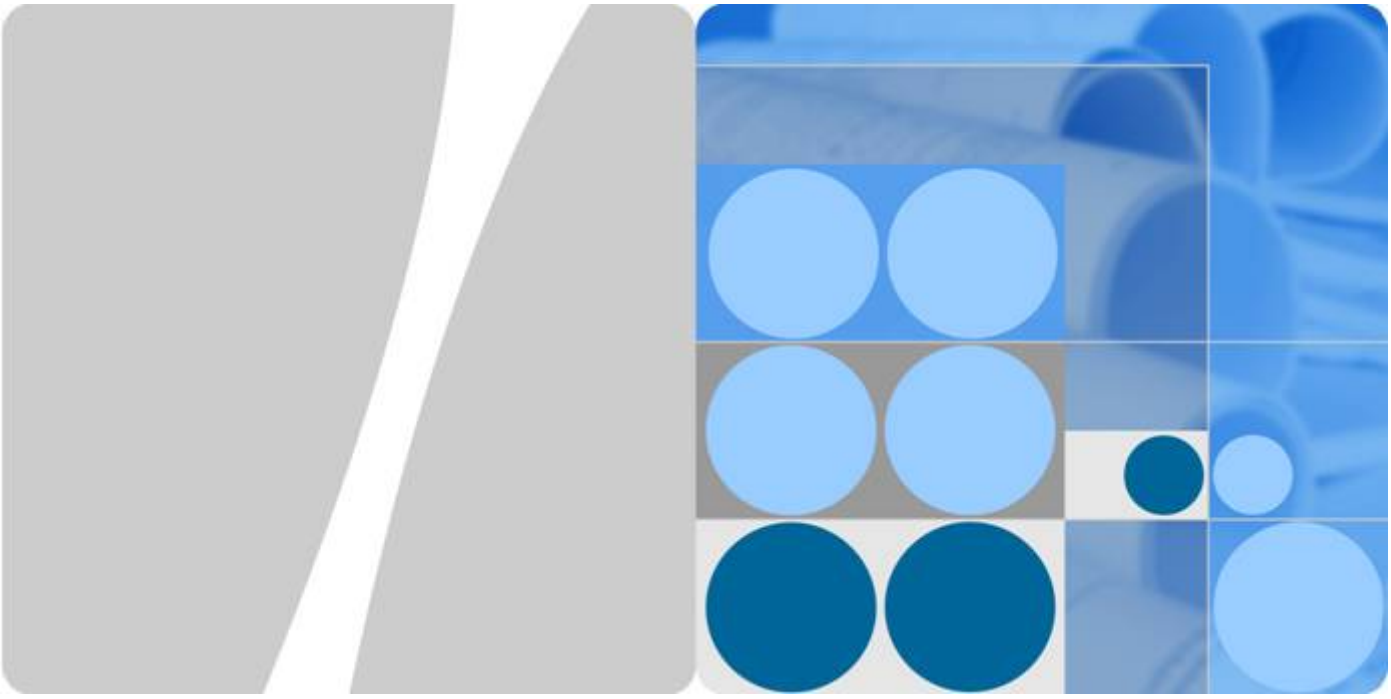


华为 IoT 平台

北向应用开发指南

文档版本　01

发布日期　2017-12-07

华为技术有限公司

HUAWEI

商标声明


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

注意

您购买的产品、服务或特性等应受华为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默示的声明或保证。

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。

华为技术有限公司

地址： 深圳市龙岗区坂田华为总部办公楼 邮编：518129

网址： http://www.huawei.com

客户服务邮箱： support@huawei.com

客户服务电话： 4008302118

目 录

1 开发者必读

按照本文档的指导，能够快速验证平台北向restful接口功能，体验业务功能，熟悉业务流程。

如果您在开发过程中遇到接口调用问题可优先参考下文提及的Lite Demo，比对发给平台的消息内容，**简单的接口调用问题可以快速定界定位**。

本文档主要涉及以下功能的指导：应用鉴权、注册设备、配置设备、消息订阅、数据上报和命令下发等。

2 概述

推荐阅读

强烈建议开发者花5分钟的时间阅读一下OceanConnect开放能力的相关**基本概念**。

为了深入了解OceanConnect生态集成的总体情况，建议您花2个小时阅读**《华为IoT OceanConnect生态集成开发入门指南.doc》**。

开发求助方式

如果您在开发过程中遇到任何问题，有如下几种求助方式：

- 华为开发者社区**OceanConnect论坛**查找答案或发贴求助
- 华为开发者社区**个人中心->我的提单->提交问题**（选择产品“物联网->OceanConnect”）

图 2-1 华为开发者社区 OceanConnect 论坛

图 2-2 开发者社区的问题提交

3 前期准备

加入华为 IoT 解决方案伙伴计划

参考如下两个文档完成申请：

MAR系统使用简要手册 - 伙伴（中文）

华为IoT合作伙伴计划政策

获取 OceanConnect OpenLab 对接信息

对接信息主要包括：

- 应用对接的平台IP和端口
- AppId和密码
- 开发者Portal（待上线）的账号密码

一般情况下，MAR审批完成后OpenLab会主动发布对接信息给开发者。

注意：

如果在MAR电子流审批后，没有收到OpenLab对接信息，您可以发送MAR电子流ID到**mailto:iotadmin@huawei.com**以索取环境对接信息。

注册华为开发者社区帐号并下载开发资料

打开**华为OceanConnect生态社区网页**，点击右上角“注册”， 按要求填写相关信息即可（如果已注册过帐号请跳过注册这一步骤）。

打开以下链接并点击或者“下载”按钮，下载相关开发资源：

- **华为IoT Platform API参考(北向)**
- **华为IoT Platform Lite Demo(North)**
- **设备描述文件profile模板**（即Profile资源包样例）

4 开发环境准备

本步骤以java语言为例，提供了安装JDK、配置环境变量、安装Eclipse的方法。如果使用的开发语言不是java，请自行准备开发环境。

4.1 安装 JDK1.8

下载jdk-8u45-windows-x64.exe，双击安装，选择路径后，默认安装。

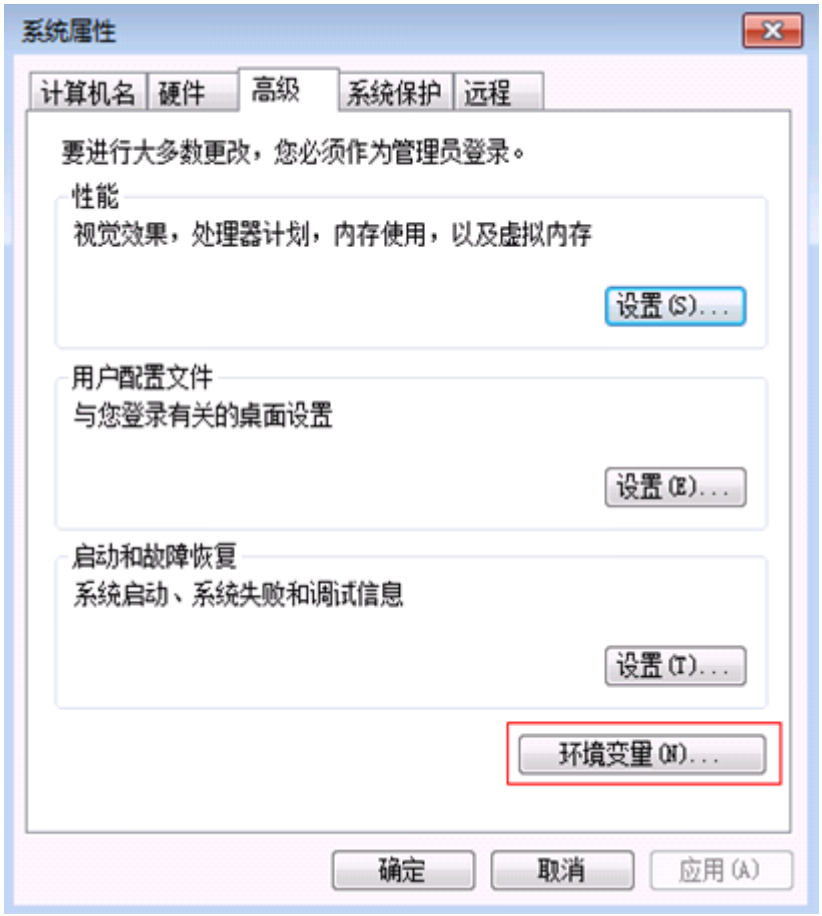
说明

- 请务必安装JDK1.8版本。JDK1.8官网下载地址：**http://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html**

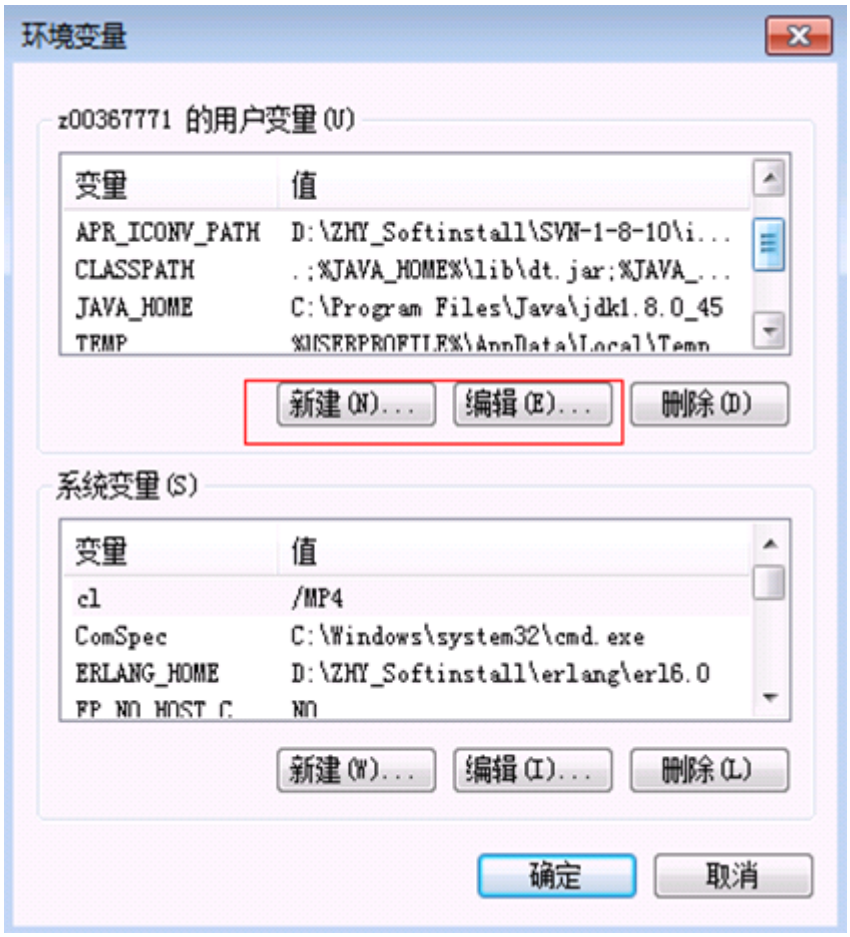
4.2 配置 Java 环境变量（Windows 操作系统）

单击右键计算机，选择属性，进入如下页面。

点击“高级系统设置”



点击“环境变量”

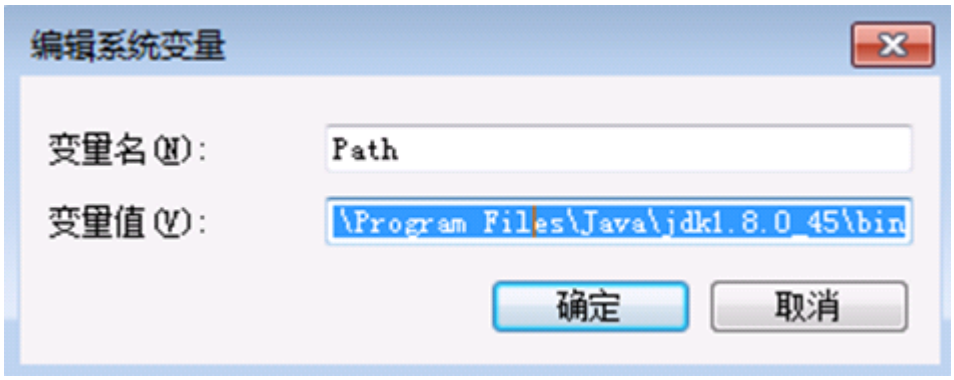


需配置3个变量：JAVA_HOME、PATH、CLASSPATH（大小写不限），若已经存在则点击“编辑”，不存在则点击“新建”。通常只有PATH存在，另外两个变量需要新增。

JAVA_HOME指明JDK安装路径：C:\ProgramFiles\Java\jdk1.8.0_45。此路径下包括lib，bin等文件夹。

PATH变量使系统可以在任何路径下识别java命令。需在该变量最后添加内容

“; C:\Program Files\Java\jdk1.8.0_45\bin”。两个路径间注意使用“;”分割，分号是英文半角。



CLASSPATH为java加载类（class or lib）路径。只有配置CLASSPATH，java命令才能识别。CLASSPATH的取值为“.;%JAVA_HOME%\lib\dt.jar;%JAVA_HOME%\lib\tools.jar”。

注意：此句首有一个’.’，表示当前路径。%JAVA_HOME%即表示引用签名指定的JAVA_HOME。

重启windows系统，使环境变量生效。

最后可以通过DOS命令来检查环境变量是否配置成功。打开Windows的命令行，运行如下的命令来检查你的Java安装，命令行如下：

C:\Users\username>echo %JAVA_HOME%

C:\Users\ username >javac －version

其中username 为开发者的用户名。正确的返回如下：

4.3 安装 Eclipse

下载eclipse-jee-luna-SR2-win32-x86_64 .zip，直接解压缩到本地即可使用。

说明

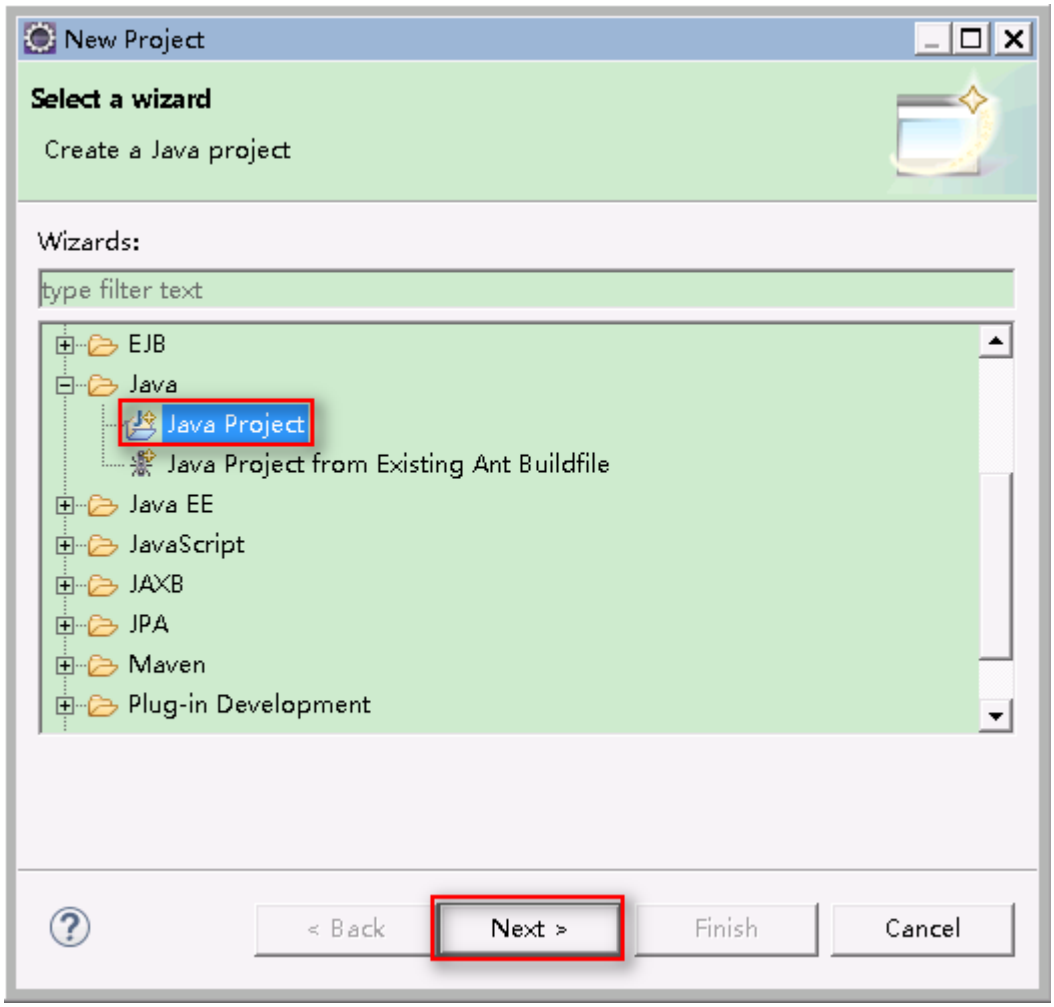
Eclipse版本和jdk必须都是32位的，或者都是64位的，否则启动会找不到jvm.dll。

官网下载地址：

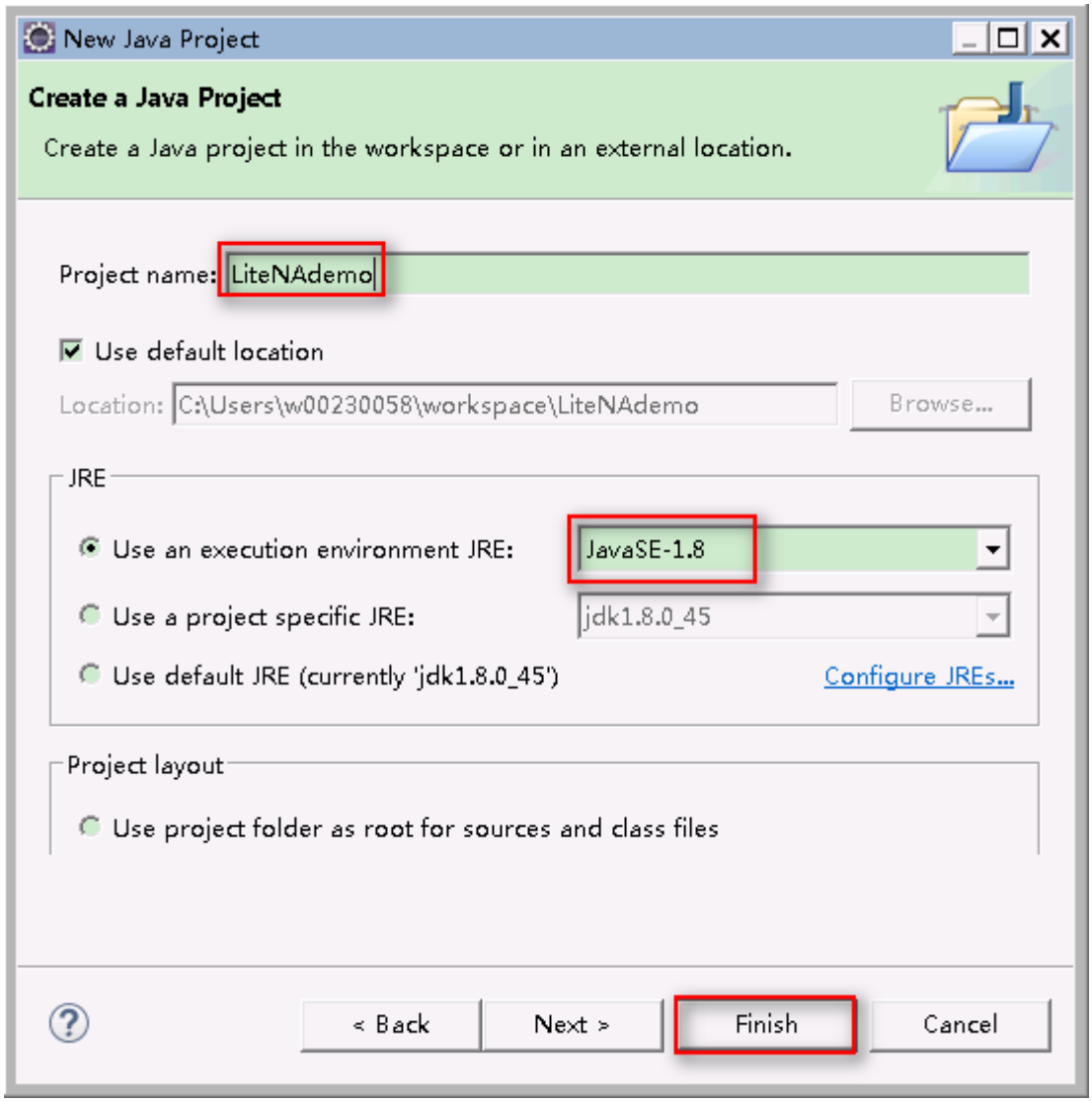
http://www.eclipse.org/downloads/packages/eclipse-ide-java-developers/lunasr2

4.4 新建工程

打开Eclipse，选择菜单File->New->Project，在弹出的对话框中选择Java Project，点击Next（本文档以Java Project为例，可以根据实际情况选择其他工程，如果您已有自己的工程，请跳过这一步）。

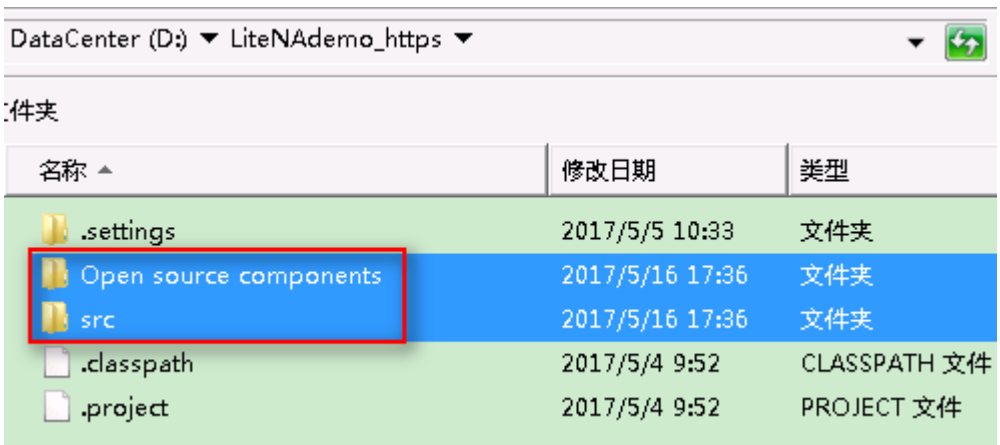


填写工程名，注意选择JRE版本为1.8，点击Finish。

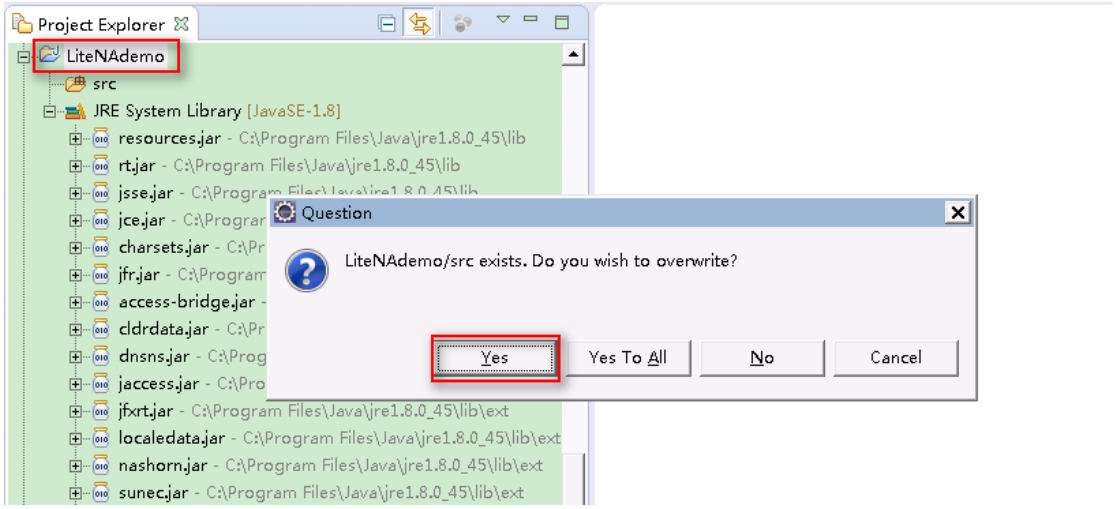


4.5 导入样例代码

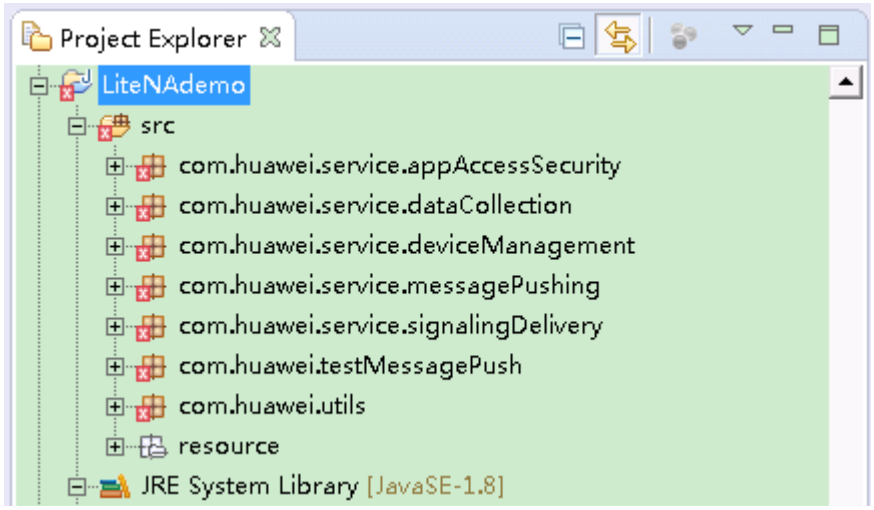
将下载到的样例代码解压，进入LiteNADemo(https)\DemoCode\src，拷贝如下图所示的两个文件夹（选中文件夹后，使用Ctrl+C快捷键）



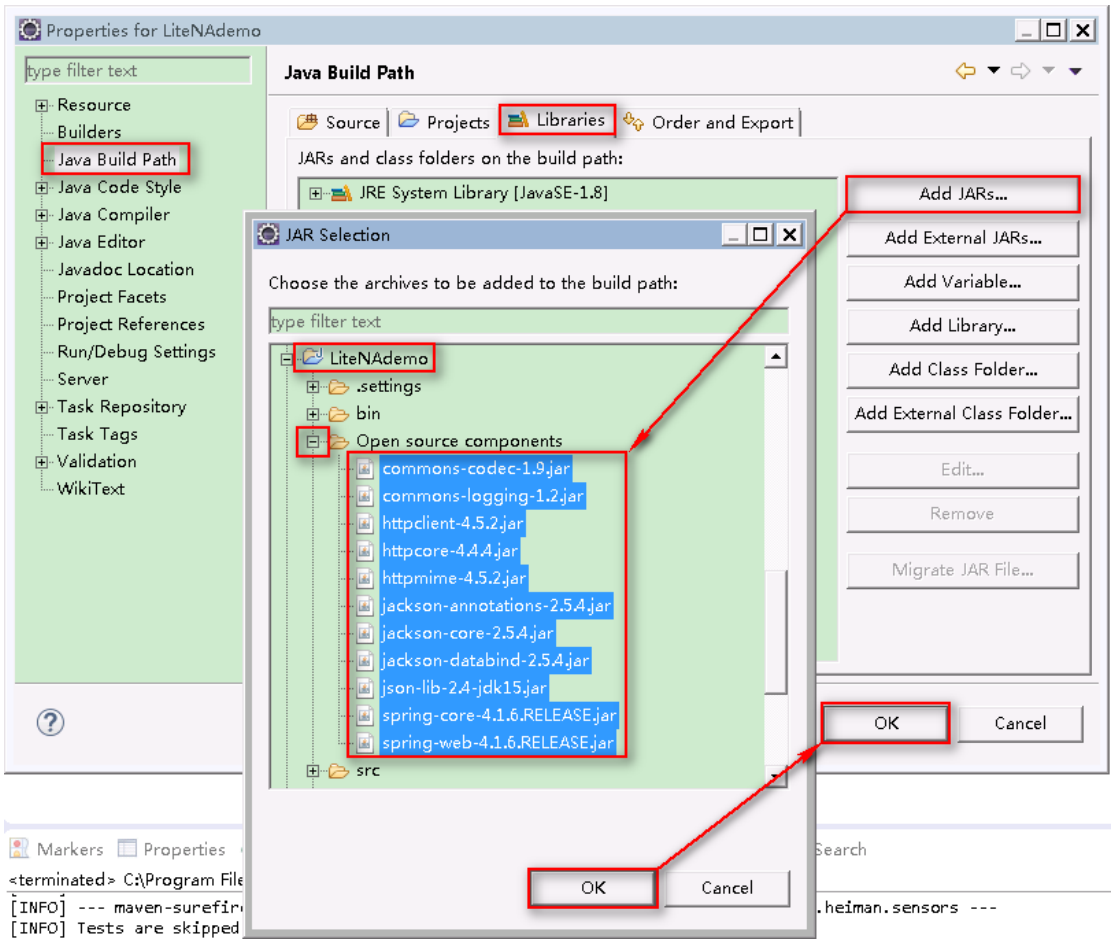
打开之前新建的工程，选中工程名，使用快捷键Ctrl+V将文件夹粘贴到工程目录下：



会发现src目录下多出几个包，且工程会有很多错误：



右击工程名，在右键菜单中选择Properties。在弹出的对话中依次选择Java Build Path-> Libraries-> Add JARs，进入工程目录\Open source components，选中所有的jar，最后点击OK。

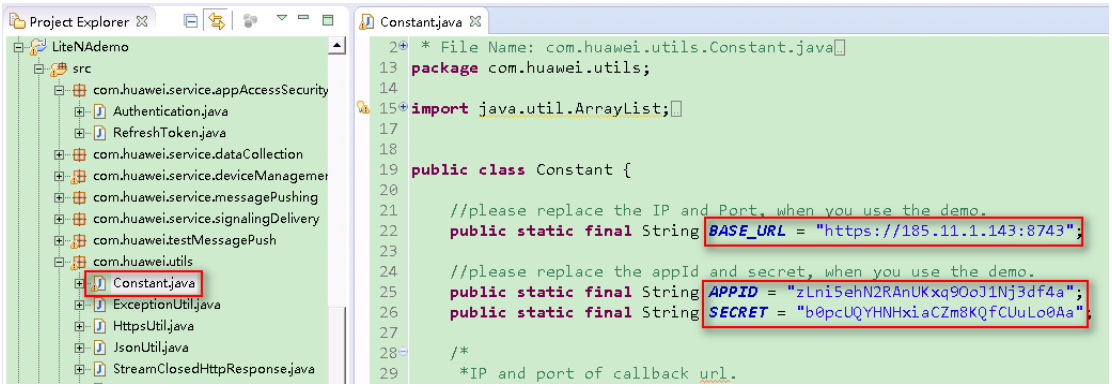


导入jar包之后工程中的错误就会全部消失。

5 调用鉴权接口，获取 accessToken

在调用接口前要先修改平台IP端口、appId和密码等常量的值。

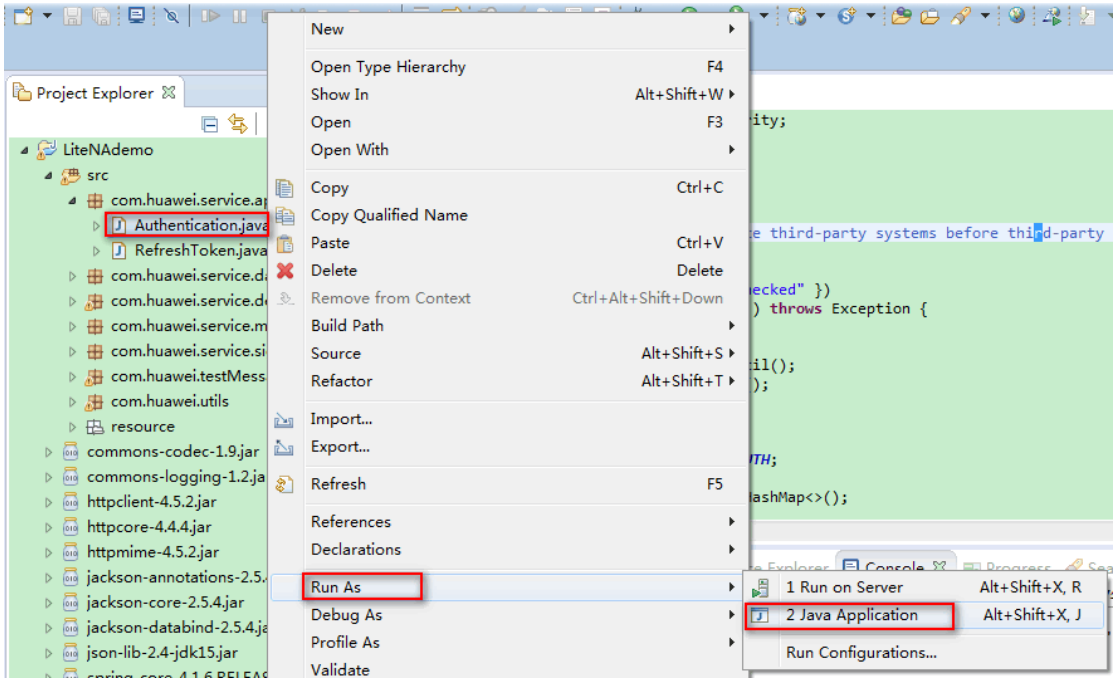
打开Constants.java，修改如下常量的值：



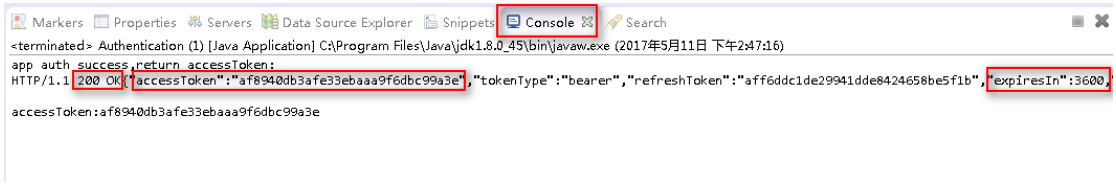
说明

如果样例代码的注解中出现“please replace”的字样，开发者需要根据实际情况修改常量或变量的值。

修改完全局常量后，如下图所示，右击Authentication.java，选择Run As -> Java Application，就可以在控制台看到响应消息的日志打印。



取得accessToken说明鉴权成功，得到的accessToken会在调用其他接口时使用到：



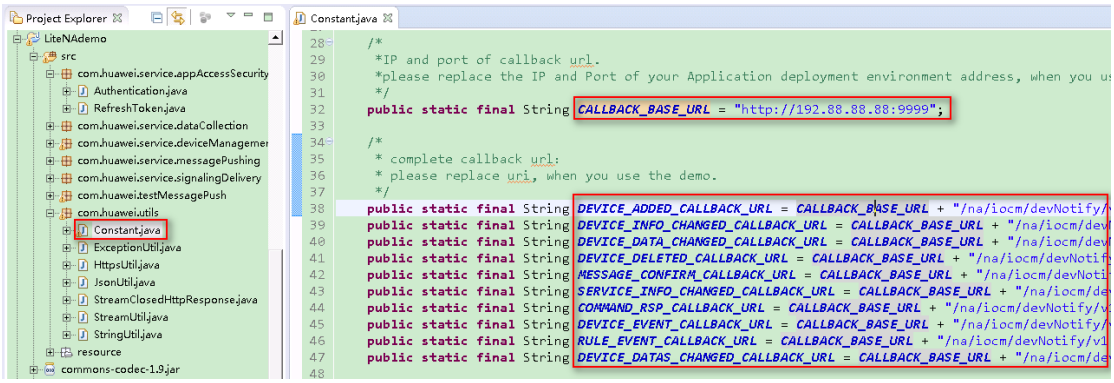
说明

- 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。还可以参考附录8.2章节的内容进行单步调测。
- accessToken会在expiresIn所标志的时间内过期，expiresIn的单位为秒。
- accessToken过期后需要重新获取。可以使用鉴权接口重新获取，也可以使用上一次鉴权得到的refreshToken来获取新的accessToken。Refresh Token接口请参考API文档和样例代码中的RefreshToken.java。
- Lite Demo中还提供了各接口调用的抓包示例，参见LiteNAdemo_https\src\resource\demo_TCP_message.json。

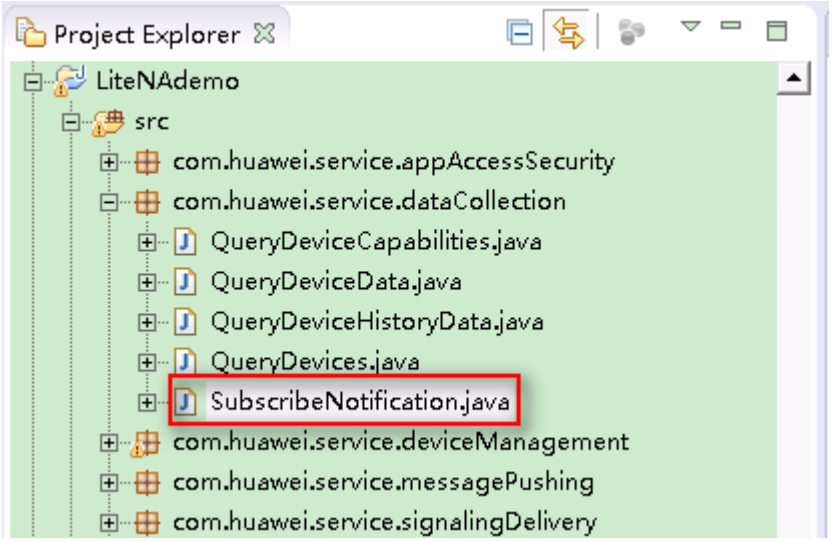
6 订阅平台消息

平台将设备消息分类推送给北向应用（具体可参考API文档中“Application 订阅平台数据”的接口说明），订阅则是将应用服务器的地址（即回调地址）设置到IoT平台，这样平台才知道将消息推送到哪里。

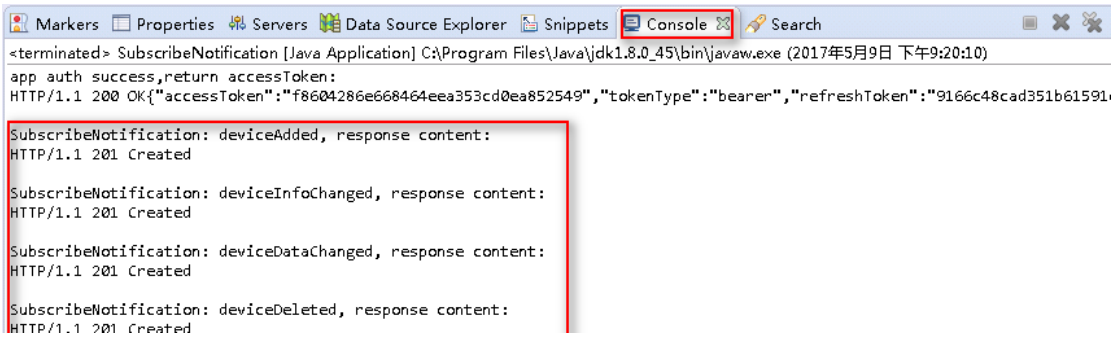
打开Constants.java，修改CALLBACK_BASE_URL常量中的IP和端口（回调地址中必须设置端口，否则会订阅失败）。下方不同类型消息的回调地址可以不同，也可以相同。开发者可以自行设计回调地址的路径。



右击Authentication.java，选择Run As -> Java Application，就可以在控制台看到响应消息的日志打印。



订阅的消息类型及响应如下：



说明

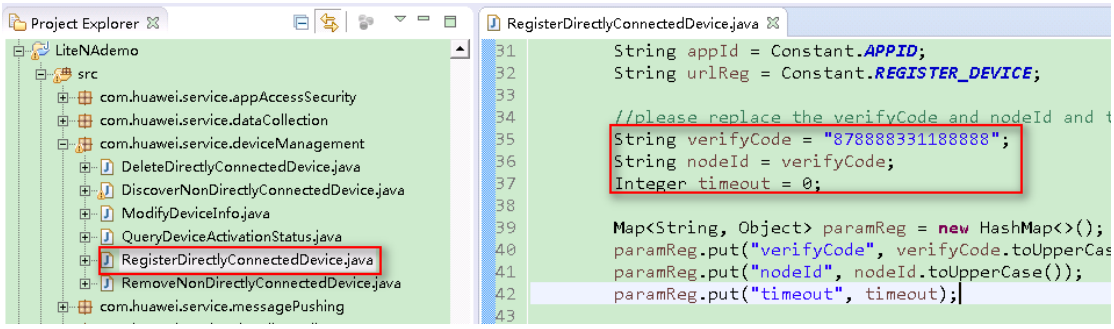
- 如需修改订阅的回调地址，在Constants.java类中修改各回调地址的值，再次订阅即可。新的回调地址会覆盖原来的回调地址。
- 订阅完成后，开发者可参考SubscribeNotifyResource.java搭建一个应用服务器来接收平台推送的Post消息。SubscribeNotifyResource.java仅供参考。如果需要在本地测试平台回调功能和查看回调内容，可以使用Lite Demo提供的类TestSubscribeAllNotification.java，下文在11 步骤中会说明。

7 注册直连设备

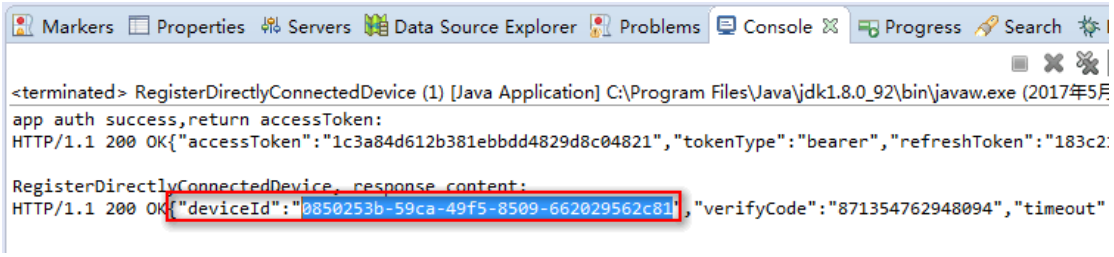
打开RegisterDirectlyConnectedDevice.java，修改verifyCode/nodeId和timeout的值：

- verifyCode/nodeId需要与真实设备的唯一标识符（IMEI或MAC）一致。如果使用的是模拟器设备，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，但不得与其他设备的verifyCode重复（本文使用模拟器设备）。
- timeout时间单位是秒，timeout的取值有如下几种情况

1. timeout = 0，注册的设备不会过期。
2. timeout > 0，真实设备必须在设置的时间内上线，否则注册的设备会过期而被IoT平台删除。如果不携带timeout，则默认过期时间是180秒
3. 在设备绑定成功后timeout字段不再起作用，设备不会过期。



同样，右击RegisterDirectlyConnectedDevice.java，选择Run As -> Java Application，就可以在控制台看到响应消息的日志打印。



说明

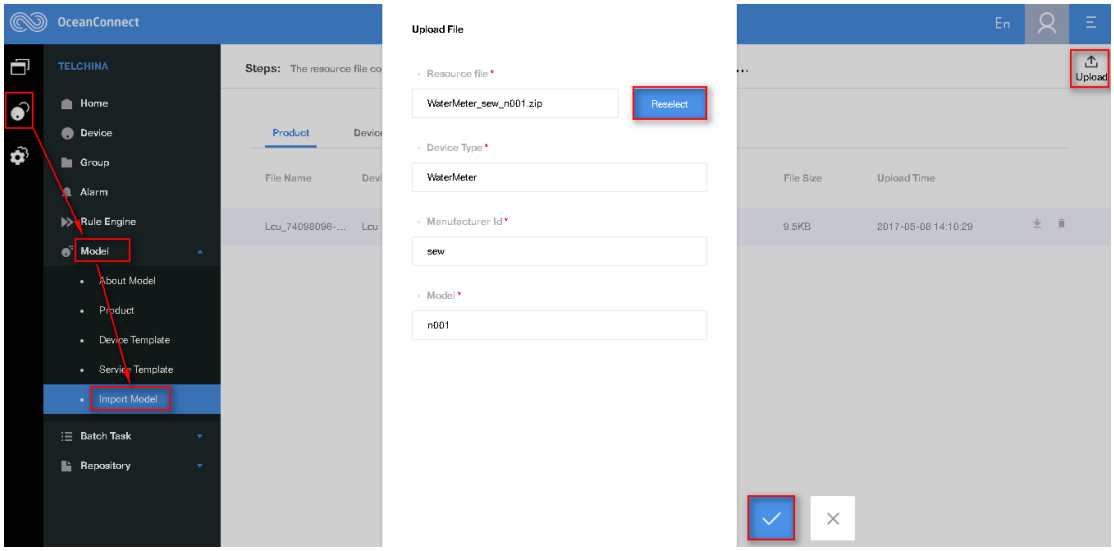
- 由于Lite Demo中没有建立数据库，需要手动拷贝并保存deviceId，以备调用其他设备相关的接口时使用。在设备未上线之前，再次调用注册设备接口则可刷新deviceId。
- 注册设备只是新增一个设备第一个步骤。注册设备需要与真实设备对接起来才能使用，否则，无法上报真实设备的数据。
- 设备注册成功后可以使用查询单个设备信息或按条件批量查询设备信息的接口查询设备信息。具体使用方法请参考QueryDeviceData.java、QueryDevices.java及API文档。

8 上传 Profile 资源包样例

登录SP portal后，在设备管理->模型->导入模型页面上传下载到的Profile资源包样例。

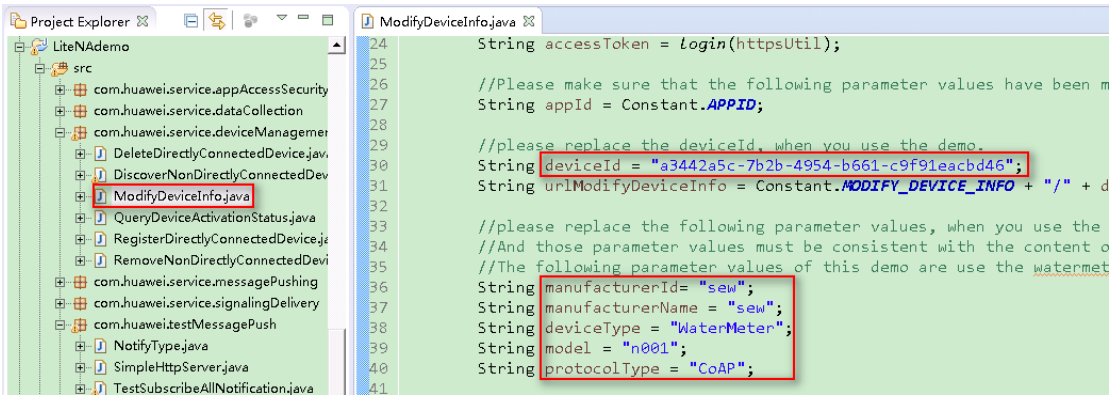
说明

本文档使用Profile资源包直接导入平台。开发者可自行设计上报数据和下发命令的字段，在设备模板和服务模板中进行建模。

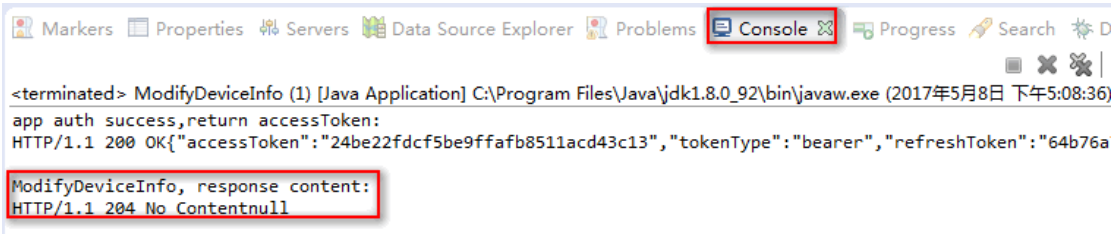


9 根据 Profile 修改设备信息

打开ModifyDeviceInfo.java，修改deviceId、deviceType、manufacturerId、manufacturerName、protocolType、model。其中deviceId是注册设备时得到的。deviceType、manufacturerId、protocolType、model是必须配置的参数，其值要与Profile中的定义保持一致。对于NB-IoT设备，protocolType必须为CoAP，另外三个字段可自行定义，Profile样例中给的值分别为WaterMeter、sew、n001。



同样，右击ModifyDeviceInfo.java，选择Run As -> Java Application，就可以在控制台看到响应消息的日志打印。



说明

- 此接口使用到的deviceId是注册设备时得到的。如果之前注册的设备过期了，需要重新注册设备，获得新的deviceId，否则修改设备信息时会出错。
- 平台根据manufacturerId和model来找到设备对应的Profile文件和编解码插件，所以设备信息的配置至关重要。
- 一个北向应用可以添加多种型号的设备，每种型号对应一个Profile。在修改了设备信息后，设备被赋予了Profile中所配置的设备能力。
- 设备修改信息成功后可以使用查询单个设备信息或按条件批量查询设备信息的接口查询设备信息。具体使用方法请参考QueryDeviceData.java、QueryDevices.java及API文档。

10 绑定设备（设备上线）

说明

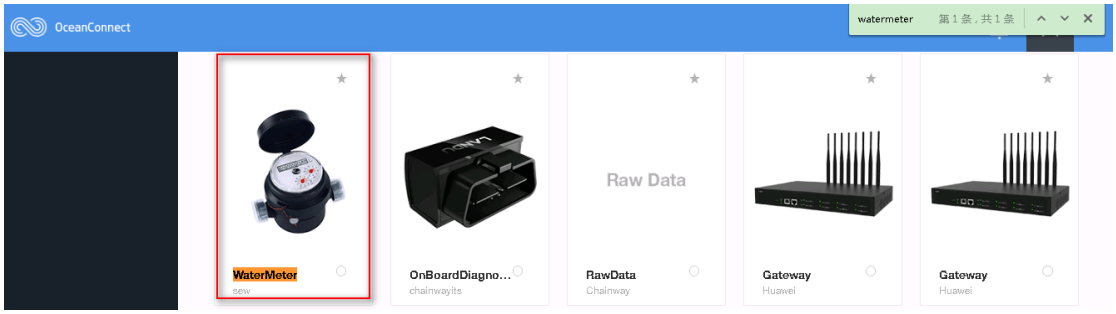
- 绑定设备，即设备首次上线的动作。设备上报的数据中携带设备标识符（IMEI或Mac地址等），经编解码插件解码成功并与注册的verifyCode/nodeId匹配起来，则绑定成功，设备状态就变成online。
- 使用真实设备需要使用AT命令将设备标识符和上报的码流设置到真实设备中。以下使用的是开发者Portal提供的模拟设备。

登录开发者Portal，在设备模拟器页面依次选择设备模拟器->系统产品->WaterMeter，如下图所示。

图 10-1 点击设备模拟器

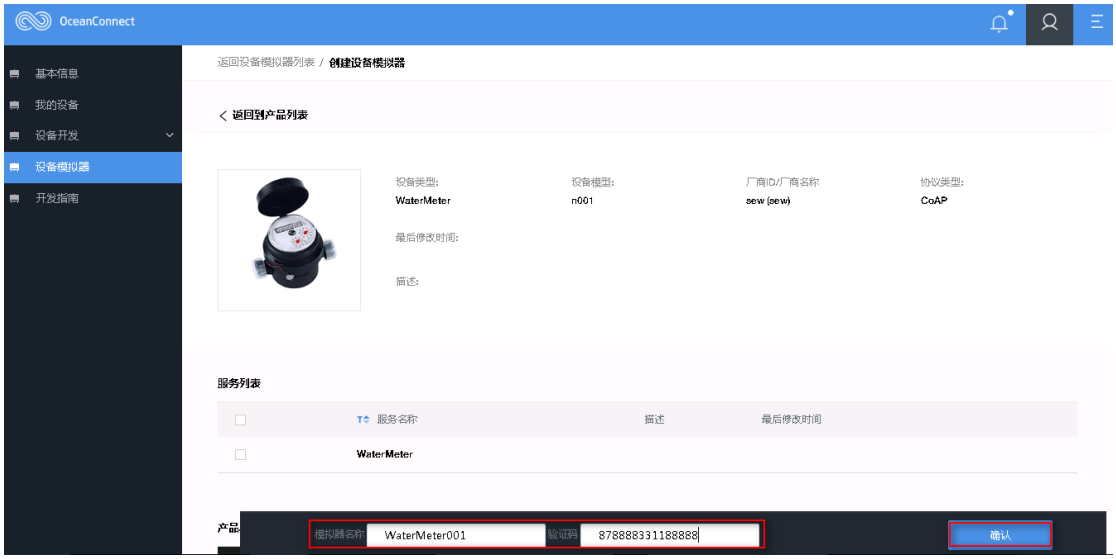
图 10-2 点击系统产品

图 10-3 点击 WaterMeter



在创建设备模拟器页面下方填入设备名称与验证码（必须与注册设备时使用的verifyCode保持一致），如下图。

图 10-4 填入设备名称与验证码



在设备模拟器列表找到刚添加的模拟器设备，点击该设备进入模拟器控制台。

图 10-5 找到设备

在模拟器控制台页面，选择Set State，将设置设备状态的按钮置为打开状态，则设备就上线了。

图 10-6 打开设备状态

设备绑定成功，如果应用订阅了绑定消息，则会收到平台推送的消息。

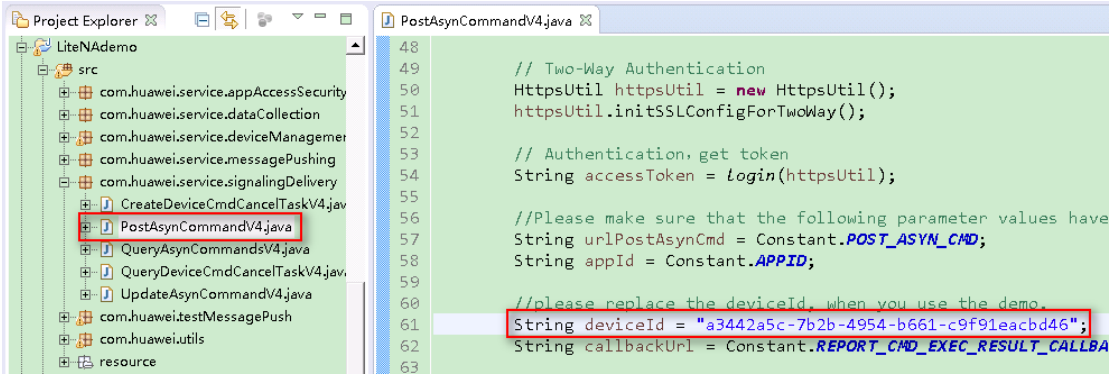
设备上线后还可以使用查询单个设备信息或按条件批量查询设备信息的接口查询设备是否上线。具体使用方法请参考QueryDeviceData.java、QueryDevices.java及API文档。

11 创建设备命令

使用LiteDemo下发一条设备命令。

打开PostAsynCommandV4.java，将修改deviceId为注册设备时得到的deviceId。

图 11-1 调用创建命令的 API 接口



右击PostAsynCommandV4.java，选择Run As -> Java Application，就可以在控制台看到调用接口的结果。

说明

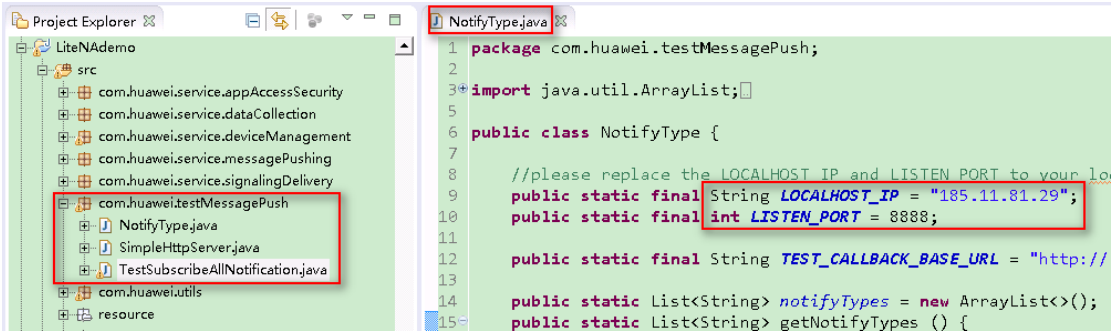
- 北向应用调用创建设备命令接口成功，只表示命令到达平台，不代表命令下发给设备。命令下发请参考Step 10。
- 如果应用需要知道命令执行结果，可以在调用创建命令接口时设置回调地址，平台会将命令执行结果推送到该回调地址。详见该接口的callbackUrl字段。

12 设备数据上报和平台命令下发

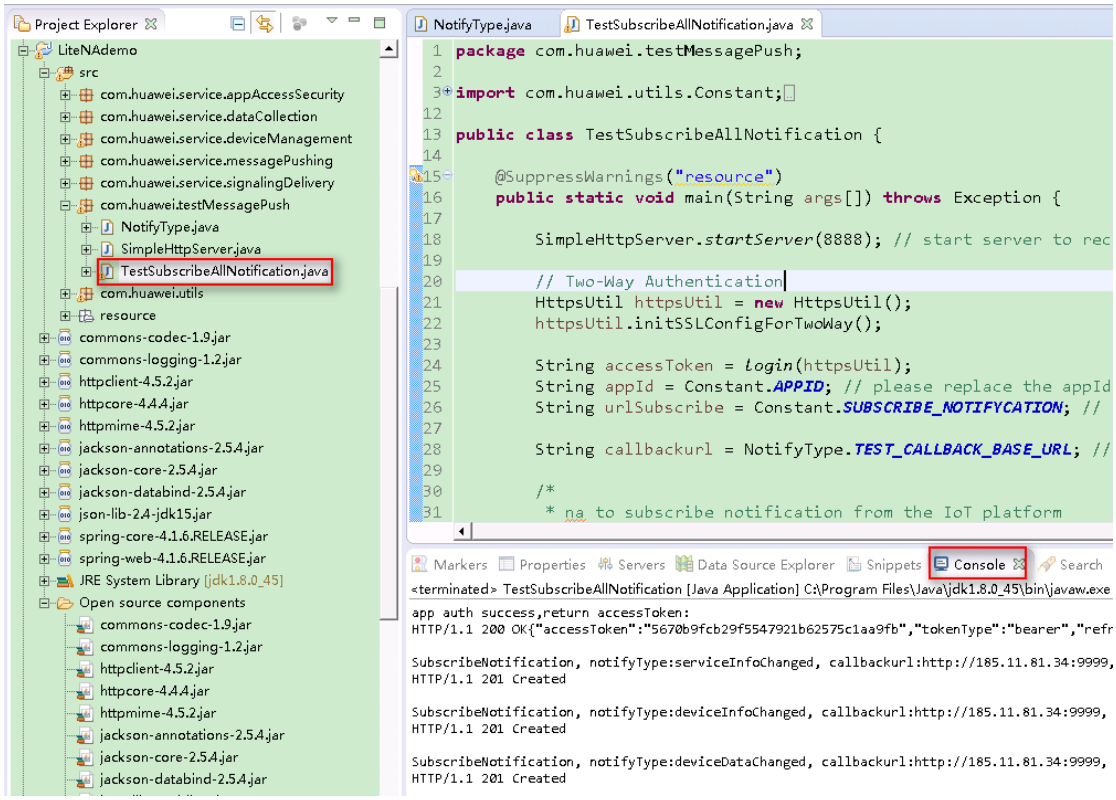
为了方便开发者了解平台是否已推送消息给应用，Lite Demo中提供了一个简单的HTTP服务器。打开NotifyType.java，修改TEST_CALLBACK_BASE_URL常量中的本地IP和端口。

说明

- Demo使用的简单的HTTP服务器只监听本地端口，TEST_CALLBACK_BASE_URL中不能添加路径。
- 设置的端口不能被本地其他程序占用。

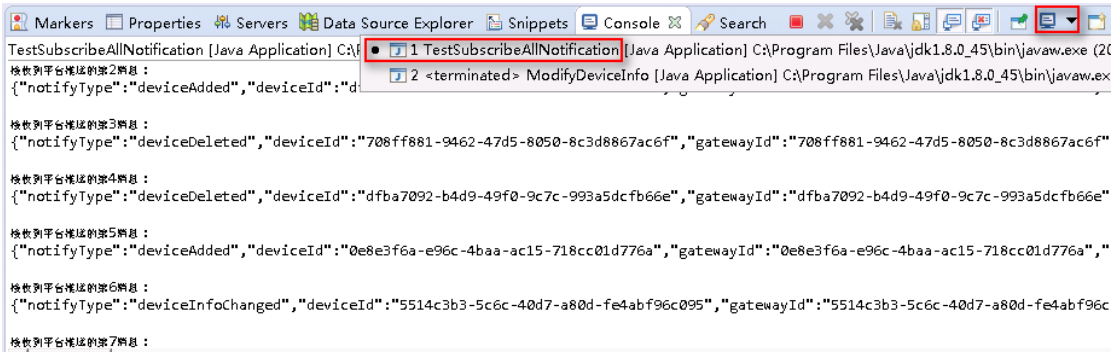


右击TestSubscribeAllNotification.java，选择Run As -> Java Application，就可以在控制台看到启动了端口监听和订阅的回调地址。



在开发者portal模拟器控制台页面，选择一个服务（如WaterMeter），填写相关字段信息，点击“确认”按钮。

在eclipse控制台就能看到平台推送给应用的消息（注意选择TestSubscribeAllNotification对应的页面）：



设备数据上报后，平台会将缓存的命令下发给设备。可以在开发者Portal设备模拟器的控制台看到平台下发的命令。

13 更多内容

更多平台开放能力请参见**OceanConnect开发者中心**。

14 附录

Postman 是一个 Chrome 扩展工具，提供功能强大的 Web API & HTTP 请求调试，可以用来调用API接口。因此可以用来模拟APP server，提供与LiteDemo相似的功能，并且不需要配置IDE。

14.1 使用 Postman 测试平台北向接口

使用本方法的最大好处是不需要任何代码开发就可以快速调测IoT平台北向接口。如果您不使用Postman或者HTTP协议测试接口可以跳过本步骤。

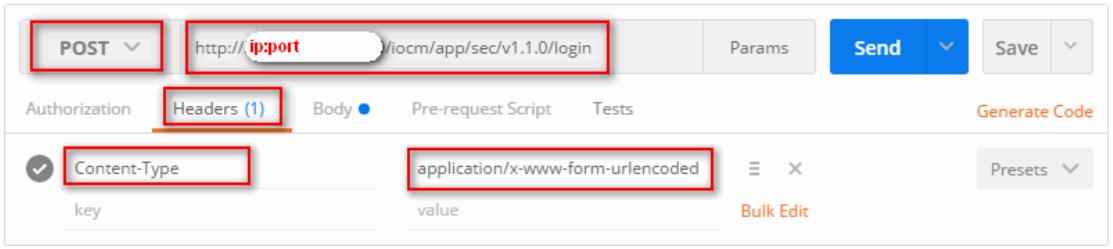
由于最新版本的Postman不支持https接口测试，此方法只适用于http协议的接口测试。

在使用本方法前，您需要在MAR流程中说明需要使用HTTP进行测试，并附上应用服务器的公网IP，华为工程师会将您的公网IP配置到IoT平台的HTTP白名单中，并将HTTP协议使用的IP与端口发送给您。

下载并安装Chrome浏览器，并安装Chrome的扩展程序Postman。Postman是免费的，在Chrome网上商店即可下载安装Postman，可以跳过账号注册的步骤。安装完成后，可以在Chrome的扩展程序中启动Postman：

鉴权接口

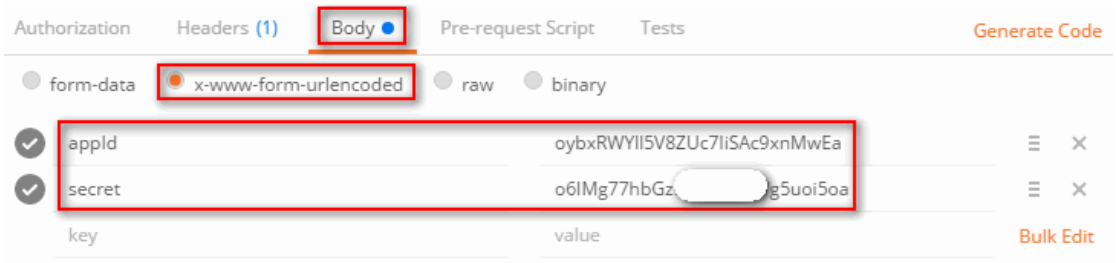
配置http方法、URL和头域



如上图所示，配置http方法、URL和头域，将URL中的https修改为http，并且手动将接口中的server:port修改为平台ip及http端口。

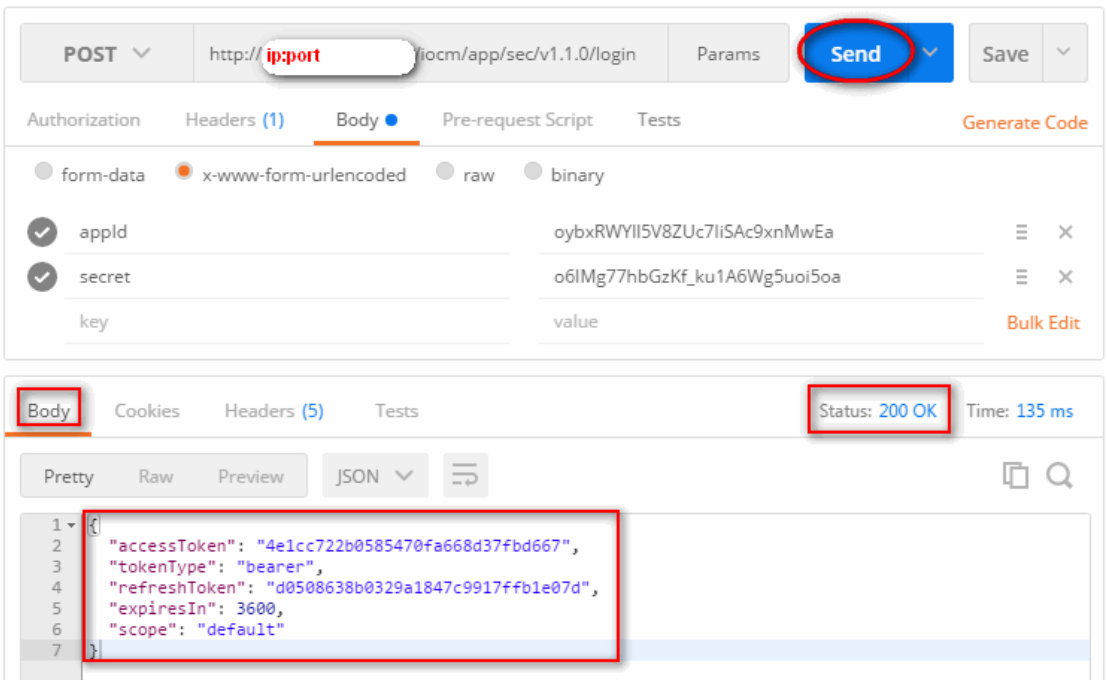
注意：URL和头域等参数请从API示例中拷贝，不要手动输入。

配置Body



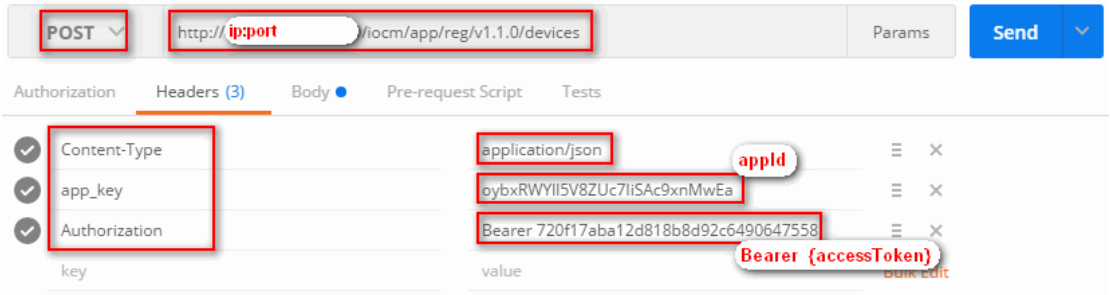
返回的消息

点击Send按钮，就能在下方看到返回码及消息内容。将返回的内容（尤其是accessToken）复制并保存起来，以备后续调用其他接口时使用。

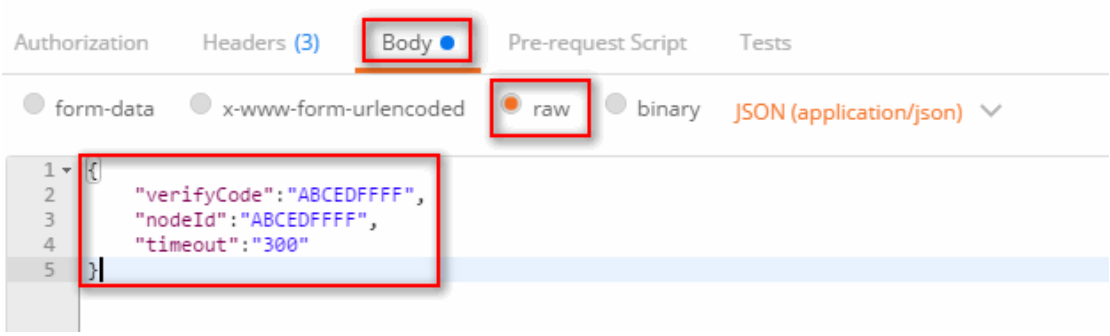


注册直连设备接口

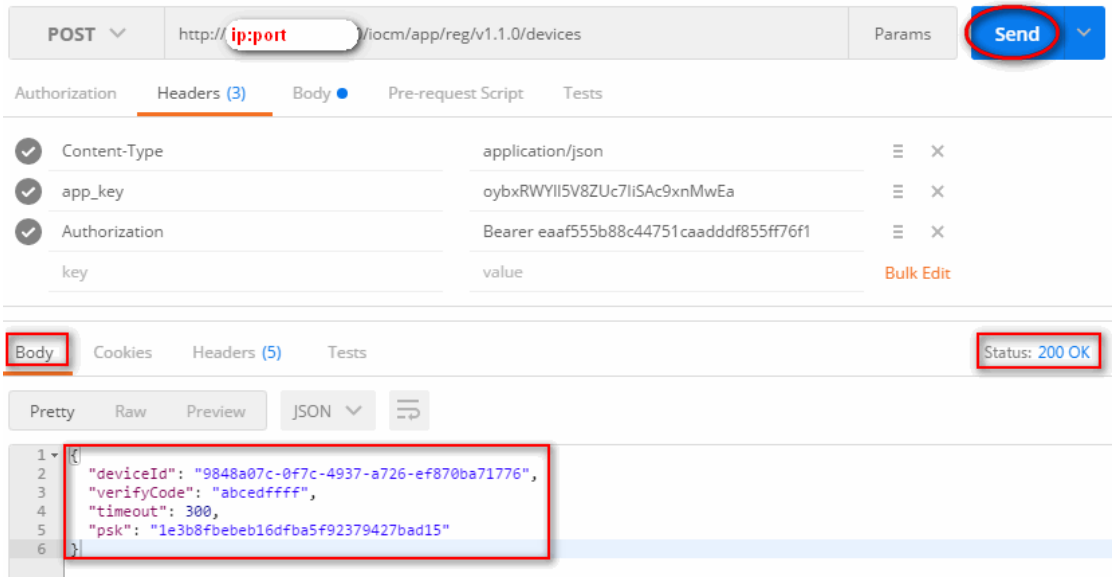
配置http方法、URL和头域



配置Body



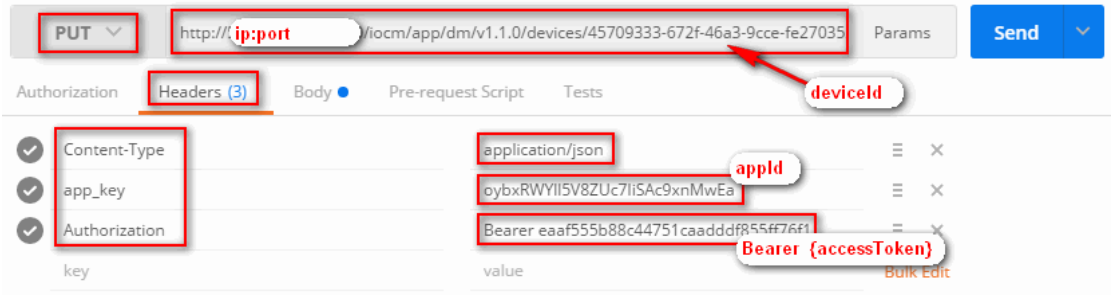
返回的消息



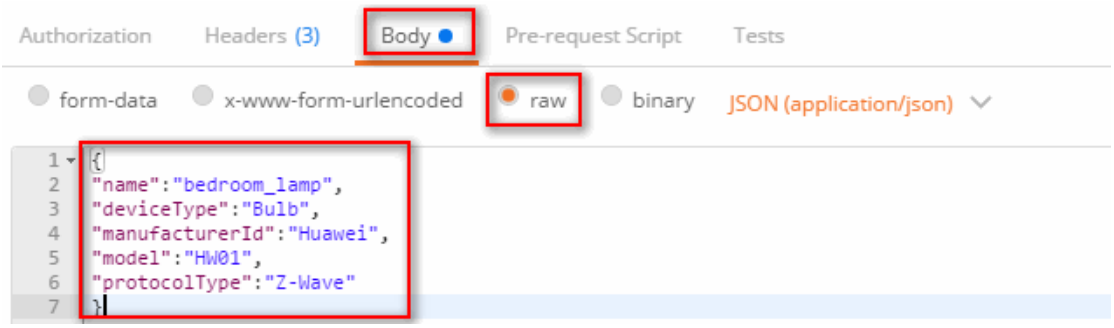
将返回的内容（尤其是deviceId）复制并保存起来，以备后续调用其他接口时使用。

修改设备信息接口

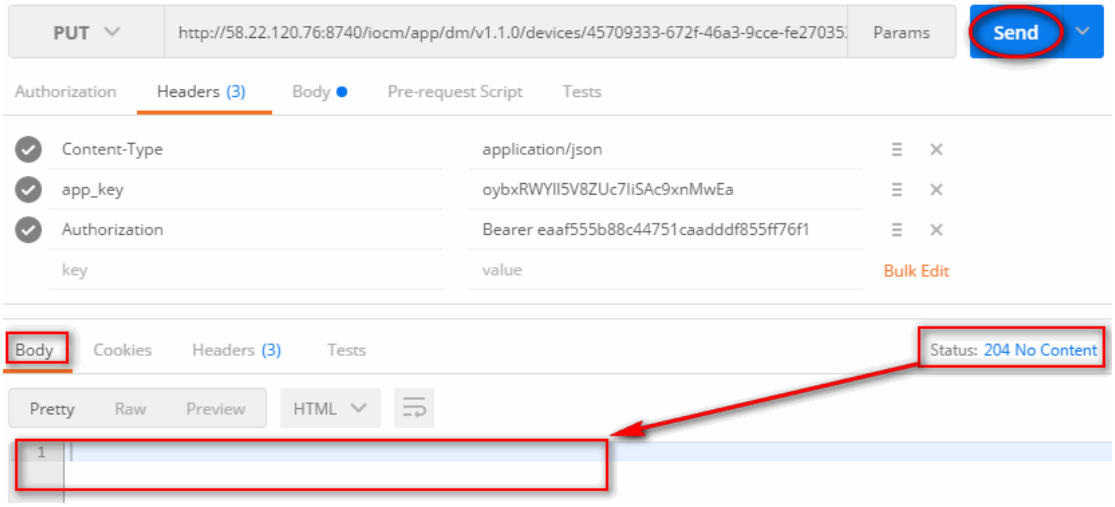
配置http方法、URL和头域



配置Body



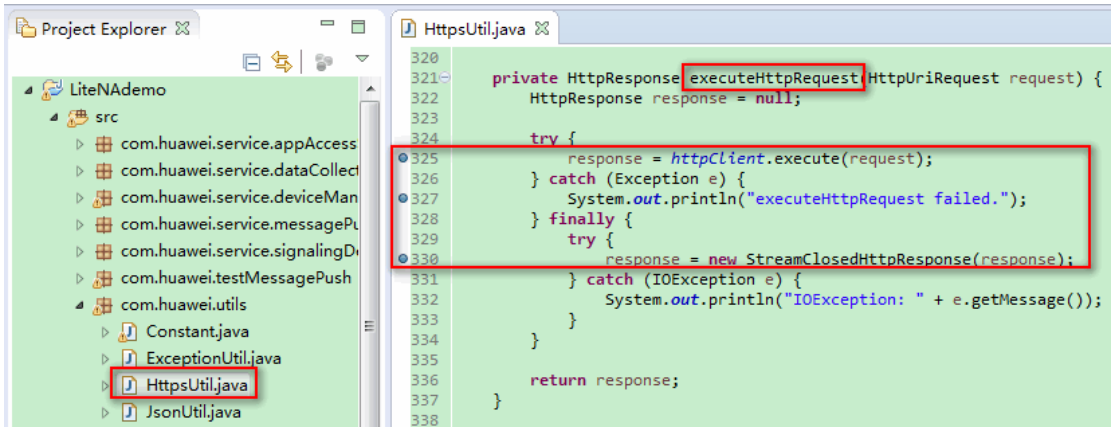
点击“Send”，就可以看到返回的消息



14.2 单步调测、查看消息内容

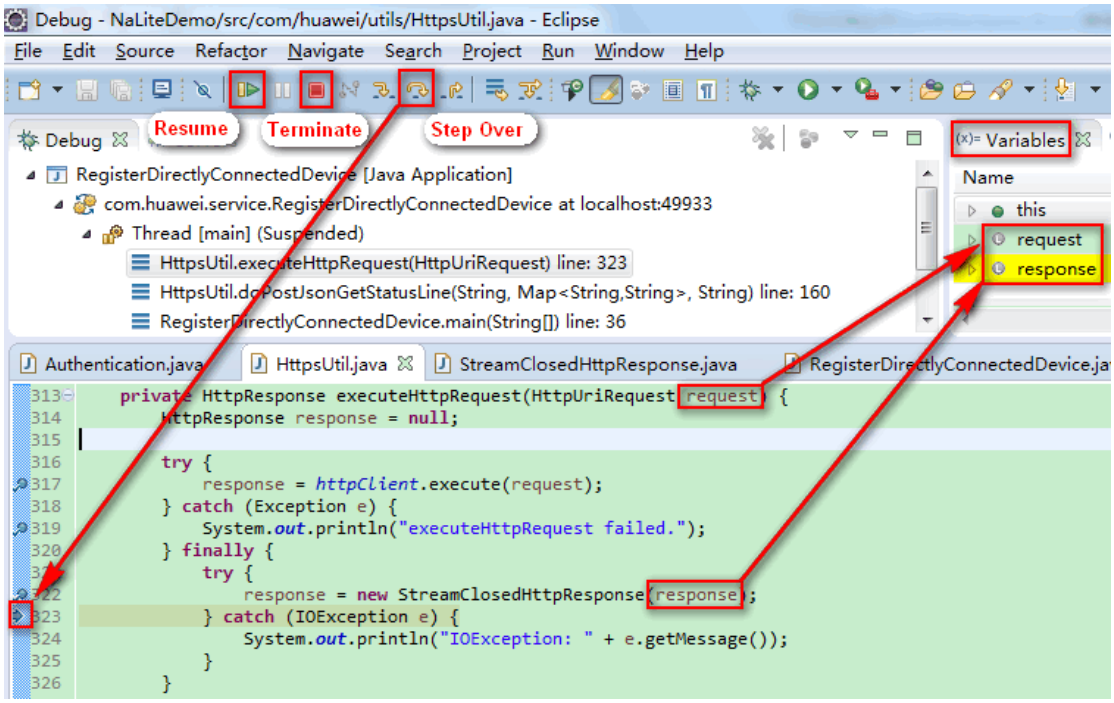
为了更直观地看到应用程序发出去的消息及平台的响应消息，以下方法使用了Eclipse的断点调试方法。如果使用Postman测试接口，请参考附录第一小节的内容。

先在最终发出http/https消息的代码处打上断点，以便调试，如在样例代码HttpsUtil.java中的executeHttpRequest方法内打3个断点（请根据您代码的实际情况打断点）：

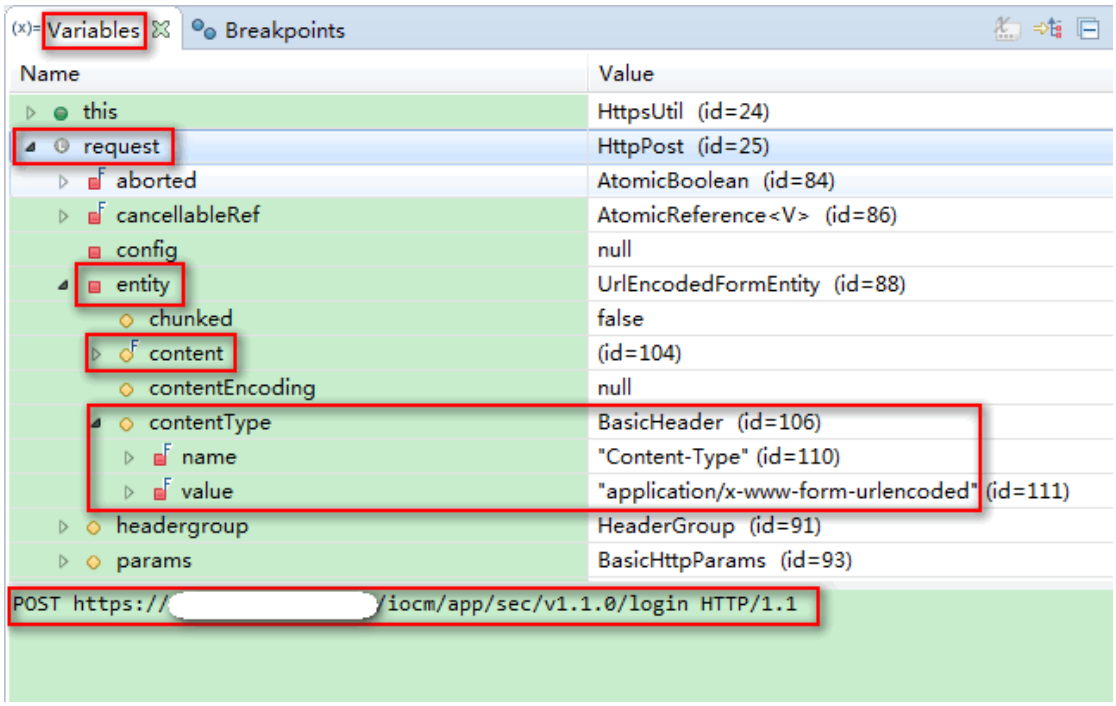


右击要调测的类（Authentication.java），选择Debug As-> Java Application（根据您建立的工程类型进行选择）。

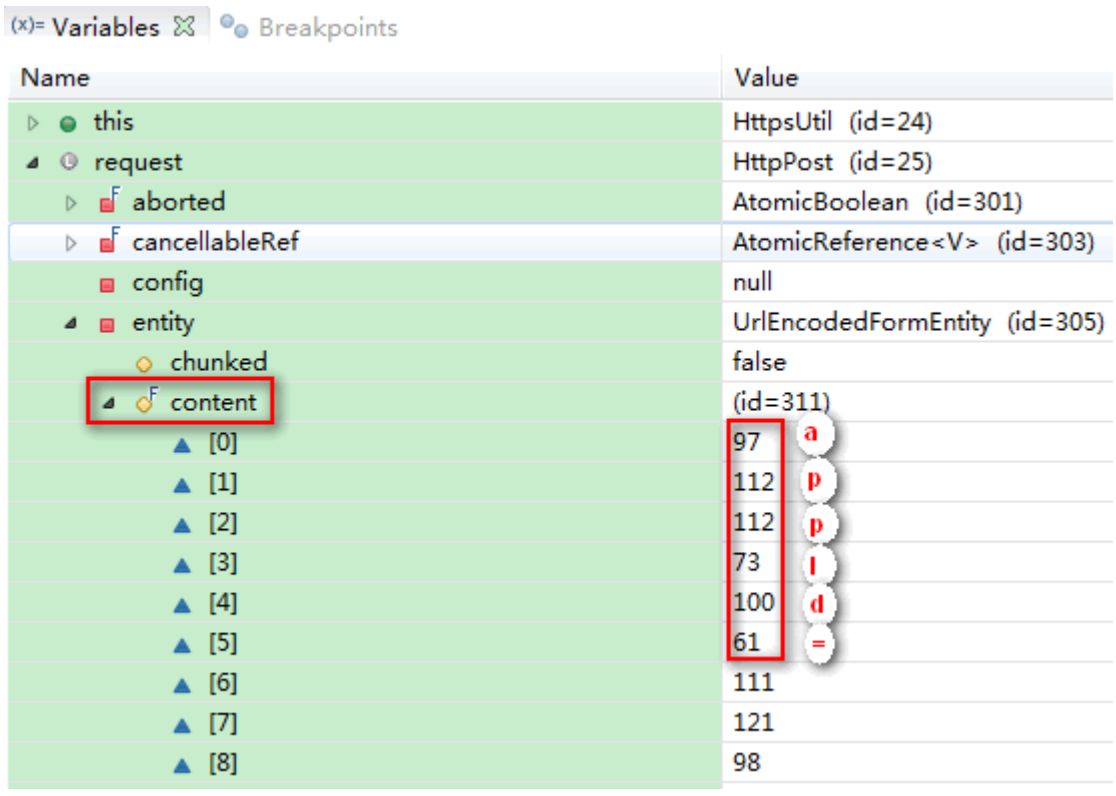
当程序运行到我们设置的断点时就会停下来，使用Step Over按钮运行到如图所示的位置，此时可以在Variables窗口看到相应的变量的内容，包括发出去的消息及平台的响应消息。



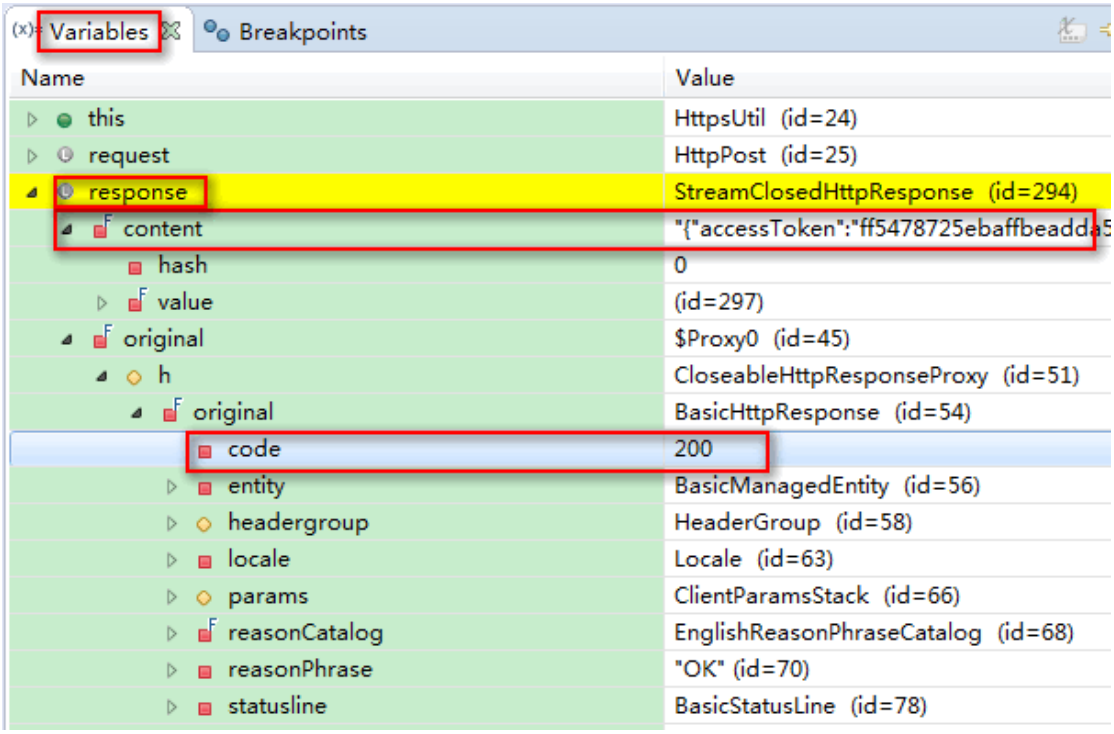
在Variables窗口中展开request变量。选中request变量时，可以在下方内容展示区看到应用程序发出的请求的URL。在entity中可以看到发出去的消息内容。



appId与secret在content字段内，使用了十进制的ASCII码表示，所以不能一眼就看出所带的字符串内容。可以对照ASCII码表将其一一转化为字母和符号。

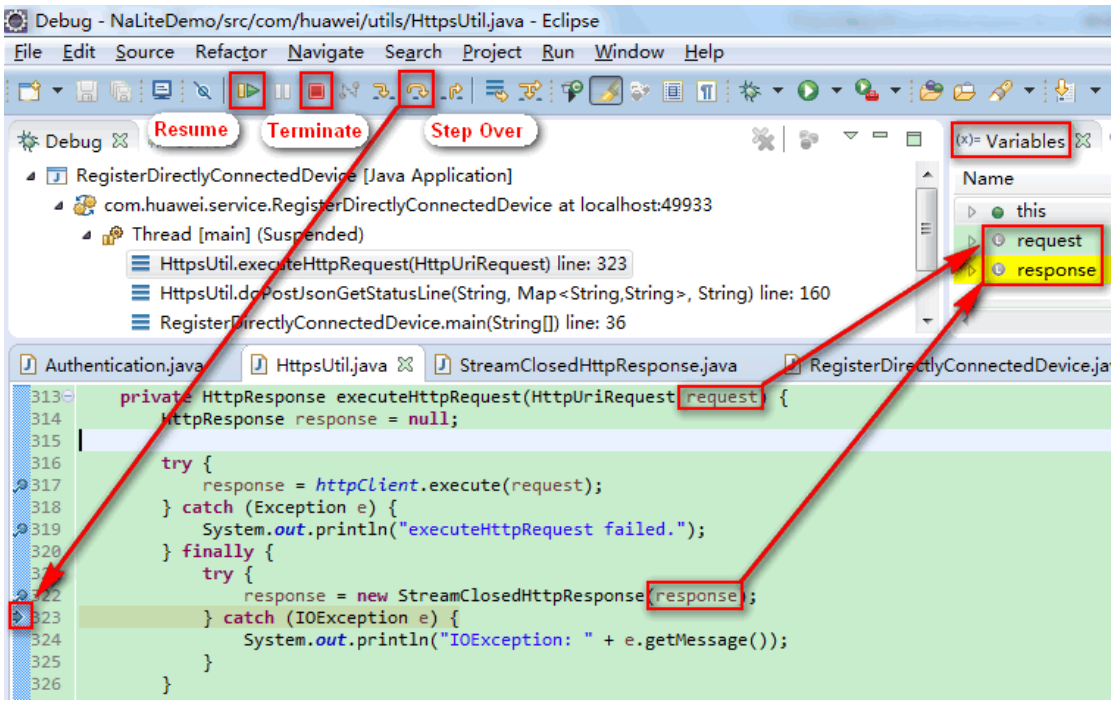


Response内容如下：

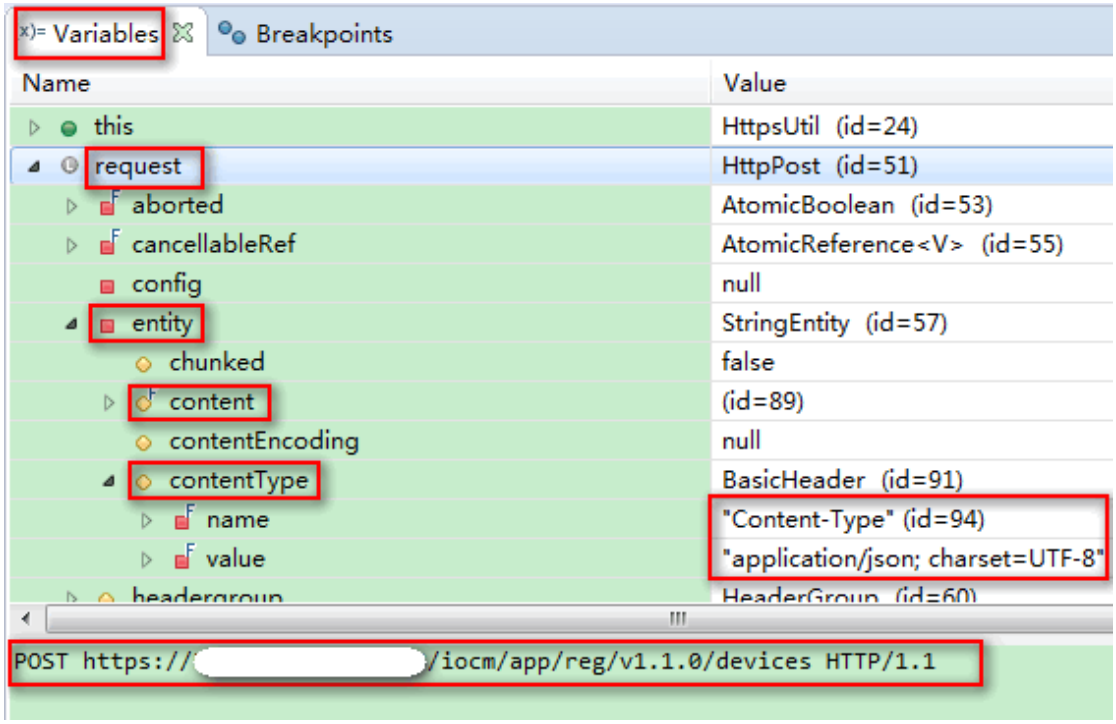


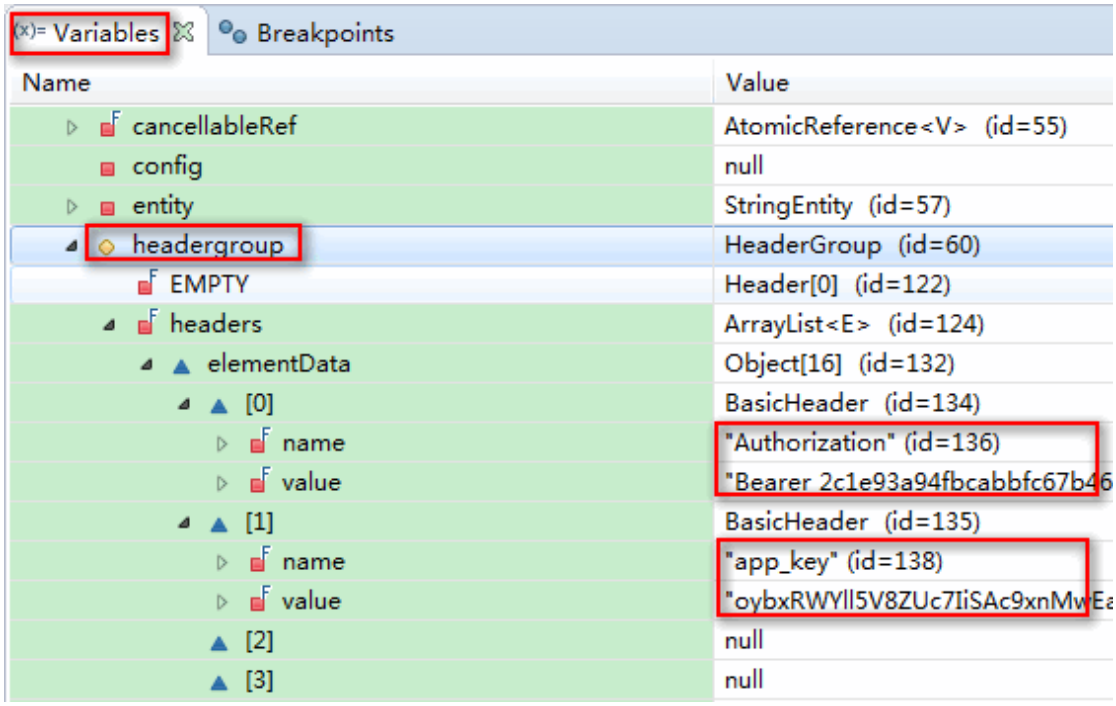
说明

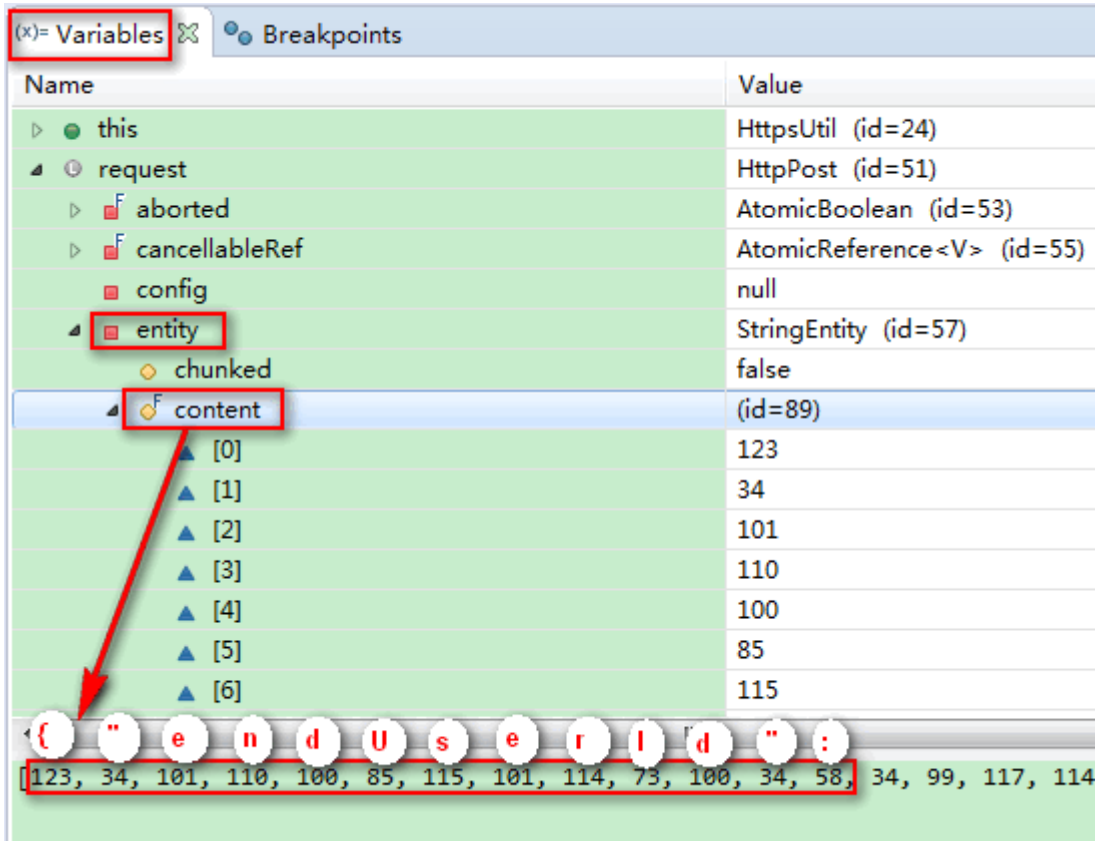
如果调用的类不是Authentication.java，由于第一次发的消息为鉴权消息，所以我们使用Resume按钮使程序向下运行，直到断点再第二次命中executeHttpRequest中发送http/https消息的地方，使用Step Over按钮运行到如下图所示的位置再查看变量内容。



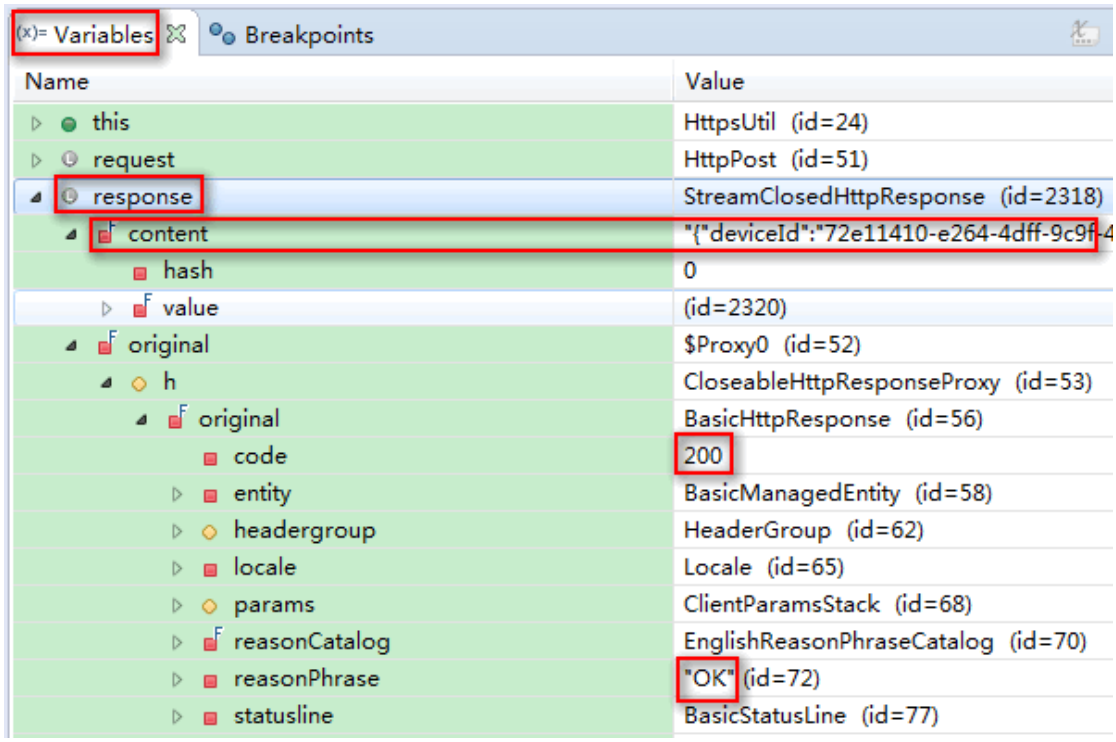
在Variables窗口中展开request变量：







Response内容如下：



15 应用集成调测典型问题分析

15.1 Profile 文件的写作问题

Profile文件的编写是较容易出现问题的地方，本节通过介绍常见的问题，从而增加开发者对Profile文件的认识，进而避免问题的发生。

Profile最快的写作方法就是根据Profile文件模版直接修改。Profile文件的格式多，字段多，在模版的基础上修改可以避免部分问题。

根据与伙伴APP server对接调测过程，总结问题如下。

如何理解 Profile 文件服务的定义

Profile写作的遇到的第一个问题就是如何定义一个服务。当前服务的定义没有明确的规定，但最好保证每个服务是一个单独的功能单元。下面以例说明。

假设一款智能彩灯可以改变颜色，如何定义它的服务。由于Profile文件是一款设备的“全量”定义与描述，因此描述这款彩灯还需要考虑其亮度、开关状态、一些基本物理参数等。以下几种拆分服务的定义都是可以的：

1. 仅定义一个服务

   所有彩灯的描述都放在一个服务里面。仅定义一个服务一般用于描述较简单，业务不复杂的设备。

2. 定义四个服务，分别描述其亮度、开关、颜色、基本物理属性

   如果物理属性APP server不关心，也可以不定义。

3. 定义两个服务，服务一定义其基本特性，服务二定义其智能特性

彩灯是智能彩灯，APP server可以通过命令控制彩灯，将智能功能归纳为一种服务也可以。

Profile 文件的格式问题

Profile文件的格式也是比较容易出错的地方。本小节通过对Profile文件格式的梳理，加深Profile编写者的理解。

Profile文件的格式包含四种，分别是json格式、命名格式、打包格式和Profile文件对字段的描述格式，分别描述如下：

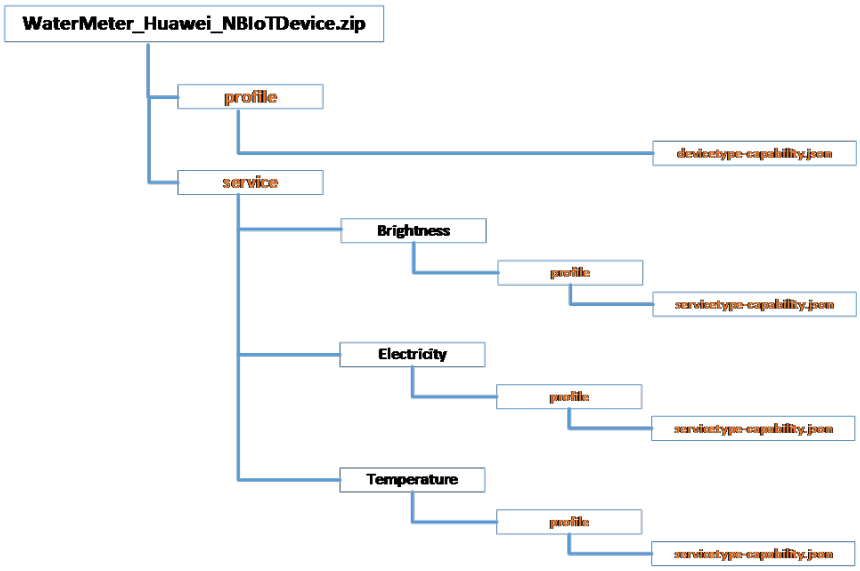
1. Profile文件的文件格式都是json。

   devicetype-capability.json和servicetype-capability.json首先要符合json的格式。通过网上搜索引擎查询在线json格式校验网站，建议完成Profile文件后可以通过这些网站进行校验。通过校验可以排除掉一些比较基本错误，如携带中文字符、缺少标点符号、额外添加非法内容等。

2. Profile文件有一套自己的命令格式。

   如图14-1所示，比较重要的注意地方是①图中标黄的文件名不可以改变；②Profile的命名需以deviceType_manufacturerId_model的方式命名。

图 15-1 Profile 文件的目录



3. 打包格式是也是较多出现问题的。

打包格式出现过的问题如下：

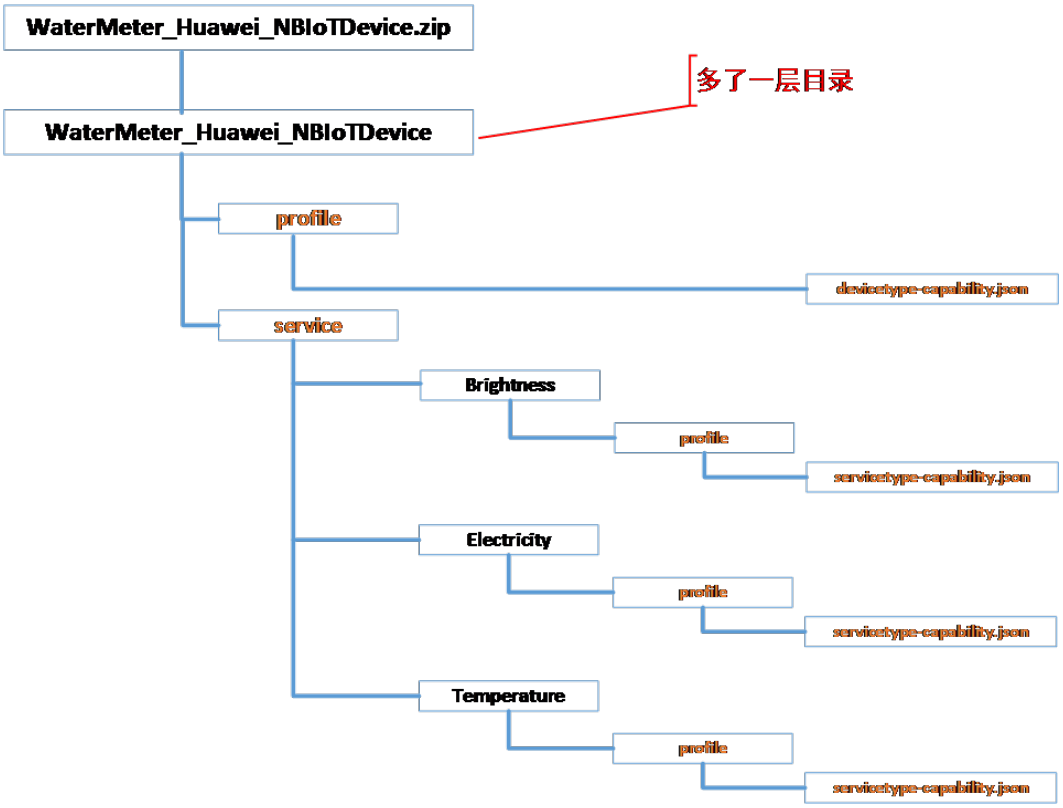
1. 压缩包没有以zip格式打包

   有伙伴以rar打包后，手动更改后缀名，导致平台解压失败。

2. 压缩包的层级目录被可变

目前网上的打包工具打包后，目录结构如图14-2所示，显然多了一层目录。

图 15-2 错误的 Profile 文件的目录



1. Profile文件中对每个字段的描述都是多维度的，因此也有自己的格式。

出现过的问题如下：

1. “任意”删除或增加描述字段的维度。

   例如，描述命令中的一个字段有九个维度，分别是paraName、dataType、required、min、max、step、maxLength、unit、enumList，九个纬度构成对字段的全量描述。某一维度可能对特定字段无意义，比如某字段的dataType是int，那么这中间的maxLength维度就无意义了，这时请根据《华为IoT 设备能力描述文件profile开发指南》对其进行赋值。每一个命令中的字段都是这九个维度描述的，不能因为某一维度某些时候无意义就删除这些维度。而设备上报业务数据字段是通过十个维度来描述的，与命令字段的九个维度同理。建议在Profile文件模版的基础上修改可以避免这种问题。

2. 命令中paras字段赋值为null。

   命令需要定义具体的参数，否则该命令不允许下发。

3. 命令字段的描述维度与业务数据字段的描述维度搞混了。

描述一个命令字段需要paraName、dataType、required、min、max、step、maxLength、unit、enumList九个维度。描述一个业务数据字段需要propertyName、dataType、required、min、max、step、maxLength、method、unit、enumList共十个维度，不能搞混了，否者Profile文件校验失败。

Profile 文件字段的取值说明

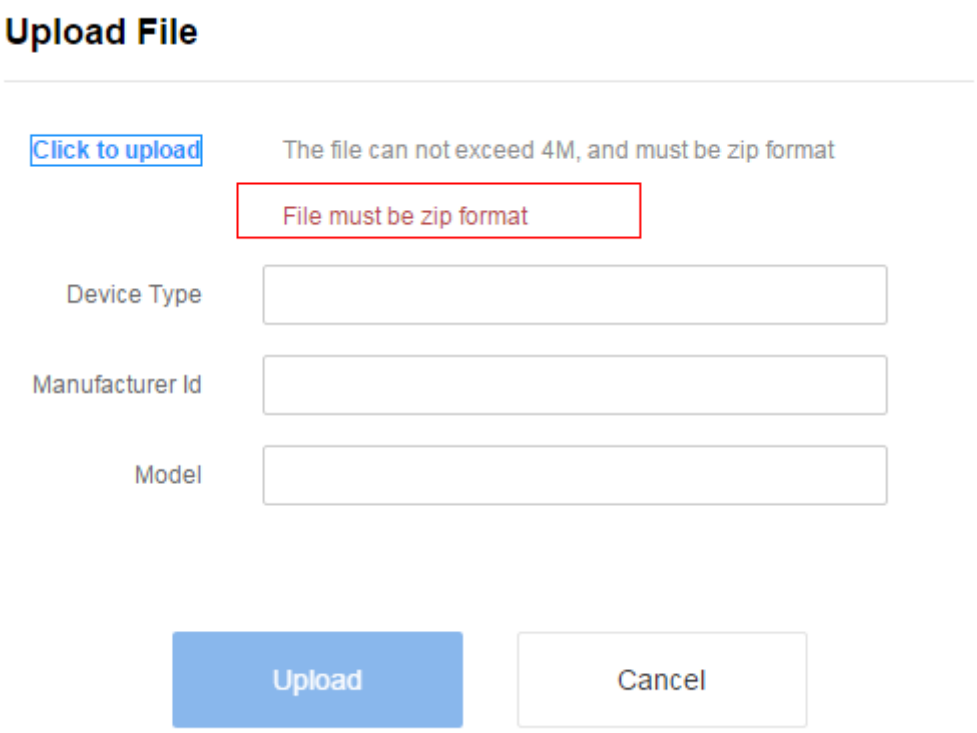
Profile文件中一些字段的取值经常有较多伙伴咨询，这些字段的具体描述请见《华为IoT 设备能力描述文件profile开发指南》，不做赘述。现总结如下：

字段名	位置	描述
option	devicetype-capability.json	以例说明。假设一款智能彩灯主要用途是用来阅读，并且可以改变颜色来增加趣味性。我们定义了三个服务，分别是Switch/Brigthness/Color。其中Switch是开关，彩灯要想被使用必须打开，因此它是阅读的必要条件，option可以取值为Mandatory；发光发亮是彩灯的基本属性，若不能发光发亮彩灯就不能用来阅读，因此option可以取值为Master；颜色并不影响是否可以用来阅读，只是附属的属性，因此Color可以定义为Optional。
dataType	servicetype-capability.json	本字段非功能性字段，仅是描述性字段。不必严格区别某数据的确切类型，如float还是double。

Profile 文件上传问题

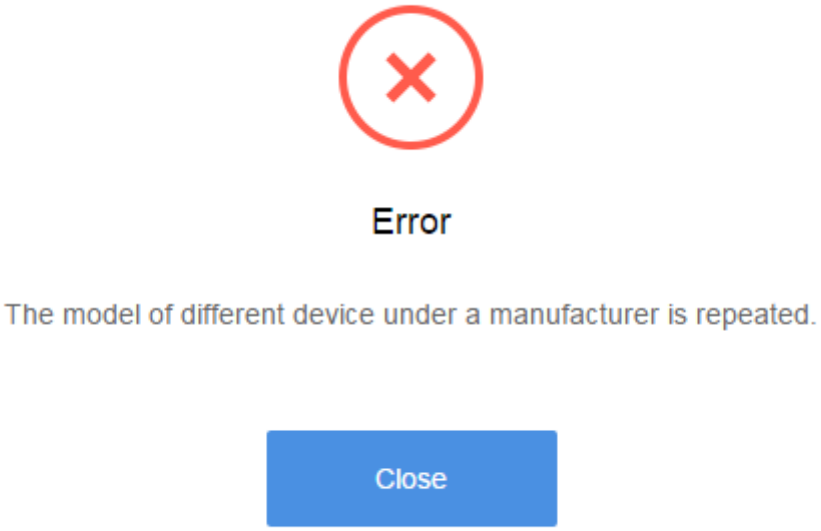
在Profile文件编写完成后需要上传到平台上，在SP portal上传的时候可能会报错，下面整理了常见上传报错并给出了指导的解决办法。

1. File must be zip format.



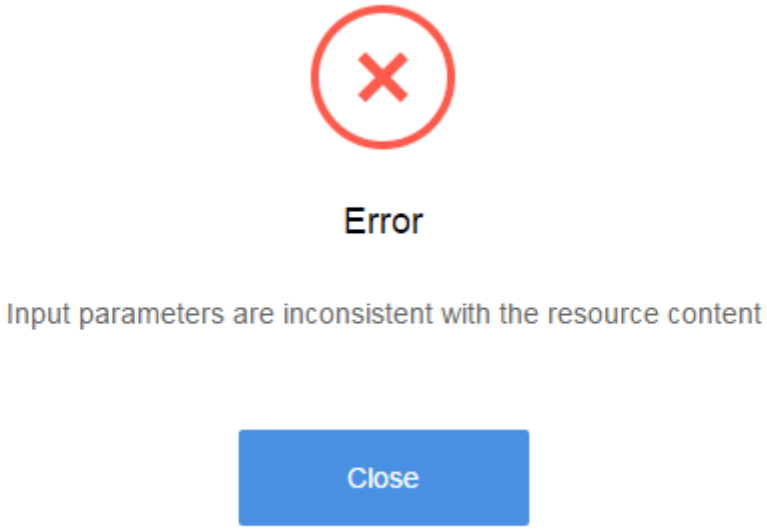
解决办法：请核对Profile文件压缩包是否zip格式，不支持其他压缩格式。

2. The model of different device under a manufacturer is repeated.



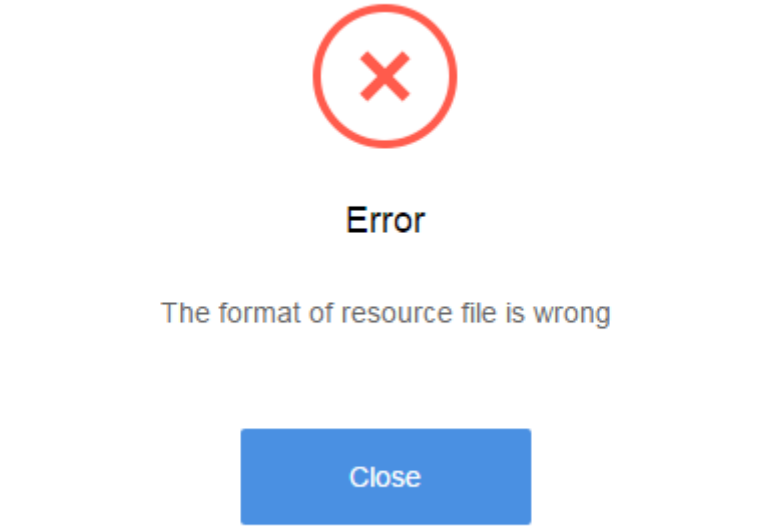
解决办法：请核对待上传的Profile文件中的{model，manufacturerId}这个组合是否已经存在于该应用了，也就是说一个应用中，不允许出现两个设备的model和manufacturerId都一样。

3. Input parameters are inconsistent with the resource content.



解决办法：这个错误可能是由于Profile名字中的deviceType、manufacturerId、model参数的取值与devicetype-capability.json中的定义不一致，请核对。

4. The format of resource file is wrong.



解决办法：

a. 对比《**华为IoT_设备能力描述文件profile开发指南**》，确认文件中是否有缺少或增加了一些字段；

b. 确认是否文件中有非英文字符；

c. 在json校验网站上核对每个json文件是否合法。

15.2 接口的调用问题

关于接口的调用我们在开发者社区上发布了**LiteDemo**，建议在调用API接口前或调用出现问题时运行一下liteDemo中的相关接口，通过对比解决问题。

API接口的调用需要一定http协议的知识，这样不仅方便进行接口的调用，而且可以快速进行问题定界。比如调用接口返回http status code为400，就需要排查是否请求结构体拼装的问题了。

在开发者论坛中总结了几个帖子，以供参考，如下：

1. **IoT平台开放给应用服务器的所有API接口的调用、需要注意的地方、抓包**
2. **OceanConnect错误码参照表大全**

有可选参数的接口调用问题

由于部分接口的参数需要根据业务场景选择是否携带，故将其设置为可选参数。但当可选参数的字段名拼写错误，接口的返回结果仍然是调用成功，不会提示错误。因此在使用可选参数时请注意拼写。常见的问题如下。

1. 修改设备信息接口model或manufacturerId拼写错误。

   特别是manufacturerId这个参数由于字符较多，非常容易拼写错误。若这个两个参数拼写错误，会造成平台找不到该设备的Profile文件，进而影响业务处理。

2. 创建设备命令接口callbackUrl拼写错误。

由于并不是所有开发者都要求平台向APP server推送命令消息，因此callbackUrl是可选的。callbackUrl可能被误写做callbackurl或callbackURL。callbackUrl拼写错误会导致命令的消息无法从平台推送到APP server。

15.3 订阅消息推送问题

消息的推送是平台调用厂商APP server提供的callbackUrl，因此常见问题如下：

1. callbackUrl不合法。

   请参考http的url的要求。

2. 网络不可达。

   排除本问题一般需要华为支撑人员的协助，需要从平台侧查看是否网络不可达。

3. 厂商服务不可用。

   若厂商的APP server服务不可用，推送不成功，可以根据平台侧获取推送消息返回的http状态确定。不过建议厂商完成APP server的开发后，检测一下服务是否可用。

4. 平台未配置APP server证书

   平台向厂商APP server推送https消息，需要平台配置ssl证书。配置证书需要平台支撑人员的协助。

5. APP server单向认证和双向认证

若证书配置的是单向认证，而APP server开启的双向认证，也会导致推送失败。因此在对接推送前，一定要和平台的支撑人员交流清楚。