

设备管理

最佳实践

文档版本

11

发布日期

2021-04-15



版权所有 © 华为技术有限公司 2023。保留一切权利。

非经本公司书面许可，任何单位和个人不得擅自摘抄、复制本文档内容的部分或全部，并不得以任何形式传播。

商标声明



HUAWEI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 数据转发	1
1.1 数据转发至 OBS 长期储存（联通用户专用）.....	1
2 设备联动	6
2.1 设备触发告警并邮件或短信通知（联通用户专用）.....	6
2.2 温度过高时自动关闭设备（联通用户专用）.....	12
3 构建应用服务器	16
3.1 基于 IoT Studio 构建应用服务器（联通用户专用）.....	16
3.2 基于软件开发平台构建应用服务器（联通用户专用）.....	30
4 设备接入	51
4.1 基于小熊派开发智慧路灯（联通用户专用）.....	51
4.2 基于唯传 LoRa 网关和体验板接入物联网平台（联通用户专用）.....	84

1 数据转发

数据转发至OBS长期储存（联通用户专用）

1.1 数据转发至 OBS 长期储存（联通用户专用）

非联通用户请查看[设备接入服务](#)。

场景说明

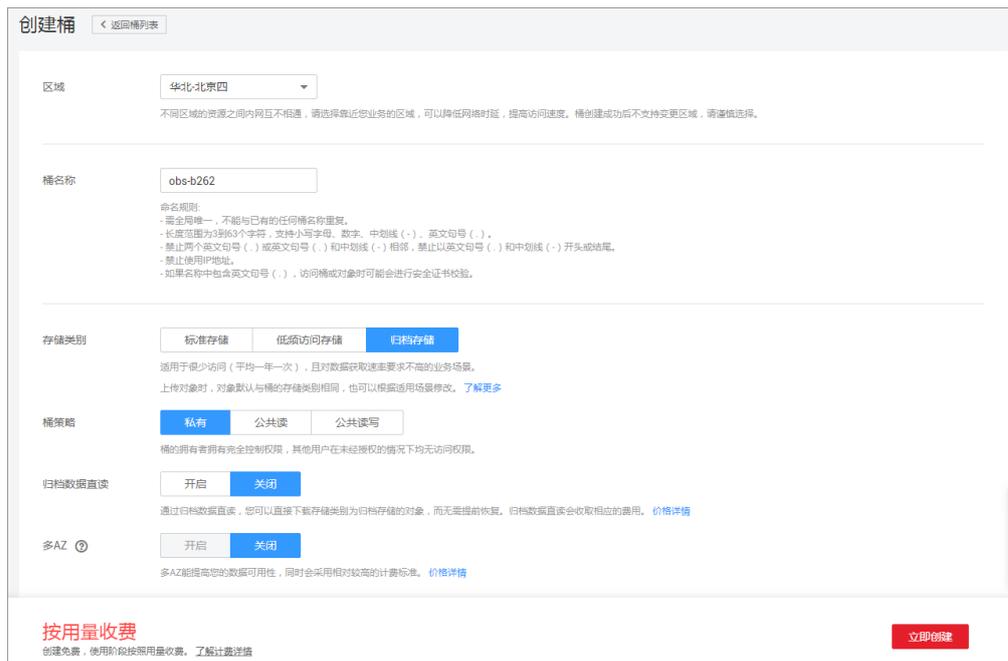
对于设备上报的数据，即使创建应用时开启了存储数据能力，设备管理服务也只保存7天的历史数据。

若您希望长期保存数据，可以选择让平台将设备上报数据推送给应用服务器，由应用服务器进行保存；还可以选择让平台将设备上报数据转发给对象存储服务（OBS），由OBS进行存储。

本示例为将所有设备上报的数据转发至OBS存储。

创建 OBS 桶

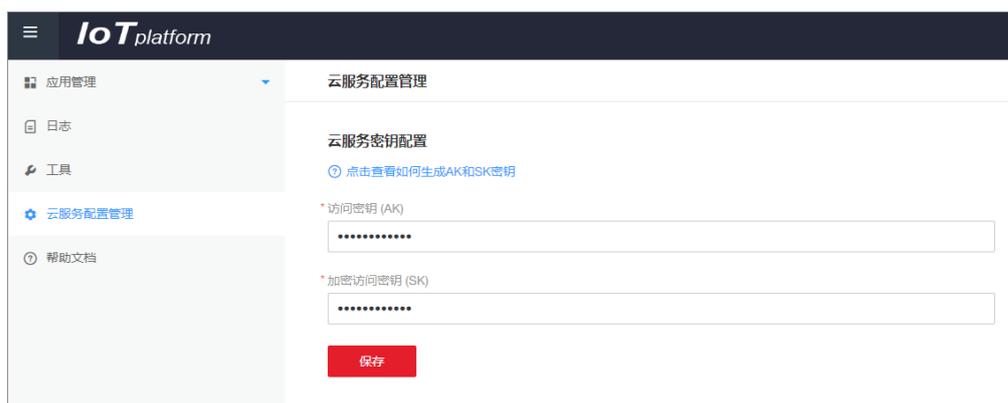
1. 登录华为云官方网站，访问[对象存储服务](#)。
2. 单击“管理控制台”进入对象存储服务管理控制台。
3. 单击页面右上角的“创建桶”，根据需求选择桶规格后，单击“立即创建”。



配置设备管理服务

在设备管理服务中创建产品模型、注册设备并设置数据转发规则，实现当设备上报数据时将数据转发至OBS。

1. 登录华为云官方网站，访问“设备管理服务”。
2. 单击“立即使用”进入设备管理控制台。
3. 单击页面右上角的“进入设备管理服务”进入控制台。
4. 进入“云服务配置管理”页面，按照页面提示配置与华为云对接。



5. 单击页面左下角的  切换左侧菜单，打开“产品模型”页面，单击页面右上角的“新增产品模型 > 本地导入”。

注：本文中使用的产品模型和设备仅为示例，您可以使用自己的产品模型和设备进行操作。

6. 导入样例模型文件，“产品名称”自定义，如“MQTT_Device”。点此获取样例模型文件

手动导入产品
✕

*** 产品名称**

*** 资源文件**

⤴

兼容旧版本的设备模型

确定

取消

7. 进入“设备 > 设备注册”页面，单击“创建”，填写设备注册参数。

单设备注册

*** 选择产品**

*** 设备识别码** ?

*** 预置密钥** ?

*** 确认密钥**

确定

取消

参数名称	说明
选择产品	选择您导入的产品模型，如“MQTT_Device”。

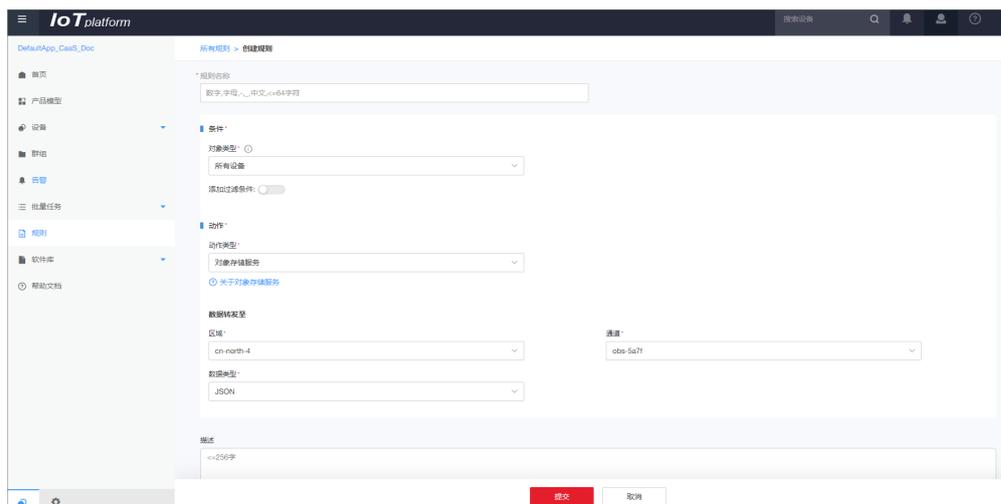
参数名称	说明
设备标识码	<p>设备唯一物理标识，如IMEI、MAC地址等，用于设备在接入物联网平台时携带该标识信息完成接入鉴权。</p> <ul style="list-style-type: none"> 原生MQTT设备：通过注册成功后生成的“设备ID”（与设备标识码一一对应）和“密钥”接入平台。 NB-IoT设备、集成Agent Lite SDK的设备：设备通过注册时填写的“设备标识码”和“预置密钥”接入平台。 <p>无真实设备时可随意填写英文字母和数字的组合字符串。</p>
预置密钥	<ul style="list-style-type: none"> NB-IoT设备、集成Agent Lite SDK的设备接入时，用于设备和物联网平台之间的传输通道安全加密。 原生MQTT设备接入时，暂无作用，可随意填写。
确认密钥	再次填写预置密钥。

填写完成后单击“确定”，请注意保存注册成功返回的“设备ID”和“密钥”。

8. 进入“规则”页面，单击“创建规则”，再单击“数据转发规则”。



9. 填写规则内容，规则名称自定义，动作类型选择“对象存储服务”，“区域”选择您开通OBS的区域，“通道”选择您创建的桶，“数据类型”选择“JSON”，填写完成后单击“提交”。



验证操作

- 您可以使用配置设备管理服务时注册的真实设备接入平台，上报任意数据。
- 您也可以使用模拟器模拟设备上报数据，操作方法请参考[通过MQTT.fx体验设备接入](#)。

期望结果：

登录OBS管理控制台，单击桶名称进入桶管理页面后，在“对象”页面可以查看到设备上报的数据。



您也可以使用OBS的API进行文件的读取。

2 设备联动

[设备触发告警并邮件或短信通知（联通用户专用）](#)

[温度过高时自动关闭设备（联通用户专用）](#)

2.1 设备触发告警并邮件或短信通知（联通用户专用）

非联通用户请查看[设备接入服务](#)。

场景说明

很多物联网设备24小时运行，设备管理者并不需要知道设备每一刻的状态，只需要设备在上报某些特定状态时能通知到管理者。

设备管理服务提供规则引擎功能可以满足该诉求，您可以通过设置规则简单实现当设备上报的数据满足某个条件时，物联网平台触发指定动作进行通知。

本示例为设备上报的电池电量小于20%时，物联网平台会上报告警并发送邮件或短信通知给指定的手机号码。

配置消息通知服务

在消息通知服务中创建主题并添加订阅，供设备管理服务调用以发送邮件或短信。

1. 登录华为云官方网站，访问[消息通知服务](#)。
2. 单击“立即使用”进入消息通知服务控制台。若您未开通消息通知服务，请先开通服务。
3. 进入“主题管理 > 主题”页面，单击“创建主题”。
4. 输入主题名称，如“Battery_Low_Notify”，单击“确定”。

创建主题 ✕

*** 主题名称** ?
主题创建后，不允许修改主题名称。

显示名 ?

标签 如果您需要使用同一标签标识多种云资源，即所有服务均可在标签输入框下拉选择同一标签，建议在TMS中创建预定义标签。 [查看预定义标签](#) ↻

该主题还可以创建10个标签

确定
取消

5. 进入“主题管理 > 订阅”页面，单击“添加订阅”。
6. 填写订阅信息。

添加订阅 ✕

主题名称 +

协议 ▼

订阅终端 ?

确定
取消

参数名称	说明
主题名称	选择步骤4创建的主题。
协议	<ul style="list-style-type: none"> 若需要发送邮件通知，选择“邮件”。 若需要发送短信通知，选择“短信”。
订阅终端	<ul style="list-style-type: none"> 若协议为“邮件”，填写接收通知邮件的邮箱地址； 若协议为“短信”，输入接收短信通知的手机号码。 需要添加多个终端时，每个终端地址占1行，最多填写10行。

填写完成后单击“确定”。

配置设备管理服务

在设备管理服务中创建产品模型、注册设备并设置设备联动规则，实现当设备上报特定数据时触发告警并发送邮件或短信。

1. 登录华为云官方网站，访问“设备管理服务”。
2. 单击“立即使用”进入设备管理控制台。
3. 单击页面右上角的“进入设备管理服务”进入控制台。
4. 进入“云服务配置管理”页面，按照页面提示配置与华为云对接。



5. 单击页面左下角的  切换左侧菜单，打开“产品模型”页面，单击页面右上角的“新增产品模型 > 本地导入”。
注：本文中使用的产品模型和设备仅为示例，您可以使用自己的产品模型和设备进行操作。
6. 导入样例模型文件，“产品名称”自定义，如“MQTT_Device”。点此获取样例模型文件



7. 进入“设备 > 设备注册”页面，单击“创建”，填写设备注册参数。

单设备注册

*选择产品

*设备标识码

*预置密钥

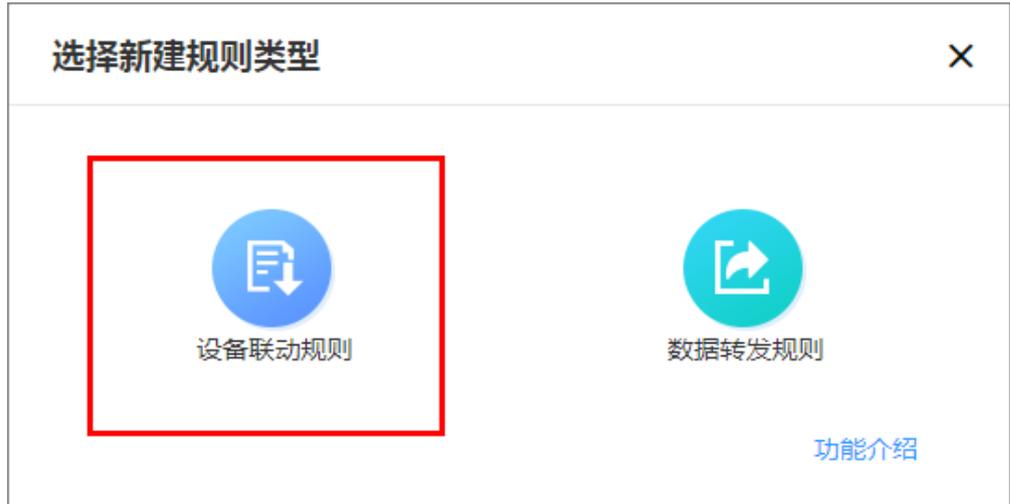
*确认密钥

确定
取消

参数名称	说明
选择产品	选择您导入的产品模型，如“MQTT_Device”。
设备标识码	<p>设备唯一物理标识，如IMEI、MAC地址等，用于设备在接入物联网平台时携带该标识信息完成接入鉴权。</p> <ul style="list-style-type: none"> 原生MQTT设备：通过注册成功后生成的“设备ID”（与设备标识码一一对应）和“密钥”接入平台。 NB-IoT设备、集成Agent Lite SDK的设备：设备通过注册时填写的“设备标识码”和“预置密钥”接入平台。 <p>无真实设备时可随意填写英文字母和数字的组合字符串。</p>
预置密钥	<ul style="list-style-type: none"> NB-IoT设备、集成Agent Lite SDK的设备接入时，用于设备和物联网平台之间的传输通道安全加密。 原生MQTT设备接入时，暂无作用，可随意填写。
确认密钥	再次填写预置密钥。

填写完成后单击“确定”，请注意保存注册成功返回的“设备ID”和“密钥”。

8. 进入“规则”页面，单击“创建规则”，再单击“设备联动规则”。



9. 填写规则内容，以下参数取值仅为示例，您可参考[用户指南](#)创建自己的规则。

参数名称	说明
规则名称	创建的规则名称，例如“Battery_Low_Mail”或“Battery_Low_SMS”。
立即触发	选择“是”。
规则类型	选择“云端规则”。
时间设置	选择“总是”。
条件	<ol style="list-style-type: none"> 1. 添加“设备行为”； 2. “条件类型”选择“设备类型”，设备模型选择6导入的产品模型。 3. “服务类型”选择“Battery”，“属性名字”选择“batteryLevel”，“操作”选择“<”，“值”填写“20”，“数据时效”填写“3600”，“延后触发”填写“0”。 4. 单击“提交”。

参数名称	说明
动作	<p>首先添加告警：</p> <ol style="list-style-type: none"> 1. 添加“发送信息或告警”。 2. “动作类型”选择“事件告警”。 3. “告警类型”选择“待处理”，“告警等级”选择“一般”，“告警名字”输入“设备电量低”，“告警内容”输入“设备电量低于20%，请及时确认并更换电池。” 4. 单击“提交”。 <p>再添加通知：</p> <ol style="list-style-type: none"> 1. 添加“发送信息或告警”。 2. “动作类型”选择“主题通知”。 3. 选择您开通的SMN服务区域，如“cn-north-4”（可登录SMN服务控制台后在左上角查看）。 4. 主题名称选择配置消息通知服务时创建的主题； <ul style="list-style-type: none"> ● 若主题对应的订阅协议是邮件，消息标题填写为邮件标题，如“[Huawei IoT Platform] Low Battery Warning”；消息内容填写“You have a device with less than 20% charge, please log in to the Huawei IoT Platform for details.” ● 若主题对应的订阅协议是短信，消息标题无需填写，消息内容填写“[Huawei IoT Platform] You have a device with less than 20% charge, please log in to the Huawei IoT Platform for details.” 5. 单击“提交”。
描述	对该规则的描述，如“设备上报电量低于20%时上报告警并发送通知”。

验证操作

- 您可以使用配置设备管理服务时注册的真实设备接入平台，上报“batteryLevel”小于20的数据。
- 您也可以使用模拟器模拟设备上报“batteryLevel”小于20的数据，操作方法请参考[通过MQTT.fx体验设备接入](#)。

期望结果：

- 设备管理服务控制台的“告警”页面会出现一条名称为“设备电量低”的一般告警。



- 若订阅了邮件通知，接受订阅的邮箱会收到一封通知设备电量低的邮件。
- 若订阅了短信通知，接受订阅的手机会收到一条通知设备电量低的短信。

2.2 温度过高时自动关闭设备（联通用户专用）

非联通用户请查看[设备接入服务](#)。

场景说明

物联网平台支持设备数据上报和设备命令下发，但若要将两者联动起来，一般需要由应用服务器实现对应逻辑。

设备管理服务提供规则引擎功能，平台上简单几步操作即可实现数据上报特定数据时平台自动下发指定命令，减少应用服务器开发工作量。

本示例为设备自带的温度传感器上报的温度大于80度时，平台自动下发命令关闭设备。

配置设备管理服务

在设备管理服务中创建产品模型、注册设备并设置设备联动规则，实现当设备上报特定数据时平台自动下发命令。

1. 登录华为云官方网站，访问“设备管理服务”。
2. 单击“立即使用”进入设备管理控制台。
3. 单击页面右上角的“进入设备管理服务”进入控制台。
4. 进入“云服务配置管理”页面，按照页面提示配置与华为云对接。



5. 单击页面左下角的  切换左侧菜单，打开“产品模型”页面，单击页面右上角的“新增产品模型 > 本地导入”。
- 注：本文中使用的产品模型和设备仅为示例，您可以使用自己的产品模型和设备进行操作。
6. 导入样例模型文件，“产品名称”自定义，如“MQTT_Device2”。[点此获取样例模型文件](#)

手动导入产品
✕

*** 产品名称**

*** 资源文件**

MQTTDevice2_Profile_Sample.zip
📁

兼容旧版本的设备模型

确定

取消

7. 进入“设备 > 设备注册”页面，单击“创建”，填写设备注册参数。

单设备注册

*** 选择产品**

*** 设备识别码** ?

*** 预置密钥** ?

*** 确认密钥**

确定

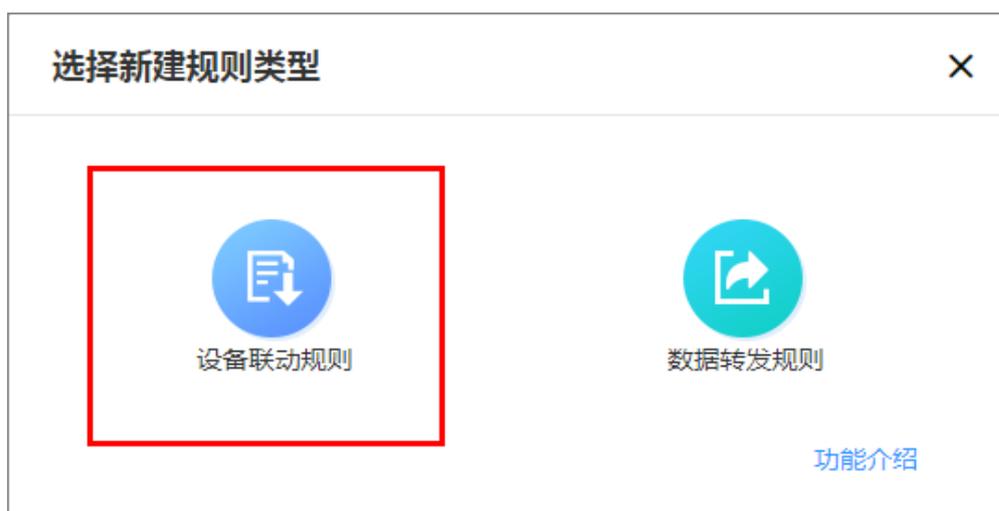
取消

参数名称	说明
选择产品	选择您导入的产品模型，如“MQTT_Device2”。

参数名称	说明
设备标识码	<p>设备唯一物理标识，如IMEI、MAC地址等，用于设备在接入物联网平台时携带该标识信息完成接入鉴权。</p> <ul style="list-style-type: none"> 原生MQTT设备：通过注册成功后生成的“设备ID”（与设备标识码一一对应）和“密钥”接入平台。 NB-IoT设备、集成Agent Lite SDK的设备：设备通过注册时填写的“设备标识码”和“预置密钥”接入平台。 <p>无真实设备时可随意填写英文字母和数字的组合字符串。</p>
预置密钥	<ul style="list-style-type: none"> NB-IoT设备、集成Agent Lite SDK的设备接入时，用于设备和物联网平台之间的传输通道安全加密。 原生MQTT设备接入时，暂无作用，可随意填写。
确认密钥	再次填写预置密钥。

填写完成后单击“确定”，请注意保存注册成功返回的“设备ID”和“密钥”。

8. 进入“规则”页面，单击“创建规则”，再单击“设备联动规则”。



9. 填写规则内容，以下参数取值仅为示例，您可参考[用户指南](#)创建自己的规则。

参数名称	说明
规则名称	创建的规则名称，例如“Overheated”。
立即触发	选择“是”。
规则类型	选择“云端规则”。
时间设置	选择“总是”。
条件	<ol style="list-style-type: none"> 1. 添加“设备行为”； 2. “条件类型”选择“设备类型”，设备模型选择6导入的产品模型。 3. “服务类型”选择“tempSensor”，“属性名字”选择“temperature”，“操作”选择“>”，“值”填写“80”，“数据时效”填写“300”，“延后触发”填写“0”。 4. 单击“提交”。
动作	<ol style="list-style-type: none"> 1. 添加“设备行为”； 2. “动作类型”选择“设备”，设备模型选择6导入的产品模型，设备选择7创建的设备。 3. “服务类型”选择“deviceSwitch”，“命令名字”选择“ON_OFF”，参数选择“power”，“值”选择“OFF”，“命令状态”选择“启用”。 4. 单击“提交”。
描述	对该规则的描述，如“设备温度高于80度时自动关闭设备”。

填写完成后单击“提交”。

验证操作

- 您可以使用配置设备管理服务时注册的真实设备接入平台，上报“temperature”大于80的数据。
- 您也可以使用模拟器模拟设备订阅Topic: /huawei/v1/devices/{deviceId}/command/json（其中{deviceId}需要替换成实际设备ID）后，上报“temperature”大于80的数据，操作方法请参考[通过MQTT.fx体验设备接入](#)。

期望结果：

- 若您使用真实设备上报数据，设备会收到一条“power”取值为“OFF”的“ON_OFF”命令。
- 若您使用模拟器上报数据，可在“Subscribe”页签查看到平台下发的“power”取值为“OFF”的“ON_OFF”命令。

3 构建应用服务器

基于IoT Studio构建应用服务器（联通用户专用）

基于软件开发平台构建应用服务器（联通用户专用）

3.1 基于 IoT Studio 构建应用服务器（联通用户专用）

非联通用户请查看[设备接入服务](#)。

场景说明

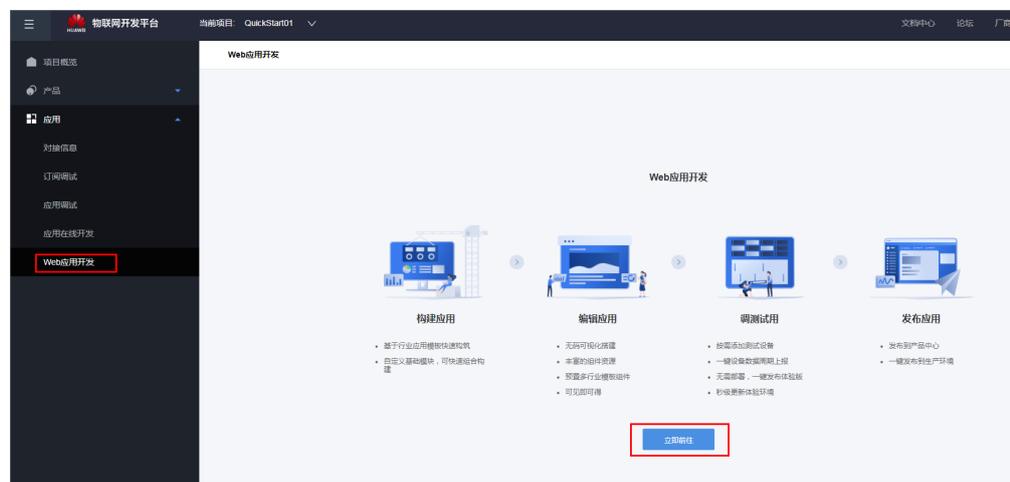
传统的物联网应用需要开发者使用物联网平台提供的API或SDK进行二次开发，对开发者的编程技能有着较高的要求，且开发周期较长。

华为云物联网平台提供IoT Studio服务，实现在线可视化的应用服务器开发，简单的拖曳式操作，所见即所得的开发界面，大大降低开发周期。

本示例基于小熊派开发板的智慧路灯方案构建一个应用服务器，请先参考[基于小熊派的智慧路灯](#)完成平台侧和设备的开发。

构建应用

步骤1 在开发中心中，选择“应用 > Web应用开发”，点击“立即前往”，进入IoT Studio平台。



步骤2 在IoT Studio平台首页，点击“构建应用”。



步骤3 在“新建应用”中，填写应用信息，点击“创建”。

- 应用资料
应用名称：StreetLightAPP
- 构建方式选择“自定义”。
选择基础功能模块：设备注册（必选）、设备列表、规则。



---结束

编辑应用

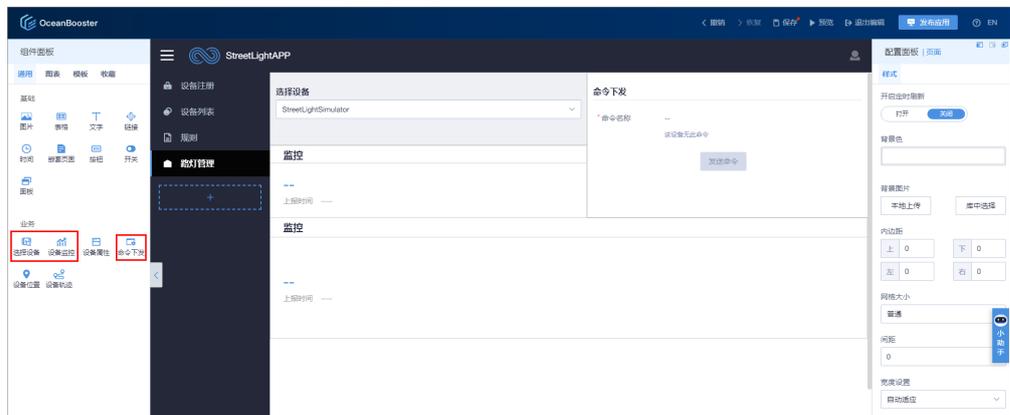
步骤1 将鼠标移至“自定义页面1”上，在弹出的列表中选择“修改”，修改页面信息。

在弹出的窗口中，修改“菜单名称”为“路灯管理”，其他保持默认，然后点击“确定”。



步骤2 选择“路灯管理”页面，设计页面组件布局。

1. 拖动1个“选择设备”组件、2个“设备监控”组件和1个“命令下发”组件至页面中，并按下图所示的布局进行摆放。



2. 分别单击页面中的“设备监控”组件，在右侧“配置面板 > 样式”中设置组件的样式。

参数	光强监控	光强变化
标题	光强监控	光强变化
显示类型	简易	图表
样式	保持默认	

3. 分别单击页面中的“设备监控”组件，在右侧“配置面板 > 数据”中设置组件的数据源。

因为2个“监控”组件都是用于监控路灯的光强，只是显示方式不同，所以“数据”页面参数设置一样。

- 产品：选择[创建产品](#)中已创建的产品
- 服务：Sensor
- 属性：luminance

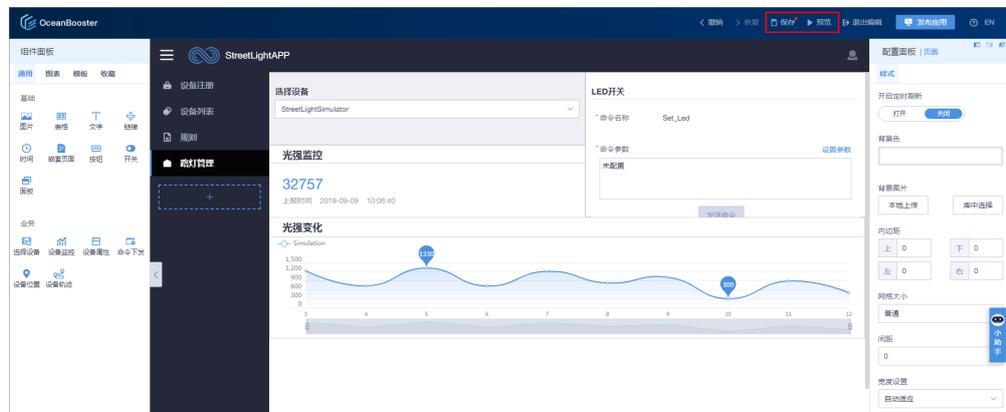


4. 单击页面中的“命令下发”组件，在右侧“配置面板”设置对应功能的属性参数。

- 设置“样式”页签参数。
 - 标题：LED开关
 - 其他参数：保持默认
- 设置“数据”页签参数。
 - 产品：选择[创建产品](#)中已创建的产品
 - 服务：LED
 - 命令：Set_Led



步骤3 路灯管理页面构建完成，点击右上角“保存”，然后点击“预览”查看应用页面效果。



----结束

调试应用

在本章节中，我们会为智慧路灯设计三种开关模式，分别是终端联控模式、分段定时模式和自动调光模式。

步骤1 在已构建应用的预览界面中，选择“设备注册 > 单个注册”，点击“创建”。

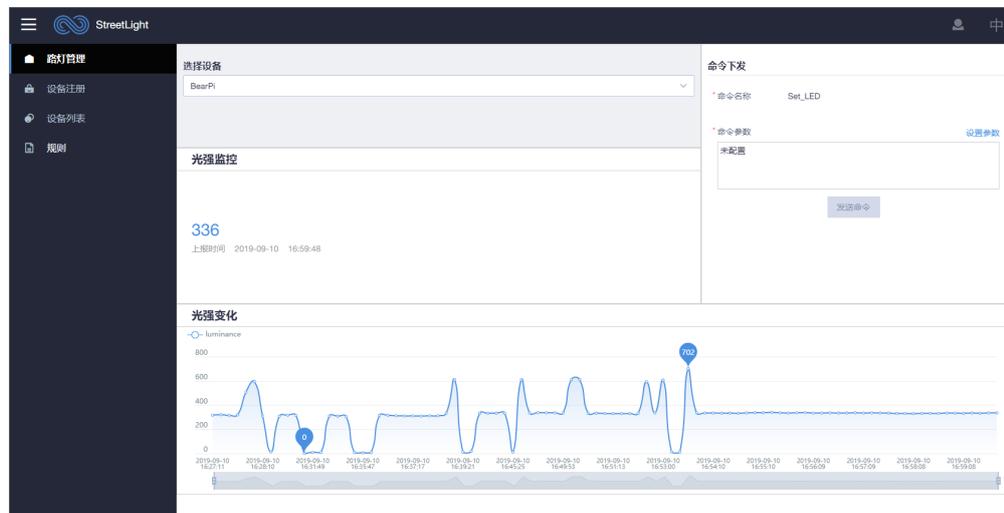


步骤2 系统将弹出“单个设备注册”窗口，填写设备相关信息，点击“确定”。

- 产品选择：选择已创建的产品
- 设备识别码：填写开发板的IMEI号，可以在开发板上进行查看。
- 预置密钥：自定义
- 确认密钥：与预置密钥一致



步骤3 开发板上电，在“路灯管理”页面，可以观察光强监控和光强变化。

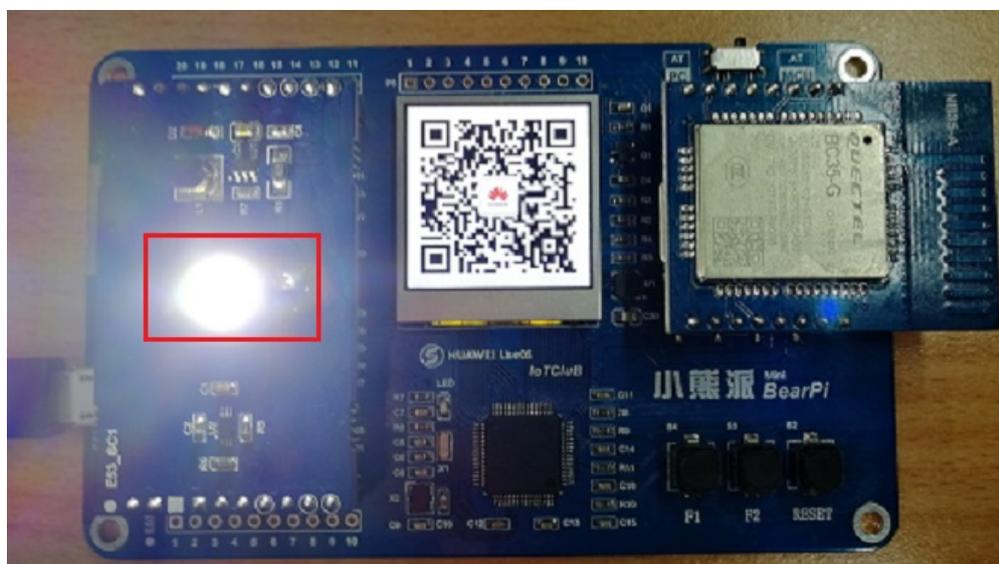


步骤4 首先验证终端联控模式。终端联控模式是指用户在平台直接控制路灯的开关。

选择“路灯管理 > LED开关”，点击“设置参数”，“led”选择“ON”，单击“发送命令”。



此时开发板的照明灯为打开状态。



关灯命令与开灯命令操作步骤一样，差别是“led”选择“OFF”。

步骤5 然后实现分段定时模式。分段定时模式是指路灯在指定时间段自动开关。

新建两条规则，分别用于控制照明灯在不同时间下的开和关。

选择“规则”，单击“创建规则”。



步骤6 在“创建规则”界面，分别填写指定时间开、关灯规则信息。

参数	开	关
规则名称	TIME_ON	TIME_OFF
条件		

参数	开	关
时间点	为方便验证，建议选择当前时间的数分钟后。	为方便验证，建议选择开灯时间的一分后。
动作		
动作类型	设备	
选择设备模型	选择已创建的产品	
点击选择设备	选择本章节中新增的设备	
服务类型	LED	
命令名称	Set_Led	
参数	led	
值	ON	OFF
命令状态	启用	
描述	指定时间点自动开灯。	指定时间点自动关灯。

1. 参照上述表格，填写开灯的规则信息。
规则名称：TIME_ON，红框内其他参数取默认值。

所有规则 > 创建规则

*规则名称
TIME_ON

*立即触发 ①
是

*规则类型 ①
云端规则

*时间设置
总是

标签 ①

条件

需满足以下 全部 条件:

设备行为 + 添加

时间 + 添加

动作

设备行为 + 添加

发送消息或告警 + 添加

描述
≤ 256字

提交 取消

2. 设置开灯规则的条件信息。在“条件”模块，点击“时间”所在行右侧的“添加”。

条件

需满足以下 全部 条件:

设备行为 + 添加

时间 + 添加

3. 按照参数表中开灯规则的条件参数填写条件信息。

时间

时间点*

18:00

重复次数

重复间隔 (分钟)

提交

取消

4. 设置开灯规则的动作。在“动作”模块，点击“设备行动”所在行右侧的“添加”。

条件

需满足以下 全部 条件:

设备行为	+ 添加
时间	+ 添加

5. 按照参数表中开灯规则的动作参数填写动作信息。

动作类型*

设备

选择设备模型*

选择设备模型

设备模型*

OC_StreetLight001

设备类型*

StreetLight

厂商*

8b31ec1ded0c4e01ab63ead32fb2411f

点击选择设备

已选设备1

服务类型*

LED

命令名字*

Set_LED

参数*

led

值*

ON

命令状态*

启用

命令请求ID

回调地址

过期时间(秒)

命令状态变化通知地址, 当命令状态变化的时候执行失败, 执行成功, 超时, 发送, 已通知会通知NA

命令有效的超期时间, 单位为秒, 当为0时, 表示立即下载命令, 其他值表示缓存下发命令, 不传时, 默认为48小时

提交

取消

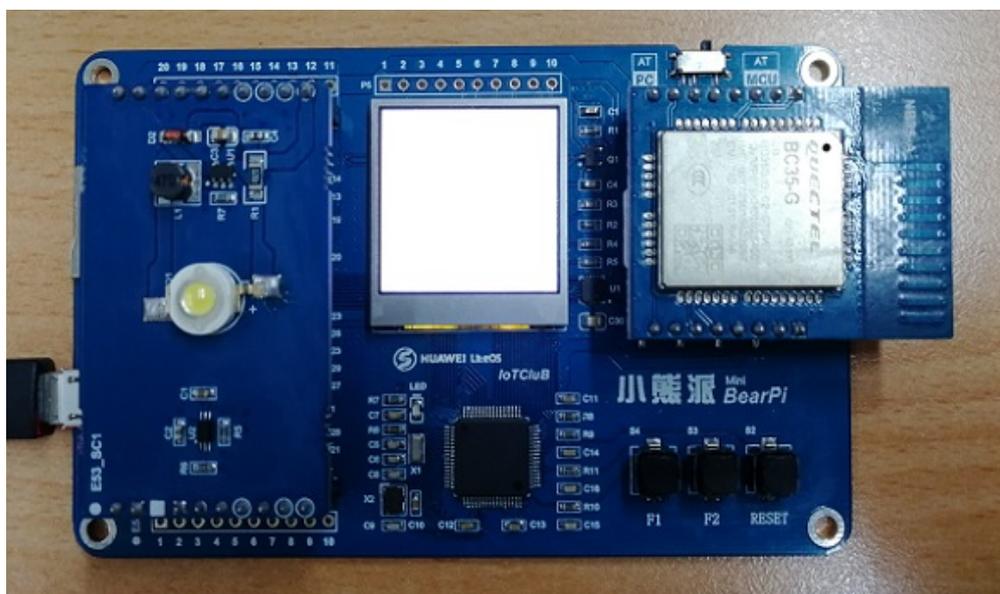
然后单击右上角的“提交”，开灯规则创建完成。关灯规则的创建操作和开灯的一样，只是规则名称、条件的取值和动作执行不同。

步骤7 测试指定时间自动开关灯。

1. 等到TIME_ON规则设置的时间点，开发板的照明灯应该自动打开。



2. 再等到TIME_OFF规则设置的时间点，开发板的照明灯应该自动关闭。



步骤8 最后实现自动调光模式。自动调光模式是指路灯在指定的光强下自动开关。

新建两条规则，分别用于控制照明灯在不同光强下的开和关。

选择“规则”，单击“创建规则”。

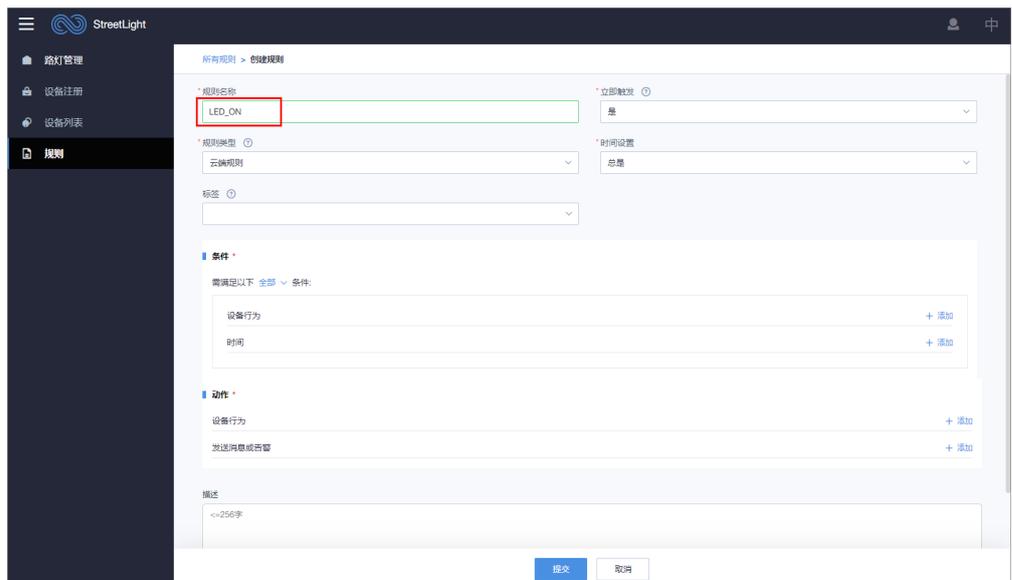
步骤9 在“创建规则”界面，分别填写光强开、关灯规则信息。

参数	开	关
规则名称	LED_ON	LED_OFF
条件		
条件类型	设备类型	
选择设备模型	选择已创建的产品	

参数	开	关
服务类型	Sensor	
属性名字	luminance	
操作	<	>
值	50	500
动作		
动作类型	设备	
选择设备模型	选择已创建的产品	
点击选择设备	选择本章节中新增的设备	
服务类型	LED	
命令名称	Set_Led	
参数	led	
值	ON	OFF
命令状态	启用	
描述	光强小于50时，照明灯开启。	光强大于500时，照明灯关闭。

1. 参照上述表格，填写开灯的规则信息。

规则名称：LED_ON



2. 设置开灯规则的条件信息。在“条件”模块，点击“设备行为”所在行右侧的“添加”。

3. 按照参数表中开灯规则的条件参数填写条件信息。

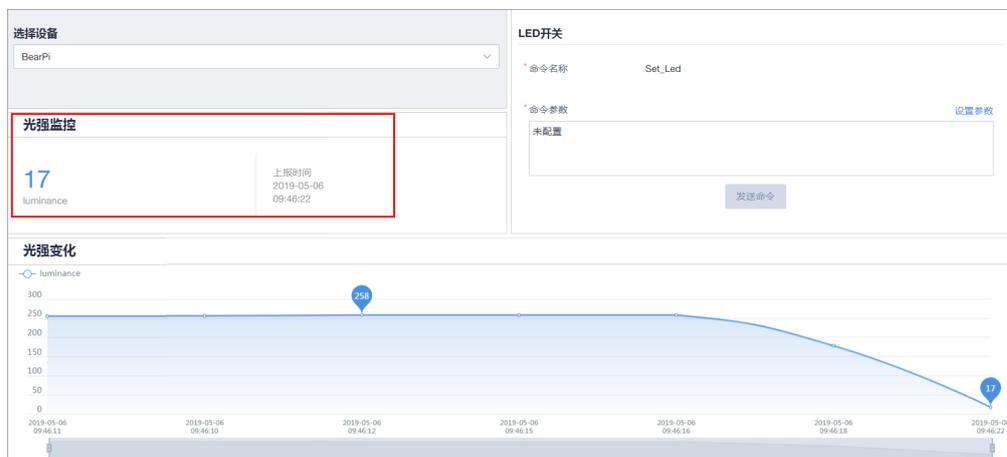
4. 设置开灯规则的动作。在“动作”模块，点击“设备行动”所在行右侧的“添加”。

5. 按照参数表中开灯规则的动作参数填写动作信息。

然后单击右上角的“提交”，开灯规则创建完成。关灯规则的创建操作和开灯的一样，只是规则名称、条件的取值和动作执行不同。

步骤10 测试光强变化自动开关灯。

1. 遮住光强传感器，使开发板处于黑暗环境中（亮度 <50 ），查看“路灯管理”中光强监控数值和开发板的照明灯：开发板的照明灯应该自动打开。

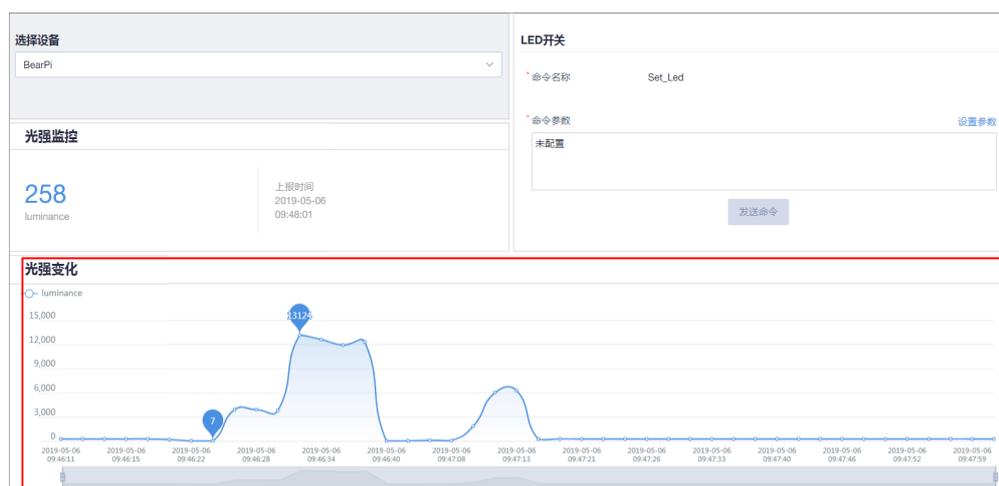


2. 移除遮挡物，使开发板处于明亮环境中（亮度 >500 ），查看“路灯管理”中光强监控数值和开发板的照明灯：开发板的照明灯应该自动关闭。





步骤11 如果需要观察开发板的照明灯在一段时间的变化情况，可以查看“路灯管理”中光强变化。



---结束

3.2 基于软件开发平台构建应用服务器（联通用户专用）

非联通用户请查看[设备接入服务](#)。

场景说明

线下开发一个物联网应用，您需要一个代码库用于管理应用代码，还需要一个可与外网通信的服务器用于部署应用，准备和维护这两者都需要一定的成本。

华为云物联网平台提供基于软件开发平台的托管式应用开发与部署服务，助您摆脱这些杂务，专注于应用开发。

本示例基于小熊派开发板的智慧路灯方案，请先参考[基于小熊派的智慧路灯](#)完成平台侧和设备的开发。

构建应用

步骤1 在开发中心中，选择“应用 > 应用在线开发”，点击“立即前往”。



步骤2 在软件开发平台（DevCloud）首页，点击“新建项目”。



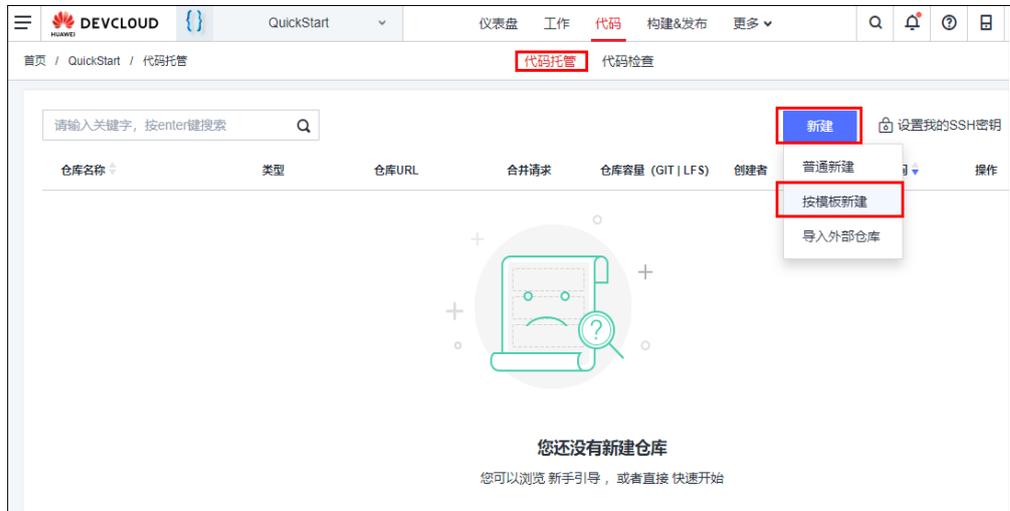
步骤3 在“新建项目”窗口中，项目模板选择“Scrum”，并配置项目信息，点击“确定”。

项目名称：QuickStart

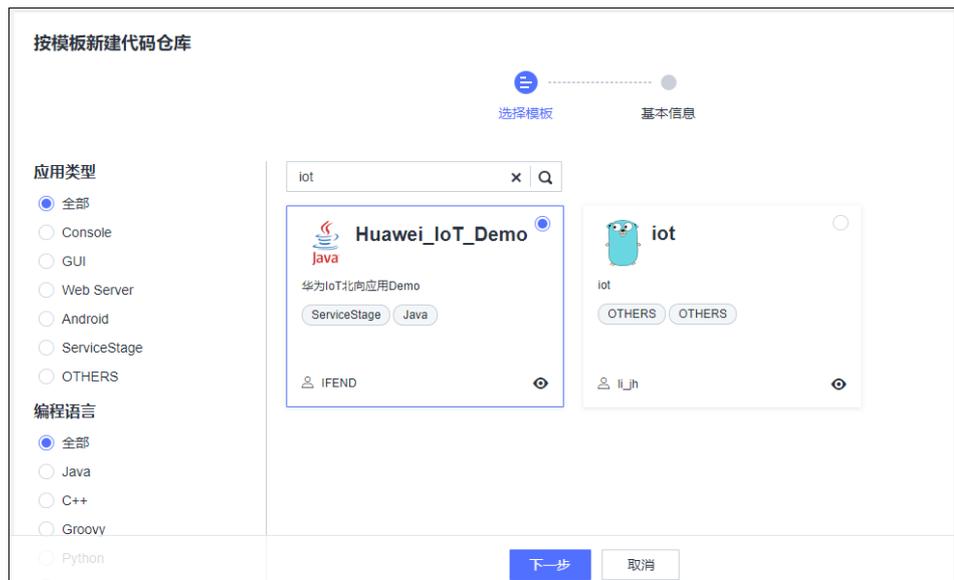


步骤4 在新建的项目中选择“代码 > 代码托管”可进入代码仓库列表。软件开发平台提供三种新建代码仓库的方式：

- 点击“新建 > 普通新建”，新建一个空白的代码仓库。
- 点击“新建 > 按模版新建”，使用模版新建一个代码仓库。本示例中使用这种方式。



- a. 搜索“iot”，点击“Huawei_IoT_Demo”，点击“下一步”。

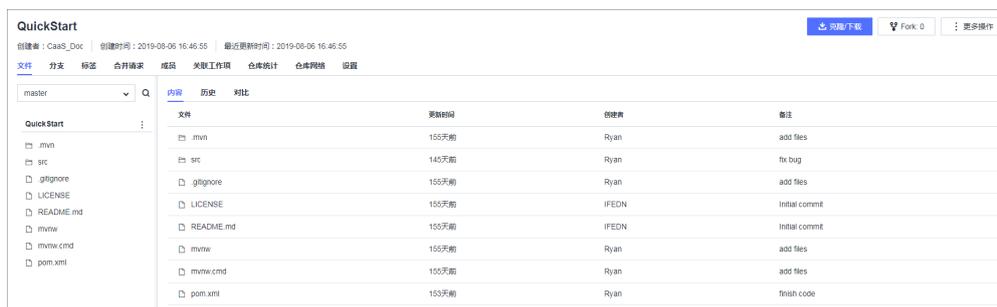


- b. 完成各项参数配置后，点击“确定”。配置说明：
- 请输入代码仓库名称：根据规划进行填写，以“QuickStart”为例
 - 选择.gitignore：Java
 - 其余参数：保持默认即可



- 点击“新建 > 导入外部仓库”，将外部Git或者SVN代码仓库导入软件开发平台。

步骤5 单击新创建的仓库名称，进入该代码仓库。



若您想要定制应用，可以使用Git工具将代码拉到本地修改后再重新提交至代码仓库。

业务代码位于“src > main > java > com > huawei”目录下，几个关键参数的修改方法如下：

- 物联网平台应用接入地址：请修改业务代码目录下“utils > Constants.java”文件中的常量“NORTH_IP”和“BASE_URL”。
- 应用服务器接收推送消息的地址：该样例接收推送的地址格式为“http://IP:8080/receive-data”
 - 其中“IP”为应用服务器的弹性IP，无法通过业务代码修改。
 - “8080”为Spring应用的启动端口，可在“src > main > resources > application.properties”文件中修改“server.port”的值。
 - “/receive-data”为映射地址，可在业务代码目录下“controller > CollectDeviceData.java”文件中修改“PostMapping”的值。

注：本文档后续的操作均基于未修改的代码样例。

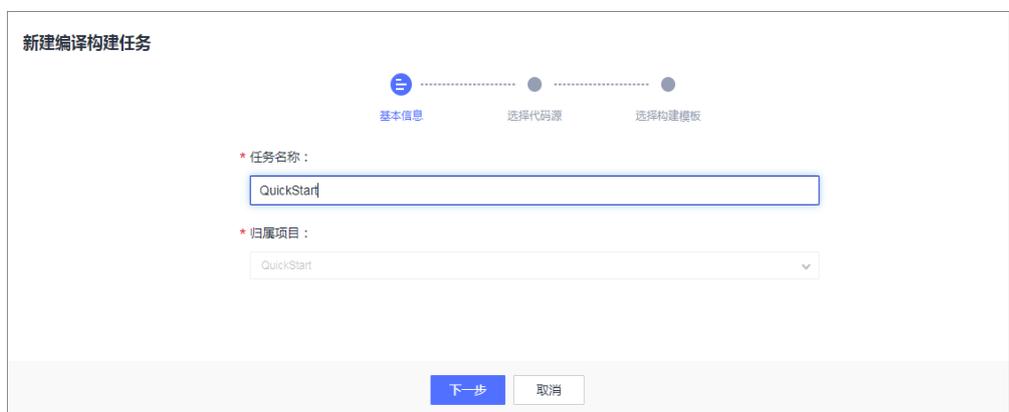
步骤6 选择“构建&发布 > 编译构建”。



步骤7 在“编译构建”中，点击“新建任务”。



步骤8 在“新建编译构建任务 > 基本信息”中，配置任务名称，以“QuickStart”为例，点击“下一步”。

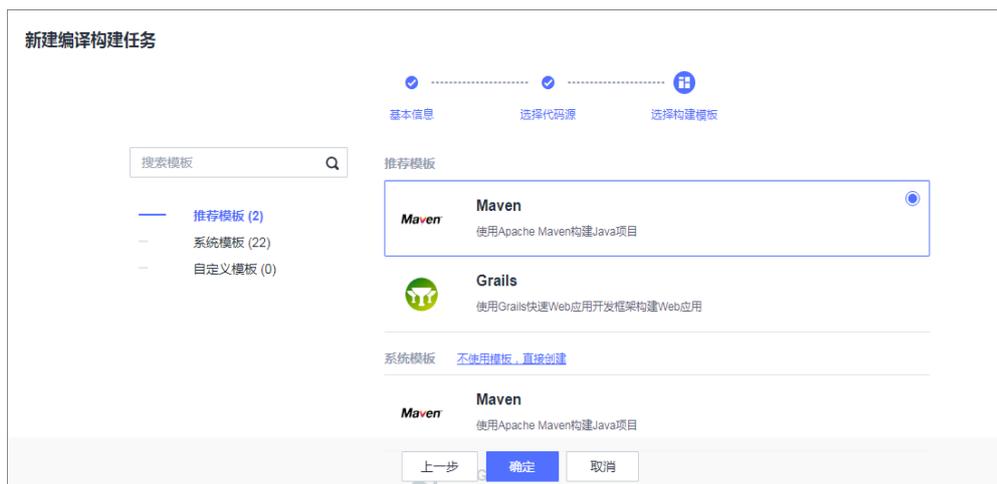


步骤9 在“新建编译构建任务 > 选择源代码”中，配置各项参数，点击“下一步”。配置说明：

- 源码源：DevCloud
- 源码仓库：与创建的代码仓库名称一致，以“QuickStart”为例
- 分支：master



步骤10 在“新建编译构建任务 > 选择构建模板”中，选择“Maven”，点击“确定”。



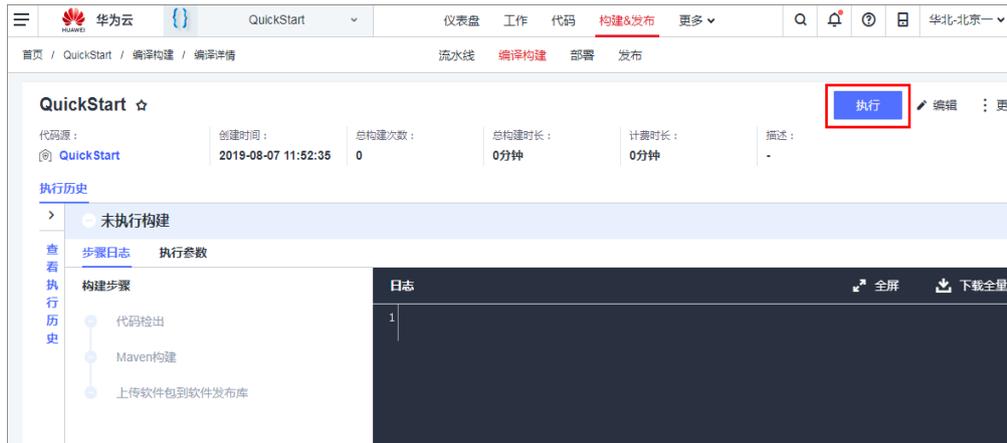
步骤11 在“构建步骤 > Maven构建”中，配置“步骤显示名称”和“工具版本”，此处以默认配置为例。



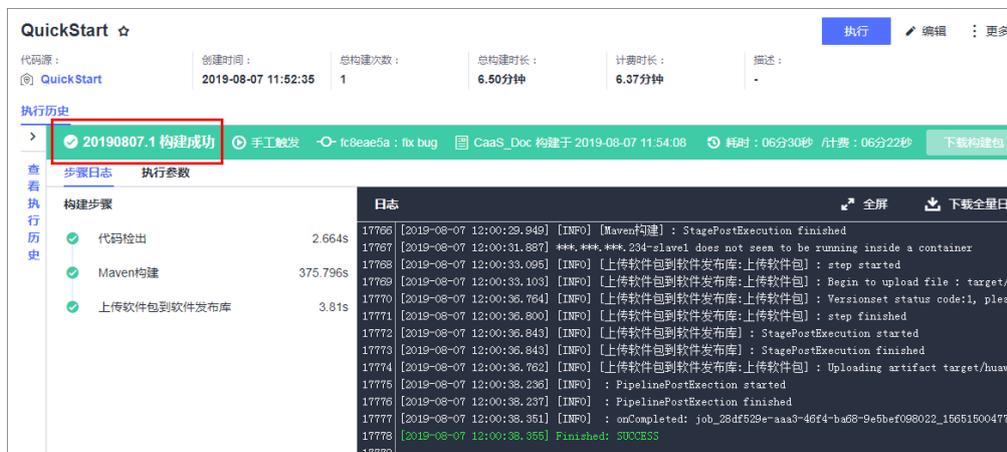
步骤12 在“构建步骤 > 上传软件包到软件发布库”中，配置各项参数，此处以默认配置为例，点击“新建”。



步骤13 点击“执行”。



等待工程完成构建。



步骤14 在工具栏选择“部署”，点击“新建任务”。



步骤15 在“新建部署任务 > 基本信息”中，配置任务名称，此处以默认配置为例，点击“下一步”。



步骤16 在“新建部署任务 > 选择部署模板”中，选择“SpringBoot应用部署”，点击“确定”。



步骤17 在“部署步骤 > 停止SpringBoot服务”中，配置各项参数：

📖 说明

如果是第一次部署该应用，则可以在“控制选项”中去勾选“启用”。

- 步骤显示名称：以默认值为例
- 主机组：如果无可用主机组，则点击“新建”完成创建主机组后，再进行选择。操作详见[参考：创建主机组](#)
- 服务操作类型：停止服务
- 服务对应的绝对路径：若之前已部署过该应用，填写该应用在主机上的实际路径，以“/home/huawei-0.0.1-SNAPSHOT.jar”为例



步骤18 在“部署步骤 > 安装JDK”中，配置各项参数：

说明

如果主机上已经安装JDK，则可以在“控制选项”中去勾选“启用”。

- 步骤显示名称：以默认值为例
- 主机组：如果无可用主机组，则点击“创建主机组”完成创建后，再进行选择。操作详见[参考：创建主机组](#)。
- jdk版本：根据需要选择jdk版本
- jdk安装路径：填写jdk在主机的安装路径，以“/usr/local/jdk”为例



步骤19 在“部署步骤 > 选择部署来源”中，配置各项参数：

- 步骤显示名称：以默认值为例
- 选择源类型：构建任务

说明

本指导基于在DevCloud构建的任务来进行部署。

- 请选择构建任务：选择已经在DevCloud构建的任务，以“QuickStart”为例
- 构建序号：配置已选择构建任务的序号，以“Latest”为例
- 下载到主机的部署目录：构建任务生成的软件部署到主机的路径，以“/home”为例



步骤20 在“部署步骤 > 启动SpringBoot服务”中，配置各项参数：

- 步骤显示名称：以默认值为例
- 主机组：如果无可用主机组，则点击“创建主机组”完成创建后，再进行选择。操作详见[参考：创建主机组](#)。
- 服务操作类型：启动服务
- 服务对应的绝对路径：由[步骤19](#)设置的部署目录（例如“/home”）加上[步骤12](#)设置的构建包名称（例如“huawei-0.0.1-SNAPSHOT.jar”）决定，例如“/home/huawei-0.0.1-SNAPSHOT.jar”
- 其余参数保持默认。

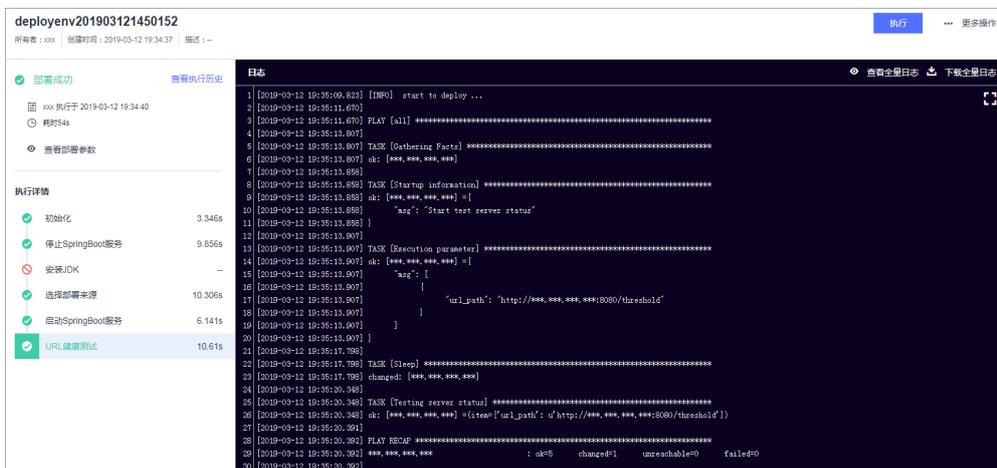


步骤21 在“部署步骤 > URL健康测试”中，配置各项参数，点击“保存并执行”。

- 步骤显示名称：以默认值为例
- 主机组：如果无可用主机组，则点击“创建主机组”完成创建后，再进行选择。操作详见[参考：创建主机组](#)。
- 等待时间：进行URL路径测试前的等待时间，以“3”为例
- 测试路径：填写测试服务状态的URL地址



步骤22 等待软件在主机部署成功。



----结束

调试应用

步骤1 为应用服务器订阅设备数据变化通知。

- 通过开发中心订阅
 - a. 登录开发中心，在对应项目中，选择“订阅调试”，并订阅设备数据变化。



- b. 输入“订阅地址”：http://IP:8080/receive-data，其中IP为**应用服务器的弹性IP**，点击“确定”。



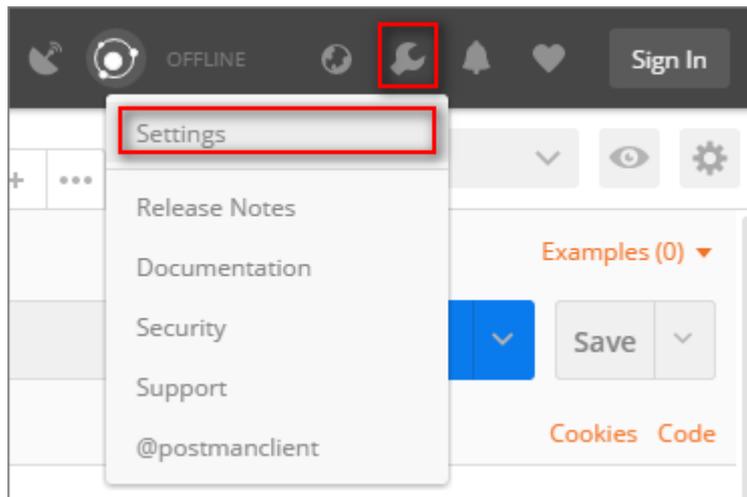
系统将对订阅地址的格式和连通性进行检测，并返回检查结果。



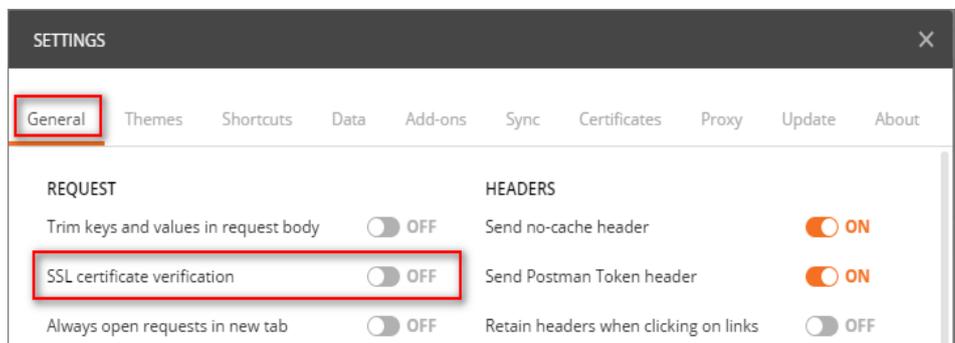
在“我的订阅”中，将显示新添加的订阅类型。



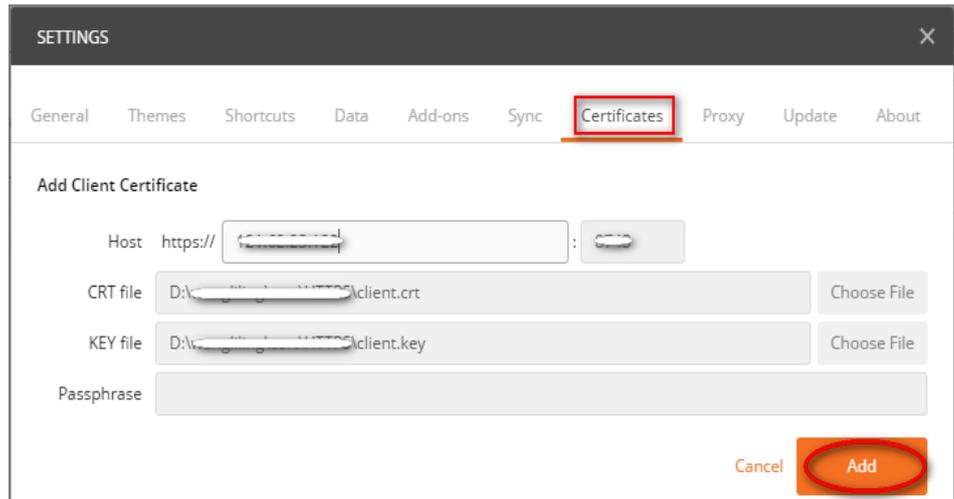
- 通过接口订阅（本示例中通过Postman调用接口进行订阅）[点击下载Postman](#)
 - 打开Postman的“Settings”菜单。



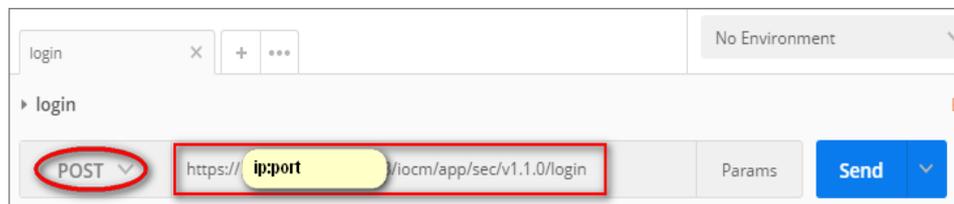
- 关闭证书校验，使Postman不再校验服务端的证书。



- 配置客户端证书，“Host”栏的地址和端口填写开发中心“应用 > 对接信息”中“应用接入信息”的IP与端口（HTTPS协议），CRT文件和KEY文件为证书文件，[点击此处获取](#)。

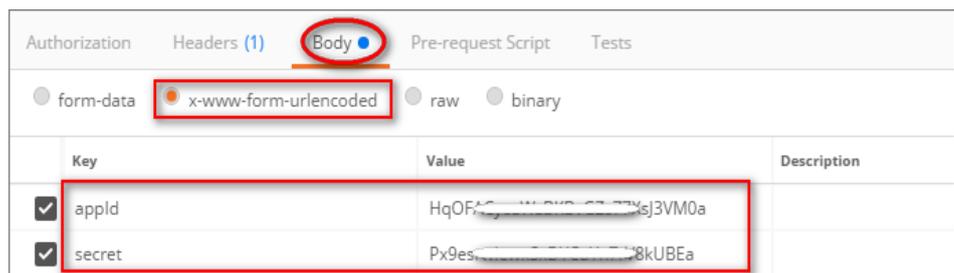


d. 调用“鉴权”接口，首先配置HTTP方法和URL。



参数	取值
HTTP方法	POST
URL	https://IP:port/iocm/app/sec/v1.1.0/login 其中IP和port为开发中心“应用 > 对接信息”中“应用接入信息”的IP与端口。

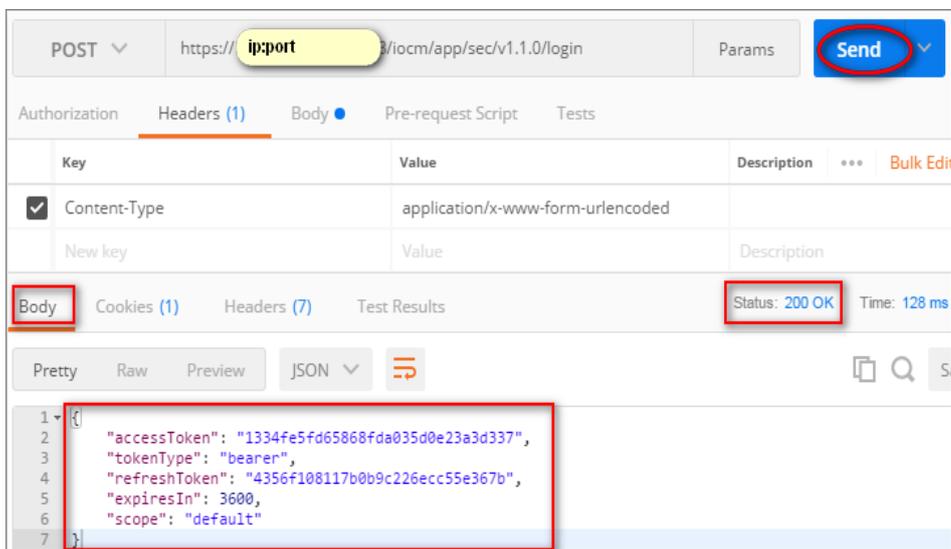
e. 然后配置Body。



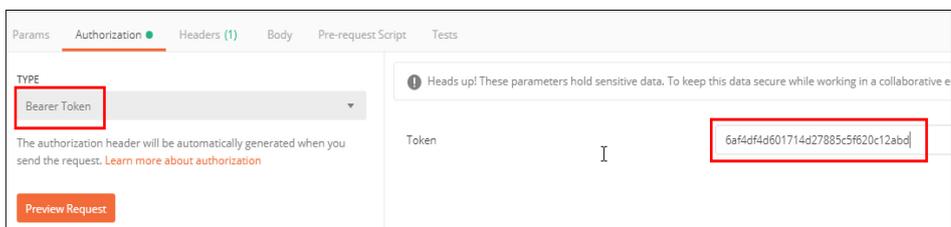
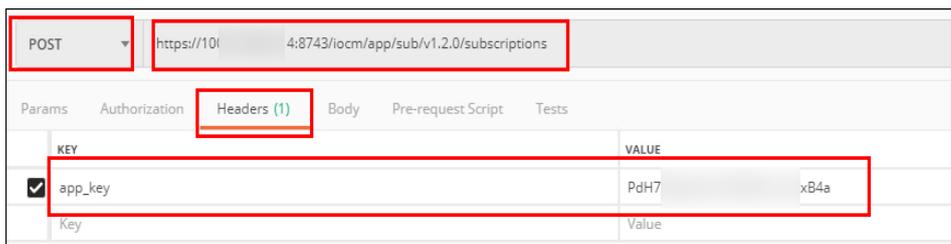
参数	取值
Content-Type	选择application/x-www-form-urlencoded。 注：选择Body格式后Postman会自动添加一个Content-Type头域。
appld	填写创建项目时获取的应用ID。
secret	填写创建项目时获取的应用密钥。

- f. 点击“Send”，在下方查看返回码和响应消息内容，响应码为200 OK表示鉴权成功。

请将返回的accessToken妥善保存，以便于在调用其它接口时使用。

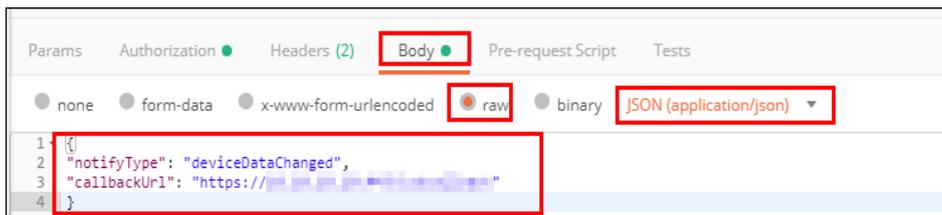


- g. 调用“订阅平台业务数据”接口，首先配置HTTP方法、URL和Headers。



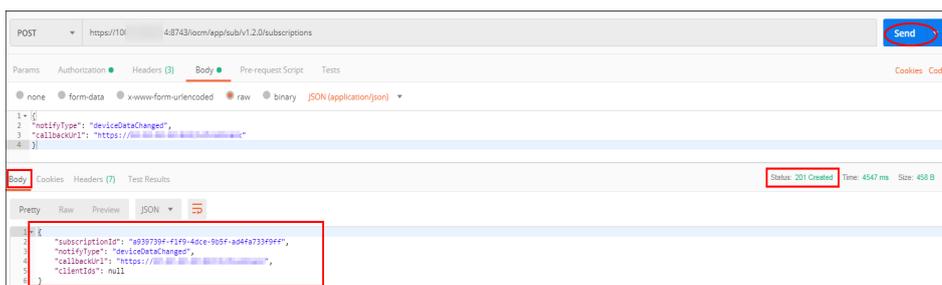
参数	取值
HTTP方法	POST
URL	https://IP:port/iocm/app/sub/v1.2.0/subscriptions 其中IP和port为开发中心“应用 > 对接信息”中“应用接入信息”的IP与端口。
app_key	填写创建项目时获取的应用ID。
Authorization	进入Authorization页签，“Type”选择“Bearer Token”，“Token”填写鉴权接口返回的accessToken。

- h. 然后配置Body。



参数	取值
Content-Type	选择raw之后选择JSON(application/json)。 注：选择Body格式后Postman会自动添加一个Content-Type头域。
notifyType	填写为“deviceDataChanged”。
callbackUrl	填写为“http://IP:8080/receive-data”，其中IP为应用服务器的弹性IP。

- i. 点击“Send”，在下方查看返回码和响应消息内容，响应码为201 Created表示订阅成功。



步骤2 访问应用系统：http://xxx.xxx.xxx.xxx:8080/index.html，xxx.xxx.xxx.xxx为应用系统的IP地址。

步骤3 在“参数设置”中，填写“应用信息”和“产品信息”后，点击“确定”。

- “应用信息”中的“应用ID”和“应用密钥”，在开发中心创建项目和产品后，由系统返回。
- “产品信息”中的“厂商ID”、“厂商名称”、“设备类型”等信息，需要与在开发中心创建项目和产品时设置的产品信息保持一致。

参数设置

应用信息

应用ID:

应用密钥:

产品信息

厂商ID:

厂商名称:

设备类型:

型号:

协议:

确定 取消

“产品信息”可以在开发中心创建的产品中查看。



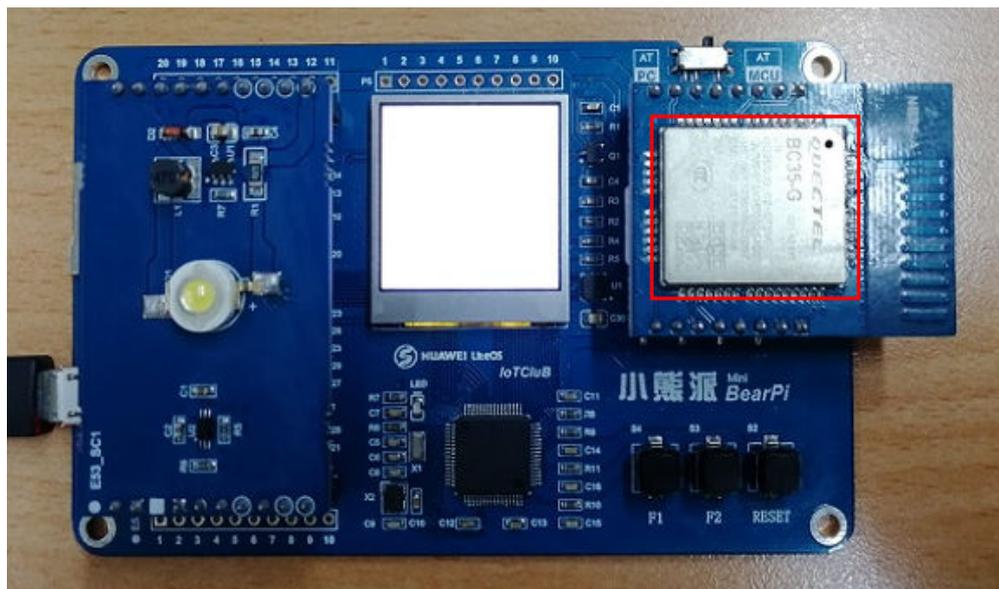
步骤4 在“属性参数”中，配置“设备标识”，点击“注册设备”。

属性参数

设备标识

注册设备 参数设置

“设备标识”填写开发板的IMEI号，可以在开发板上进行查看。



步骤5 在“属性参数”中，填写“亮度阈值”，点击“设置”。

亮度阈值请根据实际需要填写，当环境光强低于该值时，下发开灯命令；当环境光强高于该值时，下发关灯命令。此处以“10”为例。

属性参数

设备标识

亮度阈值

步骤6 进行如下测试：

1. 在应用系统的“环境亮度”区域，查看“亮度值”：“亮度值”应该随开发板所在环境不断变化。



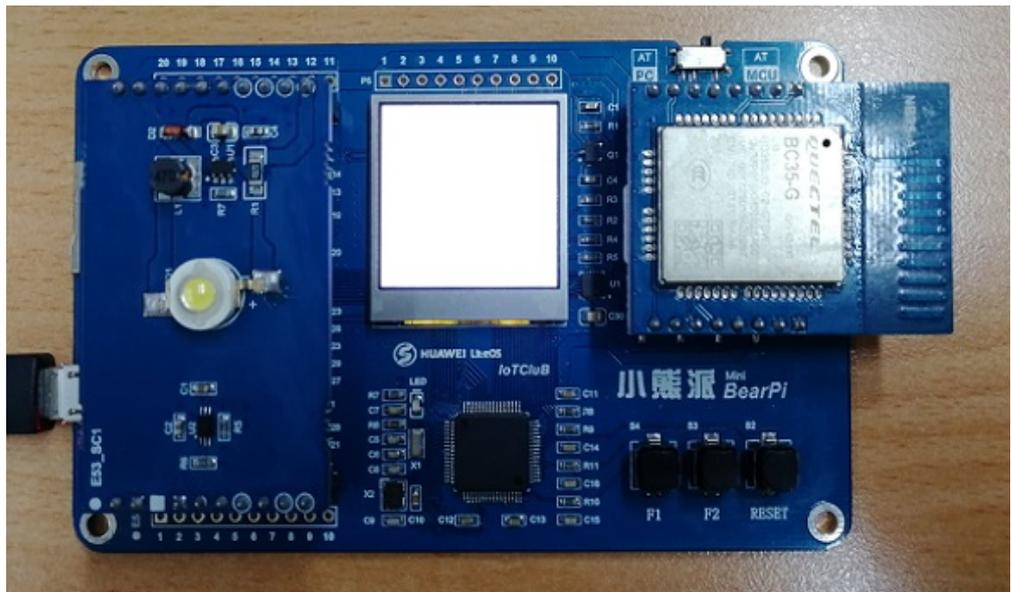
2. 遮住开发板，使开发板处于黑暗环境中（亮度<10），查看开发板的照明灯和应用系统的“灯状态”：开发板的照明灯应该打开；应用系统的“灯状态”应该切换为“开”。



状态查看

灯状态	开关灯次数	环境亮度
 <input checked="" type="radio"/> 开 <input type="radio"/> 关	统计总数 1 次	亮度值 0 LUX

3. 移除遮挡物，使开发板处于明亮环境中（亮度>10），查看开发板的照明灯和应用系统的“灯状态”：开发板的照明灯应该关闭；应用系统的“灯状态”应该切换为“关”。





----结束

参考：创建主机组

步骤1 点击“新建”。



步骤2 在新建主机组界面中配置主机组基本信息，点击“保存”。配置说明：

- 主机组名称：根据规划进行填写，以“iot”为例
- 操作系统：根据主机操作系统进行配置，以“linux”为例



步骤3 在新创建的主机组中，点击“添加主机”。



步骤4 在“添加主机”窗口中，配置主机信息，点击“添加”。各项参数请根据主机系统的实际信息进行配置。

如果没有主机或弹性IP，请参照界面提示购买虚拟机和弹性IP。



添加成功后，将在主机组中显示新添加的主机条目。

---结束

4 设备接入

基于小熊派开发智慧路灯（联通用户专用）

基于唯传LoRa网关和体验板接入物联网平台（联通用户专用）

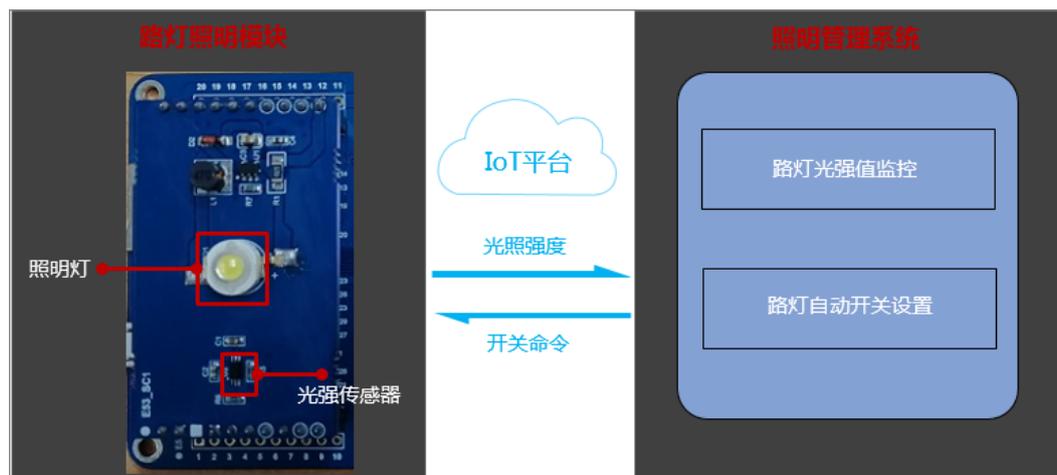
4.1 基于小熊派开发智慧路灯（联通用户专用）

非联通用户请查看[设备接入服务](#)。

场景说明

智慧路灯是城市智能化道路上重要一环，智慧路灯的实施具有节约公共照明能耗、减少因照明引起的交通事故等多种社会意义。路灯也是大家在日常生活中可以强烈感知到的公共设施，更易理解其智能化的场景。

在该文档中，我们基于华为一站式开发工具平台—开发中心，从设备、平台、应用端到端构建一款智慧路灯解决方案样例，带您体验十分钟快速上云。



方案设计

该款智慧路灯通过LWM2M协议接入物联网平台，具备如下能力：

- 支持上报光照强度。

- 支持开/关控制命令。
- 支持上报灯光状态信息。

该款智慧路灯的产品能力模型如下所示：

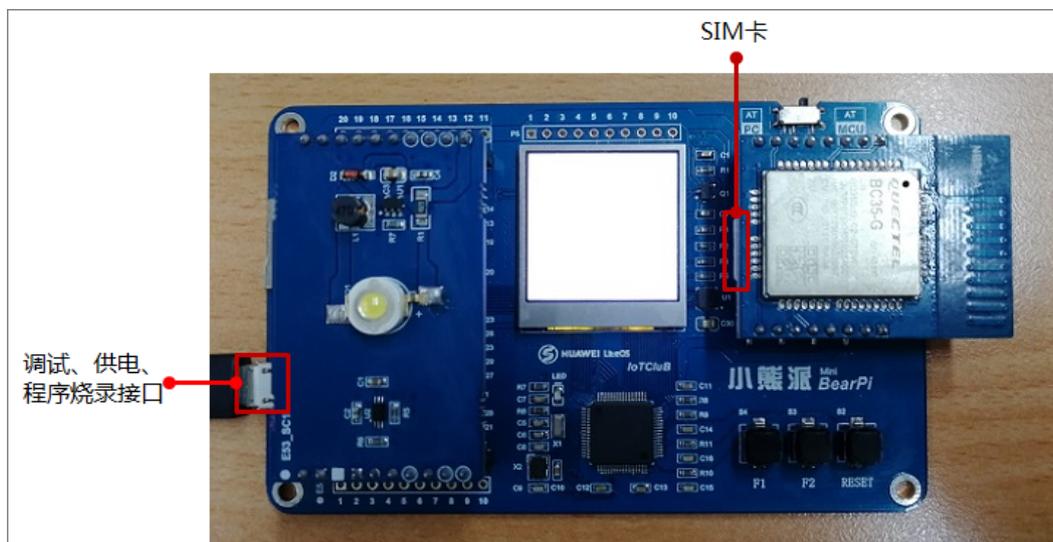
产品信息	设备类型	StreetLight	-
	设备型号	StreetLight01	-
	厂商名称	TestManuName	-
	接入协议	CoAP	-
	数据格式	二进制码流	-
服务数据	服务1	服务名称	Sensor
		属性	名称: luminance 数据类型: int 最小值: 0 最大值: 65535 步长: 1 单位: lux 访问模式: RE 是否必选: 是
	服务2	服务名称	LED
		命令	命令名称: Set_Led <ul style="list-style-type: none"> • 下发命令字段 名称: led 数据类型: string 长度: 3 枚举值: ON,OFF 是否必选: 是 • 响应命令字段 名称: light_state 数据类型: string 长度: 3 枚举值: ON,OFF 是否必选: 是

前期准备

- 开发板：小熊派开发板（含NB卡、NB模组、智慧路灯功能模块等）
- 配件：数据线
- IDE：IoT Studio

- 平台：华为云账号（开通[开发中心](#)权限）

硬件连接



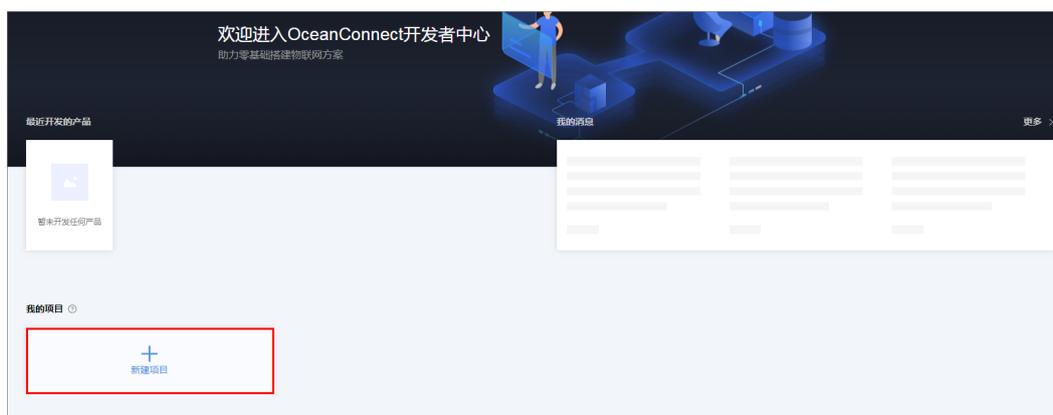
创建项目

在进行开发之前，开发者需要基于行业属性，创建一个独立的项目。在项目空间内，开发者可以开发相应的物联网产品和应用。

- 步骤1** 访问并登录华为云，打开“设备管理”服务产品首页，单击“开发中心”，进入物联网平台管理控制台，然后单击“进入开发中心”，自动跳转登录到开发中心。
- 步骤2** 在开发中心首页，单击“新建项目”。

📖 说明

如果初次登录开发中心，新建项目时会提示配置厂商信息。请根据提示，进入“厂商信息”界面进行配置即可。



- 步骤3** 填写“项目名称”、“所属行业”、“描述”等项目信息后，单击“创建”。

配置示例：

- 项目名称：StreetLight

- 所属行业：公用事业（NB-IoT）

新建项目 ×

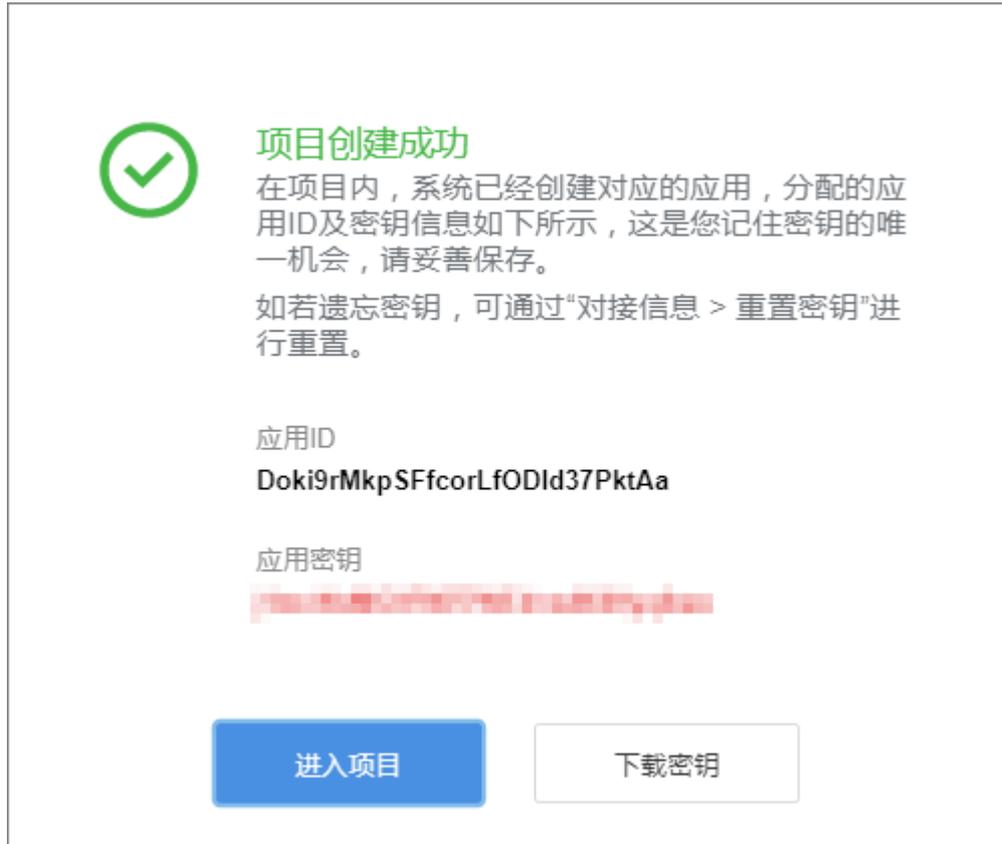
*项目名称

*所属行业

描述

确定

项目创建成功后，系统返回“应用ID”和“应用密钥”。在应用对接物联网平台时需要这两个参数，请妥善保存，如果遗忘，可以在该项目的“应用 > 对接信息 > 应用安全”中进行重置。

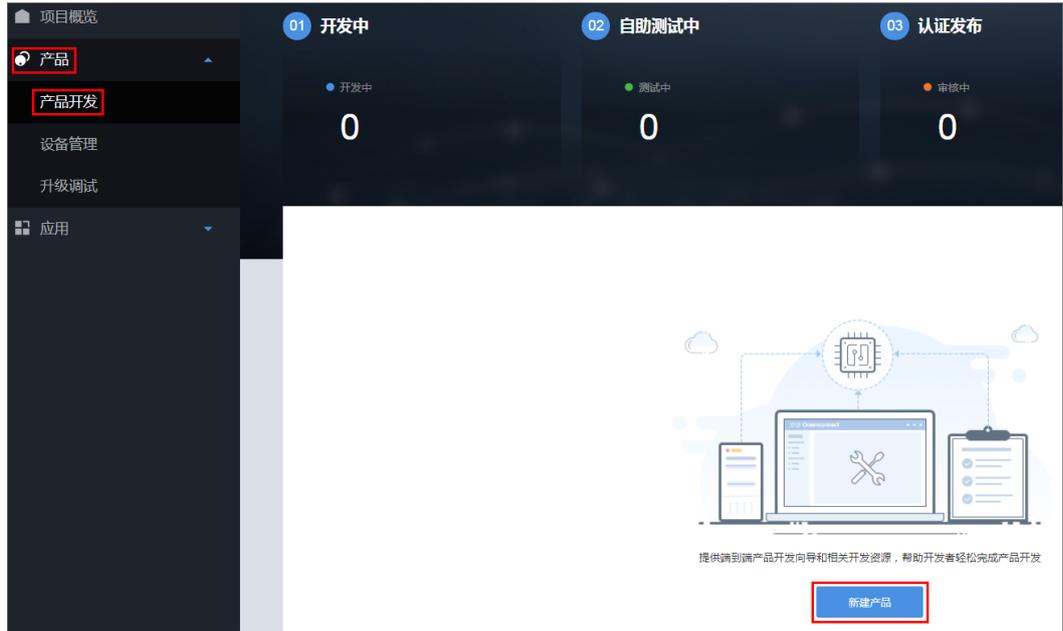


----结束

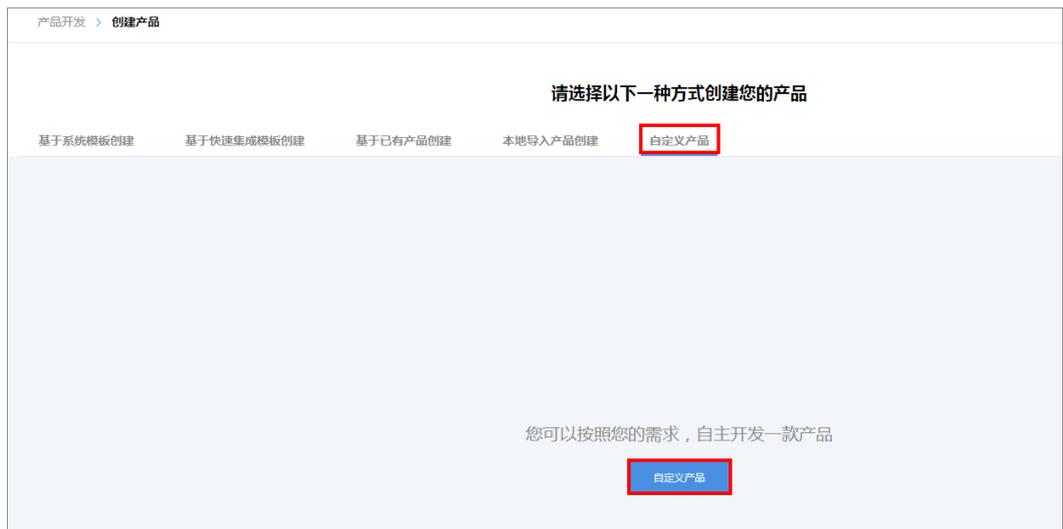
创建产品

某一类具有相同能力或特征的设备的集合称为一款产品。除了设备实体，产品还包含该类设备在物联网能力建设中的产品信息、产品模型（Profile）、插件、测试报告等资源。

步骤1 进入新创建的项目，选择“产品 > 产品开发”，单击“新建产品”。



步骤2 在“创建产品”中，选择“自定义产品”，单击“自定义产品”。



步骤3 系统将弹出“设置产品信息”窗口，填写必要产品信息，单击“创建”。

配置实例：

- 产品名称：OC_StreetLight
- 型号：OC_StreetLight001
- 厂商ID：由系统自动生成，无需配置
- 所属行业：智慧城市
- 设备类型：StreetLight
- 接入应用层协议类型：LWM2M
- 数据格式：二进制码流

设置产品信息 ? ×

* 产品名称:

* 产品型号:

* 厂商ID:

* 所属行业:

* 设备类型:

* 接入应用层协议类型 ?

注意: LWM2M协议的设备需要完善数据解析, 将设备上报的二进制数据转换为平台上的JSON数据格式

* 数据格式:

产品图片: 



步骤4 在“产品开发”界面将会呈现已经创建的产品，选择具体产品，可以进入该产品的开发界面。



----结束

Profile 定义

步骤1 在产品开发空间，选择“Profile定义”，单击“新建服务”，配置产品的服务。



步骤2 新建Sensor服务，管理路灯的光照强度。当路灯接入物联网平台后，上报光照强度值。



1. 在“属性列表”区域单击“添加属性”，填写相关信息，然后单击“确定”。

配置示例：

- 名称：luminance
- 数据类型：int
- 最小值：0
- 最大值：65535
- 步长：1
- 单位：lux
- 访问模式：RE
- 是否必选：是

新增属性 ×

* 名称

* 数据类型

* 最小值 * 最大值

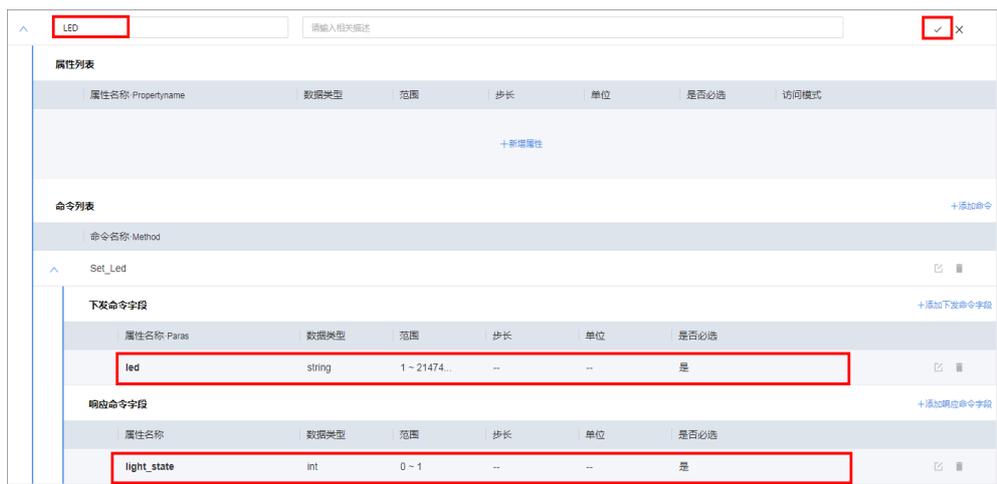
步长 单位

* 访问模式
 R 属性值可读
 W 属性值可写 (更改)
 E 属性值更改时上报事件

是否必选
 是

2. 在Sensor服务名的右边单击  ，保存服务。

步骤3 新建LED服务，管理路灯的开/关灯。当路灯收到物联网平台下发的开/关灯命令后，响应并上报当前的状态信息。



1. 在“命令列表”区域单击“添加命令”，命令名称为Set_Led，然后单击“确定”。
2. 单击“添加下发命令字段”，填写相关信息，然后单击“确定”。

配置示例：

- 名称：led
- 数据类型：string
- 长度：3
- 枚举值：ON,OFF
- 是否必选：是

新增下发命令字段 ×

* 名称

* 数据类型

* 长度

枚举值 (值之间以英文逗号分隔)

是否必选

是

- 单击“添加响应命令字段”，填写相关信息，然后单击“确定”。

配置示例：

- 名称：light_state
- 数据类型：string
- 长度：3
- 枚举值：ON,OFF
- 是否必选：是

新增响应命令字段

* 名称
light_state

* 数据类型
string

* 长度
3

枚举值 (值之间以英文逗号分隔)
ON,OFF

是否必选
 是

确定 取消

4. 在LED服务名的右边单击 ✓，保存服务。

----结束

编解码插件开发

步骤1 在产品开发空间，选择“编解码插件开发”，开发编解码插件。



步骤2 在“在线编解码插件编辑器”区域，单击“新增消息”。



步骤3 新增消息Report_Sensor，接收路灯上报的光照强度。

配置示例：

- 消息名：Report_Sensor
- 消息类型：数据上报
- 添加响应字段：是
- 响应数据：AAAA0000（默认）

The screenshot shows the '新增消息' (Add Message) configuration form. It has a '基本信息' (Basic Information) section with the following fields: '消息名' (Message Name) set to 'Report_Sensor', '消息描述' (Message Description) with a text area containing '输入描述', '消息类型' (Message Type) with '数据上报' (Data Report) selected, and a checked '添加响应字段' (Add Response Field) option. Below this is a '字段' (Fields) section with a '+ 添加字段' (Add Field) button. At the bottom, there is a '响应数据' (Response Data) field set to 'AAAA0000' and a note '不输入默认为AAAA0000' (If not input, default is AAAA0000). There are '完成' (Finish) and '取消' (Cancel) buttons at the bottom right.

1. 在“新增消息”界面，单击“添加字段”。
在“添加字段”界面，勾选“标记为地址域”，然后单击“完成”，添加地址域字段messageId。

添加字段 ×

标记为地址域 ?

*名字 只有标记为地址域时，名字固定为messageld；其他字段名字不能设置为messageld。

messageld

描述

输入字段描述

数据类型 (大端模式)

int8u(8位无符号整型) ▼

* 长度 ?

1

* 默认值 ?

0x0

偏移值 ?

0-1

完成 取消

2. 在“新增消息”界面，单击“添加字段”，填写相关信息，然后单击“完成”。

配置示例：

- 名字: data
- 数据类型: int16u (16位无符号整型)

添加字段 ×

标记为地址域 ?

***名字**

data

描述

输入字段描述

数据类型 (大端模式)

int16u(16位无符号整型)

*** 长度 ?**

2

默认值 ?

输入默认值

偏移值 ?

1-3

完成 取消

3. 在“新增消息”界面，单击“完成”，完成消息Report_Sensor的配置。

步骤4 新增消息Set_Led，管理路灯的开/关灯，并获取路灯执行命令后的状态信息。

配置示例：

- 消息名：Set_Led

- 消息类型：命令下发
- 添加响应字段：是

新增消息

基本信息

*消息名
Set_Led

消息描述
输入描述

*消息类型
 数据上报 命令下发

添加响应字段

字段
+ 添加字段

响应字段
+ 添加响应字段

完成 取消

1. 在“新增消息”界面，单击“添加字段”。
 - 在“添加字段”界面，勾选“标记为地址域”，然后单击“完成”，添加地址域字段messageId。
 - 在“添加字段”界面，勾选“标记为响应标识字段”，然后单击“完成”，添加响应标识字段mid。

添加字段 ✕

标记为地址域 ?

标记为响应标识字段 ?

*名字 只有标记为响应标识字段时，名字固定为mid；其他字段名字不能设置为mid。

mid

描述

输入字段描述

数据类型 (大端模式)

int16u(16位无符号整型) ▼

* 长度 ?

2

默认值 ?

输入默认值

偏移值 ?

1-3

完成

取消

2. 在“新增消息”界面，单击“添加字段”，填写相关信息，然后单击“完成”。
配置示例：
 - 名字：led
 - 数据类型：string
 - 长度：3

添加字段 ✕

标记为地址域 ?

标记为响应标识字段 ?

***名字**

描述

数据类型 (大端模式)

*** 长度 ?**

默认值 ?

偏移值 ?

完成

3. 在“新增消息”界面，单击“添加响应字段”。
 - 在“添加字段”界面，勾选“标记为地址域”，然后单击“完成”，添加地址域字段messageId。
 - 在“添加字段”界面，勾选“标记为响应标识字段”，然后单击“完成”，添加响应标识字段mid。

- 在“添加字段”界面，勾选“标记为命令执行状态字段”，然后单击“完成”，添加命令执行状态字段errcode。

添加字段 ✕

标记为地址域 ?

标记为响应标识字段 ?

标记为命令执行状态字段 ?

*名字 只有标记为命令执行状态字段时，名字固定为errcode；其他字段名字不能设置为errcode。

errcode

描述

输入字段描述

数据类型 (大端模式)

int8u(8位无符号整型) ▼

* 长度 ?

1

默认值 ?

0x3

偏移值 ?

1-2

完成 取消

4. 在“新增消息”界面，单击“添加响应字段”，填写相关信息，单击“完成”。
配置示例：
 - 名字: light_state
 - 数据类型: string
 - 长度: 3

添加字段 ×

标记为地址域 ?

标记为响应标识字段 ?

标记为命令执行状态字段 ?

***名字**

light_state

描述

输入字段描述

数据类型 (大端模式)

string(字符串类型) ▼

*** 长度 ?**

3

默认值 ?

输入默认值

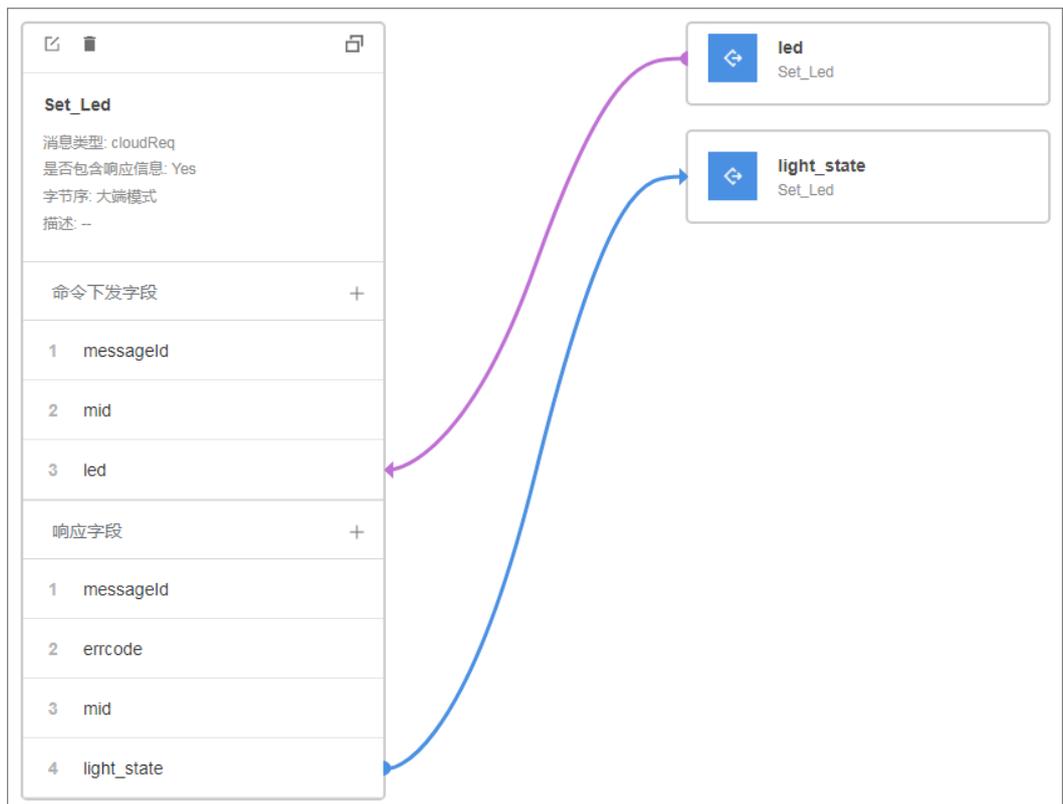
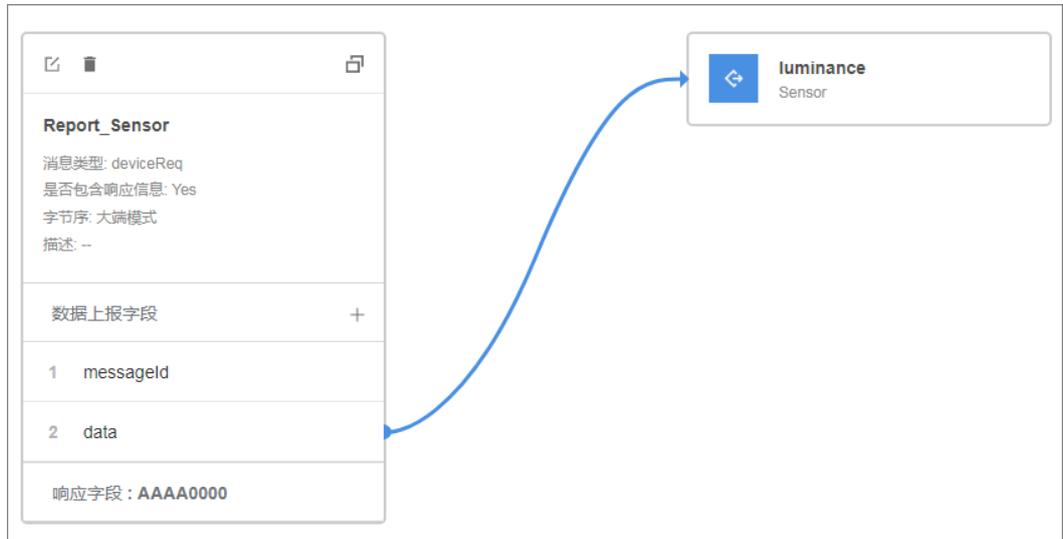
偏移值 ?

4-7

完成 取消

5. 在“新增消息”界面，单击“完成”，完成消息Set_Led的配置。

步骤5 拖动右侧“设备模型”区域的属性字段、命令字段和响应字段，与数据上报消息、命令下发消息和命令响应消息的相应字段建立映射关系。



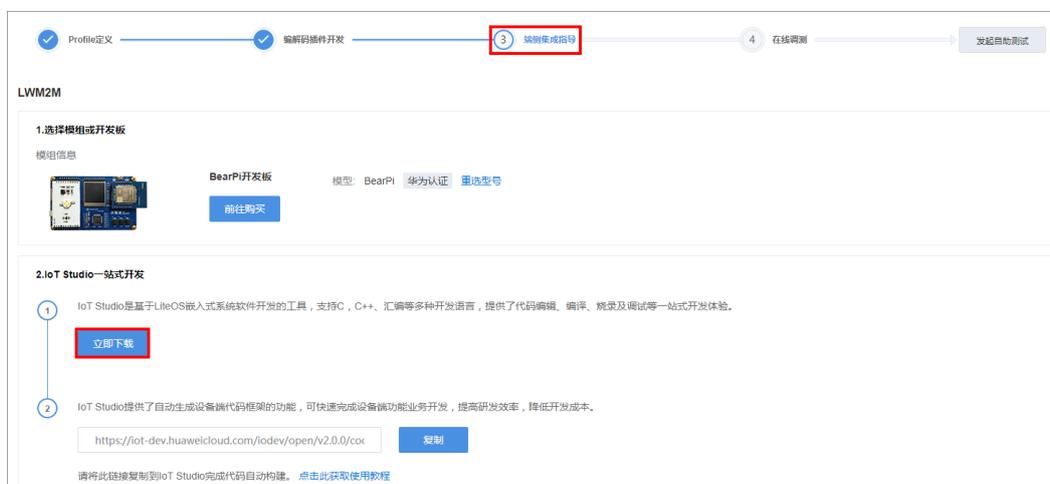
步骤6 单击“保存”，并在插件保存成功后单击“部署”，将编解码插件部署到物联网平台。



----结束

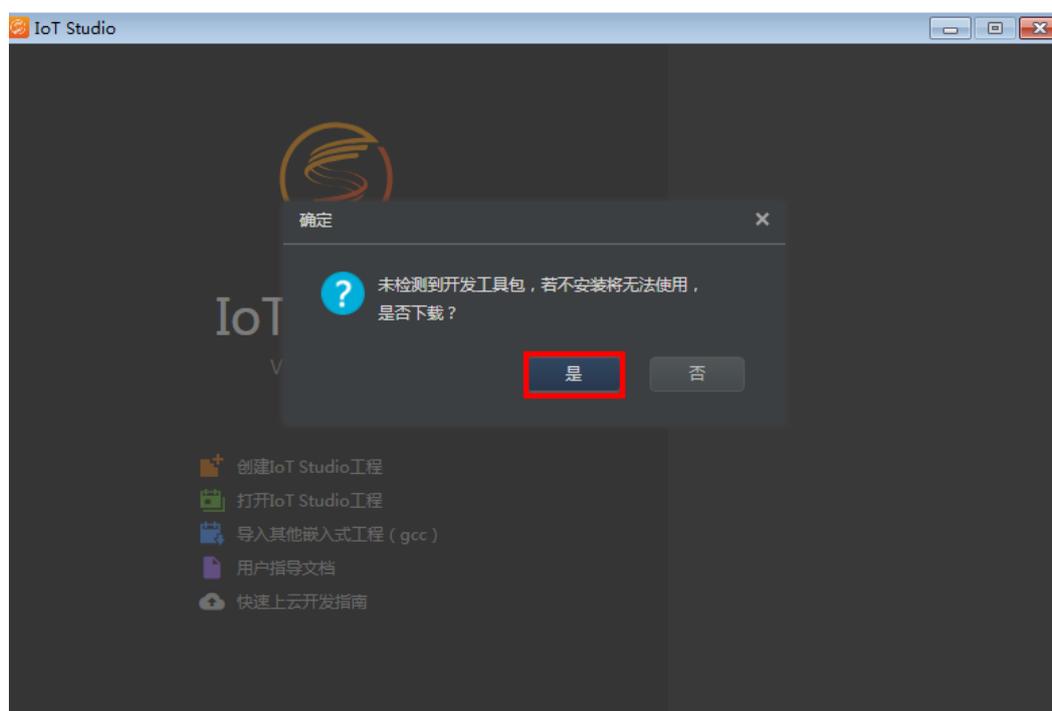
配置 IoT Studio 工程

步骤1 在产品开发空间，选择“端侧集成指导”，单击“立即下载”，下载并安装最新版的 IoT Studio。



步骤2 运行 IoT Studio 时，系统会检测是否安装开发工具包。

单击“是”，下载并安装开发工具包，此工具包用于在 windows 上搭建 IoT Studio 开发环境，编译调试 Lite OS 应用。

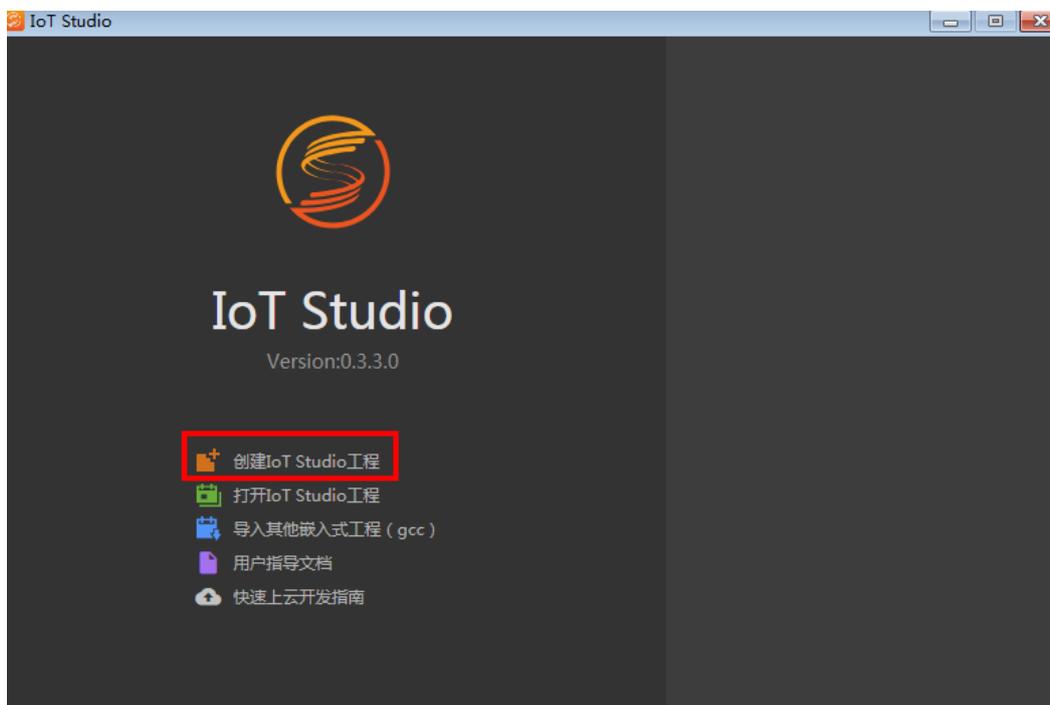


将工具包解压到任意目录，运行“developTools.exe”，安装完成后再重新运行 IoT Studio。

步骤3 在开发中心，选择“端侧集成指导”，单击“复制”，复制设备模型文件的 URL。



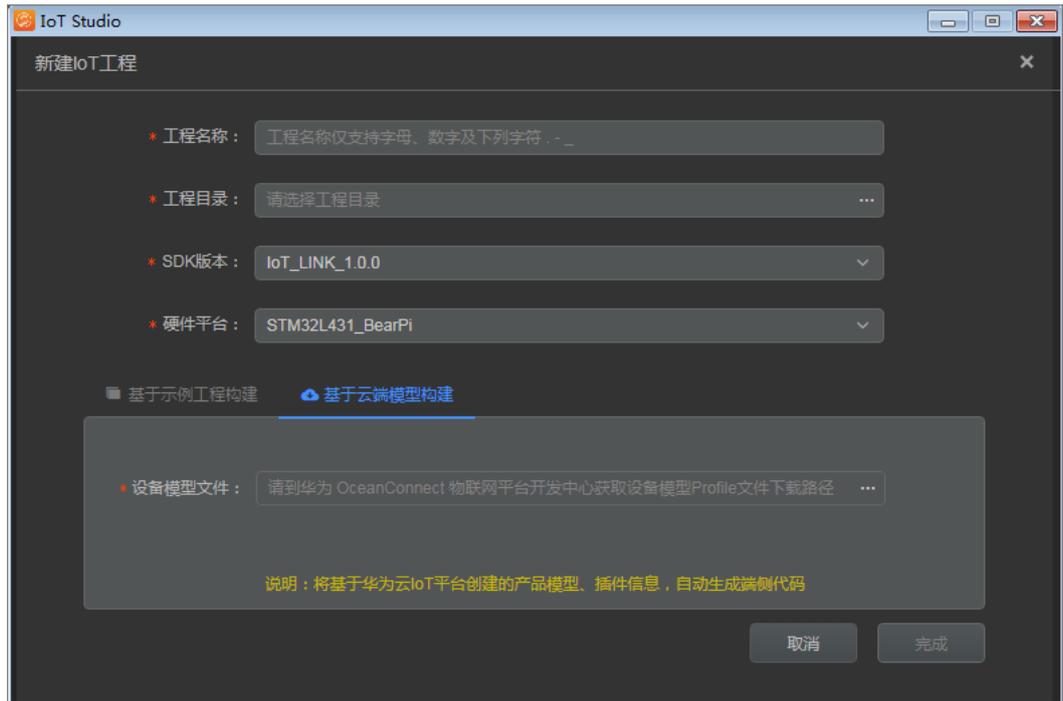
步骤4 打开IoT Studio，单击“创建IoT Studio工程”，进入“新建IoT工程”界面。



步骤5 在“新建IoT工程”中，填写工程相关信息。

配置示例：

- 工程名称：QuickStart
 - 工程目录：D:\
 - SDK版本：保持默认
 - 硬件平台：选择STM32L431_BearPi
 - 选择“基于云端模型创建”。
- 设备模型文件：步骤**步骤3**复制的设备模型文件的URL。



设备开发的业务模块主要集中在自动生成的`app.c`和`app.h`两个文件中。

- `oc_cloud_map_demo_app.c`文件中已自动生成了数据上报和命令下发两个接口。
- `oc_cloud_map_demo_app.h`文件主要描述映射生成的数据结构、硬件初始化接口。

----结束

实现设备业务处理

步骤1 实现命令下发处理。

1. 在`oc_cloud_map_demo_app.c`文件中找到`sensor_control_handler ()`函数。
2. 在 `/****** code area for cmd from IoT cloud *****/`注释下添加以下代码，

然后在工具栏中单击，保存文件。

```
//开灯
if (strstr(Set_Led->led, "ON") != NULL)
{
    HAL_GPIO_WritePin(Light_GPIO_Port, Light_Pin, GPIO_PIN_SET);
}
//关灯
if (strstr(Set_Led->led, "OFF") != NULL)
{
    HAL_GPIO_WritePin(Light_GPIO_Port, Light_Pin, GPIO_PIN_RESET);
}
```

```

56 {
57     int0_t msgid = s_rcv_buffer[0] & 0x000000FF;
58     switch (msgid)
59     {
60     #if 1
61     case cn_app_Set_Led:
62         Set_Led = (tag_app_Set_Led *)s_rcv_buffer;
63         printf("Set_Led:msgid:%d mid:%d", Set_Led->messageId, ntohs(Set_Led->mid));
64         /****** code area for cmd from IoT cloud *****/
65         //开灯
66         if (strstr(Set_Led->led, "ON") != NULL)
67         {
68             HAL_GPIO_WritePin(Light_GPIO_Port, Light_Pin, GPIO_PIN_SET);
69         }
70         //关灯
71         if (strstr(Set_Led->led, "OFF") != NULL)
72         {
73             HAL_GPIO_WritePin(Light_GPIO_Port, Light_Pin, GPIO_PIN_RESET);
74         }
75         /****** code area end *****/
76         break;
77     default:
78         break;
79     #endif
80     }

```

步骤2 实现数据上报处理。

1. 在 `oc_cloud_map_demo_app.c` 文件中找到 `report_data_handler ()` 函数。
2. 在 `/****** code area for report data to IoT cloud *****/` 注释下添加以下代码，并删除 “`// virtual sensor data example`” 注释下的代码，然后在工具栏中单击



保存文件。

```

Report_Sensor.messageId = cn_app_Report_Sensor;
Report_Sensor.data = htons(lux);
oc_lwm2m_report(context,(char *)&Report_Sensor, sizeof(Report_Sensor), 1000);

```

```

76         break;
77     default:
78         break;
79     #endif
80     }
81 }
82 }
83
84 int report_data_handler()
85 {
86     #if 1
87         tag_app_Report_Sensor Report_Sensor;
88     #endif
89
90     /****** code area for report data to IoT cloud *****/
91     Report_Sensor.messageId = cn_app_Report_Sensor;
92     Report_Sensor.data = htons(lux);
93     oc_lwm2m_report(context,(char *)&Report_Sensor, sizeof(Report_Sensor), 1000);
94     /****** code area end *****/
95
96     // virtual sensor data example

```

----结束

编译并烧录代码

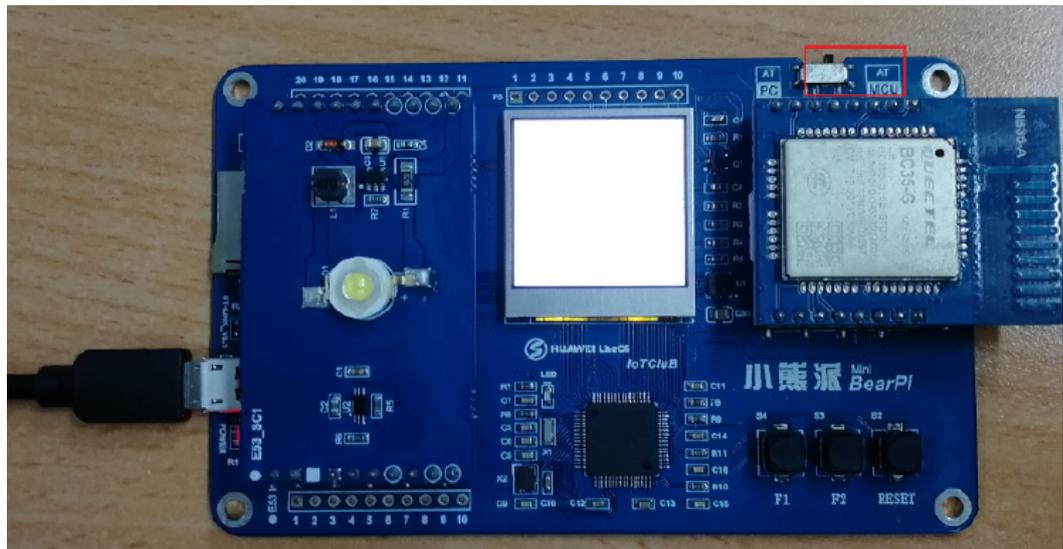
- 步骤1 在工具栏中单击 ，对当前工程进行编译。编译成功后，在控制台面板中显示“编译成功”。

```

控制台
[2019-09-10 14:39:21] 开始编译...
Compiling oc_cloud_map_demo_app.c...
D:/Bear/targets/STM32L431_BearPi/Demos/oc_cloud_map_demo/oc_cloud_map_demo_app.c:36:1: warning: /* within comment [-Wcomment] /*
In file included from D:/Bear/targets/STM32L431_BearPi/Demos/oc_cloud_map_demo/oc_cloud_map_demo_app.c:45:0: D:/Bear/targets/STM32L431_BearPi/Demos/oc_cloud_map_demo/oc_cloud_map
D:/Bear/targets/STM32L431_BearPi/Demos/oc_cloud_map_demo/oc_cloud_map_demo_app.h:185:9: warning: implicit declaration of function 'sensor_controll_handler' [-Wimplicit-function-declaration]
D:/Bear/targets/STM32L431_BearPi/Demos/oc_cloud_map_demo/oc_cloud_map_demo_app.h: In function 'app_report_task_entry': D:/Bear/targets/STM32L431_BearPi/Demos/oc_cloud_map_demo/oc_cld
function-declaration) report_data_handler(); ^~~~~~ D:/Bear/targets/STM32L431_BearPi/Demos/oc_cloud_map_demo/oc_cloud_map_demo_app.c: In function 'sensor_controll_handler'
unused variable 'Response_Set_LED' [-Wunused-variable] tag app Response_Set_LED Response_Set_LED; ^~~~~~ D:/Bear/targets/STM32L431_BearPi/Demos/oc_cloud_map_demo/oc_
D:/Bear/targets/STM32L431_BearPi/Demos/oc_cloud_map_demo/oc_cloud_map_demo_app.c: In function 'report_data_handler': D:/Bear/targets/STM32L431_BearPi/Demos/oc_cloud_map_demo/oc_cloud
file included from D:/Bear/targets/STM32L431_BearPi/Demos/oc_cloud_map_demo/oc_cloud_map_demo_app.c:45:0: At top level: D:/Bear/targets/STM32L431_BearPi/Demos/oc_cloud_map_demo/oc
void get_netstats() ^~~~~~

arm-none-eabi-size appbuild/Huawei_LiteOS.elf
text data bss dec hex filename 201800 536 8248 210584 33698 appbuild/Huawei_LiteOS.elf
arm-none-eabi-objcopy -O ihex appbuild/Huawei_LiteOS.elf appbuild/Huawei_LiteOS.hex arm-none-eabi-objcopy -O binary -S appbuild/Huawei_LiteOS.elf appbuild/Huawei_LiteOS.bin
0 Error(s), 8 Warning(s)
[2019-09-10 14:39:27] 编译成功。 编译耗时: 3874ms
    
```

步骤2 使用数据线将开发板与电脑连接，并将通信模组上的开关拨至MCU侧。



步骤3 在工具栏中单击 ，将已经编译的程序烧录至开发板。烧录成功后，控制台面板中显示“烧录成功”。

```

控制台

xPSR: 0x61000000 pc: 0x2000002e msp: 00000000
verified 78776 bytes in 2.324094s (33.101 KiB/s)
Info : Unable to match requested speed 500 kHz, using 480 kHz
Info : Unable to match requested speed 500 kHz, using 480 kHz
adapter speed: 480 kHz
shutdown command invoked
[2019-07-17 15:04:41] 烧录成功!
    
```

----结束

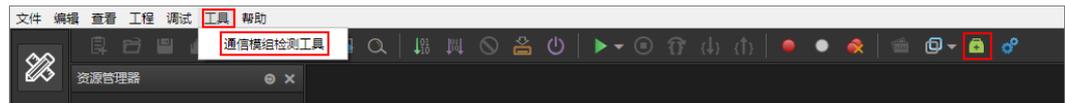
设备调测

请参考[基于IoT Studio构建应用服务器（联通用户专用）](#)或[基于软件开发平台构建应用服务器（联通用户专用）](#)完成应用服务器的构建并进行调测。

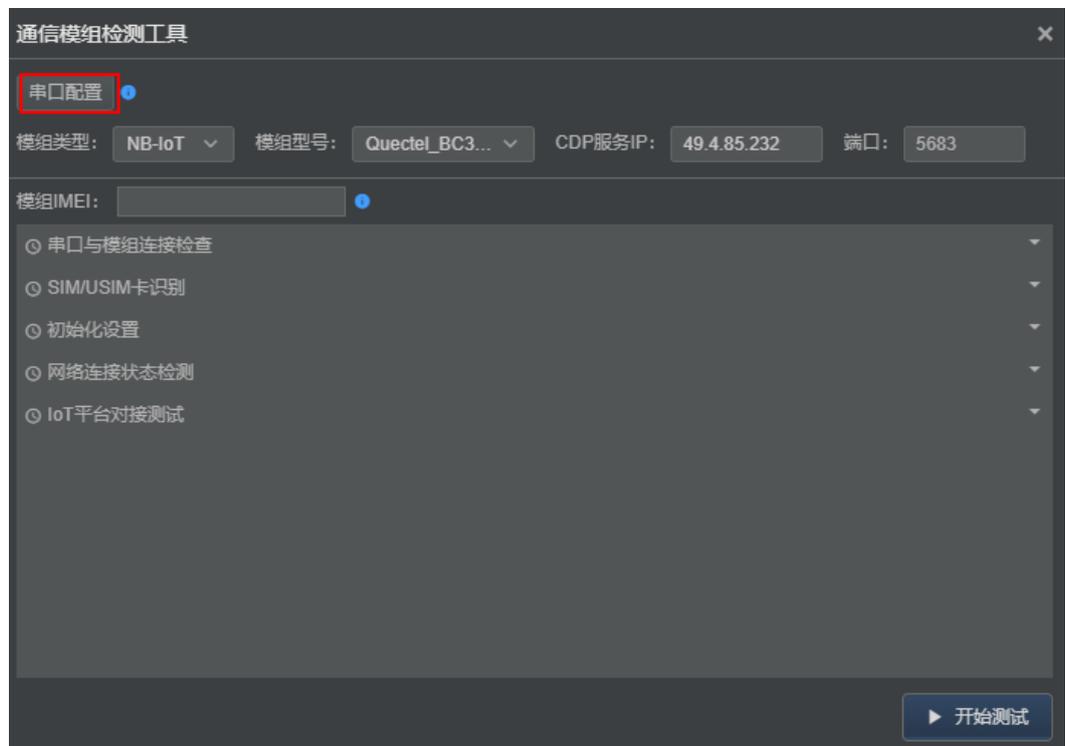
使用工具定位模组通信问题

IoT Studio在与物联网平台连通使用时，可使用通信模组检测工具快速定位 Quectel_BC35-G&BC28&BC95模组与云端连通性问题，提高开发效率。本节将以“小熊派开发板”为例，介绍如何使用通信模组检测工具定位常见问题，如设备无法上线，数据上报不成功等。

步骤1 单击菜单栏中的“工具 > 通信模组检测工具”或直接单击工具栏中的图标，打开通信模组检测工具。



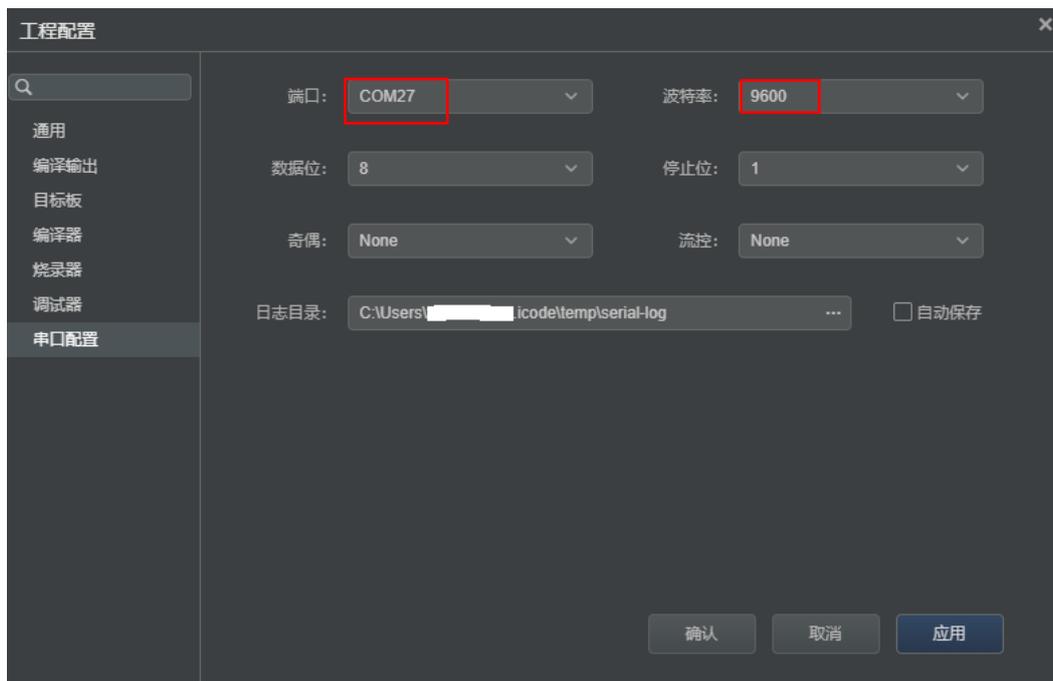
步骤2 单击“串口配置”。



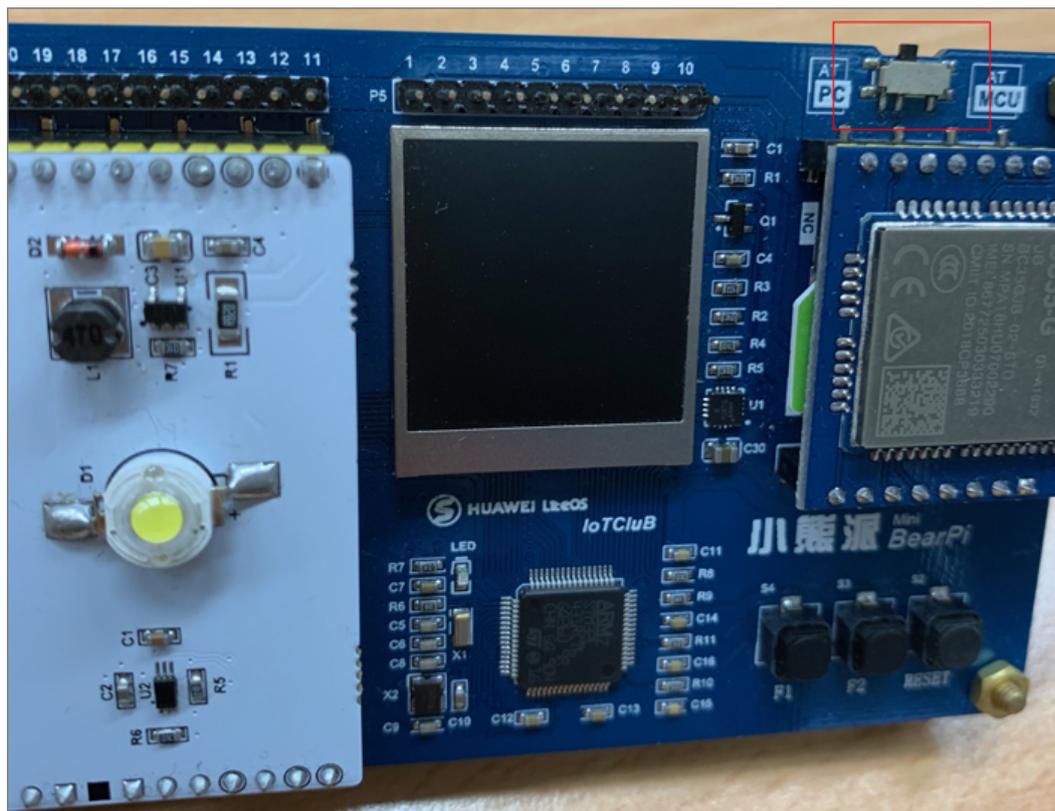
步骤3 获取PC与开发板连接的实际端口号。选择“控制面板 > 硬件和声音 > 设备管理器”，在“端口”子菜单下，找到连接设备的串口号，如COM27。



步骤4 在“串口配置”界面，“端口”选择**步骤3**获取的与开发板连接的端口号，如“COM27”；“波特率”配置为“9600”，然后单击“应用”。



步骤5 将与PC连接的通信模组上的开关拨至PC侧。



步骤6 单击“开始测试”，若通信正常，测试结果如下。

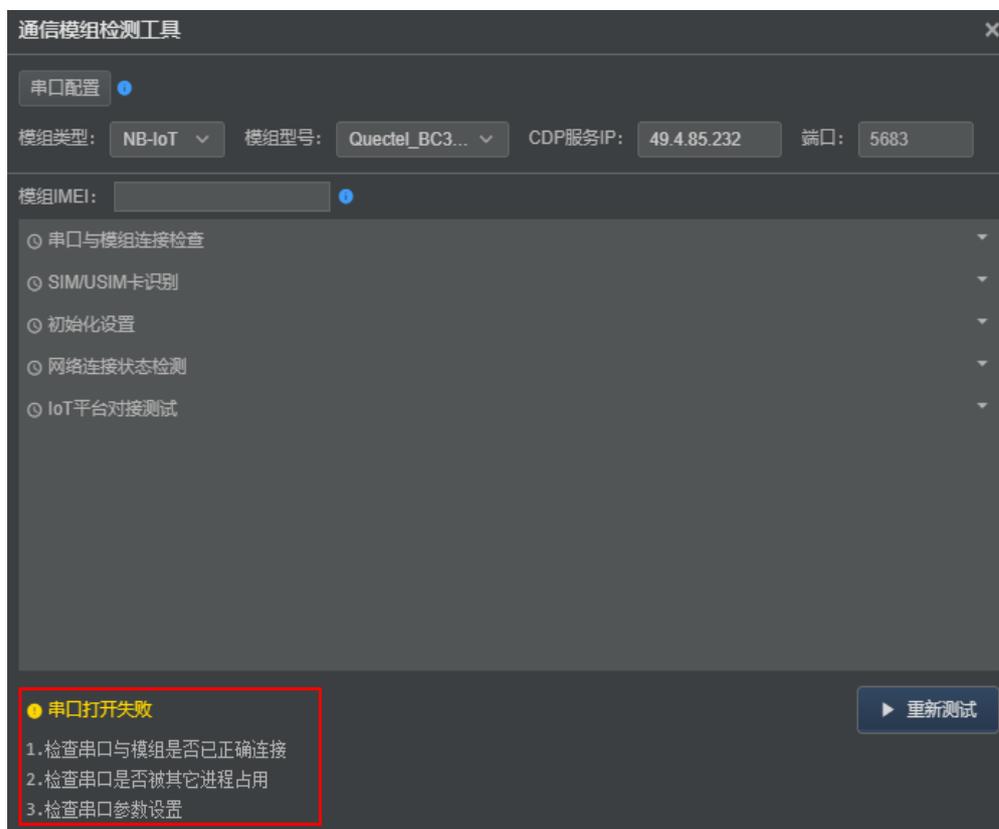


----结束

模组与物联网平台通信异常场景：

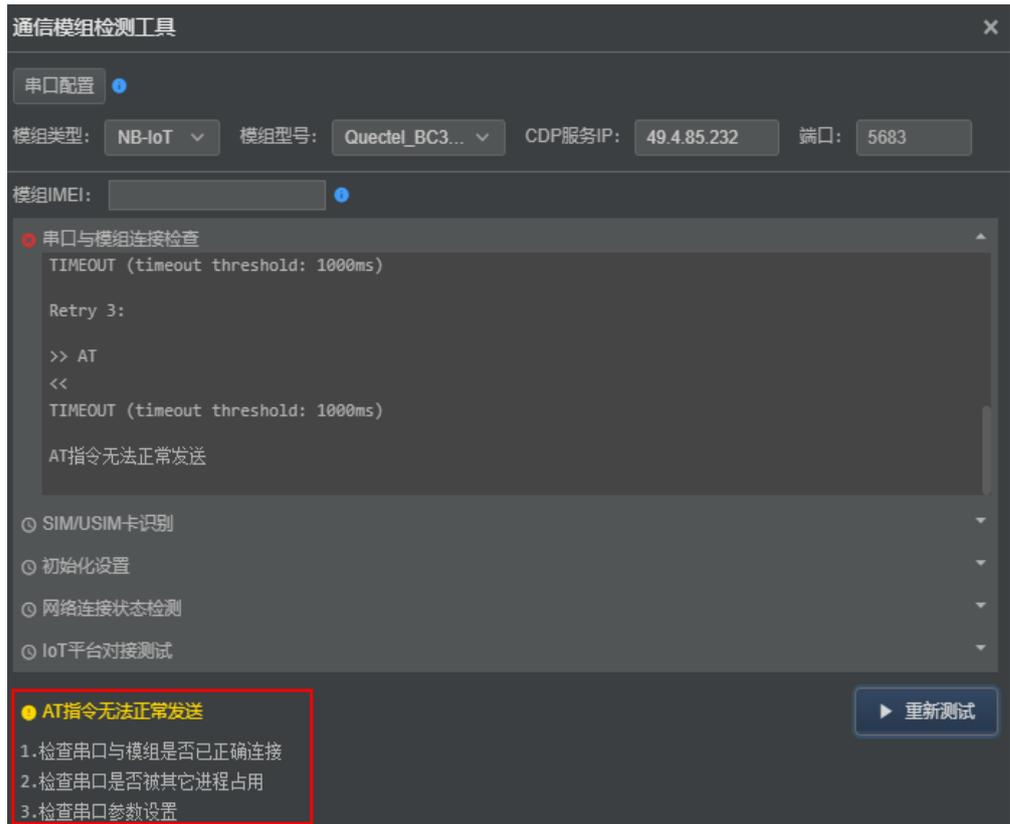
- **串口打开失败**

测试结果为“串口打开失败”，请检查串口配置是否正确，即端口号是否为实际端口号，波特率是否设置为“9600”。



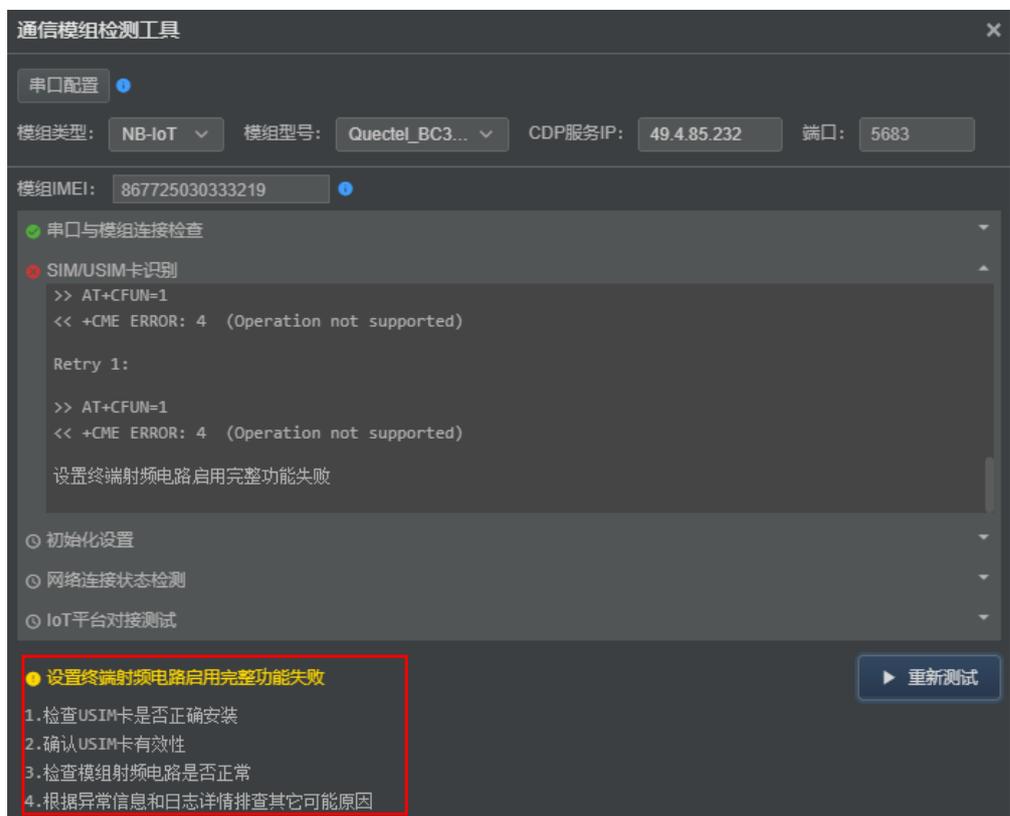
- **模组连接异常/模组损坏**

测试结果为“AT指令无法正常发送”，请检查开发板上的开关是否已拨至PC侧、模组是否损坏、以及模组是否正确的插入卡槽。



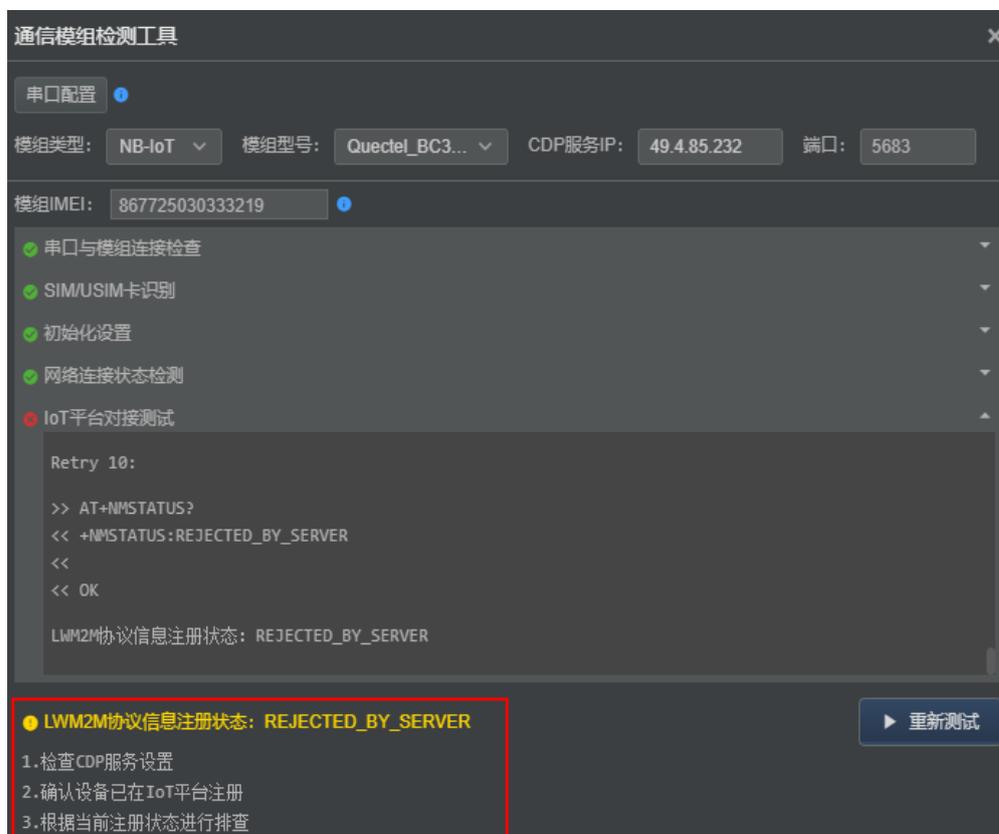
- **SIM卡未正确插入卡槽**

测试结果为“设置终端射频电路启用完整功能失败”，请检查开发板上的SIM卡是否正反面或者方向插错，以及SIM卡的有效性。



- **模组未在物联网平台注册**

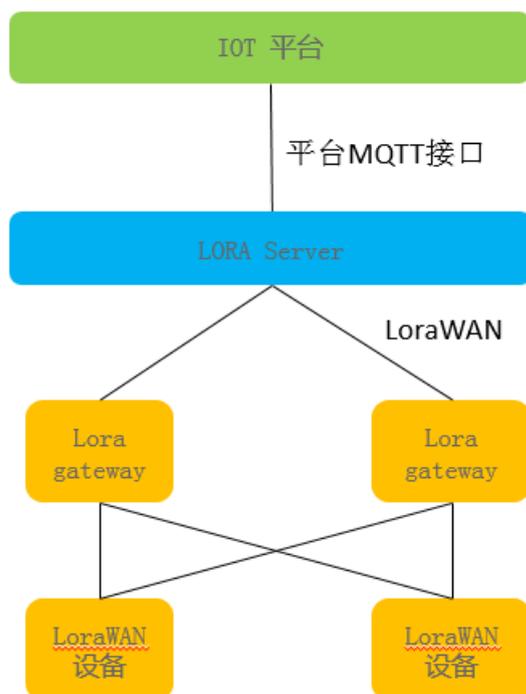
测试结果为“LWM2M协议信息注册状态：REJECTED_BY_SERVER”，请在物联网平台注册该模组。



4.2 基于唯传 LoRa 网关和体验板接入物联网平台（联通用户专用）

场景说明

LoRa（Long Range Radio）是当前应用较广的一种物联网无线接入技术，它最大的特点就是在同样功耗条件下比其他无线接入方式传播的距离更远，实现了低功耗和远距离的统一，它在同样的功耗下比传统的无线射频通信距离扩大3-5倍。而LoRaWAN是为LoRa远距离通信网络设计的一套通讯协议和系统架构。



本示例为通过登录[唯传科技官网](#)，购买LoRa网关和体验板，接入物联网平台，实现数据上报和命令下发。

对接 LoRa Server

在设备管理服务中创建LoRa Server的产品模型并注册LoRa Server设备。

1. 登录[物联网平台控制台](#)，点击右上角“进入设备管理服务”。
2. 单击页面左下角的  切换左侧菜单，选择“产品模型”，单击“新增产品模型”，选择“手动创建”。
3. 在弹框页面里填写关键参数信息后，单击“确定”。

手动创建

选择手动新增方式时，设备需要向平台上报设备能力，平台基于所上报的能力进行数据上报分析、命令下发等功能。

* 产品名称

* 型号
支持英文大小写、数字。下划线和短划线，4~20个字符

* 设备类型

* 厂商名称

协议类型 CoAP MQTT LoRaWAN

确定

取消

参数名称	说明
产品名称	自定义，如“wxLoraServer”。
型号	自定义，如“TestModel”
设备类型	固定配置为“loraServer”
厂商名称	自定义，如“TestManuName”
协议类型	选择“MQTT”

4. 在设备管理服务中选择“设备 > 设备注册”。
5. 选择页签“单个注册”，单击“创建”，填写关键参数后，单击“确定”。

单设备注册

*产品模型

*设备识别码

设备ID

密钥

确认密钥

参数名称	说明
产品模型	选择3中创建的LoRa Server的产品模型，如“wxLoraServer”
设备标识码	该参数实际未使用，可自定义设置，如“test_lora”
设备ID	由系统自动生成，无需配置
密钥	可自定义设置，如“12345678”
确认密钥	可自定义设置，如“12345678”

- 注册设备成功后，将平台返回的“设备ID”和“密钥”保存，通过线下的方式将设备id和密钥提供给唯传，唯传对LoRa Server进行配置。

注册设备成功



注册设备成功，请保存好以下设备密钥，如果丢失请查看设备设置并“重置”密钥。

设备ID 5de25b653ac49d03f5e32f94_test_lora

密钥: 12345678

保存至本地

您可以在“所有设备”列表查看详细的设备注册信息。注册设备成功后，下一步请接入实际的设备。

知道了

7. 在设备管理服务中选择“设备 > 所有设备”，确认上一步注册的LoRa Server设备状态为“在线”，则LoRa Server对接成功。



创建 LoRa 网关

在设备管理服务中创建LoRa的产品模型，并以子设备的方式在LoRa Server下创建LoRa网关。

1. 登录[物联网平台控制台](#)，点击右上角“进入设备管理服务”。
2. 单击页面左下角的 切换左侧菜单，选择“产品模型”，单击“新增产品模型”，选择“手动创建”。
3. 在弹框页面里填写关键参数信息后，单击“确定”。

手动创建

选择手动新增方式时，设备需要向平台上报设备能力，平台基于所上报的能力进行数据上报分析、命令下发等功能。

* 产品名称

* 型号

* 设备类型

* 厂商名称

支持中文、英文大小写、数字、下划线和短划线，不超过20个字符

协议类型 CoAP MQTT LoRaWAN

确定

取消

参数名称	说明
产品名称	自定义，如“loraGateway”
型号	自定义，如“TestGatewayModel”
设备类型	固定配置为“LoRaGateway”
厂商名称	自定义，如“TestGatewayManuName”
协议类型	选择“LoRaWAN”

4. 在设备管理服务中选择“设备 > 所有设备”，单击设备标识码为“test_lora”的设备进入“设备详情”页。
5. 选择“子设备”页签，单击右上角的“添加”按钮添加子设备。
6. 在弹框页面里填写关键参数后，单击“确定”。

添加子设备



* 产品模型

loraGateway

* 设备名称

test_lora_gateway

* 设备标识码(GatewayEUI)



接入码



确认

取消

参数名称	说明
选择产品	选择3中创建的LoRa网关的产品模型，如“loraGateway”
设备名称	自定义，如“test_lora_gateway”
设备标识码 (GatewayEUI)	填写LoRa网关的gatewayEUI，从LoRa网关上获取。
接入码	在LoRa Server上注册LoRa网关使用的接入码，由唯传科技提供。

📖 说明

新注册的LoRa网关处于未激活状态。如果LoRa网关是MQTT协议，接入网络2分钟后，LoRa网关变为在线状态；如果LoRa网关是UDP协议，则一直处于未激活状态。

创建 LoRaWAN 设备

在设备管理服务中创建LoRaWAN的产品模型，以子设备的方式在LoRa Server下创建LoRaWAN设备。

1. 登录[物联网平台控制台](#)，点击右上角“进入设备管理服务”。
2. 单击页面左下角的 切换左侧菜单，选择“产品模型”，单击“新增产品模型”，选择“手动创建”。
3. 在弹框页面里填写关键参数后，单击“确定”。

手动创建

选择手动新增方式时，设备需要向平台上报设备能力，平台基于所上报的能力进行数据上报分析、命令下发等功能。

*产品名称

*型号

*设备类型

*厂商名称

支持中文、英文大小写、数字、下划线和短划线，不超过20个字符

协议类型 CoAP MQTT LoRaWAN

确定

取消

参数名称	说明
产品名称	自定义，如“loraNode”
型号	自定义，如“TestNodeModel”
设备类型	非LoRaGateway的其他类型
厂商名称	自定义，如“TestNodeManuName”
协议类型	选择“LoRaWAN”

4. 在设备管理服务中选择“设备 > 所有设备”，单击已注册的LoRa Server设备进入“设备详情”页。
5. 选择“子设备”页签，单击右上角的“添加”按钮添加子设备。

6. 在弹框页面里填写关键参数后，单击“确定”。

添加子设备



* 产品模型

loraNode

* 设备名称

test_lora_node

* 设备标识码(DevEUI)



接入码



设备工作模式

ClassA ClassB ClassC

* AppEUI



入网模式

OTAA

* AppKey



确认

取消

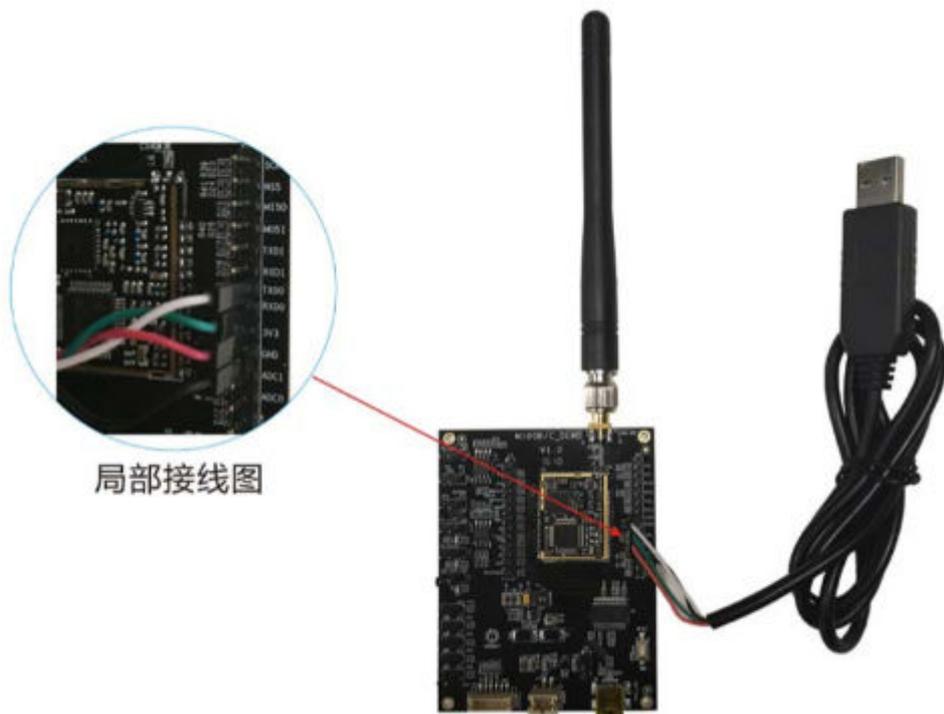
参数名称	说明
产品模型	选择3中创建的产品名称，如“loraNode”。
设备名称	自定义，如“test_lora_node”。
设备标识码 (DevEUI)	填写LoRaWAN设备的devEUI，在体验板上获取。
接入码	在LoRa Server上注册LoRaWAN设备使用的接入码，由唯传科技提供。
设备工作模式	选择classA。
appEUI	购买loRaWAN设备时提供的产品参数单上获取。
激活方式	选择OTAA。
AppKey	购买loRaWAN设备时提供的产品参数单上获取。

 说明

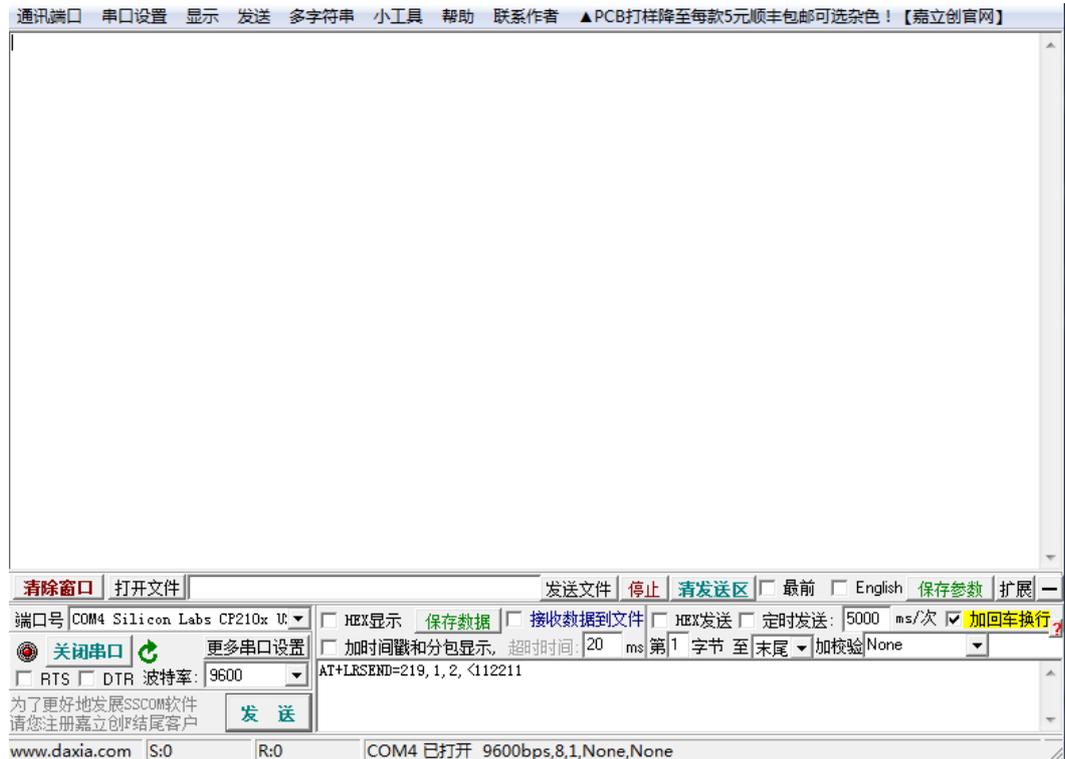
新注册的设备处于未激活状态，当上报数据后变为在线状态。

调试应用

步骤1 将体验板连接到电脑上，并且放在LoRa网关附近。接线方式如下图所示，串口线的白、绿、红、黑依次对应串口 TX、RX、电源，地线对应体验板上的TXD0、RXD0、3V3 以及 GND。

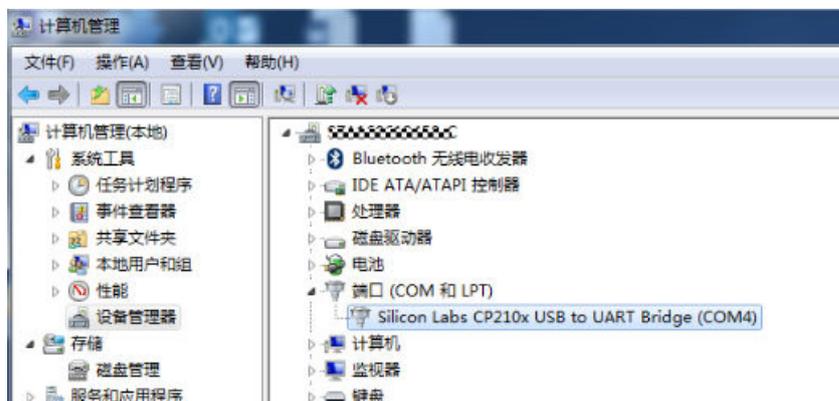


步骤2 访问[sscom](#)，下载最新版本的串口调试工具。



步骤3 获取PC与开发板连接的实际端口号。

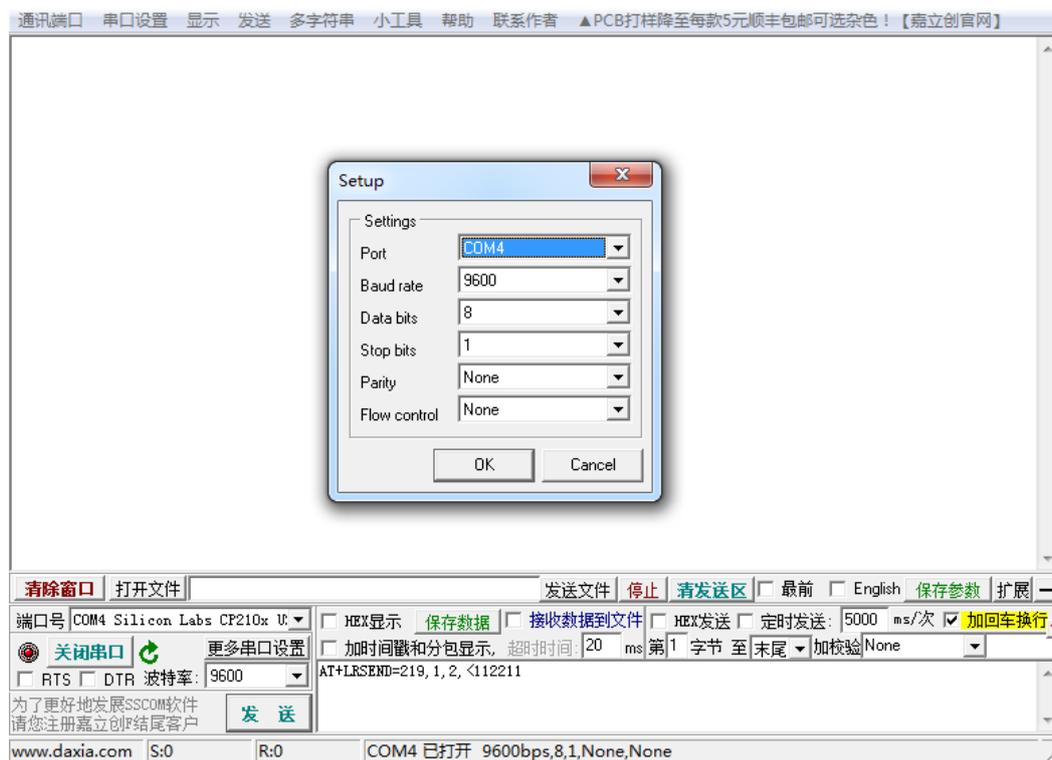
1. 选择“控制面板 > 硬件和声音 > 设备管理器”
2. 在“端口”子菜单下，找到连接设备的串口号，如COM4。



说明

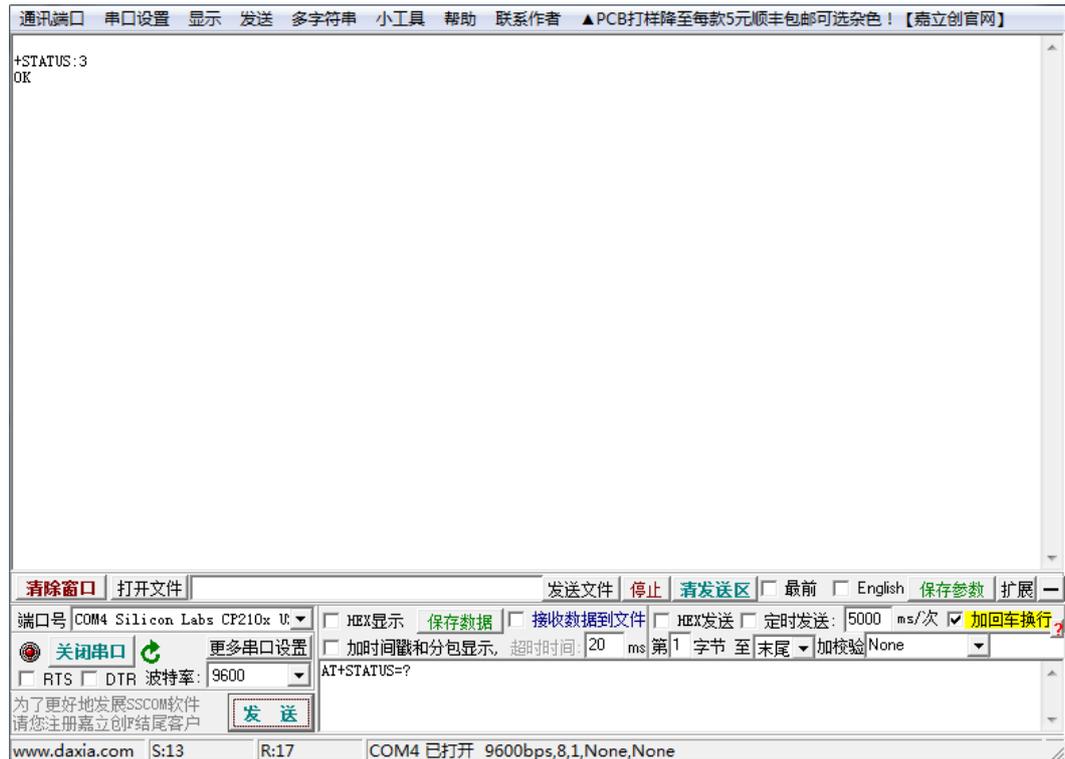
如果没有找到COM端口，请安装[驱动精灵](#)，下载对应的驱动。

步骤4 设置参数。



- **Port:** 选择3获取的与开发板连接的端口号，如COM4。
- **Baud rate:** 9600。
- **Data bits:** 8。
- **Stop bits:** 1。
- **Parity:** 配置为None。
- **Flow control:** 配置为None。

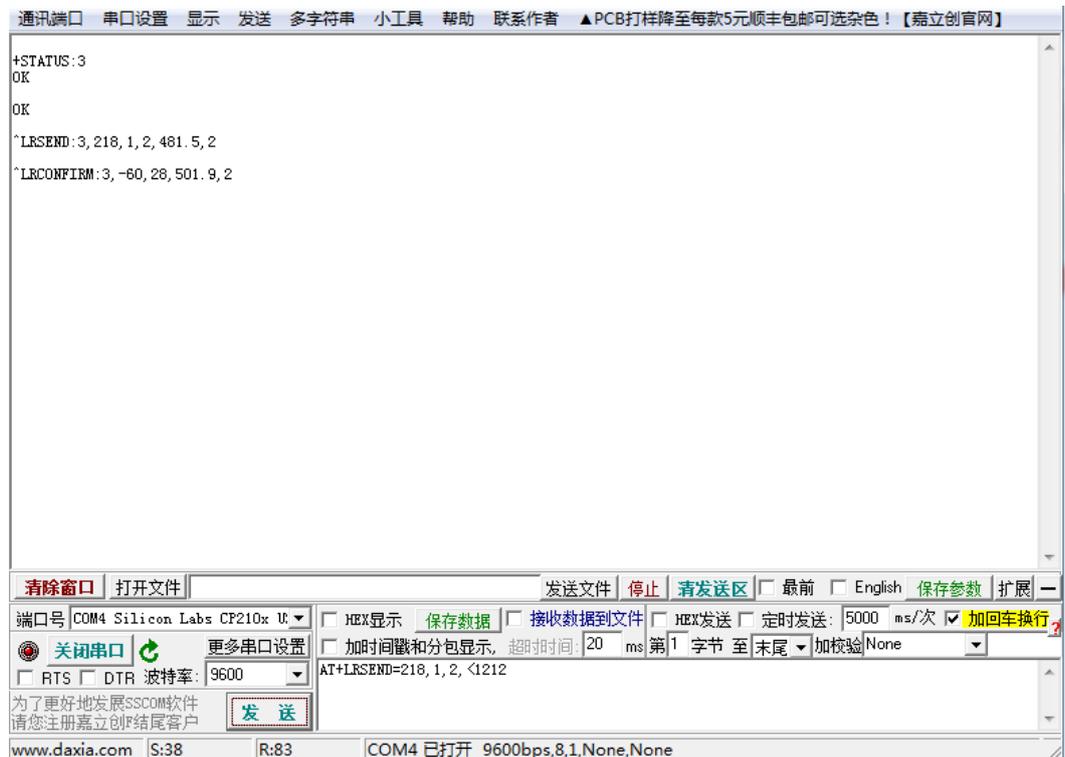
步骤5 输入AT+STATUS=?查询入网状态，如果返回“STATUS:3”表示入网成功。



说明

如果返回其他值表示入网失败，具体请联系唯传科技提供技术支持。

- 步骤6** 入网成功后，发送上报数据的A命令AT+LRSEND=<port>,<confirm>,<len>,<data>，其中<port>为端口，<confirm>为是否需要网关回复确认，<len>为发送数据的长度，<data>为发送的数据。



步骤7 在设备管理服务中选择“设备 > 所有设备”，单击设备标识码为test_lora的设备进入“设备详情”页。

步骤8 选择“子设备”页签，在子设备列表单击设备名称为“test_lora_node”的设备进入子设备详情，查看对应上报的数据。



步骤9 应用调用下发命令接口，下发请求到物联网平台，携带expireTime参数为0，其中“fport”为设备上报的数值。消息样例如下：

```
{
  "serviceId": "RawData",
  "method": "RawData",
  "expireTime": 0,
  "body": {
    "data": "test001",
    "fport": 45
  }
}
```

响应示例：

Status Code:201 Created

Content-Type:application/json

```
{
  "commandId": "string"
}
```

步骤10 在设备管理服务中“设备 > 所有设备 > 子设备”中查看设备接收到的命令。

命令详情				×
状态	SUCCESSFUL	命令ID	141558becfe644d0877db0a0e8ce0562	
命令创建时间	2019-12-10 14:47:36	等待耗时 (秒)	--	
平台发送时间	--	送达耗时 (秒)	--	
送达时间	--	执行耗时 (秒)	--	
执行时间	--	消息总耗时 (秒)	--	
指令类型	Rawdata	超时时间 (秒)	0	
服务ID	Rawdata	发送次数	--	
命令参数	{"data": "test001", "fport": "45"}			
设备回复	{"code": 200, "desc": "handle successful"}			

----结束