密级：公开资料

iOS BLE API 使用说明

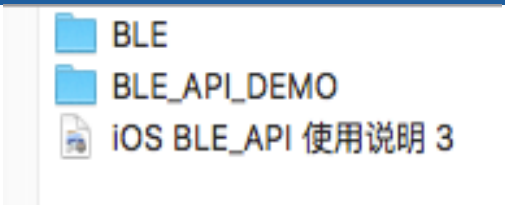
文档版本：V1.3

深圳市昇润科技有限公司
2017 年 09 月 19 日

版本	修订日期	修订人	审稿人	修订内容
1.0	2016-04-15	李炯	张眼	1. iOS BLE API 使用说明初版发布
1.1	2017-01-16	李炯	张眼	1. 增加 CC2640 数据收发相关说明
1.2	2017-03-09	李炯	张眼	1.修改 2541 发送数据长度的描述
1.3	2017-09-1	李炯	张眼	1.增加对模块参数获取和设置的说明; 2.增加对自定义服务的数据读写的说明

目 录

1. iOS BLE SDK 介绍



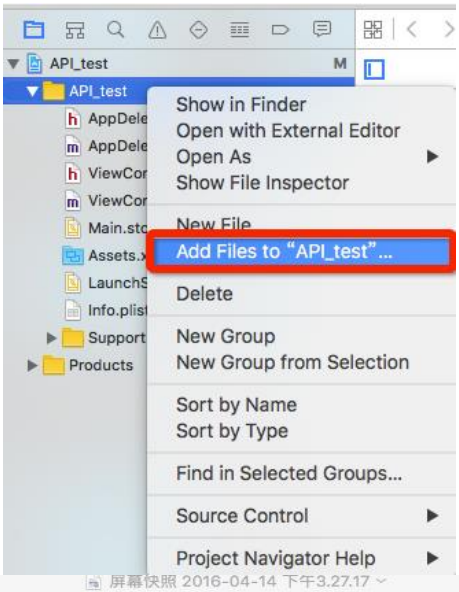
1)BLE 目录下的文件是需要添加的库文件
2)BLE_API_DEMO 为示例代码

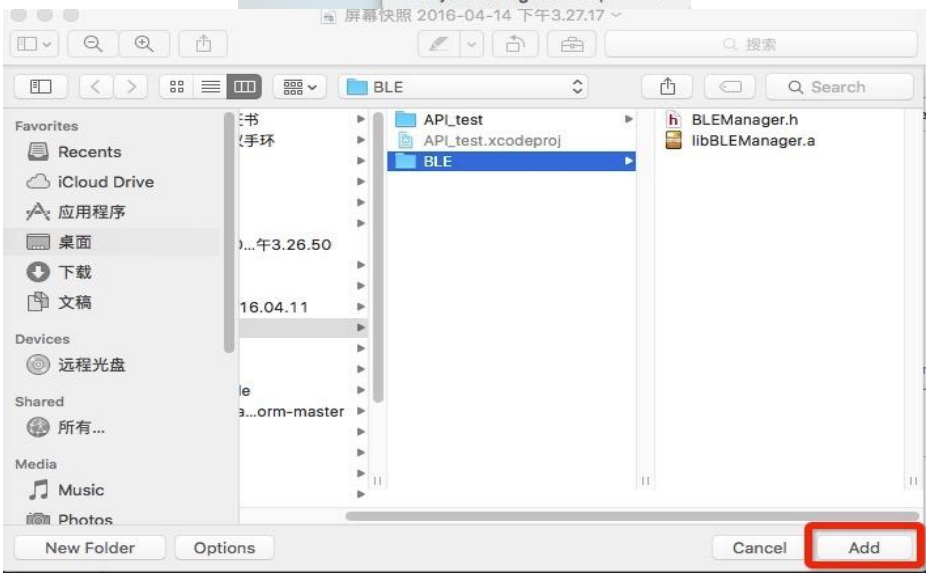
2. 工程配置

2.1. 导入蓝牙静态文件

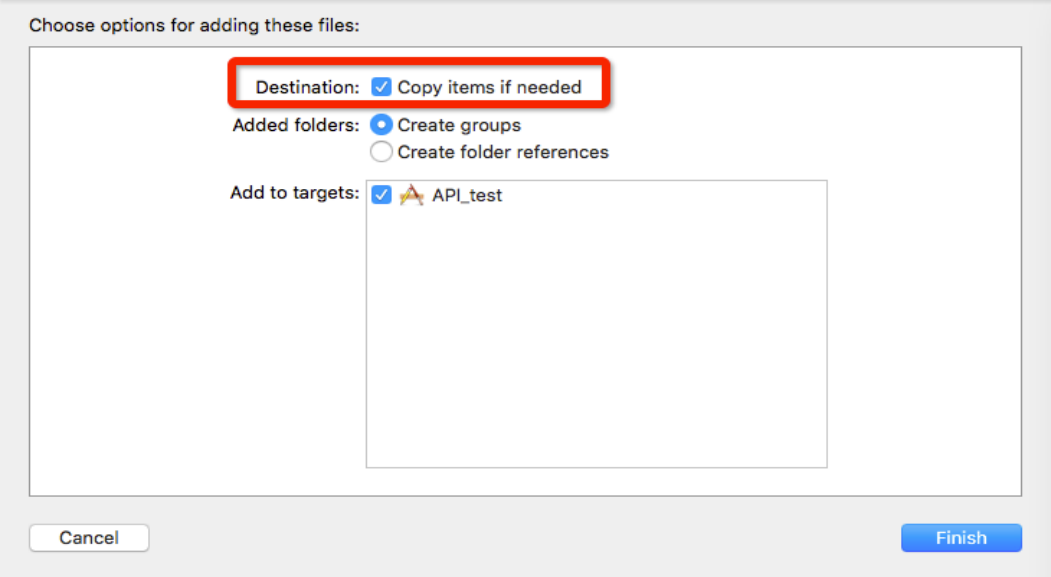
下面提供两个方法：

方法一：将静态文件添加到工程中，然后在项目中点击右键，添加到工程中，如下

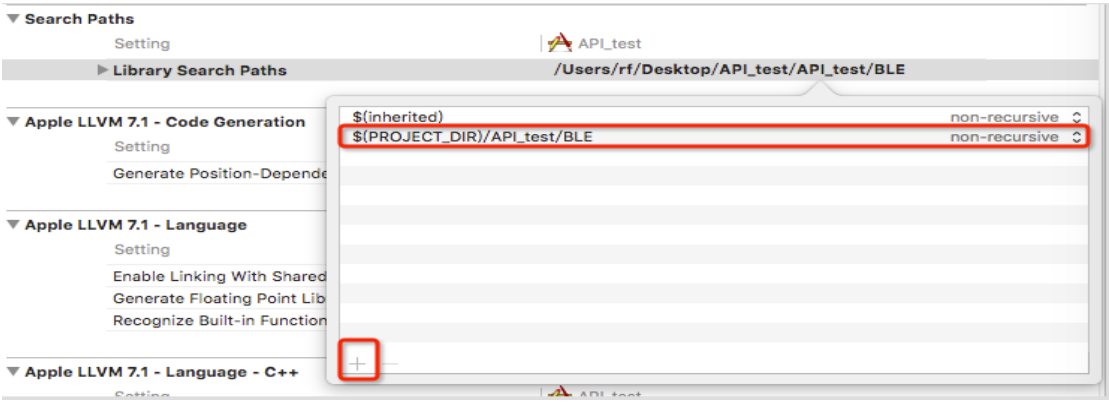




方法二：直接将静态文件拖进工程中，记住勾选 copy items if needed

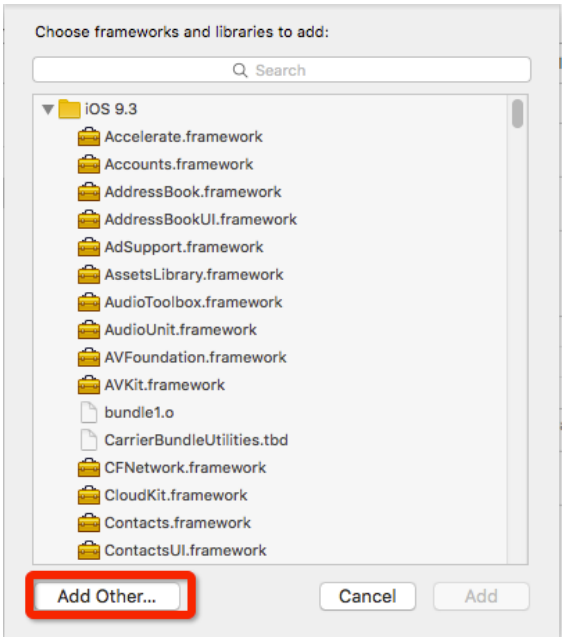


（1）导入成功后，查看工程中 TARGETS -> Build Settings -> Search Paths -> Library Search Paths,如图，如果没有静态文件路径,请手动添加，显示静态文件简介，点击上图'＋'，将文件路径复制添加到下图中。



（2）该静态文件不支持 bitcode, 在工程中 TARGETS -> Build Settings ->Build Options -> Enable Bitcode 设置为 NO

（3）添加蓝牙框架，在工程中 TARGETS -> Link Binary With Libraries 添加 CoreBluetooch.framework，直到 Link Binary With Libraries 出现 CoreBluetooch.framework；如果里面没有 libBLEManager.a，点击添加其他 直到 Link Binary With Libraries 出现 libBLEManager.a



（4）如果在调试工程的过程中, Xcode 提示未找到<DeviceInfo.h>和<oad.h>文件,请将我司提供的 API 中的这两个文件添加到工程中去,因为在 libBLEManager.a 文件中用到上面两个文件中的内容

（5）如果需要调试,请在真机上调试, libBLEManager 不支持模拟器上面的调试

3. 代码示例

3.1 创建 BLEManager 实例

BLEManager * manager = [BLEManager defaultManager];

需要设置 manager 的 delegate.

3.2 搜索设备

[manager scanDeviceTime:(NSInteger)];

这个参数传一个 NSInteger 类型的值,表示搜索持续的时间,在搜索到设备后,会回调-(void)scanDeviceRefrash:(NSMutableArray *)array;方法, array 数组中保存的是 DeviceInfo 对象,该对象所包含的属性可以在<DeviceInfo.h>文件中查看;

3.3 连接设备

-(void)connectToDevice:(CBPeripheral *)device;

参数 device: 要连接的从机对象;

可以通过-(CBPeripheral *)getDeviceByUUID:(NSString *)uuid;这个方法得到从机对象,而这个方法中的 uuid 参数可以在之前所说的搜索设备的回调方法里面的 array 中的 DeviceInfo 对象里得到;

进行连接之后,连接成功或者失败,可以在下面这两个回调里面得到答案;

```
/**
 * 连接设备成功回调方法
 * @param device 设备对象
 * @param error  错误信息
*/
-(void)connectDeviceSuccess:(CBPeripheral *)device error:(NSError *)error;
```

```
/**
 * 断开设备成功回调
 * @param device 设备对象
 * @param error  错误信息
 */
-(void)didDisconnectDevice:(CBPeripheral *)device error:(NSError *)error;
```

3.4 给从机发数据

发送数据的方式有两种,一种是加密,另一种是不加密;默认发送数据是加密的方式,如果需要的是不加密的数据方式,请将 manager 的 isEncryption 属性设置为 NO

默认的数据发送通道为 1000 服务下的 1001 通道:

-(void)sendDataToDevice1:(NSString *)dataStr device:(CBPeripheral *)device;

参数 1:发送的数据,该数据是 16 进制的字符串(注意:如果是加密的方式,字符串的长度最大是 34,如果是不加密的方式,字符串的长度最大是 40)

参数 2:从机对象.

3.5 接收从机数据

（1）如果是主动读取从机的数据可以在

-(void)receiveDeviceDataSuccess_3:(NSData *)data device:(CBPeripheral *)device;这个回调方法中得到数据

参数 1:获取到的数据

参数 2:从机对象

（2）如果是从机主动广播的数据,可以在

-(void)receiveDeviceDataSuccess_1:(NSData *)data device:(CBPeripheral *)device;这个回调方法中的到数据

3.6 读取模块参数

3.6.1 如何读取模块参数(自定义获取数据)

（1）按照底层规格书向 1005 通道发送对应参数的数据:例如,获取当前设备名称,向 1005 通道发送”0e”,使用如下的方法:

- (void)sendDataToDevice5:(NSString *)dataStr device:(CBPeripheral *)device;

（2）下一步读取 1004 通道的数据,使用如下方法

- (void)readDataWithDevice4:(CBPeripheral *)device;

然后,可以在下面的回调方法里面通过底层的说明解析数据即可:

- (void)bleManagerPeripheral:(CBPeripheral *)peripheral didUpdateValueForCharacteristic:(CBCharacteristic *)characteristic error:(NSError *)error;

3.6.2 如何读取模块参数(默认获取数据)

在我们的 API 中,我们已经对获取相应参数的流程进行了封装,比如获取设备名称的方法如下:

- (void)readDeviceSettingName:(CBPeripheral *)device;

然后,我们可以在相应的回调方法里面获取到要读取的数据,该数据我们已经进行了解析,为最终的数据.获取到设备名称后的回调方法如下:

```
-(void)receiveDeviceSettingName:(NSString*)name
device:(CBPeripheral *)
device;
```

其他参数的获取方法我们也都做了封装,这些方法都可以在 API 里面找到,只要按照上面的流程就能获取到你想要的数据.

3.7 设置模块参数

3.7.1 如何设置模块参数(自定义设置数据)

(1)按照底层规格书向 1005 通道发送对应参数的数据:例如,获取当前设备名称,向 1005 通道发送"0e",使用如下的方法:

```
- (void)sendDataToDevice5:(NSString *)dataStr device:(CBPeripheral *)device;
```

(2)下一步向 1003 通道写数据,使用如下方法

```
- (void)sendDataToDevice3:(NSString *)dataStr device:(CBPeripheral *)device;
```

3.7.2 如何设置模块参数(默认设置数据)

和获取数据一样,我们也对设置数据的所有流程进行了封装,比如客户想要设置模块名称,使用如下的方法将名称数据发送给设备即可:

```
- (void)setDeviceName:(NSString *)name device:(CBPeripheral *)device;
```

如果想要将设备的名称设置为 abcd,只需要给上面的方法的 name 参数传值为@"abcd"即可.

3.8 如何操作设备对应通道的 notify

使用如下方法操作设备对应通道的 notify 状态:

```
-(void)notification:(UInt16)serviceUUID
characteristicUUID:(UInt16)cUUID
peripheral:(CBPeripheral *)device enableState:(BOOL)isEnable;
```

参数 1 为设备的服务 UUID, 参数 2 为对应通道的 UUID, 参数 3 为设备对象,参数 4 是一个 BOOL 类型的值,传值为 YES 是打开该通道 notify, 传 NO 是关闭该通道 notify.

比如要打开 1000 服务下的 1002 通道的 notify, 我们应该按照下面的示例代码进行操作:

```
[BLEManager defaultManager] notification:0x1000
characteristicUUID:0x1002 peripheral:device enableState:YES];
```

是否设置成功,我们可以在下面的回调方法里面进行判断:

```
-(void)bleManagerPeripheral:(CBPeripheral *)peripheral
didUpdateNotificationStateForCharacteristic:(CBCharacteristic
*)characteristic error:(NSError *)error;
```

3.9 如何对非 1000 服务下的通道进行读写数据的操作

3.9.1 如何对非 1000 服务下的通道进行写数据的操作

使用如下方法:

-(void)bleManagerPeripheral:(CBPeripheral *)peripheral writeValue:(NSString*)
string serviceUUID:(UInt16)serviceUUID characteruisticUUID:(UInt16) characteruisticUUID
encryption:(BOOL)encryptionresponse(BOOL)response;

参数 1 为设备对象,参数 2 为写入的数据,参数 3 是设备服务的 UUID, 参数 4 是对应通道的 UUID,参数 5 为是否加密,参数 6 为写数据的方式, YES 代表有回应的写(
CBCharacteristicWriteResponse),为 NO 代表无回应的写
(CBCharacteristicWriteWithoutResponse).

比如要向 2000 服务下的 2001 通道写入 123456 这个数据,数据方式为加密(这个是根据模块是否加密设置的), response 参数设置为 NO, 示例代码如下:

[BLEManager DefaultManager] bleManagerPeripheral:devicewriteValue:
@“123456”erviceUUID:0x2000 characteruisticUUID:0x2001encryption:
YES response:NO];

3.9.2 如何对非 1000 服务下的通道进行读数据的操作

如同 3.8 节中介绍的那样,打开对应通道的 notify,然后当设备有数据发送给 APP 的时候,我们可以在下面的回调方法里面获取到数据:

-(void)bleManagerPeripheral:(CBPeripheral *)peripheral didUpdateValueForCharacteristic:(CBCharacteristic*)characteristic error:(NSError *)
error;

获取到的数据即为 characteristic 对象的 value 属性值.

4.联系我们

深圳市昇润科技有限公司
ShenZhen ShengRun Technology Co.,Ltd.
Tel: 0755-86233846 Fax: 0755-82970906
官网地址：www.tuner168.com
阿里巴巴网址：http://shop1439435278127.1688.com
E-mail: marketing@tuner168.com
地址：广东省深圳市南山区西丽镇龙珠四路金谷创业园 B 栋 6 楼 601-602


珠光
龙辉花园-9栋
地铁7号线(在建)
龙珠三路
永泰新屋宾馆
龙珠大道
南坪快速
鼎胜·山邻
岭南风平台
丹荔台
十五奉
龙井
城市假日
桃源社区公园
水木华庭
光前村西区75号楼
奥瑞那大厦
龙井路
金谷创业园
皇庭商务宾馆
中爱大厦
中爱花园
龙井村
食为天
桃源村
梅州大厦
吉利猪肚包鸡
广深高速
G4
海龙苑
桃源村
万和
缇花廊
欧翔路
沙河东路
大沙河公园
长寿梯
本钢大厦
桃源村综合楼
中联科技大厦
教师公寓
宜家家居深圳商场