ROBOTPEN 鲁伯特

iOS RobotPenManagerSDK

参考手册

1.RobotPenManagerSDK 下载 3

2.iOS RobotPenManagerSDK 目录结构 3

3.配置 SDK 环境 4

4.应用 5

1、扫描、连接设备 6

2、自动连接与配对 8

3、点数据上报 10

4、同步笔记 12

5、OTA 升级 14

6、OTA 外部升级 16

7、页码识别设备专用 18

简介：

iOS RobotPenManagerSDK 封装了罗博智能笔和电磁板相关的 API，应用只需准守协议，实现相应的代理方法即可获取到电磁本、电磁笔及电磁的相关数据，实现与设备通讯。

笔服务 SDK 功能：

1、 连接设备、连接设备。

2、 自动连接与配对。

3、 点数据上报。

4、 同步离线笔记。

5、 OTA 升级。

6、 OTA 外部升级。

7、 页码识别设备专用。

1.RobotPenManagerSDK 下载

请到 https://github.com/PPWrite/SDK 页面下载最新版本的 iOS RobotPenManagerSDK。

2.iOS RobotPenManagerSDK 目录结构

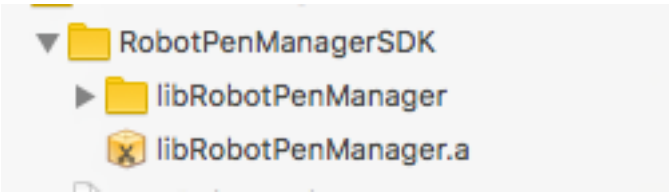
1. libRobotPenManager.a 封装了 iOS RobotPenManagerSDK 功能实现。

2. libRobotPenManager 文件包含了相关的头文件。

3. 配置 SDK 环境

1. 将 iOS SDK 中的 libRobotPenManager.a 和 libRobotPenManager 文件拷贝到应用开发的目录下。

2.将 libRobotPenManager.a 和 libRobotPenManager 文件添加到工程中。



3. 添加 SDK 依赖的系统库文件。

本 SDK 的依赖库只需添加 libsqlite3.tdb（libsqlite3. dylib）或 libsqlite3.0.tdb（libsqlite3.0. dylib ）。

4.工程设置。

PROJECT ->TAGETS->Build Settings-> Linking-> Other Linker Flags 设置成 -ObjC。

注：iOS RobotPenManagerSDK 需在 iOS8.0 及以上系统运行。

4. 应用

笔服务 SDK 主要的功能包括以下功能：

1、 扫描、连接设备

2、 自动连接与配对

3、 点数据上报

4、 同步笔记

5、 OTA 升级

以下全部功能的前提是先遵守 RobotPenDelegate 代理，此代理在同一页面只需遵守一次。

1.遵守 RobotPenDelegate 代理

```
[[RobotPenManager sharePenManager] setPenDelegate:self];
```

2.实现 RobotPenDelegate 中的代理方法即可获取相应的数据信息。详见 RobotPenManagerDemo。

电磁板有两种连接模式：1、配对模式（红蓝闪）2、连接模式（蓝闪）

1、配对模式，所有手机设备都可以连接该设备。

2、连接模式，只有之前连接过该电磁板的手机才能连接该设备。

注：已连接状态（蓝灯常亮），不可被搜索到和再次连接。

如需连接或搜索，需要先断开已连接设备。

如果使用单一类型的设备，建议调用 setDeviceType 方法设置一个默认设备，可以防止由于硬件号导致的 SDK 部分功能无法使用。

1、扫描、连接设备

1.调用 scanDevice 方法扫描设备。

2.通过协议方法 getBufferDevice 监听扫描到的设备模型。

3.调用 connectDevice 方法连接设备。

4.通过代理方法 getDeviceState（）获取设备连接的状态。

5.调用 getConnectDevice 可获得连接的设备模型获取设备数据。

6. 调用 disconnectDevice 方法即可断开连接。

代码实现：

```
//搜索设备
[[RobotPenManager sharePenManager] scanDevice];
//发现设备监听方法
- (void)getBufferDevice:(RobotPenDevice *)device{
    // device即为被发现的设备模型可通过连接方法连接。}
//连接设备
[[RobotPenManager sharePenManager] connectDevice:
RobotPenDevice];
//获取设备连接状态
- (void)getDeviceState:(DeviceState)State;
//获取设备信息
[[RobotPenManager sharePenManager] getConnectDevice];
```

注：连接设备成功后自动停止扫描周边设备。如有需要也可手动调用 stopScanDevice 方法停止扫描设备。

以下枚举类型为设备状态的部分枚举值。

```
typedef enum {
    /**设备已连接（已经存在连接设备）*/
    DEVICE_CONNECTED = 0 ,
    /**正在连接*/
    DEVICE_CONNECTING,
    /**连接成功*/
    DEVICE_CONNECTE_SUCCESS,
    /**连接错误*/
    DEVICE_CONNECT_FAIL,
    /**已断开*/
    DEVICE_DISCONNECTED,
    /**服务准备完成*/
    DEVICE_SERVICES_READY = 5,
    /**笔初始化完成*/
    DEVICE_INIT_COMPLETE,
    /**设备信息获取完成*/
    DEVICE_INFO_END,
}DeviceState;
```

其中 DEVICE_CONNECTE_SUCCESS 为蓝牙服务连接成功。

DEVICE_INFO_END 表示设备信息获取完成，在此之后通过 getConnectDevice 方法即可获取到设备模型。

2、自动连接与配对

自动连接：上一次连接过得设备，打开设备进入连接状态(蓝闪)后，可以自动连接。

配对：可以记录已经连接过得设备，打开自动连接后可自动连接。

自动连接：

1. 调用 setAutoCheckDeviceConnect 打开自动连接设备。

2. 调用 AutoCheckDeviceConnect 自动连接已配对设备。

注：自动连接功能默认开启，自动连接的设备为配对列表中的最后一个（上次连接过的）设备。

代码实现如下：

```
//打开自动连接（默认打开）
[[RobotPenManager sharePenManager]
setAutoCheckDeviceConnect:YES];
//自动连接
[[RobotPenManager sharePenManager] AutoCheckDeviceConnect];
```

配对：

设备连接成功后会自动添加到设备配对列表中。

配对功能方法如下：

1、检查是否有配对设备，调用 checkIsHaveMatch 方法获取。

2、获取配对设备列表，调用 getPairingDevice 方法获取配对列表。

3、取消当前配对，调用 deleteConnect 方法删除。

4、清空配对列表，调用 cleanAllPairingDevice 方法可清空配对列表。

5、为指定设备配对，调用 savePairingMacDevice 方法可为指定设备进行配对。

6、删除指定已配对设备，调用 deletePairingMacDevice 方法可以删除指定的已配对设备。

3、点数据上报

1. 获取原始点上报信息。（适用于使用白板服务和笔服务）

```
- (void)getPointInfo:(RobotPenPoint *)point{

    //point 即为原始点数据模型。
}
```

2. 获取优化点上报信息。（适用于只用笔服务，如果需要获取原始点数据请用 1 方法）

```
- (void)getOptimizesPointInfo:(RobotPenUtilPoint *)point{

    //point 即为优化点数据模型。
}
```

以下为优化点获取设置：

```
//SDK方法 设置报点类型，此处设置为优化点报点

    [[RobotPenManager sharePenManager] setOrigina:NO
optimize:YES transform:NO]

    //SDK方法 设置场景尺寸

[[RobotPenManager sharePenManager]

setSceneSizeWithWidth:lineViewWidth

andHeight:setLineViewHeight andIsHorizontal:setHorizontal];

 //SDK方法 设置笔迹宽度，isOptimize = YES时必须设置，即显示
宽度

[[RobotPenManager sharePenManager]

setStrokeWidth:setLineWidth];
```

设置完成后实现 2 方法即可获取优化点上报数据。

以下为原始点数据结构：

```
@property (assign , nonatomic) short originalX;
@property (assign , nonatomic) short originalY;
@property (assign , nonatomic) short pressure;
@property (assign , nonatomic) DeviceType deviceType;
@property (assign , nonatomic) RobotPenPointTouchStatus
touchState;
```

原始点数据为电磁板上报点数据。

originalX，originalY 为坐标点的 X 与 Y，单位为像素。

Pressure 为电磁笔压力值。

deviceType 为电磁板类型，可查看 DeviceType 枚举。

touchState 为状态标识。

```
typedef enum {
    RobotPenPointFloat = 0,      /** 离开(悬浮)状态 **/
    RobotPenPointTouchBegin,     /** touchBegin状态 **/
    RobotPenPointTouchMove,      /** touchMove状态 **/
    RobotPenPointTouchEnd,       /** touchEnd状态 **/
    RobotPenPointLeave           /** 离开感应范围 **/
}RobotPenPointTouchStatus;
```

注：RobotPenUtilPoint 模型中的 width 为笔迹宽度，speed 为优化点当前速度。

4、同步笔记

有无离线笔记判断：

通过[[RobotPenManager sharePenManager] getConnectDevice]获取到设备信息模型中的离线笔记数量 NoteNumber 判断，大于 0 表示有离线笔记。

1. 调用 startSyncNote 方法可以开始同步笔记。
2. 通过协议方法 SyncState（）获取同步状态。
3. 通过协议方法 getStorageNum（）获取笔记数量。
4. 调用 stopSyncNote 方法，可以中途停止同步。

代码实现：

```
//开始同步笔记
[[RobotPenManager sharePenManager] startSyncNote];
// 监听同步状态
- (void)SyncState:(SYNCState)state;
//监听离线笔记数量变化
- (void)getStorageNum:(int)num andBattery:(int)battery;
//中途停止同步笔记
[[RobotPenManager sharePenManager] stopSyncNote];
```

注：离线笔记同步过程中，中断、停止、同步错误都不会对正在同步的笔记产生影响，再次同步会从该笔记的起始点开始同步。

调用 startSyncNote 后，同步完成或者错误自动退出同步模式。也可手动调用 stopSyncNote 停止同步。

getStorageNum：此监听方法在每次离线笔记数量变化或电量变化时返回离线笔记数和电量。

SYNCState 枚举如下：

```
typedef enum {
    /** 同步错误*/
    SYNC_ERROR,
    /** 有未同步笔记*/
    SYNC_NOTE,
    /** 没有未同步笔记*/
    SYNC_NO_NOTE,
    /** 单个笔记同步成功*/
    SYNC_SUCCESS,
    /** 开始同步*/
    SYNC_START,
    /** 停止同步*/
    SYNC_STOP,
    /** 全部笔记同步完成*/
    SYNC_COMPLETE,
}SYNCState;
```

同步状态中的 SYNC_SUCCESS 表示一本笔记同步完成，如果有有 n 本笔记，则应返回 n 次 SYNC_SUCCESS 状态。SYNC_COMPLETE 表示全部笔记同步完成，只返回一次。

5、OTA 升级

连接设备后通过监听方法 getDeviceState 获取设备信息，

当 getDeviceState 接收到 DEVICE_UPDATE 状态（检查设备更新）时，此时需要手动调用 getIsNeedUpdate 方法查询是否需要升级，查询结束后通过 getDeviceState 返回 DEVICE_UPDATE_CAN（设备可更新）或 DEVICE_UPDATE_CANT（设备不可更新）。

当返回为 DEVICE_UPDATE_CAN 时可调用 startOTA 开始升级。

注：getIsNeedUpdate 方法为网络请求，可调用 cancelSession 方法取消请求。

startOTA 可在返回 DEVICE_UPDATE_CAN 状态后，需要调用的时候调用，不需要立即调用。

1、 调用 startOTA 开始升级。

2、 实现监听方法 OTAUpdateState 可以监听升级状态。

3、 实现监听方法 OTAUpdateProgress 可以监听升级进度。

4、 调用 ExitOTA 可退出升级。

代码实现：

```
  //OTA 升级
[[RobotPenManager sharePenManager] startOTA];
//OTA 升级状态  具体状态说明：请查看 OTAState
- (void)OTAUpdateState:(OTAState)state
//OTA 升级进度
- (void)OTAUpdateProgress:(float)progress
```

以下为升级状态枚举

```
typedef enum {
    /** OTA升级错误*/
    OTA_ERROR,
    /** OTA数据传输*/
    OTA_DATA,
    /** OTA升级*/
    OTA_UPDATE,
    /** OTA成功*/
    OTA_SUCCESS,
    /** 复位*/
    OTA_RESET,
    /** 单个升级成功*/
    OTA_ONE_SUCCESS,
    /** 低电不能升级*/
    OTA_STATUS_ERROR,
}OTAState;
```

其中 OTA_ONE_SUCCESS 表示单固件升级成功，适用于双固件升级。OTA_SUCCESS 表示全部固件升级成功。若为单固件设备升级成功则只会返回 OTA_SUCCESS。双固件设备升级，升级成功一个固件会先返回 OTA_ONE_SUCCESS，全部升级完成会返回 OTA_SUCCESS。若双固件设备只需升级一个固件按单固件处理。

6、OTA 外部升级

OTA 外部升级适用于固件升级包放在外部服务器上的用户。

1）直接设置下载地址

按照 5、OTA 内部升级流程，将调用 getIsNeedUpdate 方法位置替换成调用设置外部服务器方法 getIsNeedUpdateWithURL 即可设置固件检查更新及下载地址即可。

注：该方法只是设置固件服务器请求地址，要求服务上数据包的名称和格式完全按照 Robot 固件包的名称，由 SDK 内部进行下载、判断更新等操作。

如果不设置默认为 Robot 服务器接口。

2）自行下载数据包，通过 SDK 升级。

1、调用设置升级数据包外接方法 setOtherUpdateAddress，才可调用其他外部固件升级方法。

2、设置固件版本号 setHardWareUpdateVerSionWithBLEVersion，设置将要升级的 BLE 和 MCU 版本号。

3、调用设置升级包数据方法 setHardWareUpdateDataWithBLEData 设置升级数据。

4、调用 startOTA 开始升级，后续可参照 5、OTA 内部升级。

注：该方法需自行控制版本号和升级包数据，如果版本号或升级包数据设置错误可能导致升级失败甚至电磁板无法使用。

setHardWareUpdateVerSionWithBLEVersion 方法为设置将要升级的 BLE 及 MCU 版本号而非当前固件版本号。

BLE 版本号为固件版本号的后两位整数，MCU 版本号为前两位，

例：设备固件版本号（SWStr）为 0.10.0.20，其中 0.10 为 MCU 版本号，0 为大版本号，10 位小版本号，0.20 为 BLE 版本号，0 为大版本号，20 为小版本号。由于当前版本号的大版本号 0 暂不定义，所以只用到小版本号，即 MCU 版本号为 10，BLE 版本号为 20。故 setHardWareUpdateVerSionWithBLEVersion 入参分别是 20，10。

若只需升级一个固件则另一个参数传入 0 即可。

固件包数据同理。

代码实现如下：

```
//开启外部升级数据
[[RobotPenManager sharePenManager]
setOtherUpdateAddress:YES];
//设置固件升级版本号
[[RobotPenManager sharePenManager]
setHardWareUpdateVerSionWithBLEVersion:BLEVersion
andMCUVersion:MCUVersion];
//设置固件升数据包数据
[[RobotPenManager sharePenManager]
setHardWareUpdateDataWithBLEData:BLEdata
andMCUData:MCUdata];
//开始固件升级
[[RobotPenManager sharePenManager] startOTA];
```

7、页码识别设备专用

带有页码识别的设备其他功能和上述的一样。

下列方法为带有页码识别的设备专用接口。

1、 实现获取笔记编号和页码编号方法 getDevicePage 即可获取设备识别的笔记编号和页码编号。

2、 实现获取笔记和页码合并编号方法 getDevicePageNoteIdNumber 即可获取设备识别的笔记和页码的合并编号。

3、 实现获取同步笔记信息和页码信息方法 getTASyncNote 即可获取同步离线笔记信息。

4、 调用获取页码识别设备页码信息专用方法 getTAPageInfo 可从 getDevicePage 和 getDevicePageNoteIdNumber 监听方法获得页码信息。

注：页码信息连接设备后会主动上报一次。

更换页码时会主动上报一次。

调用 getTAPageInfo 方法，可以主动获取页码信息。