HPRTPrinter SDK 介绍

HPRTPinter SDK 是厦门汉印电子技术为其生产的打印机产品推出的 iOS 开发组件。客户可以使用 SDK 实现打印指令的生成以及打印机通讯。SDK 采用.framework 的形式。使用方法和系统动态框架一样。

iOS中.a与.framework有什么区别?

> - 首先用户创建的.a 和.framework 都是静态库。
> - .a是一个纯二进制文件，不能直接使用，需要.h文件配合。
> - .framework中除了有二进制文件之外还有资源文件和.h 头文件，可以直接使用。

支持的指令集：

> - ESC-POS
> - TSC-TSPL

支持的通讯方式：

> - 蓝牙4.0 (BLE)
> - WiFi。

支持的打印机：

> 同时满足通讯方式和指令集两个条件的打印机都会支持。

主要的类介绍：

- HPRTDispatcher实现和打印机的通讯（BLE、WiFi）。

  Wifi 连接接口：

```
#pragma mark HPRTNetWork

// 获取当前 Wifi 的路由信息
- (NSDictionary *)getRouterInfo;

// 通过 ip 地址连接 wifi
- (void)connectWifi:(NSString *)ip;
```

蓝牙4.0 BLE接口：

```
#pragma mark HPRTBluetooth

// 扫描蓝牙
- (void)scanBluetooth;

// 停止扫描蓝牙
- (void)stopScanBluetooth;

// 连接蓝牙
- (void)connectBluetooth:(CBPeripheral *)peripheral;

// 断开连接蓝牙
- (void)disConnectBluetooth;
```

发送数据给打印机打印：

```
// 发送一条数据打印出来
- (void)sendData:(NSData *)data;

// 打印队列，将需要的一组数据(NSData)加入 NSArray,将会依次打印出来。
- (void)sendDataQueue:(NSArray *)dataQueue;
```

断开当前连接：

```
#pragma mark CurrentConnect

// 取消当前连接（Wifi 或者 BLE）
- (void)disConnectCurrentLine;
```

打印机的连接状态：

```
typedef NS_ENUM(NSInteger, DispatchMode) {

    DispatchModeOffline = 0, //离线模式
    DispatchModeBLE,         //蓝牙模式
    DispatchModeWIFI,        //WiFi 模式
};

// 当前通讯调度模式
@property(nonatomic,assign) NSInteger currentDispatchMode;

// 当前连接的蓝牙
@property(strong,nonatomic,readwrite) CBPeripheral
*bluetoothConnected;
```

HPRTDispatcher 协议 `HPRTDispatchProtocol`

> - 同时支持协议和通知两种方式来获取连接状态。
> - 使用协议比使用通知在代码结构上更优雅。

```
@protocol HPRTDispatchProtocol <NSObject>

@optional
/**
 *  每发现一个蓝牙设备调用一次 或者观察通知:
PRTBluetoothUpdatePeripheralList（携带数据）
 */
- (void)didDiscoverPeripheral:(CBPeripheral *)peripheral RSSI:
(NSNumber *)RSSI;

/**
 *  成功连接 BLE 或者观察通知：HPRTDispatchModeChangeToBLE
 */
- (void)didConnectPeripheral:(CBPeripheral *)peripheral;

/**
 *  成功连接 WiFi 或者观察通知：HPRTDispatchModeChangeToWIFI
 */
- (void)didConnectWiFi:(NSString *)ip;
```

- TSC-TSPL 指令集：

> 广泛应用在标签打印机上。SDK 中大约有60多条生成指令的函数。

一个典型的 TSC-TSPL 指令生成流程如下：

```
HPRTPrinterTSC *tsc = [[HPRTPrinterTSC alloc] init];

[tsc initCMDQueue]; //初始化命令队列(必要)

[tsc setPrintAreaSizeWithWidth:@"100" Height:@"80"];     //设置打印区域
[tsc setCLS];    //清除缓冲区
[tsc printTexTWithXPos:.....]; //打印文本
[tsc printBarCodeWith........]; //打印条形码

[tsc sendCMDQueue];//发送队列中的指令到打印机(必要)

// NSArray *cmdQueue = [tsc geenrateCMDQueue];或者生成指令，用其他方式发送打印

/*
必须在设置指令前初始化命令队列(initCMDQueue)，设置所有指令后发送指令队列(sendCMDQueue)。
*/
```

- ESC-POS 指令集：

  > 广泛应用在票据打印机上。SDK 中大约有200条生成指令的函数。

  一个典型的 ESC-POS 指令生成流程如下：

  > 可以作为如何将基础指令封装成一个可以直接调用的功能的参考。

```
/**
 *  打印条形码
 *
 *  @param justification    page 165. 对齐方式        取值0,1,2 : 左、中、右
 *  @param type             page 339. 取值：(A:0~6),(B:65~73)
 *  @param width            page 355. 取值：(2~6),(68~76)
 *  @param height           page 337. 取值：(1~255)
 *  @param hri_position     page 332. 取值：(0~3),(48~51)
 *  @param barcode          page 339.
 *
 *  @return 生成的条形码十六进制数据
 */
- (NSData *)escPrintBarCodeWithJustification:(int)justification
                                        Type:(int)type
                                       Width:(int)width
                                      Height:(int)height
                                 HRIPosition:(int)hri_position
                                     Barcode:(NSString *)barcode
{
    //以此为例，演示如何对 ESC-POS基础指令封装，形成功能
    HPRTCommandESC *escCommand = [[HPRTCommandESC alloc] init]; //创建对象

    [escCommand initCMDQueue];  //初始化存放指令的数组

    /**
     *  生成具体的指令，一般包括但不限定：对齐方式，打印起点，打印范围，二维码和图片宽度和高度等等。
     *  每一条指令都会加入指令数组中去。
     */
    [escCommand escSelectJustification:justification];
    [escCommand escSelectPrintPositionOfHRICharacers:hri_position];
    [escCommand escSetBarCodeWidth:width];
    [escCommand escSetBarCodeHeight:height];
    [escCommand escPrintBarCodeWithType:type Barcode:barcode];

    return [escCommand generateCMDData];    //从指令数组中取出命令，可以直接通过蓝牙或者 WiFi 发送出去，就可以打印一条二维码
}
```