属性

属性是某一类产品在云端注册的核心参数或状态。有时是用户关心的，希望在 App 上看到的数据，比如空调的目标温度；有时不会直接显示在 App 上，但对产品描述是必须的，比如冰箱有几个门。

属性的值发生变化后，必须立刻上报最新的值。此外，属性可以选择周期性上报，或一次性上报。

周期性上报时，每个属性的上报周期可以不同，最短周期不得低于一小时,周期要注意并发性问题，如果某一产品定在每天晚上 8 点准时上报属性，就有可能导致云端处理不过来而丢包，造成严重的并发性问题。 另外，因为属性在云端也需要维持一个最新属性值，所以重要的属性采用 request 通信方式已保证状态同步，可用在 APP 设备列表的设备状态以及智能场景中，不重要的可采用 notify 上报。 不重要的属性的上报，为了统计作用可以使用日志上报，见下文。

属性上报时：

method = "prop." + 属性名称

属性举例：

{"method":"prop.power","params":["off"],"id":123}

批量上报属性：

设备有时需要同时上传多个属性值，如果挨个上报，效率较低，因此这里提供一个一次性上传多个属性的通用方法(params 是 kv 型的以确保任何属性都能正常上报)：

{"method":"props","params":{"prop1":int_value1,"prop2":"str_value2"},"id":123}

方法

方法是一类产品能够被远程调用而去执行的操作。例如冰箱制冰、风扇调高转速等等。

方法可以带有参数。参数是一个没有参数名称只有参数值的数组。如果没有参数，params 的值应为空数组[]，而不应该为 null。

方法一般会有返回值，用来返回执行结果，正确时执行结果结构可自定义,一般控制类命令方法的正确执行结果为{"result":["ok"]}。

举例，调用“调整音量”方法：

{"id":1,"method":"set_volumn","params":[50]}

结果如果成功： {"id":1,"result":["ok"]}

或是失败： {"id":1,"error":{"code":-xxxx,"message":"xxxxx"}}，一定不要返回 {"result":["error"]}，错误的 code 和 message 见下文的“错误编码”

此处注意，所有成功的响应都通过 result 应答，所有失败的响应都通过 error 应答。

批量获取属性:

get_prop 获取属性,该通用方法用于获取一个或多个属性的值

参数：由属性英文名称组成的数组

返回值：由属性的值组成的数组，其顺序应与参数保持一一对应。

以空调举例：

下发命令：{"id":2,"method":"get_prop","params":["current_temp","target_temp"]}

说明：【该命令可以通过开放平台云端调试接口下发】

- "id"字段是会话 id，设备根据该 id 回复到对应的命令；
- "method"字段即为产品 profile 里面定义的方法名称；
- "params"字段为方法的参数，一般为数组，也可以为 Object，具体内容和参数顺序，用户可以自己定义。

收到回复：{"id":2,"result":[29,26]}

- "id"字段即为命令下发时携带的会话 id；
- 如果是正确执行的回复，字段为"result"，后面可以是数组或 Object，为命令的执行结果；
- 如果是错误的回复，字段为“error”，后面跟错误码 code 和错误信息 message。

事件

事件是一类产品必须及时上报并让用户知晓的情况，比如探测到的突发情况，传感器的数据达到某个值，或者产品的时钟到了某个设定的时刻，又或者产品出现了故障。

事件可以带有多个参数值。参数值之间用”,”隔开。

事件上报时：

method = "event." + 事件名称

举例：

{"id":3,"method":"event.lock_broken","params":[1,"ss"]} {"id":3,"result":["ok"]}

日志

日志是有些产品的状态等必要数据的定期上报。这类数据并不需要经过场景，也并不需要让用户知晓，只需要存入数据库供后期获取统计数据或跟踪状态所用。

日志可以带有一个 Json 对参数，"log_name":val。

val 的具体含义由各产品定义。

日志上报时：

method = "_otc.log" + 统计 JSONresult

举例：

{"id":3,"method":"_otc.log","params":{"log_name":["time,power,temp","10000,on,26","70000,off,27"]}}

错误编码

业务侧错误码

MSG_ERROR_TYPE_DENIED = -1,//permission denied

MSG_ERROR_TYPE_OFFLINE = -2,//device offline

MSG_ERROR_TYPE_TIMEOUT = -3,//request time out

MSG_ERROR_TYPE_SERVER = -4,//internal exception occurred from server

MSG_ERROR_TYPE_DEVICE = -5,//internal exception occurred from device

MSG_ERROR_TYPE_INVALID = -6,//invalid request

MSG_ERROR_TYPE_BODY_TOO_LONG = -7,//msg length is too long

MSG_ERROR_TYPE_MSGPACK_FORMAT = -8,//msg format error

MSG_ERROR_TYPE_UNKNOWN = -9,//unknown error

MSG_ERROR_TYPE_NO_METHOD = -10,//no this method

MSG_ERROR_TYPE_REPEATED = -11,//repeated request

MSG_ERROR_TYPE_FREQUENT = -12,//frequent request

芯片侧错误码

//otd sys error code: -32000 < x < -30000

-30000 调用返回时,出错(常见于下行调用 ack 时没有给出结果，可能是 method 实现故障)

-30001 调用返回时,json 出错(常见于下行调用 ack 时给出错误 json 结果，可能是 method 实现故障)

-30010 程序向 ot 发布上行请求错误(可能参数故障)

-30011 尝试超限，常出现于上行调用服务,但是未答复

-30012 上行请求失败，原因是 ot 的消息队列满，消息被 skip

-30013 上行请求失败，原因是 ot 维持的 host_ip 为 0，可能是 dns 解析尚未完成

-30020 otd 服务(不是 method)不存在 -30030 redirect 失败

//third mcu error code:第三方可以定义自己的错误码

范围：-10000 <= x <= -5000

-10000 第三方芯片默认错误值

//json rpc error code: -32767 < x < -32000

-32700 json 解析出错

-32600 json 正确 但是 不是正确的 rpc 结构

-32601 方法未找到

-32602 参数错误