维亿魄VeepooBleSDK iOS版接口使用说明v2.1.13.0

维亿魄VeepooBleSDK iOS版接口使用说明v2.0.13.0 1

一、功能介绍 2

二、接入指南 2

三、蓝牙基本功能的调用和说明 2

四、基本数据的读取和调用 4

五、设备功能的设置和测试 5

六、固件升级 6

七、SDK版本更新记录 7

## 一、功能介绍

此SDK实现对iOS自带CoreBluetooth框架和维亿魄手环通讯协议的再次封装，开发者可以调用此SDK，实现和维亿魄手环设备的扫描、连接等操作，本SDK也对维亿魄手环通讯协议的所有功能进行了封装，开发者只需要通过简单的接口就可以获得相应的数据，开发者不需要再去了解相关蓝牙知识和App是如何与设备进行通讯的，只要通过正确的调用此SDK的相关接口就能完成所有的操作。

注释：更新SDK的开发者先查看SDK版本更新记录（第一次开发不用看）

## 二、接入指南

1. 将Framework添加到自己的项目中，要求必须在真机下运动
2. 要求项目必须支持手机系统在9.0以上，不能低于9.0，设备必须支持蓝牙4.0
3. 此SDK依赖于AFNetworking、FMDB、iOSDFULibrary、MJExtension，通过Pod集成如下：
4. pod 'iOSDFULibrary', '~> 4.1.0'
5. pod 'FMDB'
6. pod 'AFNetworking', '~> 4.0'
7. pod 'MJExtension'
8. 在使用处导入头文件#import <VeepooBleSDK/VeepooBleSDK.h>即可
9. Pod相关知识与OC中调用swift3.0文件，请自行查找相关资料
10. Xcode必须使用11.0以上版本
11. Other Linker Flags下添加-ObjC

## 三、蓝牙基本功能的调用和说明

1. **初始化SDK（使用SDK必须要进行的操作）**在AppDelegate.m文件中导入头文件#import <VeepooBleSDK/VeepooBleSDK.h>创建一个全局属性@property(nonatomic, strong) VPBleCentralManage \*veepooSDKBleManager; 之后在didFinishLaunchingWithOptions方法中进行初始化self.veepooSDKBleManager = [VPBleCentralManage sharedBleManager];
2. **SDK版本获取方法**

NSString \*sdkVersion = VeepooBleSDKVersion;

1. **监听手机系统蓝牙开启和关闭的状态**，代码如下：

[[VPBleCentralManage sharedBleManager] setVPBleCentralManageChangeBlock:^(VPCentralManagerState iPhoneBleChangeState) {

在这里边根据系统蓝牙状态的改变，去进行相关的操作

}];

1. 监听设备连接状态如连接成功、断开连接、密码验证成功、密码验证失败、有新的固件可以升级，代码如下:

[[VPBleCentralManage sharedBleManager] setVPBleConnectStateChangeBlock:^(VPDeviceConnectState deviceConnectState) {

在这里边根据连接状态去进行相关的操作，一般情况下密码验证成功才算是真正的建立通讯连接

}];

1. **设备的扫描** 当确保手机系统蓝牙开启的状态下，如果进入扫描设备扫描界面调用如下代码，扫描到设备后会有设备模型的回调，一个设备可能会被扫描到多次，所以展示的时候要根据模型VPPeripheralModel的设备地址属性deviceAddress进行重复性过滤，可以根据模型的属性值进行用户展示，扫描调用代码如下：

[[VPBleCentralManage sharedBleManager]veepooSDKStartScanDeviceAndReceiveScanningDevice:^(VPPeripheralModel \*peripheralModel) {

在这里对扫描到的设备进行用户展示，或其他操作

}];

1. **设备的停止扫描** 当不需要扫描设备的时候一定要关闭蓝牙的扫描一般在确定连接设备前调用，如果连接失败在重新扫描，停止扫描调用代码如下 没有回调：

[[VPBleCentralManage sharedBleManager]veepooSDKStopScanDevice];

1. **设备的连接** 当扫描到设备后根据回调的模型进行连接，调用代码如下 bleModel类型为VPPeripheralModel：

[[VPBleCentralManage sharedBleManager] veepooSDKConnectDevice:bleModel deviceConnectBlock:^(DeviceConnectState connectState) {

在这里返回连接状态的回调，根据具体情况进行相关操作

}];

1. **设备的断开连接** 当用户需要断开设备连接的时候调用 调用此代码后SDK将对当前连接的设备和已经连接过的设备不再自动重连，如下代码 在第3条中可以监听到断开状态的变化：

[[VPBleCentralManage sharedBleManager] veepooSDKDisconnectDevice];

1. SDK已经集成了设备自动连接的操作，开发者不需要另行设计

## 四、基本数据的读取和调用

1. 在连接设备并密码验证成功后可根据自己的情况去调用读取数据的接口
2. 开发者可以根据自己的实际情况去设计多久重复读取一次，建议10-20分钟，每次读取都会有相应读取的状态返回，在接口调用期间，数据没有读取完成的时候不要重复调用此接口，只有当数据全部读取完毕再去根据实际情况间隔多久去再次调用
3. 调用的代码如下，确保设备一定在连接状态，判断是否连接的代码为

[VPBleCentralManage sharedBleManager].isConnected//返回bool类型

读取数据的代码如下,这个是读取基本数据的包含心率、计步等，不包含血氧和HRV：

[[VPBleCentralManage sharedBleManager].peripheralManage veepooSdkStartReadDeviceAllDataWithReadStateChangeBlock:^(VPReadDeviceBaseDataState readState, NSUInteger totalDay, NSUInteger currentReadDayNumber, NSUInteger readCurrentDayProgress) {

这里有读取状态、读取总天数、读取当前天、读取当前天的进度

}];

1. 数据的调用，读取数据后，在SDK中根据蓝牙地址度数据进行了保存，调用数据封装在VPDataBaseOperation这个类中，所有调用都使用的是类方法接口，返回的数据格式在类接口中都有详细说明

## 五、设备功能的设置和测试

1. 信息提醒的设置和读取 包括来电、短信、微信、QQ、新浪、FaceBook、Twitter、Flickr、LinkedIn、whatsApp 调用接口案例如下：

[[VPBleCentralManage sharedBleManager].peripheralManage veepooSDKSettingMessageType:VPSettingSMS settingState:VPSettingFunctionOpen completeBlock:^(VPSettingFunctionCompleteState completeState) {

在这里有返回设置和读取的状态

}];

1. 其他功能如一些提醒的开关、读取电池的电量、设备个人信息的设置、私人血压模型的设置和读取、闹钟的设置和读取、久坐的设置和读取也在VPPeripheralManage类中，调用方法与信息提醒类似，就不一一列举了，具体开VPPeripheralManage.h文件，里边有每个参数和回调的详细说明。
2. 拍照功能的设置也在VPPeripheralManage类中，调用接口如下：

[[VPBleCentralManage sharedBleManager].peripheralManage veepooSDKSettingCameraType:VPCameraTypeEnter settingAndMonitorResult:^(VPCameraType cameraType) {

这里是进入设备拍照模式的设置，如果你自己的设备没有这个功能请不要调用此接口，回调参数有手环设置后所处的状态

}];

1. 心率测试、血压测试、血氧测试、疲劳度测试，已血压测试为案例，调用接口如下：

[[VPBleCentralManage sharedBleManager].peripheralManage veepooSDKTestBloodStart:YES testMode:0 testResult:^(VPTestBloodState testBloodState, NSUInteger testBloodProgress, NSUInteger highBlood, NSUInteger lowBlood) {

这里已开始测试血压为例，返回有测试进度和结果，心率、血氧、疲劳度也在这个类中，详细使用见类中注释；

}];

1. 设备与iphone手机配对的调用，每次连接后SDK都主动询问用户是否绑定手机，如果用户点击错了或者点了取消又想绑定，则调用此接口进行绑定：

[[VPBleCentralManage sharedBleManager].peripheralManage veepooSDKSendPairedWithIphoneCommand];

## 六、固件升级

1. 确保设备连接并发现新固件直接调用下边的接口就行，接口调用方式：

[[VPDFUOperation dfuOperationShare] veepooSDKStartDfu:^(NSInteger progress, DeviceDFUState deviceDFUState) {

回调中监听固件升级的状态和进度，升级过程中确保设备电量充足，手机和设备保持连接距离，尽可能的不要中途断开连接

}];

1. 升级失败设备名称变为DFULang，重新连接检测到固件升级消息，再次调用上边的接口即可。

## 七、SDK版本更新记录

|  |  |  |
| --- | --- | --- |
| 更新日期 | 版本 | 更改记录 |
| 2018/12/20 | V2.0.13.0 | 1. 修改bug 2. 此版本以上必须在Other Linker Flags下添加-ObjC |
| 2018/06/15(稳定版) | V2.7（对应demo为2.7版本） | 1.增加读取血氧和HRV功能  2.增加血氧自动检测、血氧过低提醒、来电自动接听开关设置 |
| 2018/03/12(稳定版) | V2.5（对应demo为2.5版本） | 1.修改2.4不能读取昨日数据的bug |
| 2017/12/20(稳定版) | V2.4（对应demo为2.4版本） | 1.增加了运动模式的开始和结束  2.增加了转腕灵敏度调节  3.增加了清除数据功能  4.更了了升级代码，pod中要重新集成'iOSDFULibrary' 4.1.0版本 |
| 2017/11/28(稳定版) | V2.0（对应demo为2.0版本） | 1.解决基本数据只能读取1天的bug |
| 2017/6/26(稳定版) | V1.8（对应demo为1.7版本） | 1.增加了log日志打印的开关  2.优化数据读取逻辑，避免每次读取的数据重复 |
| 2017/6/16(稳定版) | V1.7 | 1.修改了血压私人模式设置接口，兼容了血压动态校准功能（之前调试的要改一下这个接口的调用，很好改）  2.修改了读取数据会卡在正在读取的bug  3.修改了每天第一次读取数据都要重头开读取的bug  4.增加倒计时功能接口及Demo  5.增加了女性项目接口及Demo  6.增加了新闹钟功能的接口及Demo  7.增加了新增信息提醒的平台（信息提醒接口有注释）  8.增加了一些设置开关功能（开关接口有注释）  9.兼容了读取的总天数根据设备具体存储的天数而定（以前是固定3天，大部分设备也基本是存储3天数据） |
| 2017/3/22 | V1.5 | 增加新的接口 |
| 2017/3/4 | V1.3 | 1.修改1.2一些细节 |
| 2017/2/28 | V1.2 | 1.修改1.0连接奔溃的bug  2.增加抬手亮屏，时间段设置接口 |
| 2017/2/15 | V1.0(第一版) | 第一版更新，有一些bug，不稳定 |
|  | V |  |
|  |  |  |
|  |  |  |