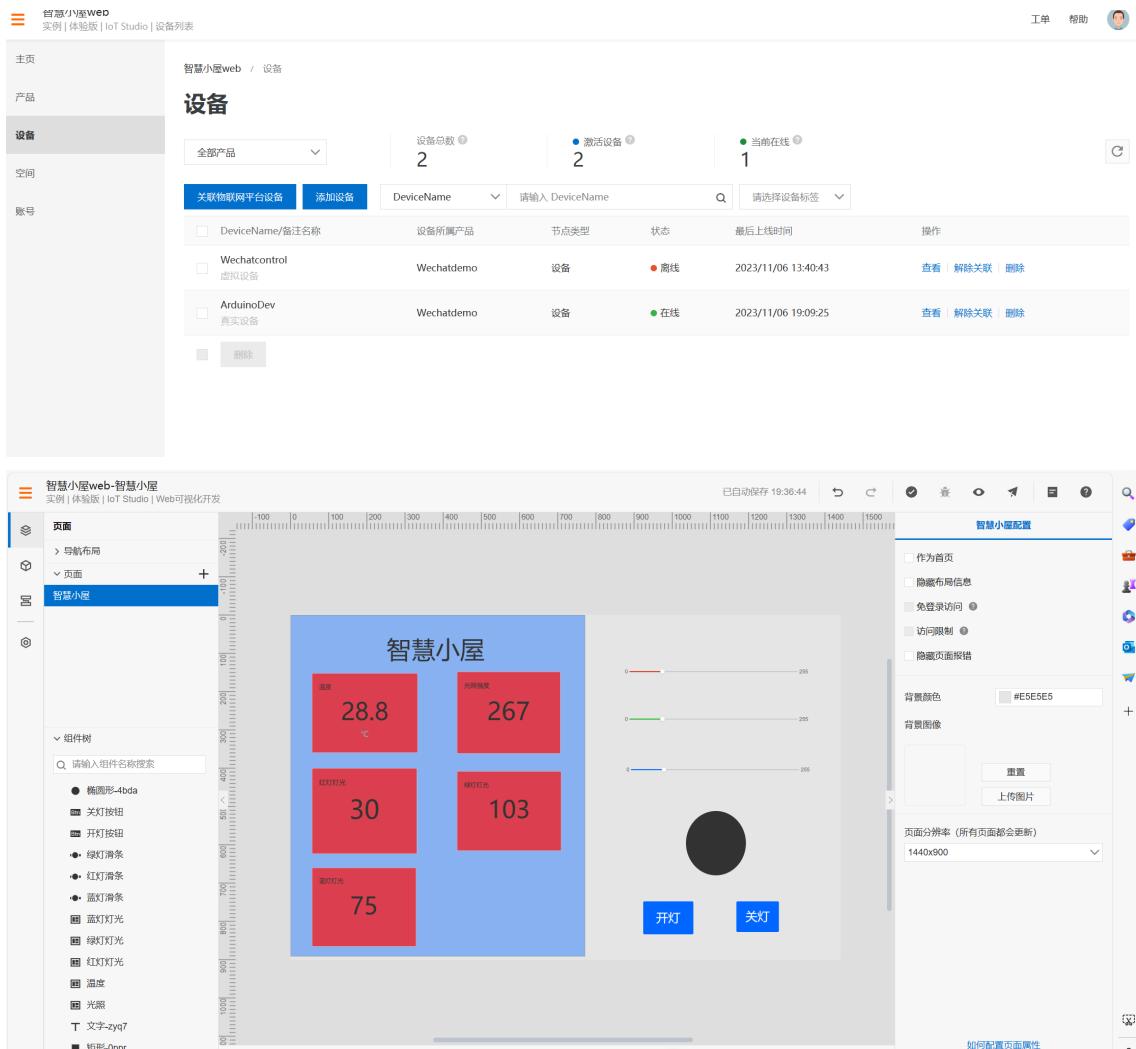


作业六 远程WEB页面控制三色灯

1.WEB页面初始化

- 打开IOT Studio中的WEB可视化开发，新建web应用并关联设备



- 设计web界面，通过拖动左侧的卡片、滑条及按钮等到web页面上，在这里我们引入了分别代表温度、光照、RGB的五个卡片来回报数据，三个滑条来调光，两个按钮来开关灯，以及一个混色表现圆形

均可以用右侧的样式进行修改

2.对页面中元素进行关联

我们希望对五个数据进行上报，但程序段中只有对温度光强的upload函数，因此对程序要进行改进

- 仿照upload () 函数的方式编写UploadRGB函数

```
bool UploadRGB()  
{  
    bool flag;  
    int len;  
    cleanBuffer(ATcmd, BUF_LEN);  
    sprintf(ATcmd, BUF_LEN, AT_MQTT_PUB_SET, ProductKey, DeviceName);  
    flag = check_send_cmd(ATcmd, AT_OK, DEFAULT_TIMEOUT);
```

```

    cleanBuffer(ATdata,BUF_LEN_DATA);
    len =
    sprintf(ATdata,BUF_LEN_DATA,JSON_DATA_PACK_4,ProductKey,DeviceName,light,colorRed,colorGreen,colorBlue);
    cleanBuffer(ATcmd,BUF_LEN);
    sprintf(ATcmd,BUF_LEN,AT_MQTT_PUB_DATA,len-1);
    flag = check_send_cmd(ATcmd,>,DEFAULT_TIMEOUT);
    if(flag) flag = check_send_cmd(ATdata,AT_MQTT_PUB_DATA_SUCC,20);
    Serial.println(ATdata);
    return flag;
}

```

注意其中JSON_DATA_PACK_4来自于前面宏定义中将light、ColorRed、ColorGreen、ColorBlue包含进去的数据包

```

#define JSON_DATA_PACK_4      "
{\\"id\\":\\"123\\",\\"version\\":\\"1.0.0\\",\\"method\\":\\"/sys/%s/%s/thing/event/property/post\\",\\"params\\":
{\\"light\\":%d,\\"colorRed\\":%d,\\"colorGreen\\":%d,\\"colorBlue\\":%d}}\r"

```

随后将UploadRGB () 函数写进loop内,同时使串口打印数据, 也即

```

if((millis()-timeStart)>10000)
{
    Upload();
    UploadRGB();
    timeStart=millis();
    Serial.print(F("Temperature = "));
    Serial.print(bmp.readTemperature());
    Serial.println(" *C");
    Serial.print(F("PhotoRes = "));
    Serial.print(analogRead(A2));
    Serial.println(F("Red = "));Serial.println(ColorRed);
    Serial.println(F("Green = "));Serial.println(ColorGreen);
    Serial.println(F("Blue = "));Serial.println(ColorBlue);
}

```

此时可将程序烧进板子, 使设备上线

- 以温度为例说明将卡片关联数据的过程, 点击温度卡片后选择配置数据源, 选择与ArduinoDev的数据相关联

数据源配置

X

选择数据源

设备



* 产品



Wechatdemo



* 设备

ArduinoDev (真实设备)

ArduinoDev (真实设备)



数据项

设备属性

* 属性



温度



格式参考

验证数据格式

确定

取消

帮助文档

配置完成页面如下

展示数据

● 已配置数据源

其他四个卡片类似

- 对滑条进行交互设置:以红色滑条为例, 首先在样式中设为取值范围0-255, 步长1

样式

交互



876 X

100 Y

0 °

480 W

96 H



组件名称

红灯滑条

 组件可见性

不透明度

100%

展示数据

配置数据源

▼ 数值范围

最小值

0

最大值

255

步长

1

▼ 两端标签

 是否隐藏

字号

14



颜色

#666666

粗细

标准

[如何配置滑条](#)

配置交互：选择值改变时设置设备属性，与红灯亮度关联

The screenshot shows a configuration dialog for a rule. At the top, it says "配置交互：选择值改变时设置设备属性，与红灯亮度关联". Below this, there's a section titled "交互1" (Interaction 1) with a trash can icon. The configuration details are as follows:

- 事件 (Event): 值改变 (Value Change)
- 动作1 (Action 1): 设置设备属性 (Set Device Property)
- 产品 (Product): Wechatdemo
- 设备 (Device): ArduinoDev (真实设备) (Real Device)
- 属性 (Property): 红灯亮度 (Red Light Brightness)
- 属性值 (Property Value): 动态来源 (Dynamic Source)

At the bottom left, there's a blue "+ 子动作" (Add Sub Action) button.

此时改变滑条位置已可以在阿里云日志收到反馈

The screenshot shows a log entry for a device event. It includes the following details:

- 查看详情 (View Details) button and close button (X).
- Topic: /sys/k09l88hcMxJ/ArduinoDev/thing/event/property/post
- 时间 (Time): 2023/11/06 20:01:47.258
- 内容 (Content): Text (UTF-8) dropdown menu. The selected content is:

```
{"id":"123","version":"1.0.0","method":"/sys/k09l88hcMxJ/Arduino Dev/thing/event/property/post","params":{"light":1,"ColorRed":111,"ColorGreen":39,"ColorBlue":113}}
```

A "复制" (Copy) button is also present next to the content.
- 关闭 (Close) button.

- 对按钮进行交互设置，与上类似，只需与灯开关这一属性关联即可

样式

交互

▼ 交互1

事件 点击

动作1 设置设备属性

产品 Wechatdemo

编辑 清除

设备 ArduinoDev (真实设备)

属性 灯开关

属性值 开

+ 子动作

This screenshot shows the 'Interaction' configuration dialog. It includes sections for 'Event' (set to 'Click'), 'Action' (set to 'Set Device Property'), 'Product' (set to 'Wechatdemo'), 'Device' (set to 'ArduinoDev (Real Device)'), 'Attribute' (set to 'Light Switch'), and 'Value' (set to 'On'). A '+' button for 'Sub Actions' is also visible.

点击开关灯收到反馈

查看详情

×

Topic /sys/k09l88hcMxJ/ArduinoDev/thing/event/property/post

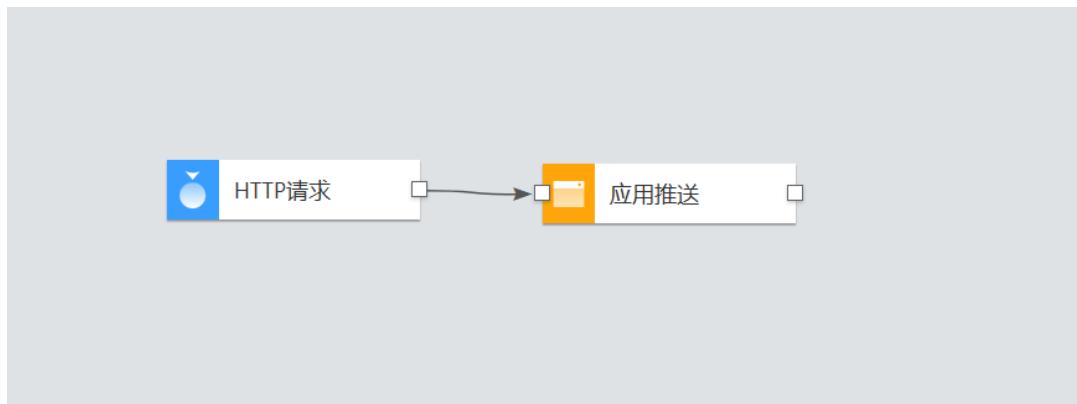
时间 2023/11/06 20:05:13.613

内容 Text (UTF-8) 复制
{"id": "123", "version": "1.0.0", "method": "/sys/k09l88hcMxJ/ArduinoDev/thing/event/property/post", "params": {"light": 1, "ColorRed": 111, "ColorGreen": 39, "ColorBlue": 113}}

关闭

- 混色表现装置的关联交互最为复杂，以下分几步进行阐述

- 创建一个业务逻辑，包括http请求和应用推送



http请求配置为三个入参rgb，应用推送设置为将上一个结点参数全部返回

配置完成后一定要先部署调试，这样才能发布，发布才可以被web逻辑调用

- 给滑条添加新的交互，当滑块值改变时，调用业务逻辑开发中的混色设置服务，将滑块数值赋值给rgb

The screenshot shows a configuration interface for a service action. At the top, there is a header with a trash icon. Below it, there are two main sections:

- 事件**: Set to "值改变" (Value Change).
- 动作1**: Set to "调用其它服务" (Call Other Service).
 - 接口来源**: Set to "业务逻辑开发" (Business Logic Development).
 - 服务名称**: Set to "混色设置" (Color Mixing).
 - 操作**: Buttons for "编辑" (Edit) and "清除" (Clear).

At the bottom left, there is a button labeled "+ 子动作" (Sub Actions).

服务配置

接口来源

业务逻辑开发

* 选择业务服务

[前往业务逻辑开发](#) C

混色设置

请求参数

静态参数 动态参数 [?](#)

* r [?](#)

值: 组件值: 红灯滑条



[参数来源](#)

* g [?](#)

值: 组件值: 绿灯滑条



[参数来源](#)

* b [?](#)

值: 组件值: 蓝灯滑条



[参数来源](#)

▶ 设置数据源默认值 [?](#)

▼ 返回结果 [?](#)

- 对圆形控件设置数据源，即业务逻辑开发中混色设置

数据源配置



选择数据源

接口



接口来源

业务逻辑开发



* 选择业务服务

[前往业务逻辑开发](#) C

混色设置



请求参数

静态参数 动态参数 [?](#)

* r [?](#) 自动更新 [?](#)

值:

组件值: 红灯滑条



[参数来源](#)

* g [?](#) 自动更新 [?](#)

值:

组件值: 绿灯滑条



[参数来源](#)

* b [?](#) 自动更新 [?](#)

值:

组件值: 蓝灯滑条



[参数来源](#)

填写初始值设置与数据过滤脚本

▾ 设置数据源默认值 ?

```
1 {
2   "b": 0,
3   "r": 0,
4   "g": 0
5 }
```

▶ 返回结果 ?

▾ 数据过滤脚本 ?

```
1 function _filter(data) {
2   // do something...
3   g = data["message"]["g"]
4   r = data["message"]["r"]
5   b = data["message"]["b"]
6   return {"b":b,"r":r,"g":g};
7 }
```

□ ▶ 数据表配置 ?

注：对业务逻辑可以这样理解，滑条设置的值作为http请求的入参触发业务逻辑运行，紧接着应用推送将rgb参数作为返回值，这个返回值又充当了混色模块的数据源，经过数据过滤后实现混色效果

- 到此应已完成混色过程，但发现问题，每次只能调一种颜色的光，其他光会自动熄灭，这是因为解析函数的问题。可以对解析函数做如下改进，使仅在数据中含有对应属性时才进行解析，避免了程序将未解析到数据自动置0

```
if strstr(tempdata, "ColorRed") !=NULL){ColorRed=doc["params"]
["ColorRed"];light=1;}
if strstr(tempdata, "ColorGreen") !=NULL){ColorGreen=doc["params"]
["ColorGreen"];light=1;}
if strstr(tempdata, "ColorBlue") !=NULL){ColorBlue=doc["params"]
["ColorBlue"];light=1;}
```

其中tempdata是将string转化为const char即

```
const char* tempdata=data.c_str();
```

strstr()函数用来检测字符串中是否有子字符串，如无返回NULL

```
char *strstr(const char *str, const char *sub_str)
{
    const char *str_local = NULL;
    const char *sub_str_local = NULL;

    if(!str || !sub_str)
    {
        printf("fun:%s param is error\n", __FUNCTION__);
        return NULL;
    }

    while(*str)
    {
        str_local = str;
        sub_str_local = sub_str;

        do
        {
            if(*sub_str_local == '\0')
            {
                return str;
            }

        }while(*str_local++ == *sub_str_local++);

        str += 1;
    }
    return NULL;
}
```

3.钉钉机器人设置

- 先在钉钉群内添加机器人，设置好对应关键词，获取webhook

设置

消息推送:

Webhook:

* 请保管好此 Webhook 地址, 不要公布在外部网站上, 泄露有安全风险
使用 Webhook 地址, 向钉钉群推送消息[查看文档](#)
设定后, 只有包含关键词的消息内容
才会被正常发送

* 安全设置 [?](#) 自定义关键词 [说明文档](#)

- 创建业务逻辑, 这里我们通过定时触发的方式进入逻辑内, 触发间隔设置为1min

* 节点名称 [如何使用该节点?](#)

定时触发

时间配置

* 触发模式

循环定时触发 (按照循环规则触发)



* 循环周期

分钟



* 时间间隔 (分钟)

1

* 选择生效时间

2023/11/06 00:00



选择结束时间

请选择日期和时间



永久生效

将产品与定时触发相连，设置为查询全部属性，方便钉钉机器人上报

* 节点名称 [如何使用该节点?](#)

Wechatdemo

产品功能定义

* 选择要控制的设备 [?](#)

ArduinoDev

* 选择操作类型 [?](#)

查询设备属性

* 选择功能定义模块

默认模块

* 选择要查询属性

全部属性

* 查询维度 [?](#)

快照值 历史值

钉钉机器人模块与产品相连，输入webhook，通过自定义修改发消息格式等

* Webhook ?

```
https://oapi.dingtalk.com/robot/send?access_token=e1c2194fc
```

配置方法

* 配置方法

使用模版 自定义

* 消息类型 ?

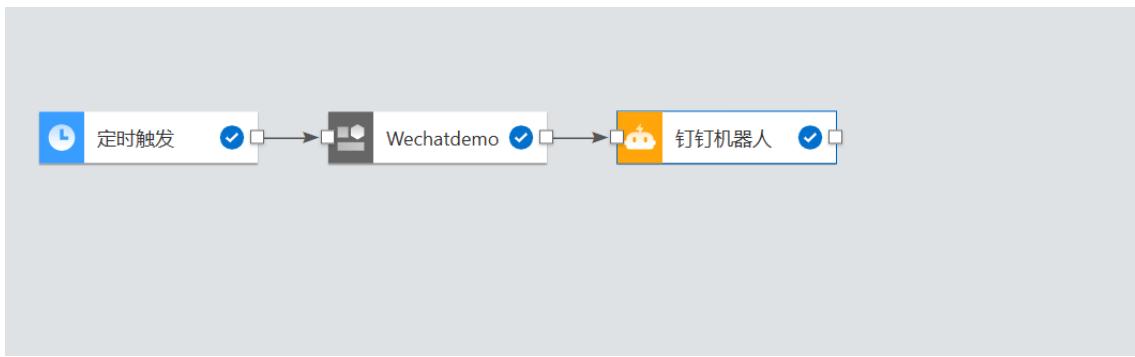
text

内容配置

```
1 {
2     "msgtype": "text",
3     "text": {
4         "content": "灯光状态: {{payload.data.0.value}}"
5     },
6     "at": {
7         "atMobiles": [
8             "156xxxx8827",
9             "189xxxx8325"
10        ],
11        "isAtAll": false
12    }
13}
```

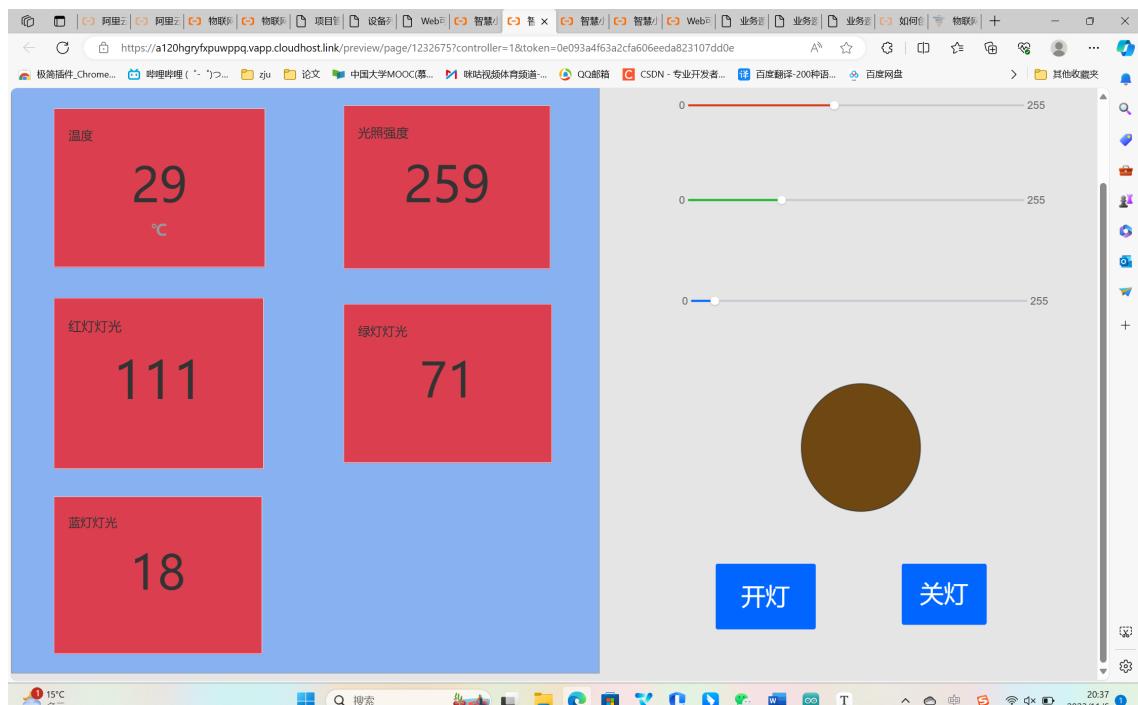
注意如何使用变量或属性，可打开对应结点日志，其中有分级的数据格式，payload表示数据源为上一个结点

完整业务逻辑如下，部署调试通过



4. 测试

- 打开web页面进行滑条调光，可观察到页面上反馈数据



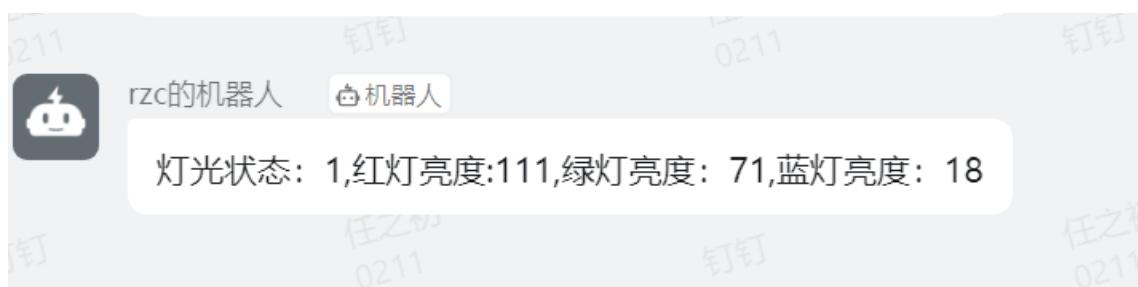
- 同时在阿里云日志发现记录

查看详情

Topic	/sys/k09l88hcMxJ/ArduinoDev/thing/event/property/post
时间	2023/11/06 20:38:47.236
内容	Text (UTF-8) <div style="margin-top: 10px;"> <code>{ "id": "123", "version": "1.0.0", "method": "/sys/k09l88hcMxJ/Arduino Dev/thing/event/property/post", "params": { "light": 1, "ColorRed": 111, "ColorGreen": 71, "ColorBlue": 18 } }</code> </div>

关闭

- 在钉钉机器人收到反馈



- 调光结果与混色比对

