LifeSmart 云平台服务接口 (v1.32)

版本	修订日期	修订人	修订内容
1.0	2015/10/20	AlexCheng	
1.01	2015/10/28	AlexCheng	修改返回信息。返回字段status修改为code,其值为错误代码。增加错误码列表
1.02	2015/11/4	AlexCheng	修改签名串例子中的错误格式
1.03	2015/11/9	AlexCheng	1：添加灯带/灯泡颜色控制 2：添加入墙开关控制 3：添加新增/删除智慧设备接口 4：添加注册用户接口
1.04	2015/11/19	Lewis Li	1．修改EpGetAll/EpSet/EpGet范例 2．添加EpAdd/EpRemove接口范例 3．细化注册用户接口 4．添加灯带动态设置
1.05	2016/10/26	Xiao Ye	1.添加EpsSet，SceneGet，SceneSet接口；
1.06	2016/11/3	Xiao Ye	1．添加UnregisterUser接口；
1.07	2017/1/10	AlexCheng	1：支持状态更新；2：支持多终端访问
1.08	2017/4/5	AlexCheng	支持多区域访问
1.10	2017/10/09	Jon Fan/ Pretty Ju	第二版整理
1.11	2018/04/20	Jon Fan	添加WebSocket事件详细说明
1.12	2018/09/20	Jon Fan	添加EpAdd扩展参数说明/ 添加EpSet扩展参数说明 修正文档描述不合理的部分
1.13	2018/10/08	Jon Fan	添加EpUpgradeAgt，EpRebootAgt， EpGetAgtLatestVersion文档说明； 设备属性增加lHeart,lDbm说明； 增加设备模型说明； EpSet增加修改Ep/IO名称的说明；
1.14	2018/11/25	Jon Fan	增加EpSearchSmart，EpAddSmart接口

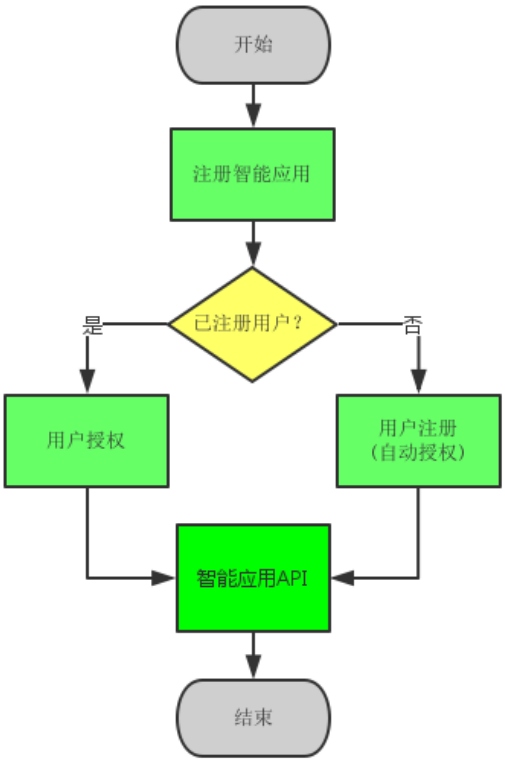
1.15	2018/12/05	Jon Fan	增加EpCmd， EpGetAgtState 接口说明，以及修改EpGetAll接口描述，增加agt_self以及智慧设备的描述。
1.16	2018/12/12	Jon Fan	WebSocket增加AI事件通知描述
1.17	2019/01/24	Jon Fan	增加设备/AI的ext_loc属性说明
1.18	2019/02/14	Jon Fan	增加EpGetAttrs(获取设备扩展属性)接口
1.19	2019/03/01	Jon Fan	增加EpSet接口修改智慧中心名称说明 增加EpTestRssi(测试射频设备信号强度)接口
1.20	2019/06/03	Jon Fan	增加EpGet/EpGetAll接口返回的设备IO属性里面v值的说明。增加授权返回的svrrgnid属性说明。 增加EpBatchSet接口
1.21	2019/10/18	Jon Fan	增加EpSearchIDev,EpAddIDev接口说明
1.22	2019/12/30	Jon Fan	增加EpMaintOtaFiles， EpMaintOtaTasks接口说明
1.23	2020/01/07	Jon Fan	增加EpMaintAgtRM接口说明
1.24	2020/03/11	Jon Fan	增加EpSetVar接口说明 EpUpgradeAgt接口增加HTTP升级方式说明
1.25	2020/04/08	Jon Fan	增加EpConfigAgt接口说明
1.26	2020/04/21	Jon Fan	设备模型增加fullCls描述
1.27	2020/07/01	Jon Fan	EpConfigAgt接口增加timezone设置 EpCmd接口增加云视户外摄像头声光警报设置
1.28	2021/06/10	Pretty Ju	更新附录1
1.29	2021/07/14	Jon Fan	“设备模型说明” 增加新属性说明
1.30	2022/02/08	Jon Fan	EpConfigAgt接口增加NIF配置
1.31	2022/03/02	Jon Fan	EpConfigAgt接口文档重新整理并增加本地互联接口说明
1.32	2022/06/23	Pretty Ju	去掉Headers["X-LS-SVRRGNID"]；更新附录1和附录2；细节描述优化；

1.介绍

LifeSmart云平台对外提供智慧设备的添加、查找、状态查询以及控制等服务。第三方应用通过HTTPS连接接入云平台，就可完成对智慧设备的操作与管理。同时，为了连接安全，所有请求请务必遵循云平台的签名规则。

1.1 接口使用流程

第三方应用使用云平台服务之前，需要先向LifeSmart注册智能应用，获得唯一的 appkey 和 apptoken ，然后第三方应用需由用户授权获取设备的访问控制权限才能提供服务。如果该用户没有注册LifeSmart账号，则可以通过云平台的用户注册接口注册用户并完成自动授权，得到 userid 和 usertoken以及过期时间expiredtime 。



2.注册智能应用

智能应用需要使用LifeSmart云平台的服务，必须先在LifeSmart云平台注册，获取**appkey**和**apptoken**，注册成功后将获取到如下信息：

名称	备注
appkey	应用的key，每个应用必须申请一个属于自己的唯一的key
apptoken	应用的token，请妥善保管好该token，不要泄漏

Note:
注册方式请与LifeSmart公司联系或通过 LifeSmart开放平台 自主注册并申请第三方应用。

3.智能应用获取用户授权

3.1.请求URL地址

```
https://api.ilifesmart.com/app/
auth.authorize?id=***&appkey=***&time=***&auth_callback=***&did=***&sign
=***&lang=zh
```

3.2.请求参数

参数名	类型	备注
id	int	消息id，标识这条消息，调用成功后将原样返回
appkey	string	智能应用申请时获得的appkey
time	int	UTC时间戳， 自1970年1月1日起计算的时间,单位为秒
auth_callback	string	授权成功后回调的URL
sign	string	签名值，签名算法见注解
did	string	(可选)终端唯一id，当有多终端的时候用于区分终端
lang	string	显示语言类型，当前支持zh,en,jp，默认为zh

Note:

- **time**：时间戳，如果请求的时间与云服务平台时间相差5分钟以上，则该请求无效。
- **did**：终端设备id，用于标识当前使用设备。如果需要支持多终端访问时，则需要传递该参数，否则不需要传递该参数。**【若授权的时候包含did值，则后续的API调用在使用授权获取的token的时候，其参数system.did的值必须等于授权的时候填写的did值】**
- **sign**：签名，生成算法如下：
  a) 签名原始串：*appkey=***&auth_callback=***&did=***&time=***&apptoken=****（签名原始字符串除*apptoken*外其他字段按照字母顺序排序生成，如果有*did*，则进行签名，否则不填入）
  b) 将"签名原始串"进行MD5编码，并转化为32位的16进制小写字符串，作为签名值sign。
  c) 注意：lang不放入签名原始串，即不需要签名。
  d) MD5算法必须正确，可用下面字符串进行对比验证：

签名原始串：

```
appkey=1111111111&auth_callback=http://localhost:8080/
CallBack.ashx&time=1445307713&apptoken=ABCDEFGHIKJLMJOBPOOFPDFDA
```

签名值sign应该为：

```
0972888fac34d1d151e4433c9dc7a102
```

请求URL服务地址 api.ilifesmart.com 并不是唯一，若明确为其它区域的应用，可以直接使用其它区域的URL地址，例如一个欧洲区域的应用，可以直接使用 api.eur.ilifesmart.com 服务地址，当前也可以仍旧使用 api.ilifesmart.com 服务地址完成应用授权，但之后的API调用必须使用授权返回的svrurl做为服务地址。

具体服务地址列表请参考 附录2 服务代号及地址对应表 。

3.3.授权过程

输入上面URL后，出现如下界面：

输入用户名和密码，验证通过之后跳转到如下页面：

点击授权之后页面跳转到URL提供的 **auth_callback** 的URL链接，URL中带有userid和usertoken等参数。智能应用端需要能读取到URL中的usertoken等内容，执行后续操作。

- 授权成功返回范例：

```
{
  "code": "success",
  "userid": "YOUR_USERID",
```

```
    "usertoken": "YOUR_TOKEN",
    "expiredtime": YOUR_EXPIRED_TIME,
    "rgn": "USER_RGN",
    "svrurl": "USER_SERVERURL",
    "svrrgnid": "USER_SEVER_RGNID"
}
```

- 授权失败返回范例：

```
{
    "code": "error",
    "message": "ERR_MESSAGE"
}
```

属性名	类型	描述
id	int	消息id号,为授权URL时传入数据
userid	string	用户id
usertoken	string	用户授权Token
expiredtime	int	Token失效过期时间，UTC时间戳，自1970年1月1日起计算的时间，单位为秒
svrurl	string	API服务地址。我们支持多区域多服务，不同用户的数据分布在不同的服务上面，该服务URL确定操作该用户数据需要访问的服务地址，调用API操作用户数据的时候请务必以该属性返回的URL地址为准，否则可能不能正确访问用户数据。
svrrgnid	string	用户所在区域ID，例如 "GS"，由于支持多区域，不同用户的所在区域可能会不同，该属性标识用户当前所在区域。

3.4.usertoken更新

usertoken必须在失效时间之前更新，否则就必须重新进行用户授权。

a) 令牌刷新地址：用户所在 svrurl + /auth.refreshtoken

例如：

用户A授权成功后所获得的svrurl="https://api.ilifesmart.com/app/",

那么用户A的令牌刷新地址为：https://api.ilifesmart.com/app/auth.refreshtoken

b) HTTP请求为POST方式，内容为JSON格式

c) 请求参数：

参数名	类型	备注
id	int	消息id号，调用成功后将原样返回
appkey	string	智能应用申请时获得的appkey

time	int	UTC时间戳，自1970年1月1日起计算的时间，单位为秒
userid	string	获取授权时得到的用户id
did	string	(可选)终端唯一id
sign	string	签名值，签名算法见注解

d) 签名原始串：

```
appkey=***&did=***&time=***&userid=***&apptoken=***&usertoken=***
```

- (签名原始字符串除apptoken和usertoken外其他字段按照字母顺序排序生成)
- usertoken为授权时获取的usertoken

e) 签名算法与授权一样

f) 调用成功后返回如下信息（JSON格式）：

属性名	类型	描述
id	int	消息id号，为授权请求时传入的数据
code	string	0:成功；其他:错误码
message	string	如果code等于0，则为空，否则返回错误信息
userid	string	用户id
usertoken	string	用户授权Token
expiredtime	int	Token失效过期时间，UTC时间戳，自1970年1月1日起计算的时间，单位为秒

g) 范例如下：

- 我们假定：

  appkey为APPKEY_XXXXXXXX，实际需要填写真实数据；
  apptoken为APPTOKEN_XXXXXXXX，实际需要填写真实数据；
  usertoken为USERTOKEN_XXXXXXXX，实际需要填写真实数据；
  sign为SIGN_XXXXXXXX，实际需要填写真实签名数据；

- 请求信息：

```
{
  "appkey": "APPKEY_XXXXXXXX",
  "time": 1445307713,
  "userid": "1111111",
  "sign": "SIGN_XXXXXXXX",
  "id": 12345
}
```

- 签名原始字符串为:

```
appkey=APPKEY_XXXXXXXX&time=1445307713&userid=1111111&apptoken=APPTOKEN_XXXXXXXX&usertoken=USERTOKEN_XXXXXXXX
```

- 回复信息：

```
{
  "code": 0,
  "message": "",
  "id": 12345,
  "userid": "1111111",
  "usertoken": "NEW_USERTOKEN_YYYYYYYY",
  "expiredtime": 1445955713
}
```

4. 智能应用API

4.1.HTTPS请求数据格式规范

4.1.1.URL

请求协议：HTTPS
请求方法：POST
请求URL：**请务必使用授权接口返回的服务器地址（svrurl字段）作为用户对应的服务地址**
具体服务地址列表请参考 **附录2 服务代号及地址对应表**
（请务必使用授权接口返回的 svrrgnid 匹配对应的WebSocket服务URL）

4.1.2.请求 Body JSON格式

id		消息序列号
system	ver	Protocol version
	lang	Language
	userid	User id
	appkey	appkey
	did	（可选）终端唯一id，如果在授权时填写，则此处必须填入相同id
	time	UTC时间戳，自1970年1月1日起计算的时间，单位为秒
	sign	签名值
method		API request method name
params	<attr>:<val>, ... <attr>:<val>	方法使用的参数集合

请求字段释义如下：

system：

- ver：版本号现是1.0
- lang：默认值是"en"，支持"en"，"zh"
- userid：授权时获取的userid

- appkey：智能应用注册时获取的appkey
- did：设备终端唯一id。如果需要多终端支持时，则必须填入。如果授权时填写该参数，则此时必须传入该参数，并且值要与授权时候填写的值相同
- time：请求时的时间戳(UTC)，零时区时间
- sign：签名值，服务端用作签名校验。签名算法见：**4.2.1**

method：目前支持的命令方式如下：

方法名称	URL后缀信息	描述
EpAddAgt	api.EpAddAgt	添加智慧设备
EpDeleteAgt	api.EpDeleteAgt	删除智慧设备
EpGetAllAgts	api.EpGetAllAgts	查询所有的智慧中心
EpAdd	api.EpAdd	添加子设备
EpRemove	api.EpRemove	删除子设备
EpSearchSmart	api.EpSearchSmart	获取智慧中心搜索到的附近其它智慧设备
EpAddSmart	api.EpAddSmart	把搜索到的附近智慧设备添加到智慧中心
EpSearchIDev	api.EpSearchIDev	获取智慧中心搜索到的附近IP网络设备
EpAddIDev	api.EpAddIDev	把搜索到的附近IP网络设备添加到智慧中心
EpGetAll	api.EpGetAll	查询该账户下授权给该App的所有智慧设备信息。若该App没有摄像头权限则返回的设备列表中不包括摄像头。
EpGet	api.EpGet	获取单个设备信息
EpSet	api.EpSet	控制单个设备
EpsSet	api.EpsSet	控制多个设备
EpCmd	api.EpCmd	控制单个设备(高级命令)
EpSetVar	api.EpSetVar	控制单个设备(低级命令)
EpBatchSet	api.EpBatchSet	批量快速设置多个设备属性
EpGetAttrs	api.EpGetAttrs	获取设备扩展属性
EpTestRssi	api.EpTestRssi	测试射频设备信息强度
EpUpgradeAgt	api.EpUpgradeAgt	升级智慧中心
EpRebootAgt	api.EpRebootAgt	重启智慧中心

EpGetAgtLatestVersion	api.EpGetAgtLatestVersion	获取智慧中心最新版本号
EpGetAgtState	api.EpGetAgtState	获取智慧设备状态
EpMaintOtaFiles	api.EpMaintOtaFiles	查看或维护智慧中心上的OTA文件列表
EpMaintOtaTasks	api.EpMaintOtaTasks	查看或维护智慧中心上的OTA任务列表
EpMaintAgtRM	api.EpMaintAgtRM	备份恢复智慧中心的配置，包括子设备以及AI的所有配置数据
EpConfigAgt	api.EpConfigAgt	设置智慧中心配置，如是否允许本地登录、修改本地登录密码
SceneGet	api.SceneGet	获取场景
SceneSet	api.SceneSet	触发场景
RegisterUser	auth.RegisterUser	注册新用户
UnregisterUser	auth.UnregisterUser	移除用户的授权，注意：该接口不会删除用户，只会回收授权。

params: 命令使用的参数

- ATTR_NAME： 属性名称，详见**《智慧设备属性定义》**
- ATTR_VALUE：属性值，详见**《智慧设备属性定义》**

4.1.4.应答JSON格式

应答字段	描述
id	消息id号，与请求的消息id号相同
code	成功：0；错误：详见 **4.3错误码**
message	如果code等于0即成功，则返回结果信息；否则返回错误文本信息

4.2.安全策略

为保障与云平台间的通讯安全，本协议要求所有通过 HTTPS 交互的请求必须携带请求源的签名信息，签名值作为 sign 存储在 HTTPS 请求的 system 参数集合中。

云平台首先会检查 HTTPS 请求中携带的 time 时间戳信息，如果与云平台系统时间差异大于 5 分钟，则视为无效的请求包，返回错误码。在时间戳检查正常的情况下，按照云平台统一的算法校验 sign 签名值，只有当签名值一致时，才接收 HTTPS 请求。

为实现上述安全机制，智能应用必须为其管理的设备和应用申请对应的 appkey 及 apptoken，申请方法见 **智能应用注册**。

4.2.1.签名算法

完成签名算法，需要如下两个步骤：

1. 收集签名原始串，"签名原始串" = method+params参数集合字串(将所有字段按升序排列后，依次连接所有字段名及对应值)+system参数集合中的did字符串(如果有)+time字串+userid+usertoken+appkey+apptoken，样式如下：

```
"<attr>:<val>,...,<attr>:<val>,did:<val>,time:<val>,userid:<val>,usertoken:<val>,appkey:<val>,apptoken:<val>"
```

2.将"签名原始串"进行MD5编码，并转化为32位的16进制小写字符串，作为sign签名值。

Note：

- 智能应用接入到云平台前必须先到云平台进行注册，申请并获取该应用的appkey和apptoken。
- 用户设备如果需要被智能应用管理时，需在该智能应用App上进行授权。智能应用拿到授权token后才能对智能设备进行控制操作。

4.2.2.签名范例

请求范例格式如下（ JSON）：

```
{
  "id": 123456,
  "system":
  {
    "ver": "1.0",
    "lang": "en",
    "userid": "1111111",
    "did": "DID_XXXXXXXX",
    "time": 1445307713,
    "appkey": "APPKEY_XXXXXXXX"
  },
  "method": "TestMethod",
  "params":
  {
    "param1": "12345",
    "param2": "abcde"
  }
}
```

将 params 部分的param1，param2按升序排序，组成无空格字符串，并在字串前加method，最后拼接 did、time、userid、usertoken、appkey、apptoken。最后组成的签名原始串如下：

```
"method:TestMethod,param1:12345,param2:abcde,did:DID_XXXXXXXX,time:1445307713,userid:1111111,usertoken:abcdefghijklmnopqrstuvwx,appkey:APPKEY_XXXXXXXX,apptoken:ABCDEFGHIKJLMJOBPOOFPDFDA"
```

签名值即对上述签名原始串计算MD5值，即：

```
sign=MD5("method:TestMethod,param1:12345,param2:abcde,did:DID_XXXXXXXX,time:1445307713,userid:1111111,usertoken:abcdefghijklmnopqrstuvwx,appkey:APPKEY_XXXXXXXX,apptoken:ABCDEFGHIKJLMJOBPOOFPDFDA")
```

最终sign值为：

```
2602efca4b1924fb1a7e62b78f2285b2
```

Note：

接口请求中经常遇到的错误返回“sign error”，一般原因如下：

- 仔细查看文档原始签名串的生成方式，留意params中参数的拼接字符是否正确；
- 用户授权获取usertoken时是否传入了did参数，在接口请求时需要将该字段信息保持一致；
- usertoken无效，即接口传入的usertoken错误或已失效，是否已经进行过令牌刷新操作等；

4.3.错误码

错误码	错误信息
0	成功
10001	请求格式错误
10002	Appkey不存在
10003	不支持HTTP GET请求
10004	签名非法
10005	用户没有授权
10006	用户授权已经过期
10007	非法访问
10008	内部错误
10009	设置属性失败
10010	Method非法
10011	操作超时
10012	用户名已存在
10013	设备没准备好
10014	设备已经被其他账户注册
10015	权限不够
10016	设备不支持该操作
10017	数据非法
10018	GPS位置非法访问拒绝
10019	请求对象不存在
10020	设备已经存在账户中
10022	请求地址需要重定向

4.4.设备模型说明

为了方便阐述API，这里我们对LifeSmart的设备模型做个简要的说明，对一些术语作出说明。
我们以EpGet/EpGetAll请求获取的数据为例：

```
{
  "name": "Smart Switch",
  "agt": "A3EAAABtAEwQXXXXXXXXXX",
  "me": "2d11",
  "devtype": "SL_SW_IF3",
  "fullCls": "SL_SW_IF3_V2",
  "stat": 1,
  "data": {
    "L1": {"type": 129, "val": 1,"name": "Living"},
    "L2": {"type": 128, "val": 0,"name": "Study"},
    "L3": {"type": 129, "val": 1,"name": "Kid"},
  },
  "ver": "0.1.6.49",
  "lDbm": -42,
  "lHeart": 1626229661
}
```

我们定义如下模型：

- **智慧中心(Agt)**： *"A3EAAABtAEwQXXXXXXXXXX"*
- **设备(EP)**： *"2d11"*，它是一个 *SL_SW_IF3* 类型的三联开关
- **设备属性(IO口)**： 设备的属性，可以用于读取状态，控制行为，*L1*、*L2*、*L3*它们都是设备的IO口，当然对于只读的IO口例如温度传感器，则只能读取状态，不能控制。
- **管理对象(MO)**： 泛指以上设备的总称，也可以包括AI对象，即MO可以是智慧中心，也可以是设备或者IO口，AI等。
- **智慧设备(SmartDevice)**：智慧设备是指可以独立工作的设备，例如传统的智慧中心，以及可以独立工作的Wi-Fi类单品，例如摄像头，超级碗SPOT等。
  可独立工作的Wi-Fi类单品通过EpSearchSmart, EpAddSmart命令又可以添加到智慧中心下面，这个时候智慧设备将会做为智慧中心下面的一个子设备(Ep)使用，加入到智慧中心之后，智慧设备将可以与其它设备一起参与到智慧中心的AI配置中。
  智慧设备本身都具有agt属性，但加入到智慧中心之后，其身份是做为一个子设备，其自身的agt属性将隐藏，可以通过Ep设备的属性:agt_self来获取其自身的agt属性。
  智慧设备加入到智慧中心之后，调用EpGetAllAgts将不会返回。

我们汇总了设备属性定义，如下表：

名称	类型	描述
agt	string	智慧中心ID

名称	类型	描述
agt_ver	string	智慧中心版本号
tmzone	int32	智慧中心时区设置
me	string	设备ID(智慧中心下面唯一)
devtype	string	设备类型(设备规格)
fullCls	string	包含版本号的完整的设备类型，一般它的值等于devtype+V[n]，V[n]指明其版本号信息，一般情况下使用devtype即可标识设备类型，在需要区分设备不同版本的特性的时候才需要用到fullCls。如果设备类型没有版本信息，则fullCls可能与devtype相同。
name	string	设备名称
stat	int32	设备在线状态， 0: Offline 1: Online
data	collection /map	设备下的IO口集合，是一个Map字典集合，键为IO口的名称(idx)，值为IO口的数据
data[IDX].type	int32	特定IO口的值类型
data[IDX].val	int32	特定IO口的值
data[IDX].v	float32	特定IO口的友好值 友好值指的是用户容易理解的值。 例如温度：val返回的是235，其友好值 v 为23.5，指示当前温度为23.5摄氏度； **注意：** • 也可以参考**《LifeSmart 智慧设备规格属性说明》**文档，基于type、val值同样可以计算出实际友好值来。 • 调用EpSet系列接口控制设备的时候，仍然需要取val值，例如设置空调的温度tT为26度，tT的val需设置为260，具体请参考**《LifeSmart 智慧设备规格属性说明》**文档。
data[IDX].name	string	特定IO口的名称(注意：如果IO口没有命名过，则不会返回这个属性值)
lHeart	int32	设备最近一次心跳时间，UTC时间戳，自1970年1月1日起计算的时间，单位为秒。
lDbm	int32	设备的dBm值，其值为负值，值越接近0表明信号质量越好。注意该属性指的是射频类设备的信号强度，Wi-Fi类设备将不存在这个属性值。

名称	类型	描述
agt_self	string	智慧设备本身的agt属性。如果一个智慧设备（例如摄像头，超级碗SPOT）被加入到智慧中心下面做为一个设备使用，则该智慧设备原先的agt属性将会被隐藏，为了显示这个属性值，将以agt_self属性提供。EpGetAll方法里面将会返回这个属性值若存在的话。 或者一个设备A级联到其它智慧中心下面做为设备B使用，则设备B将会有agt_self属性，其值指明设备A自己的agt属性
me_self	string	智慧设备本身的me属性。如果一个智慧设备（例如摄像头，超级碗SPOT）被加入到智慧中心下面做为一个设备使用，则该智慧设备原先的me属性将会被隐藏，为了显示这个属性值，将以me_self属性提供。EpGetAll方法里面将会返回这个属性值若存在的话。 或者一个设备A级联到其它智慧中心下面做为设备B使用，则设备B将会有me_self属性，其值指明设备A原本的me属性
ext_loc	string	设备扩展属性，第三方应用可以使用该字段存储需要的扩展属性。注意：该字段最大长度不能超过512字符，且必须为字符串类型。 该字段为第三方应用专用，LifeSmart应用不需要使用该字段。
ver	string	设备固件版本号

4.5.接口定义

4.5.1.EpAddAgt 增加智慧中心

4.5.1.1.JSON请求数据格式

Type	Definition		Must	Description
Interface Name	EpAddAgt			增加智慧中心
Partial URL	api.EpAddAgt			
Content Type	application/json			
HTTP Method	POST			
Request Content	system	ver	Y	1.0
		lang	Y	en
		sign	Y	签名值
		userid	Y	User ID
		appkey	Y	appkey
		did	O	(可选)终端唯一id。如果在授权时填入了，此处必须填入相同id
		time	Y	UTC时间戳，自1970年1月1日起计算的时间，单位为秒
	method		Y	EpAddAgt
	params	sn	Y	通过发现协议获取的AGT字段或 智慧中心背部条码
		name	Y	智慧中心名称
	id		Y	消息id号

4.5.1.2.范例

- 我们假定：
  appkey为APPKEY_XXXXXXXX，实际需要填写真实数据；
  apptoken为APPTOKEN_XXXXXXXX，实际需要填写真实数据；
  usertoken为USERTOKEN_XXXXXXXX，实际需要填写真实数据；
  sign为SIGN_XXXXXXXX，实际需要填写真实签名数据；

- 请求地址：
  svrurl+PartialURL，例如：

```
https://api.ilifesmart.com/app/api.EpAddAgt
```

- 请求信息：

```
{
  "id": 957,
  "method": "EpAddAgt",
  "params": {
    "sn": "xxxxxxxx",
    "name": "xxx"
  },
  "system": {
    "ver": "1.0",
    "lang": "en",
    "userid": "1111111",
    "appkey": "APPKEY_XXXXXXXX",
    "time": 1447641115,
    "sign": "SIGN_XXXXXXXX"
  }
}
```

- 签名原始字符串：

```
method:EpAddAgt,name:xxx,sn:xxxxxxxx,time:1447641115,userid:1111111,
usertoken:USERTOKEN_XXXXXXXX,appkey:APPKEY_XXXXXXXX,apptoken:APPTOKE
N_XXXXXXXX
```

- 回复信息：

```
{
  "id": 957,
  "code": 0,
  "message": [
    {
      "agt": "A3EAAABdADQQXXXXXXXXXXX",
      "name": "我的智慧中心"
    }
  ]
}
```

4.5.2.EpDeleteAgt 删除智慧中心

4.5.2.1.JSON请求数据格式

Type	Definition		Must	Description
Interface Name	EpDeleteAgt			删除智慧中心
Partial URL	api.EpDeleteAgt			
Content Type	application/json			
HTTP Method	HTTP POST			
Request Content	system	ver	Y	1.0
		lang	Y	en
		sign	Y	签名值
		userid	Y	User ID
		appkey	Y	appkey
		did	O	(可选)终端唯一id。如果在授权时填入了，此处必须填入相同id
		time	Y	UTC时间戳, 自1970年1月1日起 计算的时间, 单位为秒
	method		Y	EpDeleteAgt
	params	agt	Y	智慧中心
	id		Y	消息id号

4.5.2.2.范例

- 我们假定：
  appkey为APPKEY_XXXXXXXX，实际需要填写真实数据；
  apptoken为APPTOKEN_XXXXXXXX，实际需要填写真实数据；
  usertoken为USERTOKEN_XXXXXXXX，实际需要填写真实数据；
  sign为SIGN_XXXXXXXX，实际需要填写真实签名数据；

- 请求地址：
  svrurl+PartialURL，例如：

```
https://api.ilifesmart.com/app/api.EpDeleteAgt
```

- 请求信息：

```
{
  "id": 957,
   "method": "EpDeleteAgt",
   "params": {
     "agt": "A3EAAABdADQQXXXXXXXXXXX"
   },
   "system": {
     "ver": "1.0",
     "lang": "en",
     "userid": "1111111",
     "appkey": "APPKEY_XXXXXXXX",
     "time": 1447641115,
     "sign": "SIGN_XXXXXXXX"
   }
 }
```

- 签名原始字符串为：

```
method:EpDeleteAgt,agt:A3EAAABdADQQXXXXXXXXXXX,time:1447641115,userid:1111111,usertoken:USERTOKEN_XXXXXXXX,appkey:APPKEY_XXXXXXXX,apptoken:APPTOKEN_XXXXXXXX
```

- 回复信息：

```
{
 "id": 957,
 "code": 0,
 "message": "success"
}
```

4.5.3.EpGetAllAgts 获取所有智慧中心

4.5.3.1.JSON请求数据格式

Type	Definition		Must	Description
Interface Name	EpGetAllAgts			获取所有智慧中心
Partial URL	api.EpGetAllAgts			
Content Type	application/json			
HTTP Method	POST			
Request Content	system	ver	Y	1.0
		lang	Y	en
		sign	Y	签名值
		userid	Y	User ID
		appkey	Y	appkey
		did	O	(可选)终端唯一id。如果在授权时填入了，此处必须填入相同id
		time	Y	UTC时间戳，自1970年1月1日起计算的时间, 单位为秒
	method		Y	EpGetAllAgts
	id		Y	消息id号

4.5.3.2.范例

- 我们假定：
  appkey为APPKEY_XXXXXXXX，实际需要填写真实数据；
  apptoken为APPTOKEN_XXXXXXXX，实际需要填写真实数据；
  usertoken为USERTOKEN_XXXXXXXX，实际需要填写真实数据；
  sign为SIGN_XXXXXXXX，实际需要填写真实签名数据；

- 请求地址：
  svrurl+PartialURL，例如：

```
https://api.ilifesmart.com/app/api.EpGetAllAgts
```

- 请求信息：

```
{
  "id": 957,
  "method": "EpGetAllAgts",
   "system": {
     "ver": "1.0",
     "lang": "en",
     "userid": "1111111",
     "appkey": "APPKEY_XXXXXXXX",
     "time": 1447641115,
     "sign": "SIGN_XXXXXXXX"
   }
 }
```

- 签名原始字符串为：

```
method:EpGetAllAgts,time:1447641115,userid:1111111,usertoken:USERTOK
EN_XXXXXXXX,appkey:APPKEY_XXXXXXXX,apptoken:APPTOKEN_XXXXXXXX
```

- 回复信息：

```
{
 "id": 957,
 "code": 0,
 "message": [
    {
      "agt": "A3EAAABdADQQXXXXXXXXXX",
      "name": "我的智慧中心",
      "agt_ver": "1.0.33p1",
      "stat": 1,
      "tmzone": 8,
    }
  ]
}
```

4.5.4.EpAdd 添加设备

4.5.4.1.JSON请求数据格式

Type	Definition		Must	Description
Interface Name	EpAdd			添加射频设备
URL	api.EpAdd			
Content Type	application/json			
HTTP Method	POST			
Request Content	system	ver	Y	1.0
		lang	Y	en
		sign	Y	签名值
		userid	Y	User id
		appkey	Y	appkey
		did	O	(可选)终端唯一id。如果在授权时填入了，此处必须填入相同id
		time	Y	UTC时间戳, 自1970年1月1日起 计算的时间, 单位为秒
	method		Y	EpAdd
	params	agt	Y	智慧中心Id,为EpGetAllAgts返回时信息
		optarg	O	添加设备额外参数，这是一个可选的、高级的选项。见后面描述。 数据类型是JSON对象的序列化字符串。
	id		Y	消息id号

说明：射频设备需要添加到智慧中心才能工作，射频设备添加的时候需要对码，使得待添加的设备与智慧中心都进入相应通道，识别匹配到对方才算完成。对码过程有个超时时间，时间定义缺省是20秒，因此需要把握好时间，使得字设备进入对码状态与调用EpAdd命令的时间间隔尽可能的短。

4.5.4.2.范例

- 我们假定：
  appkey为APPKEY_XXXXXXXX，实际需要填写真实数据；
  apptoken为APPTOKEN_XXXXXXXX，实际需要填写真实数据；
  usertoken为USERTOKEN_XXXXXXXX，实际需要填写真实数据；
  sign为SIGN_XXXXXXXX，实际需要填写真实签名数据；

- 请求地址：
  svrurl+PartialURL，例如：

```
https://api.ilifesmart.com/app/api.EpAdd
```

- 请求信息：

```
{
  "id": 829,
  "method": "EpAdd",
  "system": {
    "ver": "1.0",
    "lang": "en",
    "userid": "1111111",
    "appkey": "APPKEY_XXXXXXXX",
    "time": 1447643442,
    "sign": "SIGN_XXXXXXXX"
  },
  "params": {
    "agt": "A3EAAABdADQQXXXXXXXXXX"
  }
}
```

- 签名原始字符串为：

```
method:EpAdd,agt:A3EAAABdADQQXXXXXXXXXX,time:1447643442,userid:1111111,usertoken:USERTOKEN_XXXXXXXX,appkey:APPKEY_XXXXXXXX,apptoken:APPTOKEN_XXXXXXXX
```

- 回复信息：

```
{
  "id": 829,
  "code": 0,
  "message":{
     "me": "271c"
  }
}
```

说明：返回message.me属性指明新添加设备的me属性。

4.5.4.3.optarg 参数说明

optarg参数是添加设备的额外参数，一般情况下，不使用该参数已经可以很好的工作，但对于有些设备，为了实现灵活的定制，可以使用该参数。
该参数与设备类型息息相关，不同的设备有不同的参数，如果设备没有额外参数，则将忽略该参数的值。当前可用的额外参数有如下：

多功能(CUBE)环境感应器

```
optarg = {
  "cls":"SL_SC_BE",
  "exarg":{
    "humidity_display":1/2/3,
    "temperature_display":1/2/3
  }
}
```

humidity_display 属性用于确定多功能(CUBE)环境感应器液晶屏显示的内容，可以选择 **湿度**、**光照**、**湿度与光照**，分别对应值1、2、3。
temperature_display 属性用于确定多功能(CUBE)环境感应器液晶屏对温度显示类别选择，可以选择 **摄氏温度**、**华氏温度**、**摄氏与华氏温度**，分别对应值1、2、3。

多功能(CUBE)动态感应器

```
optarg = {
  "cls":"SL_SC_BM",
  "exarg":{
    "warning_duration":[6-814]
  }
}
```

warning_duration 属性用于确定检测到移动后的警报持续时间（单位：秒），缺省为秒，可选范围有6-814秒。

耶鲁门锁模块

```
optarg = {
  "cls":"SL_LK_YL",
  "exarg":{
    "enable_remote_unlock":1/0
  }
}
```

enable_remote_unlock 属性用于确定耶鲁门锁模块是否支持远程开门，可以选择 **支持**、**不支持**，分别对应值1、0。

恒星/辰星/极星开关/开关伴侣系列

添加恒星/辰星/极星开关/开关伴侣系列必须指明设备规格，否则将不能正确的添加。

同时恒星/辰星/极星开关系列还可以设置工作模式，分别为：速度优先、电量优先。其配置如下：

```
optarg = {
  "cls":"SL_MC_ND3_V2",
  "exarg":{
    "mode_selection":"speed"
  }
}
```

cls指明其为极星三联开关；
当前恒星/辰星/极星开关系列cls定义如下：

- SL_SW_ND1_V1/SL_SW_ND2_V1/SL_SW_ND3_V1 恒星/辰星开关1联/2联3联
- SL_MC_ND1_V1/SL_MC_ND2_V1/SL_MC_ND3_V1 恒星/辰星开关伴侣1联/2联/3联
- SL_SW_ND1_V2/SL_SW_ND2_V2/SL_SW_ND3_V2 极星开关1联/2联3联
- SL_MC_ND1_V2/SL_MC_ND2_V2/SL_MC_ND3_V2 极星开关伴侣1联/2联/3联

mode_selection 属性指明工作模式，可以选择"speed"、"power" ，分别对应速度优先、电量优先，缺省模式为速度优先。

特殊指明设备

有些设备在对码的时候必须指明类型，以便API接口进行更好的添加操作，因此添加这些设备，请在optarg属性里面指明要添加的设备规格类型。

当前强制需要指明的设备规格有如下：

- PSM: PSM系列
- SL_P_IR: SPOT(MINI)

我们以恒星开关为例，参数内容如下：

```
optarg = {
  "cls":"SL_SW_ND1"
}
```

说明：其参数数据格式是JSON对象的序列化字符串，并且要参与方法签名中。

4.5.5.EpRemove 删除设备

4.5.5.1.JSON请求数据格式

Type	Definition		Must	Description
Interface Name	EpRemove			删除设备
Partial URL	api.EpRemove			
Content Type	application/json			
HTTP Method	POST			
Request Content	system	ver	Y	1.0
		lang	Y	en
		sign	Y	签名值
		userid	Y	User ID
		appkey	Y	appkey
		did	O	(可选)终端唯一id。如果在授权时填入了，此处必须填入相同id
		time	Y	UTC时间戳，自1970年1月1日起 计算的时间, 单位为秒
	method		Y	EpRemove
	params	agt	Y	欲删除设备的agt，EpGetAll返回时信息
		me	Y	欲删除设备me，为EpGetAll返回时信息
	id		Y	消息id号

4.5.5.2.范例

- 我们假定：
  appkey为APPKEY_XXXXXXXX，实际需要填写真实数据；
  apptoken为APPTOKEN_XXXXXXXX，实际需要填写真实数据；
  usertoken为USERTOKEN_XXXXXXXX，实际需要填写真实数据；
  sign为SIGN_XXXXXXXX，实际需要填写真实签名数据；

- 请求地址：
  svrurl+PartialURL，例如：

```
https://api.ilifesmart.com/app/api.EpRemove
```

- 请求信息：

```
{
 "id": 46,
  "method": "EpRemove",
  "system": {
    "ver": "1.0",
    "lang": "en",
    "userid": "1111111",
    "appkey": "APPKEY_XXXXXXXX",
    "time": 1447642457,
    "sign": "SIGN_XXXXXXXX"
  },
  "params": {
    "agt": "A3EAAABdADQQXXXXXXXXXX",
    "me": "2832"
  }
}
```

- 签名原始字符串：

```
method:EpRemove,agt:A3EAAABdADQQXXXXXXXXXX,me:2832,time:1447642457,u
serid:1111111,usertoken:USERTOKEN_XXXXXXXX,appkey:APPKEY_XXXXXXXX,ap
ptoken:APPTOKEN_XXXXXXXX
```

- 回复信息：

```
{
  "id": 46,
  "code": 0,
  "message": "success"
}
```

4.5.6.EpGetAll 获取所有设备信息

4.5.6.1.JSON请求数据格式

Type	Definition		Must	Description
Interface Name	EpGetAll			查询该账户下授权给appkey的所有设备信息。设备信息不包括摄像头信息(若没有摄像头权限)。
Partial URL	api.EpGetAll			
Content Type	application/json			
HTTP Method	POST			
Request Content	system	ver	Y	1.0
		lang	Y	en
		sign	Y	签名值
		userid	Y	User id
		appkey	Y	appkey
		did	O	(可选)终端唯一id。如果在授权时填入了，此处必须填入相同id
		time	Y	UTC时间戳, 自1970年1月1日起 计算的时间, 单位为秒
	method		Y	EpGetAll
	params	degree	N	degree: 查询返回的详细程度，若不提供则缺省为2。 0: 返回最基本的信息，包括agt,me,devtype,name,stat 1: 返回data数据 2: 返回agt_ver,lDbm,lHeart 3: 返回agt_self
	id		Y	消息id号

4.5.6.2.范例

- 我们假定：
  appkey为APPKEY_XXXXXXXX，实际需要填写真实数据；
  apptoken为APPTOKEN_XXXXXXXX，实际需要填写真实数据；
  usertoken为USERTOKEN_XXXXXXXX，实际需要填写真实数据；
  sign为SIGN_XXXXXXXX，实际需要填写真实签名数据；

- 请求地址：
  svrurl+PartialURL，例如：

```
https://api.ilifesmart.com/app/api.EpGetAll
```

- 请求信息：

```
{
  "id": 144,
  "method": "EpGetAll",
  "system": {
    "ver": "1.0",
    "lang": "en",
    "userid": "1111111",
    "appkey": "APPKEY_XXXXXXXX",
    "time": 1447396020,
    "sign": "SIGN_XXXXXXXX"
  }
}
```

- 签名原始字符串：

```
method:EpGetAll,time:1447395539,userid:1111111,usertoken:USERTOKEN_X
XXXXXXX,appkey:APPKEY_XXXXXXXX,apptoken:APPTOKEN_XXXXXXXX
```

- 回复信息：

```
{
  "id": 144,
  "code": 0,
  "message": [{
      "agt": "A3EAAABdADQQXXXXXXXXXX",
      "me": "2711",
      "devtype": "SL_SW_IF2",
      "name": "流光开关",
      "stat": 1,
      "data": {
        "L1": {"type": 129,"val": 1,"name": "客厅"},
        "L2": {"type": 128,"val": 0,"name": "餐厅"},
      },
      "lDbm": -35,
      "lHeart": 1539143970
   },{
      "agt": "A3EAAABdADQQXXXXXXXXXX",
```

```
        "me": "2713",
        "devtype": "SL_LI_RGBW",
        "name": "智慧灯泡",
        "ext_loc": "{\"key\": \"LS\", \"location\": "HangZhou"}",
        "stat": 1,
        "data": {
          "RGBW": {"type": 255,"val": 2147483648},
          "DYN": {"type": 254,"val": 0}
        },
        "lDbm": -46,
        "lHeart": 1539143970
    }]
}
```

提示：若存在ext_loc属性则将返回ext_loc属性值。

4.5.7.EpGet 获取指定设备信息

4.5.7.1.JSON请求数据格式

Type	Definition		Must	Description
Interface Name	EpGet			获取设备信息
Partial URL	api.EpGet			
Content Type	application/json			
HTTP Method	POST			
Request Content	system	ver	Y	1.0
		lang	Y	en
		sign	Y	签名值
		userid	Y	User id
		appkey	Y	appkey
		did	O	(可选)终端唯一id。如果在授权时填入了，此处必须填入相同id
		time	Y	UTC时间戳，自1970年1月1日起 计算的时间, 单位为秒
	method		Y	EpGet
	params	agt	Y	欲查询设备的agt， 为EpGetAll返回时信息
		me	Y	欲查询设备me，为GetAll返回时信息
	Id		Y	消息id号

4.5.7.2.范例

- 我们假定：
  appkey为APPKEY_XXXXXXXX，实际需要填写真实数据；
  apptoken为APPTOKEN_XXXXXXXX，实际需要填写真实数据；
  usertoken为USERTOKEN_XXXXXXXX，实际需要填写真实数据；
  sign为SIGN_XXXXXXXX，实际需要填写真实签名数据；

- 请求地址：
  svrurl+PartialURL，例如：

```
https://api.ilifesmart.com/app/api.EpGet
```

- 请求信息：

Type	Definition		Must	Description
Interface Name	EpSet			控制设备
Partial URL	api.EpSet			
Content Type	application/json			
HTTP Method	POST			
Request Content	system	ver	Y	1.0
		lang	Y	en
		sign	Y	签名值
		userid	Y	User id
		appkey	Y	appkey
		did	O	(可选)终端唯一id。如果在授权时填入了，此处必须填入相同id
		time	Y	UTC时间戳，自1970年1月1日起 计算的时间, 单位为秒
	method		Y	EpGet
	params	agt	Y	欲设置设备的agt， 为EpGetAll返回时信息
		me	Y	欲设置设备me，为EpGetAll返回时信息
		…	Y	根据不同设备传入不同参数，具体请见"智慧设备设置属性定义"
		tag	N	控制设备的自定义标记串，可以不填写
		nonVolatile	O	是否需要掉电后数据不丢失的设置。等于1表示需要。注意：仅有部分设备的属性支持这个功能。
		bandResult	O	是否在命令执行完之后在回应(Response)消息里面携带IO口最新配置。等于1表示需要。注意：可能存在部分设备在执行完之后不支持返回IO口最新配置。
	id		Y	消息id号

```
{
  "id": 974,
  "method": "EpGet",
  "system": {
    "ver": "1.0",
    "lang": "en",
    "userid": "1111111",
    "appkey": "APPKEY_XXXXXXXX",
    "time": 1447639497,
    "sign": "SIGN_XXXXXXXX"
  },
  "params": {
    "agt": "A3EAAABdADQQXXXXXXXXXX",
    "me": "2711"
  }
}
```

- 签名原始字符串：

```
method:EpGet,agt:A3EAAABdADQQXXXXXXXXXX,me:2711,time:1447639497,user
id:1111111,usertoken:USERTOKEN_XXXXXXXX,appkey:APPKEY_XXXXXXXX,appto
ken:APPTOKEN_XXXXXXXX
```

- 回复信息：

```
{
  "id": 974,
  "code": 0,
  "message": [
  {
    "agt": "A3EAAABdADQQXXXXXXXXXX",
    "me": "2711",
    "devtype": "SL_SW_IF2",
    "name": "流光开关",
    "stat": 1,
    "data": {
      "L1": {"type": 129,"val": 1,"name": "客厅"},
      "L2": {"type": 128,"val": 0,"name": "餐厅"},
    },
      "lDbm": -35,
      "lHeart": 1539143970
  }]
}
```

4.5.8.EpSet 控制设备

4.5.8.1.JSON请求数据格式

Type	Definition		Must	Description
Interface Name	EpSet			控制设备
Partial URL	api.EpSet			
Content Type	application/json			
HTTP Method	POST			
Request Content	system	ver	Y	1.0
		lang	Y	en
		sign	Y	签名值
		userid	Y	User id
		appkey	Y	appkey
		did	O	(可选)终端唯一id。如果在授权时填入了，此处必须填入相同id
		time	Y	UTC时间戳，自1970年1月1日起 计算的时间, 单位为秒
	method		Y	EpGet
	params	agt	Y	欲设置设备的agt， 为EpGetAll返回时信息
		me	Y	欲设置设备me，为EpGetAll返回时信息
		…	Y	根据不同设备传入不同参数，具体请见"智慧设备设置属性定义"
		nonVolatile	O	是否需要掉电后数据不丢失的设置。等于1表示需要。注意：仅有部分设备的属性支持这个功能。
		bandResult	O	是否在命令执行完之后在回应(Response)消息里面携带IO口最新配置。等于1表示需要。注意：可能存在部分设备在执行完之后不支持返回IO口最新配置。
	id		Y	消息id号

4.5.8.2.范例

- 我们假定：
  appkey为APPKEY_XXXXXXXX，实际需要填写真实数据；
  apptoken为APPTOKEN_XXXXXXXX，实际需要填写真实数据；
  usertoken为USERTOKEN_XXXXXXXX，实际需要填写真实数据；
  sign为SIGN_XXXXXXXX，实际需要填写真实签名数据；

- 请求地址：
  svrurl+PartialURL，例如：

```
https://api.ilifesmart.com/app/api.EpSet
```

- 请求信息：

```
{
  "id": 191,
  "method": "EpSet",
  "system": {
    "ver": "1.0",
    "lang": "en",
    "userid": "1111111",
    "appkey": "APPKEY_XXXXXXXX",
    "time": 1447640772,
    "sign": "SIGN_XXXXXXXX"
  },
  "params": {
    "agt": "A3EAAABdADQQRzM0Njg5NA",
    "me": "2832",
    "idx": "RGBW",
    "type": 128,
    "val": 0,
    "tag": "m"
  }
}
```

- 签名原始字符串：

```
method:EpSet,agt:A3EAAABdADQQRzM0Njg5NA,idx:RGBW,me:2832,tag:m,type:
128,val:0,time:1447640772,userid:1111111,usertoken:USERTOKEN_XXXXXXX
X,appkey:APPKEY_XXXXXXXX,apptoken:APPTOKEN_XXXXXXXX
```

- 回复信息：

```
{
  "id": 191,
  "code": 0,
  "message": "success"
}
```

提示：如何修改设备、IO口、智慧中心的名称?

EpSet接口支持修改设备、IO口与智慧中心的名称。
若需要修改设备名称，请指明 {agt,me,name} 属性；
若需要修改设备IO口的名称，请指明 {agt,me,idx,name} 属性。
若需要修改智慧中心的名称，请指明 {agt,me='NULL',name} 属性。
例如 *{agt="A3EAAABdADQQRzM0Njg5NA",me="2d32", name="*我的插座*"}* 说明要把设备2832命名为"我的插座"；
例如 *{agt="A3EAAABdADQQRzM0Njg5NA",me="2d33", idx="L1", name="*客厅灯*"}* 说明要把2833这个三联开关的第一路开关命名为"客厅灯"；
例如 *{agt="A3EAAABdADQQRzM0Njg5NA",me="NULL", name="*我的智慧中心*"}* 说明要把智慧中心*A3EAAABdADQQRzM0Njg5NA*命名为"我的智慧中心"；

注意：不是所有的IO口命名都有意义，当前主要是多联开关类设备，例如三联流光开关灯有意义。

注意：修改智慧中心的名称的时候，me属性必须填写为"NULL"。

提示：如何修改设备的ext_loc属性?

EpSet接口支持修改设备的ext_loc属性的值。
若需要修改设备ext_loc值，请指明 {agt,me,ext_loc} 属性；
例如 *{agt="A3EAAABdADQQRzM0Njg5NA",me="2d32", ext_loc="*{\"key\": \"LS\", \"location\": "HangZhou"}"*}* 说明要把设备2832的ext_loc属性修改为"{\"key\": \"LS\", \"location\": "HangZhou"}";
注意：ext_loc属性可以与name属性一起修改，同时指明它们的值即可。

4.5.9.EpsSet 控制多个设备

4.5.9.1.JSON请求数据格式

Type	Definition		Must	Description
Interface Name	EpsSet			控制多个设备
Partial URL	api.EpsSet			
Content Type	application/json			
HTTP Method	POST			
Request Content	system	ver	Y	1.0
		lang	Y	en
		sign	Y	签名值
		userid	Y	User ID
		appkey	Y	appkey
		did	O	(可选)终端唯一id。如果在授权时填入了，此处必须填入相同id
		time	Y	UTC时间戳, 自1970年1月1日起 计算的时间, 单位为秒
	method		Y	EpsSet
	params	args	Y	欲设置的执行列表，列表中每个内容参照EpSet的params参数； 需要将列表转化成**JSON**格式字符串赋值给args
		nonVolatile	O	是否需要掉电后数据不丢失的设置。等于1表示需要。注意：仅有部分设备的属性支持这个功能。
		bandResult	O	是否在命令执行完之后在回应(Response)消息里面携带IO口最新配置。等于1表示需要。注意：可能存在部分设备在执行完之后不支持返回IO口最新配置。
	id		Y	消息id号

4.5.9.2.范例

- 我们假定：
  appkey为APPKEY_XXXXXXXX，实际需要填写真实数据；
  apptoken为APPTOKEN_XXXXXXXX，实际需要填写真实数据；
  usertoken为USERTOKEN_XXXXXXXX，实际需要填写真实数据；
  sign为SIGN_XXXXXXXX，实际需要填写真实签名数据；

- 请求地址：
  svrurl+PartialURL，例如：

```
https://api.ilifesmart.com/app/api.EpsSet
```

- 请求信息：

```
{
  "id": 191,
  "method": "EpsSet",
  "system": {
    "ver": "1.0",
    "lang": "en",
    "userid": "1111111",
    "appkey": "APPKEY_XXXXXXXX",
    "time": 1447640772,
    "sign": "SIGN_XXXXXXXX"
  },
  "params": {
    "args": "[{\"val\":65535,\"tag\":\"m\",\"agt\":\"_-
EAAABuADoYRzUyOTc2Mg\",\"me\":\"0011\",\"idx\":\"RGBW\",\"type\":255
},{\"val\":0,\"tag\":\"m\",\"agt\":\"_-
EAAABuADoYRzUyOTc2Mg\",\"me\":\"0011\",\"idx\":\"DYN\",\"type\":128}
]",
  }
}
```

- 签名原始字符串：

```
method:EpsSet,args:[{"val":65535,"tag":"m","agt":"_-
EAAABuADoYRzUyOTc2Mg","me":"0011","idx":"RGBW","type":255},{"val":0,
"tag":"m","agt":"_-
EAAABuADoYRzUyOTc2Mg","me":"0011","idx":"DYN","type":128}],time:1447
640772,userid:1111111,usertoken:USERTOKEN_XXXXXXXX,appkey:APPKEY_XXX
XXXXX,apptoken:APPTOKEN_XXXXXXXX
```

- 回复信息：

```
{
  "id": 191,
  "code": 0,
  "message": "success"
}
```

4.5.10.SceneGet 获取场景

4.5.10.1.JSON请求数据格式

Type	Definition		Must	Description
Interface Name	SceneGet			获取场景
Partial URL	api.SceneGet			
Content Type	application/json			
HTTP Method	POST			
Request Content	system	ver	Y	1.0
		lang	Y	en
		sign	Y	签名值
		userid	Y	User ID
		appkey	Y	appkey
		did	O	(可选)终端唯一id。如果在授权时填入了，此处必须填入相同id
		time	Y	UTC时间戳, 自1970年1月1日起 计算的时间, 单位为秒
	method		Y	SceneGet
	params	agt	Y	欲查询设备的agt
	id		Y	消息id号

4.5.10.2.范例

- 我们假定：
  appkey为APPKEY_XXXXXXXX，实际需要填写真实数据；
  apptoken为APPTOKEN_XXXXXXXX，实际需要填写真实数据；
  usertoken为USERTOKEN_XXXXXXXX，实际需要填写真实数据；
  sign为SIGN_XXXXXXXX，实际需要填写真实签名数据；
  agt为AGT_XXXXXXXX，实际需要填写真实数据；

- 请求地址：
  svrurl+PartialURL，例如：

```
https://api.ilifesmart.com/app/api.SceneGet
```

- 请求信息：

```
{
  "id": 974,
  "method": "SceneGet",
  "system": {
    "ver": "1.0",
    "lang": "en",
    "userid": "1111111",
    "appkey": "APPKEY_XXXXXXXX",
    "time": 1447639497,
    "sign": "SIGN_XXXXXXXX"
  },
  "params": {
    "agt": "AGT_XXXXXXXX",
 }
}
```

- 签名原始字符串：

```
method:SceneGet,agt:AGT_XXXXXXXX,time:1447639497,userid:1111111,user
token:USERTOKEN_XXXXXXXX,appkey:APPKEY_XXXXXXXX,apptoken:APPTOKEN_XX
XXXXXX
```

- 回复信息：

```
{
  "id": 974,
  "code": 0,
  "message": [
    {
      "id": "aaaaaaaa",
      "name":"testscene",
      "desc":"testscenessss",
      "cls":"scene",
    },
    {
      "id": "bbbbbbb",
      "name":"testscene1",
      "desc":"testscene2",
      "cls":"scene",
    }
  ]
}
```

4.5.11.SceneSet 触发场景

4.5.11.1.JSON请求数据格式

Type	Definition		Must	Description
Interface Name	SceneSet			触发场景
Partial URL	api.SceneSet			
Content Type	application/json			
HTTP Method	POST			
Request Content	system	ver	Y	1.0
		lang	Y	en
		sign	Y	签名值
		userid	Y	User ID
		appkey	Y	appkey
		did	O	(可选)终端唯一id。如果在授权时填入了，此处必须填入相同id
		time	Y	UTC时间戳，自1970年1月1日起 计算的时间, 单位为秒
	method		Y	SceneSet
	params	agt	Y	欲查询设备的agt
		id	Y	欲触发场景的id
		type	O	场景需要设置的参数名字
		RGBW	O	场景需要设置的参数名字
	id		Y	消息id号

4.5.11.2.范例

- 我们假定:
  appkey为APPKEY_XXXXXXXX，实际需要填写真实数据；
  apptoken为APPTOKEN_XXXXXXXX，实际需要填写真实数据；
  usertoken为USERTOKEN_XXXXXXXX，实际需要填写真实数据；
  sign为SIGN_XXXXXXXX，实际需要填写真实签名数据；
  agt为AGT_XXXXXXXX，实际需要填写真实数据；

- 请求地址：
  svrurl+PartialURL，例如：

```
https://api.ilifesmart.com/app/api.SceneSet
```

- 请求信息：

```
{
  "id": 974,
  "method": "SceneSet",
  "system": {
    "ver": "1.0",
    "lang": "en",
    "userid": "1111111",
    "appkey": "APPKEY_XXXXXXXX",
    "time": 1447639497,
    "sign": "SIGN_XXXXXXXX"
  },
  "params": {
    "agt": "AGT_XXXXXXXX",
    "id": "aaaaaaaa",
  }
}
```

- 签名原始字符串：

```
method:SceneGet,agt:AGT_XXXXXXXX,id:aaaaaaaa,time:1447639497,userid:
1111111,usertoken:USERTOKEN_XXXXXXXX,appkey:APPKEY_XXXXXXXX,apptoken
:APPTOKEN_XXXXXXXX
```

- 回复信息：

```
{
  "id": 974,
  "code": 0,
  "message":"success"
}
```

4.5.11.3.说明

场景的cls主要有如下几种，同时执行场景时需要传入相应的参数：

cls	name	args	value	value描述	desc
scene	情景模式	N/A	N/A	N/A	
groupsw	一键开关	args	type	0x81:打开 0x80:关闭	打开或者关闭一组开关/灯泡
grouphw	极速彩灯组	args	type	0x81:打开 0x80:关闭 0xff:设置颜色并开灯 0xfe:设置颜色并关灯	设置一组灯泡。此种模式下以广播方式打开/关闭，响应速度更快
			RGBW	4byte:WRGB	
grouprgbw	彩灯组	args	type	0x81:打开 0x80:关闭 0xff:设置颜色并开灯 0xfe:设置颜色并关灯	打开或者关闭一组开关/灯泡。
			RGBW	4byte:WRGB	

4.5.12.EpUpgradeAgt 升级智慧中心

4.5.12.1.JSON请求数据格式

Type	Definition		Must	Description
Interface Name	EpUpgradeAgt			升级智慧中心
Partial URL	api.EpUpgradeAgt			
Content Type	application/json			
HTTP Method	POST			
Request Content	system	ver	Y	1.0
		lang	Y	en
		sign	Y	签名值
		userid	Y	User ID
		appkey	Y	appkey
		did	O	(可选)终端唯一id。如果在授权时填入了，此处必须填入相同id
		time	Y	UTC时间戳，自1970年1月1日起 计算的时间，单位为秒
	method		Y	EpUpgradeAgt
	params	agt	Y	欲操作的智慧中心ID
		httpUrl	O	采用HTTP方式升级提供可供下载的固件文件的URL，注意：固件文件名称必须以 **.tar.gz** 结尾。
		httpCertificate	O	采用HTTP方式升级需要的安全摘要证书内容，如果采用HTTP方式升级，则该证书必须提供。注意：证书内容需要使用标准Base64编码。
		reboot	O	是否在升级完成之后自动重启智慧中心 1表示需要，0表示不需要，缺省为1
		nonResp	O	是否不需要等待智慧中心执行完成，立即返回 1表示需要，0表示不需要，缺省为1
	id		Y	消息id号

4.5.12.2.范例

- 我们假定：
  appkey为APPKEY_XXXXXXXX，实际需要填写真实数据；
  apptoken为APPTOKEN_XXXXXXXX，实际需要填写真实数据；
  usertoken为USERTOKEN_XXXXXXXX，实际需要填写真实数据；
  sign为SIGN_XXXXXXXX，实际需要填写真实签名数据；
  agt为AGT_XXXXXXXX，实际需要填写真实数据；

- 请求地址：
  svrurl+PartialURL，例如：

```
https://api.ilifesmart.com/app/api.EpUpgradeAgt
```

- 请求信息：

```
{
  "id": 974,
  "method": "EpUpgradeAgt",
  "system": {
    "ver": "1.0",
    "lang": "en",
    "userid": "1111111",
    "appkey": "APPKEY_XXXXXXXX",
    "time": 1447639497,
    "sign": "SIGN_XXXXXXXX"
  },
  "params": {
    "agt": "AGT_XXXXXXXX",
    "reboot": "1",
  }
}
```

- 签名原始字符串：

```
method:EpUpgradeAgt,agt:AGT_XXXXXXXX,reboot:1,time:1447639497,userid
:1111111,usertoken:USERTOKEN_XXXXXXXX,appkey:APPKEY_XXXXXXXX,apptoke
n:APPTOKEN_XXXXXXXX
```

- 回复信息：

```
{
  "id": 974,
  "code": 0,
  "message":"success"
}
```

注意：
由于升级操作是比较耗时的操作，跟网络原因也有关系，因此网络环境恶劣其时间可能会需要几分钟，因此该命令缺省(nonResp=1)将不会等待设备升级完成才返回，而是立即返回。因此需要调用者间隔一段时间调用查询命令例如EpGetAllAgts去查询获取智慧中心的版本号，判断是否已经升级到最新版本。

如果nonResp=1并且reboot=0，则该命令将立即返回，但又不会自动重启，因此将无法掌握智慧中心的升级状态，因此最好不要采用这种方式。

如果nonResp=0则表示需要等待智慧中心升级结果，则reboot最好不要等于1，因为智慧中心升级完成之后自动重启，将可能导致回应包无法发送出去。并且在nonResp=0的方式下，由于耗时较长，需要设置HTTP的请求等待时间为较长的时间，例如300s，否则HTTP请求可能会提早返回超时。

因此，我们推荐的方式有：

nonResp=1, reboot=1: 不关注升级结果，命令下下去后，立即返回，缺省方式。

nonResp=0, reboot=0: 关注升级结果，升级完成之后，需手工调用EpRebootAgt执行重启命令，并且需要设置HTTP请求等待时间为较长的时间。

智慧中心的当前版本号请参考 **EpGetAllAgts** API调用返回的智慧中心 **agt_ver** 属性。

4.5.13.EpRebootAgt 重启智慧中心

4.5.13.1.JSON请求数据格式

Type	Definition		Must	Description
Interface Name	EpRebootAgt			重启智慧中心
Partial URL	api.EpRebootAgt			
Content Type	application/json			
HTTP Method	POST			
Request Content	system	ver	Y	1.0
		lang	Y	en
		sign	Y	签名值
		userid	Y	User ID
		appkey	Y	appkey
		did	O	(可选)终端唯一id。如果在授权时填入了，此处必须填入相同id
		time	Y	UTC时间戳，自1970年1月1日起 计算的时间，单位为秒
	method		Y	EpRebootAgt
	params	agt	Y	欲操作的智慧中心ID
	id		Y	消息id号

4.5.13.2.范例

- 我们假定：
  appkey为APPKEY_XXXXXXXX，实际需要填写真实数据；
  apptoken为APPTOKEN_XXXXXXXX，实际需要填写真实数据；
  usertoken为USERTOKEN_XXXXXXXX，实际需要填写真实数据；
  sign为SIGN_XXXXXXXX，实际需要填写真实签名数据；
  agt为AGT_XXXXXXXX，实际需要填写真实数据；

- 请求地址：
  svrurl+PartialURL，例如：

```
https://api.ilifesmart.com/app/api.EpRebootAgt
```

- 请求信息：

```
{
  "id": 974,
  "method": "EpRebootAgt",
  "system": {
    "ver": "1.0",
    "lang": "en",
    "userid": "1111111",
    "appkey": "APPKEY_XXXXXXXX",
    "time": 1447639497,
    "sign": "SIGN_XXXXXXXX"
  },
  "params": {
    "agt": "AGT_XXXXXXXX"
  }
}
```

- 签名原始字符串：

```
method:EpRebootAgt,agt:AGT_XXXXXXXX,time:1447639497,userid:1111111,u
sertoken:USERTOKEN_XXXXXXXX,appkey:APPKEY_XXXXXXXX,apptoken:APPTOKEN
_XXXXXXXX
```

- 回复信息：

```
{
  "id": 974,
  "code": 0,
  "message":"success"
}
```

注意：

智慧中心升级之后会自动重启，因此执行EpUpgradeAgt操作，不需要再次调用重启智慧中心操作。

4.5.14.EpGetAgtLatestVersion 获取智慧中心最新版本

4.5.14.1.JSON请求数据格式

Type	Definition		Must	Description
Interface Name	EpGetAgtLatestVersion			获取智慧中心最新版本
Partial URL	api.EpGetAgtLatestVersion			
Content Type	application/json			
HTTP Method	POST			
Request Content	system	ver	Y	1.0
		lang	Y	en
		sign	Y	签名值
		userid	Y	User ID
		appkey	Y	appkey
		did	O	(可选)终端唯一id。如果在授权时填入了，此处必须填入相同id
		time	Y	UTC时间戳，自1970年1月1日起 计算的时间，单位为秒
	method		Y	EpGetAgtLatestVersion
	params	agt	Y	欲查询的智慧中心ID
	id		Y	消息id号

4.5.14.2.范例

- 我们假定：
  appkey为APPKEY_XXXXXXXX，实际需要填写真实数据；
  apptoken为APPTOKEN_XXXXXXXX，实际需要填写真实数据；
  usertoken为USERTOKEN_XXXXXXXX，实际需要填写真实数据；
  sign为SIGN_XXXXXXXX，实际需要填写真实签名数据；
  agt为AGT_XXXXXXXX，实际需要填写真实数据；

- 请求地址：
  svrurl+PartialURL，例如：

```
https://api.ilifesmart.com/app/api.EpGetAgtLatestVersion
```

- 请求信息：

```
{
  "id": 974,
  "method": "EpGetAgtLatestVersion",
  "system": {
    "ver": "1.0",
    "lang": "en",
    "userid": "1111111",
    "appkey": "APPKEY_XXXXXXXX",
    "time": 1447639497,
    "sign": "SIGN_XXXXXXXX"
  },
  "params": {
    "agt": "AGT_XXXXXXXX"
  }
}
```

- 签名原始字符串：

```
method:EpGetAgtLatestVersion,agt:AGT_XXXXXXXX,time:1447639497,userid
:1111111,usertoken:USERTOKEN_XXXXXXXX,appkey:APPKEY_XXXXXXXX,apptoke
n:APPTOKEN_XXXXXXXX
```

- 回复信息：

```
{
  "id": 974,
  "code": 0,
  "message":{
    "newestVersion": "NEWEST_VERSION",
    "stableVersion":"STABLE_VERSION",
  }
}
```

返回参数说明：

***** `newestVersion` 当前最新版本。当前发布的最新版本，不强制用户升级，用户自由选择，可以选择升级，也可以选择忽略，一般体验最新的功能或小的Bug修复会更新最新版本。

***** `stableVersion` 当前稳定版本。若当前智慧中心版本号低于稳定版本，则必须提醒用户升级智慧中心版本号，一般有大的功能更新或大的Bug修复将会更新稳定版本。

智慧中心的当前版本号请参考 **EpGetAllAgts** API调用返回的智慧中心 **agt_ver** 属性。

4.5.15.EpSearchSmart 获取智慧中心搜索到的附近智慧设备

4.5.15.1.JSON请求数据格式

Type	Definition		Must	Description
Interface Name	EpSearchSmart			获取智慧中心搜索到的附近其它智慧设备 当需要把其它Wi-Fi类智慧设备，例如摄像头、超级碗加入到智慧中心下面，需要先执行这个接口，再调用EpAddSmart接口把搜索到的Wi-Fi智慧设备添加到智慧中心下面。
Partial URL	api.EpSearchSmart			
Content Type	application/json			
HTTP Method	POST			
Request Content	system	ver	Y	1.0
		lang	Y	en
		sign	Y	签名值
		userid	Y	User ID
		appkey	Y	appkey
		did	O	(可选)终端唯一id。如果在授权时填入了，此处必须填入相同id
		time	Y	UTC时间戳，自1970年1月1日起 计算的时间，单位为秒
	method		Y	EpSearchSmart
	params	agt	Y	欲查询的智慧中心ID
		mode	Y	查询模式，可以填写："notexist"/"auto" notexist表示仅返回搜索到的还未添加到智慧中心下面的附近智慧设备； auto表示返回所有搜索到的附近智慧设备；
	id		Y	消息id号

4.5.15.2.范例

- 我们假定：

appkey为APPKEY_XXXXXXXX，实际需要填写真实数据；
apptoken为APPTOKEN_XXXXXXXX，实际需要填写真实数据；
usertoken为USERTOKEN_XXXXXXXX，实际需要填写真实数据；
sign为SIGN_XXXXXXXX，实际需要填写真实签名数据；
agt为AGT_XXXXXXXX，实际需要填写真实数据；

- 请求地址：
  svrurl+PartialURL，例如：

```
https://api.ilifesmart.com/app/api.EpSearchSmart
```

- 请求信息：

```
{
  "id": 974,
  "method": "EpSearchSmart",
  "system": {
    "ver": "1.0",
    "lang": "en",
    "userid": "1111111",
    "appkey": "APPKEY_XXXXXXXX",
    "time": 1447639497,
    "sign": "SIGN_XXXXXXXX"
  },
  "params": {
    "agt": "AGT_XXXXXXXX",
    "mode": "notexist"
  }
}
```

- 签名原始字符串：

```
method:EpSearchSmart,agt:AGT_XXXXXXXX,mode:notexist,time:1447639497,
userid:1111111,usertoken:USERTOKEN_XXXXXXXX,appkey:APPKEY_XXXXXXXX,a
pptoken:APPTOKEN_XXXXXXXX
```

- 回复信息：

```
{
  "id": 974,
  "code": 0,
  "message":[{
      "lsid": "_wQBANxPIiTZEAAAAAAAAA",
      "name": "SPOT",
      "ip": "192.168.1.56",
      "ttl": 1,
      "sn": "dc:4f:22:24:d9:10"
    },{
      "lsid": "A9cAAEJDMzQwMDJGQTMzOA",
      "name": "Camera",
      "ip": "192.168.1.224",
      "ttl": 1
    }
  ]
}
```

返回参数说明：

* lsid 被搜索到的智慧设备的UUID；
* name 被搜索到的智慧设置的名称；
* ttl 搜索过程的TTL条数，可用户诊断网络；
* sn 被搜索到的智慧设置的MAC地址，并非所有的都会返回；

4.5.16.EpAddSmart 把搜索到的附近智慧设备添加到智慧中心

4.5.16.1.JSON请求数据格式

Type	Definition		Must	Description
Interface Name	EpAddSmart			把搜索到的附近智慧设备添加到智慧中心，需要先执行EpSearchSmart操作获取搜索到的附近智慧设备列表，才能执行添加操作。具体请参考EpSearchSmart接口说明。
Partial URL	api.EpAddSmart			
Content Type	application/json			
HTTP Method	POST			
Request Content	system	ver	Y	1.0
		lang	Y	en
		sign	Y	签名值
		userid	Y	User ID
		appkey	Y	appkey
		did	O	(可选)终端唯一id。如果在授权时填入了，此处必须填入相同id
		time	Y	UTC时间戳，自1970年1月1日起 计算的时间，单位为秒
	method		Y	EpAddSmart
	params	agt	Y	智慧中心ID
		lsid	Y	欲添加的智慧设备的LifeSmart UUID，从EpSearchSmart接口返回的智慧设备信息中获取。
		ip	Y	欲添加的智慧设备的IP，从EpSearchSmart接口返回的智慧设备信息中获取。
		name	Y	欲添加的智慧设备的名称，可以自行命名，并非一定要等于EpSearchSmart接口返回智慧设备的名称。
	id		Y	消息id号

4.5.16.2.范例

- 我们假定：
  appkey为APPKEY_XXXXXXXX，实际需要填写真实数据；
  apptoken为APPTOKEN_XXXXXXXX，实际需要填写真实数据；
  usertoken为USERTOKEN_XXXXXXXX，实际需要填写真实数据；
  sign为SIGN_XXXXXXXX，实际需要填写真实签名数据；
  agt为AGT_XXXXXXXX，实际需要填写真实数据；

- 请求地址：
  svrurl+PartialURL，例如：

```
https://api.ilifesmart.com/app/api.EpAddSmart
```

- 请求信息：

```
{
  "id": 974,
  "method": "EpAddSmart",
  "system": {
    "ver": "1.0",
    "lang": "en",
    "userid": "1111111",
    "appkey": "APPKEY_XXXXXXXX",
    "time": 1447639497,
    "sign": "SIGN_XXXXXXXX"
  },
  "params": {
    "agt": "AGT_XXXXXXXX",
    "lsid": "A9IAAEJDMzQwMDJGMUY3QQ",
    "ip": "192.168.1.145",
    "name": "CameraByOpenApi",
  }
}
```

- 签名原始字符串：

```
method:EpAddSmart,agt:AGT_XXXXXXXX,ip:192.168.1.145,lsid:A9IAAEJDMzQ
wMDJGMUY3QQ,name:CameraByOpenApi,time:1447639497,userid:1111111,user
token:USERTOKEN_XXXXXXXX,appkey:APPKEY_XXXXXXXX,apptoken:APPTOKEN_XX
XXXXXX
```

- 回复信息：

```
{
  "id": 974,
  "code": 0,
  "message":"2d3a"
}
```

返回参数说明：
若执行成功，message属性标识新添加的智慧设备在智慧中心下的 "me" 属性值。
若该设备已经存在与智慧中心下面，即已经被添加过，则message仍将返回该设备在智慧中心下的 "me" 属性值。

4.5.17.EpGetAgtState 获取智慧中心状态

4.5.17.1.JSON请求数据格式

Type	Definition		Must	Description
Interface Name	EpGetAgtState			获取智慧中心状态
Partial URL	api.EpGetAgtState			
Content Type	application/json			
HTTP Method	POST			
Request Content	system	ver	Y	1.0
		lang	Y	en
		sign	Y	签名值
		userid	Y	User ID
		appkey	Y	appkey
		did	O	(可选)终端唯一id。如果在授权时填入了，此处必须填入相同id
		time	Y	UTC时间戳，自1970年1月1日起 计算的时间，单位为秒
	method		Y	EpGetAgtState
	params	agt	Y	智慧设备ID
	id		Y	消息id号

4.5.17.2.范例

- 我们假定：
  appkey为APPKEY_XXXXXXXX，实际需要填写真实数据；
  apptoken为APPTOKEN_XXXXXXXX，实际需要填写真实数据；
  usertoken为USERTOKEN_XXXXXXXX，实际需要填写真实数据；
  sign为SIGN_XXXXXXXX，实际需要填写真实签名数据；
  agt为AGT_XXXXXXXX，实际需要填写真实数据；

- 请求地址：
  svrurl+PartialURL，例如：

```
https://api.ilifesmart.com/app/api.EpGetAgtState
```

- 请求信息：

```
{
  "id": 974,
  "method": "EpGetAgtState",
  "system": {
    "ver": "1.0",
    "lang": "en",
    "userid": "1111111",
    "appkey": "APPKEY_XXXXXXXX",
    "time": 1447639497,
    "sign": "SIGN_XXXXXXXX"
  },
  "params": {
    "agt": "AGT_XXXXXXXX"
  }
}
```

- 签名原始字符串：

```
method:EpGetAgtState,agt:AGT_XXXXXXXX,time:1447639497,userid:1111111
,usertoken:USERTOKEN_XXXXXXXX,appkey:APPKEY_XXXXXXXX,apptoken:APPTOK
EN_XXXXXXXX
```

- 回复信息：

```
{
  "id": 974,
  "code": 0,
  "message": {
    "state": 2,
    "agt_ver": "1.0.68p8",
    "name": "智慧中心MINIV2"
  }
}
```

返回参数说明：

智慧设备指拥有独立工作能力的设备，例如智慧中心，MINI智慧中心，以及Wi-Fi类独立工作设备，例如超级碗SPOT，摄像头等。

智慧设备有三种状态，其值描述如下：

- 0：离线(offline)；
- 1：正在初始化(initializing)；
- 2：工作正常(normal)；

正在初始化一般发生在添加智慧设置的过程中，智慧设备已经被添加到账号中，但还没有完成初始化。

4.5.18.EpCmd 控制设备(高级命令)

4.5.18.1.JSON请求数据格式

Type	Definition		Must	Description
Interface Name	EpCmd			控制设备(高级命令)
Partial URL	api.EpCmd			
Content Type	application/json			
HTTP Method	POST			
Request Content	system	ver	Y	1.0
		lang	Y	en
		sign	Y	签名值
		userid	Y	User ID
		appkey	Y	appkey
		did	O	(可选)终端唯一id。如果在授权时填入了，此处必须填入相同id
		time	Y	UTC时间戳，自1970年1月1日起 计算的时间，单位为秒
	method		Y	EpCmd
	params	agt	Y	智慧中心ID
		me	Y	设备ID
		cmd	Y	命令
		cmdargs	O	命令参数，为JSON格式对象的序列化字符串
	id		Y	消息id号

4.5.18.2.范例

- **我们假定：**
  appkey为APPKEY_XXXXXXXX，实际需要填写真实数据；
  apptoken为APPTOKEN_XXXXXXXX，实际需要填写真实数据；
  usertoken为USERTOKEN_XXXXXXXX，实际需要填写真实数据；
  sign为SIGN_XXXXXXXX，实际需要填写真实签名数据；

agt为AGT_XXXXXXXX，实际需要填写真实数据；

- **请求地址：**
  svrurl+PartialURL，例如：

```
https://api.ilifesmart.com/app/api.EpCmd
```

- **请求信息：**

```
{
  "id": 974,
  "method": "EpCmd",
  "system": {
    "ver": "1.0",
    "lang": "en",
    "userid": "1111111",
    "appkey": "APPKEY_XXXXXXXX",
    "time": 1447639497,
    "sign": "SIGN_XXXXXXXX"
  },
  "params": {
    "agt": "AGT_XXXXXXXX",
    "me": "2e97",
    "cmd": "audio",
    "cmdargs": "{\"adcmd\": \"play\", \"id\": 5, \"opt\": {\"vol\":
95, \"loop\": 2, \"clear\": true}}"
  }
}
```

- **签名原始字符串：**

```
method:EpCmd,agt:AGT_XXXXXXXX,cmd:audio,cmdargs:{"adcmd": "play",
"id": 5, "opt": {"vol": 95, "loop": 2, "clear":
true}},me:2e97,time:1447639497,userid:1111111,usertoken:USERTOKEN_XX
XXXXXX,appkey:APPKEY_XXXXXXXX,apptoken:APPTOKEN_XXXXXXXX
```

- **回复信息：**

```
{
  "id": 974,
  "code": 0,
  "message": "success"
}
```

EpSet命令一般用于控制设备的属性，属性可以是一个或者多个，但缺乏灵活，对于一些复杂的或者特殊的指令可以使用EpCmd，它一般用于复合指令，作用于一个设备，并且设备端还可以对某些指令进行特殊处理，因此功能要比EpSet命令强大。当前EpCmd指令有：

- 报警器播放语音指令；
- 云视·户外摄像头声光警报；

具体的指令如下：

指令Directive	设备	Description

{
cmd:"audio",
cmdargs:{
adcmd: "play",
id: 5,
opt: {
vol: 95,
loop: 2,
clear: True,
led: True,
mute: False
}
}
 	
{
cmd:"audio",
cmdargs:{
adcmd: "stop",
}
	智慧中心(MINI)设备
	
	云视·户外摄像头
		用于控制智慧中心(MINI)设备的报警音播放。
		cmd参数请务必填写 "audio";
		cmdargs参数是JOSN对象的序列化字符串。其内容如下:
		• adcmd指明动作类型，有如下值:
		"play": 开启声音播放
		"stop": 停止声音播放
		若为停止声音播放，则不需要其它参数。
		
		• id指明播放的声音序号，当前一共有7个声音序号，其值取值为:[1,7]。
		
		• opt是操作选项，对于play动作类型来说，有如下属性:
		vol: 指明播放的音量，其取值范围为:[50,100],值越大声音越响。
		loop: 指明播放的次数。
		clear: 指明是否清除正在播放的声音，开启本次新的声音播放，若值为false则原先正在播放的声音会与新的声音一起播放。
		led: 是否开启户外摄像头LED闪烁警报。
		mute: 是否静音，一般与led一起使用，若想只开启LED闪烁而没有声音警报，则可以设置该参数为True。
		
		注: led与mute参数是云视·户外摄像头才有的属性。

4.5.19.EpSetVar 控制设备(低级命令)

4.5.19.1.JSON请求数据格式

Type	Definition		Must	Description
Interface Name	EpSetVar			控制设备(低级命令)
Partial URL	api.EpSetVar			
Content Type	application/json			
HTTP Method	POST			
Request Content	system	ver	Y	1.0
		lang	Y	en
		sign	Y	签名值
		userid	Y	User ID
		appkey	Y	appkey
		did	O	(可选)终端唯一id。如果在授权时填入了，此处必须填入相同id
		time	Y	UTC时间戳, 自1970年1月1日起 计算的时间, 单位为秒
	method		Y	EpSetVar
	params	agt	Y	智慧中心ID
		me	Y	设备ID
		idx	Y	索引号 Number类型
		cmd	Y	命令号 Number类型
		cmddata	Y	命令数据，为JSON数组的序列化字符串，JSON数组成员值为Byte类型
	id		Y	消息id号

4.5.19.2.范例

- **我们假定：**
  appkey为APPKEY_XXXXXXXX，实际需要填写真实数据；

apptoken为APPTOKEN_XXXXXXXX，实际需要填写真实数据；
usertoken为USERTOKEN_XXXXXXXX，实际需要填写真实数据；
sign为SIGN_XXXXXXXX，实际需要填写真实签名数据；
agt为AGT_XXXXXXXX，实际需要填写真实数据；

- **请求地址：**
  svrurl+PartialURL，例如：

```
https://api.ilifesmart.com/app/api.EpSetVar
```

- **请求信息：**

```
{
  "id": 974,
  "method": "EpSetVar",
  "system": {
    "ver": "1.0",
    "lang": "en",
    "userid": "1111111",
    "appkey": "APPKEY_XXXXXXXX",
    "time": 1447639497,
    "sign": "SIGN_XXXXXXXX"
  },
  "params": {
    "agt": "AGT_XXXXXXXX",
    "me": "2e97",
    "idx": 129,
    "cmd": 0,
    "cmddata": "[0, 0, 0]"
  }
}
```

- **签名原始字符串：**

```
method:EpSetVar,agt:AGT_XXXXXXXX,cmd:0,cmddata:[0, 0,
0],idx:129,me:2e97,time:1447639497,userid:1111111,usertoken:USERTOKE
N_XXXXXXXX,appkey:APPKEY_XXXXXXXX,apptoken:APPTOKEN_XXXXXXXX
```

- **回复信息：**

```
{
  "id": 974,
  "code": 0,
  "message": "success"
}
```

`EpSetVar`命令是低级命令，可以对设备完成一些比较低级别的设置，请严格参考文档描述的指令进行设置，否则设置将不会成功。

当前`EpSetVar`指令有：

指令Directive	设备	Description

{
idx: 129,
cmd: 0,
cmddata: [0, 0, 0]
}
	计量插座	用于清除计量插座的累计用电量。
		
		注意：清零之后直接去查询累计电量仍然存在，不会马上生效，需要等待下一次变化上报周期到来才会生效，一般需要等待几分钟。

注：
指令Directive只列举idx、cmd、cmddata数据值，agt、me参数请填写具体设备的智慧中心Id与设备Id。

4.5.20.EpGetAttrs 获取设备扩展属性

4.5.20.1.JSON请求数据格式

Type	Definition		Must	Description
Interface Name	EpGetAttrs			获取设备扩展属性
Partial URL	api.EpGetAttrs			
Content Type	application/json			
HTTP Method	POST			
Request Content	system	ver	Y	1.0
		lang	Y	en
		sign	Y	签名值
		userid	Y	User ID
		appkey	Y	appkey
		did	O	(可选)终端唯一id。如果在授权时填入了，此处必须填入相同id
		time	Y	UTC时间戳，自1970年1月1日起 计算的时间，单位为秒
	method		Y	EpGetAttrs
	params	agt	Y	智慧中心ID
		me	Y	设备ID
		attrNames	Y	属性名称数组，为JSON数组的序列化字符串
	id		Y	消息id号

4.5.20.2.范例

- **我们假定：**
  appkey为APPKEY_XXXXXXXX，实际需要填写真实数据；
  apptoken为APPTOKEN_XXXXXXXX，实际需要填写真实数据；
  usertoken为USERTOKEN_XXXXXXXX，实际需要填写真实数据；
  sign为SIGN_XXXXXXXX，实际需要填写真实签名数据；
  agt为AGT_XXXXXXXX，实际需要填写真实数据；

- **请求地址：**
  svrurl+PartialURL，例如：

```
https://api.ilifesmart.com/app/api.EpGetAttrs
```

- **请求信息：**

```
{
  "id": 974,
  "method": "EpGetAttrs",
  "system": {
    "ver": "1.0",
    "lang": "en",
    "userid": "1111111",
    "appkey": "APPKEY_XXXXXXXX",
    "time": 1447639497,
    "sign": "SIGN_XXXXXXXX"
  },
  "params": {
    "agt": "AGT_XXXXXXXX",
    "me": "2e97",
    "attrNames": "[\"PairCfg\"]"
  }
}
```

- **签名原始字符串：**

```
method:EpGetAttrs,agt:AGT_XXXXXXXX,attrNames:["PairCfg"],me:2e97,tim
e:1447639497,userid:1111111,usertoken:USERTOKEN_XXXXXXXX,appkey:APPK
EY_XXXXXXXX,apptoken:APPTOKEN_XXXXXXXX
```

- **回复信息：**

```
{
  "id": 974,
  "code": 0,
  "message": {
    "PairCfg": {
      "warning_duration": 62
    }
  }
}
```

该接口可以获取一些设备的扩展属性，当前支持的扩展属性有：

- **PairCfg**：射频设备对码配置
  在调用EpAdd接口添加对码设备的时候，可以设置设备扩展属性，例如动态感应器的告警持续时长。这个接口可以用于获取设备添加时设置的相应值。

提示：当前仅支持 CUBE动态感应器(SL_SC_BM)、CUBE环境感应器(SL_SC_BE)、恒星/辰星/极星 获取PairCfg值。具体请参考 **添加设备(EpAdd) 接口 optarg 参数介绍** 部分。

4.5.21.EpTestRssi 测试射频设备信号强度

4.5.21.1.JSON请求数据格式

Type	Definition		Must	Description
Interface Name	EpTestRssi			测试射频设备信息强度
Partial URL	api.EpTestRssi			
Content Type	application/json			
HTTP Method	POST			
Request Content	system	ver	Y	1.0
		lang	Y	en
		sign	Y	签名值
		userid	Y	User ID
		appkey	Y	appkey
		did	O	(可选)终端唯一id。如果在授权时填入了，此处必须填入相同id
		time	Y	UTC时间戳，自1970年1月1日起 计算的时间，单位为秒
	method		Y	EpTestRssi
	params	agt	Y	智慧中心ID
		me	N	设备ID，如果设备ID为空，则表示测试智慧中心的射频信号强度
		args	N	扩展属性，为JSON对象的序列化字符串，当前不需要填写
	id		Y	消息id号

该接口获取测试射频设备的信号强度，它仅仅用于现场环境的调试与维护，并非正常使用场景下的接口。

4.5.21.2.范例

- **我们假定：**

appkey为APPKEY_XXXXXXXX，实际需要填写真实数据；
apptoken为APPTOKEN_XXXXXXXX，实际需要填写真实数据；
usertoken为USERTOKEN_XXXXXXXX，实际需要填写真实数据；
sign为SIGN_XXXXXXXX，实际需要填写真实签名数据；
agt为AGT_XXXXXXXX，实际需要填写真实数据；

- **请求地址：**
  svrurl+PartialURL，例如：

```
https://api.ilifesmart.com/app/api.EpTestRssi
```

- **请求信息：**

```
{
  "id": 974,
  "method": "EpTestRssi",
  "system": {
    "ver": "1.0",
    "lang": "en",
    "userid": "1111111",
    "appkey": "APPKEY_XXXXXXXX",
    "time": 1447639497,
    "sign": "SIGN_XXXXXXXX"
  },
  "params": {
    "agt": "AGT_XXXXXXXX",
    "me": "2e97"
  }
}
```

- **签名原始字符串：**

```
method:EpTestRssi,agt:AGT_XXXXXXXX,me:2e97,time:1447639497,userid:11
11111,usertoken:USERTOKEN_XXXXXXXX,appkey:APPKEY_XXXXXXXX,apptoken:A
PPTOKEN_XXXXXXXX
```

- **回复信息：**

```
{
  "id": 974,
  "code": 0,
  "message": {
    "dbm": {
      "back_noise_threshold": -99,
      "back_noise": -114,
      "recv": -19,
      "send": -30
    },
    "rssi": {
      "back_noise_threshold": 70,
      "back_noise": 40,
      "recv": 231,
      "send": 208
    }
  }
}
```

- **返回数据说明：**

  dbm即是分贝毫瓦。其值为负数，越接近-0则表明信号越好。

  - dbm.recv: 接收方向的射频信号强度，智慧中心 ==> 设备
  - dbm.send: 发送方向的射频信号强度，设备 ==> 智慧中心
  - dbm.back_noise: 背景噪音，其值越小(越远离-0)则说明背景噪音越小
  - dbm.back_noise_threshold: 背噪门限，当back_noise的值大于背噪门限，说明现场环境恶劣，已经影响射频设备正常的通信。

  rssi即是接收信号强度指示，其值为正数，越大则表明信号越好。

  - rssi.recv: 接收方向的射频信号强度，智慧中心 ==> 设备
  - rssi.send: 发送方向的射频信号强度，设备 ==> 智慧中心
  - rssi.back_noise: 背景噪音，其值越小则说明背景噪音越小
  - rssi.back_noise_threshold: 背噪门限，当back_noise的值大于背噪门限，说明现场环境恶劣，已经影响射频设备正常的通信。

提示：当执行测试智慧中心信号强度，也即me属性为空的时候，返回的属性为{"back_noise", "back_noise_threshold", "signal" } signal属性指明智慧中心当前的射频信号强度。其值越大说明智慧中心射频信号越好。

提示：是否所有的射频设备都支持检测?
只有支持命令下发的射频设备，例如开关、插座才支持射频信号检测，对于主动上报事件的射频设备，例如温湿度感应器、动态感应器等，由于没有命令下发接口，因此不能执行射频信号检测。查看它们的射频信息可以使用EpGetAll/EpGet命令返回的IHeart、IDbm属性。
EpGetAll/EpGet命令返回的IDbm数组指示设备侧的dbm.send属性，即设备发送方向的射频信号强度。

4.5.22.EpBatchSet 批量快速设置多个设备属性

4.5.22.1.JSON请求数据格式

该接口是一个批量操作接口，用于快速设置多个设备的属性。例如一次开启或关闭多个开关/插座/灯光等场景模式的操作。

该接口会在内部进行优化处理，减少设备处理命令的时间，保证让用户拥有良好的体验。并且该接口支持三种不同速度等级的操作，分别应用于不同的场景需求。

Type	Definition		Must	Description
Interface Name	EpBatchSet			批量快速设置同一个智慧中心下的设备属性
Partial URL	api.EpBatchSet			
Content Type	application/json			
HTTP Method	POST			
Request Content	system	ver	Y	1.0
		lang	Y	en
		sign	Y	签名值
		userid	Y	User ID
		appkey	Y	appkey
		did	O	(可选)终端唯一id。如果在授权时填入了，此处必须填入相同id
		time	Y	UTC时间戳，自1970年1月1日起 计算的时间，单位为秒
	method		Y	EpBatchSet
	params	agt	Y	智慧中心ID

		ioItems	Y	IO属性集合，为JSON数组的序列化字符串。
				每条IO属性如下：
				{
				"me":"2711", ：标识是哪个设备
				"idx":"L1", ：标识设备的IO属性
				"type": 0x81, ：设置的属性类型
				"val": 1 ：设置的属性值
				}

		speed	N	速度等级，当前支持三种，分别为： • 0：正常(normal) • 1：快速(fast) • 2：极速(extreme) 缺省为0。 详见后面描述。
		retry	N	重试次数，缺省为2。 在速度为normal、fast方式下才有效。 在由于射频通道拥塞命令执行超时再次尝试的次数，取值范围为[0,5]。该属性为高级属性，请谨慎使用。

		checkIoStat	N	是否检查IO状态，等于1表示检查IO状态。 在速度为normal、fast方式下才有效。 下发命令执行时是否先检查子设备已有的属性值，若属性值与下发值相等则忽略命令。该属性为高级属性，请谨慎使用。
		uid	N	该次命令的唯一识别码。该参数的作用是用于标识操作，如果短时间内下发两条相同uid属性的操作，则第二条操作指令将直接返回EIGN错误。这样可以防止频繁下发重复指令。
	id		Y	消息id号

4.5.22.2.范例

- **我们假定：**
  appkey为APPKEY_XXXXXXXX，实际需要填写真实数据；
  apptoken为APPTOKEN_XXXXXXXX，实际需要填写真实数据；
  usertoken为USERTOKEN_XXXXXXXX，实际需要填写真实数据；
  sign为SIGN_XXXXXXXX，实际需要填写真实签名数据；
  agt为AGT_XXXXXXXX，实际需要填写真实数据；

- **请求地址：**
  svrurl+PartialURL，例如：

```
https://api.ilifesmart.com/app/api.EpBatchSet
```

- **请求信息：**

```
{
  "id": 974,
  "method": "EpBatchSet",
  "system": {
    "ver": "1.0",
    "lang": "en",
    "userid": "1111111",
    "appkey": "APPKEY_XXXXXXXX",
    "time": 1447639497,
    "sign": "SIGN_XXXXXXXX"
  },
  "params": {
    "agt": "AGT_XXXXXXXX",
    "ioItems": "[{\"me\": \"2f14\", \"idx\": \"L2\", \"type\": 129,
\"val\": 1}, {\"me\": \"2f14\", \"idx\": \"L1\", \"type\": 129,
\"val\": 1}, {\"me\": \"2f14\", \"idx\": \"L3\", \"type\": 129,
\"val\": 1}, {\"me\": \"2f13\", \"idx\": \"L1\", \"type\": 129,
\"val\": 1}, {\"me\": \"2f0f\", \"idx\": \"L2\", \"type\": 129,
\"val\": 1}, {\"me\": \"2f0f\", \"idx\": \"L1\", \"type\": 129,
\"val\": 1}, {\"me\": \"2f0f\", \"idx\": \"L3\", \"type\": 129,
```

```
\"val\": 1}, {\"me\": \"2f50\", \"idx\": \"L2\", \"type\": 129,
\"val\": 1}, {\"me\": \"2f50\", \"idx\": \"L1\", \"type\": 129,
\"val\": 1}, {\"me\": \"2f10\", \"idx\": \"L2\", \"type\": 129,
\"val\": 1}, {\"me\": \"2f10\", \"idx\": \"L1\", \"type\": 129,
\"val\": 1}, {\"me\": \"2f72\", \"idx\": \"L1\", \"type\": 129,
\"val\": 1}, {\"me\": \"2f71\", \"idx\": \"L2\", \"type\": 129,
\"val\": 1}, {\"me\": \"2f71\", \"idx\": \"L1\", \"type\": 129,
\"val\": 1}, {\"me\": \"8141\", \"idx\": \"L\", \"type\": 129,
\"val\": 1}]",
    "speed": 1,
    "uid": "TEST"
  }
}
```

- **签名原始字符串**:

```
method:EpBatchSet,agt:AGT_XXXXXXXX,ioItems:[{"me": "2f14", "idx":
"L2", "type": 129, "val": 1}, {"me": "2f14", "idx": "L1", "type":
129, "val": 1}, {"me": "2f14", "idx": "L3", "type": 129, "val": 1},
{"me": "2f13", "idx": "L1", "type": 129, "val": 1}, {"me": "2f0f",
"idx": "L2", "type": 129, "val": 1}, {"me": "2f0f", "idx": "L1",
"type": 129, "val": 1}, {"me": "2f0f", "idx": "L3", "type": 129,
"val": 1}, {"me": "2f50", "idx": "L2", "type": 129, "val": 1},
{"me": "2f50", "idx": "L1", "type": 129, "val": 1}, {"me": "2f10",
"idx": "L2", "type": 129, "val": 1}, {"me": "2f10", "idx": "L1",
"type": 129, "val": 1}, {"me": "2f72", "idx": "L1", "type": 129,
"val": 1}, {"me": "2f71", "idx": "L2", "type": 129, "val": 1},
{"me": "2f71", "idx": "L1", "type": 129, "val": 1}, {"me": "8141",
"idx": "L", "type": 129, "val":
1}],speed:1,uid:TEST,time:1550748304,userid:1111111,usertoken:USERTO
KEN_XXXXXXXX,appkey:APPKEY_XXXXXXXX,apptoken:APPTOKEN_XXXXXXXX
```

- **回复信息**:

```
{
  "id": 974,
  "code": 0,
  "message": {
    "failedIoItems": [
      {
         "me": "2f0f",
         "idx": "L1",
         "val": 1,
         "type": 129,
        "ret": 10013,
         "retMsg": "ENR"
      },
      {
         "me": "2f0f",
         "idx": "L3",
         "type": 129,
         "val": 1,
        "ret": 10013,
         "retMsg": "ENR"
      }
    ]
  },
}
```

- **返回属性说明**:
  `message.failedIoItems`属性指明了执行失败的`Items`列表，若所有的`Item`都执行成功，则该属性为空。
  若当前`speed`处于极速`(extreme)`模式，则将不会返回`failedIoItems`属性。

4.5.22.3.speed参数说明

参数值	参数描述
0	正常(normal) 为缺省方式。 按照正常方式执行命令，所有命令操作串行执行，并且需要等待上一条命令的反馈结果之后再继续执行下一条命令。 由于需要等待每一条命令的反馈结果，因此子设备得到执行的时间也会变长，并且用户感官体验上设备执行时间也长。 **特点：【用户体验慢，接口执行时间长，保证准确性】**
1	快速(fast) 先对子设备执行一遍不需要等待反馈结果快速下发，然后再执行一遍需要等待反馈结果的正常下发。 第一遍保证大部分子设备快速得到执行机会，用户感官体验上会很快， 第二遍保证第一次快速执行过程中失败的命令再次得到执行，确保执行结果的准确性。 由于执行了两次下发，整体命令执行总耗时最长，但第一遍快速下发很好的保证了用户体验。 **特点：【用户体验快，接口执行时间最长，保证准确性】**
2	极速(extreme) 直接对子设备执行一遍不需要等待反馈结果的快速下发。 由于仅仅对子设备执行一遍快速下发，因此执行时间将最短。 但在子设备列表比较多的情况下，存在部分子设备执行失败的可能性。但该方式仍然会保证大部分子设备执行是成功的，并且接口调用耗时最短。 **特点：【用户体验快，接口执行时间短，不保证准确性】**

4.5.23.EpSearchIDev 获取智慧中心搜索到的附近IP网络设备

4.5.23.1.JSON请求数据格式

Type	Definition		Must	Description
Interface Name	EpSearchIDev			获取智慧中心搜索到的附近其它IP网络设备 当需要把其它Wi-Fi类网络设备，例如IP摄像头加入到智慧中心下面，需要先执行这个接口，再调用EpAddIDev接口把搜索到的Wi-Fi网络设备添加到智慧中心下面。
Partial URL	api.EpSearchIDev			
Content Type	application/json			
HTTP Method	POST			
Request Content	system	ver	Y	1.0
		lang	Y	en
		sign	Y	签名值
		userid	Y	User ID
		appkey	Y	appkey
		did	O	(可选)终端唯一id。如果在授权时填入了，此处必须填入相同id
		time	Y	UTC时间戳，自1970年1月1日起 计算的时间, 单位为秒
	method		Y	EpSearchIDev
	params	agt	Y	欲查询的智慧中心ID
		mode	Y	查询模式，可以填写："notexist"/"auto" notexist表示仅返回搜索到的还未添加到智慧中心下面的附近网络设备； auto表示返回所有搜索到的附近网络设备；
	id		Y	消息id号

4.5.23.2.范例

- 我们假定：
  appkey为APPKEY_XXXXXXXX，实际需要填写真实数据；

apptoken为APPTOKEN_XXXXXXXX，实际需要填写真实数据；
usertoken为USERTOKEN_XXXXXXXX，实际需要填写真实数据；
sign为SIGN_XXXXXXXX，实际需要填写真实签名数据；
agt为AGT_XXXXXXXX，实际需要填写真实数据；

- 请求地址：
  svrurl+PartialURL，例如：

```
https://api.ilifesmart.com/app/api.EpSearchIDev
```

- 请求信息：

```
{
  "id": 974,
  "method": "EpSearchIDev",
  "system": {
    "ver": "1.0",
    "lang": "en",
    "userid": "1111111",
    "appkey": "APPKEY_XXXXXXXX",
    "time": 1447639497,
    "sign": "SIGN_XXXXXXXX"
  },
  "params": {
    "agt": "AGT_XXXXXXXX",
    "mode": "notexist"
  }
}
```

- 签名原始字符串：

```
method:EpSearchIDev,agt:AGT_XXXXXXXX,mode:notexist,time:1447639497,u
serid:1111111,usertoken:USERTOKEN_XXXXXXXX,appkey:APPKEY_XXXXXXXX,ap
ptoken:APPTOKEN_XXXXXXXX
```

- 回复信息：

```
{
  "id": 974,
  "code": 0,
  "message":[{
     "uuid": "4d756c74-694d-xxxx-xxxx-xxxxxxxxxxxx",
     "devid": "4d756c74-694d-xxxx-xxxx-xxxxxxxxxxxx",
     "devType": "iControl:iCamera2",
     "name": "iCamera2",
     "host": "192.168.1.222",
     "port": 443,
     "ttl": 0,
     "auth": "Basic",
     "defUser": "administrator",
     "defPwd": "",
     "exinfo": "<?xml version=\"1.0\" encoding=\"UTF-
8\" ?><DeviceInfo version=\"1.0\"><deviceName>iCamera2F97A2A</
deviceName><deviceID>xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx</
deviceID><manufacturer>iControl</manufacturer><model>iCamera2</
model><serialNumber>1612AXN000294</
serialNumber><macAddress>b4:a5:ef:f9:7a:2a</
macAddress><firmwareVersion>3.0.01.40</
```

```
    firmwareVersion><firmwareReleasedDate>Jul 21,2016</
    firmwareReleasedDate><bootVersion>1.11</
    bootVersion><bootReleasedDate>Jul 21,2016</
    bootReleasedDate><rescueVersion>3.0.01.40</
    rescueVersion><hardwareVersion>1</hardwareVersion><apiVersion>3.3</
    apiVersion></DeviceInfo>",
         "dhcp": true
      }]
    }
```

返回参数说明:

- uuid 搜索到的网络设备的UUID;
- devType 搜索到的网络设备的类型;
- name 搜索到的网络设备的名称;
- host 搜索到的网络设备的Host地址;
- port 搜索到的网络设备的主机端口号;
- ttl 搜索过程的TTL条数，可用户诊断网络;
- defUser 搜索到的网络设备的缺省登录账号;
- defPwd 搜索到的网络设备的缺省登录账号密码;
- exinfo 搜索到的网络设备的扩展信息;

注意:

1. 并非所有的设备都会包含exinfo信息，这个与设备类型相关;
2. 若智慧中心刚刚启动或者IP设备刚刚加入本地网络中，智慧中心可能还未及时获取到设备的扩展信息，这个时候若获取不到exinfo信息，需要等待片刻再次调用该接口获取搜索信息;
3. 对于 iCamera 设备，若设备的密码已经变更，不再是初始密码，由于没有权限，我们将获取不到设备的扩展信息，这个时候exinfo将为错误信息;
4. 为了保持兼容性，接口并不解析exinfo内容，当前exinfo包含的是查询获取的完整信息，需要使用者自行从中解析出需要关注的信息;
5. 对于不支持扩展信息的设备，exinfo可能为空或者是长度为0的字符串;

4.5.24.EpAddIDev 把搜索到的附近IP网络设备添加到智慧中心

4.5.24.1.JSON请求数据格式

Type	Definition		Must	Description
Interface Name	EpAddIDev			把搜索到的附近IP网络设备添加到智慧中心，需要先执行EpSearchIDev操作获取搜索到的附近网络设备列表，才能执行添加操作。具体请参考EpSearchIDev接口说明。
Partial URL	api.EpAddIDev			
Content Type	application/json			
HTTP Method	POST			
Request Content	system	ver	Y	1.0
		lang	Y	en
		sign	Y	签名值
		userid	Y	User ID
		appkey	Y	appkey
		did	O	(可选)终端唯一id。如果在授权时填入了，此处必须填入相同id
		time	Y	UTC时间戳, 自1970年1月1日起 计算的时间, 单位为秒
	method		Y	EpAddIDev
		agt	Y	智慧中心ID
		uuid	Y	欲添加的网络设备的UUID，从EpSearchIDev接口返回的网络设备信息中获取。
		devType	Y	欲添加的网络设备的设备类型，从EpSearchIDev接口返回的网络设备信息中获取。
		host	Y	欲添加的网络设备的Host，从EpSearchIDev接口返回的网络设备信息中获取。
		port	Y	欲添加的网络设备的Port，从EpSearchIDev接口返回的网络设备信息中获取。
		name	Y	欲添加的网络设备的名称，可以自行命名，并非一定要等于EpSearchIDev接口返回网络设备的名称。

	params	user	O	欲添加的网络设备的登录账号，若用户修改了登录账号，则这里必须填写修改的登录账号，否则智慧中心将不能正确连接网络设备。若用户没有修改登录账号则根据EpSearchIDev接口返回的信息，填写缺省登录账号(defUser)，若搜索接口没有返回缺省登录账号，则可以不填写。
		pwd	O	欲添加的网络设备的登录密码，若用户修改了登录密码，则这里必须填写修改的登录密码，否则智慧中心将不能正确连接网络设备。若用户没有修改登录密码则根据EpSearchIDev接口返回的信息，填写缺省登录密码(defPwd)，若搜索接口没有返回缺省登录密码，则可以不填写。
	id		Y	消息id号

4.5.24.2.范例

- 我们假定：
  appkey为APPKEY_XXXXXXXX，实际需要填写真实数据；
  apptoken为APPTOKEN_XXXXXXXX，实际需要填写真实数据；
  usertoken为USERTOKEN_XXXXXXXX，实际需要填写真实数据；
  sign为SIGN_XXXXXXXX，实际需要填写真实签名数据；
  agt为AGT_XXXXXXXX，实际需要填写真实数据；

- 请求地址：
  svrurl+PartialURL，例如：

```
https://api.ilifesmart.com/app/api.EpAddIDev
```

- 请求信息：

```
{
  "id": 974,
  "method": "EpAddIDev",
  "system": {
    "ver": "1.0",
    "lang": "en",
    "userid": "1111111",
    "appkey": "APPKEY_XXXXXXXX",
    "time": 1447639497,
    "sign": "SIGN_XXXXXXXX"
  },
  "params": {
    "agt": "AGT_XXXXXXXX",
    "uuid": "4d756c74-694d-xxxx-xxxx-xxxxxxxxxxxx",
    "devType": "iControl:iCamera2",
```

```
        "host": "192.168.1.222",
        "port": 443,
        "name": "iCamera2BB",
        "user": "administrator",
        "pwd": ""
      }
    }
```

- 签名原始字符串：

```
method:EpAddIDev,agt:AGT_XXXXXXXX,devType:iControl:iCamera2,host:192
.168.1.222,name:iCamera2BB,port:443,pwd:,user:administrator,uuid:4d7
56c74-694d-xxxx-xxxx-
xxxxxxxxxxxx,time:1447639497,userid:1111111,usertoken:USERTOKEN_XXXX
XXXX,appkey:APPKEY_XXXXXXXX,apptoken:APPTOKEN_XXXXXXXX
```

- 回复信息：

```
{
  "id": 974,
  "code": 0,
  "message":"2d3a"
}
```

返回参数说明：

若执行成功，message属性标识新添加的智慧设备在智慧中心下的 "me" 属性值。
若该设备已经存在与智慧中心下面，即已经被添加过，则message仍将返回该设备在智慧中心下的 "me" 属性值。

4.5.25.EpMaintOtaFiles 查看或维护智慧中心上的OTA文件列表

4.5.25.1.JSON请求数据格式

Type	Definition		Must	Description
Interface Name	EpMaintOtaFiles			查看或维护智慧中心上可供子设备升级固件版本使用的OTA文件列表
Partial URL	api.EpMaintOtaFiles			
Content Type	application/json			
HTTP Method	POST			
	system	ver	Y	1.0
		lang	Y	en
		sign	Y	签名值
		userid	Y	User ID
		appkey	Y	appkey
		did	O	(可选)终端唯一id。如果在授权时填入了，此处必须填入相同id
		time	Y	UTC时间戳，自1970年1月1日起 计算的时间，单位为秒
	method		Y	EpMaintOtaFiles
		agt	Y	智慧中心ID
		act	Y	操作指令，类型为字符串，当前可以为如下： • **AddByUrl**： 告知URL地址，下载OTA文件到智慧中心； • **AddByBin**： 直接提供OTA文件的原始数据给智慧中心； • **Query**： 查询OTA文件信息 • **Remove**： 删除OTA文件

Request Content	params	actargs	O	操作指令参数，数据为JSON对象的序列化字符串。操作指令参数与操作指令一一对应，定义如下： • **AddByUrl**：{key, url}，url参数指明可被下载的OTA文件的地址，url必须为HTTP或HTTPS协议；key为可选参数，若不提供key则key为url提供的下载地址中的文件名称，请谨慎使用key参数，key参数的命名必须正确，否则升级将可能失败，若无必要，无需设置key参数，使用缺省名称即可。 • **AddByBin**：{key, cont}，key指明该OTA文件的标识，不同于AddByUrl，这里key必须提供，不能为空，cont指明OTA文件内容，为方便在JSON中传输，cont为原始OTA文件的标准Base64编码； • **Query**：actargs可以不用提供； • **Remove**：{keys}，keys指明需要移除的OTA文件，若keys=true则表明删除该智慧中心下所有的OTA文件，若keys=["key1", "key2"]则表示批量删除数组指明的OTA文件集，若keys="key1"则表示删除特定的OTA文件；
	id		Y	消息id号

4.5.25.2.范例

- 我们假定：
  appkey为APPKEY_XXXXXXXX，实际需要填写真实数据；
  apptoken为APPTOKEN_XXXXXXXX，实际需要填写真实数据；
  usertoken为USERTOKEN_XXXXXXXX，实际需要填写真实数据；
  sign为SIGN_XXXXXXXX，实际需要填写真实签名数据；
  agt为AGT_XXXXXXXX，实际需要填写真实数据；

- 请求地址：
  svrurl+PartialURL，例如：

```
https://api.ilifesmart.com/app/api.EpMaintOtaFiles
```

- 请求信息：

```
{
  "id": 974,
  "method": "EpMaintOtaFiles",
```

```
      "system": {
        "ver": "1.0",
        "lang": "en",
        "userid": "1111111",
        "appkey": "APPKEY_XXXXXXXX",
        "time": 1447639497,
        "sign": "SIGN_XXXXXXXX"
      },
      "params": {
        "agt": "AGT_XXXXXXXX",
        "act": "AddByUrl",
        "actargs": "{\"url\": \"http://www.ilifesmart.com/upgrade/test/
    FL01_03040d10_00000631.ota\"}"
       }
    }
```

- 签名原始字符串：

```
method:EpMaintOtaFiles,act:AddByUrl,actargs:{"url": "http://
www.ilifesmart.com/upgrade/test/
FL01_03040d10_00000631.ota"},agt:AGT_XXXXXXXX,time:1447639497,userid
:1111111,usertoken:USERTOKEN_XXXXXXXX,appkey:APPKEY_XXXXXXXX,apptoke
n:APPTOKEN_XXXXXXXX
```

- 回复信息：

```
{
  "id": 974,
  "code": 0,
  "message":"success"
}
```

提示：

若是查询命令，则返回的OTA文件列表如下所示：

```
{
  "code": 0,
  "message":
  [
    {
      "key": "FL01_ZG10370104_00000002.ota", // 标识该OTA文件
      "size": 165868 // 指明该OTA文件的大小
    },...
  ]
}
```

4.5.26.EpMaintOtaTasks 查看或维护智慧中心上的OTA任务列表

4.5.26.1.JSON请求数据格式

Type	Definition		Must	Description
Interface Name	EpMaintOtaTasks			查看或维护智慧中心上当前子设备正在执行OTA升级的任务列表
Partial URL	api.EpMaintOtaTasks			
Content Type	application/json			
HTTP Method	POST			
Request Content	system	ver	Y	1.0
		lang	Y	en
		sign	Y	签名值
		userid	Y	User ID
		appkey	Y	appkey
		did	O	(可选)终端唯一id。如果在授权时填入了，此处必须填入相同id
		time	Y	UTC时间戳，自1970年1月1日起 计算的时间，单位为秒
	method		Y	EpMaintOtaTasks
		agt	Y	智慧中心ID
		act	Y	操作指令，类型为字符串，当前可以为如下： • **Query**： 查询OTA任务信息 • **Remove**： 删除已经完成的OTA任务

	params	actargs	O	操作指令参数，数据为JSON对象的序列化字符串。操作指令参数与操作指令一一对应，定义如下： • **Query**： actargs可以不用提供； • **Remove**： {key}，key指明需要移除的OTA任务，若key=true则表明删除该智慧中心下所有的已经完成的OTA任务，若key=["key1", "key2"]则表示批量删除数组指明的已经完成的OTA任务集，若key="key1"则表示删除特定的已经完成的OTA任务；注意：只能删除已经升级完成的OTA任务。
	id		Y	消息id号

4.5.26.2.范例

- 我们假定：
  appkey为APPKEY_XXXXXXXX，实际需要填写真实数据；
  apptoken为APPTOKEN_XXXXXXXX，实际需要填写真实数据；
  usertoken为USERTOKEN_XXXXXXXX，实际需要填写真实数据；
  sign为SIGN_XXXXXXXX，实际需要填写真实签名数据；
  agt为AGT_XXXXXXXX，实际需要填写真实数据；

- 请求地址：
  svrurl+PartialURL，例如：

```
https://api.ilifesmart.com/app/api.EpMaintOtaTasks
```

- 请求信息：

```
{
  "id": 974,
  "method": "EpMaintOtaTasks",
  "system": {
    "ver": "1.0",
    "lang": "en",
    "userid": "1111111",
    "appkey": "APPKEY_XXXXXXXX",
    "time": 1447639497,
    "sign": "SIGN_XXXXXXXX"
  },
  "params": {
    "agt": "AGT_XXXXXXXX",
    "act": "Query"
  }
}
```

- 签名原始字符串：

```
method:EpMaintOtaTasks,act:Query,agt:AGT_XXXXXXXX,time:1447639497,userid:1111111,usertoken:USERTOKEN_XXXXXXXX,appkey:APPKEY_XXXXXXXX,apptoken:APPTOKEN_XXXXXXXX
```

- 回复信息：

```
{
  "id": 974,
  "code": 0,
  "message":{
    "ZGB752": {
      "id": "ZGB752",
      "cur": 14158,
      "size": 165566,
      "file": "FL01_ZG10370104_00000002.ota",
      "tover": "\u0000\u0000\u0000\u0002",
      "sts": 1577443530,
      "ts": 1577444053
    },...
  }
}
```

- 返回数据说明：

  id：指示ZigBee设备的Id，其值等于"ZG" + zg_nodeid的十六进制表示；

  cur：指示当前OTA升级任务的进度，若cur等于size则指明升级已经完成；

  size：当前OTA文件的总大小；

  file：当前OTA文件的文件名，也即是key；

  tover：当前OTA文件的固件版本号；

  sts：当前升级任务的开始时间，为UTC时间，单位为秒；

  ts：当前升级任务最近一次反馈的时间，为UTC时间，单位为秒；

4.5.27.EpMaintAgtRM 备份或恢复智慧中心上的配置

4.5.27.1.JSON请求数据格式

Type	Definition		Must	Description
Interface Name	EpMaintAgtRM			备份或恢复智慧中心的配置，包括子设备以及AI所有的配置数据
Partial URL	api.EpMaintAgtRM			
Content Type	application/json			
HTTP Method	POST			
	system	ver	Y	1.0
		lang	Y	en
		sign	Y	签名值
		userid	Y	User ID
		appkey	Y	appkey
		did	O	(可选)终端唯一id。如果在授权时填入了，此处必须填入相同id
		time	Y	UTC时间戳，自1970年1月1日起 计算的时间，单位为秒
	method		Y	EpMaintAgtRM
		agt	Y	智慧中心ID
		act	Y	操作指令，类型为字符串，当前可以为如下： • **backup**：备份智慧中心的配置 • **restore**：恢复智慧中心的配置

Request Content	params	actargs	Y	操作指令参数，数据为JSON对象的序列化字符串。操作指令参数与操作指令一一对应，定义如下： • **backup**：{nids, pwd}，nids参数指明需要备份的模块，若不指明则缺省备份智慧中心所有模块的数据，若指明则其数据类型为字符串数组，一般情况下不填写该字段即可。 pwd参数指明加密密码，下次执行恢复操作的时候也必须提供相同的密码，否则将不能用于restore，缺省不提供为没有密码； • **restore**：{cont, pwd, nednids}，cont指明用于恢复的原始备份数据，也即是backup指令返回的数据。 pwd指明恢复密码，也即是backup指令指明的密码，若backup操作指明了密码，则restore操作也必须提供相同的密码。 nednids指明用于仅恢复部分数据，指明仅需要恢复的模块数据，一般不需要提供该属性。
	id		Y	消息id号

4.5.27.2.范例 - Backup

- 我们假定：
  appkey为APPKEY_XXXXXXXX，实际需要填写真实数据；
  apptoken为APPTOKEN_XXXXXXXX，实际需要填写真实数据；
  usertoken为USERTOKEN_XXXXXXXX，实际需要填写真实数据；
  sign为SIGN_XXXXXXXX，实际需要填写真实签名数据；
  agt为AGT_XXXXXXXX，实际需要填写真实数据；

- 请求地址：
  svrurl+PartialURL，例如：

```
https://api.ilifesmart.com/app/api.EpMaintAgtRM
```

- 请求信息：

```
{
  "id": 974,
  "method": "EpMaintAgtRM",
  "system": {
    "ver": "1.0",
    "lang": "en",
```

```
      "userid": "1111111",
      "appkey": "APPKEY_XXXXXXXX",
      "time": 1447639497,
      "sign": "SIGN_XXXXXXXX"
    },
    "params": {
      "agt": "AGT_XXXXXXXX",
      "act": "backup",
      "actargs": "{\"pwd\": \"ls0000\"}"
    }
  }
```

- 签名原始字符串：

```
method:EpMaintAgtRM,act:backup,actargs:{"pwd":
"ls0000"},agt:AGT_XXXXXXXX,time:1447639497,userid:1111111,usertoken:
USERTOKEN_XXXXXXXX,appkey:APPKEY_XXXXXXXX,apptoken:APPTOKEN_XXXXXXXX
```

- 回复信息：

```
{
  "id": 974,
  "code": 0,
  "message":"Q2NfTiRXQB5PLjxfXzlVQ10XLjhPeioiIiA9SBJDUG1......"
}
```

- 返回数据说明：
  message：即为返回的备份的原始数据，该数据经过加密，不可读取。

注意：
备份操作是直接从智慧中心上获取数据的，所以智慧中心必须要在线，若智慧中心不在线则将无法完成备份操作。

4.5.27.3.范例 - Restore

- 请求信息：

```
{
  "id": 974,
  "method": "EpMaintAgtRM",
  "system": {
    "ver": "1.0",
    "lang": "en",
    "userid": "1111111",
    "appkey": "APPKEY_XXXXXXXX",
    "time": 1447639497,
    "sign": "SIGN_XXXXXXXX"
  },
```

```
    "params": {
      "agt": "AGT_XXXXXXXX",
      "act": "restore",
      "actargs": "{\"cont\":
\"Q2NfTiRXQB5PLjxfXzlVQ10XLjhPeioiIiA9SBJDUG1......\", \"pwd\":
\"ls0000\"}"
    }
}
```

- 签名原始字符串：

```
method:EpMaintAgtRM,act:restore,actargs:{"cont":
"Q2NfTiRXQB5PLjxfXzlVQ10XLjhPeioiIiA9SBJDUG1......","pwd":
"ls0000"},agt:AGT_XXXXXXXX,time:1447639497,userid:1111111,usertoken:
USERTOKEN_XXXXXXXX,appkey:APPKEY_XXXXXXXX,apptoken:APPTOKEN_XXXXXXXX
```

- 回复信息：

```
{
  "id": 974,
  "code": 0,
  "message": {
    "rets": {
      "cfg/rf3": true,
      "cfg/rf6": true,
      "me": true,
      "zigbee": true,
      "cfg/rf5": true
    }
  }
}
```

- 返回数据说明：
  message：rets对象指明模块恢复的结果。

注意：
执行恢复数据成功之后请务必调用 **`api.EpRebootAgt`** 接口重启智慧中心才能生效。

4.5.28.EpConfigAgt 设置智慧中心配置

4.5.28.1.JSON请求数据格式

Type	Definition	Must	Description

Interface Name	EpConfigAgt			设置智慧中心的配置，包括是否允许本地登录、修改本地登录密码等。
Partial URL	api.EpConfigAgt			
Content Type	application/json			
HTTP Method	POST			
Request Content	system	ver	Y	1.0
		lang	Y	en
		sign	Y	签名值
		userid	Y	User ID
		appkey	Y	appkey
		did	O	(可选)终端唯一id。如果在授权时填入了，此处必须填入相同id
		time	Y	UTC时间戳，自1970年1月1日起　计算的时间, 单位为秒
	method		Y	EpConfigAgt
	params	agt	Y	智慧中心ID
		act	Y	操作指令，类型为字符串， 具体参考 **4.5.28.2** 说明
		actargs	O	操作指令参数，为JSON对象的序列化字符串， 具体参考 **4.5.28.2** 说明
	id		Y	消息id号

4.5.28.2.act与actargs参数列表

Function 功能	act 操作指令，为字符串	actargs 操作指令参数，为JSON对象的序列化字符串	Response 返回结果
查询是否允许本地登录的配置	queryLocalPermission	不需要提供参数，可以为null或空对象	{stat, policy}，stat指示特定的状态，当stat为"locked"即表明禁止本地登录，stat为"normal"则表示允许本地登录； policy指示本地账号策略，当值为"[nolsi]"表明本地互联必须提供正确的账号/密码，否则不允许登录。缺省为""，表示允许。
设置是否允许本地登录	setLocalPermission	{stat, policy}，stat指明是否允许本地登录，当stat为"locked"即表明禁止本地登录，stat为"normal"则表示允许本地登录； policy指明本地账户策略，当值为"[nolsi]"表明本地互联必须提供正确的账号/密码，否则不允许登录。缺省为""，表示允许。	"success"
设置本地登录的账号密码	setLocalPwd	{pwd}，pwd指明本地登录admin账号所使用的密码，本地登录admin账号缺省密码为admin，可以修改为需要的密码。注意：pwd需要使用标准Base64编码，例如假定密码设置为"hello"，则参数pwd应该为"aGVsbG8="，密码长度没有要求，密码是不可逆的加密方式存储在智慧中心上 设置的密码不允许查询，密码是不可逆的加密方式存储在智慧中心上，因此没有queryLocalPwd动作	"success"
查询智慧中心时区	queryTimezone	{}, 不需要提供参数，可以为null或空对象	{tmzone}，tmzone指明当前所设置的时区
设置智慧中心时区	setTimezone	{tmzone}，tmzone指明本需要设置的时区，时区类型为整数，取值范围:[-11,12]	"success"

Function 功能	**act** 操作指令，为字符串	**actargs** 操作指令参数，为JSON对象的序列化字符串	**Response** 返回结果
查询智慧中心地理位置	queryLoc	{}, 不需要提供参数，可以为null或空对象；	{lnglat}，lnglat返回当前设置的地址位置，其格式为"lng,lat"的字符串
设置智慧中心地理位置	setLoc	{lnglat,fullSync}，lnglat指明本需要设置的地理位置，类型为字符串，格式为"lng,lat"，例如"120.15507,30.274084"为杭州市经纬度； fullSync=1表示全量同步，因为Loc实际上除了存储lnglat之外还可以存储其它地理位置信息，如果fullSync=1则不包含在参数里面的其它键值都会被删除，缺省fullSync=0；	"success"
查询智慧中心Mac、Ip信息	querySys	{}, 不需要提供参数，可以为null或空对象；	{mac, ip}，mac指明智慧中心的MAC地址，ip指明智慧中心当前的IP地址

查询智慧中心扩展服务	queryExSvs	{verbose=1/0}, 查询扩展服务，verbose=1表示将返回每个服务的详细信息，否则仅返回服务是否存在并处于使能状态，缺省verbose=0；	{${exSvc}: {...},}，返回扩展服务当前的配置以及状态。例如天气预报扩展服务配置：{
			"weather": {
			"state": "OK",
			"appkey": "XXX",
			"url": "YYY"
			}
			}

配置智慧中心扩展服务	setExSv	{svId,args={},reload=1/0} 配置扩展服务。svcId指明扩展服务Id，args指明扩展服务配置参数，reload指明扩展服务配置变更之后需要不需要重新加载立即生效，reload=1表示需要重新加载立即生效。具体参考 **4.5.28.4** 说明	"success"

Function 功能	**act** 操作指令，为字符串	**actargs** 操作指令参数，为JSON对象的序列化字符串	**Response** 返回结果

查询智慧中心网络信息	netInfo	{}, 不需要提供参数，可以为null或空对象；	返回网络信息，例如：{
			"gateway": [
			"? (192.168.4.1) at 24:fb:65:60:e3:03 [ether] on eth0\n"
			]
			}
			gateway是一个数组，里面的条目是ARP协议返回的网关信息，可基于ARP协议自行解析网关MAC地址。

查询智慧中心网络模块配置	queryNifCfg	{ifn,verbose=1/0}, 查询网络模块配置，verbose=1表示返回详细的信息，缺省verbose=0。当前ifn值如下可选：
		• "eth0": 网卡
		• "eth1": 华为4G Wifi USB-dongle
		• "ppp0": 2G模块
		• "wlan0": Wi-Fi
		• "usbeth0": USB网卡
		• "wwan0": WWAN-4G 模块 4G-CAT4
		• "usb0": 4G-CAT1
			{${ifn}: {....}}， 返回查询的ifn当前的配置，例如：{
			"wlan0": {
			"enable": true,
			"metric": 100,
			"C_NetworkType": "AP",
			"mode": "rtl8821cs",
			"C_SSID": "cn2",
			"C_WPAPSK": "12345678",
			"C_HWMode": "5G",
			"C_EncrypType": "hh",
			"C_AuthMode": "OPEN",
			"name": "wlan0"
			}
			}

设置智慧中心网络模块	setNifCfg	{ifn,enable=true/false}, 使能/去使能网络模块。当前ifn值如下可选： • "eth0": 网卡 • "eth1": 华为4G Wifi USB-dongle • "ppp0": 2G模块 • "wlan0": Wi-Fi • "usbeth0": USB网卡 • "wwan0": WWAN-4G 模块 4G-CAT4 • "usb0": 4G-CAT1	"success"

Function 功能	**act** 操作指令，为字符串	**actargs** 操作指令参数，为JSON对象的序列化字符串	**Response** 返回结果
查询智慧中心本地互联白名单	queryLsiWhitelist	{}, 不需要提供参数，可以为null或空对象。 有关智慧中心本地互联白名单说明请参考 **4.5.28.5** 说明	返回当前设置的白名单列表 [{lsid,user,pwd,addr},...] • lsid 为智慧设备的LSID(一般等同于智慧设备的agt属性) • user 为智慧设备的登录账号 • pwd 为智慧设备的登录密码
设置智慧中心本地互联白名单	setLsiWhitelist	[['-','lsid'], ['-','*'], ['+',lsid,user,pwd]] 设置白名单，支持多条目一起设置。每个条目为一个JSONArray， • 条目第一项为 "+" 或者 "-" ，"+"指示添加白名单，"-"指示删除白名单； • 条目第二项为智慧设备的lsid，若为删除白名单操作，第二项值为"*"指示删除所有的白名单； • 条目第三项为智慧设备的登录账号，只有添加白名单的时候才需要填写； • 条目第三项为智慧设备的登录密码，只有添加白名单的时候才需要填写； 提示：Update修改操作与添加操作参数一样，如果已经存在该lsid的白名单，则会覆盖该lsid的user、pwd设置。	[{ret,retMsg},...] 返回设置列表项每一项设置是否成功，若成功则ret=0，若失败则ret不等于0，并且retMsg指示具体的错误信息

Function 功能	act 操作指令，为字符串	actargs 操作指令参数，为JSON对象的序列化字符串	Response 返回结果
查询智慧中心本地互联搜索策略	queryLsiPolicy	{}, 不需要提供参数，可以为null或空对象；	{policy}，policy指示当前的本地互联策略，其值可为 "whitelist" 或 ""。"whitelist"表明智慧中心局域网搜索智慧设备的时候只给白名单设备发送搜索包，否则智慧中心会发送组播/广播包。
设置智慧中心本地互联搜索策略	setLsiPolicy	{policy: 'whitelist' or '' }	"success"

4.5.28.3.范例 - setLocalPermission

- 我们假定：
  appkey为APPKEY_XXXXXXXX，实际需要填写真实数据；
  apptoken为APPTOKEN_XXXXXXXX，实际需要填写真实数据；
  usertoken为USERTOKEN_XXXXXXXX，实际需要填写真实数据；
  sign为SIGN_XXXXXXXX，实际需要填写真实签名数据；
  agt为AGT_XXXXXXXX，实际需要填写真实数据；

- 请求地址：
  svrurl+PartialURL，例如：

```
https://api.ilifesmart.com/app/api.EpConfigAgt
```

- 请求信息：

```
{
  "id": 974,
  "method": "EpConfigAgt",
  "system": {
    "ver": "1.0",
    "lang": "en",
    "userid": "1111111",
    "appkey": "APPKEY_XXXXXXXX",
    "time": 1447639497,
    "sign": "SIGN_XXXXXXXX"
  },
  "params": {
    "agt": "AGT_XXXXXXXX",
    "act": "setLocalPermission",
```

```
        "actargs": "{\"stat\": \"locked\"}"
      }
    }
```

- 签名原始字符串：

```
method:EpConfigAgt,act:setLocalPermission,actargs:{"stat":
"locked"},agt:AGT_XXXXXXXX,time:1447639497,userid:1111111,usertoken:
USERTOKEN_XXXXXXXX,appkey:APPKEY_XXXXXXXX,apptoken:APPTOKEN_XXXXXXXX
```

- 回复信息：

```
{
  "id": 974,
  "code": 0,
  "message":"success"
}
```

说明：
本地登录，指的是使用LifeSmart App，在局域网内搜索到智慧中心，然后本地登录该智慧中心，从而进行操作控制。

4.5.28.4.智慧中心ExSv扩展服务说明

• 天气服务

```
svId: "weather",
args: {
      "enable" : True/False 是否使能服务
      "svt": "style1" 服务类型，当前填写"style1"即可
      "url": {WEATHER_SERVICE_URL} 天气服务URL
      "appkey": {APPKEY} 天气服务分配的AppKey
      "apptoken" {APPTOKEN} 天气服务分配的AppToken
},
"reload": 1/0 是否重新加载立即生效
```

4.5.28.5.智慧中心本地互联白名单说明

本地互联指的是LifeSmart智慧设备在局域网内的互相通信功能，例如LifeSmart智慧中心可以在局域网内搜索到LifeSmart摄像头，认证通过之后便可在局域网内播放摄像头的实时视频流、设置摄像头的播放参数、控制摄像头的播放行为等；再比如LifeSmart智慧中心具备级联的功能，可以把下级智慧中心下面的子设备、例如Nature Mini下面的子设备同步到本智慧中心下面做为子

设备，从而查询/控制Nature Mini下面的子设备。

一般家庭网络智慧设备比较少，可以不用设置本地互联白名单。但如果网络环境复杂，智慧设备较多，为了使智慧设备相互隔离、减少网络数据广播风暴、确保安全稳定的使用体检则可以使用本地互联白名单功能。一种典型的应用环境为公租屋、民宿酒店，各个房间的智慧设备都处在一个局域网内，相互都可以被搜索到。而这往往会带来如下问题：

现场施工的时候智慧中心搜索局域网内其它智慧设备会搜索到非相关的设备；

使用没有隔离，通过局域网可以控制/使用非管理域内的智慧设备；

网络拥塞，智慧中心搜索发起的广播包会在局域网内蔓延；

使用本地互联白名单可以解决如上问题。为方便阐述问题，假定有一个房间，里面有如下LifeSmart智慧设备：智慧中心 Agt1、超能面板 NatureMini1、摄像头 Camera1。

假定Agt1的agt属性为 "Agt1_agt";

假定NatureMini1的lsid/agt属性为 "NatureMini1_agt";

假定Camera1的lsid/agt属性为 "Camera1_agt";

调用 **setLocalPermission** 指令设置超能面板与摄像头本地访问策略

```
{
  ″agt″: ″NatureMini1_agt″,
  ″act″: ″setLocalPermission″,
  ″actargs″: ″{\″policy\″: \″[nolsi]\″}″
}

{
  ″agt″: ″Camera1_agt″,
  ″act″: ″setLocalPermission″,
  ″actargs″: ″{\″policy\″: \″[nolsi]\″}″
}
```

调用 **setLocalPwd** 指令设置超能面板与摄像头本地访问密码 "123456"

```
{
  ″agt″: ″NatureMini1_agt″,
  ″act″: ″setLocalPwd″,
  ″actargs″: ″{\″pwd\″: \″MTIzNDU2\″}″
}

{
  ″agt″: ″Camera1_agt″,
  ″act″: ″setLocalPwd″,
  ″actargs″: ″{\″pwd\″: \″MTIzNDU2\″}″
}
```

调用 **setLsiWhitelist** 指令设置相关的智慧设备白名单

```
{
  ″agt″: ″Agt1_agt″,
  ″act″: ″setLsiWhitelist″,
    ″actargs″: ″[[\″+\″, \″NatureMini1_agt\″, \″admin\″, \″123456\″],[\″+\″, \″Camera1_agt\″, \″admin\″,
\″123456\″]]″
```

```
}
```

调用 **setLsiPolicy** 指令设置策略为 `"whitelist"` ，只给白名单设备下发检索包

```
{
  ″agt″: ″Agt1_agt″,
  ″act″: ″setLsiPolicy″,
  ″actargs″: ″{\″policy\″: \″whitelist\″}″
}
```

5.设备属性定义

OpenAPI接口分为两类，一类是查询设备类，一类是控制设备类；
当对设备进行查询接口调用时，返回的数据信息中含有一些通用属性用于描述该设备，见：**表5-1-1**，另外不同的设备包含的其特有属性，如不同的IO控制口的属性信息说明，请参阅文档：**《LifeSmart智慧设备规格属性说明》**

Attribute	Type	RW	Decription
devtype/cls	string	R	设备类型
name	string	RW	设备名称
agt	string	R	设备所属智慧中心ID号
me	string	R	设备ID(智慧中心下面唯一)
stat	int32	R	是否处于在线状态 0: offline 1: online
data	array	R	设备下的IO控制口的信息

表5-1-1 通用属性说明

当对智慧设备进行控制类接口调用时，请参阅文档：**《LifeSmart智慧设备规格属性说明》** 中对应需要控制的设备规格的属性说明设置控制参数，需要注意的是，其中某些属性没有针对OpenAPI进行开放，且只有标识为**RW**的属性才具有设置功能。

6.发现协议

发现协议定义了在局域网中发现智慧中心等智慧设备的方法。

协议	UDP
端口号	12345
报文类型	广播报文

报文内容	Z-SEARCH * \r\n
智慧中心/智慧设备返回格式	MOD=xxxx\nSN=xxxx\nNAME=xxxx\n 或者 MGAMOD=xxxxx\nLSID=xxxxx\nNAME=xxxxx\n 其中xxxx是智慧设备相应的设置值。 新版本智慧设备也可能会添加新的属性，会以'\n'分割，如果不识别可以忽略。 • MGAMOD或MOD 为智慧设备类型； • SN或者LSID 为智慧设备序列号； • NAME 为智慧设备名称

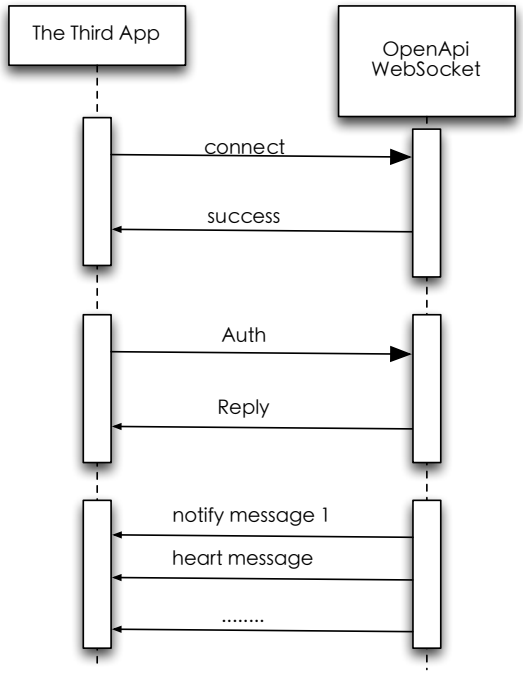
7.状态接收

第三方应用可以关注设备状态的更改事件。LifeSmart OpenAPI 服务提供以 WebSocket 方式接入，关注设备的状态更改事件。

Note:

- 第一次认证成功后，Socket将保持连接，可以对该连接发送**WbAuth**和**RmAuth**接口请求。
- **WbAuth**：WebSocket认证，一次认证一个用户，可以在一个连接多次认证，达到单连接多用户的效果，并且可以持续认证；
- **RmAuth**：WebSocket认证移除，一次移除一个用户，用户数到零时，连接自动断开；

7.1.流程



7.2.URL

根据不同的用户账号（LifeSmart用户）所在的服务区域，WebSocket服务需要使用对应的服务地址。**svrrgnid** 为服务分区标记，在用户授权成功后将会返回对应的 **svrrgnid** 信息。
根据不同的 **svrrgnid** 使用以下对应的 **WebSocket URL**：

Service Type	svrrgnid	URL
WebSocket with SSL	GS	wss://api.ilifesmart.com:8443/wsapp/
	CN0	wss://api.cn0.ilifesmart.com:8443/wsapp/
	VIP1	wss://api.cn1.ilifesmart.com:8443/wsapp/
	CN2	wss://api.cn2.ilifesmart.com:8443/wsapp/
	AME	wss://api.us.ilifesmart.com:8443/wsapp/
	EUROPE	wss://api.eur.ilifesmart.com/wsapp/
	JAP	wss://api.jp.ilifesmart.com:8443/wsapp/
	APZ	wss://api.apz.ilifesmart.com:8443/wsapp/

注意：
WebSocket URL地址的选择，必须根据用户授权成功后返回的 svrrgnid 保持一致，否则不会正常工作，WebSocket不支持跨区使用。

7.3.WebSocket认证

正确连接 WebSocket 后，请第一时间发送认证消息，否则连接会被中断。

7.3.1.JSON请求数据格式

Type	Definition		Must	Description
Interface Name	WbAuth			WebSocket认证
Content Type	application/json			
Request Content	system	ver	Y	1.0
		lang	Y	en
		sign	Y	签名值
		userid	Y	User id
		appkey	Y	appkey
		did	O	(可选)终端唯一id。如果在授权时填入了，此处必须填入相同id
		time	Y	UTC时间戳，自1970年1月1日起 计算的时间，单位为秒
	method		Y	WbAuth
	id		Y	消息id号

7.3.2.范例

- 我们假定：
  appkey为APPKEY_XXXXXXXX，实际需要填写真实数据；
  apptoken为APPTOKEN_XXXXXXXX，实际需要填写真实数据；
  usertoken为USERTOKEN_XXXXXXXX，实际需要填写真实数据；
  sign为SIGN_XXXXXXXX，实际需要填写真实签名数据；

- 请求信息：

```
{
  "id": 957,
  "method": "WbAuth",
  "system": {
     "ver": "1.0",
     "lang": "en",
     "userid": "1111111",
     "appkey": "APPKEY_XXXXXXXX",
```

```
        "time": 1447641115,
        "sign": "SIGN_XXXXXXXX"
    }
}
```

- 签名原始字符串：

```
method:WbAuth,time:1447641115,userid:1111111,usertoken:USERTOKEN_XXX
XXXXX,appkey:APPKEY_XXXXXXXX,apptoken:APPTOKEN_XXXXXXXX
```

- 回复信息：

```
{
  "id": 957,
  "code": 0,
  "message": "success"
}
```

7.4.WebSocket认证用户移除

7.4.1.JSON请求数据格式

Type	Definition		Must	Description
Interface Name	RmAuth			WebSocket认证用户移除
Content Type	application/json			
Request Content	system	ver	Y	1.0
		lang	Y	en
		sign	Y	签名值
		userid	Y	User id
		appkey	Y	appkey
		did	O	(可选)终端唯一id。如果在授权时填入了，此处必须填入相同id
		time	Y	UTC时间戳，自1970年1月1日起 计算的时间,单位为秒
	method		Y	RmAuth
	id		Y	消息id号

7.4.2.范例

- 我们假定：
  appkey为APPKEY_XXXXXXXX，实际需要填写真实数据；
  apptoken为APPTOKEN_XXXXXXXX，实际需要填写真实数据；
  usertoken为USERTOKEN_XXXXXXXX，实际需要填写真实数据；
  sign为SIGN_XXXXXXXX，实际需要填写真实签名数据；

- 请求信息：

```
{
  "id": 957,
  "method": "RmAuth",
  "system": {
    "ver": "1.0",
    "lang": "en",
    "userid": "1111111",
    "appkey": "APPKEY_XXXXXXXX",
    "time": 1447641115,
```

```
        "sign": "SIGN_XXXXXXXX"
      }
    }
```

- 签名原始字符串：

```
method:RmAuth,time:1447641115,userid:1111111,usertoken:USERTOKEN_XXX
XXXXX,appkey:APPKEY_XXXXXXXX,apptoken:APPTOKEN_XXXXXXXX
```

- 回复信息：

```
{
  "id": 957,
  "code": 0,
  "message": "success"
}
```

7.5.事件格式

所有事件数据为 JSON 编码，其格式如下：

id	消息id号
type	事件类型。type='io'表示IO状态数据更新
msg	事件体。根据不同事件类型进行不同解析

7.6.事件数据信息

事件内容属性如下：

字段名称	类型	描述
userid	string	用户ID
agt	string	智慧中心id
me	string	终端设备id 当为智慧中心级别事件则其值不存在； 当为AI类型事件其值表示的是AI的id；

idx	string	终端设备IO指示。根据设备类型不同，其内容不一样。与设备属性的DevType对应，比如T,O等。 idx="s"是特殊的类型，需要查看其它属性做进一步的判断。 如果有info字段则需要参考info信息做进一步处理，例如设备添加，设备删除，设备名称修改，AI添加/删除/变化等特定事件；否则即为设备/智慧中心上下线。
devtype	string	设备类型 当为智慧中心事件则其值固定为"agt"； 当为AI类型事件则其值固定为"ai"；
fullCls	string	包含设备版本号的完整设备类型
type	int32	终端设备IO值类型，含义同设备属性定义，当idx="s"标识特殊事件的时候，其值无效
val	int32	终端设备IO值，含义同设备属性定义，当idx="s"标识特殊事件的时候，其值无效
v	float	真实有效值。在idx!="s"情况下，其值是val值的真实友好值，例如温度变化事件，val=230，v=23.0，表示温度值是23.0摄氏度，在idx="s"情况下，其值是特定的值，具体如下: 设备上线，其值=1； 设备下线，其值=2；
ts	int64	事件发生UTC时间．从1970.1.1到现在的毫秒数
info	string	事件扩展信息，其值有："add"，"del"，"name"，"ioname"，"full"，"chg"等。 add：标识是设备/智慧中心/AI的添加事件； del：标识是设备/智慧中心/AI的删除事件； name：标识是设备名称修改的事件； ioname：标识是设备IO名称修改的事件； full：标识是智慧中心全量同步的事件； chg：标识是AI的修改事件，包括名称，状态，配置；
name	string	仅在名称修改事件有效，包括设备名称/设备IO名称/AI名称的修改事件，既info="name"或info="ioname"或info="chg"。指示新的名称。
io	string	仅在设备IO名称修改事件有效，既info="ioname"。指示设备的哪个IO发生名称变更。
stat	int32	仅AI事件有效，标识AI的状态发生变更 stat=0 表示 AI处于初始态； stat=3 表示 AI正在运行； stat=4 表示 AI运行完成；
cmdlist	bool	仅AI事件有效，为True标识AI的配置发生变更

场景定义举例：

IO变化事件

有效字段：userid,agt,me,idx,devtype,type,val,v,ts

例如插座打开事件

{"type":"io","msg":{"userid":"10001","agt":"A3EAAABtAEwQRzM0Njg5NA","me":"2ad5","idx":"L1","type":129,"val":1,"devtype":"SL_SW_RC","ts":1521455567867},"id":1}

例如环境感应器湿度变化事件

{"type":"io","msg":{"userid":"10001","agt":"A3EAAABtAEwQRzM0Njg5NA","me":"2aca","idx":"H","type":95,"val":462,"v":46.2,"devtype":"SL_SC_THL","ts":1521532876138},"id":1}

设备上线事件

有效字段：userid,agt,me,idx="s",devtype,,v=1,ts

{"type":"io","msg":{"userid":"10001","agt":"A3EAAABtAEwQRzM0Njg5NA","me":"2aca","idx":"s","devtype":"SL_SF_IF3","v":1,"ts":1521532876138},"id":1}

设备离线事件

有效字段：userid,agt,me,idx="s",devtype,,v=2,ts

{"type":"io","msg":{"userid":"10001","agt":"A3EAAABtAEwQRzM0Njg5NA","me":"2aca","idx":"s","devtype":"SL_SF_IF3","v":2,"ts":1521532876138},"id":1}

设备名称修改事件

有效字段：userid,agt,me,idx="s",devtype,info="name",name,ts

{"type":"io","msg":{"userid":"10001","agt":"A3EAAABtAEwQRzM0Njg5NA","me":"2aca","idx":"s","info":"name","name":"NEW_NAME","devtype":"SL_SF_IF3","ts":1521532876138},"id":1}

IO名称修改事件

有效字段：userid,agt,me,idx="s",devtype,info="ioname",name,io,ts

{"type":"io","msg":{"userid":"10001","agt":"A3EAAABtAEwQRzM0Njg5NA","me":"2aca","idx":"s","info":"ioname","name":"NEW_IO_NAME","io":"L2","devtype":"SL_SF_IF3","ts":1521532876138},"id":1}

设备添加事件

有效字段：userid,agt,me,idx="s",devtype,info="add",v=0,ts

{"type":"io","msg":{"userid":"10001","agt":"A3EAAABtAEwQRzM0Njg5NA","me":"2aca","idx":"s","info":"add","devtype":"SL_SF_IF3","v":0,"ts":1521532876138},"id":1}

设备删除事件

有效字段：userid,agt,me,idx="s",devtype,info="del",v=-1,ts

{"type":"io","msg":{"userid":"10001","agt":"A3EAAABtAEwQRzM0Njg5NA","me":"2aca","idx":"s","info":"del","devtype":"SL_SF_IF3","v":-1,"ts":1521532876138},"id":1}

智慧中心添加事件

有效字段：userid,agt,idx=″s″,devtype=″agt″,info=″add″,v=0,ts

{″type″:″io″,″msg″:{″userid″:″10001″,″agt″:″A3EAAABtAEwQRzM0Njg5NA″,″idx″:″s″,″info″:″add″,″devtype″:″agt″,″v″:0,″ts″:1521532876138},″id″:1}

智慧中心删除事件

有效字段：userid,agt,idx=″s″,devtype=″agt″,info=″del″,v=-1,ts

{″type″:″io″,″msg″:{″userid″:″10001″,″agt″:″A3EAAABtAEwQRzM0Njg5NA″,″idx″:″s″,″info″:″del″,″devtype″:″agt″,″v″:-1,″ts″:1521532876138},″id″:1}

智慧中心上线事件

有效字段：userid,agt,idx=″s″,devtype=″agt″,v=1,ts

{″type″:″io″,″msg″:{″userid″:″10001″,″agt″:″A3EAAABtAEwQRzM0Njg5NA″,″idx″:″s″,″devtype″:″agt″,″v″:1,″ts″:1521532876138},″id″:1}

智慧中心离线事件

有效字段：userid,agt,idx=″s″,devtype=″agt″,v=2,ts

{″type″:″io″,″msg″:{″userid″:″10001″,″agt″:″A3EAAABtAEwQRzM0Njg5NA″,″idx″:″s″,″devtype″:″agt″,″v″:2,″ts″:1521532876138},″id″:1}

智慧中心全量同步事件

有效字段：userid,agt,idx=″s″,devtype=″agt″,info=″full″,ts

{″type″:″io″,″msg″:{″userid″:″10001″,″agt″:″A3EAAABtAEwQRzM0Njg5NA″,″idx″:″s″,″info″:″full″,″devtype″:″agt″,″ts″:1521532876138},″id″:1}

AI添加事件

有效字段：userid,agt,me,idx=″s″,devtype=″ai″,info=″add″,v=0,ts

{″type″:″io″,″msg″:{″userid″:″10001″,″agt″:″A3EAAABtAEwQRzM0Njg5NA″,″me″:″AI1544609170″,″idx″:″s″,″info″:″add″,″devtype″:″ai″,″v″:0,″ts″:1544609170932},″id″:1}

AI删除事件

有效字段：userid,agt,me,idx=″s″,devtype=″ai″,info=″del″,v=-1,ts

{″type″:″io″,″msg″:{″userid″:″10001″,″agt″:″A3EAAABtAEwQRzM0Njg5NA″,″me″:″AI1544609170″,″idx″:″s″,″info″:″del″,″devtype″:″ai″,″v″:-1,″ts″:1544609265325},″id″:1}

AI变化事件-修改名称

有效字段：userid,agt,me,idx=″s″,devtype=″ai″,info=″chg″,name,ts

{″type″:″io″,″msg″:{″userid″:″10001″,″agt″:″A3EAAABtAEwQRzM0Njg5NA″,″me″:″AI1544609170″,″idx″:″s″,″info″:″chg″,″devtype″:″ai″,″name″:″NewName″,″ts″:1544609265325},″id″:1}

AI变化事件-修改配置

有效字段：userid,agt,me,idx=″s″,devtype=″ai″,info=″chg″,cmdlist,ts

{″type″:″io″,″msg″:{″userid″:″10001″,″agt″:″A3EAAABtAEwQRzM0Njg5NA″,″me″:″AI1544609170″,″idx″:″s″,″info″:″chg″,″devtype″:″ai″,″cmdlist″:True,″ts″:1544609265325},″id″:1}

AI变化事件-状态变化

有效字段：userid,agt,me,idx=″s″,devtype=″ai″,info=″chg″,stat,ts

{″type″:″io″,″msg″:{″userid″:″10001″,″agt″:″A3EAAABtAEwQRzM0Njg5NA″,″me″:″AI1544609170″,″idx″:″s″,″info″:″chg″,″devtype″:″ai″,″stat″:3,″ts″:1544609265325},″id″:1}

8. 智能应用用户API

（此类接口有限开放，非企业级用户以及没用此需求的不建议使用）

8.1. 注册用户

如果用户没有LifeSmart账号，第三方应用可以通过此接口创建用户，并自动授权。

Note:

若第三方应用无此权限，则此接口不能使用。

8.1.1.JSON请求数据格式

Type	Definition		Must	Description
Interface Name	RegisterUser			用户注册
Partial URL	auth.RegisterUser			
Content Type	application/json			
HTTP Method	POST			
Request Content	system	ver	Y	1.0
		lang	Y	en
		sign	Y	签名值
		userid	Y	User ID，固定"10001"
		appkey	Y	appkey
		time	Y	UTC时间戳, 自1970年1月1日起 计算的时间, 单位为秒
	method		Y	RegisterUser
	params	email	O	Email，与手机号二选一
		mobile	O	手机号，与email二选一
		pwd	Y	密码
		nick	O	昵称
		rgn	O	用户所在区域
	id		Y	消息id号

Note:

rgn为开发者根据实际需求需要填写的参数，表示注册用户所处区域，即所在国家或区域。原则上不可不填写，若不填即默认为中国大陆区(cn)，其它区域参照 **附录1**，rgn填写为国际域名缩写。

RegisterUser方法不涉及对已有用户的操作，故其userid和usertoken使用默认值userid=”10001”，usertoken=”10001”。

RegisterUser的接口请求地址，开发者需要根据传入的rgn找到对应的svrrgnid来使用该方法，参照 附录1 和 附录2。

8.1.2.范例

- 我们假定:
  appkey为APPKEY_XXXXXXXX，实际需要填写真实数据；

apptoken为APPTOKEN_XXXXXXXX，实际需要填写真实数据；
sign为SIGN_XXXXXXXX，实际需要填写真实签名数据；

- 请求地址：
  BaseURL+PartialURL，
  BaseURL的选择，可能与您申请的应用 appkey 所拥有的区域权限有关，若出现接口错误信息形如：“app not existed”，请检查调用接口使用的 appkey 所有的区域权限（该信息可在开放平台上进行查看）。并参考 **附录2 服务代号及地址对应表** 调整请求的BaseURL。
  BaseURL的选择：具体服务器地址可参照：**附录2 服务代号及地址对应表**

  例如：

```
https://api.ilifesmart.com/app/auth.RegisterUser
```

- 请求信息：

```
{
  "id": 963,
  "method": "RegisterUser",
  "system": {
    "ver": "1.0",
    "lang": "en",
    "userid": "10001",
    "appkey": "APPKEY_XXXXXXXX",
    "time": 1447650170,
    "sign": "SIGN_XXXXXXXX"
  },
  "params": {
    "pwd": "password_XXX",
    "email": "d@d.com",
    "nick": "nickname_XXX"
  }
}
```

- 签名原始字符串：

```
method:RegisterUser,email:d@d.com,nick:nickname_XXX,pwd:password_XXX
,time:1447650170,userid:10001,usertoken:10001,appkey:APPKEY_XXXXXXXX
,apptoken:APPTOKEN_XXXXXXXX
```

- 回复信息：

```
{
   "id": 963,
   "code": 0,
   "userid": "10010",
   "usertoken": "xxxxxxxx"
}
```

8.2.删除用户

第三方应用可以通过此接口删除已授权用户。

Note:
若第三方应用无此权限，则此接口不能使用，并且用户为已授权用户。

8.2.1.JSON请求数据格式

Type	Definition		Must	Description
Interface Name	UnregisterUser			解除用户授权
Partial URL	auth.UnregisterUser			
Content Type	application/json			
HTTP Method	POST			
Request Content	system	ver	Y	1.0
		lang	Y	en
		sign	Y	签名值
		userid	Y	user id
		appkey	Y	appkey
		time	Y	UTC时间戳，自1970年1月1日起 计算的时间, 单位为秒
	method		Y	UnregisterUser
	params		O	
	id		Y	消息id号

8.2.2.范例

- 我们假定:
  appkey为APPKEY_XXXXXXXX，实际需要填写真实数据；
  apptoken为APPTOKEN_XXXXXXXX，实际需要填写真实数据；
  userid为USERID_XXXXXXXX，实际需要填写真实数据；
  usertoken为USERTOKEN_XXXXXXXX，实际需要填写真实数据；
  sign为SIGN_XXXXXXXX，实际需要填写真实签名数据；

- 请求地址:
  svrurl+PartialURL，例如:

```
https://api.ilifesmart.com/app/auth.UnregisterUser
```

- 请求信息：

```
{
  "id": 963,
  "method": "UnregisterUser",
  "system": {
    "ver": "1.0",
    "lang": "en",
    "userid": "USERID_XXXXXXXX",
    "appkey": "APPKEY_XXXXXXXX",
    "time": 1447650170,
    "sign": "SIGN_XXXXXXXX"
  },
  "params": {}
}
```

- 签名原始字符串：

```
method:UnregisterUser,time:1447650170,userid:USERID_XXXXXXXX,usertok
en:USERTOKEN_XXXXXXXX,appkey:APPKEY_XXXXXXXX,apptoken:APPTOKEN_XXXXX
XXX
```

- 回复信息：

```
{
   "id": 963,
   "code": 0,
   "msg": "success"
}
```

附录1 国家域名缩写以及服务提供映射

国际域名缩写	国家或地区	Countries and Regions	服务提供地	服务代号 svrrgnid
ae	阿拉伯联合酋长国	United Arab Emirates	America	AME
ag	安提瓜和巴布达	Antigua and Barbuda	America	AME
am	亚美尼亚	Armenia	Europe	EUR
apz	亚太地区	Asia Pacific	Asia Pacific	APZ
ar	阿根廷	Argentina	America	AME
at	奥地利	Austria	Europe	EUR
au	澳大利亚	Australia	America	AME
bb	巴巴多斯	Barbados	America	AME
bd	孟加拉国	Bangladesh	Asia Pacific	APZ
be	比利时	Belgium	Europe	EUR
bg	保加利亚	Bulgari	Europe	EUR
bh	巴林	Bahrain	America	AME
bn	文莱	Brunei	Asia Pacific	APZ
bo	玻利维亚	Bolivia	America	AME
br	巴西	Brazil	America	AME
bs	巴哈马	Bahamas	America	AME
by	白俄罗斯	Belarus	Europe	EUR
bz	伯利兹	Belize	America	AME
ca	加拿大	Canada	America	AME
ch	瑞士	Switzerland	Europe	EUR
cl	智利	Chile	America	AME
cn	中国（旧区）	China	China(old)	CN0
cn1	酒店和公司用户	Hotel and Corporate users	China(special)	VIP1
cn2	中国（新区）	China	China(new)	CN2
co	哥伦比亚	Colombia	America	AME
cr	哥斯达黎加	Costa Rica	America	AME
cy	塞浦路斯	Cyprus	Europe	EUR

cz	捷克	Czech Republic	Europe	EUR
de	德国	Germany	Europe	EUR
dk	丹麦	Denmark	Europe	EUR
dm	多米尼克	Dominica	America	AME
do	多米尼加共和国	Dominica Rep	America	AME
dz	阿尔及利亚	Algeria	America	AME
ec	厄瓜尔多	Ecuador	America	AME
ee	爱沙尼亚	Estonia	Europe	EUR
eg	埃及	Egypt	America	AME
es	西班牙	Spain	Europe	EUR
et	埃塞俄比亚	Ethiopia	America	AME
fi	芬兰	Finland	Europe	EUR
fr	法国	France	Europe	EUR
gb	英国	United Kingdom	Europe	EUR
gd	格林纳达	Grenada	America	AME
gh	加纳	Ghana	America	AME
gr	希腊	Greece	Europe	EUR
gt	危地马拉	Guatemala	America	AME
gy	圭亚那	Guyana	America	AME
hk	中国香港	China. HongKong	America	AME
hn	洪都拉斯	Honduras	America	AME
hr	克罗地亚	Croatia	Europe	EUR
hu	匈牙利	Hungary	Europe	EUR
id	印度尼西亚	Indonesia	America	AME
ie	爱尔兰	Ireland	Europe	EUR
il	以色列	Israel	America	AME
in	印度	India	America	AME
iq	伊拉克	Republic Of Iraq	Asia Pacific	APZ
ir	伊朗	Iran	America	AME
is	冰岛	Iceland	Europe	EUR
it	意大利	Italy	Europe	EUR

jm	牙买加	Jamaica	America	AME
jo	约旦	Jordan	America	AME
jp	日本	Japan	America	AME
ke	肯尼亚	Kenya	America	AME
kh	柬埔寨	Cambodia	America	AME
kr	韩国	Korea	America	AME
kw	科威特	Kuwait	America	AME
kz	哈萨克斯坦	Kazakhstan	Asia Pacific	APZ
lc	圣卢西亚	St.Lucia	America	AME
li	列士敦士登	Liechtenstein	Europe	EUR
lk	斯里兰卡	Sri Lanka	Asia Pacific	APZ
lt	立陶宛	Lithuania	Europe	EUR
lu	卢森堡	Luxembourg	Europe	EUR
lv	拉脱维亚	Latvia	Europe	EUR
ma	摩洛哥	Morocco	America	AME
md	摩尔多瓦	Moldova, Republic of	Europe	EUR
me	黑山共和国	Montenegro	Europe	EUR
mk	马其顿	Macedonia	Europe	EUR
mm	缅甸	Myanmar	Asia Pacific	APZ
mn	蒙古	Mongolia	Asia Pacific	APZ
mo	中国澳门	China. Macao	America	AME
mt	马耳他	Malta	Europe	EUR
mu	毛里求斯	Mauritius	America	AME
mx	墨西哥	Mexico	America	AME
my	马来西亚	Malaysia	Asia Pacific	APZ
ng	尼日利亚	Federal Republic of Nigeria	America	AME
ni	尼加拉瓜	Nicaragua	America	AME
nl	荷兰	Netherlands	Europe	EUR
no	挪威	Norway	Europe	EUR
np	尼泊尔	Nepal	Asia Pacific	APZ
nz	新西兰	New Zealand	America	AME

om	阿曼	Oman	America	AME
pa	巴拿马	Panama	America	AME
pe	秘鲁	Peru	America	AME
ph	菲律宾	Philippines	Asia Pacific	APZ
pk	巴基斯坦	Pakistan	Asia Pacific	APZ
pl	波兰	Poland	Europe	EUR
pt	葡萄牙	Portugal	Europe	EUR
py	巴拉圭	Paraguay	America	AME
qa	卡塔尔	Qatar	America	AME
ro	罗马尼亚	Romania	Europe	EUR
rs	塞尔维亚	Serbia	Europe	EUR
ru	俄罗斯	Russia	Europe	EUR
sa	沙特阿拉伯	Saudi Arabia	America	AME
se	瑞典	Sweden	Europe	EUR
sg	新加坡	Singapore	Asia Pacific	APZ
si	斯洛文尼亚	Slovenia	Europe	EUR
sk	斯洛伐克	Slovakia	Europe	EUR
sr	苏里南	Suriname	America	AME
sv	萨尔瓦多	EI Salvador	America	AME
th	泰国	Thailand	Asia Pacific	APZ
tr	土耳其	Turkey	Europe	EUR
tt	特立尼达和多巴哥	Trinidad and Tobago	America	AME
tw	中国台湾省	China. Taiwan	America	AME
ua	乌克兰	Ukraine	Europe	EUR
us	美国	United States of America	America	AME
uy	乌拉圭	Uruguay	America	AME
ve	委内瑞拉	Venezuela	America	AME
vn	越南	Vietnam	Asia Pacific	APZ
za	南非	South Africa	America	AME

附录2 服务代号及地址对应表

Service Type	svrrgnid	URL
OpenAPI URL	GS	https://api.ilifesmart.com/app/
	CN0	https://api.cn0.ilifesmart.com/app/
	VIP1	https://api.cn1.ilifesmart.com/app/
	CN2	https://api.cn2.ilifesmart.com/app/
	AME	https://api.us.ilifesmart.com/app/
	EUROPE	https://api.eur.ilifesmart.com/app/
	JAP	https://api.jp.ilifesmart.com/app/
	APZ	https://api.apz.ilifesmart.com/app/