MQTT-Publisher Example

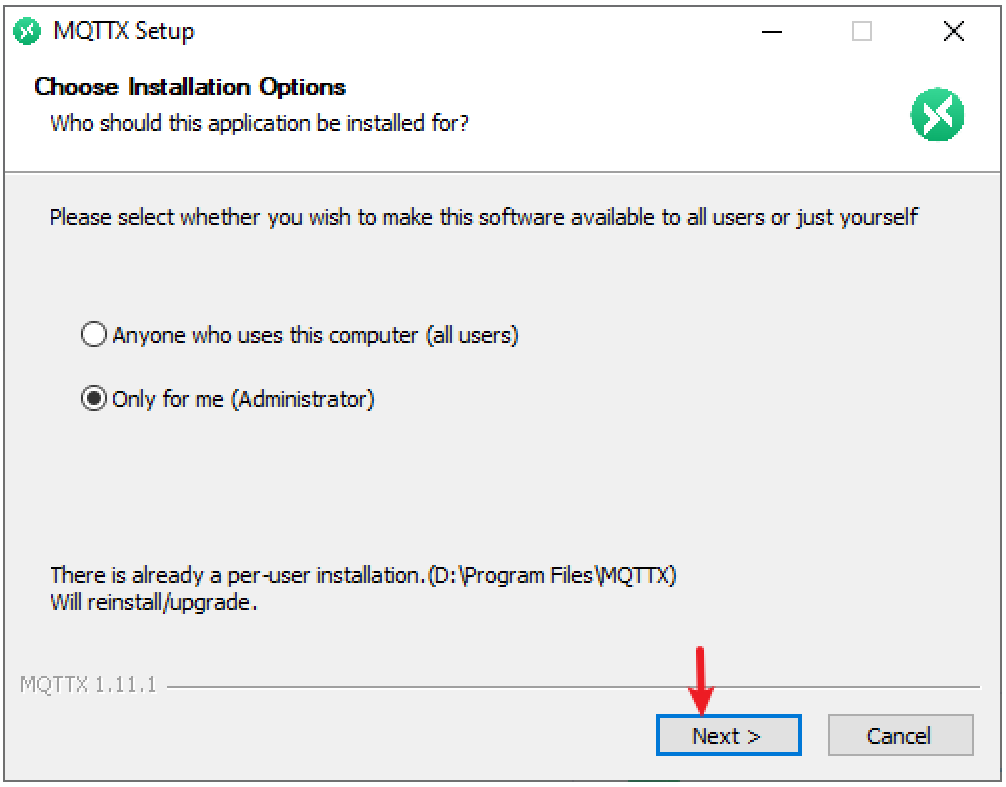
Routine explanation

Introduction

In this section, we use K230 to implement the MQTT publisher function.

Step 1: Install the debugging software MQTTX

Before running the example, we need to install MQTTX as debugging software. You can find the installation package of this software in our supporting materials (Windows version. Other system versions can be found in this link MQTTX download)


MQTTX Setup

Choose Installation Options
Who should this application be installed for?

Please select whether you wish to make this software available to all users or just yourself

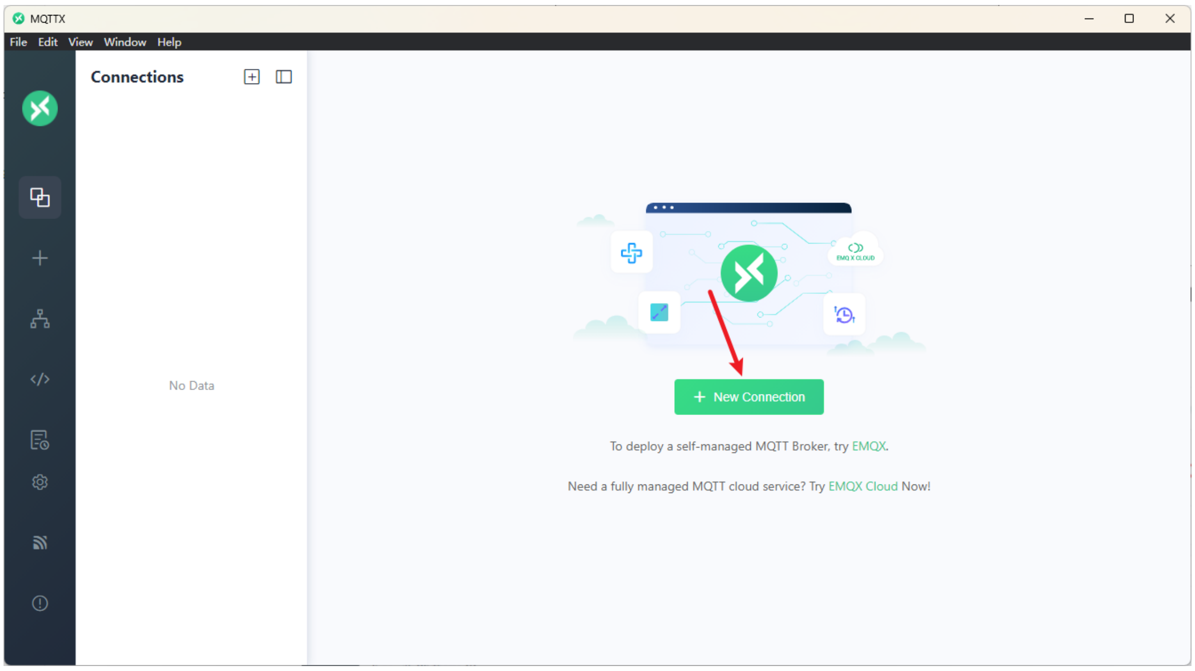
○ Anyone who uses this computer (all users)
● Only for me (Administrator)

There is already a per-user installation.(D:\Program Files\MQTTX)
Will reinstall/upgrade.

MQTTX 1.11.1

The installation steps are relatively simple, just keep clicking “Next”.

After the installation is complete, we open the program and click "New Connection"



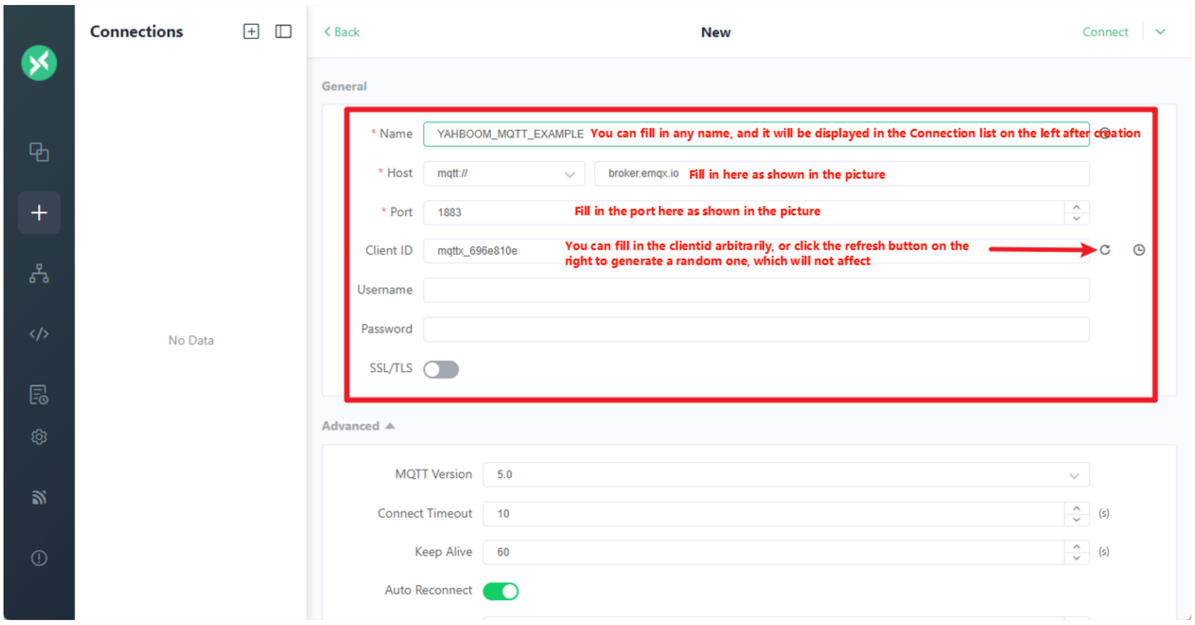
No Data

New Connection

To deploy a self-managed MQTT Broker, try EMQX.

Need a fully managed MQTT cloud service? Try EMQX Cloud Now!

Then fill in the information as shown below



* Name	YAHBOOM_MQTT_EXAMPLE
* Host	mqtt:// broker.emqx.io
* Port	1883
Client ID	mqttx_696e810e
Username	
Password	
SSL/TLS	off
MQTT Version	5.0
Connect Timeout	10 (s)
Keep Alive	60 (s)
Auto Reconnect	on

You can fill in any name, and it will be displayed in the Connection list on the left after creation

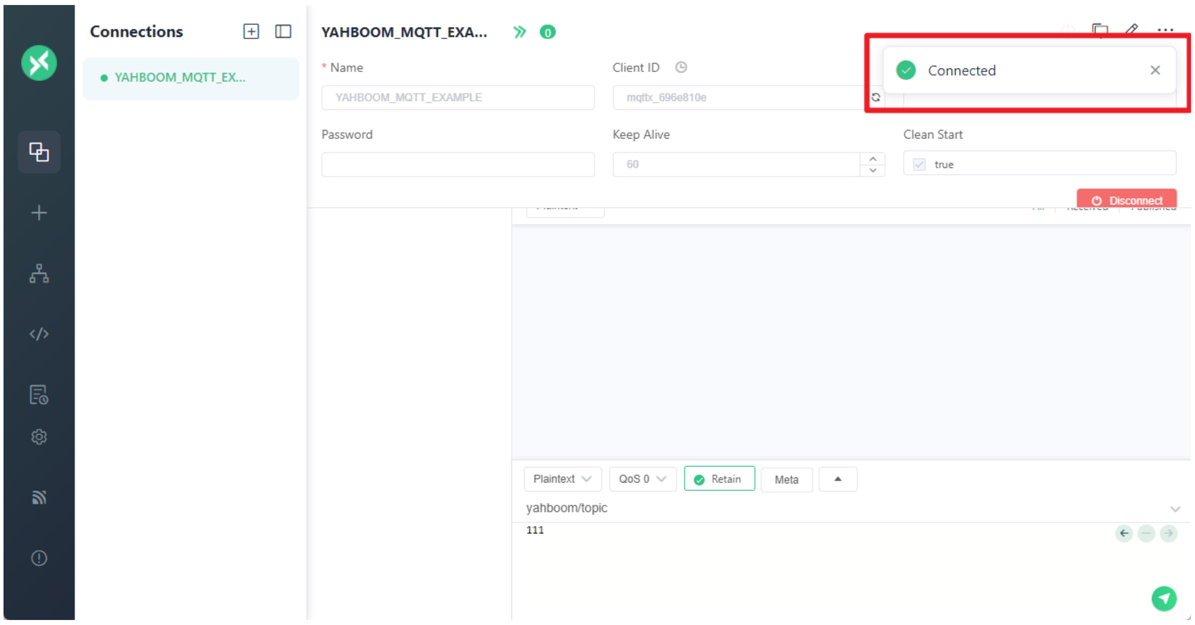
Fill in here as shown in the picture

Fill in the port here as shown in the picture

You can fill in the clientid arbitrarily, or click the refresh button on the right to generate a random one, which will not affect

> The Host and Port here refer to the address information of the MQTT server. In our example, we use the free public MQTT server provided by EMQX. The server information can be found in this link:
>
> [Free MQTT Broker: Public & Multi-Region | Connect Now | EMQ]()
>
> It should be noted that third-party services cannot guarantee long-term effectiveness. If you find that the link is invalid, you can promptly feedback to our technical support and we will find a public MQTT server to replace it and modify the tutorial.
>
> Of course, in actual scenarios (real business scenarios), please deploy the MQTT server yourself to ensure data security and service stability.

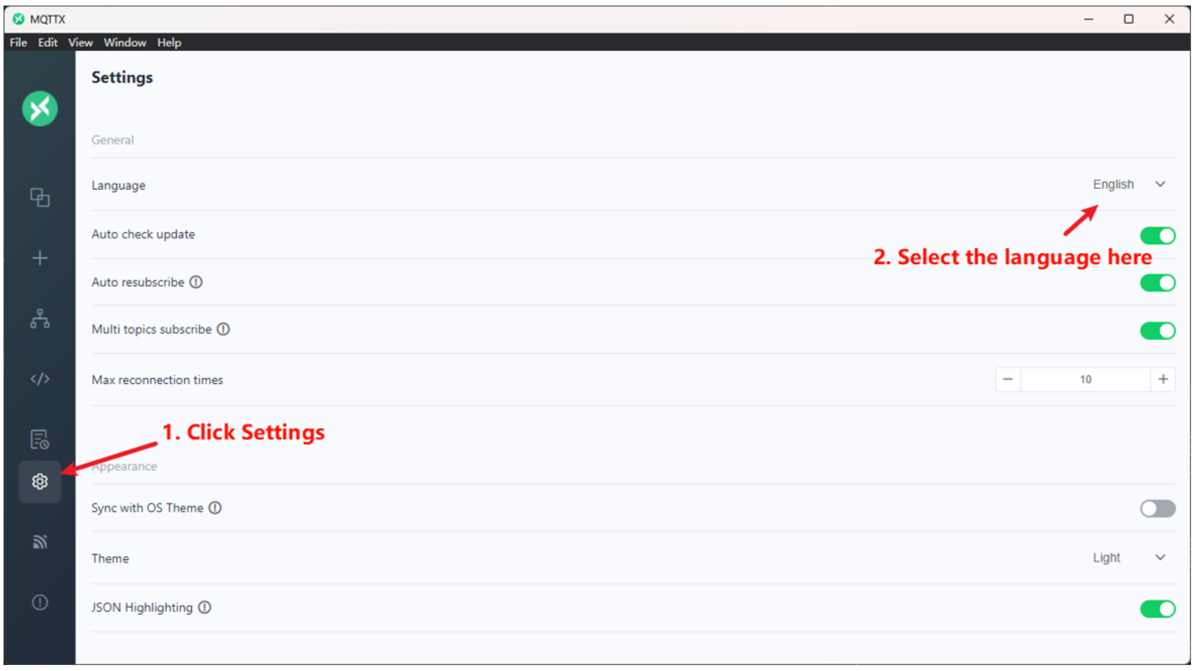
After filling in the form, click Connection in the upper right corner and wait for the Connected button to pop up, indicating that the connection is successful.



Name: YAHBOOM_MQTT_EXAMPLE
Client ID: mqttx_696e810e
Password:
Keep Alive: 60
Clean Start: true

yahboom/topic
111

If you need to adjust the language, you can do it here:



General

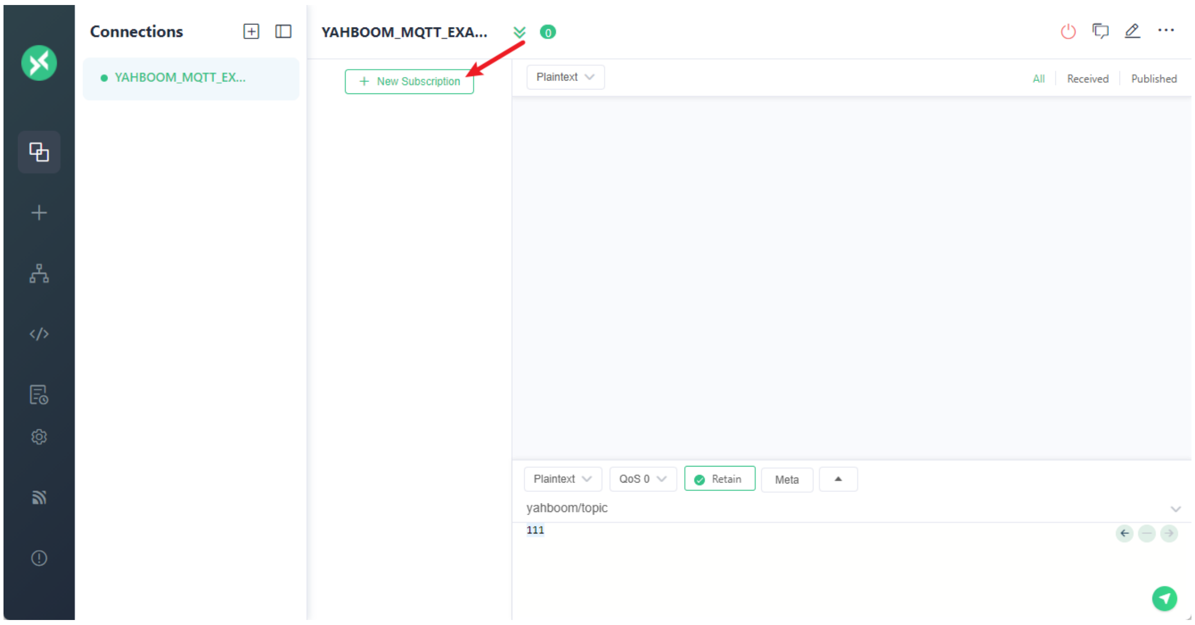
Language	English
Auto check update	on
Auto resubscribe	on
Multi topics subscribe	on
Max reconnection times	10

Appearance

Sync with OS Theme	off
Theme	Light
JSON Highlighting	on

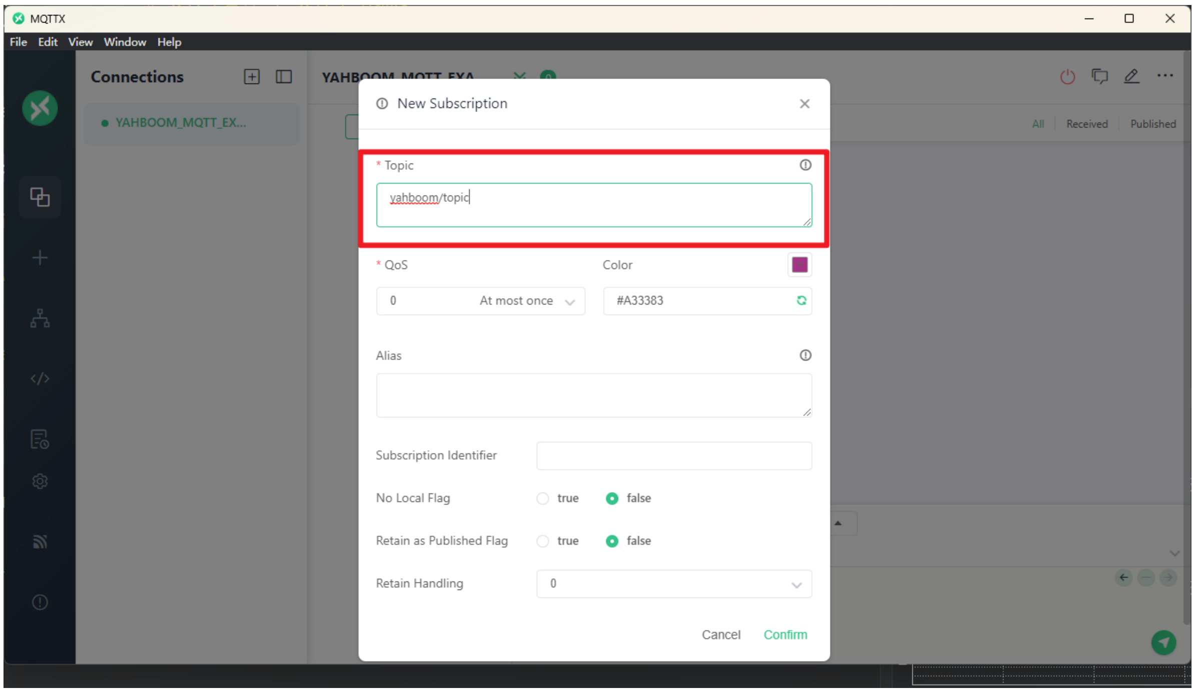
Step 2: MQTT-Publisher Startup

Let’s go to MQTTX and click New Subscription



yahboom/topic
111

Fill in Topic as yahboom/topic, then click the Confirm button in the lower right corner.



Topic: yahboom/topic
QoS: 0 At most once
Color: #A33383
Alias:
Subscription Identifier:
No Local Flag: true [✓ false]
Retain as Published Flag: true [✓ false]
Retain Handling: 0

Next, open the sample program mqtt-publisher in this section and make sure K230 is in a WIFI environment that can access the Internet.

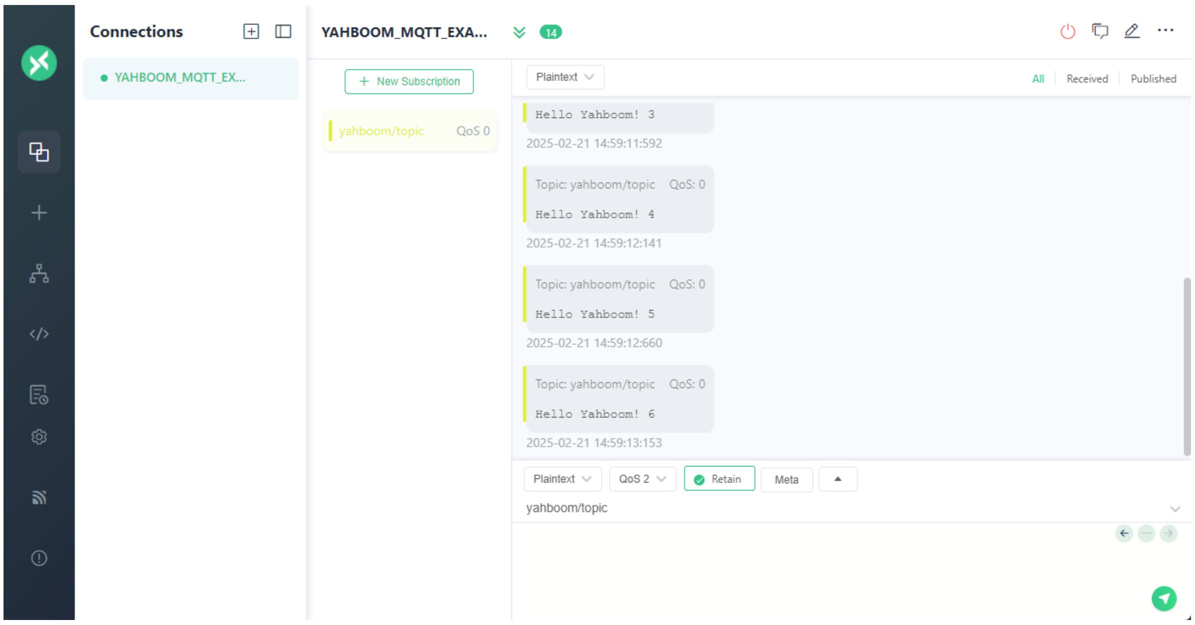
Modify WIFI connection SSID and password

```
# WiFi配置参数 / WiFi configuration parameters
WIFI_SSID = "WIFI SSID"          # WiFi名称 / WiFi name
WIFI_PASSWORD = "WIFI PASSWORD" # WiFi密码 / WiFi password
```

Click Run Program and wait for console output

```
连接WIFI: Yahboom.. Connecting..
WIFI连接成功 WIFI Connected!
```

Let's go back to MQTTX and see that we have received the content published by K230.



```
Hello Yahboom! 3
2025-02-21 14:59:11:592

Topic: yahboom/topic   QoS: 0
Hello Yahboom! 4
2025-02-21 14:59:12:141

Topic: yahboom/topic   QoS: 0
Hello Yahboom! 5
2025-02-21 14:59:12:660

Topic: yahboom/topic   QoS: 0
Hello Yahboom! 6
2025-02-21 14:59:13:153
```

yahboom/topic

Code Analysis

For the complete code, please refer to the file [Source Code Summary/11.Network/06.mqtt/mqtt_publisher.py]

1. Module import part

The code first imports several key modules:

- `from mqtt import MQTTClient`: The core module used to implement MQTT communication.
- `import network`: Used to handle WiFi connections.
- `import os`: Provides operating system related functions, which may be used for exit point checking (`os.exitpoint()`).
- `import machine`: Used for hardware control (although not used directly in this code).
- `import time`: Used to implement delayed operation.

These modules lay the foundation for subsequent WiFi connection and MQTT communication.

2. Configuration parameters

The code defines two sets of global configuration parameters:

- **WiFi Configuration** :
  - `WIFI_SSID = "WIFI SSID"`: WiFi name.
  - `WIFI_PASSWORD = "WIFI PASSWORD"`: WiFi password.
  - These are placeholders and will need to be replaced with your actual WiFi credentials.
- **MQTT Configuration** :
  - `MQTT_BROKER = "broker.emqx.io"`: Use the public EMQX MQTT server.
  - `MQTT_PORT = 1883`: The standard MQTT port.
  - `MQTT_TOPIC = "yahboom/topic"`: The topic of the message.

These parameters are the core settings for program operation.

3. `connect_wifi` Functions

Here is a function for connecting to WiFi:

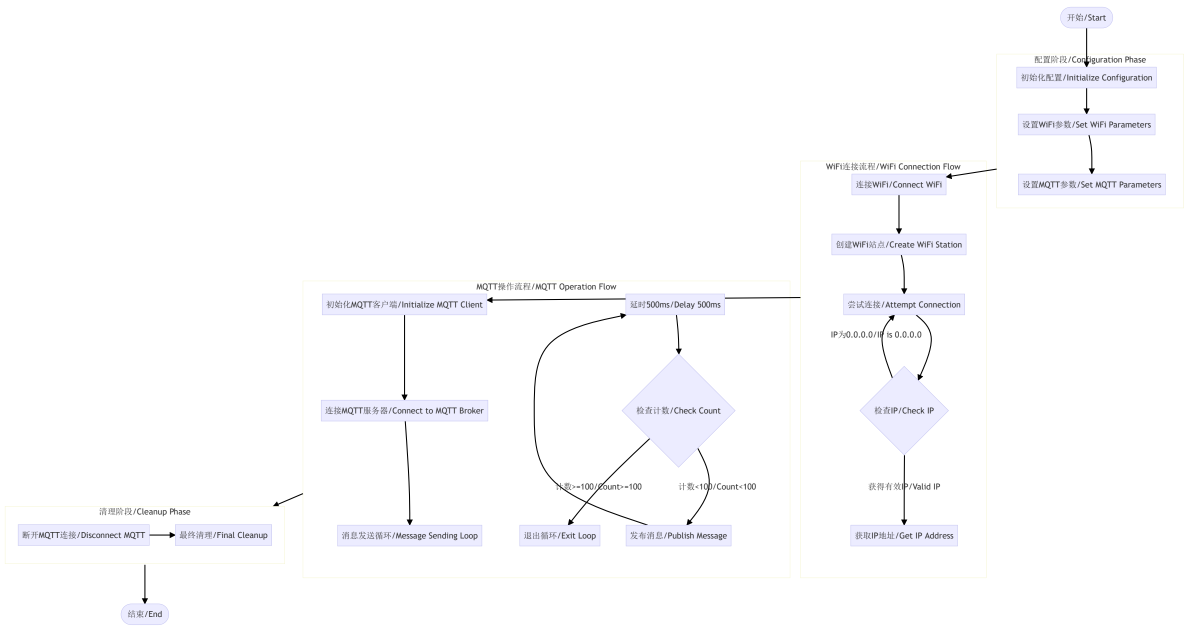
- **parameter** :
  - `ssid`: WiFi name, the default value is `"Yahboom"`.
  - `password`: WiFi password, the default value is `"yahboom890729"`.
- **Function** :
  - Create a WiFi site object: `wifi_station = network.WLAN(0)`.
  - Call `connect` the method to connect to the specified WiFi.
  - By checking `ifconfig()[0]` (IP address) in a loop, until a non-positive address is obtained `0.0.0.0`, confirm that the connection is successful.
  - Used `os.exitpoint()`, probably to handle exit points (used in MicroPython to check system events).
- **return** :
  - Returns the IP address of the device: `wifi_station.ifconfig()[0]`.
- **Output** :
  - Prints connection status information such as `"连接WIFI: {ssid}.. Connecting.."` and `"WIFI连接成功 WIFI Connected!"`.

4. Main program part

The main program `if __name__ == "__main__":` runs in the block:

- **WiFi connection** :
  - Call `connect_wifi(WIFI_SSID, WIFI_PASSWORD)` to connect to WiFi.
- **MQTT client initialization** :
  - Create an MQTT client instance: `client = MQTTClient("YAHBOOM-K230", MQTT_BROKER, port=MQTT_PORT)`.
  - The client ID is `"YAHBOOM-K230"`, connect to the specified server and port.
- **Connect to the MQTT server** :
  - Call `client.connect()` to establish a connection.
- **Message publishing cycle** :
  - Use `for` a loop to send 100 messages.
  - Each time a message is published `MQTT_TOPIC`, the content is `"Hello Yahboom! " + str(i)` (for example, `"Hello Yahboom! 0"` to `"Hello Yahboom! 99"`).
  - Delay 500 milliseconds after each release: `time.sleep_ms(500)`.
- **Disconnect** :
  - After the loop is finished, the call `client.disconnect()` is made to disconnect from the MQTT server.

flow chart


开始/Start
配置阶段/Configuration Phase
初始化配置/Initialize Configuration
设置WiFi参数/Set WiFi Parameters
设置MQTT参数/Set MQTT Parameters
WiFi连接流程/WiFi Connection Flow
连接WiFi/Connect WiFi
创建WiFi站点/Create WiFi Station
尝试连接/Attempt Connection
IP为0.0.0.0/IP is 0.0.0.0
检查IP/Check IP
获得有效IP/Valid IP
获取IP地址/Get IP Address
MQTT操作流程/MQTT Operation Flow
初始化MQTT客户端/Initialize MQTT Client
连接MQTT服务器/Connect to MQTT Broker
消息发送循环/Message Sending Loop
延时500ms/Delay 500ms
检查计数/Check Count
计数>=100/Count>=100
计数<100/Count<100
退出循环/Exit Loop
发布消息/Publish Message
清理阶段/Cleanup Phase
断开MQTT连接/Disconnect MQTT
最终清理/Final Cleanup
结束/End