浙江大学

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：	应承峻
学　　院：	计算机学院
系：	软件工程系
专　　业：	软件工程
学　　号：	3170103456
指导教师：	高艺

2019 年 10 月 11 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：夏林轩 李婧瑜 周凌畅 刘乐为 实验地点：计算机网络实验室

一、 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；

二、 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：

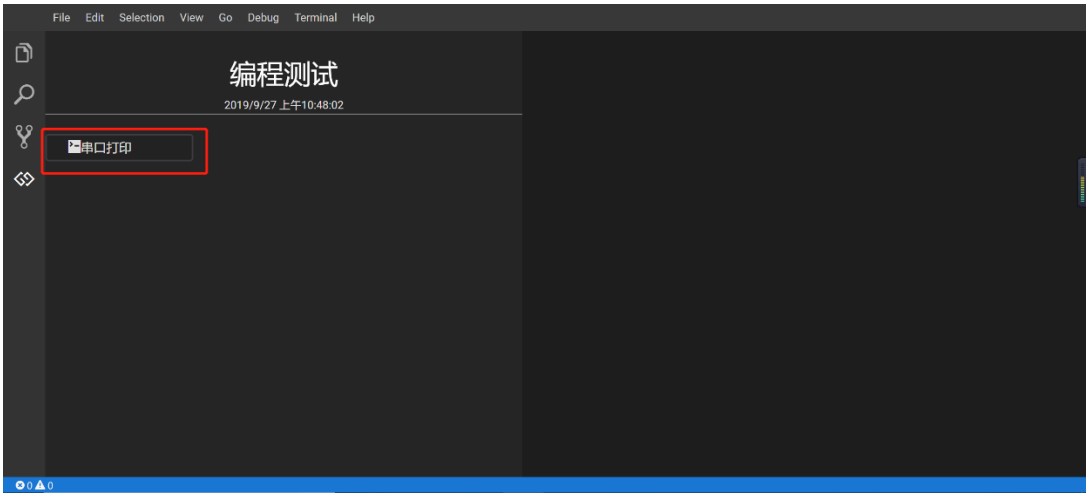


图 2.4　WebIDE 页面

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）。

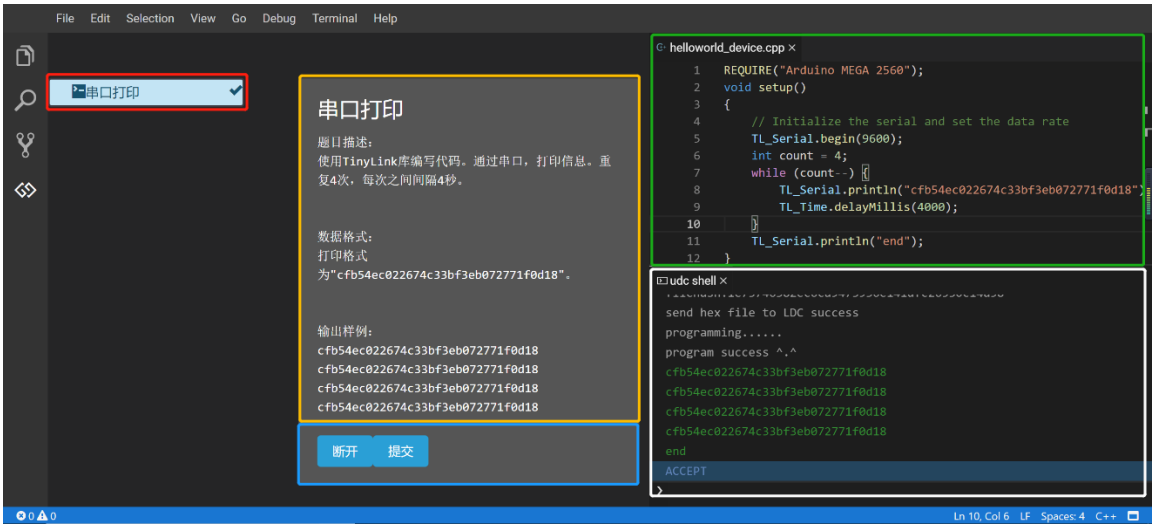


图 2.5　WebIDE 页面布局

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。

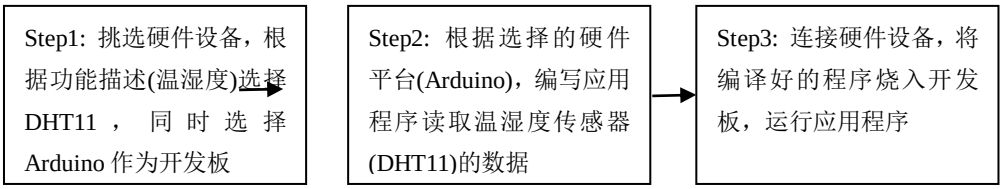


图 2.6 传统物联网应用开发流程

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。

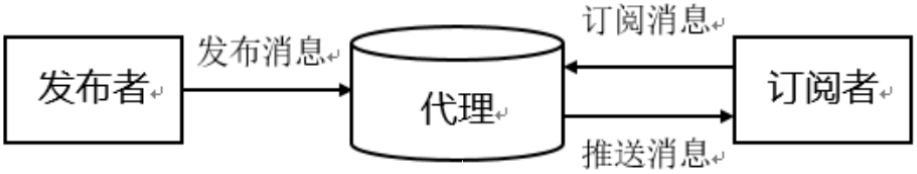


图 2.7 MQTT 协议实现方式

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。

三、 主要仪器设备

- PC(Windows/Linux/macOS)；

四、 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

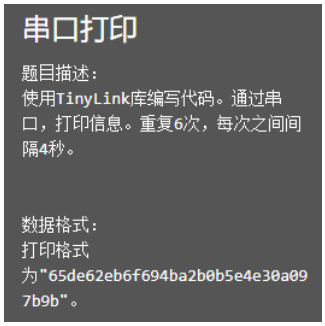


图 4.2　“串口打印”题目描述

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）

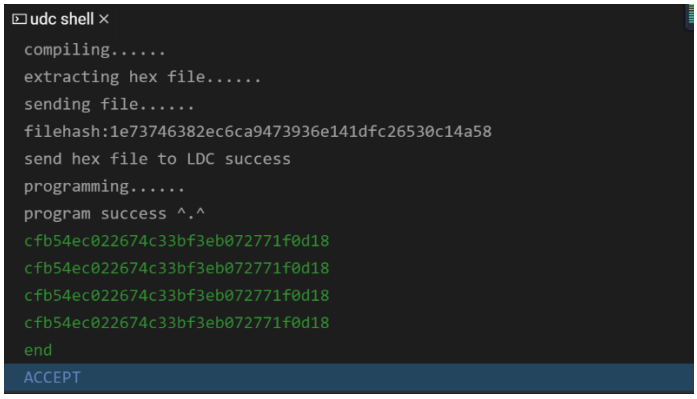


图 4.4　实验题提交后的日志和判题结果

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。

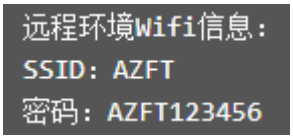


图 4.5　“MQTT 通信”实验题目的 WiFi 信息

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。

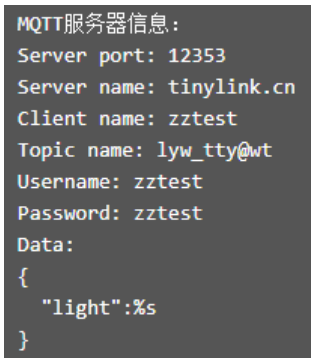


图 4.6　“MQTT 通信”实验题目的 MQTT 服务器信息

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

3. 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

属性	温度	CurrentTemperature	double (双精度浮点型)	取值范围：-40 ~ 120
属性	湿度	CurrentHumidity	double (双精度浮点型)	取值范围：0 ~ 100
属性	光照度	mlux	double (双精度浮点型)	取值范围：0 ~ 65535
属性	判题密钥	key	text (字符串)	数据长度：20

图 4.9　设备新增的功能

4. 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

5. 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

6. 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。

7. 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）。
8. 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() {

}
```

9. 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。

10. 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

五、　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

源代码：

helloworld_device.cpp

```
REQUIRE("Arduino MEGA 2560");
void setup()
{
    // Initialize the serial and set the data rate
    TL_Serial.begin(9600);
    int count = 4;
    while (count--) {
        TL_Serial.println("e5fc4173b7544ca881c796d7c0ded869");
        TL_Time.delayMillis(4000);
    }
    TL_Serial.println("end");
}

void loop(){
}

```

运行日志：

udc shell

```
*connect to server success, server ip is 47.97.253.23
*allocated devices:tinylink_platform_1-55834232832351605 2D1
*compiling......
*extracting hex file......
*sending file to LDC......
*send hex file to LDC success
*burning......
*burn success ^.^
e5fc4173b7544ca881c796d7c0ded869
e5fc4173b7544ca881c796d7c0ded869
e5fc4173b7544ca881c796d7c0ded869
e5fc4173b7544ca881c796d7c0ded869
end
ACCEPT
```

- 实验题“MQTT 通信”源代码及运行日志截图

源代码：

helloworld_device.cpp

```
TL_MQTT mqtt;
int port = 12353;
char servername[] = "tinylink.cn";
char clientname[] = "zztest";
char topicName[] = "judge/2205";
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 < 2; i++){
        TL_Humidity.read();
        String data = String("{") + "\"humidity\":" + TL_Humidity.data() + String("}");
        char buf[100];
        data.toCharArray(buf, 100);
        TL_Serial.print("Humidity data is ");
        TL_Serial.println(TL_Humidity.data());
        int res = mqtt.publish(topicName, buf, strlen(buf));
        TL_Time.delayMillis(3000);
    }
    TL_Serial.println("end");
}

void loop() {

}
```

运行日志：

udc shell

```
*compiling......
*extracting hex file......
*sending file to LDC......
*send hex file to LDC success
*burning......
*burn success ^.^
begin
0E202600001101nan
Humidity data is 0.00
0E9A2600002201judge/2205
0E813300001101nan
Humidity data is 0.00
0EDB3300002201judge/2205
end
ACCEPT
```

- 实验题“阿里云物联网平台”源代码及运行日志截图

源代码：

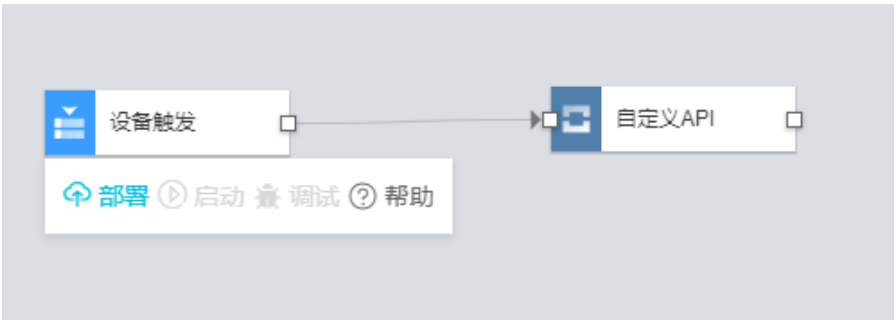
```
TL_MQTT mqtt;
int port = 1883;
char serverName[] = "a1REO4kzR1R.iot-as-mqtt.cn-shanghai.aliyuncs.com";
char clientName[] = "FESA234FBDS24|securemode=3,signmethod=hmacsha1,timestamp=789|";
char topicName[] = "/sys/a1REO4kzR1R/Z7SbnLZtqyU4Uz0JTSin/thing/event/property/post";
char userName[] = "Z7SbnLZtqyU4Uz0JTSin&a1REO4kzR1R";
char password[] = "7b7b3d2046d96efbed4cf885d096263857f967df";
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 += "\"MTYsMjg5LGE1ZTVjOTcw\"";
    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");
}
```

运行日志：

```
*compiling......
*extracting hex file......
*hexfile upload success
*burning......
*burning /dev/tinylink_platform_1-55834323832351601B0......
*program success
0
0EAD3000001201712.00
0EC1310000110125.00
0ED63200001001nan
{"id" : "123", "version":"1.0", "params" : {"CurrentTemperature":0.00, "CurrentHumidity":25.00, "mlux":712.00,
"key":"MTYsMjg5LGE1ZTVjOTcw"},"method":"thing.event.property.post"}
0EA43400002201/sys/a1REO4kzR1R/Z7SbnLZtqyU4Uz0JTSin/thing/event/property/post
0
end
ACCEPT
```

- 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



六、 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？

  物联网设备端快速开发系统 LinkLab，能够解决物联网应用开发困难（由于成本、大小、能量等因素限制）的难题。TinyLink 系统提出的自顶向下的开发模型，使得开发者只需编写应用程序代码，即可自动生成满足应用需求的设备硬件配置和编译后的二进制代码，大大加速了基于物联网设备端的软硬件开发。

  个人认为整个开发环境还是不错的，但是前面的引导页面和注册页面仍然需要完善，感觉前面的前端有点像临时糊出来的，还有就是教程部分的建设还不是特别完善。建议为初学者提供更加详细的指导。

- 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：将设备初始化和程序执行分开成两个不同的部分，使得程序比较清晰，同时也能够使得程序可以实时更新。

  缺点：Loop 死循环可能会导致计算资源浪费，即如果某一时刻没有任何程序被执行但 loop 仍然运行则会使得时间开销却没有任何成果。

- 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？

  MQTT 是一种能够在低带宽高延迟不可靠的网络下进行数据相对可靠传输的应用层协议；能够提供了不同的消息发送的服务质量以简化消息发送不同服务质量的工作量。

  由于许多物联网应用需要不间断地发送数据，因此为了节约流量往往选择 MQTT 协议，原因是 MQTT 协议能够最大限度地减少传输流量，相比与数据率量较大地 HTTP 协议更适合物联网应用。

七、 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)	4
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)	4
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	IoT Studio 入门中，需要设置的项目很多，一不小心就会弄错，因此需要足够细心。
列出你失败的 case, 并解释失败的原因	“MQTT 通信”实验题中，原来以为输出的 Begin 和 end 没有用就把它注释掉了，结果不通过。

意见反馈	
LinkLab 系统	点击连接后没有分配设备，建议完善异常提醒机制。
LinkLab 硬件	无
系统 Bug	有的时候提交反复出现 wrong answer，要重新登陆后才行
其他	无