

**本科实验报告**

|  |  |
| --- | --- |
| 课程名称： | 计算机网络基础 |
| 实验名称： | 基于LinkLab的远程物联网应用开发 |
| 姓 名： | 张佳瑶 |
| 学 院： | 计算机学院 |
| 系： |  |
| 专 业： | 软件工程 |
| 学 号： | 3170103240 |
| 指导教师： | 高艺 |

2019年 10月 1日

**浙江大学实验报告**

实验名称： 基于LinkLab的远程物联网应用开发 实验类型： 编程实验

同组学生： 实验地点： 计算机网络实验室

# 实验目的

* 熟悉LinkLab物联网远程实验平台；
* 熟悉TinyLink 语言；
* 熟悉阿里云IoT Studio平台；
* 掌握MQTT协议；

# 实验内容和原理

* LinkLab系统简介

传统的物联网实验需要学员在本地配置开发环境、购买并连接设备，实验受时间和空间的限制较大。

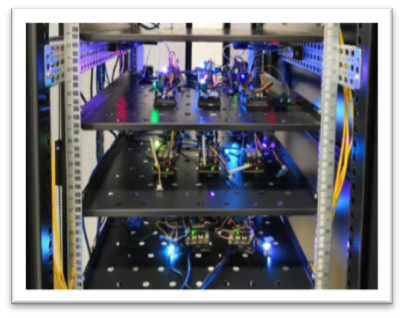


图2.1 LinkLab嵌入式设备

LinkLab物联网远程实验平台（http://linklab.tinylink.cn）采用“webIDE + 云端编译 + 远程烧写”的开发方式，学生可以基于LinkLab提供的在线开发环境编写代码，并在云端完成代码的编译，之后将代码烧写到远程的设备中，大大简化了实验流程。LinkLab集成了一套完备的物联网实验自动测试系统，基于远程设备，可以自动测试学生编写的设备端代码是否正确。

怎么使用LinkLab物联网远程实验平台？

第一步：注册账号和登录

账号注册由老师统一为学生注册。用户ID为学生学号，密码由老师统一设备并发放。学生在收到账号和密码后进入平台登录页面登录系统。



图2.2 LinkLab登录

第二步：平台主页

使用注册好的账号登录平台，进入主页面，向下滚动找到实验题列表，如下图：



图2.3 LinkLab主页

每一个实验题都由实验题标题、实验题简介和开启按钮组成，点击“开启”按钮可以进入WebIDE页面。

第三步：WebIDE的使用入门

在第二步中选择实验题并点击“开启”按钮，等待5-10秒（不同网速时间可能有偏差），进入WebIDE界面，如下图：

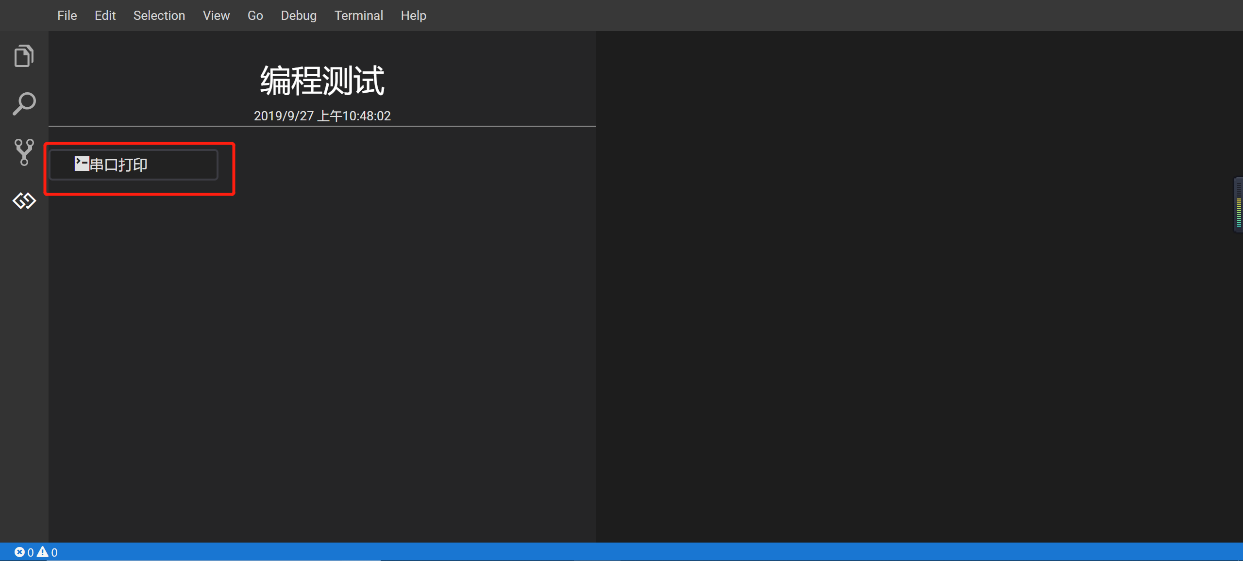


图2.4 WebIDE页面

点击“串口打印”按钮，打开题目描述信息和代码编辑器。下面对WebIDE页面布局做简单描述，红色框内为实验题列表，黄色框内为当前实验题题目描述信息，蓝色框内为实验操作（包括“连接”、“提交”，其中“连接”按钮用来连接远程物联网设备，“提交”按钮用于当代码编写完成时提交运行），绿色框内是代码编辑器，灰色框内是日志和用户输出信息（用户输出为绿色）。

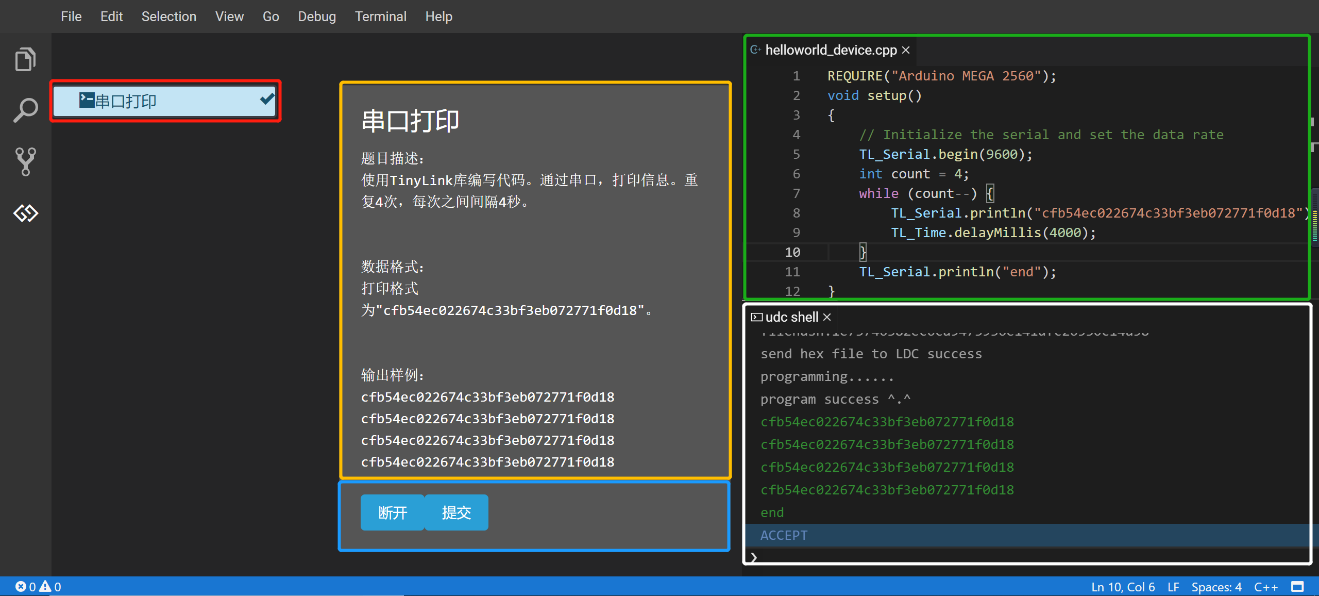


图2.5 WebIDE页面布局

WebIDE编程操作顺序：

1.在绿色框内编写代码

2.在蓝色框内点击连接按钮，连接成功则进入下一步，否则请等待设备可用

3.第二步连接成功后点击提交按钮

4.观察灰色框内的输出，如果编译成功并成功执行会提示“ACCEPT”，否则提示“WRONG ANSWER”。

* TinyLink系统简介

传统的IoT设备端应用开发流程包括硬件选择、应用开发、设备连接。假设用户需要一个测量室内温湿度的设备，根据用户的需求开发者可能会经历如图2.2所示的开发流程。

Step1: 挑选硬件设备，根据功能描述(温湿度)选择DHT11，同时选择Arduino作为开发板

Step2: 根据选择的硬件平台(Arduino)，编写应用程序读取温湿度传感器(DHT11)的数据

Step3: 连接硬件设备，将编译好的程序烧入开发板，运行应用程序

图 2.6 传统物联网应用开发流程

TinyLink是一个快速开发IoT应用的系统。不同于常规自底向上的IoT应用开发模式，TinyLink采用自顶向下的开发模型，根据用户代码自动编译生成硬件配置及相应的二进制文件。用户编写TinyLink代码需要用到TinyLink语言，Tinylink语言是一款与具体硬件平台无关的类C语言，使用类似Arduino的代码结构。用户编写完TinyLink代码后，将源代码上传到TinyLink云平台，系统自动根据上传的代码生成硬件配置和应用程序。

TinyLink编程手册参见（http://tinylink.emnets.org/TinyLink/view/en/document\_page.php）。

* 阿里云IoT Studio平台简介

IoT Studio是阿里云针对物联网场景提供的生产力工具，可覆盖各个物联网行业核心应用场景，帮助您高效经济地完成设备、服务及应用开发。物联网开发服务提供了移动可视化开发、Web 可视化开发、服务开发与设备开发等一系列便捷的物联网开发工具，解决物联网开发领域开发链路长、技术栈复杂、协同成本高、方案移植困难的问题，重新定义物联网应用开发。

IoT Studio平台的详细介绍参见阿里云官方文档（https://studio.iot.aliyun.com/doc?spm=a2c56.12526802.1304866.2.57e7107bHbunlZ）。

* MQTT协议介绍

MQTT（Message Queuing Telemetry Transport， 消息队列遥测传输协议），是由 IBM 发布的基于发布/订阅（publish/subscribe）模式的轻量级通讯协议，构建于 TCP/IP 协议之上。



图2.7 MQTT 协议实现方式

MQTT 协议的实现方式如图 2.3 所示，协议中有三种身份，分别是发布者（Publish）、代理（Broker）和订阅者（Subscribe），发布者和订阅者运行于客户端，代理运行于服务器。在 MQTT 协议中，发布者会在发布消息时指定主题（Topic），订阅者首先订阅主题，当有发布者发布该主题的消息时，订阅此主题的订阅者可收到发布者所发布的消息。

* Arduino Mega 2560平台简介

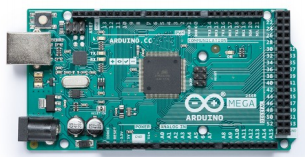


图2.8 Arduino Mega 2560开发板

Arduino Mega 2560是一款方便灵活的开源电子平台，如图2.4所示。Arduino Mega 2560平台采用简单的编程逻辑，为开发者屏蔽了底层的硬件实现细节，广泛应用于物联网原型系统开发。

* 实验内容

1.基于LinkLab远程物联网实验平台，完成“串口打印”实验题，使用TinyLink库编写代码烧录到远程物联网实验设备上，通过串口打印信息。

2.基于LinkLab远程物联网实验平台，完成“MQTT通信”实验题，使用TinyLink库编写代码，通过传感器，读取当前传感器数值，打印至屏幕。同时，连接WIFI，使用MQTT协议，将数据发送至MQTT服务器。

3.基于LinkLab远程物联网实验平台和阿里云IoT Studio平台，完成“阿里云物联网平台”实验题，使用IoT Studio实时显示传感器数据。

# 主要仪器设备

* PC(Windows/Linux/macOS)；

# 操作方法与实验步骤

* LinkLab账号注册和登陆

输入LinkLab平台（http://linklab.tinylink.cn）网址，使用本人学号登陆。LinkLab平台包括课程、实验题、场景编程和考试四个模块。如图4.1，本实验报告主要关注实验题模块中的“串口打印”、“MQTT通信”和“IoT Studio入门”。



图4.1 实验题目

* LinkLab“串口打印”实验题

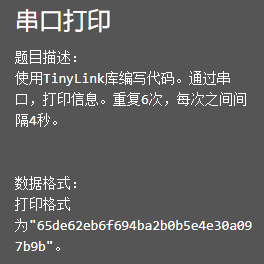


图4.2 “串口打印”题目描述

登陆进入LinkLab平台后，点击“串口打印”实验题的开启按钮。根据题目描述，编写TinyLink代码（提示：重点阅读TinyLink的Serial Module API（链接http://tinylink.emnets.org/TinyLink/view/en/api\_page.php），代码第一行添加REQUIRE("Arduino MEGA 2560");）。

REQUIRE("Arduino MEGA 2560");

void setup()

{

//-----------------------------

//Your code

//-----------------------------

TL\_Serial.println("end");

}

void loop(){

}



图4.3 实验题的设备分配和题目提交

代码编写完之后，依次点击连接和提交按钮可以提交题目（设备独占方式）

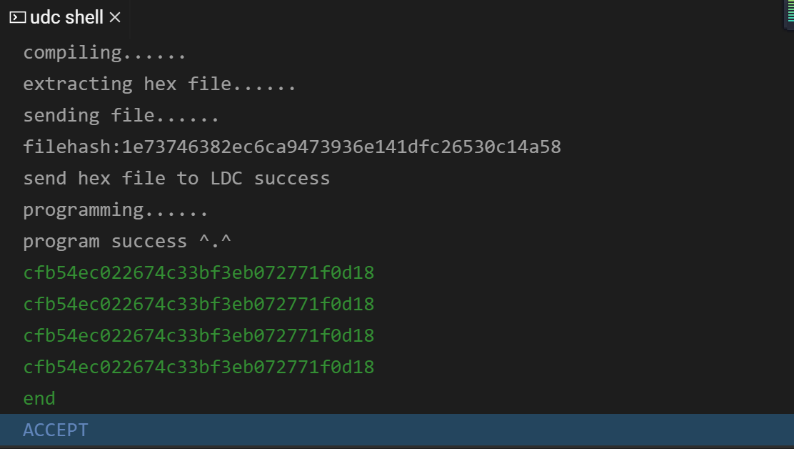


图4.4 实验题提交后的日志和判题结果

实验题目提交之后，udc shell框中会显示程序运行日志和判题结果，答案正确提示“ACCEPT”;答案错误提示“WRONG ANSWER”。

* LinkLab“MQTT通信”实验题

登陆进入LinkLab平台后，点击“MQTT通信”实验题的开启按钮。该实验题要求使用TinyLink库（提示：重点阅读TinyLink的WiFi Module和MQTT module API（链接http://tinylink.emnets.org/TinyLink/view/en/api\_page.php））编写代码，并通过传感器读取光照或者温湿度数据，并将数据打印至屏幕。同时，连接WIFI，使用MQTT协议将数据发送至云端。



图4.5 “MQTT通信”实验题目的WiFi信息

根据“MQTT通信”实验题目的描述，设备初始化时需要连接WiFi并生成MQTT文件描述符。

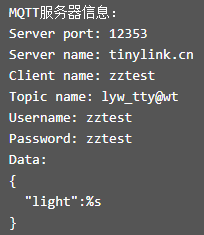


图4.6 “MQTT通信”实验题目的MQTT服务器信息

参考代码：

TL\_MQTT mqtt;

int port = 12353;

char servername[] = "tinylink.cn";

char clientname[] = "zztest";

char topicName[] = "judge/0a4c";

char username[] = "zztest";

char password[] = "zztest";

void setup() {

//-----------------------------

//Your code

//-----------------------------

TL\_Serial.println("end");

}

void loop() {

}

该实验题会检查程序是否通过TinyLink调用了传感器，并监听MQTT报文，检验两者的一致性。

* 阿里云IoT Studio物模型及服务编排入门

1. 阿里云IoT Studio（https://iot.aliyun.com/products/iotstudio）账号注册；
2. 登陆阿里云IoT Studio，创建空白项目，如图4.7；



图4.7 创建空白项目

1. 创建产品。新建设备模型，名称任意，分类为“自定义品类”，联网方式为“WiFi”，数据格式为“ICA 标准数据格式（Alink JSON）”，其他内容参考图4.8。



图4.8 创建产品



图4.9 设备新增的功能

1. 在所创建产品的“功能定义”中为设备模型添加“自定义功能”。我们新增四个属性，分别是“温度（CurrentTemperature）”、“湿度（CurrentHumidity）”、“光照度（mlux）”和“判题密钥（key）”，数据类型和取值范围参考图4.9。



图4.10 新增测试设备

1. 新增测试设备。“产品”选择第三步创建的产品，添加方式为“自动生成”，设备数量为“1”，如图4.10所示，然后点击提交按钮。



图4.11 下载激活凭证

1. 点击提交之后会弹出“新增完成”对话框，这里点击“下载激活凭证”按钮将激活凭证下载下来，如图4.11所示。
2. 现在，我们在IoT Studio上面创建了产品和设备，并下载了激活凭证，接下来就是通过激活凭证和MOTT协议将我们的远程设备和IoT Studio上面的设备建立连接。建立连接之前，需要使用激活凭证生成“MQTT域名、端口、ClientID、UserName、Password”等信息，这些信息是程序使用MQTT协议和IoT Studio设备建立关联的重要依据。生成过程请参考链接（<https://yq.aliyun.com/articles/592279>）。
3. 在WebIDE中参考以下样例编写代码：

TL\_MQTT mqtt;

int port = 1883;

char serverName[] = " MQTT域名";

char clientName[] = " ClientID ";

char topicName[] = "… /thing/event/property/post";

char userName[] = "";

char password[] = "";

char SSID[] = "AZFT";

char Pass[] = "AZFT123456";

void setup() {

}

void loop() {

}

1. 设备端和阿里云通信使用的是 Alink 协议。参考阿里云IoT Studio对Alink 通信协议（https://help.aliyun.com/knowledge\_list/89310.html）的介绍（提示：设备端传输的数据格式参考 Alink 通信协议介绍文档中的“单个设备场景->上行数据->设备属性上报”部分内容），编写设备端代码（提示：重点阅读TinyLink的WiFi Module和MQTT module API（链接http://tinylink.emnets.org/TinyLink/view/en/api\_page.php）），使用 MQTT 协议连接到阿里云，并将采集到（温度、湿度、光照度）（传感器数值可用随机数或固定值代替）上传到阿里云IoT Studio。
2. 学习阿里云IoT Studio的服务开发（https://studio.iot.aliyun.com/studioservice-doc#index.html）部分文档。学习完后，进入所创建产品的服务开发工作台，新建服务用于设备的属性上报触发。



图4.12 新建服务

新建完服务之后，进行服务编排。根据题目描述，如图4.13，整个服务设置为设备属性上报触发，并连接到自定义API，详细信息参考图4.13。确定无误后，将服务进行部署并启动。这一步骤的目的是给判题系统提供程序使用IoT Studio的依据。



图4.13 服务编排

最后，在“阿里云物联网平台(TinyLink)”题目下运行所编写代码，上传属性值。运行结果可在IoT Studio项目中通过以下路径查看“设备管理-设备-设备列表-查看-运行状态”。



图4.14 设备属性上报更新

# 实验数据记录和处理

* 实验题“串口打印”源代码及运行日志截图

实验要求：



实验代码：

REQUIRE("Arduino MEGA 2560");

void setup()

{

    // Initialize the serial and set the data rate

    TL\_Serial.begin(9600);

    int count = 5;

while (count--) {

TL\_Serial.println("eed9af14659e49c1b5c5df63cf5d0c92");

TL\_Time.delayMillis(5000);

}

TL\_Serial.println("end");

}

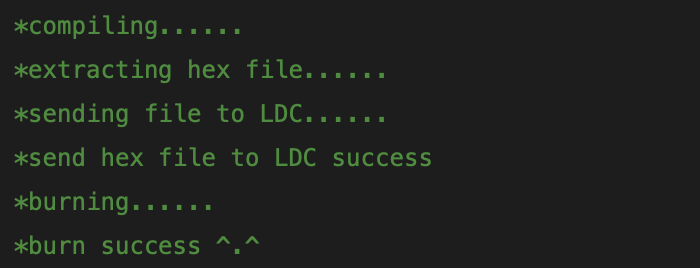
void loop(){

}

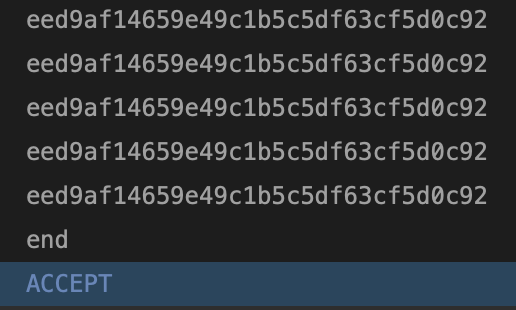
点击连接：

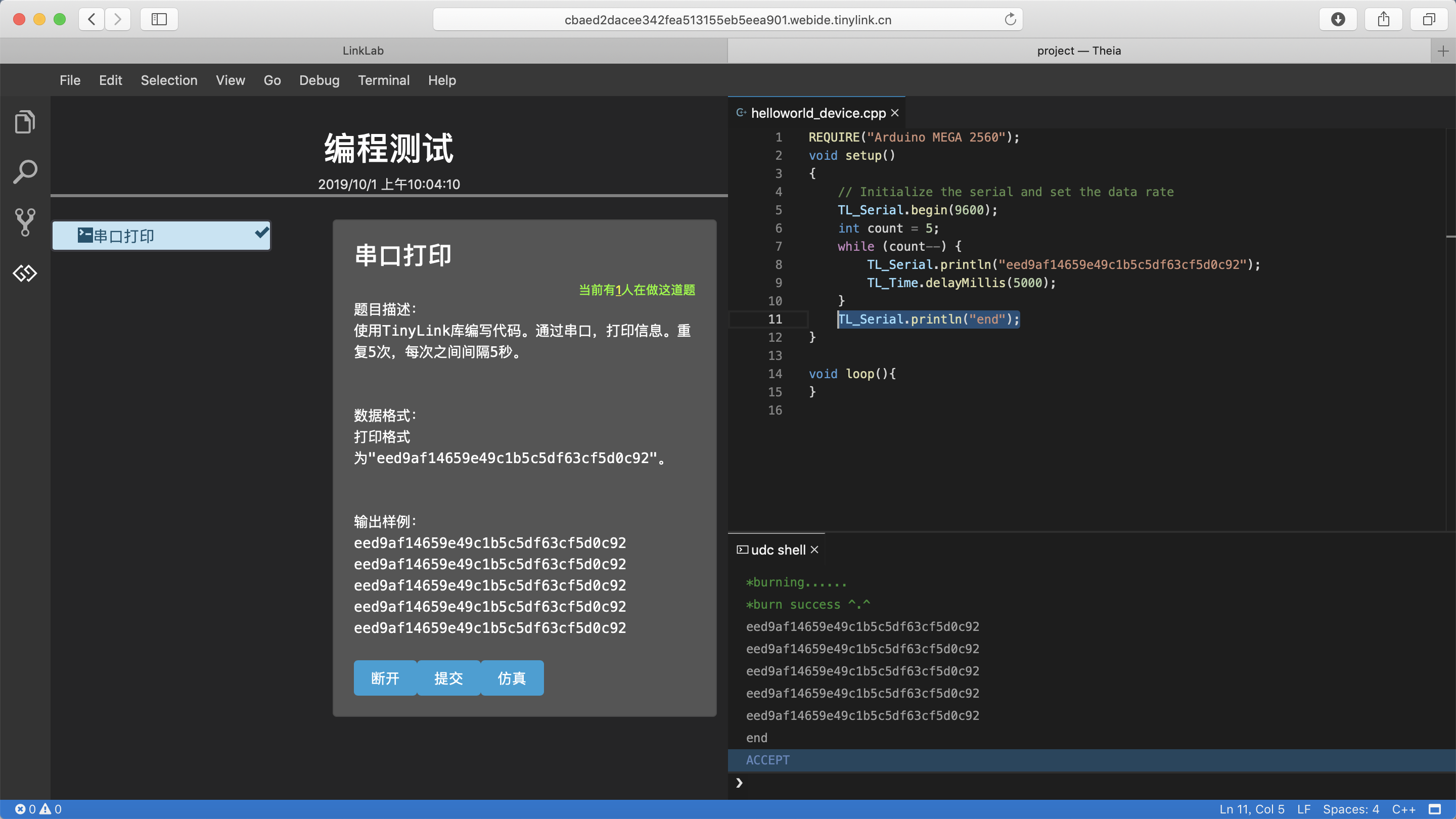


点击提交：



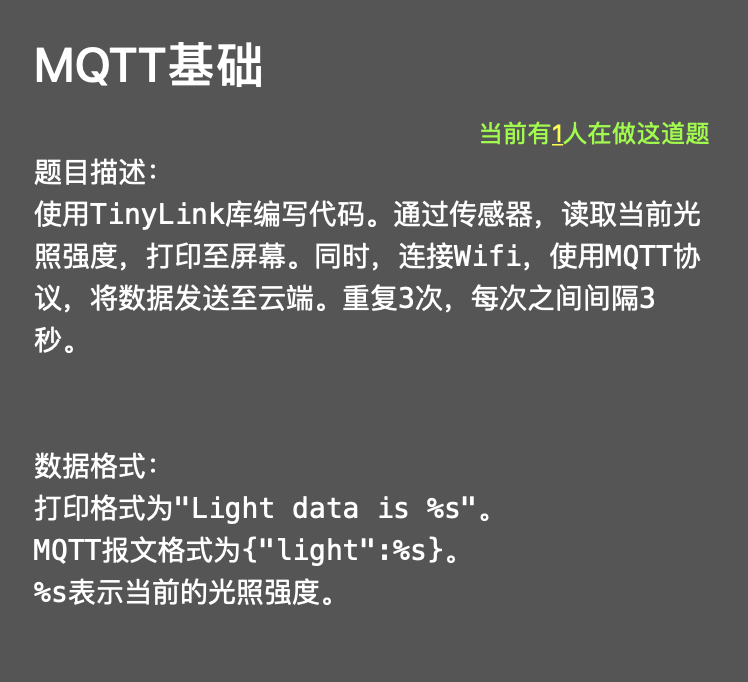
成功：





* 实验题“MQTT通信”源代码及运行日志截图

实验要求：



实验代码：

TL\_MQTT mqtt;

int port = 12353;

char servername[] = "tinylink.cn";

char clientname[] = "zztest";

char topicName[] = "judge/2315";

char username[] = "zztest";

char password[] = "zztest";

void setup() {

TL\_Serial.begin(9600);

TL\_WiFi.init();

TL\_WiFi.join("AZFT","AZFT123456");

mqtt = TL\_WiFi.fetchMQTT();

mqtt.connect(servername, port, clientname, username, password);

//TL\_Serial.println("begin");

for(int i = 0; i < 3; i++){

TL\_Light.read();

String data = String("{") + "\"light\":" + TL\_Light.data() + String("}");

char buf[100];

data.toCharArray(buf, 100);

TL\_Serial.print("Light data is ");

TL\_Serial.println(TL\_Light.data());

int res = mqtt.publish(topicName, buf, strlen(buf));

TL\_Time.delayMillis(3000);

}

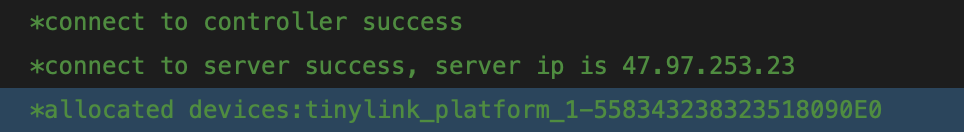
TL\_Serial.println("end");

}

void loop() {

}

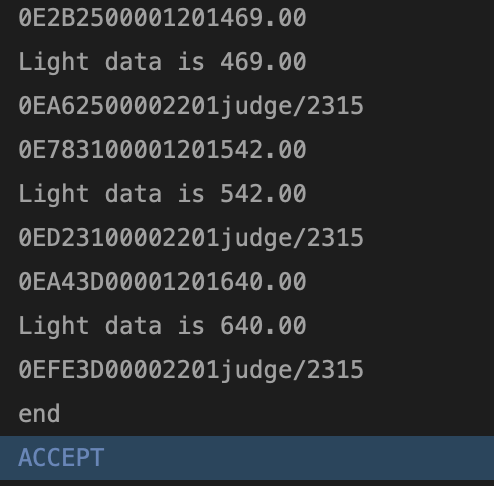
点击连接：

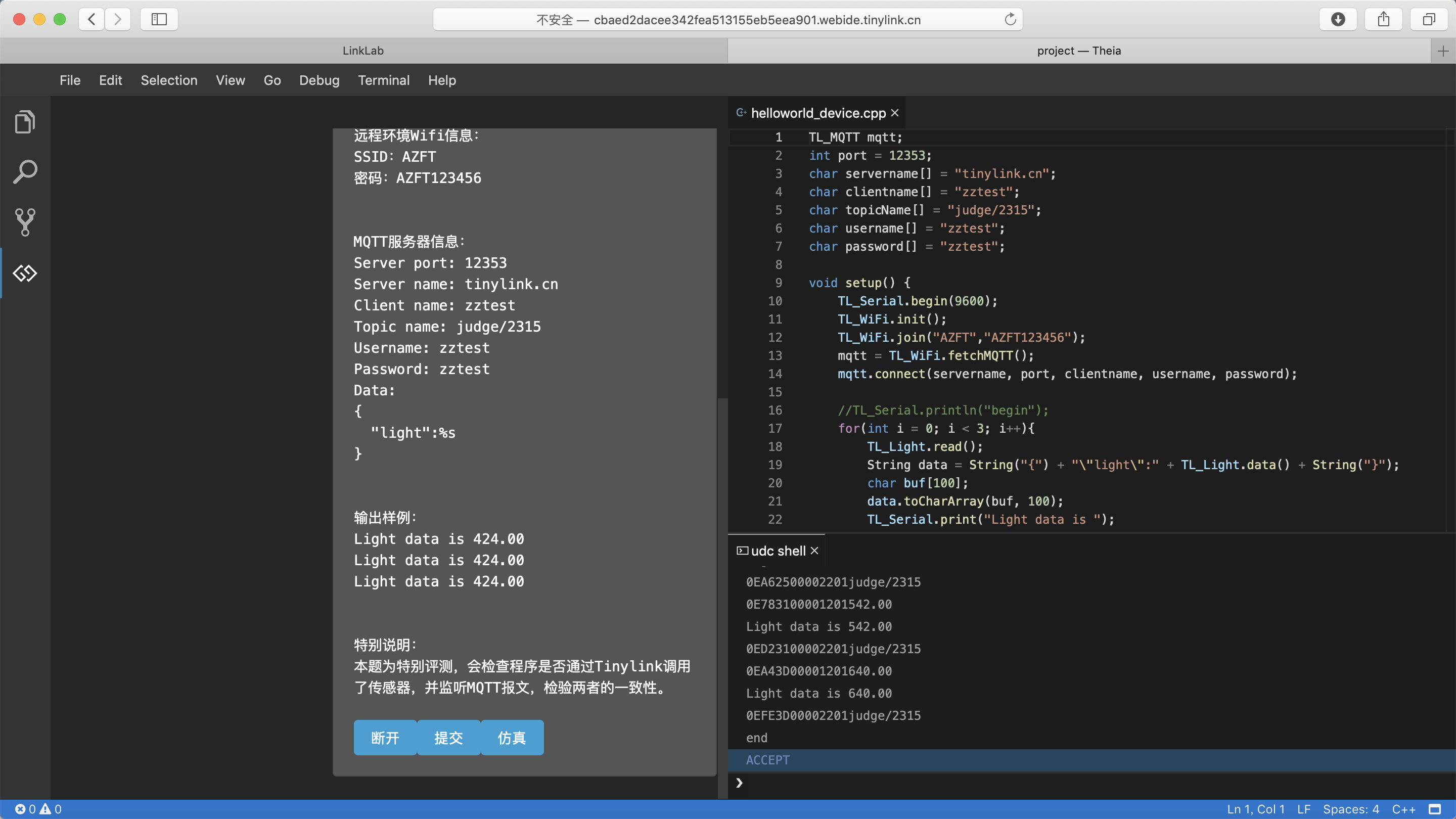


点击提交：



成功：





* 实验题“阿里云物联网平台”源代码及运行日志截图

实验代码：

TL\_MQTT mqtt;

int port = 1883;

char serverName[] = "a1EWaqIN3nk.iot-as-mqtt.cn-shanghai.aliyuncs.com";

char clientName[] = "123|securemode=3,signmethod=hmacsha1,timestamp=10|";

char topicName[] = "/sys/a1EWaqIN3nk/QAIEabQldNqfZKZjNZio/thing/event/property/post";

char userName[] = "QAIEabQldNqfZKZjNZio&a1EWaqIN3nk";

char password[] = "f8e9b9b330195a7640cd1125d45c5cfb36b9b4a8";

char SSID[] = "AZFT";

char Pass[] = "AZFT123456";

void setup() {

TL\_WiFi.init();

bool b = TL\_WiFi.join(SSID,Pass);

mqtt = TL\_WiFi.fetchMQTT();

int a = mqtt.connect(serverName, port, clientName, userName, password);

TL\_Serial.begin(9600);

TL\_Serial.println(a);

TL\_Light.read();

TL\_Humidity.read();

TL\_Temperature.read();

String data = "{\"id\" : \"123\", \"version\":\"1.0\", \"params\" : {";

data += "\"CurrentTemperature\":";

data += TL\_Temperature.data();

data += ", \"CurrentHumidity\":";

data += TL\_Humidity.data();

data += ", \"mlux\":";

data += TL\_Light.data();

data += ", \"key\":";

data += "\"MTYsMTUyLGE0MzVjZDUz\"";

data += "},\"method\":\"thing.event.property.post\"}";

TL\_Serial.println(data);

char buf[500];

data.toCharArray( buf, 500 );

int res = mqtt.publish(topicName, buf, strlen(buf),0);

TL\_Serial.println(res);

TL\_Time.delayMillis(1000);

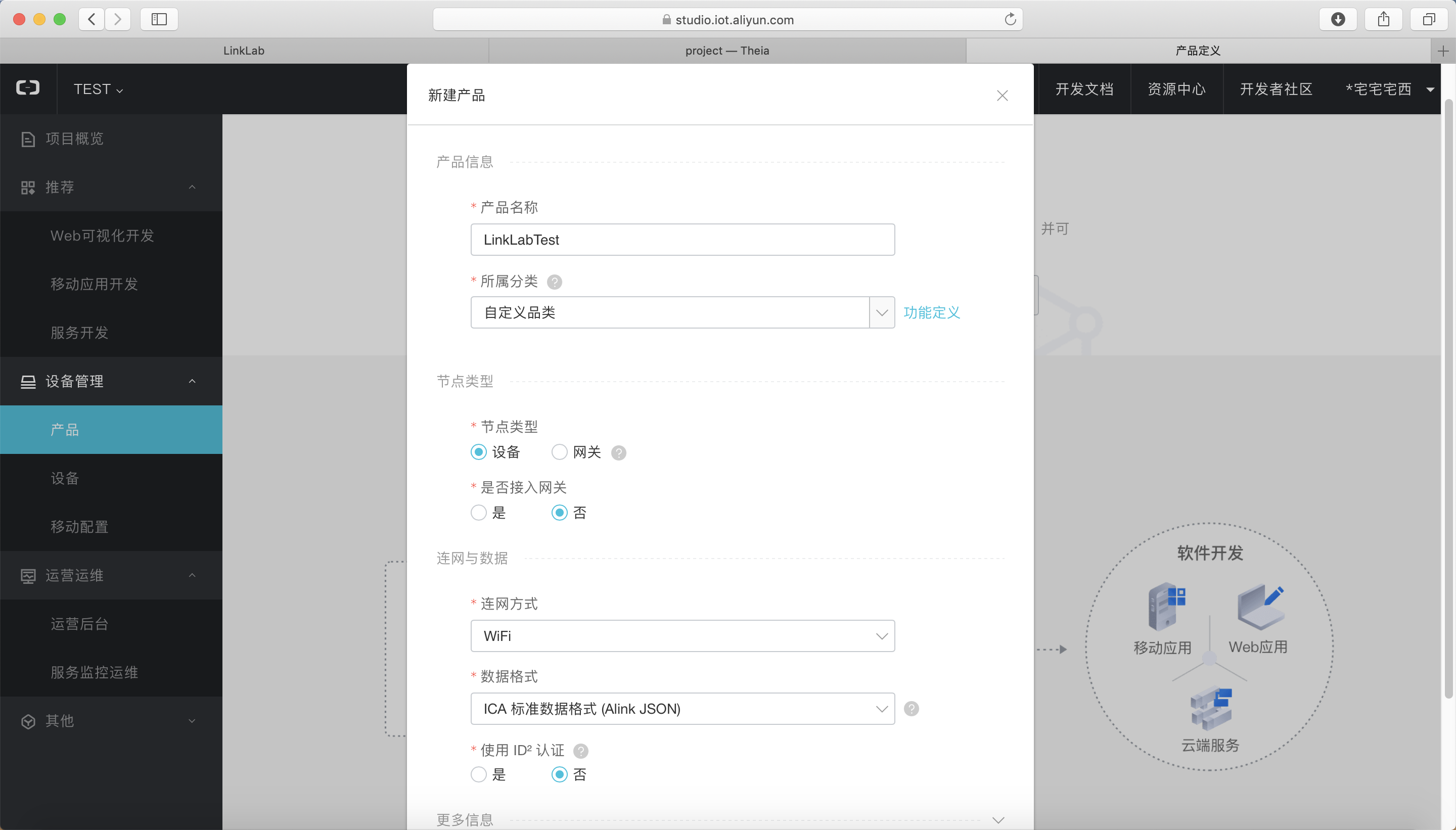
TL\_Serial.println("end");

}

void loop() {

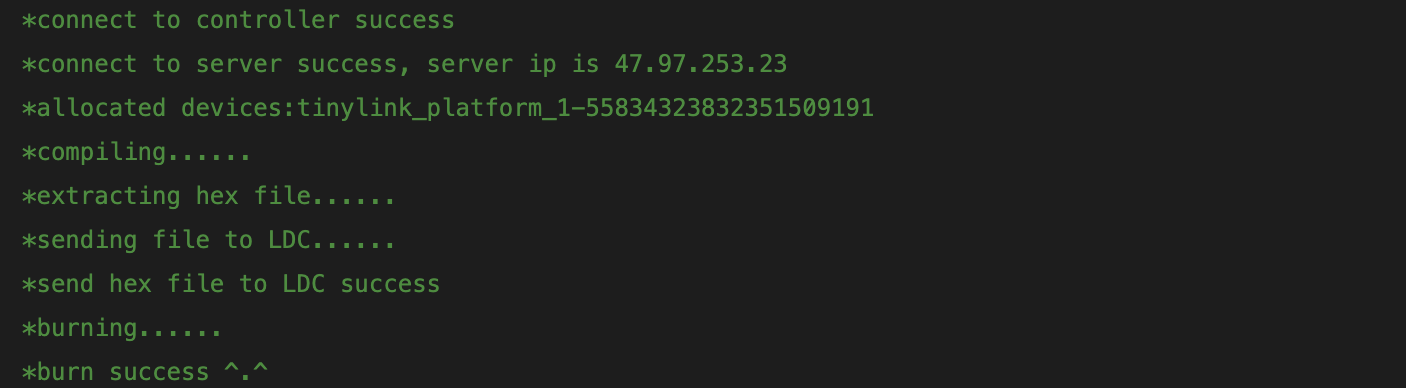
}

创建空白项目，创建产品。添加自定义功能，新增四个属性：温度、湿度、光照度、判题密钥。

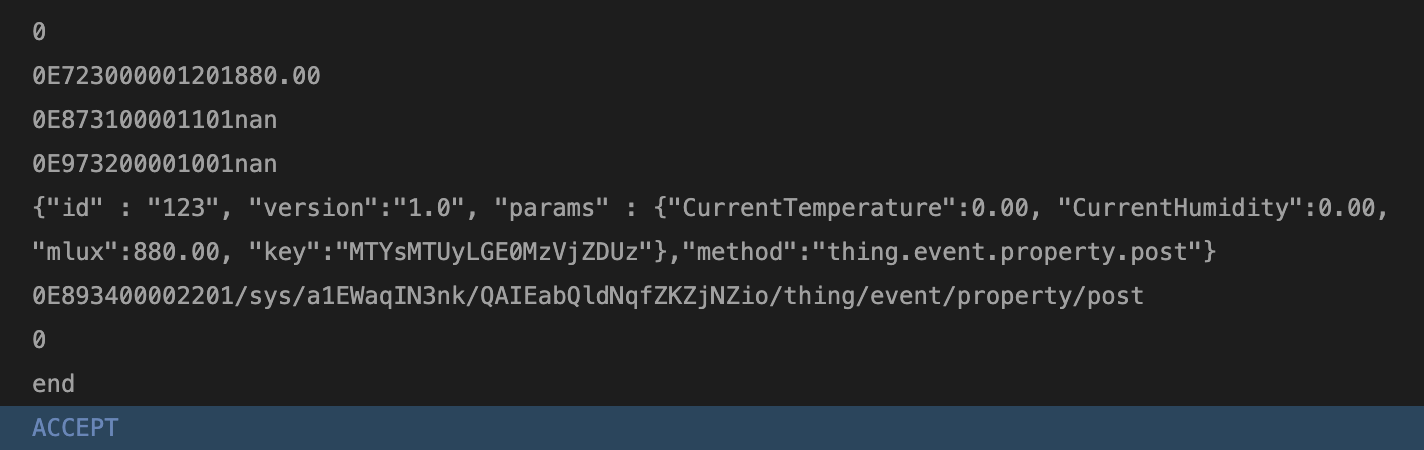


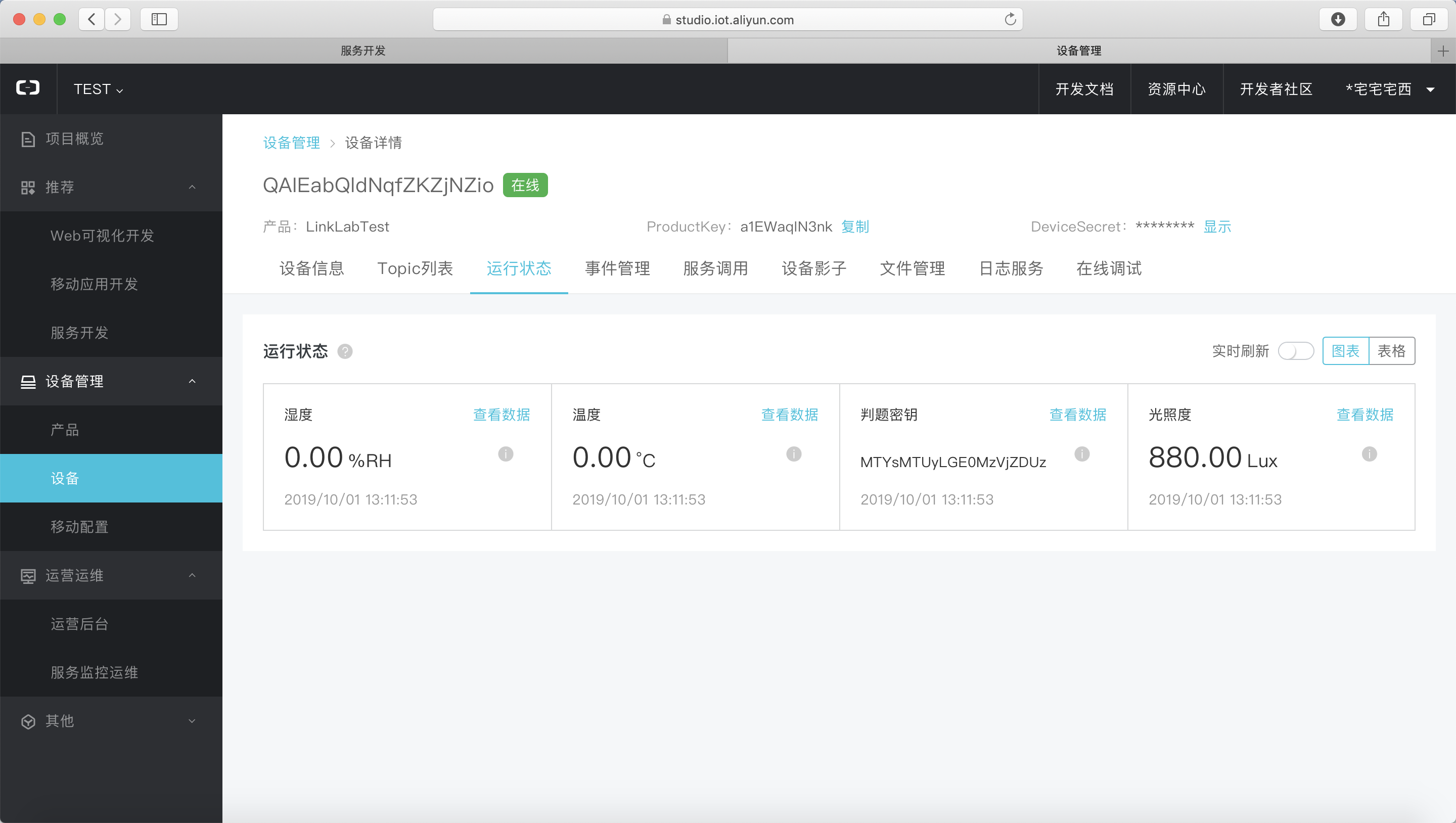
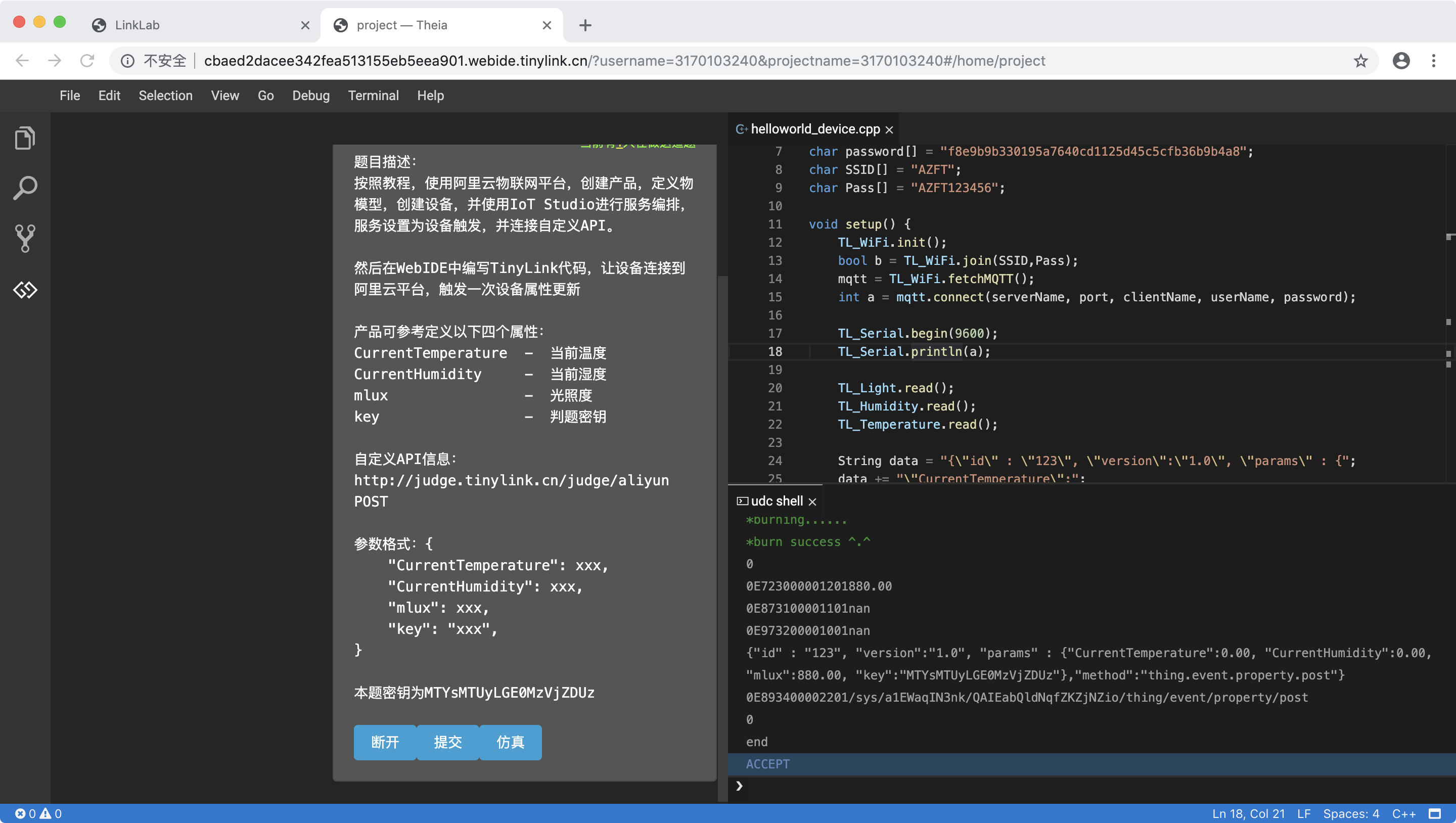


点击连接，点击提交：

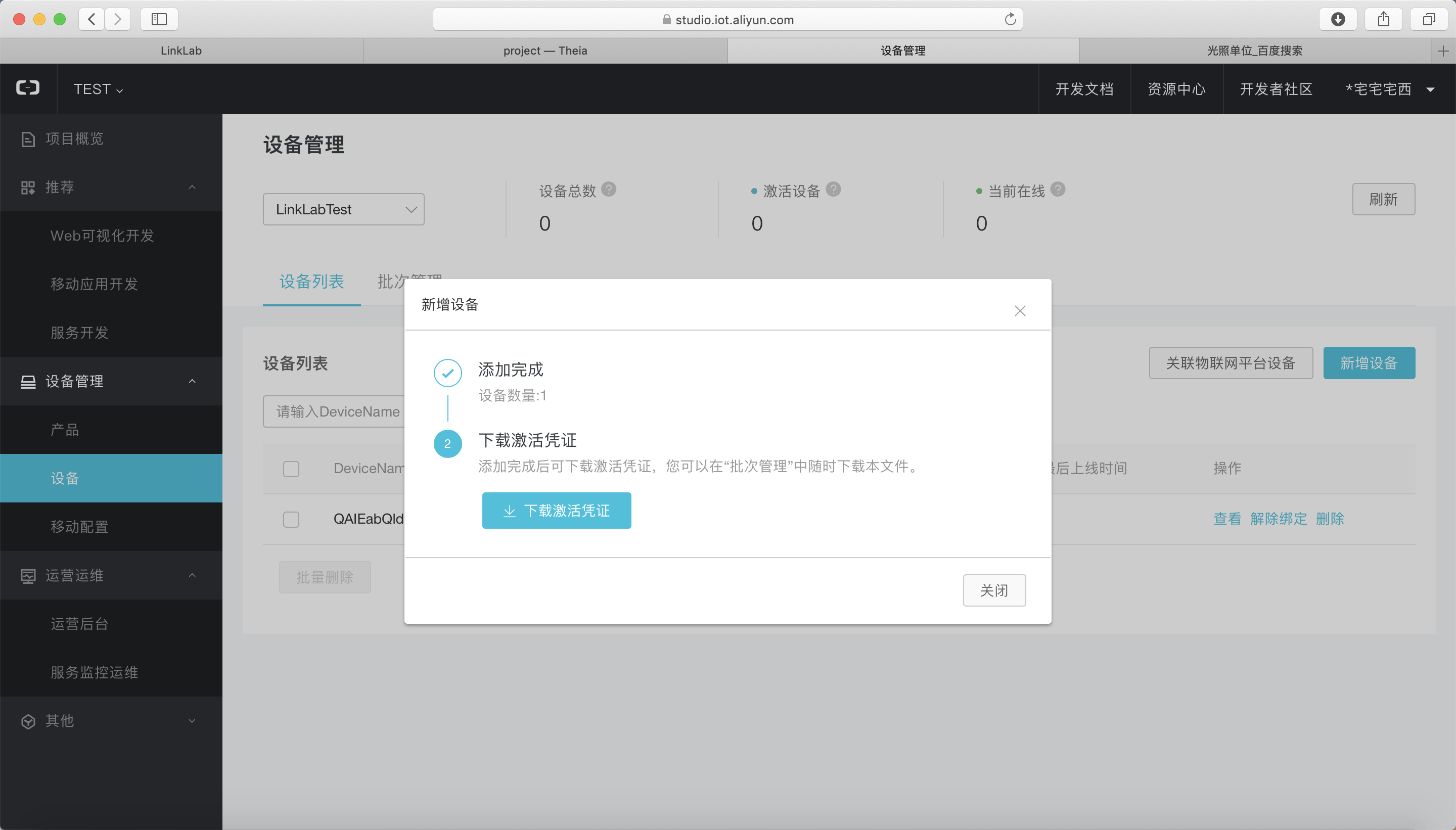


成功：



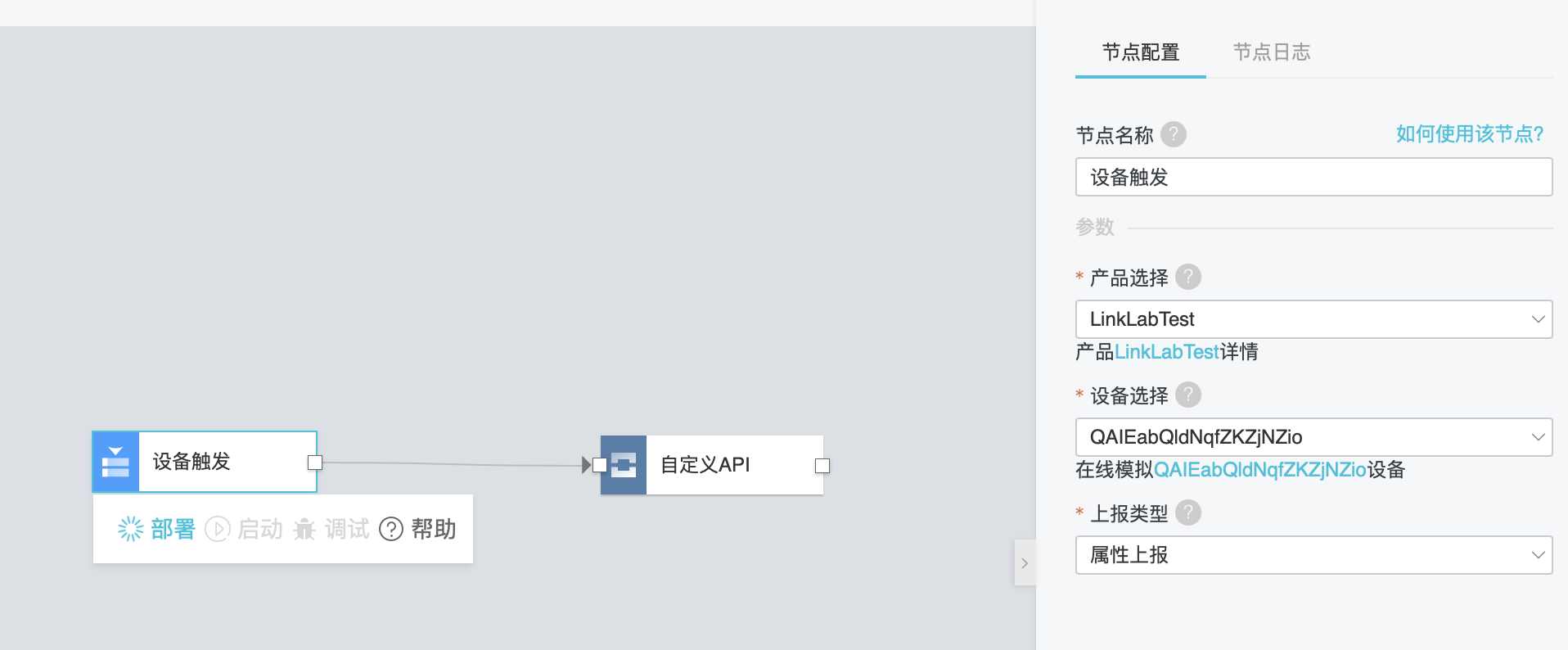


* 如图4.11，你的设备属性上报更新的截图





* 如图4.13，你的服务编排的截图







# 实验结果与分析

* LinkLab在开发中起到了什么作用？你认为的LinkLab哪些地方值得完善？

在开发中，LinkLab提供了一个实现“webIDE+云端编译+远程烧写”的平台。可以在LinkLab的网页版IDE上编写代码。用户点击连接，LinkLab选择处于空闲状态的远程设备。用户点击提交，LinkLab在云端完成代码的编译，然后将代码烧写到这个设备上，运行代码，读取各种传感器的数据。LinkLab在开发过程中是用户编写程序、设备运行程序两者之间的桥梁，用互联网技术实现物物相连，使得用户可以免去搭建设备的工作。LinkLab也是用户连接远程服务器之间的桥梁。

目前LinkLab设备较少，在实验的过程中遇到了服务器出错、连接设备失败的问题，所以需要搭建更多的远程设备。

* TinyLink 采用类似于 Arduino 的 setup-loop 的编程结构，对这种编程结构的优缺点进行分析。

优点：

封装了main函数。将初始化工作和重复执行的工作分开，显得简洁明了，降低编程难度。Loop可以实现实时更新的作用。

缺点：

无限循环的loop浪费资源。高度封装和底层脱节。

* 相比于传统的 HTTP 协议，MQTT 协议有什么特点？为什么 MQTT 更适合物联网应用？

HTTP协议是一个简单的请求-相应协议，通常运行在TCP之上，指定了客户端可能发送给服务器的信息类型和响应类型。HTTP协议需要客户端不断轮询服务器，消耗比较大，对网络要求高。MQTT协议是一种基于发布/订阅模式的“轻量级”通讯协议，该协议构建于TCP/IP协议上。MQTT协议适用于多种传感器接入互联网进行数据交换的物联网时代，它可以用少量的代码和有限的带宽为连接远程设备提供实时可靠的信息服务，具有低开销、低带宽、低网络要求的优势。MQTT确保高传输保证，具有3个级别的服务质量：最多一次、至少一次、恰好一次，为用户提供Last will & Testament和Retained消息选项。

# 讨论、心得

|  |  |  |  |
| --- | --- | --- | --- |
| 各项评分（1-差，2-可以容忍，3-满意，4-优秀） | | | |
| LinkLab系统易用性(完成IoT应用的整个流程) | | | 3 |
| LinkLab编程便捷性(根据实验文档和实验题目描述，是否便于编程) | | | 3 |
| LinkLab硬件易用性(所分配硬件设备是否可用，如wifi、传感器等) | | | 2 |
| LinkLab系统鲁棒性(系统流畅、系统容错和系统Bug等) | | | 2 |
| 实验感想 | | | |
| 简述实验中最难的case及其难点 | | 实验中最难的是阿里云IoT Studio物模型及服务编排，因为是第一次上手，按照实验报告的步骤进行，理解不深入，会不小心遗漏。  在LinkLab上的程序显示wrong\_answer时，难以很快找到出错的原因。 | |
| 列出你失败的case,并解释失败的原因 | | 1. LinkLab多次显示wrong\_answer，发现自己没有部署运行服务。 2. 在“阿里云物联网平台”实验题中，尽管LinkLab已经显示accept，但是在阿里云服务平台上设备显示未激活。最后发现自己在程序中clientName、serevrName等这些参数没有正确匹配。 | |
| 意见反馈 | | | |
| LinkLab系统 | 无。 | | |
| LinkLab硬件 | 建议增加设备。 | | |
| 系统Bug | 有几次点击连接、提交，没有提示信息打印出来，日志卡住不动了。 | | |
| 其他 | 无。 | | |