设备管理

开发指南

文档版本15发布日期2022-06-14





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

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。

1 开发前必读(联通用户专用)	1
2 资源获取(联通用户专用)	5
3 从这里开始	10
3.1 我是模组厂商(联通用户专用)	11
3.2 我是设备厂商(联通用户专用)	11
3.3 我是设备用户(联通用户专用)	14
3.4 我是应用开发者(联通用户专用)	16
4 平台侧开发	18
4.1 体验开发中心(联通用户专用)	
4.2 创建项目(联通用户专用)	39
4.3 创建产品(联通用户专用)	42
4.4 开发 Profile	47
4.4.1 什么是 Profile (联通用户专用)	47
4.4.2 在线开发 Profile (联通用户专用)	48
4.4.3 离线开发 Profile (联通用户专用)	52
4.4.4 导出和导入 Profile	64
4.5 开发编解码插件	67
4.5.1 什么是编解码插件(联通用户专用)	67
4.5.2 在线开发插件(联通用户专用)	69
4.5.3 离线开发插件(联通用户专用)	133
4.5.4 下载和上传插件(联通用户专用)	
4.6 调测产品(联通用户专用)	166
4.6.1 设备侧开发	172
4.6.1.1 使用 MQTTS 协议接入(联通用户专用)	172
4.6.1.2 使用 LoRaWAN 协议接入(联通用户专用)	173
4.6.1.3 使用 Modbus 协议接入(联通用户专用)	177
4.6.1.4 使用 Agent SDK 接入	
4.6.1.4.1 Agent Lite SDK 使用指南(C)(联通用户专用)	
4.6.1.4.2 Agent Lite SDK 使用指南(Java)(联通用户专用)	198
4.6.1.4.3 Agent Lite SDK 使用指南(Android)(联通用户专用)	
4.6.1.5 使用模组接入	226
4.6.1.5.1 Agent Tiny SDK 使用指南(联通用户专用)	

4.6.1.6 软固件升级调测	229
4.6.1.6.1 固件升级(联通用户专用)	
4.6.1.6.2 软件升级(联通用户专用)	234
4.6.1.7 应用侧开发	240
4.6.1.7.1 使用 API 对接	
4.6.1.7.1.1 API 使用指导(联通用户专用)	240
4.6.1.7.1.2 使用 Postman 调测(联通用户专用)	241
4.6.1.7.1.3 使用 Java API Demo 调测(联通用户专用)	259
4.6.1.7.2 使用 SDK 对接	
4.6.1.7.2.1 Java SDK 使用指南(联通用户专用)	
4.6.1.7.2.2 Python SDK 使用指南(联通用户专用)	
5 调测证书制作(联通用户专用)	299
6 自助测试(联通用户专用)	303
7 产品发布(联通用户专用)	312
8 商用对接(联通用户专用)	315
9 IoT 技术认证(联通用户专用)	323

目录

开发前必读(联通用户专用)

非联通用户请查看设备接入服务。

方案概述

基于设备管理服务去实现一个物联网解决方案时,需要完成以下开发操作:

开发操作	开发说明
平台侧的开发	主要包括Profile的开发和编解码插件的开发。编解码插件的开发 仅针对上报数据为二进制码流格式的设备,对于上报数据为JSON 格式的设备不需要开发编解码插件。
设备侧的开发	主要为设备与物联网平台的集成对接开发,包括设备接入物联网 平台、业务数据上报和对平台下发控制命令的处理。
应用侧的开发	主要为业务应用与物联网平台的集成对接开发,包括API接口的调 用、业务数据的获取和HTTPS证书的管理。

基于物联网平台开发一个物联网解决方案的工作流程请参考从这里开始。

开发中心与设备管理服务的差异

• 了解开发中心

开发中心是基于设备管理服务提供的物联网一站式开发工具,帮助开发者快速进行Profile(产品模型)和编解码插件的开发,同时提供在线自助测试、产品发布等多种能力,端到端指引物联网开发,帮助开发者提升集成开发效率、缩短物联网解决方案建设周期。

- 产品开发:提供产品开发向导,端到端引导开发者完成Profile开发、插件开发以及产品调测,助力物联网产品快速上线。
- 应用开发:支持对接信息、订阅调试、应用调试等能力,帮助开发者进行应用侧开发和调试,助力物联网应用便捷开发。
- 自助测试:支持对设备、应用进行自动化测试,并生成测试报告,检验产品 是否达到发布标准。
- 产品发布:产品在开发中心完成自助测试后,开发者可以一键申请发布到产品中心,已发布的产品可直接应用于商用环境。

• 了解设备管理服务

请前往<mark>平台简介</mark>查看设备管理服务的介绍。

- 开发中心和设备管理服务有哪些差异?
 - a. 两者所在的物联网平台环境不一样

开发中心所在的平台环境为测试环境,设备管理服务所在的平台环境为商用 环境。两个环境的设备数据不能互通,已对接测试环境的设备/应用要迁移商 用环境,需要在商用环境重新创建应用和注册设备,并修改设备/应用中平台 的接入信息。

b. 两者的设备数量限制不一样

开发中心单个项目(应用)下最多可注册20个设备,设备管理服务单个应用 下最多可注册1000万个设备。在其他的平台能力上,两者保持一致,例如开 发中心和设备管理服务使用同样的API,更多的平台能力使用限制请查看使用 限制。

c. 两者的计费策略不一样

开发中心为免费使用,设备管理为付费使用,设备管理服务的具体计费策略 请查看<mark>计费详情</mark>。

开发中心和设备接入服务是否有联系?

开发中心是基于设备管理服务的物联网开发工具,在开发中心上开发的产品 (Profile文件和编解码插件)只适用于设备管理服务,不能在设备接入服务中导 入和使用 。

业务概览

当用户在开通设备管理服务时,系统默认一起开通设备接入服务,即用户在使用设备 管理服务时,包含设备接入服务的能力。使用设备管理服务的完整流程如下图所示, 主要分为在开发中心(测试环境)进行产品开发、在控制台(商用环境)进行上线和 日常管理。

- 产品开发:开发者在进行设备接入前,基于开发中心进行相应的开发工作,包括 平台侧开发、设备侧开发、应用侧开发,是真实设备接入到设备管理服务的前提 条件。
- 上线(转商用):基于设备管理服务提供的控制台,将真实设备接入到设备管理 服务中,并对接用户开发的应用服务器,实现设备的远程监控和控制。
- 日常管理:真实设备接入后,基于控制台或者API接口,进行日常的应用管理和设备管理。

远端开发	转商用	设备侧开发	云端日常管理
 创建项目 创建产品 开发Profile 开发编解码插件 设备侧开发 应用侧开发 应用侧开发 4 API使用指导 使用SDK对接 调测产品 自助测试 产品发布 	 创建应用 导入产品 注册设备 接入设备 应用接入 对接验证 	 使用MQTTS协 议接入(联通用 户专用) 使用Modbus协 议接入(联通用 户专用) 使用Agent SDK接入 使用模组接入 软固件升级调测 	 应用管理 订阅推送 订役推送 授权可 设备管理 设备管理 设备注册鉴 公本 数命令配置 公备配置 公备影子 没备影子 规组与标签 设程計算 近程許多 面件升级 第二2000





商用环境

2 资源获取(联通用户专用)

平台对接信息

设备和应用接入物联网平台前需要获取平台的接入地址信息。

平台环境	获取途径		
开发中心(测试 环境)	进入 <mark>开发中心</mark> 的具体项目中, "设备接入信息"和"应用报	在"应用 > 对接信息"页面查 _{接入信息} "。	쥽
	设备接入信息		
	LWM2M(CoAP) 接入方式 ①	MQTT 接入方式 ⑦	
	4 32:5683 (CoAP) 4 32:5684 (CoAPS)	4 24:8883 (MQTTS)	
	AgentLite 接入方式 ⑦		
	4 24:8943 (HTTPS) 4 24:8883 (MQTTS)		
	应用接入信息 ⑦		
	HTTPS 接入方式		
	4 91:8743		
设备管理服务 (商用环境)	在物联网平台的 <mark>管理控制台</mark> , 接入信息"下可查看设备和应	选择设备接入,在"设备和应 团用的接入地址信息。	ī用
	接入信息		×
	关型 域名	端□ 协议	
	应用对接信息 iotmyhuaweicloud.com	m 8743 HTTPS	
	设备对接信息 iot myhuaweicloud.	com 5683 CoAP	
	设备对接信息 iot myhuaweicloud.	com 5684 CoAPS	
	设备对接信息 iot myhuaweicloud.com	m 8943 HTTPS	
	us im Apparties 25 TOU	111 0000 MULLS	

设备开发资源

物联网平台支持设备通过MQTT协议和LWM2M/CoAP协议进行接入,设备可以通过调 用设备侧的接口或者集成SDK的方式接入到物联网平台。

资源包名	描述	下载路径
Agent Lite SDK(Linux C)	设备可以通过集成Agent Lite SDK接入物联网平台, Demo提供了调用SDK接 口的样例代码。 使用指导可以参考 Agent Lite API参考(C)和Agent Lite SDK使用指南(C)。	 Agent Lite SDK(Linux通用) Agent Lite SDK(指定 工具链) Agent Lite Demo(C- Linux)
Agent Lite SDK(Windows C)	设备可以通过集成Agent Lite SDK接入物联网平台, Demo提供了调用SDK接 口的样例代码。 使用指导可以参考 Agent Lite API参考(C)和Agent Lite SDK使用指南(C)。	 Agent Lite SDK(Windows) Agent Lite Demo(C- Windows) 说明 Demo的libs文件夹下为32位 SDK,若需要使用64位 SDK,请下载并替换Demo 中的SDK。
Agent Lite SDK(Java)	设备可以通过集成Agent Lite SDK接入物联网平台, Demo提供了调用SDK接 口的样例代码。 使用指导可以参考 Agent Lite API参考(Java)和 Agent Lite SDK使用指南 (Java)。	Agent Lite Demo(Java) 说明 • SDK在Demo的libs文件 夹下。 • Agent Lite Demo(Java) 仅适用于Windows系统 环境。
Agent Lite SDK(Android)	设备可以通过集成Agent Lite SDK接入物联网平台, Demo提供了调用SDK接 口的样例代码。 使用指导可以参考 Agent Lite API参考(Android)和 Agent Lite SDK使用指南 (Android)。	Agent Lite Demo(Android) 说明 SDK在Demo的libs文件夹 下。
LiteOS SDK	设备可以通过集成LiteOS SDK接入物联网平台, Demo提供了调用SDK接 口的样例代码。	LiteOS SDK

资源包名	描述	下载路径
Profile模板	Profile模板中包含了典型 场景的Profile样例,开发 者可以在模板基础进行修 改,定义自己需要的 Profile。 使用指导可以参考 <mark>离线开</mark> 发Profile。	Profile开发示例
编解码插件样例	编解码插件的代码样例工 程,开发者可以基于该样 例工程进行二次开发。 使用指导可以参考 <mark>离线开 发插件</mark> 。	编解码插件开发样例
编解码插件检测工具	用于检测离线开发的编解 码插件的编解码能力是否 正常。	编解码插件检测工具
NB-IoT设备模拟器	用于模拟以CoAP/ LWM2M协议接入物联网 平台的NB设备,实现数据 上报和命令下发功能。 使用指导可以参考 <mark>体验开 发中心</mark> 。	NB-IoT设备模拟器
loT Studio	loT Studio是基于LiteOS 嵌入式系统软件开发的工 具,支持C,C++、汇编等 多种开发语言,提供了代 码编辑、编译、烧录及调 试等一站式开发体验。 使用指导可以参考 <mark>设备侧</mark> 开发实践。	IoT Studio

应用开发资源

为了降低应用的开发难度、提升开发效率,物联网平台开放了丰富的Restful API和 SDK包。应用通过调用物联网平台的API,实现安全接入、设备管理、数据采集、命令 下发等业务场景。

资源包名	描述	下载
应用侧开发 API JAVA Demo	物联网平台为应用服务器 提供了Restful API,能够 让开发者快速验证Restful 接口开放的能力,体验业 务功能,熟悉业务流程。 使用指南可以参考loT平台 应用侧API参考。	API JAVA Demo
应用侧开发 Java SDK	Java SDK提供JAVA方法调 用物联网平台Restful接口 与平台通信,Demo提供 调用SDK接口的样例代 码。 使用指南可以参考IoT平台 应用侧JAVA SDK API参 考和JAVA SDK 使用指南	JAVA SDKJAVA SDK Demo
应用侧开发 Python SDK	Python SDK提供Python方 法调用平台Restful接口与 平台通信,Demo提供调 用SDK接口的样例代码。 使用指南可以参考IoT平台 应用侧Python SDK API 参考和Python SDK使用 指南。	 Python SDK Python SDK Demo

证书资源

在设备和应用对接物联网平台的部分场景中,需要在设备侧和应用侧集成相应证书。 请点击获取<mark>证书文件</mark>。

🛄 说明

此证书文件只适用于华为云物联网平台。

证书类型,证书格式,以及适用开发语言,用途详见下表。

证书包名 称	一级目 录	二级目 录	三级 目录	说明
certificat	Northb	code	Java	应用服务器通过HTTPS协议调用物联网平
e	ouna API		PHP	一台接口,用于校验物联网半台的台法性 时,使用该目录下的证书。请根据应用服
			Pytho n	务器侧的编程语言选择相应目录下的证书 文件,并置于应用服务器侧。
		postm an	-	Postman通过HTTPS协议调试物联网平台 接口时,使用该目录下的证书。

证书包名 称	一级目 录	二级目 录	三级 目录	说明
	Agent Lite	Androi d	-	终端设备或网关通过集成Agent Lite SDK 接入物联网平台时,使用该目录下的证
		C- Linux	-	书。请根据终端设备或网关侧的编程语 选择相应目录下的证书文件,并置于终望 设备或网关侧。
		Java	-	



联通用户专用,非联通用户请查看设备接入服务。

使用设备管理服务的完整流程如下图所示,主要分为产品开发、上线和日常管理三个部分。用户可以根据自己角色的不同,执行对应的操作即可。

 产品开发:在开发中心(测试环境)操作,开发者在接入设备前,需要进行相应 的开发工作,包括平台侧开发(开发中心)、设备侧开发、应用侧开发,是真实 设备接入到设备管理服务的前提条件。

其中设备侧开发和应用侧开发不分先后顺序,可同步进行。



- 上线(转商用):基于设备管理服务提供的控制台,将真实设备接入到设备管理 服务中,并对接用户开发的应用服务器,实现设备的远程监控和控制。
- 日常管理:真实设备接入后,基于控制台或者API接口,进行日常的应用管理和设备管理,详情请参考使用指南。
- 3.1 我是模组厂商(联通用户专用)
- 3.2 我是设备厂商(联通用户专用)
- 3.3 我是设备用户(联通用户专用)
- 3.4 我是应用开发者(联通用户专用)

3.1 我是模组厂商(联通用户专用)

非联通用户请查看设备接入服务。

芯片是一种把集成电路小型化的方式,并时常制造在半导体晶圆表面上。模组是由若 干个显示模块、驱动电路、控制电路、芯片以及相应的结构件构成的一个独立的显示 单元。模组与芯片关系就像是U盘和Flash芯片的关系,用户99%的情况是直接使用模 组,没有人直接用芯片。当前模组厂商主要提供Wifi、NB-IoT、2/3/4/5G等通信模 组。

在物联网解决方案中,作为模组厂商的您需要让模组实现MQTTS、LWM2M、CoAP等 物联网协议栈及连接平台的能力,您只需要将Agent Tiny SDK集成在现有的模组固件 包中,这样模组就具备了接入华为物联网平台的能力。我们将提供华为IoT技术认证 支持,帮助您快速完成SDK集成,通过华为认证的模组可以申请入驻华为云市场,我 们将会推荐给物联网平台用户。查询华为已认证的模组。

Agent Tiny SDK具有普适性,可以广泛移植于WiFi模组、2/3/4/5G模组、NB-loT模 组,无需考虑模组类型(**了解Agent Tiny SDK详情**)。如何在模组中集成Agent Tiny SDK,请参考**华为loT技术认证流程**,获取相关技术支持。



3.2 我是设备厂商(联通用户专用)

非联通用户请查看设备接入服务。

角色介绍

作为开发并销售最终设备的厂商,您需要进行设备集成开发,以便让设备具备接入物 联网平台的能力。我们将提供<mark>华为loT技术认证</mark>支持,帮助您快速完成设备适配和认 证。

通过华为认证的设备可以申请入<mark>驻华为云市场</mark>,我们将会推荐给物联网平台用户。查 询:<mark>华为已认证的设备</mark>。

根据设备是否具有IP通信能力,设备接入平台分为以下两种方式:设备直接接入平台和设备通过网关接入平台,您可以根据自己的设备选择合适的方式。

前提条件

已完成平台侧开发。

直接接入平台

针对已实现TCP/IP协议栈的设备,可以直接与平台进行通信,常见的设备包括网关、 以太网设备、NB-loT设备等。根据设备自身硬件的特点不同,华为物联网平台提供四 种方式直接接入,您需要根据自身行业特征及业务情况选择合适的接入方案,四种方 案优缺点如下:

接入方式	方案优点	方案约束
方案一:集成 LiteOS接入物联 网平台	有对应的SDK,集成 难度较低,对您的开 发技能要求较低。	方案不灵活,开发者只能根据LiteOS提 供的API去实现自己的功能,超出API外 的功能,无法自定义,对于设备的硬件 要求比较高。
方案二:没有配 置模组时,通过 集成Agent Lite SDK接入物联网 平台	有对应的SDK,集成 难度低,对您的开发 技能要求较低。	方案不灵活,开发者只能根据Agent SDK提供的API去实现自己的功能,超出 API外的,无法自定义。
方案三:配置模 组时,通过集成 Agent Tiny SDK 接入物联网平台	集成难度非常低,对 您的开发技能要求 低。	需要采购指定型号的模组。
方案四:通过实 现原生协议接入 物联网平台	方案比较灵活,可根 据业务需要,实现协 议定义的功能,对设 备硬件无限制。	需要从底层协议开始实现,集成难度 大,代码开发量大,对于开发者要求 高。

方案一:设备需要具备智能操作系统,硬件满足RAM容量 > 32KB, Flash容量 > 128KB时,通过集成LiteOS操作系统接入平台。

集成开发流程详见以下链接,请根据设备支持的协议选择:

- LiteOS SDK端云互通组件CoAP/LWM2M开发指南



方案二:设备没有配置模组时,需通过集成Agent Lite SDK接入平台。了解Agent Lite SDK详情。

当设备存储及计算能力较强(满足RAM容量 > 4MB,Flash容量 > 2MB)时,推荐设备集成Agent Lite SDK。

目前Agent Lite SDK支持C、Java和Android三个版本。实际开发中,请根据开发 时使用的语言、平台,选用合适的设备端SDK集成:

- Agent Lite SDK集成开发指导(C)
- Agent Lite SDK集成开发指导(Java)
- Agent Lite SDK集成开发指导(Android)



• 方案三:当设备配置有模组时,根据模组特征,选择性集成Agent Tiny SDK接入平台。**了解Agent Tiny SDK详情**

- 如果采用经过华为认证的模组,该模组已集成Agent Tiny SDK,可以利用模组直接接入物联网平台。您可以访问**华为云市场**,购买已认证的模组。



- 如果采用未经过华为认证的模组,您需要集成Agent Tiny SDK,根据SDK集成位置,分为两种模式:
 - MCU+模组模式:此模式下,设备包含MCU(Microcontroller Unit)和 通信模组,其中MCU集成 Agent Tiny SDK及运行产品逻辑,模组作为通 信模块,提供通信网络。请参考华为IoT技术认证流程,获取相关技术支 持。



 OpenCPU模式:此模式下,设备只包含通信模组,模组集成 Agent Tiny SDK及运行产品逻辑,请参考华为IoT技术认证流程,获取相关技术 支持。



方案四:设备通过实现原生MQTT协议接入,适合一切设备。可查看《设备集成》了解操作。您可访问《MQTT接口参考》获取华为物联网平台开放的能力。



通过网关接入平台

针对未实现TCP/IP协议栈的设备,由于无法直接同物联网平台通信,它需要通过网关 进行数据转发。常见设备包括智慧园区中的照明系统、楼宇系统设备等。这些场景下 的限制如下:

制约因 素	详细描述
设备限 制	设备非常简单,没有集成TCP/IP协议栈,无法提供IP通信网络能力,只能 基于一些简单的近场通信协议如ZigBee、ZWave、Bluetooth或者是其他 的一些非IP有线方式传输协议如串口、并口等接入,此时需要一个网关设 备,先让设备接入到网关,再通过网关与华为物联网平台通信。
网络限	设备部署在独立的网络内,由于安全等考虑,无法直接与物联网平台互
制	通,需要借助网关进行网络桥接。
业务限	下层设备的管理控制逻辑比较复杂,需要利用网关软件自身具有对应的逻
制	辑控制能力配合物联网平台进行业务管理。

制约因 素	详细描述
集成限	子设备已经接入到现有系统,不愿意再进行设备改造,需要将现有系统通
制	过网关接入到物联网平台。

针对此类设备,通过在网关上集成Agent Lite SDK, 设备将数据上报给网关,通过网关 转发到华为物联网平台。华为公司协同合作伙伴,对已集成Agent Lite SDK的网关进 行了认证,您可以访问**华为云市场**,购买符合自身业务的网关产品。目前Agent Lite SDK支持C、Java和Android三个版本。实际开发中,请根据开发时使用的语言、平 台,选用合适的SDK集成:

- Agent Lite SDK集成开发指导(C)
- Agent Lite SDK集成开发指导 (Java)
- Agent Lite SDK集成开发指导(Android)



3.3 我是设备用户(联通用户专用)

非联通用户请查看设备接入服务。

作为设备用户,您既是设备的购买者,又可能是设备的实际管理者,通常您需要考虑 两件事:

- 在采购设备时,如何评估设备满足您的业务需求并且具备接入物联网平台的能力。关于此方面的知识,您可以参考设备能力评估。
- 在进行日常管理时,针对不具备接入物联网平台的设备,如何将现有设备进行改造,以便设备能接入到物联网平台中。关于此方面的知识,您可以参考设备改造。

前提条件

已完成平台侧开发。

设备能力评估

本节主要关注设备通信能力的评估。由于承接业务的不同,对于设备的要求也不同, 采购的设备通信能力评估一般由以下几个流程:

	分析业务场景	 分析接入网络类型	>	分析设备接入协议	>	确定设备选型	
L							

针对您所处的行业,您需要深度分析业务对于设备(硬件资源、电池、性能等)、网络实时性及网络覆盖度等要求,确保现有设备满足业务需求。典型案例如下表所示。

典型 行业	设备特征	技术关注点
智慧 抄表	接入设备数量多、电量有限、设备常常 分布在地上或地下多个区域,要求有信 号覆盖。	设备数量多,接入成本要 低,自身硬件资源有限,要 求设备网络要深度覆盖,低 功耗,数据量较少。
智慧园区	应用子系统多、设备传感器种类多、无 统一的通信协议标准,上报的数据无统 一格式。设备大部分无IP通信能力,需要 通过网关接入。可通过有线网络接入。	应用独立子系统多,格式不 统一,数据孤岛多,需要一 个统一平台管理。设备不具 备IP通信能力,硬件资源有 限,通常通过网关转发数 据,因此主要关注网关的通 信能力。
车联网	设备电量充足。	需要实时对车辆数据进行分 析等,因此需要接入网络实 时性高,数据传输速率高 等,主要关注接入网络选 型,要选传输速率较高的无 线网络接入。

2. 根据业务对于数据及成本要求,确认设备支持的网络是否能够满足业务需求。

业务应用	推荐的接入 网络	网络特点
智慧家庭、智 慧楼宇等	以太网	传输速率可达到10Gbit/s、随时在线、成本高。
车联网、视频 监控等	4G/5G/LTE- v	传输速率>10Mbit/s、功耗高、成本高。
电子广告、无 线ATM、梯联 网等	eMTC、 GPRS	传输速率<1Mbit/s、功耗较低、成本较低。
远程抄表、智 能停车等	NB-loT、 LoRa	传输速率<100Kbit/s、功耗低、成本低、穿透力 强、信号覆盖好。

3. 确认设备是否集成了物联网协议,以便保证设备可以连接到物联网平台,利用物 联网平台进行设备管理。华为物联网平台目前支持设备采用以下协议接入。

通信 协议	协议描述	应用场景
LWM 2M	LWM2M是开发移动联盟OMA定 义的用于设备管理的应用层通讯 协议,主要使用在资源受限的嵌 入式设备上。	NB-loT设备接入平台,业务实时性 要求不高,低功耗、信号广覆盖场 景。
СоАР	CoAP是资源受限设备和受限网络 专用的Web传输协议,专为机器 对机器的应用而设计。CoAP提供 请求/响应交互模型,支持内置的 服务和资源发现。需要底层实现 UDP协议。	NB-loT设备接入平台,业务实时性 要求不高,低功耗、信号广覆盖场 景。
MQT T	MQTT是一种物联网连接协议,提 供非常轻量级的发布/订阅消息传 输方式,用于在低带宽、不可靠 的网络的设备管理。该协议构建 于TCP/IP协议上。	对设备的可靠性和实时性要求高, 适合长连接的场景,如智能路灯 等。

4. 根据前面步骤,总结设备的特征,选择与业务相匹配的设备。例如针对智能抄表行业,要求电表通信模块具有覆盖广、穿透力强、耗电量小、成本低特点,因此接入网络选择NB-IoT网络,设备采用NB-IoT模组,模组集成了LWM2M协议,可以保证设备接入华为物联网平台。

设备改造

设备如果不具备接入华为物联网平台能力,需要对设备进行改造,改造方法跟设备厂 商进行设备集成方法一样,详见<mark>我是设备厂商</mark>介绍。

3.4 我是应用开发者(联通用户专用)

非联通用户请查看设备接入服务。

华为物联网平台面向全球各行各业提供物联网服务,并通过Restful API的形式对外开 放物联网平台丰富的设备管理能力。应用开发人员基于API接口开发所需的行业应用, 如智慧城市、智慧园区、智慧工业、车联网等行业应用,满足不同行业的需求。开发 包括如下几个关键阶段。



应用服务器开发阶段	操作指引
调用API接口或者集成 SDK	应用开发者通过 <mark>调用API接口</mark> 或者 <mark>集成SDK</mark> 调用物联网平台 的能力进行应用服务器的开发。可参考 <mark>表3-1</mark> 选择合适的开 发方式。

应用服务器开发阶段	操作指引
开发接收设备数据的 接口	应用服务器需开发接收数据对应的接口(对应订阅管理接 口中的callbackUrl地址),用于应用服务器向物联网平台 发起订阅后,接收物联网平台推送的设备相关数据。
制作调测证书与加载	 应用服务器通过HTTPS协议调用物联网平台提供的API接口,需要在应用服务器上预置CA证书,用于应用服务器 校验物联网平台的合法性,该证书可通过下载证书文件 获取。
	 物联网平台通过HTTPS或HTTP协议向应用服务器推送数据,当使用HTTPS协议时需要在物联网平台上加载CA证书,同时在应用服务器上加载设备证书,用于物联网平台校验应用服务器合法性。在调测时,您可以通过制作自签名证书进行调测。在商用时,建议您向证书知名机构申请和购买商用证书,以确保证书的安全性。
接入设备管理服务	物联网平台提供了 开发中心(调测平台) 和 商用管理平 台,您在进行应用开发和调试时,可以先接入开发调测平 台,待功能开发完善,具备商用使用条件时,再接入商用 管理平台。
	应用接入地址获取请参考 <mark>平台对接信息</mark> ,应用ID和应用密 钥为在商用管理平台上"创建应用"时分配。

表 3-1 开发方式对比说明

开发方式	优点	缺点	适用场景
调用API接 口	 开发灵活,随需调用API接口。 对于应用开发语言无限制,支持所有的开发语言 	 开发工作量、开发 难度相比集成SDK 大。 应用上线周期相对 较长。 	 企业开发能力 强,需灵活使用 物联网平台的能力。 企业已有应用服 务器,需要对接 物联网平台。
集成SDK	 代码开发工作量较小,开发能力的门槛相比直接调用API接口较低。 开发周期短,可以快速构建应用服务器。 	 与直接调用API接口 相比,开发的灵活 性稍差。 开发语言支持 Java、PHP和 Python,暂还不支 持所有的开发语 言。 	适用于企业对应用 的个性化的定制要 求不高,能够快速 构建和上线应用。



- 4.1 体验开发中心(联通用户专用)
- 4.2 创建项目(联通用户专用)
- 4.3 创建产品(联通用户专用)
- 4.4 开发Profile
- 4.5 开发编解码插件
- 4.6 调测产品(联通用户专用)

4.1 体验开发中心(联通用户专用)

非联通用户请查看设备接入服务。

本文通过"智慧路灯"为示例,通过开发中心提供的设备模拟器和应用模拟器替代真 实的设备和应用,带您快速体验设备上报数据到物联网平台和远程下发控制命令到设 备的全过程。

假设:

路灯设备上报一条数据消息,包含路灯的光照强度(Light_Intensity)和路灯的开关状 态(Light_Status); 支持远程控制路灯开关状态的命令(SWITCH_LIGHT),上报数 据的格式为二进制格式 。

前提条件

已经注册华为官方帐号,具体可参考<mark>华为账号注册</mark>。

已经在华为云上完成实名制认证,具体可参考<mark>实名认证</mark>。

开发中心申请和创建项目

用户第一次使用物联网平台云服务时,首先需要申请开通"开发中心"服务。基于开发中心,用户可以在线开发设备的Profile文件和编解码插件;另外开发中心提供了设备和应用模拟器,可以便捷地调测开发的Profile文件和编解码插件的正确性。

步骤1 登录华为云官方网站,访问IoT设备管理服务。

步骤2 在设备管理云服务首页,点击"开发中心"开通业务。

- 步骤3 填写开通信息,本示例以选择"智能路灯解决方案"为例,点击"**立即开通**"。
- **步骤4** (可选)进入"开发中心"首页后,如果用户是第一次使用,首先需要点击右上角的 "**厂商信息**",编辑并完善厂商信息后,返回主页。
- **步骤5** 在"开发中心"首页,点击"**新建项目**",创建一个新的物联网产品项目,填写项目 名称、所属行业后,点击"**确定**"。

新建项目			×
*项目名称			
StreetLight001			
*所属行业			
公用事业(NB-IoT)		-	
描述			
	确定		

步骤6 创建完项目后,会生成应用ID和应用密钥,请将密钥下载到本地并妥善保管,然后进入项目。

🛄 说明

应用密钥用于应用服务器接入的鉴权,在界面上不可见,请妥善保管,如果忘记了密钥,可以在 "**对接信息**"功能中,对密钥进行重置。



步骤7项目创建完成后,可以在"**对接信息**"中,查看项目的基本信息,例如设备接入信息、应用接入信息、重置密钥等。

≡ 🔊 OceanConnect	当前项目: StreetLight001 🗸			文档中心 论坛 厂商信息	<mark>ф[•] en</mark> q
▲ 项目概览	基本信息				
● 产品 ▲	对接信息详情				如何集成应用
产品开发	主要编取 巴		应用安全		
设备管理	应用实称 Streetl inht001	应用D WQf 352SkviTKucQaTH_bEbvQ98a	应用面明 ◎ ● ● 四字 相		
升级调试	())))))))))))))))))))))))))))))))))))	应用描述	● 通道正书 ○		
- □ 应用 ^	2019/05/10 17:17:28	督无	尚无推送证书 证书管理		
对接信息					
订阅调试	设备接入信息		行业信息 区		
应用调试	LWM2M(CoAP) 很入方式 ① 4 2 5683 (CoAP)	MQTT 把入方式 ③ 45 • • • • • • • • • • • • • • • • • • •	^{所履行业} 公用哪业(NB-IoT)	应用能力 ① 存錄模式	
应用在线开发	Annel in State (COAP 3)		(E-4)-80-9279-22-1280-		
Web应用开发	4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4		7天		Q
					信思
	应用接入信息 ⊙				助手
	HTTPS 接入方式 4 4 1 11:8743				_

----结束

产品在线开发

产品模型(也称Profile)用于描述设备具备的能力和特性。开发者通过定义Profile文件,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支持的服务、属性、命令等信息,例如温度、光强度、开关等。



本示例以一款智慧路灯为例进行描述,假设路灯设备上报一条数据消息,包含路灯的 光照强度(Light_Intensity)和路灯的开关状态(Light_Status);支持远程控制路灯 开关状态的命令(SWITCH_LIGHT)。



	当前项目: StreetLight001				
▲ 项目概览	产品开发 > 创建产品				
● 产品 ▲				注注1211	下一种大式创建你的文中
产品开发				府 应并以	下一种力式的建态的广西
设备管理	基于系统模板创建	基于快速集成模板创建	基于已有产品创建	本地导入产品创建	自定义产品
升级调试					
■ 应用 🔺					
对接信息					
订阅调试					
应用调试					
应用在线开发					
Web应用开发				您可以按照您	1的需求,自主开发一款产品
					自定义产品

步骤2 在弹出的对话框中,设置产品信息,完成后点击"创建"。



步骤3 进入到Profile定义界面后,点击"新建服务",填写服务信息,然后点击" $\sqrt{$ "。

产品开发 > St	treetLight								
\sim	StreetLight 开发中	产品型号 ①: StreetLiph001 所属行业: 智慧地市	设备类型⊙:Stre 厂商名称∶HW	etLight 数据 厂商 收起	格式:二进制码流 ID ①: 88712009c64a41;	协议类型: a19ff8aaa39a31aeb9	LWM2M	已编辑属性	自查看产品详情
	Profile定义	2 编辑码损件3	Ŧġ	3 1689	成指导		4 在线调测		发起自助测试
维护能力	配置								
软件升	级 🔵 软件升级协	<mark>й</mark> РСР		圖件升約	图件升级的	NX LWM2M			提交
服务列表	0							十新建服务 出导出P	rofile 态导入Profile
	服务名称 Service ID		描述				最后修改时间		
^	SteetLight		请输入相关描述						×
	属性列表								
	属性名称·Prop	ertyname 数据	类型 范围	步长	单位	枚举值	长度	是否必选 访问模式	
				+	新增履性				

- **步骤4** 定义路灯上报的环境光强度和路灯开关状态的属性。首先输入"**服务名称**": StreetLight,然后点击"**新增属性**"。
- **步骤5** 首先定义一条属性为:路灯采集的当前环境的光照强度,数据类型为int,光照强度范围为:0~100。

新增属性	×	<
* 名称		
Light_Intensity		
* 数据类型		
int	-	
* 最小值	* 最大值	
0	100	
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		
确定	取消	

步骤6 再点击"添加属性"按钮,定义一条属性为:路灯当前的开关灯状态,数据类型为 int,0代表关闭,1代表打开状态。

新增属性		:
* 名称		
Light_Status		
* 数据类型		
int		•
*最小值	* 最大值	
0	1	
步长	单位	
* 访问模式		
✓ R 属性值可读		
── W 属性値可写(更改)		
✓ E 属性值更改时上报事件		
是否必选		
✔ 是		
	町洋	
WELLE	积川	

- **步骤7** 接下来定义远程控制开关灯状态的命令。点击"**添加命令**"按钮,定义命令名称为: SWITCH_LIGHT。
- **步骤8** 点击"添加下发命令字段",命令名称为:SWITCH_LIGHT,数据类型为:string,长度为:3个字符,枚举值为:ON,OFF。

*名称			
SWITCH_LIC	GHT		
* 数据类型			
string			•
* 长度			
3			
3 ^{女举值} (值之间以	英文逗号分隔)		
3 ^{女举值} (值之间以 ON,OFF	英文逗号分隔)		
3 ^{女举值} (值之间以 ON,OFF	英文逗号分隔)		
3 牧举值 (值之间以 ON,OFF	英文逗号分隔)		
3 文举值 (值之间以 ON,OFF 昰否必选 ☑ 是	英文逗号分隔)		

步骤9 点击"确定"按钮,完成该路灯的Profile文件创建。

----结束

编解码插件在线开发

通常情况下设备为了省电,设备会采用"二进制"格式上报数据,编解码插件的作用 就是将设备上报的"二进制"格式数据,按照Profile文件的定义的属性转换为 "JSON"格式数据,便于物联网平台和应用服务器识别。同时,用户远程下发控制命 令时,物联网平台会将"JSON"格式的命令转换为"二进制"格式数据下发给设备。

注:如果设备本身上报的是JSON格式数据,则不需要定义编解码插件,如使用Agent Lite接入的设备。

步骤1 在"产品开发"页面进入到创建的产品中,选择"编解码插件开发"。

步骤2 在"在线编解码插件编辑器"中,点击"新增消息"。

产品开发 > StreetLight	1			
\sim	StreetLight 开发中	所尾行业: 智慧城市 亚号①: StreetLight001	厂商名称:HW_IoT 设备规型①:StreetLight	数据他式 二进制码流 厂商IDO: 96896463a8a546f882ba18a55fdfbcb7
01) Profile定义 —		— 02 编解码插件开发	→ (03) 端倒集成指导 —	(04) 在线调测 ——
在线编解码插件编辑	■器 插件管理			

步骤3 输入消息名:LightData,消息类型选择:数据上报,然后再点击"添加字段"。

新增消息		
基本信息		
*	海豐識述 通豐憲法	
字段		
	+ 添加字段	

步骤4 输入上报消息的名字: LightIntensity,数据类型: int8u(8位无符号整型),长度:1 字节,单击"**完成**"。

101/1H 1 +X			×
标记为地址域 ⑦			
*名字			
LightIntensity			
描述			
解析上报光照强度			
刻備突型(入)病模式)		-	
* 长度 ②			
1.22			
1			
1			
1 默认值 ⑦ 默认值			
1 默认值 ⑦ 默认值			
1 默认值 ⑦ 默认值			

步骤5 再次单击"**添加字段**",添加路灯上报的路灯开关状态数据。

步骤6 输入名字: LightStatus,数据类型: int8u(8位无符号整型),长度:1字节,单击 "完成"。

添加字段			:	×
标记为地址域 (D			
*名字				
LightStatus				
描述				
描述				
数据类型 (大端模式) int8u(8位无符号整	型)		•	
* 长度 ②				
1				
默认值 ②				
默认值				
偏移值 ⑦				
1-2				
	完成	取消		

步骤7单击"完成",完成路灯上报数据的编解码定义。

步骤8 再次单击"新增消息",定义远程控制路灯开关的命令对应的编解码插件消息。

步骤9 输入消息名:SwitchStatus,消息类型选择:命令下发,然后再点击"添加字段"。

基本信息		
*满意名	满意描述	
SwitchStatus	消息描述	
*消息送型 ○数据上报 ④ 命令下发		
□添加响应字段 ⑦		
字段		
	+ 添加字段	

步骤10 输入名字:SwitchStatus,数据类型:string(字符串类型),长度:3字符,单击 "完成"。

添加字段			
🗌 标记为地址域 🤅)		
□ 标记为响应标识到	神殿 ⑦		
*名字			
SwitchStatus			
描述			
远程控制路灯的开	F关状态		
数据类型 (大端模式)			
string(字符串类型)			•
* 长度 ⑦			
3			
默认值 ②			
默认值			
偏移值 ⑦			
0-3			

步骤11 单击"完成",完成远程控制命令的编解码定义。

步骤12 拖动右侧"设备模型"区域的属性字段和命令字段(Profile文件定义的字段),与编解码插件定义的数据上报消息和命令下发消息的相应字段建立映射关系。

✓ Profile定义 ——	2 编解码插件开发		3 端侧集成指导	4 在线调测	发起自助测试
在线编解码插件编辑器	插件管理				🗋 保存 🛕 部署 😑 更多
+ 新增消息					设备模型 StreetLight へ
LightData	C i	٦	Cight_Intensity StreetLight		属性 命令
SWITCH_LIGHT	LightData		Light_Status		Light_Intensity
	消息类型: deviceReq 是否包含响应信息: No 字节序: 大端模式 描述:		StreetLight		Cight_Status
	数据上报字段	+			
	1 LightIntensity				详情
	2 LightStatus				StreetLight
		5	SWITCH_LIGHT SWITCH_LIGHT		
	SWITCH_LIGHT 消息类型: cloudReq 是否包念明应信息: No				

步骤13 编解码插件与Profile文件建立映射完成后,单击右上角"**保存**",并单击"**部署**", 完成编解码插件的部署。

----结束

使用在线模拟器调试

模拟器在线调测具备设备模拟和应用模拟功能,可以对定义的Profile文件和编解码插 件进行调试,用户可以直观的感受设备上报数据到物联网平台,以及使用物联网平台 下发远程控制命令。

步骤1 在"**产品开发**"页面进入到创建的产品中,选择"**在线调测**",并单击"**新增测试设** 备"。

产品开发 > StreetLig	ht					() 查看产品详情	@如何开发产品
\sim	StreetLight 开发中	所履行业 智慧城市 型号の: StreetLight001	厂商名称: HW_JoT 设新进型O: StreetLight	数编性式: 二进制码流 厂商口⊙: 9689b463a8a5	461682ba18a58fdfbc	协议美型 LWM2M b7	
01 Profile定义 -			(13) 端倒集成指导		4 在线调测 -	······	发起自助测试
设备列表 ③							十新增测试设备
状态	设备名称	设备ID	所履产品	型号	设备类型		操作
			暂无任何设备				

步骤2 选择"没有真实的物理设备",使用模拟器进行调测,单击"创建"后,系统会默认创建一个模拟设备,并接入调测界面。
已有调试设备 : 状态 : ●	StreetLight001NBSimulator	设备类型:StreetLight	型号:StreetLight001
	▲ 立田模拟器 全部 数据接收 命令发送	命令下发 0	
服务	StreetLight 🗸		
命令	SWITCH_LIGHT -		
*SWITCH	_UGHT		<mark>请输入十六进制码流</mark> 请输入十六进制码流
设置时间	×	-	周期 5 秒 目动发送 发送

步骤3 模拟设备数据上报场景,假设上报路灯采集的光照强度为:20,路灯开关状态为:0 (关闭),则在设备模拟器中,输入十六进制码流:1400(光照强度消息为第一个字 节,对应的十六进制码流为14;路灯开关状态为第二字节,对应的十六进制码流为 00),然后单击"**发送**",我们可以在应用模拟器中看到转换为JSON格式的数据为: "Light_Intensity":20, "Light_Status":0。

已有调试设备:	StreetLight001NBSimulator +		
状态: 🌒 🛛 设	备ID:bbb4e44b-8646-4f4c-8ab1-be2bdfb598b9	设备类型:StreetLight	型号:StreetLight001
	应用模拟器	叩∀ NA 0 ◆0 数据 HR	©loT Platform
******	全部 数据接收 命令发送	MIHLIK	◆ 0 数据上报
级结接权:2 { "Light_Inte	nsity": 20, "Light_Status": 0 }		命令下发 。 ↓
			€设备模拟器
			全部 数据发送 命令接收
			数据发送 2019-05-12 11:17:51
服务	StreetLight -		1400
命令	SWITCH_LIGHT -		
*SWITCH_L	IGHT		
ON			请输入十六进制码流
			1400
缓存发送	立即发送		
	~	-	周期 5 秒 自动发送 发送

步骤4 模拟远程下发控制命令场景,在应用模拟器中,选择服务:StreetLight,命令: SWITCH_LIGHT,命令取值为:ON,单击"立即发送",我们可以在设备模拟器中看 到转换为十六进制的码流:4F4E(经ASCLL码转为为十六进制)。

▲应用模拟	器	命令下发 o→ ↓o 数据上报	Not Platform
全部 数据接收 "paras": {"SWITCH_LIGHT": "ON" }; "20190512T032350Z", "executeTimes"; "platformIssuedTime": "20190512T032 "20190512T032251Z", "IssuedTimes"; "resultCode": "DELIVERED", "resultDe "saveNBCommandDTO" } 发送消息body信息: {"serviceId": "Street "SWITCH_LIGHT", "paras": "{\"SWITCH_LIGHT", "paras": "\"SWITCH_LIGHT", "paras": "\"SWITCH_SWIT	命令发送 "reationTime": null, 3512", "deliveredTime": 0, "result": { tali": null }}, "type": HLight", "method": H_LIGHTN":\"ON\"}" }		数据上报
部令状念: 已送送 服务 命令 SWITCH_LIGHT	•		2019-05-12 11:23:51 4F4E
*SWITCH_LIGHT	-		请输入十六进制码流 1400
维存发送 设置时间 >	立即发送		周期 5 秒 自动发送 发送

----结束

使用离线模拟器调试

NB-IoT设备模拟器用于模拟以CoAP/LWM2M协议接入物联网平台的NB设备,实现数 据上报和命令下发功能。

步骤1选择"产品 > 设备管理",单击"添加真实设备"。

	设备列表		
	设备类型	いちまた 〇	皮拟& 直实
产品开发	K B X Z		E W St A
设备管理			
升级调试		在域设备 憲法役務	虚約2時 真如没新
₩ 应用 🔺		5 5	0 0
对接信息			
订阅调试			
应用调试			
应用在线开发			
Web应用开发			
		平台为通过接入真实 & 虚拟的设备,就可以对忽开发的应用和设备进行在场调则,并 进行统一的设备管理	
		添加國政保險	

步骤2 选择前面创建的产品,并填写设备相关信息,单击"确定"。

注:如果使用DTLS传输层安全协议接入时,设备注册方式选择"加密",且请妥善保 存PSK码。

如下信息根据实际情况填写:

- 设备名称: NBdevice01
- 设备标识: aaaaa11111
- 安全密钥(PSK): aaaaa11111aaaaa
- 设备注册方式:加密

新建真实设备			×
01 选择一款产品		22	
	*役無名称 NBdevice01		
	aaaaa11111 安全密钥(PSK) aaaaa11111aaaaa		
	设备注册方式 ⑦ ○ 不加密 ● 加密		
	上一步 裕定		

设备创建成功后,将返回"设备名称"、"设备ID"和"PSK码"。

步骤3 下载并解压NB-IoT设备模拟器,然后双击"NB-IoTDeviceSimulator_zh.jar",运行模拟器。

注:请确保已经安装jdk,否则,无法运行jar文件。

🐌 images	2017/11/2 15:24	文件夹	
Californium.properties	2017/7/29 12:39	PROPERTIES 文件	2 KB
NB-IoTDeviceSimulator_en.jar	2017/8/7 14:08	Executable Jar File	4,361 KB
NB-IoTDeviceSimulator_zh.jar	2017/8/7 14:07	Executable Jar File	4,361 KB
setting.properties	2017/8/7 15:21	PROPERTIES 文件	1 KB

NB-IoT设备模拟器的文件说明如下:

- Californium.properties为模拟器的配置文件。
- NB-IoTDeviceSimulator_en.jar为英文版模拟器。
- NB-IoTDeviceSimulator_zh.jar为中文版模拟器。
- setting.properties为设备模拟器接入物联网平台的配置文件。

步骤4 模拟器启动后,会提示"是否开启DTLS加密传输?",单击"是"。

注: 如果模拟设备不使用DTLS协议接入物联网平台,单击"否"。

提示		×
?	是否开启DTLS加密传输?	
	是(Y) 否(№)	

步骤5 填写"IP地址"、"VerifyCode"和"PSK",单击"注册设备",将模拟器与物联网 平台进行绑定。

注:如果未开启DTLS加密传输,则无需填写PSK。

如下信息根据实际情况填写:

- IP地址:物联网平台的接入地址,请到开发中心"应用 > 对接信息"查看。
- VerifyCode:设备标识码,如:aaaaa11111。
- PSK: aaaaa11111aaaaa

☑ NB-IoT终端模拟器					- • ×
IP地址:	VerifyCode:	aaaaa11111	psk:	1221111122223	注册设备

设备绑定成功,可以回到开发中心,选择"产品 > 设备管理",看见设备 "aaaaa11111"显示"在线",表示模拟器绑定成功。

步骤6 模拟设备数据上报场景,假设上报路灯采集的光照强度为:20,路灯开关状态为:0 (关闭),则在NB-IoT设备模拟器中,输入十六进制码流:1400(光照强度消息为第 一个字节,对应的十六进制码流为14;路灯开关状态为第二字节,对应的十六进制码 流为00),然后单击"发送数据"。

NB-IoT终端模拟器				
IP她址: VerifyCo	de: aaaaa11111	psk:	aa11111aaaaa	注册设备
日志:				
CIG下发的数据内容是: □□				_
CIG IP: 100.95.142.231, PORT: 568	4			-
====================================				=
CIG IP: 100.95.142.231, PORT: 568	4			=
=====================================				-
CIG IP: 100.95.142.231, PORT: 568	4			=
====================================				-
				=
十六进制码流 JSON				
Hex消息:				
1400				-
				•
	发送数据			
是否开启命令监听:		◉ 是	C	一否

数据上报成功,可以回到开发中心,在设备"aaaaa11111"的"历史数据"页面中查 看到转换为JSON格式的数据为:"Light_Intensity": 20, "Light_Status": 0。

步骤7 模拟远程下发控制命令场景,在开发中心,选择"设备管理",单击设备 "aaaaa11111"右边的"调试产品",进入调试界面。

在应用模拟器中,选择服务: StreetLight,命令: SWITCH_LIGHT,命令取值为: ON,单击"立即发送"。

我们可以在"日志"栏看到CIG下发的数据内容:ON,且模拟器会提示"收到平台的下行消息,是否应答?",单击"是",可以在开发中心的应用模拟器看到命令状态为"已送达"。

日志:		
CIG下发的数据内容是: □□ L□		
CIG IP: 100.95.142.231, PORT: 5684		
UE模拟器上报的数据内容是: 1400		
CIG IP: 100.95.142.231, PORT: 5684		
CIG下发的数据内容是: □□ L□	提示	
CIG IP: 100.95.142.231, PORT: 5684	? 收到平台的下行消息,是否应答?	
CIG下发的数据内容是: ON ====================================	 是(Y) 否(N)	-

----结束

进阶体验

按照本页面的指导完成开发中心快速体验,您应该已经基本了解了设备管理服务的在 线开发流程,包括创建项目,开发产品模型、编解码插件和使用模拟器在线调测。

若您想要进一步体验设备管理服务,可参考<mark>开发指南</mark>完成真实应用和设备的对接,使 用真实应用或控制台体验更多功能,控制台的操作指导请参见<mark>使用指南</mark>。

4.2 创建项目(联通用户专用)

非联通用户请查看设备接入服务。

项目是物联网平台提供物联网应用和设备的调测空间,您可以根据场景的不同创建不 同项目空间分别调测。

- 创建项目时,物联网平台会分配一个应用ID(接口调用时参数名为appld)作为项目的唯一标识。若应用服务器需要调用物联网平台的API接口进行一些业务处理,请求中必须携带appld以接入对应的项目空间,鉴权接口携带在Body中,其他接口的appld的值一般携带在Header中。
- 创建项目后,可以在项目中查看应用服务器和设备的接入地址和端口信息,方便 您快速对接应用服务器和设备。
- 项目被删除后项目内的所有资源,如设备、产品、订阅数据在平台中的信息会被 全部删除,并且不可恢复,请谨慎操作。

创建项目

在基于开发中心进行物联网开发时,您需要根据行业属性创建独立的项目,并在该项 目空间内开发物联网产品和应用。

步骤1 在开发中心首页,点击"新建项目"。

最近开发				Rinäs		亜多 >
test 开发中	WaterMeter 开发中	Vehicle 题成中		公告 2018/08/28 20133-43	公告 機械公式的增量因素并必須考於的容 管理可能可要的利用并必须注 る機能力不分加限原性素的生 2018/08/25 16 59:40	公告 《秋秋年1996) 2018/08/25 16:55:57
我的项目						
	十新建项目					

步骤2填写"项目名称"、"所属行业"、"描述"等项目信息后,点击"创建"。"项目 名称"需要保持唯一,不可以和其他项目冲突,否则会创建失败。最多支持创建10个 项目。

新建项目		×
*项目名称		
Project_01		
*所属行业		
公用事业 (5	(Toi-B	-
描述		
	确定	

步骤3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。

\oslash	项目创建成功 在项目内,系统已经 应用ID及密钥信息数 的唯一机会,请受新 如若遗忘密钥,可证 钥"进行重置。 应用ID	¥ 全创建对应的应用,分配的 如下所示,这是您记住密钥 毒保存。 通过"对接信息 > 重置密
I	应用密钥 进入项目	下载密钥

步骤4 选择新创建的项目,可以进入项目空间。在项目空间内,您可以快速完成Profile文件 和编解码插件的在线开发和调试,获取设备和应用的开发帮助,以及通过模拟器完成 设备或应用的对接调试。



----结束

查看项目信息

项目创建完成后,您可以在"**对接信息**"中,查看项目的基本信息,例如**设备接入地** 址、应用接入地址、重置密钥等。

▲ 项目概览	基本信息				
● 产品	▲ 对接信	息详情			
产品开发	主要信息	Ľ		应用安全	
设备管理	应用省称		应用ID	应用密钥 ⊙	
升级调试	Project_0	1		重置索钥	
1 应用	2019/05/1	0 17:17:28	MUTURE 新无	尚无推送证书 证书管理	
对接信息					
订阅调试	设备接入值	(思		行业值息 🖸	
应用调试		接入方式 ③ :5683 (CoAP)	「 搬入方式	所曜行业 公用事业(NB-IoT)	应用能力 ① 存储模式
应用在线开发		1999年(GUAP 3)		12 ob 80-69 70 23 12 80	
Web应用开发	#48.0 #48.0	18943 (HTTPS) 18883 (MQTTS)		7天	
	应用接入体	聰 ①			

删除项目

项目被删除后项目内的所有资源,如设备、产品、订阅数据在平台中的信息会被全部 删除,并且不可恢复,请谨慎操作。

步骤1 在开发中心首页的"我的项目"中,选择需要删除的项目,点击项目右上角的"··· ·"按钮,然后点击"删除"。

我的项目 ⑦	
Project_01 公用事业 (NB-IoT) 创建时间: 2019/09/16 11:17:20 GMT+08 删除	+

步骤2 系统将弹出"删除项目确认"窗口,点击"确认"完成项目删除。

删除项目确	认		
	您确定删除」	此项目吗?	
	确定	取消	

-----结束

4.3 创建产品(联通用户专用)

非联通用户请查看设备接入服务。

在物联网平台中,某一类具有相同能力或特征的设备的合集被称为一款产品。产品包含Profile(产品模型)、编解码插件、测试报告等资源,其中产品信息被记录在Profile中。

• 在一个项目里最多可以创建20个产品。

- 产品开发最重要的是开发Profile,Profile用于描述设备具备的能力和特性。定义 Profile,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支持的 服务、属性、命令等信息。如果设备上报的数据是二进制码流格式,就需要开发 对应的编解码插件,用于物联网平台完成二进制格式和JSON格式的转换。
- 删除产品后,该产品下的Profile、编解码插件等资源将被清空,请谨慎操作。

创建产品

开发中心上提供了多种创建产品的方法。

方式	描述
基于系统模板创建	开发中心提供了多种产品模板,这些模板涉及多个领域。模 板中提供了已经编辑好的Profile文件,您可以根据自己的需 要对Profile中的字段进行修改和增删。有的模板同时提供了 开发好的编解码插件,用户也可以进行修改。
基于快速集成模板 创建	类似系统模板,快速集成也是提供了一些产品模板,这些模 板适用快速体验平台的集成对接,或者适用一些研讨或培训 的学习场合。
基于已有产品创建	您可以基于已有的产品进行克隆,创建新的产品。
本地导入产品创建	将本地写好的Profile文件上传到平台,开发一个新产品。
自定义产品	您可以从零自定义构建产品。

接下来以创建自定义产品为例,全新定义一款产品。

步骤1 在项目空间内,选择"产品 > 产品开发",点击"新建产品"。



步骤2 在"自定义产品"界面,点击"自定义产品"。

产品开发 > 创建产品				
基于系统模板创建	基于快速集成模板创建	基于已有产品创建	请选择以下一种方式创建您的产品 ** ^{把导入产品创建}	
			您可以按照您的需求,目主升发一款产品	
			日定义广告	

- **步骤3**系统将弹出"设置产品信息"窗口,填写"产品名称"、"产品型号"等信息后,点击"创建"。
 - "产品名称"、"型号"需要在项目内保持唯一,否则会创建失败。
 - "所属行业"、"设备类型"、"接入应用层协议类型"和"数据格式"等信息 根据实际情况进行填写。
 - 当"数据格式"配置为"二进制码流"时,该产品下需要进行编解码插件开发; 当"数据格式"配置为"JSON"时,该产品下不需要进行编解码插件开发。



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空间。

▲ 项目概范	🕕 开发中	0	自助测试中		03 认证发布		
● 产品							
产品开发	• 开发中		• 3850 +		• #按中	• E%a	• Esta
设备管理	3		0		0	U	0
升级调试							
₽ 应用	全部产品(3)					Q 请输入产品名称/型号/段	
对接信息							
订阅调试		产品名称	調整	设备类型	数据格式	最后修改时间	
应用调试	e	开发中 WaterMeterD1	NBIoTDevice01	WaterMeter	二进制码流	2019/03/29 15:23:41	
应用在线开发							

----结束

完善产品详情

步骤1 在"产品开发"界面选择需要查看详情的产品,点击该产品右侧"···"按钮,然后点击"查看产品详情",进入"产品详情"界面。

▲ 项目概览	01) 开发中		🕺 自助测试中			03 认证发布		
● 产品								
产品开发	• 开发中		• 264			• 職後中	 已发布 	• Ball
设备管理	3		0			0	0	0
升级调试								
1 应用	全部产品(3)						(2) 清编入产品名称型号设备	(WSI 0 DRM) + (変形)
对接信思								
订阅调试		产品名称		코루	设备类型	数据格式	最后修改时间	
应用调试		开发中 WaterMeter01		NBIoTDevice01	WaterMeter	二进制研究	2019/03/29 15:23:41	兼要产品详细
应用在线开发								209

步骤2 在"产品详情"界面,可以查看产品的"详情介绍"、"产品文件"、"测试报告"、"厂商信息"等。在"详情介绍"界面,可以对"产品简介"、"功能亮点"、"产品规格"等信息进行完善。

▲ 项目概览	产品开发 > 产品详情					
● 产品 🔹	🕵 WaterMe	ter01 III NBIoTDevice01	设备问题: WaterMeter	政编指式:二进制码表	th设周型: CoAP	K
产品开发	开发中	所履行业:智慧城市	厂商名称: ofo	厂闸ID: 2a598d9b8f9440749728b19f	54df11a7d	
设备管理						1. 产品简介
升级调试	详情介绍	1.产品简介*			区编辑	2. 功能死点
₽ 应用 <u>^</u>	产品文件	暫无內容,请编編後加。				3. 产品规格
对接信息	测试报告	2.功能亮点			区编辑	4 82578
订阅调试	厂商信息	暫无內容,请编編添加。				5 XOVA
应用调试	备注	3.产品规格*			区编辑	3. WY 3603
应用在线开发		暫无內容,请编編表加。				
		4.服务支撑			区编辑	
		暂无内容,谦偏继承加。				
		5.客户案例			区编辑	
		智无内容,请编据表加。				

----结束

删除产品

删除产品后,该产品下的Profile、编解码插件等资源将被清空,请谨慎操作。

步骤1 在"产品开发"界面选择需要删除的产品,点击该产品右侧"···"按钮,然后点击"删除"。



步骤2 系统将弹出删除产品确认窗口,点击"确定"完成产品删除。

(!) 删除产品 您确定要删除	¥此产品吗?
确定	取消

----结束

4.4 开发 Profile

4.4.1 什么是 Profile (联通用户专用)

非联通用户请查看设备接入服务。

Profile(即产品模型)是用来描述一款产品中的设备"是什么"、"能做什么"以及 "如何控制该设备"的文件。在物联网平台集成对接中需要先创建Profile,因为 Profile里面定义了设备上报的数据和应用服务器下发的命令包含了哪些字段。定义 Profile,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支持的服 务、属性、命令、升级能力等信息。



Profile主要包含产品信息、服务能力和维护能力三部分。

• 产品信息

描述一款水表设备的基本信息,包括厂商ID、厂商名称、产品类型、型号、协议 类型。其中厂商ID和型号唯一标识一款产品。 例如,某个水表的制造厂商为"HZYB",制造商ID为"TestUtf8Manuld",型号为"NBIoTDevice",协议类型为"CoAP"。

服务能力

服务能力用于描述设备具备的业务能力。将设备业务能力拆分成若干个服务后, 再定义每个服务具备的属性、命令以及命令的参数。

以水表为例,水表具有多种能力,如上报水流、告警、电量、连接等各种数据, 并且能够接受服务器下发的各种命令。Profile文件在描述水表的能力时,可以将 水表的能力划分五个服务,每个服务都需要定义各自的上报属性或命令。说明如 下:

服务名	描述
基础 (WaterMeterBasi c)	用于定义水表上报的水流量、水温、水压等参数,如果需 要命令控制或修改这些参数,还需要定义命令的参数。
告警 (WaterMeterAlar m)	用于定义水表需要上报的各种告警场景的数据,必要的话 需要定义命令。
电池(Battery)	定义水表的电压、电流强度等数据。
传输规则 (DeliverySchedul e)	定义水表的一些传输规则,必要的话需要定义命令。
连接 (Connectivity)	定义水表连接参数。

注:具体定义几个服务是非常灵活的,如上面的例子可以将告警服务拆分成水压 告警服务和流量告警服务,也可以将告警服务合入到水表基础服务中。

维护能力

描述设备具备软固件升级、配置更新等能力。

在线开发和离线开发

Profile的开发手段有在线开发和离线开发两种,我们推荐使用开发中心**在线开发** Profile。

- 在线开发即在开发中心上,通过界面操作开发Profile,开发完成后可以下载。
- 离线开发是指通过了解Profile格式规范,在本地进行开发、打包并上传。

4.4.2 在线开发 Profile (联通用户专用)

非联通用户请查看设备接入服务。

在线创建Profile前需要先创建项目并定义产品。创建产品需要输入manufactureld、 manufactureName、deviceType、Model等信息,Profile会使用这些信息作为设备能 力字段取值。在产品创建时,如果选择使用系统模板,则系统将会自动使用相应的 Profile模板,您可以直接使用或在此基础上进行修改;如果选择自定义产品模板,则 需要完整定义Profile。 本节定义包含一个服务的Profile为示例,该Profile包含设备上报数据、下发命令、下 发命令响应、软固件升级等场景的服务和字段。

步骤1 在"产品开发"界面选择产品,选择具体产品,进入该产品的开发空间。

▲ 项目概览	01) 开发中	02 自助测试中	03 认证发布		
● 产品 🔺					
产品开发	 ● 开发中 	• #8##	• 爾松中	 ● 已发布 	• EK9
设备管理	3	0	0	0	0
升级调试					
■ 应用 🔺	全部产品(3)				(1)(1)(1)(1)(1)(1)(1)(1)(1)(1)(1)(1)(1)(
对接信息					
订阅调试	• P880	겠무	设备关型 数据格式	最后嫁放时间	
应用调试	开发中	WaterMeter01 NBIoTDevice01	WaterMeter 二进制码流	2019/03/29 15:23:41	
******					-

步骤2 在产品开发空间,点击"Profile定义",然后点击"新建服务"。

WaterMeter01	产品型号 ①: NBioTDevice01 所属行业 智慧地市	设备类型①:WaterMeter 厂商名称: HW	数据格式 二进制码流 厂前ⅠD ②: 收起人	协议类型 LWM2M	と病相關性 回臺費产品詳構
1 Profile定义	2 编解码插件开发		3 端倒集成指导	4 在线调测	发起自即测试
維护能力配置					
软件升级 🔵 软件升级协	X PCP		圖件升级 🔵 圖件升级协议		最大
服务列表 ③					十新建服务 也导出Profile ①导入Profile
服务名称 Service ID	描述	5		最后修改时间	
			您还未创建任何服务。		

步骤3 在"新建服务"区域,对服务名称、属性和命令进行定义。每个服务下,可以包含属性和命令,也可以只包含其中之一,请根据此类设备的实际情况进行配置。

퉨	务列表	ŧ ()								十新建服务	出导出Profile	₽l	导入Profile
		服务名称 Service ID		描述					最后修改时间				
						悠	四丕未创建任何服务。						
	^	请输入服务名称		请输入相关微	8							~	×
		属性列表											
		属性名称·Propertyname	数据	类型	范围	步长	单位	枚举值	长度	是否必选	访问模式		
							十新增属性						
		命令列表											
		命令名称·Method											
							十添加命令						

1. 填写"服务名称","服务名称"采用首字母大写的命名方式,比如: WaterMeter、Battery。

∧ Battery 参照人相关系统 又 X

- 2. 点击"添加属性",在弹出窗口中配置属性的各项参数,点击"确定"。
 - 名称:首位必须为字母,建议采用驼峰形式,如batteryLevel、 internalTemperature 。
 - 数据类型:配置可参考如下原则:
 - int: 当上报的数据为整数或布尔值时,可以配置为此类型。

- decimal: 当上报的数据为小数时,可以配置为此类型。配置"经纬度"属性时,数据类型建议使用"decimal"。
- string: 当上报的数据为字符串、枚举值或布尔值时,可以配置为此类型。如果为枚举值或布尔值,值之间需要用英文逗号(",")分隔。
- DateTime:当上报的数据为日期时,可以配置为此类型。
- jsonObject: 当上报的数据为json结构体时,可以配置为此类型。
- · 访问模式:设置应用服务器通过接口访问数据的模式:
 - R:通过接口可以查询该属性。
 - W:通过接口可以修改该属性值。
 - E:应用服务器订阅了数据变化通知后,设备上报了属性,应用服务器会 收到推送通知。
- 是否必选:设备上报的这个属性是不是必选。

新増属性	×
* 名称	
batteryLevel	
* 数据类型	
int	•
*最小值	*最大值
0	100
步长	单位
1	
*访问模式	
✓ R 属性値可读	
✔ W 属性值可写 (更改)	
✓ E 属性值更改时上报事件	
是否必选	
☑ 是	
确定	取消

点击"添加命令",在弹出窗口中配置"命令名称",点击"确定"。"命令名称"首位必须为字母,建议采用全大写形式,单词间用下划线连接的命名方式,如DISCOVERY,CHANGE_STATUS。

添加命令		×
* 命令名称 CHANGE_STATUS		
确定	取消	

 点击"添加下发命令字段",在弹出窗口中配置下发命令字段的各项参数,点击 "确定"。"下发命令字段名称"首位必须为字母,建议采用第一个单词首字母 小写,其余单词的首字母大写的命名方式,比如valueChange;其余参数,请根 据此类设备的实际情况进行配置。

valueChange		
* 数据类型		
int		•
*最小值	* 最大值	
0	1	
步长	单位	
1		
是否必选		
<mark>✓</mark> 是		

5. 如果要添加命令响应,点击"添加响应命令字段",在弹出窗口中配置响应命令 字段的各项参数,点击"确定"。"响应命令字段名称"首位必须为字母,建议 采用第一个单词首字母小写,其余单词的首字母大写的命名方式,比如 valueResult;其余参数,请根据此类设备的实际情况进行配置。

* 名称		
valueResult		
* 数据类型		
int	•	
*最小值	* 最大值	
0	1	
步长	单位	
1		
是否必选		
✓ 是		

步骤4 如果要添加软件/固件升级能力,在"维护能力配置"下,开启"软件升级"/"固件升级"。

维护能力配置					
软件升级 〇〇 软件升级协议 PCP	-	固件升级 🚺 固件升级协议	LWM2M	•	占次

提交后,Profile中会自动多出一个"DM服务"应用于升级。

服务列表	2 (?)								十新建服务	出导出Profile	☆导入Profile
	服务	名称 Service ID	描述					最后修改时间			
^	DM							-			
	属性药	列表									+添加屬性
		属性名称 Property Name	数据类型	范围	步长	单位	枚举值	长度	是否必选	访问模式	
		swVersion	string					256	是	R	
		fwVersion	string					256	是	R	
	命令死	列表									+添加命令
		命令名称 Method									
	^	PUSH_COMMAND									

----结束

4.4.3 离线开发 Profile (联通用户专用)

非联通用户请查看设备接入服务。

Profile本质上就是一个**devicetype-capability.json**文件和若干个**serviceType-capability.json**文件,按照如下目录打包的一个zip包。其中WaterMeter是 deviceType,TestUtf8Manuld是manufactureId,NBIoTDevice是model, WaterMeterBasic/WaterMeterAlarm/Battery是服务名。



所以离线开发profile就是按照Profile编写规则和JSON格式规范在devicetypecapability.json中定义设备能力,在servicetype-capability.json中定义服务能力。因此 离线开发Profile需要熟悉JSON的格式。

由于离线开发Profile文件相对在线开发比较耗时,因此推荐<mark>在线开发Profile(联通用</mark> **户专用**)。

命名规范

在Profile的开发过程中,需要遵循如下命名规范:

- 设备类型(deviceType)、服务类型(serviceType)、服务标识(serviceId)采用单词首字母大写的命名法。例如:WaterMeter、Battery。
- 属性使用第一个单词首字母小写,其余单词的首字母大写的命名法。例如: batteryLevel、internalTemperature。
- 命令使用所有字母大写,单词间用下划线连接的格式。例如:DISCOVERY,CHANGE_COLOR。
- 设备能力描述json文件固定命名devicetype-capability.json。
- 服务能力描述json文件固定命名servicetype-capability.json。
- 厂商ID和型号唯一标识一款产品,故这些信息的组合在不同的Profile文件中不能 重复,且仅支持英文。
- 要注重名称的通用性,简洁性;对于服务能力描述,还要考虑其功能性。例如: 对于多传感器设备,可以命名为MultiSensor;对于具有显示电量的服务,可以命 名为Battery。

Profile 模板

将一款新设备接入到物联网平台,首先需要编写这款设备的Profile。物联网平台提供了一些Profile文件模板,如果新增接入设备的类型和功能服务已经在物联网平台提供

的设备Profile模板中包含,则可以直接选择使用;如果在物联网平台提供的设备 Profile模板中未包含,则需要自己定义。

例如:接入一款水表 ,可以直接选择物联网平台上对应的Profile模板,修改设备型号 标识属性和设备服务列表 。

🛄 说明

物联网平台提供的Profile模板会不断更新,如下表格列举设备类型和服务类型示例,仅供参考。

设备型号识别属性:

属性	Profile中key	属性值
设备类型	deviceType	WaterMeter
厂商ID	manufacturerId	TestUtf8Manuld
厂商名称	manufacturerName	НΖҮВ
型号	model	NBIoTDevice
协议类型	protocolType	СоАР

设备的服务列表

服务描述	服务标识 (serviceld)	服务类型 (serviceType)	选项(option)
水表的基本功能	WaterMeterBasic	Water	Mandatory
告警服务	WaterMeterAlarm	Battery	Mandatory
电池服务	Battery	Battery	Optional
数据的上报规则	DeliverySchedule	DeliverySchedule	Mandatory
水表的连通性	Connectivity	Connectivity	Mandatory

设备能力定义样例

devicetype-capability.json记录了该设备的基础信息:

```
"supportUpgrade":true,
                             "upgradeProtocolType":"LWM2M"
                      },
"configCapability" : {
"supportConfig":true,
"configMethod":"file",
"tofoultConfigFile": {
                              "defaultConfigFile": {
                                 "waterMeterInfo" : {
                                      "waterMeterPirTime" : "300"
                                  }
                             }
                      }
         },
"serviceTypeCapabilities": [
             {
                 "serviceld": "WaterMeterBasic",
"serviceType": "WaterMeterBasic",
"option": "Mandatory"
             },
             {
                 "serviceId": "WaterMeterAlarm",
"serviceType": "WaterMeterAlarm",
                 "option": "Mandatory"
             },
             {
                 "serviceId": "Battery",
"serviceType": "Battery",
                  "option": "Optional"
             },
{
                 "serviceId": "DeliverySchedule",
"serviceType": "DeliverySchedule",
"option": "Mandatory"
             },
{
                  "serviceId": "Connectivity",
                 "serviceType": "Connectivity",
"option": "Mandatory"
             }
        ]
   }
]
```

各字段的解释:

3

字段	子字段		可选/必选	描述
devi ces			必选	包含了一个设备的完整能力信息(根节 点不能修改)。
	manufactur erld		必选	指示设备的制造商ID。
	manufactur erName		必选	指示设备的制造商名称(只允许英 文)。
	model		必选	指示设备的型号,考虑到一款设备下的 多种型号,建议包含字母或数字以保证 可扩展性。
	protocolTyp e		必选	指示设备接入物联网平台的协议类型。 如NB-loT的设备取值为CoAP。
	deviceType		必选	指示设备的类型。

字段	子字段		可选/必选	描述
	omCapabili ty		可选	定义设备的软件升级、固件升级和配置 更新的能力,字段含义详情见下文中 的:omCapability结构描述。
				如果设备不涉及软件/固件升级,本字 段可以删除。
	serviceType Capabilities		必选	包含了设备具备的服务能力描述。
		servic eld	必选	服务的ld,如果设备中同类型的服务类型只有一个则serviceld与serviceType相同,如果有多个则增加编号,如三键开关 Switch01、Switch02、Switch03。
		servic eType	必选	服务类型,与servicetype- capability.json中serviceType字段保持 一致。
		optio n	必选	标识服务字段的类型,取值范围: Master(主服务), Mandatory(必选 服务), Optional(可选服务)。 目前本字段为非功能性字段,仅起到描 述作用。

omCapability结构描述

字段	子字段	可选/ 必选	描述
upgradeCa pability		可选	设备软件升级能力。
	supportUpg rade	可选	true:设备支持软件升级。 false:设备不支持软件升级。
	upgradePro tocolType	可选	升级使用的协议类型,此处不同于设备的 protocolType,例如CoAP设备软件升级协议使 用PCP。
fwUpgrad eCapabilit y		可选	设备固件升级能力。
	supportUpg rade	可选	true:设备支持固件升级。 false:设备不支持固件升级。

字段	子字段	可选/ 必选	描述
	upgradePro tocolType	可选	升级使用的协议类型,此处不同于设备的 protocolType,当前物联网平台仅支持 LWM2M固件升级。
configCap ability		可选	设备配置更新能力。
	supportConf ig	可选	true:设备支持配置更新。 false:设备不支持配置更新。
	configMeth od	可选	file:使用文件的方式下发配置更新。
	defaultConf igFile	可选	设备默认配置信息(Json格式),具体配置信 息由设备商自定义。物联网平台只储存该信息 供下发时使用,不解析处理配置字段的具体含 义。

服务能力定义样例

{

servicetype-capability.json记录了该设备的服务信息:

```
"services": [
   {
      "serviceType": "WaterMeterBasic",
"description": "WaterMeterBasic",
"commands": [
          {
             "commandName": "SET_PRESSURE_READ_PERIOD",
             "paras": [
                {
                    "paraName": "value",
"dataType": "int",
                    "required": true,
                    "min": 1,
"max": 24,
                    "step": 1,
"maxLength": 10,
                    "unit": "hour",
                    "enumList": null
                }
             ],
"responses": [
                {
                    "responseName": "SET_PRESSURE_READ_PERIOD_RSP",
                     "paras": [
                       {
                          "paraName": "result",
"dataType": "int",
                           "required": true,
                           "min": -1000000,
                           "max": 1000000,
                           "step": 1,
"maxLength": 10,
                           "unit": null,
                           "enumList": null
                       }
```

```
]
                        }
                  ]
              }
         ],
"properties": [
              {
                   "propertyName": "registerFlow",
"dataType": "int",
"required": true,
                   "min": 0,
"max": 0,
                   "step": 1,
                   "maxLength": 0,
"method": "R",
"unit": null,
"enumList": null
              },
{
                   "propertyName": "currentReading",
                   "dataType": "string",
"required": false,
                   "min": 0,
                   "max": 0,
                   "step": 1,
"maxLength": 0,
"method": "W",
                   "unit": "L",
"enumList": null
              },
              {
                   "propertyName": "timeOfReading",
"dataType": "string",
"required": false,
                   "min": 0,
"max": 0,
"step": 1,
                   "maxLength": 0,
"method": "W",
"unit": null,
                   "enumList": null
              },
               .....
         ]
    }
]
```

各字段的解释:

}

字段	子字	段		必 选/ 可选	描述
serv ices				必选	包含了一个服务的完整信息(根节点不可 修改)。
	ser vic eTy pe			必选	指示服务的类型,与devicetype- capability.json中serviceType字段保持一 致。

字段	子字	段		必 选/ 可选	描述
	des cri pti on			必选	指示服务的描述信息。 非功能性字段,仅起到描述作用,可置为 null。
	co m ma nds			必选	指示设备可以执行的命令,如果本服务无 命令则置null。
		com man dNa me		必选	指示命令的名字,命令名与参数共同构成 一个完整的命令。
		para s		必选	命令包含的参数。
			para Nam e	必选	命令中参数的名字。
			dataT ype	必选	指示命令参数的数据类型。 取值范围: string、int、string list、 decimal、DateTime、jsonObject、 enum、boolean。 上报数据时,复杂类型数据格式如下: • string list: ["str1","str2","str3"] • DateTime: yyyyMMdd' T' HHmmss' Z' 如:
					 jsonObject: 自定义json结构体,物联 网平台不解析,仅进行透传
			requir ed	必选	指示本命令是否必选,取值为true或 false,默认取值false(非必选)。 目前本字段是非功能性字段,仅起到描述 作用。
			min	必选	指示最小值。 仅当dataType为int、decimal时生效。
			max	必选	指示最大值。 仅当dataType为int、decimal时生效。
			step	必选	指示步长。 暂不使用,填0即可。

字段	子字	段			必 选/ 可选	描述
			maxL ength		必选	指示字符串长度。 仅当dataType为string、string list、 DateTime时生效。
			unit		必选	指示单位,英文。 取值根据参数确定,如: 温度单位: "C"或"K" 百分比单位: "%" 压强单位: "Pa"或"kPa"
			enum List		必选	指示枚举值。 如开关状态status可有如下取值: "enumList":["OPEN","CLOSE"] 目前本字段是非功能性字段,仅起到描述 作用,建议准确定义。
		resp onse s			必选	命令执行的响应。
			respo nseN ame		必选	命名可以在该responses对应命令的 commandName后面添加"_RSP"。
			paras		必选	命令响应的参数。
				pa ra Na m e	必选	命令中参数的名字。
				da ta Ty pe	必选	指示数据类型。 取值范围: string、int、string list、 decimal、DateTime、jsonObject 上报数据时,复杂类型数据格式如下: • string list: ["str1","str2","str3"] • DateTime: yyyyMMdd' T' HHmmss' Z' 如: 20151212T121222 • jsonObject: 自定义json结构体,物联 网平台不解析,仅进行透传

字段	子字	段		必 选/ 可选	描述
			re qu ire d	必选	指示本命令响应是否必选,取值为true或 false,默认取值false(非必选)。 目前本字段是非功能性字段,仅起到描述 作用。
			mi n	必选	指示最小值。 仅当dataType为int、decimal时生效,逻 辑大于等于。
			m ax	必选	指示最大值。 仅当dataType为int、decimal时生效,逻 辑小于等于。
			ste p	必选	指示步长。 暂不使用,填0即可。
			m ax Le ng th	必选	指示字符串长度。 仅当dataType为string、string list、 DateTime时生效。
			un it	必选	指示单位,英文。 取值根据参数确定,如: 温度单位: "C"或"K" 百分比单位: "%" 压强单位: "Pa"或"kPa"
			en u m Lis t	必选	指示枚举值。 如开关状态status可有如下取值: "enumList":["OPEN","CLOSE"] 目前本字段是非功能性字段,仅起到描述 作用,建议准确定义。
	pro per ties			必选	上报数据描述,每一个子节点为一条属 性。
		prop erty Nam e		必选	指示属性名称。

字段	子字段	ł		必 选/ 可选	描述
	C T	data Type		必选	指示数据类型。 取值范围: string、int、string list、 decimal、DateTime、jsonObject 上报数据时,复杂类型数据格式如下: • string list: ["str1","str2","str3"] • DateTime: yyyyMMdd' T' HHmmss' Z' 如: 20151212T121222 • jsonObject: 自定义json结构体,物联 网平台不解析,仅进行透传
	r	requi red		必选	指示本条属性是否必选,取值为true或 false,默认取值false(非必选)。 目前本字段是非功能性字段,仅起到描述 作用。
	r	min		必选	指示最小值。 仅当dataType为int、decimal时生效,逻 辑大于等于。
	r	max		必选	指示最大值。 仅当dataType为int、decimal时生效,逻 辑小于等于。
	S	step		必选	指示步长。 暂不使用,填0即可。
	r t	met nod		必选	指示访问模式。 R:可读;W:可写;E:可订阅。 取值范围:R、RW、RE、RWE 、null。
	L	unit		必选	指示单位,英文。 取值根据参数确定,如: 温度单位: "C"或"K" 百分比单位: "%" 压强单位: "Pa"或"kPa"
	r L t	max Leng th		必选	指示字符串长度。 仅当dataType为string、string list、 DateTime时生效。

字段	子字	段		必 选/ 可选	描述
		enu mLis t		必选	指示枚举值。 如电池状态(batteryStatus)可有如下取 值: "enumList":[0, 1, 2, 3, 4, 5, 6] 目前本字段是非功能性字段,仅起到描述 作用,建议准确定义。

Profile 打包

Profile写作完成后,需要按如下层级结构打包:



Profile打包需要遵循如下几点要求:

- Profile文件的目录层级结构必须如上图所示,不能增删。例如:第二层级只能有 "profile"和"service"两个文件夹,每个服务下面必须包含"profile"文件夹 等。
- 图中橙色字体的命名不能改动。
- Profile文件以zip形式压缩。
- Profile文件的命名必须按照deviceType_manufacturerId_model的格式命名,其中的deviceType、manufacturerId、model必须与devicetype-capability.json中对应字段的定义一致。例如:本实例中devicetype-capability.json的主要字段如下:

```
{
    "devices": [
        {
            "manufacturerId": "TestUtf8ManuId",
            "manufacturerName": "HZYB",
            "model": "NBIoTDevice",
            "protocolType": "CoAP",
            "deviceType": "WaterMeter",
            "serviceTypeCapabilities": ****
        }
    ]
}
```

• 图中的WaterMeterBasic、WaterMeterAlarm、Battery等都是devicetypecapability.json中定义的服务。

Profile文件中的文档格式都是JSON,在编写完成后可以在互联网上查找一些格式校验 网站,检查JSON的合法性。

4.4.4 导出和导入 Profile

联通用户专用,非联通用户请查看设备接入服务。

Profile作为一种资源,可以从物联网平台导出,也可以导入到物联网平台。

- 当产品开发完成并测试验证后,需要将在线开发的Profile移植时,则可以将 Profile导出到本地。
- 当您已经有完备的Profile时(线下开发或从其他项目/平台导出),可以将Profile 直接导入到开发中心或"设备管理服务控制台"。

导出 Profile

当产品开发完成并测试验证后,需要将在线开发的Profile移植时,则可以将Profile导出到本地。

- 在开发中心的"Profile定义"界面导出
 - a. 在开发中心的具体项目中,选择"产品 > 产品开发",单击进入某个产品, 在"Profile定义"界面,点击"导出Profile"。

01 Profile定义) 编解码插件开发	- (03) 端侧集成指导	(04)	在线调测 ————————————————————————————————————	发起自动测试
OM维护					
() 软件升级	国件升级				
软件开级协议:	圈件升级协议:				
PCP	LWM2M	提交			
服务列表 ①				十新建服务	と身出Profile
服务名称 描述			最后修改时间	摄作	
V BusinessService			2019/03/29 11:49:35		

- b. 选择Profile的存放路径,将Profile下载到本地。
- 在开发中心的"产品详情"界面导出
 - a. 在开发中心的具体项目中,选择"产品 > 产品开发",点击产品右侧"··· ·"按钮,然后点击"查看产品详情",进入"产品详情"界面。



b. 在"产品详情"界面,在右侧相关资料区域,点击"下载",将Profile下载 到本地。

▲ 项日概览		产品开发 > 产品评情						
0 Fil		WaterMeter01	产品带号 ① NBlaTDevice91	设备部型①:WaterMater	FORMAT: - MANDA	1010/061	I I WADM	
产品开发			ATRICUL WINNER	厂度名称: HW	F800			13 编辑题性
设备管理								
升级调试		产品就介 功能观点 产品	品规格 服务支撑 套户案例				相关资料	
■ 成用 対接信息	•	产品简介。				C 4948	Profile文件	58 78

导入 Profile

当您已经有完备的Profile时(线下开发或从其他项目/平台导出),可以将Profile直接 导入到开发中心或"设备管理服务控制台"。

- 在开发中心导入
 - a. 在开发中心的具体项目中,选择"产品 > 产品开发",单击进入某个产品, 在"Profile定义"界面,点击"导入Profile"。

01 Profile定义 ······	• 02	编解码插件开发 ————————————————————————————————————	(3) site	继成指导 ——		04) 在线洞测 ————————————————————————————————————		2	站自动测试
OM维护									
() 软件开级		國性升级							
软件升级协议:		圈件升级协议:							
PCP		LWM2M		提交					
服务列表 ③							十级建服务	出导出Profile	☆每入Profile
服务名称	描述				最后修改时间		攝作		
			您还未创建	任何服务。					

b. 在弹窗中选择Profile文件后,点击"上传"。在选择"资源文件"后,"设备类型"、"厂家ID"、"设备型号"将会根据Profile文件的命名自动填充,且需要和创建产品时填写的信息保持一致。

*设备类型 WaterMeter
*设备类型 WaterMeter
WaterMeter
*厂家ID caa2dd65168543d49aa1b70e4d443d0e
•设备型号

- 在"设备管理服务控制台"导入
 - a. 登录"设备管理服务控制台",单击左下角的 切换左侧菜单,打开 "产品模型"页面,点击页面右上角的"新增产品模型 > 本地导入"。

产品模型							+ 新增产品模
Q 名称/型号/it 搜索结果: 1	2备关型/	搜索					
	产品名称 StreetLight_hw_SL01	型号 SL01	设备类型 StreetLight	厂商名称 hw	协议类型 CoAP	创建时间 2019-05-06 15:18:15	详情 删除
							10条/页 > (1 >

b. 在弹窗中输入产品名称并选择要上传的Profile文件,单击"确定",导入 Profile文件。

产品名称		
产品名称长属	度限制为4-32。	
资源文件		
文件大小不能	舀过4M,且必须为zip文件。	企
兼容旧版本的设	新模型	

4.5 开发编解码插件

4.5.1 什么是编解码插件(联通用户专用)

非联通用户请查看设备接入服务。

一款产品的设备上报数据时,如果"数据格式"为"二进制码流",则该产品需要进 行编解码插件开发;如果"数据格式"为"JSON",则该产品下不需要进行编解码插 件开发。

以NB-IoT场景为例,NB-IoT设备和物联网平台之间采用CoAP协议通讯,CoAP消息的 payload为应用层数据,应用层数据的格式由设备自行定义。由于NB-IoT设备一般对 省电要求较高,所以应用层数据一般不采用流行的JSON格式,而是采用二进制格式。 但是,物联网平台与应用侧使用JSON格式进行通信。因此,您需要开发编码插件,供 物联网平台调用,以完成二进制格式和JSON格式的转换。



数据上报流程



在数据上报流程中,有两处需要用到编解码插件:

- 将设备上报的二进制码流解码成JSON格式的数据,发送给应用服务器。
- 将应用服务器响应的JSON格式数据编码成二进制码流,下发给设备。
命令下发流程



在命令下发流程中,有两处需要用到编解码插件:

- 将应用服务器下发的JSON格式数据编码成二进制码流,下发给设备。
- 将设备响应的二进制码流解码成JSON格式的数据,上报给应用服务器。

在线开发和离线开发

编解码插件的开发手段有在线开发和离线开发两种,由于插件离线开发较为复杂,且 耗时比较长,我们**推荐**使用开发中心**在线开发编解码插件**。

- 在线开发是指借助开发中心,通过可视化的方式快速开发一款产品的编解码插件。
- <mark>离线开发</mark>是指使用编解码插件的Java代码Demo进行二次开发,实现编解码功能、 完成插件打包和质检等。

4.5.2 在线开发插件(联通用户专用)

非联通用户请查看设备接入服务。

借助开发中心,我们可以通过可视化的方式快速开发一款产品的编解码插件。在自定 义新建产品时,如果选择使用系统模板,部分模板会包含编解码插件,您可以直接使 用或在此基础上进行修改;如果选择自定义产品模板,则需要完成编解码插件的开 发。 编解码插件的开发需要基于Profile定义的设备能力进行开发,本节首先以一个烟感设备的例子讲解如何开发一个支持数据上报和命令下发的编解码插件,然后再以四个场 景举例说明如何完成复杂的插件开发以及调试。

- 数据上报和命令下发
- 多条数据上报消息
- 字符串及可变长字符串的编解码插件在线开发
- 数组及可变长数组数据类型
- 含命令执行结果的编解码插件

数据上报和命令下发

场景说明

有一款烟感设备,具有如下特征:

- 具有烟雾报警功能(火灾等级)和温度上报功能。
- 支持远程控制命令,可远程打开报警功能。比如火灾现场温度,远程打开烟雾报警,提醒住户疏散。

Profile定义

在烟感产品的开发空间,完成Profile定义。

- level:火灾级别,用于表示火灾的严重程度。
- temperature:温度,用于表示火灾现场温度。
- SET_ALARM: 打开或关闭告警命令,value=0表示关闭,value=1表示打开。

	服务名称	描述		最后修改时间	提倡
^	Smoke			2018/09/17 10:25:15	î
	属性列表				+ 添加属性
	+> level	数据关型 范围 步长 int 0~3	单位 	是否必选 访问模式 ✓ R	2
	↔ temperature	数据类型 范围 步长 int 0~1000	单位 	是否必选 访问模式 ✓ R	
	命令列表				+ 添加命令
	∧ ♦ SET_ALARM				2
	下发命令宇段				+ 添加下发字段
	value	数据类型 范围 # int 0~3 -	·K 単位 	是否必选	C i
	响应命令宇段				+ 添加响应字段
			您还未创建任何响应命令。		

编解码插件开发

步骤1 在烟感产品的开发空间,选择"编解码插件开发"。

步骤2 配置数据上报消息。

新增消息		×
基本信息		
	演員描述	
字段		
	+ 添加字段	
	完成 取消	

添加level字段,表示火灾级别。

- "字段名"只能输入包含字母、数字、_和\$,且不能以数字开头的字符。
- "数据类型"根据设备上报数据的实际情况进行配置,需要和Profile相应字段的 定义相匹配。
- "长度"和"偏移值"根据"数据类型"的配置自动填充。

 ● 标记为地址域 ② *名本 level 細还 >>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>	添加字段				×
	□ 标记为地址域 ⑦)			
level 編述 就研究型 int8u(8位无符号整型) ・ 长度 ① 1 繁以值 ② 佛彩值 ③ 0 0-1	*名字				
無述 数据地型 int8u(8位无符号整型) * 松度 ⑦ 1 数以值 ⑦ 傳移值 ⑦ 0-1	level				
数据映型 int8u(8位无符号整型) ・ ・ K度 ① 1 数以值 ① 備序値 ① 0-1	描述				
款法地型 int8u(8位无符号整型) ・ * K度 ③ 1 数以值 ⑤ 何・1					
数据映型 int8u(8位元符号整型) ・ ・ 长度 ○ 1 数以值 ⊙ 命移值 ⊙ 0-1					
該無受型 int8u(8位元符号整型) ・ ・ K度 ⑦ 1 默以值 ⑦ の-1					
int8u(8位元符号整型) ・ * 长度 ① 1 默认值 ① 師邸値 ① 0-1	数据类型				
 · 长度 ③ 1 数认值 ③ (689) ④ (6-1) 	int8u(8位无符	号整型)		•	
1 默认值 ① 编移值 ① 0-1	* 长度 ⑦				
默认值 ⊙ 编移值 ⊙ 0-1	1				
m标值 ① 0-1	默认值 ⑦				
编修值 ⑦ 0-1					
0-1	偏移值 ⑦				
	0-1				
完成 取消		完成	取消		

添加temperature字段,表示温度。在Profile中,temperature属性最大值1000,因此 在插件中定义temperature字段的"数据类型"为"int16u",以满足temperature属 性的取值范围。

标记为地址域 ⑦	
名字	
temperature	
苗述	
#b12346开I	
100 A 200	
int16u(16位无符号整型)	•
* K	
2	
影り値の	
偏移值 ②	
1.2	
1-3	

步骤3 配置命令下发消息。

新增消息		×
基本信息		
消息 名。	減意調送	
SET_ALARM		
 * 消息类型 ○ 数据上报 ● 命令下发 		
一汤加明应字段		
字段		
	十 添加字段	
7 532	取到時	

添加value字段,表示下发命令的参数值。

添加字段			
□ 标记为地址域	2		
*名字			
value			
描述			
数据类型			
int8u(8位升	符号整型)		•
* / # @			
1			
默认值 ⑦			
偏移值 ⑦			
0-1			
	完成	取消	

步骤4 拖动右侧"设备模型"区域的属性字段和命令字段,数据上报消息和命令下发消息的 相应字段建立映射关系。

		设备模型
c i	F level	Smoke
mokeinfo	Smoke	居住
《息洗型: deviceReq 是否包含酮应消息: No 新述:	smoke	¢
效据上报字段	+	¢
level		
temperature		
• ć	Stralapm	
ET_ALARM		
调类型: cloudReq 语包含响应消息: No		详情
iž:-		Smoke
命令下发字段 -	+	
1 value		

步骤5 点击"保存",并在插件保存成功后点击"部署",将编解码插件部署到物联网平 台。

01) Profile定义	02	编解码插件开发	04 在线调测	发起自动测试
在线编解码插件编辑器	插件管理			🖹 保存 🔺 部署 📃 更多

----结束

调测编解码插件

步骤1 在烟感产品的开发空间,选择"在线调测",使用虚拟设备调试编解码插件。

01 Profile定义		(02) 编解码插件开发 ————————————————————————————————————	(03) 端倒集成指导		— 04 <u>在线调测</u>	发起自助测试
设备列表 💿						十新墙測试设备
状态	设备名称	设备ID	所屬产品	型号	设备类型	操作
			智无任何设备			

勾选"没有真实的物理设备",点击"创建"。

新增测试设备	×
您现在 〇 有真实的物理设备 ② 有真实的物理设备 ③ 没有真 	实的物理设备
您正在注册一个虚拟的设备	
创建	取消

步骤2 使用设备模拟器进行数据上报。十六进制码流示例: 02013A。02表示火灾级别,长度 为1个字节;013A表示温度,长度为2个字节。

在"应用模拟器"区域查看数据上报的结果: {level=2, temperature=314}。2为十六进制数02转换为十进制的数值; 314为十六进制数013A转换为十进制的数值。

	命令下发	
▲ 应用模拟器	0≯	©≫loT平台
	* o	оо.от н
<u>人刻</u> 数据技巧 令人失兴	数据上报	▲ 0
王即 数据接收 叩令反达		2 1 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
数据接收		
7018-00-17 17:18:75		
(level=2, temperature=314)		
		() 设备模拟器
		全部 数据发送 命令接收
		数据 5 法
		2018-09-17 14:18:44
		02013A
服务 Smoke ▼		
命令 SET_ALARM ▼		
value * 輸入的整数范围[0, 1]		
		请输入十六进制码流
		02013A
缓存发送 立即发送 · · · · · · · · · · · · · · · · · · ·		自动发送发送
2011 2011 2011 2011 2011 2011 2011 2011		

步骤3 使用应用模拟器进行命令下发: { "serviceId": "Smoke", "method": "SET_ALARM", "paras": "{\"value\":1}" }。

在"设备模拟器"区域查看命令接收的结果:01。01为十进制数1转换为十六进制的数 值。

	命令下发	
▲ 应田焟圳哭	0>	
一世而失财命	+o	
	数据上报	▲ 0
全部 数据接收 命令友运		1 1 数据 上掲 1 1
命令发送: 2018-09-17 14:34:35		
安洋洋自创,信白: / "timo": "Mon Son 17 1/:2/:25		i i 前令卜友 0 ▼
GMT+08:00 2018", "requestid": "bc367ab4-14da-8302-		
a50d-00bd0617ce7b_7669", "callbackUrl": null,		() 设备模拟器
"expireTime": 0, "command": { "serviceId": "Smoke",		
"method": "SET_ALARM", "paras": "{\"value\":1}" } }		全部 数据发送 命令接收
发送消息body信息: { "serviceId": "Smoke", "method":		命令接收
"SET_ALARM", "paras": "{\"value\":1}" }		2018-09-17 14:34:35
		01
照冬 Smoke		
Sinoke -		
命会 SET ALARM ▼		
value ~ 输入的整数池围[U,1]		
1		
		清諭 》十六进制码法
援存发送 立即发送 · · · · · · · · · · · · · · · · · · ·		自动发送发送
And		

----结束

多条数据上报消息

如果该烟感设备需要支持同时上报烟雾报警(火灾等级)和温度,也支持单独上报温度,则按照以下步骤创建消息。

编解码插件开发

步骤1 在烟感产品的开发空间,选择"编解码插件开发"。

步骤2 配置数据上报消息,上报火灾等级和温度。

新增消息		×
基本信息		
满息名。 smokeinfo	消息描述	
 消息共型 ● 数据上报 ○ 命令下发 		
□ 添加响应字段		
字段	1.254-56	
	T NUMITR	
	为此成 取消	

添加messageld字段,表示消息种类。

- 在本场景中,数据上报消息有两种,所以需要用messageld来标识消息种类。
- "数据类型"根据数据上报消息种类的数量进行配置。在本场景中,仅有两种数据上报消息,配置为"int8u"即可满足需求。
- "默认值"可以修改,但必须为十六进制格式,且数据上报消息的对应字段必须 和默认值保持一致。在本场景中,用0x0标识上报火灾等级和温度的消息。

が川ナゼ	
✓ 标记为地址域 ⑦	
* 名字 当标记为地址域时,名字固定为messageld;否则,名字不能设置为messageld。	
messageId	
備述	
数据类型	
int8u(8位无符号整型)	•
* 长度 ⑦	
1	
0x0	
偏移值 ⑦	
0-1	

添加level字段,表示火灾级别。

- "字段名"只能输入包含字母、数字、_和\$,且不能以数字开头的字符。
- "数据类型"根据设备上报数据的实际情况进行配置,需要和Profile相应字段的 定义相匹配。
- "长度"和"偏移值"根据"数据类型"的配置自动填充。

添加字段		×
□ 标记为地址域 ⑦		
* 名字		
level		
描述		
数据类型		
int8u(8位无符号整型)		•
* 长度 ②		
1		
默认值 ⑦		
偏移值 ⑦		
1-2		
完成	取消	

添加temperature字段,表示温度。在Profile中,temperature属性最大值1000,因此 在插件中定义temperature字段的"数据类型"为"int16u",以满足temperature属 性的取值范围。

添加字段	×
标记为地址域 ②	
*名字	
temperature	
描述	
新信光田	
如病史型 int16u(16位无符号整型)	•
* 长度 ⑦	
2	
默认值 ②	
偏移值 ③	
2-4	
完成	取消

步骤3 配置数据上报消息,只上报温度。

新增消息		×
基本信息		
消息名。 <u>temperature</u> ・	消息描述	
字段		
	+ 添加字段	
stat	取消	

添加messageld字段,表示消息种类。在本场景中,用0x1标识只上报温度的消息。

✔ 标记为地址域 ⑦	
名字 当标记为地址域时,名字固定为messageld;否则,名字不能设置为messageld。	
messageId	
笛述	
int&u(&位于22中裁刑)	•
int8u(8位无符号整型)	•
int8u(8位无符号整型) * 长度 ⑦	·
int8u(8位无符号整型) * 长度 ⑦ 1	•
int8u(8位无符号整型) * 长度 ⑦ 1	•
<pre>int8u(8位无符号整型) * 长度 ⑦ 1 * 默认值 ⑦ </pre>	•
int8u(8位无符号整型) * 长度 ⑦ 1 I Ox1	•
<pre>int8u(8位无符号整型) * 长度 ⑦ 1 * 默认值 ⑦ Ox1 </pre>	•
int8u(8位无符号整型) * 长度 ② 1 * 默认值 ③ のx1 のx1	•
int8u(8位无符号整型) ・ 长度 ⑦ 1 1 SUMPPER	•

添加temperature字段,表示温度。

添加子段			×
- 标记为地址域	0		
*名字			
temperature	9		
描述			
数据类型			
数据类型 int16u(16位	无符号整型)		•
数据类型 int16u(16位	无符号整型)		•
数据类型 int16u(16位 * 长度 ⑦ 2	无符号整型)		T
数据类型 int16u(16位 * 长度 ⑦ 2	无符号整型)		•
数据类型 int16u(16位 * 长度 ⑦ 2 默认值 ⑦	无符号整型)		•
数据类型 int16u(16位 * 长度 ⑦ 2 默认值 ⑦	无符号整型)		•
数据类型 int16u(16位 * 长度 ⑦ 2 默认值 ⑦	无符号整型)		▼
数据类型 int16u(16位 * 长度 ⑦ 2 默认值 ⑦ 偏移值 ⑦ 1-3	无符号整型)		▼
数据美型 int16u(16位 * 长度 ⑦ 2 默认值 ⑦ 偏移值 ⑦ 1-3	无符号整型)		▼

步骤4 配置命令下发消息。

新增消息	×
基本信息	
消息名 *	消息描述
SET_ALARM	
* 消息交型 ② 数据上报 ④ 命令下发	
一汤加响应李段	
字段	
	+ 添加字段
完成	取び神道

添加value字段,表示下发命令的参数值。

添加字段		
标记为地址域	3	
*名字		
value		
描述		
数据类型		
数据类型 int8u(8位无	(符号整型)	Ţ
数据类型 int8u(8位无 * 长度 ⑦	行号整型)	•
数据类型 int8u(8位ス * 长度 ⑦ 1	符号整型)	•
数据类型 int8u(8位大 * 长度 ⑦ 1 默认值 ⑦	符号整型)	•
数据类型 int8u(8位元 * 长度 ⑦ 1 默认值 ⑦	5符号整型)	▼
数据类型 int8u(8位元 * 长度 ⑦ 1 默认值 ⑦	5符号整型)	▼
数据类型 int8u(8位元 * 长度 ⑦ 1 默认值 ⑦ 偏移值 ⑦	5符号整型)	▼
数据类型 int8u(8位元 * 长度 ⑦ 1 默认值 ⑦ 偏移值 ⑦ 0-1	5符号整型)	
数据类型 int8u(8位元 * 长度 ⑦ 1 默认值 ⑦ 偏移值 ⑦ 0-1	5符号整型)	

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

level字段和temperature字段分别与Profile中的对应属性建立映射关系,messageld用 于帮插件识别消息种类,不需要建立映射关系。

			设备模型
		laval	Smoke
- keinfo		Smoke	扇性
型: deviceReq (含明应消息: No 		Smoke	ire
上报字段	+		¢
messadeld			
evei			
.emperature			
1	a	Contraction temperatu	ire
erature 趣: deviceReq			详情
,含啊应消息: No -			Smoke
上报字段	+		
messageld			
temperature			
i i	8		
ALARM			vi
型: cloudReq 含啊应消息: No			
-			
下发字段	+		

步骤6点击"保存",并在插件保存成功后点击"部署",将编解码插件部署到物联网平 台。

01) Profile定义	02	\$	18指导	发起目盼潮试
在线编解码插件编辑器	插件管理			〇 保存 合 部署 三 更多

----结束

调测编解码插件

步骤1 在烟感产品的开发空间,选择"在线调测",使用虚拟设备调试编解码插件。

01 Profile定义		(02) 编解码插件开发 ————————————————————————————————————	(03) 靖國集成指导 ————		— 04 <u>在线调测</u>	发起自动测试
设备列表 💿						十新墙测试设备
状态	设备名称	设备ID	所屬产品	型号	设备类型	操作
			智无任何设备			

勾选"没有真实的物理设备",点击"创建"。

新增测试设备	×
您现在 〇 有真实的物理设备 ○ 没有真实的 	的物理设备
您正在注册一个虚拟的设备	
创建	取消

步骤2 使用设备模拟器进行数据上报。

十六进制码流示例:000100F1。00表示messageld,此消息上报火灾等级和温度;01 表示火灾级别,长度为1个字节;00F1表示温度,长度为2个字节。

在"应用模拟器"区域查看数据上报的结果: {level=1, temperature=241}。1为十六进制数01转换为十进制的数值; 241为十六进制数00F1转换为十进制的数值。

服务 Smoke ▼ 命令 SET_ALARM ▼ value * 输入的整数范围[0,1]	
value * 输入的整数范围[0 , 1]	
<u> </u>	请输入十六进制码流 000100F1 □ 自动发送 发送

十六进制码流示例:0100F1。01表示messageld,此消息只上报火灾温度;00F1表示 温度,长度为2个字节。

在"应用模拟器"区域查看数据上报的结果:{temperature=241}。241为十六进制数00F1转换为十进制的数值。

	命令下发	
▲ 应用塔彻器	0>	
一些世界政府	40	©©101+-Д
	教権上掲	
全部 数据接收 命令发送	XILLAK	A 0
		数据上报
数据接收		
2018-09-17 17:33:59		命令下发
{temperature=241}		0 🔻
		(•) 政备模拟器
		全部 数据发送 命令接收
		数据发送
		2018-09-17 17:33:59
		0100F1
服务 Smoke ▼		
PP SET_ALARIVI		
value * 榆入的整数范围[0_1]		
		请输入十六进制码流
		010051
		010011
发行友送 立即发送		目动发送

步骤3 使用应用模拟器进行命令下发: { "serviceId": "Smoke", "method": "SET_ALARM", "paras": "{\"value\":1}" }。

在"设备模拟器"区域查看命令接收的结果:01。01为十进制数1转换为十六进制的数 值。

	命令下发	
▲ 应田焟圳哭	0>	
一世而失财命	+o	
	数据上报	▲ 0
全部 数据接收 命令友运		1 1 数据 ト招 コー
命令发送: 2018-09-17 14:34:35		
安洋洋自创,信白: / "timo": "Mon Son 17 1/:2/:25		i i 前令卜友 0 ▼
GMT+08:00 2018", "requestid": "bc367ab4-14da-8302-		
a50d-00bd0617ce7b_7669", "callbackUrl": null,		() 设备模拟器
"expireTime": 0, "command": { "serviceId": "Smoke",		
"method": "SET_ALARM", "paras": "{\"value\":1}" } }		全部 数据发送 命令接收
发送消息body信息: { "serviceId": "Smoke", "method":		命令接收
"SET_ALARM", "paras": "{\"value\":1}" }		2018-09-17 14:34:35
		01
照冬 Smoke		
Sinoke -		
命会 SET ALARM ▼		
value ~ 输入的整数池围[U,1]		
1		
		清諭 》十六进制码法
援存发送 立即发送 · · · · · · · · · · · · · · · · · · ·		自动发送发送
And		

----结束

字符串及可变长字符串的编解码插件在线开发

如果该烟感设备需要支持描述信息上报功能,描述信息支持字符串和可变长度字符串 两种类型,则按照以下步骤创建消息。

Profile定义

在烟感产品的开发空间完成Profile定义。

	服务	名称	描述				最后	修改时间		操作		
^	Smo	ke					2018	3/09/17 10:25:15				i.
	属性	生列表									+ 添加属性	
	÷	level	数据类型 int	范围 0~3	步长 	单位 		是否必选 ✓	访问模式 R		Ľ i	
	÷	temperature	数据 类型 int	范围 0~1000	步长 	单位 		是否必选 ✓	访问模式 R		Ľ i	
	÷	other_info	数据类型 string	长度 100	单位 			是否必选 <mark>✓</mark>	访问模式 R		Ľ 🛙	

编解码插件开发

步骤1 在烟感产品的开发空间,选择"编解码插件开发"。

01) Profile定义	02 编解码插件开发	- (04) 在线调测	发起自动测试

步骤2 配置数据上报消息,上报字符串类型的描述信息。

新增消息		×
基本信息		
Ther_info ・ 滿意送型 ④ 数据上报 ○ 命令下发	道思描述	
液加响应李段		
字段		
	十 添加字段	
	完成 取消	

添加messageld字段,表示消息种类。在本场景中,0x0用于标识上报火灾等级和温度的消息,0x1用于标识只上报温度的消息,0x2用于标识上报描述信息(字符串类型)的消息。

净川子段 ————————————————————————————————————	
✔ 标记为地址域 ⑦	
* 名字 当标记为地址域时,名字固定为messageld;否则,名字不能设置为messageld。	
messageId	
描述	
数据类型	
int8u(8位无符号整型)	•
int8u(8位无符号整型)	•
int8u(8位无符号整型) * 长度 ⑦ 1	•
int8u(8位无符号整型) * 长度 ⑦ 1	•
<pre>int8u(8位无符号整型) * 长度 ⑦ 1 * 默认值 ⑦</pre>	•
 int8u(8位无符号整型) * 长度 ⑦ 1 * 默认值 ⑦ 0x2 	▼
 int8u(8位无符号整型) * 长度 ⑦ 1 * 默认值 ⑦ 0x2 	▼
 int8u(8位无符号整型) 长度 ⑦ 1 * 默认值 ⑦ 0x2 偏移值 ⑦ 0-1 	•
int8u(8位无符号整型) * 长度 ⑦ 1 * 默认值 ⑦ 0x2 偏移值 ⑦ 0-1	▼

添加other_info字段,表示字符串类型的描述信息。在本场景中,"长度"配置 6个字节。

添加字段		×
标记为地址域 ⑦		
*名字		
other_info		
描述		
14422		
※/·IC·2년 편]		
string(字符串类型)		•
* 长度 ⑦		
6		
默认值 ②		
偏移值 ⑦		
偏移值 ⑦ 1-7		
偏移值 ⑦ 1-7		

步骤3 配置数据上报消息,上报可变长度字符串类型的描述信息。

Ŷ	所增消息	×
	基本信息	
	時息名・	
	字段	
	+ 添加字段	
	完成	

添加messageld字段,表示消息种类。在本场景中,0x0用于标识上报火灾等级和温度的消息,0x1用于标识只上报温度的消息,0x3用于标识上报描述信息(可变长度字符串类型)的消息。

messageId	
GL/+++	
田込	
int8u(8位无符号整型)	•
* 长府 ③	
1	
1	
* 默认值 ⑦	
0.2	
UX5	
0x5	
(病移值 ⑦	

添加length字段,表示字符串长度。"数据类型"根据可变长度字符串的长度进行配置,长度在255以内,配置为"int8u"。

标记为地址域 ②		
* 名字		
length		
ente		
		_
数据类型 int8u(8位无符号整型)		T
数据类型 int8u(8位无符号整型) * 长度 ⑦		T
数据类型 int8u(8位无符号整型) * 长度 ⑦ 1		•
数据类型 int8u(8位无符号整型) * 长度 ⑦ 1		•
数据类型 int8u(8位无符号整型) * 长度 ② 1 默认值 ③		•
数据类型 int8u(8位无符号整型) * 长度 ⑦ 1 默认值 ⑦		▼
数据类型 int8u(8位无符号整型) * 长度 ② 1 默认值 ③		•
数据类型 int8u(8位无符号整型) * 长度 ⑦ 1 默认值 ⑦ 偏移值 ⑦ 1-2		▼
数据类型 int8u(8位无符号整型) * 长度 ⑦ 1 第认值 ⑦ 偏移值 ⑦ 1-2		

添加other_info字段,表示可变长度字符串类型的描述信息。 "长度关联字段"选择 "length", "长度关联字段差值"和 "数值长度"自动填充。 设备管理 开发指南

步骤4 拖云	加右侧"设备模型	"区域的属性字段,	与数据上报消息的相应字段建立映射关系
---------------	----------	-----------	--------------------

标记为地址域 ②	
*名字	
other_info	
2447-23	
田 <u>心</u>	
数据类型	
varstring(可变长度字符	符串类型)
* 长度关联字段 ⑦	* 长度关联字段差值 ⑦
length •	0
length •	
length ▼ 数值长度 ⑦	0 * 默认值 ⑦
length 数值长度 ⑦ 1	0 * 默认值 ⑦
length ▼ 数值长度 ⑦ 1 掩码 ⑦	0 * 默认值 ⑦
length ▼ 数值长度 ⑦ 1 掩码 ⑦	0 * 默认值 ⑦
length ▼ 数値长度 ⑦ 1 掩码 ⑦	* 默认值 ⑦
length ▼ 数值长度 ② 1 1 掩码 ③ (扇移值 ③ 2-2	* 默认值 ⑦
length ▼ 数值长度 ⑦ 1 1 1 掩码 ⑦ 2-3	* 默认值 ⑦
length ▼ 数值长度 ⑦ 1 1 1 掩码 ⑦ 2-3	● 默认值 ⑦

C 🛙	đ
smokeinfo	
消息类型: deviceReq 是否包含响应消息: No	
描述:	
数据上报字段	+
1 messageld	
2 level	
3 temperature	
Ľ i	æ
temperature	
消息类型: deviceReq 是否包含响应消息: No	
描述:	
数据上报字段	+
1 messageld	
2 temperature	
Ľ 🕯	ð
other_info	
消息类型: deviceReq 是否包含响应消息: No	
描述:	
数据上报字段	+
1 messageld	
2 other_info	
Ľ	Ð
other_info_2	
消息类型: deviceReq	
描述:-	
数据上报字段	+
1 messageld	
2 length	
3 other_info	

步骤5 点击"保存",并在插件保存成功后点击"部署",将编解码插件部署到物联网平 台。

01) Profile定义 ———	02 Mit	₩1000 新聞集成指导	①4) 在线词则	发起目盼测试
在线编解码插件编辑器	插件管理			🖹 保存 📥 部署 😑 更多

----结束

调测编解码插件

步骤1 在烟感产品的开发空间,选择"在线调测",使用虚拟设备调试编解码插件。

0	Profile定义 -		(12) 编解码插件开发	03) 端倒集成指导 ————————————————————————————————————	(04 在线词测	发起自动测试
设	▲列表 ⑦						十新增测试设备
	状态	设备名称	设备ID	所屬产品	型号	设备类型	操作
				智无任何设备			

勾选"没有真实的物理设备",点击"创建"。

新增测试设备	×
您现在 〇 有真实的物理设备 ② 行真实的物理设备 	
您正在注册一个虚拟的设备	
创建取消	

步骤2 使用设备模拟器上报字符串类型的描述信息。

十六进制码流示例:0231。02表示messageld,此消息上报字符串类型的描述信息;31表示描述信息,长度为1个字节。

在"应用模拟器"区域查看数据上报的结果:{other_info=null}。描述信息不足6个字节,编解码插件无法解析。

	命令下发 ●● 教掘上报	 ・ ・ ・
服务 ▼ 命令 ▼		
		请输入十六进制码流 0231
缓存发送 立即发送 设置时间 v		自动发送 发送

十六进制码流示例:02313233343536。02表示messageld,此消息上报字符串类型的 描述信息;313233343536表示描述信息,长度为6个字节。

在"应用模拟器"区域查看数据上报的结果:{other_info=123456}。描述信息长度为 6个字节,编解码插件解析成功。

▲应用模拟器	命令下发 ○→ ◆o 数据上报	€SloT平台
全部 数据接收 命令发送 数据接收 2018-09-17 22:07:27 {other_info=123456}		数据上报 命令下发
数据接收 2018-09-17 22:07:12 {other_info=null}		 设备模拟器
		 新加速 前令接收 数据发送 2018-09-17 22:07:27 02313233343536 数据发送 2010 17 20:07:40
服务 ▼		2018-09-17 22:07:12 0231
命 令 ▼		
		请输入十六进制码流
		02313233343536
缓存发送		自动发送发送

十六进制码流示例:023132333435363738。02表示messageld,此消息上报字符串 类型的描述信息;3132333435363738表示描述信息,长度为8个字节。

在"应用模拟器"区域查看数据上报的结果:{other_info=123456}。描述信息长度超过6个字节,编解码插件截取前6个字节进行解析。

	命令下发 ● ◆● 数据上报	 ・ ・・・・・・・・・・・・・・・・・・・・・・・・・・・
服务 •		02313233343536 数据发送 2018-09-17 22:07:12 0231
		请输入十六进制码流 023132333435363738
立即发送 设置时间 >		白动发送发送

十六进制码流示例:02013132333435。02表示messageld,此消息上报字符串类型的 描述信息;013132333435表示描述信息,长度为6个字节。

在"应用模拟器"区域查看数据上报的结果: {other_info=\u000112345}。01在ASCII 码表里表示"标题开始",无法用具体字符表示,因此编解码插件解析为\u0001。

	命令下发	
●应用模拟器	0» 40	©♥IoT平台
全部 数据接收 命令发送 数据接收	<u> </u>	◆ 0 1 数据上报
2018-09-17 22:16:31 {other_info=\u000112345} 数据接收		命令下发 o ▼
2018-09-17 22:13:08 {other_info=123456} 数据接收		
2018-09-17 22:07:27 {other_info=123456} 数据接收 2018-09-17 22:07:12 {other_info=null}		王日 ² 数据发送 2018-09-17 22:16:31 02013132333435 数据发送
服务		2018-09-17 22:13:08 023132333435363738 数据发送 2018-09-17 22:07:27
命令 •		02313233343536 数据发送 2018-09-17 22:07:12 0231
		请输入十六进制码流
		02013132333435
经在发 送 立即发送		白动发送
2000年 设置时间 ↓		

步骤3 使用设备模拟器上报可变长度字符串类型的描述信息。

十六进制码流示例:030141。03表示messageld,此消息上报可变长度字符串类型的 描述信息;01表示描述信息长度(1个字节),长度为1个字节;41表示描述信息,长 度为1个字节。

在"应用模拟器"区域查看数据上报的结果: {other_info=A}。41是A的十六进制 ASCII码。

▲ 应用模拟器 全部 数据接收 命令发送 数据接收 2018-09-18.09:46:04 {other_info=A}	命令下发 ● ◆● 数据上报	◎ IoT平台 数据上报
服务 命令		
		请输入十六进制码流 030141
<u> 集存发送</u> 立即发送 设置时间 ~		□ 自动发送 发送

十六进制码流示例:03024142。03表示messageld,此消息上报可变长度字符串类型的描述信息;02表示描述信息长度(2个字节),长度为1个字节;4142表示描述信息,长度为2个字节。

在"应用模拟器"区域查看数据上报的结果: {other_info=AB}。4142是AB的十六进制 ASCII码。

应用模拟器	命令下发 oo 数据上报	 ・ ・ ・
<u>续存发送</u> 设置时间 ∨		请输入十六进制码流 03024142 自动发送

十六进制码流示例:030341424344。03表示messageld,此消息上报可变长度字符串 类型的描述信息;03表示描述信息长度(3个字节),长度为1个字节;41424344表示 描述信息,长度为4个字节。

在"应用模拟器"区域查看数据上报的结果: {other_info=ABC}。描述信息长度超过3 个字节,编解码插件截取前3个字节进行解析,414243是ABC的十六进制ASCII码。

	命令下发	
●应用模拟器	0>	©⊗ loT平台
_	◆o 数据上据	
全部 数据接收 命令发送	300AT1X	
数据接收		刻浩上报
2018-09-18 09:51:45 {other_info=ABC}		; ; 命令下发 ○ ▼
数据接收		
2018-09-18 09:48:25 {other_info=AB}		(•) 设备模拟器
数据接收		今郊 数据告送 会会按收
2018-09-18 09:46:04 {other_info=A}		
		剱塘友运 2018-09-18 09:51:45
		030341424344
		2018-09-18 09:48:25
服务 ▼		03024142 数据发送
		2018-09-18 09:46:04
命令		030141
		请输入十六进制码流
		030341424344
经存货关 立即将关		白动发送

十六进制码流示例:0304414243。03表示messageld,此消息上报可变长度字符串类型的描述信息;04表示字符串长度(4个字节),长度为1个字节;414243表示描述信息,长度为4个字节。

在"应用模拟器"区域查看数据上报的结果: {other_info=null}。描述信息长度不足4 个字节,编解码插件解析失败。
	1 会会下省	
●,应用模拟器	₩マトQ 0 +0 数据上掲	© IoT平台
全部 数据接收 命令发送 数据接收	MMLTH.	★ ● 数据上报
2018-09-18 09:58:40 {other_info=null} 数据接收		
2016-09-18 09:51:45 {other_info=ABC} 数据接收 2019-09-18 09:48:25		 2) 收留保扒器 全部 数据发送 命令接收
{other_info=AB} 数据接收 2018-09-18 09:46:04		数据发送 2018-09-18 09:58:40
{other_info=A}		03/04414243 数据发送 2018-09-18 09:51:45 020244101244
服务 •		数据发送 2018-09-18 09:48:25 03024142
۲ ۲		数据发送 2018-09-18 09:46:04 030141
		请输入十六进制码流
		0304414243
生 <		□ 自动发送 发送

----结束

总结

- 当数据类型为字符串或可变长度字符串时,插件是按照ASCII码进行编解码的:上报数据时,将16进制码流解码为对应字符串,比如:21解析为"!"、31解析为"1"、41解析为"A";下发命令时,将字符串编码对应的16进制码流,比如:
 "!"编码为21,"1"编码为31,"A"编码为41。
- 当某字段的数据类型为可变长度字符串时,该字段需要关联长度字段,长度字段 的数据类型必须为int。
- 针对可变长度字符串,命令下发和数据上报的编解码插件开发方式相同。
- 在线开发的编解码插件使用ASCII码16进制的标准表对字符串和可变长度字符串进行编解码。解码时(数据上报),如果解析结果无法使用具体字符表示,如:标题开始、正文开始、正文结束等,则使用\u+2字节码流值表示(例如:01解析为\u0001,02解析为\u0002);如果解析结果可以使用具体字符表示,则使用具体字符。

数组及可变长数组数据类型

如果该烟感设备需要支持描述信息上报功能,描述信息描述信息支持数组和可变长度 数组两种类型,则按照以下步骤创建消息。

Profile定义

在烟感产品的开发空间完成Profile定义。

	服务	名称	描述				最后	修改时间		操作		
^	\$mo	ke					2018	8/09/17 10:25:15				ii.
	属的	生列表									+ 添加属性	
	÷	level	数据类型 int	范围 0~3	步长 	单位 		是否必选 <mark>/</mark>	访问模式 R		Ľ i	
	÷	temperature	数据类型 int	范围 0~1000	步长 	单位 		是否必选 ✓	访问模式 R		12 Î	
	÷	other_info	数据类型 string	长度 100	单位 			是否必选 ✓	访问模式 R		12 ii	

编解码插件开发

步骤1 在烟感产品的开发空间,选择"编解码插件开发"。

(01) Profile定义	02 编解码插件开发	→ (03) 端侧集成指导	(04) 在线调测	发起自助测试

步骤2 配置数据上报消息,上报数组类型的描述信息。

新増消息		
基本信息		
消息名* other_info	消息描述	
* <u>消息类型</u> ③ 数据上报 命令下发		
□ 添加响应字段		
字段		
	十 添加字段	

添加messageld字段,表示消息种类。在本场景中,0x0用于标识上报火灾等级和温度的消息,0x1用于标识只上报温度的消息,0x2用于标识上报描述信息(数组类型)的消息。

添加字段	
✔ 标记为地址域 ⑦	
*名字 当标记为地址域时,名字固定为messageld;否则,名字不能设置为messageld。	
messageId	
描述	
牧据类型	
int8u(8位无符号整型)	•
* 长度 ⑦	
* 长度 ⑦ 1	
* 长度 ⑦ 1 * 默认值 ⑦	
* 长度 ⑦ 1 * 默认值 ⑦ 0x2	
* K度 ⑦ 1 * 默认值 ⑦ 0x2	
* 长度 ⑦ 1 * 默认值 ⑦ 0x2 偏移值 ⑦	
* K度 ② 1 * 默认值 ③ 0x2 偏移值 ③	
* 长度 ⑦ 1 * 默认值 ⑦ 0x2 偏移值 ⑦ 0-1	

添加other_info字段,表示数组类型的描述信息。在本场景中,"长度"配置为5个字节。

 添加字段	×
标记为地址域 ⑦	
*名字	
other_info	
描述	
数据类型	
array(数组类型)	
* 长度 ⑦ 5	
默认值 ⑦	
偏移值 ⑦	
1-0	
完成取消	

步骤3 配置数据上报消息,上报可变长度数组类型的描述信息。

新增消息		×
基本信息		
消息名。	消息描述	
○ 飲養上报 ○ 命令下发		
□ 添加响应字段		
字段		
	十 添加字段	
党成	取到	

添加messageld字段,表示消息种类。在本场景中,0x0用于标识上报火灾等级和温度 的消息,0x1用于标识只上报温度的消息,0x3用于标识上报描述信息(可变长度数组 类型)的消息。

名字 当标记为地	班上域时,名字固定为messageld;合则,名字 	小能设置为messageld。
messageId		
带述		
加星光刑		
+0/0/>T/	~ ㅋ ㅎ ㅋ ㅋ	
Int8u(81 <u>∨</u> 7⊑4	す 5 登空)	•
* 长度 ②		
1		
1		
* 默认值 ⑦		
0		
0x3		
0x3		
0x3 偏移值 ⑦		
0x3 偏移值 ⑦ 0-1		

添加length字段,表示数组长度。"数据类型"根据可变长度数组的长度进行配置, 长度在255以内,配置为"int8u"。

标记为地址域 ⑦			
名字			
length			
苗述			
数据类型			
数据类型 int8u(8位无符号	整型)		•
做据类型 int8u(8位无符号	整型)		•
数据类型 int8u(8位无符号 * 长度 ②	整型)		•
数据类型 int8u(8位无符号 * 长度 ⑦ 1	整型)		•
数据类型 int8u(8位无符号 * 长度 ⑦ 1	整型)		•
数据类型 int8u(8位无符号 * 长度 ⑦ 1 默认值 ⑦	<u>整型</u>)		•
数据类型 int8u(8位无符号 * 长度 ⑦ 1 默认值 ⑦	整型)		•
数据类型 int8u(8位无符号 * 长度 ⑦ 1 默认值 ⑦	整型)		▼
数据类型 int8u(8位无符号 * 长度 ⑦ 1 默认值 ⑦ 偏移值 ⑦	整型)		
数据类型 int8u(8位无符号 * 长度 ⑦ 1 默认值 ⑦ 偏移值 ⑦ 1-2	整型)		▼
数据类型 int8u(8位无符号語 * 长度 ⑦ 1 默认值 ⑦ 偏移值 ⑦ 1-2	整型)		

添加other_info字段,表示可变长度数组类型的描述信息。"长度关联字段"选择 "length","长度关联字段差值"和"数值长度"自动填充。 设备管理

开发指南

添加字段		×
- 标记为地址域 ⑦		
*名字		
other_info		
描述		
数据类型		
variant(可变长度数组	类型)	•
* 长度关联字段 ⑦	* 长度关联字段差值 ⑦	
length •	0	
数值长度 ⑦	* 默认值 ⑦	
1		
掩码 ⑦		
偏移値の		
2-3		
2.5		
_		
	完成 取消	

步骤4 拖动右侧"设备模型"区域的属性字段,与数据上报消息的相应字段建立映射关系。

1	a		level	
			Smoke	
keinfo				
退类型: deviceReq 否包含响应消息: No			temperature	
記述:			Smoke	
2010 L10 cp.0				
9X1811R7-12	+			
1 messageld				
2 level				
a terrester.				
3 temperature				
F% 🔳			temperature	
			Smoke	
temperature				
消息类型: deviceReq				
是白包含明应滴思: No 描述:				
数据上报字段	+			
1 messageld				
2 temperature				
C I	8		other_info	
other info			Smoke	
眉感突盛. devicerceq 是否包含明应消息: No				
描述:				
数据上报字段	+			
1 messageld				
2 other_info				
C i	8		↔ other_info	
other info 2			Smoke	
送自杀刑: deviceRen				
是否包含响应消息: No				
描述:				
数据上报字段	+			
1 messageld				
-				
2 length				
3 other info		-		

步骤5 点击"保存",并在插件保存成功后点击"部署",将编解码插件部署到物联网平 台。

①1 Profile定义		发起自盼潮航
在线编解码插件编辑器	插件管理	🖺 保存 💧 部署 📃 更多

----结束

调测编解码插件

步骤1 在烟感产品的开发空间,选择"在线调测",使用虚拟设备调试编解码插件。

01	Profile定义 -		(12) 编解码插件开发	03) 端侧集成指导 ————————————————————————————————————		4 在线调测	发起自动测试
设备	例表 ⑦						十新增测试设备
:	状态	设备名称	设备ID	所屬产品	퓗룩	设备类型	操作
				智无任何设备			

勾选"没有真实的物理设备",点击"创建"。

新增测试设备		×
您现在 〇 有真实的物理设备	● 没有真实的物理设备	
您正在注册一个虚拟的讨	设备	
êU	建取消	

步骤2 使用设备模拟器上报数组类型的描述信息。

十六进制码流示例:0211223344。02表示messageld,此消息上报数组类型的描述信息;11223344表示描述信息,长度为4个字节。

在"应用模拟器"区域查看数据上报的结果:{other_info=null}。描述信息不足5个字节,编解码插件无法解析。

▲ <u> </u>	節令下友 o→ ◆o 数据上报	 ・・・・・・・・・・・・・・・・・・・・・・・・・・・・・
服务 命令		
		请输入十六进制码流 0211223344
缓存发送 立即发送 设置时间 ~		□ 自动发送 发送

十六进制码流示例:021122334455。02表示messageld,此消息上报数组类型的描述 信息;1122334455表示描述信息,长度为5个字节。

在"应用模拟器"区域查看数据上报的结果:{other_info=ESIzRF=}。描述信息长度为 5个字节,编解码插件解析成功。

▲」应用模拟器 全部 数据接收 命令发送 数据接收 2018-09-18 11:50:41	命令下发 ○◆ ◆○ 数据上报	◎ loT平台 数据上报
{other_info=ESIzRF=} 数据授收 2018-09-18 11:29:13 {other_info=null}		 ● ● ● ● ● ● ● ●
服务		2018-09-18 11:29:13 0211223344
		请输入十六进制码流 021122334455
缓存发送 立即发送 设置时间 ~		自动发送发送

十六进制码流示例:02112233445566。02表示messageld,此消息上报数组类型的描述信息;112233445566表示描述信息,长度为6个字节。

在"应用模拟器"区域查看数据上报的结果:{other_info=ESIzRF=}。描述信息长度超过5个字节,编解码插件截取前5个字节进行解析。

	命令下发	
▲ 应用模拟哭	0≯	◎◎IoT平台
	4o	
	数据上报	A 9
全部 数据接收 命令发送		
数据接收		致店上按 ·
2018-09-18 11:56:31		命令下发
{other_info=ESIzRF=}		0 🔻
災防疫収 2018_00_18 11·50·11		() 沿客構切架
{other_info=ESIzRF=}		
数据接收		
2018-09-18 11:29:13		全部 数据发送 命令接收
{other_info=null}		数据发送
		2018-09-18 11:56:31
		02112233445566
		刻病及125 2018-09-18 11:50:41
		021122334455
服务 ▼		数据发送
		2018-09-18 11:29:13
命令 •		0211223344
		清輸λ+六进制码流
		02112233445566
缓存发送立即发送		自动发送发送

步骤3 使用设备模拟器上报可变长度数组类型的描述信息。

十六进制码流示例:030101。03表示messageld,此消息上报可变长度数组类型的描述信息;01表示描述信息长度(1个字节),长度为1个字节;01表示描述信息,长度为1个字节。

在"应用模拟器"区域查看数据上报的结果: {other_info=AQ==}。AQ==是01经过 base64编码后的值。

	命令下发	
●应用模拟器	o>	©loT平台
	*	
全部 数据接收 命令发送		数据上报!!
致据接收 2018-09-18 12:07:14		命令下发
{other_info=AQ==}		0 🔻
		② 设备模拟器
		全部 数据发送 命令接收
		数据发送
		2018-09-18 12:07:14 030101
服务		
命令		
		· 博输入十八进制的流
		030101
<i>生物的</i> 全部发送 一位		自动发送 发送 发送
设置时间 ∨		

十六进制码流示例:03020102。03表示messageld,此消息上报可变长度数组类型的 描述信息;02表示描述信息长度(2个字节),长度为1个字节;0102表示描述信息, 长度为2个字节。

在"应用模拟器"区域查看数据上报的结果:{other_info=AQI=}。AQI=是01经过 base64编码后的值。

	命令下发	
▲ 应用模拟器	0≯	◎◎IoT平台
	<o< th=""><th>CONT L</th></o<>	CONT L
	数据上报	▲ 0
全部 数据接收 命令发送		
数据接收		
2018-09-18 14:17:45		命令下发
{other_into=AQI=}		0 4
2018-09-18 12:07:14		(二) 设备模拟哭
{other_info=AQ==}		CT WERE DETAILED
		全部 数据发送 命令接收
		2018-09-18 14:17:45
		03020102
		数据发送
		2018-09-18 12:07:14 030101
服务		565151
命令 •		
		请输入十六进制码流
		03020102
缓存发送 <u>立即发送</u>		自动发送发送
设备时间。		

十六进制码流示例:03030102。03表示messageld,此消息上报可变长度数组类型的 描述信息;03表示描述信息长度(3个字节),长度为1个字节;0102表示描述信息, 长度为2个字节。

在"应用模拟器"区域查看数据上报的结果: {other_info=null}。描述信息长度不足3 个字节,编解码插件解析失败。

	命令下发	
▲ 应用模拟器	0≯	◎◎IoT平台
	♦o	001111
△○○ 数据技術 会会告送	数据上报	♠ 0
土即 数据按权 叩令及达		し し 数据上报 し
数据接收		
2018-09-18 14:22:30		
数据接收		
2018-09-18 14:17:45		(•)设备模拟器
{other_info=AQI=}		
数据接收 2019-00-19-12:07:14		全部 数据发送 命令接收
2018-09-18 12:07:14 {other_info=AO==}		
[outor_intervier-]		数据发送
		03030102
		数据发送
		2018-09-18 14:17:45
服务		03020102
		致病反达 2018-09-18 12:07:14
승수 · · · · · · · · · · · · · · · · · · ·		030101
		请输入十六进制码流
		03030102
缓存发送		自动发送发送
·····································		

十六进制码流示例:0303010203。03表示messageld,此消息上报可变长度数组类型的描述信息;03表示描述信息长度(3个字节),长度为1个字节;010203表示描述信息,长度为3个字节。

在"应用模拟器"区域查看数据上报的结果:{other_info=AQID}。AQID是010203经过base64编码后的值。

	命令下发	
▲ 应用模拟器	0≯	◎◎IoT平台
2.2.1 D D-3.7 Mill	* o	00.011
	数据上报	▲ 0
王		
数据接收		NILERIA
2018-09-18 14:31:18		命令下发
{OTNEF_INTO=AQID} 数据接收	-	0 4
2018-09-18 14:31:09		()设备模拟器
{other_info=null}		Contraction and a second
数据接收		全郊 数据学送 会会按 断
2018-09-18 14:30:55		
約据接收		数据发送
2018-09-18 14:30:24		2018-09-18 14:31:18
{other_info=AQ==}		数据发送
		2018-09-18 14:31:09
服务		03030102
1002		
<u>~</u>		03020102
H5 <		数据发送
		2018-09-18 14:30:24
		030101
		请输入十六进制码流
		0202010202
		0303010203
(###UV#		白动华兴
发行反达		
	1	

十六进制码流示例:030301020304。03表示messageld,此消息上报可变长度数组类型的描述信息;03表示描述信息长度(3个字节),长度为1个字节;01020304表示描述信息,长度为4个字节。

在"应用模拟器"区域查看数据上报的结果:{other_info=AQID}。描述信息长度超过 3个字节,编解码插件截取前3个字节进行解析,AQID是010203经过base64编码后的 值。

Control (1) Control (1) Sumption Control (1) Sumpt	▲ 応用模拟器	命令下发 ○≯	© loT平台
 登録 数据接收 命令分送 数据接收 ②119-09-18 14:35:24 ③今下送 ②119-09-18 14:35:18 (ofterinfo=AQID) 数据接收 ②119-09-18 14:31:18 (ofterinfo=AQID) 数据接收 ②10-109-16 14:31:09 (ofter_info=AQID) 数据接收 ②10-109-16 14:31:09 (ofter_info=AQID) 数据接收 ②10-109-16 14:30:05 (ofter_info=AQID) 数据接收 ③10-109-16 14:30:05 (ofter_info=AQID) 数据接收 ③10-109-16 14:30:05 (ofter_info=AQID) 数据接收 ①19-09-18 14:31:09 ①03001020304 数据接送 ②1019-09-18 14:31:09 ①0300102030 数据接送 ②1019-09-18 14:31:09 ①0300102 数增量 ②119-09-18 14:31:09 ①0300102 数增量 ②119-09-18 14:31:09 ①0300102 数增量 ②119-09-18 14:30:05 ①0300102 数增量 ②119-09-18 14:30:05 ①0300102 ③10 ③10300102 ③10300102 ③10 ③10		♦o 数据上报	
2013-02-13 14:35:24 (Cher_info-AQID) 茨爾接收 2019-03 14:31:09 (other_info-AQID) 茨爾接收 2019-03 14:31:09 (other_info-AQID) 茨爾接收 2019-03 14:30:05 (other_info-AQID) 茨爾接收 ①19-04 11:40:055 (other_info-AQID) 茨爾皮 廠令 ・	全部 数据接收 命令发送 ************************************		
Conter_info=null; 変研接検 2018-09-18 14:33:09 (other_info=null; 変研接検 2018-09-18 14:33:05 (other_info=ACle); 変研接検 電子	2018-09-18 14:35:24 (other info=AQID)		命令下发
(chrucinto=AQID) 数据数次 2018-09-181 14:31:09 (chrucinto=null) 数据数次 2018-09-181 14:30:55 (chrucinto=null) 数据数法 服务 • 命令 • 命令 • (chrucinto=null) 数据数法 数据数数 • ②019-09-181 14:30:25 (chrucinto=null) 数据数数 • 命令 • ● </th <th>数据接收 2018-09-18 14:31:18</th> <th></th> <th>(1)设备模拟器</th>	数据接收 2018-09-18 14:31:18		(1)设备模拟器
2010-09-18 14:31:09 (cher_info=nul) 愛媛愛敬 2010-09-18 14:30:55 (cher_info=AOL=) (cher_info=AOL=) 愛媛愛愛 マ 命令 マ 命令 マ (cher_info=AOL=) (cher_info=AOL=) 愛媛愛送 マ (cher_info=AOL=) (cher_info=AOL=) 愛媛愛送 マ (cher_info=AOL=) (cher_info=AOL=) 愛媛愛愛送 マ (cher_info=AOL=) (cher_info=AOL=) 愛媛愛送 マ (cher_info=AOL=) (cher_info=AOL=) 愛媛愛送 マ (cher_info=AOL=) (cher_info=AOL=) (cher_info=AOL=) (cher_info=AOL=) <t< th=""><th>{other_info=AQID} 数据接收</th><th></th><th></th></t<>	{other_info=AQID} 数据接收		
※店時設付 2018-09-18 14:30:55 (chter_info=AQI=) 数据接続 ● ● ● ● ● 10 ● 18 14:35:24 030301020304 数据接送 2018-09-18 14:31:18 0303010203 数据发送 2018-09-18 14:30:55 03020102 数据发送 2018-09-18 14:30:55 03030102 数据发送 2018-09-18 14:30:55 030301020304 ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ●	2018-09-18 14:31:09 {other_info=null}		全部 数据发送 命令接收
(ther_info-AQI=) 数据接收 服务 命令 (1)	数据接收 2018-09-18 14:30:55		2018-09-18 14:35:24 030301020304
服务 ● 命令 ● ● ● <	{other_info=AQI=} 数据接收		数据发送 2018-09-18 14:31:18
	服务		0303010203 数据发送
数据发送 2018-09-18 14:30:55 03020102 数据发送 2018-09-18 14:30:24	命 令 ▼		2018-09-18 14:31:09 03030102
03020102 数環发送 2018-09-18 14:30:24			数据发送 2018-09-18 14:30:55
2018-09-18 14.30.24			03020102 数据发送 2010-00-10-14/20/24
译用如人十六世期的流 030301020304 030301020304 回 自动发送 发送			
			時絶八十一八社市が当応
使置时间 ∨			030301020304
遂存发送 立即发送 □ 自动发送 发送 设置时间 ~ -			
遂存发送 立即发送 □ 自动发送 发送 设置时间 ∨			
遂存发送 立即发送 □ 自动发送 发送 设置时间 ~ <			
	修方华兴 一 一 一 一 一 一 一 一 一 一 一 一 一 一 一 一 一 一 一		

----结束

base64编码方式说明

base64编码方式会把3个8位字节(3*8=24)转化为4个6位字节(4*6=24),并在每个 6位字节前补两个0,构成4个8位字节的形式。如果要进行编码的码流不足3个字节,则 在码流后用0填充,使用0填充的字节经编码输出的字符为"="。

base64可以将16进制码流当做字符或者数值进行编码,两种方式获得的编码结果不同。以16进制码流01为例进行说明:

- 把01当作字符,不足3个字符,补1个0,得到010。通过查询ASCII码表,将字符 转换为8位二进制数,即:0转换为00110000、1转换为00110001,因此010可以 转换为00110000001100010010000(3*8=24)。再转换为4个6位字节: 001100、000011、000100、110000,并在每个6位字节前补两个0,得到: 00001100、00000011、00000100、00110000。这4个8位字节对应的10进制数 分别为12、3、4、48,通过查询base64编码表,获得M(12)、D(3)、E (4),由于3个字符中,最后一个字符通过补0获得,因此第4个8位字节使用 "="表示。最终,把01当做字符,通过base64编码得到MDE=。
- 把01当作数值(即1),不足3个字符,补两个0,得到100。将数值转换为8位2进制数,即:0转换为0000000、1转换为0000001,因此100可以转换为00000000000(3*8=24)。在转换为4个6位字节:000000、

010000、000000、000000,并在每个6位字节前补两个0,得到:00000000 00010000、00000000、00000000。这4个8位字节对应的10进制数分别为:0、 16、0、0,通过查询base64编码表,获得A(0)、Q(16),由于3个数值中, 最后两个数值通过补0获得,因此第3、4个8位字节使用"="表示。最终,把01 当作数值,通过base64编码得到AQ==。

总结

- 当数据类型为数组或可变长度数组时,插件是按照base64进行编解码的:上报数据时,将16进制码流进行base64编码,比如:01编码为"AQ==";命令下发时,将字符进行base64解码,比如:"AQ=="解码为01。
- 当某字段的数据类型为可变长度数组时,该字段需要关联长度字段,长度字段的数据类型必须为int。
- 针对可变长度数组,命令下发和数据上报的编解码插件开发方式相同。
- 在线开发的编解码插件使用base64进行编码时,是将16进制码流当做数值进行编码。

含命令执行结果的编解码插件

如果该烟感设备需要支持支持上报命令执行结果,则按照以下步骤创建消息。

Profile定义

在烟感产品的开发空间完成Profile定义。

Smoke 属性列表 令 level	数据类型 范围 int 0~3	步长 -	单位		2018/09/1	17 20:36:25		+ 添加居性	
属性列表 ◇ level	数据类型 范围 int 0~3	步长 	单位					+ 添加属性	
+> level	数据类型 范围 int 0~3	步长 	单位						
					是 	否必选	访问模式 R	C 1	
↔ temperature	数据类型 范围 int 0~100	步长 00	单位 		是 ~	否必选	访问模式 R	Ľ i	
命令列表								+ 添加命令	
∧								2	
下发命令字段								+ 添加下发字段	
value	数据关型 Int	范围 ; 0~1 ·	步长 	单位 		是否必选 ✓		2	
响应命令宇段								+ 添加响应字段	٦
result	数据类型 int	范围 : 0~3 ·	步长 	单位 		是否必选 ✓		2	

编解码插件开发

步骤1 在烟感产品的开发空间,选择"编解码插件开发"。

01) Profile定义	02 编解码插件开发	□	①4) 在线调测	发起自动测试
---------------	------------	---	----------	--------

步骤2 配置命令下发消息。

新增消息	×
基本信息	
消息名。	
SET_ALARM_ 满息描述	
* 海島映型 ② 数据上报 ③ 命令下发	
≥ 添加调应字段	
李段	
+ 添加字段	
响应字段	
十 添加明应字段	
完成 取消	

添加messageld字段,表示消息种类。如果只有一种命令下发消息,则可以不配置此字 段。

	×
✔ 标记为地址域 ②	
标记为响应标识字段 ⑦	
* 名字 当标记为地址域时,名字固定为messageld;否则,名字不能设置为messageld。	
messageId	
描述	
描述	
数据类型	
int8u(8位无符号整型)	•
* 长度 ⑦	
1	
1 * 默认值 ⑦	
1 * 默认值 ② 0x1	
1 * 默认值 ⑦ 0x1	
1 * 默认值 ⑦ 0x1 偏移值 ⑦	
1 * 默认值 ⑦ 0x1 6移值 ⑦ 0-1	
1 * 默认值 ⑦ 0x1 @移值 ⑦ 0-1	

添加mid字段,用于将下发的命令和命令执行结果进行关联。

添加字段	×	(
标记为地址域 ②		
✓ 标记为响应标识字段 ②		
*名字 当标记为响应标识字段时,名字固定为mid;否则	,名字不能设置为mid。	
mid		
描述		
描述		
数据类型		
int16u(16位无符号整型)	•	
2		
2		
默认值 ⑦		
默认值		
偏移值 ⑦		
1-3		
完成	取消	

添加value字段,表示下发命令的参数值。

添加字段			X
🔄 标记为地址域 📀			
标记为响应标识字	段 ②		
*名字			
value			
描述			
描述			
数据类型			
in+0山/0/六王/尔/	ヨ 史6 开川)		
Intou(o1y/bity)	5292)		
* 上座 ①			
1			
款认但 (?)			
默认值			
偏移值 ⑦			
3-4			
	完成	取消	

步骤3 配置命令下发响应消息。

添加messageld,表示消息种类。命令执行结果为上行消息,需要通过messageld和数据上报消息进行区分。

~ 枝	
	示记为地址域(②
	记为响应标识字段 ②
□ #	
<u> </u>	
* 名字	当标记为地址域时,名字固定为messageld;否则,名字不能设置为messageld。
n	nessageId
描述	
指	
数据当	(刊)
数据学	型
数据学 i	型 nt8u(8位无符号整型) ▼
数据学 i	型 nt8u(8位无符号整型) ▼
数据学 iI * 长	型 nt8u(8位无符号整型) ▼ 度 ⑦
数据纠 i * 步 1	型 nt8u(8位无符号整型) ▼ 度 ⑦
数据封 iI * 步 1	型 ht8u(8位无符号整型) ▼ 度 ⑦
数据封 i * b 1	型 ht8u(8位无符号整型) ▼ 度 ⑦ 以值 ⑦
数据学 iI * b 1 * 累 0	型 ht8u(8位无符号整型) ▼ 度 ⑦ 以值 ⑦
数据学 iI * 世 1 * 里 0	型 ht8u(8位无符号整型) ▼ 度 ⑦ K以值 ⑦
数据 ³ * ¹ 1 * 累 0 偏移	型 tBu(8位无符号整型) ▼ 度 ② 以值 ③ k2
数据 ³ * ¹ * ¹ * ¹ 0 偏移 0	型 talau(8位无符号整型) 、 度 ⑦ 以值 ⑦ x2 值 ⑦
数据¥ iI * H 1 * 里 0 偏移 0	型 ht8u(8位无符号整型) 、 度 ⑦ 以值 ⑦ x2 值 ⑦
数据 iii * 步 1 * 累 0 偏移 0	型 ht&u(&位无符号整型) 、 定 ⑦ 以值 ⑦ x2 值 ⑦



法加宁的	
※加予技	Х
标记为地址域 ②	
✔ 标记为响应标识字段 ②	
标记为命令执行状态字段 ②	
*名字 当标记为响应标识字段时,名字固定为mid;否则,名字不能设置为mid。	
mid	
描述	
描述	
<u> 数据采用</u>	
int16u(16位于符异教刑)	•
	•
* 长度 ②	
2	
默认值 ⑦	
默认值	
偏移值 ⑦	
1-3	
完成 10 22	

添加errcode字段,用于表示命令执行状态:00表示成功,01表示失败,如果未携带该字段,则默认命令执行成功。

The second second				
添加字段				×
标记为地址域 ⑦				
标记为响应标识字的	2 ?			
✔ 标记为命令执行状态	字段 ⑦			
* 夕今 · 当标词为会会#	"行业太守的时,文学团	宁为arroda,不则。 夕文不	部沿军为arroada	
errcode		22/Jencode , EX3 , E344	ASKE/Jentode.	
circouc				
描述				
数据类型				
数据类型 int8u(8位无符号	整型)		T	
数据类型 int8u(8位无符号 * 长度 ①	整型)		T	
数据类型 int8u(8位无符号 * 长度 ⑦ 1	整型)			
数据类型 int8u(8位无符号 * 长度 ⑦ 1	整型)		▼	
数据类型 int8u(8位无符号 * 长度 ⑦ 1 默认值 ⑦	整型)			
数据类型 int8u(8位无符号 * 长度 ⑦ 1 默认值 ⑦	整型)			
数据类型 int8u(8位无符号 * 长度 ⑦ 1 默认值 ⑦	整型)			
数据类型 int8u(8位无符号 * 长度 ⑦ 1 默认值 ⑦ 偏移值 ⑦	整型)			
数据类型 int8u(8位无符号 * 长度 ⑦ 1 默认值 ⑦ 偏移值 ⑦ 3-4	整型)			
数据类型 int8u(8位无符号 * 长度 ⑦ 1 默认值 ⑦ 偏移值 ⑦ 3-4	整型)			

添加result字段,用于表示命令执行结果。

添加字段				×
标记为地址域(3			
标记为响应标识	字段 ⑦			
标记为命令执行	状态字段 ②			
*名字				
result				
描述				
数据类型				
数据类型 int8u(8位无名	·号整型)		.	
数据类型 int8u(8位无符	·号整型)		Ţ	
数据类型 int8u(8位无余 * 长度 ⑦	·号整型)		· · · · · · · · · · · · · · · · · · ·	
数据类型 int8u(8位无符 * 长度 ⑦ 1	F号整型)		▼	
数据类型 int8u(8位无条 * 长度 ⑦ 1 默认值 ⑦	·号整型)		■	
数据类型 int8u(8位无名 * 长度 ⑦ 1 默认值 ⑦	F号整型)		■	
数据类型 int8u(8位无符 * 长度 ⑦ 1 默认值 ⑦	F号整型)			
数据类型 int8u(8位无条 * 长度 ⑦ 1 默认值 ⑦ 偏移值 ⑦	子号整型)		· · · · · · · · · · · · · · · · · · ·	
数据类型 int8u(8位无名 * 长度 ⑦ 1 默认值 ⑦ 偏移值 ⑦ 4-5	F号整型)			
数据类型 int8u(8位无名 * 长度 ⑦ 1 默认值 ⑦ 偏移值 ⑦ 4-5	F号整型)			

步骤4 拖动右侧"设备模型"区域的属性字段和命令字段,数据上报消息和命令下发消息的 相应字段建立映射关系。

	8
smokeinfo	
消息类型: deviceReq 是否包含响应消息: No	
描述:	
数据上报字段	+
1 messageld	
2 level	
3 temperature	
C Î	5
SET_ALARM	
洞意央型: CloudRed 是否包含响应消息: Yes 描述:	

叩女 トステロ	+
i messageiu	
2 mid	
2 mid	
2 mid 3 value	
2 mid 3 value 晚应学段	+
2 mid 3 value 响应于段 1 messageld	+
2 mid 3 value 响应于段 1 messageld 2 mid	+
2 mid 3 value 均应字段 1 messageld 2 mid 3 errcode	+

步骤5点击"保存",并在插件保存成功后点击"部署",将编解码插件部署到物联网平台。

01 Profile定义	02	编解码插件开发 → (03) 编图集成指导 → (03)	(04) 在线调制	发起自动测试
在线编解码插件编辑器	插件管理			🖹 保存 🔺 部署 三 更多

----结束

调测编解码插件

步骤1 在烟感产品的开发空间,选择"在线调测",使用虚拟设备调试编解码插件。

01 Profile定义		- 02 编解码插件开发	(03) 端创集成指导 ————————————————————————————————————		04 在按問题	发起自动测试
设备列表 ⑦						十新增测试设备
状态	设备名称	设备ID	所屬产品	副中	设备类型	操作
			智无任何设备			

勾选"没有真实的物理设备",点击"创建"。

新增测试设备	×
您现在 〇 有真实的物理设备	品的物理设备
您正在注册一个虚拟的设备	
创建	取消

步骤2 使用应用模拟器进行命令下发: { "serviceId": "Smoke", "method": "SET_ALARM", "paras": "{\"value\":0}" }。

在"设备模拟器"区域查看命令接收的结果:01000100。01为messageld字段,0001为mid字段,00为value字段。

	命令下发	
▲ 应用增加器	0>	
●,四田侯拟岙	¢o	◎◎101半台
	数据上据	
全部 数据接收 命令发送	XILLEXX	▲ 0
		数据上报
命令发送: 2018-09-18 15:56:59		
		「「前令下友」
友运消息头信息: { "time": "Tue Sep 18 15:56:59		0 4
GMT+08:00 2018", "requestId": "05be64da-daaf-d414-		(1) 沿久措训型
645f-f24554ba4cb7_9927", "callbackUrl": null,		(1) 以面(天)(品
"expireTime": 0, "command": { "serviceId": "Smoke",		
"method": "SET_ALARM", "paras": "{\"value\":0}" } }		全部 数据发送 命令接收
华泽巡白hady/言白: ["appring]d": "@maka" "mothod":		
友达消息Dody1言息、{ Serviceid . Smoke , method .		中文波仪 2010-00-19-15:56:50
"SET_ALARM", "paras": {\"value\":u}" }		01000100
		01000100
服务 Smoke ▼		
命令 SET ALARM ▼		
value * 输入的整数范围[0,1]		
0		
		· 哼锏入丁八进制饰派
		白动份子 (1556
发行反达		
设置时间 🗸		
L		

步骤3 使用设备模拟器进行数据上报。

十六进制码流示例:0200010000。02表示messageld,此消息上报命令执行结果;0001表示mid,长度为2个字节;00表示命令执行状态,长度为1个字节;00表示命令执行结果,长度为1个字节。

在"设备详情 > 历史命令"查看命令执行状态:执行成功。

设备详情	历史数据	设备日志	历史命令			
						ひ 刷新
状态	命令ID			命令创建时间	命令内容	命令调应
执行成功	963c1d3	c37304828a1c97727	c23bd268	2018/09/18 15:56:59	$\label{eq:serviceId} \ensuremath{\left\{ \begin{subarray}{c} \mbox{"serviceId": "Smoke", "method": "SET_ALARM", "paras": { "value": 0 } \ensuremath{\right\}} \end{subarray} \end{subarray}$	{ "result": 0 }

----结束

总结

- 如果插件需要对命令执行结果进行解析,则必须在命令和命令响应中定义mid字段。
- 命令下发的mid是2个字节,对于每个设备来说,mid从1递增到65535,对应码流为0001到FFFF。
- 设备执行完命令,命令执行结果上报中的mid要与收到命令中的mid保持一致,这 样平台才能刷新对应命令的状态。

4.5.3 离线开发插件(联通用户专用)

非联通用户请查看设备接入服务。

编解码插件实现二进制消息转JSON格式的功能,Profile定义了该JSON格式的具体内容。因此,编解码插件开发前需要先编写设备的Profile。

为了提高离线开发的集成效率,我们提供了编解码插件的<mark>编解码插件开发样例</mark>,建议 您基于DEMO工程进行二次开发。

注:由于插件离线开发较为复杂,且耗时比较长,我们推荐<mark>在线开发插件(联通用户</mark> 专用)。

开发环境准备

- 前往<mark>官网</mark>下载Eclipse安装包,直接解压缩到本地即可使用。
- 前往<mark>官网</mark>下载Maven插件包(zip格式),直接解压缩到本地。
- 安装JDK并配置Java的开发环境。

Maven的配置涉及Windows环境变量的配置与在Eclipse中的配置,环境变量的配置请参考网上资源,本节仅介绍Maven在Eclipse中的配置。

步骤1 选择Eclipse菜单"Windows"->"Preferences",打开Preferences窗口,选择 "Maven"->"Installations"->"Add"。

ype filter text	Installations		() Y () Y Y
> General	Select the installation used to launch Maven:		
 Data Management 	Name	Details	Add
 Help Testell (Le data 	EMBEDDED	3.2.1/1.5.0.20140605-2032	Edit
> Install/Update	WURKSPACE	C INOT AVAILABLE (3.0,1	Remove
> Java EE			1
> Java Persistence			
> JavaScript			
4 Maven			
Archetypes			
Errors/Warnings			
Installations			
Java EE Integratio			
Lifecycle Mapping			
Templates			
User Interface			
b Mylyn			
Plug-in Development	Note: Embedded ru	intime is always used for dependency	
Remote Systems	resolution		
Run/Debug +		Restore Defau	Apply
		L	

步骤2 选择maven插件包路径,点击"Finish",导入Maven插件。

New Maven Run	time	- • •
Specify attributes	for a Maven installation	
Installation type:	External Workspace	
Installation home:	D:\temp\apache-maven-3.5.0-bin\apache-maven-3	Directory
Installation name:	apache-maven-3.5.0	1
Additional extension	on libraries:	
		Project
		Remove
	/	Up
		Down
		Restore Default
?	Finish	Cancel

步骤3选择导入的maven插件,点击"OK"。

ype filter text	Installations	¢.*	ф·••
General	Select the installation used to	launch Maven:	
P Ant Data Management	Name	Details	Add
Help	EMBEDDED	3.2.1/1.5.0.20140605-2032	Contraction
Install/Lindate	WORKSPACE	NOT AVAILABLE (3.0.)	<u></u> £dit
> Java	apache-mayen-3.5.0	D:\temp\apache-mayen-3.5.0	Remove
Java EE		- the second sec	
Java Persistence			
JavaScript ≡			
Maven			
Archetypes		-	
Discovery			
Errors/Warnings	1		
Installations			
Java EE Integration		-	
Lifecycle Mapping			
Templates			
User Interface			
User Settings			
> Mylyn	* M	+	
Plug-in Development	Note: Embedded runtime is a	ways used for dependency	
Remote Systems	resolution		
Run/Debug *		Restore Defaulte	Annh
e m +		Restore Delauits	дрру

----结束

导入编解码插件 Demo 工程

步骤1 下载编解码插件DEMO工程,在"source_code"文件夹中获取"codecDemo.zip", 将其解压到本地。



步骤2 打开Eclipse,右击Eclipse左侧 "Project Explorer"空白处,选择"Import > Import..."。

Project Exp	plorer X 🗢 🗆	
	E	
	New	*
	Show In Alt+Shit	ift+W >
10	Сору	trl+C
10	Copy Qualified Name	
1	Paste Ci	tri+V
-34	Delete D	Delete
	Import	 App Client JAR fi
24	Export	📇 EAR file
5	Refresh	F5 EJB JAR file
-		BAR EL Utility Ja
		a, roan me

步骤3 展开"Maven",选择"Existing Maven Projects",点击"Next"。

Import	
elect	-V-
Import Existing Maven Projects	
Select an import source:	
type filter text	
> 🗁 CVS	*
> 👝 EJB	
Git	
Install	
D Dava EE	8
A Co Maven	
Check out Mayen Projects from SCM	
Install or deploy an artifact to a Mayer repository.	
Materialize Mayer Projects from SCM	
Plug-in Development	
Remote Systems	
A Dua Dahua	·

步骤4 点击"Browse",选择步骤1解压获得的"codecDemo"文件夹,勾选"/pom.xml",点击"Finish"。

a block the block of the		
naven Projects		
Select Maven projects		
Root Directory: D:\temp\co	decDemo	· Browse
Projects:		1
V /pom.xml com.thrid	party:WaterMeter-Huawei-NBIoTDevice:1.0.0tb	undle Select All
		Deselect All
		Select Tree
		Deselect Tree
		Befresh
Add project(s) to working	j set	
Add project(s) to working WaterMeter-Huawei-NB	j set IoTDevice	
Add project(s) to working WaterMeter-Huawei-NB Adyanced	g set IoTDevice	
Add project(s) to working WaterMeter-Huawei-NB Adyanced	set IoTDevice Back Next >	Finish Cancel

----结束

实现样例讲解

导入的编解码插件Demo工程结构如下图所示。



本工程是一个Maven工程,您可在此样例工程的基础上修改如下部分,适配成自己需要的编解码插件。

```
步骤1 修改Maven的配置文件
```

<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"</pre>

xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd"> <modelVersion>4.0.0</modelVersion>

<groupId>com.thrid.party</groupId>
<!-- **请修改为你的编解码插件的名字,命名规范: 设备类型-厂商ID-设备型号,例如: WaterMeter -Huawei-**NBIoTDevice -->
<artifactId>WaterMeter-Huawei-NBIoTDevice</artifactId>
<version>1.0.0</version>
<!-- 请检查这里的值为bundle,不能为jar -->
cpackaging>bundle/packaging>

•••••

<dependencies>

```
<!-- Huawei提供的编解码接口,必须引入 -->
<!-- systemPath请替换成你本地的目录 \codecDemo\lib\com.huawei.m2m.cig.tup-1.3.1.jar -->
<dependency>
<groupId>com.huawei</groupId>
<artifactId>protocal-jar</artifactId>
<version>1.3.1</version>
<scope>system</scope>
<systemPath>${basedir}/lib/com.huawei.m2m.cig.tup-1.3.1.jar</systemPath>
</dependency>
```

```
</dependencies>
       <build>
       <plugins>
         <!-- OSGI规范打包配置 -->
         <plugin>
           <configuration>
             <instructions>
               <!-- 请修改为你的编解码插件的名字,命名规范:设备类型-厂商ID-设备型号,例如:WaterMeter-
       Huawei-NBIoTDevice -->
               <Bundle-SymbolicName>WaterMeter-Huawei-NBIoTDevice</Bundle-SymbolicName>
             </instructions>
           </configuration>
         </plugin>
       </plugins>
       </build>
       </project>
步骤2 在ProtocolAdapterImpl.java中,修改厂商ID(MANU_FACTURERID)和设备型号
        (MODEL)的取值。物联网平台通过厂商ID和设备型号将编解码插件和Profile文件进
       行关联。
       private static final Logger logger = LoggerFactory.getLogger(ProtocolAdapterImpl.class);
       // 厂商名称
       private static final String MANU_FACTURERID = "Huawei";
       // 设备型号
       private static final String MODEL = "NBIoTDevice";
步骤3 修改CmdProcess.java中的代码,实现插件对下发命令和上报数据响应的编码能力。
       package com.Huawei.NBIoTDevice.WaterMeter;
       import com.fasterxml.jackson.databind.JsonNode;
       import com.fasterxml.jackson.databind.node.ObjectNode;
       public class CmdProcess {
         //private String identifier = "123";
```

```
//private String identifier = "123";
private String msgType = "deviceReq";
private String serviceId = "Brightness";
private String cmd = "SET_DEVICE_LEVEL";
private int hasMore = 0;
private int errcode = 0;
private int mid = 0;
private JsonNode paras;
```

public CmdProcess() {
}

public CmdProcess(ObjectNode input) {

```
try {
   // this.identifier = input.get("identifier").asText();
  this.msgType = input.get("msgType").asText();
  平台收到设备上报消息,编码ACK
  {
      "identifier":"0",
     "msgType":"cloudRsp",
"request": ***,//设备上报的码流
     "errcode":0.
     "hasMore":0
  }
  * */
  if (msgType.equals("cloudRsp")) {
     //在此组装ACK的值
     this.errcode = input.get("errcode").asInt();
     this.hasMore = input.get("hasMore").asInt();
  } else {
```

```
平台下发命令到设备,输入
       {
          "identifier":0,
         "msgType":"cloudReq",
"serviceId":"WaterMeter",
         "cmd":"SET_DEVICE_LEVEL",
          "paras":{"value":"20"},
          "hasMore":0
      }
* */
         //此处需要考虑兼容性,如果没有传mld,则不对其进行编码
         if (input.get("mid") != null) {
            this.mid = input.get("mid").intValue();
         this.cmd = input.get("cmd").asText();
         this.paras = input.get("paras");
         this.hasMore = input.get("hasMore").asInt();
       }
    } catch (Exception e) {
       e.printStackTrace();
    }
  }
  public byte[] toByte() {
     try {
       if (this.msgType.equals("cloudReq")) {
         应用服务器下发的控制命令,本例只有一条控制命令: SET_DEVICE_LEVEL
         如果有其他控制命令,增加判断即可。
          * */
         if (this.cmd.equals("SET_DEVICE_LEVEL")) {
            int brightlevel = paras.get("value").asInt();
            byte[] byteRead = new byte[5];
            ByteBufUtils buf = new ByteBufUtils(byteRead);
            buf.writeByte((byte) 0xAA);
            buf.writeByte((byte) 0x72);
            buf.writeByte((byte) brightlevel);
            //此处需要考虑兼容性,如果没有传mld,则不对其进行编码
            if (Utilty.getInstance().isValidofMid(mid)) {
               byte[] byteMid = new byte[2];
               byteMid = Utilty.getInstance().int2Bytes(mid, 2);
              buf.writeByte(byteMid[0]);
               buf.writeByte(byteMid[1]);
            }
            return byteRead;
         }
       }
       平台收到设备的上报数据,根据需要编码ACK,对设备进行响应,如果此处返回null,表示不需要对设备
响应。
       * */
       else if (this.msgType.equals("cloudRsp")) {
          byte[] ack = new byte[4];
          ByteBufUtils buf = new ByteBufUtils(ack);
          buf.writeByte((byte) 0xAA);
          buf.writeByte((byte) 0xAA);
         buf.writeByte((byte) this.errcode);
         buf.writeByte((byte) this.hasMore)
         return ack;
       return null;
    } catch (Exception e) {
```
```
// TODO: handle exception
               e.printStackTrace();
               return null;
             }
          }
        }
步骤4 修改ReportProcess.java中的代码,实现插件对设备上报数据和命令执行结果的解码能
        力。
        package com.Huawei.NBIoTDevice.WaterMeter;
        import com.fasterxml.jackson.databind.ObjectMapper;
        import com.fasterxml.jackson.databind.node.ArrayNode;
        import com.fasterxml.jackson.databind.node.ObjectNode;
        public class ReportProcess {
          //private String identifier;
          private String msgType = "deviceReq";
          private int hasMore = 0;
          private int errcode = 0;
          private byte bDeviceReq = 0x00;
          private byte bDeviceRsp = 0x01;
          //serviceId=Brightness字段
          private int brightness = 0;
          //serviceId=Electricity字段
          private double voltage = 0.0;
           private int current = 0;
          private double frequency = 0.0;
           private double powerfactor = 0.0;
          //serviceId=Temperature字段
          private int temperature = 0;
          private byte noMid = 0x00;
          private byte hasMid = 0x01;
           private boolean isContainMid = false;
           private int mid = 0;
           * @param binaryData 设备发送给平台coap报文的payload部分
                       本例入参: AA 72 00 00 32 08 8D 03 20 62 33 99
                       byte[0]--byte[1]: AA 72 命令头
                       byte[2]: 00 mstType 00表示设备上报数据deviceReq
                       byte[3]: 00 hasMore 0表示没有后续数据,1表示有后续数据,不带按照0处理
                       byte[4]--byte[11]:服务数据,根据需要解析//如果是deviceRsp,byte[4]表示是否携带mid,
        byte[5]--byte[6]表示短命令Id
           * @return
          public ReportProcess(byte[] binaryData) {
             // identifier参数可以根据入参的码流获得,本例指定默认值123
             // identifier = "123";
             如果是设备上报数据,返回格式为
             {
                "identifier":"123",
                "msgType":"deviceReq",
                "hasMore":0,
                "data":[{"serviceId":"Brightness",
                      "serviceData":{"brightness":50},
                     {
"serviceId":"Electricity",
                      "serviceData":{"voltage":218.9,"current":800,"frequency":50.1,"powerfactor":0.98},
                      "serviceId":"Temperature",
```

```
"serviceData":{"temperature":25},
           ]
  }
*/
  if (binaryData[2] == bDeviceReq) {
     msgType = "deviceReq";
     hasMore = binaryData[3];
     //serviceId=Brightness 数据解析
     brightness = binaryData[4];
     //serviceId=Electricity 数据解析
     voltage = (double) (((binaryData[5] << 8) + (binaryData[6] & 0xFF)) * 0.1f);
     current = (binaryData[7] << 8) + binaryData[8];</pre>
     powerfactor = (double) (binaryData[9] * 0.01);
     frequency = (double) binaryData[10] * 0.1f + 45;
     //serviceId=Temperature 数据解析
     temperature = (int) binaryData[11] & 0xFF - 128;
  }
   /1
  如果是设备对平台命令的应答,返回格式为:
  {
     "identifier":"123",
     "msgType":"deviceRsp",
     "errcode":0,
     "body":{****} 特别注意该body体为一层json结构。
  }
*/
  else if (binaryData[2] == bDeviceRsp) {
     msgType = "deviceRsp";
     errcode = binaryData[3];
     //此处需要考虑兼容性,如果没有传mld,则不对其进行解码
     if (binaryData[4] == hasMid) {
       mid = Utilty.getInstance().bytes2Int(binaryData, 5, 2);
       if (Utilty.getInstance().isValidofMid(mid)) {
          isContainMid = true;
       }
     }
  } else {
     return;
  }
}
public ObjectNode toJsonNode() {
  try {
     //组装body体
     ObjectMapper mapper = new ObjectMapper();
     ObjectNode root = mapper.createObjectNode();
     // root.put("identifier", this.identifier);
     root.put("msgType", this.msgType);
     //根据msgType字段组装消息体
     if (this.msgType.equals("deviceReq")) {
        root.put("hasMore", this.hasMore);
       ArrayNode arrynode = mapper.createArrayNode();
       //serviceId=Brightness 数据组装
        ObjectNode brightNode = mapper.createObjectNode();
       brightNode.put("serviceId", "Brightness");
       ObjectNode brightData = mapper.createObjectNode();
       brightData.put("brightness", this.brightness);
       brightNode.put("serviceData", brightData);
       arrynode.add(brightNode);
       //serviceId=Electricity 数据组装
```

ObjectNode electricityNode = mapper.createObjectNode(); electricityNode.put("serviceId", "Electricity"); ObjectNode electricityData = mapper.createObjectNode(); electricityData.put("voltage", this.voltage); electricityData.put("current", this.current); electricityData.put("frequency", this.frequency); electricityData.put("powerfactor", this.powerfactor); electricityNode.put("serviceData", electricityData); arrynode.add(electricityNode); //serviceId=Temperature 数据组装 ObjectNode temperatureNode = mapper.createObjectNode(); temperatureNode.put("serviceId", "Temperature"); ObjectNode temperatureData = mapper.createObjectNode(); temperatureData.put("temperature", this.temperature); temperatureNode.put("serviceData", temperatureData); arrynode.add(temperatureNode); //serviceId=Connectivity 数据组装 ObjectNode ConnectivityNode = mapper.createObjectNode(); ConnectivityNode.put("serviceId", "Connectivity"); ObjectNode ConnectivityData = mapper.createObjectNode(); ConnectivityData.put("signalStrength", 5); ConnectivityData.put("linkQuality", 10); ConnectivityData.put("cellId", 9); ConnectivityNode.put("serviceData", ConnectivityData); arrynode.add(ConnectivityNode); //serviceId=battery 数据组装 ObjectNode batteryNode = mapper.createObjectNode(); batteryNode.put("serviceId", "battery"); ObjectNode batteryData = mapper.createObjectNode(); batteryData.put("batteryVoltage", 25); batteryData.put("battervLevel", 12); batteryNode.put("serviceData", batteryData); arrynode.add(batteryNode); root.put("data", arrynode); } else { root.put("errcode", this.errcode); //此处需要考虑兼容性,如果没有传mid,则不对其进行解码 if (isContainMid) { root.put("mid", this.mid);//mid //组装body体,只能为ObjectNode对象 ObjectNode body = mapper.createObjectNode(); body.put("result", 0); root.put("body", body); } return root; } catch (Exception e) { e.printStackTrace(); return null; } ----结束

decode 接口说明

decode接口的入参binaryData为设备发过来的CoAP报文的payload部分。



设备的上行报文有两种情况需要插件处理(消息④是模组回复的协议ACK,无需插件 处理):

• 设备上报数据(对应图中的消息①)

字段名	类型	是否 必填	参数描述
identifier	String	否	设备在应用协议里的标识,物联网平台通 过decode接口解析码流时获取该参数,通 过encode接口编码时将该参数放入码流。
msgType	String	是	固定值"deviceReq",表示设备上报数据。
hasMore	Int	否	表示设备是否还有后续数据上报,0表示没 有,1表示有。 后续数据是指,设备上报的某条数据可能 分成多次上报,在本次上报数据后,物联 网平台以hasMore字段判定后续是否还有 消息。hasMore字段仅在PSM模式下生 效,当上报数据的hasMore字段为1时,物 联网平台暂时不下发缓存命令,直到收到 hasMore字段为0的上报数据,才下发缓存 命令。如上报数据不携带hasMore字段, 则物联网平台按照hasMore字段为0处理。
data	ArrayNode	是	设备上报数据的内容。

表 4-1 ArrayNode 定义

字段名	类型	是否 必填	参数描述
serviceld	String	是	服务的id。
serviceDat a	ObjectNod e	是	一个服务的数据,具体字段在profile里定 义。
eventTime	String	否	设备采集数据时间(格式: yyyyMMddTHHmmssZ)。 如:20161219T114920Z。

示例:

```
{
    "identifier": "123",
    "msgType": "deviceReq",
    "hasMore": 0,
    "data": [{
    "serviceId": "NBWaterMeterCommon",
    "serviceData": {
        "meterId": "xxxx",
        "dailyActivityTime": 120,
        "flow": "565656",
        "cellId": "5656",
        "cellId": "5656",
        "cellId": "5656",
        "signalStrength": "99",
        "batteryVoltage": "3.5"
    },
    "eventTime": "20160503T121540Z"
    },
    {
        "internalTemperature": 256
     },
        "eventTime": "20160503T121540Z"
    }]
}
```

• 设备对平台命令的应答(对应图中的消息⑤)

字段名	类型	参数描述	是否必填
identifier	String	设备在应用协议里的标识,物联网平台通 过decode接口解析码流时获取该参数, 通过encode接口编码时将该参数放入码 流。	石
msgType	String	固定值"deviceRsp",表示设备的应答消 息。	是

字段名	类型	参数描述	是否必填
mid	Int	2字节无符号的命令id。在设备需要返回 命令执行结果(deviceRsp)时,用于将 命令执行结果(deviceRsp)与对应的命 令进行关联。	是
		物联网平台在通过encode接口下发命令 时,把物联网平台分配的mid放入码流, 和命令一起下发给设备;设备在上报命令 执行结果(deviceRsp)时,再将此mid 返回给物联网平台。否则物联网平台无法 将下发命令和命令执行结果 (deviceRsp)进行关联,也就无法根据 命令执行结果(deviceRsp)更新命令下 发的状态(成功或失败)。	
errcode	Int	请求处理的结果码,物联网平台根据该参 数判断命令下发的状态。 0表示成功,1表示失败。	是
body	ObjectNo de	命令的应答,具体字段由profile定义。 注 :body体不是数组。	否

示例:

```
{
    "identifier": "123",
    "msgType": "deviceRsp",
    "mid": 2016,
    "errcode": 0,
    "body": {
        "result": 0
    }
}
```

encode 接口说明

encode接口的入参是JSON格式的数据,是平台下发的命令或应答。



平台的下行报文可以分为两种情况:

• 平台对设备上报数据的应答(对应图中的消息②)

字段名	类型	参数描述	是否必填
identifier	String	设备在应用协议里的标识,物联网平台通 过decode接口解析码流时获取该参数, 通过encode接口编码时将该参数放入码 流。	否
msgType	String	固定值"cloudRsp",表示平台收到设备的 数据后对设备的应答。	是
request	byte[]	设备上报的数据。	是
errcode	int	请求处理的结果码,物联网平台根据该参 数判断命令下发的状态。 0表示成功,1表示失败。	是
hasMore	int	表示平台是否还有后续消息下发,0表示 没有,1表示有。 后续消息是指,平台还有待下发的消息, 以hasMore字段告知设备不要休眠。 hasMore字段仅在PSM模式下生效,且需 要"下行消息指示"开启。	是

表 4-2 平台收到设备的上报数据后对设备的应答 encode 接口的入参结构定义

注:在cloudRsp场景下编解码插件检测工具显示返回null时,表示插件未定义上 报数据的应答,设备侧不需要物联网平台给予响应。

示例:

{

}

```
"identifier": "123",
"msgType": "cloudRsp",
"request": [
1,
2
],
"errcode": 0,
"hasMore": 0
```

• 平台命令下发(对应图中的消息③)

表 4-3 平台下发命令 encode 接口的入参结构定义

字段名	类型	参数描述	是否必填
identifier	String	设备在应用协议里的标识,物联网平台通 过decode接口解析码流时获取该参数, 通过encode接口编码时将该参数放入码 流。	否
msgType	String	固定值"cloudReq",表示平台下发的请 求。	是
serviceId	String	服务的id。	是
cmd	String	服务的命令名,参见profile的服务命令定 义。	是
paras	ObjectNo de	命令的参数,具体字段由profile定义。	是
hasMore	Int	表示平台是否还有后续命令下发,0表示 没有,1表示有。 后续命令是指,平台还有待下发的消息, 以hasMore字段告知设备不要休眠。 hasMore字段仅在PSM模式下生效,且需 要"下行消息指示"开启。	是
mid	Int	2字节无符号的命令id,由物联网平台内 部分配(范围1-65535)。 物联网平台在通过encode接口下发命令 时,把物联网平台分配的mid放入码流, 和命令一起下发给设备;设备在上报命令 执行结果(deviceRsp)时,再将此mid 返回物联网平台。否则物联网平台无法将 下发命令和命令执行结果(deviceRsp) 进行关联,也就无法根据命令执行结果 (deviceRsp)更新命令下发的状态(成 功或失败)。	是

示例:

文档版本 15 (2022-06-14)

```
"identifier": "123",

"msgType": "cloudReq",

"serviceld": "NBWaterMeterCommon",

"mid": 2016,

"cmd": "SET_TEMPERATURE_READ_PERIOD",

"paras": {

"value": 4

},

"hasMore": 0}
```

getManufacturerId 接口说明

返回厂商ID字符串。物联网平台通过调用该接口获取厂商ID,以实现编解码插件和 Profile文件的关联。只有厂商ID和设备型号都一致时,才关联成功。

示例:

{

}

@Override
public String getManufacturerId() {
 return "TestUtf8ManuId";
}

getModel 接口说明

返回设备型号字符串。物联网平台通过调用该接口获取设备型号,以实现编解码插件和Profile文件的关联。只有设备型号和厂商ID都一致时,才关联成功。

示例:

```
@Override
public String getModel() {
    return "TestUtf8Model";
}
```

接口实现注意事项

接口需要支持线程安全

decode和encode函数需要支持线程安全,不得添加成员变量或静态变量来缓存过程数据。

错误示例:多线程并发时A线程将status设置为"Failed",B线程可能会同时设置为"Success",从而导致status不正确,引起程序运行异常。
 public class ProtocolAdapter {
 private String status;

```
@Override
public ObjectNode decode(finalbyte[] binaryData) throws Exception {
    if (binaryData == null) {
        status = "Failed";
        return null;
    }
    ObjectNode node;
    ...;
    status = "Success";// 线程不安全
    return node;
}
```

正确示例:直接使用入参编解码,编解码库不做业务处理。

mid字段的解释

物联网平台是依次进行命令下发的,但物联网平台收到命令执行结果响应的次数未必 和命令下发的次序相同,mid就是用来将命令执行结果响应和下发的命令进行关联的。 在物联网平台,是否实现mid,消息流程也有所不同:

实现mid



若实现了mid,并且命令执行结果已上报成功,则:

- a. 命令执行结果响应中的状态(SUCCESSFUL/FAILED)会刷新到平台数据库中 该命令的记录;
- b. 平台推送给应用服务器的命令执行结果通知中携带commandId;
- c. 应用服务器查询会得到该命令的状态为SUCCESSFUL/FAILED。
- 不实现mid



若不实现mid,并且命令执行结果已上报成功,则:

- a. 命令执行结果响应中的状态(SUCCESSFUL/FAILED)不会刷新到平台数据库 中该命令的记录;
- b. 平台推送给应用服务器的命令执行结果通知中不携带commandId;
- c. 应用服务器查询会得到该命令的最终状态为DELIVERED。

🗀 说明

上述两个消息流程旨在解释mid字段的作用,部分消息流程在图中简化处理。

针对只关注命令是否送达设备,不关注设备对命令执行情况的场景,设备和编解码插 件不需要实现对mid的处理。

如果不实现mid,则应用服务器不能在物联网平台获取命令的执行结果,需要应用服务 器自行实现解决方案。比如应用服务器在收到命令执行结果响应(不带commandld) 后,可以根据如下方法来进行响应匹配:

- 根据命令下发的顺序。使用此方法,平台在对同一设备同时下发多条命令时,一 旦发生丢包,将会导致命令执行结果和已下发的命令匹配错误。因此,建议应用 服务器每次对同一设备仅下发一条命令,在收到命令执行结果响应后,再下发下 一条命令。
- 编解码插件可以在命令响应消息的resultDetail里加上命令的相关信息来帮助识别 命令,比如命令码。应用服务器根据resultDetail里的信息来识别命令执行结果响 应和已下发命令的对应关系。

禁止使用DirectMemory

DirectMemory是直接调用操作系统接口申请内存,不受JVM的控制,使用不当很容易 造成操作系统内存不足,因此编解码插件代码中禁止使用DirectMemory。

错误示例:使用UNSAFE.allocateMemory申请直接内存

```
if ((maybeDirectBufferConstructor instanceof Constructor))
{
    address = UNSAFE.allocateMemory(1L);
    Constructor<?> directBufferConstructor;
    ...
}
else
{
    ...
}
```

编解码插件的输入/输出格式样例

假定某款水表支持的服务定义如下:

服务类型	属性名称	属性说明	属性类型(数据类型)
Battery	-	-	-
-	batteryLevel	电量(0100)%	int
Meter	-	-	-
-	signalStrength	信号强度	int
-	currentReading	当前读数	int
-	dailyActivityTi me	日激活通讯时长	string

那么数据上报时decode接口的输出:

```
{
  "identifier": "12345678",
"msgType": "deviceReq",
  "data": [
     {
        "serviceld": "Meter",
        "serviceData": {
          "currentReading": "46.3",
          "signalStrength": 16,
          "dailyActivityTime": 5706
        },
        "eventTime": "20160503T121540Z"
     },
     {
        "serviceld": "Battery",
        "serviceData": {
          "batteryLevel": 10
        },
        "eventTime": "20160503T121540Z"
     }
  ]
}
收到数据上报后,平台对设备的应答响应,调用encode接口编码,输入为
{
  "identifier": "123",
"msgType": "cloudRsp",
```

"request": [



假定某款水表支持的命令定义如下:

基本功能名 称	分类	名称	命令参数	数据类型	枚举值
WaterMete r	水表	-	-	-	-
-	CMD	SET_TEMPE RATURE_RE AD_PERIO D	-	-	-
-	-	-	value	int	-
-	RSP	SET_TEMPE RATURE_RE AD_PERIO D_RSP	-	-	-
-	-	-	result	int	0表示成 功,1表示 输入非 法,2表示 执行失败

那么命令下发调用encode接口时,输入为

```
{
  "identifier": "12345678",
  "msgType": "cloudReq",
  "serviceId": "WaterMeter",
  "cmd": "SET_TEMPERATURE_READ_PERIOD",
  "paras": {
      "value": 4
    },
      "hasMore": 0
}
```

收到设备的命令应答后,调用decode接口解码,解码的输出

```
{
    "identifier": "123",
    "msgType": "deviceRsp",
    "errcode": 0,
    "body": {
        "result": 0
    }
}
```

编解码插件打包

插件变成完成后,需要使用Maven打包成jar包,并制作成插件包。

Maven打包

- 步骤1 打开DOS窗口,进入"pom.xml"所在的目录。
- 步骤2 输入maven打包命令: mvn package。
- **步骤3** DOS窗口中显示"BUILD SUCCESS"后,打开与"pom.xml"目录同级的target文件 夹,获取打包好的jar包。

jar包命名规范为:设备类型-厂商ID-设备型号-版本.jar,例如:WaterMeter-Huawei-NBIoTDevice-version.jar。

WaterMeter-Huawei-NBIoTDevice-1.0.0.jar		
◆ 名称	大小	
→(上层目录)		
Jacom Com	7.66 KB	
META-INF	1 KB	
J OSGI-INF	1 KB	
📾 json-lib-2.4-jdk15.jar	155.39 KB	

- com目录存放的是class文件。
- META-INF下存放的是OSGI框架下的jar的描述文件(根据pom.xml配置生成的)。
- OSGI-INF下存放的是服务配置文件,把编解码注册为服务,供平台调用(只能有 一个xml文件)。
- 其他jar是编解码引用到的jar包。

----结束

制作插件包

步骤1 新建文件夹命名为 "package" ,包含一个 "preload/" 子文件夹。

步骤2 将打包好的jar包放到"preload/"文件夹。

名称		修改日期	<u>美型</u>		
퉬 preload 📝 package-inte ison	编解码插件 描述文件	2017/2/13 15:31 2017/2/13 15:31	文件夹 JSON 文件		
WaterMeter-Huawei-NBIoTDevice-1.0.0.jar					

步骤3 在"package"文件夹中,新建"package-info.json"文件。该文件的字段说明和模板如下:

注: "package-info.json"需要以UTF-8无BOM格式编码。仅支持英文字符。

表 4-4 "package-info.json"字段说明

字段名	字段描述	是否必填
specVersion	描述文件版本号,填写固定值:"1.0"。	是

字段名	字段描述	是否必填
fileName	软件包文件名,填写固定值:"codec-demo"	是
version	软件包版本号。描述package.zip的版本,请与 下面的bundleVersion取值保持一致。	是
deviceType	设备类型,与Profile文件中的定义保持一致。	是
manufacturerName	制造商名称,与Profile文件中的定义保持一 致,否则无法上传到平台。	是
model	产品型号,与Profile文件中的定义保持一致。	是
platform	平台类型,本插件包运行的物联网平台的操作 系统,填写固定值:"linux"。	是
packageType	软件包类型,该字段用来描述本插件最终部署 的平台模块,填写固定值:"ClGPlugin"。	是
date	出包时间,格式为:"yyyy-MM-dd HH-mm- ss",如"2017-05-06 20:48:59"。	否
description	对软件包的自定义描述。	否
ignoreList	忽略列表,默认为空值。	是
bundles	一组bundle的描述信息。 注 :bundle就是压缩包中的jar包,只需要写一 个bundle。	是

表 4-5 bundles 的字段说明

字段名	字段描述	是否必填
bundleName	插件名称,和上文中pom.xml的Bundle- SymbolicName保持一致。	是
bundleVersion	插件版本,与上面的version取值保持一致。	是
priority	插件优先级,可赋值默认值:5。	是
fileName	插件jar的文件名称。	是
bundleDesc	插件描述,用来介绍bundle功能。	是
versionDesc	插件版本描述,用来介绍版本更迭时的功能特 性。	是

package-info.json文件模板:

"specVersion":"1.0", "fileName":"codec-demo", "version":"1.0.0",

{

设备管理 开发指南

```
"deviceType":"WaterMeter",
  "manufacturerName":"Huawei",
  "model":"NBIoTDevice",
  "description":"codec",
  "platform":"linux"
  "packageType":"CIGPlugin",
  "date":"2017-02-06 12:16:59",
  "ignoreList":[],
  "bundles":[
  {
     "bundleName": "WaterMeter-Huawei-NBIoTDevice",
     "bundleVersion": "1.0.0",
     "priority":5,
     "fileName": "WaterMeter-Huawei-NBIoTDevice-1.0.0.jar",
     "bundleDesc":""
     "versionDesc":""
  }]
}
```

步骤4 选中"package"文件夹中的全部文件,打包成zip格式("package.zip")。

```
注: "package.zip"中不能包含"package"这层目录。
```

----结束

编解码插件质检

编解码插件的质检用于检验编解码是否可以正常使用。

- 步骤1 获取**编解码插件检测工具**。
- **步骤2** 将检测工具"pluginDetector.jar"、Profile文件的"devicetype-capability.json"和需要检测的编解码插件包"package.zip"和tool文件夹放在同一个目录下。



- **步骤3**获取设备数据上报的码流,并在检测工具的"data report"页签,将码流以十六进制格式输入,例如:AA72000032088D0320623399。
- 步骤4 点击检测工具的"start detect",查看解码后的json数据。

日志文本框会打印解码数据,如果提示"report data is success",表示解码成功。

🖆 pluginDetector
"serviceId": "Temperature", "serviceData": { "temperature": 25 }
} [2017-02-21 14:45:05] INFO report data is success! decode report data success [2017-02-21 14:45:05] INFO input encode ack paras is:
<pre>{ "identifier": "123", "msgType": "cloudRsp", "request": "AA72000032088D0320623399", "errcode": 0, "hasMore": 0 } [2017.02.21.14:45:05]]NEC.epsedde.sek.result is 444400000</pre>
[2017-02-21 14:45:05] INFO encode ack result success! encode ack success [2017-02-21 14:45:05] INFO encode ack result success! encode ack success data report send command
Please enter hexadecimal stream:
AA72000032088D0320623399
Start detect Clear log

如果提示"ERROR",表示解码出现错误。

🖆 pluginDetector	
[2017-06-21 15:10:36] INFO begin detecting [2017-06-21 15:10:36] ERROR CODE = 3001, decode data failed: input binary data is invalid,data is agasdf [2017-06-21 15:10:36] ERROR detect data failed : java.lang.NullPointerExcept	ion
data report send command	
Please enter hexadecimal stream:	
agasdf	
start detect clear log	

步骤5 当解码成功后,检测工具会继续调用编解码插件包的encode方法,对应答消息进行编码。

当提示"encode ack result success"时,表示对设备的应答消息编码成功。

- **步骤6** 获取应用服务器下发的命令(应用服务器通过调用物联网平台的"创建设备命令"接口进行命令下发),并在检测工具的"data report"页签输入。
- 步骤7 点击检测工具的"start detect",检测工具会调用encode接口对控制命令进行编码。

如果提示"encode cmd result success",表示对命令编码成功;如果提示 "ERROR",表示对命令编码出现错误。

🖆 pluginDetector	- • ×
<pre>[2019-10-16 17:33:55] INFO begin detecting [2019-10-16 17:33:55] INFO input encode command paras is: { "identifier": "123", "msgType": "cloudReq", "serviceld": "NBWaterMeterCommon", "cmd": "SET_DEVICE_LEVEL", "mid": 2016, "paras": { "value": "10" }, "hasMore": 0 } [2019-10-16 17:33:55] INFO encode command result is AA720A07E0 [2019-10-16 17:33:55] INFO encode cmd result success!</pre>	
data report send command	-
Please enter Json message format:	
<pre>{ "identifier": "123", "msgType": "cloudReq", "serviceId": "NBWaterMeterCommon", "cmd": "SET_DEVICE_LEVEL", "mid": 2016, "paras": { "value": "10" . start detect clear log</pre>	

命令示例:

```
{
   "identifier": "123",
"msgType": "cloudReq",
"serviceld": "NBWaterMeterCommon",
    "cmd": "SET_DEVICE_LEVEL",
"mid": 2016,
   "paras": {
"value": "10"
   },
"hasMore": 0
}
```

步骤8 获取设备命令执行结果上报的码流,并在检测工具的"data report"页签,将码流以 十六进制格式输入,例如: AA7201000107E0。

步骤9 点击检测工具的"start detect",查看解码后的Json数据。

日志文本框会打印解码数据,如果提示"report command result success",表示解 码成功;如果提示"ERROR",表示解码出现错误。

🖆 pluginDetector	- • •
<pre>[2019-10-16 17:27:10] INFO begin detecting [2019-10-16 17:27:10] INFO encode report result is: { "msgType": "deviceRsp", "errcode": 0, "mid": 2016, "body": { "result": 0 } } [2019-10-16 17:27:10] INFO report command result success,errcode = 0 report command result s </pre>] uccess
data report send command	 ▼
Please enter hevadecimal stream:	
AA7201000107E0	

----结束

编解码插件包离线签名

当编解码插件开发完后,在安装到物联网平台之前,您需要先对插件包进行签名。此 时需要下载离线签名工具,并进行签名操作。

步骤1 登录"设备管理服务控制台",选择"系统管理 > 工具",单击┵,完成工具下载。

	> 業成型名
	华为惠派登名工具
	生成数字装名公私期对
a —	登名解決
	生成公私签制
0—	
	107+1587+金46 私規文件
	导入私规文件
	需要数字签编的软件包
	A 1
	进行数字蓝名
	软件包装金贴证
	公城文件 写入公城文件
	需要验验的软件包
	进行软件包绘器

步骤2 解压 "signtool.zip",双击 "signtool.exe",生成数字签名公私钥对。

- 1. 根据需要选择"签名算法",当前提供两种签名算法:。
 - ECDSA_256K1+SHA256
 - RSA2048+SHA256
- 2. 输入"私钥加密口令",口令复杂度需要满足如下条件:
 - 口令长度至少6个字符
 - 必须包括A-Z、a-z、0–9、:~`@#\$%^&*()-_=+|?/<>[]{},;'!"中至少两种字符 组合。
- 单击"生成公私密钥",在弹出框中选择保存密钥的路径,单击"确定",弹出 "Success!"提示框表示密钥生成完成。
 - 工具将生成公私密钥共2个文件:
 - 公钥文件: "public.pem"
 - 私钥文件: "private.pem"。

步骤3进行软件包数字签名。



- 1. 单击"导入私钥文件",导入生成的私钥文件"private.pem"。
- 在弹出的窗口中输入"私钥加密口令"。
 口令输入正确,则状态栏显示私钥文件路径;口令输入错误,则状态栏显示"私 钥导入失败"。
- 4击 送 选择待签名软件路径,单击"进行数字签名"按钮,。
 数字签名成功后,将会在原软件包所在目录下生成名称为"xxx_signed.xxx"的带签名软件包。

步骤4 验证软件包签名。

3. 第 483 华大田	####################################	⊡ ∲
	软件包数字签名 私 视文件 导入私 颁文件 重要数字签名的软件包 通行数字签名	
a b c	软件包括名脑证 公视文件 等入公视文件 需要验验的软件包 通行软件包贴证	-

- 1. 单击"导入公钥文件",导入公钥文件"public.pem"。
- 2. 单击 送选择待验证的软件包路径,单击"进行软件包验签"。
 - 验证成功则弹出"验证签名成功!"提示框。
 - 验证失败则弹出"验签异常!"提示框。
 - **注**: 在进行软件包验签时,带签名软件包的存放路径不能包含中文字符。

----结束

4.5.4 下载和上传插件(联通用户专用)

非联通用户请查看设备接入服务。

在线开发完成编解码插件后,可以将插件下载到本地。本地的插件也可以上传到其他任意的物联网平台上。

下载编解码插件

步骤1 编解码插件可以在"编解码插件开发"中下载,也可以在"产品详情"中下载。

• 在"编解码插件开发"界面,选择"更多 > 下载"。

		发起自动测试
在线輪解码插件轴错器 插件管理		🗋 保存 💧 部署 📃 更多
-+ #####	设备模型	 会 金麻 业 下版 血 即紙 ① 新手描引

在"产品开发"界面选择需要查看详情的产品,点击该产品右侧"···"按钮,然后点击"查看产品详情",进入"产品详情"界面。

▲ 项目概览	01 开发中	@2 自助	测试中		03 认证发布		
● 产品 🔹							
产品开发	• 开发中	• *	at+		• #k+	• E256	• E858
设备管理	3)		0	0	0
升级调试							
■ 应用 🔺	全部产品(3)						+ 18510 O RIRF
对接信息							
订阅调试		FESR	型号	设备类型	数据格式	最后律政时间	
应用调试	-	开始中 WaterMeter01	NBIoTDevice01	WaterMeter	二进制码流	2019/03/29 15:23:41	★要产品洋植
应用在线开发							2010

- **步骤2** 系统弹出"请选择插件签名方式"窗口,请选择"生成新的公私钥签名"或"使用已有公私钥签名":
 - 如果选择生成新的公私钥进行签名,则输入"私钥密码"后,点击"下载"。系统将生成已签名的插件包和公私钥文件。
 - 注:密码长度至少6个字符,必须为字母和数字的组合。

请选择插件签名方式	×
● 生成新的公私钥签名 ○ 使用已有私钥签名	
* 私钥密码	
下载签名插件包需要设置签名密码。	
下载	

- 如果选择已有私钥进行签名,则上传私钥文件,填写私钥密码后,点击"下载"。系统将生成已签名的插件包和公私钥文件。
 - 注:下载的压缩包里的私钥文件即为上传的私钥。

请选择插件签名方式		×
○ 生成新的公私钥签名 ● 使用已有私钥签名	4	
*上传私钥		_
private.pem	选择文件	
使用私钥给插件包签名		
* 私钥密码		
下载签名插件包需要设置签名密码。		
下载取	消	



上传编解码插件

当本地已经准备好编解码插件(如线下开发)时,则可以通过开发中心将插件上传至 物联网平台。

- 步骤1 在"编解码插件开发"界面,选择"插件管理"。
- 步骤2 选择需要上传的插件包,点击"上传"。

注:插件包的设备类型、型号、厂商ID等信息需要与该产品保持一致,才可以成功上传。

01) Profile定义 02 篇例	●摘件开发 03	端倒集成指导 ————————————————————————————————————	(04) 在线词测	发起	自助測試
在线编解码插件编辑器 插件管理				Q 全屏	 新手描引
插件部署状态信息					
浙什名称 - 浙仁大志 - 浙什来選 - 學蜀时间 -					
离线插件上传					
开发者Porture合目的电上修改加终度行整点并上修之袋。加快上修成功后会自 Doort.ock_Doort.ock001_CIGPlug-in_sign.zip 上作	1998年。	۵			

当界面提示"离线插件上传成功"时,表示插件已经完成上传。

01 Profile定义 02	编解码插件开发	3) 端倒集成指导	(04) 在线调测	- XI	目時測试
在线编解码插件编辑器 插件管理 离线描件	上传成功			Q 全屏	③ 新手描引
插件部署状态信息					
播件各称: DoorLock-2a59609b8/9440749728b1954df11 播件状态: ●末部導 邮件来意: 兩线上传 操作时间: 2019-04-02 14:25:15	a74-DoorLock001 高部書 創題後 主下戦 ()				
离线插件上传					
开致着Potal会目动鱼上伸动插件进行签名并上伸公钥,插件上伸缩 文件大力不超过35M,且必须为20p文件 上传	动后会自动原著。	۵			

-----结束

4.6 调测产品(联通用户专用)

非联通用户请查看设备接入服务。

当Profile和编解码插件开发完成后,应用服务器就可以通过物联网平台接收设备上报 的数据以及向设备下发命令。

开发中心提供了产品在线调测的功能,您可以根据自己的业务场景,在开发真实应用 和真实设备之前,使用应用模拟器和设备模拟器对数据上报和命令下发等场景进行调 测;也可以在真实设备开发完成后使用应用模拟器验证业务流,真实设备开发请参照 从这里开始进行。

当设备侧开发和应用侧开发均未完成时,您可以创建虚拟设备,使用应用模拟器和设备模拟器对Profile、插件等进行调测。虚拟设备调测界面结构如下:

▲应用模拟器	©©lo T Platform	《黑开
全部 数据接收 命令发送	武 <u>御上照</u> 武 <u>御上照</u> 武 <u>御上照</u> 武 <u>御上照</u> 前令下波 和助开发者进行定界定	据上报和命 的处理日志 , 位
2007년和6周30章. • 橫拟立用服务器下发命令 • 呈现应用服务器换收的数据 和下发的命令		
服务 BinarySwitch ・ 命令 TOGGLE ・	 横刈设备上报数据 呈现设备上报的数据和接收 	
toggie MB-Jahrennie BBI(0 , 1)	100mマ 車品入十八道時開度 商品入十八道時開度 前面入十八道時開度	Q 信息助 手
affaila China chona China v	输入设备模拟器上报的十六进制 码流 用用 5 秒 □ @ @@2023 202	

 当设备侧开发已经完成时,但应用侧开发还未完成时,您可以创建真实设备,使 用应用模拟器对设备、Profile、插件等进行调测。真实设备调测界面结构如下:

▲ 107日時以常 全部 数据除收 命令2018	应用機拟硼区域: • 模拟应用服务器下发命令 • 呈现应用服务器接收的数 握和下发的命令	消息器器 《 ^{集件}
副物 BnarySwitch ・ 命令 TOGGLE ・ Toggle 編入が歴史で語(), 1]	●◆下次 ◎ → → → → → → → → → → → → → → → → → → →	5
জনসময়ে ওঁটালমানে ইউলিপান 🗸	消息服踪区域:呈现数据上 报和命令下发时,物联网平 台的处理日志,帮助开发者 进行定界定位	

使用虚拟设备调测

步骤1 在产品开发空间,点击"在线调测"。

▲ 项目概览		产品开发 > WaterMe	eter01					 · 查看产品详情 	●如何开发产品
● 产品	-								
产品开发		0	WaterMeter01 开发中	所選行业:智慧城市 型号: NBIoTDevice01 ①	「第名称: ofo 设备到型: WaterMeter ⊙	10387651: 二进制681円 /1月10: 28598d9b8f94407497	1972801954df11a7d ()	CoAP	
设备管理									
升级调试		~		-	-				
■ 应用	*	(01) Profile定义		— (02) 编解码插件开发 ——	(03) 城创集成指导	04	在线调测	> [发起自动测试
对接信息		设备列表 ①							十新增制试设备
订阅调试		状态	设备名称	设施口	所履产品	코무	设施关型		操作
应用调试					智元任何设备				

步骤2 在"设备列表"区域,点击"新增测试设备"。

设备列表						十新增测试设备
状态	设备名称	设备ID	所屬产品	产品型号	产品类型	攝作
			暂无任何设备			

步骤3 系统将弹出"新增测试设备"窗口,勾选"没有真实的物理设备",点击"创建"。

新增测试设备	×
您现在 ○ 有真实的物理设备 <i> ⊙ 没有真实的物理设备</i>	
您正在注册一个虚拟的设备	
创建 取消	

步骤4 在设备列表中,选择新创建的虚拟设备,进入调试界面。虚拟设备名称组成为: "产品名称"+"Simulator",每款产品下只能够创建一个虚拟设备。

设备列表 +新t								
	状态	设备名称	设备ID	所属产品	产品型导	产品类型	操作	
	● 在线	WaterMeter01NBSimulator	d1ec9268-240c-4168-abe4-7bbaf66e583b	WaterMeter01	WaterMeter01	WaterMeter	=	
	● 高线	testdevice001	c2c3ffb2-f8f5-48f1-b44e-5943ec64d575	WaterMeterD1	WaterMeter01	WaterMeter	≘ ≣	

步骤5 在"设备模拟器"区域,输入十六进制码流或者JSON数据(以十六进制码流为例), 点击"发送",在"应用模拟器"区域查看数据上报的结果,在"消息跟踪"区域查 看物联网平台处理日志。

	命令下发		消息跟踪
●应用模拟器	¢o	©》 IoT平台	zhaojinyong1536377412910
	数据上报	ا الله ا	2018-09-08 11:30:21
		数据上报	[Cicijig received message nom device, deld = 2nao 23 m
張援収 119-00-09 11:30:21			zhaojinyong1536377412910
attervLevel=1}			2018-09-08 11:30:21
			COSICI get plug-in success, { key ='e08ac2b7703849c784cb0ae23e0283c5&&WaterMeter013
		(。)设备模拟器	
			zhaojinyong1536377412910
		全部 数据发送 命令接收	2018-09-08 11:30:21 ICICIaia decadas data sussess - Decade BenertDTO Edentific
		数据发送	=null_msoType=DEVICEREQ_deviceInfo= *****
		2018-09-08 11:30:21	hasMore=0,dtag=null, data=List <decodedatadto> is</decodedatadto>
		0001	[DecodeDataDTO [serviceId =Battery,
		PP→12500 2018-09-08 11:30:21	serviceData=******,eventTime=null],]]
		AAAADDOD	zhaojinyong1536377412910
帝 Battery	•		2018-09-08 11:30:21
			[CIG]cig encode ack message successI encodeResult = [-86,
CHANGE_STATUS	•		-86, 0, 0]
			zhaojinyong1536377412910
lue * 输入的整数范围[0 , 1]			2018-09-08 11:30:21
			[CIG] cig is ready to response to us. Ready to encode the
			"request"."*******." "errcode":0. "hasMore"."0"}
		请输入十六进制码流	
			zhaojinyong1536377412910
		0001	2018-09-08 11:30:21 (CiCicia condidate to kafee suscession, [DeviseDate(constTur
			DATA, service= Battery, datas= (betteryLevel=1), time=
			20180908T033021Z, hasMore=false)]
			zbaojinyon@1536377412010
			2018-09-08 11:30:21
			[CIG]cig send ack message success, Response have beed
			posted to device, response : [-86, -86, 0, 0]
			zhaojinyong1536377412910
10.514		自动发送发送	2018-09-08 11:30:21
大法			[IOCM]iocm receives data from cig by kafka,
			UpdateUeviceDatasDTOGW2Cloud [header=Header [requestid=null_method=POST_timestamn=null_basMore =
			false), body=Body [services=[DeviceServiceA [serviceId=Batt
			eventTime=20180908T033021Z]]]] templateId = ruleEngine

步骤6 在"应用模拟器"区域进行命令下发,在"设备模拟器"区域查看接收到的命令(以 十六进制码流为例),在"消息跟踪"区域查看物联网平台处理日志。

命令下	发不分命	消息跟踪
●		BIOT毕会 T中会社 ThatBIRE ThatGING State ThatGING S

----结束

使用真实设备调测

请先参照**设备侧开发**将设备接入开发中心,开发中心的接入地址请参照<mark>平台对接信息</mark> 获取。

确保设备上报的数据信息与Profile中定义的一致,且如果设备上报的是二进制码流, 请确保已经开发了编解码插件。

步骤1 在产品开发空间,点击"在线调测"。



步骤2 在"设备列表"区域,点击"新增测试设备"。



- **步骤3** 系统将弹出"新增测试设备"窗口,勾选"有真实的物理设备",完成各项参数配置后,点击"创建"。
 - "设备名称"只允许大小写字母、数字和下划线,需要在产品下保持唯一。
 - "设备标识"需要在产品下保持唯一,如设备的IMEI、mac等。
 - "验证码加密"根据设备的实际情况进行配置。

AUCHIVIAN CH	
您现在	
 有真实的物理设备 没有真实的物理设备 	
*设备名称	
testdevice001	
*/7.2.1-10	
. And the second s	

设备创建成功后,将返回"设备ID"和"PSK码"。如果设备使用DTLS协议接入 物联网平台,请妥善保存PSK码。

设备创建成功	×
请根据设备指导说明书为设备接通电源,配置好网络,开启设备,观察设备是否成功接入到平台,如果状态是在线(online)表示设备已经成功的接入到平台,接着就可以接收设备的数据。 以下是您的设备信息,请牢记!	
设备ID c2c3ffb2-f8f5-48f1-b44e-5943ec64d575	
PSK码(使用DTLS协议时需要使用到该psk码,请您牢记!) ceb95d32c2da62b4a2cf9530096b178d	
确定	

步骤4 在设备列表中,选择新创建的真实设备,进入调试界面。

设备列表							
	状态	设备名称	设备ID	所屋产品	产品型号	产品类型	提作
	● 高线	testdevice001	c2c3ffb2-f8f5-48f1-b44e-5943ec64d575	WaterMeter01	WaterMeterD1	WaterMeter	⊡ ≣

步骤5 将真实设备接入到物联网平台,并进行数据上报,在"应用模拟器"区域查看数据上报的结果,在"消息跟踪"区域查看物联网平台处理日志。

	1	消息跟踪 く展开
<u> </u>		123456789 2018-09-08 10:38:03 [CIGkig received message from device, ueld = 12****89 123456789 2018-09-08 10:38:03 [CIGkig decodes data success, DecodeReportDTO [identifier -null, mg7/pe-DEVICERE_041=14***** hasMore-0.dtag=null, deta=14**CecedeRebatTOT> is
膨弱 Battery •		[DecodeDataD70 [perviceId =Battery, serviceData=******,eventTime=null],]] 12456789 2018.09.08 10:58:03 [CIGI6] og el olug-in success, [Vey =e08ac2bi7703849c784c00ac25e0283c58.8WaterMeter01) 122456789
命令 CHANGE_STATUS V	● 真实物理设备	2018.49-49 to 55:8:03 (CIG)b is ready to response to ue, Ready to encode the reponse to device, msgType. cloudRsg, input ["dentifier".null, "request" "*******", "encode"0, "hasMore".0") 124345/789
(A)		2018.09.09 10:58:00 [CIGIpig send data to kafka success, [DeviceData[reportType= DATA, service=Battery, datas= (battery Level=1), time= 2018/08/0810258032, hasMore=fate]]
		122456789 2019.49.49 10:58:03 [CIGIbij encode ack message successt encodeResult = [-96, -96, 0, 0]
		12/46/6/89 2016.49-04 91:058:03 (CIG)cji send ack message success, Response have beed posted to device, response : [-86, -86, 0, 0]
使存发送 设置时间 ↓		L2490165 2016_40/64 1055803 10C/M/jccm receive staft from cig by Isafira, Index received a staft from cig by Isafira, Index received a staft for the staff received a staff received a received a staff received a staff received a staff received a table). body-Body (services-TDexics/ServiceA (service16-staffs), event Time>1016/07/071011 tendeded a cindE-tonine

步骤6 在"应用模拟器"区域进行命令下发,在"消息跟踪"区域查看物联网平台处理日志,在真实设备上查看接收到的命令。

 ▲ CHINERIZE ▲ CONTRACT STATUS ▲ CANCE_STATUS ▲ CANCE_STATUS<!--</th--><th></th><th>1</th><th></th><th>消息跟踪</th><th></th>		1		消息跟踪	
またがある または、 または、 または、 または、 または、 または、 または、 または、		●令下致 • \$2度上报	政策上报 ● 命令下没 ● 真实物理设备	123456789 120C0/jicen statis to send command to cip to lafa using immediate strategy. command:chetalipolication-Aepti impediate strategy. command:chetalipolication-Aepti impediate strategy. command:chetalipolication-Aepti application and an annual command to a strate the strategy application and an annual command to a strategy. Common medical-huir, car-huir, hold-huir, hold-huir, hold-hail, hold-hail, and annual command: annual command to a strategy. The strategy (command: Postice Strategy). Command: the strategy of the strategy (command: the strategy). Common common and the strategy of the strategy (command: the command to lafata, Platform get one command: the command to lafata, Platform, get one catabactulif-infut; the	:ation 18-09- :=null, o=null, o=null, 'null}
	资存发送 立即发送 设置时间 ~			123456789 2018-09-08 11:02:00 [CIG]cig generate payload success, null	

----结束

设备管理

平台为每一个注册到平台上的设备生成了一个**deviceld**,这是设备在平台上的唯一 ID,在接口调用时,每个与设备的相关操作都需要**deviceld**。开发中心的"设备管 理"功能模块呈现本项目中的所有产品下的真实设备和虚拟设备,并提供分类统计、 在线调测、设备日志等功能,以便于进行设备管理和问题定位。

在"设备管理"新增真实设备,设备名称由开发者定义。当设备侧开发已经完成时,您可以在开发中心创建真实设备,对真实物理设备、编解码插件、应用服务器等进行端到端调试。

在"设备管理"新增虚拟设备,设备名称由系统生成,名称组成为: "产品名称"+"Simulator",每款产品下只能够创建一个虚拟设备。当设备侧开发还未完成时,您可以在开发中心创建虚拟设备,对编解码插件、应用服务器等进行调测。

4.6.1 设备侧开发

4.6.1.1 使用 MQTTS 协议接入(联通用户专用)

非联通用户请查看设备接入服务。

MQTTS是安全的基于TLS的加密协议,采用MQTTS协议接入平台的设备,设备与物联网平台之间的通信过程,数据都是加密的,具有一定的安全性。

MQTT主要应用于计算能力有限,且工作在低带宽、不可靠的网络的远程传感器和控制设备,适合长连接的场景,如智能路灯等。更多关于MQTT协议语法及接口信息,请访问http://docs.oasis-open.org/mqtt/mqtt/v3.1.1/os/获取。

🛄 说明

若选择MQTTS协议接入平台,建议通过使用Agent SDK接入。

使用限制

描述	限制	
支持的MQTT协议版本	3.1.1	
与标准MQTT协议的区别	 支持Qos 0和Qos 1 不支持QoS2 不支持will、retain msg 不支持Topic自定义 	
MQTTS支持的安全等级	采用TCP通道基础 + TLS协议(TLSV1、 TLSV1.1和TLSV1.2 版本)	
单帐号每秒最大MQTT连接请求数	无限制	
单个设备每分钟支持的最大MQTT连 接数	1	
单个MQTT连接每秒的吞吐量,即带 宽,包含直连设备和网关	3KB/s	
MQTT单个发布消息最大长度,超过 此大小的发布请求将被直接拒绝	1MB	
MQTT连接心跳时间建议值	心跳时间限定为30至1200秒,推荐设置为 120秒	
产品是否支持自定义Topic	不支持	
消息发布与订阅	设备只能对自己的Topic进行消息发布与订 阅	

描述	限制
每个订阅请求的最大订阅数	无限制

相关资源

您可以参考**MQTT接口**文档,把具备MQTT通信能力的设备接入物联网平台。您也可以 通过MQTT.fx体验设备接入,快速验证是否可以与物联网平台服务交互发布或订阅消 息。

4.6.1.2 使用 LoRaWAN 协议接入(联通用户专用)

非联通用户请查看设备接入服务。

概述

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。

物联网平台支持设备通过LoRa网络接入,但LoRaWAN设备不能直连物联网平台,需要经过LoRa服务商提供的LoRa Server接入物联网平台,如下图所示。



- LoRa Server: 由LoRa服务商提供的LoRa接入服务器,LoRa Server与物联网平台 之间通过MQTT协议对接,与LoRa网关之间通过标准IP网络对接。
- LoRa网关: LoRa网关向设备提供LoRa网络接入,设备通过LoRa网关对接到LoRa Server。LoRa网关通常也由LoRa服务商提供。
- LoRaWAN设备: LoRaWAN设备即LoRa节点,其通过LoRa网络接入LoRa网关, 然后通过LoRa Server最终接入到物联网平台。LoRaWAN设备可以通过以下几种 方式获取:
 - 直接购买合适的LoRaWAN设备。
 - 购买LoRa模组,并集成到设备中。

🛄 说明

LoRaWAN设备和LoRa网关并无一一对应关系,LoRaWAN设备可通过LoRa网络向任一个LoRa网关发送数据。若多个LoRa网关同时接收到一个LoRaWAN设备的数据并上报到LoRa Server, LoRa Server会进行数据去重。

接入流程

🗀 说明

测试场景:平台提供LoRaWAN协议的测试环境,多个用户共用一个Network Server,您可以通过提工单的方式获取测试账号,目前账号使用期限为7天,到期回收。

商用场景:LoRa服务商在华为云为每个客户部署一个Network Server。

前置条件:

在LoRaWAN设备接入物联网平台前,您需要完成以下前置条件:

- 选择LoRa服务商:选择一个LoRa服务商并与其达成合作关系,您可以通过提交工
 单咨询华为云物联网平台当前支持的LoRa服务商。
- 完成LoRa网络的对接调试:在选择LoRa服务商后,您需要根据LoRa服务商的要求 和指导,完成LoRaWAN设备、LoRa网关和LoRa Server的对接调试。

接入流程:

步骤	说明
对接LoRa Server	LoRa Server作为一个MQTT设备在物联网平台上注册并 上线。
创建LoRa网关	LoRa网关作为LoRa Server下的一个子设备,在物联网 平台上注册。
创建LoRaWAN设备	LoRaWAN设备作为LoRa Server下的一个子设备,在物 联网平台上注册。
业务调试	对接入物联网平台的LoRaWAN设备进行数据上报和命 令下发测试,具体操作可参考 <mark>数据上报</mark> 和 <mark>命令下发</mark> 。

对接 LoRa Server

在LoRaWAN设备接入物联网平台前,需要先完成物联网平台与LoRa Server的对接。

文档版本 15 (2022-06-14)

步骤1 新增LoRa Server的产品模型。

您可在产品中心搜索"设备类型"为"loraServer","产品范围"为"第三方公开" 的产品模型,导入使用;也可以按照以下步骤手动创建。

- 1. 登录物联网平台控制台,点击右上角"进入设备管理服务"。
- 3. 在弹框页面里填写关键参数信息后,单击"确定"。
 - 产品名称: 自定义该新建产品的产品名。
 - 型号: LoRa Server的型号,可自定义设置。
 - 设备类型:固定设置为"loraServer"。
 - 厂商名称:LoRa服务商的名称,可自定义设置。
 - 协议类型:选择"MQTT"。
- 步骤2 注册LoRa Server。
 - 1. 在设备管理服务中选择"设备 > 设备注册"。
 - 选择页签"单个注册",单击右上角"创建",填写关键参数后,单击"确定"。
 - 选择产品:选择1中添加的LoRa Server产品模型。
 - 设备标识码:对于LoRa Server,该参数实际未使用,可自定义设置。
 - 密钥:可自定义设置。
 - 确认密钥:可自定义设置。
 - 3. 注册设备成功后,将平台返回的"设备ID"和"密钥"保存并提供给LoRa服务 商。
- 步骤3 LoRa服务商部署LoRa Server, LoRa Server通过设备ID和密钥接入物联网平台。
- **步骤4** 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。

----结束

创建 LoRa 网关

在LoRa Server与物联网平台对接成功后,以子设备的方式在LoRa Server下创建LoRa 网关。

在物联网平台上创建LoRa网关,平台会同步在LoRa Server上注册LoRa网关的信息, 使LoRa网关可以正常接入LoRa Server。

- 步骤1 新增LoRa网关的产品模型。如果物联网平台上已添加了LoRa网关的产品模型,可跳过此步。
 - 1. 登录物联网平台控制台,点击右上角"进入设备管理服务"。

 - 3. 在弹框页面里填写关键参数信息后,单击"确定"。

- 产品名称: 自定义该新建产品的产品名。
- 型号:LoRa网关的型号,可自定义设置。
- 设备类型:固定配置为LoRaGateway。
- 厂商名称:LoRa网关的厂商名称,可自定义设置。
- 协议类型:选择"LoRaWAN"。

步骤2 添加LoRa网关。

- 1. 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。
- 2. 选择"子设备"页签,单击右上角的"添加"按钮添加子设备。
- 3. 在弹框页面里填写关键参数后,单击"确定"。
 - 选择产品:选择步骤1中添加的LoRa网关产品模型。
 - 设备名称: 自定义该新建设备的名称。
 - 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网关或LoRaWAN设备使用的接入码,由 LoRa服务商提供。
 - 🛄 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。

----结束

创建 LoRaWAN 设备

在LoRa Server与物联网平台对接成功和LoRa网关在线后,以子设备的方式在LoRa Server下创建LoRaWAN设备。

在物联网平台上创建LoRaWAN设备,平台会同步在LoRa Server上注册LoRaWAN设备的信息,使LoRaWAN设备可以正常接入LoRa Server。

- 步骤1 新增LoRaWAN设备的产品模型。如果物联网平台上已添加了LoRaWAN设备的产品模型,可跳过此步。
 - 1. 登录物联网平台控制台,点击右上角"进入设备管理服务"。
 - 单击左下角 ,选择"产品模型",单击"新增产品模型",产品模型可通 过手动创建和从本地导入两种方式。
 - 选择"本地导入",进入到本地导入产品的页面。
 - i. 在弹出的窗口中输入产品名称,并上传Profile资源文件。
 - ii. 单击"确定",等待导入完成。

🛄 说明

产品ID和产品密钥用于设备注册,请单击"保存密钥至本地",以保存产品密钥信息,密钥信息在产品模型详情里不可见,请妥善保管。

- 选择"手动创建",在弹框页面里填写关键参数信息后,单击"确定"。
- 产品名称: 自定义该新建产品的产品名。
- 型号: LoRaWAN设备的型号,可自定义设置。
- 设备类型:非LoRaGateway的其他类型。
- 厂商名称:LoRaWAN设备的厂商名称,可自定义设置。
- 协议类型:选择"LoRaWAN"。
- 步骤2 添加LoRaWAN设备。
 - 1. 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。
 - 2. 选择"子设备"页签,单击右上角的"添加"按钮添加子设备。
 - 3. 在弹框页面里填写关键参数后,单击"确定"。
 - 选择产品:选择<mark>步骤1</mark>中添加的LoRaWAN设备产品模型。
 - 设备名称: 自定义该新建设备的名称。
 - 设备标识码 (devEUI):填写LoRaWAN设备的devEUI,在LoRaWAN设备上获 取 。
 - 接入码:在LoRa Server上注册LoRa网关或LoRWAN设备使用的接入码,由 LoRa服务商提供。
 - 设备工作模式:可选择classA,classB或classC,根据LoRaWAN设备的实际 设置选择。
 - appEUI:LoRa应用ID,购买loRaWAN设备时提供的产品参数单上获取。
 - · 激活方式:可选择ABP或OTAA方式,根据LoRaWAN设备的实际设置选择。
 - appSKey:激活方式选择ABP时需要填写,购买loRaWAN设备时提供的产品 参数单上获取。
 - devAddr: 激活方式选择ABP时需要填写,购买loRaWAN设备时提供的产品 参数单上获取。
 - nwkSKey:激活方式选择ABP时需要填写,购买loRaWAN设备时提供的产品 参数单上获取。
 - appKey:激活方式选择OTAA时需要填写,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。

----结束

4.6.1.3 使用 Modbus 协议接入(联通用户专用)

非联通用户请查看设备接入服务。

概述

Modbus协议目前已经成为工业领域通信协议的业界标准,是工业电子设备之间最常用的连接方式。使用Modbus协议的方式包括直连和非直连两种方式,组网方案如下图所示。



接入方式	方案说明	适用场景
直连接入	Modbus设备直接与边缘节点进行连 接,通过TCP协议通信。边缘节点作为 客户端,Modbus设备作为服务端进行 连接。该方式中,在传输通道绑定设备 时,边缘节点与Modbus设备之间的通 道只能绑定一个设备。	Modbus设备具备TCP通信能力。 说明 Modbus设备具备TCP能力,但 如果IP资源比较受限,无法为每 一个Modbus设备分配一个IP地 址,也可采用非直连方式进行连 接组网。
非直连接 入	Modbus设备通过RTU串口与网关进行 连接,然后边缘节点通过TCP协议与网 关进行通信。网关作为Master节点, Modbus设备作为Slave节点进行连接。 该方式中,在传输通道(边缘节点与网 关之间的TCP传输通道)绑定设备时, 需要将该网关下的Modbus设备绑定到 指定的通道,并且,在同一个传输通道 中,通过"从站号"区分不同的 Modbus设备。	Modbus设备不具备TCP通信 能力,只能通过RTU(串口) 方式进行通信。

在数据采集时,边缘节点可根据指定的数据采集间隔时间,主动向Modbus设备或通过 网关向Modbus设备进行数据的采集,数据格式为二进制格式。数据采集完成后,IoT 边缘服务将采集的数据规整(Json格式)后,通过MQTT的方式将数据上报到设备管 理。

接入流程

业务场景	操作步骤	说明	
设备准备	准备Modbus网关	购买支持Modbus协议的网关(Modbus设 备与网关支持通过RTU通信,网关与边缘节 点采用TCP通信)	
	准备Modbus设备	购买支持Modbus协议的传感器设备	
物联网平台 侧操作	开发Modbus设备产品 模型	在产品中心开发Modbus设备的产品模型	
	添加Modbus设备	在边缘节点下,添加Modbus设备	
	创建Modbus通道和绑 定设备	将Modbus设备绑定到指定的TCP传输通道	

开发 Modbus 设备产品模型

产品模型(也称Profile)是用于描述Modbus设备具备的能力,通过定义产品模型,在 物联网平台构建一款设备的抽象在线开发Profile型,使平台理解该款Modbus设备支 持的服务、属性、命令等信息,如温度、电量等。Modbus设备的产品模型定义分为两 部分:包含Modbus设备的能力(图中红色框内容)和定义Modbus点位表(图中蓝色 框内容),如图所示。属性和命令的定义请根据在线开发Profile,在定义属性和命令 的基础上,再定义Modbus点位表,下面将详细介绍点位表的定义方法。

新增属性		×
* 名称		
* 数据类型		
int		•
*最小值	* 最大值	
0	65535	
步长	单位	

新增扩展描述

modbus读功能码	modbus写功能码
* 寄存器地址	
*奇存器数量	
* 交换寄存器内高低字节	
true	•
* 交换寄存器顺序	
true	•
* 缩放因子	
*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	
确定	取消

步骤1 登录物联网平台控制台,点击右上角进入"开发中心"。

步骤2 定义Modbus设备的属性和Modbus点位表,属性的定义请参考在线开发Profile文件,然后定义定位表(上图中的新增扩展描述)字段,原则如下表所示。

参数名称	参数说明
Modbus读功 能码	读功能码分为"比特访问(Bit Access)"和"16比特访问(16 Bit Access)",功能码均为十进制。 比特访问功能码: • 01:读线圈(Read Coils) • 02:读输入离散量(Read Discrete Inputs) 16比特访问功能码: • 03:读多个寄存器(Read Holding Registers)
	● 04:读单个寄存器(Read Input Register)
Modbus写功 能码	 写功能码分为"比特访问(Bit Access)"和"16比特访问(16 Bit Access)",功能码均为十进制。 说明 定义属性时,无需填写该字段。定义命令时,需要填写。 比特访问功能码: 05:写单个线圈(Write Single Coil) 15:写多个线圈(Write Multiple Coils) 16比特访问功能码: 06:写单个寄存器(Write Single Register) 16:写多个寄存器(Write Multiple Registers)
寄存器起始地 址	寄存器起始地址,占16比特,例如:00 01
寄存器数量	属性对应的数据所在的寄存器个数,占16比特,例如:00 02
交换寄存器内 高低字节	是否对寄存器内的数据进行高低位交换,默认:True。例如寄存器 中存储的属性数据为:10011011,则loT边缘节点获取的数据经过 高低位交互后为:01100111。
交互寄存器顺 序	是否对寄存器的位置进行交换,默认:True。例如寄存器的起始地 址为0001,寄存器数量为0002,则将0002地址寄存器与0001地址 寄存器进行交换。
缩放因子	对寄存器内的数据进行乘以缩放因子,得到所需的数据,例如获取 的温度数据为365,缩放因子为0.1,则得到实际的温度数据为 365*0.1=36.5

- 步骤3 定义Modbus设备的属性和Modbus点位表,命令的定义请参考在线开发Profile文件,然后定义定位表(上图中的新增扩展描述)字段,点位表的填写原则如步骤2中的表格所示。
- **步骤4** 定义完属性和命令后,请将Profile文件进行导入,然后根据添加Modbus设备操作,在设备管理服务中导入开发的Modbus设备产品模型。

产品开发 > Modbus					ala da anga	
Modbus H20P	产品型号 ⊙ Modbus01 所属行业 智益城市	设备类型① DoorLock 厂商名称 HW	数据格式 二进制际级 厂商D ① 46/5b3de957e4ab6b4e &22A	协议类型 modbus 39e34311acf73	じ 編編属性 🛛	查看产品详情
维护能力配置						
软件升级 () 软化	中升级协议 PCP		图件升级 🔵 图件升级协议			经交
服务列表 ⑦					十新建服务	出导出Profile
服务名称	ervice ID	描述		最后修改时间		
✓ Lock_Status				-		к.

----结束

添加 Modbus 设备

前提条件:

- 已完成loT边缘应用的部署。
- 已在开发中心完成Modbus设备产品模型开发。

操作步骤:

部署完loT边缘应用后,在设备管理服务控制台的设备列表中,会自动生成边缘节点设 备,现在需要在边缘节点设备下,添加Modbus设备。

- 步骤1 登录物联网平台控制台,点击右上角"进入设备管理服务"。
- **步骤2** 单击左下角 · · ,在"产品模型"界面中,单击右上角的"新增产品模型",选择 "本地导入",将Modbus设备的产品模型导入到设备管理中。
- 步骤3 选择IoT边缘节点所属的行业应用,并在"设备 > 所有设备"中,找到边缘节点设备。
- **步骤4** 单击边缘节点设备,进入设备详情界面,在"子设备"页签中,单击"添加"按钮, 创建Modbus设备。
- **步骤5** 在弹出的对话框中,选择上传的Modbus设备的产品模型,填写设备名称后,单击"确定",完成Modbus设备的添加。添加完成后,Modbus设备的状态为"离线"。

🛄 说明

Modbus设备的状态由IoT边缘节点进行上报,如果IoT边缘节点不能正常上报子设备的状态信息 到设备管理服务,则展示的子设备状态不会刷新。

- 在线: 设备管理服务采集到Modbus设备上报的数据后,状态就会刷新为在线。
- 离线:如果IoT边缘节点连续5个数据采集间隔无法获取Modbus设备的数据,则IoT边缘节点 上报Modbus设备的状态为离线,设备管理服务将设备的状态刷新为离线。

----结束

为 Modbus 设备绑定通道

根据概述中的Modbus接入组网方案可以知道,Modbus设备在接入loT边缘节点时,需 要将Modbus设备绑定到具体的通道中。如果Modbus设备与loT边缘节点直连,则该通 道只能绑定该设备;如果Modbus设备与loT边缘节点通过网关连接,则需要创建loT边 缘节点与网关之间的传输通道,并将Modbus设备绑定到指定的通道中。 步骤1 在边缘节点设备的详情界面中,选择"配置"页签。

步骤2 单击右上角的"通道"按钮,在弹出的对话框中单击"添加"按钮,填写通道信息后单击"确定",完成TCP传输通道的添加。

参数名称	参数说明
通道名称	定义IoT边缘节点与Modbus设备或网关之间的传输通道。如: Channel_TCP01
传输模式	固定选择"TCP"。
IP地址	loT边缘节点与Modbus设备或网关建立TCP传输通道时,loT边缘节 点作为客户端,Modbus设备或网关作为服务端,因此该地址需要 填写为Modbus设备(直连)或网关(Modbus设备非直连接入loT 边缘节点)的IP地址。如:192.168.10.11
端口	TCP服务端(Modbus设备或网关)的端口号。如:8000

🛄 说明

Modbus设备通过网关接入到IoT边缘节点时,如果添加设备前,已经创建了相应的通道,可以 不用再执行该步骤,直接按照<mark>步骤3</mark>绑定Modbus设备即可。

- 步骤3 在"配置"页签中,单击"绑定子设备"按钮,然后单击"添加"按钮。
- **步骤4** 在弹出的添加子设备配置页面中,选择需要绑定的Modbus设备,然后点击"下一步"。
- **步骤5** 在弹出的对话框中,填写相关信息后,单击"提交",完成Modbus设备的通道绑定。 配置完成后,设备管理服务会将配置信息下发给loT边缘节点。

参数名称	参数说明
关联通道	选择需要将Modbus设备绑定的通道,如:Channel_TCP01
从站号	用于标识同一个通道下的不同Modbus设备。填写原则:需要与 Modbus设备规划的从站号保持一致。
数据采集间隔	loT边缘节点采集Modbus设备数据的时间间隔,单位为:秒,最小 采集间隔可设置为 1 秒。请根据Modbus设备数据采集的实际周期 进行灵活设置。

----结束

执行完以上操作后,Modbus设备上电接入到loT边缘节点后,经过一个数据采集周期 即可在设备管理服务的设备列表中查看采集的设备数据。

4.6.1.4 使用 Agent SDK 接入

4.6.1.4.1 Agent Lite SDK 使用指南(C)(联通用户专用)

非联通用户请查看设备接入服务。

按照本文档的指导,开发者可以体验直连设备通过集成Agent Lite快速接入平台,体验 "数据上报"、"命令接收"、"添加非直连设备"等功能。

Agent Lite以SDK的形式嵌入第三方软件中。本文档以Agent Lite Linux Demo为例, 指导开发者使用Agent Lite SDK中的接口,实现"直连设备登录"、"数据上报"和 "命令下发"等功能。

- 开发者可以基于Agent Lite Linux Demo开发,也可参考Agent Lite Linux Demo,自行集成Agent Lite SDK(Linux)。
- Agent Lite Linux Demo使用的IDE工具为**Visual Studio**。开发者可以使用其他 IDE工具。

使用必读

开发环境要求:

操作系统	工具链	
当前支持的系统:	需要支持以下工具链之一:	
ARM Linux (Embedded Linux)	• gcc-linaro-arm-linux-gnueabihf-	
MIPS Linux (Embedded Linux)	raspbian	
• x86 Linux	 arm-none-linux-gnueabi 	
• x86_64 Linux	arm-linux-uclibceabi	
• x86 Windows		
• x86_64 Windows		

Demo工程目录结构及文件说明:

目录结构	目录	说明
Demo	demo.c	demo源文件
	demo.h	demo头文件
⊢—demo.c ⊢—demo.h	*.h	Agent Lite头文件
*.h	libcrypto.so/libssl.so	openssl库文件
├──libcrypto.so/libssl.so ├──libuspsdk.so	libuspsdk.so	Agent Lite编译后的库文 件
└—conf	conf	存放TLS证书文件

🛄 说明

如果开发者没有设备,可以直接在X86 Linux系统进行开发。

交叉编译环境检测:

步骤1 准备网关或设备。本文档将以树莓派为例,说明如何集成网关。

步骤2 将Agent Lite SDK (Linux)解压到本地。

DataCenter (D:) ▼ OceanConnect ▼ Demo ▼			
1. 1 .▲	i i i i i i i i i i i i i i i i i i i		
🔑 Huawei_IoT_Agent_Lite_SDK_Linux	2017/10/31 11:34		
🗎 Huawei_IoT_Agent_Lite_Demo_C_Linux.zip	2017/10/31 9:36		
📒 Huawei_IoT_Agent_Lite_SDK_Linux.zip	2017/10/30 10:59		

步骤3 用sftp工具把mytest测试工具上传到树莓派上。

因为使用的设备是树莓派,所以使用RaspberryPi目录下的mytest测试工具。开发者可以选择根据实际情况选择不同目录下的mytest测试工具进行测试,尽量选择与自己的 系统信息相近的目录下的测试工具进行测试。

• RaspberryPi ×		• •	×
← → Connect\Demo\ <mark>Huawei_IoT_Ag</mark>	ent_Lite_SDK_Linux\linux\test\R	aspberryPi 🔻 🖬 😋 🔺	■ 🔿 퉬 /opt/agentLite
名称 🔺	大小 美型 修	改时间 名科	尔 🔺
<u></u> .	文件夹		
mytest	707KB 文件 20	17/6/7, 10:54 📂 🚺	mytest

步骤4 修改所有文件的权限,进入mytest所在目录,运行"mytest"。

cd /opt/agentLite chmod -R 777 * ./mytest

如果最后出现"AUTO TEST END SUCC"字样,说明通过检测。

----结束

导入样例代码

在交叉编译环境检测通过后,再进行设备集成开发。下面将以Agent Lite Gateway Demo(C)中的样例代码为例,说明如何进行Agent Lite SDK的集成。

步骤1 将Agent Lite Demo(C-Linux)压缩包中的Agent Lite Gateway Demo(C)解压到本 地。

DataCenter (D:) ▼ OceanConnect ▼ Demo ▼	
共享 ▼ 新建文件夹	
名称 ▲ Agent Lite Gateway Demo(C) Huawei_IoT_A_ent_Lite_SDK_Linux Huawei_IoT_Agent_Lite_Demo_C_Linux.zip Huawei_IoT_Agent_Lite_SDK_Linux.zip	Huawei_IoT_Agent_Lite_Demo_C_Linux.zip - WinRAR File Commands Tools Favorites Options Help Add Extract To Test View Delete F Huawei_IoT_Agent_Lite_Demo_C_Linux.zip
	Agent Lite API Demo(C) Agent Lite Gateway Demo(C)

步骤2 将SDK库中的头文件和so库都放到"Agent Lite Gateway Demo(C)/linux/demo"目录下。





🛄 说明

因为使用的设备是树莓派,所以使用RaspberryPi目录下的库。

步骤3 打开Visual Studio,选择"File > Open > Project/Solution"。

X	Microsoft Visual Studio (Administrator)											
File	Edit	View	Debug	Team	Tools	Test	Ana	lyze	Window	Help		
	New					•	×86		-	🕨 Attach י	- 🏓 🗕	
	Open					•	ð	Proj	ect/Solution		Ctrl+Shift+C	2
	Close						4	Web	Site		Shift+Alt+O	
×	Close S	olution						Оре	n from Sou	rce Control		
	Save Se	lected It	ems		Ctrl+S		ta.	Tear	m Project			
	Save Se	lected It	ems As				e	File.			Ctrl+O	
1 2	Save All				Ctrl+Sh	ift+S		Com	vert			

步骤4 进入"Agent Lite Gateway Demo(C)/linux"目录,选择"agentLiteDemo.sln",点击"打开"按钮即可。

在Solution Explorer窗口中,双击打开"demo.c"文件。



-----结束

初始化

在发起业务前,需要先初始化Agent Lite相关资源,调用API接口**IOTA_Init()**,初始化 Agent Lite资源。具体API接口的参数使用请参考Agent Lite API接口文档。可参考 demo.c中**main()**方法对**IOTA_Init()**的调用。

IOTA_Init(const CONFIG_PATH,const HW_NULL);

- "CONFIG_PATH"为工作路径,不能为空,该参数必须带结束符'\0'。
- 第二个参数为打印日志路径,当它为空时,打印路径默认为工作路径。开发者也可以自己定义打印日志路径,该参数必须带结束符'\0'。

绑定和登录



设备或网关第一次接入物联网平台时需要进行绑定操作,从而将设备或网关与平台进 行关联。开发者通过传入设备序列号以及设备信息,将设备或网关绑定到物联网平 台。

设备或网关绑定成功后或重启后,需要进行登录的流程,在设备或网关成功登录物联 网平台后,才可以进行其它服务操作,比如接入其他传感器,数据上报等等。如果设 备或网关登录成功,那么设备或网关在平台的状态显示为已在线。

步骤1 修改绑定参数。

绑定时使用的设备固有信息(如设备型号等)是从"gwreginfo.json"文件中读取的, 所以需要修改demo目录下"gwreginfo.json"文件中的如下信息:

- "platformaddr":物联网平台的设备对接地址(MQTTS),可参考平台对接信 息获取。
- "mac": MAC地址,每个设备对应一个MAC地址,不可重复,所以建议使用 IMEI或者MAC地址等天然的设备标识。测试时只要输入一个没有使用过的MAC地 址即可。

🛄 说明

- 如果开发者通过"设备管理服务控制台"注册设备,则"mac"填写为设备注册时的 "preSecret"(预置密钥)。
- 如果通过开发中心注册设备,则"verifyCode"填写为设备注册时设置的"nodeld" (设备标识)。
- "manufacturerId"(厂商Id)、"deviceType"(设备类型)、"model"(设 备模型)和"protocolType"(协议类型)与Profile文件中的定义保持一致。

"gwreginfo.json"文件中设备固有示例:

```
"mac":"1234567",
"platformAddr":"127.0.0.1",
"platformPort":8943,
"manufacturerId":"Huawei",
"deviceType":"Gateway",
"model":"AgentLite01",
"protocolType":"HuaweiM2M",
"loglevel":255
```

步骤2 绑定设备。

{

绑定前先调用API接口IOTA_ConfigSetXXX()设置物联网平台的IP与端口。

IOTA_ConfigSetStr(EN_IOTA_CFG_IOCM_ADDR, const pucPlatformAddr); IOTA_ConfigSetUint(EN_IOTA_CFG_IOCM_PORT, uiPort);

注册广播接收器对设备绑定结果进行相应处理。

HW_BroadCastReg(IOTA_TOPIC_BIND_RSP, Device_RegResultHandler);

调用API接口**IOTA_Bind()**进行设备绑定,主要入参为MAC地址和必要的设备信息,包括"nodeld"(设备标识码)、"manufacturerId"(厂商Id)、"deviceType" (设备类型)、"model"(设备模型)和"protocolType"(协议类型),其中 MAC地址与"nodeld"的值保持一致。

HW_UINT DEVICE_BindGateWay()

```
stDeviceInfo.pcMac = HW_JsonGetStr(json, IOTA_DEVICE_MAC);
stDeviceInfo.pcNodeId = stDeviceInfo.pcMac;
stDeviceInfo.pcManufacturerId = HW_JsonGetStr(json, IOTA_MANUFACTURE_ID);
stDeviceInfo.pcDeviceType = HW_JsonGetStr(json, IOTA_DEVICE_TYPE);
stDeviceInfo.pcModel = HW_JsonGetStr(json, IOTA_MODEL);
stDeviceInfo.pcProtocolType = HW_JsonGetStr(json, IOTA_MODEL);
stDeviceInfo.pcProtocolType = HW_JsonGetStr(json, IOTA_PROTOCOL_TYPE);
.....
IOTA_ConfigSetStr(EN_IOTA_CFG_IOCM_ADDR, pucPlatformAddr);
IOTA_ConfigSetUint(EN_IOTA_CFG_IOCM_PORT, uiPort);
IOTA_Bind(stDeviceInfo.pcMac, &stDeviceInfo);
```

设备或网关绑定成功,后续就不需要再绑定了,除非设备或网关被删除,才需要重新 绑定。

设备绑定成功会收到广播,广播内容请参考Agent Lite API接口文档中设备绑定接口的 返回结果说明和demo中Device_RegResultHandler函数的处理。 Device_RegResultHandler把从广播中得到的网关设备信息保存在 "gwbindinfo.json"文件中。

步骤3 配置登录参数。

登录前需要通过参数配置API接口IOTA_ConfigSetXXX()传入所需的登录信息。

- "设备Id"("EN_IOTA_CFG_DEVICEID"), "appld"
 ("EN_IOTA_CFG_APPID")和"密码"
 ("EN_IOTA_CFG_DEVICESECRET")从绑定成功的广播中得到的。
- "HTTP地址"("EN_IOTA_CFG_IOCM_ADDR")和"MQTT地址"
 ("EN_IOTA_CFG_MQTT_ADDR")一般为同一个地址,可以从绑定成功的广播 中得到。一般情况下,这个地址和Agent Lite设备或网关对接的平台地址一致。
- 绑定成功的广播参数获取可以参考Device_RegResultHandler函数的处理。

IOTA_ConfigSetStr(EN_IOTA_CFG_DEVICEID, g_stGateWayInfo.pcDeviceID); IOTA_ConfigSetStr(EN_IOTA_CFG_IOCM_ADDR, g_stGateWayInfo.pcIOCMAddr); IOTA_ConfigSetStr(EN_IOTA_CFG_APPID, g_stGateWayInfo.pcAppID); IOTA_ConfigSetStr(EN_IOTA_CFG_DEVICESECRET, g_stGateWayInfo.pcSecret); IOTA_ConfigSetStr(EN_IOTA_CFG_MQTT_ADDR, g_stGateWayInfo.pcIOCMAddr); IOTA_ConfigSetUint(EN_IOTA_CFG_MQTT_PORT, g_stGateWayInfo.pcMqttPort); IOTA_ConfigSetUint(EN_IOTA_CFG_IOCM_PORT, g_stGateWayInfo.pcIOCMPort);

注册广播接收器对设备登录结果进行相应处理。

HW_BroadCastReg(IOTA_TOPIC_CONNECTED_NTY, Device_ConnectedHandler);

步骤4 设备登录。

调用API接口**LoginService.login()**进行直连设备登录,具体API的参数使用参考Agent Lite API接口文档中设备登录接口的说明。

IOTA_Login();

----结束

编译并运行程序

绑定和登录功能完成后,可以先测试一下设备或网关是否能正常与平台对接,再进行 后续功能开发。

步骤1 用sftp工具把demo目录上传到树莓派上。

e linux ×	4	
← → D:\OceanConnect\Demo	Agent Lite Gateway Demo(C)\linux 🗾 🐨	O ← →]] /opt/agentLite
188: - 213	美型 (1000)	(4)目 ▲
🏨	文件夹	D
\mu .vs	文件夹 2017/11/17, 15:54	🗩 🕕 demo
<u>]]</u> Debug	文件主 2017/11/17, 15:54	
🖟 demo	文件夹 2017/11/17, 15:55	
agentLiteDemo.sln 1KB	Microsoft 2017/11/17, 15:54	

步骤2 进行交叉编译。

- 1. 进入"/opt/agentLite/demo/"路径。
 - "/opt/agentLite/demo/"路径需要根据实际情况修改。
- 2. 输入 "chmod -R 777 *" 命令修改权限。
- 3. 输入 "export LD_LIBRARY_PATH=./" 设置程序共享库位置。
- 4. 输入 "gcc demo.c -L /opt/agentLite/demo/ -lpthread -ldl -lrt -luspsdk -lssl lcrypto -o sdk.out"命令,如果交叉编译成功,会产生sdk.out库文件。
 - "luspsdk"、"lssl"和"lcrypto"分别表示库文件"libuspsdk.so"、
 - "libssl.so"和"libcrypto.so",具体linux交叉编译命令gcc请开发者自行学习。

root@raspberrypi:/ \$cd /opt/agentLite/demo/
root@raspberrypi:/opt/agentLite/demo \$chmod -R 777 *
root@raspberrypi:/opt/agentLite/demo \$export LD LIBRARY PATH=,/
root@raspberrypi:/opt/agentLite/demo \$gcc demo.c -L /opt/agentLite/demo/ -lpthread -ldl -lrt -luspsdk -lssl -lcrypto -o sdk.ou
root@raspberrypi:/opt/agentLite/demo \$

步骤3运行Demo。

启动demo命令为"./ sdk.out",输入该命令程序就能运行起来了。

程序运行后,Agent Lite SDK就会主动发送绑定消息给平台。因为此时还没有使用应用侧接口在平台上注册此设备,所以会提示"The device has not registered yet"。

----结束

上传 Profile 并注册设备

- 下载Profile开发示例,并上传模板中的profile文件:
- "Gateway_Huawei_AgentLite01.zip"和"Motion_Huawei_test01.zip"。
- **步骤1** 登录开发中心,创建一个项目,在该项目空间内,选择"产品 > 产品开发",点击 "新建产品"。



步骤2 在"创建产品"中,选择"本地导入产品创建",单击"上传Profile"上传 "Gateway_Huawei_AgentLite01.zip"和"Motion_Huawei_test01.zip"。

产品开发 > 创建产品					
			请选择以1	下一种方式创建您的	产品
基于系统模板创建	基于快速集成模板创建	基于已有产品创建	本地导入产品创建	自定义产品	
				₽_ ☆/告↓/左羽玉/、	
		心可	以将平地与好的Pro	mie又件上传到半台	,
				上传Profile	

步骤3 选择"产品 > 设备管理",单击"添加真实设备",进入"新建增实设备"页面。



步骤4 根据向导注册设备。

1. 选择产品。

产品: AgentLite001

新建真实设备		×
01		02 注册设备
$\overset{O}{\swarrow}$	\sim	
AgentLite001 Huawei	test001 Huawei	

- 2. 填写设备相关信息,单击"确定"。
 - 设备名称: AgentLiteDevice
 - 设备标识: aaa123456,需要与AgentLiteDemo中网关的设备标识一致。
 - 接入方式: 网关 (Agentlite)

新建真实设备		
01		02
1559年—1377年前6		注册设备
	*设备名称	
	AgentLiteDevice	
	*设备标识 aaa123456	
	接入方式) 直连 · 网关(Agentile)	
	上一步 确定	

步骤5 注册设备后,Agent Lite SDK发送bind消息,则在开发中心的"产品 > 设备管理 > 设备列表"界面可以看到设备状态变成"在线"。

▲ 项目概览	设备列表			
 ● 产品 ▲ 产品开发 	设备类型	设备状态 ①	虚拟&真	实
设备管理 升级调试	Тор 3		100% 	耳实设备
₽ 应用 ·	• GateWay	1 100% • H#	o%. U o%.	1
	设备列表		Q 请输入设备名称、设备ID	+新增真实设备 +新增虚拟设备
	状态 设备名称	设备ID 所屬产品	型号 设备类型	
	●在线 AgentLiteDevice	41865bab-895f-453e-bfe3-9eb68b5c348f AgentLile01	AgentLite01 GateWay	續試产品 續試应用 删除

如果demo日志中出现"lota_BindDestroy",说明Agent Lite SDK多次发送bind消息,并且没有收到正确响应,设备绑定已超时。这时需要再次运行"sdk.out"程序 (先用"ctrl+c"退出,再用"./sdk.out"运行程序),Agent Lite SDK就会再发起 bind消息。平台收到正确的bind消息后设备就会变成"在线"状态。

----结束

数据上报和数据发布

设备或网关向物联网平台上报数据可以通过调用SDK的"设备服务数据上报"接口或 "数据发布"接口:

 "设备服务数据上报"接口:pcDeviceId、pcRequstId和pcServiceId由SDK组装 为消息的header;pcServiceProperties由SDK组装为消息的body。消息组装格式 为JSON。

设备或网关登录成功后可以调用IOTA_ServiceDataReport接口上报数据。

- 当设备主动上报数据时,"pcRequstId"可以为空。
- 当上报的数据为某个命令的响应时, "pcRequstId"必须与下发命令中的 requstId保持一致。 "pcRequestId"可以从广播中获取,请参考API文档中 "设备命令接收"接口的"EN_IOTA_DATATRANS_IE_REQUESTID"参数。
- "pcServiceID"要与Profile中定义的某个"serviceId"保持一致,否则无法 上报数据。

```
    "pcServiceData"实际上是一个JSON字符串,内容是健值对(可以有多组
健值对)。每个健是Profile中定义的属性名(propertyName),值就是具体
要上报的数据。
    HW_VOID Gateway_DataReport(HW_CHAR **pcJsonStr)

    HW_JSON json;
HW_JSONOBJ hJsonObj;
    hJsonObj = HW_JsonObjCreate();
json = HW_JsonGetJson(hJsonObj);
    HW_JsonAddUint(json, (HW_CHAR*)"storage", (HW_INT)10240);
HW_JsonAddUint(json, (HW_CHAR*)"usedPercent", (HW_INT)20);
*pcJsonStr = HW_JsonEncodeStr(hJsonObj);
    Device_ServiceDataReport(g_stGateWayInfo.pcDeviceID, "Storage", pcJsonStr);
//another method
```

//Device_ServiceDataReport(pcDeviceId, "Storage", "{\"storage\":20240,\"usedPercent\":20}");

HW_INT **Device_ServiceDataReport**(const HW_CHAR *pcSensorDeviceID, const HW_CHAR *pcServiceId, const HW_CHAR *pcServiceProperties)

HW_CHAR aszRequestId[BUFF_MAX_LEN];

HW_GetRequestId(aszRequestId);

```
if (HW_TRUE != g_uiLoginFlg)
{
    //TODO e.g. resend service data
    HW_LOG_INF("Device_MApiMsgRcvHandler():GW discon,pcJsonStr=%s", pcServiceProperties);
    return HW_ERR;
}
```

IOTA_ServiceDataReport(HW_GeneralCookie(), aszRequestId, pcSensorDeviceID, pcServiceId, pcServiceProperties);

```
return HW_OK;
```

注册广播接收器对网关数据上报结果进行相应处理。广播过滤参数为 "IOTA_TOPIC_DATATRANS_REPORT_RSP/{deviceId}"。

sprintf(acBuf, "%s/%s", IOTA_TOPIC_DATATRANS_REPORT_RSP, g_stGateWayInfo.pcDeviceID); HW_BroadCastReg(acBuf, Device_ServiceDataReportResultHandler);

"数据发布"接口:topic固定为"/cloud/signaltrans/v2/categories/data";
 "pbstrServiceData"参数作为消息体(包括header和body),SDK只进行透传,不进行格式调整和组装。

设备或网关登录成功后可以调用IOTA_MqttDataPub接口发布数据。

- "pucTopic"为发布数据的topic。
- "uiQos"为mqtt协议中的一个参数。
- · "pbstrServiceData"实际上是一个json字符串,内容是健值对(可以有多组 健值对)。每个健是profile中定义的属性名(propertyName),值就是具体 要上报的 。

HW_INT Device_ServiceDataPub(const HW_UCHAR *pucTopic, HW_UINT uiQos, const HW_BYTES *pbstrServiceData)

IOTA_MqttDataPub(HW_GeneralCookie(),pucTopic,uiQos,pbstrServiceData); return HW_OK;

注册广播接收器对网关数据上报结果进行相应处理。广播过滤参数为 "IOTA_TOPIC_DATATRANS_PUB_RSP"。

HW_BroadCastReg(IOTA_TOPIC_DATATRANS_PUB_RSP, Device_ServiceDataPubResultHandler);

命令接收

当开发者希望设备或网关只接收topic为"/gws/deviceid/signaltrans/v2/categories/"的消息,且对消息中的header进行解析时,可以调用"设备命令接收"接口

应用服务器可以调用物联网平台的应用侧API接口给设备或网关下发命令,所以设备或 网关需要随时检测命令下发的广播,以便在接收到命令时进行相应业务处理。

注册IOTA_TOPIC_SERVICE_COMMAND_RECEIVE/{deviceId}广播接收器对命令下发 进行相应处理。

sprintf(acBuf, "%s/%s", IOTA_TOPIC_SERVICE_COMMAND_RECEIVE, g_stGateWayInfo.pcDeviceID); HW_BroadCastReg(acBuf, Device_ServiceCommandReceiveHandler);

具体的处理函数Device_ServiceCommandReceiveHandler。

HW_INT Device_ServiceCommandReceiveHandler(HW_UINT uiCookie, HW_MSG pstMsg) HW_CHAR *pcDevId; HW_CHAR *pcReqId; HW CHAR *pcServiceId; HW_CHAR *pcMethod; HW_BYTES *pbstrContent; pcDevId = HW_MsgGetStr(pstMsg,EN_IOTA_DATATRANS_IE_DEVICEID); pcReqId = HW_MsgGetStr(pstMsg,EN_IOTA_DATATRANS_IE_REQUESTID); pcServiceId = HW_MsgGetStr(pstMsg,EN_IOTA_DATATRANS_IE_SERVICEID); pcMethod = HW_MsgGetStr(pstMsg,EN_IOTA_DATATRANS_IE_METHOD); pbstrContent = HW_MsgGetBstr(pstMsg,EN_IOTA_DATATRANS_IE_CMDCONTENT); HW LOG_INF(" ------ CommandReceiveHandler ------ DeviceId=%s", pcDevId); if ((HW_NULL == pcDevId) ||(HW_NULL == pcReqId) ||(HW_NULL == pcServiceId) ||(HW_NULL == pcMethod)) { HW_LOG_ERR("RcvCmd is invalid, pcDevId=%s, pcReqId=%s, pcServiceId=%s, pcMethod=%s.", pcDevId, pcReqId, pcServiceId, pcMethod); return HW_ERR; } if (0 == strncmp(METHOD_REMOVE_GATEWAY,pcMethod,strlen(METHOD_REMOVE_GATEWAY))) ł IOTA_RmvGateWay(); } return HW_OK; 在开发中心的"产品 > 设备管理"界面中,单击非直连设备列后的"删除"按钮,这

在开友中心的"产品 > 设备管理" 界面中,里击非直连设备列后的"删除"按钮,这 样就能在demo界面上看到广播接收时的日志打印命令下发。

非直连设备的删除需要网关的确认,正常业务情况下,网关又需要跟具体的设备确 认,所以网关收到删除非直连设备的命令也不会立刻将设备删除。

添加非直连设备

在添加非直连设备前,确认非直连设备的profile已经上传,详见**上传Profile并注册设** <mark>备</mark>步骤 。

在设备或网关登录成功后,可以调用IOTA_HubDeviceAdd接口添加非直连设备。

此处非直连设备的设备固有信息是测试数据。真实情况下,网关需要与具体的非直连设备交互,才能得到非直连设备的固有信息。

HW_VOID AddSensors()

```
ST_IOTA_DEVICE_INFO stDeviceInfo = {0};
FILE *fp = NULL;
HW_CHAR szdeviceInfoFileName[BUFF_MAX_LEN] = {0};
HW INT file size;
HW_CHAR *pcJsonStr;
HW_JSONOBJ jsonObj;
HW_JSON json;
//get device info
stDeviceInfo.pcNodeId = "SN Number_8532157";
stDeviceInfo.pcManufacturerName = "Huawei";
stDeviceInfo.pcManufacturerId = "Huawei";
stDeviceInfo.pcDeviceType = "Motion";
stDeviceInfo.pcModel = "test01";
stDeviceInfo.pcProtocolType = "Z-Wave";
IOTA_HubDeviceAdd(g_uiCookie, &stDeviceInfo);
return;
```

注册广播接收器对添加设备结果进行相应处理。添加非直连设备成功后就能从广播中得到非直连设备的deviceld。

HW_BroadCastReg(IOTA_TOPIC_HUB_ADDDEV_RSP, Device_AddResultHandler);

非直连设备添加成功后可以在"设备列表"中看到新增一条记录。

🗀 说明

Demo在设备添加成功一段时间后会再调用删除设备接口进行设备删除。如果再次运行sdk.out 前,还没执行删除设备接口,再次添加相同设备会失败。可以修改"Nodeld"的值或者调用设 备删除接口把原来的设备删除掉再进行测试。

 VERMAX
 VERMAX

 P PG
 PG

 P PG
 PG

 P PG
 VERMAX

 P PG

非直连设备状态更新

非直连设备添加上时,一般情况下是"离线"状态。所以在非直连设备添加成功后, 或者在非直连设备上报数据前,要调用IOTA_DeviceStatusUpdate()进行设备状态更 新。

IOTA_DeviceStatusUpdate(g_uiCookie, g_cDeviceId, "ONLINE", "NONE");

AgentLiteDemo中只添加了一个非直连设备,所以**IOTA_DeviceStatusUpdate**中的 **g_cDeviceId**直接使用全局变量。

注册广播接收器对非直连设备状态更新结果进行相应处理。

HW_BroadCastReg(IOTA_TOPIC_DEVUPDATE_RSP, Device_DevUpDateHandler);

非直连设备数据上报

请参考<mark>数据上报和数据发布</mark>章节,调用IOTA_ServiceDataReport或 IOTA_MqttDataPub接口进行数据上报, 各个参数使用非直连设备的相关数据即可, 此处不再复述。

设备数据上报成功后,可以在非直连设备的"历史数据"中查看上报的数据。

▲ 项目概览	我的设备 >			
● 产品 🔹	设备详情 历史数据 设计	新日志 历史命令		
产品开发				の思新
设备管理	服务	2013 15举作作	时间	
升级调试	Battery	{ "batteryLevel": 98 }	2019/05/28 14:22:10	
■ 应用 🔹	Motion	("motion": "DETECTED")	2019/05/28 14:22:10	

4.6.1.4.2 Agent Lite SDK 使用指南(Java)(联通用户专用)

非联通用户请查看设备接入服务。

按照本文档的指导,开发者可以体验直连设备通过集成Agent Lite快速接入平台,体验 "数据上报"、"命令接收"、"添加非直连设备"等功能。

Agent Lite以SDK的形式嵌入第三方软件中。本文档以Agent Lite Java Demo为例,指导开发者使用Agent Lite SDK中的接口,实现"直连设备登录"、"数据上报"和 "命令下发"等功能。

- 开发者可以基于Agent Lite Java Demo开发,也可参考Agent Lite Java Demo, 自行集成Agent Lite SDK(Java)。
- Agent Lite java Demo使用的IDE工具为Eclipse。

使用必读

开发环境要求:

使用的SDK版本为jdk1.8.0_45,适用的操作系统为Windows系统。

工程目录结构及文件说明:

目录结构	目录	说明
AgentLiteDemo	src	存放Agentlite Demo代码。
│ │──src │ └──com │ └──huawei │ └──agentlitedemo	libs	存放Agentlite提供的jar包和第三方jar 包,以及.dll动态库(工程必须的三个 文件, "agentlite-0.0.1- SNAPSHOT", "usp_agentlite.jar",".dll"文 件)。
└──agentlite-0.0.1- SNAPSHOT └──usp_agentlite.jar └──.dll └──workdir	workdir	存放工程日志文件,由agentlite初始 化资源设置。

目录结构	目录	说明
└──conf	conf	存放TLS证书文件。

🛄 说明

如果开发者没有设备,可以直接在X86 Linux系统进行开发。

导入代码样例

步骤1 将Agent Lite Demo(Java) 解压到本地。

🌗 AgentLiteDemo	2018/5/9 10:03	文件夹	
RemoteSystemsTempFiles	2018/5/8 17:30	文件夹	
🔚 AgentLiteDemo.rar	2018/5/23 16:42	WinRAR archive	1,401 KB

步骤2 导入AgentLiteDemo工程。

- 打开Eclipse,单击"File > Import"进入导入现有工程界面。
- 选择"General > Existing Projects into Workspace ",点击"Next"。

) Import	
e lect Create new projects from an archive file or directory.	
<u>S</u> elect an import wizard:	
type filter text	
General General General Existing Projects into Workspace	
Preferences	
Projects from Folder or Archive	
🕨 🧀 Git	
👂 🗁 Gradle	
> 🥟 Install	
🖻 🗁 Maven	
🖻 🧀 Oomph	
> 🗁 Run/Debug	
Tasks	
▷ 🗁 Team	
D 🦳 XML	
	Next > Finish Cancel

• 点击"Browse",选择AgentLiteDemo解压后的路径,点击"Finish"。

⊜ Import		- • •
Import Projects		
Select a directory to sear	ch for existing Eclipse projects.	
Select root directory:	D:\AgentLiteDemo_windows_java\Ager 👻	B <u>r</u> owse
○ Select <u>a</u> rchive file:	•	B <u>r</u> owse
<u>P</u> rojects:		
🔽 AgentLiteDemo (I	D:\AgentLiteDemo_windows_java\AgentLite	Select All
		Deselect All
		R <u>e</u> fresh
Ontions		
Search for nested pro	ojects	
Copy projects into we	orkspace	
Hide projects that alr	eady exist in the workspace	
Working sets		
Add projec <u>t</u> to work	ing sets	Ne <u>w</u>
W <u>o</u> rking sets:	•	S <u>e</u> lect



初始化

在发起业务前,需要先初始化Agent Lite相关资源,调用API接口**BaseService.init()**, 初始化Agent Lite资源,具体API的参数使用参考Agent Lite API接口文档。

调用BaseService.init(String workPath, String logPath)初始化AgentLite资源。

res = BaseService.init("./workdir", null);

绑定和登录



设备或网关**第一次**接入物联网平台时需要进行绑定操作,从而将设备或网关与平台进 行关联。开发者通过传入设备验证码以及设备信息,将设备或网关绑定到物联网平 台。

步骤1 修改绑定参数。

绑定时使用的设备固有信息(如设备型号等)是从"AgentLiteBind.java"文件中读取的,所以需要修改./src/main目录下"AgentLiteBind.java"文件中的如下信息:

物联网平台的设备对接地址(MQTTS)和端口,可参考平台对接信息获取。

//设置配置参数

private static final String PLATFORM_IP = "100.100.100.100"; private static final String HTTPS_PORT = "8943";

"verifyCode"(设备验证码)和必要的设备信息,包括"nodeld"(设备标识码)、"manufactureld"(厂商Id)、"deviceType"(设备类型)、"model"(设备模型)和"protocolType"(协议类型)。"verifyCode"的值与"nodeld"保持一致,"manufactureld"(厂商Id)、"deviceType"(设备类型)、"model"(设备模型)和"protocolType"(协议类型)与Profile文件中的定义保持一致。

🛄 说明

- 如果开发者通过"设备管理服务控制台"注册设备,则"verifyCode"填写为设备注册时的 "preSecret"(预置密钥)。
- 如果通过开发中心注册设备,则"verifyCode"填写为设备注册时设置的"nodeld"(设备标识)。

```
public void bindAction() {
```

```
String nodeld = "1234568";

String verifyCode = "1234568";

String manufactrueld = "Huawei";

String deviceType = "Gateway";

String model = "AgentLite01";

String protocolType = "LWM2M";

deviceInfo = new lotaDeviceInfo(nodeld, manufactrueld, deviceType, model, protocolType);

...
```

步骤2 绑定设备。

注册观察者对设备绑定结果进行相应处理。

//注册观察者

```
AgentLiteBind agentLiteBind = AgentLiteBind.getInstance();
BindService bindService = BindService.getInstance();
bindService.registerObserver(agentLiteBind);
//网关绑定
agentLiteBind.bindAction();
```

调用API接口BindConfig.setConfig()设置绑定配置,接着调用API接口 BindService .bind(String verifyCode, lotaDeviceInfo deviceInfo)绑定设备。

```
public void bindAction() {
    System.out.println(" ======== start bind ========= ");
    ...
    //绑定配置
    configBindPara();
    //发起绑定请求
    BindService.bind(verifyCode, deviceInfo);
}
//绑定配置
private static void configBindPara() {
    boolean res = false;
    res = BindConfig.setConfig(BindConfig.BIND_CONFIG_ADDR, PLATFORM_IP);
    res = BindConfig.setConfig(BindConfig.BIND_CONFIG_PORT, HTTPS_PORT);
    ...
}
```

设备或网关绑定成功,后续就不需要再绑定了,除非设备或网关被删除,才需要重新 绑定 。

设备绑定成功会收到**BindService**发出的通知,通知内容请参考Agent Lite API接口文 档中设备绑定接口的返回结果说明和demo中**update**函数的处理。

步骤3 修改登录参数。

在demo的./src/main/AgentLiteLogin.java设置物联网平台的接入IP与端口。

private static final String PLATFORM_IP = "100.100.100.100"; private static final String MQTTS_PORT = "8883"; private static final String HTTPS_PORT = "8943";

步骤4 设备登录。

注册观察者对设备登录结果进行相应处理。

```
//注册观察者
AgentLiteLogin agentLiteLogin = AgentLiteLogin.getInstance();
LoginService loginService = LoginService.getInstance();
loginService.registerObserver(agentLiteLogin);
//网关登录
agentLiteLogin.loginAction();
```

调用API接口**LoginConfig.setConfig()**传入所需的登录信息,接着调用API接口 **LoginService.login()**进行直连设备登录,具体API的参数使用参考Agent Lite接口文 档中的设备登录接口说明。



----结束

上传 Profile 并注册设备

下载**Profile开发示例**,并上传模板中的profile文件: "Gateway_Huawei_AgentLite01.zip"和"Motion_Huawei_test01.zip"。

步骤1 登录开发中心,创建一个项目,在该项目空间内,选择"产品 > 产品开发",点击 "新建产品"。

▲ 项目概范 ● 产品 ▲	01 开发中	222 自助测试中	03 认证发布			
产品开发	• #20+	• ###		• E326	• = # =	
设备管理	Ū	J	v	Ū	0	
升级调试						
对接信息						
订阅调试						
应用调试						
应用在线开发						
		据(共调至De产品开)	£向导和相关开发资源,帮助开发者轻松完成产品开发			
			新建产品			

步骤2 在"创建产品"中,选择"本地导入产品创建",单击"上传Profile"上传 "Gateway_Huawei_AgentLite01.zip"和"Motion_Huawei_test01.zip"。

产品开发 > 创建产品					
			建进权削下	サナゴの地をついて	20
			追应持以下	一种力式的建态的广	- 00
基于系统模板创建	基于快速集成模板创建	基于已有产品创建	本地导入产品创建	自定义产品	
		您可	以将本地写好的Profi	le文件上传到平台,	开发一个新产品
				上传Profile	

步骤3选择"产品 > 设备管理",单击"添加真实设备",进入"新建增实设备"页面。

▲ 项目概览	设备类型	设备状态 💿	虚拟&真实
⑦ 产品 产品开发			
设备管理		在地设备 0 0	ence Xx22 0 0
升级调试			
■ 应用	•		
		The set of	

步骤4 根据向导注册设备。

1. 选择产品。



建真实设备		
01		02
选择一款产品		注册设备
$\overset{\circ}{\sim}$	$\overset{\circ}{\sim}$	
AgentLite001 Huawei	test001 Huawei	

填写设备相关信息,单击"确定"。
 - 设备名称: AgentLiteDevice

- 设备标识: aaa123456,需要与AgentLiteDemo中网关的设备标识一致。
 - 接入方式: 网关(Agentlite)

新建真实设备		
01		02
选择一款产品		注册设备
	*设备名称	
	AgentLiteDevice	
	*设备标识	
	aaa123456	
	接入方式	
	○ 直连	
	上一步 織定	

步骤5 注册设备后,Agent Lite SDK发送bind消息,则在开发中心的"产品 > 设备管理 > 设 备列表"界面可以看到设备状态变成"在线"。

▲ 项目概览	设备列表			
 ● 产品 ▲ 	设备类型	设备状态 ⑦		虚拟8真实
设备管理 升级调试	Тор 3		• 在說 100%	#131段時 再本说後
₩ <u>w</u> 用 •	• GateWay	1 100%	● 异和 0% ● 兩柱 0%	0 1
	设备列表 状态 设备名称	设备D / 所属产品	Q 消除入设备名称、 证 型号 设备关注	1980 +新雄進武政藝 +新雄進初政藝 型
	●在线 AgentLiteDevice	41865bab-895f-453e-bfe3-9eb68b5c348f AgentLite0	AgentLite01 GateWa	ay Wistrian Wistazi Hing

如果demo日志中出现"lota_BindDestroy",说明Agent Lite SDK多次发送bind消息,并且没有收到正确响应,设备绑定已超时。这时需要再次运行"sdk.out"程序 (先用"ctrl+c"退出,再用"./sdk.out"运行程序),Agent Lite SDK就会再发起 bind消息。平台收到正确的bind消息后设备就会变成"在线"状态。

----结束

数据上报和数据发布

设备或网关向物联网平台上报数据可以通过调用SDK的"设备服务数据上报"接口或 "数据发布"接口:

• "设备服务数据上报"接口: deviceId, requstId和serviceId由SDK组装为消息的 header; serviceProperties由SDK组装为消息的body。消息组装格式为JSON。

```
注册观察者对网关数据上报结果进行相应处理。
```

```
//注册观察者
AgentLiteDataTrans agentLiteDataTrans = AgentLiteDataTrans.getInstance();
DataTransService dataTransService = DataTransService.getInstance();
dataTransService.registerObserver(agentLiteDataTrans);
//数据上报
agentLiteDataTrans.gwDataReport();
```

设备或网关登录成功后可以调用DataTransService.dataReport(int cookie, String requstId, String deviceId, String serviceId, String serviceProperties) 接口上报数据。

public void gwDataReport() {

```
System.out.println(" ========= gwDataReport! ======== ");
int cookie;
Random random = new Random();
cookie = random.nextInt(65535);
```

String deviceId = GatewayInfo.getDeviceID();

JsonObject data = new JsonObject(); data.addProperty("storage", "10240"); data.addProperty("usedPercent", "20"); DataTransService.dataReport(cookie, null, deviceId, "Storage", data.toString());

}

数据上报成功后可以在设备的"历史数据"中看到上报的数据了。

▲ 项目概览	我的设备 > aaa123456		
● 产品 -	设备详情 历史数据 设备日志	历史命令	
产品开发			ORIM
设备管理	服务	数编译编	#\$(ii)
升级调试	Storage	("storage": "10240", "usedPercent": "18")	2019/05/28 14:22:02
■ 应用 👻	Storage	("storage": "10240", "usedPercent": "20" }	2019/05/28 14:22:00

"数据发布"接口:topic固定为"/cloud/signaltrans/v2/categories/data";
 "serviceData"参数作为消息体(包括header和body),SDK只进行透传,不进行格式调整和组装。

注册观察者对网关数据上报结果进行相应处理。

//注册观察者

AgentLiteDataTrans agentLiteDataTrans = AgentLiteDataTrans.getInstance(); DataTransService dataTransService = DataTransService.getInstance(); dataTransService.registerObserver(agentLiteDataTrans); //数据发布

agentLiteDataTrans.gwDataReportByMqttDataPub();

设备或网关登录成功后可以调用DataTransService. mqttDataPub(int cookie, String topic, int qos, byte[] serviceData)接口发布数据。

- "topic"是要发布数据的topic。
- "qos"是mqtt协议的一个参数。
- "serviceData"实际上是一个json字符串,内容是健值对(可以有多组健值 对)。每个健是profile中定义的属性名(propertyName),值就是具体要上 报的内容了。

public void gwDataReportByMqttDataPub() {

Random random = new Random(); cookie = random.nextInt(65535);

String deviceId = GatewayInfo.getDeviceID();

JsonObject headerData = new JsonObject(); headerData.addProperty("method", "PUT"); String fromStr = "/device/"+deviceld+"/services/Storage"; String toStr = "/data/v1.1.0/devices/"+deviceld+"/services/Storage"; headerData.addProperty("from", fromStr); headerData.addProperty("to", toStr);

headerData.addProperty("access_token", GatewayInfo.getAccessToken());

SimpleDateFormat df = new SimpleDateFormat(MSG_TIMESTAMP_FORMAT); df.setTimeZone(TimeZone.getTimeZone("UTC")); String curTime = df.format(new Date(System.currentTimeMillis())); headerData.addProperty("timestamp", curTime); headerData.addProperty("eventTime", curTime);

JsonObject bodyData = new JsonObject(); bodyData.addProperty("storage", "10240"); bodyData.addProperty("usedPercent", "18");

JsonObject mqttMsg = new JsonObject(); mqttMsg.add("header", headerData); mqttMsg.add("body", bodyData);

DataTransService.mqttDataPub(cookie, "/cloud/signaltrans/v2/categories/data", 1, mqttMsg.toString().getBytes());
}

命令接收

当开发者希望设备或网关只接收topic为"/gws/deviceid/signaltrans/v2/categories/"的消息,且对消息中的header进行解析时,可以调用"设备命令接收"接口。

应用服务器可以调用物联网平台的应用侧API接口给设备或网关下发命令,所以网关得随时监听命令下发的广播,以便接收到命令时进行相应业务处理。

注册观察者对命令接收进行相应处理。

//注册观察者 AgentLiteDataTrans agentLiteDataTrans = AgentLiteDataTrans.getInstance(); DataTransService dataTransService = DataTransService.getInstance(); dataTransService.registerObserver(agentLiteDataTrans); //命令接收 agentLiteDataTrans.getCmdReceive();

被动接收命令的方法getCmdReceive:

```
private void getCmdReceive(lotaMessage iotaMsg) {
  System.out.println("=====receive iotCMD ========");
  String deviceId = iotaMsg.getString(DataTransService.DATATRANS_IE_DEVICEID);
  String requestId = iotaMsg.getString(DataTransService.DATATRANS_IE_REQUSTID);
  String serviceId = iotaMsg.getString(DataTransService.DATATRANS_IE_SERVICEID);
  String method = iotaMsg.getString(DataTransService.DATATRANS_IE_METHOD);
  String cmd = iotaMsg.getString(DataTransService.DATATRANS_IE_CMDCONTENT);
  if (method.equals("REMOVE_GATEWAY")) {
    //rmvGateway(context);
  }
  System.out.println ("Receive cmd :"
            + "ndeviceId = " + deviceId
            + "nrequestId = " + requestId
            + "nserviceId = " + serviceId
            + "nmethod = " + method
                       = " + cmd);
            + "ncmd
```

添加非直连设备

在添加非直连设备前,确认非直连设备的profile已经上传了,详见**上传Profile并注册** 设备步骤。

修改非直连设备信息,包括"nodeld"(设备标识码)、"manufactureld"(厂商 ld)、"deviceType"(设备类型)、"model"(设备模型)和"protocolType" (协议类型)。这里非直连设备的设备固有信息是测试数据,真实情况下,网关往往 需要跟具体的非直连设备交互,才能得到具体的设备固有信息。

```
public void addSensor() {
    System.out.println(" ======== addSensor! ========== ");
```

```
int cookie;
Random random = new Random();
cookie = random.nextInt(65535);
String nodeld = "5432154321";
String manufatrueld = "Huawei";
String deviceType = "Motion";
String model = "test01";
String protocolType = "MQTT";
deviceInfo = new lotaDeviceInfo(nodeld, manufatrueld, deviceType, model, protocolType);
....
}
```

注册观察者对添加设备结果进行相应处理。

```
//注册观察者
AgentLiteHub agentLiteHub = AgentLiteHub.getInstance();
HubService hubService = HubService.getInstance();
hubService.registerObserver(agentLiteHub);
//sensor添加
agentLiteHub.addSensor();
```

在设备或网关登录成功后就可以调用HubService.addDevice(int cookie, lotaDeviceInfo deviceInfo)接口添加非直连设备。添加非直连设备成功后就能从广播 中得到非直连设备的"deviceId"。

```
public void addSensor() {
    System.out.println(" ======= addSensor! ======= ");
    int cookie;
    Random random = new Random();
    cookie = random.nextInt(65535);
    ...
    HubService.addDevice(cookie, deviceInfo);
}
```

非直连设备添加成功后在"设备列表"中看到新增一条记录。

▲ 项目概览	设备列	表										
● 产品 产品开发		设备类型				设备状态 💿			虚拟&真实			
设备管理 升级调试			Top 3		50%		• 在线		虚拟设备		其实设备	
12 成用			• Gateway				・ ・ 第版	0% 50%	0		2	
		设备列表						Q 消除入设计	络称、设备ID	- +	新增真实设备	+新増虚拟设备
		状态	设备名称	设备ID			所属产品	型号	设备类型			
		●商线		79//Dc62-fe26	6-4ab7-9d96-4578t	of1eaf53	test001	test001	Motion	调试产品 弯	试应用 删除	
		●在线	aaa123456	9292e7e9-a7	/da-416b-9bc7-574c	d8d02d1b9	AgentLite001	AgentLite001	Gateway	调试产品 调	试应用 删除	

非直连设备状态更新

注册观察者对非直连设备状态更新结果进行相应处理。

```
//注册观察者
AgentLiteHub agentLiteHub = AgentLiteHub.getInstance();
HubService hubService = HubService.getInstance();
hubService.registerObserver(agentLiteHub);
//sensor状态更新
agentLiteHub.updataDeviceStatus();
```

非直连设备添加上时,一般情况下是"离线"状态。所以在非直连设备添加成功后, 或者在非直连设备上报数据前,要调用HubService.updateDeviceStatus(int cookie, String deviceId, String status, String statusDetail)进行设备状态更新。

非直连设备数据上报

请参考<mark>数据上报和数据发布</mark>章节,调用DataTransService.dataReport或 DataTransService.mqttDataPub接口进行数据上报,各个参数使用非直连设备的相 关数据即可,此处不再复述。

设备数据上报成功后,可以在非直连设备的"历史数据"中查看上报的数据。

▲ 项目概览	_	我的设备 >			
● 产品	•	设备详情 历史数据	设备日志 历史命令		
产品开发				08	BI NT
设备管理		服务	2013 67年16月	83(R)	
升级调试		Battery	("batteryLevel": 98)	2019/05/28 14:22:10	
≌ 应用	*	Motion	("motion": "DETECTED")	2019/05/28 14:22:10	

4.6.1.4.3 Agent Lite SDK 使用指南(Android)(联通用户专用)

非联通用户请查看设备接入服务。

按照本文档的指导,开发者可以体验直连设备通过集成Agent Lite快速接入平台,体验 "数据上报"、"命令接收"、"添加非直连设备"等功能。

Agent Lite以SDK的形式嵌入第三方软件中。本文档以Agent Lite Android Demo为例,指导开发者使用Agent Lite SDK中的接口,实现"直连设备登录"、"数据上报"和"命令下发"等功能。

- 开发者可以基于Agent Lite Android Demo开发,也可参考Agent Lite Android Demo,自行集成Agent Lite SDK(Android)。
- Agent Lite Android Demo使用的IDE工具为Android Studio。

使用必读

开发环境要求:

Android系统: API_LEVEL21及以上。

工程目录结构及文件说明:

目录结构	目录	说明
Agent LiteDemo	src	存放Agent Lite Demo代 码
├──src │ └──com │ └──huawei	libs	存放Agent Lite提供的jar 包和第三方jar包

目录结构	目录	说明
│ └──agentlitedemo ├──gen	armeabi	存放Agent Lite编译后的 库文件和第三方库文件
│	conf	存放TLS证书文件、配置 文件
├──bin ├──libs		
└──armeabi		
∣ └──res		

🗀 说明

如果开发者没有设备,可以直接在X86 Linux系统进行开发。

导入样例代码及配置 AndroidStudio

导入样例代码:

步骤1 将Agent Lite Demo(Android) 解压到本地。

(D:) ► MyWork ► OceanConnect ► Demo	•
新建文件夹	
名称 ^	修改日期
퉬 AgentLiteDemo	2017/7/10 14:39
🔚 AgentLiteDemo.zip	2017/10/18 19:38

SDK在Demo中libs文件中。



步骤2 导入android工程。

注:请确保本地计算机网络正常,才能下载需要的SDK和gradle包等资源。

- 打开Android Studio,在"file"菜单中选择"open"。
- 在打开的界面中输入AgentLiteDemo所在的路径,选择"AgentLiteDemo",点击"OK"。

😥 Open File or Project
🗌 📼 👁 🕞 📴 🗙 💋 💷 Hide path
D:\MyWork\OceanConnect\Demo\AgentLiteDemo
▶ L Git_Code
▶ □ Log
MyDocument
MyWork
AgentLiteDemo
 Gradle
▶ 🗖 .idea
▶ (• app
build
▶ 🗖 gradle
.gitignore
Drag and drop a file into the space above to quickly locate it in the tree
OK Cancel Help

• 打开菜单"Tools > Android > SDK Manager",在弹出的对话框中选择 "Android 7.1.1(Nougat)"。

<u>Tools</u> VC <u>S</u> <u>W</u> ind	dow <u>H</u> elp				
Tasks & Contex	xts	🕑 😲 🛫		1 ?	
Generate Java[Doc		1 - 1		
-					
New Scratch Fil	e Ctrl+Alt+Shift+Inse	rt			
IDE Scripting C	onsole				
5 Firebase					
App Links Assis	stant				
Android		🚺 🧿 Sync	Project	with Gradle F	iles
		Andr	oid Devi	ce Monitor	
		And A	old Devi	ce wontor	
		🔛 AVD	Manage	r	
		🖕 SDK	Manage	r	
		V Enab	le ADB I	ntegration	
		G Lavo	ut Inspe	tor	
		Then	an Editor		
			ne Eultor		
		Q Fireb	ase App	Indexing Te	st
Default Settings					
Q)	Appearance & Behavior > System Settings > Androi				F
Appearance & Behavior	Manager for the Android SDK and Tools used by Andr	oid Studio			
Appearance	Android SDK Location: D:\ProgramFiles\Android\sdk	2	E	<u>dit</u>	
Menus and Toolbars	SDK Platforms SDK Tools SDK Update Sites				
 System Settings Passwords 	Each Android SDK Platform package includes the And level by default. Once installed, Android Studio will an	droid platform and sourd itomatically check for up	ces pertaining to a dates. Check "sho	an API ow	
HTTP Proxy	package details" to display individual SDK componer	nts.	De datas	Chatar	
Updates	Android API 27	27	1	Not installed	
Usage Statistics	Android API 26	26	2	Not installed	
Android SDK	Android 7.1.1 (Nougat)	25	3	Installed	
Notifications	Android 6.0 (Marshmallow)	23	3	Not installed	
Quick Lists	Android 5.1 (Lollipop)	22	2	Not installed	
	Android 5.0 (Lollipop)	21	2	Not installed	
Path variables	Android 4.4 (KitKat Wear)	20	4	Not installed	
Keymap	Android 4.3 (Jelly Bean)	18	3	Not installed	
Editor	Android 4.2 (Jelly Bean)	17	3	Not installed	
Plugins	Android 4.1 (Jelly Bean)	16	5	Not installed	
Build, Execution, Deployment	Android 4.0.3 (IceCreamSandwich)	15	4	Not installed	
▶ Tools	Android 3.2 (Honeycomb)	13	1	Not installed	
	Android 3.1 (Honeycomb)	12	3	Not installed	
	Android 3.0 (Honeycomb)	11	2	Not installed	
	Android 2.3.3 (Gingerbread)	9	2	Not installed	
	Android 2.2 (Froyo)	8	3	Not installed	
	Android 2.1 (Eclair)	7	3	Not installed	
					-
				Show Packag	ge Deti
				Show Packag	ge Deti

步骤3 等待SDK下载安装完毕,重新编译工程,选择菜单"Build > Rebuild Project"即可。 如果编译过程出现问题,需要修改AndroidStudio配置,请参考下一章节的内容。


----结束

配置AndroidStudio:

推荐使用AndroidStudio 2.X.X版本,如2.3.1。推荐理由:版本相对稳定,下载插件时 不会出现后续版本的各种问题,各版本在功能方面实际上差不多。

步骤1 设置Gradle和Plugin版本。

Gradle和Plugin的版本之间有配套关系,请参考: https://developer.android.com/ studio/releases/gradle-plugin。

设置Gradle和Plugin版本时,需要注意在https://services.gradle.org/distributions/ 和http://jcenter.bintray.com/com/android/tools/build/gradle/目录下是否有所设 置的版本(可在浏览器中打开两个网址查看)。

• 打开\gradle\wrapper目录下的gradle-wrapper.properties,设置Gradle版本。



• 打开工程目录下的build.gradle,设置Plugin版本。



步骤2 设置Android SDK版本。

• 可以打开app目录下的build.gradle根据需要修改SDK版本号。



SDK版本号修改后需要注意是否已安装对应版本的Android SDK,可以在 "File > Settings > Android SDK > SDK Platforms"中查看并下载。

😨 Settings					×				
(g.sdk)	8	Appearance & Behavior > System Settings > And	roid SDK						
Appearance & Behavior		Manager for the Android SDK and Tools used by A	ndroid Studio						
System Settings		Android SDK Location: D:\P			Edit				
Android SDK		SDK Platforms SDK Tools SDK Update Sites							
Keymap		Each Android SDK Platform package includes the	Android platform ar	nd sources pertaini	ng to an API				
▼ Editor		level by default. Once installed, Android Studio wi	Il automatically chec	k for updates. Chee	:k "show				
Inspections	6	package details" to display individual SUK components.							
		Name	API Le	evel Revis	sion Status				
		Android API 28	28	6	Not installed				
		Android API 27	27	3	Installed				
		Android API 26	26	2	Installed				
		Android 7.1.1 (Nougat)	25	3	Installed				
		Android 7.0 (Nougat)	Android 7.0 (Nougat) 24 2 Not installed						
		Android 6.0 (Marshmallow)	23	3	Update available				
		Android 5.1 (Lollipop)	22	2	Partially installed				

步骤3代理配置。

打开 "File > Settings > HTTP Proxy"设置好相关配置即可(如下图所示)。如果不需要使用代理,请选择No proxy。

Oproxy Appearance & Behavior > System Settings > HTTP Proxy Appearance & Behavior O No proxy
Appearance & Behavior No proxy
System Settings HTTP Proxy Inspections Version Control Subversion Manual proxy configuration URL: Clear passwords Manual proxy configuration HTTP O SOCKS Host name: prc

• 打开工程目录下的gradle.properties,设置工程的http和https代理配置(一般两 个配置是一样的)。

E	AgentLiteDemo 🛛 📊 gradle.properties 🖉				
tg	🗊 Project 🔹 😌 🌞 👫	🔒 gradl	e.properties ×	🕼 gradle-wrapper.properties × 📀 Agentl	lite
[oi	🔻 🗔 AgentLiteDemo D:\MyWork\OceanConnect\tr	5	#		
a	▶ 🗖 .gradle	6	# Specifies the	. JVM arguments used for the daemon process.	
2	idea	7	# The setting i	s particularly useful for tweaking memory set	tin
	▶ ■ app	8	# Default value	: -Xmx1024m -XX:MaxPermSize=256m	
1 ŧ	N D build	9	# org.gradle.jv	margs=-Xmx2048m -XX:MaxPermSize=512m -XX:+Hes	ıpDw
tr		10	#		
1 S		11	# When configur	ed, Gradle will run in incubating parallel mo	ode.
7	i .gitignore	12	# This option s	should only be used with decoupled projects. M	lore
	La AgentLiteDemo.iml	13	# http://www.gr	adle.org/docs/current/userguide/multi_project	_bu
I les	🕑 build.gradle	14	# org.gradle.ps	urallel=true	
aptr	gradle.properties	15	#Thu Oct 19 11:	35:00 CST 2017	
	gradlew 3	16	systemProp. ht	tp. proxyPassword=13	
0	gradlew.bat	17	systemProp. ht	tp.proxyHost=r:m	
	import-summary.txt	18	systemProp. ht	tp.proxyPort=CCC0	
		19	systemProp. ht	tp.proxyUser=3	
	Continent and a setting a	20			
	setungs.gradie	21	systemProp. ht	tps.proxyPassword=P23	
	External Libraries	22	systemProp. ht	tps.proxyHost=pm	
	1	23	systemProp. ht	tps.proxyPort=CLC0	
	1	24	systemProp. ht	tps.proxyUser=w3	



初始化

在发起业务前,需要先初始化Agent Lite相关资源,调用API接口**BaseService.init()**, 初始化Agent Lite资源,具体API的参数使用参考Agent Lite API接口文档。可参考 "AgentLiteMain.java"中**onCreate()**方法对**BaseService.init()**的调用。

BaseService.init(WORK_PATH, LOG_PATH, context);

- "WORK_PATH"为工作路径,不能为空。
- "LOG_PATH"为打印日志路径,当"LOG_PATH"为空时,打印路径默认为工作路径,开发者也可以自己定义打印日志路径。

绑定和登录



设备或网关第一次接入loT联接管理平台时需要进行绑定操作,从而将设备或网关与平 台进行关联。开发者通过传入设备序列号以及设备信息,将设备或网关绑定到物联网 平台。

设备或网关绑定成功后或重启后,需要进行登录的流程,在设备或网关成功登录物联 网平台后,才可以进行其它服务操作,比如接入其他传感器,数据上报等等。如果设 备或网关登录成功,那么设备或网关在平台的状态显示为已在线。

步骤1 修改绑定参数。

绑定时使用的设备固有信息(如设备型号等)是从"config.properties"文件中读取 的,所以需要修改./app/src/main/assets/conf目录下config.properties文件中的如下信 息:

- "platformIP":物联网平台的设备对接地址(MQTTS),可参考**平台对接信息** 获取。
- "verifyCode"(设备的标识)和必要的设备信息,包括"Manufacture"(厂商 Id)、"deviceType"(设备类型)、"HardwareModel"(设备模型)和 "protocolType"(协议类型),其中"manufacturerId"(厂商Id)、 "deviceType"(设备类型)、"HardwareModel"(设备模型)和 "protocolType"(协议类型)与Profile文件中的定义保持一致。

🛄 说明

- 如果通过"设备管理服务控制台"注册设备,则"verifyCode"填写为设备注册时的 "preSecret"(预置密钥)。
- 如果通过开发中心注册设备,则"verifyCode"填写为设备注册后返回的"nodeld" (设备标识)。

"config.properties" 文件中设备固有示例:

platformIP=100.100.100.100 mqttPort=8883 httpPort=8943 verifyCode=0123456789 Manufacturer=Huawei HardwareModel=AgentLite01 deviceType=Gateway protocolType=HuaweiM2M

步骤2 绑定设备。

调用API接口BindConfig.setConfig()设置绑定配置。

private void startBind() {

```
....
configBindPara();
registerBindReceiver();
```

ı

```
private void configBindPara() {
```

BindConfig.setConfig(BindConfig.BIND_CONFIG_ADDR, AgentLiteUtil.get(ConfigName.platformIP)); **BindConfig.setConfig**(BindConfig.BIND_CONFIG_PORT, AgentLiteUtil.get(ConfigName.httpPort));

```
}
```

...

注册广播接收器对设备绑定结果进行相应处理。

LocalBroadcastManager.getInstance(this).registerReceiver(mBindStatusReceiver, new IntentFilter(**BindService.TOPIC_BINDDEVICE_RSP**));

调用API接口**BindService**.bind(String verifyCode, lotaDeviceInfo deviceInfo)绑定 设备,主要入参为"verifyCode"(设备验证码)和必要的设备信息,包括

"nodeld"(设备标识码)、"Manufacture"(厂商Id)、"deviceType"(设备类型)、"HardwareModel"(设备模型)和"protocolType"(协议类型),其中 "verifyCode"的值与"nodeld"保持一致。

private void startBind() {

```
LogUtil.i(this, TAG, "startBind");

String nodeld = AgentLiteUtil.get(ConfigName.verifyCode);

String verifyCode = AgentLiteUtil.get(ConfigName.verifyCode);

String manufacturerId = AgentLiteUtil.get(ConfigName.Manufacturer);

String deviceType = AgentLiteUtil.get(ConfigName.deviceType);

String model = AgentLiteUtil.get(ConfigName.HardwareModel);

String protocolType = AgentLiteUtil.get(ConfigName.protocolType);
```

lotaDeviceInfo deviceInfo = new lotaDeviceInfo(nodeld, manufacturerId, deviceType, model, protocolType);

BindService.bind(verifyCode, deviceInfo);

设备或网关绑定成功,后续就不需要再绑定了,除非设备或网关被删除,才需要重新 绑定。

设备绑定成功会收到广播,广播内容请参考Agent Lite API接口文档中设备绑定接口的返回结果说明和demo中**mBindStatusReceiver**函数的处理。

步骤3 配置登录参数。

登录前需要通过参数配置接口LoginConfig.setConfig()传入所需的登录信息。

- "设备Id"(即网关Id, "LOGIN_CONFIG_DEVICEID"), "appId"
 ("LOGIN_CONFIG_APPID")和"密码"("LOGIN_CONFIG_SECRET"),
 这些信息是都是从网关绑定成功的广播中得到的。
- "平台HTTP地址"("LOGIN_CONFIG_IOCM_ADDR")和"MQTT地址"
 ("LOGIN_CONFIG_MQTT_ADDR")一般是同一个地址,可以从绑定成功的广播中得到。一般情况下,这个地址和Agent Lite设备或网关对接的平台地址一致。
- 绑定成功的广播参数获取可以参考mBindStatusReceiver函数的处理。

private void configLoginPara() {

LoginConfig.setConfig(LoginConfig.LOGIN_CONFIG_DEVICEID, GatewayInfo.getDeviceID()); LoginConfig.setConfig(LoginConfig.LOGIN_CONFIG_APPID, GatewayInfo.getAppID()); LoginConfig.setConfig(LoginConfig.LOGIN_CONFIG_SECRET, GatewayInfo.getSecret()); LoginConfig.setConfig(LoginConfig.LOGIN_CONFIG_IOCM_ADDR, GatewayInfo.getHaAddress()); LoginConfig.setConfig(LoginConfig.LOGIN_CONFIG_IOCM_PORT, "8943"); LoginConfig.setConfig(LoginConfig.LOGIN_CONFIG_MQTT_ADDR, GatewayInfo.getHaAddress()); LoginConfig.setConfig(LoginConfig.LOGIN_CONFIG_MQTT_ADDR, GatewayInfo.getHaAddress());

步骤4 设备登录。

注册广播接收器对设备登录结果进行相应处理。

LocalBroadcastManager.getInstance(this).registerReceiver(LoginConnectReceiver, new IntentFilter(LoginService.TOPIC_LOGIN_CONNECTED));

调用**LoginService.login()**进行直连设备登录,具体API的参数使用参考Agent Lite接 口文档中设备登录接口的说明。

private void startLogin() {
 LogUtiLi(this, TAG, "startLogin");
 configLoginPara();
 LoginService.login();
}

----结束

编译安装程序

绑定和登录功能完成后,可以先测试一下网关是否能正常与平台对接,再进行后续功 能开发。

步骤1 将Android设备连到计算机上,点击" 🕨"按钮编译安装工程到Android设备上。

<u>B</u> uild R <u>u</u> n <u>T</u> ools VC <u>S</u> <u>W</u> indow <u>H</u> elp						
🔨 🕞 app 💌 🕨 🕴 🕼 🕼 💷 🖳 🧕 💱 🍄 🏜 🖙 📁 🔚 🚣 ?						
va 🔁 com 🔁 huawei 👌 🗈 agentlitedemo 👌 🗈 activity 🛇 G AgentLiteLogin 🔪						
C AgentLiteBind.java × C AgentLiteLogin.java ×						
🕖 Select Deployment Target						
4 Connected Devices						
5 HUAWEI HUAWEI GRA-CL00 (Android 5.0.1, API 21)						
5						
5						
6						
Create New Virtual Device <u>Don't see your device?</u>						
G Use same selection for future launches OK Cancel						
67 LogUtil. 2 (this, TAG, "startLogin");						

- **步骤2** 在Android设备上运行AgentLiteDemo,修改"平台IP"、"MQTT端口"、"HTTP 端口"和"verifyCode"等值。
 - "平台IP"、"MQTT端口"和"HTTP端口"可以从物联网平台环境开发中心的 "对接信息"界面获取。
 - "verifyCode"则是设备的标识,每个设备对应一个verifyCode,不可重复,所以 建议使用IMEI或者MAC地址等天然的设备标识。AgentLiteDemo可以在界面上人 工输入设备的verifyCode,测试时只要输入一个没有使用过的verifyCode即可。

AgentLiteDemo	BIND
Basic platformIP	
mqttPort	
httpPort	
verifyCode	



上传 Profile 并注册设备

下载**Profile开发示例**,并上传模板中的profile文件: "Gateway_Huawei_AgentLite01.zip"和"Motion_Huawei_test01.zip"。

步骤1 登录开发中心,创建一个项目,在该项目空间内,选择"产品 > 产品开发",点击 "新建产品"。

▲ 项目概览	011 开发中	02 自助测试中	03 认证发布			
● 产品						
产品开发	• 开版中	• 3820P		 已没布 	• E&B	
设备管理	0	U	0	0	0	
升级调试						
1 应用						
对接信息						
订阅调试						
应用调试						
应用在线开发						
		提供加至100 ²² 员	开发肉导和相关开发资源,导致开发者经经完成产品开发			
			86287455			

步骤2 在"创建产品"中,选择"本地导入产品创建",单击"上传Profile"上传 "Gateway_Huawei_AgentLite01.zip"和"Motion_Huawei_test01.zip"。

产品开发 > 创建产品					
			请选择以下	一种方式创建您的	的产品
基于系统模板创建	基于快速集成模板创建	基于已有产品创建	本地导入产品创建	自定义产品	
		您可	以将本地写好的Pro	file文件上传到平台	7,开发一个新产品
			[上传Profile	

步骤3选择"产品 > 设备管理",单击"添加真实设备",进入"新建增实设备"页面。

▲ 项目概览	设备类型	设备状态 ③	虚拟&真实
● 产品 -			
产品开发			
设备管理	暂无 设备	Cesterate asserted	出現で動 真宗保険 0 0
升级调试			
₽ 应用 👻			
		平台为通过接入真实 & 虚拟的设备,就可以对您开发的应用和设备进行在线谍测,并 进行统一的设备管理	
		183.10.18.10.18. 183.10.18.10.18.	

步骤4根据向导注册设备。

1. 选择产品。 产品: AgentLite001

真实设备		
01 选择一款产品		(02) 注册设备
0	0	
AgentLite001 Huawel	test001 Huawei	

- 2. 填写设备相关信息,单击"确定"。
 - 设备名称: AgentLiteDevice
 - 设备标识: aaa123456,需要与AgentLiteDemo中网关的设备标识一致。
 - 接入方式:网关(Agentlite)

新建真实设备	
	02
12月94——2017 ¹⁹ 88	注册公爵
*设备名称 AgentLi	teDevice
ेव्हामालप्पर aaa1234	56
接入方式	● 圆学(Anentitie)
	C 4244 Brunek
上一步	确定

----结束

设备上线

步骤1 点击AgentLiteDemo上的"BIND"按钮,查看界面上打印出的"设备日志"。

AgentLiteDemo	BIND
Basic	
platformIP	

步骤2 当出现"AgentLiteLogin Login Success",表示AgentLite登录平台成功。

Login	:
AgentLiteLogin onCreate AgentLiteLogin startLogin AgentLiteLogin Login Success!!!	

----结束

数据上报和数据发布

- 设备或网关向物联网平台上报数据可以通过调用SDK的"设备服务数据上报"接口或 "数据发布"接口:
- "设备服务数据上报"接口:deviceId,requstId和serviceId由SDK组装为消息的 header;serviceProperties由SDK组装为消息的body。消息组装格式为JSON。
 设备或网关登录成功后可以调用DataTransService.dataReport(int cookie, String requstId, String deviceId, String serviceId, String serviceProperties) 接口上报数据。
 - 当设备主动上报数据时," requstId "可以为空 。
 - 当上报的数据为某个命令的响应时,"requstld"必须与下发命令中的 "requstld"保持一致。requestld可以从广播中获取,请参考API文档中"设 备命令接收"接口的广播参数"DATATRANS_IE_REQUESTID"的说明。
 - "serviceId"要与profile中定义的某个serviceId保持一致,否则无法上报数据。
 - "serviceProperties"实际上是一个json字符串,内容是健值对(可以有多组 健值对)。每个健是profile中定义的属性名(propertyName),值就是具体 要上报的内容了。

private void gatewayDataReport() {
 LogUtiLi(this, TAG, "gatewayDataReport!");
 int cookie;
 String deviceId = GatewayInfo.getDeviceID();

```
Random random = new Random();
cookie = random.nextInt(65535);
LogUtil.i(this, TAG, "cookie = " + cookie);
DataTransService.dataReport(cookie, null, deviceId, "Storage", "{\"storage\":10240,\"usedPercent
\":20}");
}
```

注册广播接收器对网关数据上报结果进行相应处理。

LocalBroadcastManager.getInstance(this).registerReceiver(dataReportRsp, new IntentFilter(DataTransService.TOPIC_DATA_REPORT_RSP));

"数据发布"接口:topic固定为"/cloud/signaltrans/v2/categories/data";
 "serviceData"参数作为消息体(包括header和body),SDK只进行透传,不进行格式调整和组装。

设备或网关登录成功后可以调用DataTransService. mqttDataPub(int cookie, String topic, int qos, byte[] serviceData)接口发布数据。

- "Topic"是要发布数据的topic。
- "Qos"是mqtt协议的一个参数。
- "serviceData"实际上是一个json字符串,内容是健值对(可以有多组健值 对)。每个健是profile中定义的属性名(propertyName),值就是具体要上 报的内容了。

private void gatewayDataPub(int cookie, String topic, int qos, byte[] serviceData) { DataTransService. mqttDataPub(cookie, topic, qos, serviceData); }

注册广播接收器对网关数据上报结果进行相应处理。

LocalBroadcastManager.getInstance(this).registerReceiver(mqttDataPubRsp, new IntentFilter(DataTransService.TOPIC_MQTT_PUB_RSP));

命令接收

当开发者希望设备或网关只接收topic为"/gws/deviceid/signaltrans/v2/categories/"的消息,且对消息中的header进行解析时,可以调用"设备命令接收"接口。

应用服务器可以调用物联网平台的应用侧API接口给设备或网关下发命令,所以设备或 网关需要随时检测命令下发的广播,以便在接收到命令时进行相应业务处理。

注册DataTransService.TOPIC_COMMAND_RECEIVE广播接收器对命令下发进行相应处理。

LocalBroadcastManager.getInstance(this).registerReceiver(commandReceiver, new IntentFilter(DataTransService.TOPIC_COMMAND_RECEIVE));

具体的处理函数commandReceiver:

```
private BroadcastReceiver commandReceiver = new BroadcastReceiver() {
    @Override
    public void onReceive(Context context, Intent intent) {
        lotaMessage iotaMsg = (lotaMessage)
    intent.getSerializableExtra(DataTransService.DATATRANS_BROADCAST_IE_IOTAMSG);
        String deviceId = iotaMsg.getString(DataTransService.DATATRANS_IE_DEVICEID);
        String requestId = iotaMsg.getString(DataTransService.DATATRANS_IE_REQUSTID);
        String serviceId = iotaMsg.getString(DataTransService.DATATRANS_IE_REQUSTID);
        String method = iotaMsg.getString(DataTransService.DATATRANS_IE_SERVICEID);
        String cmd = iotaMsg.getString(DataTransService.DATATRANS_IE_METHOD);
        String cmd = iotaMsg.getString(DataTransService.DATATRANS_IE_CMDCONTENT);
        if (method.equals("REMOVE") && deviceId.equals(sensorId)) {
            //rmvSensor();
        }
        }
    }
}
```

```
LogUtil.i(AgentLiteLogin.this, TAG, "Receive cmd :"
+ "\ndeviceId = " + deviceId + "\nrequestId = " + requestId
+ "\nserviceId = " + serviceId + "\nmethod = " + method
+ "\ncmd = " + cmd);
}
```

在开发中心的"产品 > 设备管理"界面中,单击非直连设备列后的"删除"按钮,这 样就能在demo界面上看到广播接收时的日志打印命令下发。

非直连设备的删除需要网关的确认,正常业务情况下,网关又需要跟具体的设备确 认,所以收到删除非直连设备的命令也不会将设备删除。

添加非直连设备

};

在添加非直连设备前,确认非直连设备的profile已经上传了,详见**上传Profile并注册** <mark>设备</mark>步骤 。

在设备或网关登录成功后就可以调用HubService.addDevice(int cookie, lotaDeviceInfo deviceInfo)接口添加非直连设备。

这里非直连设备的设备固有信息是测试数据。真实情况下,网关往往需要跟具体的非 直连设备交互,才能得到具体的设备固有信息。

private void addSensor() {



HubService.addDevice(cookie, deviceInfo);

注册广播接收器对添加设备结果进行相应处理。添加非直连设备成功后就能从广播中 得到非直连设备的"deviceld"。

LocalBroadcastManager.getInstance(this).registerReceiver(addDeviceReceiver, new IntentFilter(HubService.TOPIC_ADDDEV_RSP));



非直连设备添加成功后可以在"设备列表"中看到新增一条记录。

非直连设备状态更新

非直连设备添加上时,一般情况下是"离线"状态。所以在非直连设备添加成功后, 或者在非直连设备上报数据前,要调用HubService.updateDeviceStatus(int cookie, String deviceId, String status, String statusDetail)进行设备状态更新。

private void updateDeviceStatus(String status, String statusDetail) {
 LogUtil.i(this, TAG, "updateDeviceStatus!");
 int cookie;
 Random random = new Random();
 cookie = random.nextInt(65535);
 SharedPreferences preferences = getSharedPreferences("AgentLiteDemo", MODE_PRIVATE);

String deviceId = preferences.getString("SENSORID", null);

if (deviceId != null) {
 HubService.updateDeviceStatus(cookie, deviceId, status, statusDetail);
}

1

}

AgentLiteDemo中只添加了一个非直连设备,所以**updateDeviceStatus()**方法中使用的"deviceId"是直接从SharedPreferences中读取的。

注册广播接收器对非直连设备状态更新结果进行相应处理。

LocalBroadcastManager.getInstance(this).registerReceiver(devStatusUpdateReceiver, new IntentFilter(HubService.TOPIC_DEVSTATUS_RSP));

非直连设备数据上报

请参考<mark>数据上报和数据发布</mark>章节,调用DataTransService.dataReport或 DataTransService.mqttDataPub接口进行数据上报,各个参数使用非直连设备的相 关数据即可,此处不再复述。

设备数据上报成功后,可以在非直连设备的"历史数据"中查看上报的数据。

▲ 项目概览		我的设备 >			
● 产品	•	设备详情历史数据	设备日志 历史命令		
产品开发					ORM
设备管理		服务	数据评价	a)(i)	
升级调试		Battery	("batteryLevel": 98)	2019/05/28 14:22:10	
₩ 应用	.	Motion	{ "motion": "DETECTED" }	2019/05/28 14:22:10	

4.6.1.5 使用模组接入

集成NB-IoT模组的设备,可以通过NB-IoT网络接入物联网平台。



特点	 覆盖广,相比LTE提升20dB以上的增益 低功耗,聚焦小数据量、小速率应用 海量连接,单扇区支持5万个连接 低成本,低速率、低功耗、低带宽等特点使NB-IoT芯片或模组具 备低成本优势
应用场景	对数据时效性要求低,数据包较小,设备位置变化较小,需要电池供 电,例如:智能抄表,智能路灯等。
网络需求	 NB-IoT网络:由运营商构建。 NB-IoT SIM卡:向NB-IoT网络运营商购买。 NB-IoT模组:向模组厂商购买已认证模组。您可以访问华为云市场,购买已认证的模组。
物联网接入 协议	CoAP/LWM2M
相关资源	请从模组厂商获取更多信息和支撑。

AT 指令集

AT指令用于控制设备。如下AT命令仅供参考,具体命令集请向相应的模组厂商获取。

AT命令	作用	备注
AT+CMEE=1	报错查询。	标准AT指令
AT+CFUN=0	关机。设置IMEI和平台IP端口前要先关 机。	标准AT指令
AT+CGSN=1	查询IMEI。IMEI为设备标识,应用服务 器调用API接口注册设备时,nodeld/ verifyCode都需要设置为IMEI。	标准AT指令
AT +NTSETID=1,xxxx	xxxx为IMEI。如果查询不到可自行设置 IMEI,IMEI必须是唯一的,不能与其他 设备重复,且只能设置一次。 IMEI为设备标识,应用服务器调用API接 口注册设备时,如果设备使用海思芯 片,则nodeld/verifyCode都需要设置成 IMEI;如果设备使用高通芯片,则 nodeld/verifyCode都需要设置成 urn:imei:IMEI。	海思芯片私有AT指 令,在flash中保存 IMEI。应用服务器 在向平台进行设备 注册时,使用此参 数,其他芯片或模 组厂商可参考实 现。

AT命令	作用	备注
AT +NCDP="IP","port "	设置设备对接的物联网平台的IP地址和端 口号,5683为非加密端口,5684为DTLS 加密端口。	海思芯片私有AT指 令,在flash中保存 IP和端口。应用服 务器在向平台进行 设备注册时,使用 此参数,其他芯片 或模组厂商可参考 实现。
AT+CFUN=1	开机。	标准AT指令
AT+NBAND=频段	设置频段。	海思芯片私有AT指 令,在flash中保存 频段。设备在入网 时,使用此参数, 其他芯片或模组厂 商可参考实现。
AT +CGDCONT=1,"IP ","CTNB"	设置核心网APN。APN与设备的休眠、 保活等模式有关,需要与运营商确认。	标准AT指令
AT+CGATT=1	入网。	标准AT指令
AT+CGPADDR	获取终端IP地址。	标准AT指令
AT+NMGS=x,xxxx	发送上行数据。第1个参数为字节数,第 2个参数为上报的16进制业务码流。	海思芯片私有AT指 令,初次发送数据 时,完成设备注 册;后续发送数据 时,仅发送数据。 其他芯片或模组厂 商可参考实现。
AT+NQMGR	接收下行数据。	海思芯片私有AT指 令,查询接收 buffer中可以接收 的数据量,以及当 前总共接收的消息 数和丢弃的消息 数。其他芯片或模 组厂商可参考实 现。
AT+NMGR	读取数据。	海思芯片私有AT指 令,读取从物联网 平台(LWM2M SERVER)接收到的 数据。其他芯片或 模组厂商可参考实 现。

4.6.1.5.1 Agent Tiny SDK 使用指南(联通用户专用)

非联通用户请查看设备接入服务。

Agent Tiny是部署在具备广域网能力、对功耗/存储/计算资源有苛刻限制的终端设备上 的轻量级互联互通中间件,您只需调用API接口,便可实现设备快速接入到物联网平台 以及数据上报和命令接收等功能,相关集成指导请参见<mark>端云互通组件开发指南</mark>。

4.6.1.6 软固件升级调测

4.6.1.6.1 固件升级(联通用户专用)

非联通用户请查看设备接入服务。

固件(Firmware)一般是指设备硬件的底层"驱动程序",承担着一个系统最基础最底层工作的软件,比如计算机主板上的基本输入/输出系统BIOS(Basic Input/output System)。固件升级又称为FOTA(Firmware Over The Air),是指用户可以通过OTA的方式对支持LWM2M协议的设备进行固件升级,固件升级协议为LWM2M协议。

本章节将介绍基于开发中心的固件升级将如何进行调测。基于设备管理服务的固件升级可参考**设备管理服务固件升级**。

检查设备的固件升级能力

在执行固件升级操作前,请确认该设备支持固件升级能力。

- 步骤1 登录开发中心,并进入到对应的项目。
- 步骤2 选择"产品 > 产品开发",选择具体产品,进入该产品的开发空间。

▲ 项目概范	01 开发中	02 自助测	试中	👊 认证发布			
产品开发 设备管理 开级调试	• ^{开发中} 2	• #### 0		• #### 0	• Bite 0	• BRR 0	
н олн 🗸	全部产品(2)	MERT	한다 전문원들 WaterMeter WaterMeter	奴原他式 二进制间间	Q 初始入7 最后待约时间 2019/02/28 19:53:23	数+ (<u>世</u> 州和中国中国	tu O 刷新

步骤3 在"Profile定义 > OM维护"下,确认"固件升级"功能已开启。

OM维护			
软件升级	國件升级		
软件升级协议:	固件升级协议:		
PCP	LWM2M	•	提交

----结束

上传固件包

步骤1 选择"产品 > 升级调试 > 升级包管理 > 固件包管理",点击"上传未签名固件包"。

▲ 項目概況		升级调试 > 升级包管理								十上传末签名图件包
	•	因件包管理 软件包管理								
设备管理		图件名称	版本	设备处型	厂商名称	했음	协议	描述	上传时间	
升级调试						暂无数据				
11 应用	*									< 1 > BMB: 1 >

步骤2 根据固件包上传向导,完成固件包上传。

1. 选择产品,点击"下一步"。

因为固件包中只包含用于固件升级的bin文件,开发中心无法从固件包直接获取产 品模型信息,所以此处需要选择产品并与固件包进行关联。

古	件包上传	ŧ					×
	1 产品遗	· · · · ·				2 因件包上传	
选	择产品		产品名称	设备型号	设备类型	接入协议	
	۲	6	WaterMeter01	NBIoTDevice01	WaterMeter	LWM2M	
	0	\sim	twx2	twx02	BikeLock	CoAP	
			Bulb	BuLB001	Bulb	MQTT	
			DoorLock	DoorLock001	DoorLock	LWM2M	
				取消	下一步		

2. 选择需要上传的未签名固件包,填写固件包的版本号,点击"提交"。

固件包上传	×
02 产品选择 図件包上传	
上传固件包	
patch_656to656sp1.zip	
取消上一步提交	

----结束

创建固件升级任务

步骤1 选择"产品 > 升级调试 > 固件升级",点击"创建升级任务"。

▲ 项目概选	lini4字升级 软件升级		+ 部建升级任务 开级包管理
		~你还没有创建任何固件升级任务哦~	
产品开发		创建任务	
设备管理			
升级调试			

步骤2 根据固件升级向导,逐步完成升级任务的创建。

- 1. 填写任务基本信息,点击"下一步"。
 - 如果需要配置升级任务的执行类型和重试参数,"高级"选择"是"。
 - 执行类型:开发中心在什么时候给设备下发升级任务,取值范围:现在、设备在线时或自定义时间段。
 - 重试参数:执行任务失败后,是否重新执行任务。

目件升级			
01	(02)	03	04
任务信息	产品选择	固件包选择	设备选择
创建任务信息			
*任务名	task001		
高级	是		•
选择高级参数			
执行类型	现在		•
重试参数	自定义		•
*重试次数	5		*
*重试间隔(秒)	120		*
	取消	下一步	
译产品,点击" ^一	下一步"。		
牛升级			
 任务信息	2 产品选择	3 固件包选择	
译产品			

۲ **_G!** WaterMeter01 NBIoTDevice01 WaterMeter LWM2M twx02 CoAP twx2 BikeLock Bulb BuLB001 Bulb MQTT DoorLock DoorLock001 DoorLock LWM2M 1 取消 上一步

2.

选择需要升级的固件包,点击"下一步"。
 如果"历史固件包"列表中没有需要的固件包,可以在此界面上传新的未签名固件包,请确保上传的固件包是用于已选产品的固件升级。

● ●	● 3 4 ● <th>国件升级</th> <th></th> <th></th> <th></th>	国件升级			
上传末签名固件包,大小200M以内,需为zip格式 ① 请输入固件版本 確定 历史固件包 @ patch_656to656sp1.bin 1.0 WaterMeter LWM2M	上传未签名固件包,大小200M以内,需为zip格式 ① 请输入固件版本 历史固件包	✓ 任务信息 #26月44句	/ 产品选择	3 固件包选择	(4) 设备选择
 历史固件包 patch_656to656sp1.bin 1.0 WaterMeter LWM2M 	● patch_656to656sp1.bin 1.0 WaterMeter LWM2M	上传末签名固件包,方	大小200M以内,需为zip格式	▲ 请输入固件版本	确定
		• patch_656	历史 o656sp1.bin 1.0	固件包 WaterMeter	LWM2M

4. 在"设备选择"界面,会呈现已选产品对应的所有设备,可以选择一个或多个设备进行升级,点击"提交"。

✓ 任务信息	() 产品选择	✓ ● 通选择	4 设备选择
译设备			
- 状态	设备名称	设备型号	设备类型
ONLINE	watermeter	NBIoTDevice01	WaterMeter
ONLINE	NBIoTDevice01NB Simulator	NBIoTDevice01	WaterMeter
ONLINE	111111111	NBIoTDevice01	WaterMeter

步骤3 固件升级任务创建后,可以在"固件升级"界面对固件升级任务进行管理。点击具体 任务可以查看任务的"基本信息"和"升级详情"。

国件升级 软件升级				十 创建升级任务 开级包管
升级状态	任务名称	开始时间	成功率	操作
● 运行中	task01	2019-03-01 11:21:00	0.00%	停止 暴出 幽陰
				< 1 > 創時到: 1 >

🗋 说明

设备在升级期间,设备不可进行业务交互。

步骤4 任务结束后,点击"导出",导出任务详情文件。

固件升级 软件升级				+ 创建升级任务 升级包管理
升级状态	任务名称	开始时间	成功率	進作
● 成功	task01	2019-03-01 15:35:50	100.00%	豆油 艶絵
				< 1 > 3U4931: 1 >

----结束

4.6.1.6.2 软件升级(联通用户专用)

非联通用户请查看设备接入服务。

软件(Software)一般分为系统软件和应用软件,系统软件实现设备最基本的功能, 比如编译工具、系统文件管理等;应用软件可以根据设备的特点,提供不同的功能, 比如采集数据、数据分析处理等。软件升级又称为SOTA(SoftWare Over The Air), 是指用户可以通过OTA的方式支持对LWM2M协议的设备进行软件升级,软件升级协议 为PCP协议。 本章节将介绍基于开发中心的软件升级将如何进行调测。基于设备管理服务的软件升级可参考**设备管理服务软件升级**。

检查设备软件升级能力

在执行软件升级操作前,请确认该设备支持软件升级能力。

- 步骤1 登录开发中心,并进入到对应的项目。
- 步骤2 选择"产品 > 产品开发",选择具体产品,进入该产品的开发空间。



步骤3 在 "Profile 定义 > OM 维护 "下,确认 "软件升级"功能已开启。

OM维护		
软件升级	国件升级	
软件升级协议:	固件升级协议:	
PCP	► LWM2M	提交
	-	

----结束

制作软件升级版本包

设备升级的软件包文件由各设备厂商提供,在物联网平台上传设备的软件升级包前, 需要制作软件升级的版本包,用于修改软件包的描述文件,如软件版本、厂商名称、 设备类型、产品模型等信息。下面将详细介绍版本包的制作方法。

- 步骤1 新建文件夹命名为"DM",在DM文件夹下新建文件夹,命名为"linux"。
- **步骤2** 使用Notepad++文本工具新建一个文本文件,拷贝如下内容到文本中,在Notepad+ +工具的"编码"菜单中选择"使用UTF-8编码",然后将文本进行存储,存储路径选 择步骤1中的"linux"文件夹,文件名称命名为"UpgradeDesc",保存类型选择 ".json"。

```
{
    "specVersion": "",
    "fileName": "",
    "packageType": "",
    "version": "",
    "deviceType": "",
    "manufacturerName": "",
    "model": "",
    "protocolType":"",
    "description":"",
    "versionCheckCode":"",
    "deviceShard":"",
    "platform":"",
    "supportSourceVersionList":[],
    "date":""
}
```

DM	UpgradeDesc.jso	n 2019/8/29 15:38	JSON 文件	1 KB
📙 linux				
linux				

步骤3 打开创建的"UpgradeDesc.json"文件,修改软件升级描述文件,相关字段如下表所示。

字段名	字段描述	是否必填
specVersion	描述文件版本号,固定值:"1.0"。	是
fileName	软件包文件名,例如: "ExamplePackage_V1.0.10.xx"。	是
packageType	软件包类型,必须设置为: "softwarePackage"。	是
version	软件包版本号长度不超过16个字节,例如: "V1.0.10"	是
deviceType	设备类型,需要与产品模型保持一致。例如: "WaterMeter"。	是
manufacturerNam e	制造商名称,需要与产品模型保持一致。例如: "TestUtf8ManuName"。	是
model	产品型号,需要与产品模型保持一致。例如: "TestUtf8ModelM2M"。	是
protocolType	设备接入协议类型,需要与产品模型保持一致。 例如: "CoAP"。 说明 设备接入的协议类型有三种: "CoAP"、 "LWM2M"、"MQTT"。	是
description	对软件包的自定义描述。	是
versionCheckCode	软件升级包校验码,长度为4个字符。软件包下载 时支持断点续传,根据该字段标识前后两次下载 的软件包分片是否为同一个软件包。	否
deviceShard	终端下载软件包的每个分片的大小,单位为 byte,如果不设置默认为500byte。大小为 32~500之间。	否
platform	标识设备的操作系统,如linux。	否
supportSourceVers ionList	支持用于升级此版本包的设备源版本列表。 支持通配符配置,*代表匹配任意0~n个字符,?代 表匹配单个任意字符,如果存在多个版本,请使 用英文";"隔开。	否
date	出包时间,格式为:"yyyy-MM-dd"。	否

步骤4 在与"DM"同级目录下创建文件夹,命名为"linux",该文件夹名称必须同<mark>步骤1</mark>中的文件夹命令保持一致,将厂商软件包(软件包格式无限制)置于该文件中。

DM ExamplePackage_V1.0.10.tb 2017/2/6 18:20 TB 文件 154 KB linux

步骤5 选中"DM"和"linux"文件夹,使用压缩工具打包成ZIP格式的压缩包,建议命令为 "xx_package.zip"。

须知

- 文件 "DM" 和 "linux" 的命名是固定的。
- "xx_package.zip"下不能包含package这层目录。
- 仅支持ZIP格式的压缩包,不能压缩成其他格式后,例如rar,再手动修改文件类型为zip。

----结束

上传软件包

步骤1 选择"产品 > 升级调试 > 升级包管理 > 软件包管理",点击"上传未签名软件包"。

▲ 项目概況	升级调试 > 升级包管理				十上待未签名软件包
	周件包管理 软件包管理				
产品开发					
设备管理	软件名称	版本描述	版本	上他的問	
升级调试	V OFO_TEST_V1.1.10.tb	Test packageType made by WYH	OFO_TEST_V1.1.10	2019-03-01 11:06:53	不 農
112 应用 🗸					< 1 > NAMBI: 1 >

步骤2 在"软件包上传"界面,选择需要上传的未签名软件包,然后点击"提交"。

上传软件包前请确保开发中心已存在对应的产品模型。

🛄 说明

因为开发中心可以从软件包里的json文件获取产品模型信息,所以此处不需要像"固件包上传" 一样,选择产品关联软件包。

----结束

创建软件升级任务

步骤1 选择"产品 > 升级调试 > 软件升级",点击"创建升级任务"。

▲ 项目概范	固件升级	杭特升级	+ 创建升级任务 升级包管理
<u>◆</u> 产品		~ 你还没有创建任何软件升级任务哦 ~	
产品开发		他能任何	
设备管理			
升级期试			

步骤2 根据软件升级向导,逐步完成升级任务的创建。

- 填写任务基本信息,点击"下一步"。
 如果需要配置升级任务的执行类型和重试参数,"高级"选择"是"。
 - 执行类型:开发中心在什么时候给设备下发升级任务,取值范围:现在、设备在线时或自定义时间段。

1	(2)	3
任务信息	软件包选择	设备选择
创建任务信息		
*任务名	upgrade001	
高级	是	•
选择高级参数		
执行类型	设备在线时	•
重试参数	自定义	•
*重试次数	2	* *
*重试间隔(秒)	120	*

重试参数:执行任务失败后,是否重新执行任务。

 选择需要升级的软件包,点击"下一步"。
 如果"历史软件包"列表中没有需要的软件包,可以点击"上传未签名软件包" 上传新的未签名软件包。上传软件包前请确保开发中心已存在对应的产品模型。

软件升级				×
✓ - 任务信息		2 软件包选择		3 设备选择
选择软件包				十上传未签名软件包
		历史软件包		
۲	OFO_TEST_V1.1.10.tb	OFO_TEST_V1.1.10	BikeLock	CoAP
	取消	上一步	下一步	

3. 在"设备选择"界面,可以选择一个或多个设备进行升级,点击"提交"。

 任务信息 			R.	3 设备选
违择设备				
	状态	设备名称	设备型号	设备类型
~	OFFLINE	ofo01	twx02	BikeLock
	OFFLINE	ofo02	twx02	BikeLock

步骤3 软件升级任务创建后,可以在"软件升级"界面对软件升级任务进行管理。点击具体 任务可以查看任务的"基本信息"和"升级详情"。

71580-0029 012592-0190 01259	3061 F
◎ 运行中 upgrade001 2019-04-24 18:12:20 0.00%	
	使止 异出 删除

步骤4 任务结束后,点击"导出",导出任务详情文件。

固件升级	软件升级			+ 创建升级任务 升级包管理
升级状态	任务名称	开始的力间	成功率	编作
●成功	upgrade001	2019-04-26 10:21:30	100.00%	要進 趙徐

----结束

4.6.1.7 应用侧开发

4.6.1.7.1 使用 API 对接

4.6.1.7.1.1 API 使用指导(联通用户专用)

非联通用户请查看设备接入服务。

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Restful API(Application Programming Interface)。您可以调用开放的API,快速 集成物联网平台的功能,如设备管理、数据采集、命令下发和消息推送等功能。

接口介绍

API分组	应用场景
应用安全接 入	本部分接口提供"鉴权"以及"刷新token"两个接口。应用服务器 通过调用这两个接口获取鉴权令牌,是调用其他平台API的前提,其 他接口调用都需要在请求的Header中携带参数app_key和 Authorization。
	● app_key取值与请求参数中appId相同。
	• Authorization的格式为Authorization: Bearer {accessToken}。
	关于应用安全接入的接口一般有两种调用模式,一种是每次业务处理 前都调用一下鉴权接口重新获取token,然后用token来调用其他业务 接口;另外一种是应用服务器有一个有效管理机制,通过定期地刷新 token,保证在调用其他业务接口前token都是有效的。

API分组	应用场景
设备管理	本部分接口提供设备管理的相关功能。设备作为一种资源,设备管理 接口提供了注册设备,删除设备,修改设备信息、位置信息等。查询 设备信息的接口归类在"数据采集"中。
	平台的接入的设备有两种,直连设备和非直连设备。
	• 直连设备指已实现TCP/IP协议栈的设备,它可以直接与平台进行通信,常见的设备包括网关、以太网设备、NB-IoT设备等。
	 非直连设备指未实现TCP/IP协议栈的设备,只能基于一些简单的近场通信协议如ZigBee、ZWave、Bluetooth或者是其他的一些非IP 有线方式传输协议如串口、并口等接入,此时需要一个网关设备,先让设备接入到网关,再通过网关与华为物联网平台通信。
	设备管理的接口中提供了发现非直连设备、删除非直连设备等接口。 如果您的方案中不包含非直连设备,则不需要调用这两个接口。
数据采集	数据采集接口提供设备的查询、批量设备查询、设备历史数据和历史 命令的查询、设备能力(即profile中定义的设备能力)查询。
	当应用服务器向平台查询设备的历史数据和历史命令时,这些数据已 经存储在物联网平台上,因此与设备是否在线无关。
订阅管理	物联网平台允许应用服务器订阅其所关注的事件,每一种事件的每一次订阅都会生成一个subscriptionld,应用服务器可以使用 subscriptionld对本次订阅进行查询、更新、删除等操作。
	平台是通过Restful接口向应用服务器推送数据的,因此应用服务器需要开发一个Restful接口来接收推送数据,这个接口的URL也需要在订阅时提供给物联网平台。由于不同事件的推送消息结构不同,因此应用服务器可以考虑为每一种事件实现一个Restful接口。
	注意事项:
	● 关于平台上的事件种类和推送的消息样例,请参考 <mark>推送通知</mark> 。
	• https推送的前提是先在平台上上传应用服务器的CA证书。
命令下发	本部分接口提供命令的创建、删除、查询、批量创建等功能。
批量处理	目前仅支持创建批量下发缓存命令任务,您也可以查询批量任务信 息。
设备组管理	应用服务器可以通过本部分接口在物联网平台上管理设备组。设备组 是把设备进行分组管理。一个设备可以归属到多个设备组内。
	在对设备进行某些操作时(如升级设备软固件、批量下发命令等), 可通过设备组来指定要进行操作的设备。
设备升级	若需要对设备进行软固件版本升级,应用服务器可调用本部分接口为 多个设备创建升级任务,升级前请确保目标版本包已经上传到物联网 平台。当前仅支持对CoAP接入的设备进行软固件版本升级。

4.6.1.7.1.2 使用 Postman 调测(联通用户专用)

非联通用户请查看设备接入服务。

调用API接口前建议完成Profile文件和编解码插件的开发。尽管接口的调用不依赖于 Profile文件和编解码插件,但是由于profile文件定义了设备数据的字段,编解码插件 是上报数据和下发命令时的必要条件,因此为了使得调用接口前的业务都正常,需要 先行完成产品的开发。

为充分了解接口,建议提前获取《应用侧API参考》查阅。我们已经写好了Postman的 Collection,在Collection中接口的请求结构体已经完成可以直接使用。

本文档以Postman为例,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:

- "鉴权"接口
- "注册设备"接口
- "修改设备信息"接口
- "删除设备"接口

配置 Postman

步骤1 导入Postman环境变量。

1. 点击右上角的 ^{See} 图标,打开"MANAGE ENVIRONMENTS"窗口。



2. 点击"Import",导入OceanConnect.postman_environment.json文件。

sent is a set of variables that allow you to				
Terrers a secon variables char allow you to	switch the conte	xt of your rec	uests. Environme	nts can be shared
ultiple workspaces. Learn more about env	vironments			
lare a variable in an environment and giv	/e it a starting valu	ue, then use it	in a request by p	utting the variable
curly-braces. Create an environment to	get started.			0
				_
		Globals	Import	Add

 点击"Manage Environments",选择导入的"OcceanConnect"环境,修改 IAM终端节点"IAMEndpoint"、物联网平台终端节点"IOTDMEndpoint"、 华为云用户名"IAMUserName"、华为云密码"IAMPassword"、华为云账号 名"IAMDoaminId"、区域"region"。这里已经填写好北京四站点对应的IAM 终端节点、物联网平台终端节点和区域,如果您是在北京四站点订购的"设备管 理"服务,只需要修改华为云用户名、华为云密码和华为云账号名。

OceanConnect	•	•			
	Manage Er	nvironments			
MANAGE ENVIRONMENTS X An environment is a set of variables that allow you to switch the context of your requests. Environments can be shared between multiple workspaces. Learn more about environments					
MANAGE ENVIRONMENTS An environment is a set of variables that allow you to switch between multiple workspaces. Learn more about environm	n the context of your requests. I <mark>ents</mark>	X			

		Globals	Import	Add
МА	NAGE ENVIRONMENTS			;
Env	vironment Name			
0	ceanConnect			
	VARIABLE	CURRENT VALUE	••• Persis	t All Reset All
	✓ IAMEndpoint	iam.cn-north-4.myh	uaweicloud.com	
	✓ IOTDMEndpoint	iotdm.cn-north-4.m	yhuaweicloud.coi	n
	✓ IAMUserName	hwstaff_xxxxxxxx		
	✓ IAMPassword	******		
	✓ IAMDoaminId	hwstaff_xxxxxxxx		
	✓ region	cn-north-4		
	X-Auth-Token	MIIbkAYJKoZIhvcNA	QcCollbgTCCG30	CAQExDTALBglg
	✓ project_id	06f54d66be8026682	2f21c014815a69b	a
【档版本 15 (2022-06-14)	✓ product_id 版权所有 © 华为技术	、有限公司5e5dfa4258590c072	4217738	
	✓ device_id	5e5dfa4258590c072	4217738_ABC123	3456789
	Add a new variable			

4. 返回主页,选择环境变量为刚导入的"OcceanConnect"。



步骤2 导入"应用侧API调用(V5版本).postman_collection.json"。

💋 Postman		
File Edit View Help		
🕂 New 🔻 Import Runner 📭		🚦 My Workspace 🔻 🍰 Invite
Q Filter	+	
History Collections APIs BETA		IMPORT ×
		Import a Postman Collection, Environment, data dump, curl command, or a RAML / WADL / Open API (1.0/2.0/3.0) / Runscope file.
No internet connection You need to be online to view your APIs. Check your internet connection and try again. Try Again		Import File Import Folder Import From Link Paste Raw Text

成功后:

History	Collections	APIs beta
+ New Co	Trash	
▼ ■ ^{应用} 8 req	侧API调用(V5版本) uests	
▼ ■ 01	Token管理	•••
POST	英取IAM用户Token	
• • 02	项目管理	•••
GET 곁	E询IAM用户可以访问的 ¹	项目列表
▼ ■ 03;	产品管理	
POST 仓	刘建产品	
GET Ž	室询产品	
DEL 册	削除产品	
• • 04	设备管理	•••
POST 注	È 册 设 备	
GET 곁	至询设备	
DEL 册	削除设备	

----结束

调测 "获取 IAM 用户 Token" 接口

在访问物联网平台业务接口前,应用服务器需要调用"获取IAM用户Token"接口鉴权,华为云认证通过后向应用服务器返回鉴权令牌**X-Subject-Token**。

应用服务器需要构造一个HTTP请求,请求示例如下:

POST https://iam.cn-north-4.myhuaweicloud.com/v3/auth/tokens Content-Type: application/json

"auth": {
"identity": {
"methods" [.] [
"password"
passworu
],
"password": {
"user"·{
name : username ,
"password": "********",
"domain": {
"name". "domainname
וומותי uomannanna
}
}
}
3
Ji "ccono": [
scope : {
"project": {
"name": " <i>xxxxxxx</i> "
}
ر ۱
3
}

接下来参考API文档,调测"获取IAM用户Token"接口。

步骤1 配置"获取IAM用户Token"接口的HTTP方法、URL和Headers。

POS	T 获取	IAM用户To	oken	× + •	••				
▶ 获I	取IAM	用户 Tok	en						
POS	ST	Ŧ	https://	{{IAMEndpoint}}/v	3/auth/toke	ns			
Para	ms	Author	ization	Headers (1)	Body ●	Pre-requ	iest Script	Tests ●	Setting
• +	leade	rs (1)							
	KEY						VALUE		
~	Content-Type						application/json;charset=utf-8		
	Key						Value		

步骤2 配置"获取IAM用户Token"接口的Body。

Post 获取IAM用户Token × + ••••								
▶ 获取IAM	▶获取IAM用户Token							
POST	 https:// 	//{{IAMEndpoint}}/	v3/auth/toke	ens				
Params	Authorization	Headers (1)	Body 🌢	Pre-ree	quest Script	Tests	Settings	
none	form-data	x-www-form-u	urlencoded	🖲 raw	binary	GraphQL ^B	eta json	•
1 • { 2 • 3 • 4 · 5 • 6 • 7 · 8 · 9 • 10 · 11 · 12 · 13 · 14 · 15 • 16 • 17 · 18 · 19 · 20 · 21 }	<pre>"auth":{ "identity" "metho "passw "u "u " " " scope":{ "proje " " } } }</pre>	:{ ds":["password"] ord":{ "name":"{{IAMU "password":"{{ "domain":{ "name":"{{ } } ct":{ ame": "{{region}	, IAMPasswor IAMDoaminI ;}"	, d}}", d}}"				

步骤3 点击"Send",在下方查看返回码和响应消息内容。

Body Cookies Headers (16) Test Results	Status: 201 Created Time: 473ms Size: 27.77 KB Save Response 💌
KEY	VALUE
Date 0	Wed, 04 Mar 2020 01:00:53 GMT
Content-Type 🕕	application/json; charset=UTF-8
Content-Length	18468
Connection 🕕	keep-alive
X-IAM-Trace-Id	token_cn-north-4_null_5e627fb3ddfc776374456e059c3666a8
Cache-Control 🕕	no-cache, no-store, must-revalidate
Pragma 🕕	no-cache
Expires 🕕	Thu, 01 Jan 1970 00:00:00 GMT
X-Subject-Token 🕕	${\sf MIIbZAYJKoZIhvcNAQcCoIIbVTCCG1ECAQExDTALBglghkgBZQMEAgEwghl2BgkqhkiG9w0BBwGggh}$
X-Request-Id 🔘	c93c1b0311803c589f61b89ef900b48d
Server 💿	api-gateway
Strict-Transport-Security	max-age=31536000; includeSubdomains;
X-Frame-Options 🕕	SAMEORIGIN
X-Content-Type-Options 💿	nosniff
X-Download-Options	noopen
X-XSS-Protection ()	1; mode=block;

步骤4 请将返回头域中的的X-Subject-Token更新到"OcceanConnect"环境的"X-Auth-Token"参数中,以便于在调用其它接口时使用。若超过令牌有效时间,需要重新调用鉴权接口。

Ocean	Connect					
,	VARIABLE	CURRENT VALUE Persist All Reset A				
\checkmark	IAMEndpoint	iam.cn-north-4.myhuaweicloud.com				
\checkmark	IOTDMEndpoint	iotdm.cn-north-4.myhuaweicloud.com hwstaff_y00465615				
\checkmark	IAMUserName					
\checkmark	IAMPassword	Huawei@123				
\checkmark	IAMDoaminId	hwstaff_y00465615				
\checkmark	region	cn-north-4				
	X-Auth-Token	MIIbZAYJKoZlhvcNAQcCoIIbVTCCG1ECAQExDTAX •				
project_id 06f54d66be8026682f21c014815a69ba product_id 5e5efefc9071cb07289e7733						
					\checkmark	device_id
	Add a new variable					

Cancel

这里我们已经在postman中自动更新了"**X-Auth-Token**"参数,使用时无需手动操作。

▶ 获取IAM用户Token												
POST	Г	•	https://{{ <mark>IAMEndpoint}</mark> }/v3/auth/tokens									
Param	าร	Author	ization	Headers (9)	Body	Pre-request Script	Tests	Settings				
1 2 3 4	//获I var t //将t pm.er	权响应参 coken = coken设 nvironm	头域X-Sub = pm.resp :置到X-Au nent.set(oject-Token ponse.headers.g uth-Token环境变i ("X-Auth-Token",	et("X-Subjec 量,作为后续 、token);	t-Token"); 请求鉴权头域						

----结束

调测 "查询 IAM 用户可以访问的项目列表"接口

在访问物联网平台业务接口前,应用服务器需要调用"查询IAM用户可以访问的项目 列表"接口获取用户的项目ID,用于后续访问物联网平台业务接口。

应用服务器需要构造一个HTTP请求,请求示例如下:
GET https://iam.cn-north-4.myhuaweicloud.com/v3/auth/projects Content-Type: application/json X-Auth-Token: *******

接下来参考API文档,调测"查询IAM用户可以访问的项目列表"接口。

步骤1 配置"查询IAM用户可以访问的项目列表"接口的HTTP方法、URL和Headers。

▶ 查询IAM用户可以访问的项目列表								
GET								
Params	Author	ization	Headers (9)	Body	Pre-reques	t Script	Tests	Settings
▼ Heade	rs (2)							
KEY						VALUE		
Cont	ent-Type					applicat	ion/json	
X-Au	th-Token					{{X-Auth	-Token}}	

步骤2 点击"Send",在下方查看返回码和响应消息内容。

Body	Cookies	Headers (15) Test Results	Status: 200 O
Pre	tty Ra	uw Preview Visualize BETA JSON 🔻 🚍	
1 2 3	{	'projects": [{ 	
4		domain_i0 : Dazirbizcrc440559954a2aC9a99525a ,	
5		15_0000411 : Talse, "papant id", "bal;	
7		"name": "an-southeast-1"	
8		"description": "".	
9		"links": {	
10		"self": "https://iam.myhuaweicloud.com/v3/projects/072a8dcbc980100d2f0ec0146f237196"	
11			
12		id": "072a8dcbc980100d2f0ec0146f237196",	
13		"enabled": true	
14		},	
15		{	
16		"domain_id": "ba21fb12cfc440569954a2ac9a99323a",	
17		"is_domain": false,	
18		<pre>"parent_id": "ba21fb12cfc440569954a2ac9a99323a",</pre>	
19		"name": "MOS",	
20		"description": "",	
21		"links": {	
22		"self": "https://iam.myhuaweicloud.com/v3/projects/b6c7508ff62e4beb91cee1c1ce49ecd9"	
23		},	
24		"id": "b6c7508ff62e4beb91cee1c1ce49ecd9",	
25		"enabled": true	
26		},	

步骤3 返回body中包含一个projects列表,查找其中"name"参数值与"OceanConnect" 环境中"region"参数值相同的条目,取其"id"参数值更新到"OceanConnect" 环境中"project_id"参数,以便于在调用其它接口时使用。

ody Cookie	s Headers (15) Test Results	Status: 200 O
Pretty	Raw Preview Visualize BETA JSON V	
95	},	
96	"id": "072a8dcbd08026542f00c014ee62ff50",	
97	"enabled": true	
98	},	
99	(
100	"domain_id": "ba21fb12cfc440569954a2ac9a99323a",	
101	"is_domain": false,	
102	"parent_id": "ba21fb12cfc440569954a2ac9a99323a",	
103	"name": "cn-north-4",	
104	"description": "",	
105	"links": {	
106	<pre>"self": "https://iam.myhuaweicloud.com/v3/projects/06f54d66be8026682f21c014815a69ba"</pre>	
107		
108	"id": "06f54d66be8026682f21c014815a69ba",	
109	"enabled": true	
110	},	
111	(
112	"domain_id": "ba21fb12cfc440569954a2ac9a99323a",	
113	"is_domain": false,	
114	"parent_id": "ba21fb12cfc440569954a2ac9a99323a",	
115	"name": "ap-southeast-3",	
116	"description": "",	
117	"links": {	
118	"self": "https://iam.myhuaweicloud.com/v3/projects/072a8dcbcd0026502fb1c014ead6fc7a"	
119	},	
120	"id": "072a8dcbcd0026502fb1c014ead6fc7a",	
121	"enabled": true	
122	·	

MANAGE ENVIRONMENTS

Environment Name

OceanConnect

	VARIABLE	CURRENT VALUE () ···· Persist All Reset All
~	IAMEndpoint	iam.cn-north-4.myhuaweicloud.com
✓	IOTDMEndpoint	iotdm.cn-north-4.myhuaweicloud.com
✓	IAMUserName	hwstaff_y00465615
✓	IAMPassword	Huawei@123
✓	IAMDoaminId	hwstaff_y00465615
~	region	cn-north-4 X ••••
\checkmark	X-Auth-Token	MIIbZAYJKoZIhvcNAQcCoIIbVTCCG1ECAQExDTALBglg
\checkmark	project_id	06f54d66be8026682f21c014815a69ba
✓	product_id	5e5efefc9071cb07289e7733
✓	device_id	5e5efefc9071cb07289e7733_ABC123456789
	Add a new variable	

Cancel

Update

这里我们已经在postman中自动更新了"**project_id**"参数,使用时无需手动操作。

▶ 查询IAM用户可以访问的项目列表							
GET	https://{{IAMEndpoint}}/v3/auth/projects						
Params	Params Authorization Headers (9) Body Pre-request Script Tests Setting:						
1 var 2 var 3 var 4 • for 5 • 6 7 8 }	<pre>region jsonDat project (i = 0; if (pro pm. }</pre>	= pm.env a = pm.r s = json i < pro jects[i] environm	<pre>vironment.get("r vesponse.json(); Data.projects; ojects.length; i .name == regior ment.set("project")</pre>	region"); ; i++) { i) { ct_id", pr	ojects[i].id);		

----结束

调测"创建产品"接口

在设备接入物联网平台前,应用服务器需要调用此接口创建产品,后续注册设备时需 要使用这里创建的产品。

应用服务器需要构造一个请求,请求示例如下:

POST https://iotdm.cn-north-4.myhuaweicloud.com/v5/iot/{project_id}/products Content-Type: application/json X-Auth-Token: *******

```
{
 "name" : "Thermometer",
 "device_type" : "Thermometer",
"protocol_type" : "LWM2M",
"data_format" : "binary",
"manufacturer_name" : "ABC",
  "industry" : "smartCity",
  "description" : "this is a thermometer produced by Huawei",
  "service_capabilities" : [ {
   "service_type" : "temperature",
"service_id" : "temperature",
"description" : "temperature",
    "properties" : [ {
     "unit" : "centigrade",
"min" : "1",
"method" : "R",
     "max" : "100",
"data_type" : "decimal",
"description" : "force",
      "step" : 0.1,
      "enum_list" : [ "string" ],
      "required" : true,
"property_name" : "temperature",
      "max_length" : 100
   }],
    "commands" : [ {
      "command_name" : "reboot",
      "responses" : [ {
        "response_name" : "ACK",
        "paras" : [ {
          "unit" : "km/h",
"min" : "1",
"max" : "100",
          "para_name" : "force",
"data_type" : "string",
```

步骤1	<pre> }], "paras": [{ "unit": "km/h", "min": "1", "max": "100", "para_name": "force", "data_type": "string", "description": "force", "step": 0.1, "enum_list": ["string"], "required": false, "max_length": 100 }],], "option": "Mandatory" }], "app_id": "jeQDJQZltU8iKgFFoW060F5SGZka" } 按下来参考API文档, 调测物联网平台"创建产品"接口 注: 在以下步骤中,只呈现样例调测用到的参数。 配置"创建产品"接口的HTTP方法、URL和Headers。 </pre>	l.
	POST Mttps://{{IOTDMEndpoint}}/v5/iot/{{project_id}}/p	products
	Params Authorization Headers (10) Body • Pre-rea	quest Script Tests • Settings
	▼ Headers (2)	
	KEY	VALUE
	Content-Type	application/json
	X-Auth-Token	{{X-Auth-Token}}

步骤2 配置"创建产品"接口的BODY。

"description" : "force",

"step" : 0.1, "enum_list" : ["string"], "required" : false, "max_length" : 100 }]

POST	https://{{IOTDMEndpoint}}/v5/iot/{{project_id}}/products	
Params	Authorization Headers (10) Body • Pre-request Script Tests • Settings	
none	● form-data ● x-www-form-urlencoded ● raw ● binary ● GraphQL ^{BETA} _JSON	
4	"protocol_type": "LWM2M", "data format": "hinary"	

步骤3 点击"Send",在下方查看返回码和响应消息内容。

Body Co	Lookies Headers (7) Test Results Status: 201 Crea	ated
Pretty	Raw Preview Visualize BETA JSON V	
1	R	
2	"product_id": "5e5efefc9071cb07289e7733",	
3	"name": "Thermometer111",	
4	"device_type": "Thermometer",	
5	"protocol_type": "LWM2M",	
6	"data_format": "binary",	
7	"manufacturer_name": "ABC",	
8	"industry": "smartCity",	
9	"description": "this is a thermometer produced by Huawei",	
10	"service_capabilities": [
11		
12	"service_id": "temperature",	
10	Harman Farman Harman	

步骤4 将返回的"product_id"更新到"OceanConnect"环境中的"product_id"参数 中,用于后续其它接口使用。

	nment Name		
Ocear	nConnect		
	VARIABLE	CURRENT VALUE ••• Persist All Reset	t A
\checkmark	IAMEndpoint	iam.cn-north-4.myhuaweicloud.com	
\checkmark	IOTDMEndpoint	iotdm.cn-north-4.myhuaweicloud.com	
~	IAMUserName	hwstaff_y00465615	
~	IAMPassword	Huawei@123	
~	IAMDoaminId	hwstaff_y00465615	
\checkmark	region	cn-north-4	
\checkmark	X-Auth-Token	MIIbZAYJKoZIhvcNAQcCoIIbVTCCG1ECAQExDTALB	gl
~	project_id	06f54d66be8026682f21c014815a69ba	
~	product_id	5e5efefc9071cb07289e7733	
\checkmark	device_id	5e5efefc9071cb07289e7733_ABC123456789	

Cancel

这里我们已经在postman中自动更新了"product_id"参数,使用时无需手动操作。



-----结束

调测"查询产品"接口

应用服务器如果需要查询之前创建的产品详情,可以调用此接口。

应用服务器需要构造一个请求,请求示例如下:

GET https://iotdm.cn-north-4.myhuaweicloud.com/v5/iot/{project_id}/products/{product_id} Content-Type: application/json X-Auth-Token: ********

接下来参考API文档,调测物联网平台"查询产品"接口。

注: 在以下步骤中,只呈现样例调测用到的参数。

步骤1 配置"查询产品"接口的HTTP方法、URL和Headers。

▶ 査	▶ 查询产品								
GET • https:/				{{IOTDMEndpoint	}}/v5/iot/ <mark>{{</mark>	oroject_id}}/pi	roducts/ { {	[product_id]	}}
Params Authorization			ization	Headers (9)	Body	Pre-request Script Test		Tests	Settings
•	Heade	rs (2)							
	KEY						VALUE		
\checkmark	Cont	ent-Type					applicat	ion/json	
~	X-Au	th-Token					{{X-Auth	n-Token}}	

步骤2 点击 "Send",在下方查看返回码和响应消息内容。

iy Coo	skies Headers (14) Test Results	Status: 200
Pretty	Raw Preview Visualize BETA JSON 🔻 🚍	
1	0	
2	"product_id": "5e5efefc9071cb07289e7733",	
3	"name": "Thermometer111",	
4	"device_type": "Thermometer",	
5	"protocol_type": "LWM2M",	
6	"data_format": null,	
7	"manufacturer_name": "ABC",	
8	"industry": "smartCity",	
9	"description": "this is a thermometer produced by Huawei",	
10	"service_capabilities": [
11	{	
12	"service_id": "temperature",	
13	"service_type": "temperature",	
14	"properties": [
15	{	
16	"property_name": null,	
17	"required": true,	
18	"data_type": null,	
19	"enum_list": [],	
20	"min": "1",	
21	"max": "100",	
22	"max_length": 1024,	
23	"step": 0.1,	
24	"unit": "centigrade",	
25	"method": "R",	
26	"description": null	
27		
28],	
29	"commands": [
30		

----结束

调测"注册设备"接口

在设备接入物联网平台前,应用服务器需要调用此接口在物联网平台注册设备。在设 备接入物联网平台时携带设备唯一标识,完成设备的接入认证。

应用服务器需要构造一个HTTP请求,请求示例如下:

POST https://iotdm.cn-north-4.myhuaweicloud.com/v5/iot/{project_id}/devices Content-Type: application/json X-Auth-Token: *******

{

"node_id" : "ABC123456789",	
"device_name" : "dianadevice",	
"product_id" : "b640f4c203b7910fc3cbd446ed437cbd",	
"auth_info" : {	
"auth_type" : "SECRET",	
"secure_access" : true,	
"fingerprint" : "dc0f1016f495157344ac5f1296335cff725ef22f",	
"secret" : "3b935a250c50dc2c6d481d048cefdc3c",	
"timeout" : 300	
},	
"description" : "watermeter device"	

接下来参考API文档,调测物联网平台"注册设备"接口。

注: 在以下步骤中,只呈现样例调测用到的参数。

步骤1 配置"注册设备"接口的HTTP方法、URL和Headers。

▶ 注册设备	۲ H							
POST	ST https://{{IOTDMEndpoint}}/v5/iot/{{project_id}}/devices							
Params	Authoriz	zation	Headers (10)	Body •	Pre-req	uest Script	Tests •	Settings
▼ Heade	ers (2)							
KEY						VALUE		
Content-Type						application/json		
🗸 X-Au	th-Token					{{X-Auth-To	ken}}	

步骤2 配置"注册设备"接口的Body。

POST	https://{{IOTDMEndpoint}}/v5/iot/{{project_id}}/devices	
Params	Authorization Headers (10) Body • Pre-request Script Tests • Settings	
none	● form-data ● x-www-form-urlencoded ● raw ● binary ● GraphQL ^{BETA} JSO	N
2 3 4	<pre>"node_id": "ABC123456789", "device_name": "dianadevice", "product_id": "{{product_id}}", "auth_info": {</pre>	

步骤3 点击"Send",在下方查看返回码和响应消息内容。

Body C	ookies Headers (7) Test Results	Status: 201 Created
Pretty	Raw Preview Visualize BETA JSON 🔻 🚍	
1	(
2	<pre>"app_id": "PAutVGQZoEVJCncftia5MFeeUlEa",</pre>	
3	"device_id": "5e5efefc9071cb07289e7733_ABC123456789",	
4	"node_id": "ABC123456789",	
5	"gateway_id": "5e5efefc9071cb07289e7733_ABC123456789",	
6	"device_name": "dianadevice",	
7	"node_type": "GATEWAY",	
8	"description": "watermeter device",	
9	"fw_version": null,	
10	"sw_version": null,	
11	"auth_info": {	
12	"auth_type": "SECRET",	
13	"secret": "3b935a250c50dc2c6d481d048cefdc3c",	
14	"fingerprint": null,	
15	"secure_access": true,	
16	"timeout": 300	
17	},	
18	"product_id": "5e5efefc9071cb07289e7733",	
19	"status": "INACTIVE",	
20	"create_time": "20200304T010621Z",	
21	"tags": []	
22	}	

步骤4 请将返回的"device_id"更新到"OceanConnect"环境中的"device_id"参数中, 用于后续其它接口使用。

MANAGE ENVIRONMENTS	>
Environment Name	
OceanConnect	
VARIABLE	CURRENT VALUE () ···· Persist All Reset All
IAMEndpoint	iam.cn-north-4.myhuaweicloud.com
IOTDMEndpoint	iotdm.cn-north-4.myhuaweicloud.com
IAMUserName	hwstaff_y00465615
IAMPassword	Huawei@123
IAMDoaminId	hwstaff_y00465615
✓ region	cn-north-4
X-Auth-Token	MIIbZAYJKoZlhvcNAQcCoIIbVTCCG1ECAQExDTALBglg
✓ project_id	06f54d66be8026682f21c014815a69ba
product_id	5e5efefc9071cb07289e7733
device_id	5e5efefc9071cb07289e7733_ABC123456789
Add a new variable	

Cancel

opuate

这里我们已经在postman中自动更新了"device_id"参数,使用时无需手动操作。

▶ 注册设备	ŕ						
POST	•	https://	{{IOTDMEndpoint}	}/v5/iot/ <mark>{{pro</mark> j	<mark>ject_id}</mark> /devices		
Params	Authori	zation	Headers (10)	Body •	Pre-request Script	Tests 🌢	Settings
1 var 2 var 3 pm.	jsonDat device_ environm	a = pm.r id = jsc ent.set(<pre>response.json(); onData.device_id; "device_id", device_id</pre>	; /ice_id);			

----结束

调测"查询设备"接口

应用服务器需要查询在物联网平台注册的设备详情时,可以调用此接口。

应用服务器需要构造一个HTTP请求,请求示例如下:

GET https://iotdm.cn-north-4.myhuaweicloud.com/v5/iot/{project_id}/devices/{device_id} Content-Type: application/json X-Auth-Token: *******

接下来参考API文档,调测物联网平台"查询设备"接口。

注: 在以下步骤中,只呈现样例调测用到的参数。

步骤1 配置"查询设备"接口的HTTP方法、URL和Headers。

▶ 查询设谷	<u>r</u>							
GET • https://{{IOTDMEndpoint}}/v5/iot/{{project_id}}/devices/{{device_id}}								
Params	Author	ization	Headers (9)	Body	Pre-reques	t Script	Tests	Settings
▼ Heade	ers (2)							
KEY						VALUE		
Content-Type						applicat	ion/json	
X-Auth-Token						{{X-Auth	-Token}}	

步骤2 点击"Send",在下方查看返回码和响应消息内容。

Body Cook	okies Headers (14) Test Results Statu	s: 200 OK
Pretty	Raw Preview Visualize BETA JSON 🔻 🚍	
1 { 2 3 4 5 6 7 8 9 9 10 11 12 13	<pre>("app_id": "PAutVSQZoEVJCncftia5MFeeUlEa", "device_id": "SeSefefc9071cb07289e7733_ABC123456789", "node_id": "ABC123456789", "device_name": "dianadevice", "device_name": "GATEWAY", "device_tope": "GATEWAY", "device_tope": "Watermeter device", "fw_version": null, "sw_version": null, "auth_type": "SECRET", "secret": "*******, "figregneniat": null </pre>	
15	"secure access": true,	
16	"timeout": 0	
17	},	
18	"product_id": "5e5efefc9071cb07289e7733",	
19	"status": "INACTIVE",	
20	"create_time": "20200304T010621Z",	
21	"tags": []	
22 }	}	

----结束

4.6.1.7.1.3 使用 Java API Demo 调测(联通用户专用)

本文档基于调用API接口的代码样例(Java)进行指导,接口信息详见API参考文档。

准备 Java 开发环境

本小节以Windows操作系统为例,介绍安装JDK1.8以及安装Eclipse的方法,如果您使用其它类型的开发环境,请根据自己的需要完成部署。

- **步骤1** 进入Java JDK官网,下载JDK1.8版本(比如: jdk-8u161-windows-x64.exe),双击进行安装。
- 步骤2 配置Java环境变量。
 - 1. 右键单击"计算机",选择"属性"。

	1		_
200		打开(0)	
计算		用 FAR 打开	
		用 FAR 编译到 HTML 帮助	
		用 FAR 复制或备份	
	۲	管理(G)	
		映射网络驱动器(N)	
		断开网络驱动器(C)	
		创建快捷方式(S)	
		删除(D)	
		重命名(M)	
		属性(R) ▶	

2. 点击"高级系统设置"。



3. 点击"环境变量"。

系统属性
计算机名 硬件 高级 系统保护 远程
要进行大多数更改,您必须作为管理员登录。 一性能 视觉效果,处理器计划,内存使用,以及虚拟内存
设置 (S)
一用户配置文件 与您登录有关的桌面设置
设置(2)
启动和故障恢复 系统启动、系统失败和调试信息
设置(T)
环境变里 (8)
确定 取消 应用 (A)

4. 配置系统变量。需配置3个变量:JAVA_HOME、Path、CLASSPATH(不区分大小写)。若变量名已经存在,则点击"编辑";若变量名不存在,则点击"新建"。一般Path变量存在,JAVA_HOME变量和CLASSPATH变量需要新增。

的用户	(U) 重变
受量	值
PRO_LANG	zh-en
TEMP	%USERPROFILE%\AppData\Local\Temp
TMP	%USERPROFILE%\AppData\Local\Temp +
	新建 01) 编辑 02) 删除 00)
74大亦母 (c)	〔新建 01)〕[编辑 02) 聞除 00)
< <p> 4.</p>	(新建 01)) [编辑 02) 開除 02)
< <p> (5) (5) (5) (5)</p>	「新建 (M)」「编辑 (E)」 開除 (D) 値
《统变量 (S) 变量 PRO_LANG	新建 (M) [编辑 (Z) 删除 (D) 值 zh=cn
《统变里 (S) 变里 PRO_LANG PROCESSOR_AR	新建 (M) 編辑 (E) 刪除 (D) 值 zh=cn x86
《统变里 (S) 变里 PRO_LANG PROCESSOR_AR PROCESSOR_ID	新建 (M) (编辑 (E) 剛除 (D) 值 zh-cn x86 x86 Family 6 Model 62 Stepping
《统变量 (S) 变量 PRO_LANG PROCESSOR_AR PROCESSOR_ID PROCESSOR_LEVEL	新建(M) 编辑(E) 删除(D) 值 zh-cn x86 x86 Family 6 Model 62 Stepping

JAVA_HOME指明JDK安装路径,配置示例:C:\ProgramFiles\Java\jdk1.8.0_45。 此路径下包括lib,bin等文件夹。

新建系统变量	×
变 里 名(M):	JAVA_HOME
变 里 值(V):	C:\ProgramFiles\Java\jdk1.8.0_45
	确定即消

Path变量使系统可以在任何路径下识别Java命令。如果Path变量已经存在,则需 在变量值最后添加路径,配置示例:;C:\Program Files\Java\jdk1.8.0_45\bin;C: \Program Files\Java\jdk1.8.0_45\jre\bin。

两个路径之间需要使用	";"	分割,	分号是英文半角。
------------	-----	-----	----------

编辑系统变量	X
变量名(M): 变量值(V):	Path gram Files\Java\jdk1.8.0_45\jre\bin
	确定 取消

CLASSPATH为Java加载类(class或lib)路径,只有配置CLASSPATH,Java命令才 能识别。配置示**例:** .;%JAVA_HOME%\lib\dt.jar;%JAVA_HOME%\lib\tools.jar。 **注:** 路径以"."开始,表示当前路径。

新建系统变量	—
变量名(M):	CLASSPATH
变 重 值(V):	.ib\dt.jar;%JAVA_HOME%\lib\tools.jar
	确定 取消

5. 选择"开始 > 运行",输入"cmd",执行命令:java -version、java、javac。 如果命令可以执行,则说明环境变量配置成功。

```
C:\Users\z00293999>java -version
java version "1.8.0_45"
Java(TM) SE Runtime Environment (build 1.8.0_45-b15)
Java HotSpot(TM) Client VM (build 25.45-b02, mixed mode)
```

步骤3 前往Eclipse官网,下载Eclipse安装包,直接解压缩到本地即可使用。

----结束

导入 Demo 工程

步骤1 运行Eclipse,选择"File > New > Project",在弹出的对话框中选择"Java Project",点击"Next"。

💽 New Project	_ — X
Select a wizard	>
Create a Java project	
Wizards:	
type filter text	
EJB	_
i≘⊱ Java <mark>⊮</mark> Java Project	
🖉 🥼 🛣 Java Project from Existing Ant Buildfile	
🗈 🗁 Java EE	
I II → B JavaScript	
H-C- Mayen	
E-> Plug-in Development	_
	•
A Back Next > Finish	Cancel

步骤2 填写"Project name",JRE版本选择"JavaSE-1.8",点击"Finish"。

💽 New Java Project	_ 🗆 🗙		
Create a Java Project			
Create a Java project in the workspace or in an external location.			
Project name: LiteNAdemo			
✓ Use default location			
Location: C:\Users\w00230058\workspace\LiteNAdemo	Browse		
JRE			
Use an execution environment JRE: JavaSE-1.8	•		
C Use a project specific JRE: jdk1.8.0_45	-		
C Use default JRE (currently 'jdk1.8.0_45')	onfigure JREs		
Project layout CUse project folder as root for sources and class files			
? < Back Next > Finish	Cancel		

步骤3 下载API JAVA Demo,并解压。

步骤4 完成解压后,拷贝(Ctrl+C) "Open source components"和 "src"文件夹。

LiteNAdemo_https	• 😽 搜索 LiteNAd	lemo_https 👂
	:=-	• 🔟 🔞
名称	修改日期	类型
Open source components src	2017/5/10 19:30 2017/5/10 19:30	文件夹 文件夹

步骤5 打开在Eclipse创建的工程,点击选中工程名称,将拷贝的文件粘贴(Ctrl+V)到该工程目录下。

Project Explorer 🕱	
🗄 🖾 LiteNAdemo	
🥶 src	
🖹 📄 🚔 JRE System Library [Jav	/aSE-1.8]
🕂 🐻 resources.jar - C:\Pi	rogram Files\Java\jre1.8.0_45\lib
🖪 🚾 rt.jar - C:\Program I	Files\Java\jre1.8.0_45\lib
🗄 🚾 jsse.jar - C:\Program	n Filach IavaNina1 8 0 45Nlih
😟 💮 jce.jar - C:\Program	Question 🗙
🛓 🐻 charsets.jar - C:\Pr	
🚊 🐻 j fr.jar - C:\Program	LiteiNAdemo/src exists. Do you wish to overwrite?
🛓 👼 access-bridge.jar -	•
🛓 💮 🔂 🔂 🕂 📴 🗄	
🚊 🐻 dnsns.jar - C:\Prog	Yes Yes To All No Cancel
im 👼 jaccess.jar - C:∖Pro	
🛓 💮 🧰 jfxrt.jar - C:\Program	n Files\Java\jre1.8.0_45\lib\ext
😟 🐻 localedata.jar - C:\P	rogram Files\Java\jre1.8.0_45\lib\ext
⊡… nashorn.jar - C:\Pro	gram Files\Java\jre1.8.0_45\lib\ext
🖃 🚾 sunecjar - C:\Progr	am Files\Java\jre1.8.0_45\lib\ext

粘贴完成后,"src"目录下的文件存在错误。



步骤6 右键单击工程名称,选择"Properties > Java Build Path > Libraries > Add JARs", 在弹出框中选中"Open source components"目录下的所有jar文件,点击"OK"。





应用接入

应用服务器需要调用物联网平台的"鉴权"接口,完成应用服务器和物联网平台的对接,接口信息详见API参考文档。

本文档基于调用API接口的代码样例(Java)进行指导,帮助开发者理解"鉴权"接口的调用。

步骤1 在eclipse中,选择"src > com.utils > Constant.java",修改BASE_URL、APPID、 SECRET,然后保存(Ctrl+S)。

🗸 🕞 src	3 package com.utils;	
> 🔠 com.service.deviceAccess.appAccessSecurity	4	
	5 public class Constant {	
> # com.service.deviceAccess.dataCollection	6	
> # com.service.deviceAccess.deviceManagement	provide the product of the and product of the for the former of the set of	
> com.service.deviceManagement.batchTask	9	
> com.service.deviceManagement.deviceGroupManagement	0 //please replace the appId and secret, when you use the demo.	
> # com.service.deviceManagement.deviceUpgrade	1 public static final String APPID = "lzfuCR6V5*******HOnhMhy7WAa";	
> # com.service.deviceManagement.messagePushing	2 public static final String SECRET = "OpLXC5fN********pDM24P5fw8a";	
> com.service.deviceManagement.subscribtionManagement	3 @ /*	
> com.tools.testMessagePush	5 *IF and port of callback url.	
✓	6 *please replace the IP and Fort of your Application deployment environment address, when you use the	demo.
Constantiava	7 */	
DefaultHortnameVerifier inva	8 public static final String CALLBACK_BASE_URL = "http://10.10.10.99:9999";	
> M Denanti losti ante vermer Java	9	

配置说明如下:

- BASE_URL: 填写应用对接地址/端口号。
- APPID: 填写创建应用或项目后获取的应用ID。
- SECRET:填写创建应用或项目后获取的应用密钥。
- **步骤2** 在eclipse中,选择"src > com.huawei.service.appAccessSecurity",右键单击 "Authentication.java",选择"Run As > Java Application"。

	4 🔛 LiteNAdemo_https				300	import java util HachMan-[]
JRE System Library [JavaSE-1.8]		10	where a ferring and the			
4 进 src			110	/**		
4 🌐 com.huawei.service.appAccessSecurity		12	* Auth:			
	> D Authent	·			13	Inis interface is used to authenticate third-party systems before third-party systems access open APIS. */
	⊳ 🕡 Refresh	New				ublic class Authentication {
	b # com huawe	Open Ty	pe Hierarchy		F4	<pre>@SuppressWarnings("unchecked")</pre>
	> # com huawa	Chan In		Also Chife -	141.5	<pre>public static void main(String args[]) throws Exception {</pre>
	b = com huawe	SHOW IN		AILTSHILT		// Two-Way Authentication
	b m continuawe	Open			F3	<pre>HttpsUtil httpsUtil = new HttpsUtil();</pre>
	D 🖶 com.nuawe	Open Wi	th		•	httpsUtil.initSSLConfigForTwoWay();
	b to com.huawe	Com		Ctrl		
	b com.huawe	Copy	10 L M	cur		String appld = Constant_APPlD; String servet = Constant_SFCPET:
	E com.huawe	Copy Qu	aimed Name			String urllogin = Constant.APP AUTH;
	🖻 🌐 com.huawe 👔	Paste		Ctrl	+V	
	> 🌐 com.huawe 💥	Contraction Contractic Contr		Dele	ete	Map <string, string=""> param = new HashMap<>();</string,>
	👂 🌐 com.huawe 🔬	Remove	from Context	Ctrl+Alt+Shift+Do	wn	param.put("appld", appld);
	E resource	Build Pat	h			poloniped server, server,
	E resource.ce	Source		Alt+Shift		<pre>StreamClosedHttpResponse responseLogin = httpsUtil.doPostFormUrlEncodedGetStatusLine(urlLogin, param);</pre>
	Referenced Lil	D ()		Alter Shine		
	Den source e	Refactor		Alt+Shift	+1.	System out.printin(app auth success,return accessioken:); System out printin(response) ogin getStatue[ing());
	b 🔛 south matt dema	Import				System.out.println(responseLogin.getContent());
		Evnort				System.out.println();
	-	- coportan				
	8	Refresh			F5	ManCString. Strings data = new HashManc>():
		0.6				<pre>data = JsonUtil.jsonString2SimpleObj(responseLogin.getContent(), data.getClass());</pre>
		Keterenc	85		'	<pre>String accessToken = data.get("accessToken");</pre>
		Declarati	ons			System. <i>out</i> .println("accessToken:" + accessToken);
		Coverage	e As			3
		Run As			•	I lave Application Att-Shift+Y
		Dahua A			-	z zara Application - Automiti A, 2
		Debug A	`		•	Run Configurations
		Validate				

步骤3 在控制台查看响应消息的打印日志,如果获得accessToken,说明鉴权成功。

accessToken请妥善保存,以便于在调用其它接口时使用。

🖹 Problems 📮 Console 🕱	• × × • • • • • • • • •
<terminated> Authentication [.</terminated>	ava Application] C:\Program Files\Java\jdk1.8.0_45\bin\javaw.exe (2018年3月14日 下午4:37:54)
app auth success,return a HTTP/1.1 200 OK{"accessTo	ccessToken: ken":"8d44bc6cdb60e863dfdbb8ccfed4e51","tokenType":"bearer","refreshToken":"9e11397df47877fe2afe4cd5c463621","expiresIn":3600,
accessToken:8d44bc6cdb60e	863dfdbb8ccfed4e51

如果没有得到正确的响应,请检查全局常量是否修改正确,并排除网络问题。

----结束

订阅数据

应用服务器通过调用物联网平台的"订阅平台业务数据"接口,告知物联网平台消息 推送的地址和通知类型,比如设备业务数据、设备告警等,接口信息详见API参考文 档。

在订阅场景下,物联网平台是客户端,应用服务器是服务端,物联网平台调用应用服务器的接口,并向应用服务器推送消息。在Java API Demo中,使用HTTP协议接收物 联网平台的推送消息,不需要在物联网平台加载CA证书。 本文档基于调用API接口的代码样例(Java)进行指导,帮助开发者理解"订阅平台业务数据"接口的调用。

步骤1 在eclipse中,选择 "src > com.huawei.utils > Constant.java",修改 "CALLBACK_BASE_URL",填写回调的IP地址和端口号。

同一个应用下,所有订阅类型的回调地址的IP和端口必须一致。回调地址的合法性和 连通性可以通过开发中心的"订阅调试"功能进行检测。



步骤2 在eclipse中,选择"src > com.huawei.service.subscribtionManagement",右键单击"SubscribeServiceNotification.java",选择"Run As > Java Application"。

🗉 😂 LiteNAdemo_https	2 2 import inva util MachManu[]
JRE System Library [JavaSE-1.8]	
🔺 🕮 src	130 /**
temperature de la companya de la	14 * SubscribeServiceNotification :
temperature de la companya de la	16 */
temperature de la command de livery	17 public class SubscribeServiceNotification {
temperature de la comparation de la comparati	18 anyble static usid main(Staing ange[1) throws Exception (
temperature de la completación de la completació	20
temperature de la companya de la	21 // Two-Way Authentication
temperature de la com.huawei.service.deviceUpgrade	22 HttpsUtil httpsUtil = new HttpsUtil(); 22 httpsUtil idit50 confidentiamelar();
temperature de la companya de la	24
4 🌐 com.huawei.service.subscribtionManagement	25 // Authentication. get token
DeleteBatchSubscriptions.java	26 String accessToken = login(httpsUtil); 27
DeleteSpecifi/Subscription inva	//Please make sure that the following parameter values have been modified in the Constant file.
QueryB New	String appId = Constant.APPID;
QueryS Open Type Hierarchy	F4 String urlSubscribeServiceNotification = Constant.SUBSCRIBE_SERVICE_NOTIFYCATION;
Subscri Show In Alt	+Shift+W)
Subscri Open	F3 /*
> com.huawe	*/
Open with	<pre>String callbackurl_deviceAdded = Constant.DEVICE_ADDED_CALLBACK_URL;</pre>
resource Copy	Ctrl+C String notifyType_deviceAdded = Constant.DEVICE_ADDED;
E resource.c Copy Qualified Name	Map <string, object=""> paramSubscribe deviceAdded = new HashMap<>();</string,>
A Referenced Li Paste	Ctrl+V paramSubscribe_deviceAdded.put("notifyType", notifyType_deviceAdded);
Delete	Delete paramSubscribe_deviceAdded.put("callbackUrl, callbackurl_deviceAdded);
south_mqtt_demc Remove from Context Ctrl+Alt+Sh	paramsubscribe_deviceAdded.put(appid ; appid);
Build Path	<pre>String jsonRequest_deviceAdded = JsonUtil.jsonObj25ting(paramSubscribe_deviceAdded);</pre>
Source	tachifas } ManyString String hander deviceAdded = new MarkManyS();
Beferter	() header deviceAdded_bet(Constant.HEADER APP KEY, appId);
Relactor Al	<pre>trsnitt+1 / header_deviceAdded.put(Constant.HEADER_APP_AUTH, "Bearer" + " " + accessToken);</pre>
🔤 Import	HttpResponse httpResponse deviceAdded = https//til doPostIson(urlSubscribeServiceNotification, beader devi
🗹 Export	hetpicsporse nephesporse_acreenaaca - nepsocration oscialitar asabet receiver receiver receiver and a same receiver as the same receiver and the same rece
8 Refresh	String bodySubscribe_deviceAdded = httpsUtil.getHttpResponseBody(httpResponse_deviceAdded); F5
Peferencer	<pre>System.out.println("SubscribeServiceNotification: " + notifyType_deviceAdded + ", response content:"); SubscribeServiceNotification: " + notifyType_deviceAdded + ", response content:");</pre>
Dedeetier	System.out.println(hcbySubsrie deviceAdded);
Declarations	System.out.println();
Coverage As	>
Run As	J Java Application Alt+Shift+X, J
Debug As	Pue Configurations
Validate	kun Configurations

步骤3 在控制台查看响应消息的打印日志,如果所有类型的订阅均获得"201 Created"响 应,则说明订阅成功。

app auth success,return accessToken: HTTP/1.1 200 OK{"accessToken":"c5166051d466226b1f1a3590d17e4a3a","tokenType":"bearer","refreshToken":"357e88ea50a45d523b7e5ff9165cf58","expiresIn":3600," SubscribeNotification: deviceAdded, response content: HTTP/1.1 201 Created

如需修改订阅的回调地址,在"Constants.java"类中修改"CALLBACK_BASE_URL"的值,再次运行

"SubscribeServiceNotification.java"即可,新的回调地址会覆盖原来的回调地址。

----结束

注册设备

应用服务器通过调用物联网平台的"注册直连设备"接口,在物联网平台添加设备, 接口信息详见API参考文档。

本文档基于调用API接口的代码样例(Java)进行指导,帮助开发者理解"注册直连设 备"接口的调用。

步骤1 在eclipse中,选择 "src > com.huawei.service.deviceManagement >

RegisterDirectConnectedDevice.java ",修改"verifyCode"、"nodeld"、 "timeout"、"manufacturerId"、"manufacturerName"、"deviceType"、 "model"、"protocolType"的取值。



配置说明如下:

- "verifyCode/nodeld"需要与真实设备的唯一标识符(IMEI或mac)一致。如果 使用的是设备模拟器,则"verifyCode"可以是数字、字母和特殊符号的组合, 开发者可自行定义,但不可以与其它设备的verifyCode重复。
- "timeout"单位是"秒", "timeout"的取值作用如下:
 - timeout = 0, 注册的设备不会过期。
 - timeout > 0, 真实设备必须在设置的时间内上线, 否则注册的设备会因为过 期而被物联网平台删除。如果不携带timeout,则默认过期时间是180秒。
 - 在设备绑定成功后,"timeout"不再起作用,注册的设备不会过期。
- "manufacturerId"、"manufacturerName"、"deviceType"、"model"、 "protocolType"需要与对应的Profile保持一致。
- 步骤2 右键单击"RegisterDirectConnectedDevice.java",选择"Run As > Java Application "。
- 步骤3 在控制台查看响应消息的打印日志,如果获得"deviceId",则说明注册成功。

可以在开发中心的"产品 > 设备管理"中,查看新注册的设备是否已经显示。此时, 注册设备只有设备ID(deviceId)信息。

app auth success,return accessToken: HTTP/1.1 200 OK{"accessToken":"1f337bfa6cb85f83243b99fda22d2lb", "tokenType":"bearer", "refreshToken":"be3ccdce5390ed14b81c85f58e2712b2", "expiresIn":3600, RegisterDirectlyConnectedDevice, response content: HTTP/1.1 200 OK{"deviceId":"d0ac5cac-d3af-4e0c-8166-5a67e1c6f0a7","verifyCode":"9999","timeout":0,"psk":"5a7f074ffdb1c46ed104a8efcafd0a0e"}

----结束

其他接口

请参考API参考文档完成其他接口业务的开发。

文档版本 15 (2022-06-14)

单步调测

为了更直观地查看应用程序发送的消息及物联网平台的响应消息,以下方法使用了 Eclipse的断点调试方法。

步骤1 在最终发出http/https消息的代码处设置断点。例如:在样例代码"HttpsUtil.java"中的"executeHttpRequest"方法设置3个断点(请根据您代码的实际情况打断点)。



- **步骤2** 右键单击需要调测的类,例如: "Authentication.java (根据您建立的工程类型进行选择)",选择"Debug As > Java Application"。
- 步骤3 当程序在断点位置停止运行后,点击"Step Over"进行单步调试。此时可以在 "Variables"窗口查看相应变量的内容,包括发送的消息及物联网平台的响应消息。



步骤4 在"Variables"窗口中展开"request"变量,查看请求消息的内容。

1. 选中"request"变量时,可以在下方内容展示区看到应用程序发送请求的URL; 在"entity"中可以看到发送的消息内容。

🗱 🖋 🗣 Breakpoints	🏝 🕫 🖻
Name	Value
▷ ● this	HttpsUtil (id=24)
▲ ○ request	HttpPost (id=25)
aborted	AtomicBoolean (id=84)
⊳ 🧧 cancellableRef	AtomicReference <v> (id=86)</v>
config	null
a entity	UrlEncodedFormEntity (id=88)
chunked	false
⊳ o ^F content	(id=104)
 contentEncoding 	null
⊿ 📀 contentType	BasicHeader (id=106)
Image:	"Content-Type" (id=110)
value	"application/x-www-form-urlencoded" (id=111)
b o headergroup	HeaderGroup (id=91)
> o params	BasicHttpParams (id=93)
POST https:// /iocm/app/sec/v1.:	1.0/login HTTP/1.1

2. 应用ID(appld)和应用密钥(secret)在"content"字段内,使用十进制的 ASCII码表示,需要对照ASCII码表将其转化为字母和符号。

(x)= Variables 🕱 💁 Breakpoints				
Name	Value			
b e this	HttpsUtil (id=24)			
⊿ ⊙ request	HttpPost (id=25)			
⊳ 🝯 aborted	AtomicBoolean (id=301)			
all cancellableRef	AtomicReference <v> (id=303)</v>			
config	null			
a 🍵 entity	UrlEncodedFormEntity (id=305)			
<u>chunked</u>	false			
	(id=311)			
▲ [0]	97 a			
▲ [1]	112 P			
▲ [2]	112 p			
▲ [3]	73 ()			
▲ [4]	100 d)			
▲ [5]	61 =			
▲ [6]	111			
▲ [7]	121			
▲ [8]	98			

步骤5 在"Variables"窗口中展开"response"变量,查看响应消息的内容。

(x)+ Variables 🔀 💁 Breakpoints	<u>*</u> =
Name	Value
▷ ⊜ this	HttpsUtil (id=24)
▷ ○ request	HttpPost (id=25)
🧉 🕴 response	StreamClosedHttpResponse (id=294)
🖌 🧧 content	"{"accessToken":"ff5478725ebaffbeadda5
🗧 hash	0
⊳ 🧧 value	(id=297)
🔺 🧧 original	\$Proxy0 (id=45)
⊿	CloseableHttpResponseProxy (id=51)
🏼 🧧 original	BasicHttpResponse (id=54)
code	200
entity	BasicManagedEntity (id=56)
b o headergroup	HeaderGroup (id=58)
Iocale	Locale (id=63)
params	ClientParamsStack (id=66)
reasonCatalog	EnglishReasonPhraseCatalog (id=68)
reasonPhrase	"OK" (id=70)
statusline	BasicStatusLine (id=78)

在代码样例中,"Authentication.java"之外的类均会先调用鉴权接口。因此,在对 "Authentication.java"之外的类进行单步调测时,需要程序第二次运行到设置断点的 位置时,再查看变量内容。

----结束

4.6.1.7.2 使用 SDK 对接

4.6.1.7.2.1 Java SDK 使用指南(联通用户专用)

非联通用户请查看设备接入服务。

本文以提供的应用侧Java SDK Demo为例说明如何使用JAVA SDK与物联网平台对接,包括证书配置及回调等。Demo以Java工程为例,每个类(除工具类外)都包含了main方法,可单独运行,旨在演示如何调用SDK接口。

开发环境要求

开发平台	开发环境	配套要求	推荐的操作系统
IoT	1) J2EE for Java Developers	JDK 1.8 及以上版 本	Windows7
	2) Maven 插件: m2e - Maven Integration for Eclipse (includes Incubating components)		

SDK包为纯JAVA的JAR包,在使用上没有特殊限制,JDK在1.8及以上版本即可。

导入 Demo 工程

- 步骤1 将下载的JAVA SDK Demo解压到本地。
- **步骤2** 打开eclipse,选择菜单"File > Import",再选择"Maven > Existing Maven Projects",单击"Next"。

€ Import	_ 🗆 X
Select Import Existing Maven Projects	è
Select an import source:	
CVS CVS CVS Constall Check out Maven Projects from SCM Check out Maven Projects Check out Maven Check out Mav	
? Sack Next > Finish	Cancel

步骤3 单击"Browse",选择demo解压后的路径,然后单击"Finish"。

Import Maven Projects	
Maven Projects	
Select Maven projects	
Root Directory: C:\Users\Desktop\North_JAVA_SDK_Demo	Browse
Projects:	
/sourcecode/pom.xml com.huawei.m2m.iotplatform:api-client-demo:1.0.0;jar	Select All
	Deselect All
	Select Tree
	Deselect Tree
	Refresh
Add project(s) to working set	
api-client-demo	-
▶ Advanced	
? Kext > Finish	Cancel

步骤4 如果开发者无法从maven仓库下载Demo工程依赖的jar包,可以从Demo文件的 components文件夹下手动导入工程依赖包。

- 📧 httpmime-4.5.2.jar
- 📧 json-lib-2.4.jar
- 📓 netty-all-4.0.27.Final.jar
- 📓 spring-boot-starter-web-1.5.9.RELEASE.jar

----结束

初始化及证书配置

新建一个**NorthApiClient**实例,设置好**ClientInfo**(包括平台IP、端口、appld和密码),再初始化证书。

平台IP、端口、appld和密码都是从配置文件./src/main/resources/ application.properties中读取的,因此,当这些信息发生变化时,只要修改配置文 件,不用修改应用服务器的代码。本章节所指的证书是平台提供的,在调用平台接口 过程中使用;一般情况下,与回调使用的证书不一样。

使用测试证书:
NorthApiClient northApiClient = new NorthApiClient();
PropertyUtil.init("./src/main/resources/application.properties");
ClientInfo clientInfo = new ClientInfo(); clientInfo.setPlatformIp(PropertyUtil.getProperty("platformIp")); clientInfo.setPlatformPort(PropertyUtil.getProperty("platformPort")); clientInfo.setAppId(PropertyUtil.getProperty("appId")); clientInfo.setSecret(PropertyUtil.getProperty("secret"));
northApiClient.setClientInfo(clientInfo); northApiClient.initSSLConfig();//默认使用测试证书,且不进行主机名校验
如果不使用测试证书,可使用指定证书(如商用证书): NorthAniCliant northAniCliant - now NorthAniCliant():
NorthApictient horthApictient – new NorthApictient(),
PropertyUtil.init("./src/main/resources/application.properties");
ClientInfo clientInfo = new ClientInfo(); clientInfo.setPlatformIp(PropertyUtil.getProperty("platformIp")); clientInfo.setPlatformPort(PropertyUtil.getProperty("platformPort")); clientInfo.setAppId(PropertyUtil.getProperty("appId")); clientInfo.setSecret(getAesPropertyValue("secret"));
SSLConfig sslConfig= new SSLConfig(); sslConfig.setTrustCAPath(PropertyUtil.getProperty("newCaFile")); slConfig.setTrustCAPwd(getAesPropertyValue("newCaPassword")); slConfig.setSelfCertPath(PropertyUtil.getProperty("newClientCertFile")); slConfig.setSelfCertPwd(getAesPropertyValue("newClientCertPassword"));
northApiClient.setClientInfo(clientInfo); northApiClient.initSSLConfig(sslconfig); //使用指定的证书,且默认使用严格主机名校验
使用指定证书时,如果不使用严格主机名校验,在调用 northApiClient.initSSLConfig(sslconfig) 之前可以自行设置主机名校验方法: northApiClient.setHostnameVerifier(new HostnameVerifier() { public boolean verify(String arg0, SSLSession arg1) { // 自定义主机名校验 ······ return true; }
1//
须知

主机名校验方法应以安全为原则,不应该直接返回true。

业务接口调用方法

设置好NorthApiClient实例后才能调用其他业务接口。以如下几个接口为例说明如何 调用业务接口。

关于哪些参数需要设置,请查看《应用侧JAVA SDK API参考》。对于可选参数,如果 业务不需要,可以不设置或者设置为null。

鉴权

//得到NorthApiClient实例后,再使用northApiClient得到鉴权类实例 Authentication authentication = new Authentication(northApiClient);

//调用鉴权类实例authentication提供的业务接口,如getAuthToken AuthOutDTO authOutDTO = authentication.getAuthToken(); //从返回的结构体authOutDTO中获取需要的参数,如accessToken,不同接口的token是通用的,只用获取一次,token有效期为1小时。在即将到1小时前,可调用刷新鉴权接口重新获取token

String accessToken = authOutDTO.getAccessToken();

订阅

//得到NorthApiClient实例后,再使用northApiClient得到订阅类实例 SubscriptionManagement subscriptionManagement = new SubscriptionManagement(northApiClient);

```
//先设置好subDeviceData的第一个入参SubDeviceDataInDTO结构体
SubDeviceDataInDTO sddInDTO = new SubDeviceDataInDTO();
sddInDTO.setNotifyType("deviceDataChanged");
//需要根据实际情况修改回调的ip和端口
ddInDTO.setCallbackUrl("https://XXX.XXX.XXX.8099/v1.0.0/messageReceiver");
try {
    //调用订阅类实例subscriptionManagement提供的业务接口,如subDeviceData
    SubscriptionDTO subDTO = subscriptionManagement.subDeviceData(sddInDTO, null, accessToken);
    System.out.println(subDTO.toString());
} catch (NorthApiException e) {
    System.out.println(e.toString());
}
```

注册设备

//得到NorthApiClient实例后,再使用northApiClient得到设备管理类实例 DeviceManagement deviceManagement = new DeviceManagement(northApiClient);

```
//设置好注册设备接口的第一个入参RegDirectDeviceInDTO2结构体
RegDirectDeviceInDTO2 rddInDTO = new RegDirectDeviceInDTO2();
String nodeid = "86370303XXXXXX"; //this is a test imei
String verifyCode = nodeid;
rddInDTO.setNodeId(nodeid);
rddInDTO.setVerifyCode(verifyCode);
rddInDTO.setTimeout(timeout);
```

//调用设备管理类实例deviceManagement提供的业务接口,如regDirectDevice RegDirectDeviceOutDTO rddod = deviceManagement.regDirectDevice(rddInDTO, null, accessToken);

//从返回的结构体rddod中获取需要的参数,如deviceld String deviceld = rddod.getDeviceld();

回调接口实现及证书制作

回调接口实现

新建一个类并继承**PushMessageReceiver**,可以参考Demo中的 **PushMessageReceiverTest**类,需要接收哪一类消息就重写对应的方法,如: @Override

public void handleDeviceAdded(NotifyDeviceAddedDTO body) {
 System.out.println("deviceAdded ==> " + body);

//TODO deal with deviceAdded notification

}

接收到平台推送的消息后,开发者需要根据业务进行处理,但不建议进行复杂计算、 I/O操作或者可能长时间等待的动作,可以先写数据库,应用进入相应界面或者刷新界 面再从数据库取数据并进行数据处理。

回调路径已在SDK中设置好了,所以在订阅时要注意设置对应的回调地址。回调的IP地 址则是服务器的地址,需要是公网地址。Demo工程的回调端口配置在src\main \resource\application.properties中: #specify the port of the web application server.port=8099

回调证书制作

本章节以自签名证书为例。如果是使用商用证书,请直接向CA机构申请。

步骤1 打开windows命令行窗口,输入where java,找到jdk所在路径,进入jdk的bin路径。 where java

```
cd /d {jdk的bin路径}
```



步骤2 使用如下命令生成tomcat.keystore文件。

- 如果jdk的bin目录下已有tomcat.keystore,建议先将已有的tomcat.keystore移到 别的路径下。
- "您的名字与姓氏是什么"要输入应用服务器的IP或域名。
- <tomcat>的密钥口令要与密钥库口令设置一致(最后一步按回车即可), 输入的 密钥库口令要记住,后续配置会使用到。

keytool -genkey -v -alias tomcat -keyalg RSA -keystore tomcat.keystore -validity 36500



步骤3 将物联网平台提供的根证书ca.pem放到jdk的bin目录下,并使用如下命令将其加到 tomcat.keystore的信任证书链中。

注: 平台的测试根证书ca.pem可以在JAVA SDK包的cert目录下找到。

keytool -import -v -file ca.pem -alias iotplatform_ca -keystore tomcat.keystore

输入密钥库口令,查看导入的证书内容,确认无误后,输入y即可。



将物联网平台提供的根证书ca.pem加到tomcat.keystore的信任证书链后,由ca.pem签 发的子证书就能得到应用服务器的信任。

步骤4 将tomcat.keystore放到Demo工程目录下,例如: src\main\resources,打开src\main \resource\application.properties,添加如下配置。其中,server.ssl.key-store 为 tomcat.keystore 所在路径,server.ssl.key-store-password为密钥库口令。

#one-way authentication (server-auth) server.ssl.key-store=./src/main/resources/tomcat.keystore server.ssl.key-store-password=741852963.

步骤5 右键单击**PushMessageReceiverTest**,选择 "Run As > Java Application ",运行 Demo中的**PushMessageReceiverTest**类。

当有数据推送到应用服务器时,就会进入相应的回调函数中。

	77	((avannida the callback functions if needed otherwise you can delete them
	37 380	Woverside the callback functions if needed, otherwise, you can delete them.
	39	public woid handleDeviceAdded(NotifyDeviceAddedDTO_body) {
	40	System.out.println("deviceAdded ==> " + body):
	41	}
	42	
	43 0	@Override
4	44	<pre>public void handleBindDevice(NotifyBindDeviceDTO body) {</pre>
	45	<pre>System.out.println("bindDevice ==> " + body);</pre>
	46	}
	47	
	480	@Override
^	49	public void handleDeviceInfoChanged(NotifyDeviceInfoChangedDTO body) {
	4	
		1
R	Probl	= ems @ Javadoc 🗟 Declaration 🔗 Search 🥃 Console 🕱 🗧 🖉
.	Proble	コー ems @ Javadoc 隠 Declaration ※ Search 国 Console 窓 sageReceiverTest Llava Application () Program Files) Java) idk1 8 0 45\ire\bin\iavaw eve (2018年7月13日 下午715:09)
רי נויני	Probl	コ ems @ Javadoc 隠 Declaration 🥓 Search 📮 Console 窓 🗧 sageReceiverTest [Java Application] C:\Program Files\Java\jdk1.8.0_45\jre\bin\javaw.exe (2018年7月13日下午7:15:09)
ים ים: י	Proble shMess	ロート・ション・ション・ション・ション・ション・ション・ション・ション・ション・ション
ים: ים: (Problems	ロー・コー・コー・コー・コー・コー・コー・コー・コー・コー・コー・コー・コー・コー
ייים אייייייייייייייייייייייייייייייייי	Problems	ロート・ション・ション・ション・ション・ション・ション・ション・ション・ション・ション
Pu: / (Proble shMess ()) / _ () / _ /	U ems @ Javadoc 隠 Declaration <i>※</i> Search 回 Console 図 ■ 第 sageReceiverTest [Java Application] C:\Program Files\Java\jdk1.8.0_45\jre\bin\javaw.exe (2018年7月13日下午7:15:09)
) / ((\ = ·	Proble shMess 	u ems @ Javadoc 隐 Declaration
)u: / ((\ = :	Proble shMess () \ () \ 	
)) () = : de	Proble shMess ()\ ()\ 	ems @ Javadoc Declaration Search Console sageReceiverTest [Java Application] C:\Program Files\Java\jdk1.8.0_45\jre\bin\javaw.exe (2018年7月13日下午7:15:09) \

-----结束

回调证书导出

步骤1 使用浏览器打开回调地址https://server:8099/v1.0.0/messageReceiver,以Google为例,并查看证书。

server是应用服务器的地址(即本机地址),8099是在application.properties中配置的 端口。

▲ 不安全 https:// 1.0.0/m	lessageReceiver
冬日。 您与此网站之间建立的连接不安全 ————————————————————————————————————	
请勿在此网站上输入任何敏感信息(例如密码或信用卡 信息) 因为政夫者可能会盗取这些信息 乙酸洋库	
☑ 证书(无效)	
(目前使用了 0 个) Cookie (目前使用了 0 个) Cookie	
✿ 网站设置	

步骤2 系统将弹出证书窗口,选择"详细信息",单击"复制到文件"。

正书	>
常规 详细信息 证书路径	
	•
字段	值 _ _
画版本	V3
	UI ee od a9 -b-056854
	sha250n5k
□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□	
	2018年7月2日 16:57:50
三 到	2118年6月8日 16:57:50
	10 00 100 101 1 1
了解 <u>证书详细信息</u> 的更多信息	编辑属性 (B) 复制到文件 (C)
	确定

步骤3 单击"下一步",进入"导出文件格式"界面,选择"Base64编码",然后单击"下一步"。

证书导出向导	x
导出文件格式 可以用不同的文件格式导出证书。	
选择要使用的格式:	
💽 Base64 编码 X.509(.CER)(S)	
○ 加密消息语法标准 - PKCS #7 证书(.P7B)(C)	
🗖 如果可能,则数据包括证书路径中的所有证书 (I)	
○ 个人信息交换 - PKCS #12(.PFX)(P)	
🗖 如果可能,则数据包括证书路径中的所有证书 (U)	
🗖 如果导出成功,删除私钥 (K)	
□ 导出所有扩展属性 (A)	
Ĉ Microsoft 序列化证书存储(.SST)(T)	
了解证书文件格式的详细信息	
〈上一步(36) 下一步(37) > 取消	

步骤4 指定证书的保存路径,完成证书导出。

如果应用服务器最后部署到云上,可能会有多级证书,**建议在部署完成后再导出证** 书。此时需要将证书链**上面几级的证书——导出**。

 在"要导出的文件"界面,单击"浏览",选择一个路径,输入文件名,单击 "保存",回到证书导出向导,单击"下一步"。

证书导出向导	X
要导出的文件 指定要导出的文件名。	
文件名(F): C:\Vsers\\Desktop\mycert.cer	
< 上一步 (8) 下一步 (8) > 取消	j

2. 单击"完成",完成证书导出。

证书导出向导		×
	正在完成证书导出向导	
	您已成功地完成证书导出向导。	
	您已指定下列设置:	
	文件名	C:\Users\
	导出密钥	否
	数据包括证书路径中所有证书 文件格式	合 Rana64 偏码 V 50
		Daseo4 5HHHJ A. 30
	4	► F
	p	
	〈上一步(B) [3	完成 取消

步骤5 若存在多级证书,需一一导出。

1. 在"证书"窗口,选择"证书路径",查看多级证书,选中证书路径中的某个证书,单击"查看证书"。



- 系统将弹出证书窗口,选择"详细信息",然后重复上述导出证书的步骤,导出已选证书。
- **步骤6** 使用文本编辑器,将所有导出的证书以首尾相连的方式,合并为一个.pem格式的文件。该文件需要上传到物联网平台相应的应用下。
 - 24 5gaRQBi5+MHt39tBquCWIMnNZBU4gcmU7qKEKQsTb47bDN0lAtukixlE0kF6BWlK 25 WE9qyn6CaqsCqiUXObXbf+eEZSqVir2G316BFoMtEMze/aiCKm0oHw0LxOXnGiYZ 26 4fQRbxC1lfznQgUy286dUV4otp6F01vvpX1FQHKOtw5rDgb7MzVIcbidJ4vEZV8N hnacRHr21Vz2XTIIM6RUthq/aFzyQkqF0FSDX9HoLPKsEdao7WNq 27 28 ----END CERTIFICATE-----BEGIN CERTIFICATE---29 30 MIIFODCCBCCgAwIBAgIQUT+5dDhwtzRAQYOwkwaZ/zANBgkqhkiG9w0BAQsFADCB 31 yjELMAkGA1UEBhMCVVMxFzAVBgNVBAoTDlZlcmlTaWduLCBJbmMuMR8wHQYDVQQL ExZWZXJpU2lnbiBUcnVzdCBOZXR3b3JrMTowOAYDVQQLEzEoYykqMjAwNiBWZXJp 32 33 U2lnbiwqSW5jLiAtIEZvciBhdXRob3JpemVkIHVzZSBvbmx5MUUwQwYDVQQDEzxW 34 ZXJpU2lnbiBDbGFzcyAzIFB1YmxpYyBQcmltYXJ5IENlcnRpZmljYXRpb24gQXV0

----结束

回调证书上传

- 步骤1 登录开发中心,进入相关项目。
- 步骤2选择"应用 > 对接信息",单击"证书管理"。
- 步骤3 单击"添加",上传证书。

----结束

业务接口调用流程及注意事项

• 请按如下流程调用业务接口。



 Demo中使用的Profile如下图所示,只有一个Brightness服务,Brightness服务下 有一个brightness属性和一个PUT命令。在调用创建设备命令或设备服务调用等接 口时,如果不是使用以下Profile内容,请将相关服务、属性或者命令名称修改为 相应的名称。

服务列表	ŧ											+ #	版建服务
		服务名称			掤	<u>*</u>			最后修改时间		攝作		
~		Brightness							2018/07/16 16:26	:59			Î
ţ	属性										+ 1	所增属性	:
	⇔ brightness		属性类 int	型 取值范围 0~100	步长 •	单位 -		访问ł R	莫式	是否必选: <mark>></mark>	/	Î	
g	命令										+ 1	所增命令	
/	∧ ⇔ PUT											Î	
ł	命令下发字段										十 新増命令	下发字段	
	brightness	压 in	E性美型 nt	取值范围 0 ~ 100	步长 -	单位 -			是否必选: <mark>✓</mark>		1	Î	
1	向应字段												
						十新	(增响应字段						

• 创建新的Profile方法:

登录开发者中心,选择"产品 > 产品开发 > 添加 > 自定义产品",点击"自定义 产品",进入"设置产品信息"页面。填写"产品名称"、"型号"、"厂商 ID"、"所属行业"、"设备类型"、"接入应用层协议类型"等产品信息,点 击"创建"。然后点击"+新建服务"(根据设备功能添加属性和命令)最后点击 "保存"。

 修改设备信息接口使用到的字段值如"设备类型"、"厂商名"、"厂商ID"、 "设备型号"要与Profile定义的保持一致。 accessToken可以由SDK管理,也可由第三方应用自行管理,具体参考具体可参考 《JAVA SDK API参考文档》中"应用安全接入 > 定时刷新token"章节的说明。

SDK 独立运行测试

SDK包中提供了可独立运行的jar包,用于测试平台Restful接口。可独立运行测试的jar 包在testSDK目录下:

图 4-1 可独立运行的 jar 包

퉬 api-client-test_lib
📓 api-client-test.jar
📄 ca.jks
config.properties
📄 logback.xml
outgoing.CertwithKey.pkcs12
readMe_cn.txt
🚳 runMe.bat

步骤1 修改config.properties后再双击运行runMe.bat即可进行测试。

图 4-2 修改 config.properties

1	#please modify the value of platformIp/platformPort,
2	platformIp=1CI.iiI.iii.iii
3	platformPort=8743
4	appId=xo1D12,
5	secret=gPnTWCkkf12P8f4a
6	#the value of newCaFile and newClientCertFile should
7	newCaFile <mark>=</mark>
8	newCaPassword=
9	newClientCertFile=
10	newClientCertPassword =
11	#hostNameVerify default value is true, true means se
12	hostNameVerify=

- **步骤2** 如果使用商用证书,请直接将证书放在testSDK目录下面(证书名字不可以与ca.jks或者outgoing.CertwithKey.pkcs12相同),并在config.properties中配置证书名及密码;如果使用测试证书,则不需要修改config.properties中的证书信息。
- **步骤3**测试结果会在最前面输出:[y]表示测试通过;[x]则表示出错,请仔细查看该行的错误 提示或说明。
- 步骤4 运行jar包需要依赖JDK,请确认已安装JDK并设置了系统环境变量。

运行runMe.bat的结果如下:
	Begin test
	==> D: \0 ;
	[Y] getAuthToken() 1, get accesstoken successfully with inner certificates
	AuthOutDTO [accessToken=57a37baa92cbbfca3d763ffe8e9e9d73, tokenType=bearer, refreshToken=4ba44a
	[Y] refreshAuthToken<> succeeded, AuthRefreshOutDTO [accessToken=309c198bc768ca31eb83996936fba2
	3762bcf2]
	[Y] regDirectDevice() succeeded, DirectDeviceRegOutDTO [deviceId=ef72daa2-772d-4a9b-a2bc-71a48a
	[Y] modifyDeviceInfo<> succeeded
ļ	[Y] deleteDevice() succeeded

----结束

4.6.1.7.2.2 Python SDK 使用指南(联通用户专用)

非联通用户请查看设备接入服务。

本文以提供的应用侧Python SDK Demo为例说明如何使用Python SDK与物联网平台 对接。Demo以Python工程为例,以invokeapiTest下的每个类都包含了main方法,可 单独运行,旨在演示如何调用SDK接口。

开发环境要求

开发平 台	开发环境	配套要求	推荐的操作系统
loT	1. 开发工具:JetBrains PyCharm 2018.1.4 x64	Python 3.7	Windows7
	2. Python Project Interpreter: Python 3.7或Anaconda 3		

SDK包为纯Python的egg包,在使用上没有特殊限制,Python在3.7及以上版本即可。

导入 Demo 工程

步骤1 将Python SDK解压至本地Python安装路径的"Lib\site-packages"目录下,如,C: \Program Files\Python37\Lib\site-packages。

本地磁盘 (C:)	7 ▶ Lib ▶ site-pac	kages 🕨 OceanConr	ect_Python_SDK →	
工具(T) 帮助(H)				
共享 ▼ 新建文件夹				
名称	修改日期	类型	大小	
鷆 com 鷆 EGG-INFO	2019/8/9 11:24 2019/8/9 11:24	文件夹 文件夹		

- 步骤2 将Python SDK Demo解压到本地。
- 步骤3 打开PyCharm,选择"Open",再选择Demo解压后的路径,单击"OK"。
- 步骤4 配置Project Interpreter。
 - 1. 选择"File > Settings > Project: OceanConnect _ Python _SDK_Demo > Project Interpreter"。



 选择SDK所在的解释器,如,C:\Program Files\Python37\python.exe,单击 "Apply",最后单击"OK"即可正确配置工程所需要的SDK。

Q.↓ Appe Keyn > Edito Plugi	earance & Behavior nap or	Pro	roject: OceanConnect_Python_S > Project Interpreter Interpreter Interpreter Interpreter:	
> Appe Keym > Edito Plugi	earance & Behavior nap or	Pro	roject Interpreter: <no interpreter=""></no>	
Keym > Edito Plugi	nap or		9	~
> Edito Plugi	or		4	
Plugi			Package Version Latest versi	on
	jins			
> Versi	ion Control	ē		
~ Proje	ect: OceanConnect_Python_S	- 0		
Pre	roject Interpreter	n.		
Pro	roject Structure	li li		
▶ ∎ 🖭	Add Python Interpreter			E
> L	Virtualeny Environment	New environme	ent	
> 1	O Conda Environment	Location:	D:\PythonWorkspaces\IOT_Python_Demo\OceanConnect_Python_SDK_Demo\venv	<u> </u>
	System Interpreter	Base interpret	eter: C:\Program Files\Python37\python.exe	~
	Pineny Environment	Inherit alak	shal site-nackanes	
		Make avail	ilable to all projects	
		Existing environ	nment	
		Interpreter:	C:\Program Files\Python37\python.exe	~
	L	Make avail	include to all presidents	

步骤5 Python SDK Demo目录结构如下,Demo工程中已包含SDK库。

OceanConnect_Python_SDK_Demo
👻 🛅 com
🗸 🖿 huawei
🗸 🖿 iotplatform
> 🛅 client
> 🛅 constant
> 🛅 utils
💑initpy
💑initpy
💑initpy
> 🖿 venv
💑initpy
🛃 setup.py

----结束

配置平台信息及证书

配置文件(**application.ini**)以及证书都分别放置在工作路径"d:/python_sdk/"下面的**cert**和**resources**文件夹中,用户也可自行设置工作路径。

平台IP、端口、appId和密码都是从配置文件中读取的,因此,当这些信息发生变化时,只要修改配置文件,不用修改应用服务器的代码。

[CLIENT_INFO] platformIp = 100.100.100 platformPort = 8743 appId = FT8EjQ8O****VbW60Qb8xvga secret = TtuK4Paf*****yAv66himUk8a

```
在"Constant.py"中,读取证书和CLIENT_INFO。
```

```
class Constant(object):
# 工作路径
   workPath = os.path.join('d:/python_sdk/')
# 读取证书
def readCertificate(self):
  certFilePath = os.path.join(Constant.workPath, 'cert/client.crt')
  certFilePath2 = os.path.join(Constant.workPath, 'cert/client.key')
  cert = (certFilePath, certFilePath2)
return cert
# 读取配置文件
def readConfFile(self):
  configFilePath = os.path.join(Constant.workPath, 'resources/application.ini')
  cf.read(configFilePath)
  platformIp = cf.get("CLIENT_INFO", "platformIp")
platformPort = cf.get("CLIENT_INFO", "platformPort")
  appId = cf.get("CLIENT_INFO", "appId")
secret = cf.get("CLIENT_INFO", "secret")
return platformIp, platformPort, appId, secret
# 设置平台对接信息
def clientInfo(self):
  clientInfo = ClientInfo()
  clientInfo.setPlatformIp((Constant().readConfFile())[0])
  clientInfo.setPlatformPort((Constant().readConfFile())[1])
```

clientInfo.setAppId((Constant().readConfFile())[2])

```
clientInfo.setSecret((Constant().readConfFile())[3])
clientInfo = DictUtil.dto2dict(clientInfo)
return clientInfo
```

业务接口调用方法

设置好**Constant.py**后才能调用其他业务接口。所有业务接口的测试都在 "invokeapiTest"中。以如下几个接口为例说明如何调用业务接口:

关于哪些参数需要设置,请查看《应用侧Python SDK API参考》对于可选参数,如果 业务不需要,可以不设置。

配置日志

```
# 在每个业务接口中都配置了日志,具体日志实现可参考LogUtil.py:
# 开发者可通过修改入参来控制日志的输出,以下三个参数都可自行设置;
# logPath:日志的输出路径,默认在工作路径d:/python_sdk/log/
# level:日志等级,默认DEBUG(最低等级,即大于等于该等级的日志级别都会输出),如果level=0没有日志输出
# logFilename:日志名称
def setLogConfig(self,
```

```
logPath=os.path.join(Constant.workPath, 'log/'),
level=logging.DEBUG,
logFilename="python_sdk.log"):
```

鉴权

```
# 在AuthenticationTest.py中对鉴权和刷新token接口进行测试:
if __name__ == "__main__":
  from com.huawei.iotplatform.utils.LogUtil import Log
  # 实例化Authentication()
  authentication = Authentication()
  # 调用鉴权类实例authentication提供的业务接口,如getAuthToken
  ag = authentication.getAuthToken(Constant().clientInfo())
  print("====== get access token ======")
```

```
print("result:", ag + "\n")
```

```
# 从返回的ag中获取需要的参数,如accessToken
authOutDTO = AuthOutDTO()
authOutDTO.setAccessToken(json.loads(ag)['accessToken'])
print("token", authOutDTO.getAccessToken())
```

订阅

```
# 在SubscriptionManagementTest.py中对订阅订阅平台业务数据接口进行测试:
class SubscriptionManagementTest(object):
    def subDeviceBusinessData(self):
        sdbdInDTO = SubDeviceBusinessDataInDTO()
        sdbdInDTO.notifyType = "bindDevice"
        sdbdInDTO.callbackUrl = "https://XXX.XXX.XXX.XXX:443/callbackurltest"
        return sdbdInDTO
if __name__ == "__main__":
```

from com.huawei.iotplatform.utils.LogUtil import Log

```
# 实例化
smTest = SubscriptionManagementTest()
subscriptionManagement = SubscriptionManagement()
```

```
# get accessToken at first
result = Authentication().getAuthToken(Constant().clientInfo())
authOutDTO = AuthOutDTO()
authOutDTO.setAccessToken(json.loads(result)['accessToken'])
accessToken = authOutDTO.getAccessToken()
```

```
# 调用订阅类实例subscriptionManagement提供的业务接口,如subDeviceData
  ss = subscriptionManagement.subDeviceBusinessData(smTest.subDeviceBusinessData(), accessToken)
  print("===== subscribe to device business data notification ======")
  print("result:", ss + "\n")
注册设备
# 在DeviceManagementTest.py中对注册设备接口进行测试:
class DeviceManagementTest(object):
  def regDirectDeviceInfo(self):
    rddInDto = RegDirectDeviceInDTO()
     rddInDto.nodeId = "AAA" + str(random.randint(0, 9999))
    return rddInDto
if name == " main ":
  from com.huawei.iotplatform.utils.LogUtil import Log
  # 实例化
  dmTest = DeviceManagementTest()
  deviceManagement = DeviceManagement()
  # get accessToken at first
  result = Authentication().getAuthToken(Constant().clientInfo())
  authOutDTO = AuthOutDTO()
  authOutDTO.setAccessToken(json.loads(result)['accessToken'])
  accessToken = authOutDTO.getAccessToken()
  # 调用设备管理类实例deviceManagement提供的业务接口,如regDirectDevice
  dr = deviceManagement.regDirectDevice(dmTest.regDirectDeviceInfo(), None, accessToken)
  print("===== register a new device ======")
  print("result:", dr + "\n")
# 从返回的dr中获取需要的参数,如deviceld
rddod = RegDirectDeviceOutDTO()
rddod.setDeviceId(json.loads(dr)['deviceId'])
```

回调接口实现及证书制作

回调接口实现

deviceId = rddod.getDeviceId()
print("deviceId==", deviceId + "\n")

回调接口的启动在 "PushMessageReceiverTest.py "中,配置请看以下说明。 if __name__ == '__main_':

回调地址和端口

callbackUrl = "XXX.XXX.XXX.XXX" port = 8099

这里使用了pyOpenSSL自带证书,注意ssl_context的值必须是adhoc app.run(host=callbackUrl, port=port, ssl_context='adhoc')

使用自己配置的证书, ssl_context的值配置如下: 此处放置的是第8章节步骤3生成的server证书 app.run(host=callbackUrl, port=port, ssl_context=('d:/python_sdk/cert/server.crt', 'd:/python_sdk/cert/ server.key'))

具体业务实现在"PushMessageReceiver.py"中的"PushMessageReceiver"类中, 可以参考Demo中的"PushMessageReceiver"类,需要接收哪一类消息就改写对应的 方法,如:

```
class PushMessageReceiver(object):
    def handleDeviceAdded(self):
        print("deviceAdded ==> ", request.json)
#TODO deal with deviceAdded notification
        pass
```

接收到平台推送的消息后,开发者需要根据业务进行处理,但不建议进行复杂计算、 I/O操作或者可能长时间等待的动作,可以先写数据库,应用进入相应界面或者刷新界 面再从数据库取数据并进行数据处理。

回调路径已在SDK中设置好了,所以在订阅时要注意设置对应的回调地址。回调的IP地 址则是服务器的地址,需要是公网地址,端口自行配置。

回调证书制作

本章节以自签名证书为例。如果是使用商用证书,请直接向CA机构申请。

注: 每个步骤的**创建证书请求**中的配置,请自行设置。"Common Name (e.g. server FQDN or YOUR name) []:"要输入应用服务器的IP或域名。

步骤1 打开windows命令行窗口,输入where openssl,查询是否安装openssl。 where openssl



步骤2 创建测试CA证书。

- 1. 创建私钥。 openssl genrsa -out ca-key.pem 1024
- 2. 创建证书请求。 openssl req -new -out ca-req.csr -key ca-key.pem
- 3. 自签署证书。 openssl x509 -req -in ca-req.csr -out ca-cert.pem -signkey ca-key.pem -days 3650

root%52X1004480858./usr/makeCert#[openss] genrsa -out ca-key.pem 1024
denerating now private key, 1024 pit 10m wolding
e is obs// (NAINNI) root#SZX1000480858:/wsr/makeCert# openssl req -new -out ca-req.csr -key ca-key.pem You are about to be asked to enter information that will be incorporated into your certificate request. What you are about to enter is what is called a Distinguished Name or a DN. There are quite a few fields but you can leave some blank For some fields there will be a default value.
If you enter '.', the field will be left blank.
Country Name (2 letter code) [AU]:cn State or Frowince Name (full name) [Some-State]:gd Locality Name (eg. city) []:sz Organization Nome (eg. company) [Internet Widgits Pty Ltd]:test Organizational Unit Name (eg. section) []:test Comon Name (e.g. server FODN or YOUR name) []:test Email Address []:test
Please enter the following 'extra' attributes to be sent with your certificate request A challenge password []:1234 An optional company name []:test requestributed (serve) attribute forement v503 area aim carred cer anut carcent per asignbeu carkeu per adaug 3650
Signature ok
subject*/C*cn/ST*gd/L*sz/O*test/OU*test/CN*test/email&ddress*test Getting Private key
root@SZX1000480858:/usr/makeCert#

步骤3 生成server证书。

1. 创建私钥。

openssl genrsa -out server.key 1024

- 2. 创建证书请求。 openssl req -new -out server-req.csr -key server.key
- 3. 自签署证书。

openssl x509 -req -in server-req.csr -out server.cert -signkey server.key -CA ca-cert.pem -CAkey ca-key.pem -CAcreateserial -days 3650



步骤4 生成client证书。

- 1. 创建私钥。 openssl genrsa -out client.key 1024
- 2. 创建证书请求。 openssl req -new -out client-req.csr -key client.key
- 3. 自签署证书。

openssl x509 -req -in client-req.csr -out client.cert -signkey client.key -CA ca-cert.pem -CAkey cakey.pem -CAcreateserial -days 3650

root#SZX100480858:/usr/makeCert# openss1 genrsa -out client key 1024 Generating KSA private key, 1024 bit long madelus
e is 65537 (%x0001) root#SZX100480858 usr/makeCert# openesl req -new -out client-req.csr -key client.key Tou are about to be asked to ented information that will be incorporated into your cortificate request. That you are about to enter is what is called a Distinguished Name or a DN. There are guite a few fields but you can leave beceb blank For more fields there will be a default value. If you enter ', the field will be left blank.
Country Name (2 letter code) [AU]:cn State or Province Name (full mame) [Some-State]:gd Locality Name (eg. civ)[]:sz Dyganization Name (eg. company) [Internet Widgits Pty Ltd]:test Common Name (eg. server FOLW or YOUR name) []:test Easil Address []:test Easil Address []:test
Please enter the following 'extra' attributes to be sent with your certificate request A challenge password [1]] A challenge password [1]] A challenge variable (1)] A challenge variable (1)] A challenge variable(2) Fill -dear 3559
SiyunNur uk subject×C+cm/ST=gd/1=sz/O+test/OU+test/CN+test/emailAddress+test Getting Pirvate key Getting CA Frivate Key root#SZID00480858 /usr/amkeCert# ■

步骤5 在Demo工程中,已经配置好生成的server证书。

步骤6 单击"启动"**PushMessageReceiverTest**,选择"Run""PushMessageReceiverTest"。"。

当有数据推送到应用服务器时,就会进入相应的回调函数中。



Rur	u 🦷	PushMessageReceiverTest ×
C	1	100.115.187.142 [02/Nov/2018 18:30:03] "POST /v1.0.0/messageReceiver/cmd HITP/1.1" 200 -
	+	deviceAdded ==> {'notifyType': 'deviceAdded', 'deviceId': '100000000000000000000000000000000000
ш	s	

----结束

回调证书导出

步骤1 使用浏览器打开回调地址https://server:8099/v1.0.0/messageReceiver,以Google为例,并查看证书。

server是应用服务器的地址(即本机地址),8099是在application.properties中配置的端口。

▲ 不安全 https://1000/m	essageReceiver
※ 您与此网站之间建立的连接不安全	
请勿在此网站上输入任何敏感信息(例如密码或信用卡	
信息),因为攻击者可能会盗取这些信息。 了解详情	
证书(无效)	
🍪 (目前使用了 0 个)Cookie	
🔹 网站设置	

步骤2 系统将弹出证书窗口,选择"详细信息",单击"复制到文件"。

征书	×
常规 详细信息 证书路径	
並示 (5): < <u><</u> 所有> 字段 ■ 版本	
展 旅空 開 序列号 目 签名算法 日 签名哈希算法	01 ee 8d a9 sha256RSA sha256
一一一一一一一一一一一一一一一一一一一一一一一一一一一一一一一一一一一一一	2018年7月2日 16:57:50 2118年6月8日 16:57:50
了解 <u>证书详细信息</u> 的更多信息	编辑属性 (E) 复制到文件 (C)
	确定

步骤3 单击"下一步",进入"导出文件格式"界面,选择"Base64编码",然后单击"下一步"。

选	择要使用的格式:
	◎ DER 编码二进制 X.509(.CER)(D)
	● Base64 编码 X.509(.CER)(S)
	○ 加密消息语法标准 - PKCS #7 证书(.P7B)(C)
	🔲 如果可能,则数据包括证书路径中的所有证书 (I)
	○ 个人信息交换 - PKCS #12(.PFX)(P)
	🔲 如果可能,则数据包括证书路径中的所有证书 (V)
	□ 如果导出成功,删除私钥 (K)
	□ 导出所有扩展属性 (A)
	C Microsoft 序列化证书存储(.SST)(T)
了解正	其文件格式的详细信自

步骤4 指定证书的保存路径,完成证书导出。

如果应用服务器最后部署到云上,可能会有多级证书,**建议在部署完成后再导出证** 书。此时需要将证书链**上面几级的证书——导出**。

 在"要导出的文件"界面,单击"浏览",选择一个路径,输入文件名,单击 "保存",回到证书导出向导,单击"下一步"。

证书导出向导	X
要导出的文件 指定要导出的文件名。	
文件名 @): C:\Vsers\⊷】\Desktop\mycert.cer	
< 上一步 (B) 下一步 (B) > 取消	Í.

2. 单击"完成",完成证书导出。

证书导出向导			×
	正在完成证书导出向导		
	您已成功地完成证书导出向导。		
	您已指定下列设置:		
	文件名	C:\Users\	
A Street	导出密钥 数据包括证书路径中所有证书	告 否	
	文件格式 	Base64 编码 X.50	
	•		
	〈上一步 (3)	完成 取消	

步骤5 若存在多级证书,需一一导出。

1. 在"证书"窗口,选择"证书路径",查看多级证书,选中证书路径中的某个证书,单击"查看证书"。



- 系统将弹出证书窗口,选择"详细信息",然后重复上述导出证书的步骤,导出已选证书。
- **步骤6** 使用文本编辑器,将所有导出的证书以首尾相连的方式,合并为一个.pem格式的文件。该文件需要上传到物联网平台相应的应用下。

24	5gaRQBi5+MHt39tBguCWIMnNZBU4gcmU7gKEKQsTb47bDN01Atukix1E0kF6BW1K
25	WE9gyn6CagsCqiUXObXbf+eEZSqVir2G316BFoMtEMze/aiCKm0oHw0LxOXnGiYZ
26	4fQRbxC1lfznQgUy286dUV4otp6F01vvpX1FQHKOtw5rDgb7MzVIcbidJ4vEZV8N
27	hnacRHr2lVz2XTIIM6RUthg/aFzyQkqF0FSDX9HoLPKsEdao7WNq
28	END CERTIFICATE
29	BEGIN CERTIFICATE
30	MIIFODCCBCCgAwIBAgIQUT+5dDhwtzRAQY0wkwaZ/zANBgkqhkiG9w0BAQsFADCB
31	${\tt yjELMAkGA1UEBhMCVVMxFzAVBgNVBAoTD1Z1cmlTaWduLCBJbmMuMR8wHQYDVQQL}$
32	ExZWZXJpU2lnbiBUcnVzdCBOZXR3b3JrMTowOAYDVQQLEzEoYykgMjAwNiBWZXJp
33	U2lnbiwgSW5jLiAtIEZvciBhdXRob3JpemVkIHVzZSBvbmx5MUUwQwYDVQQDEzxW
34	ZXJpU2lnbiBDbGFzcyAzIFB1YmxpYyBQcmltYXJ5IENlcnRpZmljYXRpb24gQXV0

----结束

回调证书上传

- 步骤1 登录开发中心,进入相关项目。
- 步骤2选择"应用 > 对接信息",单击"证书管理"。
- 步骤3 单击"添加",上传证书。

----结束

业务接口调用流程及注意事项

• 请按如下流程调用业务接口。



 Demo中使用的Profile如下图所示,只有一个Brightness服务,Brightness服务下 有一个brightness属性和一个PUT命令。在调用创建设备命令或设备服务调用等接 口时,如果不是使用以下Profile内容,请将相关服务、属性或者命令名称修改为 相应的名称。

服务列表							十 新建服务
服务名	称	描述		最后修改时间		播作	
Bright	nëss			2018/07/16 16:26	:59		Î
属性						十 新増	属性
+> brightness	属性类型 取值范围 int 0~100	步长 ^其	柏立	访问模式 R	是否必选: ▼	1	î
命令						十 新増	命令
∧ ↔ PUT						1	î
命令下发字段						十 新增命令下发	;字段
brightness	庫性美型 取值范围 int 0~100	步长 i -	单 位 -	是否必选: ▼		/ î	
响应字段							
			十 新增响应字段				

• 创建新的Profile方法:

登录开发者中心,选择"产品 > 产品开发 > 添加 > 自定义产品",点击"自定义 产品",进入"设置产品信息"页面。填写"产品名称"、"型号"、"厂商 ID"、"所属行业"、"设备类型"、"接入应用层协议类型"等产品信息,点 击"创建"。然后点击"+新建服务"(根据设备功能添加属性和命令)最后点击 "保存"。

 修改设备信息接口使用到的字段值如"设备类型"、"厂商名"、"厂商ID"、 "设备型号"要与Profile定义的保持一致。 accessToken可以由SDK管理,也可由第三方应用自行管理,具体信息可参考 《Python SDK API参考文档》中"应用安全接入 > 定时刷新token"章节的说 明。

5 调测证书制作(联通用户专用)

非联通用户请查看设备接入服务。

调测证书,又叫做自签名证书,用于客户端通过HTTPS访问服务端时进行安全认证。 在物联网平台的使用中,可用于物联网平台向应用服务器采用HTTPS协议推送数据 时,物联网平台认证应用服务器的合法性。本文以Windows环境为例,介绍通过 Openssl工具制作调测证书的方法,生成的证书为PEM编码格式的证书,后缀为.cer。

常见的证书存储格式如下表所示。

存储格式	说明
DER	二进制编码,后缀名.der/.cer/.crt
PEM	BASE 64编码,后缀名 .pem/.cer/.crt
JKS	Java的证书存储格式,后缀名 .jks

🛄 说明

自签名证书仅用于调测阶段,在商用时,您需要向知名CA机构申请证书,否则可能会带来安全 风险。

- **步骤1** 在浏览器中访问https://slproweb.com/products/Win32OpenSSL.html,下载并进行安装OpenSSL工具。
- 步骤2 以管理员身份运行cmd命令行窗口。
- 步骤3 执行cd c:\openssl\bin(请替换为openssl实际安装路径),进入openssl命令视图。
- 步骤4 执行如下命令生成CA根证书私钥文件ca_private.key。 openssl genrsa -passout pass:123456 -aes256 -out ca_private.key 2048
 - aes256: 代表加密算法。
 - passout pass: 代表私钥密码。
 - 2048:代表密钥长度。
- 步骤5 执行如下命令使用CA根证书私钥文件生成csr文件ca.csr,用于步骤6生成CA根证书。 openssl req -passin pass:123456 -new -key ca_private.key -out ca.csr -subj "/C=CN/ST=GD/L=SZ/O=Huawei/ OU=IoT/CN=CA"

如下信息您可以根据实际情况进行修改。

- C: 代表国家,填写CN。
- ST: 地区, 如GD。
- L: 城市, 如SZ。
- O: 组织,如Huawei。
- OU: 组织单位, 如IoT。
- CN:Common Name,填写为CA的组织名,如CA。

步骤6 执行如下命令生成CA根证书ca.cer。

openssl x509 -req -passin pass:123456 -in ca.csr -out ca.cer -signkey ca_private.key -CAcreateserial -days 3650

如下信息您可以根据实际情况进行修改。

- passin pass:必须与步骤4中设置的私钥密码保持一致。
- days: 代表证书有效期。
- 步骤7 执行如下命令生成应用服务器端私钥文件。

openssl genrsa -passout pass:123456 -aes256 -out server_private.key 2048

步骤8 执行如下命令生成应用服务器端csr文件,用于生成服务端证书。

openssl req -passin pass:123456 -new -key server_private.key -out server.csr -subj "/C=CN/ST=GD/L=SZ/ O=Huawei/OU=IoT/CN=appserver.iot.com"

如下信息您可以根据实际情况进行修改。

- C: 代表国家,填写CN。
- ST: 地区,如GD。
- L: 城市, 如SZ。
- O: 组织,如Huawei。
- OU:组织单位,如IoT。
- CN:Common Name,一般填写为应用服务器的域名或IP。
- **步骤9** 通过CA私钥文件ca_private.key对服务端csr文件server.csr进行签名,生成服务端证书 文件server.cer。

openssl x509 -req -passin pass:123456 -in server.csr -out server.cer -sha256 -CA ca.cer -CAkey ca_private.key -CAserial ca.srl -CAcreateserial -days 3650

步骤10 (可选)如果您需要.crt/.pem后缀的证书,可以根据如下命令进行转换。下面将以 server.cer转为为server.crt为例进行说明,需要转换ca.cer证书时,请将命令中的server 替换为ca。

openssl x509 -inform PEM -in server.cer -out server.crt

步骤11 在openssl安装目录的bin文件夹下,获取生成的CA证书(ca.cer/ca.crt/ca.pem)、应用服务器证书(server.cer/server.crt/server.pem)和私钥文件 (server_private.key)。其中CA证书用于加载到物联网平台,应用服务器证书和私钥文件用于加载到应用服务器。

----结束

加载证书

物联网平台采用HTTPS协议向应用服务器推送消息时,需要在物联网平台上加载CA证书,下面将详细介绍在开发中心上传证书的方法。在设备管理服务控制台上加载证书请参考加载推送证书。

步骤1 登录开发中心控制台,进入到开发中心。

步骤2 选择"应用 > 对接信息",在"推送证书"区域,点击"证书管理"。

▲ 项目概览	基本信息		
 ● 产品 	对接信息详情		
₽ 应用 ▲	主要信息。		应用安全
对接信息	应用名称:	应用ID	応用密钥 の
订阅调试	Test_App001	BG49xVd8BEHAdOTdrbZAYdNxAhQa	重置察钥
应用调试	创建时间	应用描述	推送证书 ⑦
应用在线开发	2019/09/04 19:39:17 GMT+08:00	请填写应用描述信息	尚无推送证书 证书管理

步骤3 系统弹出"CA证书"窗口,检查相应的CA证书是否已上传,如未上传,点击"添加"。

CA证非	5 (该应用尚未	≂上传应用证书,	订阅功能将无法	使用)			×	
							+添加	
	CA证书	域名/IP与端口	lb昵称	是否检查 CNAME	CNAME			
				暂无数据				

步骤4 系统弹出"上传证书"窗口,选择证书文件,并填写"域名/IP与端口",其它信息不用修改。点击"上传"。

*上传证书文件				
文件大小不超过	±1M,且必须为pei	m文件	Ę	法选择文件
·域名/IP与端口				
请输入域名/	P与端口			
My api.ct10649.0 輸入合法的域名/I	com:9001或127.0. P及端口	1.1:8080		
Mg: api.ct10649.(動入合法的域名/II ™b昵称 default	com:9001或127.0. P及端口	1.1:8080		
bstab/ (costic) (例如: api.ct10649.(情輸入合法的域名/I *Ib昵称 default	com:9001或127.0. P及端口	1.1:8080		
Mgu: api.ct10649.(薄输入合法的域名/I *Ib昵称 default ≧否检查CNAME @	om:9001或127.0. P及端口	1.1:8080		

----结束

6 自助测试(联通用户专用)

非联通用户请查看设备接入服务。

自助测试提供了端到端的测试用例,帮助开发者自助完成产品的基础能力测试,如数据上报、命令下发等。旨在通过物联网检测技术帮助开发者发现自身产品中存在的缺陷或问题,缩短产品上市时间。测试完成后,开发中心将生成测试报告,用于进行产品发布认证。当开发者已完成产品profile定义和编解码插件开发,并且部署了编解码插件后,可以进行自助测试。

- 步骤1 在产品开发空间,单击"发起自助测试"。
- **步骤2**进入"选择测试用例"界面,开发者可以自行选择需要测试的用例,系统会自动检查 已选测试用例是否满足测试能力并返回检查结果。
 - 如果所有已选测试用例检查通过,则单击"下一步"继续下一阶段测试。
 - 如果有测试用例检查未通过,则单击该测试用例的"信息缺失",根据提示信息 进行修改。

注:产品通过测试的用例越多,产品发布到产品中心的审核通过率就越高。建议软/固 件升级测试任选其一,其余测试用例全选。

终止则试,返回	开发 4	1 选择测试用例		
选择测试用例			Ri	新 下—步
<mark>ீ</mark> முக				
~	设备绑定测试。公别	测试真实设备在良好的网络下接入IoT云平台的能力		Mit 🔵
~	数据上报测试	测试设备是否具备数据上报的能力		演試 🔵
\sim	无结参数上报测试	测试设备是否具备上报堆测数据的能力		測試
\sim	控制命令下发测试	測试设备是否具备接收IoT平台下发命令的能力		Mit 🔵
~	命令下发明应测试	测试设备在接收到IoT平台命令后,是否具备主动上报执行结果的解	8 5	演試 🔵
~	软件升级测试	测试设备是否具备软件升级的能力		測試
\sim	圖件升級測试 💿 信息缺失	測试设督是否具备固件升级的能力		Mit 🔵
	Profile - OM層性	您还来打开圆件升级服务,请补充		修改
參应用			有应用	无应用
~	应用订间事件测试	测试应用装置否可以调用IoT平台北向API接口并订阅平台设备数据	变化通知事件	● 測試
\sim	应用接收推送数据测试	測试应用號是否可以正确接收10T平台推送设备数据		🔵 Mit

步骤3根据测试用例说明,依次完成自助测试。完成测试后,可以预览测试报告或申请发布 产品。

----结束

设备绑定测试

设备绑定测试用于测试设备接入物联网平台的能力,包括设备注册和设备上线两个步骤。

🛄 说明

设备绑定测试是进行其他测试的前提条件,如果测试失败,则无法进行其他测试。

- 步骤1 在设备绑定测试界面,单击"下一步",进入注册设备界面。
- **步骤2**根据向导进入测试界面,选择"安全模式",并填写"设备标识"和"模组名称", 单击"下一步"。
 - 如果安全模式选择"加密",还需要填写"PSK"。
 - 如果不是使用模组进行测试,则"模组名称"填写无。

*安全模式:)加速	密 💿 非加密		
*设备标识: 11	11111111		
*模组名称 无			

步骤3 根据向导操作真实设备接入物联网平台进行绑定,查看测试用例执行结果。

- 测试成功,单击"下一步"进入下一阶段测试。
- 测试失败,排查并处理问题后,单击"重新测试"重测测试用例。

设备标识: 111111111 设备1d: 1e029ae8-1306-48f6-b07e-cec62ae7c70f 设备上线时间: 2019-06-22 16:48:27 用例完成时间: 2019-06-22 16:48:28 日志内容: [CIG]Bind request have reached platform. Bind request have reached platform, IMEI = 1111111 1, coapMID = 5890, coapToken = [56, -15, 89, -36], sourceAddress = 10.61.93.131, sourcePort = 50006		设备上线成功
设备ld: 1e029ae8-1306-48f6-b07e-cec62ae7c70f 设备上线时间: 2019-06-22 16:48:27 用例完成时间: 2019-06-22 16:48:28 日志内容: [CIG]Bind request have reached platform. Bind request have reached platform, IMEI = 1111111 11, coapMID = 5890, coapToken = [56, -15, 89, -36], sourceAddress = 10.61.93.131, sourcePort = 50006	设备标识	111111111
设备上线时间: 2019-06-22 16:48:27 用例完成时间: 2019-06-22 16:48:28 日志内容: [CIG]Bind request have reached platform. Bind request have reached platform, IMEI = 1111111 11, coapMID = 5890, coapToken = [56, -15, 89, -36], sourceAddress = 10.61.93.131, sourcePort = 50006	设备ld	1e029ae8-1306-48f6-b07e-cec62ae7c70f
用例完成时间: 2019-06-22 16:48:28 日志内容: [CIG]Bind request have reached platform. Bind request have reached platform, IMEI = 11111111 11, coapMID = 5890, coapToken = [56, -15, 89, -36], sourceAddress = 10.61.93.131, sourcePort = 50006	设备上线时间	2019-06-22 16:48:27
日志内容: [CIG]Bind request have reached platform. Bind request have reached platform, IMEI = 11111111 11, coapMID = 5890, coapToken = [56, -15, 89, -36], sourceAddress = 10.61.93.131, sourcePort = 50006	用例完成时间	2019-06-22 16:48:28
	日志内容	[CIG]Bind request have reached platform. Bind request have reached platform, IMEI = 1111111 11, coapMID = 5890, coapToken = [56, -15, 89, -36], sourceAddress = 10.61.93.131, sourcePort = 50006

----结束

文档版本 15 (2022-06-14)

数据上报测试

数据上报测试用于测试设备上报数据的能力,目的是验证Profile中定义的属性字段是 否正确。如果物联网平台与设备交互的"数据格式"为二进制码流,还会验证编解码 插件与Profile的映射关系是否正确。

- **步骤1** 在数据上报测试界面,单击"下一步"开始测试。
- **步骤2** 根据向导进入测试界面,操作真实设备上报Profile中定义的属性数据,如果设备的所有属性数据都已上报完成,可以点击"终止测试"完成数据上报测试并查看测试用例执行结果。
 - 测试成功,单击"下一步"进入下一阶段测试。
 - 测试失败,排查并处理问题后,单击"重新测试"重测测试用例。

平台会检测所有上报成功的属性数据,并记录到测试报告中,重复上报的属性只会记录一次。

1、设备上报数据	> 2、查看上报结果		
	上报数据成功		
上报的数据:	已上报的数据: {"toggle":35}		
	上报时间: 2019-06-22 17:02:50		
未上报的属性:	luminance swVersion SNR fwVersion CellID		
		重新测试	下一步

----结束

无线参数上报测试

无线参数上报测试用于测试设备上报无线参数属性数据的能力,包括信号强度、覆盖 等级、信噪比和小区ID。

如果测试此用例,需要在Profile中定义如下无线参数属性,并在编解码插件中建立对 应的映射关系。

参数	类型	描述
RSRP/rsrp/ signalStrength/ SignalPower	int	信号强度,取值范围-140~-40。
ECL/signalECL	int	覆盖等级,取值范围0~2。
SNR/snr/SINR/sinr/ signalSNR	int	信噪比,取值范围-20~30。
CellID/cellId	int	小区ID,取值范围0~2147483647。

步骤1 在无线参数上报测试界面,单击"下一步"开始测试。

- **步骤2**根据向导进入测试界面,操作真实设备上报Profile中定义的无线参数属性数据,查看测试用例执行结果。
 - 测试成功,单击"下一步"进入下一阶段测试。
 - 测试失败,排查并处理问题后,单击"重新测试"重测测试用例。

🗋 说明

上报的无线参数值必须在数据范围内才算有效数据。

Ⅰ、设备上报无线	参数数据 > 2、查看上报结果
上报的数据:	已上版的数据: {"SignalPower":-86,"ECL":0,"SNR":3,"CellID":11171992}
	上报时间: 2019-06-24 11:25:11
未上报的属性:	空
用例完成时间:	2019-06-24 11:25:44
日志内容:	[CIG]Cig received message from device. IMEI = 1111111111, reqMID = 23089, coapToken = [31, -71, 103, -3, 4, -33, -64, 1], rspMID = 22758, sourceAddress = 10.61.93.131, sourcePort = 56759
	重新测试 下一步

----结束

控制命令下发测试

控制命令下发测试用于测试设备接收和处理控制命令的能力,目的是验证Profile中定 义的命令字段是否正确。如果平台与设备交互的"数据格式"为二进制码流,还会验 证编解码插件与Profile的映射关系是否正确。由于下发的命令是立即下发模式,设备 需要在线。

如果使用业务应用进行测试,还会测试业务应用是否正确调用物联网平台"创建设备 命令"接口给设备下发命令的能力。

- 步骤1 在控制命令下发测试界面,单击"下一步"开始测试。
- 步骤2 根据向导进入测试界面,物联网平台会根据Profile的定义向设备下发一条命令。在真实设备应答后,查看测试用例执行结果。

如果是使用业务应用接入物联网平台,则操作业务应用向设备下发一条命令,在真实 设备应答后,进入"上传应用下发命令截图"界面,单击界面中的"+",上传业务应 用命令下发成功的截图。此图作为业务应用正确调用物理网平台"创建设备命令"接 口的凭证。

- 测试成功,单击"下一步"进入下一阶段测试。
- 测试失败,排查并处理问题后,单击"重新测试"重测测试用例。

1、loT平台下发台	▶ 2、查看设备接收结果
	命令下发成功
下发的命令:	命令内容: {"led":"OFF"}
	命令状态: 已送达
用例完成时间:	2019-06-24 11:55:57
日志内容:	[CIG]Cig received the command from cmdh. CIG get command from cmdh successfully, comma ndId = cdf6ecbfe5aa4b9884e672bc16618c1b, callbackUrl = null, serviceId = LED [CIG]Cig get protocol interpreter successfully. Have found Plugin, ready to encode, Manufacture rId = b651245873b54e06827afce9b9882e4e, Model = StreetLight01, Mid = 1, serviceId = LED, c md = Strt Log nerve = [CIG] ¹¹ CIET] between = 0
	重新测试下一步

----结束

命令下发响应测试

命令下发响应测试用于测试设备在接收物联网平台命令后主动上报执行结果的能力。 当Profile中定义了"命令下发响应字段",即设备需要返回命令执行结果时,开发者 才需要进行命令下发响应测试。由于下发的命令是立即下发模式,设备需要在线。

- 步骤1 在命令下发响应测试界面,单击"下一步"开始测试。
- **步骤2**根据向导进入测试界面,物联网平台会根据Profile的定义向设备下发一条命令,如果 是业务应用接入物联网平台,则操作业务应用向设备下发一条命令。

如果真实设备支持自动返回命令执行结果,则在真实设备收到命令后,直接查看测试 用例执行结果。如果真实设备不支持自动返回命令执行结果,则根据真实设备收到的 命令,手动操作真实设备上报命令执行结果后,查看测试用例执行结果。

- 测试成功,单击"下一步"进入下一阶段测试。
- 测试失败,排查并处理问题后,单击"重新测试"重测测试用例。

1、loT平台下发命	命令 > 2、命令响应测试
	命令执行结果上报成功
下发的命令:	命令内容: {"led":"OFF"}
	命令状态:执行成功
	命令结果:
用例完成时间:	2019-06-24 11:57:10
日志内容:	[CIG]Cig received the command from cmdh. CIG get command from cmdh successfully, comma ndld = 7163b0191e914772bdddade44ad9642d, callbackUrl = null, serviceId = LED [CIG]Cig get protocol interpreter successfully. Have found Plugin, ready to encode, Manufacture
	重新刻试下一步

----结束

固件升级测试

固件升级测试用于测试设备是否具备固件升级能力。在执行固件升级测试前,请确认 "Profile定义 > OM维护"下,"固件升级"已开启。

步骤1 在固件升级测试界面,单击"下一步"开始测试。

文档版本 15 (2022-06-14)

步骤2根据向导进入测试界面,上传未签名的固件升级包,填写版本号,单击"下一步", 系统自动创建固件升级任务。

请确保上传的升级包用于该设备的固件升级,且文件为zip格式压缩包。

1、上传升级包 > 2、设备	上报数据 > 3、查看升级结果 > 4、升级能力验证	
	请上传一个 未签名 的固件包,并输入固件版本号。	
	patch_656to656sp1.zip & 2.0	/
		七一步

- **步骤3**操作真实设备上报一条属性数据,触发升级任务,待升级任务执行完成后,查看升级结果。
 - 升级成功,单击"下一步"验证设备升级后是否正常工作。
 - 升级失败,排查并处理问题后,单击"重新测试"重测固件升级。

1、上传升级包 > 2、设备上报数据 > 3、查看升级结果	> 4、升级能力验证		
~	设备升级成功,版本号为2.0		
		重新测试	下一步

- **步骤4** 操作真实设备上报一条属性数据,验证设备升级后是否能与物联网平台正常通信,查 看测试用例执行结果。
 - 测试成功,单击"下一步"进入下一阶段测试。
 - 测试失败,排查并处理问题后,单击"重新测试"重测测试用例。

1、上传升级包 > 2、设备上报数据 > 3、查看升级结果 > 4、升级能力验证		
数据上报成功		
升级完成时间: 2019-06-24 15:06:42		
升级前上报的数据: 空		
升级后版本号: 2.0		
升级后上报的数据: {"toggle":36}		
	重新测试	下一步

----结束

软件升级测试

软件升级测试用于测试设备是否具备软件升级能力。在执行软件升级测试前,请确认 "Profile定义 > OM维护"下,"软件升级"已开启。

- 步骤1 在软件升级测试界面,单击"下一步"开始测试。
- **步骤2**根据向导进入测试界面,上传未签名的软件升级包,单击"下一步",系统自动创建 软件升级任务。

请确保上传的升级包用于该设备的软件升级,且文件为zip格式压缩包。

1、上传升级包 > 2、设备上报数据 >	3、查看升级结果 > 4、升级能力验证	
	请上传一个 未签名 的软件包	
	package zip	۵
	L1	
		8 —⊀

步骤3 操作真实设备上报一条属性数据,触发升级任务,查看测试用例执行结果。

- 升级成功,单击"下一步"验证设备升级后是否正常工作。
- 升级失败,排查并处理问题后,单击"重新测试"重测软件升级。

1、上传升级包 > 2、设备上报数据 > 3、查看升级结果 > 4、升级能力验证		
设备升级成功,版本号为OFO_TEST_V1.1.10		
	重新测试	下一步

- **步骤4** 操作真实设备上报一条属性数据,验证设备升级后是否能与物联网平台正常通信,查 看测试用例执行结果。
 - 测试成功,单击"下一步"进入下一阶段测试。
 - 测试失败,排查并处理问题后,单击"重新测试"重测测试用例。

1、上传升级包 > 2、	设备上报数据 > 3、查看升级结果 > 4、升级能力验证	
	数据上报成功	
升级完成时间:	2019-06-24 15:14:43	
升级前上报的数据:	호	
升级后版本号:	OFO_TEST_V1.1.10	
升级后上报的数据:	{"toggle":102}	
	重新测试下一步	

----结束

应用订阅事件测试

应用订阅事件测试主要是测试业务应用是否可以正确调用物联网平台"订阅平台业务数据"接口,订阅设备数据变化通知的能力。

如果物联网平台使用HTTPS协议向业务应用推送数据时,需要在物联网平台上传业务 应用提供的CA证书。可以在开发中心的"应用 > 对接信息 > 推送证书",单击"证书 管理"上传证书,证书获取参考<mark>如何导出https推送证书</mark>。

步骤1 在应用订阅事件测试界面,单击"下一步"开始测试。

步骤2 在"应用名称填写"界面,填写应用名称,单击"下一步"。

1、应用名称填	写 > 2、应用订阅消息 :	- 3、查看订阅结果 >	4、上传订阅截图	
请输入应用信	愿			
*应用名称:	StreetLightAPP			
				下一步

- **步骤3**根据向导进入测试界面,在业务应用订阅设备数据变化消息,查看测试用例执行结果。
 - 应用消息订阅成功,单击"下一步"进入"上传订阅截图"界面。
 - 应用消息订阅失败,在业务应用排查并处理订阅故障后,单击"重新测试"重测测试用例。
- **步骤4** 单击"上传订阅截图"界面中的"+",上传业务应用订阅成功的截图。此图是业务应用正确调用物联网平台"订阅平台业务数据"接口的凭证。

图片上传成功后,单击"下一步"进入下一阶段测试。

🛄 说明

上传的图片大小不能超过20MB。

----结束

应用接收推送数据测试

应用接收推送数据测试主要是测试业务应用是否可以正确接收物联网平台推送设备数 据的能力。

- 步骤1 在应用接收推送数据测试界面,单击"下一步"开始测试。
- 步骤2 根据向导进入测试界面,操作真实设备上报一条profile中定义的属性数据,物联网平台获取数据并推送给业务应用,查看测试用例执行结果。
 - 数据推送成功,单击"下一步"进入"上传应用接收截图"界面。
 - 数据推送失败,排查并处理故障后,单击"重新测试"重测测试用例。
- 步骤3 单击"上传应用接收截图"界面中的"+",上传业务应用接收到物联网平台推送数据的截图。此图是业务应用正确接收物联网平台推送数据的凭证。

🛄 说明

上传的图片大小不能超过20MB。

----结束

7 产品发布(联通用户专用)

非联通用户请查看设备接入服务。

如果开发中心已经对接产品中心,则开发者在完成产品的<mark>自助测试</mark>后,可以把产品发 布到产品中心,已发布的产品可直接应用于商用环境。

申请发布产品

- 步骤1 产品在通过所有测试用例后,单击"申请发布"。
- **步骤2**系统自动完成厂商信息和产品信息完整性的检查。如果没有重要信息缺失,则单击 "发布"。
 - 黄色信息缺失提示:部分信息不完整,不影响发布产品,但发布到产品中心可能 会审核不通过,建议补充。
 - 红色信息缺失提示:重要信息缺失,需补充完整才能发布产品。

终止剥试,返回开发		1 預检查	2 自助测试	3 申请发布
产品信息				发布
厂家LOGO检查		(A)		/ 修改
厂家名称检查		HW		/ 修改
厂家联系方式		12345678901		/ 修改
厂家简介检查		提供更好的物联网产品及服务。		/ 修改
产品简介	①信息缺失	补充完整之后可点击"发布"按钮,将产品发布至产品	品中心	/ 修改
功能亮点	①信息缺失	发布至产品中心可能会审核不通过,建议补充		/ 修改
产品规格		智慧路灯		/ 修改
服务支撑	①信息缺失	发布至产品中心可能会审核不通过,建议补充		/ 修改
客户案例	①信息缺失	发布至产品中心可能会审核不通过,建议补充		/ 修改
产品图片				/ 修改

步骤3选择发布方式:"公开发布"或"私有发布",单击"发布",提交发布申请。 注:一旦发布不可修改发布方式。

选择发布方式	
发布之后会有专, 审核通过后可以在产品中心	人审核是否合格 2发布,也可以供自己使用
● 公开发布(所有人可见)	○ 私有发布(仅自己可见)
发布	取消

步骤4 查看产品是否发布到产品中心。

登录认证产品中心,选择"产品",在产品列表查看通过审核的产品。

- **步骤5** (可选)申请关联云市场产品,以便用户通过产品中心直接购买该产品。
 - 进入认证产品中心,鼠标移至右上角的用户名,单击下拉列表中的"我的产品"。



2. 在产品列表,单击产品右侧的"云市场发布"。

全部 产品 1 个							
	公开	产品名称 智能路灯	型号 lightD01	设备类型 StreetLight	接入协议 CoAP	最后修改日期 2019-09-12 16:10:35	云市场发布

系统弹出"云市场发布"窗口,单击"填入商品ID",然后单击"确定"。
 注: "商品ID"为云市场产品的产品ID,在产品审核上架后,可在云市场中该产品的商品详情页面获取。云市场发布产品请参考云市场商品接入。

×

云市场发布

填入商品ID之后,产品中心的商品即可出售,若您还没上架,请先去云市场完成相关操作



----结束



非联通用户请查看设备接入服务。

当产品发布到产品中心后,我们在"开发中心"定义好的产品就可以在"设备管理服务"中使用了。由于开发中心所在的平台环境为测试环境,设备管理服务所在的平台环境为商用环境,两个环境的数据不能互通,已对接测试环境的设备/应用服务器要迁移商用环境,您需要通过本章节的指引完成商用对接。

总体说明

- 迁移对接的过程中,开发中心数据不割接,由应用服务器负责保存设备在开发中 心的历史数据。
- 建议应用服务器先迁移商用环境再迁移设备,设备迁移采用API逐个重新注册的方式。

创建应用

步骤1选择"设备管理",登录设备管理服务控制台。

()) 物联网平台	设备管理 设备管理在设备接入基础上,提供了更丰富完备的设备管	理能力,等化海量设备管理复杂性,节省人工操作,	操开管理效率。		进入设备管理服务
设備统入 设备管理 设备发放 OceanLink 开发中心	職点(KP)機計 (1) 合語 (1) (1) (1) (1) (1) (1) (1) (1)	_{ёл}	^{規則引擎} O	秋回件包 0/0	11.001.35 O
	伏臣保作	ビングログログ ビングログログ ジングの注意である ジングの注意である ジングの注意である ジングの注意である ジングの注意である ジングの注意である ジングの注意である ジングング ジングングング ジングングング ジングング ジングング ジングングング ジングングング ジングング ジングングング ジングングング ジングングング ジングングング ジングングング ジングングング ジングングングング ジングングングング ジングングング ジングングング ジングングング ジングングング ジングングングング ジングングングング ジングングングング ジングングングング ジングングングング ジングングングング ジングングングングング ジングング ジングング ジングングング ジングングング ジング ジングング ジングングング ジングング ジング ジング ジング ジング ジング ジングング ジング ジング ジング ジング ジング ジングング ジング ジング ジングング ジングング ジング ジング ジングング ジング ジングング ジングング ジングング ジング ジングング ジング ジングング ジングング ジングンジング ジングング ジングング ジングンジング ジング ジング ジング ジングング ジング ジング ジング ジング ジング ジングング ジング ジング ジングンジング ジング ジングン ジング ジング ジング ジング ジングン ジング ジングン ジング ジングン ジングン ジングン ジンジンジン ジンジンジンジンジンジン ジンジンジンジン	しまた たまでの しまた にの にの にの にの にの にの にの にの にの にの	け数使式 (760年、 ・ 公室管理 3607/0万、 県に日本市 第7日数後 1909年2	2.4 集人和设备管理服务,所有资源并共称以下农業标准进 第:每天先展期运び个设备的计者。就出先局期运动分前设 每级运行进度(时间增 3.5 最有点最新运动分前用 美国的注意(前时有 这种分离

步骤2系统已自动为用户创建一个应用,若使用默认应用,需要在"应用定义 > 安全"中重置密钥,且无需执行**步骤3**至步骤4。

如果用户想要自己创建应用,需要将默认应用删除,重新创建。

Z用管理 > 创建应用		
基本信息		
*应用名称:	请输入	
	名称只能包含大写字母、小写字母、数字和下划线,且不能大于50个字符	
企业:	华为 ~	
所属行业:	智慧家庭行业	
*消息跟踪数据授权:		
	大切12001天、千百日は以て18時の17日7月12日は東南50kmに近め3、71時6月86日また13505年、1月時19年の支援が登場したします。 息、将会降低回動定位放率、建议認須投合平台管理员进行业务数据的限定。	
消息推送		
洗探执锁 亚公士持应用服务婴	MinutTDfiluTTDc开始协议体》之中,继续应用uTTDc计议, 也今州再算	
推送协议:	HTTPS ~	
	此方式平台需要对接入的应用服务器采用CA证书进行认证,认证通过后才允许接入,以保证数据的安全	7
CA证书 ⑦:	文件大小不超过1M, 且必须为pem、crt、cer或der文件 1	

步骤3 (使用默认应用无需执行此步骤)参考下表按照实际情况填写配置参数。

参数名称	参数说明
基本信息	
应用名称	定义用户的应用的名称,应用名称必须为帐号下唯一,且创建后 不可更改。
所属行业	根据用户的应用的行业属性进行选择。
消息跟踪数据授权	 设置物联网平台运营管理员可以跟踪发生故障的设备的权限。 打开授权,表示平台管理员在辅助租户进行设备的故障定位时,可以跟踪设备上报的业务数据,便于快速解决问题。授权打开的状态下需要设置"授权时效",可设置"指定时间"或者"永久有效"。为了保证用户的数据权益,物联网平台运维管理员跟踪的设备数据保留时间不超过3天。 关闭授权,表示平台管理员在辅助租户进行设备的故障定位时,不能跟踪设备上报的业务数据,可能导致没有足够的信息,将会降低问题定位效率,建议您授权给平台管理员进行业务数据的跟踪。
消息推送	

参数名称	参数说明
选择协议	推送协议 应用服务器向物联网平台进行消息订阅,物联网平台在推送数据 时可以采用加密的HTTPS协议或者非加密的HTTP协议。推荐采用 HTTPS协议。
	• HTTPS方式:表示物联网平台与应用服务器之间采用加密的传输协议,需要应用服务器侧上传CA证书。
	 HTTP方式:表示物联网平台与应用服务器之间采用非加密的 传输协议。此方式的安全性较低,存在物联网平台与应用服务 器之间通信信息泄露风险。
	CA证书
	创建应用时无需上传CA证书,请创建完应用后根据 <mark>配置数据推送</mark> 业务上传CA证书。
平台能力	
设备数据处理	物联网平台提供设备上报数据的存储能力,用户可以通过"存储 历史数据"的开关进行控制,默认为"打开"状态。
	 打开开关:即物联网平台会对上报数据进行存储,存储时间以 界面显示的存储时间为准。
	• 关闭开关:即物联网平台不对上报数据进行存储。
	用户可以通过 <mark>数据转发规则</mark> 转发到华为云其他云服务上进行存储 和处理。
推送服务	应用服务器向物联网平台订阅设备信息,物联网平台能够向应用 服务器进行消息推送。
其他	•
应用描述	对该应用的描述。
应用图标	为该应用添加自定义图标。

- **步骤4** (使用默认应用无需执行此步骤)勾选"我已阅读并同意《个人数据使用条款》", 单击"确定",完成创建应用。创建完成后,系统弹出"成功"对话框,显示应用的 基本信息,包含应用ID、应用密钥、应用对接地址和设备对接地址。
 - 请单击"保存密钥至本地",以保存应用密钥信息,密钥信息在应用详细页内不可见,请妥善保管。如果遗忘应用密钥时,可在"应用列表"中单击 ²,选择 "重置密钥",或者通过应用详情页内"应用定义 > 安全"进行重置密钥。

🗀 说明

应用ID和应用密钥用于应用服务器接入物联网平台,如果重置密钥,旧的密钥将不能使 用,您的应用服务器需要更新为新的密钥才能重新接入平台,请谨慎操作。

- 单击"查看应用详情"进入应用详情页,具体功能介绍参照"应用详情"。
- 单击"返回应用列表",返回到创建应用页。单击"应用列表"中应用的图标, 可直接进入该应用的应用详情页。

----结束

导入产品模型

- 从产品中心导入产品模型
 - a. 单击页面左下角的 切换左侧菜单,打开"产品模型"页面,点击页面 右上角的"新增产品模型 > 从产品中心导入"。

Default1		产品模型	+ 新電产品構築
▲ 首页		产品模型(也称Profile)用于描述设备具备的能力和特性。开发来通过定义Profile,在IoT平台构建一款设备的抽象模型,使平台理解该款设备支持的服务、属性、命令等信息,如颜色、开关等。	从产品中心导入
12 产品模型		Q. 各称/型号/设备类型/ 被救	本地导入
@ 10%		· 建双环带 = 0	中切出短
 8年4日 			
▲ 告答			
Ⅲ 批量任务	-		
10 MERI			
62/19/8	-		
③ 輸助文档			
		暂无数据	
0			

b. 在产品中心选择已发布的产品,导入即可。

产品模型 > 产品中心				出 第三方私有产品
*晶名称	设备类型	产品范围		厂商名称
请输入产品名称	请输入产品类型	第三方公开	~	请输入厂商名称
		自有		
		第三方公开		技來 重要
资本结果: 83				
模组与开发板	智慧城市	智慧城市	智慧城市	智慧城市
HC_OCLINK_DEMO	DD_GPS_11 DODEM	HDC003MBUS HONGDACHIP	TSTH010 BMTS	无线手动报警按钮通信底座 FourFaith
智慧城市	车联网	智慧城市	智慧城市	智慧城市
FIRE *				EXE:
于代表来很感觉演信应应	NP IoT干扰数据终端	184403926-34 主	NRIGT	10E

🗀 说明

如果您需要导入第三方私有产品,单击右上角"第三方私有产品",填写产品验证 码。产品验证码由第三方私有产品提供方在认证产品中心的"我的产品"页面,单击 该产品的"获取产品码"得到。

- 本地导入产品模型
 - a. 单击页面左下角的 切换左侧菜单,打开"产品模型"页面,点击页面 右上角的"新增产品模型 > 本地导入"

resourcetest	•	产品模型							十 新增产品模型
▲ 盖页		产品模型(也称Pn	ofile)用于描述设备具备的能力和#	种生,开发黄逝过定义Profile,在lo	T平台构建一款设备的抽象模	型,使平台理解该款设备支持	的服务、属性、命令等信息,如	调色、开关等。	从产品中心导入
2 产品模型		Q 各称/型号/S	2备类型/	Bitte					本地导入
₽ 128		搜索结果: 24							手动的AE
a (2-6			产品名称	型号	设备类型	厂商名称	协议类型	自动的时间	1245 BID9 T-80
 8748 			streetlight	d77f80fbeb3c9404	light	siko	MQTT	2020-04-07 16:09:24	
▲ 告盼									
二 出意任务			产品名称	型号	设备类型	厂商名称	协议类型	包括10月间	详稿 删除 下载
D IBN			Simulator	54560068886005110	Simulator	LICENSI	MQT	2020-04-07 19:37:33	
			*522		10 M M R	F-96.0224	24.27136/001	of Beneric	
家以午時	*		Simulator01	a12e4c854a5f2b0a	Simulator	HUAWEI	MQTT	2020-04-08 18:47:00	详情 删除 下载
 ● 報助文档 		· ·							
			产品名称	型号	设备类型	厂商名称	协议类型	包细胞时间	
			OC_StreetLight	888c4e7810fd8af2	Simulator	HUAWEI	MQTT	2020-04-08 18:53:25	详情 删除 下载
			产品名称	型号	设备类型	厂商名称	协议类型	89309140	1948 BIS TH
			smoke_Detector	ef64ccf28cce9b93	smokeDetector	huawei	MQTT	2020-04-24 14:37:47	PT10 8370- 1144
0			产品名称	型号	设备类型	厂商名称	协议类型	自動的作用	详情 删除 下载
•			ctstest	8a295b2971e94816	ctstest	huawei	MQTT	2020-04-30 09:46:14	

b. 在弹出的页面,填写"产品名称",导入本地产品模型的资源文件,点击 "确认"按钮。

手动导入产品	×
*产品名称	
test101	
*资源文件	
Profile.zip	企
兼容旧版本的设备模型	
确定取消	

🛄 说明

选择"本地导入"的方式导入产品模型,如果产品模型的协议类型为LWM2M/ CoAP,则需要联系华为技术支持上传插件。推荐您将产品发布到产品中心后再选择 "从产品中心导入产品模型"。

迁移应用

迁移应用是指将对接了开发中心(测试环境)的应用服务器迁移到设备管理服务(商 用环境)中去,您需要完成以下步骤:

- 步骤1 为了使应用服务器能对接这个业务使用空间,您需要将对接的平台地址修改为"设备管理服务"的应用对接地址,应用对接地址获取请参考平台对接信息。
- **步骤2** 在"设备管理服务"中, **应用**对应的是开发中心的**项目**。为了使应用服务器能调用平台的接口,您需要将应用服务器中设定的**应用ID**和**应用密钥**修改为新的值。

步骤3 如果应用服务器调用"鉴权"接口成功,表明应用服务器已对接到商用环境。

----结束

迁移设备

迁移设备是指将对接了开发中心(测试环境)的设备迁移到设备管理服务(商用环 境)中去。迁移对接的过程中,开发中心数据不割接,由应用服务器负责保存设备在 开发中心的历史数据。

- 步骤1 您需要将设备根据通信协议修改对接的平台地址为"设备管理服务"的设备对接地 址,设备对接地址获取请参考平台对接信息。
- **步骤2** 您需要在"设备管理服务"中注册设备,可以使用应用服务器调用平台注册设备接口,也可以在界面上注册设备。
- 步骤3 平台为每一个注册到平台上的设备生成了一个deviceId,这是设备在平台上的唯一 ID,您需要将设备和ID映射关系刷新为新的ID。

----结束

对接验证

将真实的设备和应用接入到物联网平台后,需要对设备上报数据到物联网平台,平台 能正常将设备上报的数据推送给应用服务器进行验证;同时,验证应用服务器向设备 下发命令,设备能正常收到命令并执行成功。

- 步骤1 设备上电,基于在设备上定义的业务逻辑进行数据采集,向物联网平台上报数据。
- **步骤2** 登录"设备管理服务控制台",选择"设备管理 > 设备 > 所有设备",在设备列表中 查看对应设备的状态。如果状态为"在线",则表示设备已经成功接入物联网平台。

▲ 首页	所有设备			建成 设备名称 ID 设备识别码等 Q > 高級	現案 出 号出 〇 刷新
11 产品模型	白教	● 在纬 の	東洋の	● 手搬注 ④	
 ● 设备 	16	1			
所有设备	10		0	0	
设备注册	搜索结果: 16				✿ 列表设置
标签	- 秋志 - 设建	新D 设备识别码	设备类型	厂商名称 协议类型	创操作
■ 群组	• 在线	Contraction in the second	WaterMeter	HZYB COAP	20 🔊 🗑

步骤3 点击对应的设备,进入设备详情页,在详情页中查看"最近上报数据",如果能正常解析和显示对应的数据,则表示设备上报数据成功。

如果需要查看所有上报的历史数据,则可以在设备详情的"历史数据"中进行查看。

▲ 首页	设面列表 > 设备详简					
■ 产品模型	设备信息 历史数据 操作记录 消息服除 告票 设备影子 设置 命令 子设备					
所有设备	最近上版数据 (2019-06-11 15.02-27)	查看全部属性 >				
设备注册	pressure temperature Visua Mater					
标选	0 25					
■ 群組	2019-06-11 15:02:27 2019-06-11 15:02:27					

- **步骤4** 在CallBackURL对应的服务端中,查看是否收到物联网平台推送的数据,如果能正常接收,则表示物联网平台推送消息成功。
- **步骤5** 通过应用服务器向设备下发命令,在设备侧查看设备的执行结果,如果设备的执行动 作与下发的命令相符,且在"设备管理服务控制台"中查看下发命令任务的执行结果 为"已送达"或"成功",则表示应用服务器向设备下发命令成功。
- 对于NB-IoT设备,如果采用的是缓存下发模式,需要触发设备再次上报数据后, 命令才会下发给设备。
- 如果设备会给物联网平台返回命令的执行结果(成功或失败),则命令下发的任务状态会根据执行结果刷新为"成功"或"失败"。

----结束

开始使用

物联网平台设备提供海量设备的接入和管理能力,配合华为云其他产品同时使用,帮助快速构筑物联网应用,简化海量设备管理复杂性,节省人工操作,提升管理效率。

功能	简介
应用管理	应用可以理解为在物联网平台中为用户的业务划分一个项目空间,当 用户在开发中心、线下环境里完成应用服务器侧的开发、物联网平台 侧的开发以及设备侧的开发后,就需要在控制台上创建应用,将开发 完的应用服务器与真实设备接入到这个项目空间中,实现设备的数据 采集和设备管理。
产品模型	又称Profile,用于定义一款接入设备所具备的属性(如颜色、大小、采 集的数据、可识别的指令或者设备上报的事件等信息),然后通过厂 家、设备类型和设备型号,唯一标识一款设备,便于平台识别。产品 模型可通过开发中心进行无码化开发。
设备注册 鉴权	物联网平台对接入平台的设备进行鉴权认证。待真实设备上电后,设 备可以上报数据到物联网平台,物联网平台根据应用服务器的订阅消 息类型,把消息推送给应用服务器。
订阅推送	订阅:是指应用服务器通过调用物联网平台的API接口,向平台获取发 生变更的设备业务信息(如设备注册、设备数据上报、设备状态等) 和管理信息(软固件升级状态和升级结果)。 推送:是指订阅成功后,物联网平台根据应用服务器订阅的数据类 型,将对应的变更信息推送给指定的URL地址。
数据上报	当设备完成和物联网平台对接后,一旦设备上电,设备基于在设备定 义上的业务逻辑进行数据采集和上报,可以是基于周期或者事件触 发。
命令下发	为能有效地对设备进行管理,设备的产品模型中定义了物联网平台可 向设备下发的命令,应用服务器可以调用物联网平台开放的API接口向 单个设备或批量设备下发命令,或者用户通过物联网平台直接向单个 设备下发命令,配置或修改设备的服务属性值,以实现对设备的远程 控制。
设备配置 更新	物联网平台提供设备配置更新功能,即用户可通过控制台对单个设备 或批量设备的设备属性值进行修改,满足用户频繁、快捷、方便的管 理设备的诉求。
设备影子	设备影子是一个JSON文件,用于存储设备的在线状态、设备最近一次 上报的设备属性、应用服务器期望下发的配置。每个设备有且只有一 个设备影子,设备可以获取和设置设备影子以此来同步状态,这个同 步可以是影子同步给设备,也可以是设备同步给影子。

功能	简介
规则引擎	指用户可以在物联网平台上可以对接入平台的设备设定相应的规则, 在条件满足所设定的规则后,平台会触发相应的动作来满足用户需 求。包含设备联动和数据转发两种类型。
群组与标 签	群组是一系列设备的集合,用户可以对应用下所有设备,根据区域、 类型等不同规则进行分类建立群组,以便处理对海量设备的批量管理 和操作。 物联网平台支持定义不同的标签,并对设备打标签,通过标签,可以 快速筛选设备。
设备监控	提供查看设备详情、设备状态管理、查看报表、查看操作记录、查看 审计日志、告警管理、设备消息跟踪等设备监控与运维能力,提升设 备的可维护性。
远程诊断	支持用户对接入的设备进行远程维护操作,快速定位问题及恢复业 务,减少近端维护引入的高成本。当前支持的远程维护操作包括设备 的运行日志收集、重启模组。
固件升级	用户可以通过OTA的方式对支持LWM2M协议的设备进行固件升级,升 级协议为LWM2M协议。
软件升级	用户可以通过OTA的方式支持对LWM2M协议的设备进行软件升级,升 级协议为PCP协议。
网关与子 设备	物联网平台支持设备直连,也支持设备挂载在网关上,作为网关的子 设备,由网关直连,通过网关进行数据转发。

9 IoT 技术认证(联通用户专用)

非联通用户请查看设备接入服务。

认证概述

华为IoT技术认证是华为面向合作伙伴提供的IoT技术认证服务,旨在通过严谨而专业 的检测技术帮助合作伙伴发现并解决自身产品的不足,缩短产品上市周期,同时联合 合作伙伴向共同的客户提供经过验证的方案。通过该认证的伙伴产品将获得华为颁发 的IoT技术认证证书,该证书标志着华为对双方物联网产品预集成的技术认可。

技术认证特点:

- 权威性:提供华为商业化平台认证服务、国家/区域重要法规认证服务、重要行业 安全标准认证服务,得到许多行业客户以及运营商的认可。
- 全面性:完备的测试环境和专业的测试服务,从功能、性能、可靠性、安全、可 维护性等为物联网产品提供端到端的软硬件能力测试,高效打造优质产品。
- 多渠道推广:达到一定标准的产品可以在产品中心发布,获得证书的产品不仅可以在产品中心、华为云市场发布,而且还可以享受华为提供的商业推广机会,例如展会活动、联合营销等。
- 方便快捷: OpenLab认证实验室全球覆盖、线上一站式申请、线下专业人员指导,为全球合作伙伴提供贴身的技术认证服务。

🛄 说明

华为会针对行业特点对认证产品的安全进行必要的验证,但不承担合作伙伴所属产品的安全责任。

认证分类

根据产品的认证方式、认证测试内容和认证目的不同,分为Enabled和Compatible两 种认证证书 。

认证方式	线上自助测试	OpenLab现场测试
认证测试内容	验证业务应用和设备能否正确接入物联 网平台,并且实现基础功能,如数据上 报、命令下发、应用订阅事件等。	全面验证业务应用和设备能否可靠安全的接入 物联网平台 , 功能、性能、可靠性、安全、可 维护性等能达到华为物联网平台标准。
认证目的	指定的产品是基于华为开放的产品能力 和技术支持服务开发的,经认证测试有 效调用了华为产品的ICT能力	指定的产品同华为产品完成了互联互通测试, 能够可靠安全地接入华为物联网平台,且满足 协议规定的性能要求。

因为设备的通信技术和集成方式不同,所以将Compatible证书细化为NB-IoT Compatible、Agent Lite Compatible和Agent Tiny Compatible三种认证类型。

认证证书	认证类型	认证对象	认证测试指导
Enabled	Enabled	业务应用	测试指导请参考 <mark>自助测试</mark> (<mark>联通用户专用</mark>)。
Compatible	NB-IoT Compatible	使用NB-loT通信技 术的设备	测试指导请参考OpenLab <mark>现场测试</mark> 。
	Agent Lite Compatible	集成Agent Lite SDK的设备	
	Agent Tiny Compatible	集成LiteOS SDK的 设备	

如需申请技术认证请联系oceanconnect@huawei.com。

OpenLab 现场测试

合作伙伴在物联网平台上完成产品开发(包括<mark>平台侧开发、设备侧开发</mark>和应用侧开发)后,然后才可以进行OpenLab技术认证,认证流程如下。



关键操作	说明	相关资源
成为合作 伙伴	提交企业相关信息,成为合作伙伴,可以申 请合作伙伴的权益。 注: 如果未注册华为账号,需先进行华为账 号注册。	成为合作伙伴
合作方案 构建	合作伙伴根据华为提供的构建指导书,提交 合作方案资料。通过专业检测服务完成该方 案的构建和优化。	-

关键操作	说明	相关资源
合作方案 技术认证	合作方案完成构建后,合作伙伴可以申请华 为技术认证。通过认证的方案,将获得华为 颁发的技术认证证书,同时该技术认证结果 将公布到华为技术认证证书网站供合作伙伴 和客户查询。	 华为IoT应用认证指导 (Enabled) 华为NB-IoT终端认证 指导(Compatible) 华为IoT Agent Lite终 端认证指导 (Compatible) 华为IoT Agent Tiny 终端认证指导 (Compatible)