBM-Russian)
65	3841 (Gost)
66	3843 (Polish)
67	3844 (CS2)
68	3845 (Hungarian)
69	3846 (Turkish)
70	3847 (Brazil-ABNT)
71	3848 (Brazil-ABICOMP)

72	1001 (Arabic)
73	2001 (Lithuanian-KBL)
74	3001 (Estonian-1)
75	3002 (Estonian-2)
76	3011 (Latvian-1)
77	3012 (Latvian-2)
78	3021 (Bulgarian)
79	3041 (Maltese)
80	8859
81	Persia

【说明】

几种类型的打印机可能不能支持所有的代码页。

返回值

返回值	情况
POS_SUCCESS	成功
ERR_TEXT_SELECT_CHAR_SET	失败
ERR_TEXT_SELECT_CODE_PAGE	失败
ERR_ ERR_PARAM	参数错误

示例代码

```
String txtbuf = “123456789abcdABCD”;
error_code = pos_sdk.textSelectCharSetAndCodePage(CharacterSetUSA,0);
error_code = pos_sdk.textPrint(txtbuf.getBytes(),txtbuf.getBytes().length);
```

- **textSetLineHeight**

设置行高。

函数

- public int **textSetLineHeight**(int Height)

参数

- **Height** 行高点数

Height 取值	描述
0-255	合法值
其他	参数错误

【说明】

1)如果参数 Height 为 0 时，行高将设置为默认值(1/6 inch)。

2)如果参数 Height 小于字符的高度，打印机将设置行高为字符的高度值。

返回值

返回值	情况
POS_SUCCESS	成功
ERR_TEXT_SET_LINE_HEIGHT	失败
ERR_PARAM	参数错误

示例代码

```
error_code = pos_sdk.textSetLineHeight(34);
```

● **textSetCharacterSpace**

设置字符间距。

函数

- public int **textSetCharacterSpace**(int LeftSpace,int RightSpace,int Mode)

参数

- **LeftSpace** 字符左间距（设置中文字符间距时，此参数需有效）

LeftSpace 取值	描述
0-255	合法值
其他	参数错误

- **RightSpace** 字符右间距

RightSpace 取值	描述
0-255	合法值
其他	参数错误

- **Mode** 字符模式

Mode 取值	描述
ChineseCharacterMode	中文字符模式
EnglishCharacterModes	英文字符模式
其他	参数错误

【说明】

1)如果参数 Mode 为 ChineseCharacterMode 时，参数 LeftSpace 和 RightSpace 必须同时合法，如果参数 Mode 为 EnglishCharacterMode 时，只要参数 RightSpace

合法便可，此时参数 LeftSpace 被忽略。

2)调用一次函数 textSetCharacterSpace:只可以单独更改英文或中文的字符间距，若想更改中英文字符间距需要调用函数两次。

返回值

返回值	情况
POS_SUCCESS	成功
ERR_TEXT_SET_CHARACTER_SPACE	失败
ERR_PARAM	参数错误

示例代码

```
error_code = pos_sdk.textSetCharacterSpace(10,50,EnglishCharacterMode);
```

- **textStandardModeAlignment**

设置文本打印的对齐方式（只在标准模式的行首有效）。

函数

- public int **textStandardModeAlignment**(int Alignment)

参数

- **Alignment** 对齐方式

Alignment 取值	描述
TextAlignmentLeft	左对齐
TextAlignmentCenter	居中
TextAlignmentRight	右对齐
其他	参数错误

返回值

返回值	情况
POS_SUCCESS	成功
ERR_TEXT_STANDARD_MODE_ALIGNMENT	失败
ERR_PARAM	参数错误

示例代码

```
error_code = pos_sdk.textStandardModeAlignment(TextAlignmentLeft);
```

● **textStandardModeUpsideDown**

选择是否倒置打印（只在标准模式的行首有效）。

函数

- public int **textStandardModeUpsideDown**(int UpsideDown)

参数

- **UpsideDown**　选择是否倒置打印

UpsideDown　取值	描述
1	倒置打印
0	不倒置打印
其他	参数错误

【说明】见 textStandardModeRoate

1)调用 textStandardModeUpsideDown:的参数 UpsideDown 为 FontStyleUpsideDown 的打印效果和调用- textStandardModeRotate(int Rotate)将参数 Rotate 设置为旋转 180°效果相同，且同时调用效果不叠加。

2)因调用- textStandardModeRotate(int Rotate)时，认为所有旋转情况不存在，因此先调用倒置再调用旋转，结果同旋转。先调用旋转（不旋转、顺时针 90 度、180 度、顺时针 270 度）再调用倒置，因倒置和旋转 180 度效果不叠加，结果为倒置、顺时针 270 度、180 度、顺时针 270 度。

3)建议不要同时调用倒置（textStandardModeUpsideDown）和旋转（textStandardModeRotate）。

返回值

返回值	情况
POS_SUCCESS	成功
ERR_TEXT_STANDARD_MODE_UPSIDEDOWN	失败
ERR_PARAM	参数错误

示例代码

```
error_code = pos_sdk.textStandardModeUpsideDown(1);
```

● **textPrint**

打印字符串。

函数

- public int **textPrint**(byte[] Buffer,int BytesOfBuffer)

参数

- Buffer　　要打印的文本内容缓冲区
- BytesOfBuffer　打印文本内容的字节数

返回值

返回值	情况
POS_SUCCESS	成功
ERR_TEXT_PRINT	失败
ERR_PARAM	字符串空

示例代码

见方法举例中的文本打印。

● **textSelectFontMagnifyTimes**

设置字符大小。

函数

- public int **textSelectFontMagnifyTimes**(int HorizontalTimes,int VerticalTimes)

参数

- **HorizontalTimes**　横向放大倍数

HorizontalTimes　取值	描述
1-6	合法横向放大倍数
其他	非法参数

- **VerticalTimes**　横向放大倍数

VerticalTimes　取值	描述
1-6	合法纵向放大倍数
其他	非法参数

【说明】

1)在标准模式下，纵向是进纸方向，横向是垂直于进纸的方向。但是当字符顺时针旋转 90°时，横向和纵向颠倒。

2)页模式下，横向和纵向取决于区域的方向。

3)同一行字符的放大倍数不同时，所有的字符以底线对齐。

返回值

返回值	情况
POS_SUCCESS	成功
ERR_TEXT_SELECT_MAGNIFY_TIMES	失败
ERR_PARAM	参数错误

示例代码

```
error_code = pos_sdk.textSelectFontMagnifyTimes(2,3);
```

● **textStandardModeRotate**

设置字符旋转打印的旋转度数。

函数

- public int **textStandardModeRotate**(int Rotate)

参数

- **Rotate** 设置旋转度数

Rotate 取值	描述
RotatePrintNormal	不旋转
RotatePrintR90	顺时针旋转 90 度
RotatePrint180	旋转 180 度
RotatePrintL90	逆时针旋转 90 度
其他	非法参数

【说明】

1)调用 textStandardModeUpsideDown:的参数 UpsideDown 为 FontStyleUpsideDown 的打印效果和调用- textStandardModeRotate(int Rotate)将参数 Rotate 设置为旋转 180°效果相同，且同时调用效果不叠加。

2)因调用- textStandardModeRotate(int Rotate)时，认为所有旋转情况不存在，因此先调用倒置再调用旋转，结果同旋转。先调用旋转（不旋转、顺时针 90 度、180 度、顺时针 270 度）再调用倒置，因倒置和旋转 180 度效果不叠加，结果为倒置、顺时针 270 度、180 度、顺时针 270 度。

3)建议不要同时调用倒置（textStandardModeUpsideDown:）和旋转（textStandardModeRotate:）

返回值

返回值	情况
POS_SUCCESS	成功
ERR_TEXT_STANDARD_MODE_ROTATE	失败
ERR_PARAM	参数错误

示例代码

```
error_code = pos_sdk.textStandardModeRotate(RotatePrintR90);// 顺时针旋转 90 度
```

● **textSelectFont**

选择打印字符的字体及字体风格。

函数

- public int **textSelectFont**(int FontType, int FontStyle)

参数

- **FontType** 字体类型

FontType 取值	描述
FontTypeStandardASCII	标准 ASCII
FontTypeCompressedASCII	压缩 ASCII
FontTypeUserDefined	用户自定义字符
FontTypeChinese	中文字符
其他	参数错误

- **FontStyle** 字体风格

FontStyle 取值	描述
FontStyleReverse	反显
FontStyleBold	粗体
FontStyleUpsideDown	倒置
FontStyleUnderlineOneDotThick	一点粗下划线
FontStyleUnderlineTwoDotThick	两点粗下划线

【说明】

1)在黑白反显打印模式选择时，下划线模式不起作用

2)不对顺时针旋转 90 度和 270 度的字符加下划线。

3)对于中文选择参数 FontStyle 为 FontStyleUnderlineOneDotThick 和 FontStyleUnderlineTwoDotThick 时，效果相同，均打印一点粗下划线。

4)函数第二个参数 FontStyle 可以为上述字符风格值的组合，如设置反显和粗

体，该参数可设为 FontStyleReverse| FontStyleBold。

返回值

返回值	情况
POS_SUCCESS	成功
ERR_TEXT_SELECT_FONT_TYPE	选择字体失败
ERR_TEXT_SET_FONT_STYLE_REVERSE	反显失败
ERR_TEXT_SET_FONT_STYLE_BOLD	粗体失败
ERR_TEXT_SET_FONT_STYLE_UNDERLINE	下划线失败
ERR_PARAM	参数错误

示例代码

```
error_code = pos_sdk.textSelectFont(FontTypeStandardASCII, FontStyleReverse| FontStyleBold);
```

- **textEnterOrQuitColorPrint**

  进入/退出双色打印模式。

函数

- public int **textEnterOrQuitColorPrint**(int ColorPrint)

参数

- **ColorPrint**　字体

ColorPrint 取值	描述
0	退出双色打印
1	进入双色打印
其他	参数错误

返回值

返回值	情况
POS_SUCCESS	成功
ERR_TEXT_ENTER_QUIT_COLOR_PRINT	进入双色打印失败
ERR_PARAM	参数错误

示例代码

```
error_code = pos_sdk. textEnterOrQuitColorPrint(1);
```

● **textSetColorPrint**

设置打印字符的颜色。

函数

- public int **textSetColorPrint**(int Color)

参数

- **Color** 打印字体颜色

Color 取值	描述
0	选择颜色 1
1	选择颜色 2
其他	参数错误

返回值

返回值	情况
POS_SUCCESS	成功
ERR_TEXT_SET_COLOR_PRINT	设置颜色失败
ERR_PARAM	参数错误

示例代码

```
error_code = pos_sdk. textSetPrintColor(1);
```

● textUserDefinedCharacterEnable

选择或取消用户自定义字符。

函数

- public int **textUserDefinedCharacterEnable**(int Enable)

参数

- **Enable** 选择或者取消用户自定义字符

Enable 取值	描述
FontUserDefinedDisable	取消用户自定义字符
FontUserDefinedEnable	选择用户自定义字符
其他	非法参数

【说明】

可以选择和取消全部的自定义字符。

返回值

返回值	情况
POS_SUCCESS	成功
ERR_TEXT_FONT_USER_DEFINED_ENABLE	失败
ERR_PARAM	参数错误

示例代码

详细使用见方法举例中的用户自定义字符。

● **textUserDefinedCharacterDefine**

定义用户自定义字符。

函数

- public int **textUserDefinedCharacterDefine**(int BytesOfHeight,int DotsOfWidth, int StartingCode,int EndingCode, Bitmap []image)

参数

- **BytesOfHeight** 指定纵向字节数
- **DotsOfWidth** 指定横向点数
- **StartingCode** 起始字符代码
- **EndingCode** 终止字符代码
- **image** 要下载的图像数组

各参数合法值

参数	合法取值
BytesOfHeight	3
DotsOfWidth	9 或 12
StartingCode	32-127
EndingCode	32-127

【说明】

1)参数 BytesOfHeight 必须为 3。

2)自定义的字符的图像必须为 9*17 或 12*24。

3)字符打印的反显、下划线、间距设置、行高设置、对齐方式设置、旋转打印等字符打印风格，均能影响自定义字符打印。

返回值

返回值	情况
POS_SUCCESS	成功
ERR_TEXT_FONT_USER_DEFINED	失败
ERR_PARAM	参数错误或数据个数不匹配

示例代码

详细使用见方法举例中的用户自定义字符。

- **textUserDefinedCharacterCancel**

  取消指定的自定义字符。

函数

- public int **textUserDefinedCharacterCancel**(int CharCode)

参数

- **CharCode** 被取消的用户自定义字符的代码

CharCode 取值	描述
32-127	合法值
其他	非法参数

【说明】

可以取消指定 CharCode 代码的单个用户自定义字符。

返回值

返回值	情况
POS_SUCCESS	成功
ERR_TEXT_FONT_USER_DEFINED_CANCEL	失败
ERR_PARAM	参数错误

示例代码

详细使用见方法举例中的用户自定义字符。

- **imageCreateRasterBitmap**

  创建光栅化图像。

函数

- public Bitmap **imageCreateRasterBitmap** (String printText, int textSize, int

bold)

参数

- **printText** 要创建图像的字符串
- **textSize** 画布尺寸

textSize 取值	描述
1-200	合法值
其他	非法参数

- **bold** 打印内容粗细程度

bold 取值	描述
1-5	合法值
其他	非法参数

返回值

返回值	情况
Bitmap 类对象	成功
null	失败

示例代码

见方法举例中文本光栅化打印。

● **imageStandardModePrint**

下载并打印位图（只在标准模式下有效）。

函数

- public int **imageStandardModePrint**(Bitmap image,int SingleDoubleFlag,int StartHorPos,int PrinterWidth)

参数

- **image** 要打印的图像
- **SingleDoubleFlag** 状态标志，非负整数

Mode 取值	描述
SingleDensity_8	8 点单精度
DoubleDensity_8	8 点双精度
SingleDensity_24	24 点单精度
DoubleDensity_24	24 点双精度

其他	非法参数

- **StartHorPos** 初始打印位置
- **PrinterWidth** 打印头宽度

PrinterWidth 取值	描述
0	图像不缩放
64-65535	合法的打印头宽度，图像正常缩放
其他	非法参数

返回值

返回值	情况
POS_SUCCESS	成功
ERR_IMAGE_DOWNLOAD_AND_PRINT	失败
ERR_PARAM	参数错误或图像为空

示例代码

见方法举例中的位图打印。

● **imageDownloadToPrinterRAM**

下载 RAM 图像。

函数

- public int **imageDownloadToPrinterRAM**(int ImageID, Bitmap image,int PrinterWidth)

参数

- **ImageID** 下载图像的 ID 号

ImageID 取值	描述
0-7	合法 ID 号
其他	非法参数

- **Image** 要下载的图像
- **PrinterWidth** 打印头宽度

PrinterWidth 取值	描述
0	图像不缩放
64-65535	合法的打印头宽度，图像正常缩放
其他	非法参数

【说明】

1)每次只能下载单幅图像，并指定其 ID 号。

2)打印机重新上电或者重新建立端口连接时，已下载的图像被删除。

返回值

返回值	情况
POS_SUCCESS	成功
ERR_IMAGE_DOWNLOAD_RAM	失败
ERR_PROCESSING	图像缩放失败
ERR_PARAM	参数错误

示例代码

见方法举例中的 RAM/Flash 位图下载及打印

- **imageRAMPrint**

打印 RAM 中下载的位图。

函数

- public int **imageRAMPrint**(int ImageID,int Mode)

参数

- **ImageID** 打印图像的 ID 号

ImageID 取值	描述
0-7	合法 ID 号
其他	非法参数

- **Mode** 图像打印模式

Mode 取值	描述
NormalMode	正常大小
Double_width	图像倍宽
Double_height	图像倍高
Quadruple	图像倍宽倍高
其他	非法参数

返回值

返回值	情况
POS_SUCCESS	成功
ERR_IMAGE_RAM_PRINT	失败

ERR_PARAM	参数错误

示例代码

见方法举例中的 RAM/Flash 位图下载及打印

- **imageDownloadToPrinterFlash**

下载 Flash 位图。

函数

- public int **imageDownloadToPrinterFlash**(int image_num, Bitmap image[],int PrinterWidth)

参数

- **image_num** 下载到 Flash 中的图像个数
- **image** 图像数据数组
- **PrinterWidth** 打印头宽度

PrinterWidth 取值	描述
0	图像不缩放
64-65535	合法的打印头宽度，图像正常缩放
其他	非法参数

【说明】

1)每次下载删除所有的之前由函数 imageDownloadToPrinterFlash:下载到 Flash 中的图像。

2)下载多幅图像时，若其中有下载不成功的，清除已下载的图像。

3)打印机重新上电或者重新建立端口连接时，已下载的图像不会被删除。

4)多幅图像下载时，图像名称之间以“@”分隔开，如：S .bmp@Jpg.jpg@face.PNG 。

返回值

返回值	情况
POS_SUCCESS	成功
ERR_IMAGE_DOWNLOAD_FLASH	失败
ERR_PROCESSING	图像缩放失败
ERR_PARAM	参数错误

示例代码

见方法举例中的 RAM/Flash 位图下载及打印

● **imageFlashPrint**

打印 Flash 位图。

函数

- public int **imageFlashPrint**(int ImageID,int Mode)

参数

- **ImageID** 打印图像的 ID 号

ImageID 取值	描述
1-255	合法 ID 号
其他	非法参数

- **Mode** 图像打印模式

Mode 取值	描述
NormalMode	正常大小
Double_width	图像倍宽
Double_height	图像倍高
Quadruple	图像倍宽倍高
其他	非法参数

返回值

返回值	情况
POS_SUCCESS	成功
ERR_IMAGE_FLASH_PRINT	失败
ERR_PARAM	参数错误

示例代码

见方法举例中的 RAM/Flash 位图下载及打印

● **imageStandardModeRasterPrint**

标准模式下打印光栅位图，在准备数据光栅打印需要发送 1B 40 指令，该指令不会对其他操作产生影响。

函数

- public int **imageStandardModeRasterPrint**(Bitmap image, int PrinterWidth)

参数

- **Image** 要打印的图像
- **PrinterWidth** 打印头宽度

PrinterWidth 取值	描述
0	图像不缩放
64-2040	合法的打印头宽度，图像正常缩放
其他	非法参数

返回值

返回值	情况
POS_SUCCESS	成功
ERR_IMAGE_STANDARD_MODE_RASTER_PRINT	失败
ERR_PROCESSING	图像缩放失败
ERR_PARAM	参数错误

示例代码

见方法举例中的光栅化位图打印。

● **barcodePrint1Dimension**

打印一维条码。

函数

- public int **barcodePrint1Dimension**(String pszBuffer,int DataLength,int nType, int nWidthX,int nHeight, int nHriFontType, int nHriFontPosition)

参数

- **pszBuffer** 条码数据
- **DataLength** 条码数据字节长度
- **nType** 条码类型

nType 取值	描述	条码数据长度 取值
BarcodeUPC_A	UPC-A	11 -12
BarcodeUPC_E	UPC-E	11-12
BarcodeJAN13orEAN13	EAN13	12-13
BarcodeJAN8orEAN8	EAN-8	7-8
BarcodeCODE39	Code39	1-255
BarcodeITF	交叉 25 码	1-255
BarcodeCODABAR	CodaBar	1-255

BarcodeCODE93	Code93	1-255
BarcodeCODE128	Code128	2-255
其他	非法参数	

- **nWidthX** 基本模块宽度

nWidthX 取值	描述
2-6	合法的条码基本模块宽度
其他	非法参数

- **nHeight** 条码高度

nHeight 取值	描述
1-255	合法的条码高度参数
其他	非法参数

- **nHriFontType** 字体类型

nHriFontType 取值	描述
FontTypeStandardASCII=0	标准 ASCII
FontTypeCompressedASCII=1	压缩 ASCII
其他	非法参数

- **nHriFontPosition** HRI 字符的打印位置

nHriFontPosition 取值	描述
HRINone	不打印 HRI 字符
HRIAbove	HRI 字符打印于条码上方
HRIBelow	HRI 字符打印于条码下方
HRIAboveAndBelow	HRI 字符在条码上方和下方均打印
其他	非法参数

【说明】具体说明见附录 B. 条码说明 和 附录 C. 128 码。

返回值

返回值	情况
POS_SUCCESS	成功
ERR_BARCODE_PRINT_1D	失败
ERR_BARCODE_SELECT_MODULE_WIDTH	设置模块宽度失败
ERR_BARCODE_SELECT_BARCODE_HEIGHT	设置条码高度失败
ERR_BARCODE_SELECT_HRI_FONT_TYPE	设置 HRI 字体类型失败
ERR_BARCODE_SELECT_HRI_FONT_POSITION	设置 HRI 字符打印位置失败

ERR_PARAM	参数错误

示例代码

```
String str_data = “123456789012”
int data_lengt = str_data.getBytes().length;
error_code = pos_sdk barcodePrint1Dimension(str_data, data_lengt,BarcodeUPC_A,3,100,0,1);
```

- **barcodePrintQR**

设置 QR 码参数并打印 QR 码。

函数

- public int **barcodePrintQR**(String pszBuffer,int DataLength,int nOrgx,int BasicElementWidth,int SymbolType,int LanguageMode)

参数

- **pszBuffer** 要打印的条码数据
- **DataLength** 条码数据字节数
- **nOrgx** 起始点与左边界距离点数
- **BasicElementWidth** 条码基本元素宽度

BasicElementWidth 取值	**描述**
1-10	合法的条码基本元素宽度
其他	非法参数

- **SymbolType** 符号类型

SymbolType 取值	**描述**
OriginalType	原始类型
EnhancedType	增强型（建议）
其他	非法参数

- **LanguageMode** 语言模式

LanguageMode 取值	**描述**
LanguageChinese	中文
LanguageJapanese	日文
其他	非法参数

【说明】

1)设置条码数据和基本元素宽度使条码超出打印的页面时，不打印条码。

2)建议使用 EnhancedType 类型。

返回值

返回值	情况
POS_SUCCESS	成功
ERR_BARCODE_QR_SET_PARAM	失败
ERR_BARCODE_PRINT_2D	选择条码类型为 QR 失败
ERR_PROCESSING	发送条码数据失败
ERR_PARAM	参数错误

示例代码

见方法举例中的 QR 码打印。

- **barcodePrintPDF417**

设置 PDF417 条码参数并打印 PDF417 条码。

函数

- public int **barcodePrintPDF417**(String pszBuffer,int DataLength,
int AppearanceToHeight,int AppearanceToWidth,int RowsNumber,
int ColumnsNumber,int Xsize, int LineHeight,int nCorrectGrade)

参数

- **pszBuffer** 要打印的条码数据
- **DataLength** 条码数据字节数
- **AppearanceToHeight** 外观比高度
- **AppearanceToWidth** 外观比宽度
- **RowNumber** 行数
- **ColumnNumber** 列数
- **Xsize** X 尺寸
- **LineHeight** 行高
- **nCorrectionGrade** 纠错等级

各参数合法值列表如下：

参数	合法值
AppearanceToHeight	1-10
AppearanceToWidth	1-100
RowNumber	3-90
ColumnNumber	1-30

Xsize	1-7
LineHeight	2-25
CorrectionGrade	0-8

【说明】

1)数据超界，不打印条码。

2)条码大小超出页面范围不打印。

返回值

返回值	情况
POS_SUCCESS	成功
ERR_BARCODE_PDF417_SET_SIZE	失败
ERR_BARCODE_PDF417_SELECT_CORRECTION_GRADE	设置纠错等级失败
ERR_BARCODE_PRINT_2D	选择打印 PDF417 条码失败
ERR_PROCESSING	发送条码数据失败
ERR_PARAM	参数错误

示例代码

见方法举例中的 PDF417 条码打印。

- **barcodePrintMaxicode**

打印 Maxicode 条码。

函数

- public int **barcodePrintMaxicode**(String pszBuffer, int DataLength)

参数

- **pszBuffer** 要打印的条码数据
- **DataLength** 条码数据的字节数

【说明】

数据超界，不打印条码。

返回值

返回值	情况

POS_SUCCESS	成功
ERR_BARCODE_PRINT_2D	选择打印 Maxicode 条码失败
ERR_PROCESSING	发送条码数据失败
ERR_PARAM	参数错误

示例代码

见方法举例中 Maxicode 条码打印

● **barcodePrintGS1DataBar**

设置 GS1 DataBar 条码参数并采用指令模式打印 GS1 条码，GS1 条码是独立条码或者复合条码，通过编码数据中是否存在数据分隔符“|”区分，存在“|”为复合码，否则是独立的 DataBar 条码。

函数

- public int **barcodePrintGS1DataBar**(String pszBuffer,int DataLength,int BarcodeType,int BasicElementWidth,int BarcodeHeight,int BasicElementHeight,int SeparatorHeight,int SegmentHeight,int HRI,int AI)

参数

- **pszBuffer** 要打印的条码数据
- **DataLength** 条码数据的字节数
- **BarcodeType** GS1 条码类型

BarcodeType 取值	**描述**
GS1DataBarOmnidirectional	全向型 GS1DataBar Omnidirectional
GS1DataBarTruncated	截短型 GS1DataBar Truncated
GS1DataBarStacked	层排型 GS1 DataBar Stacked
GS1DataBarStackedOmnidirectional	全向层排型 GS1 DataBar Stacked Omnidirectiona
GS1DataBarLimited	限定性 GS1 DataBar Limited
GS1DataBarExpanded	扩展型 GS1 DataBar Expanded
GS1DataBarExpandedStacked	扩展层排型 GS1 DataBar ExpandedStacked
其他	非法参数

- **BasicElementWidth** 基本元素宽度

BasicElementWidth 取值	**描述**
1-6	合法基本元素宽度

其他	非法参数

- **BarcodeHeight** DataBar 的高度，如果条码是 GS1DataBarStacked、GS1DataBarStackedOmnidirectional、GS1DataBarExpandedStacked 是指多行条码符号中每一行的高度。

BarcodeHeight 取值	**描述**
2-250	合法高度
其他	非法参数

- **BasicElementHeight** 复合码中 2D 条码符号基本元素高度

BasicElementHeight 取值	**描述**
1-10	合法基本元素高度
其他	非法参数

- **SeparatorHeight** 分隔符的高度。 条码类型是 DataBar 复合码或者独立的 GS1DataBarStacked、GS1DataBarStackedOmnidirectional、GS1DataBarExpandedStacked 需要设置此参数。

SeparatorHeight 取值	**描述**
1-10	合法分隔符的高度
其他	非法参数

- **SegmentNumber** 每行条码符号的段数，只有条码类型是 GS1DataBarExpandedStacked 时需要设置此参数。

SegmentNumber 取值	**描述**
2-20	合法的独立的扩展层排型条码参数范围
4-20	合法的复合的扩展层排型条码时参数范围
其他	非法参数

- **HRI** 注释字符内容

HRI 取值	**描述**
1	复合码时注释字符是 DataBar 和 2D 两部分的数据，独立码时注释字符是 DataBar 数据
2	复合码或独立码 打印 DataBar 部分的数据
3	复合码时打印 2D 部分的数据 独立码时不打印
4	无注释字符
其他	非法参数

- **AI** 是否应用 AI（应用标识符）：0 表示不应用 AI；1 表示应用 AI。

【说明】

1)几种型号打印机无法打印 GS1 的所有类型条码。

2)数据超界，不打印条码。

3)条码大小超出页面范围不打印。

返回值

返回值	情况
POS_SUCCESS	成功
ERR_BARCODE_GS1DATABAR_SET_PARAM	失败
ERR_BARCODE_PRINT_2D	选择 GS1 DataBar 类型失败
ERR_PROCESSING	发送条码数据失败
ERR_PARAM	参数错误

示例代码

见方法举例中 GS1 DataBar 和 GS1 复合条码打印。

- **standardModeSetPrintAreaWidth**

标准模式设置打印左边距及区域宽度（只在标准模式的行首有效）。

函数

- public int **standardModeSetPrintAreaWidth**(int LeftMargin,int Width)

参数

- **LeftMargin** 左边距

LeftMargin 取值	描述
0-65535	合法左边距
其他	非法参数

- **Width** 打印区域宽度

Width 取值	描述
0-65535	合法打印区域宽度
其他	非法参数

【说明】

1)参数 LeftMargin 和 Width 设置，不影响用户自定义字符、光栅化位图、光栅化字符的打印范围。

2)页模式此接口函数无效。

返回值

返回值	情况

POS_SUCCESS	成功
ERR_STANDARD_MODE_SET_PRINTAREA_WIDTH	设置区域宽度失败
ERR_STANDARD_MODE_SET_LEFT_MARGIN	设置左边距失败
ERR_PARAM	参数错误

示例代码

```
eror_code = pos_sdk. standardModeSetPrintAreaWidth (100,500);
```

- **standardModeSetStartingPosition**

标准模式设置打印起始横坐标位置。

函数

- public int **standardModeSetStartingPosition**(int X)

参数

- X　　横向绝对起始坐标

X　取值	描述
0-65535	合法值
其他	非法参数

【说明】

参数 X 设置，不影响用户自定义字符、光栅化位图、光栅化字符的打印范围。

返回值

返回值	情况
POS_SUCCESS	成功
ERR_STANDARD_MODE_SET_HORIZONTAL_STARTING_POSITION	失败
ERR_PARAM	参数错误

示例代码

```
error_code = pos_sdk. standardModeSetStartingPosition (100);
```

- **pageModeSetStartingPosition**

页模式设置打印的横向、纵向起始位置。

函数

- public int **pageModeSetStartingPosition**(int X,int Y)

参数

- X　　横向绝对起始坐标

X　取值	描述
0-65535	合法值
其他	非法参数

- Y　　纵向绝对起始坐标

Y　取值	描述
0-65535	合法值
其他	非法参数

返回值

返回值	情况
POS_SUCCESS	成功
ERR_STANDARD_MODE_SET_HORIZONTAL_STARTING_POSITION	设置横向起始坐标失败
ERR_PAGE_MODE_SET_VERTICAL_STARTING_POSITION	设置纵向起始坐标失败
ERR_PARAM	参数错误

示例代码

```
error_code = pos_sdk.pageModeSetStartingPosition(203,203);
```

● **pageModeSetPrintArea**

页模式的打印区域设置。

函数

- public int **pageModeSetPrintArea**(int X,int Y, int AreaWidth,int AreaHeight,int Direction)

参数

- **X**　　横向起始位置
- **Y**　　纵向起始位置
- **AreaWidth**　　打印区域宽度
- **AreaHeight**　　打印区域高度

X/Y/AreaWidth/AreaHeight 取值	描述
0-65535	合法值
其他	非法参数

- **Direction** 打印方向

Direction 取值	描述
LeftToRight=0	从左向右打印
BottomToTop=1	从下向上打印
RightToLeft=2	从右向左打印
TopToBottom=3	从上向下打印
其他	非法参数

返回值

返回值	情况
POS_SUCCESS	成功
ERR_PAGE_MODE_SET_PRINT_AREA	设置打印区域失败
ERR_PAGE_MODE_SET_PRINT_DIRECTION	设置打印方向失败
ERR_PARAM	参数错误

示例代码

```
error_code = pos_sdk.pageModeSetPrintArea(100,0,400,1000,LeftToRight);
```

● **pageModePrint**

页模式下打印。

函数

- public int **pageModePrint**()

返回值

返回值	情况
POS_SUCCESS	成功
ERR_PAGE_MODE_PRINT	失败

示例代码

```
error_code = pos_sdk.pageModePrint();
```

- **pageModeClearBuffer**

  页模式下清空 Buffer。

函数

- public int **pageModeClearBuffer**()

返回值

返回值	情况
POS_SUCCESS	成功
ERR_PAGE_MODE_CLEAR_BUFFER	失败

示例代码

```
error_code = pos_sdk.pageModeClearBuffer();
```

5. 附录

附录 A. 错误码列表

错误码	描述
POS_SUCCESS	成功
ERR_PARAM	参数错误
ERR_SYSTEM_RESET	打印机初始化失败
ERR_SYSTEM_SELECT_PRINT_MODE	设置打印模式失败
ERR_SYSTEM_SELECT_PAPER_TYPE	选择纸张类型失败
ERR_SYSTEM_SET_MOTION_UNIT	设置横纵向移动单位失败
ERR_SYSTEM_QUERY_STATUS	查询打印机状态失败
ERR_SYSTEM_FEED_LINE	打印机进纸失败
ERR_SYSTEM_CUT_PAPER	打印机切纸失败
ERR_CASH_DRAWER_OPEN	钱箱弹出失败
ERR_TEXT_SELECT_CHAR_SET	选择字符集失败
ERR_TEXT_SELECT_CODE_PAGE	选择代码页失败
ERR_TEXT_SET_LINE_HEIGHT	设置字符行高失败
ERR_TEXT_SET_CHARACTER_SPACE	设置字符间距失败
ERR_TEXT_STANDARD_MODE_ALIGNMENT	标准模式下设置对齐方式失败
ERR_TEXT_SELECT_FONT_TYPE	选择字体类型失败
ERR_TEXT_SET_FONT_STYLE_REVERSE	反显打印失败
ERR_TEXT_SET_FONT_STYLE_BOLD	粗体打印失败
ERR_TEXT_SET_FONT_STYLE_UNDERLINE	打印下划线效果失败
ERR_TEXT_STANDARD_MODE_UPSIDEDOWN	倒置打印失败
ERR_TEXT_SELECT_MAGNIFY_TIMES	选择放大倍数失败
ERR_TEXT_STANDARD_MODE_ROTATE	标准模式下旋转打印失败
ERR_TEXT_ENTER_QUIT_COLOR_PRINT	选择/取消双色打印失败
ERR_TEXT_SET_COLOR_PRINT	设置颜色打印失败
ERR_TEXT_FONT_USER_DEFINED_ENABLE	选择用户自定义字符功能失败

ERR_TEXT_FONT_USER_DEFINED	用户自定义字符失败
ERR_TEXT_FONT_USER_DEFINED_CANCEL	取消用户自定义字符失败
ERR_TEXT_PRINT	打印文本失败
ERR_IMAGE_DOWNLOAD_AND_PRINT	图像下载并打印失败
ERR_IMAGE_DOWNLOAD_RAM	下载 RAM 位图失败
ERR_IMAGE_RAM_PRINT	打印 RAM 中下载的位图失败
ERR_IMAGE_DOWNLOAD_FLASH	下载 Flash 位图失败
ERR_IMAGE_FLASH_PRINT	打印 Flash 中下载的位图失败
ERR_IMAGE_STANDARD_MODE_RASTER_PRINT	标准模式下打印光栅位图失败
ERR_STANDARD_MODE_SET_PRINTAREA_WIDTH	设置标准模式的打印区域宽度失败
ERR_STANDARD_MODE_SET_LEFT_MARGIN	设置标准模式的打印左边距失败
ERR_STANDARD_MODE_SET_HORIZONTAL_STARTING_POSITION	设置标准模式的横向起始位置失败
ERR_PAGE_MODE_SET_VERTICAL_STARTING_POSITION	设置页模式的纵向起始坐标失败
ERR_PAGE_MODE_SET_PRINT_AREA	设置页模式的打印区域失败
ERR_PAGE_MODE_SET_PRINT_DIRECTION	设置页模式的打印方向失败
ERR_PAGE_MODE_PRINT	页模式下打印失败
ERR_PAGE_MODE_CLEAR_BUFFER	页模式清空 Buffer 失败
ERR_BARCODE_PRINT_1D	选择要打印一维条码失败
ERR_BARCODE_PRINT_2D	选择要打印的二维条码失败
ERR_BARCODE_SELECT_MODULE_WIDTH	选择条码基本模块宽度失败
ERR_BARCODE_SELECT_BARCODE_HEIGHT	选择条码高度失败
ERR_BARCODE_SELECT_HRI_FONT_TYPE	选择条码的 HRI 字体失败
ERR_BARCODE_SELECT_HRI_FONT_POSITION	选择 HRI 字符的打印位置失败

ERR_BARCODE_QR_SET_PARAM	设置 QR 码参数失败
ERR_BARCODE_PDF417_SELECT_CORRECTION_GRADE	设置 PDF417 的纠错等级失败
ERR_BARCODE_PDF417_SET_SIZE	设置 PDF417 的大小失败
ERR_BARCODE_GS1DATABAR_SET_PARAM	设置 GS1 DataBar 的参数失败

附录 B. 条码说明

各条码类型对应的条码数据长度及字符集，如下表：

条码类型	数据长度	字符集	备注
UPC-A	11-12	48 ≤ d ≤ 57	
UPC-E	11-12	48 ≤ d ≤ 57,d1=48	第一个字符必须为 0
JAN13(EAN13)	12-13	48 ≤ d ≤ 57	
JAN 8 (EAN8)	7-8	48 ≤ d ≤ 57	
CODE39	1-255	45 ≤ d ≤ 57, 65 ≤ d ≤ 90, 32, 36, 37,43	
ITF	1-255	48 ≤ d ≤ 57	
CODABAR	1-255	48 ≤ d ≤ 57 65 ≤ d ≤ 68, 36, 43,45,46,47 58	CODEBAR 码起始符和结束符都必须为 A、B、C、D 四个字母中的一个，结束符可以使用 T、E、*、N 四字符来代替
CODE93	1-255	0 ≤ d ≤ 127	
CODE128	2-255	0 ≤ d ≤ 127	条码数据前必须先选择字符集
PDF417	1-255	0 ≤ d ≤ 255	
QRCODE	4-255	0 ≤ d ≤ 255	
MAXICODE	1-138	48 ≤ d ≤ 57,65 ≤ d ≤ 90	
GS1	1-255	由选择的 GS1 参数决定	

GS1条码类型对应的条码数据长度及字符集

条码类型	数据长度	字符集
全向型 GS1DataBar Omnidirectional	14 位， 13 位数字+1 位校验字符	数字 0-9

截短型 GS1DataBar Truncated	14 位， 13 位数字+1 位校验字符	数字 0-9
层排型 GS1 DataBar Stacked	14 位， 13 位数字+1 位校验字符	数字 0-9
全向层排型 GS1 DataBar Stacked Omnidirectiona	14 位， 13 位数字+1 位校验字符	数字 0-9
限定性 GS1 DataBar Limited	14 位， 13 位数字+1 位校验字符	数字 0-9
扩展型 GS1 DataBar Expanded	最大 74 个数字或 41 字母	0 ~ 9、A ~ Z、 a ~ z ！ " % & ' ( ) * + , - ./ : ; < = > ? _ 空格 FNC1
扩展层排型 GS1 DataBar ExpandedStacked	最大 74 个数字或 41 字母	0 ~ 9、A ~ Z、 a ~ z ！ " % & ' ( ) * + , - ./ : ; < = > ? _ 空格 FNC1

[应用注释**]**

• 当选择UPC-A 、UPC-E、JAN13 (EAN13)或者JAN8 (EAN8)码时，如果条码数据个数超出了规定的取值范围，不打印条码。

[应用注释 （标准模式）]

• 如果条码数据超出了规定的字符集取值范围，不打印条码。

• 如果条码横向超出了打印区域，不打印条码。

• 不管由textSetLineHeight:设定的行高是多少，走纸距离都与设定的条码高度相等。

• 只有在打印缓冲区没有数据时才有效，如果打印缓冲区有数据，该命令被忽略。

• 打印条码后，将打印位置设置在行首。

• 打印模式设置- public int textSelectFont(int FontType, int FontStyle)（**FontStyle** 为 FontStyleReverse/FontStyleBold/FontStyleUnderlineOneDotThick/FontStyleUnderlineTwoDotThick）不影响打印条码，但是倒置模式对条码打印有影响。

[应用注释 （页模式）]

• 将条码图形生成到打印缓冲区，但是并不打印。处理完条码数据后将打印位置移到条码的右边。

•如果条码数据超出了规定的字符集取值范围，不打印条码。

• 如果条码宽度超出了打印区域，不打印条码。

当选择 CODE128时：

•CODE 128的相关信息和字符集，见附录 128 码。

• 在使用CODE 128 时，按照下列说明进行编码：

①在条码数据前必须先选择字符集（CODE A、CODE B 和 CODE C中的一个）。

②选择字符集是通过发送字符“{” 和另外一个字符结合来完成的；ASCII字符“{”通过连续发送字符“{”两次来完成。

指定字符集	发送数据
SHIFT	{S
CODE A	{A
CODE B	{B
CODE C	{C
FNC1	{1
FNC2	{2
FNC3	{3
FNC4	{4
"{"	{{

例如：选择CODE B打印“ 123456”，需要输入：{B123456

- 如果在条码数据的最前端不是字符集选择，则不打印条码。
- 如果“{”和紧接着它的那个字符不是上面所指定的组合则不打印条码。
- 如果打印机接收的字符不是条码字符集数据，则不打印条码。
- 打印机打印HRI字符时，不打印shift字符和字符集选择数据。
- 功能字符的HRI字符不打印。
- 控制字符（<00>H to <1F>H and <7F>H）的HRI字符也不打印。
- 一定要保证条码的左右间隙，间隙因条码类型不同而不同。

附录 C. 128 码

1. 128码综述

128码通过交替使用字符集A、字符集B和字符集C，能够对128个ASCII字符和00~99的100个数字以及一些特殊字符进行编码。每个字符集编码的字符如下：

- 字符集 A： ASCII 字符 00H 到 5FH
- 字符集 B： ASCII 字符 20H 到 7FH
- 字符集 C： 00~99的100个数字

128码也能对下列特殊字符进行编码：

- SHIFT 字符

“SHIFT”能使条码符号SHIFT字符后边第一个字符从字符集A转换到字符集B，或从字符集B转换到字符集A，从第二个字符开始恢复到SHIFT以前所用的字符集。“SHIFT”字符仅能在字符集A和字符集B之间转换使用，它无法使当前的编

码字符进入或退出字符集C的状态。

- 字符集选择字符（CODE A、CODE B、 CODE C）

  这些字符能将其后边的编码字符转换到字符集A、B或C。

- 功能字符（FNC1、 FNC2、FNC3、FNC4）

  这些功能符的用处取决于应用软件。在字符集C中，只有FNC1 可用。

2. 字符集

字符集A中的字符

	发送数据		字符	发送数据		字符	发送数据	
	Hex	Decimal		Hex	Decimal		Hex	Decimal
NUL	00	0	(	28	40	P	50	80
L	01	1	)	29	41	Q	51	81
SOH	02	2	*	2A	42	R	52	82
STX	03	3	+	2B	43	S	53	83
ETX	04	4	,	2C	44	T	54	84
EOT	05	5	-	2D	45	U	55	85
ENQ	06	6	.	2E	46	V	56	86
ACK	07	7	/	2F	47	W	57	87
BEL	08	8	0	30	48	X	58	88
BS	09	9	1	31	49	Y	59	89
HT	0A	10	2	32	50	Z	5A	90
LF	0B	11	3	33	51	[	5B	91
VT	0C	12	4	34	52	\	5C	92
FF	0D	13	5	35	53	]	5D	93
CR	0E	14	6	36	54	^	5E	94
SO	0F	15	7	37	55	_	5F	95
SI	10	16	8	38	56	FN	7B,31	123,49
DLE	11	17	9	39	57	C1	7B,32	123,50
HC1	12	18	:	3A	58	FN	7B,33	123,51
HC2	13	19	;	3B	59	C2	7B,34	123,52
HC3	14	20	<	3C	60	FN	7B,53	123,83
HC4	15	21	=	3D	61	C3	7B,42	123,66
NAK	16	22	>	3E	62	FN	7B,43	123,67
SYN	17	23	?	3F	63	C4		
ETB	18	24	@	40	64	SH		
CAN	19	25	A	41	65	IFT		
EM	1A	26	B	42	66	CO		
SUB	1B	27	C	43	67	DEB		
ESC	1C	28	D	44	68	CO		
FS	1D	29	E	45	69	DEC		
GS	1E	30	F	46	70			
RS	1F	31	G	47	71			
US	20	32	H	48	72			

SP	21	33	I	49	73			
!	22	34	J	4A	74			
"	23	35	K	4B	75			
#	24	36	L	4C	76			
$	25	37	M	4D	77			
%	26	38	N	4E	78			
&	27	39	O	4F	79			
'								

字符集 B 中的字符

字符	发送数据		字符	发送数据		字符	发送数据	
	Hex	Decimal		Hex	Decimal		Hex	Decimal
SP	20	32	H	48	72	p	70	112
!	21	33	I	49	73	q	71	113
"	22	34	J	4A	74	r	72	114
#	23	35	K	4B	75	s	73	115
$	24	36	L	4C	76	t	74	116
%	25	37	M	4D	77	u	75	117
&	26	38	N	4E	78	v	76	118
'	27	39	O	4F	79	w	77	119
(	28	40	P	50	80	x	78	120
)	29	41	Q	51	81	y	79	121
*	2A	42	R	52	82	z	7A	122
+	2B	43	S	53	83	{	7B,7B	123,123
,	2C	44	T	54	84	|	7C	124
-	2D	45	U	55	85	}	7D	125
.	2E	46	V	56	86	—	7E	126
/	2F	47	W	57	87	DEL	7F	127
0	30	48	X	58	88	FNC1	7B,31	123,49
1	31	49	Y	59	89	FNC2	7B,32	123,50
2	32	50	Z	5A	90	FNC3	7B,33	123,51
3	33	51	[	5B	91	FNC4	7B,34	123,52
4	34	52	\	5C	92	SHIFT	7B,53	123,83
5	35	53	]	5D	93	CODEA	7B,41	123,65
6	36	54	^	5E	94	CODEC	7B,43	123,67
7	37	55	_	5F	95			
8	38	56	`	60	96			
9	39	57	a	61	97			
:	3A	58	b	62	98			
;	3B	59	c	63	99			
<	3C	60	d	64	100			
=	3D	61	e	65	101			
>	3E	62	f	66	102			
?	3F	63	g	67	103			
@	40	64	H	68	104			
A	41	65	i	69	105			

B	42	66	j	6A	106			
C	43	67	k	6B	107			
D	44	68	l	6C	108			
E	45	69	m	6D	109			
F	46	70	n	6E	110			
G	47	71	o	6F	111			

字符集C中的字符

字符	发送数据		字符	发送数据		字符	发送数据	
	Hex	Decimal		Hex	Decimal		Hex	Decimal
00	00	0	40	28	40	80	50	80
01	01	1	41	29	41	81	51	81
02	02	2	42	2A	42	82	52	82
03	03	3	43	2B	43	83	53	83
04	04	4	44	2C	44	84	54	84
05	05	5	45	2D	45	85	55	85
06	06	6	46	2E	46	86	56	86
07	07	7	47	2F	47	87	57	87
08	08	8	48	30	48	88	58	88
09	09	9	49	31	49	89	59	89
10	0A	10	50	32	50	90	5A	90
11	0B	11	51	33	51	91	5B	91
12	0C	12	52	34	52	92	5C	92
13	0D	13	53	35	53	93	5D	93
14	0E	14	54	36	54	94	5E	94
15	0F	15	55	37	55	95	5F	95
16	10	16	56	38	56	96	60	96
17	11	17	57	39	57	97	61	97
18	12	18	58	3A	58	98	62	98
19	13	19	59	3B	59	99	63	99
20	14	20	60	3C	60	FNC1	7B,31	123,49
21	15	21	61	3D	61	CODE	7B,41	123,65
22	16	22	62	3E	62	A	7B,42	123,66
23	17	23	63	3F	63	CODE		
24	18	24	64	40	64	B		
25	19	25	65	41	65			
26	1A	26	66	42	66			
27	1B	27	67	43	67			
28	1C	28	68	44	68			
29	1D	29	69	45	69			
30	1E	30	70	46	70			
31	1F	31	71	47	71			
32	20	32	72	48	72			
33	21	33	73	49	73			
34	22	34	74	4A	74			

35	23	35	75	4B	75			
36	24	36	76	4C	76			
37	25	37	77	4D	77			
38	26	38	78	4E	78			
39	27	39	79	4F	79			

附录 D. 程序流程