Rokid Mobile iOS SDK

NO.	修改	版本	说明
01	新增	v1.0.0	SDK 文档初版
02	修改	v1.0.0	SDK 初完成版
03	增加	v1.0.1	添加 Skill 闹钟、提醒
04	增加	v1.0.2	新增 Web Bridge 接入说明

一. SDK导入方式

目前只支持手动添加,后续会添加CocoaPods

- 暂时只支持 Debug 版本的 Framework 包
- 提供了测试工程，在 RokidSDKDemo 下，包括 Objc 和 swift(暂未实现) 两个版本

1、使用时的工程设置

- Podfile

```
Demo 工程中提供了 Podfile，里面已经写好了依赖的工程
```

- 工程设置

```
在工程的 Embed Library 中加上我们的 framework
在 Framework Search Path 中把 framework 所在路径设置好
```

- 注意目前只有 Debug 版本，且运行时注意是模拟器或真机环境

二、Demom 工程

1、介绍

Demo 有 Swift 和 OC 版本，请开发者各取所需。

2、GitHub 地址：

https://github.com/Rokid/RokidMobileSDKiOSDemo

三. SDK介绍

1、SDK 初始化

1.1 初始化

- 异步初始化RokidMobileSDK, 请确保初始化成功，否则其余API都会失败.

  参数说明：

字段	类型	必须?	说明
appKey	String	是	Rokid 发放的 appKey
appSecret	String	是	Rokid 发放的 appSecret

示例代码：

```
// complete: 初始化结果，成功success = true，失败success = false
RokidMobileSDK.shared.initSDK(appKey: String,
                              appSecret: String,
                              escaping complete:(success: Bool) -> Void)
```

1.2 开启日志

此API 未启用

示例代码：

```
// Log默认关闭
RokidMobileSDK.log.enable = true;
```

```
Log DEMO:

RokidMobileSDK:开始SDK初始化：appKey:xxxxxxxx, appSecret:yyyyyy
RokidMobileSDK:初始化0:开始验证app...
RokidMobileSDK:初始化1:开始生成本地权限...
RokidMobileSDK:初始化0：校验app成功（失败,失败原因：无效网络）
RokidMobileSDK:初始化1:SDK权限已生成，具有权限：账户管理-true、设备管理-true

RokidMobileSDK: 开始登陆: userId:xxxxxxxx, token: yyyyyy
RokidMobileSDK: 登陆成功(失败, 失败原因：token校验失败)
```

2、账户模块 Account

2.0 临时调试登录接口

```
// 临时登录接口，不需要账号密码
RokidMobileSDK.account.tempLogin(name, password, complete)
```

2.1 授权登陆

- 接口说明：需要传入token和用户id，再监听回调

参数说明:

字段	类型	必须?	说明
userId	String	是	用户id
token	String	否	用户登录token

示例代码:

```
// token 有些平台没有
RokidMobileSDK.account.login(userId: String, token: String, complete:((RKError?) -
> Void))
```

2.2 退出登陆

示例代码：

```
RokidMobileSDK.account.logout()
```

3、配网模块 Bind

3.1 获取蓝牙开启状态,未授权时先授权再check

示例代码

```
RokidMobileSDK.binder.getBLEStatus()
```

3.2 开启蓝牙扫描

参数说明

字段	类型	必须?	说明
type	String	是	设备名称类型前缀

示例代码

```
RokidMobileSDK.binder.startBLEScan(type: String, onNewDeviceCallback: (BTDevice)->
Void) ->RKError?
```

3.3 停止蓝牙扫描

示例代码:

```
RokidMobileSDK.binder.stopBLEScan()
```

3.4 连接设备

接口说明 接口需传入蓝牙名称（蓝牙address重启后会变）

参数说明

字段	类型	必须?	说明
name	String	是	设备名称

示例代码

```
// name为RokidMobileSDK.BTDevice.name
RokidMobileSDK.binder.connectBLEDevice(name: String, complete:(RKError?)->Void)
```

3.5 蓝牙配网

参数说明

字段	类型	必须?	说明
binderData	DevicebinderData	是	蓝牙发送信息

示例代码

```
let binderData: DevicebinderData = DevicebinderData()
binderData.userId("your ueserId")      //绑定的masterId（不能为空）
binderData.wifiPwd("your wifiPwd")     //wifi密码（可以为空）
binderData.wifiSsid("your wifiSsid")   //wifi名字（可以为空）
binderData.wifiBssid("your wifiBssid") //wifi地址（可以为空）

RokidMobileSDK.binder.sendBLEBinderData(binderData: DevicebinderData, complete: (R
KError?)->Void)
```

3.6 监听蓝牙状态改变

参数说明

字段	类型	必须?	说明

statusChange	CBCentralManagerState) -> Void	是	状态改变回调

示例代码

```
RokidMobileSDK.binder.onBLEStatusChange(statusChange: @escaping (CBCentralManagerS
tate) -> Void)
```

4、设备管理模块 Device

4.1 获取设备列表

接口说明

```
目前获取服务端masterId对应的设备列表，
注意 RKDevice此时里面只有 rokiId,rokidNick,basic_info信息，
底层会默认给用户选择一个当前设备，逻辑图如下：
```

示例代码

```
// 设备信息 icon、name、alive
RokidMobileSDK.device.queryDeviceList(complete:(RKError?,[RKDevice])->Void)
```

```
[
    {
        //设备昵称
        "rokidNick": "xxx",
        //设备Id
        "rokiId": "0011111111111111",
        "basic_info": {
            //设备地区
            "region": "CN",
            "sn": "02010217020001ED",
            //系统版本号
            "ota": "2.2.2-20171027.091126",
            //mac地址
            "mac": "02:00:00:00:00:00",
            "ip": "183.129.185.66",
            //局域网IP
            "lan_ip":"192.168.1.156",
            //自定义TTS音色
            "ttsList": "[{\"name\":\"111777\",\"vibrato\":0,\"speed\":0,\"formant\
```

```
":0,\"tone\":0},{\"name\":\"234566777777\",\"vibrato\":4,\"speed\":5,\"formant\":-
5,\"tone\":5},{\"name\":\"5555\",\"vibrato\":-5,\"speed\":5,\"formant\":-5,\"tone\
":-5}]",
            //系统当前选择TTS音色
            "tts": "{\"formant\":-3,\"speed\":-2,\"tone\":1,\"ttsName\":\"蜡笔小新\
"}"
        }
    }
]
```

4.2 获取 设备基本信息包括：ip、局域网ip、mac、nick、cy、sn、version

参数说明

字段	类型	必须?	说明
deviceId	String	是	设备Id

接口定义

```
RokidMobileSDK.device.getBasicInfo(deviceId: String) -> [String: Any]?
```

4.3 获取设备的 Loaction 信息

参数说明

字段	类型	必须?	说明
deviceId	String	是	设备Id
completion		是	结果回调

接口定义

```
RokidMobileSDK.device.getLocation(deviceId: String, completion: @escaping (_ error
: RKError?, _ location: Location?) -> Void)
```

4.4 更新设备的 Location 信息

参数说明

字段	类型	必须?	说明
deviceId	String	是	设备Id
location	Location	是	位置信息
completion		是	结果回调

接口定义

```
RokidMobileSDK.device.updateLocation(deviceId: String, location: Location, completion: @escaping (_ error: RKError?) -> Void)
```

4.5 更新当前设备的昵称

参数说明

字段	类型	必须?	说明
deviceId	String	是	设备Id
newNick	String	是	新昵称
completion		是	结果回调

接口定义

```
RokidMobileSDK.device.updateNick(deviceId: String, newNick: String, completion: @escaping (_ error: RKError?) -> Void)
```

4.6 获取系统版本信息

参数说明

字段	类型	必须?	说明
deviceId	String	是	设备Id
completion		是	结果回调

接口定义

```
RokidMobileSDK.device.getVersion(deviceId: String, completion: @escaping (_ error:
 Error?, _ versionInfo: RKDeviceVersionInfo?) -> Void )
```

4.7 开始系统升级

参数说明

字段	类型	必须?	说明
deviceId	String	是	设备Id

接口定义

```
RokidMobileSDK.device.startSystemUpdate(deviceId: String) -> Bool
```

4.8 设备恢复出厂设置

参数说明

字段	类型	必须?	说明
deviceId	String	是	设备Id

接口定义

```
RokidMobileSDK.device.resetDevice(deviceId: String) -> Bool
```

4.9 解绑设备

参数说明

字段	类型	必须?	说明
deviceId	String	是	设备Id

completion		是	结果回调

接口定义

```
RokidMobileSDK.device.unbindDevice(deviceId: String, completion: @escaping (_ erro
r: RKError?) -> Void)
```

4.10 设置当前设备(本地缓存)

参数说明

字段	类型	必须?	说明
device	RKDevice	是	设备Entity

接口定义

```
RokidMobileSDK.device.setCurrentDevice(device: RKDevice)
```

4.11 获取当前设备(本地缓存)

参数说明

字段	类型	必须?	说明

接口定义

```
RokidMobileSDK.device.getCurrentDevice() -> RKDevice?
```

4.12 通过 id 获取 device 信息

参数说明

字段	类型	必须?	说明
deviceId	String	是	设备Id

接口定义

```
RokidMobileSDK.device.getDevice(deviceId: String) -> RKDevice?
```

5、Home 模块

5.1 获得 card 列表

参数说明

字段	类型	必须?	说明
maxDbId	Int	是	card 的 db id
pageSize	Int	否	页大小

接口定义

```
RokidMobileSDK.home.getCardList(maxDbId: Int, completion: @escaping (_ error: RKEr
ror?, _ cardList: [RKCard]?) -> Void) -> Void
```

```
RokidMobileSDK.home.getCardList(maxDbId: Int, pageSize: Int = 20, completion: @esc
aping (_ error: RKError?, _ cardList: [RKCard]?) -> Void) -> Void
```

5.2 发送 ASR

发送 ASR，就是 发送一条 能让设备立即执行的一条指令。

参数说明

字段	类型	必须?	说明
asr	String	是	ASR 内容
device	RKDevice	是	需要发送的设备

接口定义

```
RokidMobileSDK.home.sendAsr(asr: String, to device: RKDevice)
```

5.3 发送TTS

发送 TTS，就是 发送一条 能让设备立即说的 内容。

参数说明

字段	类型	必须?	说明
device	String	是	需要发送的设备
tts	String	是	让设备说的话

示例代码:

```
RokidMobileSDK.home.sendTts(tts: String, to device: RKDevice)
```

6、Skill 模块

6.1 闹钟

6.1.1 获取闹钟列表

获取设备的闹钟列表

接口定义

```
RokidMobileSDK.skill?.alarm.getList(deviceId: String,
                                    completion: @escaping (error: RKError?, alarms: [RKAla
rm]?) -> Void)
```

RKAlarm 字段说明:

参数	类型	必要?	说明
id	int	是	闹钟Id

year	int	是	年
month	int	是	月
day	int	是	日
hour	int	是	小时
minute	int	是	分钟
date	String	是	重复模式的文案
ext	Map	是	扩展字段，根据自己业务进行扩展

注：目前只有Lua版Linux系统支持该字段
ext字段是手机App与系统通信特有的字段，添加或修改时传入，获取列表时原样返回，以下划线_开始的key是预定义key。
ext字段可以为空；因为暂时没有删除字段的接口，所以修改时(SpecificTime)需要传入所有的key和value。
系统不支持时间完全相同的闹钟，所以更新和删除时不会校验ext是否匹配。

名称	类型	描述
_ringtone	string	闹钟铃声地址，会覆盖全局的闹钟主题

第三方需求可以由他们自定义字段，比如小雅小雅的标签需求

6.1.2 新建闹钟

新建一个闹钟

接口定义

```
RokidMobileSDK.skill?.alarm.create(deviceId: String, alarm: RKAlarm)
```

repeatType 解释:

```
RKAlarmRepeatModeOnce : 仅此一次
RKAlarmRepeatModeEveryday : 每天
RKAlarmRepeatModeWeekday : 工作日
RKAlarmRepeatModeWeekend : 每周末
RKAlarmRepeatModeEveryMonday : 每周一
```

```
RKAlarmRepeatModeEveryTuesday : 每周二
RKAlarmRepeatModeEveryWednesday : 每周三
RKAlarmRepeatModeEveryThursday : 每周四
RKAlarmRepeatModeEveryFriday : 每周五
RKAlarmRepeatModeEverySaturday : 每周六
RKAlarmRepeatModeEverySunday : 每周日
```

6.1.3 删除闹钟

删除一个闹钟

接口定义

```
RokidMobileSDK.skill?.alarm.delete(deviceId: String, alarm: RKAlarm)
```

6.1.4 更新闹钟

更新一个闹钟

接口定义

```
RokidMobileSDK.skill?.alarm.update(deviceId: String, alarm: RKAlarm, to: RKAlarm)
```

6.2 提醒

6.2.1 获取提醒列表

请求获取设备上的提醒列表:

```
RokidMobileSDK.skill?.remind.getList(deviceId: String,
                          completion: @escaping (_ error: RKError?, _ reminds: [R
KRemind]?) -> Void)
```

RKRemind 字段说明:

参数	类型	必要?	说明

id	int	是	闹钟Id
year	int	是	年
month	int	是	月
day	int	是	日
hour	int	是	小时
minute	int	是	分钟
content	String	是	提醒内容

6.2.2 删除提醒

删除一个提醒:

```
RokidMobileSDK.skill?.remind.delete(deviceId: String, remind: RKRemind)
```

7、RKWebBridge

7.1 简介

如果需要 接入 智能家居 等一些 H5 页面，需要接入 RKWebBridge，否则 H5 页面无法正常使用。

7.2 快速接入

SDK 提供了 快速接入方法 供开发者集成，请安装下面 Demo 代码使用即可，具体 Native UI View 组件可根据APP业务需求进行实现。

```
class WebviewViewController: UIViewController {

    var webbridge: RKWebBridge?
    var webView: WKWebView?

    override func viewDidLoad() {
        super.viewDidLoad()
```

```
        self.webbridge = RKWebBridge.injectWebBridge(to: self.webView!)
        self.webbridge?.setAppDelegate(delegate: self)
        self.webbridge?.setViewDelegate(delegate: self)
    }

}

extension WebviewViewController: RKBridgeModuleAppDelegate  {

    // 关闭当前页面
    func close() {
    }

    // 在当前的 webview , 打开Url
    func open(title: String, urlStr: String) {
    }

    // 在一个新的 ViewController 中打开Url
    func openNewWebView(title: String, urlStr: String) {
    }

    // 使用外部浏览器 打开Url
    func openExternal(urlStr: String) {
    }

}

extension WebviewViewController: RKBridgeModuleViewDelegate  {

    // 显示 Toast
    func showToast(message: String) {
    }

    // 显示 加载中UI组件
    func showLoading(message: String) {
    }

    // 隐藏 加载中UI组件
    func hideLoading() {
    }

    // 设置 标题栏标题
    func setNavigationBarTitle(title: String) {
    }

    // 设置 标题栏风格
    func setNavigationBarStyle(style: String) {
    }

    // 设置 标题栏 右侧按钮
    func setNavigationBarRight(button: [String : Any]) {
    }

    // 设置 标题栏 右侧按钮小红点状态
```

```
    func setNavigationBarRightDotState(state: Bool) {
    }

    // 显示 异常UI组件
    func errorView(state: Bool, retryUrl: String) {
    }

}
```

（1）标题栏风格:
请标题栏风格 根据业务需求设置。

（2）标题栏 右侧 RightButton 参数解释:

名称	类型	必须?	描述
icon	String	否	按钮图片
text	String	否	按钮标题
loadSelf	Bool	否	默认`false`: 是否在当前页面打开
targetUrl	String	是	`close`: 关闭当前页面 <url>: 打开这个url

SDK 通知

接口定义

```
NotificationCenter.rokid.addObserver()
```

通知定义

通知名都定义在 RKNotificationName

通知名	说明
.CurrentDeviceUpdated	当前设备改变
.DeviceStatusUpdated	设备在线状态改变
.DeviceListUpdated	设备列表改变 增加或解绑
.CardListUpdated	card列表改变，增加或减少

.AlarmVolumeChanged	音量变化
.ShouldLogout	被登出