ALIBABA CLOUD

阿里云

物联网络管理平台

最佳实践

文档版本：20210427

阿里云

法律声明

阿里云提醒您在阅读或使用本文档之前仔细阅读、充分理解本法律声明各条款的内容。如果您阅读或使用本文档，您的阅读或使用行为将被视为对本声明全部内容的认可。

1. 您应当通过阿里云网站或阿里云提供的其他授权通道下载、获取本文档，且仅能用于自身的合法合规的业务活动。本文档的内容视为阿里云的保密信息，您应当严格遵守保密义务；未经阿里云事先书面同意，您不得向任何第三方披露本手册内容或提供给任何第三方使用。

2. 未经阿里云事先书面许可，任何单位、公司或个人不得擅自摘抄、翻译、复制本文档内容的部分或全部，不得以任何方式或途径进行传播和宣传。

3. 由于产品版本升级、调整或其他原因，本文档内容有可能变更。阿里云保留在没有任何通知或者提示下对本文档的内容进行修改的权利，并在阿里云授权通道中不时发布更新后的用户文档。您应当实时关注用户文档的版本变更并通过阿里云授权渠道下载、获取最新版的用户文档。

4. 本文档仅作为用户使用阿里云产品及服务的参考性指引，阿里云以产品及服务的“现状”、“有缺陷”和“当前功能”的状态提供本文档。阿里云在现有技术的基础上尽最大努力提供相应的介绍及操作指引，但阿里云在此明确声明对本文档内容的准确性、完整性、适用性、可靠性等不作任何明示或暗示的保证。任何单位、公司或个人因为下载、使用或信赖本文档而发生任何差错或经济损失的，阿里云不承担任何法律责任。在任何情况下，阿里云均不对任何间接性、后果性、惩戒性、偶然性、特殊性或刑罚性的损害，包括用户使用或信赖本文档而遭受的利润损失，承担责任（即使阿里云已被告知该等损失的可能性）。

5. 阿里云网站上所有内容，包括但不限于著作、产品、图片、档案、资讯、资料、网站架构、网站画面的安排、网页设计，均由阿里云和/或其关联公司依法拥有其知识产权，包括但不限于商标权、专利权、著作权、商业秘密等。非经阿里云和/或其关联公司书面同意，任何人不得擅自使用、修改、复制、公开传播、改变、散布、发行或公开发表阿里云网站、产品程序或内容。此外，未经阿里云事先书面同意，任何人不得为了任何营销、广告、促销或其他目的使用、公布或复制阿里云的名称（包括但不限于单独为或以组合形式包含“阿里云”、“Aliyun”、“万网”等阿里云和/或其关联公司品牌，上述品牌的附属标志及图案或任何类似公司名称、商号、商标、产品或服务名称、域名、图案标示、标志、标识或通过特定描述使第三方能够识别阿里云和/或其关联公司）。

6. 如若发现本文档存在任何错误，请与阿里云取得直接联系。

通用约定

格式	说明	样例
危险	该类警示信息将导致系统重大变更甚至故障，或者导致人身伤害等结果。	危险 重置操作将丢失用户配置数据。
警告	该类警示信息可能会导致系统重大变更甚至故障，或者导致人身伤害等结果。	警告 重启操作将导致业务中断，恢复业务时间约十分钟。
注意	用于警示信息、补充说明等，是用户必须了解的内容。	注意 权重设置为0，该服务器不会再接受新请求。
说明	用于补充说明、最佳实践、窍门等，不是用户必须了解的内容。	说明 您也可以通过按Ctrl+A选中全部文件。
>	多级菜单递进。	单击**设置**> **网络**> **设置网络类型**。
粗体	表示按键、菜单、页面名称等UI元素。	在**结果确认**页面，单击**确定**。
Courier字体	命令或代码。	执行 `cd /d C:/window` 命令，进入Windows系统文件夹。
斜体	表示参数、变量。	`bae log list --instanceid` *Instance_ID*
[] 或者 [a|b]	表示可选项，至多选择一个。	`ipconfig [-all|-t]`
{} 或者 {a|b}	表示必选项，至多选择一个。	`switch {active|stand}`

目录

1.LoRaWAN温湿度传感器方案

本文提供LoRaWAN温湿度传感器通过**Link WAN**接入，同时采用**阿里云物联网平台**实现端到端应用的方案。

背景信息

- 开通物联网络管理平台

  完成账号的注册之后，使用账号登录Link WAN 开通服务。

- LoRa节点设备接入
- 搭建与管理网络

  参见搭建与管理网络搭建和管理网络、创建节点组并添加节点。

- 配置数据流转

  目前数据流转支持阿里云物联网平台、消息队列MQ两种方式，这里选择将数据流转至阿里云物联网平台，详情请参见数据接入物联网平台-1对1。

- 物联网平台LoRa节点设备接入

  本章介绍如何在物联网平台开发平台上进行设备接入的开发。主要的开发内容包括：

  - 创建产品和设备
  - 产品功能定义
  - 平台脚本开发

本文以一个空气温湿度传感器为例，同时可以配置温湿度的阈值，在温湿度超出阈值时上报事件。

创建产品和设备

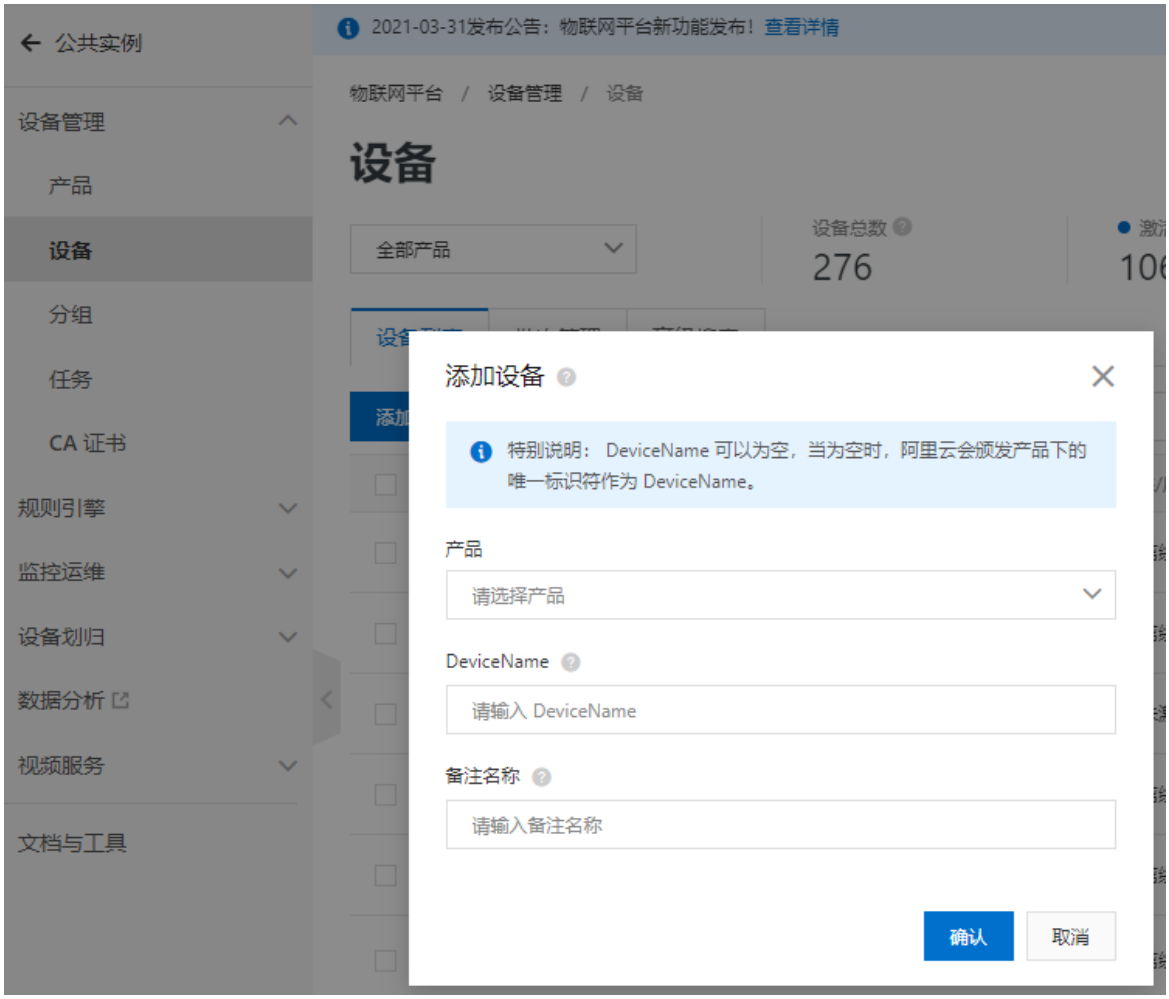
1. 登录物联网平台控制台。
2. 在**实例概览**页面，找到对应的实例，单击实例进入**实例详情**页面。
3. 在左侧导航栏上选择**设备管理 > 产品**，单击**创建产品**， 填写产品信息后单击**确认**。详情请参见数据接入物联网平台-1对1。

参数	描诉
产品名称	可填写任意名称
所属品类	自定义品类
节点类型	直连设备
连网方式	LoRaWAN
入网凭证	从表单选择。如无，可单击**创建凭证**

参数	描诉
数据格式	透传/自定义

4. 为产品添加LoRa设备。

   在左侧导航栏上单击**设备**，参见单个创建设备添加设备。



> **说明** 使用LoRaWAN设备的DevEUI需小写作为DeviceName。

- 添加完成后，显示如下，此时设备状态为**未激活**。

- 数据流转已自动同步。

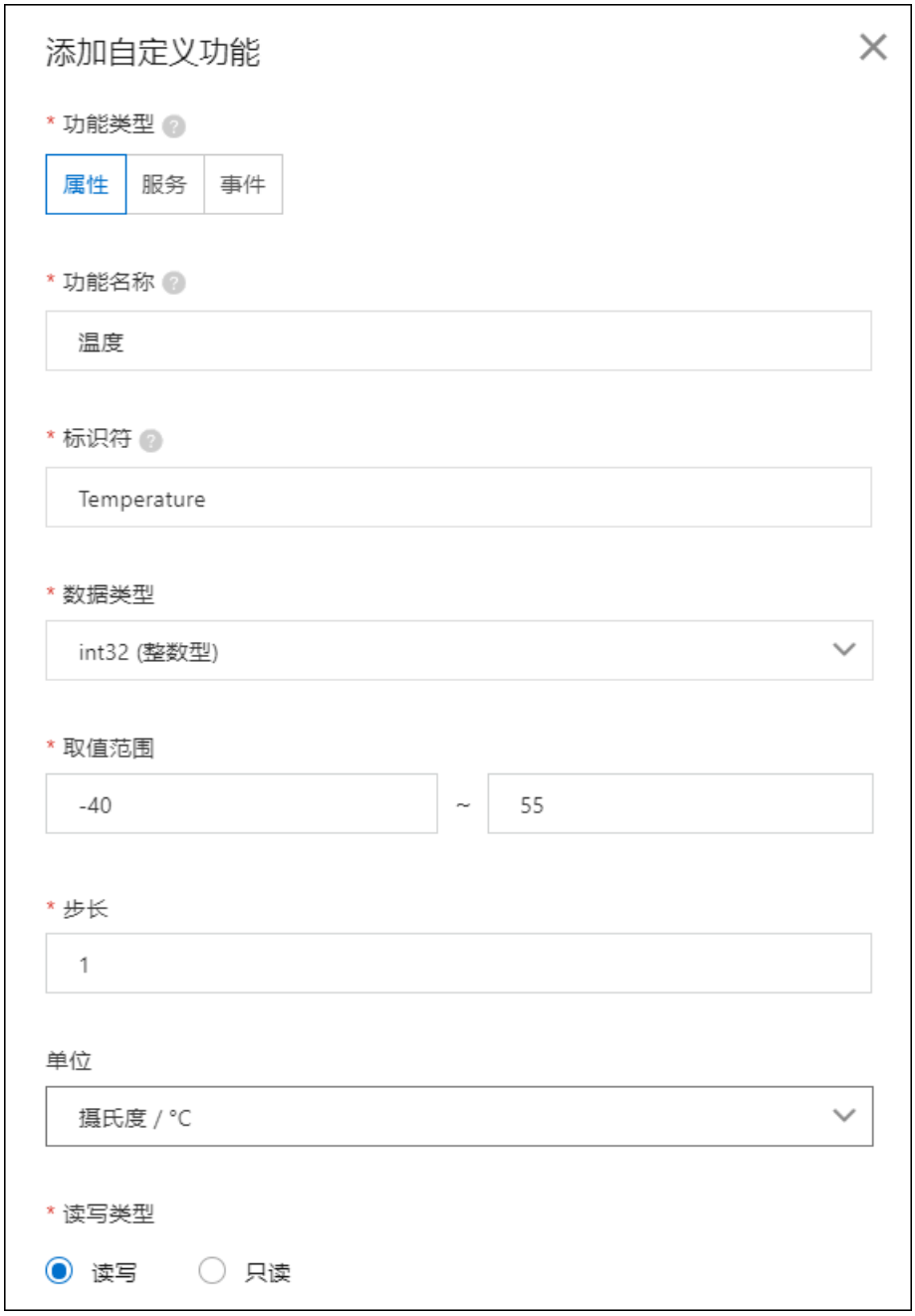
产品创建完成后，可在Link WAN里看到自动同步的数据流转设置。

说明 在网管平台只能查阅，新增终端请移步至物联网平台维护。

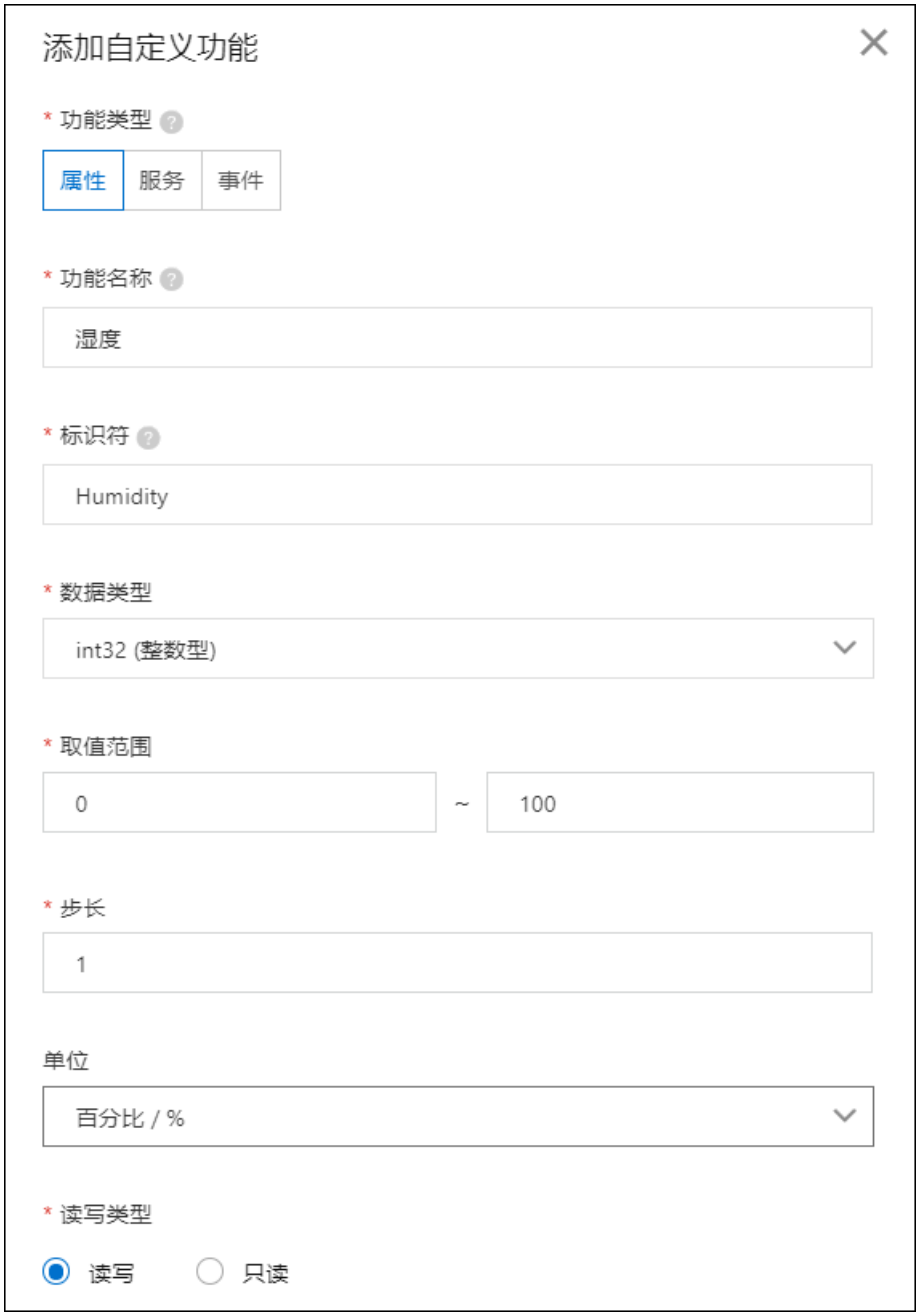
产品功能定义

产品创建完成之后，需要在平台上定义产品有哪些功能。功能定义是为了让平台能够理解设备上下行的数据定义，便于上层应用的读写。

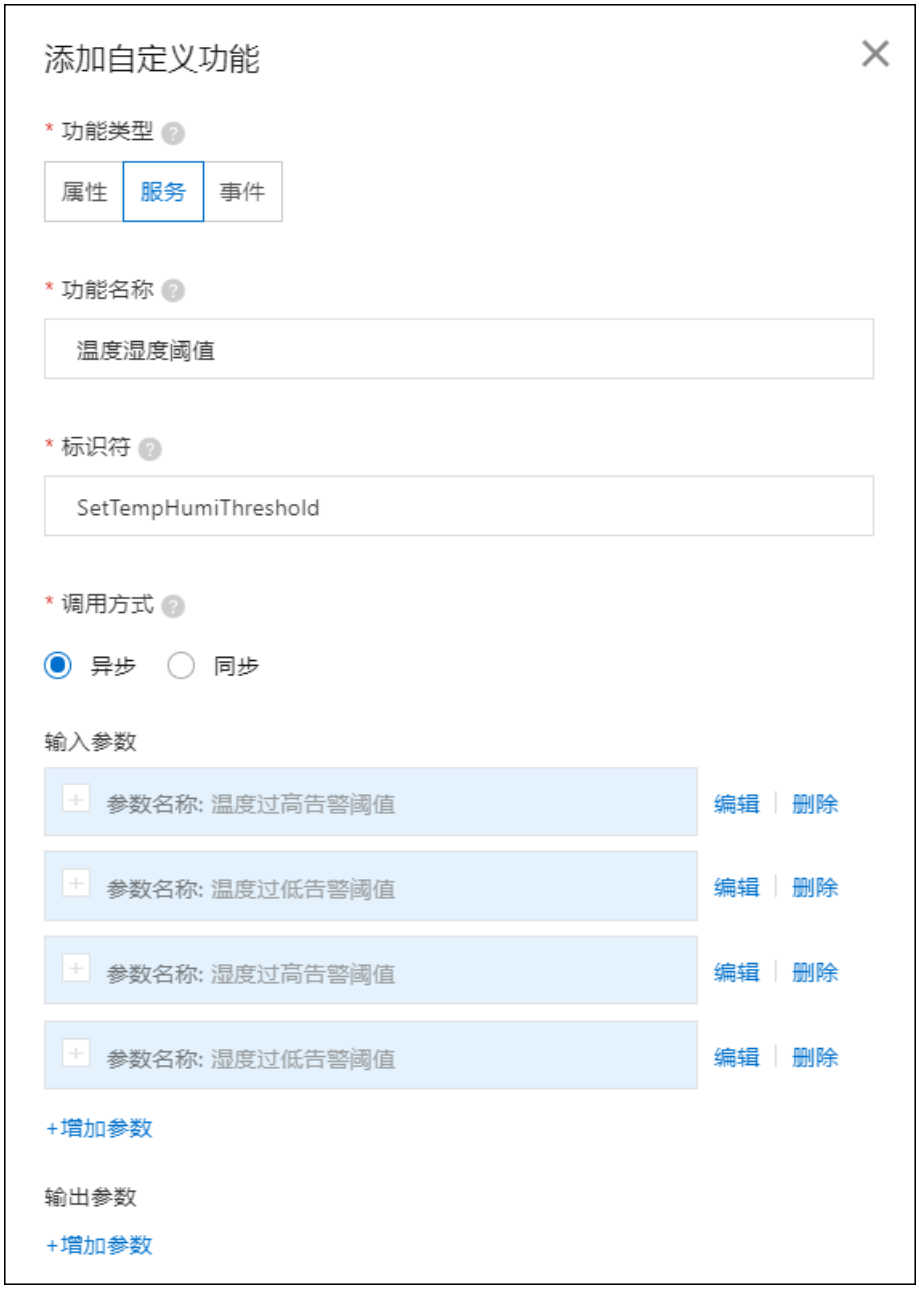
1. 在左侧导航栏上选择**设备管理 > 产品**，单击产品对应**操作**栏中的**查看**。
2. 选择**功能定义 > 编辑草稿**，单击**添加自定义功能**。
3. 在**添加自定义功能**弹框中，**功能类型**选择**属性**，添加温湿度属性。
   - 添加温度属性，配置参数如下图所示。



- 添加湿度属性，配置参数如下图所示。



4. **功能类型**选择**服务**，添加温度湿度阈值，参数配置如下图所示。



其中输入参数设置如下图所示。

参数名称	标志符	数据类型	取值范围	步长	单位
温度过高阈值	MaxTemp	int32（整数型）	-40~55	1	摄氏度/℃
温度过低阈值	MinTemp	int32（整数型）	40~55	1	摄氏度/℃

参数名称	标志符	数据类型	取值范围	步长	单位
湿度过高阈值	MaxHumi	int32（整数型）	1~100	1	百分比/%
湿度过低阈值	MinHumi	int32（整数型）	1~100	1	百分比/%

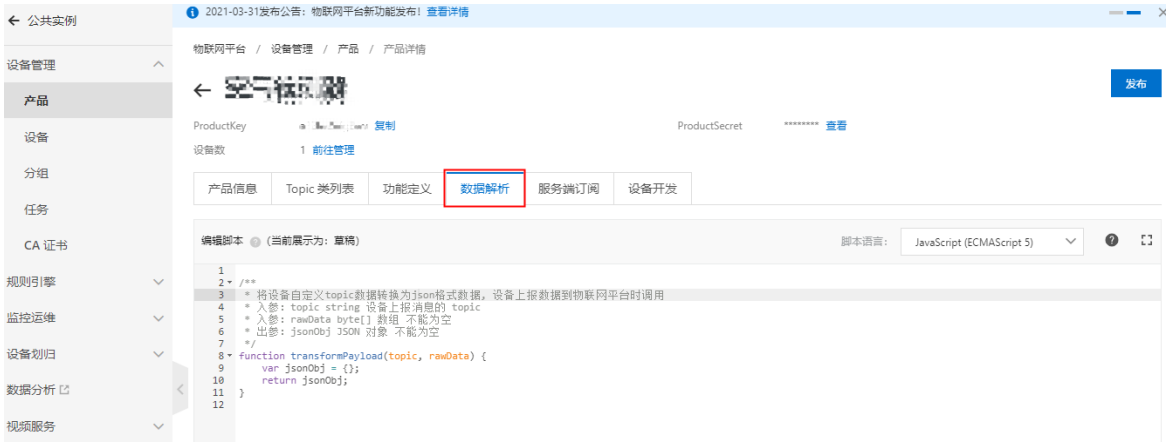
5. **功能类型**选择**事件**，添加湿度过高／过低告警事件。告警输出参数为当前湿度。

其中输出参数设置如下。

6. 单击**确认**，单击页面左下方的**发布更新**。
上述属性、服务、事件添加完成后，在自定义功能一栏下方可确认添加的结果。

平台脚本开发

1. 进入**产品**的**数据解析**标签页，可以添加解析脚本。由于数据是以自定义格式透传到平台，所以需要添加脚本来解析自定义协议。



2. 将下列代码添加到上图的**脚本编辑区**。

```
var ALINK_ID = "12345";
var ALINK_VERSION = "1.1";
var ALINK_PROP_POST_METHOD    = 'thing.event.property.post';
var ALINK_EVENT_TEMPERR_METHOD = 'thing.event.TempError.post';
var ALINK_EVENT_HUMIERR_METHOD = 'thing.event.HumiError.post';
var ALINK_PROP_SET_METHOD     = 'thing.service.property.set';
var ALINK_SERVICE_THSET_METHOD = 'thing.service.SetTempHumiThreshold';
/*
 * 示例数据:
 * 传入参数 ->
 *    000102 // 共3个字节
 * 输出结果 ->
 *    {"method":"thing.event.property.post", "id":"12345", "params":{"Temperature":1,"Humidity":2}, "
version":"1.1"}
 * 传入参数 ->
 *    0102 // 共2个字节
 * 输出结果 ->
 *    {"method":"thing.event.TempError.post","id":"12345","params":{"Temperature":2},"version":"1.1
"}
 * 传入参数 ->
 *    0202 // 共2个字节
 * 输出结果 ->
 *    {"method":"thing.event.HumiError.post","id":"12345","params":{"Humidity":2},"version":"1.1"}
 */
function rawDataToProtocol(bytes)
{
  var uint8Array = new Uint8Array(bytes.length);
  for (var i = 0; i < bytes.length; i++)
  {
    uint8Array[i] = bytes[i] & 0xff;
  }
  var params = {};
  var jsonMap = {};
  var dataView = new DataView(uint8Array.buffer, 0);
  var cmd = uint8Array[0]; // command
  if (cmd === 0x00)
  {
    params['Temperature']  = dataView.getInt8(1);
    params['Humidity']    = dataView.getInt8(2);
```

```
    jsonMap['method'] = ALINK_PROP_POST_METHOD;
  }
  else if (cmd == 0x01)
  {
    params['Temperature'] = dataView.getInt8(1);
    jsonMap['method'] = ALINK_EVENT_TEMPERR_METHOD;
  }
  else if (cmd == 0x02)
  {
    params['Humidity'] = dataView.getInt8(1);
    jsonMap['method'] = ALINK_EVENT_HUMIERR_METHOD;
  }
  else
  {
    return null;
  }
  jsonMap['version'] = ALINK_VERSION;
  jsonMap['id']    = ALINK_ID;
  jsonMap['params'] = params;
  return jsonMap;
}
/*
 * 示例数据:
 * 传入参数 ->
 *   {"method":"thing.service.SetTempHumiThreshold", "id":"12345", "version":"1.1", "params":{"Max
Temp":50, "MinTemp":8, "MaxHumi":90, "MinHumi":10}}
 * 输出结果 ->
 *   0x5d0a000332085a0a
 */
function protocolToRawData(json)
{
  var id = json['id'];
  var method = json['method'];
  var version = json['version'];
  var payloadArray = [];
  // 追加下行帧头部
  payloadArray = payloadArray.concat(0x5d);
  payloadArray = payloadArray.concat(0x0a);
  payloadArray = payloadArray.concat(0x00);
  if (method == ALINK_SERVICE_THSET_METHOD)
  {
    var params = json['params'];
    var maxtemp = params['MaxTemp'];
    var mintemp = params['MinTemp'];
    var maxhumi = params['MaxHumi'];
    var minhumi = params['MinHumi'];
    payloadArray = payloadArray.concat(0x03);
    if (maxtemp !== null)
    {
      payloadArray = payloadArray.concat(maxtemp);
    }
    if (mintemp !== null)
    {
      payloadArray = payloadArray.concat(mintemp);
```

```
    }
    if (maxhumi !== null)
    {
      payloadArray = payloadArray.concat(maxhumi);
    }
    if (minhumi !== null)
    {
      payloadArray = payloadArray.concat(minhumi);
    }
  }
  return payloadArray;
}
// 以下是部分辅助函数
function buffer_uint8(value)
{
  var uint8Array = new Uint8Array(1);
  var dv = new DataView(uint8Array.buffer, 0);
  dv.setUint8(0, value);
  return [].slice.call(uint8Array);
}
function buffer_int16(value)
{
  var uint8Array = new Uint8Array(2);
  var dv = new DataView(uint8Array.buffer, 0);
  dv.setInt16(0, value);
  return [].slice.call(uint8Array);
}
function buffer_int32(value)
{
  var uint8Array = new Uint8Array(4);
  var dv = new DataView(uint8Array.buffer, 0);
  dv.setInt32(0, value);
  return [].slice.call(uint8Array);
}
function buffer_float32(value)
{
  var uint8Array = new Uint8Array(4);
  var dv = new DataView(uint8Array.buffer, 0);
  dv.setFloat32(0, value);
  return [].slice.call(uint8Array);
}
```

脚本解析下行数据的函数protocolToRawData中必须设定输出结果的起始三个字节（用于指定下行的端口号以及下行消息类型），否则系统会丢掉下行帧。另外，节点实际接收到的数据将不会包含起始的三个字节。

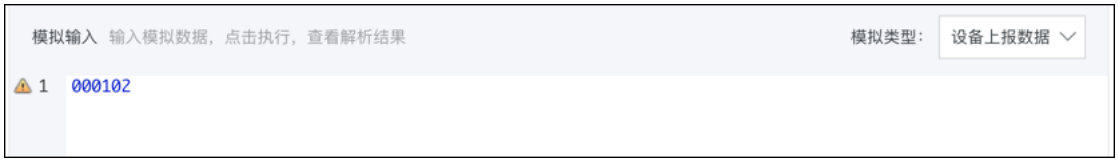
起始三字节的说明如下表所示。

Size（bytes）	LoRa Downlink	描述
1	DFlag	固定为0x5D
1	FPort	下行端口号

Size（bytes）	LoRa Downlink	描述
1	DHDR	○ 0表示 “Unconfirmed Data Down”数据帧 ○ 1表示 “Confirmed Data Down”数据帧

示例： `0x5D 0x0A 0x00` 表示：下行帧端口号为10，数据帧为Unconfirmed Data Down。

3. 脚本模拟运行。
   i. 设备上报数据调试。在**脚本调试区1**里输入下面数据，**模拟类型**选择**设备上报数据**后，单击**运行**按钮。



> **说明** 000102中的00表示后面的两个字节分别表示温度和湿度，01表示温度为1摄氏度，02表示湿度为2%。

ii. 设备接收数据调试。

在**脚本调试区1**里输入以下数据，**模拟类型**选择**设备接收数据**后，单击**运行**按钮。

```
{
  "method": "thing.service.SetTempHumiThreshold",
  "id": "12345",
  "version": "1.1",
  "params": {
    "MaxTemp": 50,
    "MinTemp": 8,
    "MaxHumi": 90,
    "MinHumi": 10
  }
}
```

查看**脚本调试区2**的运行结果如下：

4. 脚本调试无误后，单击**提交**按钮提交脚本。

设备在线调试

脚本提交后，可以结合节点测试数据的上下行链路是否打通，LoRa节点如何发送以及接收数据请参考各模组厂商的相关手册。

1. 节点数据上行。

i. 上报温湿度属性。

a. 在LoRa节点侧选择输入十六进制的000102后发送数据。

b. 从左侧导航栏的**设备 > 设备列表**选择对应节点，单击**查看**。

c. 在**设备详情**页面选择**物模型数据 > 运行状态**。

d. 确认节点的湿度与温度信息是否已经上报且设置如下。

ii. 上报温湿度告警事件。

- 温度告警事件上报

a. 在LoRa节点侧选择输入十六进制的0102后发送数据。

b. 在**设备详情 > 事件管理**中确认温度告警事件是否已经上报。

- 湿度告警事件上报

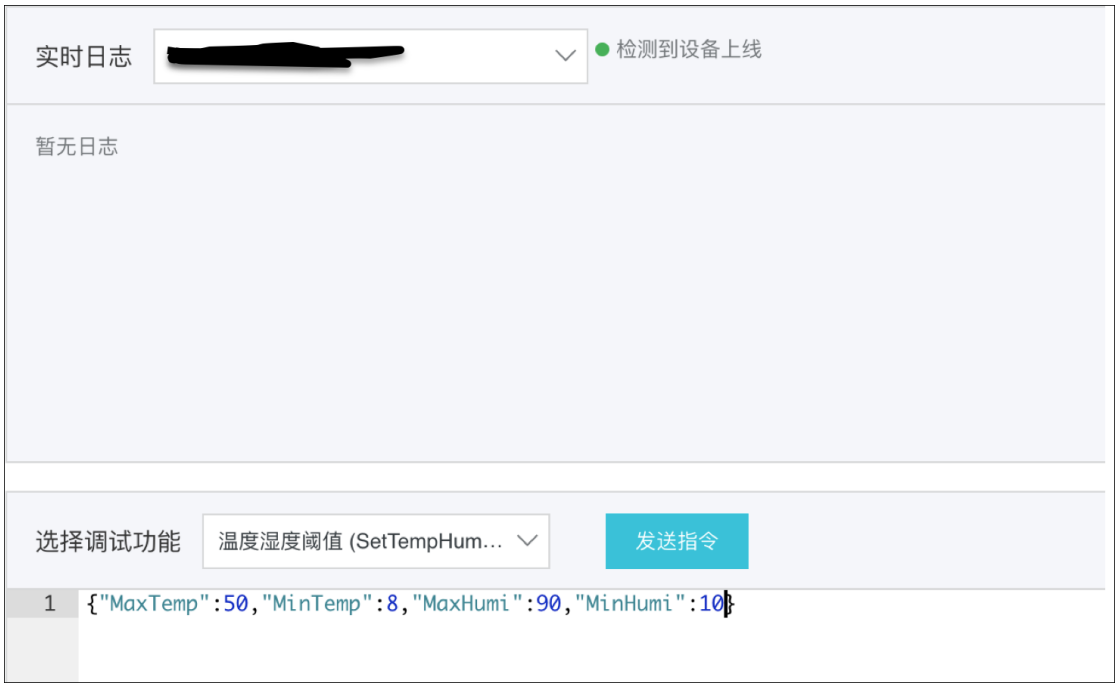
a. 在LoRa节点侧选择输入十六进制的0202后发送数据。

b. 在**设备详情 > 事件管理**中确认湿度告警事件是否已经上报。

2. 节点数据下行。

i. 在**产品详情 > 设备开发**，单击对应节点的**调试**按钮，进入**在线调试**页面。

ii. 选择调试功能为之前添加的温度湿度阈值，具体格式如下所示，单击发送指令。



发送完成后在节点侧确认输出是否是16进制的 `0332085a0a` 。

> **说明** 对于Class A类型的节点，需要先发送数据才能启动接收。

固件升级

LoRa节点设备可以通过本地端烧录方式升级固件，目前不支持网络在线升级（FUOTA）。

2.LoRaWAN智能厕所

本实践推荐使用阿里云物联网平台一站式完成应用开发，物联网平台可直接调用Link WAN网管服务。

背景信息

- 应用概述

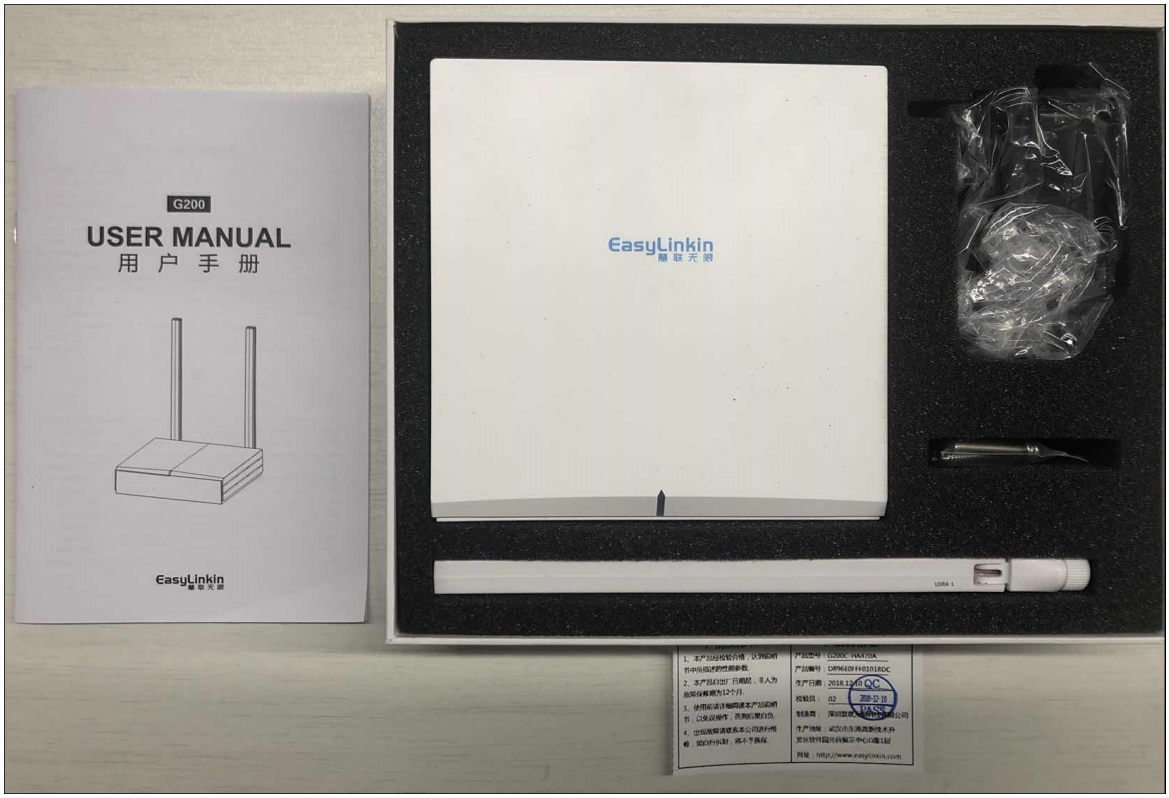
  为了增加厕所使用效率，减少被味道“熏陶”的等待时间，同时也为了增加厕所的清洁效率，可以做一个非侵入式的智能厕所改造方案。通过红外热释电检测每个坑位有没有人，在web/app上实时显示，方便如厕人员查询。并且可以检测厕所的臭味，达到阈值时通知清洁工进行清扫。

  本文中将使用物联网平台搭建了一个基于LoRaWAN连接的智能厕所demo。

- 物料清单
  - 慧联无限G200 LoRaWAN网关
  - 慧联无限LoRa红外传感器
  - 一台能联网的电脑

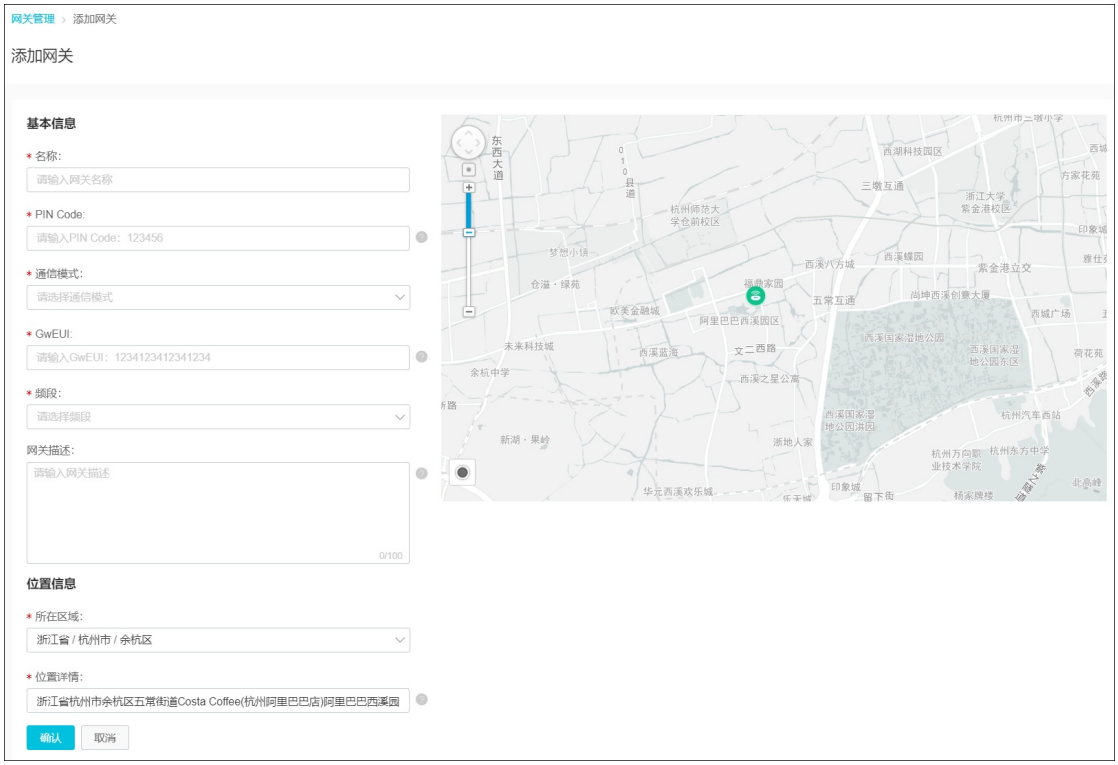
配置LoRaWAN网关-自建LoRaWAN网络

1. 选择从阿里云IoT市场上购买网关并自己组成网络。以下是本实践使用的网关



2. 插上网线与电源，登录物联网络管理平台控制台注册网关。
   i. 在左侧导航栏上，选择**网络管理 > 网关管理**。
   ii. 在**网关列表**页签，单击**添加网关**。

   iii. 配置信息以激活网关，如下图所示。



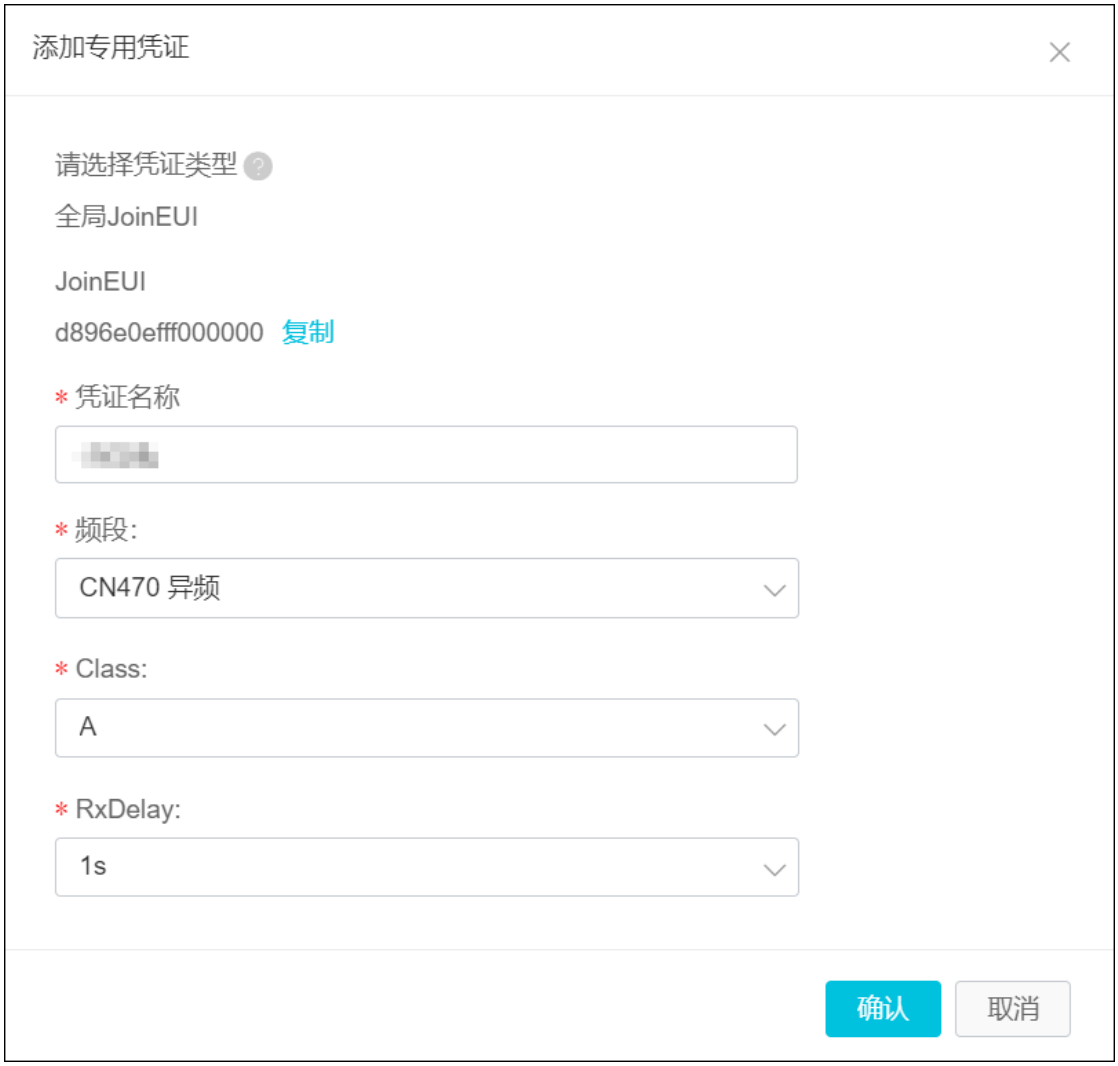
参数	说明
基本信息	
名称	填写添加的网关名称，支持中文、英文字母、数字和下划线，长度限制4～30，中文算两位。
PIN Code	Pin Code为6位数字，通常会贴在网关标签上。
通信模式	通信模式分为**全双工**和**半双工**两种，此处以选择**半双工**为例。
GwEUI	请在网关标签上查看该参数。
频段	支持以下三种频段的选择： ■ CN470 同频 ■ CN470 异频 ■ AS923 同频 此处以选择**CN470 异频**为例。
网关描述	选填，可以填写所添加网关的备注信息。网关描述不多于100字符。
位置信息	
	说明 这里需要手动选择网关位置，因G200不带GPS模块，需要手动填写，之后会显示在地图上。带GPS网关的，上报数据后会自动刷新位置。
所在区域	请选择网关的所在区域。

参数	说明
位置详情	填写网关的具体的位置信息，您也可于右侧地图单击做定位。

G200回传网默认是DHCP上网配置，如果您是固定IP或PPPoE，需依照实际环境调整，相关操作请参考G200网关手册。配置成功后可以在**网关列表**下显示**在线**状态。

3. 创建自己的网络凭证，然后把网络分配给自己账号使用。分配后透过入网凭证来取得网络使用权利。
   i. 在左侧导航栏上，选择**网络管理 > 入网开通**。
   ii. 在**入网开通**页面，单击**添加专用凭证**，如下图所示。

   iii. 在下图中配置参数，配置完成后单击**确认**即可创建一个凭证。



参数	说明
凭证名称	填写添加的凭证的名称，支持中文、英文字母、数字和下划线，长度限制4～30，中文算两位。
频段	支持以下三种频段的选择： ■ CN470 同频 ■ CN470 异频 ■ AS923 同频 此处以选择**CN470 异频**为例。
Class	支持以下三种类的选择： ■ A ■ B ■ C 此处以选择**A**为例，按需发送。

参数	说明
RxDelay	选择Class A可配置 RXDelay，用于配置上行后的接收窗口延迟时间，默认1s，可选范围在1s~15s。

iv. 单击该凭证**操作**所在列的**凭证授权**，将添加好的凭证授权给自己，如下图所示。

授权成功之后，新凭证会在出现在**凭证清单**里，如果是别人将凭证授权给您，接受后也会在此清单显示。

拆封LoRaWAN硬件

从网上购买来的认证过的LoRa传感器，拆出来的时候可以看到后面附了一个16位码的贴纸，这个是节点的DevEUI。同时还会有一个6位的PINCODE字段。

按照说明书，默认是5分钟模式，即收到一次警报之后，5分钟之内收到的警报不会再上传。这跟厕所的使用场景不一致，我们需要调成测试模式，即每次收到的警报都会上传。

操作方法：把背部的壳子掰开（不需要螺丝刀）后，调整跳针针帽位置到ON，然后合上盖子，先不拔掉塑料片。

在物联网平台上配置LoRa节点

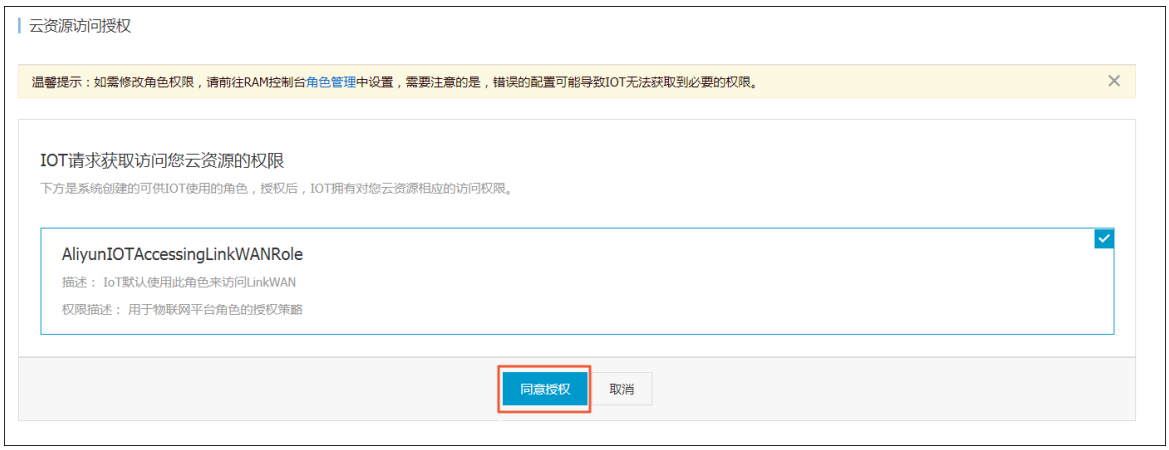
1. 登录物联网平台控制台
2. 在左侧导航栏上选择**设备管理 > 产品**，单击**创建产品**

3. 填写产品信息，然后单击**保存**。

参数	说明
产品名称	智能咖啡机（可自定义）
所属品类	选择**自定义品类**
节点类型	选择**直连设备**
连网方式	选择**LoRaWAN**

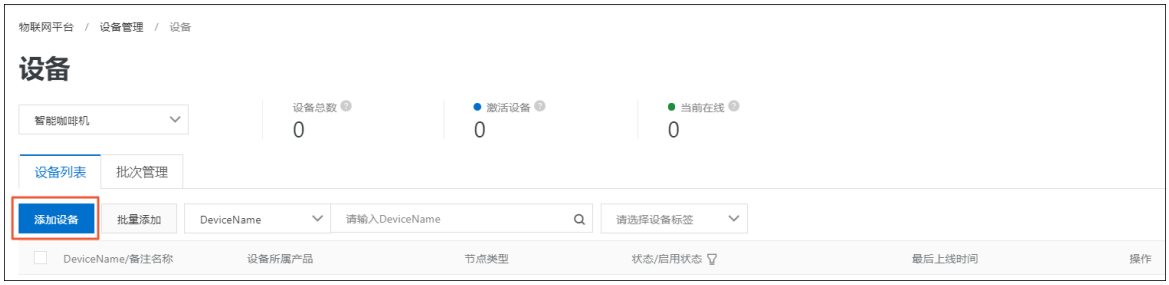
参数	说明
入网凭证	从清单里选择，或单击**添加专用凭证** **说明** 请确认已经在Link WAN取得入网凭证。
数据格式	透传/自定义
产品描述	非必填，用于描述产品的相关信息。

如首次使用Link WAN需授权数据权限。

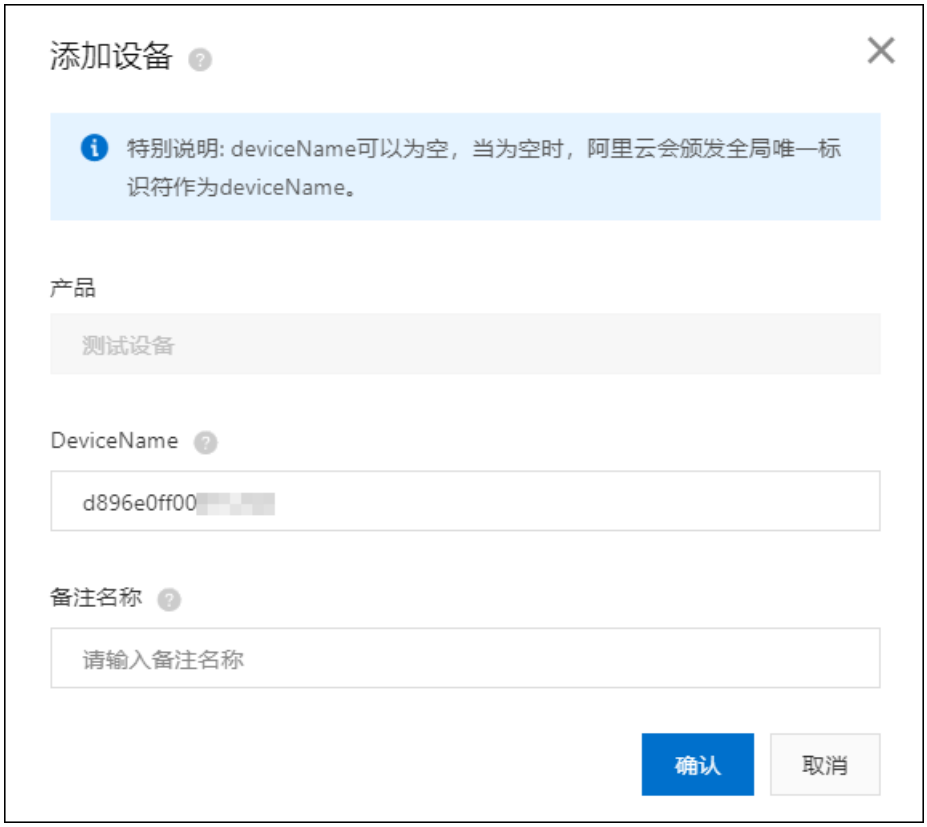


4. 创建完毕后，单击**管理设备**前往管理。

5. 单击**添加设备**，即可完成LoRa设备接入。



6. 输入16位DevEUI作为DeviceName（需小写）。

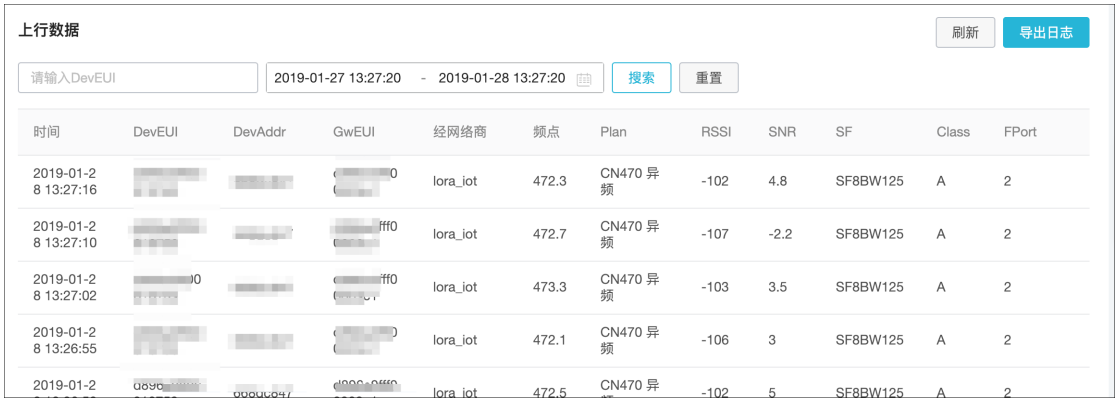


查看上下行数据及数据日志

1. 回到物联网络管理平台控制台，可以查看LoRaWAN链路层数据通信上下行。
   i. 在左侧导航栏上选择**节点管理 > 节点分组**。

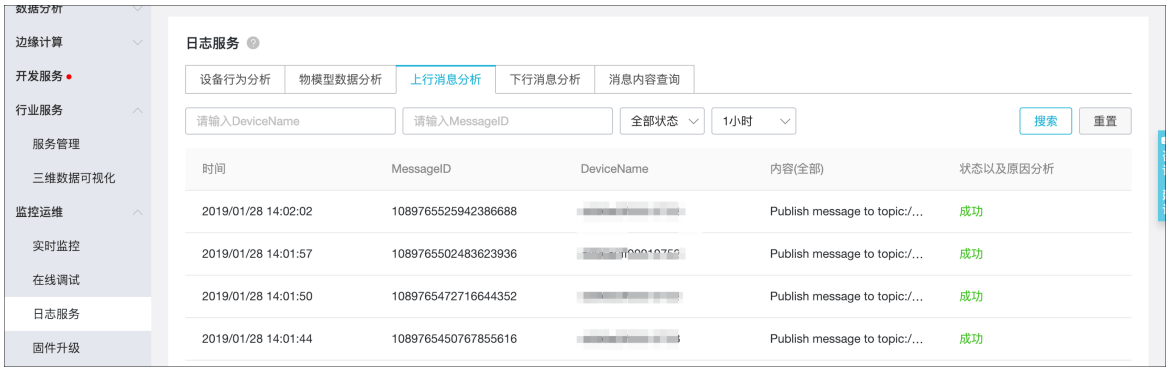
> **说明** 此节点已授权给物联网平台接入使用，这里只提供查看功能。

   ii. 单击该节点对应**操作**栏下的**查看**，选择**上行数据**页签，查看上行数据。



2. 拔掉LoRa传感器上的塑料片，让传感器上电。
3. 查看物联网平台上收到的上行日志。

   在物联网平台控制台左侧导航栏上选择**监控运维 > 日志服务**，在**上行消息分析**页签查看到如下日志。



此时设备也收到了数据日志，即表示所有通讯链路都打通，下图为原始的上行日志。

解析设备上传的信息

从传感器的说明文件可以得知，传感器上报的是二进制数据。我们如何把二进制数据转化为可以理解的属性名称呢？具体请看下文的操作步骤。

下图是厂家提供的传感器的二进制配置文件

1、LoraWan payload上行格式定义：

字段定义：

属性	值
字段名称	报警状态
协议版本	01
占用字节	B1
字段类型	数值型
字段单位	-
偏移量	-
分度值	-

字段定义：

属性	值
字段名称	传感器状态
协议版本	01
占用字节	B2
字段类型	数值型
字段单位	度
偏移量	-
分度值	-

字段定义：

属性	值
字段名称	电池状态
协议版本	01
占用字节	B2
字段类型	数值型
字段单位	-
偏移量	-
分度值	-

字段定义：

属性	值
字段名称	防拆状态
协议版本	01
占用字节	B2
字段类型	布尔型
字段单位	-
偏移量	-
分度值	-

字段定义：

属性	值
字段名称	按键状态
协议版本	01
占用字节	B2
字段类型	数值型
字段单位	-
偏移量	-
分度值	-

其中BYTE2字段（传感器状态、电池状态、防拆状态、按键状态）位序定义如下：

位序	7	6	5	4	3	2	1	0
意义	传感器状态		电池状态		防拆状态	按键状态		

BYTE0	BYTE1	BYTE2
1	0-3（所用到的数值）	1
协议版本位（01/最新的为02）	报警状态(如右边表格)	传感器状态
	若是门磁：00 表示 门关闭 03 表示门磁报警即门开	
	若是红外：00 表示 无红外感应 01 表示探测到红外	
	若是可燃：00 表示无可燃气体 02 表示探测到可燃气体	

注：BYTE0 协议版本

BYTE1：哪个类型的产品报警

报警状态	状态值
无报警	0x00
红外报警	0x01
可燃气体报警	0x02
门磁报警	0x03
保留	0x06-0xFF
烟感	0x04

1. 定义物模型。

   对于该型号的传感器，`020100`中的第一个BYTE `02`表示协议，`01`对红外传感器表示有人，`00`表示传感器状态正常。我们首先需要在产品里定义**室内人体探测开关**和**传感器属性**两个功能，用于记录这两个属性。

   i. 在物联网平台控制台左侧导航栏上选择**设备管理 > 产品**，单击产品对应的**查看**，进入产品详情页。

   ii. 选择**功能定义**页签，单击**编辑草稿 > 添加自定义功能**。

   iii. 在**添加自定义功能**弹框中，**功能类型**选择**属性**，添加以下两个属性。

■ 添加**传感器属性**


添加自定义功能
* 功能类型
属性
服务
事件
* 功能名称
传感器属性
* 标识符
SensorProperty
* 数据类型
enum (枚举型)
* 枚举项
参数值
参数描述
8
~
正常
删除
3
~
没电
删除
+添加枚举项
* 读写类型
读写
只读

- 添加**室内人体探测开关**

属性添加完成后，如下图所示。

> **说明** 读写类型都选择*读写*。

2. 使用产品定义里的数据解析，把二进制数据自动转化为ALink-JSON格式，以应对定义物模型中的两个属性。

> 说明
> - 转化规则可以参考文档，这里只提供最后的代码。
> - 数据解析需要产品为**开发中**状态。如果已经发布请单击右上角**撤回发布**。

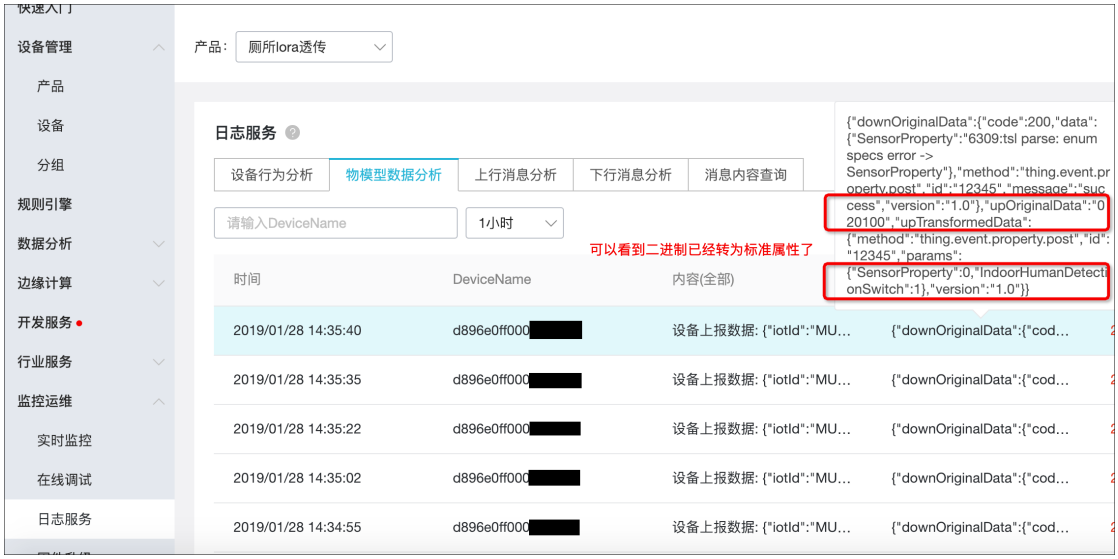
i. 在**产品详情**页，单击**数据解析**页签。

ii. 在**编辑脚本**下方，输入如下的解析脚本，然后单击页面下方的**保存**。

```
var COMMAND_REPORT = 02;
var COMMAND_SET = 01;
var ALINK_PROP_REPORT_METHOD = 'thing.event.property.post'; //标准ALink JSON格式topic， 设备 上传属性数据到 云端
var ALINK_PROP_SET_METHOD = 'thing.service.property.set';
function rawDataToProtocol(bytes) {
  var uint8Array = new Uint8Array(bytes.length);
  for (var i = 0; i < bytes.length; i++) {
    uint8Array[i] = bytes[i] & 0xff;
  }
  var dataView = new DataView(uint8Array.buffer, 0);
  var jsonMap = new Object();
  var fHead = uint8Array[0]; // 第0个BYTE为上报协议
  if (fHead == COMMAND_REPORT) {
    jsonMap['method'] = ALINK_PROP_REPORT_METHOD; //ALink JSON格式 - 属性上报topic
    jsonMap['version'] = '1.0'; //ALink JSON格式 - 协议版本号固定字段
    jsonMap['id'] = '' + 12345; //ALink JSON格式 - 标示该次请求id值
    var params = {};
    params['IndoorHumanDetectionSwitch'] = uint8Array[1]; //第1个BYTE为传感器读数判断有没有人
    params['SensorProperty'] = uint8Array[2]; //第2个BYTE为传感器本身的状态，对应产品属性中 prop_float
    jsonMap['params'] = params; //ALink JSON格式 - params标准字段
  }
  return jsonMap;
```

iii. 输入原始数据*020100*进行调试，可以看到右边解析成功，然后单击页面下方的**执行**即可让脚本生效。

在日志里可以看到二进制的020100已经转为 `{"SensorProperty":0,"IndoorHumanDetectionSwitch":1}` ，表示已完成设备接入。



在IoT Studio上配置智能厕所看板

IoT Studio（原Link Develop）是阿里云针对物联网场景提供的生产力工具，可覆盖各个物联网行业核心应用场景，解决物联网开发领域开发链路长、技术栈复杂、协同成本高、方案移植困难的问题。入口在物联网平台控制台左侧导航栏上的**IoT Studio**项。

1. 在IoT Studio上新建一个项目，以项目维度管理我们的智能厕所应用。

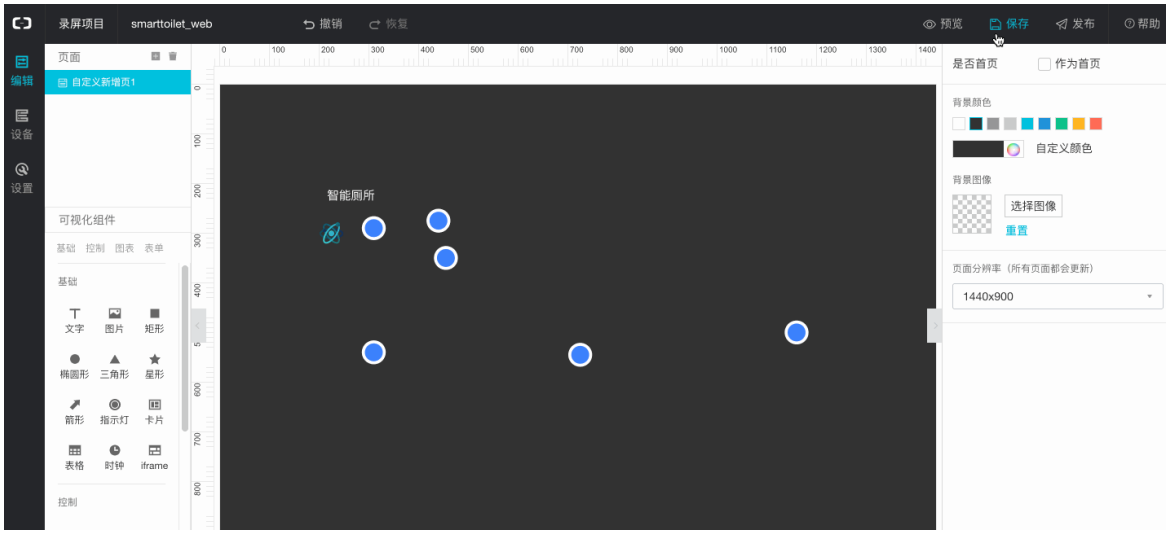
2. 创建项目之后可以看到右边栏多了一项SmartToilet，单击**查看**进入项目详情。然后在项目概览页单击右上角**导入产品**，把刚才创建的产品导入到项目内。

3. 在左侧栏上选择**推荐 > Web可视化开发**，单击**新建应用**进入Web可视化开发工作台，然后选择空白模板进行新建。

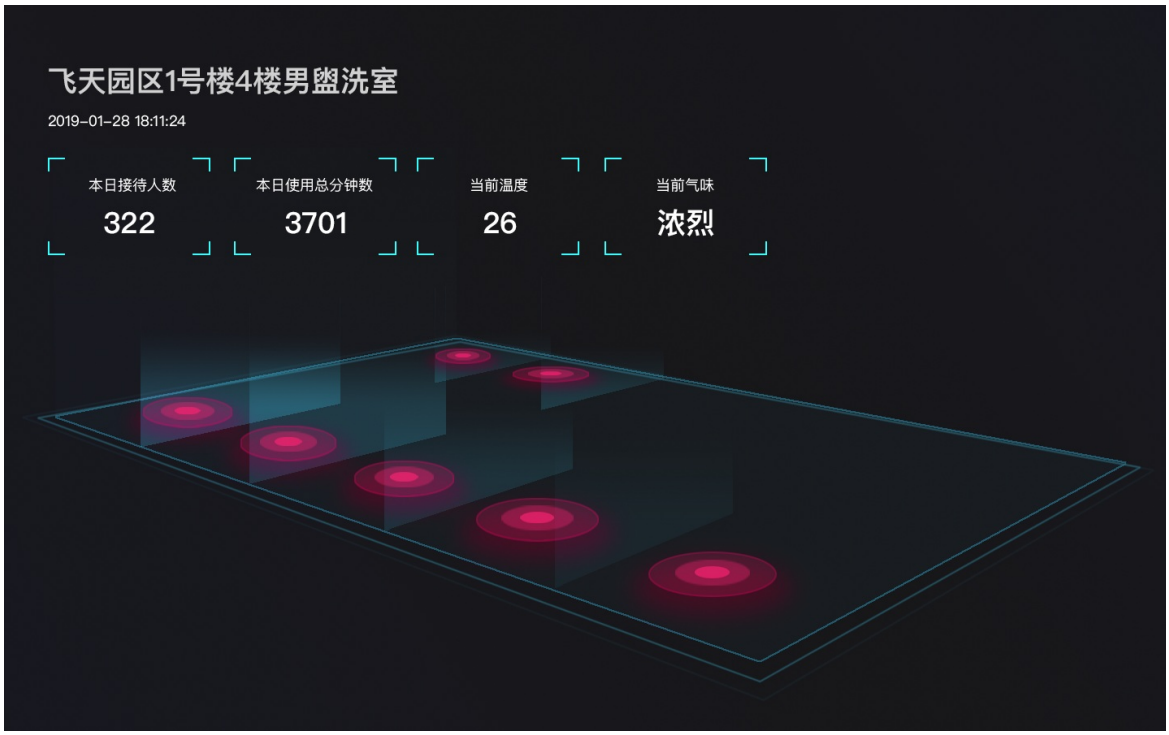
4. 进入编辑中，可以改变背景色，添加标题文字等。对最重要的厕所占位状态，可以理解为一个“有人”、“没人”的指示灯，因此把指示灯组件放入画布中，并配置数据源。

5. 配置数据源时，请选择产品、关联的设备以及关联的布尔型属性。单击**验证数据格式**获得通过，之后单击**确定**完成配置。配置之后可以通过上传图片等修改指示灯开关状态的样式。

6. 单击**预览**，完成整个Web应用的链路调试。



7. 假设有7个坑位，只需要重复上述步骤即可（可以使用上传excel格式批量添加设备）。另外GUI配置的部分与之前的版本操作一致，在此不再赘述。最终效果如下图所示。

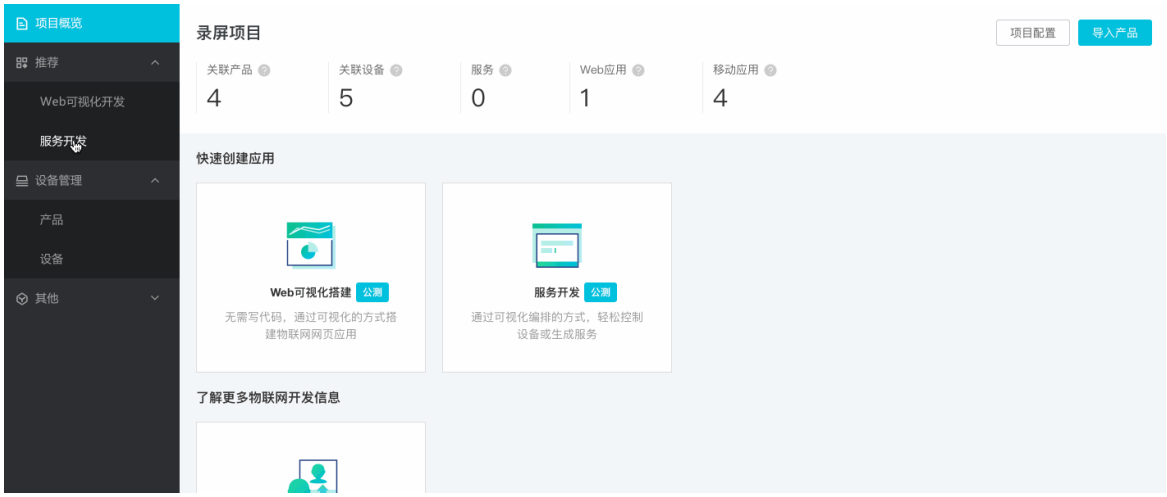


在IoT Studio上配置智能厕所状态推送&转储

服务开发（原服务编排）可以通过可视化拖拽的方式快速完成所需业务逻辑的设计，例如，设备联动、可视化搭建数据联动、云服务连接、API 生成、数据处理与转储、生成App的后端服务。在本文中，我们要使用服务开发完成智能厕所坑位状态的推送，以及把厕所使用状况转储到表格存储Table Store上。

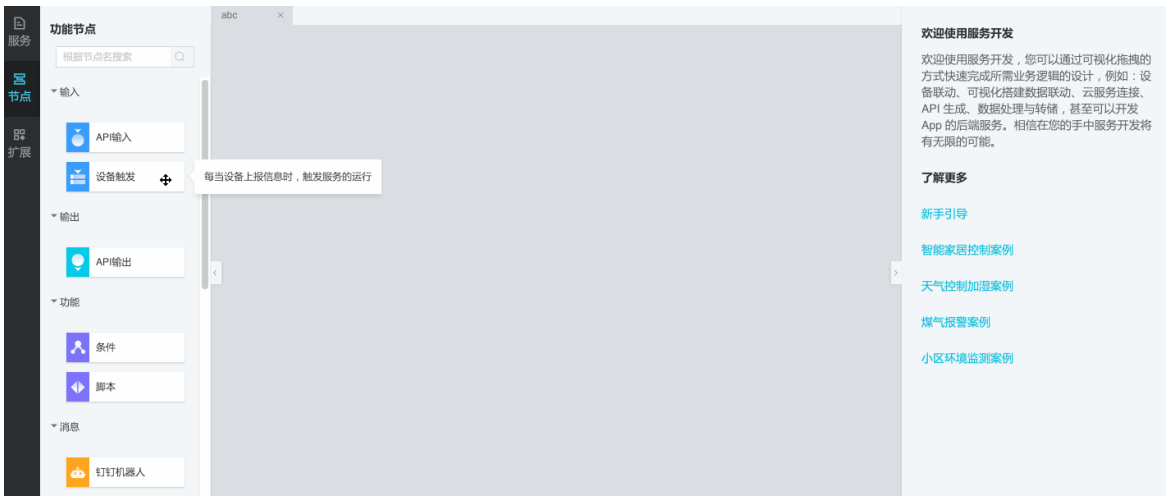
1. 在IoT Studio上新建一个服务。

   从项目概览页选择**推荐 > 服务开发**，单击**新建服务**即可进入服务开发工作台的新建页面，选择空白模板进行新建。

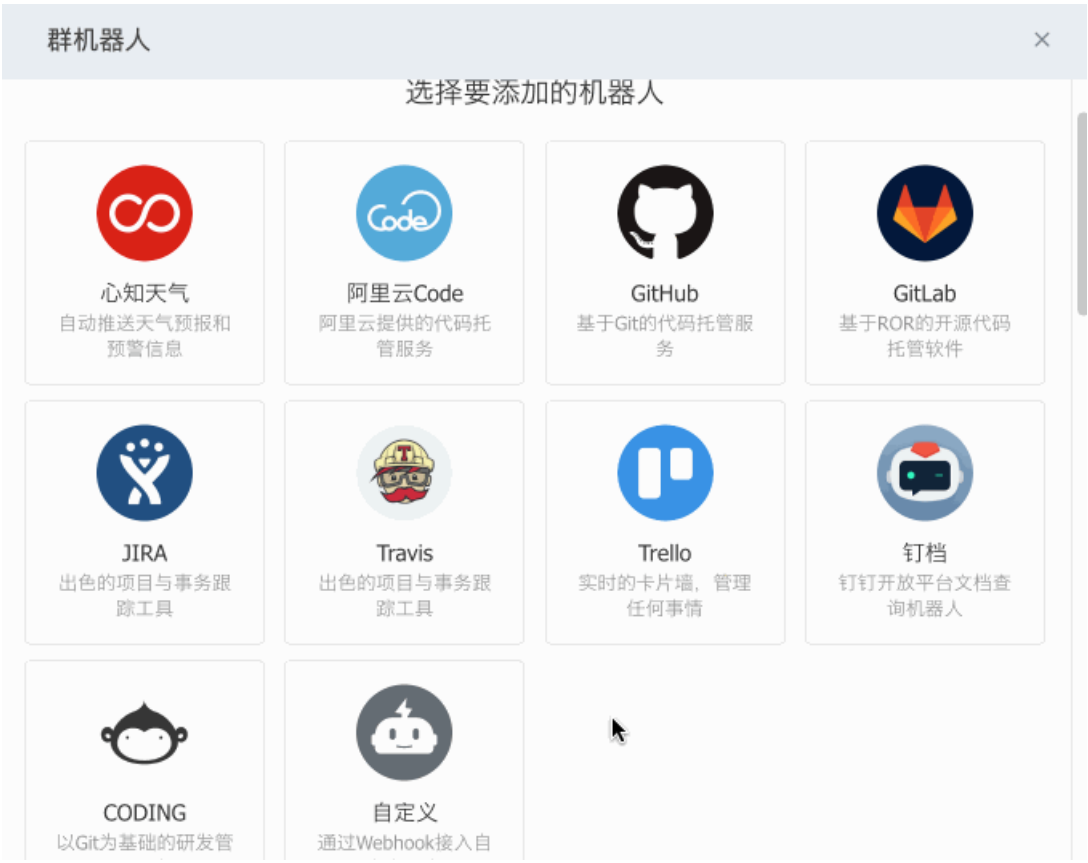


2. 把厕所的占用时间推送到钉钉机器人上。

   首先配置一个**设备触发**节点作为输入，选择之前创建好的产品，**设备选择**选择**所有设备**，以监听所有设备上报的信息。**上报类型**选择**属性上报**，最后配置一个钉钉机器人节点作为消息推送节点。



3. 在要推送的钉钉群内添加机器人，获得webhook，如下图所示。

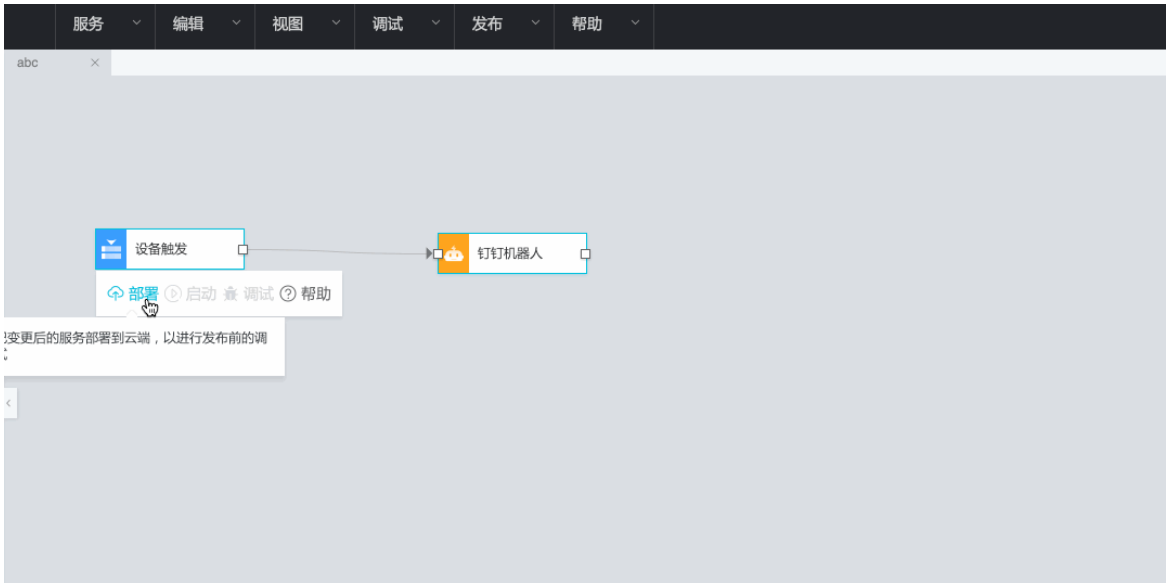


4. 配置webhook以后可以配置推送的信息。钉钉机器人节点目前支持多种消息推送类型，并且支持动态调用设备数据。

   框内配置项如下：

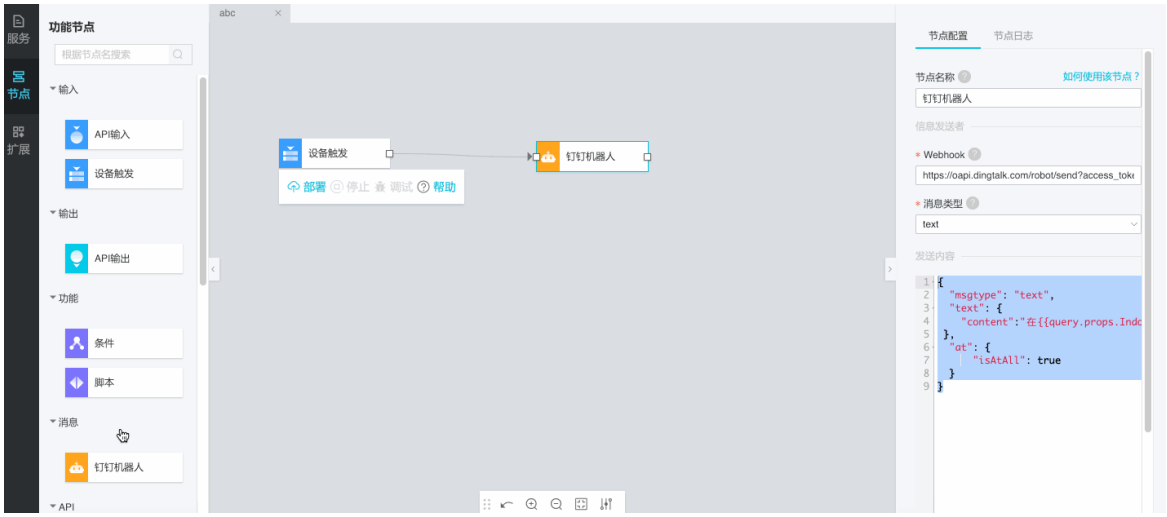
```
{
 "msgtype": "text",
 "text": {
  "content":"在{{query.props.IndoorHumanDetectionSwitch.time}}时候有{{query.props.IndoorHumanDetectionSwitch.value}}人进坑！"
 },
 "at": {
   "isAtAll": true
 }
}
```

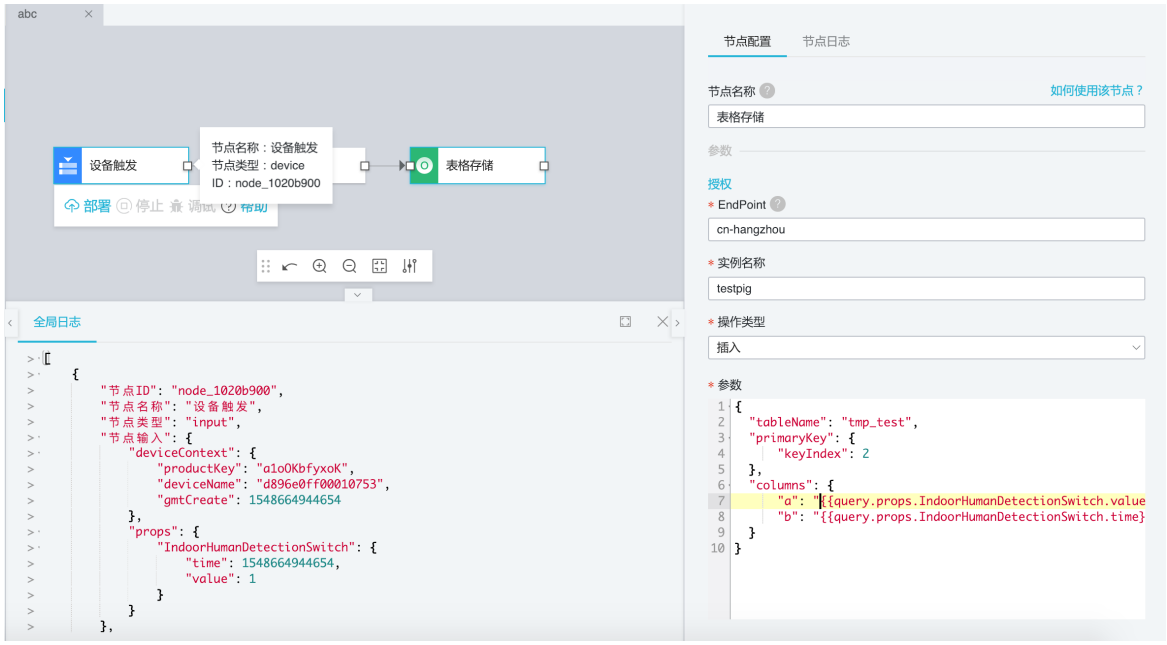
5. 配置完成之后，单击**部署**对服务进行部署。然后单击**启动**让服务生效。



如果设备已经上线，则可以直接看到机器人的消息推送，实现厕所使用状态的实时推送了，如下图所示。

6. 如果需要把厕所的使用状况使用TableStore，云数据库MySQL等云产品存储起来，可以使用存储节点。




abc
设备触发
节点名称：设备触发
节点类型：device
ID：node_1020b900
表格存储
部署 停止 调试 帮助
全局日志
[
{
"节点ID": "node_1020b900",
"节点名称": "设备触发",
"节点类型": "input",
"节点输入": {
"deviceContext": {
"productKey": "a1oOKbfyxoK",
"deviceName": "d896e0ff00010753",
"gmtCreate": 1548664944654
},
"props": {
"IndoorHumanDetectionSwitch": {
"time": 1548664944654,
"value": 1
}
}
},
节点配置
节点日志
节点名称
如何使用该节点？
表格存储
参数
授权
EndPoint
cn-hangzhou
实例名称
testpig
操作类型
插入
参数
{
"tableName": "tmp_test",
"primaryKey": {
"keyIndex": 2
},
"columns": {
"a": "{{query.props.IndoorHumanDetectionSwitch.value
"b": "{{query.props.IndoorHumanDetectionSwitch.time}
}
}

最终结果如图：


tmp_test
表格数据
插入数据
查询数据
更新数据
删除数据
数据源：tmp_test
表格数据最多显示50行。
详细数据
keyIndex(主键)
a
b
详细数据
1
aaa
bbb
详细数据
2
1
1548664932279
共有2条，每页显示：10条
1

结语


IoT Studio
焕新发布
10分钟搭建物联网应用
立即体验