浙江大学

本科实验报告

课程名称：计算机网络基础

姓　　名：

学　　院：计算机学院

系：计算机科学与技术

专　　业：计算机科学与技术

学　　号：

指导教师：

2018 年　　05 月　　06 日

浙江大学实验报告

课程名称：计算机网络基础 实验类型：操作实验

实验项目名称：使用 Tinylink 系统体验 IoT 应用的开发

学生姓名：　专业：计算机科学与技术 学号：

同组学生姓名：　指导老师：

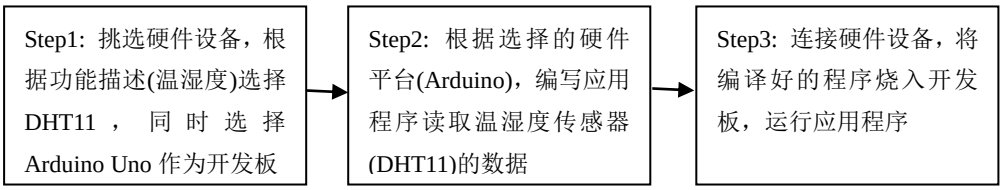
实验地点：计算机网络实验室 实验日期：2018 年 05 月 06 日

一、 实验目的和要求：

- 了解常用的硬件平台(Arduino, LinkIt One)；
- 熟悉 Tinylink 进行 IoT 应用开发的流程。

二、 实验内容和原理

- 传统的 IoT 应用开发流程包括硬件选择和应用开发。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下列流程：



- 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，根据用户自定义代码自动生成硬件配置及相应的应用程序。
- Tinylink 提供多种硬件平台(Arduino, LinkIt, BeagleBone, Raspberry Pi, Mbed)和丰富的传感器(DHT11, Grove UART WiFi, SDS018 ...)，最终会根据用户需求生成价格最优的解决方案，同时也提供一些满足用户需求的推荐方案。
- 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 的编程环境。使用 Tinylink 语言编写用户自定义代码，Tinylink 系统会根据上传的代码生成硬件配置和应用程序。
- 使用 Tinylink 开发 IoT 应用，将设备节点连接到远程的云平台，在终端（手机 APP、电脑 Client APP）实现数据的查看以及远程设备的控制。

三、 主要仪器设备

- PC
- 手机
- Arduino Uno、LinkIt One、传感器

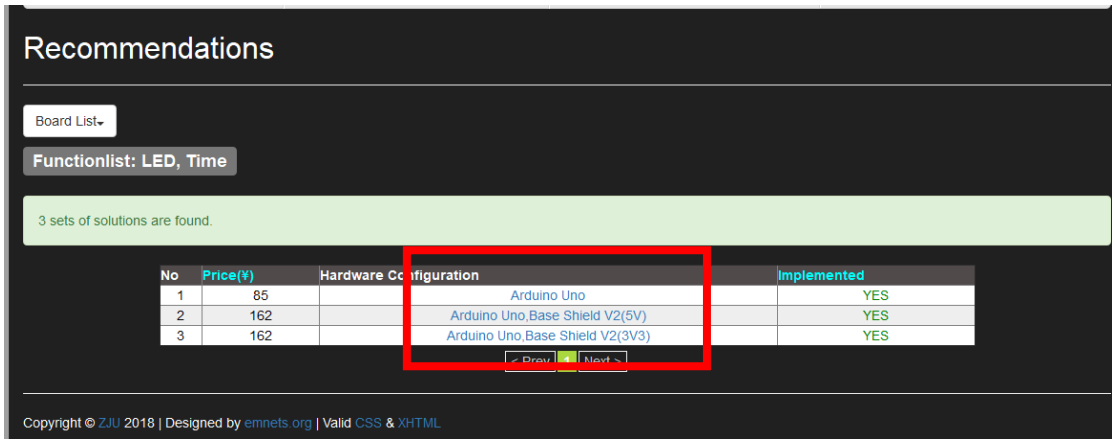
四、 操作方法与实验步骤

- 下载 Tinylink Client
- 下载 Tinylink APP 并安装到手机(目前仅支持 Android)
- 安装 Arduino 和 LinkIt One 的驱动
- 结合 Tinylink API 手册，使用 Tinylink 语言编写用户自定义代码
- 打开 Tinylink 桌面应用，登录后上传代码，根据硬件配置结果选择开发板，将开发板连接到 PC，烧写程序，根据硬件配置选择传感器并连接至开发板
- 如果需要编写移动端应用逻辑，打开 Tinylink 桌面应用，对移动端应用逻辑进行设置
- 根据上述 Tinylink 开发流程完成以下案例：

Level1	Case1	完成 LED 灯点亮实验，控制 LED 灯周期性闪烁
	Case2	完成温度测量实验，使用串口间隔输出温度数值
Level2	Case3	完成蓝牙点灯实验。在手机端安装蓝牙 APP(nRF Connect for Mobile)，利用蓝牙 4.0 协议，通过手机控制开发板 LED 灯的亮灭。(注意：蓝牙设备名称命名为学号)
	Case4	完成室内环境监测实验。使用传感器测量室内温度、湿度及 PM2.5 含量。将测得数据使用 get 请求周期性上传到云端（可以使用客户端 Generate uploading configuration 选项中的 Generate URL 生成对应链接)，通过 TinyLink 手机端 APP 实时获取室内环境状态。
	Case5	完成土壤环境监测实验。使用传感器测量土壤湿度、温度、光照，将测得数据使用 MQTT 协议周期性上传到云端（可以使用客户端 Generate uploading configuration 中的 Generate MQTT 生成对应参数)，通过 TinyLink 手机端 APP 实时获取土壤环境状态。(注意：由于 Arduino 仅有一个物理串口，在本 Case 中，不要执行串口打印操作，否则，系统无法选出满足条件的设备)
Level3	Case6	完成智能温度控制应用。基于 MQTT 协议，在移动端实现温度控制。主要实现两个功能。一是温度高于 30℃，自动打开风扇，二是用户在手机端可查看风扇的开关状态，控制风扇的打开与关闭。(注意：两个功能均通过移动端设置)

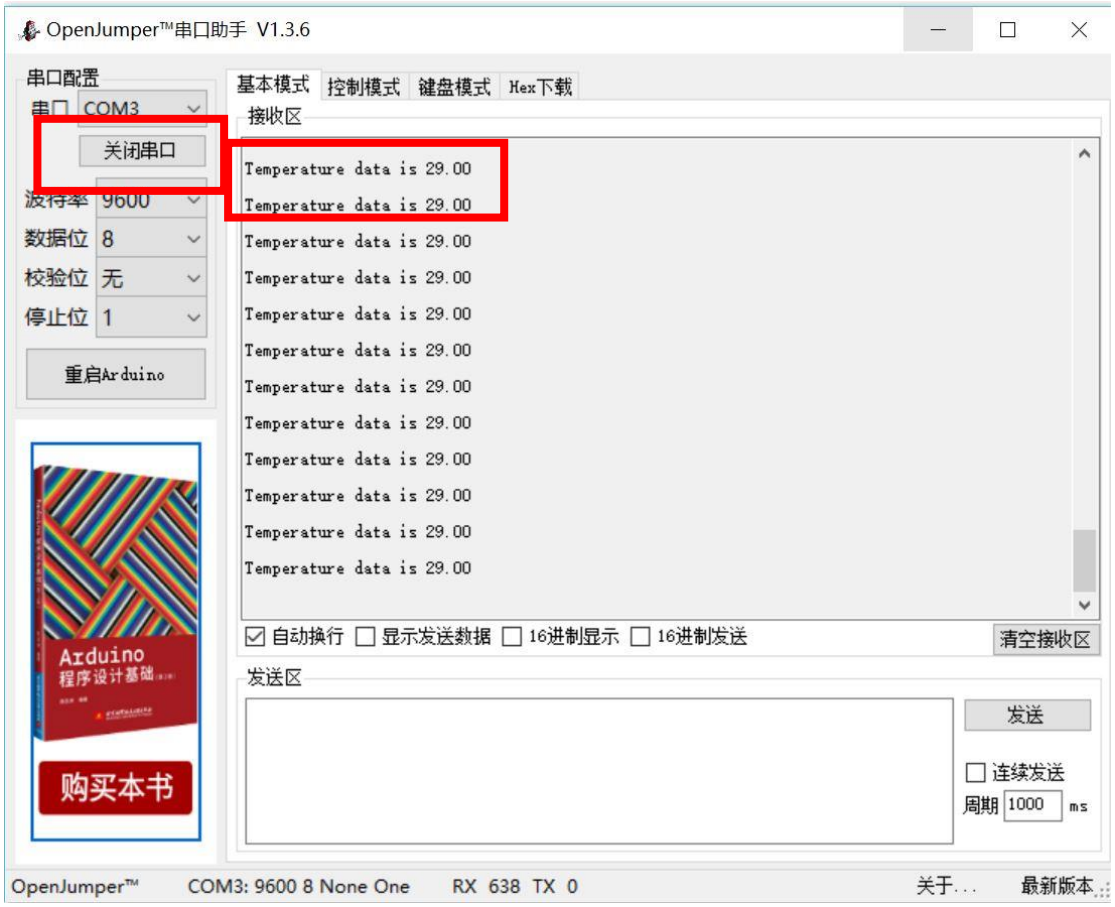
五、 实验数据记录和处理

- **在 case 1 中，根据 recommendations，列出哪些设备可以提供 LED 功能。**



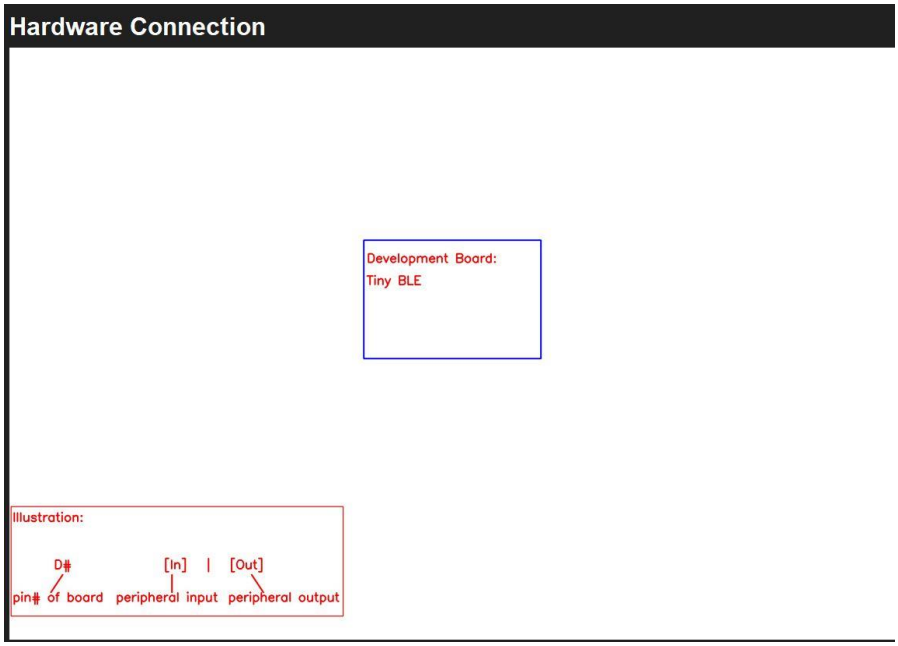
上传代码后查看 recommendations，选择 Arduino Uno 后出现三种设备，由上图可知 Arduino Uno 和 Base Shield（5V & 3V3）两种都可以实现 LED 功能。在本次实验中，我们组采用了 Arduino Uno 完成该实验。

- **记录 case 2 的串口显示数据。**



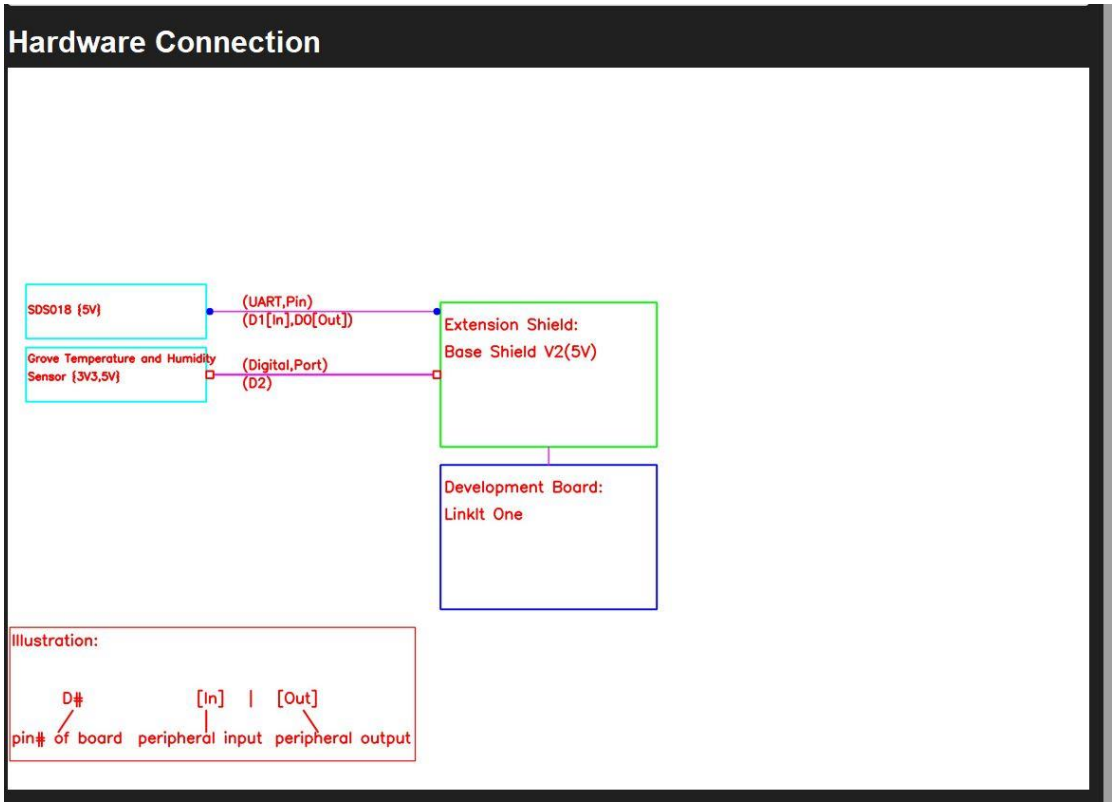
在本实验中首先要记得打开串口，这样才可以接受到数据。在接受区显示接收到的温度数据。

- **在 case 3 中，列出所使用的设备。**



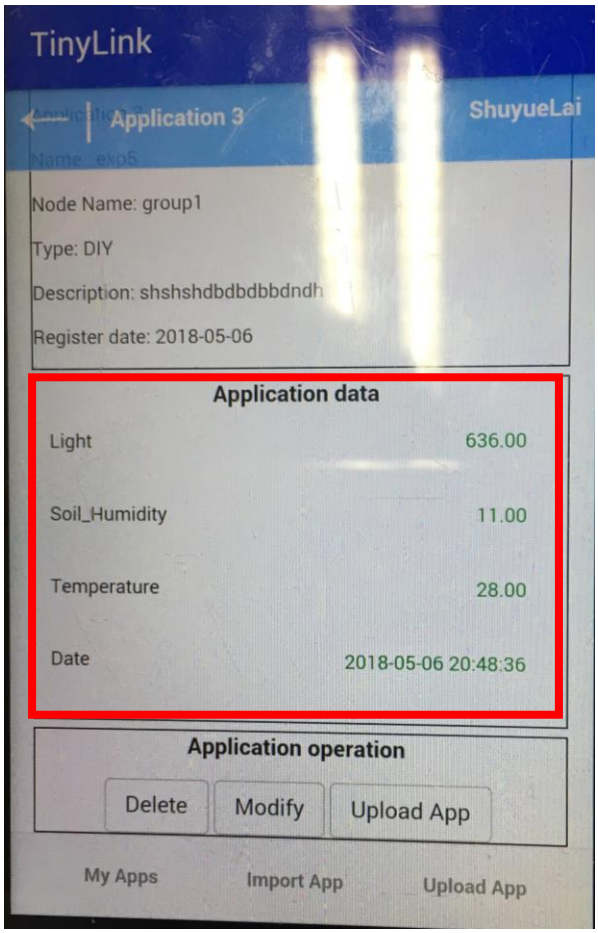
Arduino Uno + Base Shield V2 + Grove BLE

- **在 case 4 中，根据 recommendations，列出哪些设备可以提供 WiFi 功能。**



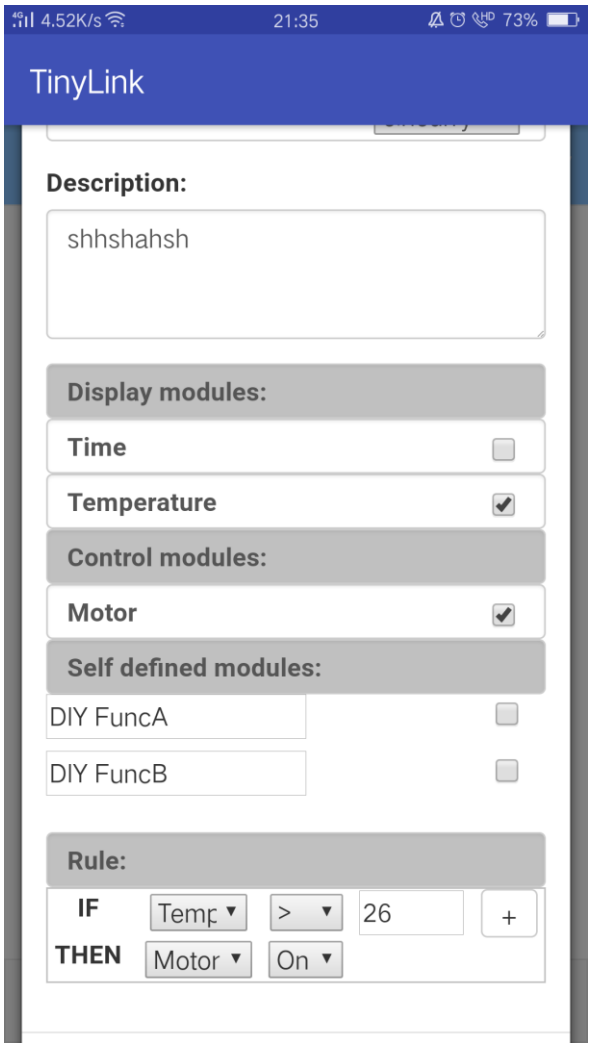
Linklt One + Base Shield V2(5V) + Grove Temperature and Humidity Sensor + SDS018

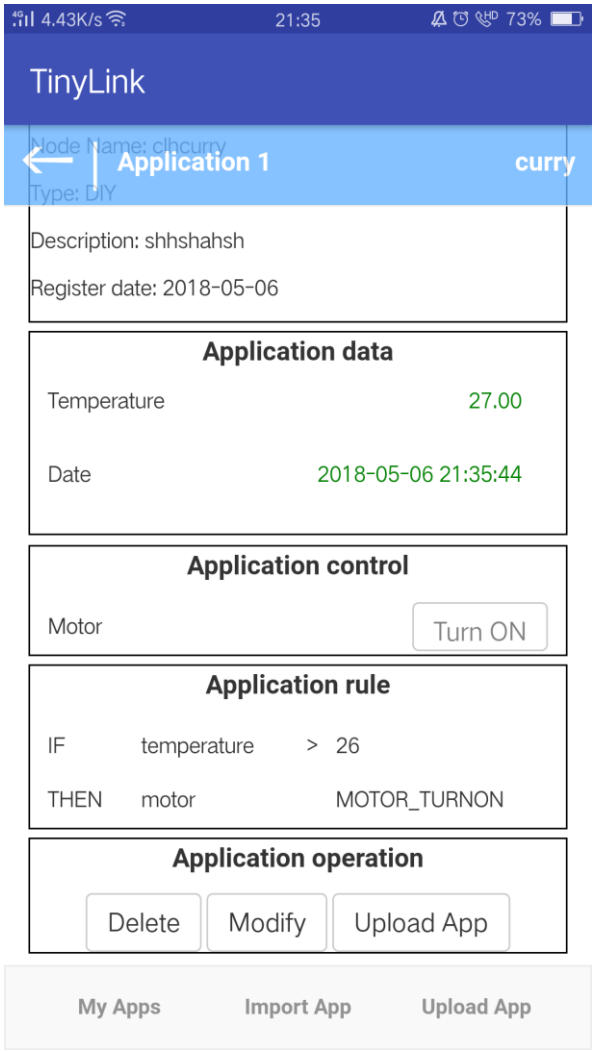
- **记录 case 5 的手机显示数据。**



在手机客户端显示土壤的相关信息。

- **记录 case 6 的手机显示数据，简述代码工作流程。**


4.52K/s
21:35
73%
TinyLink
Description:
shhshahsh
Display modules:
Time
Temperature
Control modules:
Motor
Self defined modules:
DIY FuncA
DIY FuncB
Rule:
IF
Temp
>
26
+
THEN
Motor
On

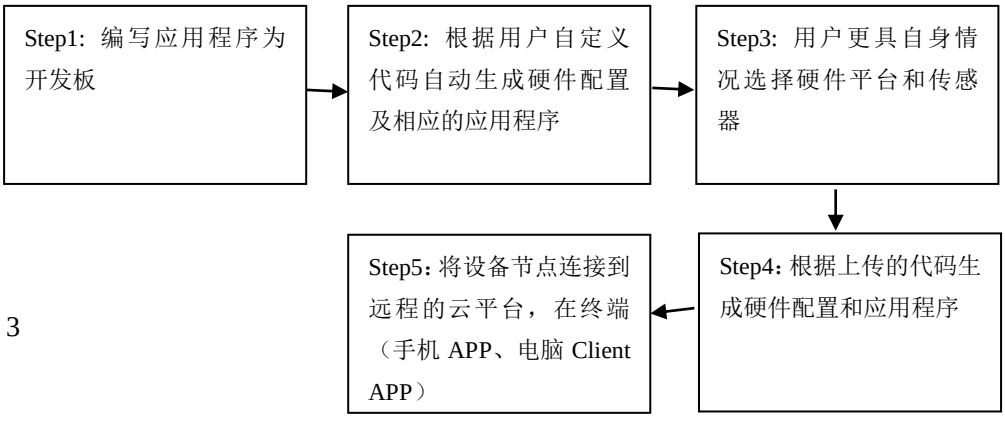


首先配置好 Wifi 和 Mqtt，然后进行循环读取温度和风扇状态信息，并发布；然后对手机操作进行判断，从而控制风扇。

```
TL_MQTT mqtt;

int port = 1883;
char serverName[]= "10.214.149.119";          // Server ip
char clientName[] = "3150105275";              // Device ID
char topicName[] = "clhcurry@wt";              // Topic name
char userName[] = "3150105275";                // Product ID
char password[] = "clhclh19971123";           // Authentication information

void setup() {
      TL_WiFi.init();
      bool b = TL_WiFi.join("EmNets-301","eagle402");
      mqtt = TL_WiFi.fetchMQTT();
      int a = mqtt.connect(serverName, port, clientName, userName, password);
      mqtt.subscribe("clhcurry@rt", messageArrived, 0);
```

```
}

void messageArrived(MQTT::MessageData& md){
    MQTT::Message &message = md.message;
    char res[20];
    strncpy(res, (char*)message.payload, 19);
    if(strncmp(res, "MOTOR_TURNON", strlen("MOTOR_TURNON")) == 0){
        TL_Motor.turnOn();
    }else if(strncmp(res, "MOTOR_TURNOFF", strlen("MOTOR_TURNOFF")) == 0){
        TL_Motor.turnOff();
    }
}

void loop() {
    TL_Temperature.read();
    String data = "{";
    data += "\"temperature\":";
    data += TL_Temperature.data();
    data += ",\"motor\":";
    data += TL_Motor.state();
    data += "}";
    char buf[100];
    data.toCharArray( buf,100 );
    int res = mqtt.publish(topicName, buf, strlen(buf));
    mqtt.yield(2000);
    TL_Time.delayMillis(1000);
}
```

六、 实验结果与分析



- 结合上述实验，比较 HTTP 协议与 MQTT 协议的异同。

同：两者都是用来进行通信的协议；

异：HTTP: request/response, 通过 URL 即链接和 get 等请求操作来完成信息通信；

MQTT：publish/subscribe, 通过生成对应参数来完成信息的通信；

七、 讨论、心得

各项评分（1-差，2-可以容忍，3-满意，4-优秀）	
Tinylink API 易用性 (根据 API 手册完成特定需求的用户自定义代码)	3
Tinylink 硬件易用性 (根据连接关系图挑选对应的硬件设备并完成节点组装)	4
Tinylink 系统易用性 (完成 IoT 应用的整个流程，硬件生成、程序烧写等)	3
TInylink 系统鲁棒性 (系统流畅、系统容错和系统 Bug)	3

实验感想	
简述实验中最难的 case 及其难点	Case 6 在代码的细节上没有有所遗漏，例如 mqtt.yield(2000); 总的来说是对 TinyLink 编程还不是非常熟悉。
列出你失败的 case, 并解释失败的原因	没有失败的，撒花~

意见反馈	
Tinylink API	挺好的
Tinylink 系统	Choose File 之后 File 是在黑色背景之上显示黑色文件名，有点不友好
系统 Bug	暂没有还没有发现
其他	