SunmiPrinter 开发者文档

目录

简介

商米的 V1、P1、T1 内置了热敏打印机，允许 App 直接打印热敏小票出来。商米产品的内置打印机一共有 2 种规格：

- 80mm 宽，带切刀。兼容 58mm， T1 搭载了这种打印机。
- 58mm 宽，不带切刀。V1、P1 搭载了这种打印机。

App 开发者可以使用 3 种方式调用内置热敏打印机：

1.通过 AIDL 连接打印机

2.通过蓝牙连接打印机

3.通过 JS 桥调用打印机

下面以 V1 设备为例，介绍两种调用方式，该方案也适用于 P1、T1 设备。

1 连接打印服务

1.1 方式 1 ： AIDL

1.1.1. AIDL 简介

AIDL 是 Android Interface Definition language 的缩写，它是一种 Android 内部进程通信接口的描述语言，通过它我们可以定义进程间的通信接口。AIDL 提供封装好的常用打印指令，方便开发者快速接入 Sunmi 打印机，同时也支持 Sunmi《ESC / POS》指令集。

1.1.2. AIDL 使用

建立连接可分以下 5 步骤：

1.在项目中添加资源文件中附带的AIDL文件（部分机型还包含java文件）。

2.在控制打印的代码类中实现ServiceConnection。

3.调用ApplicationContext.bindService()，并在ServiceConnection实现中进行传递。

注意：bindservice是非阻塞调用，意味着调用完成后并没有立即绑定成功，必须以serviceConnected为准。

4.在ServiceConnection.onServiceConnected()实现中，你会接收一个IBinder实例(被调用的Service)。调用 IWoyouService.Stub.asInterface(service)将参数转换为IWoyouService类型。

5.现在就可以调用IWoyouService接口中定义的各种方法进行打印了。

绑定服务事例
private ServiceConnection connService = **new** ServiceConnection() { @Override

```
        public void onServiceDisconnected(ComponentName name) {
            Toast.makeText(PrinterTestDemoAct.this, "service disconnected",  Toast.LENGTH_LONG).show();
            setButtonEnable(false);
            woyouService = null;
            try {
                Thread.sleep(2000);
            } catch (InterruptedException e) {
                // TODO Auto-generated catch block
                e.printStackTrace();
            }
            Binding();
        }
        @Override
        public void onServiceConnected(ComponentName name, IBinder service) {
            woyouService = IWoyouService.Stub.asInterface(service);
            setButtonEnable(true);
            try {
                serviceVersion = woyouService.getServiceVersion();
                info.setText("service version :" + serviceVersion + "\n");
            } catch (RemoteException e) {
                // TODO Auto-generated catch block
                e.printStackTrace();
            }
        }
    };
private void Binding(){
        Intent intent=new Intent();
        intent.setPackage("woyou.aidlservice.jiuiv5");
        intent.setAction("woyou.aidlservice.jiuiv5.IWoyouService");
        startService(intent);
        bindService(intent, connService,Context.BIND_AUTO_CREATE);
}
```


1.1.3. AIDL 接口说明

通过 AIDL 方式与打印服务建立连接后，就可调用 IWoyouService 封装好的常用打印指令接口（仅 aidl 连接方式才可调用 IWoyouService 接口函数）。

函数原型	printerInit()
功能说明	打印机初始化
参数说明	无
返回值	void
补充说明	重置打印机的逻辑程序，但不清空缓存区数据，因此未完成的打印作业将在重置后继续

函数原型	printerSelfChecking()
功能说明	打印机自检
参数说明	无
返回值	void
补充说明	无

函数原型	int getPrinterMode()
功能说明	获取打印机模式
参数说明	无
返回值	0:普通模式 1:黑标模式
补充说明	仅支持 T1 设备

函数原型	int getPrinterBBMDistance()
功能说明	获取黑标模式打印机自动走纸距离(点行)
参数说明	无
返回值	走纸距离
补充说明	仅支持 T1 设备

函数原型	getPrinterSerialNo()
功能说明	获取打印机板序列号
参数说明	无
返回值	String 打印机版序列号
补充说明	无

函数原型	getPrinterVersion()
功能说明	获取打印机固件版本号
参数说明	无
返回值	String 打印机固件版本号
补充说明	无

函数原型	getPrintedLength(in ICallback callback)
功能说明	获取打印头打印长度
参数说明	Callback 结果回调
返回值	void
补充说明	无

函数原型	lineWrap(int n, in ICallback callback)
功能说明	打印机走纸
参数说明	Callback 结果回调
返回值	void
补充说明	强制换行，结束之前的打印内容后走纸 n 行

函数原型	sendRAWData(in byte[] data, in ICallback callback)
功能说明	打印 esc/pos 格式指令
参数说明	Data 指令 Callback 结果回调
返回值	void
补充说明	无

函数原型	setAlignment (int alignment, in ICallback callback)
功能说明	设置对齐模式
参数说明	alignment 对齐方式 0--居左 , 1--居中, 2--居右 Callback 结果回调
返回值	void
补充说明	全局方法，对之后打印有影响，初始化能取消设置

函数原型	setFontName(String typeface, in ICallback callback)
功能说明	设置打印字体
参数说明	typeface 字体名称 Callback 结果回调
返回值	void
补充说明	全局方法，对之后打印有影响，初始化能取消设置 目前只支持一种字体"gh"，gh 是一种等宽中文字体

函数原型	setFontSize(float fontsize, in ICallback callback)
功能说明	设置字体大小
参数说明	fontsize 字体大小 Callback 结果回调
返回值	void
补充说明	全局方法，对之后打印有影响，初始化能取消设置 字体大小是超出标准国际指令的打印方式，调整字体大小会影响字符宽度，每行字符数量也会随之改变，因此按等宽字体形成的排版可能会错乱

函数原型	printText(String text, in ICallback callback)
功能说明	打印文字
参数说明	text 打印内容 Callback 结果回调
返回值	void
补充说明	文字宽度满一行自动换行排版，不满一整行不打印除非强制换行

函数原型	**printTextWithFont(String text, String typeface, float fontsize, in ICallback callback)**
功能说明	打印指定字体，大小的文本
参数说明	**text** 要打印文字 **typeface** 字体名称 **fontsize** 字体大小 Callback 结果回调
返回值	**void**
补充说明	字体设置只对本次有效

函数原型	**printColumnsText(in String[] colsTextArr, in int[] colsWidthArr, in int[] colsAlign, in ICallback callback)**
功能说明	打印表格的一行
参数说明	**colsTextArr** 各列文本字符串数组 **colsWidthArr** 各列宽度数组，以英文字符计算，每个中文字符占两个英文字符，每个宽度大于 **0** **colsAlign** 各列对齐方式：**0** 居左, **1** 居中, **2** 居右 Callback 结果回调
返回值	**void**
补充说明	三个参数的数组长度应该一致，如果 **colsText[i]**的宽度大于 **colsWidth[i]**，则文本换行

函数原型	**printColumnsString(in String[] colsTextArr, in int[] colsWidthArr, in int[] colsAlign, in ICallback callback)**
功能说明	打印表格的一行，可以指定列宽、对齐方式
参数说明	**colsTextArr** 各列文本字符串数组 **colsWidthArr** 各列宽度权重即各列所占比例 **colsAlign** 各列对齐方式(**0** 居左, **1** 居中, **2** 居右) Callback 结果回调
返回值	**void**
补充说明	无

函数原型	printBitmap(in Bitmap bitmap, in ICallback callback)
功能说明	打印图片
参数说明	bitmap 图片 bitmap 对象 Callback 结果回调
返回值	void
补充说明	最大宽度 384 像素，超过无法打印并且回调 callback 异常函数

函数原型	printBarCode(String data, int symbology, int height, int width, int textposition, in ICallback callback)
功能说明	打印一维条码
参数说明	data 条码数据 symbology 条码类型 * 0 -- UPC-A， * 1 -- UPC-E， * 2 -- JAN13(EAN13)， * 3 -- JAN8(EAN8)， * 4 -- CODE39， * 5 -- ITF， * 6 -- CODABAR， * 7 -- CODE93， * 8 -- CODE128 height 条码高度，取值 1 到 255，默认 162 width 条码宽度，取值 2 至 6，默认 2 textposition 文字位置 0--不打印文字，1--文字在条码上方，2--文字在条码下方，3--条码上下方均打印 Callback 结果回调
返回值	void
补充说明	每种编码的最多打印内容

函数原型	printQRCode(String data, int modulesize, int errorlevel, in

	ICallback callback)
功能说明	打印二维条码
参数说明	data 二维码数据 modulesize 二维码块大小，单位:点，取值 1 至 16 errorlevel 二维码纠错等级(0 至 3)， 0 -- 纠错级别 L (7%)， 1 -- 纠错级别 M (15%)， 2 -- 纠错级别 Q (25%)， 3 -- 纠错级别 H (30%) Callback 结果回调
返回值	void
补充说明	无

函数原型	printOriginalText (String text, in ICallback callback)
功能说明	打印矢量文字
参数说明	text 打印内容 Callback 结果回调
返回值	void
补充说明	打印文字，文字宽度满一行自动换行排版，不满一整行不打印除非强制换行 文字按矢量文字宽度原样输出，即每个字符不等宽

函数原型	commitPrint(in TransBean[] transbean, in ICallback callback)
功能说明	lib 包事务打印专用接口
参数说明	transbean 打印任务列表 Callback 结果回调
返回值	void
补充说明	使用 lib 的用户调用此接口开始事物打印事物打印 需打印服务 2.0.8 以上

函数原型	commitPrinterBuffer()
功能说明	打印缓冲区内容

参数说明	无
返回值	void
补充说明	事物打印 需打印服务 2.0.8 以上

函数原型	enterPrinterBuffer(in boolean clean)
功能说明	进入缓冲模式
参数说明	clean 是否清除缓冲区内容 true 清除 false 不清除
返回值	void
补充说明	事物打印 所有打印调用将缓存，调用 commitPrinterBuffe()后打印事物打印 需打印服务 2.0.8 以上

函数原型	exitPrinterBuffer(in boolean commit)
功能说明	退出缓冲模式
参数说明	commit 是否打印出缓冲区内容 true 打印 false 不打印事物打印
返回值	void
补充说明	事物打印事物打印 需打印服务 2.0.8 以上

函数原型	commitPrinterBufferWithCallback(in ICallback callback)
功能说明	带反馈打印缓冲区内容
参数说明	Callback 结果回调
返回值	void
补充说明	仅支持机型 P1，V1S

函数原型	exitPrinterBufferWithCallback(boolean commit, ICallback callback)
功能说明	带反馈打印缓冲区内容
参数说明	Commit 是否提交缓冲区内容 true 提交 false 不提交 Callback 结果回调
返回值	void
补充说明	仅支持机型 P1，V1S

注：当进入缓冲（事务）打印后，提交打印成功将返回成功结果，但遇到打印机异常如缺纸、过热等，将会丢掉本次提交事务中所有指令任务，同时反馈异常，即当一单任务执行前或执行中打印机异常，则此单不会打出；

注意：

当打印机处于异常状态仍连续使用缓冲（事务）打印（发送了多单），且不判断每单的反馈，打印机恢复正常后将不打出第一单！

当指令打印和缓冲（事务）打印交替使用时，如果打印机异常，不会清除指令打印的内容！

函数原型	cutPaper(in ICallback callback)
功能说明	切纸
参数说明	Callback 结果回调
返回值	void
补充说明	无

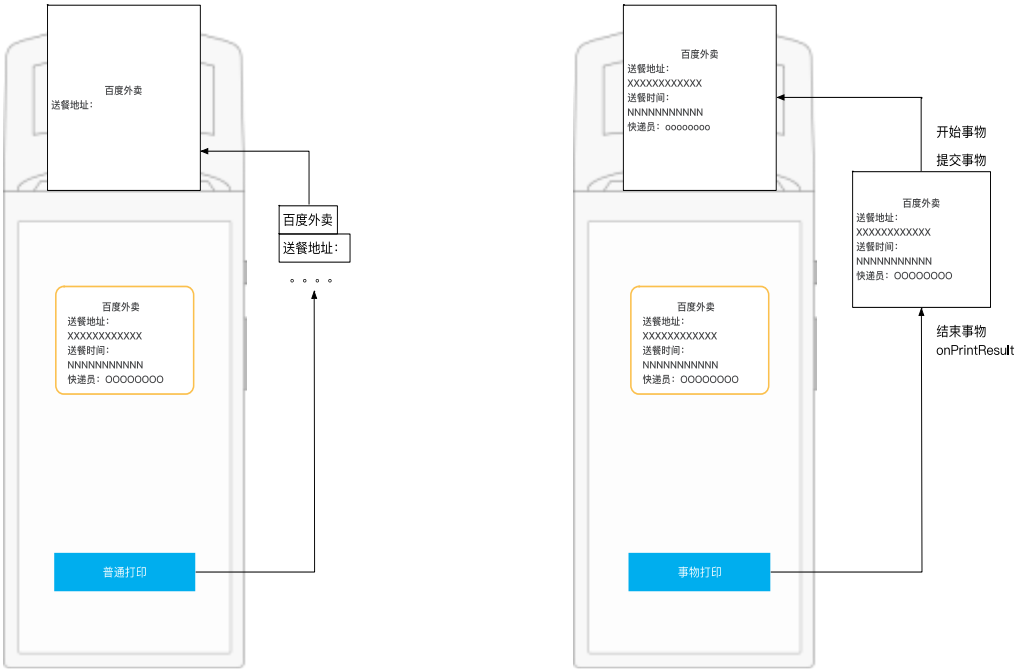
函数原型	getCutPaperTimes()
功能说明	获取切刀次数
参数说明	无
返回值	int 切刀次数
补充说明	无

函数原型	openDrawer(in ICallback callback)
功能说明	打开钱柜
参数说明	Callback 结果回调
返回值	void
补充说明	无

函数原型	getOpenDrawerTimes()
功能说明	取钱柜累计打开次数
参数说明	无
返回值	void
补充说明	无

1.1.3.1. 事务打印说明

事务打印模式适用于需要控制打印内容并得到打印结果反馈(是否打印出小票)的需求，此模式相当于建立一个事务队列缓冲区，当开发者进入事务打印模式，将开启一个事务队列，可以向其中增加打印方法，此时打印机不会立刻打印内容，当提交事务后，打印机才会依次执行队列中的任务，执行结束将获得此次事务的结果反馈。



事务打印模式包括以下方法：

仅支持机型 P1，V1S

函数原型	void enterPrinterBuffer(in boolean clean)
功能说明	启用并进入事务打印模式： 启用并进入事务打印模式： 在这个模式下不会立刻打印数据直到提交事物或退出提交事物
参数说明	in boolean clean true：清除上一次事务打印未提交的内容 false :不清除上一次事务打印未提交的内容，下次提交将包含上次的内容
返回值	void

补充说明	**无**

函数原型	void commitPrinterBuffer()
功能说明	提交事务： 将事务队列中的所有内容提交并打印,之后仍然处于事务打印模式
参数说明	**无**
返回值	void
补充说明	**无**

函数原型	void commitPrinterBufferWithCallback(in ICallback callback)
功能说明	提交事务： 将事务队列中的所有内容提交并打印同时回调接口 callback 中回调 onPrintResult 方法,之后仍然处于事务打印模式，onPrintResult 返回打印结果
参数说明	Callback **结果回调**
返回值	void
补充说明	**无**

函数原型	void exitPrinterBuffer(in boolean commit)
功能说明	退出事务打印
参数说明	in boolean commit true：会打印出事务队列中的所有内容 false :不会打印事务队列中的内容，此内容将保存直到下次提交
返回值	void
补充说明	**无**

函数原型	void exitPrinterBufferWithCallback(in boolean commit, in ICallback callback)
功能说明	退出事务打印：

	退出事务打印同时对提交打印后的结果通过回调接口 callback 的 onPrintResult 方法回调
参数说明	in boolean commit true ：会打印出事务队列中的所有内容 false ：不会打印事务队列中的内容，此内容将保存直到下次提交
返回值	Callback **结果回调**
补充说明	**无**

事务打印结果反馈回调接口：

code：

0 打印成功 string null

1 打印失败 string “woyou.aidlservice.jiuv5.ERROR_ACTION”

伪代码示例如下：

```
----------------------------------------
enterPrinterBuffer(true)
printText(/*something*/)
printBitmap(/*bitmap resource*/)
... ...
commitPrinterBuffer()//exitPrinterBuffer(true)//
commitPrinterBufferWithCallback(callback)/
exitPrinterBufferWithCallback(true, callback)//在 callbackOnPrintResult 中回掉打印结果
----------------------------------------
```

1.2 方式 2 ： 虚拟蓝牙

1.2.1 虚拟蓝牙简介

在V1的蓝牙设备列表中可以看到一个已经配对，且永远存在的蓝牙设备“ InnerPrinter”，这是由操作系统虚拟出来的打印机设备，实际并不存在。虚拟蓝牙支持Sunmi《esc/pos》指令。其中有部分特殊的指令属于sunmi自定义指令，如：

开钱箱指令	byte [5] ：0x10 0x14 0x00 0x00 0x00
切刀指令 全部切割	byte [4] ：0x1d 0x56 0x42 0x00
切刀指令 切割（左边留一点不切）	byte [4] ：0x1d 0x56 0x41 0x00

1.2.2 虚拟蓝牙使用

1.与该蓝牙设备建立连接

2.将指令和文本内容拼接转码为Bytes

3.发送给InnerPrinter

4.底层打印服务驱动打印设备完成打印

注：BluetoothUtil 一个蓝牙工具类，用于连接虚拟蓝牙设备InnerPrinter。

工具类BluetoothUtil， 标准的蓝牙连接工具类

```
public class BluetoothUtil {

    private static final UUID PRINTER_UUID = UUID.fromString("00001101-0000-1000-8000-00805F9B34FB");

    private static final String Innerprinter_Address = "00:11:22:33:44:55";

    public static BluetoothAdapter getBTAdapter() {

    return BluetoothAdapter.getDefaultAdapter();

    }


    public static BluetoothDevice getDevice(BluetoothAdapter bluetoothAdapter) {

        BluetoothDevice innerprinter_device = null;

        Set<BluetoothDevice> devices = bluetoothAdapter.getBondedDevices();

        for (BluetoothDevice device : devices) {

            if (device.getAddress().equals(Innerprinter_Address)) {

                innerprinter_device = device;

                break;

            }

        }

        return innerprinter_device;

    }
```

```
    public static BluetoothSocket getSocket(BluetoothDevice device) throws IOException {
        BluetoothSocket socket = device.createRfcommSocketToServiceRecord(PRINTER_UUID);
        socket.connect();
        return socket;
    }

    public static void sendData(byte[] bytes, BluetoothSocket socket) throws IOException {
        OutputStream out = socket.getOutputStream();
        out.write(bytes, 0, bytes.length);
        out.close();
    }
}
```

蓝牙连接打印服务事例

```
// 1: Get BluetoothAdapter
BluetoothAdapter btAdapter = BluetoothUtil.getBTAdapter();
    if (btAdapter == null) {
        Toast.makeText(getBaseContext(),"Please Open Bluetooth!", Toast.LENGTH_LONG)
                                                .show();
    return;
    }
```

```
// 2: Get Sunmi's InnerPrinter BluetoothDevice
```

```
BluetoothDevice device = BluetoothUtil.getDevice(btAdapter);
            if (device == null) {
            Toast.makeText(getBaseContext(),"Please Make Sure Bluetooth have          InnterPrinter!",
            Toast.LENGTH_LONG).show();
            return;
        }
```

```
// 3: Generate a order data ,   user add data here
```

```
byte[] data = null;
```

```
// 4: Using InnerPrinter print data
```

```
BluetoothSocket socket = null;
socket = BluetoothUtil.getSocket(device);
BluetoothUtil.sendData(data, socket);
```

```
```

注意：.需要在App的项目中添加蓝牙权限声明才能使用蓝牙设备。

```
<manifest>
    <uses-permission
android:name="android.permission.BLUETOOTH"></uses-permission>
    <uses-permission
android:name="android.permission.BLUETOOTH_ADMIN"></uses-permission>
</manifest>
```

1.3 方式 3 ： HTML 中的 JS

1.3.1 JS 简介

通过 JS 调用打印机本质上是通过 JS 桥在 HTML 中操作 android 原生的代码实现打印，通过蓝牙打印或者 AIDL 的方式实现打印。（sunmi 打印机本身不是网络打印机，网页应用无法直接与打印机通信，需要在 android 上有接受数据的应用）

1.3.2 HTML 使用

以下为具体的调用流程:

1.在 HTML 中定义 android 中要使用的方法。

```
document.getElementsByTagName('a')[0].addEventListener('click', function(){
var a = "wellcome to sunmi";
javascript:lee.funAndroid(a);
    return false;
      }, false);
```

2.初始化 WebView.

```
WebView mWebView = (WebView) findViewById(R.id.wv_view);
    // 设置编码
    mWebView.getSettings().setDefaultTextEncodingName("utf-8");
    // 支持 js
    mWebView.getSettings().setJavaScriptEnabled(true);
    mWebView.setWebChromeClient(new WebChromeClient());
```

初始化打印服务

```
Intent intent = new Intent();
intent.setPackage("woyou.aidlservice.jiuiv5");
intent.setAction("woyou.aidlservice.jiuiv5.IWoyouService");
startService(intent);//Start printer service
bindService(intent, connService, Context.BIND_AUTO_CREATE);
```

3.给 WebView 添加监听，在 onPageFinished 回调方法中调用 WebView.addJavascriptInterface(new JsObject(), 'lee');在 JsObject 类中通过 @JavascriptInterface 定义在 HTML 中指定要操作的方法 ,在方法中通过调用 AIDL 打印方式打印一张小票。

```
mWebView.setWebViewClient(new WebViewClientDemo());//添加一个页面相应监听类
class WebViewClientDemo extends WebViewClient {
      @Override
      public boolean shouldOverrideUrlLoading(WebView view, String url) {
          // 当打开新链接时，使用当前的 WebView，不会使用系统其他浏览器
           view.loadUrl(url);
           return true;
        }
       @Override
       public void onPageFinished(WebView view, String url) {
```

```
        super.onPageFinished(view, url);
            /**
             * 注册 JavascriptInterface，其中"lee"的名字随便取，如果你用"lee"，那么在 html 中只要用 lee.方法名()
             * 即可调用 MyJavascriptInterface 里的同名方法，参数也要一致
             */
              mWebView.addJavascriptInterface(new JsObject(), "lee");
                    }
          }
          class JsObject {
                    @JavascriptInterface
                    public void funAndroid(final String i) {
                              Toast.makeText(getApplicationContext(), "calling the local method funAndroid by JS. " + i,
                              Toast.LENGTH_SHORT).show();
                              try {
                                        woyouService.printerSelfChecking(callback);//Using AIDL to print something.
                              } catch (RemoteException e) {
                                        e.printStackTrace();
                              }
                    }
          }
```

4.加载 HTML 文件，在点击 HTML 中的按钮的时候就会打印一张小票了

```
// 载入包含 js 的 html
mWebView.loadData("", "text/html", null);
mWebView.loadUrl("file:///android_asset/test.html");//这里是您业务 html 所在的网页
```

2 状态反馈

2.1 打印机状态反馈

通过广播的形式

用户需要建立一个广播接收者来监听广播。

```
// 缺纸异常
public final static String OUT_OF_PAPER_ACTION =
"woyou.aidlservice.jiuv5.OUT_OF_PAPER_ACTION";
```

```
// 打印错误
public final static String ERROR_ACTION = "woyou.aidlservice.jiuv5.ERROR_ACTION";
```

```
// 可以打印
public final static String NORMAL_ACTION =
"woyou.aidlservice.jiuv5.NORMAL_ACTION";
```

```
// 开盖子
public final static String COVER_OPEN_ACTION =
"woyou.aidlservice.jiuv5.COVER_OPEN_ACTION";
```

```
// 关盖子异常
public final static String COVER_ERROR_ACTION =
"woyou.aidlservice.jiuv5.COVER_ERROR_ACTION";
```

```
// 切刀异常1 - 卡切刀
public final static String KNIFE_ERROR_1_ACTION =
"woyou.aidlservice.jiuv5.KNIFE_ERROR_ACTION_1";
```

```
// 切刀异常2 - 切刀修复
public final static String KNIFE_ERROR_2_ACTION =
"woyou.aidlservice.jiuv5.KNIFE_ERROR_ACTION_2";
```

```
// 打印头过热异常
public final static String OVER_HEATING_ACITON =
"woyou.aidlservice.jiuv5.OVER_HEATING_ACITON";
```

```
// 打印机固件开始升级
```

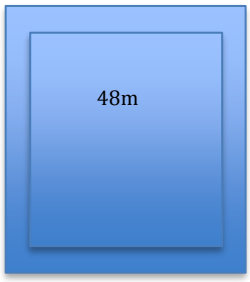
public final static String FIRMWARE_UPDATING_ACITON = "woyou.aidlservice.jiuv5.FIRMWARE_UPDATING_ACITON";

2.2 指令回调反馈

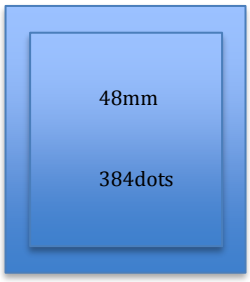
反馈函数	返回	
onRunResult	指令执行结果 boolean isSuccess	true执行成功，false 执行失败
onReturnString	指令执行结果 final String result result: 结果	result: 结果
onRaiseException	异常信息 int code, String msg code： 异常代码 msg: 异常描述	public static final int UNSUPPORT = -1; public static final String UNSUPPORT_MSG = "command is not support,index #"; //#代表第#个byte出错
		public static final int UNSUPPORTENCODING = -2; public static final String UNSUPPORTUNSUPPORTENCODING_MSG = "# encoding is not support";
		public static final int ADDTASKFAILED = -3; public static final String ADDTASKFAILED_MSG = "oops,add task failed (the buffer of the task queue is 10M),please try later";
		public static final int CODEFAILED = -4; public static final String CODEFAILED_MSG = "create command failed";
		public static final int IllegalParameter = -5; public static final String IllegalParameter_MSG = "Illegal parameter";
onPrintResult	Code 0 成功 1 失败 Msg 失败时的描述	针对事物打印的反馈 （需要打印服务 P1V1s 2.3.2 及以上版本支持）

3 打印服务参数介绍

一张 58 打印纸的宽度



有效打印宽度 48mm



有效打印像素点 384

注：Sunmi 打印机支持 58mm，80mm 打印纸，本文档以 58mm 打印纸为案例说明打印机的支持参数，80mm 打印纸规格类似。

一张 58 打印纸宽度为 58mm，有效打印宽度为 48mm。有效打印宽度一行为 384 个像素点。V1 纸槽深度 40mm，最多能放直径 40mm 的纸

3.1. 打印机分辨率

打印机分辨率为 205DPI，计算公式如下

DPI=384dots/48mm=8dots/1mm=205dots/in=205

3.2. 查询有无打印机

T1 设备分为有打印机硬件和无打印机硬件两个版本，用户可以通过打印机查询接口从软件上判断。

仅针对 T1 设备用户

接口	返回
Settings.Global.getInt(getContentResolver(), "hasPrinter", 0);	类型 int 内容： 0 无打印机 1 有打印机

	−1	正在查询

3.3. 字体说明

默认字体 24，中文为 24*24 的矩阵，英文为 12*24 的矩阵。

3.4. 二维码说明

sunmi 打印机打印二维码，每个二维码块为 4 个像素点（小于 4 扫码解析不出）。最大支持 version19（93*93）的模式。

3.5. 图片说明

sunmi 打印机支持最大打印图片大小为 1M，最大支持宽度为 384 个像素点。如果用户需要打印超过 1M 或者超过宽度 384 像素的图片，需要手动压缩图片。

```
if( mBitmap == null ){
            / * * * * 1M 以内的图片 * * * * * /
            mBitmap = BitmapFactory.decodeResource(getResources(), R.raw.sunmi);
            / * * * * 超过 1M 的图片 * * * * * * * * * * * /
            mBitmap1 = BitmapFactory.decodeResource(getResources(), R.raw.sunmi1);
           }
            / * * * * * 压缩图片 * * * * * * * * * * /
            double gh=(double)mBitmap.getWidth()/384;
            if( mBitmap1 == null ){
mBitmap2 = BitmapUtils.zoomBitmap(mBitmap1, 384, (int)(mBitmap.getHeight()/gh)); }
            / * * * * * * 压缩完成 * * * * * * * * * /
try {
            woyouService.setAlignment(1, callback);
            woyouService.printBitmap(mBitmap, callback);
//          woyouService.printBitmap(mBitmap2, callback);
```

```
            woyouService.lineWrap(3, null); }
catch (RemoteException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();}//当图片过大，也没有压缩，打印服务会反馈异常
```

3.6. 条码说明

依赖版本 V1_printerservice2.1.13V1（不同机型有略为不同）

code39　最长打印 13 个数字

code93　最长打印 17 个数字

code128　最长打印 15 个数字

ean8　要求 8 位数字（最后一位校验位），有效长度 8 个数字

ean13　有效长度 13 个数字，其中最后一位为校验位

ITF　要求输入数字，且有效小于 14 位，必须是偶数位

Codebar　要求 0-9 及 6 个特殊字符，最长打印 18 个数字

UPC-E　要求 8 位数字（最后一位校验位）

UPC-A　要求 12 位数字（最后一位校验位）

3.7. 字符集设置

默认 charracter 设置为简体中文。

切换字符集命令如下：

```
close multibyte(single byte fit for europe area)：0x1C 0x2E
open multibyte(fit for east asia) ：0x1C 0x26
Set Single byte  character set：0x1B 0x74  [parameter]
```

Single byte character set parameter list:

```
[para] [character set] [area]
0 "CP437";
2 "CP850";
```

```
3 "CP860";
4 "CP863";
5 "CP865";
13 "CP857";
14 "CP737";
15 "CP928";
16 "Windows-1252";
17 "CP866";
18 "CP852";
19 "CP858";
21 "CP874";
33 "Windows-775";
34 "CP855";
36 "CP862";
37 "CP864";
254 "CP855";

Set multibyte character set：1C 43 [parameter]
```

multibyte character set parameter list:

```
[para] [character set]
0x00 || 0x48 "GB18030";
//0x01 || 0x49 "BIG5";
0x02 || 0x50 "KSC5601";
(byte) 0xff "utf-8";
```

3.8. 黑标打印说明

商米 T1 可使用符合规范的黑标打印纸进行打印。

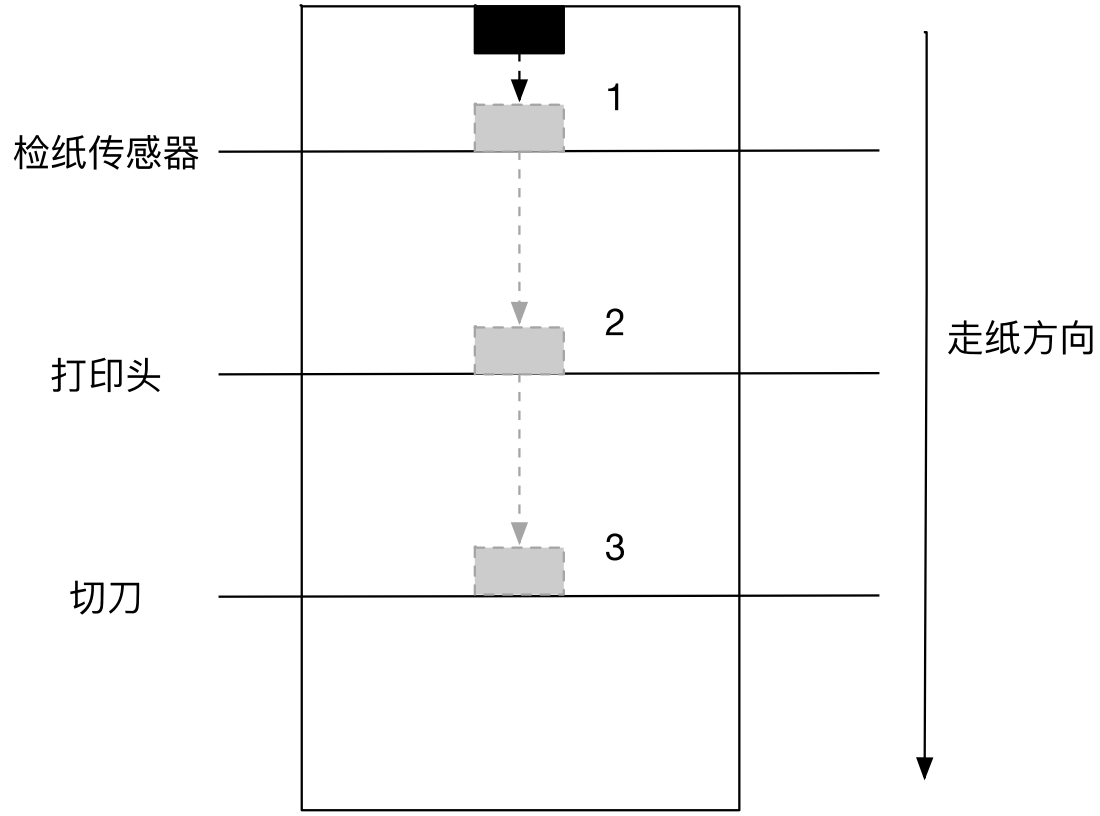
对黑标打印纸的要求：商米 T1 的黑标打印与一般的黑标打印机不同，商米 T1 硬件上并不具备检测黑标的传感器，而是利用了检纸传感器对反射率不超过 6%

（红外波长 700-1000nm 范围内）的材料认为是无纸的原理，制造了类似黑标打印的功能。因为原理不同，所以商米 T1 黑标打印模式无法像黑标打印机那样精确的定位黑标位置，存在一定误差）

对黑标位置的要求：黑标需要在纸张水平中间位置才能被检纸传感器器检测到。

商米 T1 黑标实现原理：

检纸传感器与打印头和切刀位置不在一个水平线上，纸张上的黑标会先经过检纸传感器（图 1 位置），再经过打印头（图 2 位置），最后到达切刀位置（图 3 位置）切纸出仓。



在黑标模式下，检纸传感器在检测到没纸后，打印机会自动走 7mm 的距离，如果走完之前检纸传感器还是没检测到有纸，就按照没纸处理。如果走完之前检测到有纸，就按照黑标处理。

当打印内容覆盖黑标位置，会继续打印即当黑标走到图 2 位置，仍然有打印任务，会覆盖黑标继续打印任务，直到打印完成。

按照此机制，打印最后的位置与检纸传感器需要保持一定距离（打印最后的内容与黑标边沿距离 6mm）才能保证打印内容在 2 个黑标范围内。

黑标模式指令序列：

1， 发送打印内容

2， 输入走纸到黑标的指令{0x1c,0x28,0x4c,0x02,0x00,0x42,0x31}

3， 输入切刀指令

打印会在完成打印内容后，自动检索下一个黑标，并在指定位置切刀。

用户可以在 设置->打印->内置打印管理中对黑标模式和切刀位置进行修改。

文档更新记录

文档版本	更新内容
1.0.0	原始版本
1.1.170301	增加：打印图片的规格说明 增加：有无打印机硬件查询接口说明
1.1.170315	增加：aidl 切刀接口 增加：aidl 开钱柜接口 增加：aidl 获取切刀次数接口 增加：aidl 获取钱柜打开次数接口
1.1.170322	增加：aidl 带反馈的事物打印接口 增加：callback 事物打印结果反馈

	修改：条码格式说明
1.1.170329	增加：字符集修改说明
1.1.170615	增加：事物打印说明
1.1.170726	修改：AIDL 调用的说明
1.1.170802	增加：T1 黑标打印说明
1.1.170803	增加：T1 黑标指令说明