Sunmi Printer 开发者文档

SUNMI Printer Developer Documentation

文档更新说明 Document Revisions

编号 Version	更新日期 Undate Date	组件版本 Component Ver.	更新内容 Updates	撰写人 Owner
1.0.0			原始版本 Original version	徐赟庭
1.1.170301	2017/03/01		增加：打印图片的规格说明 Addition: description on the spec of images to be printed 增加：有无打印机硬件查询接口说明 Addition: description on whether there is a hardware query interface	徐赟庭
1.1.170315	2017/03/15		增加：aidl 切刀接口 Addition: AIDL cutter interface 增加：aidl 开钱柜接口 Addition: AIDL interface to open the cash drawer 增加：aidl 获取切刀次数接口 Addition: AIDL interface to obtain the number of times the cutter has been used 增加：aidl 获取钱柜打开次数接口 Addition: AIDL interface to obtain the number of times the cash drawer has been opened	徐赟庭
1.1.170322	2017/03/22		增加：aidl 带反馈的事物打印接口 Addition: AIDL transaction print interface with feedback 增加：callback 事物打印结果反馈 Addition: result feedback on callback transaction print 修改：条码格式说明 Modification: description on barcode format	徐赟庭
1.1.170329	2017/03/29		增加：字符集修改说明 Addition: description of character sets change	徐赟庭
1.1.170615	2017/06/15		增加：事物打印说明 Addition: description of transaction printing	徐赟庭
1.1.170726	2017/07/26		修改：AIDL 调用的说明 Modification: description of AIDL calling	徐赟庭
1.1.170802	2017/08/02		增加：T1 黑标打印说明 Addition: description of T1 black mark print	徐赟庭

1.1.170803	2017/08/03		增加：T1 黑标指令说明 Addition: description of T1black mark instructions	徐赟庭
1.1.180523	2018/05/23		修改：全新文档格式，标准化整理 Modification: new document formate; standardization	Darren
1.1.180601	2018/06/01		添加：打印图片扩展接口，ICallback 实例说明 Addition: extension interface for image printing; instance description of ICallback	Darren
1.1.180612	2018/06/12		修改：加粗指令及部分描述 Modification: bold instruction and some descriptions.	Darren
1.1.180629	2018/06/29		修改：ICallback 接口方法错误及描述 Modification: ICallback interface method errors and descriptions	Darren
1.1.180912	2018/09/12		修改：重新整理文档内容 Modification: rearrange the document	田昆龙
1.1.190225	2019/02/25		修改：整理 faq Modification: FAQ	田昆龙
1.1.191008	2019/10/08		修改：整理 faq Modification: FAQ	田昆龙
1.1.191112	2019/11/12		修改：更新资源链接和介绍 Modification: update resource links and the introduction	田昆龙
1.1.191211	2019/12/11		增加：标签功能介绍 Addition: introduction to label printing function	田昆龙

目录 Content

简介 Introduction

商米机器内置了**缓存式热敏打印机**，允许 App 通过 sdk 直接打印热敏小票，具有打印机的商米产品有:

Some SUNMI devices come with an inbuilt thermal printer (with a buffer), which allow Apps to directly print thermal receipts through SDK. The SUNMI devices which are equipped with a printer are:

手持非金融系列 —— V1、V1s、V2 Pro 等
手持金融系列 —— P1、P1 4g 等
台式收银机设备 —— T1、T2、T1mini、T2mini 等
台式收银秤设备 —— S2 等

Mobile products – V1, V1s, V2 Pro, etc.
Smart payment products – P1, P1 4g, etc.
Desktop POS – T1, T2, T1 MINI, T2 MINI, etc.
POS scale – S2, etc.

商米产品的内置打印机一共有 2 种规格：

There are two specs of inbuilt printers for SUNMI devices:

- 80mm 纸张宽度，带切刀，兼容 58mm， 如 T1 搭载了这种打印机。
- Support 80mm paper width, paper cutter equipped, compatible with the 58mm spec. T1 is equipped with this spec of printer.
- 58mm 纸张宽度，不带切刀，如 V1 搭载了这种打印机。
- Support 58mm paper width, no paper cutter. V1 is equipped with this spec of printer.

App 开发者可以使用 3 种方式调用内置热敏打印机：

An App developer can call the inbuilt printer with 3 methods:

- **通过内置打印服务接口调用打印机**——此种方式适用于初次开发打印相关 app 且对 epson 指令没有了解的开发者，通过商米打印服务提供的多个打印接口实现自己需要的打印效果；
- **Call the printer via the inbuilt print service interface** – this method works best for the developers who are not familiar with Epson command and develop Apps related to printing for the first time. Developers can realize the desired printing effects through multiple printing interfaces provided by SUNMI print service;

•**通过内置虚拟蓝牙设备调用打印机**——此种方式适用于之前有过开发蓝牙、usb 打印机经验或是开发者 app 已经实现蓝牙打印机打印的开发者，仅需稍微改动代码即可实现打印效果；

•**Call the printer via the inbuilt virtual Bluetooth device –** this method works best for developers who have developed Bluetooth or USB printers before, or the App being developed has already realized Bluetooth printing. Developers can change some codes and realize the desired printing effects.

•**H5 Web 页面通过 JS 桥调用打印机**——此种方式适用于接入 H5 页面的 app；

• **Call the printer via JS bridge on H5 Web page** – this method works best for the app accessing to H5 webpage.

1. 通过内置打印服务接口调用打印机 Call a Printer via the Inbuilt Print Service Interface

1.1. 连接内置打印服务 Connect to the Inbuilt Print Service

1.1.1. 通过远程依赖方式 Via Remote Dependency

为了方便调用内置打印服务，对使用 gradle 配置的开发者，我们推荐使用远程依赖库，同时由于我们的库对机型问题做了适配，使用远程依赖方式会自动区分不同机型的接口，不用因为机型不同需要配置不同的 AIDL 文件而发愁，当用了错误的接口也会有相应异常提示；

For developers using gradle in configuration, we recommend to use remote dependency library in order to call the inbuilt print service conveniently. Meanwhile, our library has adapted itself to different device models, and interfaces for different models can be differentiated automatedly when using remote dependency. There is no need to configure different AIDL files to be in accordance with different device models, and reminders will be prompted if the developer uses a wrong interface.

集成方式 Integration

在 app 目录下的 build.gradle文件中添加依赖和属性配置：
Add dependency and attribute configuration to the build.gradle under the directory of the App:

dependencies { implementation 'com.sunmi:printerlibrary:1.0.7' }

初始化 Initialization

像绑定一个服务组件一样，这里通过单例调用绑定方法
Like binding a service component, we can call the binding method through singleton call

```
boolean result = InnerPrinterManager.getInstance().bindService(context, innerPrinterCallback);

InnerPrinterCallback innerPrinterCallback = new InnerPrinterCallback(){

        @Override
        protected void onConnected(SunmiPrinterService service){
            //这里即获取到绑定服务成功连接后的远程服务接口句柄 Here is the remote service interface handle after the binding service has been successfully connected
            //可以通过 service 调用支持的打印方法 Supported print methods can be called through service
        }

        @Override
        protected void onDisconnected() {
            //当服务异常断开后，会回调此方法，建议在此做重连策略 The method will be called back after the service is disconnected. A reconnection strategy is recommended here

        }
```

```
    }
```

result:表示绑定成功或失败 to show the binding result (succeeded or failed)

简单调用打印方法 A Simple Case to Call Print

```
    try{
        sunmiPrinterService.printText("要打印的内容 content to be printed\n",
            new InnerResultCallbcak() {
                @Override public void onRunResult(boolean isSuccess) throws
RemoteException {
                //返回接口执行的情况(并非真实打印):成功或失败 Return the execution
result (not a real printing): succeeded or failed
                }



                @Override
            public void onReturnString(String result) throws
    RemoteException {
                    //部分接口会异步返回查询数据 Some interfaces return inquired
data asynchronously
                }

                @Override
            public void onRaiseException(int code, String msg) throws RemoteException {
                    //接口执行失败时，返回的异常状态 The exception returned
when the interface failed to execute
                }

                @Override
                public void onPrintResult(int code, String msg) throws      RemoteException {
                    //事务模式下真实的打印结果返回 The real printing result
returned in transaction printing mode
                }
        }
        });
    } catch (RemoteException e) {
        e.printStackTrace();
        //如部分接口只能用于指定机型所以会跑出调用接口异常，如钱箱接口只能用于台
式机 If some interfaces can only be used for specified device models, the call error reminder will pop
out. For example, the interface of a cash drawer can only be used for a desktop device.
    }
```

结束释放 Disconnection

正常调用结束后可以调用如下方法，断开服务

After a normal call, you can call the following method to disconnect the service.

InnerPrinterManager.getInstance().unBindService(context, innerPrinterCallback);

类说明 Class Specifications

InnerPrinterManager 打印库的管理类，主要用来实现连接、断开打印服务
InnerPrinterManager is the management class of the print interface library, which is used to connect and disconnect print services.

InnerPrinterCallback 连接远程服务的回调接口，用来获取远程服务连接结果
InnerPrinterCallback is a callback interface to connect remote service, which is used to obtain the result of the remote service connection.

SunmiPrinterService 商米打印服务接口类，定义了所有的打印方法，通过连接服务后获取此类实例，同 AIDL IWoyouService 接口相同
SunmiPrinterService is the SUNMI print service interface class, which defines all printing methods. The instance of this type can be obtained through connecting to a service. It is the same as the AIDL IWoyouService interface.

InnerPrinterException 打印异常类，调用方法出现错误或异常； 由于部分机型不具备相应方法功能（如手持机器不具备钱箱功能，当调用开钱箱接口时会抛 出打印异常，相应接口不能使用）
InnerPrinterException is the printing exception class. An error or exception may occur when calling a method because some devices are not equipped with some functions (for example, a handheld device is not equipped with a cash drawer, printing error will pop out when calling the interface of a cash drawer).

InnerResultCallbcak 打印方法的回调接口，用来获取接口方法的执行情况
InnerPrinterException is the callback interface of printing method, which is used to obtain the execution result of an interface method.

InnerLcdCallback 客显方法的回调接口，用来获取客显方法的执行情况
InnerLcdCallback is a callback interface of customer display method, which is used to obtain the implementation result of customer display method.

InnerTaxCallback 税控方法的回调接口，用来获取发送税控的结果
InnerTaxCallback is the callback interface of tax control method, which is used to obtain the result of sending tax control.

1.1.2. 通过 AIDL 方式 Via AIDL

AIDL 简介 Intro to AIDL

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。商米 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

AIDL (Android Interface Definition language) is a description language of an Andriod inner progress communication interface, which can be used to define the commnucation interface in progress. SUNMI AIDL supplies capsulated common printing cammands to facilitate developers' quick access to SUNMI printers.

AIDL 使用 How to Use AIDL

建立连接可分以下 5 步骤：

Follow the following 5 steps to establish a connection:

1. 在项目中添加 AIDL 文档资源文件中附带的 AIDL 文件且不能修改包路径和包名（部分机型还包含 java 文件）；

1. Add the AIDL file in the AIDL resource document to the project, and please be aware that the package path and package name shall not be changed (java files might be included in some versions of devices);

2. 在控制打印的代码类中实现 ServiceConnection；

2. Realize ServiceConnection in the code class for print control;

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 onServiceConnected 回调为准。

3. Call ApplicationContext.bindService(), and pass the interface object of _x0005_AIDL during the realization of ServiceConnection. Please note: bindservice is not a blocking call, which means that the binding has not been completed after calling, and the binding result can only be confirmed by the callback of onServiceConnected.

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。

4. During the realization of ServiceConnection.onServiceConnected(), you will receive an Ibinder instance (the Service called). Transfer the parameter into IwoyouService type by calling **IWoyouService.Stub.asInterface(service).**

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

5. Now you can call all kinds of methods defined in the IwoyouService interface to print.

AIDL 资源下载 AIDL Resources Download

⚠注意：由于机型差异各个机型会有部分接口不同，所以请区分机型使用 AIDL 资源文件，目前有如下四个资源包：

⚠Please note: Please use AIDL resources according to device models, for the interfaces of different devices may vary. There are four resource packages currently:

old（V1）用于商米首款 V1 机型 old (V1) for the first SUNMI V1 model

——主要包含 Mainly includes:

IWoyouService.aidl 打印机接口文件
ICallback.aidl 打印机回调接口文件
TransBean.aidl
TransBean.java 页打印类文件

handheld（手持机 除 **V1**）用于商米手持机型，如 V1s、V2、P2 等
handheld (handheld devices except V1) for V1s, V2, P2, etc.

——主要包含 Mainly includes:

IWoyouService.aidl 打印机接口文件
ICallback.aidl 打印机回调接口文件
TransBean.aidl
TransBean.java 页打印类文件
ITax.aidl 税控回调接口文件

desktop（台式机）用于商米台式打印机，如 **T1**、**T2** 等
desktop (desktop devices) for SUNMI T1, T2, etc.

——主要包含 Mainly includes:

IWoyouService.aidl 打印机接口文件
ICallback.aidl 打印机回调接口文件
ITax.aidl 税控回调接口文件

desktop+lcd（台式+客显）用于商米带客显的打印机，如 **T1mini**、**T2mini**
Desktop + lcd (desktop + customer display) for SUNMI desktop devices with a customer display like **T1mini**, **T2mini.**

——主要包含 Mainly includes:

IWoyouService.aidl 打印机接口文件
ICallback.aidl 打印机回调接口文件
ITax.aidl 税控回调接口文件
ILcdCallback.aidl 客显回调接口文件

通用资源 General resource

WoyouConsts.java 定义打印机暴露的常量类，用来设置配置

绑定服务示例 An Example on Binding a Service

实现 ServiceConnection

Realize ServiceConnection

```
private ServiceConnection connService = new ServiceConnection() {
@Override
public void onServiceDisconnected(ComponentName name) {
    woyouService = null;
}
@Override
public void onServiceConnected(ComponentName name, IBinder service) {
     woyouService = IWoyouService.Stub.asInterface(service);
}
```

绑定打印服务
Bind print service

```
private void Binding(){
    Intent intent=new Intent();
    intent.setPackage("woyou.aidlservice.jiuiv5");
    intent.setAction("woyou.aidlservice.jiuiv5.IWoyouService");
    bindService(intent, connService, Context.BIND_AUTO_CREATE);
}
```

1.2. 接口定义说明 Description of Interface Definition

通过以上方式与内置打印服务建立连接后，并获得 IWoyouService 或 SunmiPrinterService 对象后即可调用如下的接口实现自己的打印

After establishing connection with the inbuilt print service using the methods stated above and obtaining the object of IWoyouService or SunmiPrinterService, you can call the interfaces below to print.

1.2.1. 打印机初始化及设置 Printer Initialization and Setting

编号 No.	方 法 Method
1	void **printerInit( )** 打印机初始化 Printer initialization
2	void **printerSelfChecking(ICallback** callback**)** 打印机自检 Printer self-inspection

1.打印机初始化 Printer Initialization

函数： void **printerInit( )**

Function: void printerInit( )

备注：重置打印机的逻辑程序（例如：排版、加粗等样式设置），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。

Note: reset the logic programs (type setting, bold, etc.) of a printer but not to clear the data in the buffer. Therefore, uncompleted print jobs will be continued after resetting.

2.打印机自检 Printer Self-Inspection

函数： void **printerSelfChecking (ICallback** callback**)**

Function: void **printerSelfChecking (ICallback** callback**)**

参数：callback → 结果回调。

Parameter: callback → result callback.

示例：

Example:

```
woyouService.printerSelfChecking(callback);
```

1.2.2. 获取设备及打印机信息 Get Device and Printer Information

编号 No.	方 法 Method
1	String **getPrinterSerialNo( )** 获取打印机板序列号

	Get the SN of a printer board
2	String **getPrinterModal( )** 获取打印机类型接口 Get printer type interface
3	String **getPrinterVersion( )** 获取打印机固件版本号 Get printer firmware version
4	Build**.MODEL**（常量） Build**.MODEL** (constant) 设备名称 Device name
5	int **updatePrinterState()** 获取打印机的最新状态 Get the latest status of a printer
6	String **getServiceVersion()** 获取打印服务版本号 Get the version number of a print service
7	int **getPrintedLength(ICallback** callback**)** 获取打印头打印长度 Get the print length of a printhead
8	int **getPrinterPaper()** 获取打印机当前的纸张规格 Get the current paper spec of a printer

1.获取打印机的最新状态 Get the Latest Status of a Printer

函数：String **updatePrinterState ()**

Function: String **updatePrinterState ()**

返回值：

1 → 打印机工作正常
2 → 打印机准备中
3 → 通讯异常
4 → 缺纸
5 → 过热
6 → 开盖
7 → 切刀异常
8 → 切刀恢复
9 → 未检测到黑标
505 →未检测到打印机
507 →打印机固件升级失败

Return value:

1 → The printer works normally
2 → Preparing printer

3 → Abnormal communication
4 → Out of paper
5 → Overheated
6 →Open the lid
7 → The paper cutter is abnormal
8 → The paper cutter has been recovered
9 → No black mark has been detected
505 →No printer has been detected
507 →Failed to upgrade the printer firmware

备注：目前 V1 设备暂不支持该接口；除主动获取状态，也可以通过注册广播异步获取，参考附录 A

Note: this interface is not available to V1 for now. aside from getting the status actively, you can also obtain them asynchronously by registering broadcast. Please refer to Appendix A.

2.获取打印机已打印长度 Get the Length the Printer Has Printed

函数：int **getPrintedLength(ICallback** callback**)**

Function: int **getPrintedLength(ICallback** callback**)**

说明：目前可获取到上电以来的打印长度，由于台式机和手持机的硬件区别，获取打印结果的返回略有不同，即手持机通过 ICallback callback 接口获取打印长度，台式机通过返回值直接获取长度。

Description: currently, the print length of a printer (since power on) can be obtained. For a desktop device and a handheld device, the ways to get the print length of a printer may vary due to the hardware difference, i.e. the print length can be obtained through the ICallback callback interface for a handheld device, and the print length can be obtained through the return value for a desktop device.

3.获取打印机当前的纸张规格 Get the Paper Spec of a Printer

函数：int **getPrinterPaper()**

Function: int **getPrinterPaper()**

说明：手持打印机默认为 58mm 的纸张规格，台式打印机默认为 80mm 的纸张规格，但可以通过增加挡板并进行打印机配置设置为使用 58mm 的纸张规格，此接口会返回当前打印机设置的纸张规格；

Description: by default, a handheld device uses 58mm printing paper and a desktop device uses 80mm printing paper. However, adding a partition and adjusting some configurations can be used to enable a desktop device uses 58mm printing paper. This interface will return the printing paper spec that is currently being used by the printer.

备注：目前台式机器 T1 在 v2.4.0 版本以上支持此接口；T2、S2 在 v1.0.5 以上支持此接口；其他机型在 4.1.2 版本后均支持此接口查询纸张规格;

Note: currently, desktop devices that support this interface are T1 (v2.4.0 and above), T2 (v1.0.5 and above) and S2 (v1.0.5 and above); other devices which with a version 4.1.2 and above also can use this interface to get the printing paper spec.

1.2.3. ESC/POS 指令 ESC/POS Instruction

编号 No.	方 法 Method

1	void **sendRAWData (**byte[] data, **ICallback** callback**)** 打印 ESC/POS 格式指令 Print ESC/POS format instruction

1.打印 ESC/POS 格式指令 Print ESC/POS Format Instruction

函数： void **sendRAWData(**byte[] data, **ICallback** callback**)**

Function: void **sendRAWData(**byte[] data, **ICallback** callback**)**

参数：

data → ESC/POS 指令。

callback → 结果回调。

Parameters:

data → ESC/POS instruction.

callback → result callback.

备注：相关指令，请参考《ESC/POS》指令集。

Note: for relevant commands, please refer to ***ESC/POS instruction set***.

示例：

Example:

```
woyouService.sendRAWData(new byte[]{0x1B,0x45,0x01}, callback);
// 1B,45, 01 是字体加粗指令 1B,45, 01 are instructions to realize bold font
```

1.2.4 打印样式设置接口说明 Description of the Interface for Print Style Setting

编号 No.	方 法 Method
1	void **setPrinterStyle(**int key, int value**)** 设置打印机的样式 Set printer style

1.设置打印机的样式 Set Printer Style

函数： void **setPrinterStyle(**int key, int value**)**

Function: void **setPrinterStyle(**int key, int value**)**

参数：参考 WoyouConsts.java 打印机样式设置常量类

key →通过常量类接口中的定义设置不同的属性，一般分 **ENABLE_XXX** 和 **SET_XXX**

value →对应属性设置状态或大小

使能 ENABLE_XXX 属性需要选择 ENABLE 或 DISABLE

设置 SET_XXX 属性需要设置具体的大小

Parameter: Refer to WoyouConsts.java printer style set constant class

key →Set attributes which are usually divided into **ENABLE_XXX** and **SET_XXX** through the definition in constant type interface.

value → corresponds to the attribute setting includes status or size

ENABLE or DISABLE should be chosen for the Enable attribute ENABLE_XXX.

The specific size should be set for the SET attribute SET_XXX.

示例：

Example:

```
woyouService.setPrinterStyle( WoyouConsts.ENABLE_ILALIC,WoyouConsts.ENABLE); //使之后打印文本内容变为斜体 to print italic text for the content printed subsequently
woyouService.setPrinterStyle(WoyouConsts.SET_LINE_SPACING, 123); //使之后打印文本行间距变为 123 个像素 to realize the leading (123 pixels) for the content printed subsequently
```

备注：此接口需要打印服务 v4.2.22 版本以上支持！

Note: this interface is only available to the print service (v4.2.22 and above)!

1.2.5. 黑标打印 Black Mark Print

编号 No.	方 法 / 指 令 Method/Instruction
1	int **getPrinterMode**() 获取打印机模式 Get the printer mode
2	int **getPrinterBBMDistance**() 走纸距离 Paper feed distance
3	相关模式设置，请在系统“设置”→”打印设置”中进行设置。 For relevant mode settings, please complete in “Setting” →”Print Setting”.

1.获取打印机模式 Get Printer Mode

函数： int **getPrinterMode**()

Function: int **getPrinterMode**()

返回值：

0 → 普通模式；

1 → 黑标模式；

备注：仅支持 T1 、T2 设备。

Return values:

0 → Normal mode;

1 → Black mark mode;

Note: only available to T1 and T2.

2.获取黑标模式打印机自动走纸距离 Get the Paper Feed Distance of a Printer in Black Mark Mode

函数： int **getPrinterBBMDistance**()

返回值： 走纸距离(点行)。

备注： 仅支持 T1 、T2 设备。

Function: int **getPrinterBBMDistance**()

Return value: paper feed distance (pixel line).

Note: only available to T1 and T2.

1.2.6. 文字打印 Character Printing

编号 No.	方 法 / 指 令 Method/Instruction
1	void **setAlignment**(int alignment, **ICallback** callback) 设置对齐模式 Set alignment
2	~~void **setFontName**(String typeface, **ICallback** callback)~~ ~~设置打印字体（暂不支持）~~ ~~Set print typeface (unavailable for now)~~
3	void **setFontSize**(float fontsize, **ICallback** callback) 设置字体大小 Set font size
4	esc/pos 指令：字体加粗{0x1B, 0x45, 0x1}、取消加粗{0x1B, 0x45, 0x0} 设置与取消加粗 esc/pos instruction: bold font {0x1B, 0x45, 0x1}, cancel bold font {0x1B, 0x45, 0x0} Set and cancel the bold font effect
5	void **printText**(String text, **ICallback** callback) 打印文字 Print text
6	void **printTextWithFont**(String text, String typeface, float fontsize, **ICallback** callback) 打印指定字体，大小的文本 Print text in a specified typeface and size
7	void **printOriginalText** (String text, **ICallback** callback) 打印矢量文字 Print vector font

1.设置对齐模式 Set Alignment

函数： void **setAlignment** (int alignment, **ICallback** callback)

Function: void **setAlignment** (int alignment, **ICallback** callback)

参数：

alignment → 对齐方式：0→居左 , 1→居中, 2→居右 。

callback → 结果回调。

Parameter:

alignment → alignment: 0→align left, 1→center, 2→align right.

callback → Result callback.

备注：全局方法，对之后执行的打印有影响，打印机初始化时取消相关设置。

Note: it is a global method which will affect the subsequent printing. If a printer initialization method has been called, this method will be restored to its default setting.

示例：

Example:

```
woyouService.setAlignment(1, callback);
```

2.设置打印字体（暂不支持）Set Typeface (Unavailable for Now)

函数： void **setFontName (**String typeface, **ICallback** callback**)**

Function: void **setFontName (**String typeface, **ICallback** callback**)**

参数：

typeface → 字体名称，目前中支持字体**"gh"，** gh 是一种等宽中文字体。

callback → 结果回调。

Parameters:

typeface → font name. Currently the font **"gh"** is available (a Chinese monospaced font).

callback → Result callback.

备注：全局方法，对之后执行的打印有影响，打印机初始化时取消设置（现有版本暂时不支持设置字体）。

Note: it is a global method which will affect the subsequent printing. If a printer initialization method has been called, this method will be restored to its default setting (setting fonts is unavailable to existing versions).

示例：

Example:

```
woyouService.setFontName("gh", callback);
```

3.设置字体大小 Set Font Size

函数： void **setFontSize (**float fontsize, **ICallback** callback**)**

Function: void **setFontSize (**float fontsize, **ICallback** callback**)**

参数：

fontsize → 字体大小。

callback → 结果回调。

Parameters:

fontsize → font size.

callback → Result callback.

备注：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。

Note: it is a global method which will affect the subsequent printing. The settings will be canceled if an initialization has been called. Setting font size is not included in the print method in the standard international instruction, and it will affect character width and the number of characters in a line correspondingly. As a result of this, the type setting based on monospace font will be changed.

示例：

Example:

```
woyouService.setFontSize(36, callback);
```

4.设置与取消加粗 Set and Cancel Bold

指令：字体加粗{0x1B, 0x45, 0x1}、取消加粗{0x1B, 0x45, 0x0}

Instruction: make fonts bold {0x1B, 0x45, 0x1}, cancel bold {0x1B, 0x45, 0x0}

备注：请参考本文 **1.1.3.3. ESC/POS 指令**。

Note: please refer to the **1.1.3.3. ESC/POS instruction** in this document.

示例：

Example:

```
woyouService.sendRAWData(new byte[]{0x1B,0x45,0x0}, callback);
// 取消字体加粗指令 Cancel the instruction of making fonts bold
```

5.打印文字 Print Text

函数：void **printText(**String text, in **ICallback** callback**)**

Function: void **printText(**String text, in **ICallback** callback**)**

参数：

text → 打印内容，文字宽度**超出一行自动换行排版**，不满一行或超出一行不满一行部分需要在结尾加**强制换行符"\n"**才会即时打印出来，否则会缓存在缓存区。

callback → 结果回调。

Parameters:

text → the text to be printed. The part of the text **which exceeds a line will be warped onto the next line.** The **force line break "\n"** must be added to the end of a line of content which not fully occupy the line in order to print this line of content, otherwise it will be cached in the buffer.

callback → Result callback.

备注：若要修改打印文本的样式（如：对齐方式、字体大小、加粗等），请在调用 **printText** 方法前设置。

Note: if a style (alignment, font size, bold, etc.) of the text to be printed needs to be changed, please set before calling **printText** method.

示例：

Example:

```
woyouService.setAlignment(1, callback);
woyouService.setFontSize(36, callback);
woyouService.printText("商米科技 SUNMI Technology\n", callback);
```

6.打印指定字体，大小的文本 Print Text in a Specialized Font and Size

函数： void **printTextWithFont(**String text, String typeface, float fontsize, **ICallback** callback**)**

Function: void **printTextWithFont(**String text, String typeface, float fontsize, **ICallback** callback**)**

参数：

text → 打印内容，文字宽度**超出一行自动换行排版**，不满一行或超出一行不满一行部分需要在结尾加**强制换行符"\n"**才会即时打印出来，否则会缓存在缓存区。

typeface → 字体名称（现有版本暂时不支持设置字体，默认）。

fontsize → 字体大小，只对该方法有效。

callback → 结果回调。

Parameters

text → the text to be printed. The part of the text which **exceeds a line will be wrapped onto the next line.** The **force line breaker "\n"** must be added to the end of a line of content which not fully occupy a line in order to print this line of text, otherwise it will be cached in the buffer.

typeface → font name (setting font is not available to the existing version. Default).

fontsize → font size. It takes effect only to this method.

callback → Result callback.

备注：字体设置只对本次有效。

Note: font setting is only valid for this time.

示例：

Example:

```
woyouService.printTextWithFont("商米 SUNMI\n","",36,callback);
```

7.打印矢量文字 Print Vector Font

函数： void **printOriginalText(**String text, **ICallback** callback**)**

Function: void **printOriginalText(**String text, **ICallback** callback**)**

参数：

text → 打印内容，文字宽度超出一行自动换行排版，不满一行或超出一行不满一行部分需要在结尾加强制换行符"\n"才会即时打印出来，否则会缓存在缓存区。

callback → 结果回调。

Parameters:

text　the text to be printed. The part of the text which exceeds a line will be wrapped onto the next line. The force line breaker "\n" must be added to the end of a line of content which not fully occupy a line in order to print this line of text, otherwise the line of content will be cached in the buffer.

callback → Result callback.

返回值：

Return value:

备注：文字按矢量文字宽度原样输出，即每个字符不等宽。

Note: output characters are **in the same width of vector fonts**, which means that **they are not monospace.**

示例：

Example:

```
woyouService.printOriginalText ("κρχκμνκλρκνκνμρτυφ\n", callback);
```

1.2.7. 表格打印 Print Tables

编号 No.	方 法 / 指 令 Method/Instruction
1	void **printColumnsText(**String[] colsTextArr, int[] colsWidthArr, int[] colsAlign, **ICallback** callback) 打印表格的一行(不支持阿拉伯字符) Print a row of a table (unavailable to Arabic characters)
2	void **printColumnsString(**String[] colsTextArr, int[] colsWidthArr, int[] colsAlign, **ICallback** callback) 打印表格的一行，可以指定列宽、对齐方式 Print a row of a table. Columnn width and alignment can be specified.

1.打印表格的一行(不支持阿拉伯字符) Print a Row of a Table (Unavailable to Arabic Characters)

函数： void **printColumnsText(**String[] colsTextArr, int[] colsWidthArr, int[] colsAlign, **ICallback** callback**)**

Function: void **printColumnsText(**String[] colsTextArr, int[] colsWidthArr, int[] colsAlign, **ICallback** callback**)**

参数：

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。

Parameters:

colsTextArr → each column's array of text string.

colsWidthArr →each column's width array, which is calculated based on English characters. Each Chinese character takes the space of two English characters, and the width exceeds 0.

colsAlign →each column's alignment: 0 align left, 1 center, 2 align right.

callback → Result callback.

备注：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，不支持阿拉伯字符。

Note: the array length of the three parameters should be the same. If the colsText[i] is wider than the colsWidth[i], the exceeding part will be wrapped onto the next line. This is unavailable to Arabic characters.

示例：

Example:

```
woyouService.printColumnsText(new String[]{"商米 SUNMI","商米 SUNMI","商米 SUNMI"}, new int[]{4,4,8}, new int[]{1,1,1},callback);
```

2.打印表格的一行，可以指定列宽、对齐方式 Print a Row of a Table With Specified Column Width and Alignment

函数： void **printColumnsString(**String[] colsTextArr, int[] colsWidthArr, int[] colsAlign, **ICallback** callback**)**

Function: void **printColumnsString(**String[] colsTextArr, int[] colsWidthArr, int[] colsAlign, **ICallback** callback**)**

参数：

colsTextArr → 各列文本字符串数组。

colsWidthArr → 各列宽度权重即各列所占比。

colsAlign → 各列对齐方式：0 居左, 1 居中, 2 居右。

callback → 结果回调。

Parameters:

colsTextArr → each column's array of text string.

colsWidthArr → the weight of each column's width, i.e. the width proportion of each column.

colsAlign → each column's alignment: 0 align left, 1 center, 2 align right.

callback → Result callback.

备注：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。

Note: the array length of the three parameters should be the same. If the colsText[i] is longer than the colsWidth[i], the exceeding part will be fed to the next line.

示例：

Example:

```
woyouService.printColumnsString(new String[]{"商米 SUNMI","商米 SUNMI","商米 SUNMI"}, new int[]{1,1,2}, new int[]{1,1,1},callback);
```

1.2.8. 打印图片 Print Images

编号 No.	方 法 Method

1	void **printBitmap**(Bitmap bitmap, **ICallback** callback) 打印图片 Print image
2	void **printBitmapCustom**(Bitmap bitmap, int type, **ICallback** callback) 打印图片(2) Print image (2)

1. 打印图片 Print Image

函数： void **printBitmap** (Bitmap bitmap, **ICallback** callback)

Function: void **printBitmap** (Bitmap bitmap, **ICallback** callback)

参数：

bitmap → 图片 Bitmap 对象。

callback → 结果回调。

Parameters:

bitmap → image Bitmap object.

callback → Result callback.

备注：图片像素分辨率小于 2M，且宽度根据纸张规格设置（58 为 384 像素，80 为 576 像素），如果超过纸张宽度将不显示

Note: the resolution of an image should be within 2M, and the width should be set in accordance with the paper spec (58: 384 Pixel, 80: 576 Pixel). It cannot be shown if it exceeds the paper width.

示例：

Example:

```
woyouService.printBitmap(bitmap,callback);
```

2. 打印图片(2) Print Image (2)

函数： void **printBitmapCustom** (Bitmap bitmap,int type **ICallback** callback)

Function: void **printBitmapCustom** (Bitmap bitmap,int type **ICallback** callback)

参数：

bitmap → 图片 bitmap 对象(最大宽度 384 像素，图片超过 1M 无法打印)。

type→ 目前有两种打印方式：

0 → 同方法 **printBitmap**();

1 → 阈值 200 的黑白化图片

2 → 灰度图片

callback → 结果回调。

Parameters:

bitmap → picture bitmap object (the maximum width is 384 Pixel, and a picture exceeding 1M cannot be printed).

type→ currently, two print methods are available:

0 → the same method of **printBitmap**();

1 → black and white picture (with the threshold value of 200)

2 → grayscale picture

callback → Result callback.

备注：图片像素分辨率小于 200 万，且宽度根据纸张规格设置（58 为 384 像素，80 为 576 像素），如果超过纸张宽度将不显示

支持版本：P1-v3.2.0 以上、P14g-v1.2.0 以上、V1s-v3.2.0 以上、V2-v1.0.0 以上、T1-v2.4.0 以上、T2、S2-v1.0.5 以上、T1mini-v2.4.1 以上、T2mini-v1.0.0 以上

Note: the resolution of an image should with within 2M, and the width should be set in accordance with the paper spec (58mm: 384 Pixel, 80mm: 576 Pixel). It cannot be shown if it exceeds the paper width.

Available to: P1 (v3.2.0 and above), P14g (v1.2.0 and above), V1s (v3.2.0 and above), V2 (v1.0.0 and above), T1 (v2.4.0 and above), T2 (v1.0.5 and above), S2 (v1.0.5 and above), T1mini (v2.4.1 and above), T2mini (v1.0.0 and above).

示例：

Example:

```
woyouService.printBitmapCustom(bitmap,callback);
```

1.2.9. 一维码与二维码的打印 Print 1D/2D Barcodes

编号 No.	方 法 Method
1	void **printBarCode**(String data, int symbology, int height, int width, int textPosition, **ICallback** callback) 打印一维条码 Print 1D barcodes
2	void **printQRCode**(String data, int modulesize, int errorlevel, **ICallback** callback) 打印 QR 条码 Print QR barcodes
3	void **print2DCode**(String data, int symbology, int modulesize, int errorlevel, **ICallback** callback) 打印二维条码 Print 2D barcodes

1. 打印一维条码 Print 1D Barcodes

函数： void **printBarCode**(String data, int symbology, int height, int width, int textPosition, **ICallback** callback)

Function: void **printBarCode**(String data, int symbology, int height, int width, int textPosition, **ICallback** callback)

参数：

data → 一维码内容。

symbology → 条码类型（0 - 8）：

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 - 255, 默认：162。
width → 条码宽度, 取值 2 - 6, 默认：2。
textPosition → 文字位置（0 - 3）：

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

Parameters:

data → the 1D barcode to be printed
symbology → barcode type (0-8):

0 → UPC-A
1 → UPC-E
2 → JAN13(EAN13)
3 → JAN8(EAN8)
4 → CODE39
5 → ITF
6 → CODABAR
7 → CODE93
8 → CODE128

height → barcode height (ranges from 1-255. 162 by default).
width → barcode width (ranges from 2-6. 2 by default).
textPosition → text position (0-3)

0 → don't print text
1 → text above the barcode
2 → text under the barcode
3 → text above and under the barcode

callback → Result callback

备注：条码种类不同有如下区别

Note: the differences caused by different codes are listed below

编码 Code	说明 Description
code39	最长打印 13 个数字 Numbers within 13 digits can be printed
code93	最长打印 17 个数字 Numbers within 17 digits can be printed

ean8	要求 8 位数字（最后一位校验位），有效长度 8 个数字 8-digit numbers (the last digit is for parity check). The effective length is 8 digits (numbers).
ean13	有效长度 13 个数字，其中最后一位为校验位 The effective length is 13 digits, and the last digit is for parity check.
ITF	要求输入数字，且有效小于 14 位，必须是偶数位 Number (even number of digits) within 14 digits is required.
Codabar	要求 0-9 及 6 个特殊字符，最长打印 18 个数字 Numbers within 0-9 plus 6 special characters. Maximum 18 digits can be printed.
UPC-E	要求 8 位数字（最后一位校验位） 8-digit number (the last digit is for parity check)
UPC-A	要求 12 位数字（最后一位校验位） 12-digit number (the last digit is for parity check)
code128	Code128 分三类： A 类：包含大写字母、数字、标点等； B 类：大小写字母，数字； C 类：纯数字，复数字符，若为单数位，最后一个将忽略； 接口默认使用 B 类编码，若要使用 A 类、C 类编码需在内容前面加“{A”、“{C”，例如：“{A2344A”，”{C123123”，”{A1A{B13B{C12”。 Code128 can be devided into subset A, B and C: 128A: numbers, upper-case letters, and control characters, etc. 128B: numbers, upper- and lower-case letters and special character. 128C: numbers only. It must have an even number of digits (if it has an odd number of digits, the last digit will be omitted). By default, the interface uses subset B. “{A” or “{C” should be added before the content if you need to use subset A or C, for example: “{A2344A”, ”{C123123”, ”{A1A{B13B{C12”.

示例：

Example:

```
woyouService.printBarCode("1234567890", 8, 162, 2, 2, callback);
```

2. 打印 QR 条码 Print QR Barcodes

函数： void **printQRCode** (String data, int modulesize, int errorlevel, **ICallback** callback)

Function: void **printQRCode** (String data, int modulesize, int errorlevel, **ICallback** callback)

参数：

data → QR 码内容。

modulesize → QR 码块大小，单位:点, 取值 4 至 16。

errorlevel → 二维码纠错等级(0 - 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。

Parameters:

data → QR code to be printed.

modulesize → the size of the QR code block. Unit: dot, which should be within 4-16.

errorlevel → QR code error correction level (0-3):

0 → error correction level L (7%)

1 → error correction level M (15%)

2 → error correction level Q (25%)

3 → error correction level H (30%)

callback → Result callback.

备注：普通打印状态下在调用该方法后会直接输出打印，每个二维码块为 4 个像素点（小于 4 扫码解析有可能失败）。最大支持 version19（93*93）的模式。

Note: after calling this method, the content will be printed under normal print status, and every QR code block is 4 Pixel (when smaller than 4 Pixel, the code parsing might fail). version19 (93*93) is a maximum mode supported.

示例：

Example:

```
woyouService.printQrCode("商米科技 SUNMI Technology", 4, 3, callback);
```

3. 打印二维条码 Print 2D barcodes

函数： void **print2DCode(**String data, int symbology, int modulesize, int errorlevel, **ICallback** callback)

Function: void **print2DCode(**String data, int symbology, int modulesize, int errorlevel, **ICallback** callback)

参数：

data → 二维码内容。

symbology → 二维码类型

1 Qr（同 **printQRCode** 接口）

2 PDF417

3 DataMatrix

modulesize → 二维码有效块大小，根据码类型不同，支持的最佳块大小不同

Qr 码　　4～16（同 **printQRCode** 接口）

PDF417　　1～4

DataMatrix　　4～16

errorlevel → 二维码纠错等级，根据码类型不同，支持等级范围不同

Qr 码　　0～3（同 **printQRCode** 接口）

PDF417　　0～8

DataMatrix 默认使用 ECC200 自动纠错 不支持设置

callback → 结果回调。

Parameters:

data → the 2D barcode to be printed.

symbology → the type of 2D barcode

1 Qr (same as the **printQRCode** interface)
2 PDF417
3 DataMatrix

modulesize → the effective code block size, and it varies in accordance with different types of 2D barcode

Qr code 4～16 (same as the **printQRCode** interface)
PDF417 1～4
DataMatrix 4～16

errorlevel → 2D barcode error correction level, and it varies in accordance with different types of 2D barcodes

Qr code 0～3 (same as the **printQRCode** interface)
PDF417 0～8
DataMatrix ECC200 error correction is used by default, and adjustment on setting is unavailable.

callback → Result callback.

备注：普通打印状态下在调用该方法后会直接输出打印；此接口在 4.1.2 版本后支持；

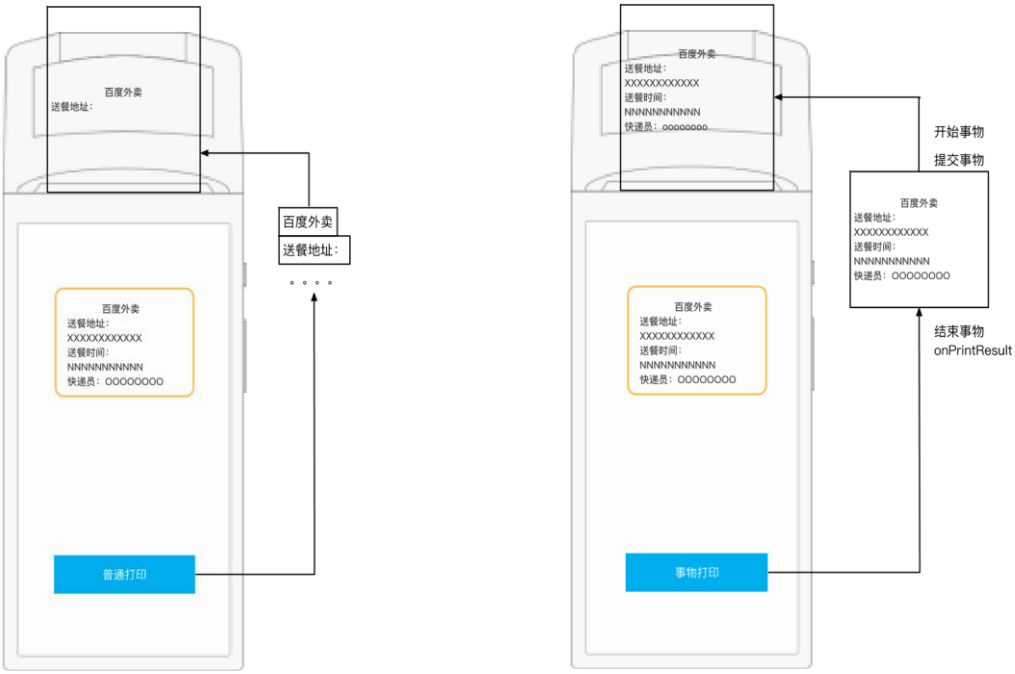
Note: after calling this method, the content will be printed under normal print status. This interface is available to version 4.1.2 and above.

1.2.10. 事务打印 Transaction printing

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。

The mode transaction printing caters to the needs to control content to be printed as well as get feedback on printing results. It is similar to establish a buffer for a transaction queue, when a developer uses this mode, a transaction list will be opened and print methods can be added. The printer will not print the content instantly, it

will print by the order of transactions after the transactions being submitted. After printing, the feedback on printing results will be obtained.



事务打印注意事项：

Notes on transaction printing:

1. 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；

1. when printing the queue of transactions, the printing results will be fed back after printing. However, **all print jobs submitted this time will loss if errors including out of paper, overheated occured, and an error feedback will be provided.** Which means, if the printer error occurred before or during a print job, the job won't be printed;

2. 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！

2. When alternately using (buffer) transaction printing and instruction printing, if a printer error occurred, the content of instruction printing won't be cleared!

3. 进入事务打印模式后，同样使用 **1.2.x.** 中的其它接口方法输出内容，但**不会**立即打印输出，会将输出内容缓存到缓存区，当调用 **exitPrinterBuffer()或 commitPrinterBuffer()**等方法才会进行打印输出。

3. When using transaction printing mode, other interfaces in **1.2.x.** can be used to output content, but the content will not be printed instantly but saved in the buffer, and it will be printed after methods **exitPrinterBuffer()** or **commitPrinterBuffer()** being called.

4. 事务打印结果回调在 **ICallback** 方法中的 **onPrintResult**(int code, String msg)方法（会有一定的**耗时**，要等**物理打印出纸**，不推荐**单行频繁使用**事务打印，将会**影响打印速度**，推荐**整张小票**使用事务打印），对应返回 code 如下：

a) 0 → 打印成功，msg 为“Transaction print successful!”；

b) 1 → 打印失败，msg 为“Transaction print failed!”；

4. The result of a transaction printing will be returned in the **onPrintResult** (int code, String msg) method of **ICallback interface** (it may **take some time,** for you have to wait for **the paper to be printed out**. Transaction printing is **not recommended to repeatedly print a single line,** for it will slow down the print speed. Transaction printing is suitable to print a **whole receipt**). The corresponding return codes are:

c) 0 → printed successfully, and the msg is “Transaction print successful!”;

d) 1 → failed to print, and the msg is “Transaction print failed!”.

5. 整个事务打印伪代码示例如下：

5. Below is a complete pseudocode example on transaction printing:

```
enterPrinterBuffer(true) ——进入事务模式，此后所有命令不会立刻输出
printText(/*something*/)
printBitmap(/*bitmap resource*/)
…… 其它打印相关方法——打印一些内容
commitPrinterBuffer()/commitPrinterBufferWithCallback(callback)——提交一次事务，此时打印机将开始打印，当打印成功或失败将在 callback 中返回
……等待上一次事务的返回
printText(/*something*/)
printBitmap(/*bitmap resource*/)
…… 其它打印相关方法——可以选择等待或不等待上一次事务的返回继续打印内容
commitPrinterBuffer()/commitPrinterBufferWithCallback(callback)——继续提交下一次事务，此时打印机将继续打印
exitPrinterBuffer(true)/exitPrinterBufferWithCallback(true, callback)——退出事物模式时调用，如果在上一次提交后又输入新的数据则会继续打印否则不打印
```

enterPrinterBuffer(true) ——enter into the transaction mode, the subsequent orders won't be outputted instantly
printText(/*something*/)
printBitmap(/*bitmap resource*/)
…… other print-relevant methods——print some content
commitPrinterBuffer()/commitPrinterBufferWithCallback(callback)——submit a transaction, and the printer will start printing. The result (succeeded or failed) will be returned in callback.
……Wait for the return of the last transaction
printText(/*something*/)
printBitmap(/*bitmap resource*/)
…… other print-relevant methods——you can choose to wait for the result of the last transaction printing or proceed printing
commitPrinterBuffer()/commitPrinterBufferWithCallback(callback)——continue to submit the next transaction, and the printer will continue to print at this time
exitPrinterBuffer(true)/exitPrinterBufferWithCallback(true, callback)——call it when you need to exit transaction mode, and printing will be continued if new data has been entered after the last submission, otherwise it won't continue printing.

6.

6. 具体方法说明：

6. Detailed methods:

编号 No.	方 法 Method
1	void **commitPrint**(TransBean[] tranBean, **ICallback** callback) lib 包事务打印专用接口 The interface specially for lib package transaction printing
2	void **enterPrinterBuffer**(boolean clean) 进入事务打印模式 Enetr into the transaction printing mode
3	void **exitPrinterBuffer**(boolean commit) 退出事务打印模式 Exit the transaction printing mode
4	void **exitPrinterBufferWithCallback**(boolean commit, **ICallback** callback) 退出事务打印模式并回调结果 Exit the transaction printing mode and callback the result
5	void **commitPrinterBuffer**() 提交事务打印 Submit transaction printing
6	void **commitPrinterBufferWithCallback**(**ICallback** callback) 提交事务打印并回调结果 Submit transaction printing and callback the result

7.

1. lib 包事务打印专用接口 The lib package transaction printing interface

函数： void **commitPrint (**TranBean[] tranBean, **ICallback** callback**)**

Function: void **commitPrint (**TranBean[] tranBean, **ICallback** callback**)**

参数：

tranBean → 打印任务列表。

callback → 结果回调。

Parameters:

tranBean → Print job list.

callback → Result callback.

示例：

Example:

```
woyouService.commitPrint(tranBean, callback);
```

2. 进入事务模式 Enetr into the transaction printing mode

函数： void **enterPrinterBuffer(**Boolean clear**)**

Function: void **enterPrinterBuffer(**Boolean clear**)**

参数：

clear → 是否清除缓冲区内容：

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。

备注：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，

支持版本: 除 V1 设备

Parameters:

clear → whether to clear the cache in the buffer:

true → clear the unsubmitted content of the last transaction printing;

false → not to clear the unsubmitted content of the last transaction printing, and the content will be included in the submission next time.

Note: enable and enter into the transaction printing mode. In this mode, content will not be printed instantly until the submission of a transaction or exiting the submission of a transaction.

Version supported: except V1 device.

示例：

Example:

```
woyouService.enterPrinterBuffer(false);
```

3. 退出事务模式 Exit the transation printing mode

函数： void **exitPrinterBuffer(**Boolean commit**)**

Function: void **exitPrinterBuffer(**Boolean commit**)**

参数：

commit → 是否打印出缓冲区内容：
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

支持版本: 除 V1 设备

Parameters:

commit → whether to print the content in the buffer:
true → print all content in transaction queue
false → not to print the content in transaction queue, and the content will be saved and submitted next time.

Versions supported: except V1 device.

示例：

Example:

```
woyouService.exitPrinterBuffer(true);
```

4. 退出事务打印模式并回调结果 Exit the transaction printing mode and callback the result

函数： void **exitPrinterBuffer(**Boolean commit, **ICallback** callback**)**

Function: void **exitPrinterBuffer(**Boolean commit, **ICallback** callback**)**

参数：

commit → 是否打印出缓冲区内容：
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

callback → 结果回调。

Parameters:

commit → whether to print the content in the buffer:
true → print all content in the buffer
false → not to print the content in the queue of transacion and save it to submit next time

callback → Result callback

支持版本: 除 V1 设备

Versions supported: except V1 device.

示例：

Example

```
woyouService.exitPrinterBuffer(true);
```

5. 提交事务打印 Submit transaction printing

函数： void **commitPrinterBuffer()**

Function: void **commitPrinterBuffer()**

备注：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

Note: submit and print all content of a transaction queue, and the transaction mode will be remained after the submission and printing.

支持版本: 除 V1 设备

Versions supported: except V1 device.

示例：

Example:

```
woyouService.commitPrinterBuffer();
```

6. 提交事务打印并回调结果 Submit Transaction Printing and Callback the Result

函数： void **commitPrinterBufferWithCallbacka(ICallback** callback**)**

Function: void **commitPrinterBufferWithCallbacka(ICallback** callback**)**

参数：

callback → 结果回调。

Parameter:

callback → Result callback

支持版本: 除 V1 设备

Versions supported: except V1 device.

示例：

Example:

```
woyouService.commitPrinterBufferWithCallback(callback);
```

1.2.11. 走纸相关 Line feed

编号 No.	方 法 Method
1	void **lineWrap**(int n, **ICallback** callback) 打印机走纸 n 行 Implement n LFs on the paper

1. 打印走纸 n 行 Implement n LFs on the paper

函数： void **lineWrap(**int n, **ICallback** callback**)**

Function: void **lineWrap(**int n, **ICallback** callback**)**

参数：

n → 走纸行数。

callback → 结果回调。

Parameters:

n → The number of LFs on the paper

callback → Result callback.

备注：强制换行，结束之前的打印内容后走纸 **n** 行。

Note: line feed by force. Implement **n** LFs on the paper after the content has been printed.

示例：

Example:

```
woyouService.printBitmap(3, callback);
```

1.2.12. 切刀（切纸）相关 Cut Paper

编号 No.	方 法 Method
1	void **cutPaper(ICallback** callback**)** 切纸 Cut paper
2	int **getCutPagerTimes()** 获取切刀累计次数 Get the number of times a cutter has been used

1. 切纸 Cut Paper

函数： void **cutPaper (ICallback** callback**)**

Function: void **cutPaper (ICallback** callback**)**

参数：

callback → 结果回调。

Parameter:

callback → Result callback

备注：由于打印头和切刀有一定距离，调用接口将自动补全这段距离；

仅支持台式机带切刀功能机器。

Note: the gap between the printhead and the paper cutter will be filled up by calling the interface to make sure the printed content won't be cut.

It is only available to the desktop devices with a cutter.

示例：

Example:

```
woyouService.cutPager(callback);
```

2. 获取切刀次数 Get the Number of Times a Cutter Has Been Used

函数： int **getCutPaperTimes ()**

Function: int **getCutPaperTimes ()**

返回值： 切刀次数。

Return value: the times the cutter has been used.

备注：仅支持台式机带切刀功能机器。

Note: it is only available to the desktop devices with a cutter function.

1.2.13. 钱箱相关 Cash Drawer

编号 No.	方 法 Method
1	void **openDrawer(ICallback** callback**)** 打开钱箱 Open a cash drawer
2	int **getOpenDrawerTimes()** 获取钱箱累计打开次数 Get the times a cash drawer has been opened
3	int **getDrawerStatus()** 获取当前的钱箱状态 Get the current status of a cash drawer

1. 打开钱箱 Open a Cash Drawer

函数： void **openDrawer (ICallback** callback**)**

Function: void **openDrawer (ICallback** callback**)**

参数：

callback → 结果回调。

Parameter:

callback → Result callback.

备注：仅支持台式机带钱箱功能机器。

Note: it is only available to the desktop devices with a cash drawer function.

示例：

Example:

```
woyouService.openDrawer(callback);
```

2. 获取钱箱累计打开次数 Get the Number of Times a Cash Drawer Has Been Opened

函数： int **getCutPaperTimes ()**

Function: int **getCutPaperTimes ()**

返回值： 钱箱累计打开次数。

Return value: the times a cash drawer has been opened.

备注：仅支持台式机带钱箱功能机器。

Note: it is only available to the desktop devices with a cash drawer function.

3、获取当前的钱箱状态 Get the Current Status of a Cash Drawer

函数： int **getDrawerStatus()**

Function: int **getDrawerStatus()**

说明： 可以通过此接口在部分具有连接钱箱功能的机型上获取钱箱开关状态

Description: this interface can be used to get the status of a cash drawer which is connected to a device (available to devices that can be connected to a cash drawer)

备注： 目前仅对 S2、T2、T2mini 机器 v4.0.0 版本以上支持此接口

Note: currently, it is available to S2, T2 and T2 mini with v4.0.0 and above.

1.2.14. 获取全局设置属性 Get the Attributes of Global Settings

商米机器可以通过设置配置一些打印样式，从而覆盖掉开发者自定义的样式，通过下列接口可以获取当前启用的配置状态；

You can override the style set by a developer through some print style setting and configuration. The interfaces below can be used to obtain the enabled configuration status.

编号 No.	方 法 Method
1	int **getForcedDouble()** 获取全局字体倍高倍宽状态 Get the status of global fonts in double height and double width
2	boolean **isForcedAntiWhite()** 获取全局字体反白样式使能 Get the anti-white style enable of global fonts
3	boolean **isForcedBold()** 获取全局字体加粗样式使能 Get the bold style enable of global fonts
4	boolean **isForcedUnderline()** 获取全局字体下划线样式使能 Get the underline style enable of global fonts
5	int **getForcedRowHeight**() 获取全局行高设定值 Get the set value of global line height
6	int **getFontName**() 获取当前的使用字体 Get the font currently being used
7	int **getPrinterDensity**() 获取打印机浓度 Get the printer density

备注： 目前除获取打印机浓度接口其他接口仅在手持机 V1、V1s、P1 v3.2.0 版本以上及 P14g v1.2.0 以上支持

Note: the interfaces above (except the interface to get the printer density) are available to V1, V1s, P1 with a version v3.2.0 and above and P14g with a version v1.2.0 and above.

1.2.15. 顾显(客显）接口说明 Description of the Customer Display Interface

mini 系列机器具有顾显（客显）功能，可通过如下方法使用；

This is available to the devices of mini series. Please see the table below for instruction:

编号 No.	方 法 Method
1	void **sendLCDCommand**(int flag) 发送控制命令 Send control command
2	void **sendLCDString**(String string, ILcdCallback callback) 发送单行文本内容 Send a single-line text
3	void **sendLCDDoubleString**(String topText, String bottomText, ILcdCallback callback) 发送双行文本内容 Send a double-line text
4	void **sendLCDMultiString**(String[] text, int[] align, ILcdCallback callback) 发送多行文本内容，每行内容根据权重自动分配大小 Send a multi-line text. The content size of each line will be adjusted according to the weight.
5	void **sendLCDFillString**(String string, int size, boolean fill, ILcdCallback callback) 发送单行文本，文本可设置字体大小，可设置填充方式 Send a single-line text. The filling method and font size can be set.
6	void **sendLCDBitmap**(Bitmap bitmap, ILcdCallback callback) 发送 bitmap 图显示 Send bitmap and show it on the customer screen

1. 发送顾显控制命令 Send Customer Display Control Command

函数： void **sendLCDCommand**(int flag)

Function: void **sendLCDCommand**(int flag)

参数： flag → 1 初始化 2 唤醒 LCD 3 休眠 LCD 4 清屏

Parameter: flag → 1 initialization 2 awake LCD 3 hibernate LCD 4 clear screen

备注：仅支持台式机 mini 系列带顾显功能机器。

Note: only available to the desktop devices of mini series which have a customer display.

2. 发送单行文本内容 Send a Single-Line Text

函数： void **sendLCDString**(String string, ILcdCallback callback)

Function: void **sendLCDString**(String string, ILcdCallback callback)

参数： string → 显示的文本内容

callback → 异步回调执行结果

Parameters:

string → the text displayed

callback → the asynchronous callback result

备注：仅支持台式机 mini 系列带顾显功能机器，文本内容过长将不显示。

Note: only available to the desktop devices of mini series which have a customer display. The text cannot be displayed if it is too long.

3. 发送双行文本内容 Send a Double-Line Text

函数：void **sendLCDDoubleString**(String topText， String bottomText, ILcdCallback callback)

Function: void **sendLCDDoubleString**(String topText， String bottomText, ILcdCallback callback)

参数： topText → 显示上半区文本内容

bottomText → 显示下半区文本内容

callback → 异步回调执行结果

Parameters

topText → display the top half text

bottomText → display the bottom half text

callback → the asynchronous callback result

备注：仅支持台式机 mini 系列带顾显功能机器，文本内容过长将不显示。

Note: only available to the desktop devices of mini series which have a customer display. The text cannot be displayed if it is too long.

4. 发送多行文本内容 Send a Multi-Line Text

函数：void **sendLCDMultiString**(String[] text, int[] align, ILcdCallback callback)

Function: void **sendLCDMultiString**(String[] text, int[] align, ILcdCallback callback)

参数： text → 显示各行文本内容的数组，根据数组元素数量确定行数，某行可为空则不显示内容

align → 各行文本所占区域的权重比，此数组元素大小必须同文本数组一致

callback → 异步回调执行结果

Parameters:

text → display the array of each line, and determine the number of lines based on the array. No content will be displayed if the line is a blank.

align → The weight ratio of each line takes in the area. The element size of this array must be consistent with the text array.

callback → the asynchronous callback result

备注：仅支持台式机 mini 系列带顾显功能机器，需要 v4.0.0 以上版本支持，文本内容过长将不显示。

Note: only available to the desktop devices (v4.0.0 and above) of mini series which have a customer display. The text cannot be displayed if it is too long.

5. 发送自定义大小单行文本内容 Send a single-line text in a customized size

函数：void **sendLCDFillString**(String string, int size, boolean fill, ILcdCallback callback)

Function: void **sendLCDFillString**(String string, int size, boolean fill, ILcdCallback callback)

参数： string → 要显示的文本内容

size → 显示文本内容的字体大小，目前默认居中

fill → 是否拉伸字体填充满显示区域
callback → 异步回调执行结果

Parameters

string → the text to be displayed
size → the font size of the text. The font is centered by default
fill → whether to stretch the text to fully fill the display area
callback → the asynchronous callback result

备注：仅支持台式机 mini 系列带顾显功能机器，需要 v4.0.0 以上版本支持，文本内容过长将不显示。

Note: only available to the desktop devices (v4.0.0 and above) of mini series which have a customer display. The text cannot be displayed if it is too long.

6. 发送图片显示内容 Send a bitmap and show it on the customer screen

函数：void **sendLCDBitmap**(Bitmap bitmap, ILcdCallback callback)

Function: void **sendLCDBitmap**(Bitmap bitmap, ILcdCallback callback)

参数： bitmap → 要显示的图片内容
callback → 异步回调执行结果

参数： bitmap → the picture to be displayed
callback → the asynchronous callback result

备注：仅支持台式机 mini 系列带顾显功能机器，顾显可显示最大图片像素为 128*40。

Note: only available to the desktop devices of mini series with a customer display. The picture with a maximum pixel128*40 can be shown on a customer display.

1.2.16.标签打印说明 Introduction to label printing

目前商米 V2, V2 Pro 机器均支持打印标签纸的功能，**要求标签纸的规格为宽度 50mm，高度 30mm 以上，标签间隙 2mm 以上（达到最佳定位效果）**;

Now, SUNMI V2 and V2 Pro support label printing functions. Label paper should be 50mm wide and 30mm high and a gap between labels should be at least 2mm (to reach the best positioning effect);

前置条件：

Precondition:

1.设置模式—使用标签打印功能请打开设置->内置打印->打印模式选择，选择标签热敏模式，选择一次后机器将记录并作为标签打印机使用，如下图：

1. Set mode – to use label printing function, please open setting->inbuilt printing->printing mode selection and choose label thermal mode. Your selection will be recorded by the device and turned the device into a label printer when printing, please see the picture below:

开发者可以通过获取打印机模式接口 getPrinterMode()查询当前的打印机模式

A developer can get te printer mode interface getPrinterMode() to find out the mode of a printer.

2.学习标签纸—如果首次使用或者更换了不同类型的标签纸，请打开设置->内置打印->标签学习，点击学习标签纸，如下图：

2.Learn label paper – if it is the first time for you to use it or you have changed a different type of label paper, please open setting->inbuilt printing->label learning and click the button learn label paper which shows as the picture below.

这将启动学习标签纸程序，请保证至少有三张标签，学习结束后会通过弹窗提示学习结果；

This will initiate label paper learning process. Please ensure there are at least 3 pieces of labels. A popoup box will show to indicate the result of learning when the learning process ends.

标签相关接口：

Label-related interfaces:

编号 **No.**	**方法** **Method**
1	void **labelLocate**() 定位下一张标签

	Position the next label
2	void **labelOutput**() 输出标签 Output a label

打印标签：

Print a label:

标签接口仅提供了针对标签的定位输出操作，而标签内容需要开发者根据自己的需求，像热敏打印一样构造自己的内容，其中需要注意的是要控制打印内容的高度在标签纸内，否则将造成打印内容超出标签纸，打印到下一张标签或定位下一张标签不准的问题；

Label-related interfaces only provide how to position a label, so printing a label content shall be formed by developers to their needs, just like what should be done with thermal printing. What should be paid attention to is the height of a label content should be within the hight of a label, otherwise the content may exceed a label paper to the next label or lead to an inaccurate label positioning.

如果使用高度为 30mm 的标签纸，那么支持的高度是 240（30mmx8 点）像素高度的打印内容，打印机默认行间距是 32 点行，那么可以打印 8 行左右的文本内容或者一张 384x240 高度的图片

If you use a 30mm-high label paper, a content of 240 (30mmx8 dots) pixel-high can be printed. The default print line spacing is 32 pixel line, so an 8-line text content or a 384x240 high picture can be printed normally.

举例来说—构造一个简单标签内容如下

For example – how to form a simple label content:

```
labelLocate()                                   //打印前需要定位标签 Positioning the label before printing is required
setPrinterStyle(WoyouConsts.ENABLE_BOLD, WoyouConsts.ENABLE)
                                                //设置字体加粗 Set to realize bold fonts
linewrap(1, null)                               //头部留一个空行 Leave a blank line at the head
setAlignment(0, null)                           //内容居左 Set the content to align left
printText("商品 Item:    豆豆浆 Soybean milk\n", null)        //打印内容，需要\n 换行 Print content, \n line feed is needed
printText("到期时间 Expiration time:    12-13 14 时 12-13 14:00\n", null)    //打印内容，需要\n 换行 Print content, \n line feed is needed
printBarCode("{C1234567890123456", 8, 90, 2, 2, null)
//打印条码，这里打印 code128c 类型条码，高度为 90，将占据 154 点行（条码会预留上下一个空行）Print a barcode. Here the barcode printed is a code128c type barcode with a height of 90, which occupies 154 pixel line (a blank line above and below the barcode will be left)
printText("\n", null)                            //条码需要换行输出 The barcode needs to be outputted with a line feed
labelOutput()                                   //打印后根据需要输出标签 The label will be outputted in accordance with needs after printing
```

这将输出一个基本充满标签区域的内容，如下

A content basically take all the area of a label will be outputted, please see the picture below:

开发者可以根据增加其他 API 来设计自己需要的标签内容
A developer can add other APIs to design his/her desired label content.

打印多张标签：
Print more than one label:
如果只打印一张标签，只需要执行以下操作
If to print only one label, you can:
labelLocate -> 打印标签 -> labelOutput
labelLocate -> Print label -> labelOutput
如果要打印多张标签，那么不用每次发送标签内容后执行 labelOutput 输出标签，只需要
If to print more than one label, you do not have to send label content and conduct labelOutput to output labels every time, you only need to:
labelLocate->打印标签 1->labelLocate->打印标签 2->labelLocate->打印标签 3……
->labelLocate->打印标签 n->labelOutput
labelLocate->Print label 1->labelLocate->Print label 2->labelLocate->Print label 3……
->labelLocate->Print label n->labelOutput

备注：当前标签只能通过接口方式使用，暂不支持蓝牙 ESC 指令
Note: Currently label can only be used through interface method instead of Bluetooth ESC command.

1.3. 接口返回说明 Description of Interface Return Result

1.3.1. ICallback 接口方法说明 Description of ICallback Interface Methods

编号 No.	方 法 Method
1	void **onRunResult**(boolean isSuccess) 指令执行结果 The result of instruction excution
2	void **onReturnString**(String result) 指令执行结果返回 String

	The return result string of instruction execution
3	void **onRaiseException**(int code, String msg) 异常信息返回 The return exception
4	void **onPrinterResult**(int code, String msg) 针对事物打印的反馈 The feedback on transaction printing

1. 指令执行结果 Result on Instruction Execution

函数： void **onRunResult** (boolean isSuccess)

参数：

isSuccess → 执行结果：

true → 执行成功

false → 执行失败

备注： 该接口返回处理结果是指命令处理执行结果，而不是打印出纸的处理结果。

Function: void **onRunResult** (boolean isSuccess)

Parameters:

isSuccess → execution result:

true → succeeded

false → failed

Note: this interface return result indicates the result of executing an instruction, instead of the result of printing a paper.

2. 指令执行结果返回 String　Return Result on Instruction Execution

函数： void **onRrturnString** (String result)

参数：

result → 执行结果

Function: void **onRrturnString** (String result)

Parameters

result → result on execution

3. 异常信息返回 Return Exception Message

函数： void **onRaiseException** (int code, String msg)

参数：

code → 异常代码。

msg → 异常描述。

备注： 见异常信息对照。

Function: void **onRaiseException** (int code, String msg)

Parameter:

code → exception code

msg → description on an exception

Note: see Exception Table.

4. 针对事物打印的反馈 Feedback on Transaction Print

函数： void **onPrintResult (**int code, String msg**)**

参数：

code → 状态码：

0 →成功。

1 →失败。

msg → 成功时为 null，失败时的异常描述。

备注：该接口返回结果是指令处理及物理打印输出后的结果（会有一定的**耗时**，要等**物理打印出纸**）。

Function: void **onPrintResult (**int code, String msg**)**

Parameters:

code → status code:

0 →succeeded

1 →failed

msg → succeeded: null; failed: description on the exception.

Note: this interface return result indicates the results of instrctuction execution and printing (it may **take some time for printing**).

1.3.2. Callback 对象示例　Examples on Callback Object

```
new ICallback.Stub(){
        @Override
        public void onRunResult(boolean b) throws RemoteException {
            // 指令执行结果 Instruction execution result
        }
        @Override
        public void onReturnString(String s) throws RemoteException {
            // 指令执行结果返回 String Return String on instruction execution
        }
        @Override
        public void onRaiseException(int i, String s) throws RemoteException {
            // 异常信息返回  Return Exception Message
        }
        @Override
        public void onPrintResult(int i, String s) throws RemoteException {
             // 针对事物打印的反馈 Feedback on Transaction Print
        }
    }
```

1.3.3. 异常信息对照表 Exception Table

code	msg
-1	"command is not support,index #"; //＃代表第＃个 byte 出错 ＃indicates that the ＃th byte went wrong
-2	"# encoding is not support"; //＃代表第＃个 byte 出错＃indicates that the ＃th byte went wrong
-3	"oops,add task failed (the buffer of the task queue is 10M),please try later";
-4	"create command failed";
-5	"Illegal parameter";
-6	"param found null pointer"

2. 通过内置虚拟蓝牙调用打印机 Call a Printer via the Inbuilt Virtual Bluetooth

2.1. 虚拟蓝牙简介 Intro on Virtual Bluetooth

在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。

You can find a Bluetooth device “InnerPrinter” (it has already been paired and always stays in the list of Bluetooth list), it is a virtual Bluetooth printer invented by the OS. It supports SUNMI esc/pos instructions.

其中有部分特殊的指令属于 sunmi 自定义指令，如：

Among all the instructions, some are SUNMI customized instructions. For example:

功能 Function	指令 Instruction
开钱箱指令 Open the cash drawer	byte［5］：0x10 0x14 0x00 0x00 0x00
切刀指令 全部切割 Cutter – cut the receipt paper	byte［4］：0x1d 0x56 0x42 0x00
切刀指令 切割（左边留一点不切） Cutter – cut the receipt paper (but save a little bit in the left side)	byte［4］：0x1d 0x56 0x41 0x00

2.2. 虚拟蓝牙使用 How to Use the Virtual Bluetooth

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。

1. Connect to the Bluetooth device.
2. Combine and transcode instructions and text into Bytes.
3. Send the Bytes to InnerPrinter.
4. Drive the printer with the underlying printing service to complete printing.

Note: **BluetoothUtil** is a Bluetooth tool class, which is used to connect the virtual Bluetooth device InnerPrinter.

2.2.1. 工具类 BluetoothUtil，标准的蓝牙连接工具类 The Tool Class BluetoothUtil, a Standard Tool for Bluetooth Connection

```
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();
    }
}
```

2.2.2. 蓝牙连接打印服务事例 Examples on Connecting Printing Service with Bluetooth

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);
```

2.2.3. 注意事项 Notes

需要在 App 的项目中**添加蓝牙权限声明才能使用蓝牙设备：**

A bluetooth devices can only be used by adding Bluetooth Permission Decalaration to the project in the App:

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

3. HTML 中的 JS　Via JS in HTML

3.1. JS 简介 Description of JS

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）

Essentially, calling a printer via JS is using JS bridge in HTML to operate Android native codes to print, and print via Bluetooth or via AIDL. (SUNMI printer itself is not a network printer, which means that the webpage application can not directly communicate with the printer and an app to receive data on Andriod is needed.)

3.2. HTML 使用 How to Use HTML

以下为具体的调用流程:

The detailed calling processes are as follows:

a. 在 HTML 中定义 android 中要使用的方法。 Define the method to be used in Android in HTML.

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

2.

b. 初始化 WebView. Initialize WebView.

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

3.

c. 初始化打印服务 Initialize Print Service

```
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);
```

4.

d. 给 WebView 添加监听，在 onPageFinished 回调方法中调用 Add Monitor to WebView and Call in the onPageFinished Callback Method

```
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。

Define the operation method in HTML through @JavascriptInterface in the JsObject class, and print a receipt by calling via AIDL print method in the method.

```
//添加一个页面相应监听类 class  Add a corresponding webpage monitor class
mWebView.setWebViewClient(new WebViewClientDemo());
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 The current
    WebView will be used when opening a new link; other system browsers won't be used.
        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 Register JavascriptInterface. You can
    name "lee" whatever you want.
        * 如果你用"lee"，那么在 html 中只要用  lee.方法名() If you use "lee", you only need to
    use lee. method name () in html
        * 即可调用 MyJavascriptInterface 里的同名方法，参数也要一致 And you can call the
method with the same name in MyJavascriptInterface, and the parameters shall be the same.
        */
        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();
        }
    }
}
```

e. 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
Load HTML File, and a Receipt Will be Printed When Clicking the Button in HTML

```
// 载入包含 js 的 html Load the html which contains js
mWebView.loadData("", "text/html", null);
mWebView.loadUrl("file:///android_asset/test.html");//这里是您业务 html 所在的网页
here is the webpage containing your business html
```

6.

附录 A 打印服务广播

Appendix A Printing Service Broadcast

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。

Through broadcast: users need to create a broadcast receiver to monitor broadcasts

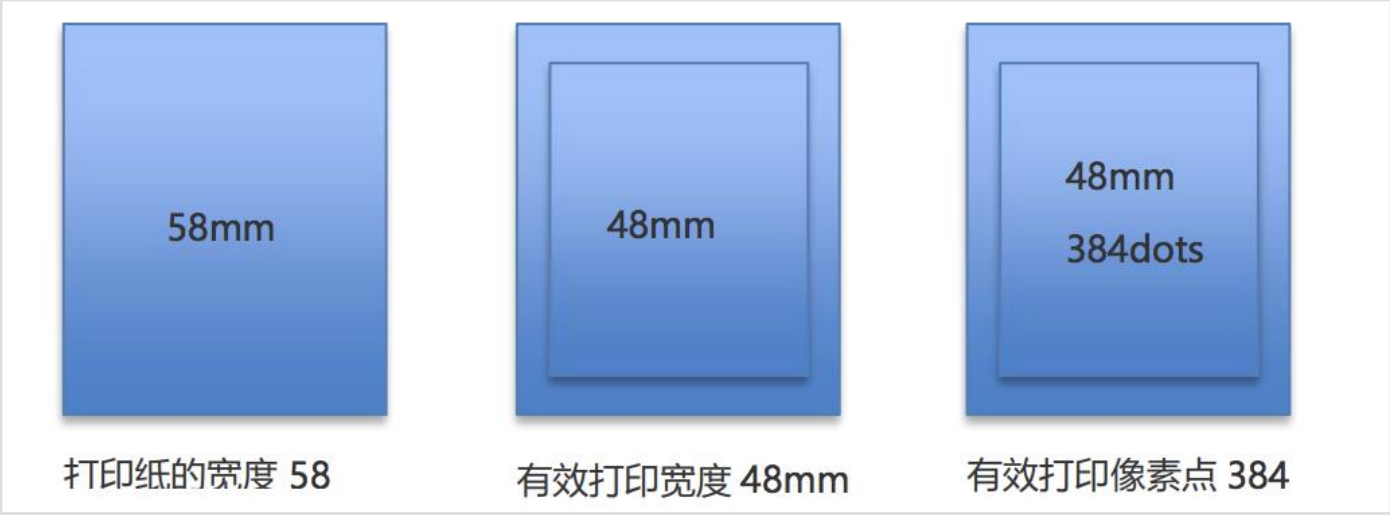
功能 Function	Action
打印机准备中 Preparing the printer	"woyou.aidlservice.jiuv5.INIT_ACTION"
打印机更新中 Updating the printer	"woyou.aidlservice.jiuv5.FIRMWARE_UPDATING_ACITON"
可以打印 Ready to print	"woyou.aidlservice.jiuv5.NORMAL_ACTION"
打印错误 Print error	"woyou.aidlservice.jiuv5.ERROR_ACTION"
缺纸异常 Out of paper	"woyou.aidlservice.jiuv5.OUT_OF_PAPER_ACTION"
打印头过热异常 Overheated printhead	"woyou.aidlservice.jiuv5.OVER_HEATING_ACITON"
打印头温度恢复正常 The temperature of the printhead is back to normal	"woyou.aidlservice.jiuv5.NORMAL_HEATING_ACITON"
开盖子	"woyou.aidlservice.jiuv5.COVER_OPEN_ACTION"

Open the lid	
关盖子异常 Something went wrong when closing the lid	"woyou.aidlservice.jiuv5.COVER_ERROR_ACTION"
切刀异常 1－卡切刀 Cutter error1 - block	"woyou.aidlservice.jiuv5.KNIFE_ERROR_ACTION_1"
切刀异常 2－切刀修复 Cutter error2 - fixed	"woyou.aidlservice.jiuv5.KNIFE_ERROR_ACTION_2"
打印机固件开始升级 The printer firmware starts upgrading	"woyou.aidlservice.jiuv5.FIRMWARE_UPDATING_ACITON"
打印机固件升级失败 Failed to upgrade the printer firmware	"woyou.aidlservice.jiuv5.FIRMWARE_FAILURE_ACITON"
未发现打印机 No printer has been found	"woyou.aidlservice.jiuv5.PRINTER_NON_EXISTENT_ACITON"
未检测到黑标 No black mark has been found	"woyou.aidlservice.jiuv5.BLACKLABEL_NON_EXISTENT_ACITON"

附录 B 打印服务 F&Q

Appendix B Printing Service FAQ

1. 打印纸张规格说明 Description on the Specs of Printing Paper



打印纸宽度 58：Paper width: 58mm

有效打印宽度 48mm：Active print width: 48mm

有效打印像素点 384： Active print pixels: 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；

Note: SUNMI printers support printing papers of 58mm, 80mm width. This document takes the printing paper of 58mm width as an example to illustrate; the spec of paper of 80mm width is similar to the example in this document.

The width of the 58 printing paper is 58mm, and the active print width is 48mm, the active print pixels per line are 384. The depth of V1 paper compartment is 40mm which can accommodate a 40mm-wide paper roll maximum.

2. 打印机分辨率是多少？

2. The Resolution of a Printer

打印机分辨率为 205DPI，计算公式如下

A printer’s resolution is 205DPI. The calculation formula is:

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

3. 如何查询有无打印机？

3. How to Find Whether There Is a Printer?

台式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。

查询接口：

函数： Settings.Global.getInt(getContentResolver(), "sunmi_printer", 0);

参数： 固定值。

返回值：

0 → 无打印机

1 → 有打印机

-1 → 正在查询中

Desktop devices can be divided into two types – with printer hardware or without printer hardware; users can find whether there is a printer through the query interface for printers.

Query interface:

Function: Settings.Global.getInt(getContentResolver(), "sunmi_printer", 0);

Parameter: fixed value.

Return values

0 → there is a printer

1 → there is no printer

-1 → query in process

4. 为什么我执行了打印图片却没有打印出来?

4. Why can’t I print the image even though I execute the image printing?

首先，打印机是行缓冲的，目前机器对于除二维码外指令接口均以满一行打印输出，不满一行缓冲，所以当像调用 printBitmap 接口后发现图片没有打印，很可能是由于图片宽度达不到纸张宽度，没有输出，这个时候只要调用 linewrap 走纸接口即可将缓存的图片打印出来了。

First of all, the content to be printed (except QR code) which cannot fill a line will be saved in the buffer. If you find the image cannot be printed out after calling the printBitmap interface, the reason might be the image width is smaller than the paper width. You can call the linewrap interface to print the buffered image.

但如果仍然没有打印出图片，那么就要检查一下接口的回调，看是否有失败的异常出现，一般情况下可能由于送的图片过大导致接口调用失败，sunmi 打印机支持最大打印图片分辨率大小 2M(非图片实际大小)，最大可显示宽度根据纸张规格决定。所以开发者想要打印合适的图像时需要自行缩放图片大小；

If the method stated above failed, you need to check the callback of the interface to see whether there is an exception. Commonly, an oversized image might be the cause of failing to call the interface. SUNMI printers

support images with a resolution smaller than 2M (not the actual size of an image), and the maximum displayable width depends on the paper spec. As a result of this, developers need to adjust the size of the image to be printed.

另外，如果开发者是通过蓝牙十六进制指令发送光栅位图，导致图片打印失败时，需要自行检查指令是否有错误，此时一个字节数据的错误就有可能影响整条数据的实现；

Additionally, if a developer sends a raster bitmap through Bluetooth hexadecimal instruction and the printing failed, the developer needs to check whether there is a mistake in the instruction, for a byte data error might affect the realization of the whole instruction.

5. 我的条形码内容很长，小票显示不下，我应该选择哪种条码?

5. My bar code is too long to fit in a receipt. Which barcode should I use?

由于打印纸张宽度限制（手持机一般 384 像素点宽、台式机 576 像素点宽），打印位数较多的条码将不能全部显示，这个时候最好先调整条码的宽度设置，看是否能解决问题；

Due to the width limit of a printing paper (handheld device – 384 pixels wide; desktop device – 576 pixels wide), the barcode with pixels exceeds the limit cannot be displayed fully. The width of the barcode should be adjusted first to see whether this problem can be solved.

如果条码内容过于多，超过 20 位，这种情况一般是使用 code128 类型条码，请调用 **printBarCode** 接口选择 code128 码类型，目前此接口实现了混合编码，即可通过增加{A、{B、{C 三种表示动态切换编码类型（A：数字、大写字母、控制字符；B:数字、大小字母、字符;C:双位数字），当是数字时切换为 C 类型，当是其他字符可选择切换为 A、B 类型，即可打印长条码，例如打印 ABab1212：传入 "{BABab{C1212";

If the barcode exceeds 20 digits, code 128 is recommended to use. Please call the interface **printBarCode** and select code128. This interface has realized mixed encoding, which means you can change encoding type by adding {A, {B, {C to dynamically switch encoding types(A: numbers, upper-case letters, and control characters, etc. B: numbers, upper- and lower-case letters and some additional characters. C: numbers with even number of digits). C type will be used when it is a number, and A/B type will be used when it is other characters, then you can print a long barcode for example to print ABab1212: pass "{BABab{C1212";

混合编码需要打印服务 4.0.0 版本以上支持，如果开发者打印服务不支持或使用蓝牙方式打印，那么需要引入下列方法获取一组 epson 指令数组，通过 sendRawData 接口或蓝牙发送返回的数组即可；

Mixed encoding is only available to a version 4.0.0 and above. If it is not available to the developer's print service or the developer prints via Bluetooth, an array of Epson instructions needs to be obtained by introducing the following method, and send the return array through sendRawData interface or Bluetooth.

其中，data 为直接传入需要打印的条码内容即可；width 由于默认的宽度最小值是 2 个像素，当更小时扫码速度严重下降，但为了打印超过 25 甚至更多的条码内容，可以选择设置为 1；

Among which, data is the direct incoming barcode content to be printed; the minimum width value is 2 pixels by default, and a width smaller than this value will drastically slow the code scanning speed. However, to print a barcode exceeding 25 or more, the value can be set as 1.

```
public static byte[] getPrintBarCode(String data, int height, int width, int textposition){
    if(width < 1 || width > 6){width = 2;}
    if(textposition < 0 || textposition > 3){textposition = 0;}
    if(height < 1 || height>255){height = 162;}
    ByteArrayOutputStream buffer = new ByteArrayOutputStream();
    try{
        buffer.write(new byte[]{0x1D,0x66,0x01,0x1D,0x48,(byte)textposition,
                0x1D,0x77,(byte)width,0x1D,0x68,(byte)height,0x0A});
        byte[] barcode = checkCode128Auto(data.getBytes());
        buffer.write(new byte[]{0x1D,0x6B,0x49,(byte)(barcode.length)});
        buffer.write(barcode);
    }catch(Exception e){
        e.printStackTrace();
    }
    return buffer.toByteArray();
}
```

```
    public static byte[] checkCode128Auto(byte[] data){
        ByteArrayOutputStream temp = new ByteArrayOutputStream();
        int pos = 0;
        boolean mode_C = true;
        temp.write(0x7b);
        temp.write(0x43);
        while(pos < data.length){
            if(data[pos] == '{' && pos + 1 != data.length){
                switch (data[pos + 1]){
                    case 'A':
                    case 'B':
                        if(mode_C){mode_C =
false;temp.write(data[pos++]);temp.write(data[pos++]);
                        }else{pos++;pos++;}
                        break;
                    case 'C':
                        if(!mode_C){mode_C =
true;temp.write(data[pos++]);temp.write(data[pos++]);
                        }else{pos++;pos++;}
                        break;
                    default:
                        break;}
            }else if(pos + 1 == data.length){
                if(mode_C){temp.write(0x7b);temp.write(0x42);mode_C = false;}
                temp.write(data[pos++]);
            }else if(data[pos] < '0' || data[pos] > '9'){
                if(mode_C){temp.write(0x7b);temp.write(0x42);mode_C = false;}
                temp.write(data[pos++]);
            } else if(data[pos+1] < '0' || data[pos+1] > '9'){
                if(mode_C){temp.write(0x7b);temp.write(0x42);mode_C = false;}
                temp.write(data[pos++]);
                temp.write(data[pos++]);
            }else{
                if(!mode_C){temp.write(0x7b);temp.write(0x43);mode_C = true;}
                int left = data[pos] - '0';
                int right = data[pos + 1] - '0';
                int num = left*10 + right;
                if(num < 0 || num > 99){
                    return null;
                }else{
                    temp.write(num);
                }
                pos++;
                pos++;
            }
        }
        return temp.toByteArray();
    }
```

P系列和V1S和V2 AIDL文档资源，T系列和S系列AIDL文档资源，Mini系列AIDL文档资源，V系列AIDL文档资源

6. 为什么我收不到回调结果?

6. Why cannot I receive callback results?

AIDL resource document – P series, V1S and V2; AIDL resource document - T series and S series; AIDL resource document - Mini series; AIDL resource document - series:

首先，如果开发者用的是 AIDL 方式接入接口，需要注意是否使用匹配的 AIDL 资源：

First of all, if a developer accesses an interface via AIDL method, attention should be paid on whether the suitable AIDL resource has been used.

在确认资源文件正确的前提下，需要注意当我们创建回调对象时，需要用引入 Aidl 生成的回调类的内部静态类 Stub！

On the premise of using the suitable resource document, please be sure that you have introduced the static nested class Stub! in callback class generated by AIDL introduced.

~~new ICallback(…)~~ —> new ICallback.Stub()

对于使用远程导入库的开发者，只要遵循文档使用已经封装好的 InnerPrinterCallback 等类就不会出现收不到结果的问题了。

This problem won't occur if the developer using remote introduction library follows the document and uses the encapsulated InnerPrinterCallback class, etc.

7. 字符集的选择和设置

7. How to select and set a character set.

背景：由于内置打印机兼容二进制字节流的形式传输，所以通过字节码发送的打印文本内容涉及到字符集的选择和设置；默认的机器设置是多字节、GB18030 字符编码；

Background: the inbuilt printers are compatible with the transmission in a form of binary byte stream, so a character set must be selected and set for the text to be printed which is sent in a form of byte code; multi-byte, GB18030 character encoding are the default device settings.

内置打印机支持的单字节编码类型如下：

The single-byte encoding types for inbuilt printers are:

[参数] [编码] [国家]
[Parameter] [Encoding] [Country]
0 "CP437"; 美国欧洲标准 The USA, European standard
2 "CP850"; 多语言 Multiple languages
3 "CP860"; 葡萄牙语 Portuguese
4 “CP863"; 加拿大 -法语 Canada - French
5 “CP865"; 北欧 Northern Europe
13 "CP857"; 土耳其语 Turkish
14 "CP737"; 希腊语 Greek
15 "CP928";
16 "Windows-1252";
17 "CP866"; 西里尔文 Cyrillic
18 "CP852"; 拉丁-中欧语 Latin - Central European
19 "CP858";
21 "CP874";
33 "Windows-775"; 波罗的海语 Baltic
34 "CP855"; 西里尔文 Cyrillic
36 "CP862"; 希伯来语 Hebrew
37 "CP864";
254 "CP855"; 西里尔文 Cyrillic

内置打印机支持的多字节编码类型如下：

The multiple-byte encoding types for inbuilt printers are:

[参数] [编码]
[Parameter] [Encoding]
0x00 || 0x48 "GB18030";
0x01 || 0x49 "BIG5";
0xFF “utf-8”;

不同的国家可以根据本国或需要修改不同的机器设置使打印机识别接收到的打印内容数据流，比如打印 CP437 页码表的内容需要发送：

The device settings can be adjusted in accordance to the needs of different countries or other requirements to enable the printers to identify the data stream of the content to be printed. To print CP437, please send:

0x1C 0x2E ——设置为单字节编码类型

0x1B 0x74 0x00 ——设置为单字节编码页中 CP437 代码页表

打印 CP866 页码表的内容需要发送：

0x1C 0x2E ——设置为单字节编码类型

0x1B 0x74 0x11 ——设置为单字节编码页中 CP866 代码页表

打印繁体内容需要发送：

0x1C 0x26 ——设置为多字节编码类型

0x1C 0x43 0x01—— 设置为多字节编码页中 BIG5 编码

打印 UTF-8 编码内容（使用 utf-8 编码可以支持所有 unicode 字符集，可以保证打印所有内容）发送：

0x1C 0x26 ——设置为多字节编码类型

0x1C 0x43 0xFF—— 设置为多字节编码页中 UTF8 编码

0x1C 0x2E ——set it as the single-byte encoding type

0x1B 0x74 0x00 ——set it as the CP437 of the single-byte code page

To print CP866, please send:

0x1C 0x2E ——set it as the single-byte encoding type

0x1B 0x74 0x11 ——set it as the CP866 of the single-byte code page

To print traditional characters, please send:

0x1C 0x26 ——set it as the multiple-byte encoding type

0x1C 0x43 0x01—— set it as the BIG5 encoding of the multiple-byte code page

To print UTF-8 encoding content (all Unicode character sets are supported if using UTF-8 encoding, and all content can be printed), please send:

0x1C 0x26 ——set it as the multiple-byte encoding type

0x1C 0x43 0xFF—— Set it as the UTF-8 encoding of the multiple-type code page

8. 黑标打印说明 Description of black mark print

商米 T1 可使用符合规范的黑标打印纸进行打印。

SUNMI T1 can print using the printing paper with black marks which comply to the standard.

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）

Requirements on the printing paper with black marks: different from other black mark printers, SUNMI T1 printer has no sensor to detect black marks but provide a black mark printing function through reflectivity (if the reflectivity is not bigger than 6% (Infrared wave lengths within 700-1000nm), the paper sensor will detect it as no paper). Due to the different theory used, a SUNMI T1 printer in black mark print mode cannot accurately locate the black mark as other black mark printers and there are some errors.

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。

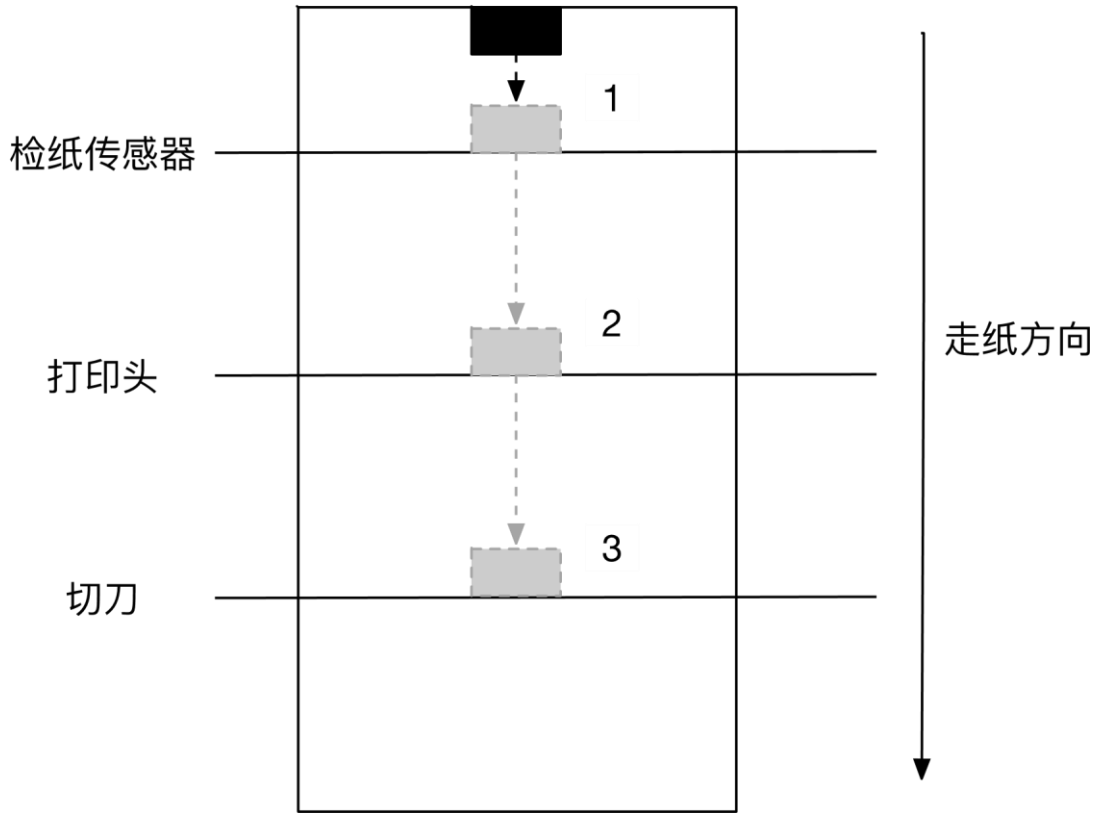
Requirement on the location of the black mark: the black mark should be in the horizontal center for the paper sensor to detect it.

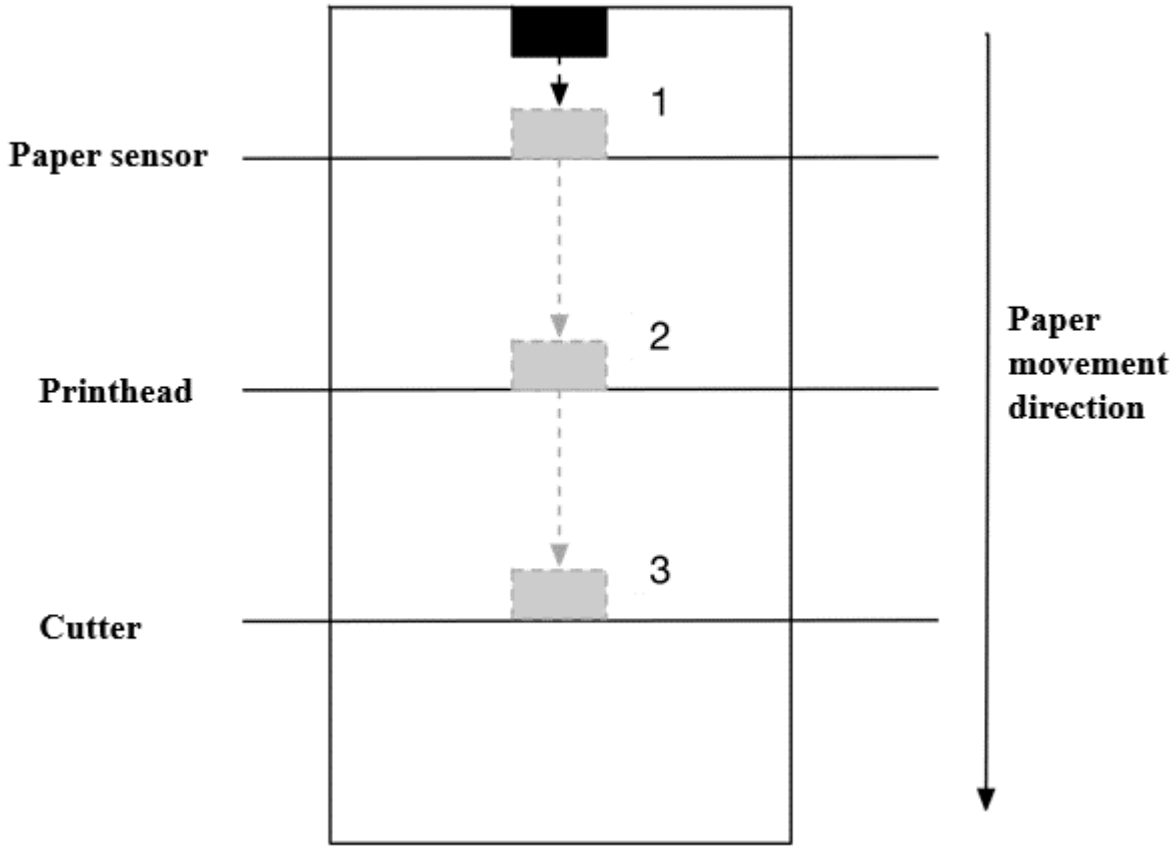
商米 T1 黑标实现原理：

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。

The theory SUNMI T1 used to realize black mark print:

The paper sensor and the cutter are not in the same horizontal line, and the black mark on the paper will first run through the paper sensor (the location of 1) and then run through the printhead (the location of 2) and finally arrive the location of the cutter (the location of 3).d





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。

In black mark print mode, if the printer detects that there is no paper, it will keeping moving 7mm. After moving 7mm, if no paper has been detected, the printer will process it as no paper, and if paper is detected after moving 7mm, the printer will process it as black mark print.

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。

If the content to be printed covers the black mark, the printing will be proceeded, which means if the black mark arrives at the location of 2 and there is still a print job, the print job will continue till it is completed.

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。

In this mechanism, the end of content to be printed should keep a certain distance with paper sensor (the end of content should keep a distance of 6mm with the black mark edge) to ensure that the content is between 2 black marks.

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{0x1d, 0x56, 0x00}

这样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。

Instruction sequence in the black mark print mode:

1. send the content to be printed
2. enter the instruction on moving the paper till the black mark has been detected {0x1c, 0x28, 0x4c, 0x02, 0x00, 0x42, 0x31}
3. enter the cutter instruction {0x1d, 0x56, 0x00}

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。

Users can change the black mark print mode and the location of cutter in Setting->Print->Inbuilt print management.