青莲云 iOS SDK 快速开发文档

目录

一、概要

青莲云作为物联网后端云服务，用户可以使用青莲云提供的 iOS SDK 快速进行 App 开发。青莲云 iOS SDK 提供了与硬件设备、青莲云通讯的接口封装，快速开发包括以下功能：

账户体系（SDK 初始化，手机号注册、登录，获取产品列表，获取设备列表）

硬件设备相关（配网、搜索、绑定、控制、监听状态上报）

二、SDK 介绍

1、SDK 名称

SDK 名称：IOTSDK.framework

2、接入说明

下载 IOTSDK.framework，将该文件拷贝到工程目录下。按照如下步骤接入：

- 在项目 TAGETS—>Build Phases—>New Copy Files Phase 下点击加号，加入 IOTSDK.framework。
- 在项目 TAGETS—>Build Phases—>Link Binary With Libraries 下点击加号，加入 IOTSDK.framework。
- 在项目的预编译文件 PrefixHeader.pch 中添加以下内容：

```
#import <IOTSDK/IOTSDK.h>
```

3、注意事项

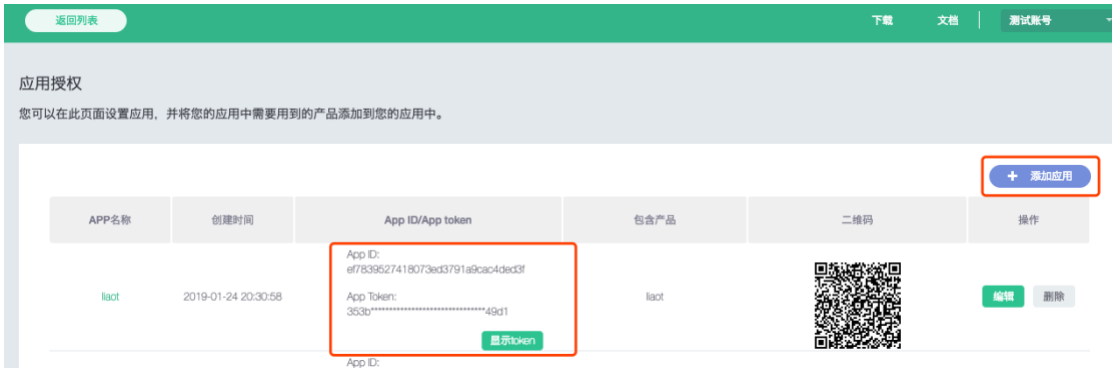
SDK 内置了 Esptouch、HFSmartLink、EasyLink、LTLink、JMAirKiss 和 MQTTClient 第三方框架，并进行了再次封装，以下有详细说明。所以，为避免冲突，客户端 App 不宜再导入相同类库。

三、开发前准备

前往青莲云官网 https://www.qinglianyun.com,注册开发者账号，新建应用获取到 AppId ，AppToken，SDK 初始化时会用到。如下图：

1、在官网左上角点击授权管理

2、添加应用即可获得 AppID 和 AppToken



四、SDK 使用

1、SDK 初始化

打开 AppDelegate.m 文件，在[AppDelegate application:didFinishLaunchingWithOptions:]方法中使用开发平台获取的 AppID 和 AppToken 调用接口初始化 SDK：

请求参数：

参数名	类型	说明	备注	必需
AppId	String	应用 id	云平台获取	是
AppToken	String	应用 token	云平台获取	是

例子：

```
[[IOTSmartSDK sharedInstance] IOTCloudSDK_startWithAppID:<#your_app_id#>
AppToken:<#your_app_token#> success:^{
        NSLog(@"install success");
} failure:^(int errNo,NSString *errMessage) {
        NSLog(@"install failure: %@", errMessage);
}];
```

2、用户注册

用户相关的所有功能对应 IOTSmartUser 类。

1、发送注册验证码到手机：

请求参数：

参数名	类型	说明	备注	必需
phone	String	手机号码	接收验证码的手机号	是
zone	String	区号（如中国大陆手机号写“0086”）	默认四位，不足四位前面补 0	是

例子：

```
- (void)sendVerifyCode {
        [[IOTSmartUser sharedInstance]
IOTCloudSDK_registerSendVerifyCodeByPhoneNum:@"your_phone_number" zone:
@"your_phone_zone"    success:^{
        NSLog(@"sendVerifyCode success");
} failure:^(int errNo,NSString *errMessage) {
        NSLog(@"sendVerifyCode failure: %@", errMessage);
}];
}
```

2、使用手机验证码注册

请求参数：

参数名	类型	说明	备注	必需
phone	String	手机号码	接收验证码的手机号	是
zone	String	区号（如中国大陆手机号写“0086”）	默认四位，不足四位前面补 0	是
password	String	注册的账号密码		是
smscode	String	手机接收到的验证码		是

例子：

```
- (void)registerByPhone {
        [[IOTSmartUser sharedInstance]
IOTCloudSDK_registerByPhoneNum:@"your_phone_number" zone: @"your_phone_zone"
password:@"your_password" smsCode:@"verify_code" success:^{
        NSLog(@"register success");
} failure:^( int errNo,NSString *errMessage) {
        NSLog(@"register failure: %@", errMessage);
}];
}
```

3、用户登录

1、使用手机号与密码进行登录：

请求参数：

参数名	类型	说明	备注	必需

phone	String	手机号码	接收验证码的手机号	是
password	String	注册的账号密码		是
zone	String	区号（如中国大陆手机号写“0086”）	默认四位，不足四位前面补 0	是

例子：

```
- (void)loginByPhoneAndPassword {
        [[IOTSmartUser sharedInstance]
IOTCloudSDK_loginByPhoneNum:@"your_phone_number" password:@"your_password"
zone: @"your_phone_zone"    success:^{
        NSLog(@"login success");
} failure:^( int errNo,NSString *errMessage) {
        NSLog(@"login failure: %@", errMessage);
}];
}
```

2、用户退出登录

请求参数：无

例子：

```
- (void) signOut {
        [[IOTSmartUser sharedInstance] IOTCloudSDK_Signout];
}
```

4、获取产品列表

添加产品配置到应用中就会在产品列表接口中获取到

产品相关的所有功能对应 IOTSmartProduct 类。

通过调用以下接口获取所有产品列表。返回 IOTSmartProductListModel 类对象数组。

请求参数：无

例子：

```
- (void)getProductList{
        [IOTSmartProduct IOTCloudSDK_getProductListSuccess:
^(NSArray<IOTSmartProductListModel *> *productListModelArr) {
```

```
        NSLog(@"getProductList success");
} failure:^( int errNo,NSString *errMessage) {
        NSLog(@"getProductList failure: %@", errMessage);
}];
}
```

5、设备配网

设备配网有两种类型：

1、直接调用硬件厂商的配网 SDK 给已知模组配网，目前支持的模组：乐鑫 ESP8266、汉枫 LPB120、庆科 WIFI 模组、高通 WIFI、AirKissWIFI 模组。

请求参数：

参数名	类型	说明	备注	必需
activatorType	enum	模组类型	IOTSmartActivatorType 枚举类型中的一种	是
password	String	当前 Wifi 的 password		是

例子：

```
- (void)start{
        [[IOTSmartActivator shareInstance] IOTCloudSDK_startConfigWithActivatorType:@"your_choose_activatorType" password: @"your_wifi_password" success:^{
        NSLog(@"activator success");
} failure:^{
        NSLog(@"activator failure");
}];
}
```

2、把设备作为热点，手机连到设备热点，把目标网络的 ssid 的 password 传给设备实现 AP 配网。

请求参数：

参数名	类型	说明	备注	必需
ssid	String	目标 Wifi 的 ssid		是
password	String	目标 Wifi 的 password		是

例子：

```
- (void)APstart{
        [[IOTSmartActivator shareInstance] IOTCloudSDK_startAPConfigWithSSID:@"wifi_ssid" password: @" wifi_password" success:^{
        NSLog(@"activator success");
} failure:^{
        NSLog(@"activator failure");
}];
}
```

3、停止配网

请求参数：

参数名	类型	说明	备注	必需
activatorType	enum	模组类型	IOTSmartActivatorType 枚举类型中的一种	是

例子：

```
- (void)stop{
        [[IOTSmartActivator shareInstance] IOTCloudSDK_stopConfigWithActivatorType:
@"your_choose_activatorType"];
}
```

6、搜索设备

硬件正常上网后，搜索设备发现设备，采用 udp 发包形式，发现设备相关的所有功能对应 IOTSmartFindDevices 类。

1、调用以下方法发现设备后立即成功回调返回 IOTSmartFindDeviceModel 实体类对象，可以多次成功回调。

请求参数：

参数名	类型	说明	备注	必需
productId	int	设备的产品 Id	填写产品 id 就只能发现改产品下的设备，若填写 0 则可以发现该局域网内所有正常上网的设备	是
timeout	int	超时时间		是

例子：

```
- (void)startFind{
        [[IOTSmartFindDevices shareInstance] IOTCloudSDK_getDeviceWithProductId:
@"your_choose_productId" timeout : @"your_choose_timeout"
success:^( IOTSmartFindDeviceModel    *findDeviceModel){
        NSLog(@"findDevices success");
} failure:^( int errNo,NSString *errMessage) {
        NSLog(@"findDevices failure: %@", errMessage);
}];
}
```

2、停止搜索设备

请求参数：无

例子：

```
- (void)stopfind{
        [[IOTSmartFindDevices shareInstance] IOTCloudSDK_stopUDP];
}
```

7、绑定设备

一个设备只能被一个用户绑定，不允许多个用户绑定一个设备。设备相关的所有功能对应 IOTSmartDevice 类。绑定成功返回 IOTSmartDeviceListModel 实体类对象。

1、调用以下接口实现用户与搜索到的设备进行绑定：

请求参数：

参数名	类型	说明	备注	必需
iotId	String	绑定设备的 iotId		是
iotToken	String	绑定设备的 token		是

例子：

```
- (void)bindDevice{
        [IOTSmartDevice IOTCloudSDK_bindDeviceWithIotId: @"your_findDevice_iotId"
iotToken ： @"your_ findDevice_iotToken" success:^(IOTSmartDeviceListModel *
deviceModel){
        NSLog(@"bindDevice success");
} failure:^( int errNo,NSString *errMessage) {
        NSLog(@"bindDevice failure: %@", errMessage);
}];
}
```

2、解绑设备

请求参数：无

例子：

```
- (void)unbindDevice{
        [self.device IOTCloudSDK_unbindDeviceSuccess: ^{
        NSLog(@"unbindDevice success");
} failure:^( int errNo,NSString *errMessage) {
        NSLog(@"unbindDevice failure: %@", errMessage);
}];
}
```

8、获取设备列表

通过调用以下接口获取所有设备列表。返回 IOTSmartDevice 类对象数组。

请求参数：无

例子：

```
- (void)getDeviceList{
        [IOTSmartDevice IOTCloudSDK_getDeviceListSuccess:
^(NSArray<IOTSmartDevice *> * deviceLisArr) {
        NSLog(@"getDeviceList success");
} failure:^( int errNo,NSString *errMessage) {
        NSLog(@"getDeviceList failure: %@", errMessage);
```

```
}];
}
```

9、控制设备

控制设备之前需要为该设备所属的产品增加功能点，点击进入产品功能后，添加功能点即可。

请求参数：

参数名	类型	说明	备注	必需
dps	NSArray	要控制的数据点数组	数组元素为 IOTSmartDP 实体类对象	是

例子：

```
- (void)cmdDps{
        [self.device IOTCloudSDK_publishDps：@"your_dps_Arr"    sucess:^{
        NSLog(@"cmdDps success");
} failure:^(NSString *errMessage) {
        NSLog(@"cmdDps failure: %@", errMessage);
}];
}
```

10、监听设备状态上报

手机端要监听该用户下的所有设备的数据点更新，需要将自身作为 IOTSmartNotificationDeviceNewDPsUpdateArr 通知的观察者。在相应的通知方法里，取出 notice.userInfo，根据[notice.userInfo objectForKey:@"iotId"],判断哪个设备有数据点更新。