

苹果标签打印机集成 SDK

简介:

本 SDK 已简化，去掉多余代码，封装了.a 静态库，已尽量通过引用最少代码文件来达到集成的目的。

标签打印机采用打印机业内通用的 TSPL 指令，SDK 封装了蓝牙发现，连接，发送数据，通用的 TSPL 指令（文字，条形码，二维码），SDK 集成分 3 步。

第一步:

引入两个头文件 BluetoothManager.h、PrintfTSPLManager.h 和一个静态库 libBluetoothManager.a，如下图，把三个文件拷贝到项目下



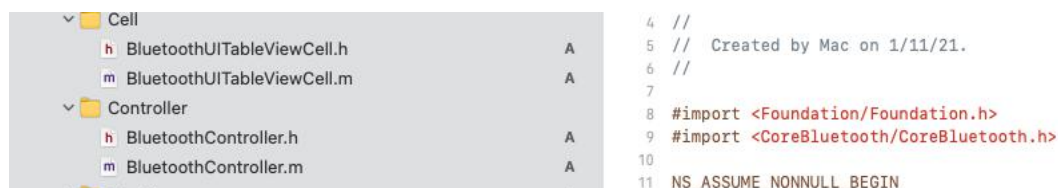
第二步:

第二步是打印机蓝牙设备“发现（搜索）”，“连

接”，“发送数据”的功能集成。

1) 涉及到 **BluetoothManager.h** 头文件, 里面封装了蓝牙发现, 发现设备回调, 蓝牙连接, 连接结果回调, 发送数据等, 详情看头文件方法的注释

2) 拷贝四个文件到项目中 **BluetoothController.h**,
BluetoothController.m,
BluetoothUITableViewController.h,
BluetoothUITableViewController.m, 这 4 个的主要作用是, 用于发现打印机蓝牙设备, 并进行连接用于数据传输, 如下图



3) 可在需要的页面跳转到 **BluetoothController** 进行蓝牙连接, 可参照 **demo** 进行跳转

第三步:

在上一步中, 已经与打印机建立了连接, 下面开始发送数据给打印机。常用的 **TSPL** 指令已经封装, 打印文字、条形码, 二维码等封装好了方法。

1) 涉及 **PrintfTSPLManager.h** 头文件, 该头文件封装了常用的 **TSPL** 指令

2) 先初始化一个可变 **Data** 数组，用于存放指令

```
NSMutableData *instructionData = [[NSMutableData alloc] initWithCapacity:5];
```

3) 清除打印机画板缓存

```
//清除画板缓存  
[instructionData appendData:[PrintfTSPLManager clearCanvas]];
```

4) 初始化标签纸大小（标签宽和高，单位 mm）

```
[instructionData appendData:[PrintfTSPLManager initCanvas:60 height:30]];
```

5) 打印方向设置（参数值为 0 和 1，一正一反）

```
[instructionData appendData:[PrintfTSPLManager setDirection:1]];
```

上面是标签设置指令，下面是打印内容指令

6) 打印文字，文字的坐标定位采用 xy 坐标轴，坐标在标签纸上的计算单位是，1mm=8px，参数定义，请看方法注释

```
[instructionData appendData:[PrintfTSPLManager printText:0 y:0 fontWidth:1 fontHeight:1 content:@"打印标签文字测试ABCDE"]];
```

7) 打印条形码

```
[instructionData appendData:[PrintfTSPLManager printBarCode:0 y:0 codeType:@"128" height:106 barcodeTextAlign:2 rotation:0 content:@"11043001"]];
```

8) 打印二维码

```
[instructionData appendData:[PrintfTSPLManager printQRCode:0 y:0 level:@"H" width:1 rotation:0 content:@"测试的二维码的内容 1"]];
```

9) 开始打印，参数是打印张数

```
[instructionData appendData:[PrintfTSPLManager beginPrint:1]];
```

10) 发送给打印机

```
[blueToothManager writeNSData:instructionData];
```

11) 拼接起来，就是一个标签

```
NSMutableData *instructionData = [[NSMutableData alloc]
initWithCapacity:5];
//清除画板缓存
[instructionData appendData:[PrintfTSPLManager clearCanvas]];
//初始化标签纸大小（标签宽和高，单位 mm）
[instructionData appendData:[PrintfTSPLManager initCanvas:60
height:30]];
//打印方向设置（参数值为 0 和 1，一正一反）
[instructionData appendData:[PrintfTSPLManager
setDirection:1]];
//打印文字（参数说明，请看方法注释）
[instructionData appendData:[PrintfTSPLManager printText:0 y:0
fontWidth:1 fontHeight:1 content:@"打印标签文字测试 ABCDE"]];
//开始打印（参数是打印的张数）
[instructionData appendData:[PrintfTSPLManager beginPrint:1]];
//发送数据给打印机
if([blueToothManager writeNSData:instructionData]){
    NSLog(@"正在打印...");
}
```

第三步也已经完成了，可以正常与打印机进行通信了，感谢您看到这里，SDK 文档书写过程中难免有错漏，如遇开发问题，可联系我们，再次感谢

附录一：常用的 TSPL 指令

1) SIZE 指令：

功能：定义标签纸的宽度及高度

语法：SIZE m mm, n mm

参数说明：m 定义标签纸的宽度(毫米)

n 定义标签纸的高度(毫米)

2) CLS 指令：

功能：清除数据缓存

语法：CLS

3) DIRECTION 指令：

功能：设定打印方向

语法: DIRECTION n

参数说明: n 0 或 1, 0 是正向, 1 是反向

4) PRINT 指令

功能: 将存于数据缓存的标签打印

语法: PRINT m [,n]

参数说明: m 打印张数

5) BARCODE 指令

功能: 打印条形码

语法: BARCODE X,Y,"code type",height,human
readable,rotation,narrow,wide,[alignment,]"content"

参数说明: X 明定条码左上角 X 坐标

Y 明定条码左上角 Y 坐标

其他参数, 有点多, 就不详述了, 见另一个 TSPL 文档第 48 页

6) QRCODE 指令

功能: 打印二维码

语法: QRCODE x,y,ECC Level,cell width,mode,rotation,[model,mask,]"content"

参数说明: X 明定条码左上角 X 坐标

Y 明定条码左上角 Y 坐标

其他参数, 有点多, 就不详述了, 见另一个 TSPL 文档第 74 页

7) TEXT 指令

功能: 打印文字

语法: TEXT x,y,"font",rotation,x-multiplication,y-multiplication,[alignment,]"content"

参数说明: X 明定条码左上角 X 坐标

Y 明定条码左上角 Y 坐标

其他参数, 有点多, 就不详述了, 见另一个 TSPL 文档第 74 页

8) BITMAP 指令:

功能: 打印图片

语法: BITMAP X, Y, width, height, mode, bitmap data,,

参数说明:

X 明定图形左上角 X 坐标

Y 明定图形左上角 Y 坐标

Width 图形的宽度, 单位为 byte

Height 图形的高度, 单位为 dot

Mode 绘制图形的方式

0 OVERWRITE

1 OR

2 XOR

bitmap data 图形资料

好的, TSPL 最常用的几个指令已经介绍完了, 来看几个例子。

例子 1: 打印文字,

```
SIZE 50 mm,30 mm
DIRECTION 1
CLS
TEXT 0,50,"0",0,1,1,"打印标签，文字测试文字"
PRINT 1,1
```

例子 2: 打印条码

```
SIZE 72 mm,30 mm
DIRECTION 1
CLS
BARCODE 0,0,"128",106,0,0,3,3,"11043001"
PRINT 1,1
```

例子 3: 打印二维码

```
SIZE 72 mm,30 mm
DIRECTION 1
CLS
TEXT 0,50,"0",0,1,1,"打印标签 abc_12342"
QRCODE 0,80,H,2,M,0,M1,S1,"测试的二维码的内容 1"
PRINT 1,1
```

附录二：重要头文件的定义

BluetoothManager.h: 蓝牙发现，连接，数据传输类

方法:

1. 获取单个实例: `+(instancetype)bluetoothManagerInstance`
2. 扫描到设备的回调: `-(void)blePeripheralFound:(void (^)(CBPeripheral *peripheral))block;`
3. 连接蓝牙设备成功的回调: `-(void)blePeripheralConnected:(void (^)(CBCentralManager *central,CBPeripheral *peripheral))block;`
4. 蓝牙设备断开的回调: `-(void)blePeripheralDisonnected:(void (^)(CBCentralManager *central,CBPeripheral *peripheral))block;`
5. 开始搜索蓝牙: `-(void)beginScan;`
6. 停止扫描: `-(void)stopScan;`
7. 连接蓝牙: `-(void)connect:(CBPeripheral *)peripheral;`

参数说明: peripheral 的外设

8. 向蓝牙发送 Data 数据: `-(BOOL)writeNSData:(NSData *)nsData;`

参数说明: nsData 发送的数据 NSData

9. 向蓝牙发送数据并接收回传数据: `-(void)writeAndRead:(NSData *)nsData
readNSData:(ReadNSData) readNSData;`

参数说明: nsData 发送的数据 NSData

readNSData 返回的数据 NSData

10. 得到当前的蓝牙名称: `-(NSString *)getCurrentName;`

PrintTSPLManager.h: 常用 TSPL 指令封装

方法:

1. 初始化标签大小: `+(NSData *)initCanvas:(int)width
height:(int)height;`

参数说明: width 标签宽度 单位 mm height 标签高度 单位 mm

2. 清除画板缓存: `+(NSData *)clearCanvas;`

其他说明: 每打一个不同的标签, 都需要清除画板, 不然会有上一个标签的内容出来

3. 打印文字: `+(NSData *)printText:(int)x y:(int)y
fontWidth:(int)fontWidth fontHeight:(int)fontHeight
content:(NSString *)content;`

参数说明: x 文字 x 坐标

y 文字 y 坐标

fontWidth 文字宽度 (1-10)

fontHeight 文字高度 (1-10)

content 文字内容

其他说明: 文字的宽度和高度范围是 1 到 10, 整数

4. 打印条形码: `+(NSData *)printBarCode:(int)x y:(int)y
codeType:(NSString *)barCodeType height:(int)height
barcodeTextAlign:(int)textAlign rotation:(int)rotation
content:(NSString *)content;`

参数说明: x 文字 x 坐标

Y 文字 y 坐标

codeType 条形码类型:

CODE128: "128"

UPCA: "UPCA"

EAN13: "EAN13"

EAN8: "EAN8"

CODE39: “39”

CODABAR: “CODA”

height: 条形码高度(dots)

barcodeTextAlign: 条形码文字位置

NONE: 0

LEFT: 1

CENTER: 2

RIGHT: 3

rotation: 旋转角度 (0, 90, 180, 270)

content: 条形码内容

其他说明: 条形码的高度是点数, 200dpi 打印机按照 1mm=8dot 来计算

5. 打印二维码: +(NSData *)printQRCode:(int)x y:(int)y level:(NSString *)level width:(int)width rotation:(int)rotation content:(NSString *)content;

参数说明: x 二维码 x 坐标

Y 二维码 y 坐标

level 容错率 (“L”, “M”, “Q”, “H”)

width 二维码宽度 (1-10)

rotation 旋转角度 (0, 90, 180, 270)

content 二维码内容

6. 打印图片: +(NSData *)printfUIImage:(int)x y:(int)y UIImage:(UIImage *)image concentration:(int)concentration;

参数说明: x 图片 x 坐标

Y 图片 y 坐标

UIImage 图片

concentration 阈值 (0-255) 默认 128 就行

其他说明: 阈值默认是 128, 值越大越黑

7. 开始打印: +(NSData *)beginPrintf:(int)number;

参数说明: number 打印的张数