设备接入

快速入门

文档版本 1.0

发布日期 2024-09-23

华为云计算技术有限公司

商标声明


和其他华为商标均为华为技术有限公司的商标。
本文档提及的其他所有商标或注册商标，由各自的所有人拥有。

注意

您购买的产品、服务或特性等应受华为云计算技术有限公司商业合同和条款的约束，本文档中描述的全部或部分产品、服务或特性可能不在您的购买或使用范围之内。除非合同另有约定，华为云计算技术有限公司对本文档内容不做任何明示或暗示的声明或保证。

由于产品版本升级或其他原因，本文档内容会不定期进行更新。除非另有约定，本文档仅作为使用指导，本文档中的所有陈述、信息和建议不构成任何明示或暗示的担保。

目 录

1 设备快速接入—属性上报与命令接收

1.1 开通服务

本章节介绍如何在“中国-香港”区域开通一个标准版免费实例单元，以进行IoTDA平台快速入门的体验。

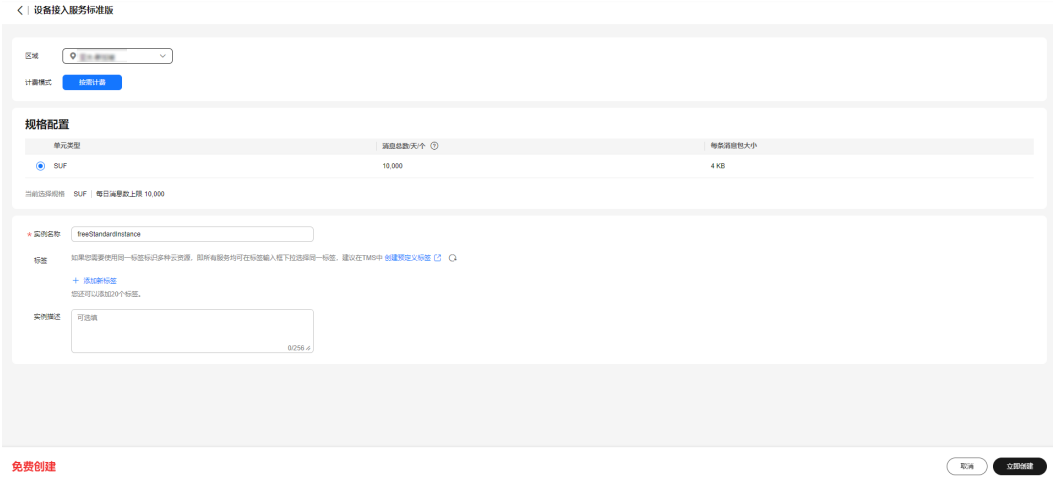
步骤1 访问**设备接入服务**，单击“控制台”进入设备接入控制台。

步骤2 在左侧导航栏，选择“IoTDA实例”，单击“开通免费单元”。

图 1-1 实例-标准版-开通免费实例

步骤3 按下图选择配置信息，均采用默认配置即可。

图 1-2 实例-免费实例配置



步骤4 单击“立即创建”，进入实例页面，刷新页面，等待实例状态变为“运行中”，即表示免费实例成功创建。

图 1-3 实例-免费实例创建完成

----结束

1.2 向导式体验智慧烟感接入平台

场景说明

如果您没有IoT设备，但想快速体验设备数据采集和接收控制命令，可以用Windows或者Linux个人计算机作为虚拟设备，体验设备与云端的双向通信。本文以一款虚拟的智慧烟感器为例，带您快速体验平台的三个基本功能：设备连接平台，设备上报数据到平台，平台下发命令给设备。

前提条件

- 已注册华为云官方账号。未注册可单击**注册页面**完成注册。
- 已开通设备接入服务。未开通则访问**设备接入服务**，单击“管理控制台”后开通该服务。

操作步骤

步骤1 进入**设备接入服务**，单击“管理控制台”。选择您的实例，单击实例卡片进入。

步骤2 在左侧的“总览”菜单栏里，单击“快速体验”按钮即可开始体验。

图 1-4 向导式极速体验-开始

步骤3 本次向导式体验已经给您预先定义好了一款智慧烟感模型。在弹出界面中查看模型的属性和命令，然后单击“创建产品”。

图 1-5 向导式极速体验-创建产品

步骤4 接下来您可以创建一个虚拟的智慧烟感设备。您可以自定义设备标识码和设备名称。单击“注册设备”。

图 1-6 向导式极速体验-注册设备

步骤5 根据您的实际情况，选择设备演示包。

图 1-7 向导式极速体验-选择设备演示包

步骤6 根据界面提示，单击“下载设备演示包”并解压，执行huaweicloud-iot-device-quickstart.exe，您会发现设备状态从“未激活”变为“在线”，并且temperature等属性已有上报值，说明设备已成功接入平台。

图 1-8 向导式极速体验-配置模拟设备

图 1-9 向导式极速体验-模拟设备状态变化

步骤7 您可以手动设置不同的“setReportingFrequency”设备属性上报频率，单击“下发命令”将新的属性上报频率值下发给设备，然后感受设置前后属性上报值刷新速度的变化。

图 1-10 向导式极速体验-设备属性上报频率

----结束

进阶体验

按照本页面的指导，您应该已经基本了解设备连接平台的步骤及相关概念。

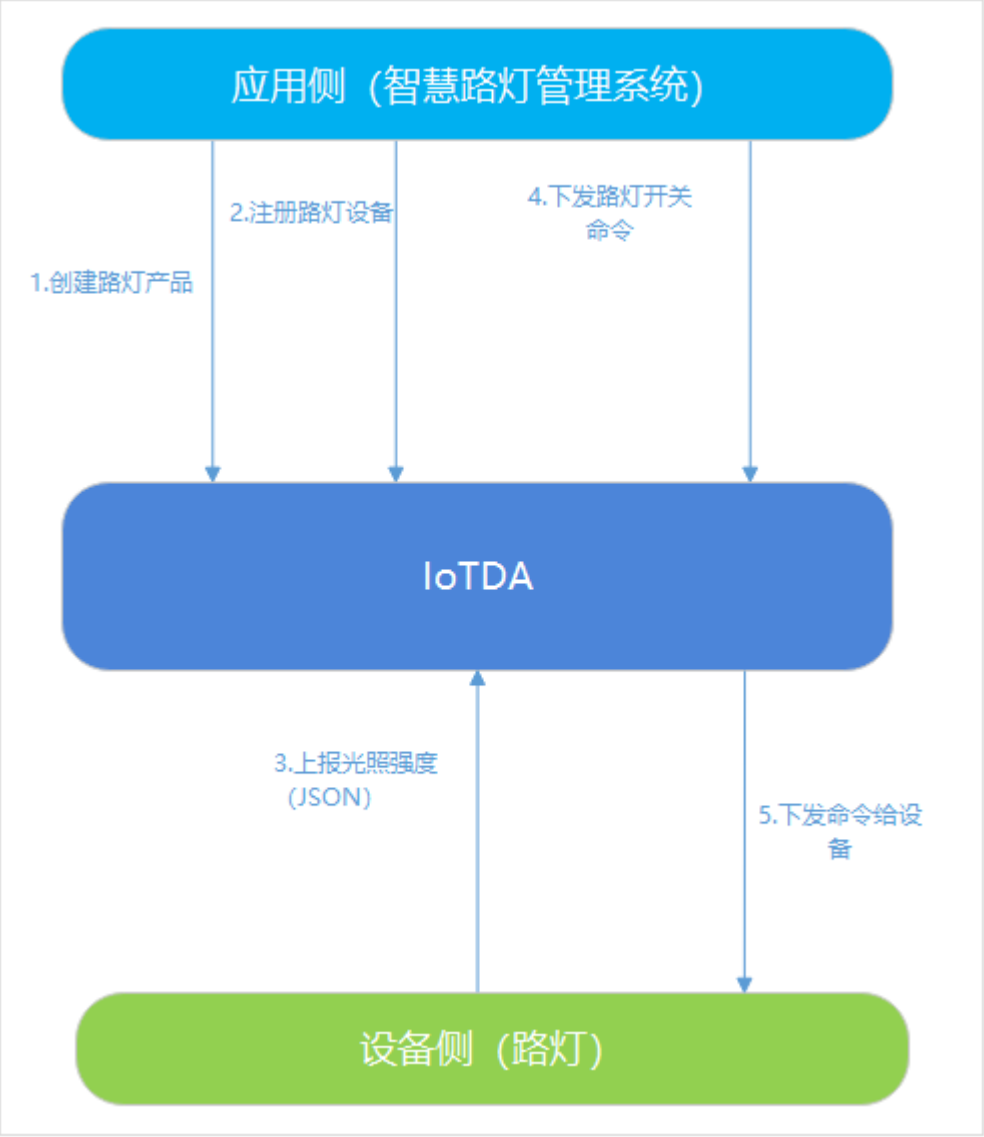
若您想要进一步体验设备接入服务，可参考**虚拟智慧路灯与平台通信**进行自定义产品，用虚拟设备和虚拟应用来体验平台的基本功能及开发流程。

1.3 注册“智慧路灯”模拟设备

场景说明

本文以“智慧路灯”为示例，通过MQTT.fx设备模拟器模拟智慧路灯，给您介绍基于IoTDA平台实现一款智慧路灯联接到平台、上报光照强度数据、以及平台下发开灯命令给智慧路灯三个场景。

图 1-11 模拟智慧路灯与平台通信流程图



前提条件

- 已注册华为云官方账号。未注册可单击**注册页面**完成注册。
- 已开通设备接入服务。未开通则访问**设备接入服务**，单击“控制台”后开通该服务。

业务流程

基于MQTT.fx体验平台功能是指使用MQTT.fx模拟器工具，进行数据上报、命令下发等业务的体验。您可点此下载**MQTT.fx**（默认是64位操作系统，如果是32位操作系统，单击此处下载**MQTT.fx**），安装MQTT.fx工具。整体的业务流程如下：

1. 创建产品。在控制台上创建一个MQTT协议的智慧路灯产品。通过定义产品模型，构建一款路灯设备，支持上报光照强度、下发路灯开关状态命令。
2. 注册设备。在控制台上注册一个MQTT协议的智慧路灯设备。
3. 设备建链。使用MQTT.fx模拟智慧路灯，完成连接鉴权，激活在物联网平台上注册的设备。
4. 数据上报。使用MQTT.fx模拟智慧路灯向物联网平台上报路灯光照强度数据。
5. 命令下发。在管理控制台下发路灯开关命令，远程控制MQTT.fx模拟智慧路灯。

创建产品

步骤1 登录**管理控制台**，选择您的实例，单击实例卡片进入。单击左侧导航栏“产品”，单击页面左侧的“创建产品”。

图 1-12 产品-创建产品

步骤2 创建一个协议类型为MQTT协议、设备类型为StreetLamp的产品，参考页面提示填写参数后，单击“确定”。

图 1-13 创建产品-MQTT

----结束

开发产品模型

步骤1 找到新增的产品，单击产品进入产品界面。

步骤2 在产品详情“基本信息”页面，单击“自定义模型”，配置产品的服务。

图 1-14 自定义模型-MQTT

步骤3 新增服务类型“BasicData”。

1. 在“添加服务”页面，根据页面提示填写“服务ID”、“服务类型”和“服务描述”，单击“确定”。

   图 1-15 添加服务-BasicData

2. 在“BasicData”服务列表右侧区域，单击“新增属性”，填写相关信息后，单击“确定”。

图 1-16 新增属性-luminance

步骤4 新增服务类型“LightControl”。

1. 在“基本信息”下单击“添加服务”，根据页面提示填写后，单击“确定”。
   - “服务ID”： LightControl
   - “服务类型”： 建议和服务ID保持一致
   - “服务描述”： 路灯开关控制
2. 在“LightControl”的下拉菜单下单击“添加命令”，输入命令名称“Switch”。

图 1-17 新增命令-Switch

3. 在“新增命令”页面，单击“新增输入参数”，填写相关信息后，单击“确定”。

图 1-18 新增命令参数-value

----结束

注册设备

步骤1 在设备接入控制台页面，选择您的实例，选择左侧导航栏“设备 > 所有设备”，单击“注册设备”。

图 1-19 所有设备-注册设备

步骤2 根据页面提示信息填写参数，然后单击“确定”。

参数名称	说明
所属资源空间	确保和所属产品归属在同一个资源空间。
所属产品	选择对应产品。

参数名称	说明
设备标识码	即nodeID，设备唯一物理标识。可自定义，由英文字母和数字组成。
设备名称	即device_name，可自定义。
设备认证类型	选择“密钥”。
密钥	此处如不填写，物联网平台会自动生成。

图 1-20 单设备注册-MQTT

步骤3 成功注册设备后，平台会自动生成设备ID和密钥，请妥善保管好设备ID（deviceId）和密钥（deviceSecret），用于设备接入。

图 1-21 设备-注册设备成功

----结束

1.4 MQTT.fx 模拟智慧路灯与平台通信

通过 MQTT.fx 模拟智慧路灯连接平台

使用MQTT.fx工具激活在物联网平台上注册的设备。

步骤1 下载**MQTT.fx**（默认是64位操作系统，如果是32位操作系统，单击此处下载**MQTT.fx**），安装MQTT.fx工具。

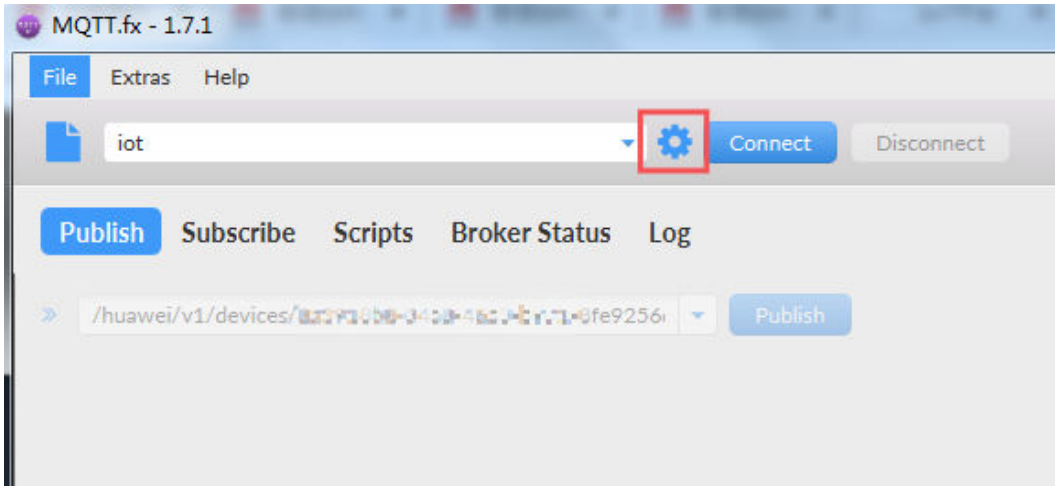
步骤2 进入设备详情页面，找到“MQTT连接参数”，单击“查看”，查看其中的clientId、username、password和hostname。

图 1-22 设备-设备详情

图 1-23 设备-设备详情-MQTT 连接参数

步骤3 打开MQTT.fx软件，单击设置图标。

图 1-24 MQTT.fx 设置



步骤4 单击“User Credentials”，参考下表配置鉴权参数。

图 1-25 配置鉴权参数

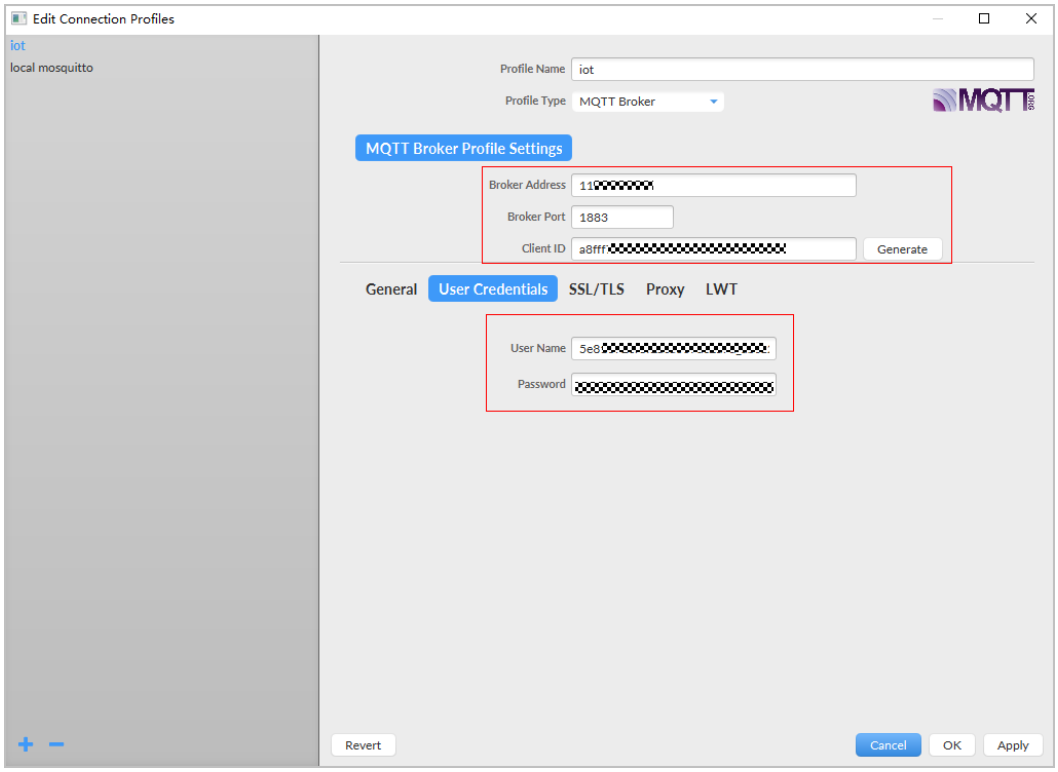
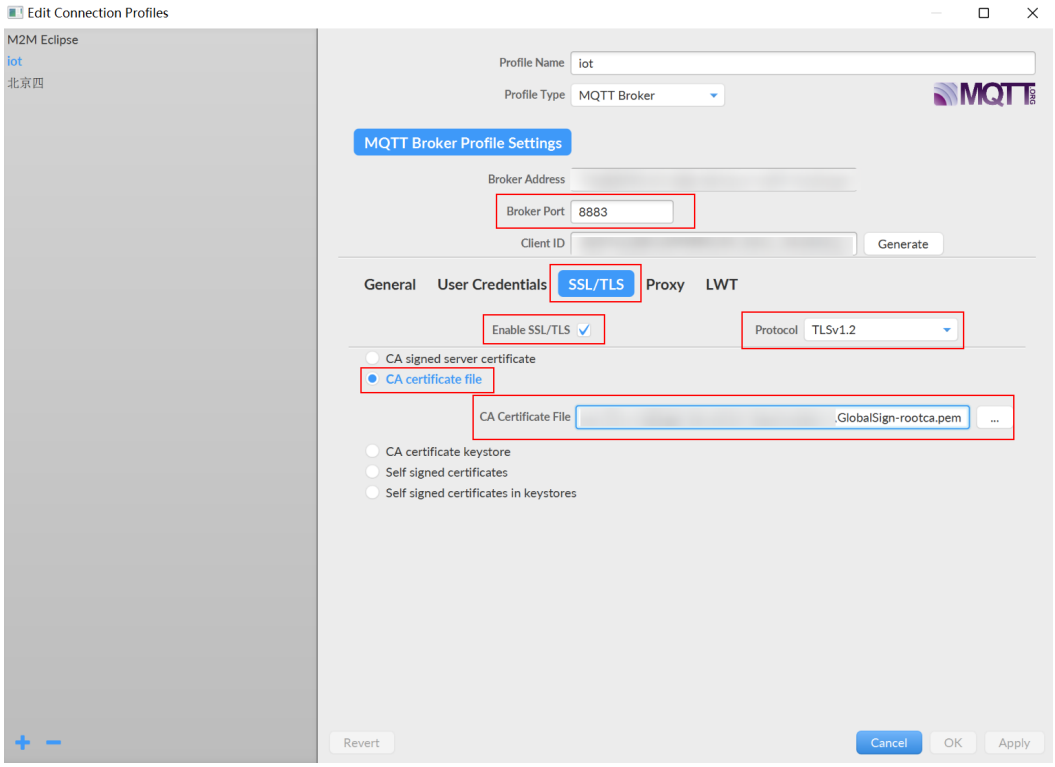


表 1-1 参数说明

参数名称	说明
Broker Address	即hostname，请参考2中获取。此接入地址为域名信息。不能通过域名接入的设备，通过在cmd命令框中执行“ping 域名”获取IP地址，用IP地址接入平台。由于IP地址不固定，您需要将IP地址做成可配置项。
Broker Port	8883，本次快速入门实践采用8883安全连接port。
Cliend ID	设备cliendID，请参考2中获取。
User Name	即设备ID，请参考2中获取。
Password	加密后的设备密钥，请参考2中获取。

步骤5 单击“SSL/TLS”，然后单击“Enable SSL/TLS”，“Protocol”推荐选择“TLSv1.2”。选择“CA certificate file”，前往**证书资源**下载您对应Region和实例版本的证书文件，将证书文件的完整本地路径填入栏目中。最后单击“Apply”，再单击“Cancel”退出配置界面。

图 1-26 配置 SSL/TLS 参数



步骤6 单击“Connect”，看到MQTT.fx界面右上角圆圈转为绿色，即说明设备模拟器鉴权连接成功。设备连接成功后，在物联网平台可以看到设备处于在线状态。

图 1-27 设备模拟器连接成功

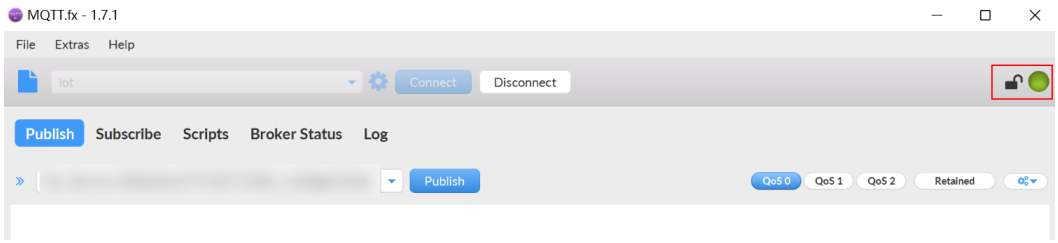


图 1-28 设备在线

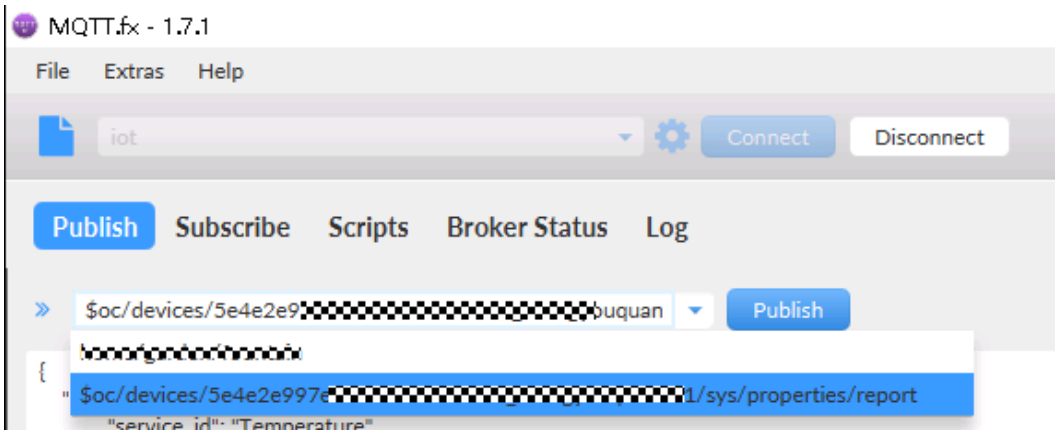
----结束

光照强度数据上报

使用MQTT.fx工具向物联网平台上报光照强度数据。设备若通过MQTT通道上报数据，需要发给指定的topic，上报消息的topic格式为：$oc/devices/**{device_id}**/sys/properties/report，其中对于一机一密设备，使用deviceId接入时需填写为设备注册成功后返回的deviceId值。

步骤1 填写接口地址，此处以$oc/devices/**{device_id}**/sys/properties/report为例。

图 1-29 填写接口地址



步骤2 在工具中间的空白处填写上报的数据后，单击“Publish”。

表 1-2 设备服务数据列表

字段名	必选/可选	类型	参数描述
services	必选	List<ServiceProperty>	设备服务数据列表（具体结构参考下表ServiceProperty定义表）

表 1-3 ServiceProperty 定义表

字段名	必选/可选	类型	参数描述
service_id	必选	String	设备服务的ID。
properties	必选	Object	设备服务的属性列表，具体字段在设备关联的产品模型中定义。
eventTime	可选	String	设备采集数据UTC时间（格式：yyyyMMddTHHmmssZ），如：20161219T114920Z。 设备上报数据不带该参数或参数格式错误时，则数据上报时间以平台时间为准。

请求示例如下：

```
{
   "services": [{
          "service_id": "BasicData",
          "properties": {
             "luminance": 30
      }
    }
```

```
  ]
}
```

步骤3 可以在设备详情页中查看设备是否成功上报数据。如下图，显示光照强度luminance已更新为30。

图 1-30 查看上报数据-MQTT

----结束

远程下发开灯命令

在管理控制台下发开灯命令，远程控制智慧路灯。

步骤1 选择“设备 > 所有设备”，找到新建的设备，单击“详情”进入设备详情页面。

步骤2 单击“云端下发”页签，单击“命令下发”，选择命令为LightControl: Switch，value为ON，下发开灯命令。

图 1-31 命令下发-同步命令下发

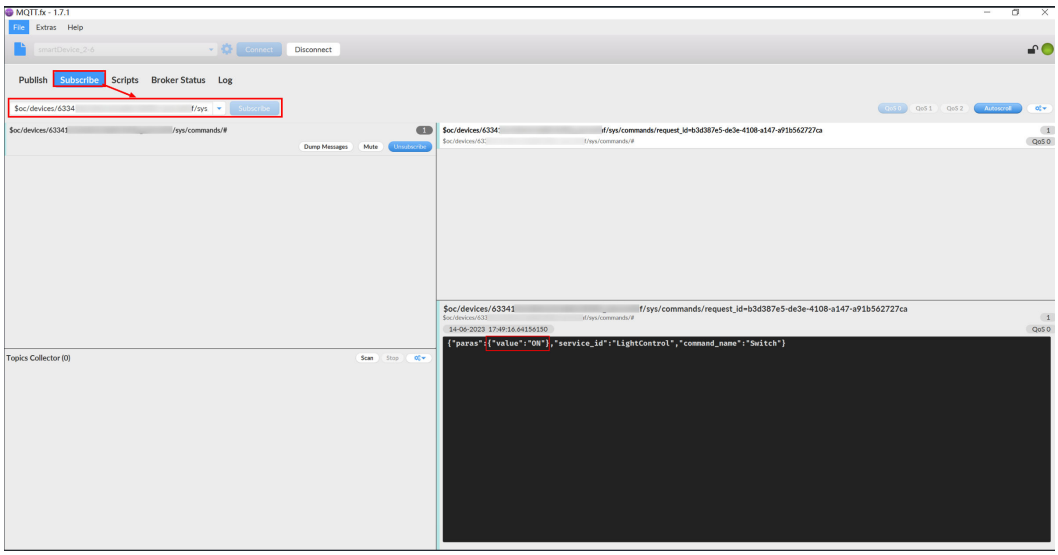
图 1-32 命令下发-LightControl

说明

MQTT协议设备仅支持同步命令下发，NB-IoT设备仅支持异步命令下发。

步骤3 MQTT.fx模拟器中选择“Subscribe”，输入命令下发Topic，订阅后则可以查看到下发的命令参数。命令下发Topic格式为：$oc/devices/**{device_id}**/sys/commands/#。如下图所示，MQTT.fx模拟器成功收到command_name为Switch的命令，value为ON。

图 1-33 查看下发的命令参数



说明

如果Console界面提示命令请求超时，是因为下发的同步命令需要设备侧及时回报响应消息，而MQTT.fx并无自动上报命令响应消息的功能。命令响应内容请参考**平台命令下发**。

----结束

1.5 智慧路灯设备 SDK 与平台通信（Java）

概述

本文基于Java代码演示设备通过MQTTS/MQTT协议接入华为云物联网平台，通过**平台接口**实现南向“数据上报”、“命令下发”的功能，通过应用侧的示例代码接收北向服务端订阅的消息示例。以智慧路灯为例，设备将光照强度等信息上报到IoT平台，应用服务器再接收从平台推送来的设备数据。

前提条件

- 确保开发环境为JDK 1.8及以上版本。
- 已安装IntelliJ IDEA开发工具。如未安装请访问**IntelliJ IDEA官网**下载并安装。

上传产品模型

产品模型是用来描述设备能力的文件，通过JSON的格式定义了设备的基本属性、上报数据和下发命令的消息格式。定义产品模型，即在物联网平台构建一款设备的抽象模型，使平台理解该款设备的功能。

操作步骤:

步骤1 访问**设备接入服务**，单击“控制台”进入设备接入控制台。

步骤2 选择左侧导航栏的“产品”，单击左侧“创建产品”。

图 1-34 产品-创建产品

步骤3 在弹出的窗口中，根据实际情况自定义填写。

图 1-35 创建产品-MQTT

步骤4 下载**模型文件**，该模型文件已开发完毕。详细开发过程指导请参考**在线开发产品模型**。

步骤5 创建成功后，单击刚创建的产品，然后单击上传模型文件（无需解压，并且压缩包的名称不能有括号），上传刚下载的模型文件

图 1-36 上传产品模型-MQTT

----结束

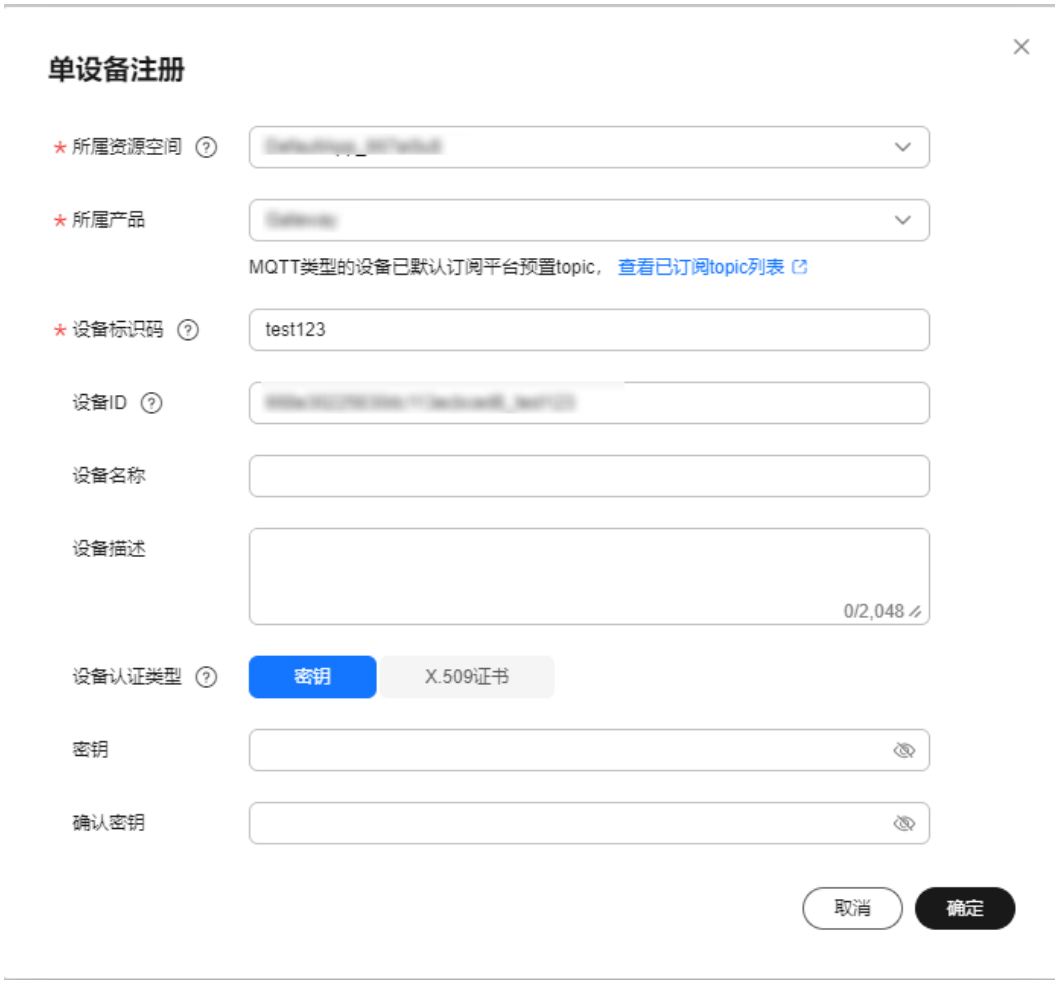
创建设备

步骤1 选择设备接入服务左侧导航栏的“设备 > 所有设备”，单击“注册设备”。

图 1-37 所有设备-注册设备

步骤2 在弹出的窗口中，可以参考图中的内容填写（所属产品需要选择上述步骤创建的产品，密钥不填写，则由平台自动生成，这里是由平台自动生成）然后单击“确定”。

图 1-38 注册设备-test123



步骤3 设备创建成功后，需要保存设备ID和密钥（后续设备连接的时候需要用到）。

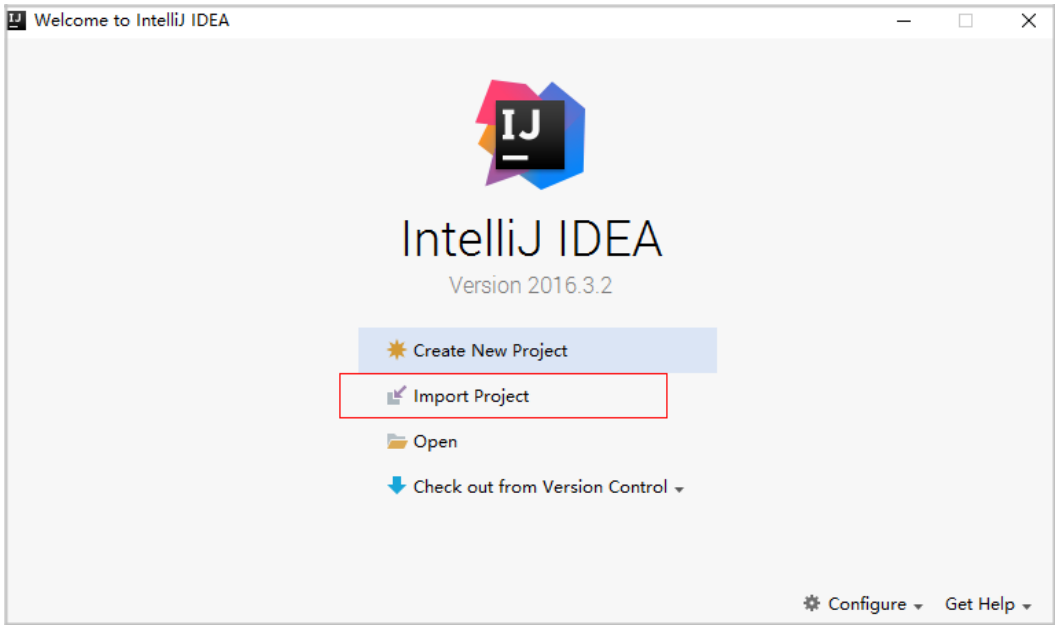
图 1-39 设备-注册设备成功

----结束

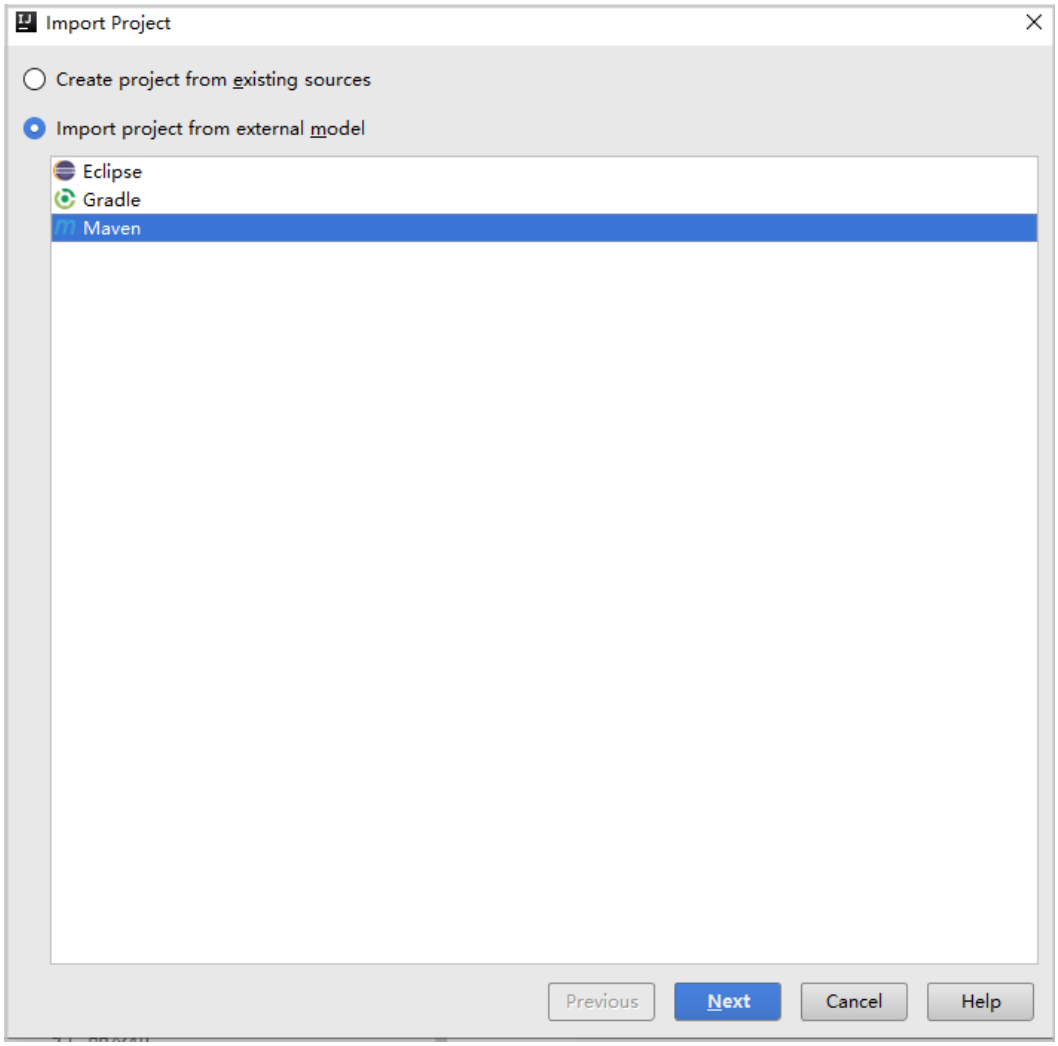
导入代码样例

步骤1 下载**JAVA样例**。

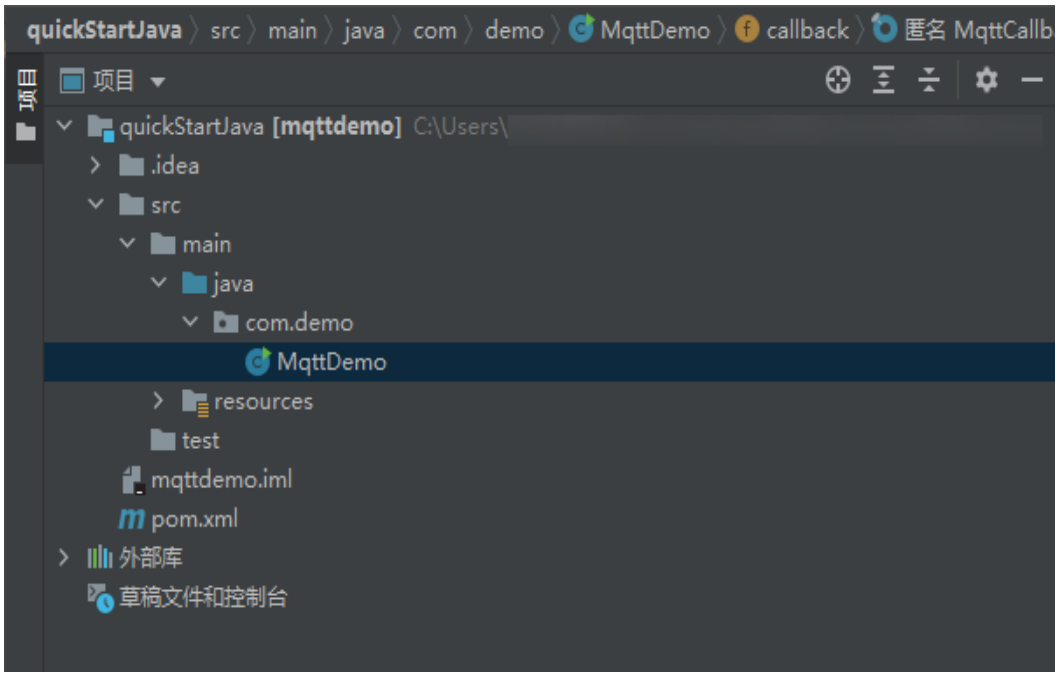
步骤2 打开IDEA开发者工具，单击“ Import Project”。



步骤3 选择步骤1中下载的样例，然后根据界面提示，单击“next”。



步骤4 完成代码导入。



----结束

建立连接

设备或网关在接入物联网平台时首先需要和平台建立连接，从而将设备或网关与平台进行关联。开发者通过传入设备信息，将设备或网关连接到物联网平台。

1. 在建立连接之前，先修改以下参数：

```
//IoT平台mqtt对接地址
static String serverIp = "iot-mqtts.cn-north-4.myhuaweicloud.com";
//注册设备时获得的deviceId,密钥（要替换为自己注册的设备ID与密钥）
static String deviceId = "yourDeviceID"; //device_id, 在创建设备时获得
static String secret = "yourSecret";     //secret，在创建设备时获得
```

   - serverIp为物联网平台设备接入MQTT协议的地址，详细获取方式请参考**资源获取**。
   - device_id和secret为设备ID和密钥，在成功**创建设备**后获取。
2. 完成上述信息的修改后，运行程序，在平台可以看到设备显示在线。

图 1-40 设备列表-设备在线

属性上报

属性上报是指设备主动向平台上报自己的属性（该示例代码已实现自动定时上报功能，可参考下一节在iot平台查看设备上报的数据内容）。

```
//上报json数据，注意serviceId要与产品模型中的定义对应
String jsonMsg = "{\"services\":[{\"service_id\":\"BasicData\",\"properties\":{\"luminance\":32},\"eventTime
\":null}]}";
```

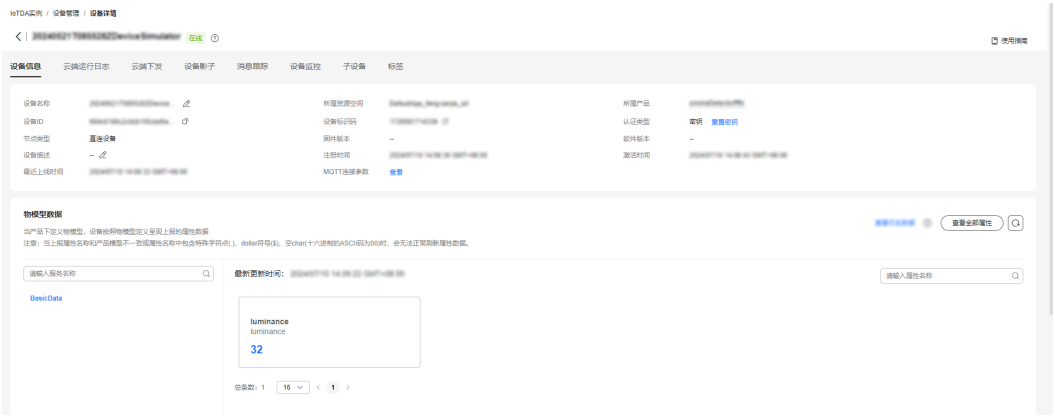
- 消息体jsonMsg组装格式为JSON，其中service_id要与产品模型中的定义对应，properties是设备的属性；
- luminance表示路灯亮度；
- event_time为可选项，为设备采集数据UTC时间，不填写默认使用系统时间。

设备或网关成功连接到物联网平台后，即可调用MqttAsyncClient的publish(String topic,MqttMessage message)方法向平台上报设备属性值。

查看上报数据

运行main方法成功启动后，即可在设备详情页面查看上报的设备属性数据。详细接口信息请参考**设备属性上报**。

图 1-41 查看上报数据-luminance



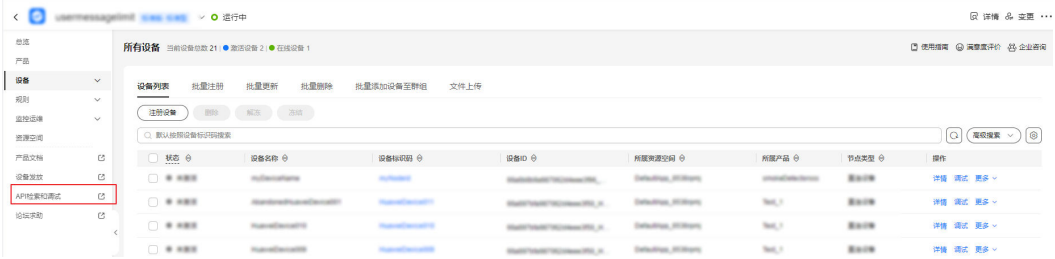
说明

如果在“设备详情”页面没有最新上报数据，请修改产品模型中服务和属性的内容，确保设备上报的服务/属性和产品模型中的服务/属性一致（如果不一致，则历史数据中看不到设备上报的数据），或者进入“产品 > 基本信息”页面，删除所有服务。

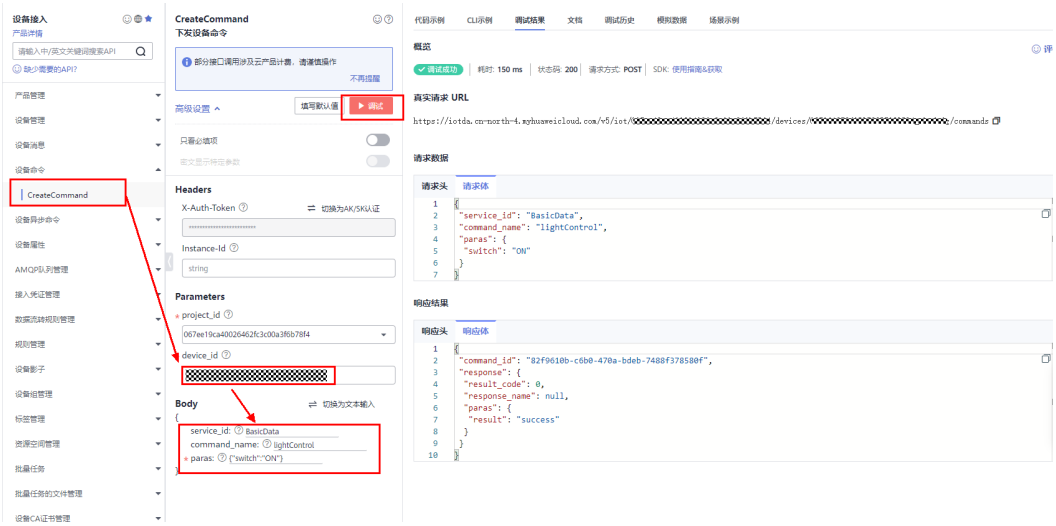
命令下发

步骤1 在控制台界面单击左侧菜单栏的“API检索和调试”。

图 1-42 导航-API 检索和调试

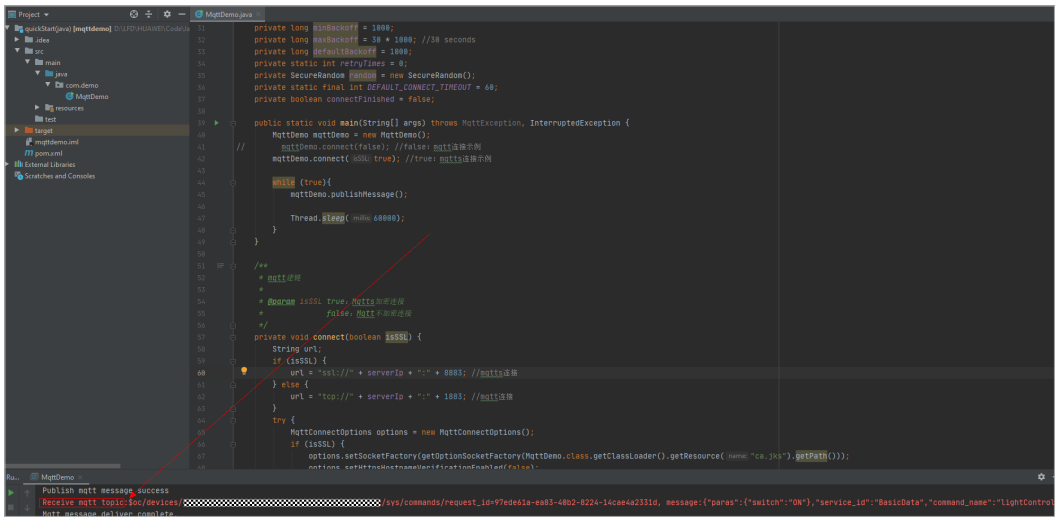


步骤2 找到“设备命令”一栏，下发的参数请参考图片内容（跟产品模型中保持一致），然后单击“调试”按钮即可发送命令。



- service_id表示服务ID，例如：BasicData。
- command_name表示命令名称，例如：lightControl。
- paras表示下发参数，例如：{"switch":"ON"}。

设备侧可查看已收到命令（示例代码已实现接收命令topic的订阅）。



----结束

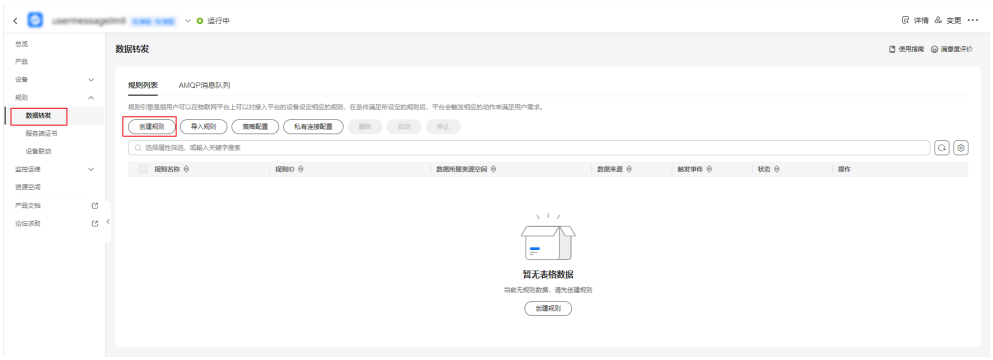
通过云端获取设备上报的数据

本文以AMQP为例，获取设备上报到云端的数据。

步骤1 单击**这里**获取Java AMQP接入示例。

步骤2 登录**管理控制台**，选择“规则 > 数据转发”，创建数据转发规则。

图 1-43 数据转发-创建规则



步骤3 设置转发数据, 填入参数，创建规则。

图 1-44 数据转发-新建属性上报流转规则

参数名	参数说明
规则名称	创建的规则名称，可自定义。
规则描述	对该规则的描述。
数据来源	选择“设备属性”
触发事件	设备属性上报
资源空间	选择所有资源空间

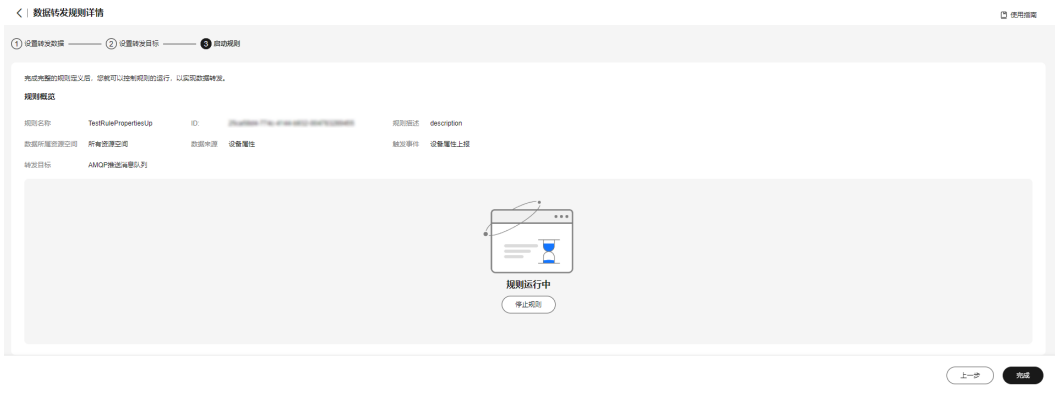
步骤4 设置转发目标（注意：需要单击“预置服务接入凭证”按钮，获取下载文件）。

图 1-45 新建转发目标-转发至 AMQP 推送消息队列

参数名	参数说明
转发目标	AMQP推送消息队列
接入凭证	单击“预置服务接入凭证”，保存下载的文件（包含access_key和access_code
消息队列	DefaultQueue

步骤5 单击启动规则。

图 1-46 启动规则-转发至 AMQP



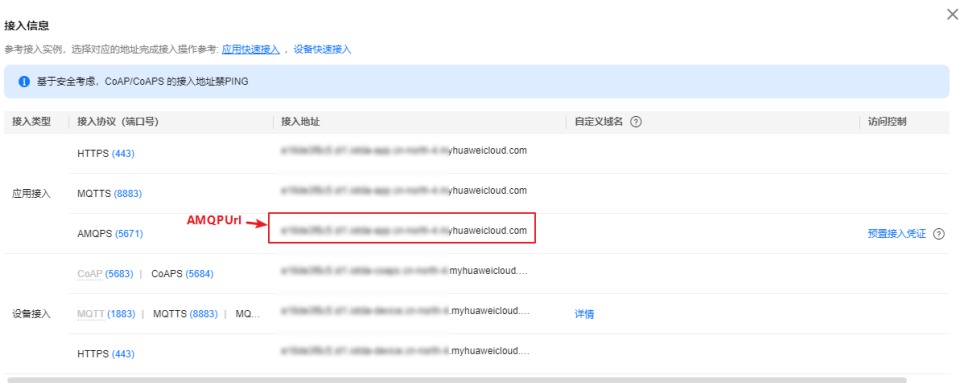
步骤6 修改**步骤1**获取的AMQP代码样例中的参数。

```
        Runtime.getRuntime().availableProcessors(), maximumPoolSize: Runtime.getRuntime().availableProcessors
        keepAliveTime: 60, TimeUnit.SECONDS, new LinkedBlockingQueue<>( capacity: 5000));

public static void main(String[] args) throws Exception {
    //连接凭证接入键值。
    String accessKey = "yourAccessKey";
    long timeStamp = System.currentTimeMillis();
    //UserName组装方法，请参见文档：AMQP客户端接入说明。
    String userName = "accessKey=" + accessKey + "|timestamp=" + timeStamp;
    //连接凭证接入码。
    String password = "yourAccessCode";
    //按照qpid-jms的规范，组装连接URL。
    String baseUrl = "yourAMQPUrl";
    String connectionUrl = "amqps://" + baseUrl + ":5671?amqp.vhost=default&amqp.idleTimeout=8000&amqp.saslMe
    Hashtable<String, String> hashtable = new Hashtable<>();
    hashtable.put("connectionfactory.HwConnectionURL", connectionUrl);
    //队列名，可以使用默认队列DefaultQueue
    String queueName = "yourQueue";
    hashtable.put("queue.HwQueueName", queueName);
    hashtable
            .put(Context.INITIAL_CONTEXT_FACTORY, "org.apache.qpid.jms.jndi.JmsInitialContextFactory");
    Context context = new InitialContext(hashtable);
```

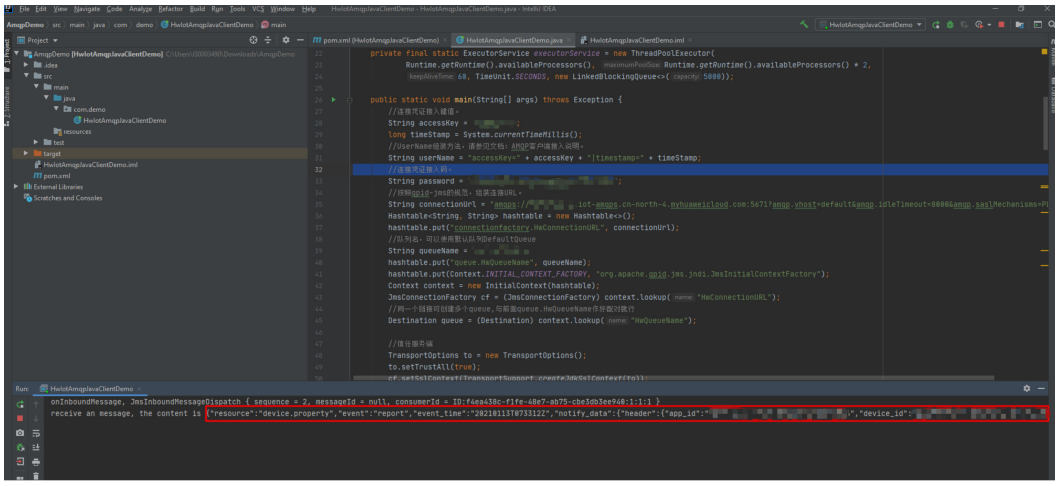
- yourAccessKey：连接凭证接入键值，参考步骤**4**获取。
- yourAccessCode：连接凭证接入码，参考步骤**4**获取
- yourAMQPUrl：amqp域名，请前往**控制台**-总览页面-平台接入地址获取，如下图：

图 1-47 接入信息-AMQP 接入地址



- yourQueue：队列名，使用默认队列DefaultQueue。

步骤7 AMQP数据成功接收。



----结束

更多

更详细开发指导请参考**设备侧SDK**接入或更多其他语言的**demo接入**。

1.6 智慧路灯设备 SDK 与平台通信（C）

概述

本文章节基于C代码演示设备通过MQTTS/MQTT协议接入华为云物联网平台，通过平台接口实现南向“数据上报”、“命令下发”的功能，通过应用侧的示例代码接收北向服务端订阅的消息示例。以智慧路灯为例，设备将光照强度等信息上报到IoT平台，应用服务器再接收从平台推送来的设备数据。

前提条件

Linux操作系统，且已安装好gcc（建议4.8及以上版本）。

上传产品模型

产品模型是用来描述设备能力的文件，通过JSON的格式定义了设备的基本属性、上报数据和下发命令的消息格式。定义产品模型，即在物联网平台构建一款设备的抽象模型，使平台理解该款设备的功能。

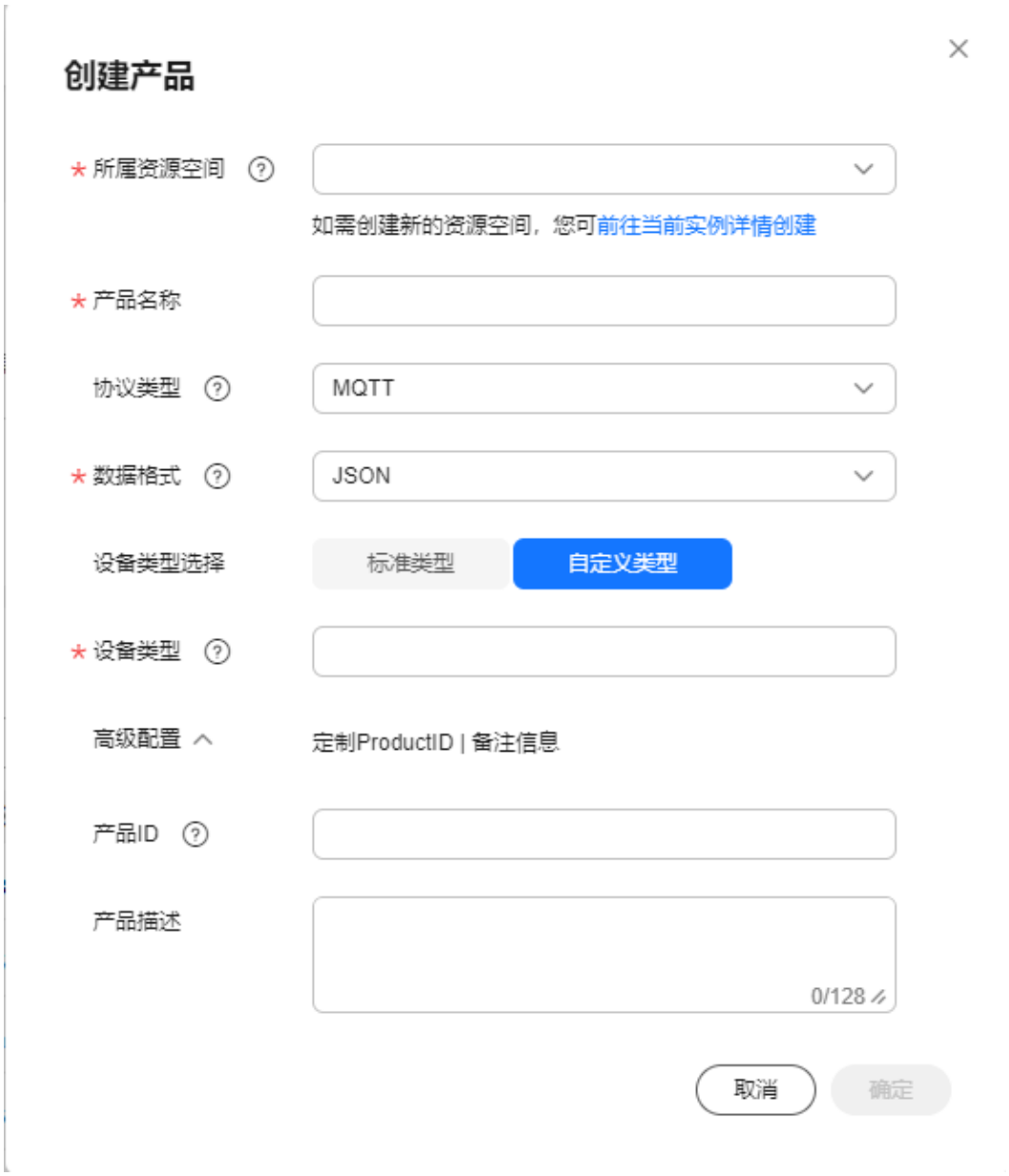
步骤1 访问**设备接入服务**，单击“管理控制台”进入设备接入控制台。选择您的实例，单击实例卡片进入。

步骤2 选择左侧导航栏的“产品”，单击左侧“创建产品”。

图 1-48 产品-创建产品

步骤3 在弹出的窗口中，内容根据实际情况自定义填写填写。

图 1-49 创建产品-MQTT



步骤4 下载**模型文件**，该模型文件已开发完毕（ 。详细开发过程指导请参考**在线开发产品模型**。

步骤5 创建成功后，单击刚创建的产品，然后单击上传模型文件（无需解压，并且压缩包的名称不能有括号），上传刚下载的模型文件。

图 1-50 上传产品模型-MQTT

----结束

创建设备

步骤1 选择设备接入服务左侧导航栏的“所有设备”，单击“注册设备”按钮。

图 1-51 所有设备-注册设备

步骤2 在弹出的窗口中，可以参考图中的内容填写（产品需要选择刚刚创建的产品，密钥不填写，则由平台自动生成，这里是由平台自动生成）然后单击“确定”按钮。

图 1-52 单设备注册-test

步骤3 设备创建成功后，需要保存设备ID和密钥（后续设备连接的时候需要用到）。

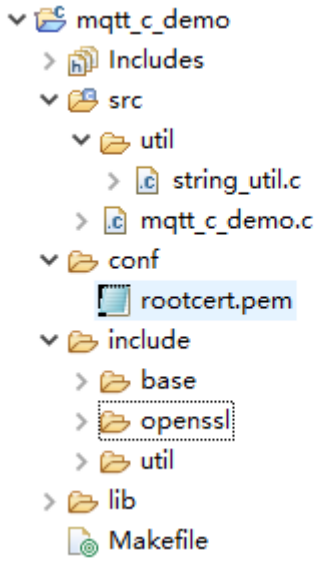
图 1-53 设备-注册设备成功

----结束

导入代码样例

步骤1 下载quickStart(C)样例。

步骤2 将代码复制到linux运行环境中。代码文件层级如下图。



代码目录简述：

- **src：源码目录**

  mqtt_c_demo：demo核心源码；

  util/string_util.c：工具资源文件；

- **conf：证书目录**

  rootcert.pem：设备校验平台身份的证书，用于设备侧接入物联网平台登录鉴权使用；

- **include: 头文件目录**

  base目录：存放依赖的paho头文件

  openssl目录：存放依赖的openssl头文件

  util目录：存放依赖的工具资源头文件

- **lib：依赖库文件**

  libcrypto.so*/libssl.so*: openssl库文件

  libpaho-mqtt3as.so*: paho库文件

- **Makefile：Makefile文件**

----结束

编译库文件

- 编译openssl库

  a. 下载**openssl**，上传到linux编译机任意目录下，并使用如下命令解压：

  ```
  tar -zxvf openssl-1.1.1d.tar.gz
  ```

  b. 配置生成makefile文件。

  执行以下命令进入openssl源码目录

  ```
  cd openssl-1.1.1d
  ```

  创建openssl编译后的目录（本文以/home/test为例）

  ```
  mkdir /home/test
  ```

  创建openssl编译后的目录

  ```
  mkdir /home/test/openssl
  ```

  创建配置文件目录：

  ```
  mkdir /home/test/openssl/ssl
  ```

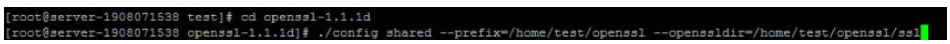
  运行如下配置命令：

  ```
  ./config shared --prefix=/home/test/openssl --openssldir=/home/test/openssl/ssl
  ```

  其中“prefix”是安装目录，“openssldir”是配置文件目录，“shared”作用是生成动态链接库（即.so库）。

  如果编译有问题，配置命令加上no-asm（表示不使用汇编代码）

  ./config no-asm shared --prefix=/home/test/openssl --openssldir=/home/test/openssl/ssl

  c. 编译出库。

  在openssl源码目录下，运行make depend命令。

  ```
  make depend
  ```

  再运行make命令进行编译。

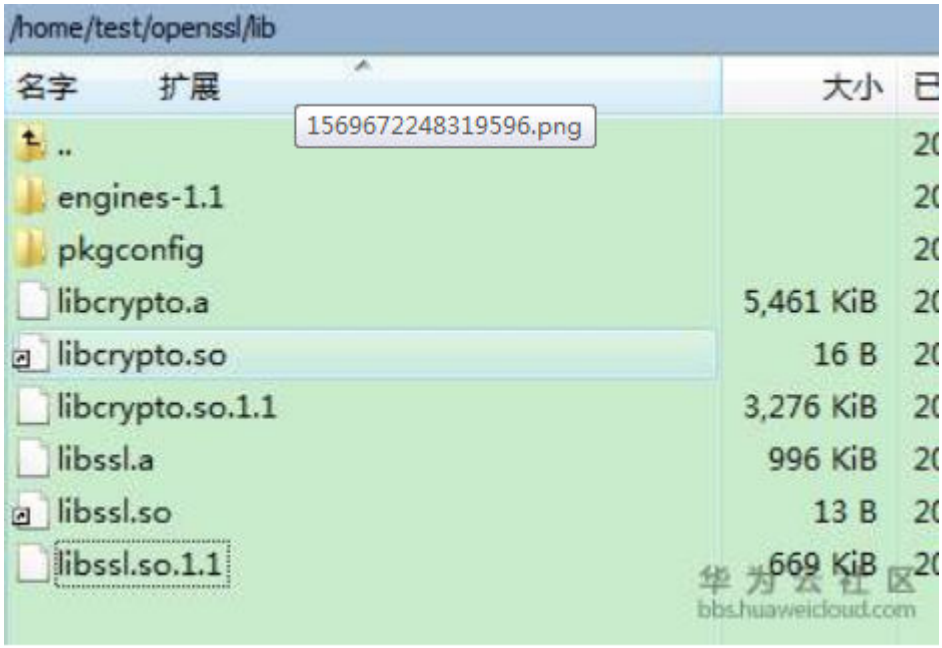
  ```
  make
  ```

  安装openssl。

  ```
  make install
  ```

在配置的openssl安装目录下home/test/openssl找到lib目录，有生成的库文件：

“libcrypto.so.1.1”、“libssl.so.1.1”和软链接“libcrypto.so”、“libssl.so”，请将这些文件复制到quickStart(C)的lib文件夹下（同时将/home/test/openssl/include/openssl里的内容复制到quickStart(C)的include/openssl下）。



注：有的编译工具是32位的，如果在64位的linux机器上使用，这时只要将Makefile中的-m64都删除，再进行编译即可。

- 编译paho库文件
  a. 下载**paho.mqtt.c源码**。
  b. 解压后上传到linux编译机。
  c. 修改makefile
     i. 通过如下命令进行编辑Makefile
     ```
     vim Makefile
     ```
     ii. 显示行数
     ```
     :set nu
     ```
     iii. 在129行之后添加下面两行（自定义的openssl的头文件和库文件）
     ```
     CFLAGS += -I/home/test/openssl/include
     LDFLAGS += -L/home/test/openssl/lib -lrt
     ```

```
127 INSTALL_PROGRAM = $(INSTALL)
128 INSTALL_DATA =  $(INSTALL) -m 644
129 DOXYGEN_COMMAND = doxygen
130 CFLAGS += -I/home/test/openssl/include
131 LDFLAGS += -L/home/test/openssl/lib -lrt
132
133 MAJOR_VERSION = 1
134 MINOR_VERSION = 0
135 VERSION = ${MAJOR_VERSION}.${MINOR_VERSION}
```

     iv. 把195行、197行、199行、201行都改成对应的地址

```
194
195 CCFLAGS_SO += -Wno-deprecated-declarations -DOSX -I /home/test/openssl/include
196 LDFLAGS_C += -Wl,-install_name,lib${MQTTLIB_C}.so.${MAJOR_VERSION}
197 LDFLAGS_CS += -Wl,-install_name,lib${MQTTLIB_CS}.so.${MAJOR_VERSION} -L /home/test/openssl/lib
198 LDFLAGS_A += -Wl,-install_name,lib${MQTTLIB_A}.so.${MAJOR_VERSION}
199 LDFLAGS_AS += -Wl,-install_name,lib${MQTTLIB_AS}.so.${MAJOR_VERSION} -L /home/test/openssl/lib
200 FLAGS_EXE += -DOSX
201 FLAGS_EXES += -L /home/test/openssl/lib
202
203 LDCONFIG = echo
204
205 endif
```

d. 编译

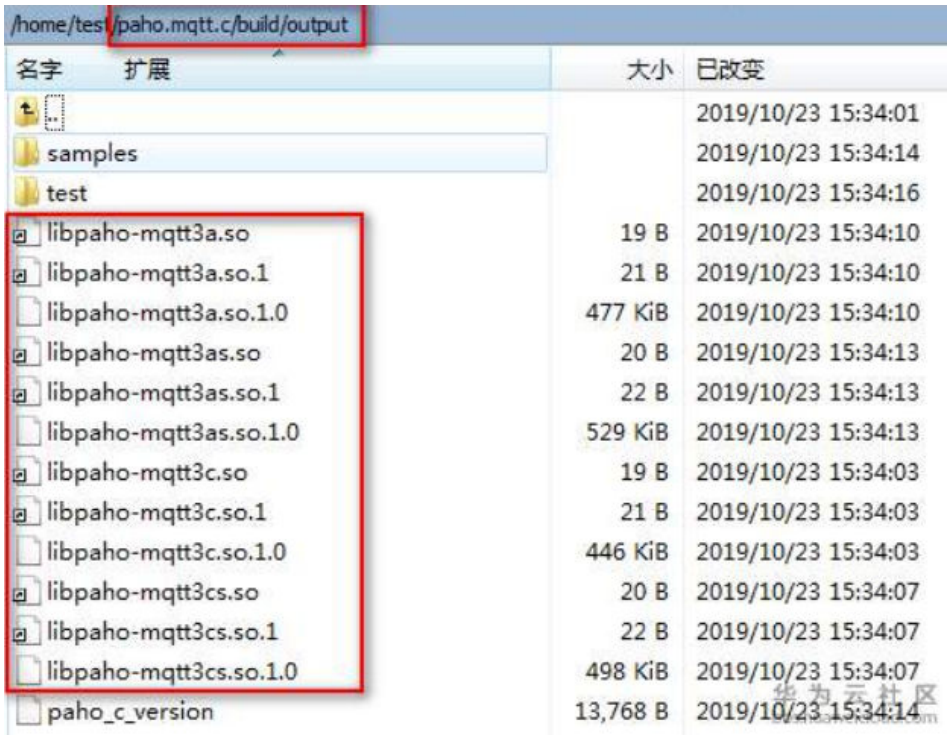
i. 执行清空命令

```
make clean
```

ii. 执行编译命令

```
make
```

e. 编译完成后，可以在build/output目录下看到编译成功的库。



f. 复制paho库文件。

当前SDK仅用到了libpaho-mqtt3as，请将“libpaho-mqtt3as.so”和“libpaho-mqtt3as.so.1”文件复制到quickStart(C)的lib文件夹下。（同时回到paho源代码路径，进入src目录，将MQTTAsync.h、MQTTClient.h、MQTTClientPersistence.h、MQTTProperties.h、MQTTReasonCodes.h、MQTTSubscribeOpts.h复制到quickStart(C)的include/base文件夹下）。

建立连接

设备或网关在接入物联网平台时首先需要和平台建立连接，从而将设备或网关与平台进行关联。开发者通过传入设备信息，将设备或网关连接到物联网平台。

步骤1 设置参数，只需修改username和password。详情请参考**资源获取**。

步骤2 连接。

1. 执行**make**进行编译。如果是32位的操作系统，请删除Makefile中的"-m64"。
2. 执行**export LD_LIBRARY_PATH=./lib/**加载库文件。

3. 运行./MQTT_Demo.o。

步骤3 连接成功后，打印“connect success”，同时在控制台可看到设备已在线。

```
begin to connect the server.
connect success.
```

图 1-54 设备列表-设备在线

----结束

属性上报

属性上报是指设备主动向平台上报自己的属性（该示例代码已实现自动定时上报功能，可参考下一节在iot平台查看设备上报的数据内容），更多信息请参考**设备属性上报**。

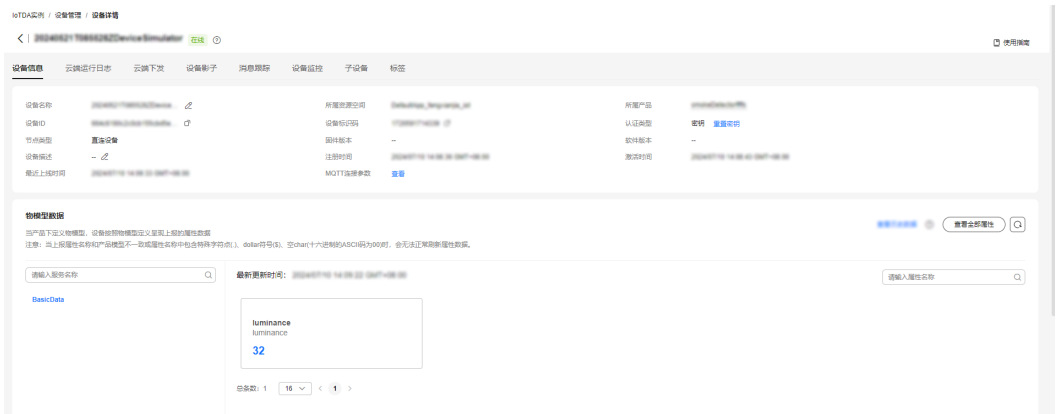
```
//publish data
char *payload = "{\"services\":[{\"service_id\":\"BasicData\",\"properties\":{\"luminance\":32},\"eventTime\":NULL}]}";
```

- 消息体payload组装格式为JSON，其中service_id要与产品模型中的定义对应，properties是设备的属性；
- luminance表示路灯亮度；
- event_time为可选项，为设备采集数据UTC时间，不填写默认使用系统时间。

设备上报属性成功后，demo控制台中会打印“publish success”字样。

同时在设备详情页面查看到上报的属性：

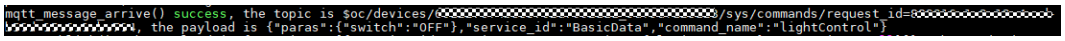
图 1-55 查看上报数据-luminance



接收下发命令

订阅了命令Topic后，可以在控制台下发同步命令。详情请参考**MQTT设备同步命令下发**。

命令下发后，demo中接收到命令：



通过云端获取设备上报的数据

当数据到达平台后，应用服务器可以使用AMQP来接收推送消息。具体可参考**通过云端获取设备上报的数据**。

更多

更详细开发指导请参考**设备侧SDK**接入或更多其他语言的**demo接入**。

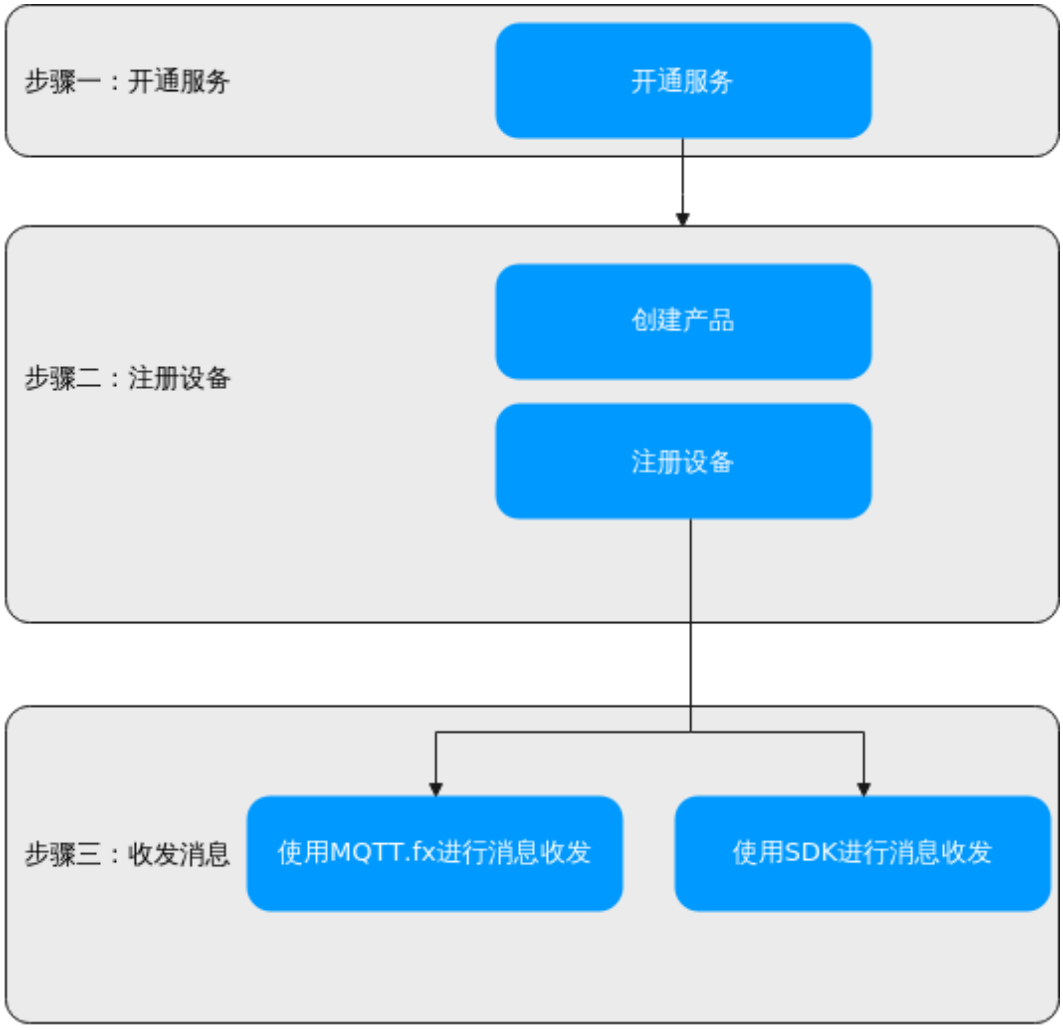
2 设备快速接入—消息收发

2.1 入门指引

IoTDA服务支持设备使用MQTTS/MQTT协议接入平台并进行消息收发，本文分别以MQTT.fx和Java SDK为例介绍设备基于MQTTS/MQTT协议进行消息收发的流程。

操作流程

图 2-1 操作流程



2.2 开通服务

本章节介绍如何在“中国-香港”区域开通一个标准版免费实例单元，以进行IoTDA平台快速入门的体验。

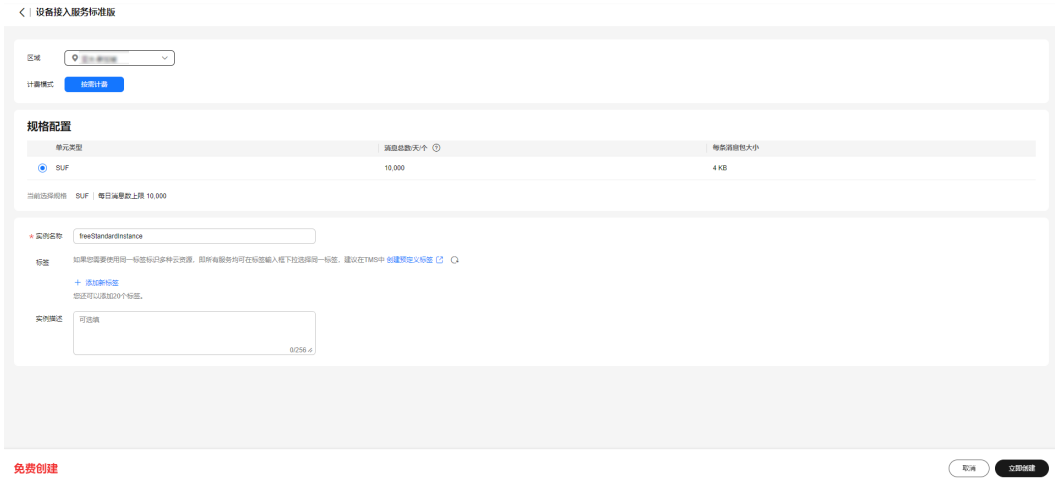
步骤1 访问**设备接入服务**，单击“控制台”进入设备接入控制台。

步骤2 在左侧导航栏，选择“IoTDA实例”，单击“开通免费单元”。

图 2-2 实例-标准版-开通免费实例

步骤3 按下图选择配置信息，均采用默认配置即可。

图 2-3 实例-免费实例配置



步骤4 单击“立即创建”，进入实例页面，刷新页面，等待实例状态变为“运行中”，即表示免费实例成功创建。

图 2-4 实例-免费实例创建完成

----结束

2.3 注册设备

在使用IoTDA进行收发消息前，您需要创建产品和设备。

步骤1 访问**设备接入服务**，单击“控制台”进入设备接入控制台。选择左侧导航栏“IoTDA实例”，单击您需要的实例卡片进入实例。

步骤2 创建产品。单击左侧导航栏“产品”，单击页面左侧的“创建产品”。根据页面提示填写参数，然后单击“确定”，完成产品的创建。

参数名称	配置说明
产品名称	填写为“MyProduct”。
数据格式	选择“JSON”
所属行业	请根据实际情况填写。
设备类型	请根据实际情况填写。

图 2-5 创建产品-MQTT

步骤3 注册设备。在左侧导航栏选择“ 设备 > 所有设备”，单击“注册设备”，按照如下表格填写参数后，单击“确定”，设备注册成功后，请妥善保管好设备ID和密钥，用于设备接入平台认证。

参数名称	配置说明
所属产品	选择设备所属的产品。
设备标识码	填写为“test001”。
设备名称	填写为“test001”。
设备认证类型	选择“密钥”。
密钥	设备密钥，可自定义，不填写平台会自动生成。

图 2-6 设备-注册密钥设备

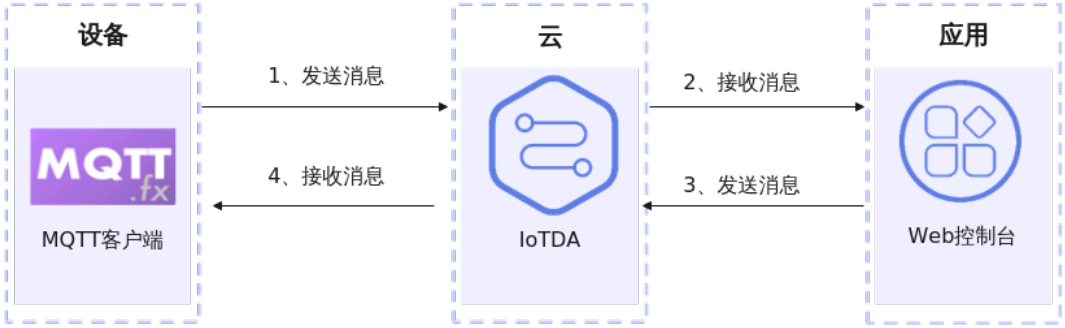
----结束

2.4 使用 MQTT.fx 进行消息收发

概述

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，可用于模拟设备通过MQTTS/MQTT协议连接华为云IoTDA并通过Topic发布和订阅消息。本文以Windows系统为例，介绍如何使用MQTT.fx接入华为云IoTDA并进行消息收发。

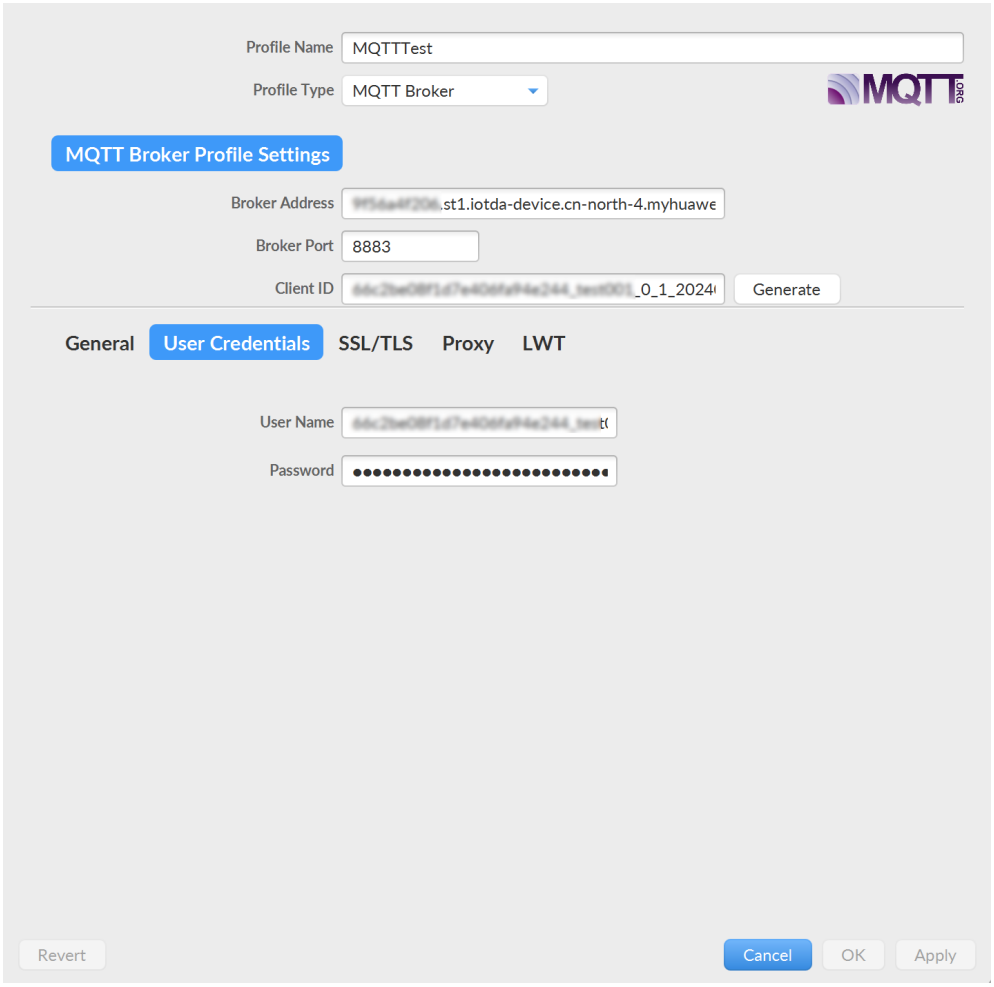
图 2-7 MQTT.fx 与 IoTDA 的消息交互流程



配置 MQTT.fx 接入 IoTDA

1. 下载MQTT.fx（默认是64位操作系统，如果是32位操作系统，单击此处下载MQTT.fx），安装MQTT.fx工具。
2. 打开MQTT.fx客户端，在其顶部菜单栏中选择“Extras > Edit Connection Profiles”。
3. 在Edit Connection Profiles页面中配置相关参数，然后单击“OK”。

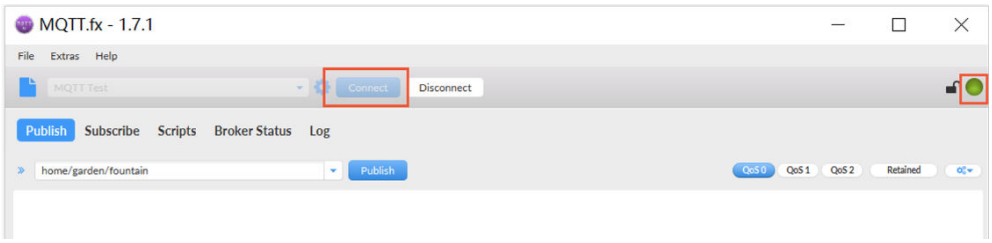
图 2-8 MQTT.fx 连接参数配置



参数名称	参数说明	取值示例
Profile Name	配置文件名称。	MQTT Test
Profile Type	配置的连接类型。	固定选择“MQTT Broker”，表示连接MQTT服务器。
Broker Address	MQTT服务器接入地址。	华为云IoTDA的MQTTS协议接入地址，请参考**此处**，进入您的实例，获取“总览”->“接入信息”->“MQTTS”协议对应的接入地址。
Broker Port	MQTT服务器接入端口。	8883
Client ID	设备接入华为云IoTDA需要完成设备接入认证，认证通过后才能进行消息收发。设备鉴权参数计算方式请参见**设备连接鉴权**。	进入设备详情页面，找到“MQTT连接参数”，单击“查看”，查看其中的clientId、username和password。
User Name		
Password		
SSL/TLS		
Enable SSL/TLS	是否使用SSL或TLS加密协议。	是
Protocol	协议版本。	TLSv1.2
CA certificate file	CA证书文件	从**证书资源**页面获取对应区域的CA证书。

4. 参数配置完成后，单击“Connect”进行连接。右侧绿灯亮起，表示MQTT.fx和华为云IoTDA已成功连接。右侧红灯亮起，表示连接失败，您可以单击“Log”查看日志，根据日志信息修改配置并重新尝试连接。

图 2-9 MQTT.fx 连接



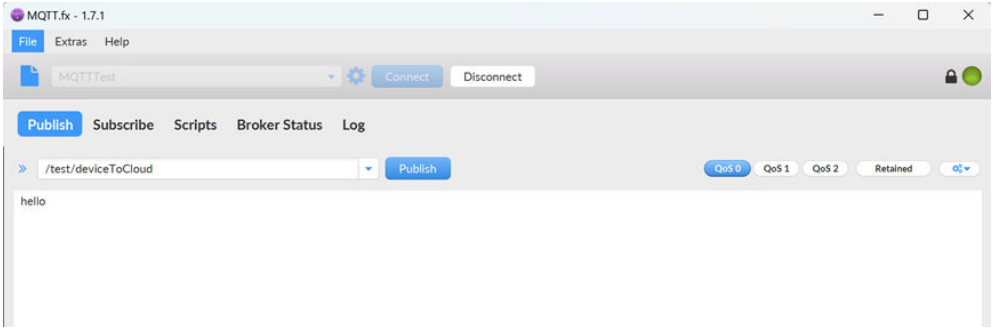
5. 访问**设备接入服务**，单击“控制台”进入设备接入控制台。选择左侧导航栏“IoTDA实例”，单击您需要的实例卡片进入实例。在左侧导航栏选择“设备>所有设备”，查看设备状态，预期设备状态为在线。

图 2-10 设备列表-设备在线

使用 MQTT.fx 发送消息

1. 在MQTT.fx客户端上方单击“Publish”页签。
2. 在“Publish”页签左侧Topic文本框输入Topic的名称，本示例以“/test/deviceToCloud”为例，消息文本框中输入要发送的消息内容，例如："hello"，单击右侧的“Publish”发送消息。

图 2-11 MQTT.fx 发送消息



3. 访问**设备接入服务**，单击“控制台”进入设备接入控制台。选择左侧导航栏“IoTDA实例”，单击您需要的实例卡片进入实例。在左侧导航栏选择“设备>所有设备”，进入设备页面后单击“详情”，在设备详情的消息跟踪页面可以查看MQTT.fx发送的消息。

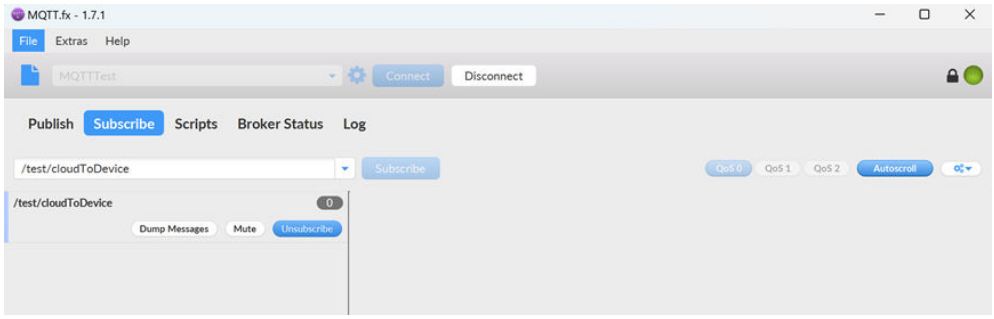
图 2-12 消息跟踪-查看消息跟踪

4. MQTT.fx将消息发送到平台后，可以通过配置**数据转发**，将MQTT.fx上报的消息平滑流转至消息中间件、存储、数据分析或业务应用。

使用 MQTT.fx 接收消息

1. 在MQTT.fx客户端上方单击“Subscribe”页签。
2. 在Subscribe页签中，在左侧Topic文本框输入Topic的名称后，单击文本框右侧的“Subscribe”。本示例以“/test/cloudToDevice”为例。订阅成功后，该Topic会显示在订阅列表中。

图 2-13 MQTT.fx 订阅 Topic

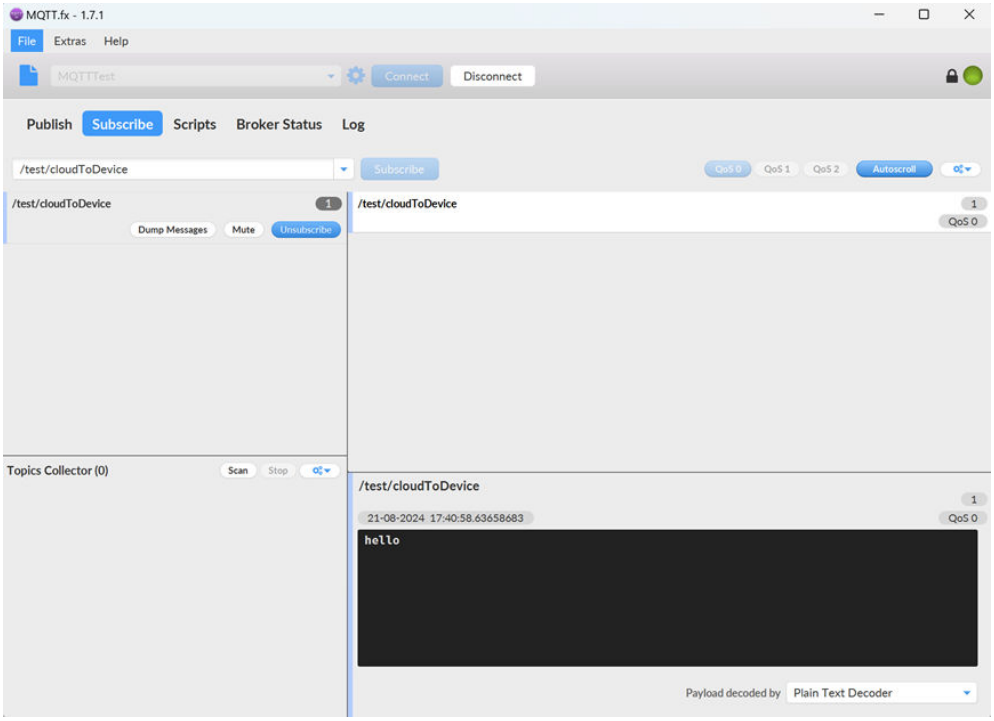


3. 访问**设备接入服务**，单击“控制台”进入设备接入控制台。选择左侧导航栏“IoTDA实例”，单击您需要的实例卡片进入实例。选择左侧导航栏的“设备 > 所有设备”，在设备列表中，单击具体的设备进入到设备的详情页面。
4. 在“云端下发>消息下发”标签页，单击“下发消息”，在弹出的窗口中选择需要下发的消息并设置消息参数。

图 2-14 下发消息-自定义 topic

5. 在MQTT.fx客户端上方单击“Subscribe”页签，可以查看到对应订阅的Topic已接收到发送的消息。

图 2-15 MQTT.fx 查看消息

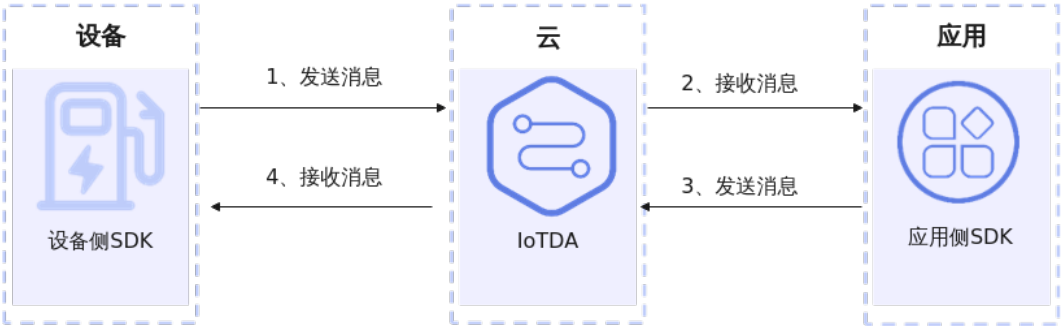


2.5 使用设备 SDK 进行消息收发

概述

华为云IoTDA服务**多种语言SDK**，通过SDK设备可以快速连接华为云IoTDA并进行消息上下行通信，本文基于Java示例代码演示设备通过MQTTS/MQTT协议接入华为云IoTDA服务并通过Topic发布和订阅消息。

图 2-16 SDK 与 IoTDA 的消息交互流程



设备侧 SDK 发送消息

1. 配置设备侧SDK的Maven依赖。

```
<dependency>
    <groupId>com.huaweicloud</groupId>
    <artifactId>iot-device-sdk-java</artifactId>
    <version>[1.2.1,)</version>
</dependency>
```

2. 建立设备与华为云IoTDA的连接。

   a. 参考**样例代码**，在建立连接之前，修改设备连接参数。

```
// 请将 xxx.st1.iotda-device.cn-north-4.myhuaweicloud.com 替换为自己的接入地址。
//域名获取方式：登录华为云IoTDA控制台左侧导航栏“总览”页签，在选择的实例基本信息中，单击“接入信息”。选择“设备接入”8883端口对应的接入地址。
IoTDevice device = new IoTDevice("ssl://xxx.st1.iotda-device.cn-
north-4.myhuaweicloud.com:8883", "your device id",
        "your device secret", tmpCAFile);
device.getClient().setConnectListener(new MessageSample(device));

if (device.init() != 0) {
    return;
}
```

   b. 设备接入平台后，参考如下代码订阅平台下发的消息。

```
    @Override
    public void connectComplete(boolean reconnect, String serverURI) {
        // 订阅下行消息
        device.getClient().subscribeTopic("/test/cloudToDevice", new ActionListener() {
            @Override
            public void onSuccess(Object context) {
                System.out.println("subscribeTopic success");
            }

            @Override
            public void onFailure(Object context, Throwable var2) {
                System.out.println("subscribeTopic failure");
            }
        } , rawMessage -> {
            System.out.println(" on receive message topic : " + rawMessage.getTopic() + " ,
```

```
payload : " + new String(
                rawMessage.getPayload(), StandardCharsets.UTF_8));
        }, 1);
    }
```

c. 完成上述信息的修改后，运行程序，在平台可以看到设备显示在线。

图 2-17 设备列表-设备在线

3. 上报设备消息，参考样例代码，在建立连接后，指定Topic上报消息。

```
device.getClient().publishRawMessage(new RawMessage("/test/deviceToCloud", "hello", 1), new
ActionListener() {
    @Override
    public void onSuccess(Object context) {
        System.out.println("reportDeviceMessage success: ");
    }
    @Override
    public void onFailure(Object context, Throwable var2) {
        System.out.println("reportDeviceMessage fail: " + var2);
    }
});
```

4. 访问**设备接入服务**，单击“控制台”进入设备接入控制台。选择左侧导航栏“IoTDA实例”，单击您需要的实例卡片进入实例。在左侧导航栏选择“设备>所有设备”，单击“详情”，在设备详情的消息跟踪页面可以查看平台是否收到对应消息。

图 2-18 消息跟踪-查看消息跟踪

应用侧 SDK 接收消息

设备通过SDK将消息发送到平台后，可以配置**数据转发**将设备上报的消息平滑流转至消息中间件、存储、数据分析或业务应用。本文以**Java SDK接入示例**接收设备上报的消息并进行业务处理。

应用侧 SDK 下发消息

配置应用侧SDK下发消息。

1. 配置Maven依赖，本示例使用的开发环境为JDK 1.8及以上版本。SDK代码获取：**SDK下载**。

```
<dependency>
    <groupId>com.huaweicloud.sdk</groupId>
    <artifactId>huaweicloud-sdk-core</artifactId>
    <version>[3.0.40-rc, 3.2.0)</version>
</dependency>
```

```
<dependency>
    <groupId>com.huaweicloud.sdk</groupId>
    <artifactId>huaweicloud-sdk-iotda</artifactId>
    <version>[3.0.40-rc, 3.2.0)</version>
</dependency>
```

2. 应用侧向指定设备下发消息样例如下：

```
public class MessageDistributionSolution {
    // REGION_ID：如果是上海一，请填写"cn-east-3"；如果是北京四，请填写"cn-north-4";如果是华南广州，请填写"cn-south-4"
    private static final String REGION_ID = "<YOUR REGION ID>";
    // ENDPOINT：请在控制台的"总览"界面的"平台接入地址"中查看“应用侧”的https接入地址。
    private static final String ENDPOINT = "<YOUR ENDPOINT>";
    // 标准版/企业版：需自行创建Region对象
    public static final Region REGION_CN_NORTH_4 = new Region(REGION_ID, ENDPOINT);
    public static void main(String[] args) {
        String ak = "<YOUR AK>";
        String sk = "<YOUR SK>";
        String projectId = "<YOUR PROJECTID>";
        // 创建认证
        ICredential auth = new BasicCredentials().withDerivedPredicate(AbstractCredentials.DEFAULT_DERIVED_PREDICATE)
            .withAk(ak)
            .withSk(sk)
            .withProjectId(projectId);
        // 创建IoTDAClient实例并初始化
        IoTDAClient client = IoTDAClient.newBuilder().withCredential(auth)
            // 基础版：请选择IoTDARegion中的Region对象
            //.withRegion(IoTDARegion.CN_NORTH_4)
            // 标准版/企业版：需自行创建Region对象
            .withRegion(REGION_CN_NORTH_4).build();
        // 实例化请求对象
        CreateMessageRequest request = new CreateMessageRequest();
        request.withDeviceId("<YOUR DEVICE_ID>");
        DeviceMessageRequest body = new DeviceMessageRequest();
        body.withPayloadFormat("raw");
        body.withTopicFullName("/test/cloudToDevice");
        body.withMessage("hello");
        request.withBody(body);
        try {
            CreateMessageResponse response = client.createMessage(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

3. 查看设备端日志，可以看到应用给设备下发的消息如下：

```
on receive message topic : /test/cloudToDevice , payload : hello
```

3 应用快速接入

为了降低应用侧的开发难度、提升应用侧开发效率，物联网平台向应用侧开放了丰富的API。本文档以本地调试（ Postman ）为例，模拟应用服务器以HTTPS协议为例接入物联网平台。

本地调试

本地调试是指以Postman方式调用应用侧接口为例介绍如何使用设备接入服务。

具体步骤如下：

步骤1：开通设备接入服务。访问**设备接入服务**，单击“管理控制台”后开通服务。

步骤2：创建产品。创建一个MQTT协议的产品。

步骤3：配置环境。下载并安装Postman，Postman建议使用7.17.0版本。

步骤4：调用服务。使用Postman调用API接口，查看返回结果或状态码与错误码。

- **步骤1：开通设备接入服务**

  目前设备接入服务仅在亚太-曼谷、亚太-新加坡、中国-香港、非洲-约翰内斯堡环境上线。

- **步骤2：创建产品**

  调用接口前，需要先在物联网平台创建一款产品。

  a. 访问**设备接入服务**，单击“管理控制台”进入设备接入控制台。选择您的实例，单击实例卡片进入。

  b. 选择左侧导航栏的“产品”，单击“创建产品”。

  c. 按照页面提示填写参数，创建一个MQTT协议的产品，然后单击“确定”。

基本信息	
所属资源空间	平台自动将新创建的产品归属在默认资源空间下。如需归属在其他资源空间下，下拉选择所属的资源空间。如无对应的资源空间，请先创建**资源空间**。
产品名称	自定义。长度不超过64，只允许中文、字母、数字、以及_?'#().,&%@!-等字符的组合。
协议类型	建议选择MQTT。

数据格式	选择JSON。
所属行业	请根据实际情况选择。
所属子行业	请根据实际情况选择。
设备类型	请根据实际情况选择。
高级配置	
产品ID	定制ProductID，用于唯一标识一个产品。如果携带此参数，平台将产品ID设置为该参数值；如果不携带此参数，产品ID在物联网平台创建产品后由平台分配获得。
产品描述	产品描述。请根据实际情况填写。

- **步骤3：配置环境**

  下载并安装Postman，详细操作请参考**安装并配置Postman**。

- **步骤4：调用服务**

  配置完Postman后，模拟应用服务器以HTTPS协议接入物联网平台，调测以下API接口：

  - **“获取IAM用户Token”接口**
  - **“查询IAM用户可以访问的项目列表”接口**
  - **“创建产品”接口**
  - **“查询产品”接口**
  - **“创建设备”接口**
  - **“查询设备”接口**

进阶体验

按照本页面的指导，使用Postman模拟应用服务器接入物联网平台后，您应该已经基本了解应用服务器如何通过调用物联网平台开放的接口与平台交互。

若您想要进一步体验设备接入服务，可参考**开发指南**开发真实应用和真实设备，并接入物联网平台，体验更多功能。