**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。

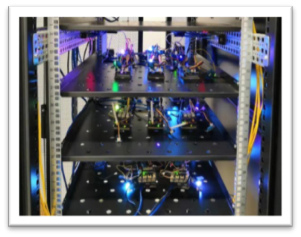


图2.1 LinkLab嵌入式设备

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：注册账号和加入课程

使用教师提供的linklab账号登录<http://linklab.tinylink.cn>。 默认登录密码为账号的后6位，建议登录后及时修改密码。

进入“个人中心”之后，点击“我的课程”，可以看到相应的计算机网络课程实验。



图2.2 “浙大计算机网络”课程

第二步：实验题界面

进入实验界面后，有各个的选项，点击实验标题或右侧的箭头能够将实验内容进行展开

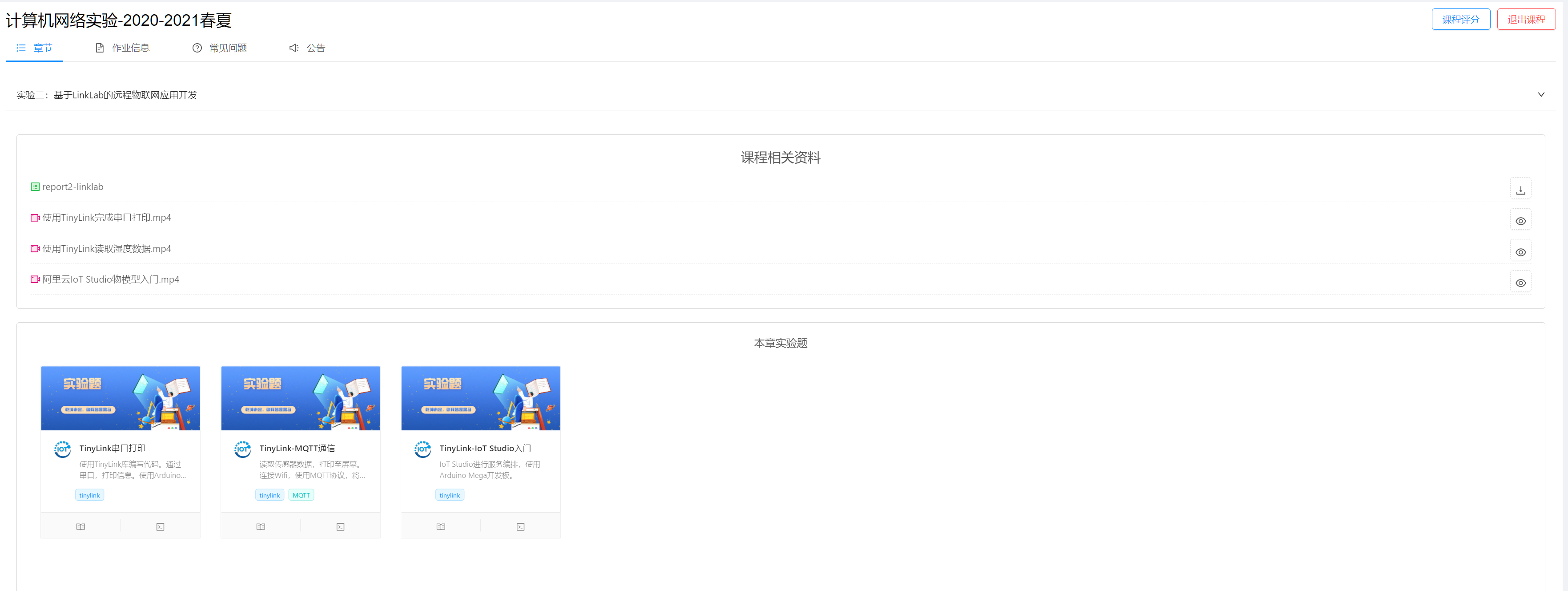


图2.3 实验题界面

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：

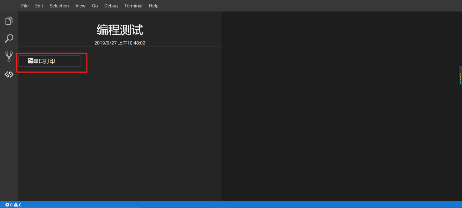


图2.4 WebIDE页面

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）。

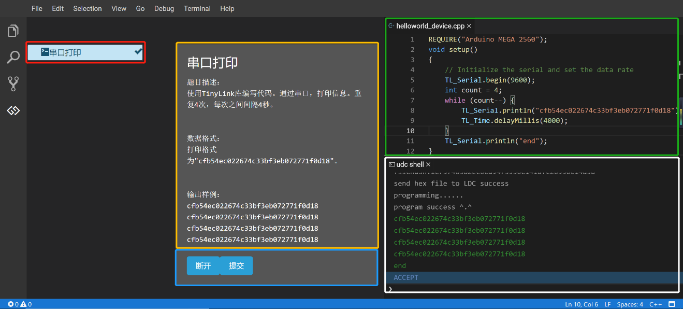


图2.5 WebIDE页面布局

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://kubernetes.tinylink.cn/tinylink/view/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

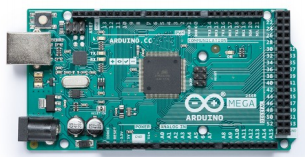


图2.8 Arduino Mega 2560开发板

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

* **1 LinkLab“串口打印”实验题**

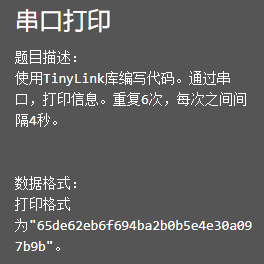


图4.2 “串口打印”题目描述

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://kubernetes.tinylink.cn/tinylink/view/api\_page.php）

C:\Users\lenovo\AppData\Local\Temp\1569589143(1).png

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）

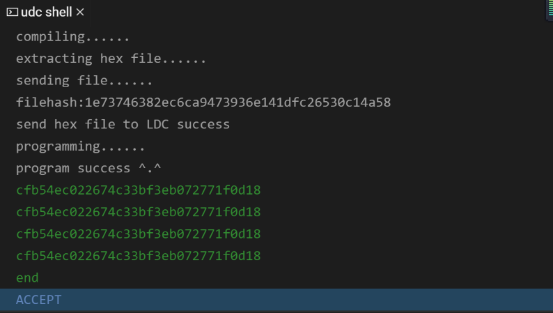


图4.4 实验题提交后的日志和判题结果

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”。

REQUIRE("Arduino MEGA 2560");

int count = 0;

void setup()

{

TL\_Serial.begin(9600);

count = 0;

}

void loop(){

if( count < 5){

TL\_Serial.println("89df0563e5554fdca202d1412416946c");

TL\_Time.delayMillis(3000);

count +=1;

}

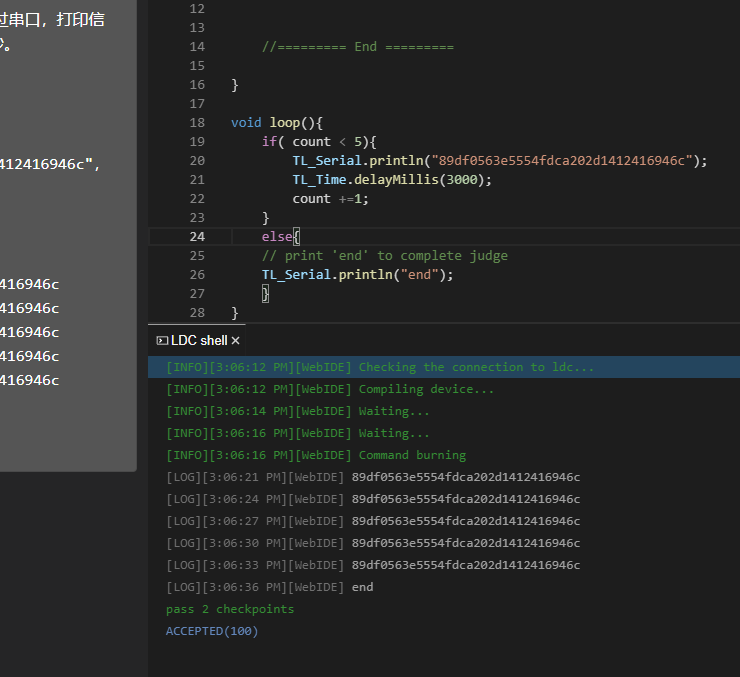
else{

// print 'end' to complete judge

TL\_Serial.println("end");

}

}



* **2 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://kubernetes.tinylink.cn/tinylink/view/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。

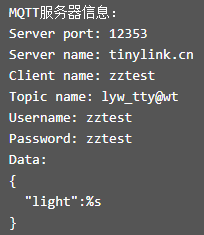


图4.6 “MQTT通信”实验题目的MQTT服务器信息

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

REQUIRE("Arduino MEGA 2560");

int port = 12353;

char serverName[] = "tinylink.cn";

char clientName[] = "zztest";

char userName[] = "zztest";

char password[] = "zztest";

char SSID[] = "AZFT";

char Pass[] = "AZFT123456";

int count = 0 ;

TL\_MQTT mqtt;

char topicName[] = "judge/9bb2";

void setup()

{

// Initialize network

TL\_Serial.begin(9600);

TL\_WiFi.init();

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

mqtt = TL\_WiFi.fetchMQTT();

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

for( int i = 0 ; i < 4; i ++){

TL\_Humidity.read();

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

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

char buf[100];

String data = "{";

data += "\"humidity\":";

data += TL\_Humidity.data();

data += "}";

data.toCharArray( buf,100 );

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

TL\_Time.delayMillis(3000);

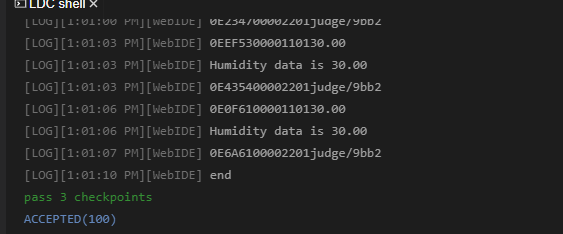
}

TL\_Serial.println("end");

}

void loop(){

}



* **3 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>； 地址好像换成了.
2. 可能是<https://iot.console.aliyun.com/product>

很奇怪, 事实上是, [应用开发 (aliyun.com)](https://studio.iot.aliyun.com/) 这里移动应用,新建应用, 然后才找到新建项目.

1. 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。

它没有说清楚要批量添加. 而且下载设备证书.



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MQTT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>）。 [阿里云IoT Studio物模型入门 (tinylink.cn)](https://linklab.tinylink.cn/static/tutorial/tutorial06.html?time=1634648133494) 参考这个可以下载技小新MQTT单片机编程小工具

这个链接介绍了它的tool, 但是这个tool打不开.

Timestamp 可以不传递 .

|  |  |  |
| --- | --- | --- |
| ﻿DeviceName | DeviceSecret | ProductKey |
| P4XToDcc5qW6WLTShmYx | 0def9fac7929d5ba43e83a7aa0e03aad | ghrx90JEQiz |

<https://files.alicdn.com/tpsservice/88413c66e471bec826257781969d1bc7.zip?spm=a2c4g.11186623.2.21.374c58001Hmxut&file=88413c66e471bec826257781969d1bc7.zip> 从这里下载了一个软件,MQTT协议的password计算器. 因为下面的评论说不能用.

Client Id 表示客户端ID，可自定义，长度不可超过64个字符。建议使用设备的MAC地址或SN码，方便您识别区分不同的客户端。 我写了 12345

Method使用 hmacsha1

密码生成了 9051BE31491AC9AD883E993515497EE285958E3D

域名根据[实例管理 (aliyun.com)](https://help.aliyun.com/document_detail/147356.htm?spm=a2c4g.11186623.0.0.15f9f24atHd9Nv#section-7lj-bvu-re8) 在**实例详情**页面应该是 iot-06z00iu2k8n6zea.mqtt.iothub.aliyuncs.com

根据技小新MQTT单片机编程小工具 生成:

端口1883

UserName: P4XToDcc5qW6WLTShmYx&ghrx90JEQiz

Password: 68f087da8f8333aea5169a5b738fc657399a3ccc

MQTT域名:ghrx90JEQiz.iot-as-mqtt.cn-shanghai.aliyuncs.com:1883

Client ID : FESA234FBDS24|securemode=3,signmethod=hmacsha1,timestamp=789|

1. 在WebIDE中参考样例编写代码：[阿里云IoT Studio物模型入门 (tinylink.cn)](https://linklab.tinylink.cn/static/tutorial/tutorial06.html?time=1634648133494)
2. 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可以在ALink协议->设备属性，事件业务->设备事件上报->上行(Alink JSON)部分），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）上传到阿里云IoT Studio。
3. 学习阿里云IoT Studio的服务开发（https://studio.iot.aliyun.com/studioservice-doc#index.html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。新建服务的位置在IOT Studio的应用开发界面中的业务服务标签页

可能是 <https://iot.aliyun.com/products/iotstudio>

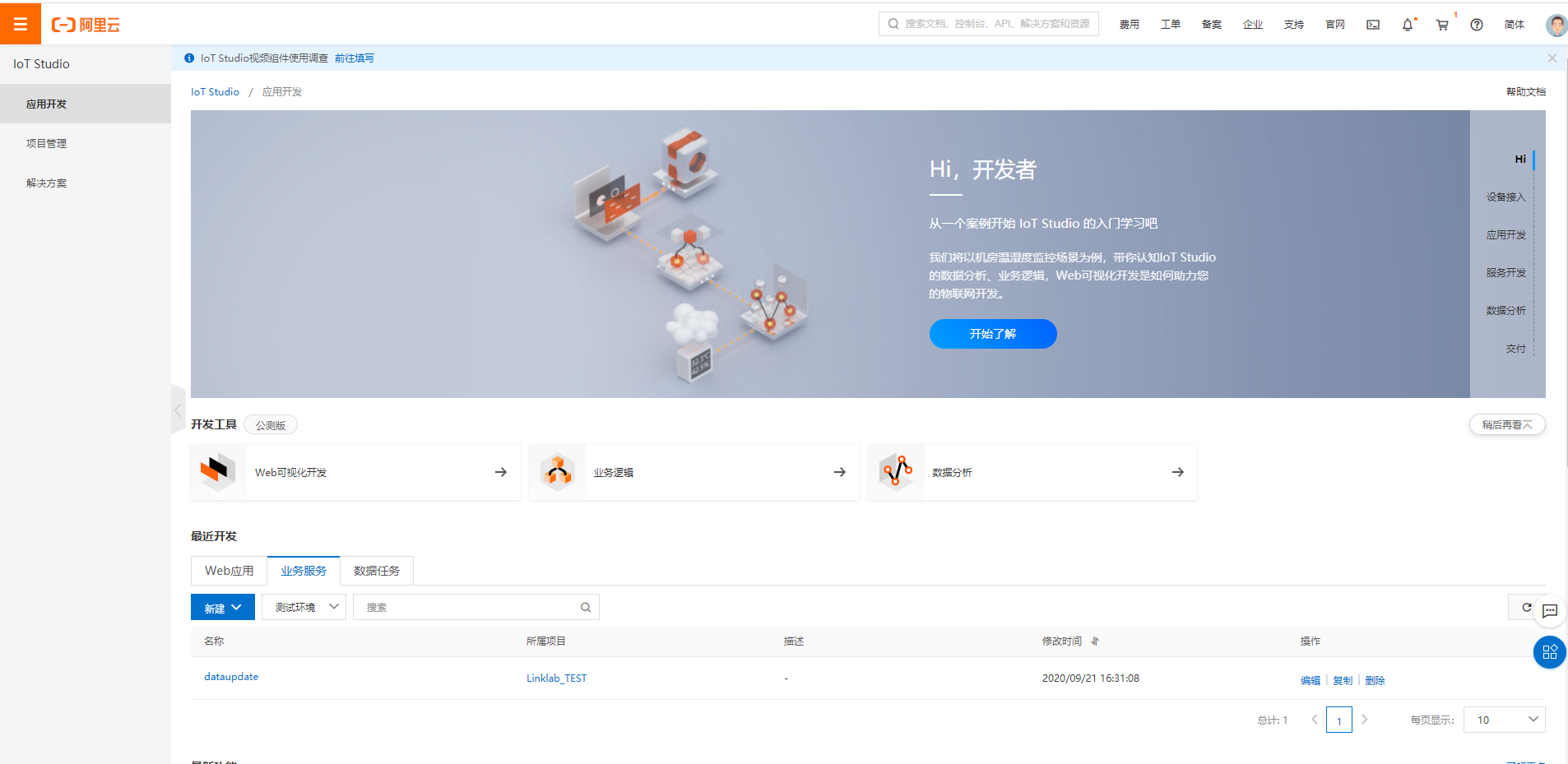


图4.12 新建服务



图4.13 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4.14，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4.14。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。

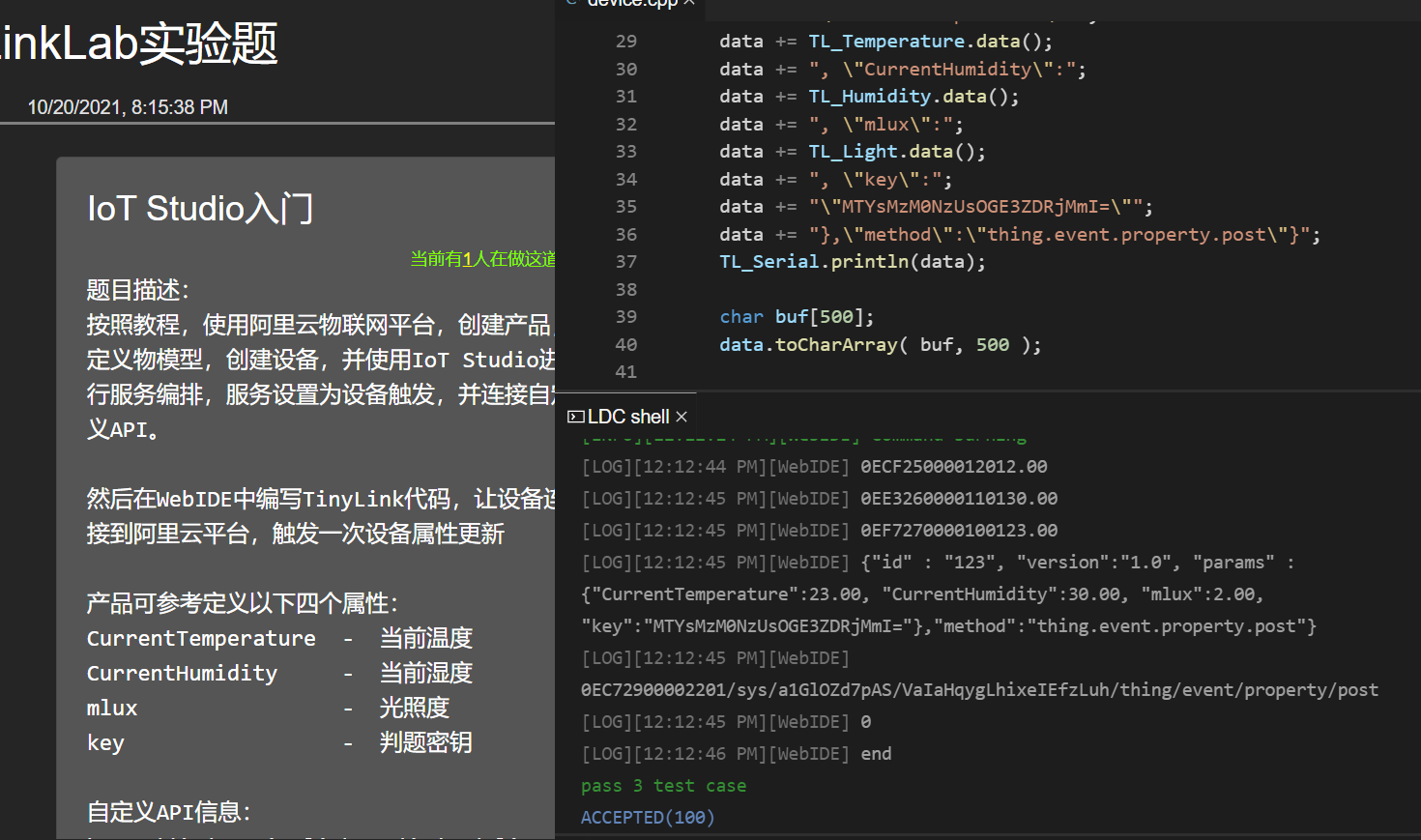


图4.14 服务编排

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4.15 设备属性上报更新



代码:

TL\_MQTT mqtt;

// remote wifi

char SSID[] = "AZFT";

char Pass[] = "AZFT123456";

int port = 1883;

// fills your Aliyun info

char serverName[] = "a1GlOZd7pAS.iot-as-mqtt.cn-shanghai.aliyuncs.com";

char clientName[] = "FESA234FBDS24|securemode=3,signmethod=hmacsha1,timestamp=789|";

char topicName[] = "/sys/a1GlOZd7pAS/VaIaHqygLhixeIEfzLuh/thing/event/property/post";

char userName[] = "VaIaHqygLhixeIEfzLuh&a1GlOZd7pAS";

char password[] = "b1d22e159500d36eb0fb81c3a2375e0bff10070e";

void setup()

{

// Initialize network

TL\_Serial.begin(9600);

TL\_WiFi.init();

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

mqtt = TL\_WiFi.fetchMQTT();

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

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 += "\"MTYsMzM0NzUsOGE3ZDRjMmI=\"";

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);

//========= End =========

// print 'end' to complete judge

TL\_Serial.println("end");

}

void loop(){

TL\_Time.delayMillis(5000);

}

# 实验数据记录和处理

* 观察shell界面产生的中间文件是什么类型的，此文件可以跨平台、跨设备使用吗？

没法看到中间文件.

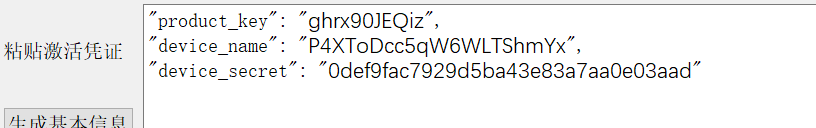
* 实验中使用MQTT模块的哪一API向服务器发送数据？如果希望实时追踪服务器上的数据状态变化，又应该使用哪一API？

使用 mqtt.publish(topicName, buf, strlen(buf)); 向服务器发送数据

实时追踪

mqtt.subscribe(topicName, messageArrived, 0);

* 记录本实验题中创建的设备的三元组信息，各个信息的含义是什么？为何要用三元组定义一个设备？



每个产品根据productKey来区分。一个产品里面有很多台设备，每个设备都有一个deviceName和deviceSecret与之对应。通常把这三者联合起来称作三元组信息。按照官方设定的协议规则得到UserName、PassWord、ClientID、IP这几个参数.

* 设备属性上报服务器，设置服务器上设备属性和从服务器接收设备属性分别应该使用哪种topic？

上报用"judge/0a4c";

|  |  |
| --- | --- |
| **Alink JSON** | * 请求Topic：/sys/{productKey}/{deviceName}/thing/event/property/post * 响应Topic：/sys/{productKey}/{deviceName}/thing/event/property/post\_reply |

设置服务器上设备属性:

通过调用[SetDeviceProperty](https://help.aliyun.com/document_detail/69579.htm#doc-api-Iot-SetDeviceProperty)或[SetDevicesProperty](https://help.aliyun.com/document_detail/96243.htm#doc-api-Iot-SetDevicesProperty)下发设置属性指令到设备。

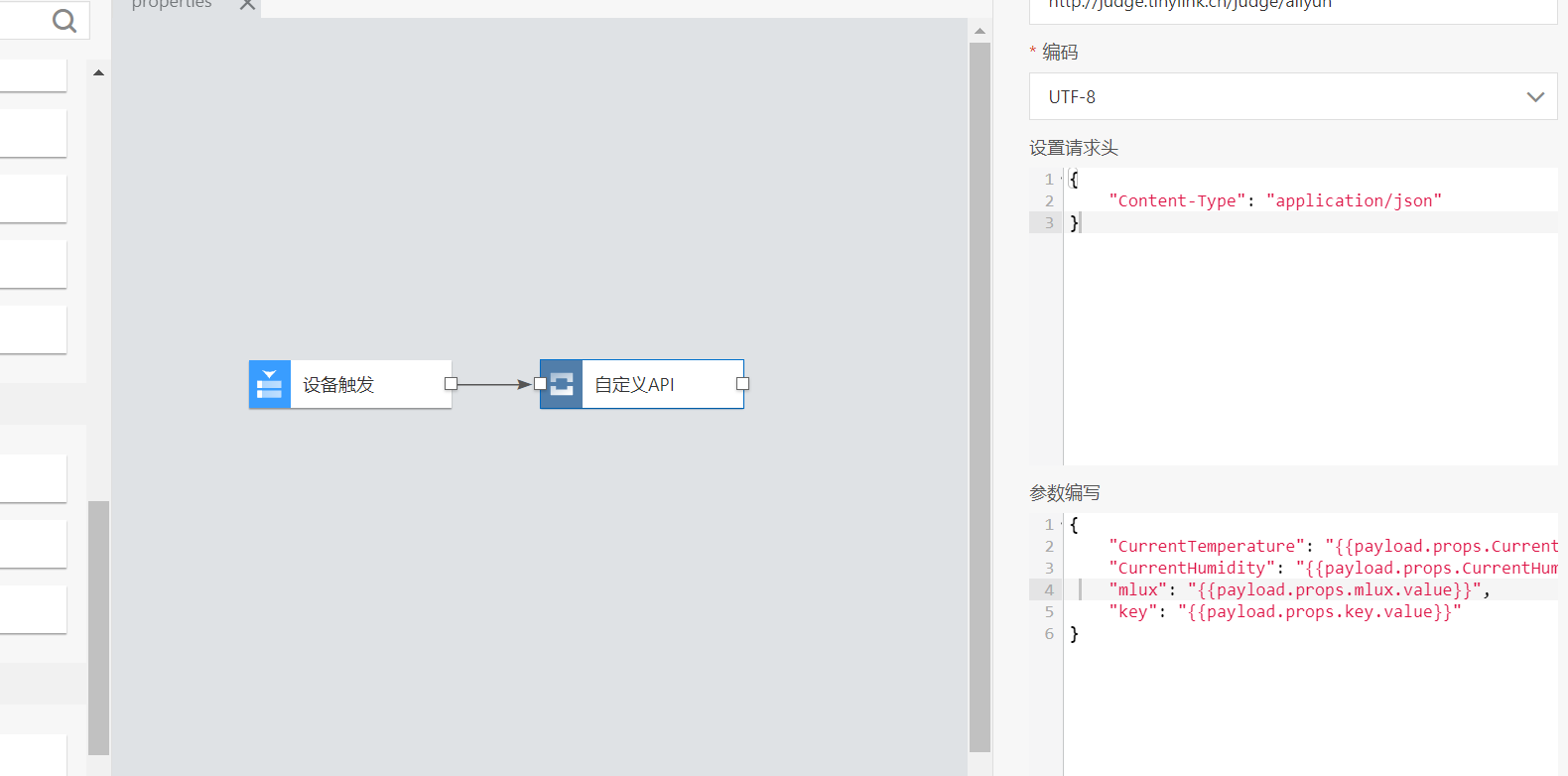
| **数据格式（下行）** | **请求和响应Topic** |
| --- | --- |
| **透传/自定义** | * 请求Topic：/sys/{productKey}/{deviceName}/thing/model/down\_raw * 响应Topic：/sys/{productKey}/{deviceName}/thing/model/down\_raw\_reply |
| **Alink JSON** | * 请求Topic：/sys/{productKey}/{deviceName}/thing/service/property/set * 响应Topic：/sys/{productKey}/{deviceName}/thing/service/property/set\_reply |

从服务器接收设备属性:

设备服务调用（异步调用）

|  |  |
| --- | --- |
| **透传/自定义** | * 请求Topic：/sys/{productKey}/{deviceName}/thing/model/down\_raw * 响应Topic：/sys/{productKey}/{deviceName}/thing/model/down\_raw\_reply |
| **Alink JSON** | * 默认模块   + 请求Topic：/sys/{productKey}/{deviceName}/thing/service/{tsl.service.identifier}   + 响应Topic：/sys/{productKey}/{deviceName}/thing/service/{tsl.service.identifier}\_reply * 自定义模块   + 请求Topic：/sys/{productKey}/{deviceName}/thing/service/{tsl.functionBlockId}:{tsl.service.identifier}   + 响应Topic：/sys/{productKey}/{deviceName}/thing/service/{tsl.functionBlockId}:{tsl.service.identifier}\_reply |

* 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LinkLab哪些地方值得完善？

**起到的作用:**

1. 可以高亮编辑cpp文件

2. mqtt上传, 在阿里云上添加一个服务 把收到的节点信息转发给判题系统

**值得完善的地方**

1. Webide 输出看不到, 不知道输出为什么不对. 它没有explicitly告诉我们 end了系统就判定程序结束了,导致没有输出. 也不告诉我们输出到底哪里不一致.

2. 云端编辑器没有最基本的检查, 比如分号检查.

3. 它没有说清楚要批量添加.

* 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: 可以让熟悉Arduino的工程师快速上手.

缺点: **串口事件不能实时响应**。

* 数据可以用HTTP传输吗？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？

可以,

MQTT特点:

  MQTT： TCP 长链接， HTTP： TCP

 MQTT： 发布订阅， HTTP：  request/ response

轻量级的通信协议比如MQTT，而不是HTTP

那为什么选择MQTT，而不是HTTP呢？

因为HTTP 协议的首部（header）比较大，而且每次发送数据都要发送一个数据包来连接/断开TCP，因此发送的数据越多，数据总通信量也就越大；

而MQTT 的首部比较小，还能在维持TCP 连接的同时，进行下一次数据的收发，所以比起HTTP，它更能抑制数据总通信量。

除此之外，在使用MQTT 时还要注意一点，即应该一边维持MQTT的TCP 连接，一边进行数据的发送和接收。因为MQTT 是通过维持TCP 连接来削减通信量的，所以要是每次进行数据通信都断开TCP 连接，MQTT 就会跟HTTP 一样在每次发送数据时都执行连接和断开处理，结果反而会增加通信量。

MQTT是订阅式服务模式,broker可以及时push消息给client。若http实现相同功能，需要client不停polling服务端。  
MQTT支持1对多消息发布方式。  
HTTP是请求回复模式。一问一答，服务端不会主动发送给客户端。

|  |  |  |
| --- | --- | --- |
| HTTP | 应用广泛，开发成本低 | 1，服务端无法主动推送数据给client; 2，采用key-value格式，占用存储和计算资源高； |
| MQTT | 小型传输，资源开销小 | 发布订阅模式，提供1对多消息发布，解除应用程序耦合 |

# 讨论、心得

实验过程中遇到的困难，得到的经验教训，对本实验安排的更好建议（看完请删除本句）

|  |  |  |  |
| --- | --- | --- | --- |
| 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) | | | 1 |
| 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) | | | 2 |
| 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 | | 第9点不应该放在第10点前面, 因为写了代码还是不知道怎么做, 应该先完整阐述一遍要做什么, 然后再分解每一点讲. 先开发服务再写代码再上传. 难点就是找不到它这些按钮在哪里, 建议录制一个视频让人来做. | |
| 列出你失败的case,并解释失败的原因 | | 第三步创建的产品, 找不到激活的地方 | |
| 意见反馈 | | | |
| LinkLab系统 | 感觉就是按着图做一遍没学到什么可迁移技能, 大部分时间都在寻找按钮. 图有些也没说清楚. | | |
| LinkLab硬件 | 无问题 | | |
| 系统Bug | 无问题 | | |
| 其他 | 无问题 | | |