ESP32 连接 MQTT Broker

淘宝店铺：

PC 端：

http://n-xytrt8gqu585po94mwj5atokcyd4.taobao.com/index.htm

手机端：

https://shop.m.taobao.com/shop/shop_index.htm?sellerId=755668508&shopId=104493595&inShopPageId=423890608&pathInfo=shop/index2

资料下载地址：

链接：https://pan.baidu.com/s/1kCjD8yktZECSGmHomx_veg?pwd=q8er
提取码：q8er

源码下载地址：

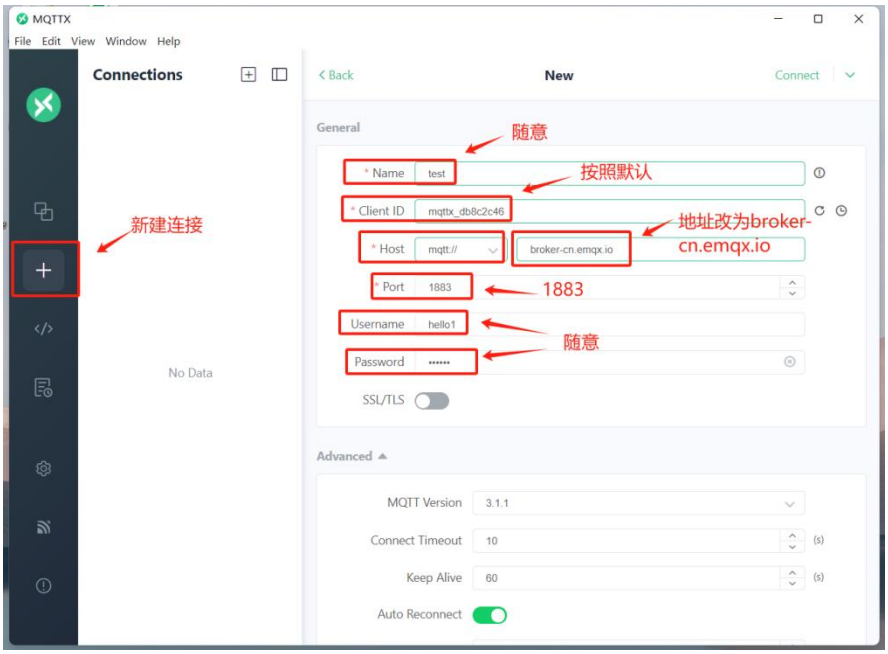
https://gitee.com/vi-iot/esp32-board.git

一、MQTT Broker

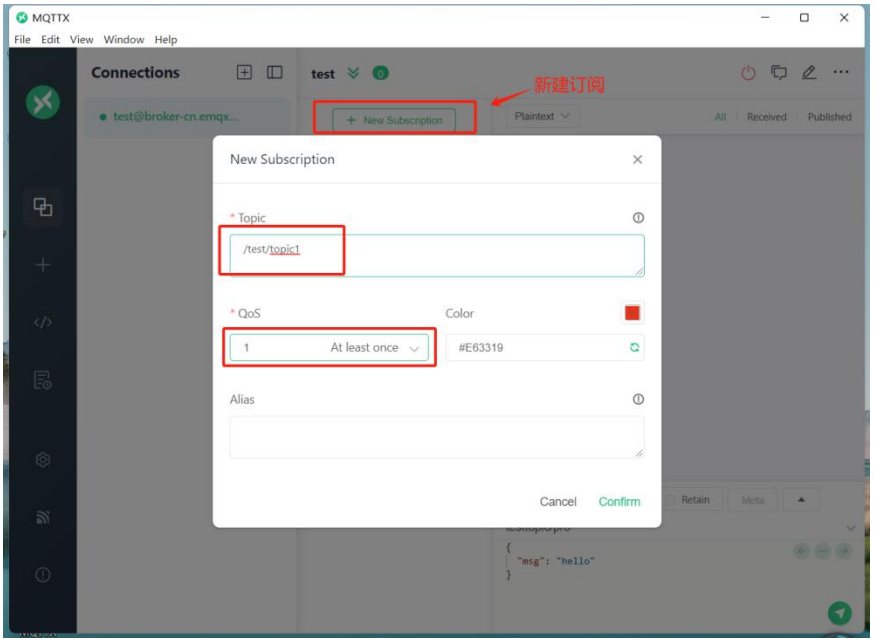
在开始 ESP32 编程之前，我们要先来看看公共主流的 MQTT 服务器可供使用，所谓的公共 MQTT 服务器就是一些网站给我们提供了在线的 MQTT Broker，我可以直接利用其进行 MQTT 学习、测试甚至是小规模使用，而无需再自行部署，方便快捷，节省时间与精力成本，这对于我们用 ESP32 来做一些 MQTT 调试再适合不过了，由于这些 MQTT 服务器很多都是外国的，它们的可访问性、延时等都有着一些问题。这里我推荐一个目前为止我一直在用的。

mqtt://broker-cn.emqx.io

为了方便测试，我们可以在电脑上下载一个 MQTTX 客户端软件，后续的话可以看到与 ESP32 是如何进行数据交互的，下载地址在 https://mqttx.app/zh。下载安装过程就不介绍了，打开软件之后设置如下，其余按照默认值即可



修改完后点击右上角 Connect，连接成功后，新建一个订阅，订阅主题是/test/topic1，消息等级 QoS=1，点击 Confirm。这样我们在电脑上的客户端就部署完毕了。



二、ESP32 上的 MQTT 程序

完整代码在 esp32-board/mqtt 中。

mqtt 连接之前，需要 WiFi 连接成功，因此需要 WIFI——STA 章节的代码，我在 WIFI——STA 章节代码上做了一些改善，增加了一个回调函数通知主函数 WiFi 已连接。

```
/** 事件回调函数
 * @param arg   用户传递的参数
 * @param event_base    事件类别
 * @param event_id      事件 ID
 * @param event_data    事件携带的数据
 * @return 无
*/
static void event_handler(void* arg, esp_event_base_t event_base,int32_t
event_id, void* event_data)
{
    if(event_base == WIFI_EVENT)
    {
        switch (event_id)
        {
        case WIFI_EVENT_STA_START:      //WIFI 以 STA 模式启动后触发此事件
            esp_wifi_connect();         //启动 WIFI 连接
            break;
        case WIFI_EVENT_STA_CONNECTED:  //WIFI 连上路由器后，触发此事件
            ESP_LOGI(TAG, "connected to AP");
            break;
        case WIFI_EVENT_STA_DISCONNECTED:   //WIFI 从路由器断开连接后触发此
事件
            esp_wifi_connect();             //继续重连
            ESP_LOGI(TAG,"connect to the AP fail,retry now");
            break;
        default:
            break;
        }
    }
    if(event_base == IP_EVENT)                  //IP 相关事件
    {
        switch(event_id)
        {
            case IP_EVENT_STA_GOT_IP:           //只有获取到路由器分配的 IP,
才认为是连上了路由器
                if(wifi_cb)
                    wifi_cb(WIFI_CONNECTED);
                ESP_LOGI(TAG,"get ip address");
                break;
        }
```

```
    }
}
```

请看 IP_EVENT_STA_GOT_IP 事件处理，会执行 wifi_cb 函数通知 WIFI_CONNECTED 事件。

接下来请看 main.c 文件实现

```
#define MQTT_ADDRESS    "mqtt://broker-cn.emqx.io"      //MQTT 连接地址
#define MQTT_PORT       1883                           //MQTT 连接端口号
#define MQTT_CLIENT     "mqttx_d11213"                 //Client ID（设备唯一，
大家最好自行改一下）
#define MQTT_USERNAME   "hello"                        //MQTT 用户名
#define MQTT_PASSWORD   "12345678"                     //MQTT 密码

#define MQTT_PUBLIC_TOPIC      "/test/topic1"          //测试用的,推送消息主
题
#define MQTT_SUBSCRIBE_TOPIC    "/test/topic2"         //测试用的,需要订阅的
主题
//定义一个事件组，用于通知 main 函数 WIFI 连接成功
#define WIFI_CONNECT_BIT     BIT0
static EventGroupHandle_t   s_wifi_ev = NULL;
//MQTT 客户端操作句柄
static esp_mqtt_client_handle_t     s_mqtt_client = NULL;
//MQTT 连接标志
static bool   s_is_mqtt_connected = false;
/**
 * mqtt 连接事件处理函数
 * @param event 事件参数
 * @return 无
 */
static void aliot_mqtt_event_handler(void* event_handler_arg,
                                     esp_event_base_t event_base,
                                     int32_t event_id,
                                     void* event_data)
{
    esp_mqtt_event_handle_t event = event_data;
    esp_mqtt_client_handle_t client = event->client;
    // your_context_t *context = event->context;
    switch ((esp_mqtt_event_id_t)event_id) {
        case MQTT_EVENT_CONNECTED:  //连接成功
            ESP_LOGI(TAG, "mqtt connected");
            s_is_mqtt_connected = true;
            //连接成功后，订阅测试主题
            esp_mqtt_client_subscribe_single(s_mqtt_client,MQTT_SUBSCRIB
E_TOPIC,1);
            break;
        case MQTT_EVENT_DISCONNECTED:   //连接断开
```

```
            ESP_LOGI(TAG, "mqtt disconnected");
            s_is_mqtt_connected = false;
            break;
        case MQTT_EVENT_SUBSCRIBED:     //收到订阅消息 ACK
            ESP_LOGI(TAG, " mqtt subscribed ack, msg_id=%d",
event->msg_id);
            break;
        case MQTT_EVENT_UNSUBSCRIBED:   //收到解订阅消息 ACK
            break;
        case MQTT_EVENT_PUBLISHED:      //收到发布消息 ACK
            ESP_LOGI(TAG, "mqtt publish ack, msg_id=%d", event->msg_id);
            break;
        case MQTT_EVENT_DATA:
            printf("TOPIC=%.*s\r\n", event->topic_len,
event->topic);       //收到 Pub 消息直接打印出来
            printf("DATA=%.*s\r\n", event->data_len, event->data);
            break;
        case MQTT_EVENT_ERROR:
            ESP_LOGI(TAG, "MQTT_EVENT_ERROR");
            break;
        default:
            break;
    }
}
/** 启动 mqtt 连接
 * @param 无
 * @return 无
*/
void mqtt_start(void)
{
    esp_mqtt_client_config_t mqtt_cfg = {0};
    mqtt_cfg.broker.address.uri = MQTT_ADDRESS;
    mqtt_cfg.broker.address.port = MQTT_PORT;
    //Client ID
    mqtt_cfg.credentials.client_id = MQTT_CLIENT;
    //用户名
    mqtt_cfg.credentials.username = MQTT_USERNAME;
    //密码
    mqtt_cfg.credentials.authentication.password = MQTT_PASSWORD;
    ESP_LOGI(TAG,"mqtt
connect->clientId:%s,username:%s,password:%s",mqtt_cfg.credentials.clie
nt_id,
    mqtt_cfg.credentials.username,mqtt_cfg.credentials.authentication.pa
ssword);
```

```
    //设置 mqtt 配置，返回 mqtt 操作句柄
    s_mqtt_client = esp_mqtt_client_init(&mqtt_cfg);
    //注册 mqtt 事件回调函数
    esp_mqtt_client_register_event(s_mqtt_client, ESP_EVENT_ANY_ID,
aliot_mqtt_event_handler, s_mqtt_client);
    //启动 mqtt 连接
    esp_mqtt_client_start(s_mqtt_client);
}
/** wifi 事件通知
 * @param 无
 * @return 无
*/
void wifi_event_handler(WIFI_EV_e ev)
{
    if(ev == WIFI_CONNECTED)
    {
        xEventGroupSetBits(s_wifi_ev,WIFI_CONNECT_BIT);
    }
}
void app_main(void)
{
    esp_err_t ret = nvs_flash_init();
    if (ret == ESP_ERR_NVS_NO_FREE_PAGES || ret ==
ESP_ERR_NVS_NEW_VERSION_FOUND) {
        //NVS 出现错误，执行擦除
        ESP_ERROR_CHECK(nvs_flash_erase());
        //重新尝试初始化
        ESP_ERROR_CHECK(nvs_flash_init());
    }
    s_wifi_ev = xEventGroupCreate();
    EventBits_t ev = 0;
    //初始化 WIFI，传入回调函数，用于通知连接成功事件
    wifi_sta_init(wifi_event_handler);
    //一直监听 WIFI 连接事件，直到 WiFi 连接成功后，才启动 MQTT 连接
    ev =
xEventGroupWaitBits(s_wifi_ev,WIFI_CONNECT_BIT,pdTRUE,pdFALSE,portMAX_D
ELAY);
    if(ev & WIFI_CONNECT_BIT)
    {
        mqtt_start();
    }
    static char mqtt_pub_buff[64];
    while(1)
    {
```

```
        int count = 0;
        //延时 2 秒发布一条消息到/test/topic1 主题
        if(s_is_mqtt_connected)
        {
            snprintf(mqtt_pub_buff,64,"{\"count\":\"%d\"}",count);
            esp_mqtt_client_publish(s_mqtt_client, MQTT_PUBLIC_TOPIC,
                                    mqtt_pub_buff, strlen(mqtt_pub_buff),1, 0);
            count++;
        }
        vTaskDelay(pdMS_TO_TICKS(2000));
    }
    return;
}
```