OneNet接入方案与RESTfulAPI接口

欢迎访问设备云门户网站<http://open.iot.10086.cn/>注册用户，获取最新文档。

|  |  |  |  |
| --- | --- | --- | --- |
| **版本号** | **修订日期** | **修订内容** | **说明** |
| V1.1 | 2014.10.30 | EDP增加存储数据报文；  REST API 增加历史数据查询接口，用于上报数据点，或者上报的同时转发数据点。 |  |
| V1.1.1 | 2014/11/3 | EDP增加消息类型9，以支持对存储数据的确认。 |  |
| V1.1.2 | 2015/01/10 | 1、修改3.5.3 AB接口，增加获取每个数据流当前值。  2、数据点添加时若无数据流将自动添加数据流。 |  |
| V1.2.2 | 2015/01/15 | 1. 修改3.5.1 ，新增数据流返回uuid，唯一标识该数据流； 2. 修改3.5.1，增加三种复合数据流的添加； 3. 修改触发器相关操作接口。一个触发器可同时监控不同设备的多个数据流。 4. 修改触发器一些条件的设定方式。 |  |
| V1.2.3 | 2015/06/18 | 1. 去掉复合数据流定义 2. 设备批量查看增加device\_id 参数 3. 触发器增加title字段 4. 新增触发器批量获取接口 5. 修改key查看输出内容格式 |  |
| V1.3.0 | 2015/06/24 | 1、命令操作，增加响应内容查询 |  |
| V1.3.1 | 2015/07/28 | 1、修改4.6.3，新增数据点抽样和统计 |  |
| V3.0 | 2016/06/20 | 1、4.3.3模糊查看设备信息,增加时间查询 |  |
| V3.1 | 2016/08/04 | 1.增加4.10.3注册码创建接口 |  |

目录

[OneNet接入方案与RESTfulAPI接口 1](#_Toc437010302)

[1 概述 3](#_Toc437010303)

[2 基于OneNet的应用系统的典型场景 3](#_Toc437010304)

[3 API资源模型 3](#_Toc437010305)

[4 REST API 5](#_Toc437010306)

[4.1 设计原则 5](#_Toc437010307)

[4.2 HTTP 原始报文格式 6](#_Toc437010308)

[4.3 设备（device） 6](#_Toc437010309)

[4.3.1 新增 6](#_Toc437010310)

[4.3.2 更新 7](#_Toc437010311)

[4.3.3 查看 8](#_Toc437010312)

[4.3.4 删除 11](#_Toc437010313)

[4.4 数据流（datastream） 12](#_Toc437010314)

[4.4.1 新增 12](#_Toc437010315)

[4.4.2 更新 12](#_Toc437010316)

[4.4.3 查看 13](#_Toc437010317)

[4.4.4 删除 14](#_Toc437010318)

[4.5 数据点（datapoint） 14](#_Toc437010319)

[4.5.1 新增 14](#_Toc437010320)

[4.5.2 更新 16](#_Toc437010321)

[4.5.3 查看 16](#_Toc437010322)

[4.5.4 删除 19](#_Toc437010323)

[4.6 触发器（trigger） 19](#_Toc437010324)

[4.6.1 新增 19](#_Toc437010325)

[4.6.2 更新 20](#_Toc437010326)

[4.6.3 查看 21](#_Toc437010327)

[4.6.4 删除 22](#_Toc437010328)

[4.7 API 权限(apikey) 23](#_Toc437010329)

[4.7.1 新增 23](#_Toc437010330)

[4.7.2 更新 23](#_Toc437010331)

[4.7.3 查看 24](#_Toc437010332)

[4.7.4 删除 25](#_Toc437010333)

[4.8 二进制数据（bindata） 25](#_Toc437010334)

[4.8.1 新增 25](#_Toc437010335)

[4.8.2 查看 26](#_Toc437010336)

[4.8.3 删除 26](#_Toc437010337)

[4.9 命令执行（cmd） 26](#_Toc437010338)

[4.9.1 发送命令 26](#_Toc437010339)

[4.9.2 查看状态 27](#_Toc437010340)

[4.9.3 获取响应 27](#_Toc437010341)

[4.10 其他 28](#_Toc437010342)

[4.10.1 触发数据内容 28](#_Toc437010343)

[4.10.2 历史数据查询 29](#_Toc437010344)

# 概述

OneNet是中移物联网有限公司基于物联网技术和产业特点打造的开放平台和生态环境，适配各种网络环境和协议类型，支持各类传感器和智能硬件的快速接入和大数据服务，提供丰富的API和应用模板以支持各类行业应用和智能硬件的开发，能够有效降低物联网应用开发和部署成本，满足物联网领域设备连接、协议适配、数据存储、数据安全、大数据分析等平台级服务需求。

中国移动物联网开放平台始终秉承开放合作的态度，为智能硬件创客和创业企业提供硬件社区服务，为中小企业客户物联网应用需求提供数据展现、数据分析和应用生成服务，为重点行业领域/大客户提供行业PaaS服务和定制化开发服务。

open.iot.10086.cn是OneNet对外公开的门户，可以通过该门户了解到OneNet的最新进展。

本文档主要描述了OneNet基于HTTP方式向外提供API接口以及这些接口的调用方式、参数说明等等。这些接口按照RESTful的方式向外提供服务。

# 基于OneNet的应用系统的典型场景

**设备层：**利用平台提供的EDPSDK，实现EDP协议，用于上报业务数据点到OnetNet。若需要实时接收业务层下发的控制命令，需要保持EDP长连接。

**业务应用层：**若要自定义实现业务平台，可通过HTTP 协议的RESTful API操作OnetNet提供的资源（设备、数据点、命令控制等资源的增删查改）。



适用场景：在充分分析业务数据模型的基础上，认为OneNet提供的设备-数据流-数据点模型适合业务数据存储。优先推荐新业务使用该模式。

# API资源模型

OneNet HTTPAPI按照RESTful的方式向外提供服务，其资源模型中包含的资源种类有：用户、设备（device）、数据流（datastream）、数据点（datapoint）、触发器（trigger）、API key、命令等。

用户是指通过open.iot.10086.cn注册的用户。每个用户可以建立多个项目，每个项目可以添加多个设备。一个设备只能属于一个项目。

一个设备可以产生多个数据流，一个数据流可以理解为一类数据。比如温度是一个数据流、经纬度是一个数据流等等。

数据流中的一个具体的数据称为数据点。数据点采用Key-Value方式的存储的。其中Key的组成包括设备id、数据流id、时间等信息，value部分可以为任何数据对象，如整数、字符串或者JSON数据类型。

API Key用于对设备云进行操作的权限控制，在访问OneNet Rest API时，http请求消息的头域必须携带api-key字段，设备云会依据api-key字段判别该用户是否具有对应操作的权限。默认在open.iot.10086.cn上注册项目时会生成一个默认的APIKey，这个APIKey是masterkey，具有对本项目下所有设备最大的访问权限。除了这个master key之外，还可以通过API命令创建具备不同权限的其他APIKey。APIKey的权限模型是通过设定对某些设备和数据流、是否可以增加、查看、修改、删除等方式来描述的，最低的权限级别可细化到对某个数据流的访问权限。

触发器用于告知设备云当满足某种条件时主动触发某个动作。

命令是让设备云主动给设备推送数据的一种方式。默认只有EDP设备或者MODBUS设备才可以接收设备云发送的命令。大多数使用场景是设备主动向设备云发送数据，也就是数据上行传递；而通过命令的方式，可以控制设备云向设备发送命令，达到数据下行的目标。这可以达到对远程终端进行控制的目的。

二进制数据是一类特别的数据，可以通过api上传到设备云，作为数据点存储在设备云上。二进制数据主要是图片等等。

设备1

数据流1

数据点

产品2

设备云

数据流2

数据点

设备2

数据流1

数据点

数据流2

数据点

设备3

数据流1

数据点

数据流2

数据点

用户

产品1

产品3

# REST API

以下API 需要在平台注册用户并创建项目的条件下使用。

REST API 基于HTTP 协议（详见http://www.w3.org/Protocols/HTTP/1.0/spec.html）和json数据格式（详见http://www.json.org/json-zh.html），适合平台与平台之间数据对接，或使用短连接上报终端数据。

<API\_ADDRESS>外网运营地址：api.heclouds.com端口80 或jjfarfapi.heclouds.com端口80

统一说明：

* 终端设备上行发送给设备云的日期时间格式统一为”2015-12-04T13:00:05”形式，日期和时间之间用大写字母T隔开。包括消息中设计数据点的时间和URL参数中涉及到的时间都按照此格式；
* 设备云下发的日期时间格式为”2015-12-0413:00:05”，中间用空格隔开，与上行格式不同，请注意；

## 设计原则

* 使用标准HTTP方法实现资源CURD操作；
* 采用json作为API输入输出；
* 以json输出错误信息。

|  |  |
| --- | --- |
| **HTTP 方法** | **描述** |
| POST | 新增资源信息 |
| PUT | 更新资源信息 |
| GET | 查看资源信息 |
| DELETE | 删除资源 |

支持的返回码列表：

|  |  |  |
| --- | --- | --- |
| **HTTP返回码** | **含义** | **详细描述** |
| 200 | OK | 在HTTP body中以json格式指明详细成功或错误信息，例如：  {  "errno": 0,  "error":“succ”  }  成功、失败指示，主要以errno是否为零来判断。 |
| 400 | Bad Request |
| 401 | Not Authorized |
| 403 | Forbidden |
| 404 | Not Found |
| 406 | Not Acceptable |
| 422 | Unprocessable Entity |
| 500 | Internal Server Error |
| 503 | No server error |

## HTTP 原始报文格式

例1，自己封装HTTP报文，新增设备，格式如下，其中\r\n是两个字符（ASCII码里面的回车换行），长度32表示两个连续\r\n\r\n后数据的长度：

|  |
| --- |
| POST /devices HTTP/1.1\r\n api-key:<your\_api\_key>\r\n Host:api.heclouds.com\r\n Content-Length: 32\r\n \r\n {"title":"898602B2221430000029"} |

例2，看某个设备的详细情况，格式如下，

|  |
| --- |
| GET /devices/<your\_device\_id> HTTP/1.1\r\n api-key:<your\_api\_key>\r\n Host:api.heclouds.com\r\n \r\n |

## 设备（device）

### 新增

新增设备用于向设备云增加一个设备，通过HTTP POST请求增加设备，增加成功后，会返回该设备的设备ID。

|  |  |
| --- | --- |
| HTTP方法 | POST |
| URL | http://<API\_ADDRESS> /devices |
| HTTP头部 | api-key:xxxx-ffff-zzzzz，必须为MasterKey |
| HTTP内容 | {  *//设备名，应在产品范围内唯一，对于MODBUS、EDP设备，应填写卡号*  "title":"my device1",  *//设备描述*  "desc":"some description",  "tags":[*//可选*  "Tag1",  "Tag2"  ],  *// 设备位置*  "location":{*//可选*  *//高度，double*  "ele":370000,  *// 经度，double*  "lat":17.609997,  *// 纬度，double*  "lon":177.03403  },  *// 本设备是否私有，公开的应用在open.iot.10086.cn上发现上会列出来*  "private":true,  "protocol":"EDP|HTTP|JTEXT",  *//可选对于MODBUS设备有效，需包含卡号（phone）和密码（svrpwd）形式为：{“登录报文的phone字段”:”登录报文的SVRPWD字段”}*  "auth\_info":{…},  *//可选，对modbus设备有效，TCP 类设备，表示设备云主动查询设备的时间间隔，单位秒*  "interval":60,  “other":{…}  } |
| HTTP响应响应消息内容 | {  "errno": 0,  "error":“succ”,  "data":  {  *//平台分配唯一ID*  "device\_id":"233444"  }  } |

说明：

1. 对于非MODBUS和EDP协议的设备，可以忽略auth\_info和interval字段，可以参阅MODBUS接入协议对应的api文档；
2. other字段如果有可填写，如果也不影响设备的创建；
3. 响应消息中errno表示错误码，error表示错误原因，如果创建设备失败，则没有device\_id字段。

### 更新

根据更新参数，需要更新哪些参数就选哪些参数？。

|  |  |
| --- | --- |
| HTTP方法 | PUT |
| URL | http://<API\_ADDRESS>/ devices /<device\_id> |
| HTTP头部 | api-key:xxxx-ffff-zzzzz，可为设备级别的Key |
| HTTP内容 | {  "title":"my device1",  "desc":"some description",  ”private":true,  *//可选*  "tags":[          "Tag1",  "Tag2"  ],  "location":{  "ele":370000,  "lat":17.609991828964787,  "lon":177.03402996826173  },  "auth\_info":{…},  "interval":60,  "other":{}  } |
| 请求返回 | {  "errno": 0,  "error":“succ”,  } |

说明：

1. 更新时，更新那个字段就只需要在json字符串中包括哪个字段，不需要包含所有内容。
2. Private字段表示是否私有。私有的设备不会列在”open.iot.10086.cn->发现”栏目中。
3. auth\_info和interval字段参见设备新增章节的解释。

### 查看

#### a查看单个设备的信息

单个设备查看是在知道待查设备的设备ID的情况下查看设备的明细。会返回待查设备的基本信息、所有数据流信息、apikey信息、二进制数据信息等等。

|  |  |
| --- | --- |
| HTTP方法 | GET |
| URL | http://<API\_ADDRESS>/devices/<device\_id> |
| HTTP头部 | api-key:xxxx-ffff-zzzzz，必须为可查看该设备的Key |
| 请求返回 | {  "errno": 0,  "error":“succ”,  "data":  {  "id":"device\_id",  *//表示EDP设备是否在线，只对EDP设备存在此字段*  "online":true|false,  "protocol":"HTTP"  "title":"my device1",  "desc":"some description,ex:url",  "create\_time":"xx-xx-xx 10:22:22",  "private":true|false,  *//设备关联的图像或二进制数据*  "binary":[{  *//二进制数据索引*  “index”:”FJWOPN9023899”,  *//上传时间*  “at”:”2014-10-23 20:22:22”,  *//二进制数据大小*  “size”:2333(字节),  ”desc”:”binary description”  }]  "tags":["Tag1","Tag2"],  "location":{"ele":370000,"lat":17.609991828964787,"lon":177.034029968273},  *//只对MODBUS和EDP设备有效*  "auth\_info":{…},  "other":{}  "keys":[  {"title":"device-auto-key","key":"2JO3B12O1"},  {"title":"key-name2","key":"DFWfsfw1wfDF233"},  ]，  *//数据流*  "datastreams":  [{  “create\_time”: ”2014-10-23 20:22:22”  "id":"datastream\_id1”,  "unit":"celsius",  "unit\_symbol":"C",  "uuid":" 231a5aa9-9de4-5f2e-9e0f-015181c98429"  }]*//end data\_streams*  }*//end data*  } |

说明：

1. 注意URL最后不能有多余的斜杠，比如http://<API\_ADDRESS>/devices/<device\_id>可以，但是http://<API\_ADDRESS>/devices/<device\_id>/会查询不到。

具体如下：

|  |  |
| --- | --- |
| 请求 | http://api.heclouds.com/devices/564280 |
| 响应 | {  "errno":0,  "data":  {  "private":true,  "protocol":"HTTP",  "create\_time":"2015-11-0414:27:31",  "location":{"lon":106.55231191544001,"lat":29.535451246673},  "id":"564280",  "auth\_info":{"SYS":"z=8QY3r7aTQ48UKfqPnB=PybYRo="},  "datastreams":  [  {  "unit":"摄氏度",  "create\_time":"2015-11-23 09:35:54",  "unit\_symbol":"℃",  "id":"温度",  "uuid":"231a5aa9-9de4-5f2e-9e0f-015181c98429"  },  {  "create\_time":"2015-11-24 15:10:26",  "id":"温度2",  "uuid":"25346b97-01a8-461d-b98b-26139e67f6a8"  },  {  "create\_time":"2015-11-24 15:05:01",  "id":"温度\_2",  "uuid":"6eb512e5-3fad-45a8-bc83-a9eec7d2c792"  }  ],  "title":"设备1",  "desc":"测试设备1",  "tags":["aa,bb"]  },  "error":"succ"  } |

1. Online字段只对EDP设备有效，表示EDP设备是否在线，对于非EDP设备不存在该属性。

#### b模糊查看设备信息

批量查看是指模糊查找设备，会把所有符合条件的设备都列出来。

批量查找通过在URL中附带参数的形式向服务器端查询设备。

|  |  |
| --- | --- |
| HTTP方法 | GET |
| URL | <http://api.onenet.com/devices> |
| URL参数 | key\_word =xxbbcc,//可选,会从设备id字段中去左匹配  tag = xxx //可选  online = true|false //可选  private = true|false //可选  page = 1 //指定页码, 可选  per\_page = 30 //指定每页输出设备个数,可选,默认30，最多100  device\_id = 235122 //指定设备ID,可选,多个用逗号分隔，最多100个  begin=2016-06-20 //可选，包括这天  end=2016-06-20//可选，包括这天 |
| HTTP头部 | api-key:xxxx-ffff-zzzzz |
| 请求返回 | {  "errno": 0,  "error":“succ”,  "data":  {  "total\_count":1,  "per\_page":30,  "page":1,  "devices":  [  {  "id":"dev\_id",  "title":"my device1",  “desc”:”device desc”,  "private":true,  "protocol":"HTTP|EDP|JTEXT",  "online":ture|false,  "tags":["aa","bb"]  "location":{"lon":106.55231191544001,"lat":29.535451246673},  "create\_time":"2013-02-11 12:22:33",  "auth\_info":{"SYS":"z=8QY3r7aTQ48UKfqPnB=PybYRo="}  }  ]  }  } |

### 删除

|  |  |
| --- | --- |
| HTTP方法 | DELETE |
| URL | http://<API\_ADDRESS>/ devices /<device\_id> |
| HTTP头部 | api-key:xxxx-ffff-zzzzz，可为设备级别的Key |
| HTTP内容 | 无 |
| 请求返回 | {  "errno": 0,  "error":“succ”  } |

说明：删除设备会删除该设备下所有数据流和数据点。删除设备动作是异步的，系统会在后续逐步删除该设备下的数据流和数据点。

## 数据流（datastream）

### 新增

|  |  |
| --- | --- |
| HTTP方法 | POST |
| URL | http://<API\_ADDRESS>/devices/<device\_id>/datastreams |
| HTTP头部 | api-key:xxxx-ffff-zzzzz，可以为设备级别的Key |
| HTTP内容 | {  "id":"datastream\_id1”,  *//以下参数可选*  "tags":["Tag1","Tag2"],  "unit":"celsius",  "unit\_symbol":"C",  *//只有MODBUS设备填写，MODBUS命令，16进制字节字符串*  "cmd":"0003000000184411",  *//只有MODBUS设备填写，采集间隔，秒*  "interval":60,  *//只有MODBUS设备填写，最终结果处理方法，寄存器地址从A0开始,多个公式用分号间隔*  "formula":"(A0+A1)\*A2",  } |
| 请求返回 | {  "errno": 0,  "error":“succ”,  "data":  {  "ds\_uuid":"FWFW-WFWW-EWFW-WW"//平台分配数据流唯一ID  }  } |

说明：

* 对于普通的非modbus设备，不要添加cmd、interval、formula等字段，否则可能会导致增加数据流失败。
* 新增数据流的id字段不能为空，否则可能会添加失败
* 新增数据流的id字段支持中文，但最好为英文，这样可以规避字符编码带来的问题。

### 更新

更新数据流主要更新数据流的单位、标签等信息。

|  |  |
| --- | --- |
| HTTP方法 | PUT |
| URL | http://<API\_ADDRESS>/devices/<device\_id>/datastreams/<datastream\_id> |
| HTTP头部 | api-key:xxxx-ffff-zzzzz 可以为数据流级别的key |
| HTTP内容 | {  "tags":["Tag1","Tag2"],  "unit":"celsius",  "unit\_symbol":"C",  "cmd":"0003000000184411",  "interval":60,  "formula":"(A0+A1)\*A2",  } |
| 请求返回 | {  "errno": 0,  "error":“succ”,  } |

说明：请求消息中的的任何字段都可以为空。对于非MODBUS设备不要更新其cmd、interval、formula等字段，否则可能失败

。

### 查看

A查看某设备单个数据流信息

|  |  |
| --- | --- |
| HTTP方法 | GET |
| URL | http://<API\_ADDRESS>/devices/<device\_id>/datastreams/<datastream\_id> |
| HTTP头部 | api-key:xxxx-ffff-zzzzz 可以为数据流级别的key |
| 请求返回 | {  "errno": 0,  "error":“succ”,  *//数据流信息*  "data":  {  *//设备范围内唯一*  "id":"datastream\_id1”,  *//平台范围内唯一*  "uuid":"EFEW-EFW-EFW-WEFE”,  "tags":["Tag1","Tag2"],  "unit":"celsius",  "unit\_symbol":"C",  "create\_time":"xxxx-xx-xx 10:22:22",  "current\_value":{…},  "update\_at":"xx-xx-xx"  }  } |

B查看某设备多个数据流信息：

|  |  |
| --- | --- |
| HTTP方法 | GET |
| URL | http://<API\_ADDRESS>/devices/<device\_id>/datastreams |
| URL参数 | datastream\_ids : 查看的datastream的id号，多个datastream之间用逗号分开  使用示例：  http://api.onenet.com/devices/1234/datastreams?datastream\_ids=temp,gps  表示查看temp和gps两个数据流的信息 |
| HTTP头部 | api-key:xxxx-ffff-zzzzz，可以为数据流级别的Key |
| 请求返回 | {  "errno": 0,  "error":“succ”,  "data":  [{  "id":"datastream\_id1”,  "uuid":"EFEW-EFW-EFW-WEFE”,  "tags":["Tag1","Tag2"],  "unit":"celsius",  "unit\_symbol":"C",  "create\_time":"xxxx-xx-xx 10:22:22",  "current\_value":{…},  "update\_at":"xxxx-xx-xx 10:22:22"  }]  } |

说明：http://api.onenet.com/devices/1234/datastreams后面不带任何参数指定数据流，用于查看设备下所有数据流的概要信息。

### 删除

|  |  |
| --- | --- |
| HTTP方法 | DELETE |
| URL | http://<API\_ADDRESS>/devices /<device\_id>/datastreams/<datastream\_id> |
| HTTP头部 | api-key:xxxx-ffff-zzzzz， 可以为数据流级别的key |
| HTTP内容 | 无 |
| 请求返回 | {  "errno": 0,  "error":“succ”,  } |

## 数据点（datapoint）

数据点是指一个数据流中的某一条具体的数据。对数据流主要包括新增、查看、删除。暂不提供更新操作的API

### 新增

|  |  |
| --- | --- |
| HTTP方法 | POST |
| URL | *http://*<API\_ADDRESS>*/devices/<device\_id>/datapoints* |
| HTTP头部 | api-key:xxxx-ffff-zzzzz，数据流key  (鉴权是根据数据内容进行的，如果添加的数据涉及多个数据流，则需要key有权限访问所有数据流) |
| HTTP内容 | {  "datastreams":[  {  "id":"temperature",  "datapoints":[  {  *//参数可选*  "at":"2013-04-22T00:35:43",  "value":42  },  {"at":"2013-04-22T00:55:43","value":84}  ]  }，  {  "id":"key",  "datapoints":[  {"at":"2013-04-22T00:35:43","value":{…}},  {"at":"2013-04-22T00:35:43","value":{…}}  ]  }，{…}  ]  } |
| 请求返回 | {  "errno": 0,  "error":“succ”,  } |

说明：

1. 可以一次性向设备云上传多个数据流，每个数据流中有可以包括多个数据点。示例程序表示向设备云上数据流temperature和数据流key上传数据。其中temperature数据流有两个数据点，key数据流也有两个数据点。
2. 如果某个数据流不存在，也就是事先没有通过4.4.1节中新增数据流的API创建本数据流，则在增加数据点时，设备云会自动创建一个新的数据流，注意新数据流的streamid不可为空。
3. 数据点表示在某个时刻，该数据流的值为多少。其at表示时间，为可选字段。如果为空，则设备云会取当前时间。如果存在其格式必须为"2013-04-22T00:35:43"的形式。
4. 数据点中的value表示具体的值，其值为JSON对象，可以为整型、字符串多种类型。如果value部分用双引号括起来，则其在设备云存储为字符串，如果不用双引号括起来，则其表示数字。
5. 数据点对应的时间是数据点的标识的一部分，相同时间的两个数据点，后一个会把前一个覆盖。一般如果at部分取空，设备云会取系统时间，精度为毫秒，如果在一条命令中某一个数据流包含多个数据点都没有at字段，会导致设备云只保存了最后一条。
6. 为了节省流量，增加数据点支持如下简写方式。在请求的URL中增加type字段，指示所使用的简写方式。目前type支持3，4，5三种情况
   * *http://*<API\_ADDRESS>*/devices/<device\_id>/datapoints?type=3*

{“temperature”:22.5,”humidity”:”95.2%”}

表示在数据流temperature中增加一个数据点22.5，在humidity中增加一个数据点95.2%。

* + *http://*<API\_ADDRESS>*/devices/<device\_id>/datapoints?type=4*

{“temperature”:{“2015-03-22T22:31:12”:22.5}}

表示在数据流temperature中加一个数据点，在2015年2月22日22点31分12秒的值为22.5

* + *http://*<API\_ADDRESS>*/devices/<device\_id>/datapoints?type=5*

,;temperature,2015-03-22T22:31:12,22.5;102;pm2.5,89;10

**补充说明：**

消息中最前面两位为用户自定义的域中分隔符和域间分隔符，这两个分隔符不能相同。比如采用逗号作为域中分隔符，分号作为域间分隔符的格式如下：

,;feild0;feild1;…;feildn

其中，每个field格式支持3种，下面以逗号作为域中分隔符进行说明：

field格式1： 3个子字段，分别是数据流ID,时间戳，数据值。通用格式：

Datastream\_id,datetime,value

field格式2： 2个子字段，分别是数据流ID和数据值，省略时间戳。通用格式：

Datastream\_id,value

field格式3： 1个子字段，省略了数据ID和时间戳，只传输数据值，平台将用该域所在的位置号（从0开始）作为数据流ID。通用格式：

注意：此格式中的数据点值均视为字符串。

示例：

(1),;temperature,2015-03-22T22:31:12,22.5;102;pm2.5,89

此段数据一共包括3个数据点，改写为基本格式为：

|  |
| --- |
| {  “datastreams”:[  {  “id”:”temperature”,  “datapoints”:  {  “at”:”2015-03-22T22:31:12”,  “value”:”22.5”  }  }  {  “id”:”1”,  “datapoints”:  {  “value”:”102”  }  }  {  “id”:”pm2.5”,  “datapoints”:  {  “value”:”89”  }  }  ]  } |

### 更新

暂不支持。

### 查看

注：

设备云识别的GPS型数据格式（value）：

{“lon”:33.2,”lat”:23,”ele”:222}

二进制数据数据索引存储格式（value）：

{ “index”:”FW3SO72FZIF3W”,“size”:2333(字节), ”desc”:字符串或json对象(根据上传时的desc参数确定)}

查看数据点时通过HTTP Get请求在URL中携带参数的形式实现的。

|  |  |
| --- | --- |
| HTTP方法 | GET |
| URL | http://<API\_ADDRESS>/devices/<device\_id>/datapoints |
| URL参数 | 接口参数包括必选的模式选项和可选的公共参数。  可选的公共参数：  1.datastream\_id:涉及的数据流：datastream\_id=<datastream\_id>,多个数据流之间用逗号分隔；  2.start:表示提取数据点的开始时间，格式为2015-01-10T08:00:35  3.end:表示提取数据点的结束时间，格式为2015-01-10T08:00:35  4.limit:限定本次请求最多返回的数据点的数量，取值>0，<=6000  5.cursor:指定本次请求继续从cursor位置开始提取数据  6.interval:指定通过采样方式返回数据点，interval值指定采样的时间间隔  7.method:指定在返回数据点时，同时返回统计结果，可能的值为sum/avg/min/max  8.first:返回结果中最值的时间点。1表示最早的时间，0表示最近的时间，默认为1 |
| HTTP头部 | api-key:xxxx-ffff-zzzzz， 需要设备级别的key |
| 请求返回 | {  "errno": 0,  "error":“succ”,  "data":  {  "cursor":"XXAABBCCDD"，  "datastreams":[  {  "id":"temperature",  "datapoints":[  {"at":" xxxx-xx-xx 10:22:22","value":42},  {"at":" xxxx-xx-xx 10:22:22","value":84}  ],  *//数据流统计信息*  "statistic":{  *// 求最值时返回的一个最值时间点*  "at":"xxxx-xx-xx 10:22:22",  "value":42  }  }，  {  "id":"key",  "datapoints":[  {"at":" xxxx-xx-xx 10:22:22","value":{…}},  {"at":"xxxx-xx-xx 10:22:22","value":{…}}  ]  }，{…}]  }  } |

说明：

1. 不携带任何参数，直接调用。则返回本设备所存在的所有数据流中最新的数据。

如果这个设备有三个数据流，则返回这三个数据流中每个数据流中最后一条数据。http://api.heclouds.com/devices/564280/datapoints 表示查询设备id为564280对应的所有数据流的最新数据点。

1. 不携带数据流id参数，携带limit参数时，会返回该设备每个数据流最多limit条数据。

http://api.heclouds.com/devices/564280/datapoints?limit=10，假设说设备564280有三个数据流，那么这三个流中每个流都会最多返回10个数据点。

1. 要查看某一条数据流数据，在上述URL中增加参数datastream\_id，如下所示：http://api.heclouds.com/devices/564280/datapoints?datastream\_id=temperature

astream\_id=后面的数据流的名称不加双引号。

会返回数据流temperature里时间最新的一个数据点

1. 用http://api.heclouds.com/devices/564280/datapoints?datastream\_id=温度2&limit=10会返回数据流”温度2”中最近的10个数据点。
2. 要查看某一条数据流在某个时间范围内的数据，可以在增加start和end参数。

http://api.heclouds.com/devices/564280/datapoints?datastream\_id=temperature&start=2013-05-12T17:12:33&end=2013-06-12T17:22:33

表示查询本设备对应的数据流在2013年5月12日17点12分33秒到17点22分33秒的数据点。注意如果start参数存在，end参数不存在，表示取start后的所有数据；如果start不存在，end存在，设备云会忽略end参数。注意：日期时间格式格式必须为示例中的格式：如2013-05-12T17:12:33, 日期和时间之间用大写字母T分隔。start和end之间的时间间隔最大为1年，超过一年会忽略。

1. 如果指定了start参数，则可能返回的数据点的数目会很多，此时默认会返回最多100个数据点。可以使用limit参数，设定返回多少个数据点，最大为6000个数据点。当实际的数据点数目多于limit限定的数目时，返回的json串中会有一个cursor字段，下一次请求的命令行中可以携带此cursor字段表示接着遍历本数据流。此cursor字段标识上次取出数据点后下一个数据点的位置。
2. method参数用于只有当数据流的value为整数时有效。如果为字符串或者其他类型，则无效。此方法用于返回一个统计值。当前支持sum/avg/min/max，当指定此参数时，返回的结果在响应的json字符串中的statistic部分，否则没有statistic部分.
3. 采样参数interval参数单位为秒，表示间隔多长时间提取一个数据点。

示例一：

|  |  |
| --- | --- |
| 请求URL | http://api.heclouds.com/devices/564280/datapoints? datastream\_id=温度2&start=2015-11-30T17:12:33&end=2015-12-01T17:22:33 |
| 响应结果 | {"errno":0,"data":{"count":26,"datastreams":[{"datapoints":[  {"at":"2015-12-01 17:06:23.535","value":"39"},  {"at":"2015-12-01 17:06:34.488","value":"39"},  {"at":"2015-12-01 17:06:35.935","value":"39"},  {"at":"2015-12-01 17:09:35.918","value":"20"},  {"at":"2015-12-01 17:09:55.915","value":"30"},  {"at":"2015-12-01 17:10:24.981","value":"35"},  {"at":"2015-12-01 17:10:53.406","value":"38"},  {"at":"2015-12-01 17:11:24.815","value":"48"},  {"at":"2015-12-01 17:11:58.126","value":"58"},  {"at":"2015-12-01 17:12:22.925","value":"68"},  {"at":"2015-12-01 17:12:32.173","value":"78"},  {"at":"2015-12-01 17:13:36.797","value":"38"},  {"at":"2015-12-01 17:13:58.436","value":"48"},  {"at":"2015-12-01 17:14:03.500","value":"18"},  {"at":"2015-12-01 17:14:11.332","value":"78"},  {"at":"2015-12-01 17:14:15.983","value":"18"},  {"at":"2015-12-01 17:15:51.763","value":"18"},  {"at":"2015-12-01 17:15:56.768","value":"19"},  {"at":"2015-12-01 17:16:01.089","value":"13"},  {"at":"2015-12-01 17:16:04.720","value":"23"},  {"at":"2015-12-01 17:16:26.593","value":"13"},  {"at":"2015-12-01 17:16:30.998","value":"33"},  {"at":"2015-12-01 17:16:35.592","value":"43"},  {"at":"2015-12-01 17:16:40.401","value":"23"},  {"at":"2015-12-01 17:17:41.498","value":"aa"},  {"at":"2015-12-01 17:18:08.904","value":"10"}],"id":"温度2"}]},"error":"succ"} |

示例二：

|  |  |
| --- | --- |
| 请求URL | http://api.heclouds.com/devices/564280/datapoints? datastream\_id=温度2&start=2015-11-30T17:12:33&end=2015-12-01T17:22:33&limit=5 |
| 响应结果 | {"errno":0,"data":{"cursor":"25971\_564280\_1448961024981","count":5,"datastreams":[{"datapoints":[  {"at":"2015-12-01 17:06:23.535","value":"39"},  {"at":"2015-12-01 17:06:34.488","value":"39"},  {"at":"2015-12-01 17:06:35.935","value":"39"},  {"at":"2015-12-01 17:09:35.918","value":"20"},  {"at":"2015-12-01 17:09:55.915","value":"30"}],"id":"温度2"}]},"error":"succ"} |

示例三：

|  |  |
| --- | --- |
| 请求URL | http://api.heclouds.com/devices/564280/datapoints? datastream\_id=温度2&start=2015-11-30T17:12:33&end=2015-12-01T17:22:33&limit=5&cursor=25971\_564280\_1448961024981 |
| 响应结果 | {"errno":0,"data":{"cursor":"25971\_564280\_1448961152173","count":5,"datastreams":[{"datapoints":[  {"at":"2015-12-01 17:10:24.981","value":"35"},  {"at":"2015-12-01 17:10:53.406","value":"38"},  {"at":"2015-12-01 17:11:24.815","value":"48"},  {"at":"2015-12-01 17:11:58.126","value":"58"},  {"at":"2015-12-01 17:12:22.925","value":"68"}],"id":"温度2"}]},"error":"succ"} |

### 删除

数据点不支持删除操作，默认数据在设备云上保存一年，过期自动淘汰，后期依据现网情况，可能会延长保存年限。

如果把设备删掉，则设备对应的所有数据点都会被删掉，由于调度的关系，删除动作的具体执行时间会有一定的延迟。

## 触发器（trigger）

### 新增

|  |  |
| --- | --- |
| HTTP方法 | POST |
| URL | http://<API\_ADDRESS>/triggers |
| HTTP头部 | api-key:xxxx-ffff-zzzzz，必须为Master Key |
| HTTP内容 | {  *//可选*  "title":"wen du jian kong",  *//可选，数据流名称（id）*  "ds\_id":"gps",  *//可选，设备ID*  "dev\_ids":["1027","23456","15523"],  *//可选，数据流uuid*  "ds\_uuids":[  "datastream1\_uuid”,  "datastream2\_uuid”,  "datastream3\_uuid”  ],  "url":"http://xx.bb.com",  "type":">|>=|<|<=|==|inout|in|out|exp|change|frozen|live",  *// 根据type不同，见以下说明*  "threshold ":100  }  注：   1. type为>|>=|<|<=|==时，threshold必须为数值 2. type为inout时，threshold设置为{“lolmt”:40, “uplmt”:52}，表示数据流的值首次进入或离开闭区间[40,52]时触发； 3. type为exp时,threshold设置为字符串类型的条件表达式，$val[0]表示第一个数据流的当前值，$val[1]为第二个…。例如，第一个数据流上报数据点格式为：{“temperature”:22, “humidity”:56},若需设置温度大于30，且湿度小于33时触发告警，则threshold中条件表达式可设置为：“$val[0][‘temperature’] > 30 && $val[0][‘humidity’]< 33 ”； 4. type为change时，threshold 参数不用传递；当上传的值有改变时触发告警。 5. type为frozen时，threshold 为数值，指定多少秒内未上报数据触发告警，同时被监控对象进入frozen状态。 6. type为live时，threshold要传递；(要传值大于0的数字)被监控对象在frozen状态下收到上报的数据点时，触发告警。 |
| 请求返回 | {  "errno": 0,  "error":“succ”,  "data":  {  "trigger\_id":233444  }  } |

说明：

1、触发器的含义是：当指定范围内的数据点满足触发条件的要求时，会向url参数指定的地址发送post请求，post的消息内容请参考4.10.1

2、触发器有三种工作触发模式

1. 在请求参数中单独指定了ds\_id，不包括其他参数，那么当前项目下所有设备的数据流中数据流名称符合ds\_id的数据都会进入触发判断逻辑；
2. 在请求参数中单独指定了ds\_uuids数组，那么只有这些指定的数据流会进入触发判断逻辑；
3. 在请求参数中指定了ds\_id和dev\_ids，那么只有这些指定的设备的数据流会进入触发判断逻辑。

### 更新

根据更新参数，全量设置对应项目。

|  |  |
| --- | --- |
| HTTP方法 | PUT |
| URL | http://<API\_ADDRESS>/triggers/<trigger\_id> |
| HTTP头部 | api-key:xxxx-ffff-zzzzz，需要master Key |
| HTTP内容 | {  "title":"wen du jian kong",  "ds\_uuids":[  "datastream1\_uuid” ,  "datastream2\_uuid” ,  "datastream3\_uuid”  ],  "url":"http://xx.bb.com",  "type":">|>=|<|<=|==|in|out|exp|change|frozen|live ",  "threshold":100  }  ***注：若要修改触发条件，必须同时设置type和threshold*** |
| 请求返回 | {  "errno": 0,  "error":“succ”  } |

### 查看

A 单个查看

|  |  |
| --- | --- |
| HTTP方法 | GET |
| URL | http://<API\_ADDRESS>/triggers/<trigger\_id> |
| HTTP头部 | api-key:xxxx-ffff-zzzzz，需要master Key |
| 请求返回 | {  "errno": 0,  "error":“succ”,  "data":{  "id":trigger\_id1,  "title":"wen du jian kong",  "ds\_uuids":[  "datastream1\_uuid” ,  "datastream2\_uuid” ,  "datastream3\_uuid”  ],  "url":"xx.bb.com",  "type":">|>=|<|<=|==|in|out|exp|change|frozen|live ",  "threshold":100,  *//触发器是否已失效*  "invalid":true,  “create\_time”:"xx-xx-xx 10:22:22"  }  } |

b批量查看

|  |  |
| --- | --- |
| HTTP方法 | GET |
| URL | http://<API\_ADDRESS>/triggers |
| URL参数 | title = name *//指定触发器名称*  page = 1 *//指定页码,可选*  per\_page = 10 *//指定每页输出个数,可选,默认10，最多100* |
| HTTP头部 | api-key:xxxx-ffff-zzzzz |
| 请求返回 | {  "errno": 0,  "error":“succ”,  "data":{  "total\_count":600,  "per\_page":30,  "page":1,  "triggers":[{  "id":trigger\_id1,  "title":"wen du jian kong",  "ds\_uuids":["datastream1\_uuid” ,"datastream2\_uuid”],  "url":"xx.bb.com",  "type":">|>=|<|<=|==|in|out|exp|change|frozen|live ",  "threshold":100,  *//触发器是否已失效*  "invalid":true,  “create\_time”:XXXXXXXX  }]  }  } |

说明：

* 触发器失效是指触发器所依赖的数据流或者设备被删除了。此时触发器永远不会有效触发事件

### 删除

|  |  |
| --- | --- |
| HTTP方法 | DELETE |
| URL | http://<API\_ADDRESS>/triggers/<trigger\_id> |
| HTTP头部 | api-key:xxxx-ffff-zzzzz， 需要master Key |
| HTTP内容 | 无 |
| 请求返回 | {  "errno": 0,  "error":“succ”  } |

## API 权限(apikey)

### 新增

|  |  |
| --- | --- |
| HTTP方法 | POST |
| URL | http://<API\_ADDRESS>/keys |
| HTTP头部 | api-key:xxxx-ffff-zzzzz  (注：必须是master-key) |
| HTTP内容 | {  "title":"sharing key",  "permissions":[  {  "resources":[  {  "dev\_id":"424"  }  ]  "access\_methods": [  "get",  "put",  "post"  ]  }  ]  } |
| 请求返回 | {  "errno": 0,  "error":“succ”,  "data":  {  "key":"FJWEFOWFNWBSNOQ"  }  } |

### 更新

根据更新参数，全量设置对应项目。

|  |  |
| --- | --- |
| HTTP方法 | PUT |
| URL | http://<API\_ADDRESS>/keys/<key\_string> |
| HTTP头部 | api-key:xxxx-ffff-zzzzz  （注：只能为master-key） |
| HTTP内容 | {  "title":"sharing key",  "permissions":[  {  *//被授权的资源*  "resources":[  {  "dev\_id":"424",  *//数据流id，可选*  "ds\_id":"fan1"  }],  *// 被授权的命令，可选。如果没有，表示可以进行任何操作*  "access\_methods": ["get","put","post" ]  }]  } |
| 请求返回 | {  "errno": 0,  "error":“succ”  } |

说明：对于master key不允许更新操作，masterkey默认具有最大权限，可以访问该用户名下所有设备、所有数据，可以进行POST、GET、PUT、DELETE操作。如果更新master key系统会返回失败。

### 查看

|  |  |
| --- | --- |
| HTTP方法 | GET |
| URL | http://<API\_ADDRESS>/keys |
| URL参数 | key = FWJPO-FZFHW-FFEWB,//可选，只查看该key相关信息  page = 1 //指定页码, 可选  per\_page = 10 //指定每页输出个数,可选,默认10，最多100  device\_id = xxbbcc,//可选,只查看与该设备相关的非master-key |
| HTTP头部 | api-key:xxxx-ffff-zzzzz  （注：只能为master-key） |
| 请求返回 | {  "errno": 0,  "data": {  "per\_page": 30,  "total\_count": 2,  "keys": [  {  "create\_time": "2015-11-04 14:25:52",  "title": "Auto-generated-master-key",  "key": "03GeDLYk0WNk1V338g7yndKZYkQA"  },  {  "create\_time": "2015-12-03 10:00:41",  "permissions": [  {  "resources": [  {  "dev\_id": "564280"  }  ]  }  ],  "title": "设备1测试Key",  "key": "vuHebEWefiU2Ueo5ouoCc1CUsYQ="  }  ],  "page": 1  },  "error": "succ"  } |

### 删除

|  |  |
| --- | --- |
| HTTP方法 | DELETE |
| URL | http://<API\_ADDRESS>/keys/<key\_string> |
| HTTP头部 | api-key:xxxx-ffff-zzzzz  (注：只能为master-key) |
| HTTP内容 | 无 |
| 请求返回 | {  "errno": 0,  "error":“succ”  } |

## 二进制数据（bindata）

### 新增

功能：将二进制的数据保存到设备云，设备云返回该二进制数据的索引；并根据参数，将二进制数据作为数据点保存到设备下的某个数据流。

|  |  |
| --- | --- |
| HTTP方法 | POST |
| URL | http://<API\_ADDRESS>/bindata |
| HTTP头部 | api-key:xxxx-ffff-zzzzz，数据流 Key |
| URL参数 | device\_id，必选，该数据所属设备。  datastream\_id，必选，该数据所属数据流。 |
| HTTP内容 | 普通二进制数据、文件、图像 |
| 请求返回 | {  "errno": 0,  "error":“succ”,  "data":{  *//该数据在设备云的索引*  "index":“AJOWFW1133OFO2Z93”  }  } |

### 查看

功能：读取二进制数据的内容。若查询没有该数据返回HTTP 404,并在返回内容中指定{errno,error}

|  |  |
| --- | --- |
| HTTP方法 | GET |
| URL | http://<API\_ADDRESS>/bindata/<index> |
| HTTP头部 | api-key:xxxx-ffff-zzzzz，可以为数据流Key |
| URL参数 |  |
| HTTP内容 |  |
| 请求返回 | 普通二进制数据、文件、图像 |

### 删除

二进制数据点不支持删除操作，默认数据在设备云上保存一年，过期自动淘汰，后期依据现网情况，可能会延长保存年限。

如果把设备删掉，则设备对应的所有数据点都会被删掉，由于调度的关系，删除的时间会有一定的延迟。

## 命令执行（cmd）

命令是指应用发送命令数据给OneNet设备云、再由OneNet设备云转发给终端设备、终端设备收到命令后可以向设备云反馈响应的一种工作机制。命令下发后，应用可以通过API查询命令状态和提取命令的响应数据。

命令的执行步骤如下：

1. 应用通过API向设备云发送命令数据；
2. 设备云生成该命令的uuid作为该命令的唯一标识并向应用返回，同时向终端设备转发命令数据；
3. 终端设备收到命令数据后执行命令数据，可以生成响应，也可以不响应；
4. 设备云跟踪记录命令的响应，如果设备有响应，就关联命令uuid和响应结果；
5. 应用通过API向设备云提取命令执行结果(API请求中携带命令uuid标识)。

特别说明：

命令下发到终端设备后，如果终端设备10秒内未收到响应，则此命令的状态将被设置为超时，应用通过API查询命令状态时，会反馈超时。超过10秒后收到的响应会被丢弃。

终端设备针对同一条命令回复多条命令响应，只有在未超时时间内收到的第一条是有效响应，后续的响应包将会被直接丢弃。因此终端设备应该对每个命令只有一个响应包。

### 发送命令

用于发送数据到登录设备云的设备。

|  |  |
| --- | --- |
| HTTP方法 | POST |
| URL | http://<API\_ADDRESS>/cmds |
| HTTP头部 | api-key:xxxx-ffff-zzzzz，必须master key |
| URL参数 | device\_id = //接收该数据的设备ID，必填。 |
| HTTP内容 | 用户自定义Json或二进制数据（小于64K） |
| 成功返回 | {  "errno": 0,  "error":“succ”，  "data":{  *//不超过64个字符字符串*  "cmd\_uuid":“2302-312-FWs”  }  } |

### 查看状态

用于查看命令发送状态

|  |  |
| --- | --- |
| HTTP方法 | GET |
| URL | http://<API\_ADDRESS>/cmds/<cmd\_uuid> |
| HTTP头部 | api-key:xxxx-ffff-zzzzz，必须master key |
| 成功返回 | {  "errno": 0,  "error":“succ”，  "data":{  *//0：设备不在线|Device not online*  *1：命令已创建| Command Created*  *2：命令已发往设备| Command Sent*  *3：命令发往设备失败| Send Command Failed*  *4：设备正常响应| Command Response Received*  *5：命令执行超时| Command Response Timeout*  *6：设备响应消息过长 | Command Response Too Large*  "status":1  "desc":“sending”  }  } |

### 获取响应

用于获取终端执行命令后的响应内容。

|  |  |
| --- | --- |
| HTTP方法 | GET |
| URL | http://<API\_ADDRESS>/cmds/< cmd\_uuid >/resp |
| HTTP头部 | api-key:xxxx-ffff-zzzzz， 必须master key |
| HTTP body | 用户自定义Json或二进制数据（小于64K）  设备返回的命令执行响应 |

响应的HTTP状态码对应的错误信息如下

|  |  |
| --- | --- |
| 400 | No api-key |
| 403 | Auth Failed |
| 404 | Not Found |
| 400 | Invalid Parameter |
| 220 | Command Created |
| 221 | Command Sent |
| 520 | Send Command Failed |
| 200 | Command Response Received |
| 432 | Command Response Timeout |
| 433 | Command Response Too Large |

## 其他

### 触发数据内容

触发数据是指当OneNet设备云判断某个触发器的触发条件满足时，会通过POST方式向外抛出的URL请求，在HTTP的POST消息体中会携带如下格式的JSON字符串。可以结合4.6.1节中的内容来理解。

|  |
| --- |
| {  "trigger":{  "id":1,  "threshold":9.0,  "type":">"  },  "current\_data":[  {  "dev\_id":"2656",  "ds\_id":"gps",  "ds\_uuid":"FWEF-ZFW-ZFW-ZFWW",  "at":"2009-09-07T12:16:02.001403Z",  "value":100  },…  {  "dev\_id":"2656",  "ds\_id":"temper",  "ds\_uuid":"FEWEF-ZFWFWW-WFWW-FWEFFWW",  "at":"2009-09-07T12:16:02.001403Z",  "value":88.2  } ]  } |

### 历史数据查询

说明：根据条件查询该用户下的数据点记录。

|  |  |
| --- | --- |
| HTTP方法 | GET |
| URL | http://<API\_ADDRESS> /datapoints |
| URL参数 | page=1 *//指定页码, 可选*  device\_id=<device\_id>,*//指定设备ID,可选*  datastream\_id=<datastream\_id>,*//指定数据流,可选*  start=2012-06-02T14:01:46,*//指定开始时间,必选*  end=2012-06-02T14:01:46,*//指定结束时间,可选，间隔最大6小时*  duration=600,*//时间间隔，可选，默认600，单位秒，最大6小时*  limit=100,*// datapoint个数，默认10，最大6000*  cursor=XXXXXXX,*//查询条件返回结果超过limit条数时遍历的游标。*  使用示例，查询该用户下，所有设备gps数据流中的数据点：  http://api.onenet.com /datapoints?datastream\_id=gps |
| HTTP头部 | api-key:xxxx-ffff-zzzzz **（必须使用master\_key）** |
| 请求返回 | {  "errno": 0,  "error":“succ”,  "data":  {  "cursor":"XXAABBCCDD"，  "datapoints":[  {  "dev\_id":"10313",//设备ID  "ds\_id":"gps",//数据流ID  "at":" xxxx-xx-xx 10:22:22",//数据点时间  "value":{"lat":23.1,"lon":89.2","ele":200}，  },{  "dev\_id":"2656",  "ds\_id":"gps",  "at":" xxxx-xx-xx 10:22:22",  "value":{"lat":23.1,"lon":69.2","ele":120}，  },{…}]  }  } |

说明：本操作必须使用masterkey

### 获取接入机的IP的端口

|  |  |
| --- | --- |
| HTTP方法 | GET |
| URL | http://<API\_ADDRESS> /s |
| URL参数 | t=1 *//接入机类型，必选*  v=””,*//版本号，选填，默认最新* |
| HTTP内容 |  |
| 请求返回 | 183.230.40.42:2987 //接入机ip:port |
| 说明 | \* t 接入机类型  // EDP 1  // nwx 2  // jtext 3  // Hiscmd 4  // jt808 5  // modbus 6  // mqtt 7  // gr20 8  // reg 9 |