设备接入

快速入门

文档版本 1.0

发布日期 2022-08-30

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（标准版实例）

1.1 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”。

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。

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 1-1 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 1-2 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 1-3 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 1-4 向导式极速体验-配置模拟设备

图 1-5 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 1-6 向导式极速体验-设备属性上报频率

----结束

进阶体验

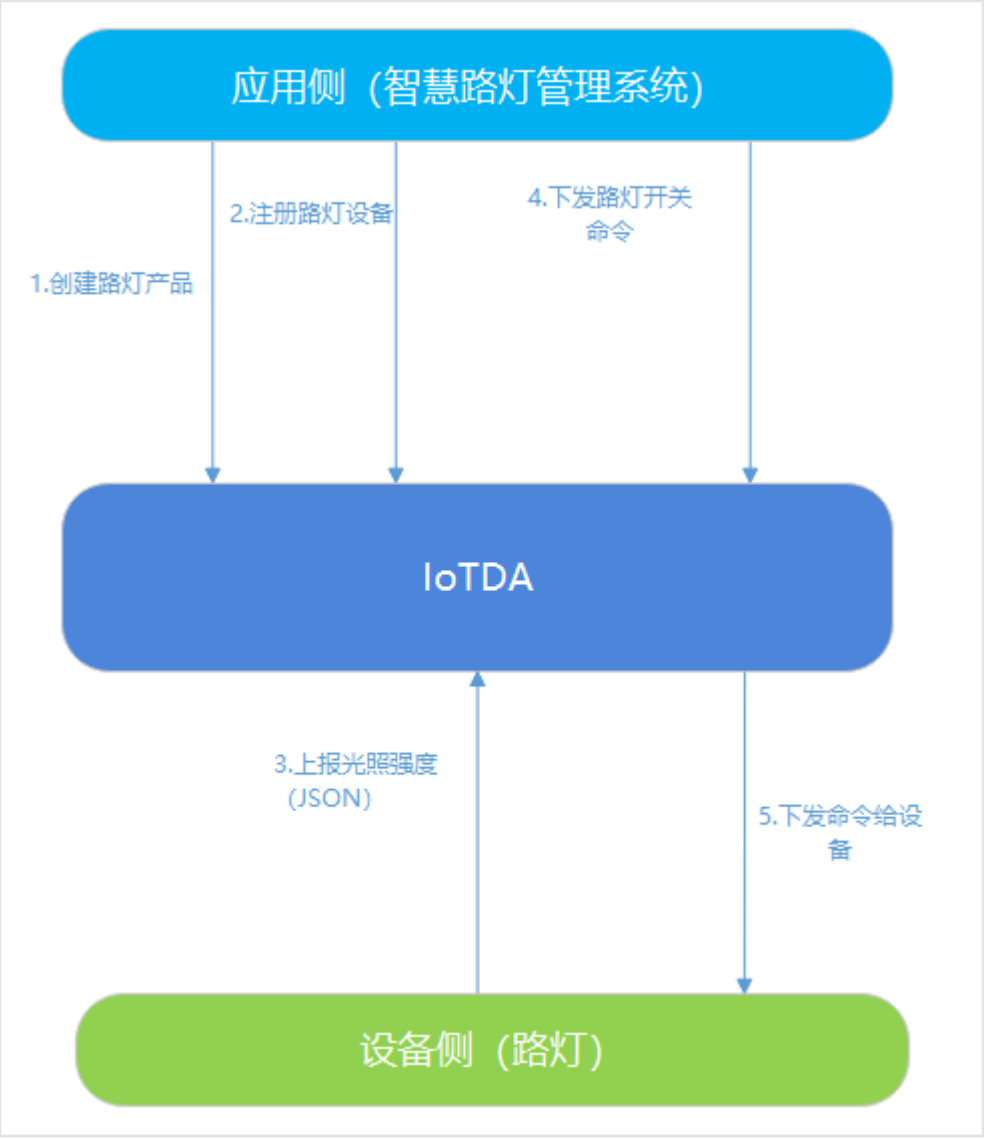
按照本页面的指导，您应该已经基本了解设备连接平台的步骤及相关概念。

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。

1.2 虚拟智慧路灯与平台通信

场景说明

本文中，您不用下载任何软件，用物联网平台的虚拟设备即可完成设备与云端的双向交互，并与平台之间进行交互，体验平台的基本功能及开发流程。本文以智慧路灯为例，给您介绍基于IoT平台操作实现一款智慧路灯联接到平台、上报光照强度数据、以及平台下发命令给智慧路灯三个场景。



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。

业务流程

虚拟设备实现端云双向通信是指基于物联网平台的在线调试功能，使用虚拟设备体验设备上报数据，平台下发远程控制命令等业务。

具体步骤如下：

步骤1：创建产品。创建一个MQTT协议的产品。

步骤2：开发产品模型。定义设备上报到平台的光照强度值（luminance）和远程控制路灯开关状态的命令（switch）。

步骤3：注册虚拟设备。创建一个虚拟设备，体验数据上报业务。

步骤4：数据上报。在设备模拟区域执行数据上报操作。

步骤5：命令下发。在应用模拟器区域执行命令下发操作。

创建产品

产品是设备的合集，您可以将相同能力或特征的设备归属在同一个产品下。

步骤1 登录管理控制台，单击左侧导航栏“产品”，单击页面左侧的“创建产品”。

步骤2 根据页面提示填写参数，然后单击“确定”。

图 1-7 创建产品-MQTT

基本信息	
所属资源空间	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，如SmartStreetlight。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	在本次操作中，选择MQTT协议。
数据格式	选择JSON。
所属行业	无。
设备类型	SmartStreetLight

----结束

开发产品模型

步骤1 找到**创建产品**章节新增的产品，单击产品进入产品界面。

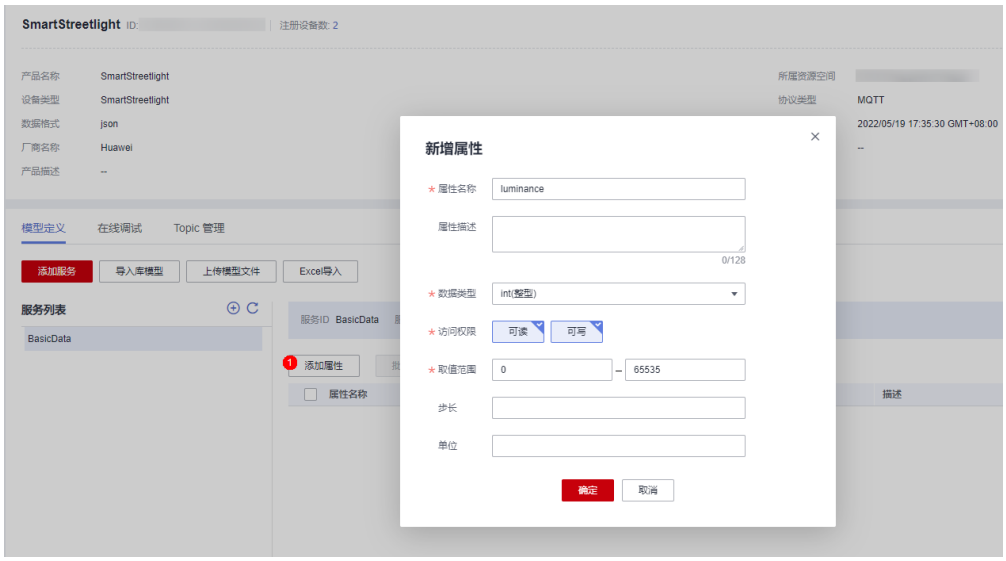
步骤2 在产品详情“模型定义”页面，单击“自定义模型”，配置产品的服务。

步骤3 添加服务类型“BasicData”。

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

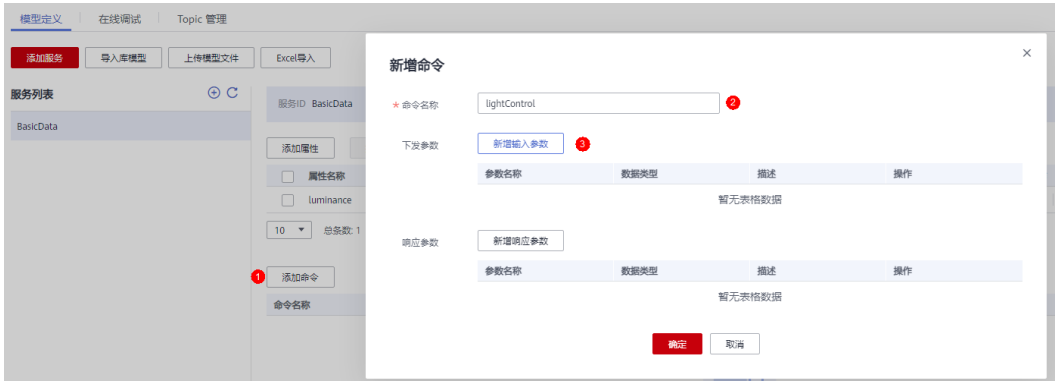
- “服务ID”：BasicData
- “服务类型”：建议和服务ID保持一致
- “服务描述”：上报路灯数据

2. 在“**BasicData**”的下拉菜单下单击“添加属性”，填写相关信息后，单击“确定”。



- “属性名称”：luminance
- “数据类型”：int（整型）
- “访问权限”：可读、可写
- “取值范围”：0~65535
- “步长”：0
- “单位”：不填写

步骤4 单击添加命令，输入命令名称“lightControl”。



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 1-8 新增输入参数"“switch”"。

- “参数名称”： switch
- “参数描述”： 下发开关命令
- “数据类型”： string(字符串)
- “长度”： 15
- “枚举值”： ON,OFF

----结束

注册虚拟设备

步骤1 选择**创建产品**章节新建的产品，单击产品名称进入产品界面。

步骤2 单击“在线调试”页签，单击“新增测试设备”，在弹出的页面中选择“虚拟设备”，并单击确认。

虚拟设备名称包含 “Simulator”字样，选择新注册的虚拟设备，单击右侧的“调试”，进入调试界面，同时设备变更为上线状态。

----结束

数据上报

在“设备模拟器”区域，输入上报的光照强度值（luminance），单击“发送”，在“应用模拟器”区域查看上报的结果。

命令下发

在“应用模拟器”区域选择命令参数后，下发远程控制开关灯命令，在“设备模拟器”区域可以查看接收到的命令。

进阶体验

按照本页面的指导，您应该已经基本了解平台的基本功能及开发流程。

若您想要进一步体验设备接入服务，可参考**智慧路灯设备SDK与平台通信（Java）**和**智慧路灯设备SDK与平台通信（C）**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。

1.3 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

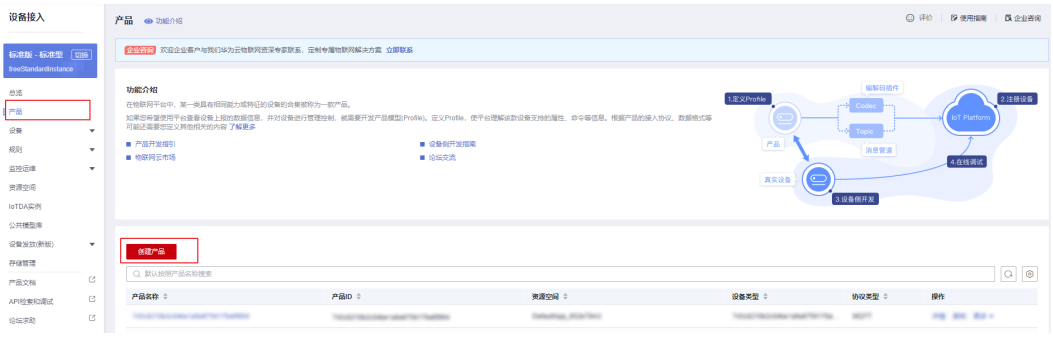
产品模型是用来描述设备能力的文件，通过JSON的格式定义了设备的基本属性、上报数据和下发命令的消息格式。定义产品模型，即在物联网平台构建一款设备的抽象模型，使平台理解该款设备的功能。

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 1-9 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 1-10 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

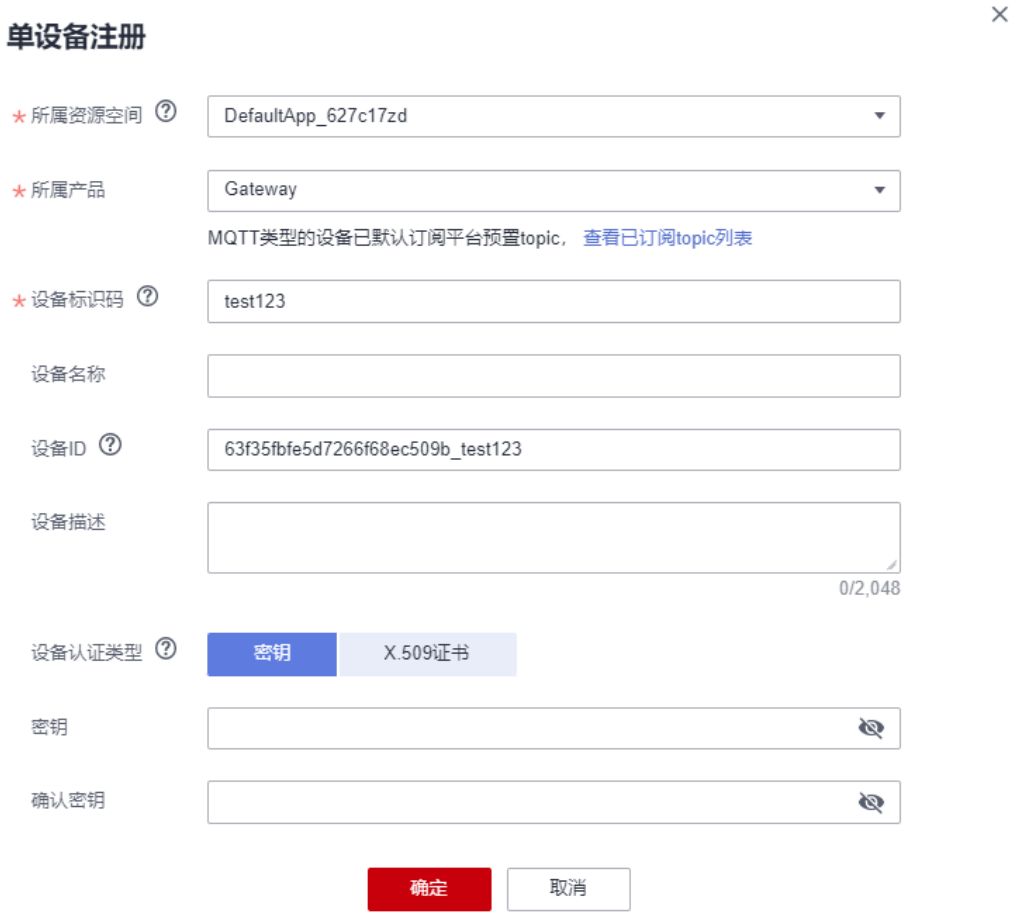
图 1-11 上传产品模型-MQTT

----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”。

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”。



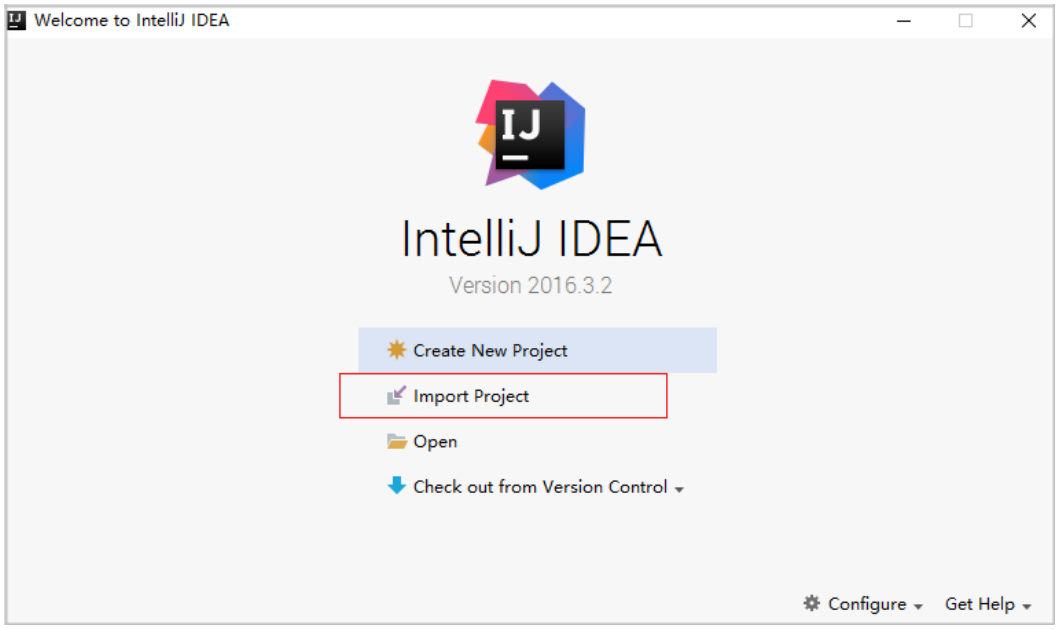
步骤3 设备创建成功后，需要保存设备ID和密钥（后续设备连接的时候需要用到）。

----结束

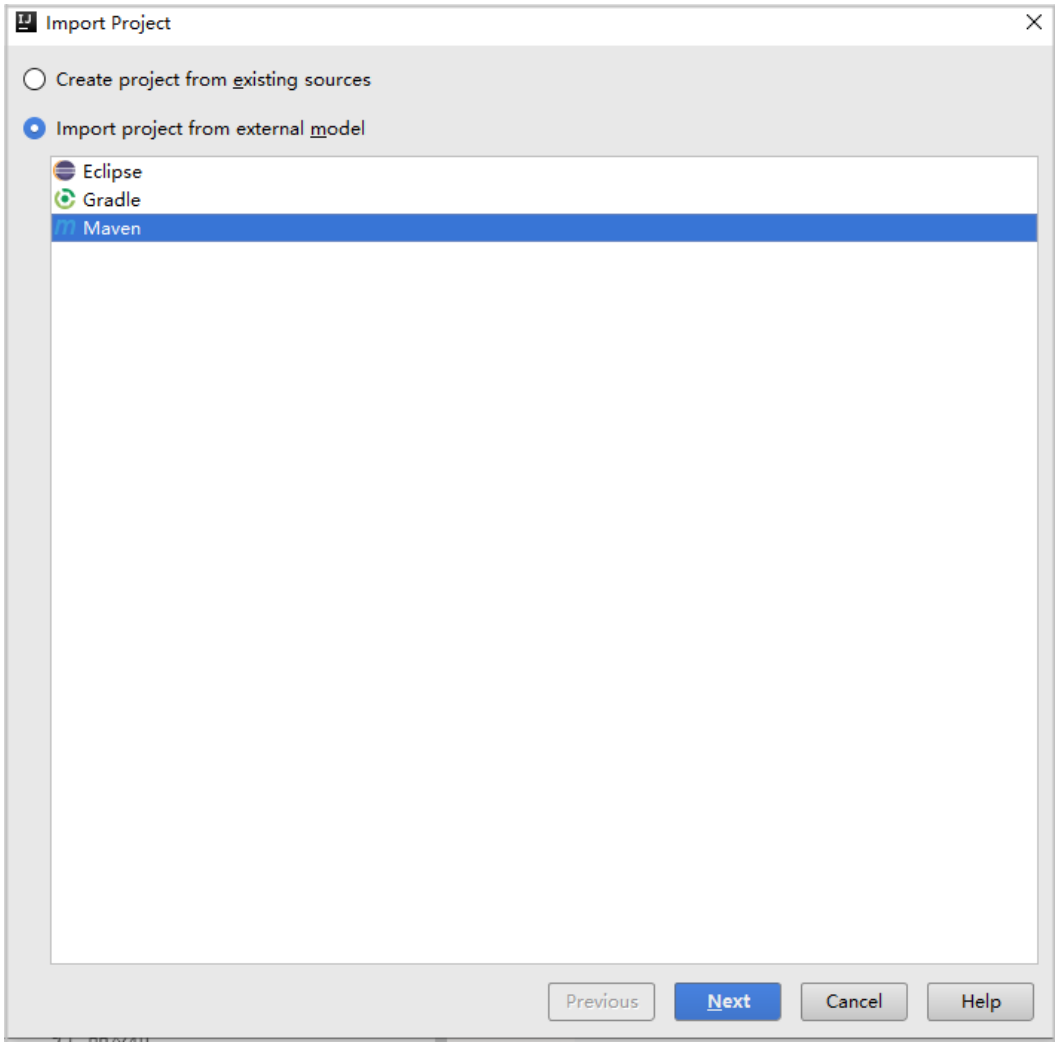
导入代码样例

步骤1 下载**JAVA样例**。

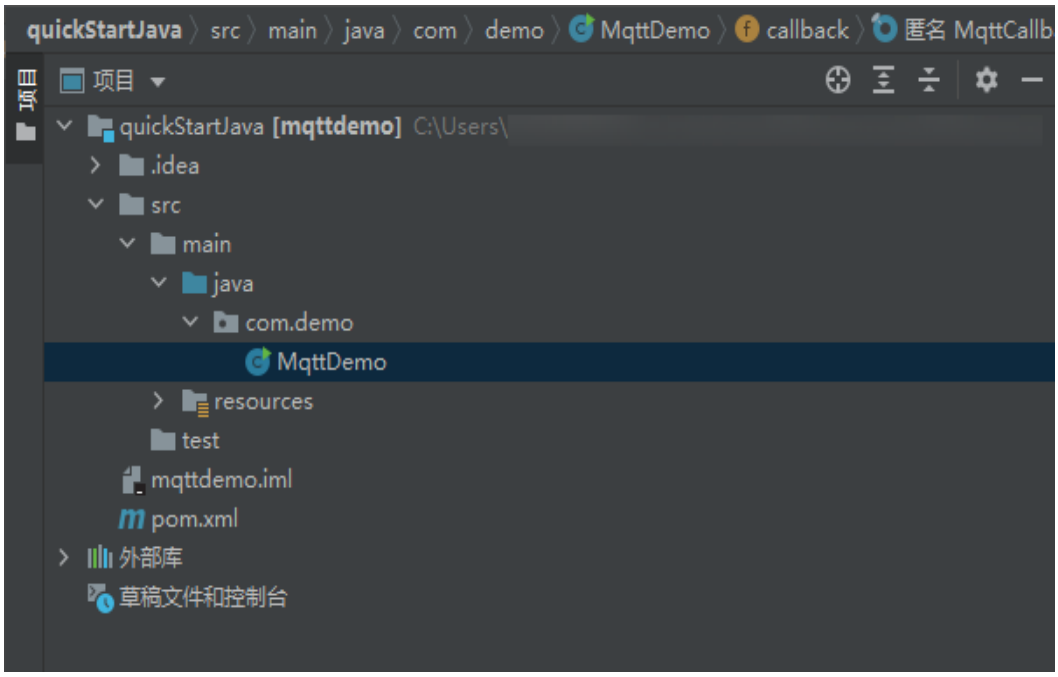
步骤2 打开IDEA开发者工具，单击“ Import Project”。



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。

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

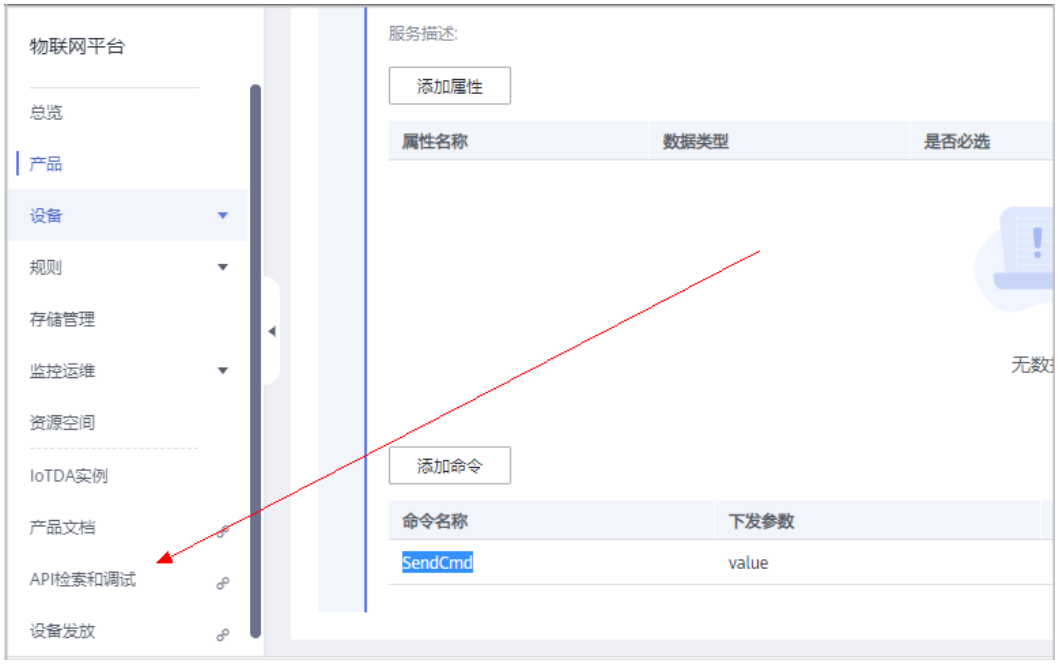
运行main方法成功启动后，即可在设备详情页面查看上报的设备属性数据。详细接口信息请参考**设备属性上报**。

说明

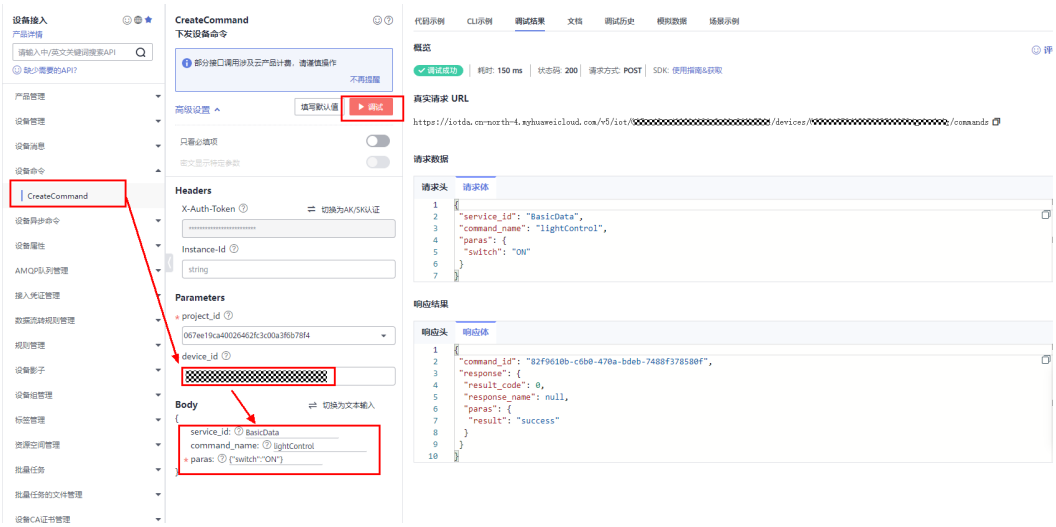
如果在“设备详情”页面没有最新上报数据，请修改产品模型中服务和属性的内容，确保设备上报的服务/属性和产品模型中的服务/属性一致（如果不一致，则历史数据中看不到设备上报的数据），或者进入“产品 > 模型定义”页面，删除所有服务。

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”。

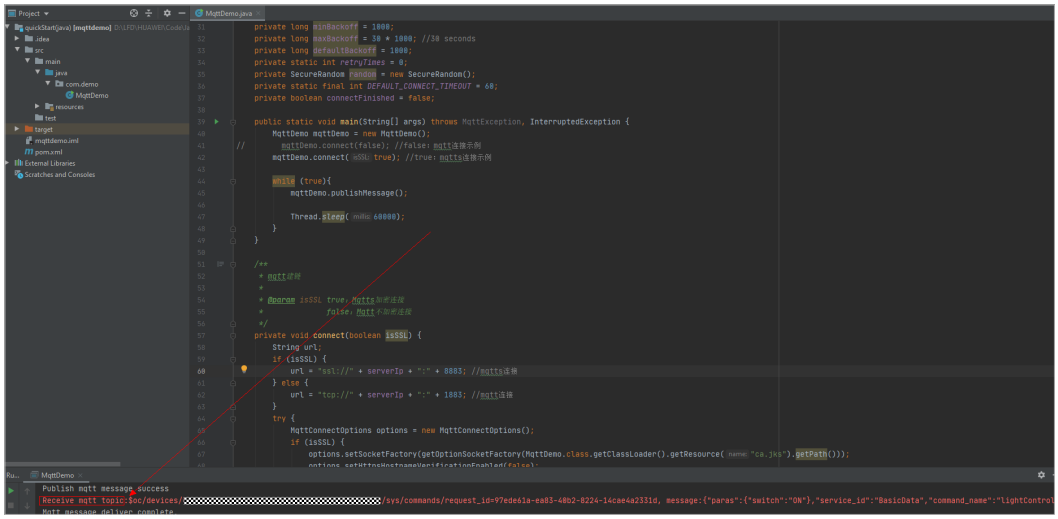


步骤2 找到“设备命令”一栏，下发的参数请参考图片内容（跟产品模型中保持一致），然后单击“调试”按钮即可发送命令。



- 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

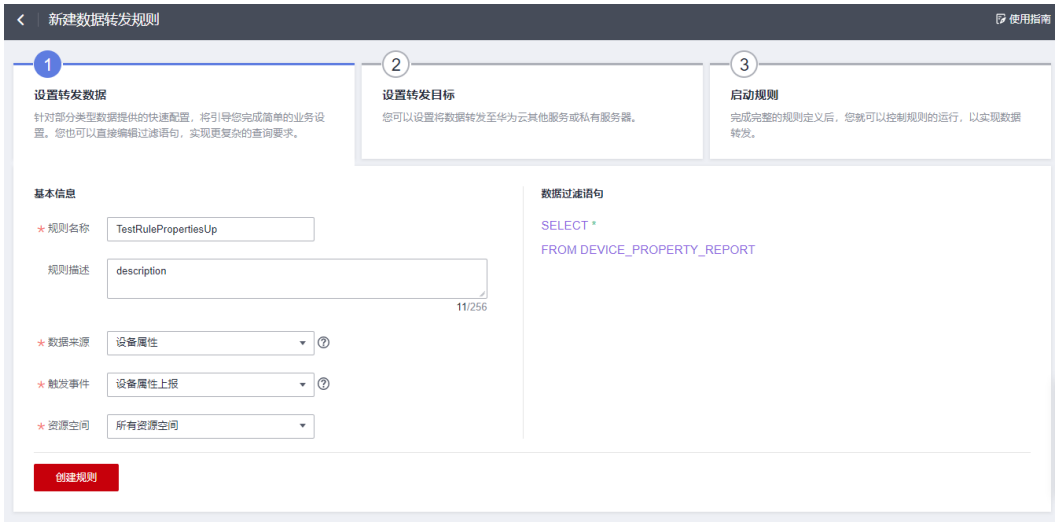
本文以AMQP为例，获取设备上报到云端的数据。

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。

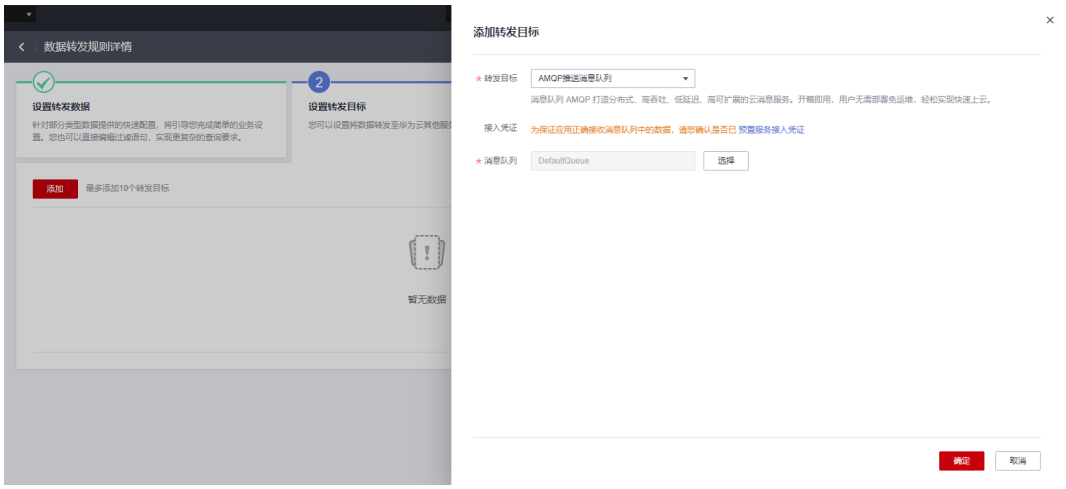
图 1-12 数据转发-创建规则

步骤3 设置转发数据，填入参数，创建规则。



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）。



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。

步骤6 修改**步骤1**获取的AMQP代码样例中的参数。

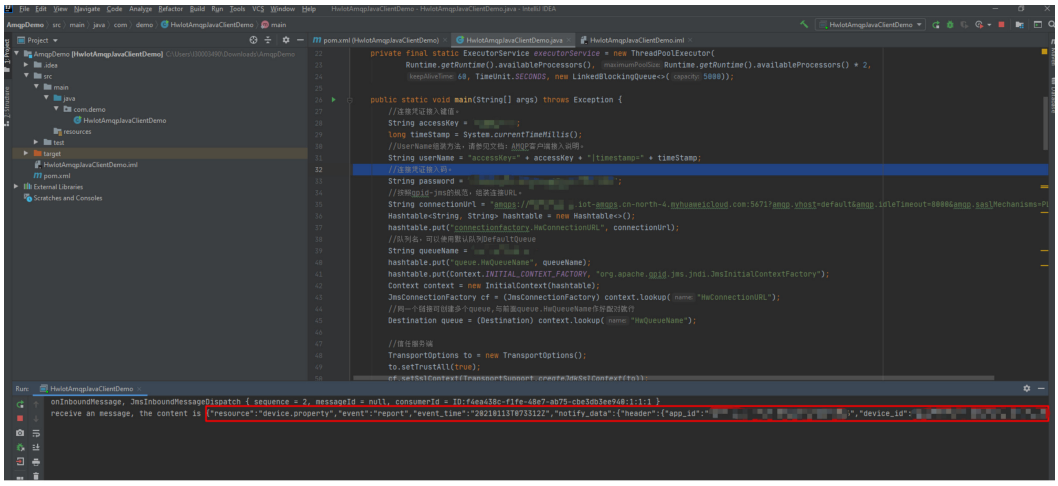
```
        Runtime.getRuntime().availableProcessors(), maximumPoolSize: Runtime.getRuntime().availableProcessors
        keepAliveTime: 60, TimeUnit.SECONDS, new LinkedBlockingQueue<>( capacity: 5000));

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：

- 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**。

1.4 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。

步骤2 选择左侧导航栏的“产品”，单击左侧“创建产品”。

图 1-13 产品-创建产品

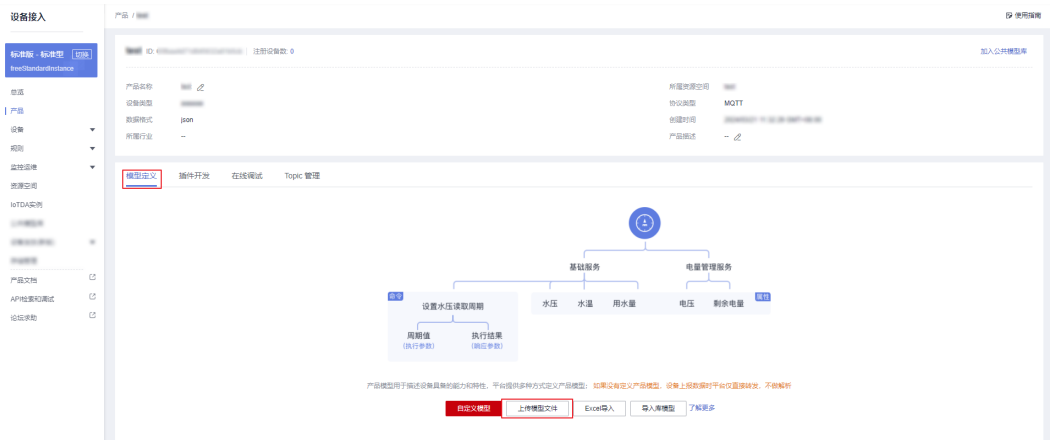
步骤3 在弹出的窗口中，内容根据实际情况自定义填写填写。

图 1-14 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 1-15 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 1-16 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。

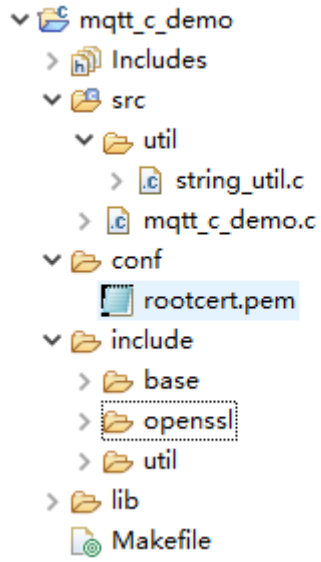
步骤3 设备创建成功后，需要保存设备ID和密钥（后续设备连接的时候需要用到）。

----结束

导入代码样例

步骤1 下载quickStart(C)样例。

步骤2 将代码拷贝到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

```
[root@server-1908071538 test]# cd openssl-1.1.1d
[root@server-1908071538 openssl-1.1.1d]# ./config 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。

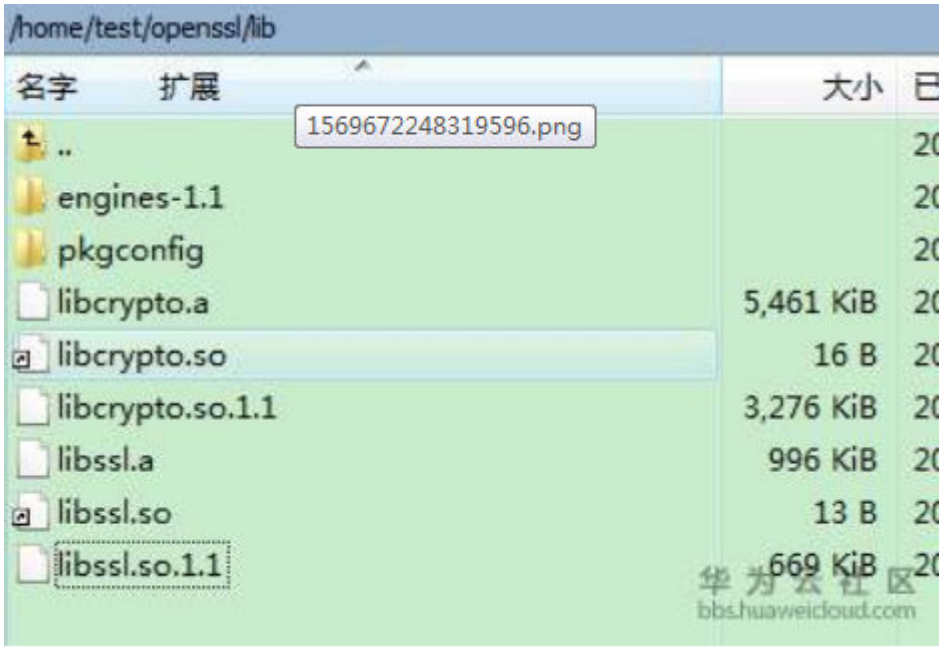
```
make
```

安装openssl。

```
make install
```

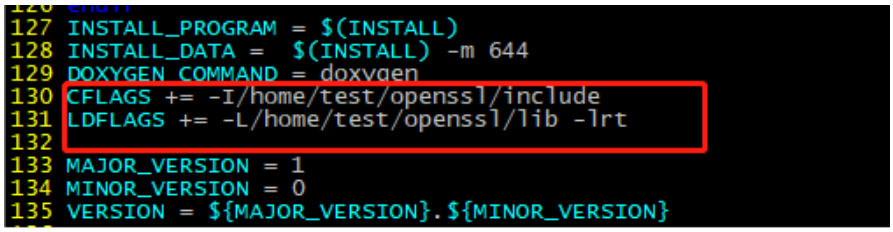
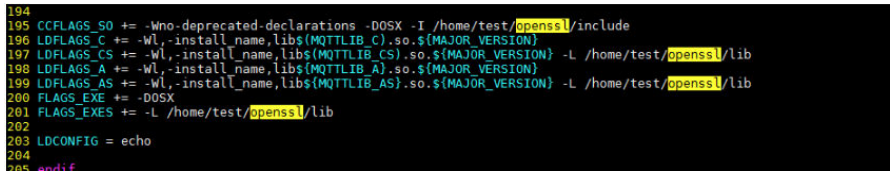
在配置的openssl安装目录下home/test/openssl找到lib目录，有生成的库文件：

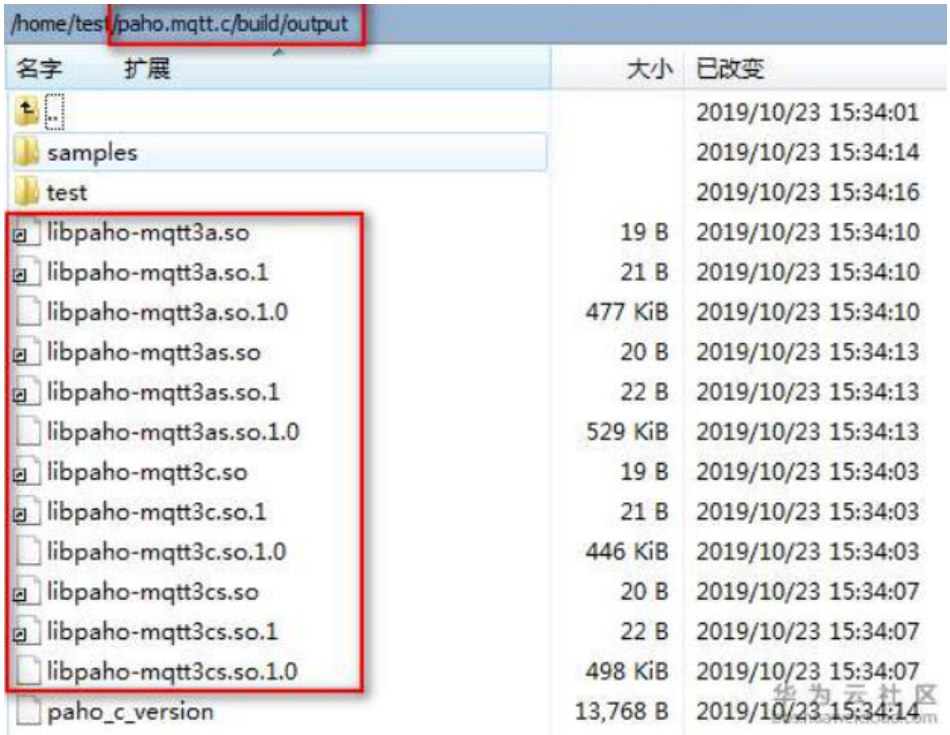
“libcrypto.so.1.1”、“libssl.so.1.1”和软链接“libcrypto.so”、“libssl.so”，请将这些文件拷贝到quickStart(C)的lib文件夹下（同时将/home/test/openssl/include/openssl里的内容拷贝到quickStart(C)的include/openssl下）。



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
   ```
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. 拷贝paho库文件。

当前SDK仅用到了libpaho-mqtt3as，请将“libpaho-mqtt3as.so”和“libpaho-mqtt3as.so.1”文件拷贝到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拷贝到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 1-17 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：

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：

```
mqtt_message_arrive() success, the topic is $oc/devices/[redacted]/sys/commands/request_id=[redacted], the payload is {"paras":{"switch":"OFF"},"service_id":"BasicData","command_name":"lightControl"}
```

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 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。
  b. 选择左侧导航栏的“产品”，单击页面右上角的“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。
厂商名称	自定义。长度不超过32，只允许中文、字母、数字、以及_?'#().,&%@!-等字符的组合。
所属行业	请根据实际情况填写。若使用平台预置的产品模型，请根据产品模型所属的行业填写。
设备类型	使用平台预置的产品模型，会自动关联设备类型，不需要再输入设备类型。
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。