数据推送服务接入说明书

版本号：V1.1

日期：2018 年 3 月 29 日

文档修订记录

标 准 修 订 履 历 记 录				
版本编号	日 期	修订人	修订页次	修 改 内 容 简 述
会 签				
经营体	签 署 意 见		签 名	日 期
编 制		标 查		
审 核		批 准		

目录

1 引言

1.1执行摘要

作为第三方云平台订阅数据推送服务开发的依据.

1.2编写目的

为第三方云平台接受海尔设备数据开发提供基础和指导。

1.3特殊记号格式说明

本文档暂未使用其他特殊文档记号或格式。

本文中的接口示例均使用的是示例数据并非真实数据。

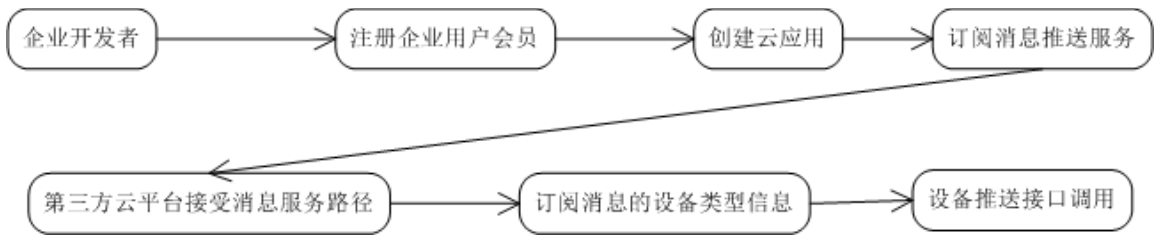
2 公共说明

2.1基本介绍

第三方可以通过数据推送服务，订阅海尔设备消息功能，实现海尔设备数据与第三方平台的互通。

2.2调用流程

第三方平台要使用文档中的接口，需遵循以下流程。



2.3推送地址

本文档提供的所有接口支持 http 和 https 协议请求。

与数据推送的交互接口统一为基于 http 或者 https 的 REST 接口;采用 UTF-8 编码;

在开发联调时，应用开发者应该提供开发者环境的推送地址，接收推送数据；在上线时，应用开发者应该提供生产环境的推送地址，接收推送数据。

2.4签名认证

2.4.1 说明

数据推送平台服务端需要对发送请求进行签名，执行签名计算的签名值需要赋值到 Header 头中的 sign 属性（见公共部分说明），以便调用方进行签名验证。

2.4.2 参数介绍

待签名字符串为：SystemID+SystemKey +timestamp；

SystemID：云应用 ID，40 位以内字符,Haier U+ 云平台全局唯一；

SystemKey：在海极网给应用申请的 SystemKey，不能明文发送；

timestamp：Unix 时间戳（*YYMMDDhhmmss*）；

2.4.3 算法

签名算法就是对待签名字符串计算 32 位小写 SHA-256 值，算法示例见附录。

2.5国际化处理

对接口响应返回的 retCode 和 retInfo 不做国际化处理，由接口调用方处理。

对于接口涉及业务数据的国际化通过在 header 中传递 language 参数来定义，具体的国际化语言代码见附录。

2.6数据类型限定

2.6.1 字段类型说明

限定类型	说明	格式	json 示例
DateTime	日期时间类型的字符串	yyyy-MM-ddhh:mm:ss	{ “lgTime”：“2013-10-08 08:00:00” }
Date	日期类型的字符串	yyyy-MM-dd	{ “lgDate”：“2013-10-08” }
String	字符串		{ “address”：“street 123” }
int	整形数字		{ “age” :1234}
long	长整形数字		{ “oid” :1234567890123}
double	浮点数数字		{ “price” :12.35}
boolean	布尔型（true 或 false）		{ “idOld” :true}。

2.6.2 null 值说明

为避免解析错误， uws 各接口的返回参数，不返回 null 值。

必填参数，无论输入还是输出，必须有值，不能为 null

非必填参数，则说明如下：

数值类型数据（int、long、double）只会返回数字，包括正数、零及负数。

布尔类型数据（boolean）只返回 true 和 false

以上基本类型本身不包含 null 值。

DateTime、Date、String 及结构体类型的数据，如果为 null 时，所对应的属性将不返回。

例如：

DateTime 类型

birthday 为 DateTime 类型，不为 null 时：

{"name":"Tom","age":23,"birthday":"2013-10-08 08:00:00","address":{"city":"beijing","street":"haidian"} }

birthday 为 null 时，则 birthday 属性不返回

{"name":"Tom","age":23, "address":{"city":"beijing","street":"haidian"} }

String 类型

name 是 String 类型，不为 null 时：

{"name":"Tom","age":23,"birthday":"2013-10-08 08:00:00","address":{"city":"beijing","street":"haidian"} }

name 为 null 时，则 name 属性不返回

{"age":23,"birthday":"2013-10-08 08:00:00","address":{"city":"beijing","street":"haidian"} }

结构体类型

address 为结构体类型，不为 null 时：

{"name":"Tom","age":23,"birthday":"2013-10-08 08:00:00","address":{"city":"beijing","street":"haidian"} }

address 为 null 时，address 属性不返回

{"name":"Tom","age":23,"birthday":"2013-10-08 08:00:00" }

3 数据推送服务接入流程说明

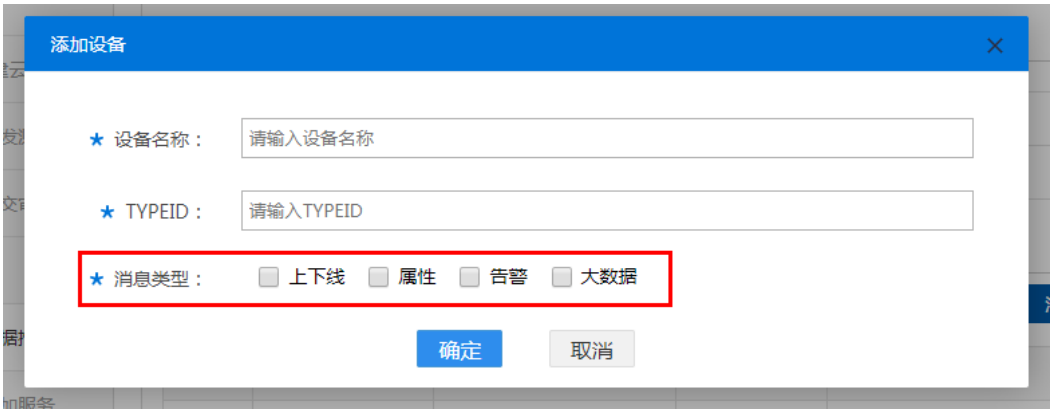
说明：接入流程是结合云应用在海极网上的接入流程。云应用没有上线之前，使用数据推送平台的话只能走线下方式完成数据推送服务的订阅。

需要提供如下信息：

1、提供 SystemID 和 SystemKey

2、提供设备信息（名称、型号）、tyepid 和消息类型

3、提供接收数据推送平台的 4 个 REST 接口上下线路径、状态路径、告警路径和大数据路径。目前，支持根据 typeid 进行推送。



4 附录

4.1公共错误码

错误码	描述
A00001	服务不可用
A00002	网络异常
A00003	访问或操作超时
A00004	系统内部错误
A00005	数据库访问异常
A00006	未知异常
A00007	邮件服务异常
A00008	邮件发送失败
A00009	邮件发送次数超限
B00001	缺少必填参数
B00002	参数类型错误
B00003	参数数值超出值域或不是枚举值
B00004	参数不符合规则要求

B00006	参数长度错误
B00007	参数与接口定义不匹配
C00001	appId 与 appKey 验证失败
C00002	appServer 无访问授权
C00003	访问权限不足
C00004	操作权限不足
C00005	重复请求
C00006	未知设备类型
C00007	appId 配置信息为空
C00008	appKey 为空
D00001	sign 签名错误
D00003	Token 不存在，未通过 token 验证
D00004	Token 已过期，未通过 token 验证。
D00005	Token 不是由此应用创建，未通过 token 验证。
D00006	会话失效
D00007	不是系统内部用户
D00008	用户不合法

4.2签名算法示例

```
String getSign(String appId,StringappKey, String timestamp) throws Exception {

        appId = appId.trim();

        appKey = appKey.trim();

        appKey = appKey.replaceAll("\"", "");

        StringBuffersb = new StringBuffer();
```

```
        sb.append(appId).append(appKey).append(timestamp);


        MessageDigest md = null;
        byte[] bytes = null;
        try {
            md = MessageDigest.getInstance("SHA-256");
            bytes = md.digest(sb.toString().getBytes("utf-8"));
        } catch (Exception e) {
            e.printStackTrace();
        }

        returnBinaryToHexString(bytes);

}

String BinaryToHexString(byte[] bytes) {
    StringBuilder hex = new StringBuilder();
    String hexStr = "0123456789abcdef";
    for (inti = 0; i<bytes.length; i++) {
            hex.append(String.valueOf(hexStr.charAt((bytes[i] & 0xF0) >> 4)));
            hex.append(String.valueOf(hexStr.charAt(bytes[i] & 0x0F)));
        }
```

```
        returnhex.toString();

    }
```