ALIBABA CLOUD

阿里云

物联网平台
入门教程

文档版本：20220112

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.企业版实例快速入门 05
1.1. 概述 05
1.2. 创建企业版实例 06
1.3. 创建产品和设备 07
1.4. 设备接入和上报数据 11
1.5. 数据转发到表格存储 13
1.6. 服务端订阅设备消息 17
1.7. 云端下发指令 21
2.公共实例快速入门 24
2.1. 概述 24
2.2. 创建产品与设备 25
2.3. 为产品定义物模型 28
2.4. 建立设备与平台的连接 32
2.5. 服务端订阅设备消息 35
2.6. 云端下发指令 39
3.使用MQTT.fx接入物联网平台 42
4.使用自定义Topic进行通信 50
5.设备消息通过RocketMQ流转到服务器 57

1.企业版实例快速入门

1.1. 概述

本文帮助您快速了解物联网平台的基础能力，包括设备上云、设备上报消息、云端订阅设备消息、云端下发指令到设备等。

前提条件

- 开通物联网平台服务。

  说明 开通服务不收费。开通后，您可先试用物联网平台服务，试用流程，请参见物联网平台试用中心。物联网平台服务的计费标准，请参见计费概述。

- 准备Node.js语言开发环境。本文中设备端开发是在Windows 10（64位）下，使用阿里云提供的Node.js语言SDK进行开发；本文中服务端下发指令，使用物联网平台的云端Node.js语言SDK，调用接口Pub。
- 准备Java开发环境。本文中服务端接收设备消息，使用Java语言的AMQP SDK。

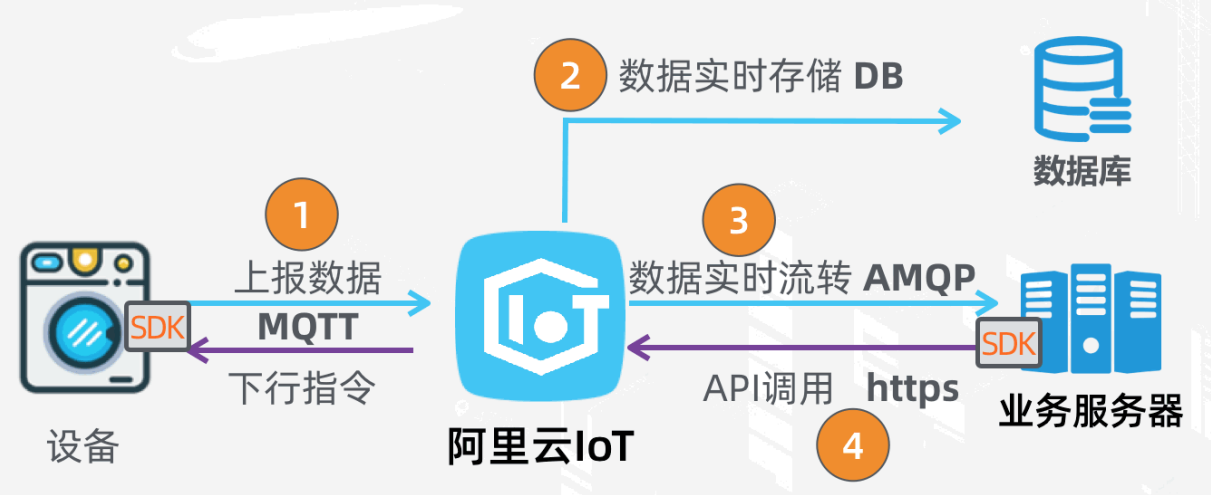
  本文使用Java开发环境如下：

  - 操作系统：Windows 10 64位
  - JDK版本：JDK8
  - 集成开发环境：Intellij IDEA社区版

背景信息

本文以家庭温控器为例，通过上报温湿度数据和使用API发送消息的操作，描述设备如何接入物联网平台及收发消息。

消息通信流程



1. 设备上报数据到阿里云物联网平台。
   i. 创建企业版实例：在物联网平台购买企业版实例，用于设备接入和业务管理。
   ii. 创建产品和设备：在物联网平台控制台创建产品和设备，获取设备证书信息（ProductKey、DeviceName和DeviceSecret）。
   iii. 设备接入和上报数据：以MQTT协议将设备接入到企业版实例，通过物模型上报数据。

2. 数据转发到表格存储：使用规则引擎数据流转功能，将数据转发到表格存储（Tablestore）中存储。
3. 服务端订阅设备消息：数据通过AMQP通道流转至企业服务器。
4. 云端下发指令：调用物联网平台云端API，向设备下发指令。

视频演示

1.2. 创建企业版实例

物联网平台支持用户购买企业版实例，用于设备接入和业务管理。

操作步骤

1. 登录物联网平台控制台。
2. 在**实例概览**页面的**购买企业版实例**卡片中，单击**购买实例**。

3. 在实例购买页面，根据您的业务需求选择实例规格，并完成购买。

   本文示例直接使用默认规格，创建**华东2（上海）**地域的**基础型**实例。

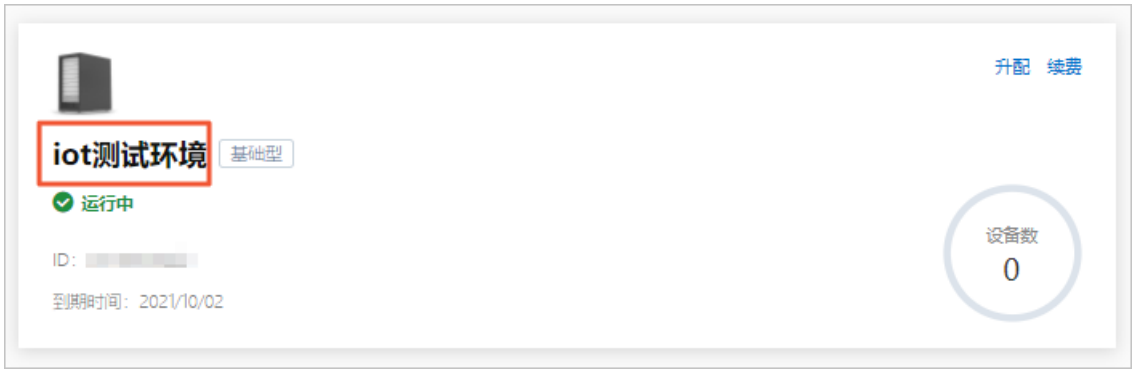
   如需了解更多信息，请参见查看实例终端节点。

   实例购买完成后，系统默认为实例分配一个实例ID。刷新**实例概览**页面，您可在**公共实例**右侧看到新购买的实例。

   您需复制并保存实例ID，用于后续AMQP接入物联网平台。

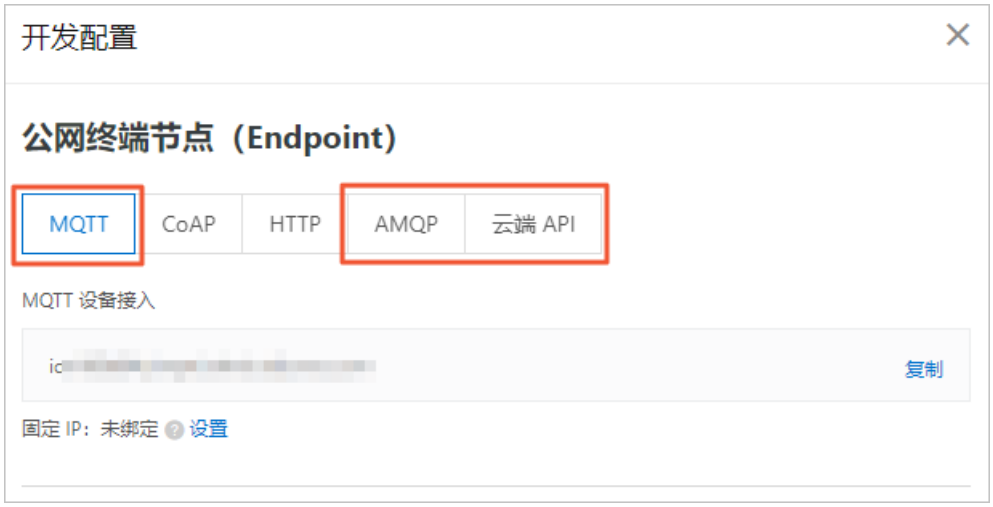
4. 在**实例概览**页面，单击该实例。
5. 在**实例详情**页面，单击**实例基本信息**区域的**编辑**，输入*iot测试环境*，单击**保存**。

   保存备注名称后，在**实例概览**页面，以备注名称显示该实例，便于您后续快速找到实例。



6. 在**实例详情**页面，单击右上角的**查看开发配置**，在右侧面板查看各节点信息。

   本文示例以公网环境为例，您需复制并保存**MQTT**、**AMQP**和**云端API**的接入信息，用于后续实现设备接入、服务端订阅消息和云端下发指令功能。



后续步骤

创建产品和设备

1.3. 创建产品和设备

使用物联网平台的第一步是在云端创建产品和对应设备，获取设备证书（ProductKey、DeviceName和DeviceSecret）。

前提条件

已完成购买企业版实例。具体操作，请参见创建企业版实例。

背景信息

- 产品相当于一类设备的集合，同一产品下的设备具有相同的功能。您可以根据产品批量管理设备，如定义物模型、自定义Topic等。
- 您的每个实际设备需对应一个物联网平台设备。将物联网平台颁发的设备证书（ProductKey、DeviceName和DeviceSecret）烧录到设备上，用于设备连接物联网平台的身份验证，请参见设备获取设备证书文档。

本文描述在物联网平台为家庭温控器创建对应的产品、定义物模型、添加设备和获取设备证书的操作步骤。

操作步骤

1. 在物联网平台控制台的**实例概览**页面，单击实例**iot测试环境**。
2. 在左侧导航栏，选择**设备管理 > 产品**，单击**创建产品**。
3. 在**新建产品**页面，配置参数后，单击**确认**。

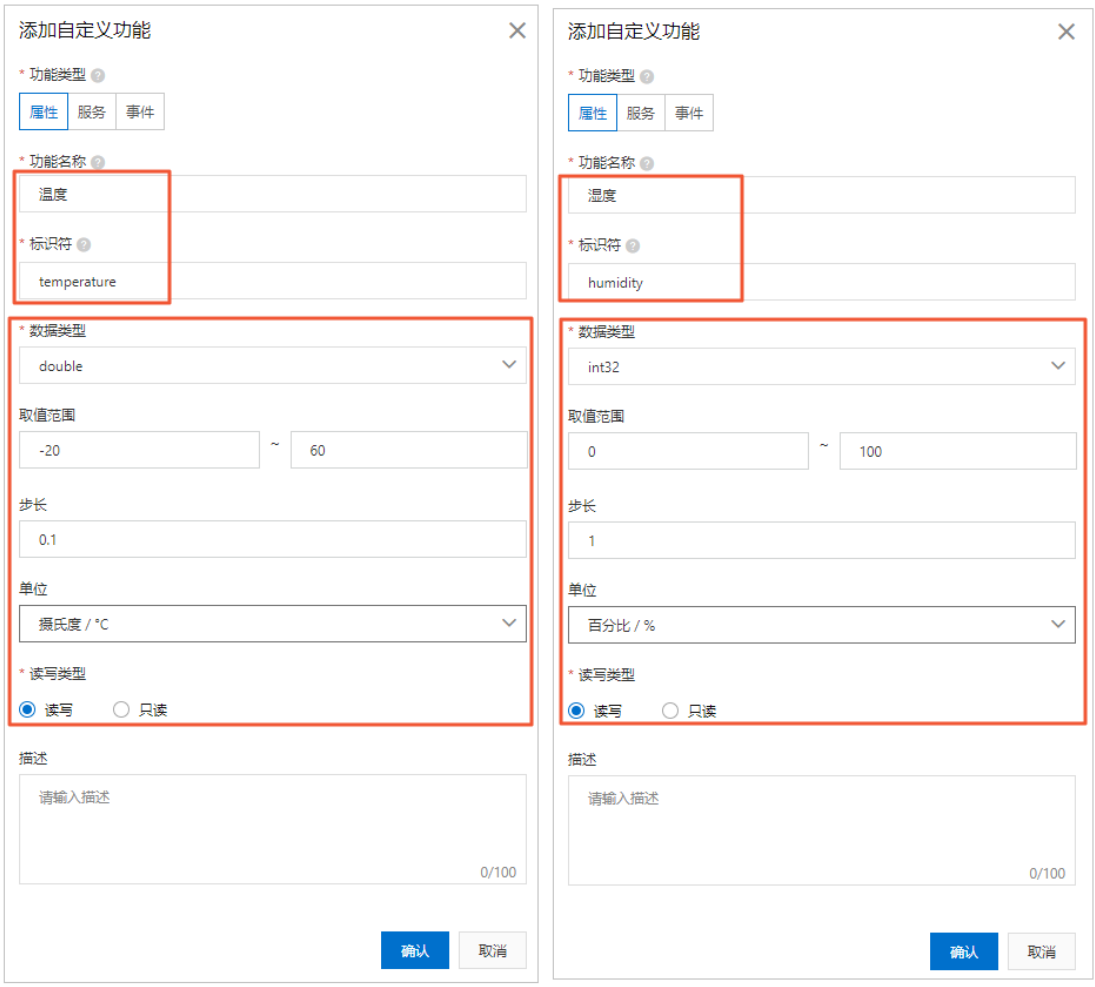
   本文示例的**产品名称**为家庭温控器，**所属品类**为**自定义品类**，其他参数使用默认值，如下图所示。

4. 在**创建产品**页面，单击**前往定义物模型**。

   为产品定义物模型，将实际产品抽象成由属性、服务、事件所组成的数据模型，便于物联网平台管理和数据交互。

5. 在**功能定义**页签，单击**编辑草稿**，按照以下步骤，为家庭温控器产品定义温度（temperature）和湿度（humidity）属性。

i. 在**默认模块**下，单击**添加自定义功能**，如下图所示，分别配置温度和湿度。



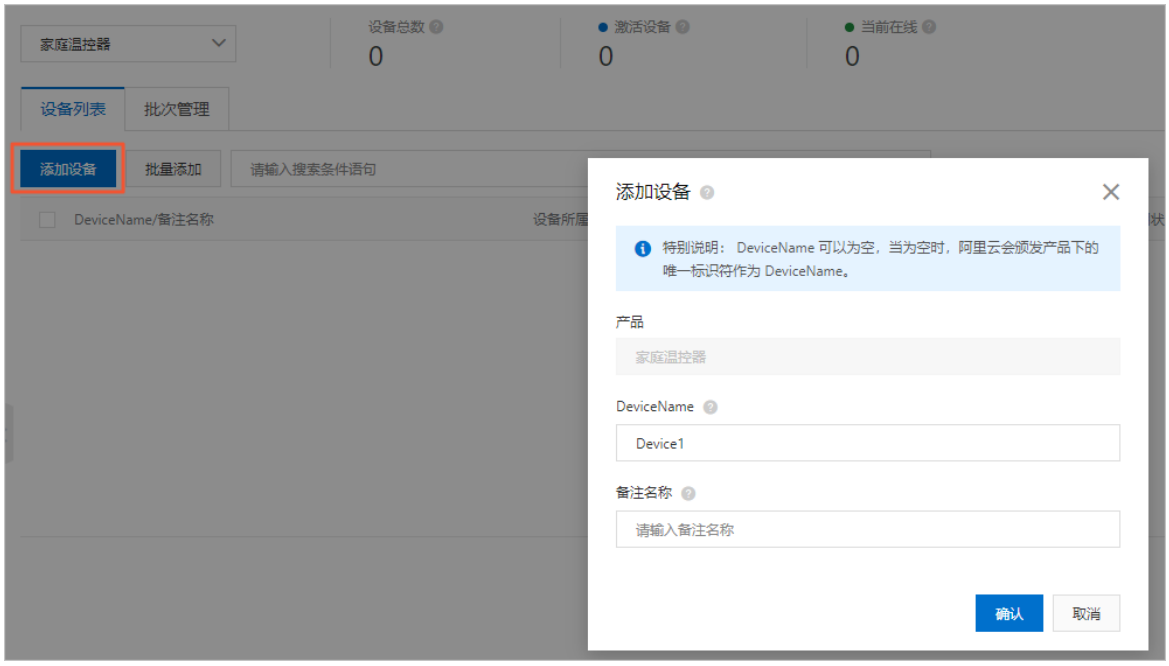
ii. 单击页面左下方的**发布上线**按钮，弹出**发布物模型上线**对话框。

iii. 选中**确认已查看当前版本与线上版本的比对结果**前复选框。

iv. 单击**确定**，发布物模型。

6. 在**产品详情**页面，单击**前往管理**。

7. 在**设备列表**页签，单击**添加设备**，在弹出对话框中，输入设备名称（例如Device1），单击**确认**。



创建设备成功后，在弹出的**添加完成**对话框，单击**一键复制设备证书**获取设备证书。

您也可单击**前往查看**，在**设备详情**页面，单击DeviceSecret右侧**查看**，获取设备证书。

设备证书包含ProductKey、DeviceName和DeviceSecret，是设备与物联网平台进行通信的重要身份认证。后续设备接入，需设置此信息，请复制后妥善保管。

参数	说明
ProductKey	设备所属产品的ProductKey，即物联网平台为产品颁发的全局唯一标识符。

参数	说明
DeviceName	设备在产品内的唯一标识符。DeviceName与设备所属产品的ProductKey组合，作为设备标识，用来与物联网平台进行连接认证和通信。
DeviceSecret	物联网平台为设备颁发的设备密钥，用于认证加密。需与DeviceName成对使用。

后续步骤

设备接入和上报数据

1.4. 设备接入和上报数据

获取设备证书（ProductKey、DeviceName和DeviceSecret）后，通过MQTT协议将设备接入企业版实例。

前提条件

已创建产品和设备。具体操作，请参见创建产品和设备。

操作步骤

> **说明** 示例通过物模型相关Topic上报数据。您也可以通过自定义Topic，向设备发送数据，更多信息，请参见自定义Topic。

1. 在Windows系统或Linux系统下载并安装Node.js。本文以Windows 10（64位）系统为例，下载安装包node-v14.15.1-x64.msi。
2. 安装成功后，打开CMD窗口，通过以下命令查看node版本。

   ```
   node --version
   ```

   显示如下版本号，表示安装成功。

   ```
   v14.15.1
   ```

3. 在本地计算机创建一个JavaScript文件（例如iot_device.js），用来存放Node.js示例代码。

   Node.js示例代码如下所示：

```
const mqtt = require('aliyun-iot-mqtt');
// 1. 设备身份信息
var options = {
    productKey: "g18l***",
    deviceName: "Device1",
    deviceSecret: "b2e6e4f102458d84***",
    host: "iot-cn-***.mqtt.iothub.aliyuncs.com"
};
// 2. 建立MQTT连接
const client = mqtt.getAliyunIotMqttClient(options);
//订阅云端指令Topic
client.subscribe(`/${options.productKey}/${options.deviceName}/user/get`)
client.on('message', function(topic, message) {
    console.log("topic " + topic)
    console.log("message " + message)
})
setInterval(function() {
    // 3.定时上报温湿度数据
    client.publish(`/sys/${options.productKey}/${options.deviceName}/thing/event/proper
ty/post`, getPostData(), { qos: 0 });
}, 5 * 1000);
function getPostData() {
    const payloadJson = {
        id: Date.now(),
        version: "1.0",
        params: {
            temperature: Math.floor((Math.random() * 20) + 10),
            humidity: Math.floor((Math.random() * 20) + 10)
        },
        method: "thing.event.property.post"
    }
    console.log("payloadJson " + JSON.stringify(payloadJson))
    return JSON.stringify(payloadJson);
}
```

参数	示例	说明
productKey	g18l***	您添加设备后，保存的设备证书信息，请参见设备证书信息。 您可在控制台中设备**Device1**的**设备详情**页面查看。
deviceName	Device1	
deviceSecret	b2e6e4f102458d84***	
host	iot-cn-***.mqtt.iothub.aliyuncs.com	您购买实例成功后，保存的MQTT设备接入域名，请参见创建企业版实例。 您可在控制台中实例**iot测试环境**的**实例详情**页面，单击**查看开发配置**，查看MQTT的信息。

4. 打开CMD窗口，使用 `cd` 命令找到iot_device.js文件所在路径，在该路径下使用 `npm` 命令下载阿里云IoT的MQTT库。下载后的MQTT库文件如下图所示。

```
npm install aliyun-iot-mqtt -S
```

5. MQTT库下载完成后，在CMD窗口输入如下命令，运行iot_device.js代码，启动设备。

```
node iot_device.js
```

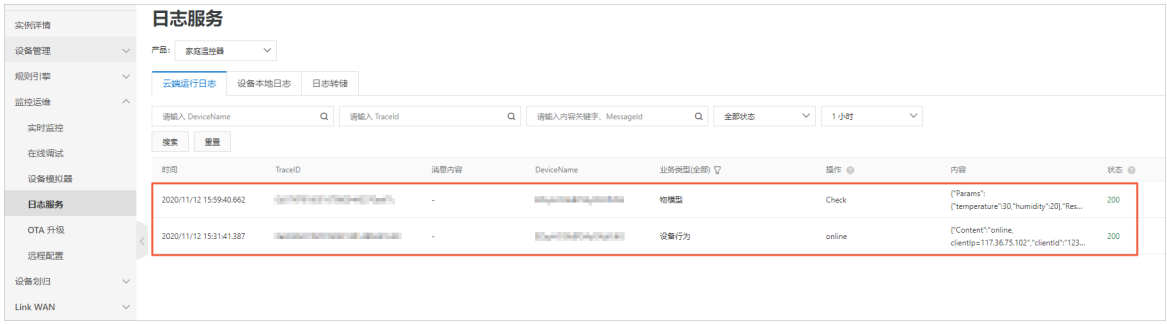
返回如下信息，表示设备接入成功，并上报数据。

```
payloadJson {"id":161848***,"version":"1.0","params":{"temperature":16,"humidity":21},"
method":"thing.event.property.post"}
topic /sys/g18l***/Device1/thing/event/property/post_reply
message {"code":200,"data":{},"id":"16184848***","message":"success","method":"thing.ev
ent.property.post","version":"1.0"}
```

6. iot_device.js代码运行成功后，设备状态显示为在线，在**设备详情**页面，单击**物模型数据**页签，可看到最新上报的温度值和湿度值。

7. 在左侧导航栏，单击**监控运维 > 日志服务**，进入**云端运行日志**页签，选择家庭温控器产品，查看设备上报数据的日志。

更多信息，请参见云端运行日志。



后续步骤

数据转发到表格存储

1.5. 数据转发到表格存储

您可以使用规则引擎数据流转功能，将数据转发到表格存储（Tablestore）中存储。

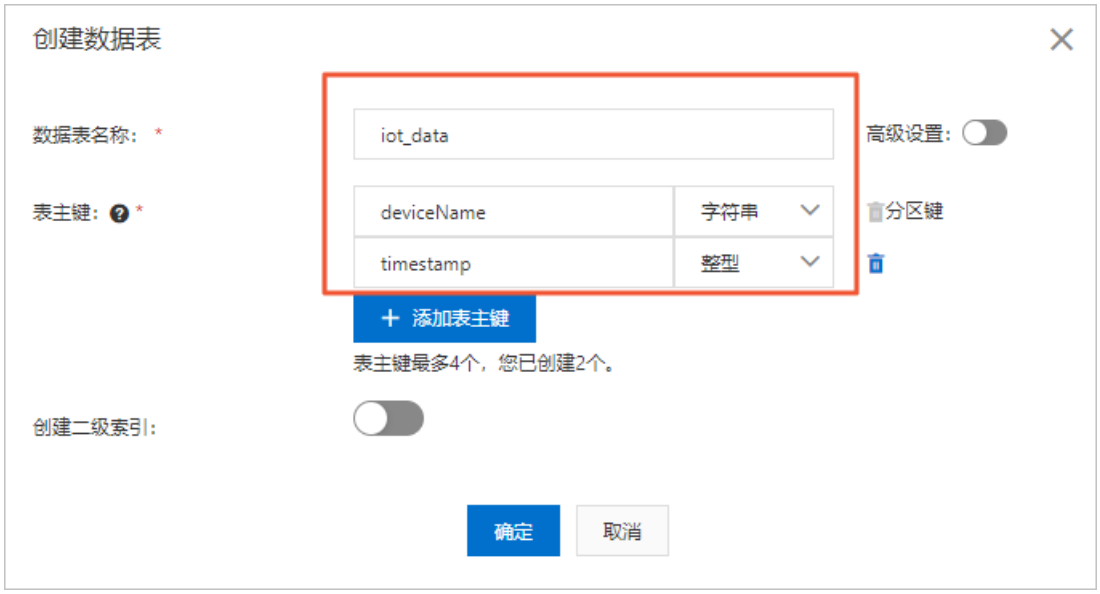
前提条件

- 已将设备接入到企业版实例。具体操作，请参见设备接入和上报数据。
- 已开通表格存储服务。具体操作，请参见开通表格存储服务。

创建数据表

1. 登录表格存储控制台，在页面上方，选择地域**华东2（上海）**。
2. 单击**创建实例**，在创建实例对话框，输入实例名称（例如IoTDataStore），其他参数使用默认值，然后单击**确定**。

   更多信息，请参见创建表格存储实例。
3. 在IoTDataStore的**实例详情**页签，单击**创建数据表**，输入数据表名称，例如iot_data，添加deviceName（字符串型）和timestamp（整型）主键。具体操作，请参见创建数据表。



设置数据流转规则

1. 在**iot测试环境**实例的左侧导航栏，选择**规则引擎 > 云产品流转**。
2. 在**云产品流转**页，单击**创建规则**。
3. 在**创建云产品流转规则**对话框，输入规则名称*温控器数据流转*，数据格式为**JSON**，单击**确认**。
4. 规则创建成功后，将跳转到**数据流转规则**页面。参照以下操作，编辑处理消息数据的SQL，设置数据转发目的地。

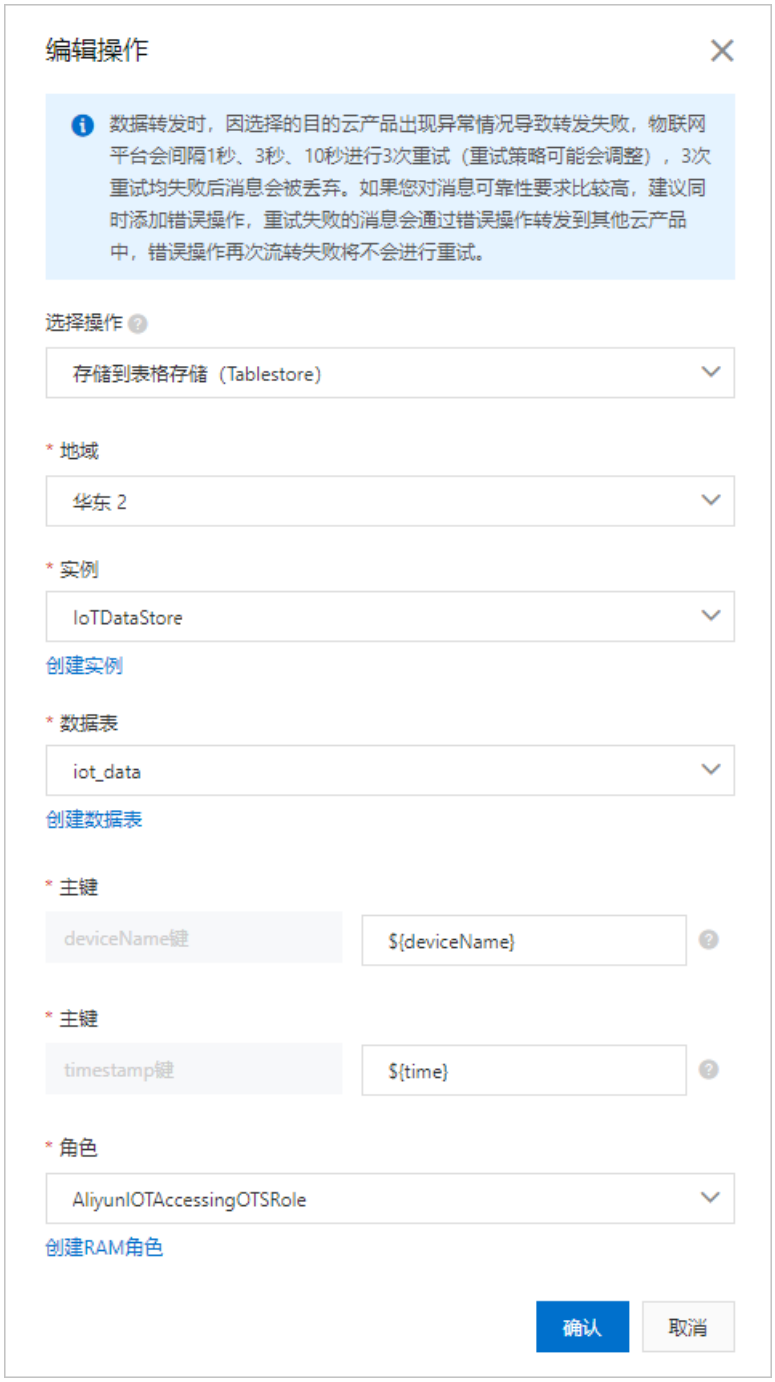
i. 单击**编写SQL**，在弹出对话框中，输入SELECT的字段，并选择Topic的数据类型为**物模型数据上报**、产品为**家庭温控器**、设备为**Device1**、物模型模块为**默认模块**、Topic为**thing/event/property/post**，然后单击**确认**。

本示例SELECT的字段如下：

```
items.temperature.value as temperature, items.humidity.value as humidity,deviceName
() as deviceName,timestamp() as time
```

SQL编写更多信息，请参见SQL表达式和函数列表。

ii. 单击**转发数据**一栏的**添加操作**，如下图所示，设置数据转发目的地为已创建的数据表iot_data。具体操作，请参见数据转发到表格存储。

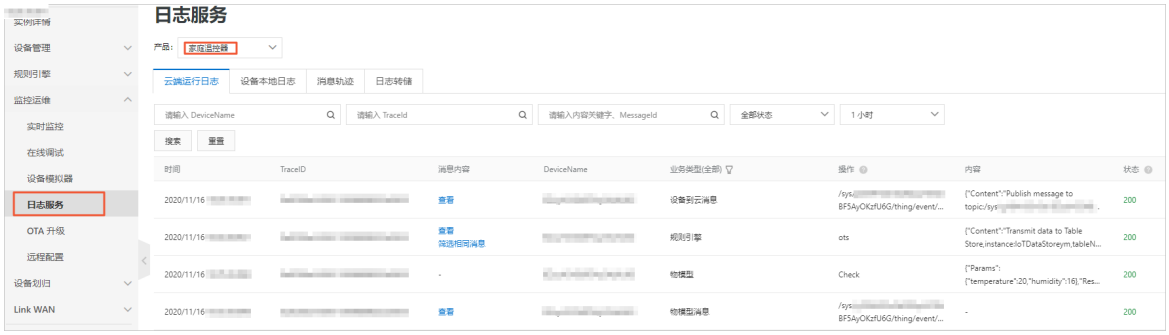


5. 所有设置完成后，返回至**云产品流转**页，单击规则对应的**启动**按钮。规则启动后，数据即可按照规则进行转发。

   当虚拟设备有数据上报后，您可以在表格存储的iot_data表中看到实时存储的数据。如下图所示。

deviceName(主键)	timestamp(主键)	humidity	temperature
Device		15	12
Device		18	11
Device		20	29
Device		23	12
Device		21	26

   您也可以在IoT企业实例的日志服务中，查看完整的流转日志，排查数据链路异常情况。如下图所示。



后续步骤

服务端订阅设备消息

1.6. 服务端订阅设备消息

设备连接物联网平台后，数据直接上报至物联网平台，平台上的数据可以通过AMQP通道流转至您的服务器。本文为您介绍通过配置AMQP服务端订阅，实现企业服务器通过接入AMQP客户端，接收路灯设备数据的完整流程。

前提条件

已接入设备和上报数据。具体操作，请参见设备接入和上报数据。

配置AMQP服务端订阅

1. 在物联网平台控制台的**实例概览**页面，单击实例**iot测试环境**。
2. 为消息消费端配置消费组身份，用于企业服务器通过消费组获取转发的消息。
   i. 在左侧导航栏，选择**规则引擎 > 服务端订阅**，单击**消费组列表**页签。
   ii. 单击**创建消费组**。
   iii. 在**创建消费组**对话框中，设置消费组名称为温控器数据接收，单击**确认**。

   创建消费组成功后，跳转到**消费组详情**页面，查看**消费组ID**。您需复制并保存**消费组ID**值，用于AMQP客户端接入。

3. 为家庭温控器产品配置服务端订阅，用于您的服务器直接订阅产品下多种类型的消息。
   i. 在左侧导航栏，选择**规则引擎 > 服务端订阅**。
   ii. 在**服务端订阅**页面的**订阅列表**页签下，单击**创建订阅**。

iii. 在**创建订阅**对话框，设置参数后单击**确认**。

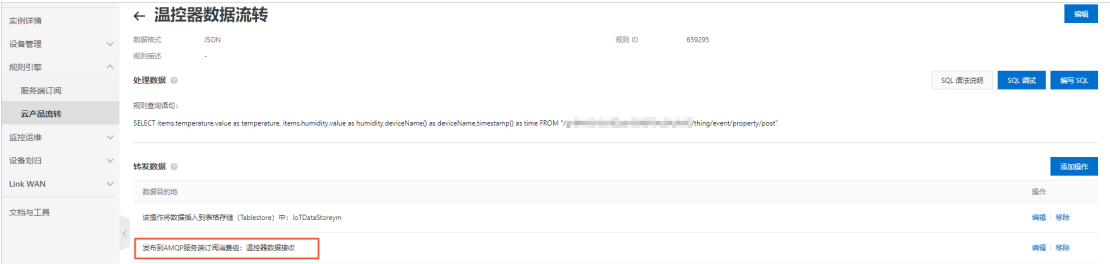
参数	说明
产品	选择家庭温控器。
订阅类型	选择AMQP。
消费组	选择上一步创建的温控器数据接收。
推送消息类型	选择设备上报消息和物模型历史数据上报。更多消息类型，请参见设置订阅。

您也可以通过云产品流转规则，将数据转发至AMQP订阅的消费组，使用AMQP客户端接收消息。例如，将流转规则**温控器数据流转**中已处理的**物模型数据上报**内容，转发到AMQP订阅的消费组温控器数据接收中。

i. 在左侧导航栏，选择**规则引擎 > 云产品流转**。

ii. 单击规则**温控器数据流转**对应的**查看**，进入**数据流转规则**页面，单击**转发数据**一栏对应的**添加操作**。

iii. 在**添加操作**对话框中，选择操作为**发布到AMQP服务端订阅消费组**，消费组为温控器数据接收，单击**确认**。



AMQP客户端接入示例

示例使用Java开发语言，推荐使用Apache Qpid JMS客户端。您可访问Qpid JMS 0.57.0，查看Qpid JMS客户端使用说明。

示例使用的开发环境如下：

- 操作系统：Windows 10 64位
- JDK版本：JDK8
- 集成开发环境：IntellIJ IDEA社区版

1. 下载Demo代码包，并解压。
2. 打开IntellIJ IDEA，导入Demo包中的示例工程*amqp-demo*。
3. 在*src/main/java/com.aliyun.iotx.demo*目录下*AmqpClient.java*文件中，参照下表，修改JMS客户端接入物联网平台代码中的参数值。

```
private final static Logger logger = LoggerFactory.getLogger(AmqpClient.class);
private static String accessKey = "${YourAccessKey}";
private static String accessSecret = "${YourAccessSecret}";
private static String consumerGroupId = "${YourConsumerGroupId}";

//iotInstanceId: 实例ID。若是2021年07月30日之前（不含当日）开通的公共实例，请填空字符串。
private static String iotInstanceId = "${YourIotInstanceId}";

//控制台服务端订阅中消费组状态页客户端ID一栏将显示clientId参数。
//建议使用机器UUID、MAC地址、IP等唯一标识等作为clientId。便于您区分识别不同的客户端。
private static String clientId = "${YourClientId}";

//${YourHost}为接入域名，请参见AMQP客户端接入说明文档。
private static String host = "${YourHost}";

// 指定单个进程启动的连接数
// 单个连接消费速率有限，请参考使用限制，最大64个连接
// 连接数和消费速率及rebalance相关，建议每500QPS增加一个连接
private static int connectionCount = 4;
```

参数	示例	说明
accessKey	LTAI4GFGQvKuqHJhFa******	登录物联网平台控制台，将鼠标移至账号头像上，然后单击**AccessKey管理**，获取AccessKey ID和AccessKey Secret。 **说明** 如果使用RAM用户，您需授予该RAM用户管理物联网平台的权限（AliyunIOTFullAccess），否则将连接失败。授权方法请参见授权RAM用户访问物联网平台。
accessSecret	iMS8ZhCDdfJbCMeA005sieKe******	
consumerGroupId	VWhGZ2QnP7kxWpeSSjt******	已创建的温控器数据接收消费组ID。请参见上文配置AMQP服务端订阅的步骤2。
iotInstanceId	"iot-cn-6ja***"	**iot测试环境**的实例ID。 您可在控制台的**实例概览**页面查看。
clientId	12345	客户端ID，可自定义，长度不可超过64个字符。建议设置为您的AMQP客户端所在服务器UUID、MAC地址、IP等唯一标识，方便您识别区分不同的客户端。 AMQP客户端接入成功后，在控制台的温控器数据接收的消费组详情页面，显示该参数。
connectionCount	4	启动AMQP客户端的连接数，最大不超过64个。用于实时消息推送的扩容。 **消费组详情**页面会以 `${clientId}+"-"+`**数字** 形式，显示连接的客户端。其中数字最小值为0。

参数	示例	说明
host	iot-***.amqp.iothub.aliyuncs.com	您购买实例成功后，保存的AMQP接入域名，请参见创建企业版实例。 您可在控制台中实例**iot测试环境**的**实例详情**页面，单击**查看开发配置**，单击**AMQP**页签查看。

4. 运行*AmqpClient.java*示例代码后，返回类似如下日志信息，表示AMQP客户端已接入物联网平台并成功接收消息。

> **说明** 本示例Demo代码中，添加了结束程序的代码（`Thread.sleep(60 * 1000);`），即程序启动成功，运行一分钟后会结束。实际场景中，您可根据需要自行设置运行时间。

```
[main] INFO com.aliyun.iotx.demo.AmqpClient - amqp demo is started successfully, and will exit after 60s
[AmqpProvider :(2):[amqps://[redacted]i.aliyuncs.com:5671]] DEBUG org.apache.qpid.jms.provider.amqp.AmqpConsumer - Dispatching received message: JmsInboundMessageDispatch { sequenc
[JmsSession [ID:d8cd8ff1-5c3[redacted]y dispatcher] DEBUG org.apache.qpid.jms.provider.failover.FailoverProvider - Executing Failover Task: message acknowledge -> JmsInboundMessag
[AmqpProvider :(2):[amqps://[redacted]i.aliyuncs.com:5671]] DEBUG org.apache.qpid.jms.provider.amqp.AmqpConsumer - Delivered Ack of message: JmsInboundMessageDispatch { sequence = 1
[JmsSession [ID:d8cd8ff1-5c3[redacted]y dispatcher] DEBUG org.apache.qpid.jms.provider.failover.FailoverProvider - Executing Failover Task: message acknowledge -> JmsInboundMessag
[AmqpProvider :(2):[amqps://[redacted]i.aliyuncs.com:5671]] DEBUG org.apache.qpid.jms.provider.amqp.AmqpConsumer - Accepted Ack of message: JmsInboundMessageDispatch { sequence = 1
[pool-1-thread-1] INFO com.aliyun.iotx.demo.AmqpClient - receive message,
topic = /a1[redacted]/device1/thing/event/property/post,
messageId = 1388044478987086465,
content = {"deviceType":"CustomCategory","iotId":"nTT9r[redacted]","requestId":"1617863381153","checkFailedData":{},"productKey":"a10[redacted]","gmtCreate":1617863218430,"deviceName":"device1","items":{"Light
```

返回参数：

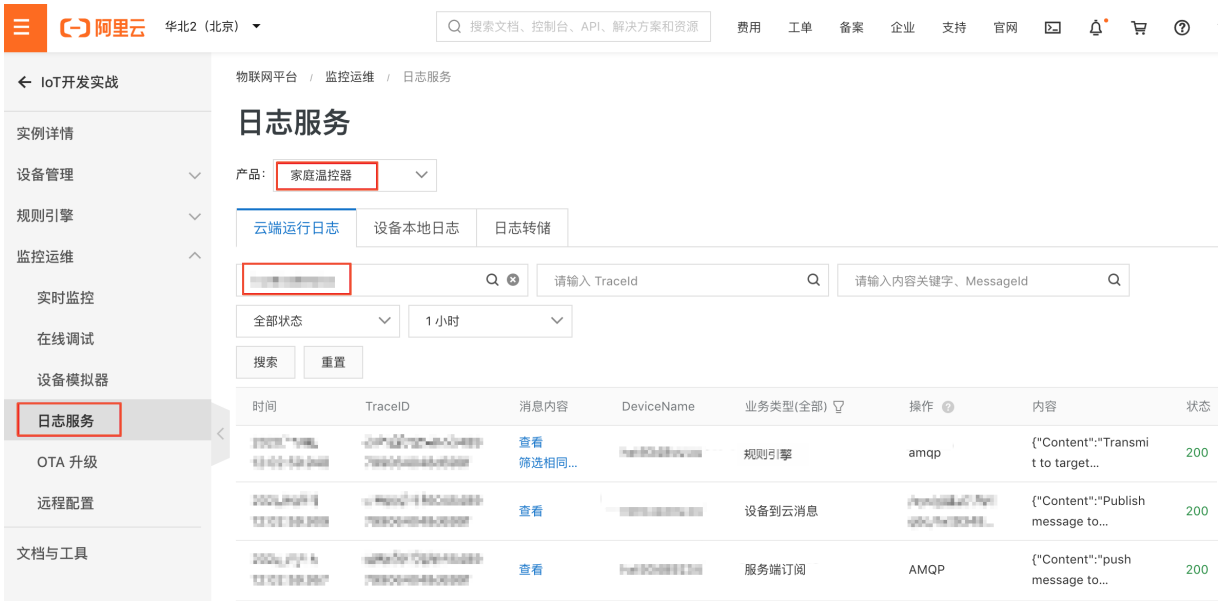
参数	参数值	说明
topic	/**********/******/thing/event/property/post	设备属性上报的Topic
messageId	1324198300680719360	消息的ID
content	{"temperature":23,"humidity":21,"time":1604548451951}	消息的内容

代码运行成功后，您可在服务端订阅中，查看消费组状态的基本信息。

具体操作，请参见管理消费组。

查看日志

所有配置完成后，登录物联网平台控制台，在企业实例下的**监控运维 > 日志服务**中，从产品列表选择家庭温控器，可在云端运行日志页签下，查看完整的日志信息。



后续步骤

云端下发指令

1.7. 云端下发指令

设备上报消息后，您可以尝试从云端下发指令到设备端。本文介绍如何通过云端应用调用Pub接口，向温控器发送指令。

前提条件

已接入设备。具体操作，请参见设备接入和上报数据。

操作步骤

> **说明** 示例通过自定义Topic下发指令。您也可以通过物模型相关Topic，发送标准化的物模型数据，更多信息，请参见什么是物模型。

1. 在本地计算机创建一个JavaScript文件（例如PubApi.js），用来存放Pub API调用的参考代码。

   Pub API调用的参考代码如下所示：

```
const Core = require('@alicloud/pop-core');
var client = new Core({
  accessKeyId: 'LTAI4GFGQvKuqHJhFa******',
  accessKeySecret: 'iMS8ZhCDdfJbCMeA005sieKe******',
  endpoint: 'https://iot.cn-shanghai.aliyuncs.com',
  apiVersion: '2018-01-20'
});
var params = {
  "RegionId": "cn-shanghai",
  "TopicFullName": "/g34q2******/Device1/user/get",
  "MessageContent": "d2FzaGluZ01vZGU6IDIsIHdhc2hpbmdUaW1lOiAzMA==",
  "ProductKey": "g34q2******",
  "IotInstanceId": "iot-cn-6ja1y******"
}
var requestOption = {
  method: 'POST'
};
client.request('Pub', params, requestOption).then((result) => {
  console.log(JSON.stringify(result));
}, (ex) => {
  console.log(ex);
})
```

本示例需修改以下参数的值。更多信息，请参见云端开发指南。

> **说明** MessageContent是要发送的消息主体，通过将消息原文转换成二进制数据，并进行Base64编码生成。示例为消息 `"washingMode: 2, washingTime: 30"` 编码后的值。

参数	示例	说明
accessKeyID	LTAI4GFGQvKuqHJhFa******	您的阿里云账号的AccessKey ID。 登录物联网平台控制台，将鼠标移至账号头像上，然后单击**AccessKey管理**，获取AccessKey ID和AccessKey Secret。 **说明** 如果使用RAM用户，您需授予该RAM用户管理物联网平台的权限（AliyunIOTFullAccess），否则将连接失败。授权方法请参见授权RAM用户访问物联网平台。
accessKeySecret	iMS8ZhCDdfJbCMeA005sieKe******	
TopicFullName	/g34q2******/Device1/user/get	要发布消息的自定义Topic。 本示例使用物联网平台中的自定义Topic：`/${productKey}/${deviceName}/user/user/get`。 其中`${productKey}`为产品家庭温控器的ProductKey值，`${deviceName}`为设备Device1。 更多信息，请参见使用自定义Topic进行通信。

参数	示例	说明
ProductKey	g34q2******	家庭温控器的ProductKey值。
IotInstanceId	iot-cn-6ja1y******	**iot测试环境**的实例ID。 您可在控制台的**实例概览**页面查看。

2. 打开CMD窗口，使用cd命令找到PubApi.js文件所在路径，在该路径下执行以下命令，安装@alicloud/pop-core模块。

```
npm install @alicloud/pop-core -S
```

返回类似信息，表示安装成功。

```
+ @alicloud/pop-core@1.7.10
updated 1 package and audited 24 packages in 1.653s
found 0 vulnerabilities
```

3. 继续在该路径下执行以下命令，安装co模块。

```
npm install co
```

返回类似信息，表示安装成功。

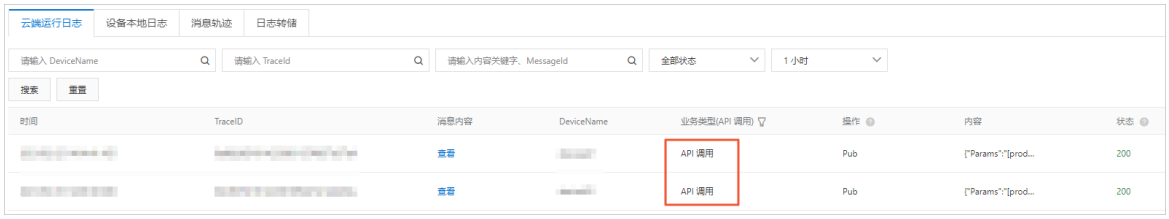
```
+ co@4.6.0
updated 1 package and audited 24 packages in 1.244s
found 0 vulnerabilities
```

4. 模块安装完成后，在CMD窗口输入如下命令，运行PubApi.js代码，从云端下发指令到设备端。

```
node PubApi.js
```

执行结果如下图所示。关于返回参数的更多信息，请参见Pub。

5. PubApi.js代码运行成功后，在左侧导航栏，单击**监控运维 > 日志服务**，进入**云端运行日志**页签，查看设备API调用的日志。



您也可以在设备端查看下发的MessageContent内容，如图所示。

2.公共实例快速入门

2.1. 概述

本文帮助您快速了解物联网平台的基础能力，包括设备上云、设备上报消息、云端订阅设备消息、云端下发指令到设备等。

本文以路灯为例，通过接入路灯设备到物联网平台控制其开关的操作，描述设备接入物联网平台并使用基础能力的步骤。

准备工作

开始接入设备前，您需要做如下准备：

- 开通物联网平台服务。

  > **说明** 开通服务不收费。开通后，您可先试用物联网平台服务，试用流程，请参见物联网平台试用中心。物联网平台服务的计费标准，请参见计费概述。

- 准备C语言开发环境。本文中设备端开发是在Linux下，使用阿里云提供的C语言SDK进行开发。
- 准备Java开发环境。本文中服务端下发指令，使用物联网平台的在线调试功能；服务端接收设备消息，使用Java语言的AMQP SDK。

  本文使用Java开发如下：

  - 操作系统：Windows 10 64位
  - JDK版本：JDK8
  - 集成开发环境：Intellij IDEA社区版

消息通信流程

1. 设备上报数据到阿里云物联网平台。
   i. 创建产品与设备：在物联网平台上为设备注册一个身份，获取设备证书信息（ProductKey、DeviceName和DeviceSecret）。该证书信息将烧录到设备上，用于设备连接物联网平台时，进行身份认证。
   ii. 为产品定义物模型：可以从属性、服务和事件三个维度定义产品功能。物联网平台根据您定义的功能构建出产品的数据模型，用于云端与设备端进行指定数据通信。
   iii. 建立设备与平台的连接：开发设备端SDK，传入设备的证书信息，将设备连接到物联网平台，使设备激活。
2. 服务端订阅设备消息：服务端通过订阅消息类型，接收设备相关消息，如设备上下线通知、设备生命周期变更、设备上报消息等。
3. 云端下发指令：使用物联网平台的在线调试功能，从控制台下发指令到设备端。

2.2. 创建产品与设备

使用物联网平台的第一步是在云端创建产品和对应设备，获取设备证书（ProductKey、DeviceName和DeviceSecret）。

背景信息

- 产品相当于一类设备的集合，同一产品下的设备具有相同的功能。您可以根据产品批量管理设备，如定义物模型、自定义Topic等。
- 您的每个实际设备需对应一个物联网平台设备。将物联网平台颁发的设备证书（ProductKey、DeviceName和DeviceSecret）烧录到设备上，用于设备连接物联网平台的身份验证，请参见设备获取设备证书文档。

本文介绍在物联网平台为路灯创建对应的产品和设备，并获取设备证书的操作步骤。

操作步骤

1. 登录物联网平台控制台。
2. 在控制台左上方，选择物联网平台所在地域，然后在**实例概览**页面，单击**公共实例**。

   注意 在中国地域，目前仅华东2（上海）地域开通了公共实例服务。

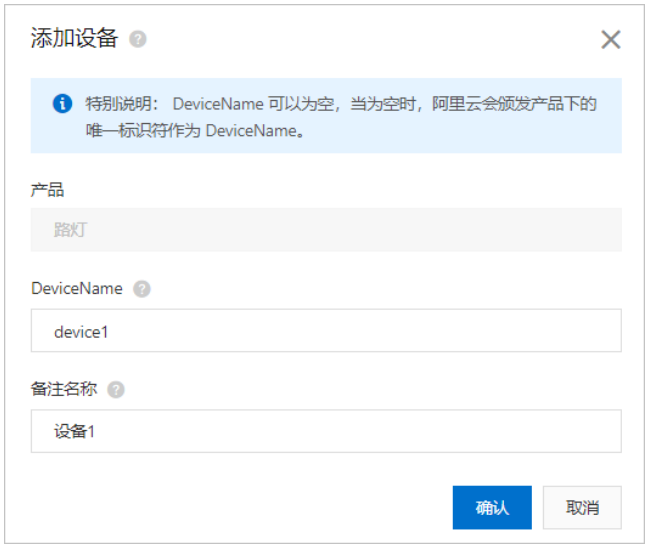
3. 在左侧导航栏，选择**设备管理 > 产品**，单击**创建产品**。
4. 在**新建产品**页面，配置参数后，单击**确认**。

   本文示例的**产品名称**为路灯，**所属品类**为**自定义品类**，其他参数使用默认值，如下图所示。

5. 在**创建产品**页面，单击**前往添加**。

6. 在**设备列表**页签，单击**添加设备**，输入设备名称（DeviceName）为device1，设置备注名称为设备1，单击**确认**。

   更多描述，请参见单个创建设备。



创建设备成功后，在弹出的**添加完成**对话框，单击**一键复制设备证书**获取设备证书。

您也可单击**前往查看**，在**设备详情**页面，单击DeviceSecret右侧**查看**，获取设备证书。

设备证书包含ProductKey、DeviceName和DeviceSecret，是设备与物联网平台进行通信的重要身份认证。后续设备接入，需设置此信息，请复制后妥善保管。

参数	说明
ProductKey	设备所属产品的ProductKey，即物联网平台为产品颁发的全局唯一标识符。
DeviceName	设备在产品内的唯一标识符。DeviceName与设备所属产品的ProductKey组合，作为设备标识，用来与物联网平台进行连接认证和通信。

参数	说明
DeviceSecret	物联网平台为设备颁发的设备密钥，用于认证加密。需与DeviceName成对使用。

后续步骤

为产品定义物模型。

2.3. 为产品定义物模型

物联网平台支持为产品定义物模型，将实际产品抽象成由属性、服务、事件所组成的数据模型，便于云端管理和数据交互。产品创建完成后，您可以为它定义物模型，产品下的设备将自动继承物模型内容。

前提条件

已完成产品创建。具体操作，请参见创建产品与设备。

背景信息

本文介绍为路灯所属的产品，定义工作电流、主灯开关的物模型属性。定义完物模型，您可以接入更多路灯到该产品下，自动继承已定义的物模型。物模型相关介绍，请参见什么是物模型。

操作步骤

1. 登录物联网平台控制台。
2. 在控制台左上方，选择物联网平台所在地域，然后在**实例概览**页面，单击**公共实例**。

> **注意** 在中国地域，目前仅华东2（上海）地域开通了公共实例服务。

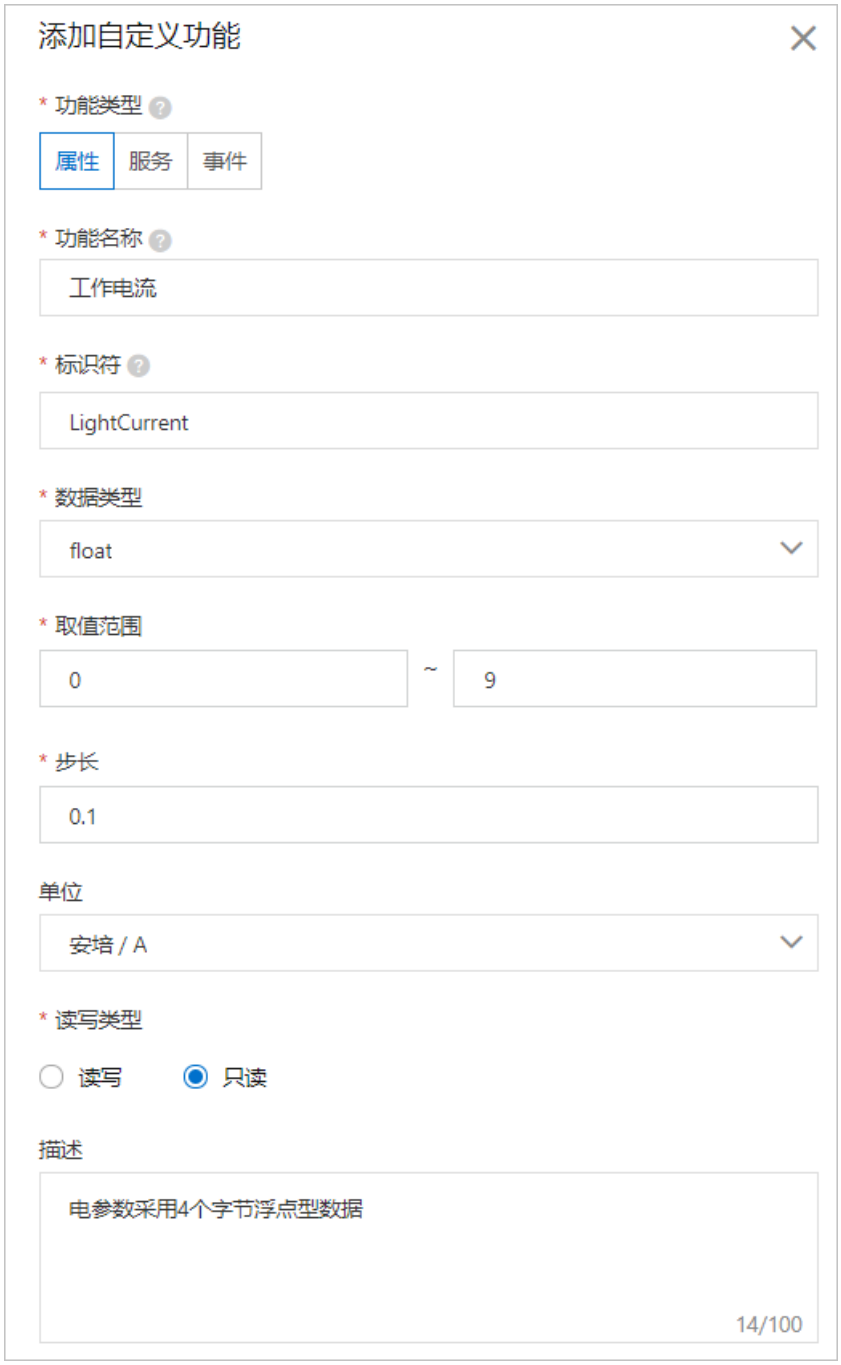
3. 在左侧导航栏，选择**设备管理 > 产品**。在产品列表中，找到已创建的路灯产品，单击操作栏的**查看**。
4. 在产品详情页，单击**功能定义**页签，然后单击**编辑草稿**。

5. 在**功能定义**页面的**默认模块**，单击**添加自定义功能**，进行物模型配置，然后单击**确认**。

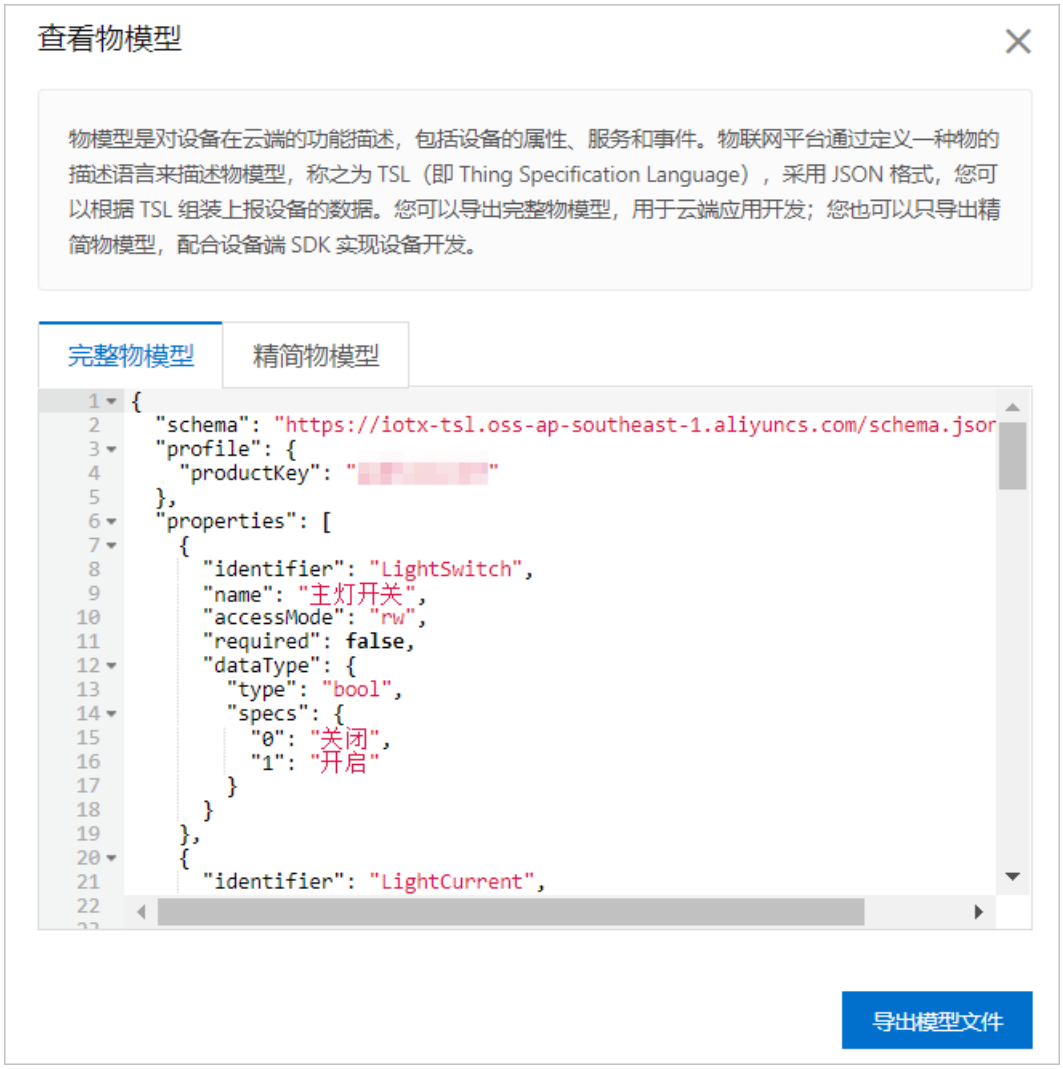
   按照如下图设置路灯产品的主灯开关和工作电流属性。

   - 主灯开关属性

   - 工作电流属性



6. （可选）单击**物模型TSL**，在**完整物模型**页签下，可看到该产品的完整物模型JSON文件。



7. 发布物模型。
   i. 单击页面左下方的**发布上线**按钮，弹出**发布物模型上线**对话框。
   ii. （可选）单击**添加发布备注**，输入版本号和版本描述。

参数	说明
版本号	设置当前物模型版本号。后期可根据版本号管理物模型。 版本号支持英文字母、数字和英文句号（.），长度限制1～16个字符。
版本描述	描述当前版本物模型。支持中文汉字、英文字母、数字和特殊符号。长度限制为100个字符。一个中文汉字算一个字符。

   iii. 如存在已上线的版本，则需要进行比对，确认新版本的修改点。

   单击**比对结果**，在**比对结果**对话框中查看修改点。确认无误后，单击**确认查阅**，回到**发布物模型上线**对话框，自动勾选“确认已查看当前版本与线上版本的比对结果”。

   iv. 单击**确定**，发布物模型。

后续步骤

建立设备与平台的连接

2.4. 建立设备与平台的连接

阿里云物联网平台提供设备端SDK，设备使用SDK与平台建立通信。本文使用平台提供的样例程序 `data_model_basic_demo` ，模拟路灯设备进行开发，实现设备与物联网平台的通信。

背景信息

- 本文使用Linux下的设备端C语言SDK。该SDK的编译环境推荐使用64位的Ubuntu16.04。
- SDK的开发编译环境会用到以下软件：

  make（4.1及以上版本）、gcc（5.4.0及以上版本）。

  可以使用如下命令行安装：

```
sudo apt-get install -y build-essential make gcc
```

操作步骤

1. 获取设备端C语言SDK。
   i. 登录物联网平台控制台。
   ii. 在控制台左上方，选择物联网平台所在地域，然后在**实例概览**页面，单击**公共实例**。

   > **注意** 在中国地域，目前仅华东2（上海）地域开通了公共实例服务。

   iii. 在左侧导航栏单击**文档与工具**，然后在**设备接入SDK**区域的**Link SDK**下，单击**SDK定制**。
   iv. 按照如下图所示，定制SDK并下载。

   在**高级能力**下，单击**物模型**，其他参数使用默认配置，然后单击**开始生成**后，会在本地下载C语言SDK的ZIP文件。

2. 解压本地的C语言SDK文件，修改SDK中的设备证书信息。

   请下载最新示例代码文件data_model_basic_demo.c，替换掉*\LinkSDK\demos*路径下的*data_model_basic_demo.c*文件，然后修改以下参数。

```
char *product_key       = "a2***";
char *device_name       = "device1";
char *device_secret     = "8c684ef***";
...
...
char  *mqtt_host = "a2***.iot-as-mqtt.cn-shanghai.aliyuncs.com";
```

参数	示例	说明
mqtt_host	○ 新版公共实例的示例：iot-cn-***.mqtt.iothub.aliyuncs.com ○ 旧版公共实例的示例：a2***.iot-as-mqtt.cn-shanghai.aliyuncs.com	设备的MQTT接入域名。 ○ 新版公共实例：在**实例详情**页面的**开发配置**面板，查看接入域名。 ○ 旧版公共实例：接入域名格式为 `${YourProductKey}.iot-as-mqtt.${YourRegionId}.aliyuncs.com` 。 实例的更多详细说明，请参见查看实例终端节点。
product_key	a2***	设备认证信息。即完成添加设备后，您保存至本地的设备证书。 您也可以在物联网平台的**设备详情**页查看设备的认证信息。更多信息，请参见获取设备认证信息。
device_name	device1	
device_secret	8c684ef***	

3. 登录Linux虚拟机，执行以下命令，安装所需软件。

```
sudo apt-get install -y build-essential make gcc
```

4. 将步骤2中已修改完成的*LinkSDK*文件，上传至Linux虚拟机的开发环境。
5. 在SDK根目录*/LinkSDK*下，执行 `make` 命令，完成样例程序的编译。

```
make clean
make
```

生成的样例程序 `data-model-basic-demo` 存放在*./output*目录下。

6. 运行样例程序。

```
./output/data-model-basic-demo
```

运行成功后，如下图所示：

```
root:~/LinkSDK# ./output/data-model-basic-demo
[1637041632.700][LK-0313] MQTT user calls aiot_mqtt_connect api, connect
[1637041632.700][LK-032A] mqtt host: a.iot-as-mqtt.cn-shanghai.aliyuncs.com
[1637041632.700][LK-0317] user name: device1&a0
[1637041632.700][LK-0318] password: 86761E21D
establish tcp connection with server(host='a0.iot-as-mqtt.cn-shanghai.aliyuncs.com', port=[443])
success to establish tcp, fd=3
local port: 35604
[1637041632.711][LK-1000] establish mbedtls connection with server(host='a.iot-as-mqtt.cn-shanghai.aliyuncs.com', port=[443])
[1637041632.766][LK-1000] success to establish mbedtls connection, (cost 45252 bytes in total, max used 47988 bytes)
[1637041632.811][LK-0313] MQTT connect success in 108 ms
AIOT_MQTTEVT_CONNECT
[1637041632.811][LK-0309] pub: /sys/a0/device1/thing/event/property/post

[LK-030A] > 7B 22 69 64 22 3A 22 31  22 2C 22 76 65 72 73 69 | {"id":"1","versi
[LK-030A] > 6F 6E 22 3A 22 31 2E 30  22 2C 22 70 61 72 61 6D | on":"1.0","param
[LK-030A] > 73 22 3A 7B 22 4C 69 67  68 74 43 75 72 72 65 6E | s":{"LightCurren
[LK-030A] > 74 22 3A 20 31 2E 35 7D  2C 22 73 79 73 22 3A 7B | t": 1.5},"sys":{
[LK-030A] > 22 61 63 6B 22 3A 31 7D  7D                      | "ack":1}}

[1637041632.833][LK-0309] pub: /sys/a0/device1/thing/event/property/post_reply

[LK-030A] < 7B 22 63 6F 64 65 22 3A  32 30 30 2C 22 64 61 74 | {"code":200,"dat
[LK-030A] < 61 22 3A 7B 7D 2C 22 69  64 22 3A 22 31 22 2C 22 | a":{},"id":"1","
[LK-030A] < 6D 65 73 73 61 67 65 22  3A 22 73 75 63 63 65 73 | message":"succes
[LK-030A] < 73 22 2C 22 6D 65 74 68  6F 64 22 3A 22 74 68 69 | s","method":"thi
[LK-030A] < 6E 67 2E 65 76 65 6E 74  2E 70 72 6F 70 65 72 74 | ng.event.propert
[LK-030A] < 79 2E 70 6F 73 74 22 2C  22 76 65 72 73 69 6F 6E | y.post","version
[LK-030A] < 22 3A 22 31 2E 30 22 7D                          | ":"1.0"}

[1637041632.833][LK-0A08] DM recv generic reply
demo_dm_recv_handler, type = 0
msg_id = 1, code = 200, data = {}, message = success
```

7. 登录物联网平台控制台，进入公共实例页面，查看设备状态和运行状态。
   - 左侧导航栏选择**设备管理 > 设备**，找到目标设备，查看设备状态。设备状态显示为**在线**，则表示设备与物联网平台成功连接。

   - 单击设备对应操作栏的**查看**，进入**设备详情**页面，单击**物模型数据**页签，然后单击**运行状态**，查看设备的运行状态。

     *data_model_basic_demo.c*文件中代码示例，上报了工作电流的值，如下图所示。

后续步骤

服务端订阅设备消息

2.5. 服务端订阅设备消息

设备连接物联网平台后，数据直接上报至物联网平台，平台上的数据可以通过AMQP通道流转至您的服务器。本文为您介绍通过配置AMQP服务端订阅，实现企业服务器通过接入AMQP客户端，接收路灯设备数据的完整流程。

背景信息

通过AMQP服务端订阅获取设备消息流程如下图所示。

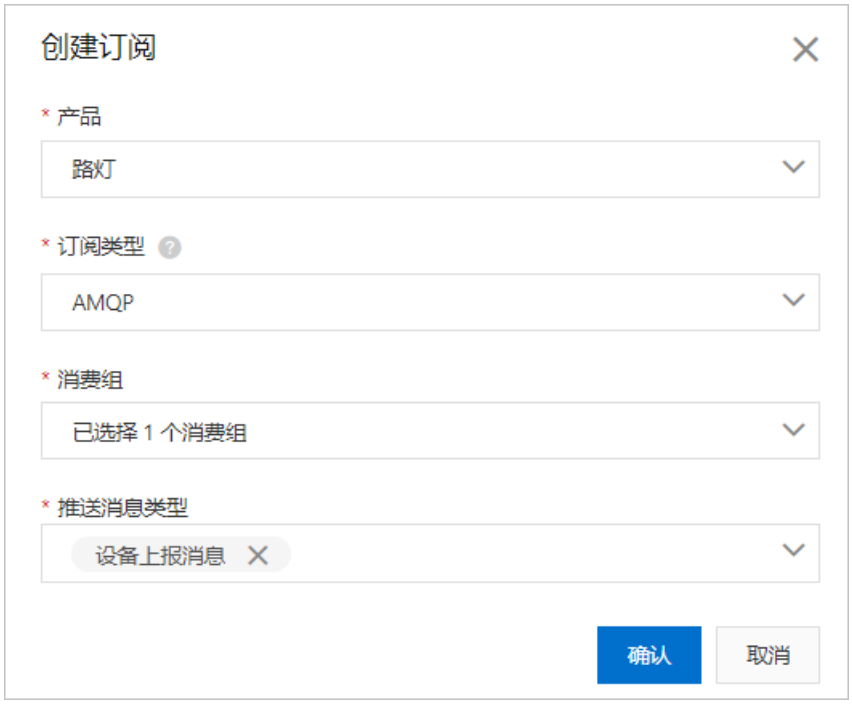
操作步骤

1. 登录物联网平台控制台。
2. 在控制台左上方，选择物联网平台所在地域，然后在**实例概览**页面，单击**公共实例**。

注意 在中国地域，目前仅华东2（上海）地域开通了公共实例服务。

3. 为消息消费端配置消费组身份，用于企业服务器通过消费组获取转发的消息。
   i. 在左侧导航栏，选择**规则引擎 > 服务端订阅**，单击**消费组列表**页签。
   ii. 单击**创建消费组**。
   iii. 在**创建消费组**对话框中，设置消费组名称为路灯消费组，单击**确认**。
4. 为路灯设备所属产品配置服务端订阅，用于您的服务器直接订阅产品下多种类型的消息 。
   i. 在左侧导航栏，选择**规则引擎 > 服务端订阅**。
   ii. 在**服务端订阅**页面的**订阅列表**页签下，单击**创建订阅**。

iii. 在**创建订阅**对话框，设置参数后单击**确认**。



参数	说明
产品	选择路灯。
订阅类型	选择AMQP。
消费组	选择上一步创建的路灯消费组。
推送消息类型	选择设备上报消息。

5. AMQP客户端接入物联网平台。

   示例使用Java开发语言，推荐使用Apache Qpid JMS客户端，实现AMQP客户端的接入。您可访问Qpid JMS 0.57.0，查看Qpid JMS客户端使用说明。

   示例使用的开发环境如下：

   - 操作系统：Windows 10 64位
   - JDK版本：JDK8
   - 集成开发环境：Intellij IDEA社区版

   使用示例代码，实现AMQP客户端接入的具体操作如下：

   i. 下载Demo代码包，并解压。

   ii. 打开Intellij IDEA，导入Demo包中的示例工程*amqp-demo*，使用Maven构建工程。

   iii. 在*src/main/java/com.aliyun.iotx.demo*目录下*AmqpClient.java*文件中，参照下表，修改JMS客户端接入物联网平台代码中的参数值。

参数	示例	说明

参数	示例	说明
accessKey	LTAI4GFGQvKuqHJhFa******	登录物联网平台控制台，将鼠标指针移至账号头像上，然后单击**AccessKey管理**，获取AccessKey ID和AccessKey Secret。 **说明** 如果使用RAM用户，您需授予该用户管理物联网平台的权限（AliyunIOTFullAccess），否则将连接失败。授权方法请参见授权RAM用户访问物联网平台。
accessSecret	iMS8ZhCDdfJbCMeA005sieKe******	
consumerGroupId	VWhGZ2QnP7kxWpeSSjt******	步骤3中创建的路灯消费组ID。请在物联网平台控制台公共实例页面，选择**规则引擎 > 服务端订阅**，单击**消费组列表**页签，找到路灯消费组获取消费组ID。
iotInstanceId	■ 新版公共实例的示例：iot-w3*** ■ 旧版公共实例的示例：""	实例ID。 ■ 新版公共实例：在**实例概览**页面的**公共实例**下，查看**ID**值。 ■ 旧版公共实例：没有实例ID，传入空值。 实例的更多详细说明，请参见查看实例终端节点。
clientId	12345	表示客户端ID，建议使用您的AMQP客户端所在服务器UUID、MAC地址、IP等唯一标识。长度不可超过64个字符。 登录物联网平台控制台，在对应实例的**规则引擎 > 服务端订阅 > 消费组列表**，单击消费组对应的**查看**，**消费组详情**页将显示该参数，方便您识别区分不同的客户端。
connectionCount	4	启动AMQP客户端的连接数，最大不超过64个。用于实时消息推送的扩容。 **消费组详情**页面会以 `${clientId}+"-"+数字` 形式，显示连接的客户端。其中数字最小值为0。
host	■ 新版公共实例的示例：iot-***.amqp.iothub.aliyuncs.com ■ 旧版公共实例的示例：198426864******.iot-amqp.cn-shanghai.aliyuncs.com	AMQP接入域名。 详细说明，请参见查看实例终端节点。

iv. 运行示例代码后，返回类似如下日志信息，表示AMQP客户端已接入物联网平台并成功接收消息。

> **说明** 本示例Demo代码中，添加了结束程序的代码（`Thread.sleep(60 * 1000);`），即程序启动成功，运行一分钟后会结束。实际场景中，您可根据需要自行设置运行时间。

```
[main] INFO com.aliyun.iotx.demo.AmqpClient - amqp demo is started successfully, and will exit after 60s
[AmqpProvider :(2):[amqps://                                    .aliyuncs.com:5671]] DEBUG org.apache.qpid.jms.provider.amqp.AmqpConsumer - Dispatching received message: JmsInboundMessageDispatch { sequenc
[JmsSession [ID:d0cd8ff1-5c3                                    ry dispatcher] DEBUG org.apache.qpid.jms.provider.failover.FailoverProvider - Executing Failover Task: message acknowledge -> JmsInboundMessag
[AmqpProvider :(2):[amqps://                                    .aliyuncs.com:5671]] DEBUG org.apache.qpid.jms.provider.amqp.AmqpConsumer - Delivered Ack of message: JmsInboundMessageDispatch { sequence =
[JmsSession [ID:d0cd8ff1-5c3                                    ry dispatcher] DEBUG org.apache.qpid.jms.provider.failover.FailoverProvider - Executing Failover Task: message acknowledge -> JmsInboundMessag
[AmqpProvider :(2):[amqps://                                    .aliyuncs.com:5671]] DEBUG org.apache.qpid.jms.provider.amqp.AmqpConsumer - Accepted Ack of message: JmsInboundMessageDispatch { sequence = 1
[pool-1-thread-1] INFO com.aliyun.iotx.demo.AmqpClient - receive message,
topic = /a1                /device1/thing/event/property/post,
messageId = 1380844478987006465,
content = {"deviceType":"CustomCategory","iotId":"nTT9r                    ","requestId":"1617863381153","checkFailedData":{},"productKey":"a10                    ","gmtCreate":1617863218430,"deviceName":"device1","items":{"Light
```

更多服务端订阅设备消息相关内容，请参见：

相关文档

- 配置AMQP服务端订阅
- AMQP客户端接入说明

2.6. 云端下发指令

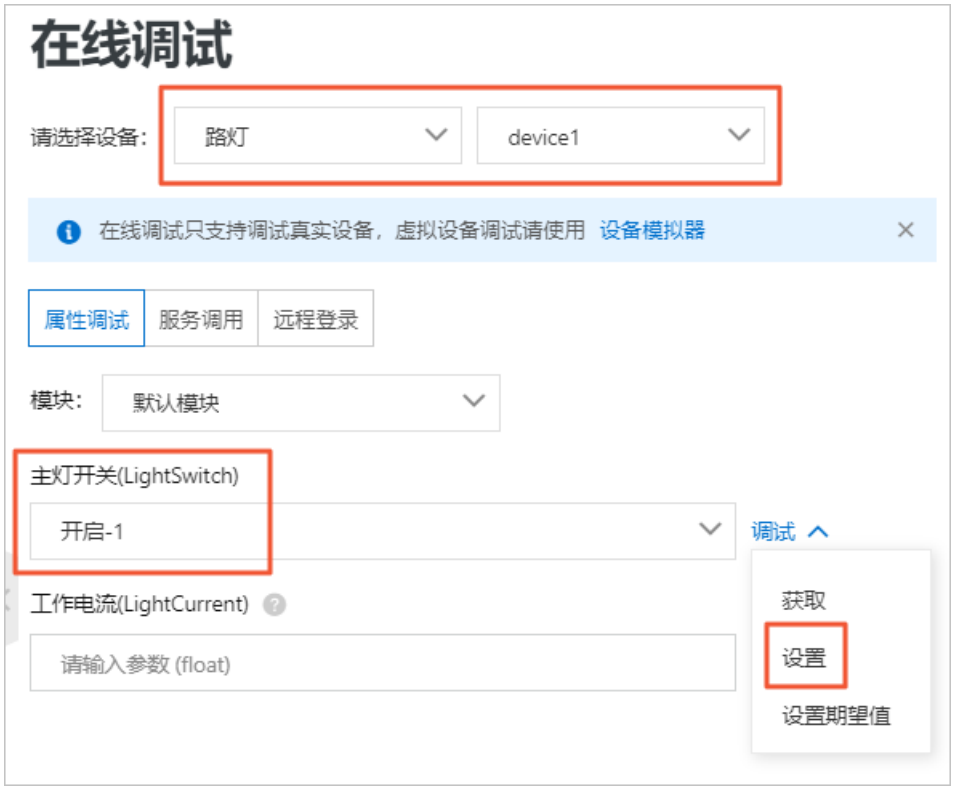
设备成功上报消息后，您可以尝试从云端下发指令到设备端。本文介绍使用物联网平台控制台上的在线调试功能，设置路灯设备属性的操作。

操作步骤

1. 登录物联网平台控制台。
2. 在控制台左上方，选择物联网平台所在地域，然后在**实例概览**页面，单击**公共实例**。

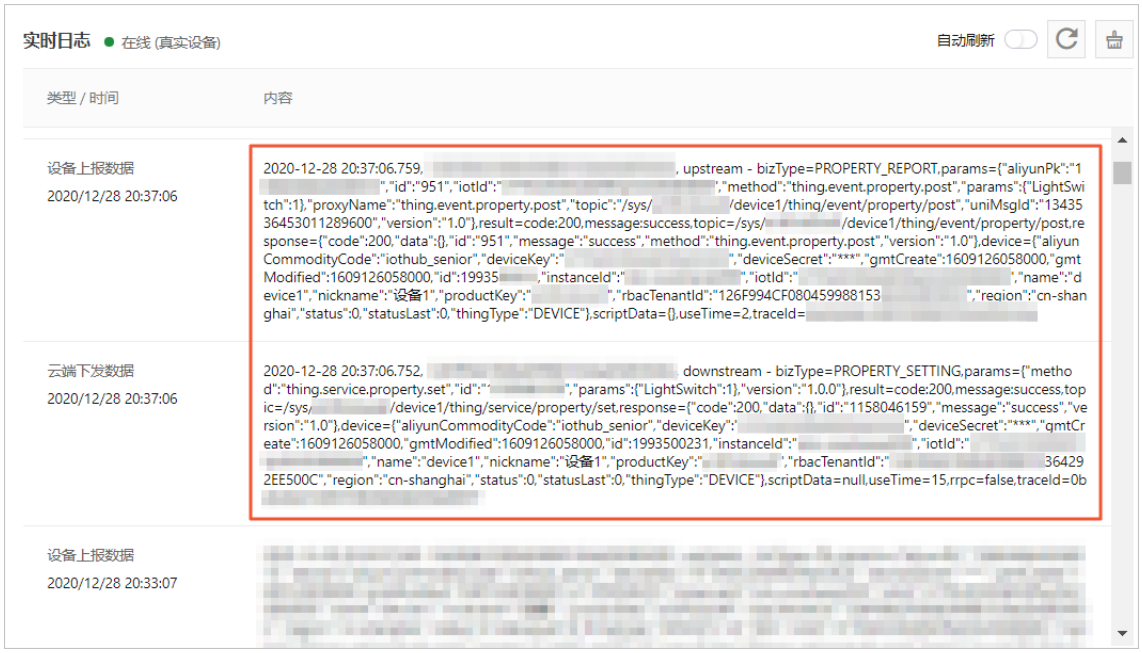
> **注意** 在中国地域，目前仅华东2（上海）地域开通了公共实例服务。

3. 在左侧导航栏，选择**监控运维 > 在线调试**。
4. 在**在线调试**页面，选择路灯产品和device1设备。
5. 在**属性调试**页签，设置调试参数，下发指令。



调试项目	操作步骤
模块	选择**默认模块**。
主灯开关（LightSwitch）	选择**开启-1**，单击下拉框右侧的**调试** > **设置**。
工作电流（LightCurrent）	工作电流是只读属性，本文中体验云端下发指令到设备的功能，因此无需设置该属性。

6. 查看设备属性状态。
   - 在**在线调试**页面的实时日志区域下，查看云端下发数据和设备上报数据。



- 左侧导航栏选择**设备管理 > 设备**，找到目标设备，单击对应操作栏中的**查看**。

  在**设备详情**页面，单击**物模型数据**页签，然后单击**运行状态**，查看设备的运行状态。

7. （可选）如果您希望使用自己的云端应用程序设置设备属性，可参考云端开发指南，下载云端SDK，调用API下发指令。

3.使用MQTT.fx接入物联网平台

MQTT.fx是一款基于Eclipse Paho，使用Java语言编写的MQTT客户端，支持Windows、Mac和Linux操作系统，可用于验证设备是否可与物联网平台正常连接，并通过Topic订阅和发布消息。本文以Windows系统下MQTT.fx为例，介绍模拟设备以MQTT协议接入物联网平台的步骤。

前提条件

已在物联网平台控制台，对应实例下，创建产品和设备，然后在**设备详情**页面，获取设备证书和**MQTT连接参数**的信息。具体操作，请参见：

- 创建产品。
- 创建设备。
- 获取MQTT签名参数值。

本文以旧版公共实例下设备为例，设备证书和MQTT连接参数如下表，参数详细说明，请参见MQTT-TCP连接通信。

参数	值
ProductKey	`a1***`
DeviceName	device1
DeviceSecret	f35***d9e
clientId	`a1***.device1|securemode=2,signmethod=hmacsha256,timestamp=2524608000000|`
username	`device1&a1***`
passwd	`86761***21d`
mqttHostUrl	`a1***.iot-as-mqtt.cn-shanghai.aliyuncs.com`
port	`1883`

> **注意** MQTT.fx模拟的在线设备，仅支持非透传消息通信。如需实现透传信息通信，您可以使用真实设备或SDK进行测试。

视频演示

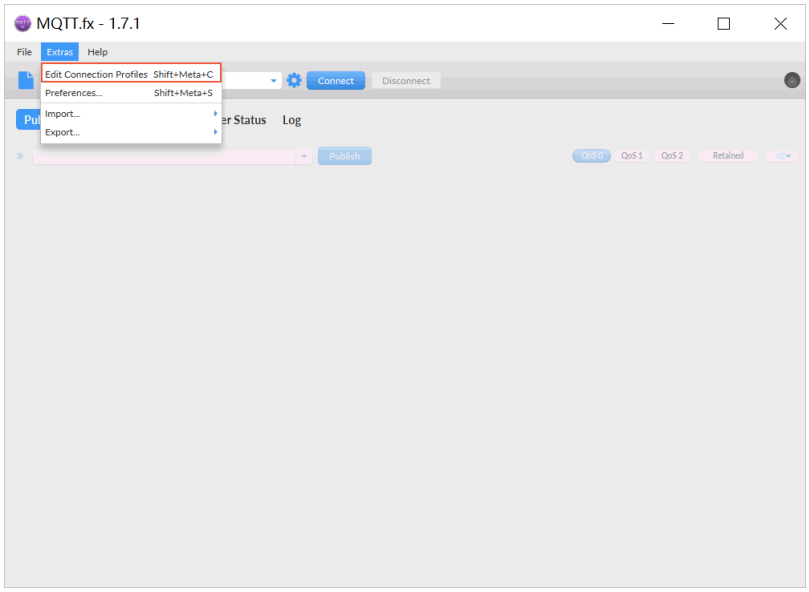
使用MQTT.fx，模拟设备接入物联网平台的操作如下。

配置MQTT.fx接入

1. 下载MQTT.fx Version 1.7.1 for Windows版本软件，完成安装。

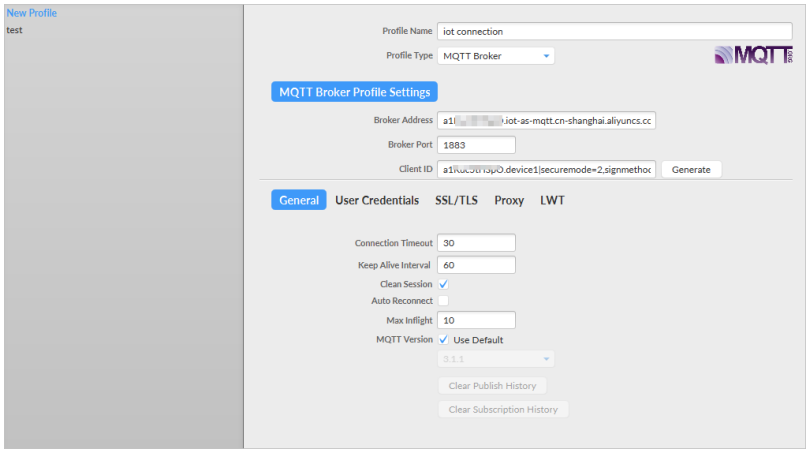
   MQTT.fx更多信息，请访问MQTT.fx官网。

2. 打开MQTT.fx软件，单击菜单栏中的**Extras**，选择**Edit Connection Profiles**。



3. 在**Edit Connection Profiles**页面，完成以下参数的设置。

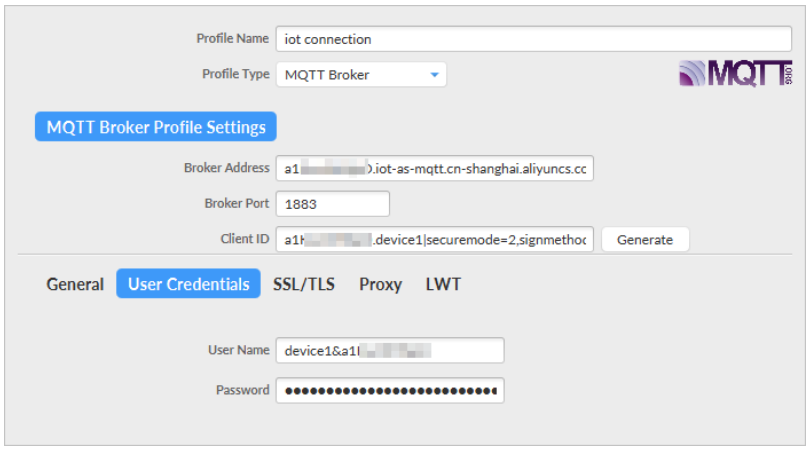
   i. 设置基本信息。



参数	说明
Profile Name	输入您的自定义名称*iot connection*。
Profile Type	MQTT服务器连接，选择**MQTT Broker**。
MQTT Broker Profile Settings	
Broker Address	MQTT接入域名，对应*前提条件*中已获取的mqttHostUrl值：*a1***.iot-as-mqtt.cn-shanghai.aliyuncs.com*。 ■ *a1****为本示例产品的ProductKey。 ■ *cn-shanghai*为本示例所在地域。 此处仅输入域名，无需携带端口号。
Broker Port	设置为*1883*。

参数	说明
Client ID	MQTT的协议字段。 固定格式：`${ClientID}\|securemode=${Mode},signmethod=${SignMethod}\|timestamp=${timestamp}|`。 输入*前提条件*中已获取的clientId值：*a1***.device1|securemode=2,signmethod=hmacsha256,timestamp=2524608000000|*。 参数说明： ■ *${ClientId}*：设备、App或Web等场景下的Client ID信息。 **注意** 该值可自定义，长度在64个字符以内。若为设备的ID信息，建议使用您设备的MAC地址或SN码，方便您识别区分不同的设备。 ■ *${Mode}*：安全模式。可选值有*2*（TLS直连模式，需要设置**SSL/TLS**信息）和*3*（TCP直连模式，无需设置**SSL/TLS**信息）。 ■ *${SignMethod}*：算法类型，支持*hmacsha256*、*hmacmd5*和*hmacsha1*。 ■ *${timestamp}*：表示当前时间毫秒值，可以不传递timestamp。 物联网平台提供的连接参数中*${ClientId}*默认为`${ProductKey} + '.' + ${DeviceName}`组成的字符串，*${Mode}*默认为*2*，*${SignMethod}*默认为*hmacsha256*，您可根据需要修改。 **注意** ■ MQTT.fx的**Client ID**和设备的*${ClientId}*，切勿混淆。 ■ 不要遗漏参数之间及最后的竖线（|）。 ■ 设置参数时，请确保参数值中或参数值的前后均没有空格。 ■ 输入**Client ID**信息后，请勿单击**Generate**。
General	本示例使用默认值。您也可以根据实际场景需求设置。

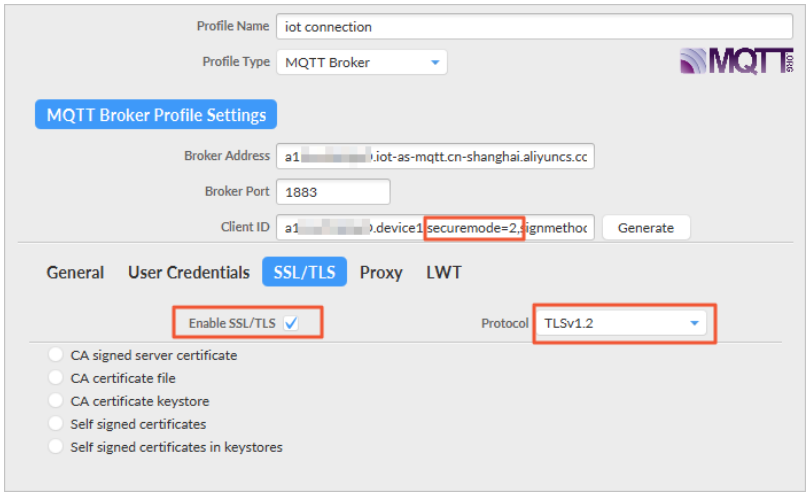
ii. 单击User Credentials，设置User Name和Password。



参数	参数说明
User Name	由设备名DeviceName、and（&）和产品ProductKey组成，固定格式为 `${DeviceName}&${ProductKey}` 。 输入*前提条件*中已获取的username值：*device1&a1****。 ■ `device1` 为设备的DeviceName。 ■ `a1***` 为设备的ProductKey。
Password	通过选择的加密方法，以设备的DeviceSecret为密钥，将参数和参数值拼接后，加密生成Password。 输入*前提条件*中已获取的passwd值：*86761****21d*。 **注意** ■ 如果您使用的MQTT.fx版本，在粘贴Password后不显示具体的字符串，只要光标已从输入框的前部移至了后部，则表示粘贴成功，请勿重复粘贴。 ■ 请注意参数和参数值中字母的大小写。 ■ 若您设置Client ID时，修改了已获取值中的*${clientId}*、*${SignMethod}*的设置，必须保证加密参数的值与Client ID中对应参数值一致，重新计算出Password。相关参数设置与计算方法，请参见使用网页工具计算和使用Node.js语言脚本计算。

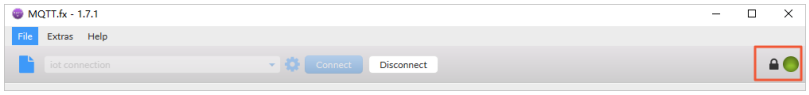
iii. TLS直连模式（即securemode=2）下，单击**SSL/TLS**，选中**Enable SSL/TLS**，设置**Protocol**为**TLSv1.2**。

> **注意** TCP直连模式（即 `securemode=3` ）下，无需设置SSL/TLS信息，直接进入下一步。



4. 设置完成后，单击右下角的**OK**。
5. 单击**Connect**。

   右侧亮绿灯，表示连接成功。



   您可在物联网平台控制台，在对应实例下，选择**设备管理 > 设备**，选择产品，查看该设备状态，预期设备为**在线**状态。

下文通过测试自定义Topic的上下行通信，验证MQTT.fx与物联网平台连接是否成功。若测试与本示例结果不符，表示通信连接失败，您需根据日志信息，进行修正。

测试下行通信

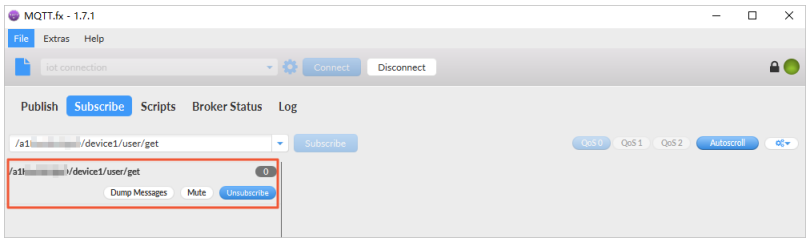
1. 在物联网平台控制台对应实例下的产品详情页面，单击**Topic类列表 > 自定义Topic**，找到一个具有**订阅**权限的自定义Topic。

   本示例使用Topic： `/a1***/${deviceName}/user/get` ，您需替换 `${deviceName}` 为设备名称 `device1` 。

   更多信息，请参见自定义Topic。

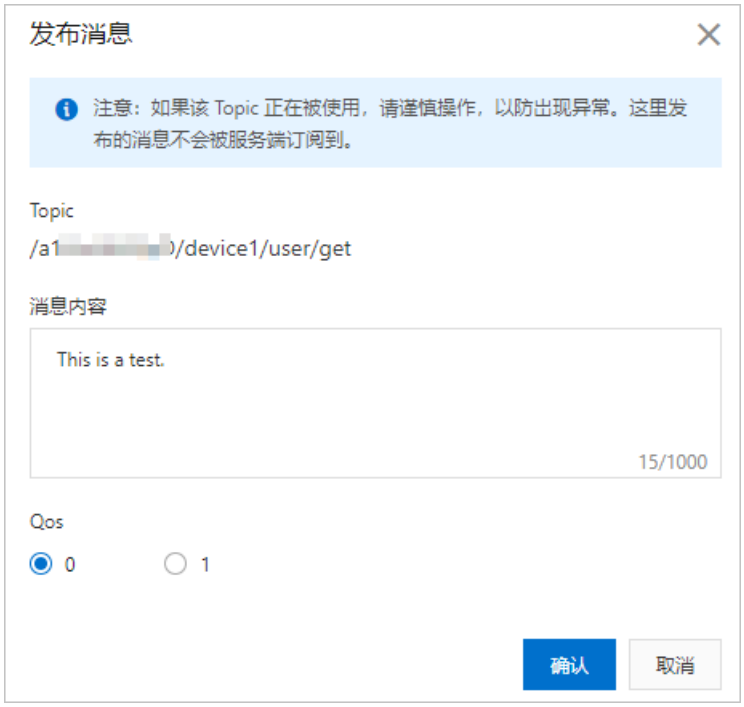
2. 在MQTT.fx上单击**Subscribe**，在**Subscribe**文本框中，输入上一步的Topic，再单击**Subscribe**。

   订阅成功后，该Topic将显示在列表中。

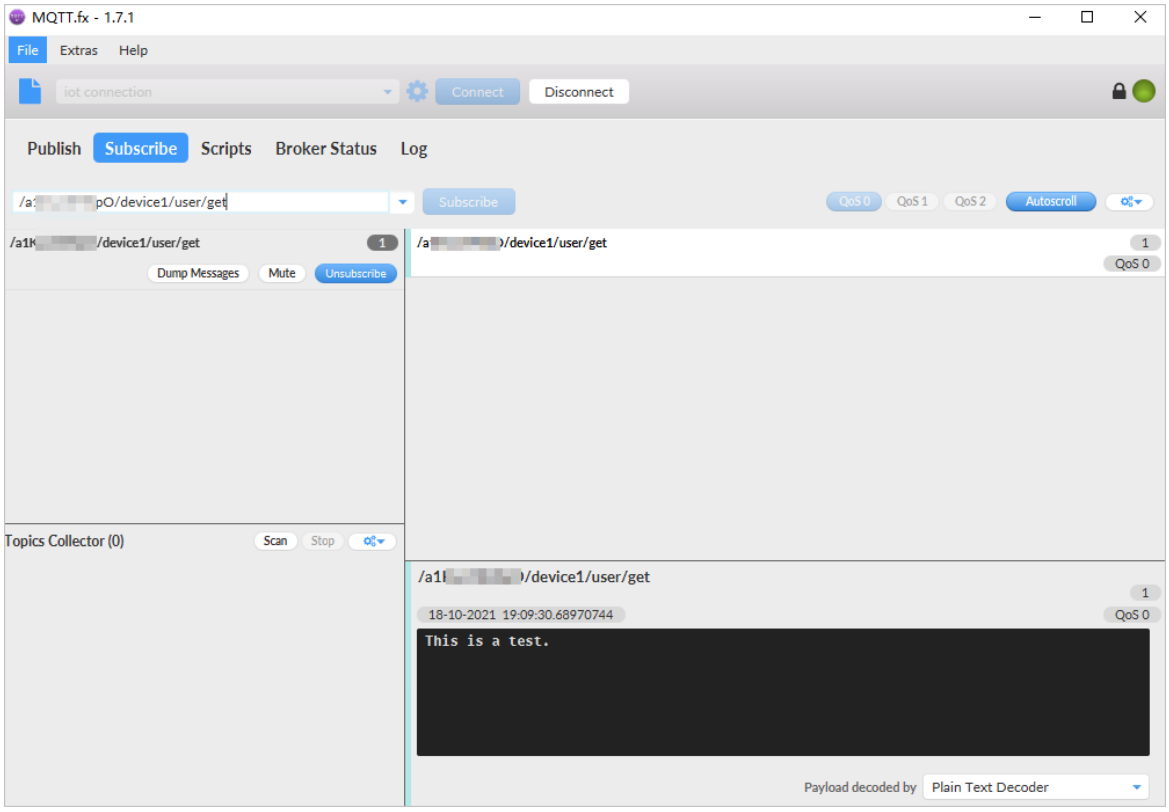


3. 返回物联网平台，进入该设备的**设备详情**页面，在**Topic列表**页签下，单击已订阅Topic对应的**发布消息**。

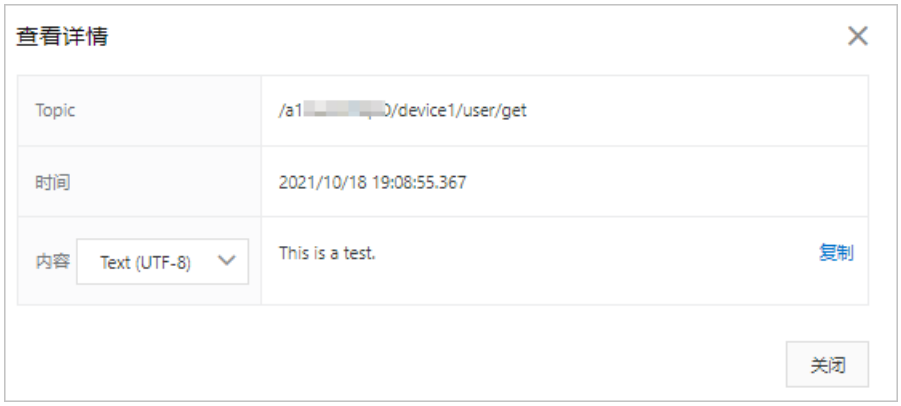
4. 输入消息内容，单击**确认**。



5. 回到MQTT.fx上，查看接收到的消息。



6. 回到物联网平台，在**设备详情**页面，单击**日志服务**页签的**前往查看**，在**日志服务**页面，查看**云到设备消息**。



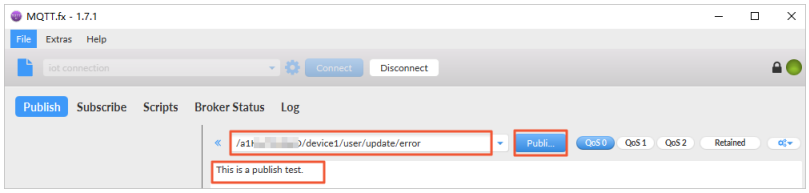
测试上行通信

1. 在物联网平台控制台对应实例下的产品详情页面，单击**Topic类列表 > 自定义Topic**，找到一个具有**发布**权限的自定义Topic。

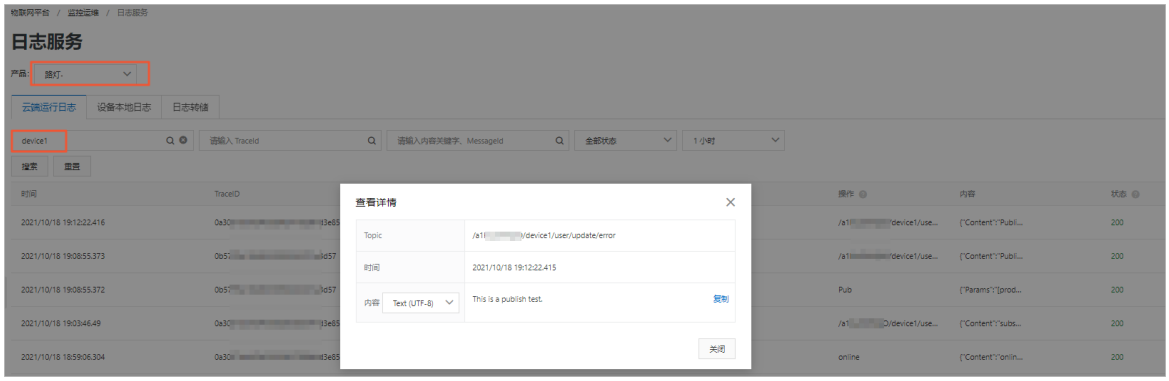
   本示例使用Topic：`/a1***/${deviceName}/user/update/error`，您需替换`${deviceName}`为设备名称`device1`。

   更多信息，请参见自定义Topic。

2. 在MQTT.fx上，单击**Publish**，在**Publish**文本框中，输入上一步的Topic。在文本编辑页面，输入要发送的消息内容，然后单击**Publish**。

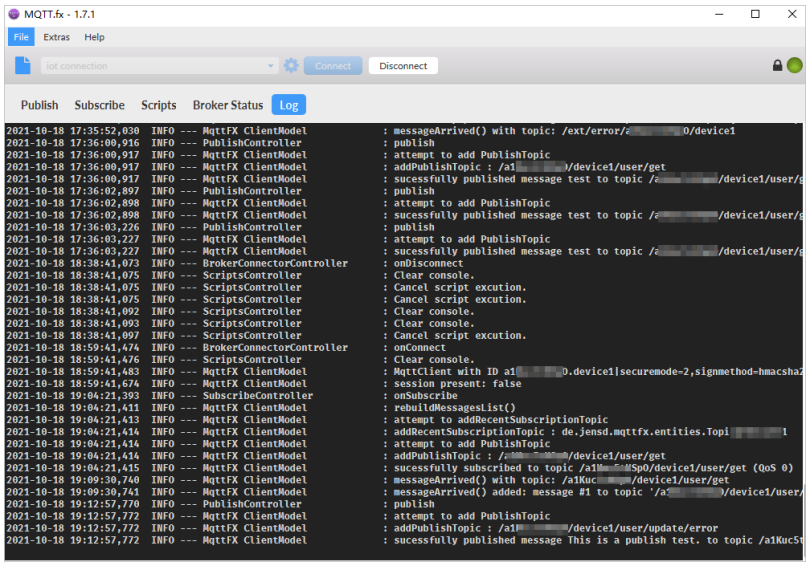


3. 回到物联网平台，在**设备详情**页面，单击**日志服务**页签的**前往查看**，在**日志服务**页面，查看**设备到云消息**。



查看日志

在MQTT.fx上，单击**Log**查看操作日志和错误提示日志。



4.使用自定义Topic进行通信

您可以在物联网平台上自定义Topic类，设备将消息发送到自定义Topic中，服务端通过AMQP SDK获取设备上报消息；服务端通过调用物联网平台接口Pub向设备发布指令。自定义Topic通信不使用物模型，消息的数据结构由您自定义。

背景信息

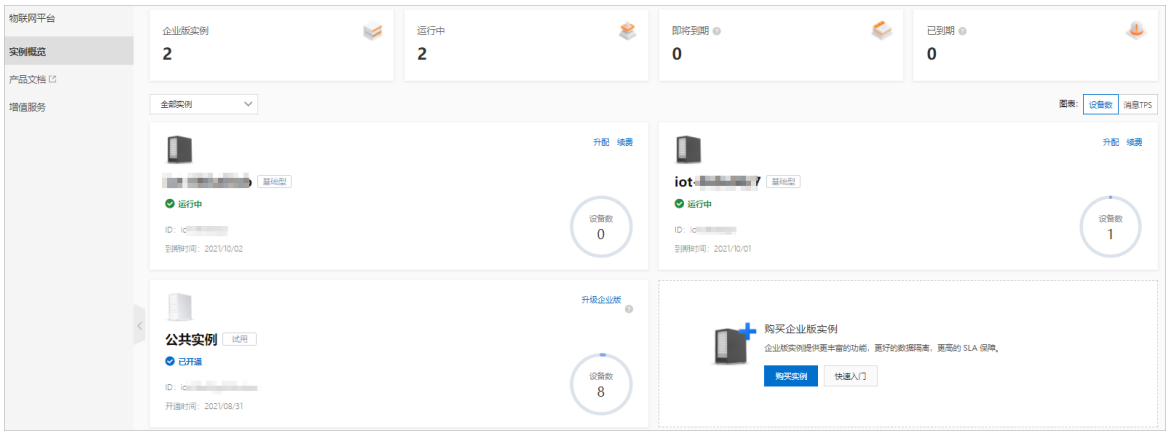
本示例中，电子温度计定期与服务器进行数据的交互，传递温度和指令等信息。温度计向服务器上行发送当前的温度；服务器向温度计下行发送精度设置指令。

准备开发环境

本示例中，设备端和云端均使用Java语言的SDK，需先准备Java开发环境。您可从Java 官方网站下载，并安装Java最新开发环境。

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，单击**设备管理 > 产品**。
4. 单击**创建产品**，创建温度计产品，获取productKey，例如 `a1uzcH0****` 。

   详细操作指导，请参见创建产品。
5. 产品创建成功后，单击该产品对应的**查看**。
6. 在**产品详情**页的**Topic类列表**页签下，单击**自定义Topic**，增加自定义Topic类。

   详细操作指导，请参见自定义Topic。

   本示例中，定义了以下两个Topic类：

   - 设备发布消息Topic：/a1uzcH0****/${deviceName}/user/devmsg，权限为发布。

- 设备订阅消息Topic：/a1uzcH0****/${deviceName}/user/cloudmsg，权限为订阅。

7. 在**服务端订阅**页签下，单击**创建订阅**，设置AMQP服务端订阅，订阅**设备上报消息**到**默认消费组**。

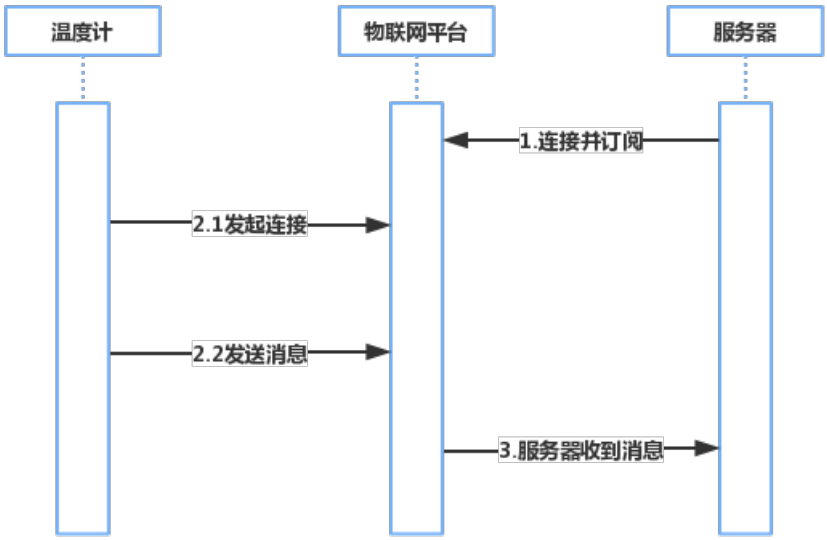
   详细操作指导，请参见配置AMQP服务端订阅。

8. 在左侧导航栏，单击**设备**，然后在刚创建的温度计产品下，添加设备**device1**，获取设备证书**ProductKey**、**DeviceName**和**DeviceSecret**。

   详细操作指导，请参见单个创建设备。

设备发送消息给服务器

流程图：



在整个流程中：

- 服务器通过AMQP客户端接收消息，需配置AMQP客户端接入物联网平台，监听设备消息。具体操作，请参见Java SDK接入示例。

  > **注意** AMQP订阅消息的消费组，必须与设备在相同的物联网平台实例下。

- 配置设备端SDK接入物联网平台，并发送消息。
  - 配置设备认证信息。

```
final String productKey = "a1uzcH0****";
final String deviceName = "device1";
final String deviceSecret = "uwMTmVAMnxB***";
final String region = "cn-shanghai";
```

- 设置初始化连接参数，包括MQTT连接配置、设备信息和初始物模型属性。

```
LinkKitInitParams params = new LinkKitInitParams();
//LinkKit底层是MQTT协议，设置MQTT的配置。
IoTMqttClientConfig config = new IoTMqttClientConfig();
config.productKey = productKey;
config.deviceName = deviceName;
config.deviceSecret = deviceSecret;
config.channelHost = productKey + ".iot-as-mqtt." + region + ".aliyuncs.com:1883";
//设备的信息。
DeviceInfo deviceInfo = new DeviceInfo();
deviceInfo.productKey = productKey;
deviceInfo.deviceName = deviceName;
deviceInfo.deviceSecret = deviceSecret;
//报备的设备初始状态。
Map<String, ValueWrapper> propertyValues = new HashMap<String, ValueWrapper>();
params.mqttClientConfig = config;
params.deviceInfo = deviceInfo;
params.propertyValues = propertyValues;
```

- 初始化连接。

```
//连接并设置连接成功以后的回调函数。
LinkKit.getInstance().init(params, new ILinkKitConnectListener() {
     @Override
     public void onError(AError aError) {
         System.out.println("Init error:" + aError);
     }
     //初始化成功以后的回调。
     @Override
     public void onInitDone(InitResult initResult) {
         System.out.println("Init done:" + initResult);
     }
 });
```

- 设备发送消息。

  设备端连接物联网平台后，向以上定义的Topic发送消息。需将onInitDone函数内容替换为以下内容：

```
@Override
 public void onInitDone(InitResult initResult) {
     //设置Pub消息的Topic和内容。
     MqttPublishRequest request = new MqttPublishRequest();
     request.topic = "/" + productKey + "/" + deviceName + "/user/devmsg";
     request.qos = 0;
     request.payloadObj = "{\"temperature\":35.0, \"time\":\"sometime\"}";
     //发送消息并设置成功以后的回调。
     LinkKit.getInstance().publish(request, new IConnectSendListener() {
         @Override
         public void onResponse(ARequest aRequest, AResponse aResponse) {
             System.out.println("onResponse:" + aResponse.getData());
         }
         @Override
         public void onFailure(ARequest aRequest, AError aError) {
             System.out.println("onFailure:" + aError.getCode() + aError.getMsg());
         }
     });
 }
```

  服务器收到消息如下：

```
Message
{payload={"temperature":35.0, "time":"sometime"},
topic='/a1uzcH0****/device1/user/devmsg',
messageId='1131755639450642944',
qos=0,
generateTime=1558666546105}
```

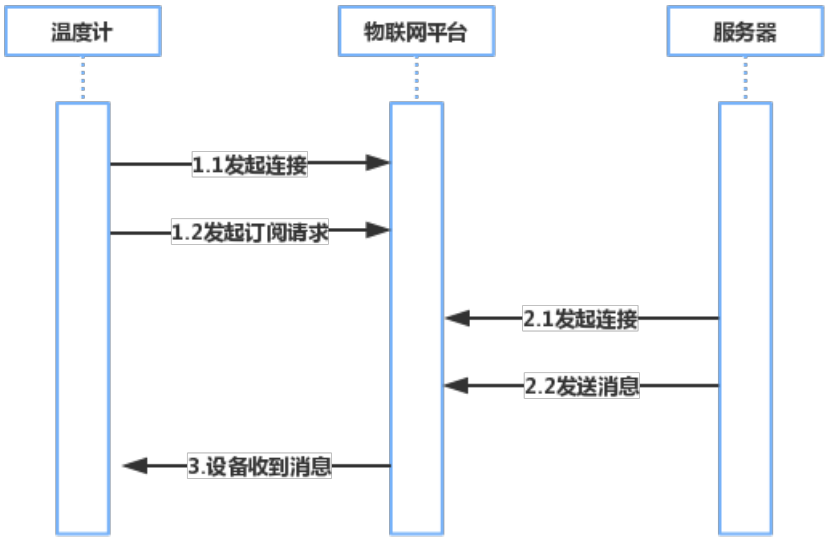
实际业务场景中，您需修改以下参数值。

参数	示例	说明
productKey	a1uzcH0****	设备认证信息，请参见创建产品和设备中的步骤8。
deviceName	device1	
deviceSecret	uwMTmVAMnxB***	
region	cn-shanghai	您的物联网平台设备所在地域代码。地域代码表达方法，请参见地域和可用区。
channelHost	iot-o***.mqtt.iothub.aliyuncs.com:1883	设备接入地址。详细说明，请参见查看实例终端节点。 注意 接入域名若没有携带端口号1883，需添加该端口号。

参数	示例	说明
request.topic	"/" + productKey + "/" + deviceName + "/user/devmsg"	具有发布权限的自定义Topic。
request.payloadObj	"{\"temperature\":35.0,\"time\":\"sometime\"}"	自定义的消息内容。

服务器发送消息给设备

流程图：



- 配置设备端SDK订阅Topic。

  配置设备认证信息、设置初始化连接参数、初始化连接，请参见设备发送消息给服务器中的相应示例代码。

  设备要接收服务器发送的消息，还需订阅消息Topic。

  配置设备端订阅Topic示例如下：

```
//初始化成功以后的回调。
@Override
public void onInitDone(InitResult initResult) {
    //设置订阅的Topic。
    MqttSubscribeRequest request = new MqttSubscribeRequest();
    request.topic = "/" + productKey + "/" + deviceName + "/user/cloudmsg";
    request.isSubscribe = true;
    //发出订阅请求并设置订阅成功或者失败的回调函数。
    LinkKit.getInstance().subscribe(request, new IConnectSubscribeListener() {
        @Override
        public void onSuccess() {
            System.out.println("");
        }
        @Override
        public void onFailure(AError aError) {
        }
    });
    //设置订阅的下行消息到来时的回调函数。
    IConnectNotifyListener notifyListener = new IConnectNotifyListener() {
        //此处定义收到下行消息以后的回调函数。
        @Override
        public void onNotify(String connectId, String topic, AMessage aMessage) {
            System.out.println(
                "received message from " + topic + ":" + new String((byte[])aMessage.getD
ata()));
        }
        @Override
        public boolean shouldHandle(String s, String s1) {
            return false;
        }
        @Override
        public void onConnectStateChange(String s, ConnectState connectState) {
        }
    };
    LinkKit.getInstance().registerOnNotifyListener(notifyListener);
}
```

其中，`request.topic`值需修改为具有订阅权限的自定义Topic。

- 配置云端SDK调用物联网平台接口Pub发布消息。参数说明，请参见Pub，使用说明，请参见Java SDK使用说明。
  - 设置身份认证信息。

```
String regionId = "XXXXXX";
String accessKey = "XXXXXX";
String accessSecret = "XXXXXXXXX";
final String productKey = "XXXXXX";
```

  - 设置连接参数。

```
//设置client的参数。
DefaultProfile profile = DefaultProfile.getProfile(regionId, accessKey, accessSecret);
IAcsClient client = new DefaultAcsClient(profile);
```

- 设置消息发布参数。

```
PubRequest request = new PubRequest();
request.setQos(0);
//设置发布消息的Topic。
request.setTopicFullName("/" + productKey + "/" + deviceName + "/user/cloudmsg");
request.setProductKey(productKey);
//设置消息的内容，一定要用Base64编码，否则乱码。
request.setMessageContent(Base64.encode("{\"accuracy\":0.001,\"time\":now}"));
```

- 发送消息。

```
try {
     PubResponse response = client.getAcsResponse(request);
     System.out.println("pub success?:" + response.getSuccess());
 } catch (Exception e) {
     System.out.println(e);
 }
```

设备端接收到的消息如下：

```
msg = [{"accuracy":0.001,"time":now}]
```

附录：代码示例

> **说明** 实际业务场景中，请按照上文描述，修改代码中相关参数的值。

下载Pub/Sub demo，包含本示例的云端SDK和设备端SDK配置代码Demo。

AMQP客户端接入物联网平台示例，请参见：

- Java SDK接入示例
- .NET SDK接入示例
- Node.js SDK接入示例
- Python 2.7 SDK接入示例
- Python3 SDK接入示例
- PHP SDK接入示例
- Go SDK接入示例

5.设备消息通过RocketMQ流转到服务器

物联网平台将设备上报的数据流转至消息队列RocketMQ的Topic中，然后，RocketMQ再将数据流转到企业服务器。本文介绍数据流转的操作步骤。

前提条件

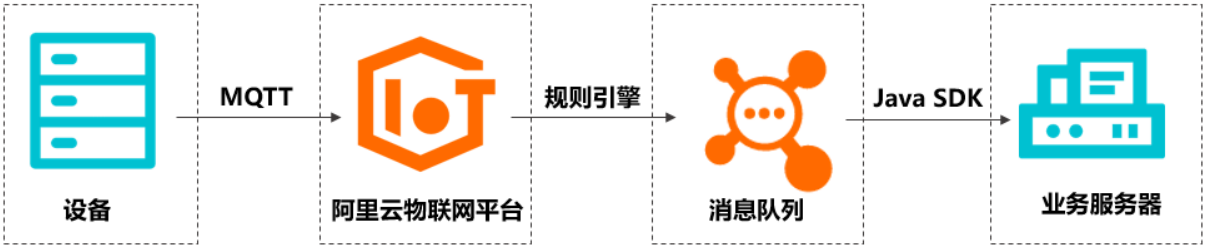
- 已注册阿里云账号。
- 已开通阿里云物联网平台服务。

  如未开通，请登录物联网平台产品页，单击**立即开通**，根据页面提示，开通服务。
- 已开通消息队列RocketMQ服务。

  如未开通，请登录消息队列RocketMQ产品页面，开通服务。
- 准备开发环境。本示例使用Java SDK开发的环境如下：
  - 操作系统：Windows 10 64位
  - JDK版本：JDK8
  - 集成开发环境：IntellIJ IDEA社区版

背景信息

数据流转流程图：

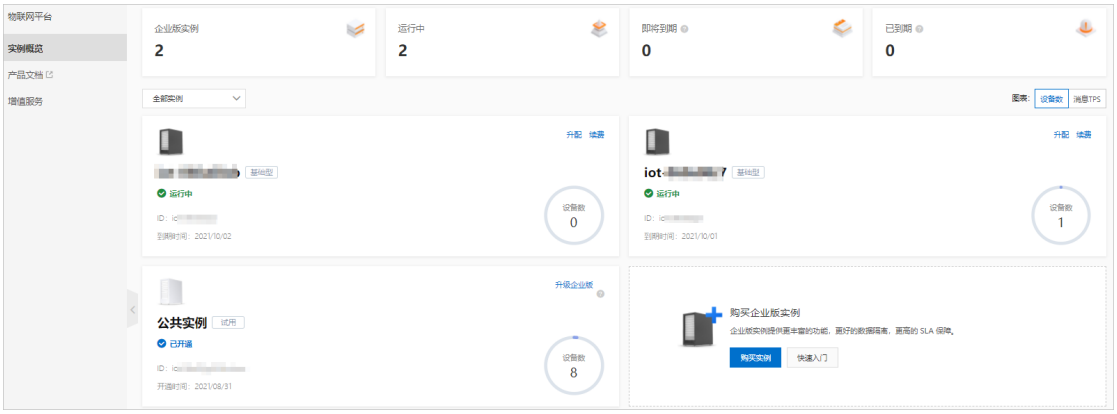


方案优势：

- 设备使用MQTT协议接入物联网平台，数据传输链路支持TLS加密，保障数据不被篡改。MQTT协议说明，请参见MQTT协议规范。
- 通过消息队列RocketMQ削峰填谷，缓冲消息，减轻服务器同时接收大量设备消息的压力。

操作步骤

1. 登录物联网控制台，创建产品和设备。
   i. 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。



ii. 在左侧导航栏选择**设备管理 > 产品**，单击**创建产品**，配置参数，单击**确认**。

本示例中，产品名称为MQ_test，所属品类为**自定义品类**，节点类型为**直连设备**，其他参数使用默认值。

iii. 单击**查看产品详情**，在产品详情页面，单击**Topic类列表 > 自定义Topic**，然后单击**自定义Topic类**，定义一个用于设备上报数据的Topic。

本示例中，定义的Topic类：`/{YourProductKey}/${YourDeviceName}/user/data`。

iv. 在左侧导航栏选择**设备管理 > 设备**，单击**添加设备**，为产品MQ_test创建设备。

本示例中，创建了一个名称为MQdevice的设备。

2. 在消息队列RocketMQ控制台，创建Topic和消费者。

i. 登录消息队列RocketMQ控制台。

ii. 单击**创建实例**，创建一个标准版实例，地域选择**华东2（上海）**。

iii. 单击**创建 Group**，如下图所示。

iv. 单击**创建Topic**，消息类型选择**普通消息**。

v. 创建消息消费者，然后在RocketMQ控制台查看消费者状态，确保消费者处于在线状态，订阅关系一致。

本文以调用TCP协议的SDK为例，进行收发消息。SDK获取和使用的详细内容，请参见调用TCP协议的SDK收发普通消息。

Java语言的SDK代码示例如下：

> **说明**
>
> - 如何查看AccessKey和SecretKey，请参见管理AccessKey。
> - RocketMQ详细操作指导，请参见消息队列RocketMQ文档。

```
import com.aliyun.openservices.ons.api.Action;
import com.aliyun.openservices.ons.api.ConsumeContext;
import com.aliyun.openservices.ons.api.Consumer;
import com.aliyun.openservices.ons.api.Message;
import com.aliyun.openservices.ons.api.MessageListener;
import com.aliyun.openservices.ons.api.ONSFactory;
import com.aliyun.openservices.ons.api.PropertyKeyConst;
import java.util.Properties;
public class ConsumerTest {
    public static void main(String[] args) {
        Properties properties = new Properties();
        // 您在控制台创建的 Group ID
        properties.put(PropertyKeyConst.GROUP_ID, "XXX");
        // AccessKey 阿里云身份验证，在阿里云服务器管理控制台创建
        properties.put(PropertyKeyConst.AccessKey, "${AccessKey}");
        // SecretKey 阿里云身份验证，在阿里云服务器管理控制台创建
        properties.put(PropertyKeyConst.SecretKey, "${SecretKey}");
        // 设置 TCP 接入域名，到控制台的实例基本信息中查看
        properties.put(PropertyKeyConst.NAMESRV_ADDR,
            "XXX");
        // 集群订阅方式 (默认)
        // properties.put(PropertyKeyConst.MessageModel, PropertyValueConst.CLUSTER
ING);
        // 广播订阅方式
        // properties.put(PropertyKeyConst.MessageModel, PropertyValueConst.BROADCA
STING);
        Consumer consumer = ONSFactory.createConsumer(properties);
        consumer.subscribe("iot_to_mq", "*", new MessageListener() { //订阅多个 Tag
            public Action consume(Message message, ConsumeContext context) {
                System.out.println("Receive: " + message);
                return Action.CommitMessage;
            }
        });
        consumer.start();
        System.out.println("Consumer Started");
    }
}
```

3. 登录物联网平台控制台，在对应实例下，设置数据流转规则，将设备上报的数据转发至消息队列（RocketMQ）。

i. 单击**规则引擎 > 云产品流转**，然后单击**创建规则**，创建一条数据流转规则MQ流转，数据格式选择为JSON。

ii. 单击**编写SQL**，设置数据处理SQL，如下图所示。

iii. 单击**添加操作**设置数据转发目的地。

iv. 所有设置完成后，返回至**云产品流转**页面，单击MQ流转规则对应的**启动**。
规则启动后，物联网平台会将规则SQL中定义的设备上报消息转发至消息队列（RocketMQ）的Topic中。

4. 使用Java SDK模拟设备，上报消息。

   i. 下载Java SDK Demo，然后解压。

   ii. 在IntelliJ IDEA中，导入Demo包中的示例工程JavaLinkKitDemo。

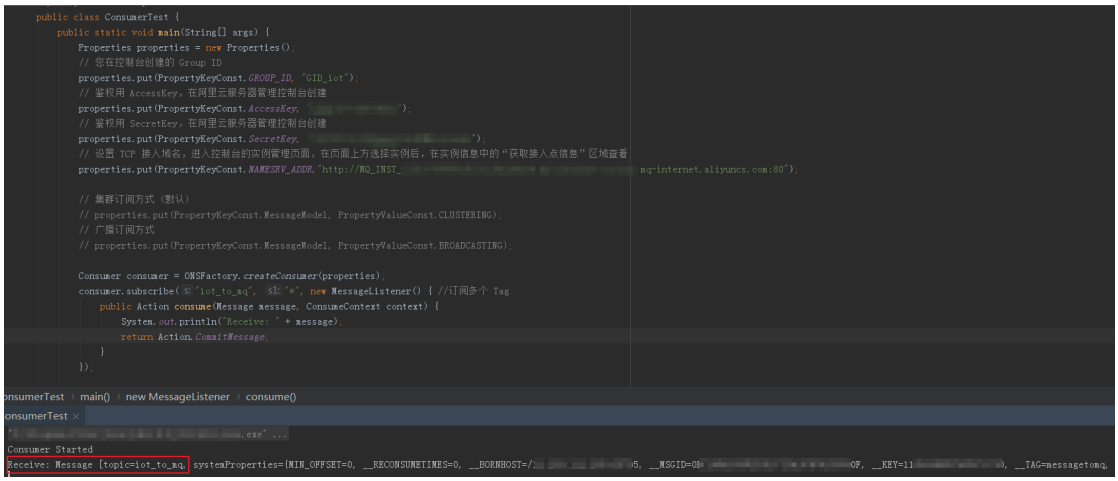
   iii. 在文件*device_id.json*中输入MQdevice的设备证书信息：productKey、deviceName和deviceSecret。

   iv. 在文件*src\main\java\com.aliyun.alink.devicesdk.demo\MqttSample.java*中修改MQTT Topic为设备上报数据的Topic。本示例中，使用的Topic是 `/{YourProductKey}/${YourDeviceName}/user/data` 。

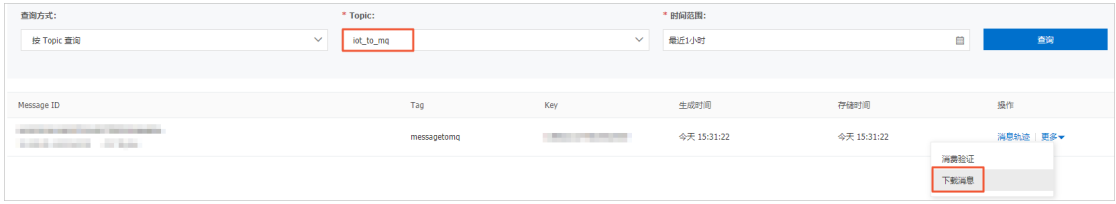
   v. 运行*src\main\java\com.aliyun.alink.devicesdk.demo\HelloWorld.java*文件，启动设备。

登录物联网平台控制台，在对应的实例下，选择**监控运维 > 日志服务**，查看该设备的日志信息，发现设备数据成功转发至RocketMQ。

5. 在RocketMQ控制台查看消息。
   i. 在本地运行订阅消息队列RocketMQ资源的代码。



   ii. 在消息队列RocketMQ控制台，**消息查询**页面，按Topic或者Message ID查询消息，您可下载流转至消息队列RocketMQ中的消息。



消息内容如下：

```
{"deviceName":"MQdevice"}
```