智能家居APP JSAPI接口文档

|  |  |  |  |
| --- | --- | --- | --- |
| 拟制 | 华为 | 日期 | 2017/06/20 |
| 审核 |  | 日期 |  |
| 批准 |  | 日期 |  |



华为技术有限公司

版权所有 侵权必究

修订记录

|  |  |  |  |
| --- | --- | --- | --- |
| 日期 | 修订版本 | 修改描述 | 作者 |
| 2017-06-20 | 1.0 | 初稿完成 | 华为 |
| 2017-06-27 | 1.1 | 新增DEVICEINFO-API | 华为 |
| 2017-07-10 | 1.2 | 将js接口格式修改为简单数据结构 | 华为 |
| 2017-07-12 | 1.3 | 新增设备开发部分功能接口 | 华为 |
| 2017-07-13 | 1.5 | 新增场景接口及扩展接口 | 华为 |
| 2017-07-13 | 1.7 | 新增跳转接口 | 华为 |
| 2017-09-11 | 1.8.1 | 新增公共控件接口，以及去掉一个容易的设备状态接口 | 华为 |

[智能家居APP JSAPI接口文档 1](#_Toc515099317)

[1. 概述 4](#_Toc515099318)

[2. JSAPI列表 5](#_Toc515099319)

[1) 设备接口 5](#_Toc515099320)

[2.1.1 获取缓存设备状态全集 5](#_Toc515099321)

[2.1.2 获取设备状态 6](#_Toc515099322)

[2.1.3 获取设备状态（带proid方式） 7](#_Toc515099323)

[2.1.4 下发命令到设备 8](#_Toc515099324)

[2.1.5 下发命令到设备（带proId方式） 8](#_Toc515099325)

[2.1.6 获取设备事件通知 9](#_Toc515099326)

[2.1.7 获取设备历史信息 10](#_Toc515099327)

[2) 路由器接口 11](#_Toc515099328)

[2.2.1 路由器通用交互接口-获取资源 11](#_Toc515099329)

[2.2.2 路由器通用交互接口-下发配置 12](#_Toc515099330)

[3) 场景接口](#_Toc515099331)~~[（暂不需要）](#_Toc515099331)~~ [13](#_Toc515099331)

[2.3.1 获取场景列表 13](#_Toc515099332)

[2.3.2 创建手动场景 14](#_Toc515099333)

[2.3.3 修改手动场景 15](#_Toc515099334)

[2.3.4 删除手动场景 16](#_Toc515099335)

[2.3.5 创建自动规则 17](#_Toc515099336)

[2.3.6 修改自动规则 19](#_Toc515099337)

[2.3.7 删除自动规则 20](#_Toc515099338)

[4) 扩展接口 21](#_Toc515099339)

[2.4.1 拉起商城页面（未做） 21](#_Toc515099340)

[2.4.2 拉起酷玩页面（未做） 22](#_Toc515099341)

[2.4.3](#_Toc515099342) ~~[拉起native自动场景添加页面（暂不需要）](#_Toc515099342)~~ [22](#_Toc515099342)

[2.4.4](#_Toc515099343) ~~[返回首页（没有作用，废弃）](#_Toc515099343)~~ [23](#_Toc515099343)

[2.4.5 拉起时间选择弹窗 23](#_Toc515099344)

[2.4.6 拉起倒计时选择弹窗 24](#_Toc515099345)

[2.4.7 弹出确认提示框 25](#_Toc515099346)

[2.4.8 拉起定时器页面 26](#_Toc515099347)

[2.4.9 调整标题栏 27](#_Toc515099348)

[2.4.10 修改副标题栏文字 27](#_Toc515099349)

[2.4.11 设置标题栏能否点击 28](#_Toc515099350)

[2.4.12 设置标题栏是否可见 28](#_Toc515099351)

[2.4.13 添加覆盖提示 29](#_Toc515099352)

[2.4.14 移除覆盖提示 30](#_Toc515099353)

[2.4.15 设置提示信息是否可见 30](#_Toc515099354)

[2.4.16 获取用户列表 31](#_Toc515099355)

[2.4.17 添加用户信息 31](#_Toc515099356)

[2.4.18 更新用户信息 32](#_Toc515099357)

[2.4.19 获取系统及app信息 33](#_Toc515099358)

[2.4.20 解压LZMA数据 34](#_Toc515099359)

[2.4.21 退出设备页面 34](#_Toc515099360)

[2.4.22 删除当前设备 34](#_Toc515099361)

[2.4.23 获取当前设备信息 35](#_Toc515099362)

[2.4.24 打印debug级别的日志信息 36](#_Toc515099363)

[2.4.25 打印用户级别的日志信息 36](#_Toc515099364)

[2.4.26 修改设备名称 36](#_Toc515099365)

[2.4.27 修改设备摆放位置 37](#_Toc515099366)

[2.4.28 拉起设备信息页面 37](#_Toc515099367)

[2.4.29 拉起设备网络信息页面 38](#_Toc515099368)

[2.4.30 拉起门锁匹配页面 39](#_Toc515099369)

[2.4.31 拉起设备设置页面 39](#_Toc515099370)

[2.4.32 启用重写安卓物理返回键及原生标题（包括IOS和Android）返回键 40](#_Toc515099371)

[2.4.33 获取设备profile 41](#_Toc515099372)

[2.4.34 App从后台恢复时调用函数 42](#_Toc515099373)

[2.4.35 启用重写原生标题（包括IOS和Android）设置按钮点击逻辑 42](#_Toc515099374)

[2.4.36 获取当前手机状态栏高度 43](#_Toc515099375)

[3. JSAPI通道的安全性 44](#_Toc515099376)

[4. 异常错误码描述 44](#_Toc515099377)

# 概述

智能家居APP提供了浏览器jsapi扩展功能，用于实现设备联动、场景配置、路由器交互功能；

智能家居APP提供了在APP内部加载呈现HTML格式web页面的功能，允许页面中通过JavaScript调用APP所特有的jsapi接口，从而实现对智能家居APP native功能的联动控制。

组网图

**功能流程介绍：**

**1）智能家居APP中集成了JSSDK，提供了一套jsapi供页面调用。**

**2）H5页面可使用JavaScript调用jsapi接口，将命令下发给APP。**

**3）此时，APP将收到该jsapi调用，进行解析处理。例如，智能设备的控制命令将通过智能家居云/HiLink网关下发到智能设备。**

HiLink智能家居APP的JSSDK中实现了JavaScript的hilink对象。该SDK的初始化接口定义为：

hilink.config(

"my\_appId", //appId，必填，应用的唯一标识，由华为提供给具体产品合作方

"", //configToken,留空

"" // jsApiList，用于web页面宣示即将调用的API列表，JSON格式。没有在此宣示的API将无权限调用

"0", // 1开启调试模式,调用的所有api的返回值会在客户端alert出来

);

当H5页面需要使用jsapi接口时，需要先调用hilink.config接口，实现hilink接口的初始config。

# JSAPI列表

## 设备接口

### 获取缓存设备状态全集

当需要一次性获取APP本地缓存的设备全部状态时，WEB中调用js方法如下。此接口一般用到设备界面刚刚打开时，快速展示设备界面。此接口仅涉及本地调用，不发出网络请求，将快速返回结果。

**请求消息**

hilink.getDevCacheAll(

"**{devid}**", //devid,string,请取”0”，表示当前设备，暂不支持其它取值

"json\_body", //body,string,可选，见下描述

"resultCallback" //resultCallback，成功或失败时，将调用传入resultStr返回结果

);

【回应】：

当调用成功时，JSSDK将调用success对应的回调函数，将结果返回到resultStr中。

resultStr的结构如下：

**{**

**"devId": "xxxxx",**

**"gatewayId": "xxxxx",**

**"nodeType": "xxxxx",**

**"devInfo": {**

**"sn": "00E0FC018008",**

**"model": "SmartSpeaker",**

**"devType": "004",**

**"manu": "002",**

**"mac": "",**

**"hiv": "1.0",**

**"fwv": "10.01",**

**"hwv": "VER.C",**

**"swv": "V100R001C01B010",**

**"protType": 1,**

**"prodId": "000b"**

**},**

**"services": [{**

**"st": "light",**

**"ts": "20151212T121212Z",**

**"sid": "/light1",**

**"data": {**

**"characteristicName1": "value1",**

**"characteristicName2": "value2"**

**}]**

**}**

【错误码】：

**{**

**"errcode": 11**

**}**

错误码取值参见最下方表格。

如果数据校验失败，返回[HILINK\_VALIDATE\_ERR](#_异常错误码描述_2)；

如果设备不在线，返回[HILINK\_DEV\_OFFLINE](#_异常错误码描述_2)；

如果设备响应超时，返回[HILINK\_DEV\_TIMEOUT](#_异常错误码描述_2)。

### 获取设备状态

一旦JSSDK config成功，就可以通过getDevInfo接口调用该应用所实现的device接口。

该接口为实时通过网络从设备获取，视网络情况可能比较耗时。

当需要获取设备状态时，WEB中调用js方法如下：

hilink.getDevInfo(

"**{devid}**", //devidstring,请取”0”，表示当前设备，暂不支持其它取值

"**{sid}**", // sid,string, 为该设备的service id，取值参见HiLink Profile定义

"json\_body", //body,string,可选，见下描述

"resultCallback" //resultCallback，成功或失败时，将调用传入resultStr返回结果

);

//web页面中需实现该函数，用来获取执行结果

function resultCallback (resultStr) {

var result = resultStr; // REST请求的返回结果

}

sid为该设备的service id，取值参见HiLink Profile定义。

body中可选的列举出需要获取的设备characteristicName。

**{**

**"characteristicName1",**

**"characteristicName2"**

**}**

如果body为空，此接口将返回该**sid**下的对应的所有**characteristicName**的取值。

【回应】：

当调用成功时，JSSDK将调用success对应的回调函数，将结果返回到resultStr中。

resultStr的结构如下：

**{**

**"characteristicName1": "value1",**

**"characteristicName2": "value2"**

**}**

【错误码】：

**{**

**"errcode": 11**

**}**

错误码取值参见最下方表格。

如果数据校验失败，返回[HILINK\_VALIDATE\_ERR](#_异常错误码描述_2)；

如果设备不在线，返回[HILINK\_DEV\_OFFLINE](#_异常错误码描述_2)；

如果设备响应超时，返回[HILINK\_DEV\_TIMEOUT](#_异常错误码描述_2)。

### 获取设备状态（带proid方式）

该接口为带proId的获取设备状态的接口，通常用于支持lua脚本的设备。

当需要获取设备状态时，WEB中调用js方法如下：

hilink.getDevInfoWithProdId(

"**{devid}**", //devidstring,请取”0”，表示当前设备，暂不支持其它取值

"**{sid}**", // sid,string, 为该设备的service id，取值参见HiLink Profile定义

"json\_body", //body,string,可选，见下描述

"resultCallback" //resultCallback，成功或失败时，将调用传入resultStr返回结果

);

//web页面中需实现该函数，用来获取执行结果

function resultCallback (resultStr) {

var result = resultStr; // REST请求的返回结果

}

sid为该设备的service id，取值参见HiLink Profile定义。

body中可选的列举出需要获取的设备characteristicName。

**{**

**"characteristicName1",**

**"characteristicName2"**

**}**

如果body为空，此接口将返回该**sid**下的对应的所有**characteristicName**的取值。

【回应】：

当调用成功时，JSSDK将调用success对应的回调函数，将结果返回到resultStr中。

resultStr的结构如下：

**{**

**"characteristicName1": "value1",**

**"characteristicName2": "value2"**

**}**

【错误码】：

**{**

**"errcode": 11**

**}**

错误码取值参见最下方表格。

如果数据校验失败，返回[HILINK\_VALIDATE\_ERR](#_异常错误码描述_2)；

如果设备不在线，返回[HILINK\_DEV\_OFFLINE](#_异常错误码描述_2)；

如果设备响应超时，返回[HILINK\_DEV\_TIMEOUT](#_异常错误码描述_2)。

### 下发命令到设备

当需要将下发命令给设备时，WEB中调用js方法如下。一般用在接收到用户对设备的操作指令时。

hilink.setDeviceInfo(

"**{devid}**", //devid,string,请取”0”，表示当前设备，暂不支持其它取值

"json\_body", //body,string,可选，见下描述

"resultCallback" //resultCallback，成功或失败时，将调用传入resultStr返回结果

);

sid为该设备的的service id，取值参见HiLink Profile定义。

json\_body中列举出需要设置的设备characteristicName以及要设置的value。

**{**

**"sid1":{**

**"characteristicName1": "value1",**

**"characteristicName2": "value2"**

**},**

**"sid2":{**

**"characteristicName1": "value1",**

**"characteristicName2": "value2"**

**}**

**}**

Body不能为空，否则将失败。

【回应】：

当调用成功时，JSSDK将调用success对应的回调函数，将如下结果返回到resultStr中。

**{**

**"errcode": 0**

**}**

【错误码】：

**{**

**"errcode": 11**

**}**

错误码取值参见最下方表格。

如果数据校验失败，返回[HILINK\_VALIDATE\_ERR](#_异常错误码描述_2)；

如果设备不在线，返回[HILINK\_DEV\_OFFLINE](#_异常错误码描述_2)；

如果设备响应超时，返回[HILINK\_DEV\_TIMEOUT](#_异常错误码描述_2)。

### 下发命令到设备（带proId方式）

当需要使用带proId的方式下发命令给设备时，WEB中调用js方法如下。一般用在支持lua脚本的设备下发命令时。

hilink. setDeviceInfoWithProdId (

"**{devid}**", //devid,string,请取”0”，表示当前设备，暂不支持其它取值

"json\_body", //body,string,可选，见下描述

"resultCallback" //resultCallback，成功或失败时，将调用传入resultStr返回结果

);

sid为该设备的的service id，取值参见HiLink Profile定义。

json\_body中列举出需要设置的设备characteristicName以及要设置的value。

**{**

**"sid1":{**

**"characteristicName1": "value1",**

**"characteristicName2": "value2"**

**},**

**"sid2":{**

**"characteristicName1": "value1",**

**"characteristicName2": "value2"**

**}**

**}**

Body不能为空，否则将失败。

【回应】：

当调用成功时，JSSDK将调用success对应的回调函数，将如下结果返回到resultStr中。

**{**

**"errcode": 0**

**}**

【错误码】：

**{**

**"errcode": 11**

**}**

错误码取值参见最下方表格。

如果数据校验失败，返回[HILINK\_VALIDATE\_ERR](#_异常错误码描述_2)；

如果设备不在线，返回[HILINK\_DEV\_OFFLINE](#_异常错误码描述_2)；

如果设备响应超时，返回[HILINK\_DEV\_TIMEOUT](#_异常错误码描述_2)。

### 获取设备事件通知

//web页面中需实现该函数，用来获取设备产生的事件

function deviceEventCallback(event) {

var result = event; //事件详情

}

当需要处理设备产生的事件时，WEB中需要实现如下方法：

event格式如下：

**{**

**"devId":"{deviceId}",**

**"gatewayId":"xxxxx",**

**"services": [{**

**"st": "air\_conditioner",**

**"ts": "20151212T121212Z",**

**"sid": "1",**

**"data": {**

**"characteristicName1": "value1",**

**"characteristicName2": "value2"**

**}**

**}**

**]**

**}**

### 获取设备历史信息

当需要获取Device历史信息时，WEB中调用js方法如下：

**请求消息**

hilink.getDevHistory(

"**{devid}**", //devid,string,请取”0”，表示当前设备，暂不支持其它取值

" 0", // pageNo,string, 缺省 0

" 10", // pageSize,string, 缺省 10

" 20150501T010203Z", // startTime,string, 格式：20150501T010203Z

" 20150601T010203Z",// endTime,string,

" button1", // sid,string, 为该设备的service id，取值参见HiLink Profile定义

" on", // character,string, 为该设备的characteristicName，取值参见HiLink Profile定义

"resultCallback" //resultCallback，成功或失败时，将调用传入resultStr返回结果

);

【回应】：

当调用成功时，JSSDK将调用success对应的回调函数，将结果返回到resultStr中。

resultStr的结构如下：

**{**

**"totalCount": 2,**

**"pageNo": 0,**

**"pageSize": 10,**

**"list": [{**

**"devId": "xxxxx",**

**"gatewayId": "xxxxx",**

**"sid": "/light1",**

**"data": {**

**"characteristicName1": "value1",**

**"characteristicName2": "value2"**

**},**

**"timestamp":"20151212T121212Z"**

**}**

**]**

**}**

【错误码】：

**{**

**"errcode": 11**

**}**

错误码取值参见最下方表格。

如果数据校验失败，返回[HILINK\_VALIDATE\_ERR](#_异常错误码描述_2)；

如果设备不在线，返回[HILINK\_DEV\_OFFLINE](#_异常错误码描述_2)；

如果设备响应超时，返回[HILINK\_DEV\_TIMEOUT](#_异常错误码描述_2)。

### 上报设备拓展数据

当需要上报设备拓展数据时，WEB中调用js方法如下：

**请求消息**

hilink. postDeviceExtendData (

"json\_body" //需要上报的数据的json字符串

"resultCallback" //resultCallback，成功或失败时，将调用传入resultStr返回结果

);

json\_body格式如下

**{**

**"type": "extend\_data" //上报的扩展数据的type，需要根据需求向云端申请白名单**

**"data":"json\_body" //需要上报的扩展数据内容，根据需求填写，需要符合json格式**

**}**

【回应】：

当调用成功时，JSSDK将调用success对应的回调函数，返回对应这次上报操作的错误码，成功时错误码为0。

【错误码】：

**{**

**"errcode": 0**

**}**

错误码取值参见最下方表格。

如果数据校验失败，返回[HILINK\_VALIDATE\_ERR](#_异常错误码描述_2)；

如果设备不在线，返回[HILINK\_DEV\_OFFLINE](#_异常错误码描述_2)；

如果设备响应超时，返回[HILINK\_DEV\_TIMEOUT](#_异常错误码描述_2)。

### 获取设备拓展数据

当需要获取设备拓展数据时，WEB中调用js方法如下：

**请求消息**

hilink. getDeviceExtendData (

"type" //需要查询的拓展数据的type,与上报时一致

"resultCallback" //resultCallback，成功或失败时，将调用传入resultStr返回结果

);

【回应】：

当调用成功时，JSSDK将调用success对应的回调函数，将结果返回到resultStr中。

resultStr的结构如下：

**{**

**"type": “extend\_data”, //上报的拓展数据的type**

**"data": “json\_body” //上报的拓展数据，符合json格式**

**}**

【错误码】：

失败时，将返回对应的错误码

**{**

**"errcode": 11**

**}**

错误码取值参见最下方表格。

如果数据校验失败，返回[HILINK\_VALIDATE\_ERR](#_异常错误码描述_2)；

如果设备不在线，返回[HILINK\_DEV\_OFFLINE](#_异常错误码描述_2)；

如果设备响应超时，返回[HILINK\_DEV\_TIMEOUT](#_异常错误码描述_2)。

### 删除设备拓展数据

当需要删除设备拓展数据时，WEB中调用js方法如下：

**请求消息**

hilink. deleteDeviceExtendData (

"type" //需要删除的拓展数据的type,与上报时一致

"resultCallback" //resultCallback，成功或失败时，将调用传入resultStr返回结果

);

【回应】：

当调用成功时，JSSDK将调用success对应的回调函数，返回对应这次删除操作的错误码，删除成功时错误码为0。

【错误码】：

**{**

**"errcode": 0**

**}**

错误码取值参见最下方表格。

如果数据校验失败，返回[HILINK\_VALIDATE\_ERR](#_异常错误码描述_2)；

如果设备不在线，返回[HILINK\_DEV\_OFFLINE](#_异常错误码描述_2)；

如果设备响应超时，返回[HILINK\_DEV\_TIMEOUT](#_异常错误码描述_2)。

## 路由器接口

当从路由器工具箱入口加载H5页面时，将启用如下的路由器接口集合（其它情况下，此接口集合是无权限调用的状态）。此时，H5页面绑定了该路由器相关的属性，可调用如下接口，和拉起该界面的路由器进行深层次的交互。

注：当通过路由器具体一个插件的入口加载H5页面时，仅允许调用该插件的提供的接口。

### 路由器通用交互接口-获取资源

当H5页面需要和路由器交互时，通过getResource接口调用获取路由器上的资源。本接口实现了路由器resource的GET方法，调用方法如下：

hilink.getResource(

"/resources/xx/status", //uri，该REST resource的uri,参见路由器接口描述

"resultCallback" //resultCallback，成功或失败时，将调用传入resultStr返回结果

);

resultCallback使用方法如下：

//web页面中需实现该函数，用来获取执行结果

function resultCallback (resultStr) {

var result = resultStr; // REST请求的返回结果

}

【回应】：

当调用成功时，若有数据回，将得到如下格式的数据，具体结构定义参见路由器接口文档。

**{**

**"characteristicName1": "value1",**

**"characteristicName2": "value2"**

**}**

当无需数据返回时，JSSDK将调用success对应的回调函数，将如下结果返回到resultStr中。

**{**

**"errcode": 0**

**}**

【错误码】：

**{**

**"errcode": 11**

**}**

【错误码】：

**{**

**"errcode": 11**

**}**

错误码取值参见最下方表格。

如果数据校验失败，返回[HILINK\_VALIDATE\_ERR](#_异常错误码描述_2)；

如果设备不在线，返回[HILINK\_DEV\_OFFLINE](#_异常错误码描述_2)；

如果设备响应超时，返回[HILINK\_DEV\_TIMEOUT](#_异常错误码描述_2)。

### 路由器通用交互接口-下发配置

当H5页面需要和路由器交互时，通过setResource接口调用实现了下发配置到路由器。对应路由器restful接口的POST方法，调用方法如下：

hilink.setResource(

"/resources/xx/para", //uri，该REST resource的uri,参见路由器接口描述

"", //body可选，POST的数据，具体格式请参考路由器接口

"resultCallback" //resultCallback，成功或失败时，将调用传入resultStr返回结果

);

resultCallback使用方法如下：

//web页面中需实现该函数，用来获取执行结果

function resultCallback (resultStr) {

var result = resultStr; // REST请求的返回结果

}

【回应】：

当调用成功时，若有数据回，将得到如下格式的数据，具体结构定义参见路由器接口文档。

**{**

**"characteristicName1": "value1",**

**"characteristicName2": "value2"**

**}**

当无需数据返回时，JSSDK将调用success对应的回调函数，将如下结果返回到resultStr中。

**{**

**"errcode": 0**

**}**

【错误码】：

**{**

**"errcode": 11**

**}**

【错误码】：

**{**

**"errcode": 11**

**}**

错误码取值参见最下方表格。

如果数据校验失败，返回[HILINK\_VALIDATE\_ERR](#_异常错误码描述_2)；

如果设备不在线，返回[HILINK\_DEV\_OFFLINE](#_异常错误码描述_2)；

如果设备响应超时，返回[HILINK\_DEV\_TIMEOUT](#_异常错误码描述_2)。

## 场景接口~~（暂不需要）~~

### 获取场景列表

当需要获取场景列表时，WEB中调用js方法如下：

hilink.getScenes(

"", //homeId, string,留空,默认当前设备所在房间

"json\_body", //body,string,留空

"resultCallback" //resultCallback，成功或失败时，将调用传入resultStr返回结果

);

//web页面中需实现该函数，用来获取执行结果

function resultCallback (resultStr) {

var result = resultStr; // REST请求的返回结果

}

【回应】：

当调用成功时，JSSDK将调用success对应的回调函数，将结果返回到resultStr中。

resultStr的结构如下：

**[{**

**"id": 101,**

**"name":"scene1",**

**"type": 0,**

**"description":"回家模式",**

**"actions": [{**

**"devId":"2323-1221-...",**

**"sid": "/switch1",**

**"data": {**

**"on": true**

**}**

**}]**

**}, {**

**"id": 102,**

**"name":"scene2",**

**"type": 0,**

**"description":"离家模式",**

**"actions": [{**

**"devId":"2323-1221-...",**

**"sid": "/switch1",**

**"data": {**

**"on": false**

**}**

**}]**

**}]**

【错误码】：

**{**

**"errcode": 11**

**}**

错误码取值参见最下方表格。

如果数据校验失败，返回[HILINK\_VALIDATE\_ERR](#_异常错误码描述_2)；

如果设备不在线，返回[HILINK\_DEV\_OFFLINE](#_异常错误码描述_2)；

如果设备响应超时，返回[HILINK\_DEV\_TIMEOUT](#_异常错误码描述_2)。

### 创建手动场景

当需要创建手动场景时，WEB中调用js方法如下：

hilink.addScene(

"", //homeId, string,留空,默认当前设备所在房间

"json\_body", //body,string, 场景json字符串

"resultCallback" //resultCallback，成功或失败时，将调用传入resultStr返回结果

);

//web页面中需实现该函数，用来获取执行结果

function resultCallback (resultStr) {

var result = resultStr; // REST请求的返回结果

}

json\_body中携带需要创建的场景，格式如下：

**{**

**"name": "Scene01",**

**"description": "Scene01 for home: default",**

**"type": 1,**

**"actions": [**

**{**

**"devId":"2323-1221-...",**

**"sid": "/switch1",**

**"messageType": "POST",**

**"data": {**

**“on”: true**

**}**

**},**

**{**

**"devId":"2323-1221-...",**

**"sid": "/switch1",**

**"messageType": "POST",**

**"data": {**

**“on”: false**

**}**

**}**

**]**

**}**

Body不能为空，否则将失败。

【回应】：

当调用成功时，JSSDK将调用success对应的回调函数，将如下结果返回到resultStr中。

**{**

**"id": 1068,**

**"name": "Scene02",**

**"type": 1,**

**"status": "active",**

**"description": "Scene01 for home: 1008",**

**"actions": [**

**{**

**"devId":"2323-1221-...",**

**"sid": "/switch1",**

**"messageType": "POST",**

**"data": {**

**“on”: true**

**}**

**},**

**{**

**"devId":"2323-1221-...",**

**"sid": "/switch1",**

**"messageType": "POST",**

**"data": {**

**“on”: false**

**}**

**}**

**]**

**}**

【错误码】：

当出错时，返回错误码：

**{**

**"errcode": 11**

**}**

错误码取值参见最下方表格。

如果数据校验失败，返回[HILINK\_VALIDATE\_ERR](#_异常错误码描述_2)；

如果设备不在线，返回[HILINK\_DEV\_OFFLINE](#_异常错误码描述_2)；

如果设备响应超时，返回[HILINK\_DEV\_TIMEOUT](#_异常错误码描述_2)。

### 修改手动场景

当需要修改手动场景时，WEB中调用js方法如下：

hilink.modifyScene(

"", //homeId, string,留空,默认当前设备所在房间

"1", //scenceId, string,需要修改的场景ID

"json\_body", //body,string, 场景json字符串

"resultCallback" //resultCallback，成功或失败时，将调用传入resultStr返回结果

);

//web页面中需实现该函数，用来获取执行结果

function resultCallback (resultStr) {

var result = resultStr; // REST请求的返回结果

}

json\_body中携带需要创建的场景，格式如下：

**{**

**"name": "Scene01",**

**"description": "Scene01 for home: default",**

**"type": 1,**

**"actions": [**

**{**

**"devId":"2323-1221-...",**

**"sid": "/switch1",**

**"messageType": "POST",**

**"data": {**

**“on”: true**

**}**

**},**

**{**

**"devId":"2323-1221-...",**

**"sid": "/switch1",**

**"messageType": "POST",**

**"data": {**

**“on”: false**

**}**

**}**

**]**

**}**

Body不能为空，否则将失败。

【回应】：

当调用成功时，JSSDK将调用success对应的回调函数，将如下结果返回到resultStr中。

**{**

**"errcode": 0**

**}**

【错误码】：

**{**

**"errcode": 11**

**}**

错误码取值参见最下方表格。

如果数据校验失败，返回[HILINK\_VALIDATE\_ERR](#_异常错误码描述_2)；

如果设备不在线，返回[HILINK\_DEV\_OFFLINE](#_异常错误码描述_2)；

如果设备响应超时，返回[HILINK\_DEV\_TIMEOUT](#_异常错误码描述_2)。

### 删除手动场景

当需要删除手动场景时，WEB中调用js方法如下：

hilink.delScene(

"", //homeId, string,留空,默认当前设备所在房间

"1", //scenceId, string,指定的场景ID

"json\_body", //body,string, 留空

"resultCallback" //resultCallback，成功或失败时，将调用传入resultStr返回结果

);

//web页面中需实现该函数，用来获取执行结果

function resultCallback (resultStr) {

var result = resultStr; // REST请求的返回结果

}

【回应】：

当调用成功时，JSSDK将调用success对应的回调函数，将如下结果返回到resultStr中。

**{**

**"errcode": 0**

**}**

【错误码】：

**{**

**"errcode": 11**

**}**

错误码取值参见最下方表格。

如果数据校验失败，返回[HILINK\_VALIDATE\_ERR](#_异常错误码描述_2)；

如果设备不在线，返回[HILINK\_DEV\_OFFLINE](#_异常错误码描述_2)；

如果设备响应超时，返回[HILINK\_DEV\_TIMEOUT](#_异常错误码描述_2)。

### 创建自动规则

当需要创建自动规则时，WEB中调用js方法如下：

hilink.addRule(

"json\_body", //body,string, 规则json字符串

"resultCallback" //resultCallback，成功或失败时，将调用传入resultStr返回结果

);

//web页面中需实现该函数，用来获取执行结果

function resultCallback (resultStr) {

var result = resultStr; // REST请求的返回结果

}

json\_body中携带需要创建的规则，格式如下：

**{**

**"name": "Rule01",**

**"conditions":[**

**{**

**"type":"DEVICE\_DATA",**

**"deviceInfo":{**

**"devId":"2323-1221-...",**

**"devType":"005",**

**"path":"switch/on"**

**},**

**“operator” :"=”,**

**“value”:”1”**

**}**

**]**

**“author”:”12222”,**

**"actions": [**

**{**

**"type":"DEVICE\_CMD",**

**"devId":"2323-1221-...",**

**"devType":"01B",**

**“cmd”:{**

**"sid": "switch",**

**"messageType": "POST",**

**" messageBody": {**

**“on”: 1**

**}**

**}**

**}**

**]**

**}**

Body不能为空，否则将失败。

【回应】：

当调用成功时，JSSDK将调用success对应的回调函数，将如下结果返回到resultStr中。

**{**

**”ruleId” : “\*\*\*\*\*\*\*”**

**"name": "Rule01",**

**"conditions":[**

**{**

**“id”:”\*\*\*\*”,**

**" type":" DEVICE\_DATA ",**

**"deviceInfo":{**

**"devId":"2323-1221-...",**

**"devType":"005",**

**"path":"switch/on"**

**},**

**“operator” :"=”,**

**“value”:”1”**

**}**

**]**

**“author”:”12222”,**

**"actions": [**

**{**

**"type":"DEVICE\_CMD",**

**“id”:”\*\*\*\*”,**

**"devId":"2323-1221-...",**

**"devType":"01B",**

**“cmd”:{**

**"sid": "switch",**

**"messageType": "POST",**

**" messageBody": {**

**“on”: 1**

**}**

**}**

**}**

**]**

**}**

【错误码】：

当出错时，返回错误码：

**{**

**"errcode": 11**

**}**

错误码取值参见最下方表格。

如果数据校验失败，返回[HILINK\_VALIDATE\_ERR](#_异常错误码描述_2)；

如果设备不在线，返回[HILINK\_DEV\_OFFLINE](#_异常错误码描述_2)；

如果设备响应超时，返回[HILINK\_DEV\_TIMEOUT](#_异常错误码描述_2)。

### 修改自动规则

当需要修改自动规则时，WEB中调用js方法如下：

hilink.modifyRule(

"json\_body", //body,string, 规则json字符串

"resultCallback" //resultCallback，成功或失败时，将调用传入resultStr返回结果

);

//web页面中需实现该函数，用来获取执行结果

function resultCallback (resultStr) {

var result = resultStr; // REST请求的返回结果

}

json\_body中携带需要创建的规则，格式如下：

**{**

**"name": "Rule01",**

**"conditions":[**

**{**

**"type":"DEVICE\_DATA",**

**"deviceInfo":{**

**"devId":"2323-1221-...",**

**"devType":"005",**

**"path":"switch/on"**

**},**

**“operator” :"=”,**

**“value”:”1”**

**}**

**]**

**“author”:”12222”,**

**"actions": [**

**{**

**"type":"DEVICE\_CMD",**

**"devId":"2323-1221-...",**

**"devType":"01B",**

**“cmd”:{**

**"sid": "switch",**

**"messageType": "POST",**

**" messageBody": {**

**“on”: 1**

**}**

**}**

**}**

**]**

**}**

Body不能为空，否则将失败。

【回应】：

当调用成功时，JSSDK将调用success对应的回调函数，将如下结果返回到resultStr中。

**{**

**”ruleId” : “\*\*\*\*\*\*\*”**

**"name": "Rule01",**

**"conditions":[**

**{**

**“id”:”\*\*\*\*”,**

**" type":" DEVICE\_DATA ",**

**"deviceInfo":{**

**"devId":"2323-1221-...",**

**"devType":"005",**

**"path":"switch/on"**

**},**

**“operator” :"=”,**

**“value”:”1”**

**}**

**]**

**“author”:”12222”,**

**"actions": [**

**{**

**"type":"DEVICE\_CMD",**

**“id”:”\*\*\*\*”,**

**"devId":"2323-1221-...",**

**"devType":"01B",**

**“cmd”:{**

**"sid": "switch",**

**"messageType": "POST",**

**" messageBody": {**

**“on”: 1**

**}**

**}**

**}**

**]**

**}**

【错误码】：

当出错时，返回错误码：

**{**

**"errcode": 11**

**}**

错误码取值参见最下方表格。

如果数据校验失败，返回[HILINK\_VALIDATE\_ERR](#_异常错误码描述_2)；

如果设备不在线，返回[HILINK\_DEV\_OFFLINE](#_异常错误码描述_2)；

如果设备响应超时，返回[HILINK\_DEV\_TIMEOUT](#_异常错误码描述_2)。

### 删除自动规则

当需要删除自动规则时，WEB中调用js方法如下：

hilink.delScene(

"1", //ruleId, string,指定的规则ID

"json\_body", //body,string, 留空

"resultCallback" //resultCallback，成功或失败时，将调用传入resultStr返回结果

);

//web页面中需实现该函数，用来获取执行结果

function resultCallback (resultStr) {

var result = resultStr; // REST请求的返回结果

}

【回应】：

当调用成功时，JSSDK将调用success对应的回调函数，将如下结果返回到resultStr中。

**{**

**"errcode": 0**

**}**

【错误码】：

**{**

**"errcode": 11**

**}**

错误码取值参见最下方表格。

如果数据校验失败，返回[HILINK\_VALIDATE\_ERR](#_异常错误码描述_2)；

如果设备不在线，返回[HILINK\_DEV\_OFFLINE](#_异常错误码描述_2)；

如果设备响应超时，返回[HILINK\_DEV\_TIMEOUT](#_异常错误码描述_2)。

## 扩展接口

### 拉起时间选择弹窗

当需要在页面中跳转时间选择弹框时，WEB中调用js方法如下，提示框将叠加展示在当前界面：

hilink.timerDiag(

"com.huawei.smarthome.timerDiag", //URI,string, 弹出时间选择窗口

"title", //param,string, 提示框标题

"json\_body", //body,string,时间选择框属性，见下描述

"resultCallback" //resultCallback，成功或失败时，将调用传入resultStr返回结果

);

效果图：



json\_body中列举出需要设置的时间选择框的属性。

**{**

**"maxHour": 24 //可选择的最大小时**

**"minHour": 0 //可选择的最小小时**

**“maxMinute”: 59 //可选择的最大分钟数**

**“minMinute”: 0 //可选择的最小分钟数**

**}**

Body不能为空，否则将失败。

【回应】：

当调用成功时，JSSDK将调用success对应的回调函数，将如下结果返回到resultStr中。

**{**

**"errcode": 0,**

**"hour": "12",**

**"miniute": "11"**

**}**

【错误码】：

**{**

**"errcode": 11**

**}**

错误码取值参见最下方表格。

如果数据校验失败，返回[HILINK\_VALIDATE\_ERR](#_异常错误码描述_2)；

如果设备不在线，返回[HILINK\_DEV\_OFFLINE](#_异常错误码描述_2)；

如果设备响应超时，返回[HILINK\_DEV\_TIMEOUT](#_异常错误码描述_2)。

### 拉起倒计时选择弹窗

当需要在页面中拉起倒计时选择弹框时，WEB中调用js方法如下，提示框将叠加展示在当前界面：

hilink.timerDiag(

"com.huawei.smarthome. delayDig", //URI,string, 弹出倒计时选择窗口

"resultCallback" //resultCallback，成功或失败时，将调用传入resultStr返回结果

);

【回应】：

当调用成功时，JSSDK将调用success对应的回调函数，将如下结果返回到resultStr中。

**{**

**"errcode": 0,**

**}**

【错误码】：

**{**

**"errcode": 11**

**}**

错误码取值参见最下方表格。

如果数据校验失败，返回[HILINK\_VALIDATE\_ERR](#_异常错误码描述_2)；

如果设备不在线，返回[HILINK\_DEV\_OFFLINE](#_异常错误码描述_2)；

如果设备响应超时，返回[HILINK\_DEV\_TIMEOUT](#_异常错误码描述_2)。

### 弹出确认提示框

当需要在页面中弹出确认提示框时，WEB中调用js方法如下，提示框将叠加展示在当前界面：

hilink.confirmDiag(

"com.huawei.smarthome.confirmDiag", //URI,string, 提示框URI

"title", //param,string, 提示框标题

"content text", //param,string, 提示框文字

"resultCallback" //resultCallback，成功或失败时，将调用传入resultStr返回结果

);

【回应】：

当调用成功时，JSSDK将调用success对应的回调函数，将如下结果返回到resultStr中。

**{**

**"errcode": 0,**

**"confirmed": false**

**}**

【错误码】：

**{**

**"errcode": 11**

**}**

错误码取值参见最下方表格。

如果数据校验失败，返回[HILINK\_VALIDATE\_ERR](#_异常错误码描述_2)；

如果设备不在线，返回[HILINK\_DEV\_OFFLINE](#_异常错误码描述_2)；

如果设备响应超时，返回[HILINK\_DEV\_TIMEOUT](#_异常错误码描述_2)。

### 拉起定时器页面

当需要在页面中跳转定时器界面时，WEB中调用js方法如下，将叠加展示定时器页面，不会关闭当前界面：

hilink.jumpTo(

"com.huawei.smarthome.timerPage", //URI,string, 跳转到定时器页面

"resultCallback" //resultCallback，成功或失败时，将调用传入resultStr返回结果

);

拉起的定时器页面效果图：



【回应】：

当调用成功时，JSSDK将调用success对应的回调函数，将如下结果返回到resultStr中。

**{**

**"errcode": 0,**

**}**

【错误码】：

**{**

**"errcode": 11**

**}**

错误码取值参见最下方表格。

如果数据校验失败，返回[HILINK\_VALIDATE\_ERR](#_异常错误码描述_2)；

如果设备不在线，返回[HILINK\_DEV\_OFFLINE](#_异常错误码描述_2)；

如果设备响应超时，返回[HILINK\_DEV\_TIMEOUT](#_异常错误码描述_2)。

### 调整标题栏

当需要在页面中设置标题栏背景及标题栏字体颜色时，WEB中调用js方法如下：

hilink.modifyTitleBar(

true, //URI,boolean, true，标题栏及状态栏文字白色，false标题栏及状态栏文字黑色

"FFFFEE00", //string, ARGB背景色值，目前已无作用

"resultCallback" //resultCallback，成功或失败时，将调用传入resultStr返回结果

);

【回应】：

当调用成功时，JSSDK将调用success对应的回调函数，将如下结果返回到resultStr中。

**{**

**"errcode": 0,**

**}**

【错误码】：

**{**

**"errcode": 11**

**}**

错误码取值参见最下方表格。

如果数据校验失败，返回[HILINK\_VALIDATE\_ERR](#_异常错误码描述_2)；

如果设备不在线，返回[HILINK\_DEV\_OFFLINE](#_异常错误码描述_2)；

如果设备响应超时，返回[HILINK\_DEV\_TIMEOUT](#_异常错误码描述_2)。

### 修改副标题栏文字

当需要在页面中设置副标题栏文字时，WEB中调用js方法如下：

hilink. modifyTitleStatus (

"status", //string, 需要显示在副标题栏的文字

"resultCallback" //resultCallback，成功或失败时，将调用传入resultStr返回结果

);

【回应】：

当调用成功时，JSSDK将调用success对应的回调函数，将如下结果返回到resultStr中。

**{**

**"errcode": 0,**

**}**

【错误码】：

**{**

**"errcode": 11**

**}**

错误码取值参见最下方表格。

如果数据校验失败，返回[HILINK\_VALIDATE\_ERR](#_异常错误码描述_2)；

如果设备不在线，返回[HILINK\_DEV\_OFFLINE](#_异常错误码描述_2)；

如果设备响应超时，返回[HILINK\_DEV\_TIMEOUT](#_异常错误码描述_2)。

### 设置标题栏能否点击

当需要在页面中设置标题栏能否点击时，WEB中调用js方法如下：

hilink. setTitleEnable(

true, //URI,boolean, true，标题可以点击，false标题无法点击

"resultCallback" //resultCallback，成功或失败时，将调用传入resultStr返回结果

);

【回应】：

当调用成功时，JSSDK将调用success对应的回调函数，将如下结果返回到resultStr中。

**{**

**"errcode": 0,**

**}**

【错误码】：

**{**

**"errcode": 11**

**}**

错误码取值参见最下方表格。

如果数据校验失败，返回[HILINK\_VALIDATE\_ERR](#_异常错误码描述_2)；

如果设备不在线，返回[HILINK\_DEV\_OFFLINE](#_异常错误码描述_2)；

如果设备响应超时，返回[HILINK\_DEV\_TIMEOUT](#_异常错误码描述_2)。

### 设置标题栏是否可见

当需要在页面中设置标题栏是否可见时，WEB中调用js方法如下：

hilink. setTitleVisible(

true, //URI,boolean, true，标题可见，false标题不可见

"resultCallback" //resultCallback，成功或失败时，将调用传入resultStr返回结果

);

【回应】：

当调用成功时，JSSDK将调用success对应的回调函数，将如下结果返回到resultStr中。

**{**

**"errcode": 0,**

**}**

【错误码】：

**{**

**"errcode": 11**

**}**

错误码取值参见最下方表格。

如果数据校验失败，返回[HILINK\_VALIDATE\_ERR](#_异常错误码描述_2)；

如果设备不在线，返回[HILINK\_DEV\_OFFLINE](#_异常错误码描述_2)；

如果设备响应超时，返回[HILINK\_DEV\_TIMEOUT](#_异常错误码描述_2)。

### 添加覆盖提示

当需要在页面中添加覆盖提示时，WEB中调用js方法如下：

hilink. addMessage(

"json\_body", //body,string，见下描述

"resultCallback" //resultCallback，成功或失败时，将调用传入resultStr返回结果

);

json\_body中列举出需要添加的覆盖提示的的属性。

**{**

**"type": 24 //自定义int值**

**"key": 1 //默认传1**

**“icon”: 1 //覆盖消息前图标的编号，可以为Null，可传值为数组对应角标，参考下图**

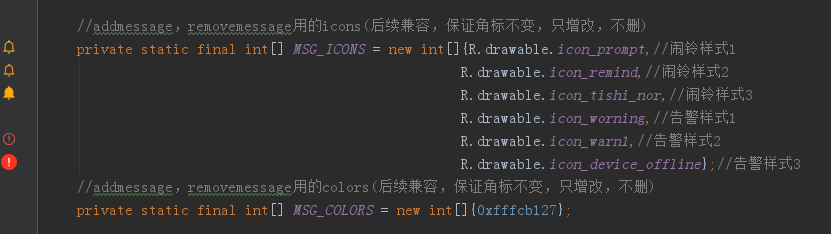
**“color”: 0 //字体颜色编号，可以为null，可传值为数组对应角标，可传值见下图**

**“msg”: “ssssss” //需要显示的信息**

**}**

Body不能为空，否则将失败。

Key传2时对应空气净化器更换滤芯的提示，对应跳转更换滤芯界面。



【回应】：

当调用成功时，JSSDK将调用success对应的回调函数，将如下结果返回到resultStr中。

**{**

**"errcode": 0,**

**}**

【错误码】：

**{**

**"errcode": 11**

**}**

错误码取值参见最下方表格。

如果数据校验失败，返回[HILINK\_VALIDATE\_ERR](#_异常错误码描述_2)；

如果设备不在线，返回[HILINK\_DEV\_OFFLINE](#_异常错误码描述_2)；

如果设备响应超时，返回[HILINK\_DEV\_TIMEOUT](#_异常错误码描述_2)。

### 移除覆盖提示

当需要在页面中移除覆盖提示时，WEB中调用js方法如下：

hilink. removeMessage(

"json\_body", //body,string，见下描述

"resultCallback" //resultCallback，成功或失败时，将调用传入resultStr返回结果

);

json\_body中需要的属性：

**{**

**"type": 24 // 添加覆盖提示时传的自定义int值，移除对应的覆盖提示**

**}**

Body不能为空，否则将失败。

【回应】：

当调用成功时，JSSDK将调用success对应的回调函数，将如下结果返回到resultStr中。

**{**

**"errcode": 0,**

**}**

【错误码】：

**{**

**"errcode": 11**

**}**

错误码取值参见最下方表格。

如果数据校验失败，返回[HILINK\_VALIDATE\_ERR](#_异常错误码描述_2)；

如果设备不在线，返回[HILINK\_DEV\_OFFLINE](#_异常错误码描述_2)；

如果设备响应超时，返回[HILINK\_DEV\_TIMEOUT](#_异常错误码描述_2)。

### 设置提示信息是否可见

当需要在页面中设置提示信息是否可见时，WEB中调用js方法如下：

hilink. setMessageVisible(

true, //URI,boolean, true，提示信息可见，false提示信息不可见

"resultCallback" //resultCallback，成功或失败时，将调用传入resultStr返回结果

);

【回应】：

当调用成功时，JSSDK将调用success对应的回调函数，将如下结果返回到resultStr中。

**{**

**"errcode": 0,**

**}**

【错误码】：

**{**

**"errcode": 11**

**}**

错误码取值参见最下方表格。

如果数据校验失败，返回[HILINK\_VALIDATE\_ERR](#_异常错误码描述_2)；

如果设备不在线，返回[HILINK\_DEV\_OFFLINE](#_异常错误码描述_2)；

如果设备响应超时，返回[HILINK\_DEV\_TIMEOUT](#_异常错误码描述_2)。

### 获取用户列表

用于获取对应设备的用户列表，主要用于智能门锁，WEB中调用js方法如下：

hilink. getUserList(

"**{devid}**", //devid,string，需要查询用户列表的设备的devid

"resultCallback" //resultCallback，成功或失败时，将调用传入resultStr返回结果

);

【回应】：

当调用成功时，JSSDK将调用success对应的回调函数，将如下结果返回到resultStr中。

**{**

**[**

**{**

**“userId”: “11111”, //用户id，String**

**“name”: “aaaaa”, //昵称，String**

**“type”: “pass”, //种类，String**

**“label”: “ssss”, //标签，String**

**“deviceid”: “111111-sss-sasas”, //设备id，String**

**},**

**{**

**“userId”: “222222”, //用户id，String**

**“name”: “bbbbbb”, //昵称，String**

**“type”: “key”, //种类，String**

**“label”: “dddd”, //标签，String**

**“deviceid”: “111111-sss-sasas”, //设备id，String**

**},**

**……**

**]**

**}**

【错误码】：

**{**

**"errcode": 11**

**}**

错误码取值参见最下方表格。

如果数据校验失败，返回[HILINK\_VALIDATE\_ERR](#_异常错误码描述_2)；

如果设备不在线，返回[HILINK\_DEV\_OFFLINE](#_异常错误码描述_2)；

如果设备响应超时，返回[HILINK\_DEV\_TIMEOUT](#_异常错误码描述_2)。

### 添加用户信息

用于添加对应设备的用户信息，主要用于智能门锁，WEB中调用js方法如下：

hilink. setUserList(

"json\_body", //body,string，见下描述

"resultCallback" //resultCallback，成功或失败时，将调用传入resultStr返回结果

);

json\_body中列举需要添加的用户信息的属性：

**{**

**[**

**{**

**“userId”: “11111”, //用户id，String**

**“name”: “aaaaa”, //昵称，String**

**“type”: “pass”, //种类，String**

**“label”: “ssss”, //标签，String**

**“deviceid”: “111111-sss-sasas”, //设备id，String**

**},**

**{**

**“userId”: “222222”, //用户id，String**

**“name”: “bbbbbb”, //昵称，String**

**“type”: “key”, //种类，String**

**“label”: “dddd”, //标签，String**

**“deviceid”: “111111-sss-sasas”, //设备id，String**

**},**

**……**

**]**

**}**

Body不能为空，否则将失败。

【回应】：

当调用成功时，JSSDK将调用success对应的回调函数，将如下结果返回到resultStr中。

**{**

**ok**

**}**

【错误码】：

**{**

**"errcode": 11**

**}**

错误码取值参见最下方表格。

如果数据校验失败，返回[HILINK\_VALIDATE\_ERR](#_异常错误码描述_2)；

如果设备不在线，返回[HILINK\_DEV\_OFFLINE](#_异常错误码描述_2)；

如果设备响应超时，返回[HILINK\_DEV\_TIMEOUT](#_异常错误码描述_2)。

### 更新用户信息

用于更新对应设备的用户信息，主要用于智能门锁，WEB中调用js方法如下：

hilink. updateUserList(

"json\_body", //body,string，见下描述

"resultCallback" //resultCallback，成功或失败时，将调用传入resultStr返回结果

);

json\_body中列举需要添加的用户信息的属性：

**{**

**“userId”: “222222”, //用户id，String**

**“name”: “bbbbbb”, //昵称，String**

**“type”: “key”, //种类，String**

**“label”: “dddd”, //标签，String**

**“deviceid”: “111111-sss-sasas”, //设备id，String**

**}**

Body不能为空，否则将失败。

【回应】：

当调用成功时，JSSDK将调用success对应的回调函数，将如下结果返回到resultStr中。

**{**

**ok**

**}**

【错误码】：

**{**

**"errcode": 11**

**}**

错误码取值参见最下方表格。

如果数据校验失败，返回[HILINK\_VALIDATE\_ERR](#_异常错误码描述_2)；

如果设备不在线，返回[HILINK\_DEV\_OFFLINE](#_异常错误码描述_2)；

如果设备响应超时，返回[HILINK\_DEV\_TIMEOUT](#_异常错误码描述_2)。

### 获取系统及app信息

用于获取打开网页的系统及app信息，，WEB中调用js方法如下：

hilink. getVersion(

"resultCallback" //resultCallback，成功或失败时，将调用传入resultStr返回结果

);

【回应】：

当调用成功时，JSSDK将调用success对应的回调函数，将如下结果返回到resultStr中。

**{**

**{**

**“os”: “Android”, //打开网页的系统，Android或IOS，String**

**“osVersion”: 21, //系统版本，int**

**“apiVersion”: 21, //api版本，int**

**“version”: “1900011000”, //app版本号，int**

**},**

**}**

注：版本号格式：如版本为9.0.1.100 ,

则版本号为1 9 00 01 100 0，1900011000

【错误码】：

**{**

**"errcode": 11**

**}**

错误码取值参见最下方表格。

如果数据校验失败，返回[HILINK\_VALIDATE\_ERR](#_异常错误码描述_2)；

如果设备不在线，返回[HILINK\_DEV\_OFFLINE](#_异常错误码描述_2)；

如果设备响应超时，返回[HILINK\_DEV\_TIMEOUT](#_异常错误码描述_2)。

### 解压LZMA数据

用于解压LZMA数据，目前主要用于解压科沃斯扫地机器人地图数据，WEB中调用js方法如下：

hilink. unCompressLzma(

" base64Str", //base64编码后的LZMA压缩数据,string

" type", //返回的数据类型,int，后续可扩展：0：byte转Base64编码字符串，1：科沃斯扫地机器人地图栅格坐标，2：直接byte转字符串

"resultCallback" //resultCallback，成功或失败时，将调用传入resultStr返回结果

);

【回应】：

当调用成功时，JSSDK将调用success对应的回调函数，根据type将对应的结果JSON字符串返回到resultStr中。

**{**

**“resultStr”**

**}**

【错误码】：

**{**

**"errcode": 11**

**}**

错误码取值参见最下方表格。

如果数据校验失败，返回[HILINK\_VALIDATE\_ERR](#_异常错误码描述_2)；

如果设备不在线，返回[HILINK\_DEV\_OFFLINE](#_异常错误码描述_2)；

如果设备响应超时，返回[HILINK\_DEV\_TIMEOUT](#_异常错误码描述_2)。

### 退出设备页面

当需要退出当前设备页面时，WEB中调用js方法如下：

hilink. finishDeviceActivity ();

【回应】：

该函数无需返回。

### 删除当前设备

当需要删除当前设备时，WEB中调用js方法如下：

hilink. deleteDevice (

true, // isAssociate,boolean, 是否删除设备下挂关联设备，目前只有桥设备用到

"resultCallback" //resultCallback，成功或失败时，将调用传入resultStr返回结果

);

【回应】：

当调用成功时，JSSDK将调用success对应的回调函数，将如下结果返回到resultStr中。

**{**

**"errcode": 0,**

**}**

【错误码】：

**{**

**"errcode": 11**

**}**

错误码取值参见最下方表格。

如果数据校验失败，返回[HILINK\_VALIDATE\_ERR](#_异常错误码描述_2)；

如果设备不在线，返回[HILINK\_DEV\_OFFLINE](#_异常错误码描述_2)；

如果设备响应超时，返回[HILINK\_DEV\_TIMEOUT](#_异常错误码描述_2)。

如果无网络导致删除失败，返回HILINK\_NO\_NETWORK

### 获取当前设备信息

当需要获取当前设备信息时，WEB中调用js方法如下：

hilink. getDeviceInfo(

"resultCallback" //resultCallback，成功或失败时，将调用传入resultStr返回结果

);

【回应】：

当调用成功时，JSSDK将调用success对应的回调函数，将如下结果返回到resultStr中。

**{**

**“deviceId”:”479\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*443”,**

**“devName”:”摇控器(8)”,**

**“roomName”:”null”,**

**“status”:”online”,**

**“gatewayId”:”a67\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*ec2”,**

**“nodeType”:”ENDPOINT”,**

**“devInfo”:{**

**“sn”:'@LH\*\*\*\*\*\*\*\*\*\*\*\*\*\*168',**

**“manu”:'021',**

**“devType”:'038',**

**“model”:'B910ZB',**

**“mac”:'00:\*\*\*\*\*\*\*\*\*\*\*:68',**

**“hiv”:'1.0.5',**

**“fwv”:'1.0.1',**

**“hwv”:'v01.3',**

**“swv”:'b00.2',**

**“id”:'1015',**

**“prototype”:'1'},**

**“services”:**

**[{**

**“sid”:'keyEvent',**

**“st”:'keyEvent',**

**“ts”:'20171225T074217Z',**

**“data”:{"key":0}**

**},**

**….]**

**}**

【错误码】：

**{**

**"errcode": 11**

**}**

错误码取值参见最下方表格。

如果数据校验失败，返回[HILINK\_VALIDATE\_ERR](#_异常错误码描述_2)；

如果设备不在线，返回[HILINK\_DEV\_OFFLINE](#_异常错误码描述_2)；

如果设备响应超时，返回[HILINK\_DEV\_TIMEOUT](#_异常错误码描述_2)。

### 打印debug级别的日志信息

当需要在Android版本生成的日志中打印debug级别的日志信息时，debug版本和release版本中都可查看到打印的信息，用来帮助开发定位， WEB中调用js方法如下：

hilink. printLogDebug (

true , // isSaveLog,boolean, true，保存日志信息，false不保存日志信息

"tag" , // tag,String,日志信息的关键字

"msg" , // msg,String,需要打印的内容或信息

);

【回应】：

该函数无需返回。

### 打印用户级别的日志信息

当需要在Android版本生成的日志中打印用户级别的日志信息时，比如涉及到用户数据（mac地址、用户隐私数据）等相关安全级别的信息打印，在debug版本的app中可以看到，在release版本中会经过加密处理，避免安全信息的泄露。WEB中调用js方法如下：

hilink. printLog (

true , // isSaveLog,boolean, true，保存日志信息，false不保存日志信息

"tag" , // tag,String,日志信息的关键字

"msg" , // msg,String,需要打印的内容或信息

);

【回应】：

该函数无需返回。

### 修改设备名称

当进入设备设置界面需要修改设备名称时，WEB中调用js方法如下：

hilink. modifyDeviceName (

"deviceName", // deviceName,String,修改的设备名称字段

"resultCallback" //resultCallback，成功或失败时，将调用传入resultStr返回结果

);

【回应】：

当调用成功时，JSSDK将调用success对应的回调函数，将如下结果返回到resultStr中。

**{**

**"errcode": 0,**

**}**

【错误码】：

**{**

**"errcode": 11**

**}**

错误码取值参见最下方表格。

如果数据校验失败，返回[HILINK\_VALIDATE\_ERR](#_异常错误码描述_2)；

如果设备不在线，返回[HILINK\_DEV\_OFFLINE](#_异常错误码描述_2)；

如果设备响应超时，返回[HILINK\_DEV\_TIMEOUT](#_异常错误码描述_2)。

如果无网络导致删除失败，返回HILINK\_NO\_NETWORK

### 修改设备摆放位置

当进入设备设置界面需要修改设备摆放位置时，WEB中调用js方法如下：

hilink. modifyDeviceRoomName (

"chooseRoomString", // chooseRoomString,String, 修改的设备摆放位置信息的字段

"resultCallback" //resultCallback，成功或失败时，将调用传入resultStr返回结果

);

【回应】：

当调用成功时，JSSDK将调用success对应的回调函数，将如下结果返回到resultStr中。

**{**

**"errcode": 0,**

**}**

【错误码】：

**{**

**"errcode": 11**

**}**

错误码取值参见最下方表格。

如果数据校验失败，返回[HILINK\_VALIDATE\_ERR](#_异常错误码描述_2)；

如果设备不在线，返回[HILINK\_DEV\_OFFLINE](#_异常错误码描述_2)；

如果设备响应超时，返回[HILINK\_DEV\_TIMEOUT](#_异常错误码描述_2)。

如果无网络导致删除失败，返回HILINK\_NO\_NETWORK

### 拉起设备信息页面

当需要跳转到设备信息的界面时，WEB中调用js方法如下：

hilink. jumpTo (

"com.huawei.smarthome.deviceinfoactivity", // uri,String, 跳转到设备信息页面的uri

"resultCallback" //resultCallback，成功或失败时，将调用传入resultStr返回结果

);

【回应】：

当调用成功时，JSSDK将调用success对应的回调函数，将如下结果返回到resultStr中。

**{**

**"errcode": 0,**

**}**

【错误码】：

**{**

**"errcode": 11**

**}**

错误码取值参见最下方表格。

如果数据校验失败，返回[HILINK\_VALIDATE\_ERR](#_异常错误码描述_2)；

如果设备不在线，返回[HILINK\_DEV\_OFFLINE](#_异常错误码描述_2)；

如果设备响应超时，返回[HILINK\_DEV\_TIMEOUT](#_异常错误码描述_2)。

如果无网络导致删除失败，返回HILINK\_NO\_NETWORK

### 拉起设备网络信息页面

当需要跳转到设备网络信息的界面时，WEB中调用js方法如下：

hilink. jumpTo (

"com.huawei.smarthome. deviceNetinfoActivity", // uri,String, 跳转到设备信息页面的uri

"resultCallback" //resultCallback，成功或失败时，将调用传入resultStr返回结果

);

【回应】：

当调用成功时，JSSDK将调用success对应的回调函数，将如下结果返回到resultStr中。

**{**

**"errcode": 0,**

**}**

【错误码】：

**{**

**"errcode": 11**

**}**

错误码取值参见最下方表格。

如果数据校验失败，返回[HILINK\_VALIDATE\_ERR](#_异常错误码描述_2)；

如果设备不在线，返回[HILINK\_DEV\_OFFLINE](#_异常错误码描述_2)；

如果设备响应超时，返回[HILINK\_DEV\_TIMEOUT](#_异常错误码描述_2)。

如果无网络导致删除失败，返回HILINK\_NO\_NETWORK

### 拉起门锁匹配页面

当需要跳转到门锁匹配的界面时，WEB中调用js方法如下：

hilink. jumpTo (

"com.huawei.smarthome. addSmartlockActivity", // uri,String, 跳转到设备信息页面的uri

"resultCallback" //resultCallback，成功或失败时，将调用传入resultStr返回结果

);

【回应】：

当调用成功时，JSSDK将调用success对应的回调函数，将如下结果返回到resultStr中。

**{**

**"errcode": 0,**

**}**

【错误码】：

**{**

**"errcode": 11**

**}**

错误码取值参见最下方表格。

如果数据校验失败，返回[HILINK\_VALIDATE\_ERR](#_异常错误码描述_2)；

如果设备不在线，返回[HILINK\_DEV\_OFFLINE](#_异常错误码描述_2)；

如果设备响应超时，返回[HILINK\_DEV\_TIMEOUT](#_异常错误码描述_2)。

如果无网络导致删除失败，返回HILINK\_NO\_NETWORK

### 拉起设备设置页面

当需要在页面中跳转设备设置页面时，WEB中调用js方法如下：

hilink.jumpTo(

URI, //string, 跳转到设备设置界面的URI

resultCallback //string，成功或失败时，将调用传入resultStr返回结果

);

示例：

hilink.jumpTo(

" com.huawei.smarthome.deviceSettingActivity",

"resultCallback"

);

回调函数示例：

//web页面中需实现该函数，用来获取执行结果

function resultCallback (resultStr) {

var result = resultStr; //请求的返回结果

}

【回应】：

当调用成功时，JSSDK将调用success对应的回调函数，将如下结果返回到resultStr中。

**{**

**"errcode": 0**

**}**

【错误码】：

**{**

**"errcode": 11**

**}**

错误码取值参见最下方表格。

如果数据校验失败，返回[HILINK\_VALIDATE\_ERR](#_异常错误码描述_2)；

如果设备不在线，返回[HILINK\_DEV\_OFFLINE](#_异常错误码描述_2)；

### 启用重写安卓物理返回键及原生标题（包括IOS和Android）返回键

当重写安卓物理返回键及原生标题（包括IOS和Android）返回键时，WEB中调用js方法如下：

hilink. overrideBackPressed (

enable, //boolean, true启用，false不启用

resultCallback //string，成功或失败时，将调用传入resultStr返回结果

);

示例：

hilink. overrideBackPressed (

true,

"resultCallback"

);

回调函数示例：

//web页面中需实现该函数，用来获取执行结果

function resultCallback (resultStr) {

var result = resultStr; //请求的返回结果

}

【回应】：

当调用成功时，JSSDK将调用success对应的回调函数，将如下结果返回到resultStr中。

**{**

**"errcode": 0**

**}**

【错误码】：

**{**

**"errcode": 11**

**}**

错误码取值参见最下方表格。

如果数据校验失败，返回[HILINK\_VALIDATE\_ERR](#_异常错误码描述_2)；

如果设备不在线，返回[HILINK\_DEV\_OFFLINE](#_异常错误码描述_2)；

当enable为true时，安卓物理返回键将重写为调用一个js方法goBack()，web页面中需实现该函数来实现自己的返回功能。

//web页面中需实现该函数，用来实现自定义的返回

function goBack() {

//web 实现自定义的返回

}

### 获取设备profile

当需要获取设备profile时，WEB中调用js方法如下：

hilink. getDeviceProfile (

resultCallback //string，成功或失败时，将调用传入resultStr返回结果

);

示例：

hilink. getDeviceProfile (

"resultCallback"

);

回调函数示例：

//web页面中需实现该函数，用来获取执行结果

function resultCallback (resultStr) {

var result = resultStr; //请求的返回结果

}

【回应】：

当调用成功时，JSSDK将调用success对应的回调函数，将如下结果返回到resultStr中。

**{**

**"quickmenu": [**

**{**

**"content": [**

**{**

**"conditions": [**

**{**

**"operator": "=",**

**"path": "switch/on",**

**"type": "DEVICE\_DATA",**

**"value": "1"**

**}**

**],**

**"logic": "or",**

**"text": "开启"**

**}**

**]**

**}**

**],**

**"services": [**

**{**

**"characteristics": [**

**{**

**"characteristicName": "on",**

**"characteristicType": "bool",**

**"enumList": [**

**{**

**"descCh": "关",**

**"enumVal": 0**

**},**

**{**

**"descCh": "开",**

**"enumVal": 1**

**}**

**],**

**"max": 0,**

**"maxLength": 0,**

**"min": 0,**

**"report": 0,**

**"step": 0**

**}**

**],**

**"serviceId": "switch",**

**"serviceType": "binarySwitch"**

**}**

**],**

**"uiType": "H5",**

**"uriInfo": {**

**"uri": "9004/h5\_001/index.html"**

**}**

**}**

【错误码】：

**{**

**"errcode": 11**

**}**

错误码取值参见最下方表格。

如果数据校验失败，返回[HILINK\_VALIDATE\_ERR](#_异常错误码描述_2)；

如果设备不在线，返回[HILINK\_DEV\_OFFLINE](#_异常错误码描述_2)；

### App从后台恢复时调用函数

当app从后台（包括锁屏和将app置于后台）恢复或从原生设备设置页面返回时，安卓将调用一个js方法onResume()，web页面中需实现该函数来实现自己需要的功能。

//web页面中需实现该函数，用来实现自定义的返回

function onResume() {

//web 实现自定义的返回

}

### 启用重写原生标题（包括IOS和Android）设置按钮点击逻辑

当需要重写原生标题（包括IOS和Android）设置按钮点击逻辑时，WEB中调用js方法如下：

hilink. overrideSettingClick (

enable, //boolean, true启用，false不启用

resultCallback //string，成功或失败时，将调用传入resultStr返回结果

);

示例：

hilink. overrideSettingClick (

true,

"resultCallback"

);

回调函数示例：

//web页面中需实现该函数，用来获取执行结果

function resultCallback (resultStr) {

var result = resultStr; //请求的返回结果

}

【回应】：

当调用成功时，JSSDK将调用success对应的回调函数，将如下结果返回到resultStr中。

**{**

**"errcode": 0**

**}**

【错误码】：

**{**

**"errcode": 11**

**}**

错误码取值参见最下方表格。

如果数据校验失败，返回[HILINK\_VALIDATE\_ERR](#_异常错误码描述_2)；

如果设备不在线，返回[HILINK\_DEV\_OFFLINE](#_异常错误码描述_2)；

当enable为true时，原生标题设置按钮点击将重写为调用一个js方法goSetting()，web页面中需实现该函数来实现自己的返回功能。

//web页面中需实现该函数，用来跳转自定义的设备设置页面

function goSetting () {

//web 实现跳转自定义的设备设置页面

}

### 获取当前手机状态栏高度

当需要获取当前手机状态栏高度时，WEB中调用js方法如下：

hilink. getStatusBarHeight (

"resultCallback" //resultCallback，成功或失败时，将调用传入resultStr返回结果

);

【回应】：

当调用成功时，JSSDK将调用success对应的回调函数，将如下结果返回到resultStr中。

**{**

**“statusBarHeigt”:24,**

**}**

【错误码】：

**{**

**"errcode": 11**

**}**

错误码取值参见最下方表格。

如果数据校验失败，返回[HILINK\_VALIDATE\_ERR](#_异常错误码描述_2)；

如果设备不在线，返回[HILINK\_DEV\_OFFLINE](#_异常错误码描述_2)；

如果设备响应超时，返回[HILINK\_DEV\_TIMEOUT](#_异常错误码描述_2)。

### 循环下发同一sid 下命令

当需要下发某sid下多次属性改变时（目前主要用于灯类颜色变化下发），WEB中调用js方法如下：

hilink. setDeviceServiceLoop(

"**{devid}**", //devid,string,请取”0”，表示当前设备，暂不支持其它取值

"sid", //sid

"json\_body",//body,string,见下描述

isLoop, //isLoop,boolean,表示是否需要多次循环下发，若为No，将json\_body数组内属性依次下发完即停止，若为YES，将json\_body数组内属性依次下发完会再次重新依次下发

intervalTime, //依次下发的时间间隔，单位为毫秒

"resultCallback" //resultCallback，成功或失败时，将调用传入resultStr返回结果

);

json\_body中列举需要设置的设备sid、以及characteristicName和value的数组

**{"colour": [{"blue": 111,"green": 111,"red": 111},**

**{"blue": 222,"green": 222,red": 222},**

**{"blue": 333,"green": 333,"red": 333}]}**

Body不能为空，否则将失败

【回应】：

当调用成功时，JSSDK将调用success对应的回调函数，将如下结果返回到resultStr中。

**{**

**"errcode": 0**

**}**

【错误码】：

**{**

**"errcode": 11**

**}**

错误码取值参见最下方表格。

如果数据校验失败，返回[HILINK\_VALIDATE\_ERR](#_异常错误码描述_2)；

如果设备不在线，返回[HILINK\_DEV\_OFFLINE](#_异常错误码描述_2)；

如果设备响应超时，返回[HILINK\_DEV\_TIMEOUT](#_异常错误码描述_2)

### 下发多个sid命令到设备

当需要下发多个sid命令给设备时，WEB中调用js方法如下：

hilink. setMultipleDeviceInfo(

"**{devid}**", //devid,string,请取”0”，表示当前设备，暂不支持其它取值

"json\_body", //body,string(JsonArray), 见下描述

isParallel, //Boolean,是否并行下发，建议使用并行，串行出错概率较高

shouldWaitAll, //Boolean,是否需要所有命令下发才回调，若为NO，只要有一个命令成功就触发成功回调，所有命令失败才触发失败回调，若为YES，只有所有命令下发成功后才触发成功回调

"resultCallback" //resultCallback，成功或失败时，将调用传入resultStr返回结果

);

json\_body中列举出需要设置的设备sid、characteristicName以及要设置的value，整个body为一个jsonArray

**[**

**{"sid1":{**

**"characteristicName1": "value1",**

**"characteristicName2": "value2"**

**}},**

**{"sid2":{**

**"characteristicName1": "value1",**

**"characteristicName2": "value2"**

**}},**

**……**

**]**

Body不能为空，否则将失败。

【回应】：

当调用成功时，JSSDK将调用success对应的回调函数，将如下结果返回到resultStr中。

**{**

**"errcode": 0**

**}**

【错误码】：

**{**

**"errcode": 11**

**}**

错误码取值参见最下方表格。

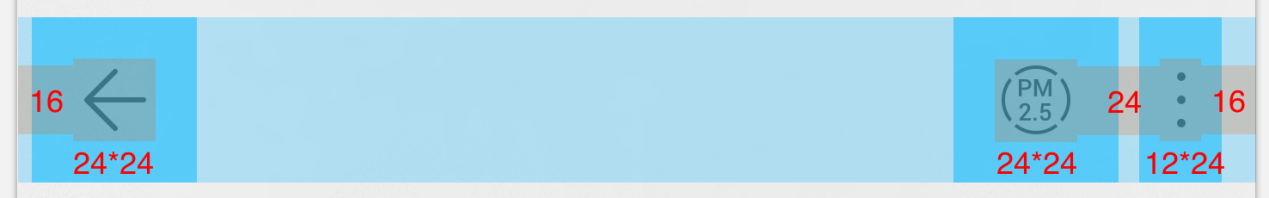
如果数据校验失败，返回[HILINK\_VALIDATE\_ERR](#_异常错误码描述_2)；

如果设备不在线，返回[HILINK\_DEV\_OFFLINE](#_异常错误码描述_2)；

如果设备响应超时，返回[HILINK\_DEV\_TIMEOUT](#_异常错误码描述_2)。

### 显示UI+特殊样式的导航栏

当需要显示UI+特殊样式的导航栏时，



WEB中调用js方法如下：

hilink. showUIplusAppBar(

"url", //url,string, 见下描述

"buttonClickCallback",// buttonClickCallback,string, 见下描述

"resultCallback" //resultCallback，成功或失败时，将调用传入resultStr返回结果

);

url中列举出需要显示在更多按钮左边的按钮图标下载相对路径（相对于本设备h5\_001文件夹路径），可以传0-2个，APP会从右往左排布。url格式如下：

**{  
          "url": "/sdk/themes/default/images/airpurify/pm25.png",  
          "url2": "/sdk/themes/default/images/airpurify/pm25.png"  
       }**

url可以为空，若为空则不显示任何图标。

buttonClickCallback中列举出上述按钮点击回调事件，回调时调用如下，只需将按钮的index传入js function即可，index对应上述url里面的index

**buttonClickCallback(‘index’)**

【回应】：

当调用成功时，JSSDK将调用success对应的回调函数，将如下结果返回到resultStr中。

**{**

**"errcode": 0**

**}**

【错误码】：

**{**

**"errcode": 11**

**}**

错误码取值参见最下方表格。

如果数据校验失败，返回[HILINK\_VALIDATE\_ERR](#_异常错误码描述_2)；

如果设备不在线，返回[HILINK\_DEV\_OFFLINE](#_异常错误码描述_2)；

如果设备响应超时，返回[HILINK\_DEV\_TIMEOUT](#_异常错误码描述_2)。

### APP本地保存用户收藏的灯光颜色

用于本地保存用户收藏的灯光颜色，包括获取与更新操作，主要用于智能灯，APP仅通过HWID/DevID作为key在APP本地进行持久化储存，不做数据校验。WEB中调用js方法如下：

hilink. collectLightColor(

"action", //action,string，目前仅有”get”/”update”

"json\_body", //body,string，见下描述，若action为get，此字段为空

"resultCallback" //resultCallback，成功或失败时，将调用传入resultStr返回结果

);

json\_body中列举用户收藏颜色的完整数组：

**[**

**{"color":"102,107,255"},**

**{"color":"102,107,255"},**

**{"color":"102,107,255"},**

**{"color":""},**

**{"color":"102,107,255"}**

**]**

若action为get，此字段为空

【回应】：

当调用成功时，JSSDK将调用success对应的回调函数，将如下结果返回到resultStr中。若action为”update”，返回字段如下

**{**

**"errcode": 0**

**}**

若action为”get”，返回字段如上述json\_body,若本地无颜色储存，返回非0错误码

【错误码】：

**{**

**"errcode": 11**

**}**

错误码取值参见最下方表格。

如果数据校验失败，返回[HILINK\_VALIDATE\_ERR](#_异常错误码描述_2)；

如果设备不在线，返回[HILINK\_DEV\_OFFLINE](#_异常错误码描述_2)；

如果设备响应超时，返回[HILINK\_DEV\_TIMEOUT](#_异常错误码描述_2)

# JSAPI通道的安全性

H5设备页面的安全性分为三个层面：

**1、传输的安全**

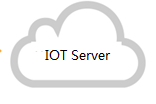
APP连接到IOT云去下载H5页面，这个过程是通过**HTTPS协议加密传输**的，APP会持有**云端数字证书**，对云的合法性进行验证，避免云端的仿冒攻击、劫持攻击、篡改攻击。

**2、运行的安全**

在APP执行页面阶段，H5运行环境会判断所加载的所有页面资源的URL进行校验，是否为IOT云地址前缀，避免加载未经授权页面内容。且运行环境中，对页面跳转做了禁止限制。

**3、第三方来源的安全（扩展）**

若第三方来源界面，将为每个设备页面**颁发所申请的接口权限的证书签名**。每个页面在调用初始化config接口时，需要传入IOT平台所发放的token及所申请使用的API列表，APP会采用预置证书公钥校验所申请的API的权限是否经过签名，避免未审批接口的调用。



APP

https

# 异常错误码描述

错误码规范：

0表示成功，其它值表示失败。

|  |  |  |
| --- | --- | --- |
| 错误值 | 枚举宏 | 说明 |
| 10 | HILINK\_DEV\_TIMEOUT | 设备请求没有响应 |
| 11 | HILINK\_DEV\_OFFLINE | 设备已离线 |
| 12 | HILINK\_VALIDATE\_ERR | 设备数据参数校验非法，在设备控制参数校验失败时，返回APP或者云端 |
| 14 | HILINK\_NO\_NETWORK | 无网络错误 |