Android APP SDK

开发文档

版本：1.0

修订记录

|  |  |  |  |
| --- | --- | --- | --- |
| **备注** | **版本** | **发布日期** | **作者** |
| 创建文档 | 1.0 | 2016-5-13 | YJ |

目录

[1. 概述 5](#_Toc450926884)

[2. 功能介绍 5](#_Toc450926885)

[3. 开发流程 5](#_Toc450926886)

[4. SDK配置 6](#_Toc450926887)

[5. SDK接口 7](#_Toc450926888)

[5.1. SDK版本 7](#_Toc450926889)

[5.2. SDK LOG信息打印 7](#_Toc450926890)

[5.3. SDK初始化 7](#_Toc450926891)

[5.4. SDK关闭 9](#_Toc450926892)

[5.5. 账号系统AccountImpl 10](#_Toc450926893)

[5.5.1. 登陆 10](#_Toc450926894)

[5.5.2. 本地登录 10](#_Toc450926895)

[5.5.3. 注册 11](#_Toc450926896)

[5.5.4. 获取手机验证码 11](#_Toc450926897)

[5.5.5. 获取邮箱验证码 11](#_Toc450926898)

[5.5.6. 获取用户昵称头像 12](#_Toc450926899)

[5.5.7. 修改用户头像 12](#_Toc450926900)

[5.5.8. 修改用户昵称 12](#_Toc450926901)

[5.5.9. 修改密码 13](#_Toc450926902)

[5.5.10. 发送修改手机号码验证码 13](#_Toc450926903)

[5.5.11. 发送修改邮箱验证码 13](#_Toc450926904)

[5.5.12. 发送找回密码验证码 13](#_Toc450926905)

[5.5.13. 修改手机 14](#_Toc450926906)

[5.5.14. 修改邮箱 14](#_Toc450926907)

[5.5.15. 重置密码 14](#_Toc450926908)

[5.5.16. 检查用户名密码是否正确 15](#_Toc450926909)

[5.5.17. 获取用户手机邮箱信息 15](#_Toc450926910)

[5.6. 设备控制ControllerImpl 16](#_Toc450926911)

[5.6.1. 设备初始化 16](#_Toc450926912)

[5.6.2. 设备配置 17](#_Toc450926913)

[5.6.3. 设备扫描 18](#_Toc450926914)

[5.6.4. 设备配对 19](#_Toc450926915)

[5.6.5. 设备控制 19](#_Toc450926916)

[5.6.6. 查询设备状态 20](#_Toc450926917)

[5.6.7. 查询局域网设备IP地址 21](#_Toc450926918)

[5.6.8. 查询设备的Profile 21](#_Toc450926919)

[5.6.9. 查询固件版本 21](#_Toc450926920)

[5.6.10. 查询设备服务器时间 21](#_Toc450926921)

[5.6.11. 固件升级 22](#_Toc450926922)

[5.6.12. 修改设备信息 22](#_Toc450926923)

[5.6.13. 设备通用定时 23](#_Toc450926924)

[5.6.14. 资源下载 27](#_Toc450926925)

[5.7. 数据采集 29](#_Toc450926926)

[6. 错误码 30](#_Toc450926927)

[7. FAQ 33](#_Toc450926928)

[8. 附录 35](#_Toc450926929)

[8.1. 产品profile格式说明 35](#_Toc450926930)

# 概述

BroadLink的智能家居平台DNA System，旨在帮助各大厂商统一、快捷地集成自己的家电设备到系统中。我们提供了完整的控制SDK，完善的云端接口供开发者调用。

但由于系统复杂、功能繁多、接口自由性，加上一系列的安全机制，使得开发者在使用SDK开发时遇到以下问题：

* 需要了解整个复杂的DNA系统
* 需要了解各接口之间的联系与限制
* 开发者需要较丰富的开发经验
* 开发者需要较长的学习时间
* 需要不断的提供技术支持

针对上述问题，APP SDK(Let)诞生，帮助开发者快速接入DNA System。

# 功能介绍

APP SDK目前分为四块：

1. 账号系统 Let.Account
2. 设备控制 Let.Controller
3. 数据统计 Let.Picker
4. 调试信息打印 Let.DebugLog

# 开发流程

1. License申请

为了设备的安全，所有使用SDK的用户必须向BroadLink Co., Ltd.申请License。 License与应用的包名(packageName)相关联，不同的应用包名需要申请不同的License。 License还用于远程资源访问的限制

1. SDK配置

将SDK jar，.so文件放入对应文件夹，并且配置SDK使用权限防止SDK使用失败。

1. SDK初始化 Let.init()
2. SDK接口调用

为了保证设备的控制权限，SDK会将添加的每一个设备与当前的账号绑定（SDK会自动完成绑定，开发者无需关心），在使用SDK初始化完成之后，必须调用SDK Let.Account.login()或者Let.Account.localLogin()方法登陆SDK，以保证设备可以正常的远程控制。

SDK 搜索设备需要先调用Let.Controller.startProbe()方法启动定时器搜索设备，通过设置Let.Controller.setOnDeviceScanListener()设备回调函数将搜索到新设备和设备属性发生改变的返回应用层。

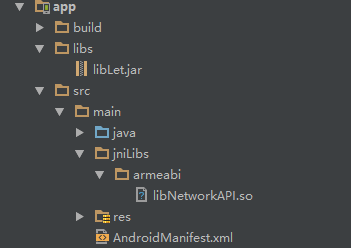
1. SDK退出

在APP退出的时候，需要调用Let. finish()关闭网络库，已经避免网络库在后台运行占用手机内存和消耗用户流量。

# SDK配置

1. 新建android app 工程
2. 导入libLet.jar 拷贝到libs/目录下
3. Android studio拷贝libNetworkAPI.so到jniLibs/armeabi/目录下

Eclipse拷贝libNetworkAPI.so到libs/armeabi/目录下



1. AndroidMainfest.xml权限添加

*<!-- 在SDCard中创建与删除文件权限 -->*<uses-permission android:name="android.permission.MOUNT\_UNMOUNT\_FILESYSTEMS" />  
*<!-- 往SDCard写入数据权限 -->*<uses-permission android:name="android.permission.WRITE\_EXTERNAL\_STORAGE" />  
<uses-permission android:name="android.permission.BLUETOOTH" />  
<uses-permission android:name="android.permission.READ\_PHONE\_STATE" />  
<uses-permission android:name="android.permission.ACCESS\_WIFI\_STATE" />  
<uses-permission android:name="android.permission.ACCESS\_NETWORK\_STATE" />

<uses-permission android:name="android.permission.CHANGE\_WIFI\_MULTICAST\_STATE" />

*<!-- 连接互联网Internet权限 -->*<uses-permission android:name="android.permission.INTERNET" />

# SDK接口

## SDK版本

|  |  |  |
| --- | --- | --- |
| 方法：getSDKVersion ()  说明：用于SDK所有模块的初始化 | | |
| 返回参数 | 类型 | 说明 |
| sdkVersion | String | APP SDK版本 |

## SDK LOG信息打印

打开SDK LOG信息打印：Let.DebugLog.on()

关闭SDK LOG信息打印：Let.DebugLog.off()

## SDK初始化

在应用Application的onCreate函数中调用初始化SDK Let.initSelf()函数。

|  |  |  |
| --- | --- | --- |
| 方法：init(Context context)  init(Context context,ConfigParam configParam)  说明：用于SDK所有模块的初始化 | | |
| 参数 | 类型 | 说明 |
| context | Context | App Application对象 |
| configParam | ConfigParam | 全局配置项：如目录，超时时间 |

调用此方法初始化需要在AndroidManifest.xml将license和channel配置。

例如：

<meta-data  
 android:name="LET\_LICENSE"  
 android:value="license1234567890" />  
  
<meta-data  
 android:name="LET\_CHANNEL"  
 android:value="cn.com.broadlink.xxx" />

|  |  |  |
| --- | --- | --- |
| 方法：init (Context context, String license, String channel)  init (Context context, String license, String channel, ConfigParam configParam)  说明：用于SDK所有模块的初始化 | | |
| 参数 | 类型 | 说明 |
| context | Context | App Application对象 |
| license | String | DNA System的license |
| channel | String | 渠道，标示Android App的发布渠道，数据采集用 |
| configParam | ConfigParam | 全局配置项：如目录，超时时间 |

ConfigParam 可配置参数：

所有超时时间配置均已毫秒为单位。

*/\*\* 全局的http超时时间 \*\*/*public static final String *HTTP\_TIMEOUT* = "HTTP\_TIMEOUT";  
*/\*\* 账号系统的http超时时间 \*\*/*public static final String *ACCOUNT\_HTTP\_TIMEOUT* = "ACCOUNT\_HTTP\_TIMEOUT";  
*/\*\* 账号域名配置 \*\*/*public static final String *ACCOUNT\_HOST* = "ACCOUNT\_HOST";  
*/\*\* 数据采集系统的http超时时间 \*\*/*public static final String *PICKER\_HTTP\_TIMEOUT* = "PICKER\_HTTP\_TIMEOUT";  
*/\*\* SDK文件存储根目录 \*\*/*public static final String *SDK\_FILE\_PATH* = "SDK\_FILE\_PATH";  
*/\*\* 控制系统sdk jni日志等级  
 \* 0不打印错误信息  
 \* 1打印error  
 \* 2打印 warn  
 \* 3打印debug  
 \* 4打印info  
 \* \*\*/*

public static final String *CONTROLLER\_JNI\_LOG\_LEVEL* = "CONTROLLER\_JNI\_LOG\_LEVEL";

*/\*\* 本地超时时间 \*\*/*public static final String *CONTROLLER\_LOCAL\_TIMEOUT* = "CONTROLLER\_LOCAL\_TIMEOUT";  
*/\*\* 远程超时时间 \*\*/*public static final String *CONTROLLER\_REMOTE\_TIMEOUT* = "CONTROLLER\_REMOTE\_TIMEOUT";  
*/\*\* 网络模式 \*\*/*public static final String *CONTROLLER\_NETMODE* = "CONTROLLER\_NETMODE";  
*/\*\* 连续发包次数 \*\*/*public static final String *CONTROLLER\_SEND\_COUNT* = "CONTROLLER\_SEND\_COUNT";  
*/\*\* 设备配置超时时间 \*\*/*public static final String *CONTROLLER\_DEVICE\_CONFIG\_TIMEOUT* = "CONTROLLER\_DEVICE\_CONFIG\_TIMEOUT";  
*/\*\* 脚本路径 \*\*/*public static final String *CONTROLLER\_SCRIPT\_PATH* = "CONTROLLER\_SCRIPT\_PATH";

## SDK关闭

在退出应用的时候用户需要调用finishSelf()关闭SDK。

|  |  |  |
| --- | --- | --- |
| 方法：finishSelf ()  说明：用于用户登录 | | |
| 参数 | 类型 | 说明 |
| 无 |  |  |

## 账号系统AccountImpl

账号系统错误定义：AccountErrCode

### 登陆

|  |  |  |
| --- | --- | --- |
| 方法：login(String username, String password)  说明：用于用户登陆 | | |
| 参数 | 类型 | 说明 |
| username | String | 登陆用户名(手机或者邮箱) |
| password | String | 登陆密码 |
| 返回 |  |  |
| loginResult | LoginResult | 登陆返回的信息 |

LoginResult字段：

|  |  |  |
| --- | --- | --- |
| 字段 | 类型 | 说明 |
| loginsession | String | 登陆Session |
| userid | String | 用户id |
| nickname | String | 用户昵称 |
| iconPath | String | 头像地址 |
| loginip | String | 登陆ip地址 |
| logintime | String | 登陆时间 |
| sex | String | 性别 |

用户登录成功之后开发者可以将登陆返回的信息缓存在本地，下次就不需要在登陆，直接调用localLogin()接口就可以的。当Session失效之后，需要重新调用登陆login()接口重新登陆。

### 本地登录

如果APP已经登陆成功，在启动APP时候需要调用此方法进行SDK本地登录。

|  |  |  |
| --- | --- | --- |
| 方法：localLogin(LoginResult loginResult)  说明：用于载入本地缓存的登陆信息(是否有必要) | | |
| 参数 | 类型 | 说明 |
| loginResult | Object | 本地保存的登陆用户信息 |

### 注册

|  |  |  |
| --- | --- | --- |
| 方法：regist(RegistParam registParam, File fileIcon)  说明：注册用户 | | |
| 参数 | 类型 | 说明 |
| phoneOrEmail | String | 注册时填的手机或邮箱 |
| password | String | 注册密码 |
| nickname | String | 昵称 |
| country | String | 国家(选填) |
| countrycode | String | 国家码(手机注册时必填) 中国：+86 或者 86 |
| sex | String | 性别 AccountConstants. Sex.male 男  AccountConstants. Sex.female 女 |
| iconpath | String | 头像是url时填(选填) |
| code | String | 手机或邮箱收到的验证码 |
| fileIcon | File | 头像是本地文件时填(选填) |

### 获取手机验证码

|  |  |  |
| --- | --- | --- |
| 方法：sendRegVCode(String phone, String countrycode)  说明：获取手机验证码 | | |
| 参数 | 类型 | 说明 |
| phone | String | 手机 |
| countrycode | String | 国家码，发送手机验证码时必填 |
| 返回值 |  |  |
|  | BaseResult | error 0表示成功，其他失败  msg错误信息 |

### 获取邮箱验证码

|  |  |  |
| --- | --- | --- |
| 方法：sendRegVCode(String email)  说明：获取手机验证码 | | |
| 参数 | 类型 | 说明 |
| email | String | 邮箱地址 |
| 返回值 |  |  |
|  | BaseResult | error 0表示成功，其他失败  msg错误信息 |

### 获取用户昵称头像

|  |  |  |
| --- | --- | --- |
| 方法：getUserInfo (List<String> useridList)  说明：用于获取用户信息(昵称&头像) | | |
| 参数 | 类型 | 说明 |
| useridList | List<String> | 用户id，登陆后可获得 |
| 返回值 |  |  |
|  | GetUserInfoResult | error 0表示成功，其他失败  msg错误信息  List<UserInfo> 用户信息列表 |

### 修改用户头像

|  |  |  |
| --- | --- | --- |
| 方法：modifyUserIcon (File picFile)  说明：修改用户头像 | | |
| 参数 | 类型 | 说明 |
| picFile | File | 头像是本地文件 |
| 返回值 |  |  |
|  | ModifyUserIconResult | error 0表示成功，其他失败  msg错误信息  icon 新图标地址 |

### 修改用户昵称

|  |  |  |
| --- | --- | --- |
| 方法：modifyUserNickname (String nickName)  说明：修改用户昵称 | | |
| 参数 | 类型 | 说明 |
| nickname | String | 新昵称 |
| 返回值 |  |  |
|  | BaseResult | error 0表示成功，其他失败  msg错误信息 |

### 修改密码

|  |  |  |
| --- | --- | --- |
| 方法：modifyPassword (String oldPassword, String newPassword)  说明：修改密码 | | |
| 参数 | 类型 | 说明 |
| oldpassword | String | 原密码 |
| newpassword | String | 新密码 |
| 返回值 |  |  |
|  | BaseResult | error 0表示成功，其他失败  msg错误信息 |

### 发送修改手机号码验证码

|  |  |  |
| --- | --- | --- |
| 方法：sendModifyVCode (String phone, String countrycode)  说明：修改手机或邮箱发送验证码 | | |
| 参数 | 类型 | 说明 |
| phone | String | 原手机号码 |
| countrycode | String | 修改手机时必填 |
| 返回值 |  |  |
|  | BaseResult | error 0表示成功，其他失败  msg错误信息 |

### 发送修改邮箱验证码

|  |  |  |
| --- | --- | --- |
| 方法：sendModifyVCode (String email)  说明：修改手机或邮箱发送验证码 | | |
| 参数 | 类型 | 说明 |
| email | String | 原邮箱地址 |
| 返回值 |  |  |
|  | BaseResult | error 0表示成功，其他失败  msg错误信息 |

### 发送找回密码验证码

|  |  |  |
| --- | --- | --- |
| 方法：sendRetrieveVCode (String phoneOrEmail)  说明：密码找回发送验证码 | | |
| 参数 | 类型 | 说明 |
| phoneOrEmail | String | 手机或邮箱 |
| 返回值 |  |  |
|  | BaseResult | error 0表示成功，其他失败  msg错误信息 |

### 修改手机

|  |  |  |
| --- | --- | --- |
| 方法：modifyPhone(String phone, String countrycode, String vcode, String password)  说明：修改手机或邮箱 | | |
| 参数 | 类型 | 说明 |
| phone | String | 新的手机或者邮箱 |
| countrycode | String | 修改手机时必填 |
| vcode | String | 返回的验证码 |
| password | String | 密码 |
| 返回值 |  |  |
|  | BaseResult | error 0表示成功，其他失败  msg错误信息 |

### 修改邮箱

|  |  |  |
| --- | --- | --- |
| 方法：modifyEmail (String email, String vcode, String password)  说明：修改手机或邮箱 | | |
| 参数 | 类型 | 说明 |
| email | String | 新的手机或者邮箱 |
| code | String | 返回的验证码 |
| password | String | 密码 |
| 返回值 |  |  |
|  | BaseResult | error 0表示成功，其他失败  msg错误信息 |

### 重置密码

|  |  |  |
| --- | --- | --- |
| 方法：retrievePassword(String phoneOrEmail, String vcode, String newPassword)  说明：根据返回的验证码修改密码 | | |
| 参数 | 类型 | 说明 |
| phoneOrEmail | String | 手机或邮箱 |
| vcode | String | 验证码 |
| newpassword | String | 新密码 |
| 返回值 |  |  |
|  | LoginResult | error 0表示成功，其他失败  msg错误信息  用户信息 |

### 检查用户名密码是否正确

|  |  |  |
| --- | --- | --- |
| 方法：checkUserPassword(String password)  说明：检查密码是否正确 | | |
| 参数 | 类型 | 说明 |
| password | String | 登陆密码 |
| 返回值 |  |  |
|  | BaseResult | error 0表示成功，其他失败  msg错误信息 |

### 获取用户手机邮箱信息

|  |  |  |
| --- | --- | --- |
| 方法：getUserPhoneAndEmail ()  说明：获取登陆用户的手机与邮箱 | | |
| 参数 | 类型 | 说明 |
|  |  |  |
| 返回值 |  |  |
|  | GetUserPhoneAndEmailResult | error 0表示成功，其他失败  msg错误信息  userid 用户id  email 邮箱地址  phone 手机号码 |

## 设备控制ControllerImpl

设备控制系统错误码定义：ControllerErrCode

### 设备初始化

通过SDK控制设备或者是获取设备状态等操作，就需要调用addDevice()接口将设备添加到网络库中初始化。

#### 单个设备初始化

|  |  |  |
| --- | --- | --- |
| 方法： addDevice(DNADevice device)  说明： 将数据库中的设备加到sdk中 | | |
| 参数 | 类型 | 说明 |
| device | DNADevice | 设备信息 |

DNADevice设备信息:

|  |  |  |
| --- | --- | --- |
| 字段 | 类型 | 说明 |
| did | String | 设备ID |
| mac | String | Mac地址 |
| pid | String | 设备分类ID |
| name | String | 设备名称 |
| type | int | 设备类型 |
| lock | boolean | 是否锁定 |
| newconfig | boolean | 是否是新配置的设备 |
| password | int | 设备控制密码 |
| id | int | 设备控制ID |
| key | String | 设备控制密钥 |
| state | int | 设备状态 |

#### 批量设备初始化

|  |  |  |
| --- | --- | --- |
| 方法： addDevice(List< DNADevice > deviceList)  说明： 将数据库中的设备加到sdk中 | | |
| 参数 | 类型 | 说明 |
| deviceList | List< DNADevice > | 设备列表 |

如果需要将设备从SDK中移除可以调用removeDevice()，移除之后将不能再对设备进行一系列操作。

#### 单个设备删除

|  |  |  |
| --- | --- | --- |
| 方法： removeDevice(DNADevice device)  说明： 将sdk中的某个设备删除 | | |
| 参数 | 类型 | 说明 |
| device | DNADevice |  |

#### 批量设备删除

|  |  |  |
| --- | --- | --- |
| 方法： removeDevice(List< DNADevice > deviceList)  说明： 将sdk中的设备删除 | | |
| 参数 | 类型 | 说明 |
| deviceList | List< DNADevice > | 设备列表 |

#### 清空SDK设备

|  |  |  |
| --- | --- | --- |
| 方法： removeAllDevice ()  说明： 将sdk中的设备全部删除 | | |
| 参数 | 类型 | 说明 |
| 无 |  |  |

### 设备配置

#### 开始配置

|  |  |  |
| --- | --- | --- |
| 方法： deviceConfig(DeviceConfigParam deviceConfigParam)  deviceConfig(DeviceConfigParam deviceConfigParam, int timeout)  说明： 设备配置(同步方法，需用户自启线程) | | |
| 参数 | 类型 | 说明 |
| ssid | String | ssid |
| password | String | 密码 |
| gatewayaddr | String | 网关地址 |
| timeout | int | 配置超时时间，如果不加超时时间，则使用SDK初始化配置的时间或者默认时间。  如果添加0表示会一直配置 |

#### 取消配置

|  |  |  |
| --- | --- | --- |
| 方法： deviceConfigCancel ()  说明： 取消设备配置 | | |
| 参数 | 类型 | 说明 |
| 无 |  |  |

### 设备扫描

该接口用于启动局域网设备扫描功能，通过设置设备扫描回调函数将搜索到的新设备返回给应用层。

#### 开始设备扫描

|  |  |  |
| --- | --- | --- |
| 方法： startProbe()  说明：开启设备扫描 | | |
| 参数 | 类型 | 说明 |
|  |  |  |

#### 关闭设备扫描

|  |  |  |
| --- | --- | --- |
| 方法： stopProbe()  说明： 停止扫描设备 | | |
| 参数 | 类型 | 说明 |
|  |  |  |

#### 设备回调

|  |  |  |
| --- | --- | --- |
| 方法：setOnDeviceScanListener (DeviceScanListener scanListener)  说明： 设置设备扫描回调 | | |
| 参数 | 类型 | 说明 |
| DeviceScanListener |  |  |
| 返回 |  |  |
| device | DNADevice | 设备信息 |
| isNewDevice | boolean | true表示为新搜索出的设备，之前没有add到SDK中  false表示已经add到SDK中，设备信息有发生改变。 |

SDK搜索到新设备默认自动调用设备配对接口pair接口,获取到设备密钥并自动添加到网络库中，此时返回的设备信息中会带有id和key值。

如果不希望自动配对，需要重写DeviceScanListener中的shouldAdd方法，将返回值置为false即可，此时返回设备信息中不会有id和key，如果需要控制设备则需要手动去调用pair接口，配对成功之后调用addDevice添加到SDK中初始化。

开发者如果需要远程控制设备则需要将扫描到设备本地保存起来，下次使用APP的时候只需要将保存的设备信息addDevice到SDK中即可对设备进行远程控制。

### 设备配对

一般不需要主动调用，只要shouldAdd返回false时，才需要主动调用获取设备控制密钥。

|  |  |  |
| --- | --- | --- |
| 方法： pair()  说明： 设备绑定 | | |
| 参数 | 类型 | 说明 |
| DNADevice |  | 需要配对的设备 |

### 设备控制

设备控制有两个接口:

1. dnaControl依赖设备脚本
2. dnaPassthrough 透传接口无需脚本

#### 通过本地脚本控制

|  |  |  |
| --- | --- | --- |
| 方法： dnaControl(String did, String subDevInfoStr, StdControlParam stdControlParam)  dnaControl(String did, String subDevInfoStr, StdControlParam stdControlParam, ConfigParam configParam)  说明： 标准控制设置设备状态 | | |
| 参数 | 类型 | 说明 |
| did | String | 要控制设备的did 必填 |
| subDevInfoStr | String | 子设备信息(需要控制子设备的时候需要带上) |
| stdControlParam | StdControlParam | 控制设备参数 |
| configParam | ConfigParam | 控制超时时间，脚本路径配置  ConfigParam可以配置以下参数：  本地超时时间：CONTROLLER\_LOCAL\_TIMEOUT  远程超时时间：CONTROLLER\_REMOTE\_TIMEOUT  脚本路径：CONTROLLER\_SCRIPT\_PATH |

#### 透传控制

|  |  |  |
| --- | --- | --- |
| 方法： dnaPassthrough(String did, String subDevInfoStr, byte[] data)  dnaPassthrough(String did, String subDevInfoStr, byte[] data, ConfigParam configParam)  说明： 透传控制 | | |
| 参数 | 类型 | 说明 |
| did | String | 要控制设备的did 必填 |
| subDevInfoStr | String | 子设备信息(需要控制子设备的时候需要带上) |
| data | byte[] | 需要透传的指令 |
| configParam | ConfigParam | 控制超时时间配置  ConfigParam可以配置以下参数：  本地超时时间：CONTROLLER\_LOCAL\_TIMEOUT  远程超时时间：CONTROLLER\_REMOTE\_TIMEOUT |

### 查询设备状态

|  |  |  |
| --- | --- | --- |
| 方法： queryDeviceState(String did)  说明： 获取设备状态 | | |
| 参数 | 类型 | 说明 |
| did | String | 设备did |
| 返回 |  |  |
|  | int | 设备状态  DeviceState.UNKNOWN：还未获取到设备状态  DeviceState.LOCAL：设备和手机在同一个局域网  DeviceState.REMOTE：设备连接到服务器，和手机不在同一个局域网  DeviceState.OFFLINE: 设备未连接到服务器，不在线 |

### 查询局域网设备IP地址

|  |  |  |
| --- | --- | --- |
| 方法： queryDeviceIp ()  说明： 获取设备ip地址，没有则返回空 | | |
| 参数 | 类型 | 说明 |
| IPAddress | String | 设备ip地址 |

### 查询设备的Profile

|  |  |  |
| --- | --- | --- |
| 方法： queryProfile(String did)  queryProfile(String did, String profilePath)  说明： 获取原始的profile字符串 | | |
| 参数 | 类型 | 说明 |
| did | String | 设备did |
| profilePath | String | 脚本绝对路径 |

Profile 详细信息见附录。

### 查询固件版本

|  |  |  |
| --- | --- | --- |
| 方法： queryFirmwareVersion (String did)  queryFirmwareVersion(String did, ConfigParam configParam)  说明： | | |
| 参数 | 类型 | 说明 |
| did | String | 设备did |
| configParam | ConfigParam | 控制超时时间配置  ConfigParam可以配置以下参数：  本地超时时间：CONTROLLER\_LOCAL\_TIMEOUT  远程超时时间：CONTROLLER\_REMOTE\_TIMEOUT |

### 查询设备服务器时间

|  |  |  |
| --- | --- | --- |
| 方法： queryDeviceTime (String did)  queryDeviceTime (String did, ConfigParam configParam)  说明： | | |
| 参数 | 类型 | 说明 |
| did | String | 设备did |
| configParam | ConfigParam | 控制超时时间配置  ConfigParam可以配置以下参数：  本地超时时间：CONTROLLER\_LOCAL\_TIMEOUT  远程超时时间：CONTROLLER\_REMOTE\_TIMEOUT |

### 固件升级

|  |  |  |
| --- | --- | --- |
| 方法： updateFirmware (String did, String url)  updateFirmware (String did, String url, ConfigParam configParam)  说明： | | |
| 参数 | 类型 | 说明 |
| did | String | 设备did |
| url | String | 固件地址 |
| configParam | ConfigParam | 控制超时时间配置  ConfigParam可以配置以下参数：  本地超时时间：CONTROLLER\_LOCAL\_TIMEOUT  远程超时时间：CONTROLLER\_REMOTE\_TIMEOUT |

### 修改设备信息

|  |  |  |
| --- | --- | --- |
| 方法： updateDeviceInfo (String did, String name, boolean lock)  updateDeviceInfo (String did, String name, boolean lock, ConfigParam configParam)  说明： | | |
| 参数 | 类型 | 说明 |
| did | String | 设备did |
| name | String | 设备名称 |
| lock | boolean | 是否锁定设备 |
| configParam | ConfigParam | 控制超时时间配置  ConfigParam可以配置以下参数：  本地超时时间：CONTROLLER\_LOCAL\_TIMEOUT  远程超时时间：CONTROLLER\_REMOTE\_TIMEOUT |

### 设备通用定时

#### 定时列表查询

查询定时任务分为两个部分：

1. 查询定时信息列表queryTask()
2. 查询任务指令 queryTimerTaskData()/ queryPeriodTaskData()

|  |  |  |
| --- | --- | --- |
| 方法： queryTask(String did)  queryTask(String did, ConfigParam configParam)  说明： 查询定时和周期任务 | | |
| 参数 | 类型 | 说明 |
| Did |  | 需要查询的设备id |
| configParam | ConfigParam | 控制超时时间配置 |
| 返回 |  |  |
| timer | ArrayList<TimerInfo> | 定时任务列表 |
| period | ArrayList<PeriodInfo> | 周期任务列表 |

TimerInfo定时信息:

|  |  |  |
| --- | --- | --- |
| 字段 | 类型 | 说明 |
| index | int | 定时任务序号  修改/删除用 |
| enable | boolean | 是否启用 |
| year | int | 年 |
| month | int | 月 |
| day | int | 日 |
| hour | int | 时 |
| min | int | 分 |
| sec | int | 秒 |

PeriodInfo 周期任务信息：

|  |  |  |
| --- | --- | --- |
| 字段 | 类型 | 说明 |
| index | int | 周期任务序号  修改/删除用 |
| enable | boolean | 是否启用 |
| hour | int | 时 |
| min | int | 分 |
| sec | int | 秒 |
| repeat | List<Integer> | 任务的周期，1-7分别表示每周一到每周日,若repeat为空， 则表示该任务仅在最近一次匹配的时间执行一次。 |

定时任务指令查询:

|  |  |  |
| --- | --- | --- |
| 方法： queryTimerTaskData(String did, int index)  queryTimerTaskData(String did, int index , ConfigParam configParam)  说明： | | |
| 参数 | 类型 | 说明 |
| did | String | 设备id |
| index | int | 任务序号 |

周期任务指令查询:

|  |  |  |
| --- | --- | --- |
| 方法： queryPeriodTaskData (String did, int index)  queryPeriodTaskData (String did, int index , ConfigParam configParam)  说明： | | |
| 参数 | 类型 | 说明 |
| did | String | 设备id |
| index | int | 任务序号 |

#### 周期任务

##### 添加

|  |  |  |
| --- | --- | --- |
| 方法： addPeriodTask(String did, PeriodInfo task, StdData stdData)  addPeriodTask(String did, PeriodInfo task, StdData stdData, ConfigParam configParam)  说明： | | |
| 参数 | 类型 | 说明 |
| did | String | 设备id |
| periodInfo | PeriodInfo | 周期对象 |
| stdData | StdData | 发送的指令 |
| configParam | ConfigParam | 控制超时时间配置  ConfigParam可以配置以下参数：  本地超时时间：CONTROLLER\_LOCAL\_TIMEOUT  远程超时时间：CONTROLLER\_REMOTE\_TIMEOUT |

##### 修改

|  |  |  |
| --- | --- | --- |
| 方法： updatePeriodTask (String did, PeriodInfo task, StdData stdData)  updatePeriodTask (String did, PeriodInfo task, StdData stdData, ConfigParam configParam)  说明： | | |
| 参数 | 类型 | 说明 |
| did | String | 设备id |
| periodInfo | PeriodInfo | 周期对象 |
| stdData | StdData | 发送的指令 |
| configParam | ConfigParam | 控制超时时间配置  ConfigParam可以配置以下参数：  本地超时时间：CONTROLLER\_LOCAL\_TIMEOUT  远程超时时间：CONTROLLER\_REMOTE\_TIMEOUT |

##### 删除

|  |  |  |
| --- | --- | --- |
| 方法： delPeriodTask (String did, int index)  delPeriodTask (String did, int index, ConfigParam configParam)  说明： | | |
| 参数 | 类型 | 说明 |
| did | String | 设备id |
| index | int | 周期任务序号 |
| configParam | ConfigParam | 控制超时时间配置  ConfigParam可以配置以下参数：  本地超时时间：CONTROLLER\_LOCAL\_TIMEOUT  远程超时时间：CONTROLLER\_REMOTE\_TIMEOUT |

#### 定时任务

##### 添加

|  |  |  |
| --- | --- | --- |
| 方法： addTimerTask (String did, TimerInfo task, StdData stdData)  addTimerTask (String did, TimerInfo task, StdData stdData, ConfigParam configParam)  说明： | | |
| 参数 | 类型 | 说明 |
| did | String | 设备id |
| task | TimerInfo | 定时任务对象 |
| stdData | StdData | 发送的指令 |
| configParam | ConfigParam | 控制超时时间配置  ConfigParam可以配置以下参数：  本地超时时间：CONTROLLER\_LOCAL\_TIMEOUT  远程超时时间：CONTROLLER\_REMOTE\_TIMEOUT |

##### 修改

|  |  |  |
| --- | --- | --- |
| 方法： updateTimerTask (String did, TimerInfo task, StdData stdData)  updateTimerTask (String did, TimerInfo task, StdData stdData, ConfigParam configParam)  说明： | | |
| 参数 | 类型 | 说明 |
| did | String | 设备id |
| task | TimerInfo | 定时任务对象 |
| stdData | StdData | 发送的指令 |
| configParam | ConfigParam | 控制超时时间配置  ConfigParam可以配置以下参数：  本地超时时间：CONTROLLER\_LOCAL\_TIMEOUT  远程超时时间：CONTROLLER\_REMOTE\_TIMEOUT |

##### 删除

|  |  |  |
| --- | --- | --- |
| 方法： delTimerTask (String did, int index)  delTimerTask (String did, int index, ConfigParam configParam)  说明： | | |
| 参数 | 类型 | 说明 |
| did | String | 设备id |
| index | int | 定时任务序号 |
| configParam | ConfigParam | 控制超时时间配置  ConfigParam可以配置以下参数：  本地超时时间：CONTROLLER\_LOCAL\_TIMEOUT  远程超时时间：CONTROLLER\_REMOTE\_TIMEOUT |

### 资源下载

SDK提供设备UI资源包和控制脚本版本查询、下载，版本维护需要应用层自己维护。

#### 脚本版本查询

|  |  |  |
| --- | --- | --- |
| 方法： queryScriptVersion(String pid)  说明： 查询 | | |
| 参数 | 类型 | 说明 |
| pid | String | 设备分类ID |
| 返回 |  |  |
| version | String | 脚本版本 |

#### 脚本下载

|  |  |  |
| --- | --- | --- |
| 方法： downloadScript (String pid)  说明： 查询 | | |
| 参数 | 类型 | 说明 |
| pid | String | 设备分类ID |
| 返回 |  |  |
| savePath | String | 下载完成之后的脚本路径 |

#### 查询pid对应脚本的路径

|  |  |  |
| --- | --- | --- |
| 方法： queryScriptPath (String pid)  说明： 查询 | | |
| 参数 | 类型 | 说明 |
| pid | String | 设备分类ID |
| 返回 |  |  |
| path | String | 下载完成之后的脚本路径 |

#### UI包版本查询

|  |  |  |
| --- | --- | --- |
| 方法： queryUIVersion (String pid)  说明： 查询 | | |
| 参数 | 类型 | 说明 |
| pid | String | 设备分类ID |
| 返回 |  |  |
| version | String | UI版本 |
| uiid | String | UIid 下载UI包需要带上UIID |

#### UI包下载

|  |  |  |
| --- | --- | --- |
| 方法： downloadUI (String pid)  说明： 查询 | | |
| 参数 | 类型 | 说明 |
| pid | String | 设备分类ID |
| uiid | String | UIid 下载UI包需要带上UIID |
| 返回 |  |  |
| savePath | String | 下载完成之后的UI路径 |

#### 查询pid对应UI的路径

|  |  |  |
| --- | --- | --- |
| 方法： queryUIPath (String pid)  说明： 查询 | | |
| 参数 | 类型 | 说明 |
| pid | String | 设备分类ID |
| 返回 |  |  |
| path | String | UI路径 |

## 数据采集

|  |  |  |
| --- | --- | --- |
| 方法： onCreate(Activity activity)  说明： 用于记录Android界面生命 | | |
| 参数 | 类型 | 说明 |
|  |  |  |

|  |  |  |
| --- | --- | --- |
| 方法： onDestroy(Activity activity)  说明： 用于记录Android界面生命 | | |
| 参数 | 类型 | 说明 |
|  |  |  |

|  |  |  |
| --- | --- | --- |
| 方法： onResume(Activity activity)  说明： 用于记录Android界面生命 | | |
| 参数 | 类型 | 说明 |
|  |  |  |

|  |  |  |
| --- | --- | --- |
| 方法： onPause(Activity activity)  说明： 用于记录Android界面生命 | | |
| 参数 | 类型 | 说明 |
|  |  |  |

事件统计

|  |  |  |
| --- | --- | --- |
| 方法： onEvent(String eventId)  onEvent(String eventId , String eventTag)  onEvent(String eventId, String eventTag, Map<String, String> data)  说明： 开发者自定义事件 | | |
| 参数 | 类型 | 说明 |
| eventId | String | 用于分类事件 |
| eventTag | String | 用于分类事件 |
| data | 键值对 | 用于开发者自定义上传数据 |

# 错误码

|  |  |  |
| --- | --- | --- |
| 错误码 | 错误描述 | 备注 |
| 0 | 成功，无错误 |  |
| -1000 | 网络超时 | 检查网络状态，并测试ping的延时时间 |
| -1001 | 设备不在局域网中 | 设备配对接口必须在局域网中才能进行操作。 |
| -1002 | 用户自行取消easyconfig配置 |  |
| -1003 | 系统调用失败 | 创建网络套接字失败 |
| -1004 | 系统调用失败 | 设置网络套接字属性失败 |
| -1005 | 系统调用失败 | 网络套接字发送数据包失败 |
| -1006 | 系统调用失败 | 操作网络套接字失败 |
| -1007 | 接收到的数据包长度有误 |  |
| -1008 | 接收到的数据包校验有误 |  |
| -1009 | 接收到的数据包消息类型有误 |  |
| -1010 | 接收到的加密数据包长度有误 |  |
| -1011 | 接收到的加密数据包校验失败 |  |
| -1012 | 设备控制ID有误 | 设备复位，且未重新进行设备配对操作 |
| -1013 | 域名解析失败 | 检查网络，若控制终端连接在路由器，可以尝试将路由器的DNS设置为8.8.8.8 |
| -1014 | SDK初始化失败 | 具体错误信息可见返回json中的msg |
| -1015 | 非法的json传参 | 具体错误信息可见返回json中的msg |
| -1016 | 传参json中的数据类型有误 | 具体错误信息可见返回json中的msg |
| -1017 | 传参设备信息的json有误 | 具体错误信息可见返回json中的msg |
| -1018 | 系统调用失败 | 动态分配内存失败 |
| -1019 | 暂未使用 |  |
| -1020 | 解析脚本有误 | 文件不存在，或者脚本语法有误。  请查看SDKInit初始化时filepath是否正确。 |
| -1021 | 暂未使用 | 保留 |
| -1022 | 不支持的命令类型 |  |
| -1023 | License被禁止使用 | 非法操作的license将被加入黑名单 |
| -1024 | 操作过于频繁 |  |
| -1025 | SDK不支持远程操作 | 调用SDKInit时，若localctrl为true，则所有的远程操作都会报该错误 |
| -1026 | 发送或接收到的数据有误 |  |
| -1027 | 暂未使用 | 保留 |
| -1028 | 电控板未及时响应 | 检查Wi-Fi模块与电控之间的链路 |
| -1029 | 不存在对应的通信服务器 | 请联系BroadLink Co., Ltd.解决 |
| -1030 | 数据缓存过小 | 一般在脚本出错时发生 |
| -1031 | 脚本解析失败 | 标准指令到电控指令失败 |
| -1032 | 脚本解析失败 | 电控指令到标准指令失败 |
| -1033 | 系统调用出错 | 所有的系统调用失败都返回该错误（前面具体定义的除外） |
| -1034 | SDK正在注册 | 等待注册完成 |
| -1035 | SDK未认证 | 先调用SDKAuth操作 |
| -1036 | 非法License | 无效的license |
| -1037 | SSL连接服务器失败 | 请检查网络状况 |
| -1038 | SSL与服务器握手失败 | 请检查网络状况 |
| -1039 | SSL证书无效 | 请联系BroadLink Co., Ltd.解决 |
| -1040 | SSL写数据失败 | 请检查网络状况 |
| -1041 | SSL读数据头失败 | 请检查网络状况 |
| -1042 | SSL读数据内容失败 | 请检查网络状况 |
| -2001 | SDK认证超时，请重新认证 | 重新调用SDKAuth操作 |
| -2002 | 非法的认证码 | 请确认认证服务器的合法性 |
| -2003 | 控制的设备未绑定至云端 | 先调用deviceBindWithServer |
| -2004 | 绑定设备到云端请求的参数不合法 | 确认参数的正确性 |
| -2005 | 请求的命令不支持 | 不支持的远程控制命令 |
| -2006 | 网络故障 | 接入服务器网络故障，请联系BroadLink Co., Ltd.解决 |
| -2007 | 远程控制数据加密类型错误 | 请联系BroadLink Co., Ltd.更新SDK |
| -2008 | 远程控制数据校验失败 |  |
| -2009 | License未在云端注册 | License非法 |
| -2010 | 远程控制数据解密失败 |  |
| -2011 | 认证码不匹配 | 重新调用SDKAuth操作 |
| -2012 | UID非法 | License非法，请联系BroadLink Co., Ltd.解决 |
| -2013 | 账号无法通过校验 | 提供的account\_id和account\_session错误，无法通过账号系统的认证，请检查相应的账号信息。 |
| -2014 | 远程认证数据解密失败 |  |
| -2015 | 绑定到云端的设备数量超过限制 | 请连续BroadLink Co., Ltd.解决 |
| -2016 | License不支持当前绑定到云端的设备类型 | 请联系我们处理 |
| -2017 | 控制时未提供account\_id |  |
| -3101 | 未知异常 |  |
| -3102 | SDK未登录 | 需要调用SDK localLogin()方法 |
| -3103 | 设备不存在 | 未将设备添加到网络库中，就进行操作。需要调用addDevice()方法 |
| -3104 | 获取Token异常 |  |
| -3105 | 查询资源异常 |  |
| -3106 | 无法找到请求的资源 |  |
| -3107 | Body格式错误 |  |
| -3108 | 数据缺少必要字段 |  |
| -3109 | Url Token过期 |  |
| -3110 | 请求方式错误 |  |
| -3111 | 用户主动停止 |  |
| -3112 | 解压压缩包失败 |  |

# FAQ

**SDK**认证为什么会失败**?**

1. License有问题。这种情况下会返回-1023的错误，请通过DNAKit来申请正确的控制License。务必保证您的APP的包名与申请License时输入的一致。
2. 网络有限制。这种情况下会返回-1034的错误，请确认路由器没有对\*.ibroadlink.com和\*.broadlink.com.cn的域名进行过滤。SDK通过服务器的12345端口进行认证，请确保没有重定向或者封闭该端口。上述问题都排除后，请联系BroadLink Co., Ltd.并且提供您的License供检查。

设备无法配置入网？

1. 请务必保证设备处于配网模式。
2. 请确认手机接入的是2.4G的网络，设备当前不支持5G(包括802.11n的5G和802.11ac)。
3. 若路由器为802.11n或者802.11bgn模式，请尝试将路由器设置为802.11bg模式，待设备配置成功之后，您可以重新改回去。这是为了保证路由器没有工作在802.11n的5G频段上。
4. 不支持路由器使用wep共享密钥的加密方式，请更换其他加密方式，推荐使用wpa2。
5. 若您将SDK运行在双网卡的宿主机上，请确认配网数据包是通过无线网卡发送的。Linux平台下可以通过添加路由表route add –host 255.255.255.255 dev br0 和route add –net 224.0.0.0 netmask 240.0.0.0 dev br0来进行配置，其中br0为您的无线网卡的名称。

设备已经成功入网了，为什么无法搜索到设备?

1. 请确认设备与控制终端处于同一个局域网中，设备发现功能仅限于发现局域网内的设备。
2. 请确认您申请的License是否具有对该类型设备的控制权限。
3. 请确认您的路由器或者AP没有开启AP隔离(局域网内隔离)模式。
4. 若您将SDK运行在双网卡的宿主机上，请确认设备发现的数据包是通过无线网卡发送的。Linux平台下可以通过添加路由表route add –host 255.255.255.255 dev br0 和route add –net 224.0.0.0 netmask 240.0.0.0 dev br0来进行配置，其中br0为您的无线网卡的名称。
5. 若是概率性的搜索不到设备，请您检查您的局域网网络状况，由于设备发现使用的是局域网广播的原理，从协议层就无法保证百分百成功，建议您在局域网中且APP处于前台时，开启定时器定时进行设备发现，建议间隔时间为3秒钟。

设备搜索到了，设备配对失败了？

1. 请确保设备与控制终端处于同一局域网中，设备配对仅限局域网内操作。
2. 请确保调用配对接口时，设备信息的完整性。
3. 请检查内网网络情况，若内网网络较差，可适当加大ltimeout的值。

设备控制失败，返回-7错误?

1. 请确保设备已经成功进行配对操作，并且调用控制接口时，设备信息中已经包含了配对成功时返回的id和key。
2. 设备已经被认为复位了，设备复位之后，必需和设备再同一个局域网内再重新配对一次，并及时更新id和key。

设备控制失败，返回-1000错误?

1. 请检查网络状况，局域网控制可适当加大ltimeout的值，远程控制可适当加大rtimeout的值。
2. 若您是使用BroadLink的模块进行产品开发，请保证您的通信数据是否正确，您可以尝试通过透传方式先调通协议，然后再通过标准接口调试您的脚本，直到数据包正确。

如何判断一个设备是否处于局域网中？

由于设备发现利用了局域网广播的原理，设备发现并不能保证每次都可以搜索到设备，根据我们的经验值来说，APP在前台时，并且处于Wi-Fi连接状态时，APP可以启动一个定时器，每3秒钟进行一次设备发现操作，若连续三次搜索不到设备，则认为设备已经离开局域网，反之可认为设备还处于局域网中。APP在后台时或者控制终端没有连接到Wi-Fi时，都可以认为设备不在局域网中。由于广播的原理，在设备离开局域网后，并不能立即知道，必然会存在一个超时时间，在这个超时时间内，会无法控制该设备。

# 附录

## 产品profile格式说明

|  |
| --- |
| {  "srvs" : ["1.1.1"], //产品的服务编码  “funcs”: [“stdctrl”, “periodctrl”, “timerctrl”, “ota”, “cyclectrl”],  "suids" : [{  "suid" : "", //标准UI编码  "intfs" : { //支持该UI编码下的接口说明  "pwr" : [ //支持的参数名称，一个参数可能有多组，以数组表示  {  "idx" : 1, //表示第几路  "act" : 3, //参数属性：1: read only 2: write only 3:read & write  "in" : [ //参数值说明  1, //第一个值表示数据类型 1:枚举类型 2: 连续值  1, //表示第一个枚举值  2 //表示第二个枚举值  ]  }  ],  "temp" : [ //支持的参数名称，一个参数可能有多组，以数组表示  {  "idx" : 1, //表示第几路  "act" : 3, //参数属性：1: read only 2: write only 3:read & write  "in" : [ //参数值说明  2, //第一个值表示数据类型 1:枚举类型 2: 连续值  1, //连续值的最小值  100, //连续值的最大值  1, //连续值的步长，例如步长5，则值以5，10，15……存在  1 //连续值的倍数,例如倍数100，则100表示协议中的1  ]  }  ]  }  }  ],  "desc" : {  "cat" : "40", //产品类别编码  "vendor" : "1149", //产品所属公司编码  "model" : "26000", //产品型号  "pid" : "00000000000000000000000090650000" //产品pid  },  "ver" : "" //profile版本号  } |