

打印机接口说明

一 概述.....	3
二 接口.....	4
IO.....	4
Write.....	4
Read.....	5
IsOpened.....	6
IOCallback.....	7
OnOpen.....	7
OnOpenFailed.....	8
OnClose.....	9
BTPrinting.....	10
Open.....	10
Listen.....	11
Close.....	12
Write.....	13
Read.....	14
SkipAvailable.....	15
IsOpened.....	16
SetCallback.....	17
BLEPrinting.....	18
Open.....	18
Close.....	19
Write.....	20
Read.....	21
SkipAvailable.....	22
IsOpened.....	23
SetCallback.....	24
NETPrinting.....	25
Open.....	25
Close.....	26
Write.....	27
Read.....	28
SkipAvailable.....	29
IsOpened.....	30
USBPrinting.....	31
Open.....	31
Close.....	32
Write.....	33
Read.....	34
SkipAvailable.....	35
IsOpened.....	36
Pos.....	37

Set.....	37
GetIO.....	38
POS_PrintPicture.....	39
POS_S_TextOut.....	40
POS_S_SetBarcode.....	41
POS_S_SetQRcode.....	43
POS_FeedLine.....	44
POS_S_Align.....	45
POS_SetLineHeight.....	46
POS_Reset.....	47
POS_SetMotionUnit.....	48
POS_S_SetAreaWidth.....	49
POS_CutPaper.....	50
POS_Beep.....	51
POS_KickDrawer.....	52
POS_SetPrintSpeed.....	53
POS_QueryStatus.....	54
POS_QueryStatus.....	55
POS_TicketSucceed.....	57

一 概述

1 PrinterLibs 是 Android 平台下控制 打印机的 jar 库

2 PrinterLibs 有以下几个公共类

A IO 类

包括 IO、BLEPrinting、BTPrinting、NETPrinting、USBPrinting

实现基本的 Open、Close、Write、Read 等功能

IOCallback 提供了 Open 和 Close 的回调接口、便于获取当前的连接状况

B

打印类

包括 Pos、Label1

Pos 类实现了诸如打印文本、打印条码、打印二维码、打印图片等功能

Label1 类实现了标签打印功能，需要标签打印机支持

C

使用方法

Android 代码片段：

```
Pos mPos = new Pos();
BTPrinting mBt = new BTPrinting();
mPos.Set(mBt);
mBt.SetCallback(this);
```

之后启用异步调用：

```
public static class TaskOpen implements Runnable
{
    BTPrinting bt = null;
    String address = null;

    public TaskOpen(BTPrinting bt, String address)
    {
        this.bt = bt;
        this.address = address;
    }

    @Override
    public void run() {
        // TODO Auto-generated method stub
        bt.Open(address);
    }
}
```

连接成功之后，调用相应的函数即可打印。

二 接口

IO

Write

Syntax

```
public int Write(byte[] buffer, int offset, int count)
```

Parameters

buffer

发送缓冲区

offset

从指定偏移开始发送数据

count

要发送的字节数

Return value

如果写入成功，返回成功写入的字节数、如果写入失败，返回-1

Remarks

IO 类的 Write 函数为空实现，始终返回-1

Read

Syntax

```
public int Read(byte[] buffer, int offset, int count, int timeout)
```

Parameters

buffer

接收缓冲区

offset

从指定偏移开始存放收到的数据

count

要接收的字节数

timeout

超时毫秒时间

Return value

如果读取成功，返回成功读入的字节数、如果读取失败，返回-1。

Remarks

IO 类的 **Read** 函数为空实现，始终返回-1

IsOpened

Syntax

```
public boolean IsOpened()
```

Parameters

Return value

如果以连接到打印机，返回 **true**、否则，返回 **false**

Remarks

IO 类的 **IsOpened** 函数为空实现，始终返回 **false**

I OCall Back

处理底层连接的 4 个类:

BLEPrinting BTPrinting NETPrinting USBPrinting

Open 成功时, 会调用 OnOpen

Open 失败时, 会调用 OnOpenFailed

Close 或异常断开时, 会调用 OnClose

OnOpen

连接成功之后, 会调用 OnOpen

Syntax

```
void OnOpen()
```

Parameters

Return value

Remarks

OnOpenFailed

连接失败，会调用 OnOpenFailed

Syntax

```
void OnOpenFailed()
```

Parameters

Return value

Remarks

OnClose

连接断开（主动断开或异常中断），会调用 OnClose

Syntax

```
void OnClose()
```

Parameters

Return value

Remarks

BT Printing

蓝牙 2.0 连接、读写封装

Open

连接指定蓝牙打印机

Syntax

```
public boolean Open(String BTAddress, Context mContext)
```

Parameters

BTAddress

蓝牙打印机地址：形如 00:11:22:33:44:55

mContext

Application Context

Return value

连接成功，返回 **true**、否则，返回 **false**。

Remarks

连接成功之后，会调用回调接口 **OnOpen**，连接失败会调用 **OnOpenFailed**

Listen

连接 2.0 蓝牙打印机（作为主模式，等待打印机主动上连）

Syntax

```
public boolean Listen(String BTAddress, int timeout, Context mContext)
```

Parameters

BTAddress

蓝牙打印机地址：形如 00:11:22:33:44:55，暂不使用

timeout

等待超时毫秒时间

mContext

Application Context

Return value

连接成功，返回 **true**、否则，返回 **false**。

Remarks

连接成功之后，会调用回调接口 **OnOpen**，连接失败会调用 **OnOpenFailed**

Close

关闭连接

Syntax

```
public void Close()
```

Parameters

Return value

Remarks

关闭连接，会调用回调接口 **OnClose**，重复 **Close** 不会多次调用回调。

Write

通过蓝牙写入数据

Syntax

```
public int Write(byte[] buffer, int offset, int count)
```

Parameters

buffer

发送缓冲区

offset

从指定偏移开始发送数据

count

要发送的字节数

Return value

如果写入成功，返回成功写入的字节数、如果写入失败，返回-1

Remarks

Read

读数据

Syntax

```
public int Read(byte[] buffer, int offset, int count, int timeout)
```

Parameters

buffer

接收缓冲区

offset

从指定偏移开始存放收到的数据

count

要接收的字节数

timeout

超时毫秒时间

Return value

如果读取成功，返回成功读入的字节数、如果读取失败，返回-1。

Remarks

SkipAvailable

忽略缓冲区中的数据

Syntax

```
public void SkipAvailable()
```

Parameters

Return value

Remarks

IsOpened

是否已连接

Syntax

```
public boolean IsOpened()
```

Parameters

Return value

返回 **true**，表示已经连接、返回 **false**，表示未连接。

Remarks

IsOpened 函数是建立在心跳的基础上，并不能实时获取连接状态。

如果打印机突然关机，**IsOpened** 可能需要几秒钟，才能返回正确的结果。

如果想确定打印机是否已连接，可以使用 POS 系列函数中的 **RTQueryStatus**。

SetCallback

设置回调接口

Syntax

```
public void SetCallback(I0Callback callback)
```

Parameters

callback

回调接口，只有设置了该项，在连接成功或连接断开的时候，才会有回调。

Return value

Remarks

BLE Printing

蓝牙 4.0 连接、读写封装

Open

连接指定蓝牙打印机

Syntax

```
public boolean Open(String BTAddress)
```

Parameters

BTAddress

蓝牙打印机地址：形如 00:11:22:33:44:55

Return value

连接成功，返回 **true**、否则，返回 **false**。

Remarks

连接成功之后，会调用回调接口 **OnOpen**，连接失败会调用 **OnOpenFailed**

Close

关闭连接

Syntax

```
public void Close()
```

Parameters

Return value

Remarks

关闭连接，会调用回调接口 **OnClose**，重复 **Close** 不会多次调用回调。

Write

通过蓝牙写入数据

Syntax

```
public int Write(byte[] buffer, int offset, int count)
```

Parameters

buffer

发送缓冲区

offset

从指定偏移开始发送数据

count

要发送的字节数

Return value

如果写入成功，返回成功写入的字节数、如果写入失败，返回-1

Remarks

蓝牙 4.0 由于标准限制，速度会比 2.0 慢不少。

Read

读数据

Syntax

```
public int Read(byte[] buffer, int offset, int count, int timeout)
```

Parameters

buffer

接收缓冲区

offset

从指定偏移开始存放收到的数据

count

要接收的字节数

timeout

超时毫秒时间

Return value

如果读取成功，返回成功读入的字节数、如果读取失败，返回-1。

Remarks

SkipAvailable

忽略缓冲区中的数据

Syntax

```
public void SkipAvailable()
```

Parameters

Return value

Remarks

IsOpened

是否已连接

Syntax

```
public boolean IsOpened()
```

Parameters

Return value

返回 **true**，表示已经连接、返回 **false**，表示未连接。

Remarks

IsOpened 可能会有延时。

如果想确定打印机是否已连接，可以使用 POS 系列函数中的 **RTQueryStatus**。

SetCallback

设置回调接口

Syntax

```
public void SetCallback(I0Callback callback)
```

Parameters

callback

回调接口，只有设置了该项，在连接成功或连接断开的时候，才会有回调。

Return value

Remarks

NETPrinting

WIFI 底层连接、读写封装

Open

连接指定网络打印机

Syntax

```
public boolean Open(String IPAddress, int PortNumber)
```

Parameters

IPAddress

打印机 IP 地址：可以在打印机自检页中获取，打印机默认 IP：192.168.1.87

PortNumber

打印机端口号：固定为 9100

Return value

连接成功，返回 **true**、否则，返回 **false**。

Remarks

连接成功之后，会调用回调接口 **OnOpen**，连接失败会调用 **OnOpenFailed**

Close

关闭连接

Syntax

```
public void Close()
```

Parameters

Return value

Remarks

关闭连接，会调用回调接口 **OnClose**，重复 **Close** 不会多次调用回调。

Write

通过网口写入数据

Syntax

```
public int Write(byte[] buffer, int offset, int count)
```

Parameters

buffer

发送缓冲区

offset

从指定偏移开始发送数据

count

要发送的字节数

Return value

如果写入成功，返回成功写入的字节数、如果写入失败，返回-1

Remarks

如果无线路由器信号不好，或网络环境不佳，可能会造成卡顿。正常情况下，打印巨量数据都不会有问题。

Read

读数据

Syntax

```
public int Read(byte[] buffer, int offset, int count, int timeout)
```

Parameters

buffer

接收缓冲区

offset

从指定偏移开始存放收到的数据

count

要接收的字节数

timeout

超时毫秒时间

Return value

如果读取成功，返回成功读入的字节数、如果读取失败，返回-1。

Remarks

SkipAvailable

忽略缓冲区中的数据

Syntax

```
public void SkipAvailable()
```

Parameters

Return value

Remarks

IsOpened

是否已连接

Syntax

```
public boolean IsOpened()
```

Parameters

Return value

返回 **true**，表示已经连接、返回 **false**，表示未连接。

Remarks

IsOpened 函数是建立在心跳的基础上，并不能实时获取连接状态。

如果打印机突然关机，**IsOpened** 可能需要几秒钟，才能返回正确的结果。

如果想确定打印机是否已连接，可以使用 POS 系列函数中的 **RTQueryStatus**。

USB Printing

USB 底层连接、读写封装

Open

连接指定 USB 打印机

Syntax

```
public boolean Open(UsbManager manager, UsbDevice device)
```

Parameters

manager

UsbManager

使用(UsbManager) getSystemService(Context.USB_SERVICE) 获得

device

UsbDevice

通过枚举 USB 设备获得 UsbDevice (mUsbManager.getDeviceList())

Return value

连接成功，返回 true、否则，返回 false。

Remarks

连接成功之后，会调用回调接口 OnOpen，连接失败会调用 OnOpenFailed

Close

关闭连接

Syntax

```
public void Close()
```

Parameters

Return value

Remarks

关闭连接，会调用回调接口 **OnClose**，重复 **Close** 不会多次调用回调。

Write

通过 USB 写入数据

Syntax

```
public int Write(byte[] buffer, int offset, int count)
```

Parameters

buffer

发送缓冲区

offset

从指定偏移开始发送数据

count

要发送的字节数

Return value

如果写入成功，返回成功写入的字节数、如果写入失败，返回-1

Remarks

USB 发送数据速度最快最稳定，建议使用 USB 进行打印。

Read

读数据

Syntax

```
public int Read(byte[] buffer, int offset, int count, int timeout)
```

Parameters

buffer

接收缓冲区

offset

从指定偏移开始存放收到的数据

count

要接收的字节数

timeout

超时毫秒时间

Return value

如果读取成功，返回成功读入的字节数、如果读取失败，返回-1。

Remarks

SkipAvailable

忽略缓冲区中的数据

Syntax

```
public void SkipAvailable()
```

Parameters

Return value

Remarks

IsOpened

是否已连接

Syntax

```
public boolean IsOpened()
```

Parameters

Return value

返回 **true**，表示已经连接、返回 **false**，表示未连接。

Remarks

IsOpened 函数是建立在心跳的基础上，并不能实时获取连接状态。

如果打印机突然关机，**IsOpened** 可能需要几秒钟，才能返回正确的结果。

如果想确定打印机是否已连接，可以使用 POS 系列函数中的 **RTQueryStatus**。

Pos

POS 通过持有一个 IO 对象来与打印机通信
使用 Set(IO)即可设置 POS 持有的 IO 对象
后续一系列指令，都是通过指定 IO 传达

Set

指定 IO 对象

Syntax

```
public void Set(IO io)
```

Parameters

io

需要使用的 IO 对象

Return value

Remarks

调用该函数，将一个底层读写类绑定到 Pos 这个上层逻辑处理类。

GetIO

获取当前的 IO 对象

Syntax

```
public IO GetIO()()
```

Parameters

Return value

当前 IO 对象

Remarks

POS_PrintPicture

打印图片

Syntax

```
public void POS_PrintPicture(Bitmap mBitmap, int nWidth, int nBinaryAlgorithm, int nCompressMethod)
```

Parameters

mBitmap

需要打印的位图

nWidth

需要打印的宽度

如果 nWidth 和 Bitmap 的宽度不一致，会等比例缩放到 nWidth 宽

2 寸打印机（58mm 打印机）最大宽度不超过 384 点

3 寸打印机（80mm 打印机）最大宽度不超过 576 点

nBinaryAlgorithm

二值化算法

0 使用抖动算法，对彩色图片有较好的效果。

1 使用平均阈值算法，对文本类图片有较好的效果

nCompressMethod

压缩算法

0 不使用压缩算法

1 使用压缩算法

Return value

Remarks

打印图片使用抖动算法，对彩色图片打印效果较好，打印的图片有灰阶的效果。

P0S_S_TextOut

按照一定的格式打印字符串

Syntax

```
public void P0S_S_TextOut(String pszString, int nOrgx, int nWidthTimes,  
                           int nHeightTimes, int nFontType, int nFontStyle)
```

Parameters

pszString

需要打印的字符串

nOrgx

指定 X 方向（水平）的起始点位置离左边界的点数。
2 寸打印机一行 384 点，3 寸打印机一行 576 点。

nWidthTimes

指定字符的宽度方向上的放大倍数。
可以为 0 到 1。

nHeightTimes

指定字符高度方向上的放大倍数。
可以为 0 到 1。

nFontType

指定字符的字体类型。
(0x00 标准 ASCII 12x24)
(0x01 压缩 ASCII 9x17)

nFontStyle

指定字符的字体风格。可以为以下列表中的一个或若干个。

- (0x00 正常)
- (0x08 加粗)
- (0x80 1 点粗的下划线)
- (0x100 2 点粗的下划线)
- (0x200 倒置、只在行首有效)
- (0x400 反显、黑底白字)
- (0x1000 每个字符顺时针旋转 90 度)

Return value

Remarks

POS_S_SetBarcode

打印条码

Syntax

```
public void POS_S_SetBarcode(String strCodedata, int nOrgx, int nType,
                             int nWidthX, int nHeight, int nHriFontType,
                             int nHriFontPosition)
```

Parameters

strCodedata

需要打印的条码的字符串

部分条码有格式要求，请按照条码规范打印条码

nOrgx

指定 X 方向（水平）的起始点位置离左边界的点数。

2 寸打印机一行 384 点，3 寸打印机一行 576 点。

nType

指定条码的类型。

可以为以下列表中所列值之一。

Value	Meaning
0x41	UPC-A
0x42	UPC-C
0x43	JAN13(EAN13)
0x44	JAN8(EAN8)
0x45	CODE39
0x46	ITF
0x47	CODEBAR
0x48	CODE93
0x49	CODE 128

nWidthX

指定条码的基本元素宽度

范围：[2,6]

nHeight

指定条码的高度点数

可以为 1 到 255

nHri FontType

指定 HRI（Human Readable Interpretation）字符的字体类型。
可以为以下列表中所列值之一。

Value	Meaning
0x00	标准 ASCII
0x01	压缩 ASCII

nHri FontPosition

指定 HRI（Human Readable Interpretation）字符的位置。
可以为以下列表中所列值之一。

Value	Meaning
0x00	不打印
0x01	只在条码上方打印
0x02	只在条码下方打印
0x03	条码上、下方都打印

Return value**Remarks**

如果条码太宽超出打印机最大打印宽度，则条码不会被打印。
如果条码格式有误，条码也不会打印。

POS_S_SetQRcode

打印二维码

Syntax

```
public void POS_S_SetQRcode(String strCodedata, int nWidthX,  
int nVersion, int nErrorCorrectionLevel)
```

Parameters

strCodedata

二维码字符串

nWidthX

二维码每个模块的单元宽度，[1, 16]

适当设置模块宽度，可以使得二维码看起来更漂亮

nVersion

二维码版本大小，该值和二维码大小相关。[0, 16]

设置为 0 自动计算二维码版本大小。

如果希望二维码大小固定不变，请设置该值为合适的值。

nErrorCorrectionLevel

纠错等级。

[1, 4]

Return value

Remarks

POS_FeedLine

走纸一行

Syntax

```
public void POS_FeedLine()
```

Parameters

Return value

Remarks

POS_S_Align

设置对齐方式

Syntax

```
public void POS_S_Align(int align)
```

Parameters

Align

设置对其方式

(0 左对齐)

(1 居中对齐)

(2 右对齐)

Return value

Remarks

POS_SetLineHeight

设置行高

Syntax

```
public void POS_SetLineHeight(int nHeight)
```

Parameters

nHeight

行高 (0, 255]

Return value

Remarks

POS_Reset

复位打印机（软件复位）

Syntax

```
public void POS_Reset()
```

Parameters

Return value

Remarks

POS_SetMotionUnit

设置打印机的移动单位

Syntax

```
public void POS_SetMotionUnit(int nHorizontalMU, int nVerticalMU)
```

Parameters

nHorizontalMU

把水平方向上的移动单位设置为 25.4 / nHorizontalMU 毫米。

nVerticalMU

把垂直方向上的移动单位设置为 25.4 / nVerticalMU 毫米。

Return value

Remarks

POS_S_SetAreaWidth

设置标准模式下的打印区域宽度

Syntax

```
public void POS_S_SetAreaWidth(int nWidth)
```

Parameters

nWidth

指定打印区域的宽度

Return value

Remarks

POS_CutPaper

切纸

Syntax

```
public void POS_CutPaper()
```

Parameters

Return value

Remarks

只对带切刀的机器有效

POS_Beep

蜂鸣器鸣叫

Syntax

```
public void POS_Beep(int nBeepCount, int nBeepMillis)
```

Parameters

nBeepCount

鸣叫次数

nBeepMillis

每次鸣叫的时间 = 100 * nBeepMillis ms

Return value

Remarks

POS_KickDrawer

打开钱箱

Syntax

```
public void POS_KickDrawer(int nDrawerIndex, int nPulseTime)
```

Parameters

nDrawerIndex

0 表示：脉冲发送到钱箱输出引脚 2

1 表示：脉冲发送到钱箱输出引脚 5

nPulseTime

脉冲时间

高电平时间: $nPulseTime * 2ms$

低电平时间: $nPulseTime * 2ms$

Return value

Remarks

POS_SetPrintSpeed

设置打印速度 注：如果打印速度大于发送速度，打印会有卡顿感。

Syntax

```
public void POS_SetPrintSpeed(int nSpeed)
```

Parameters

nSpeed

打印速度 (mm/s)

Return value

Remarks

将打印速度设置为数据发送速度，可以是打印效果达到最好。

可以通过打印一张单据，测量单据的长度和所用时间，用长度/时间，即可。

P0S_QueryStatus

查询状态

打印机忙时，该命令会一直阻塞
返回的状态保存在 `status` 中

Syntax

```
public boolean P0S_QueryStatus(byte[] status, int timeout, int MaxRetry)
```

Parameters

`status`

`status = new byte[1]` 该值目前无意义

`timeout`

单次查询状态的超时毫秒时间

`MaxRetry`

失败重试次数

Return value

返回 `true`，表明打印机状态 **OK**。否则，打印机未联机或打印机正忙。

Remarks

POS_QueryStatus

实时状态查询

无论打印机处于何种状态，只要打印机收到该命令就立刻回送状态
返回的状态保存在 `status` 中

Syntax

```
public boolean POS_RTQueryStatus(byte[] status, int type, int timeout,
    int MaxRetry)
```

Parameters

`status`

```
status = new byte[1]
```

`timeout`

`type` 可取值 [1, 4]

1: 打印机状态

位	0/1	十六进制码	十进制码	功能
0	0	00	0	固定为 0
1	1	02	2	固定为 1
2	0	00	0	一个或两个钱箱打开 (没有钱箱的机器该位固定为零)
	1	04	4	两个钱箱都关闭
3	0	00	0	联机
	1	08	8	脱机
4	1	10	16	固定为 1
5, 6		--	--	未定义
7	0	00	00	纸已撕走
	1	80	96	纸未撕走

2: 传送脱机状态

位	0/1	十六进制码	十进制码	功能
0	0	00	0	固定为 0
1	1	02	2	固定为 1
2	0	00	0	上盖关
	1	04	4	上盖开
3	0	00	0	未按走纸键

	1	08	8	按下走纸键
4	1	10	16	固定为 1
5	0	00	0	打印机不缺纸
	1	20	32	打印机缺纸
6	0	00	00	没有出错情况
	1	40	64	有错误情况
7	0	00	0	固定为 0

3: 传送错误状态

位	0/	十六进制码	十进制码	功能
	1			
0	0	00	0	固定为 0
1	1	02	2	固定为 1
2		--	--	未定义
3	0	00	0	切刀无错误
	1	08	8	切刀有错误
4	1	10	16	固定为 1
5	0	00	0	无不可恢复错误
	1	20	32	有不可恢复错误
6	0	00	00	打印头温度和电压正常
	1	40	64	打印头温度或电压超出范围
7	0	00	0	固定为 0

4: 传送纸传感器状态

位	0/	十六进制码	十进制码	功能
	1			
0	0	00	0	固定为 0
1	1	02	2	固定为 1
2, 3	0	00	0	有纸
	1	0C	12	纸将近
4	1	10	16	固定为 1
5,	0	00	0	有纸
6	1	60	96	纸尽
7	0	00	0	固定为 0

MaxRetry

失败重试次数

Return value

返回 true, 表明打印机通讯正常, 查询的状态保存在 status 中。

Remarks

POS_TicketSucceed

发送单据查询命令

Syntax

```
public boolean POS_TicketSucceed(int dwSendIndex, int timeout)
```

Parameters

dwSendIndex

单据索引:

可以从 1 开始依次递增, 目前并无实际意义

timeout

等待单据打印完成的超时 ms 时间

Return value

单据打印完成, 并且没有因为缺纸而中断, 则返回 **true**。

否则, 没有查到状态, 或返回因为缺纸或其他错误导致打印中断, 则返回 **false**。

Remarks

为了保证单据打印的可靠性, 请每批次打印任务完成之后, 调用一次该函数确认单据打印结果。