Soil Moisture sensor system

อธิบายการทำงานของโค้ด :

```
#include <Arduino.h>
#include <WiFi.h>
#include <PubSubClient.h>

const char *ssid = "LAPTOP-621";
const char *password = "Tni54321";
const char *mqtt_server = "192.168.137.167";
const int mqtt_port = 1883; // 1883 for TCP
const int soilpin = 34;
// Topic for Publish &Subscribe
const char *mqtt_publish_topic = "esp32/output";
const char *mqtt_subscribe_topic = "esp32/input";
WiFiClient espClient;
PubSubClient client(espClient);
long lastMsg = 0;
char msg[50];
int value = 0;

```

• library ที่ใช้ในการทำงานได้แก่ :
Arduino → ใช้สำหรับเชื่อมต่อArduino
WiFi → ใช้เชื่อมต่อเครือข่ายไวไฟกับ ESP32
PubSubClient → สื่อสารผ่าน MQTT Protocol

กำหนดค่าต่างๆ :
const char* ssid = "LAPTOP-621"; → ชื่อwifiที่เชื่อมต่อ
const char* password = "Tni54321"; → password-wifi ที่เชื่อมต่อ
const char* mqtt_server = "192.168.137.167"; → ที่อยู่ของ IP ของServer
const int mqtt_port = 1883; → portมาตรฐานของ MQTT
const int soilpin = 34; → ขาของ ESP32 ที่ใช้ต่อกับเซนเซอร์วัดความชื้นดิน

กำหนดค่าสำหรับ mqtt :
const char* mqtt_publish_topic = "esp32/output"; → ส่งค่าความชื้นดิน
const char* mqtt_subscribe_topic = "esp32/input"; → รอรับข้อความ

ตัวแปรต่างๆ :
WiFiClient espClient; → สร้าง Client สำหรับเชื่อมต่ออินเทอร์เน็ต
PubSubClient client(espClient); → ใช้ client เชื่อมMQTT Broker
long lastMsg = 0; → เช็คเวลาในการส่งข้อมูล
char msg[50]; → buffer สำหรับสร้างข้อความก่อนส่ง
int value = 0; → ตัวนับค่า

```
19  void setup_wifi()
20  {
21    delay(10);
22    Serial.println();
23    Serial.print("Connecting to ");
24    Serial.println(ssid);
25    WiFi.begin(ssid, password);
26    while (WiFi.status() != WL_CONNECTED)
27    {
28      delay(500);
29      Serial.print(".");
30    }
31    Serial.println("");
32    Serial.println("WiFi connected");
33    Serial.println("IP address: ");
34    Serial.println(WiFi.localIP());
35  }
36
```

ฟังก์ชัน setup_wifi() :

1. เรียก WiFi.begin(ssid, password) เพื่อให้ ESP32 เชื่อมต่อกับ Wi-Fi
2. ใช้ while(WiFi.status() != WL_CONNECTED) ทำการวนซ้ำรอจนกว่าจะเชื่อมต่อสำเร็จ ระหว่างนี้จะแสดง "." ออกมาทุก 0.5 วินาที
3. เมื่อเชื่อมต่อแล้วจะแสดงข้อความ “WiFi connected : ” และ “IP Address : ” ที่ได้รับ

```
37  // Callback function when MQTT message received
38  void callback(char *topic, byte *payload,
39                unsigned int length)
40  {
41    Serial.print("Message arrived [");
42    Serial.print(topic);
43    Serial.print("] ");
44    // display payload on serial monitor
45    for (int i = 0; i < length; i++)
46    {
47      Serial.print((char)payload[i]);
48    }
49    Serial.println();
50  }
```

ฟังก์ชัน callback(...):

1. เมื่อมีข้อความเข้ามาทาง MQTT ฟังก์ชันนี้จะทำงาน
2. แสดงชื่อ topic ของข้อความนั้นออกมาบน Serial Monitor
3. วนลูปอ่านข้อความจาก payload แล้วแปลงเป็นตัวอักษรพิมพ์ออกมา

```
52  void reconnect()
53  {
54    while (!client.connected())
55    {
56      Serial.print("Attempting MQTT connection...");
57      String clientId = "ESP32Client-";
58      clientId += String(random(0xffff), HEX);
59      if (client.connect(clientId.c_str()))
60      {
61        Serial.println("connected");
62        client.subscribe(mqtt_subscribe_topic);
63        Serial.print("Subscribed to: ");
64        Serial.println(mqtt_subscribe_topic);
65      }
66      else
67      {
68        Serial.print("failed, rc=");
69        Serial.print(client.state());
70        Serial.println(" try again in 5 seconds");
71        delay(5000);
72      }
73    }
74  }
```

ฟังก์ชัน reconnect() :

1. ฟังก์ชันนี้จะถูกเรียกใช้เมื่อ ESP32 ยังไม่ได้เชื่อมต่อกับ MQTT Broker
2. เริ่มจากการแสดงข้อความ "Attempting MQTT connection..."
3. สร้าง Client ID แบบสุ่ม เพื่อไม่ให้ซ้ำกับอุปกรณ์อื่นที่อาจต่ออยู่ใน broker เดียวกัน
4. เรียกใช้เงื่อนไข if(client.connect(clientId.c_str()) เพื่อเชื่อมต่อกับ broker
   - ถ้าเชื่อมต่อสำเร็จ →
     - แสดงข้อความ "connected"
     - subscribe topic ที่ตั้งไว้ เช่น "esp32/input"
   - ถ้าเชื่อมไม่ต่อสำเร็จ →
     - แสดงข้อความ “failed, rc=”
     - แสดงข้อความ "try again in 5 seconds" แล้วหน่วงเวลา 5 วินาที ก่อนลองใหม่
5. ลูปจะทำงานซ้ำไปเรื่อย ๆ จนกว่าจะเชื่อมต่อสำเร็จ

```
76  void setup()
77  {
78    Serial.begin(9600);
79    setup_wifi();
80    pinMode(soilpin, INPUT);
81    client.setServer(mqtt_server, mqtt_port);
82    client.setCallback(callback);
83  }
84
```

ฟังก์ชัน setup():

1. set ความเร็ว Serial monitor ไว้ที่ 9600
2. เชื่อมต่อ Wi-Fi
3. ตั้งค่า pin ของ Soil Moisture Sensor เป็นขา Input
4. เชื่อมต่อไปยัง MQTT broker ด้วย IP/Host และ Port
5. กำหนด callback function ไว้คอยจัดการข้อความจาก MQTT

```
85  void loop()
86  {
87    if (!client.connected())
88    {
89      reconnect();
90    }
91    client.loop();
92
93    long now = millis();
94    if (now - lastMsg > 10000)
95    { // ทุกๆ 10 วินาที
96      lastMsg = now;
97      ++value;
98      client.publish(mqtt_publish_topic, msg);
99      int humid = analogRead(soilpin); // ความชื้นสัมพัทธ์
100
101     Serial.print("Soil Moisture: ");
102     Serial.print(humid);
103
104     // สร้าง JSON payload อย่างถูกต้อง
105     sprintf(msg, "{\"humid\": %d}", humid);
106
107     Serial.print("Publishing JSON to topic '");
108     Serial.print(mqtt_publish_topic);
109     Serial.print("': ");
110     Serial.println(msg);
111
112     client.publish(mqtt_publish_topic, msg);
113   }
114 }
```

ฟังก์ชัน loop() :

1. ตรวจสอบการเชื่อมต่อ MQTT
   - ถ้าไม่ได้เชื่อม → เรียก reconnect()
2. รัน MQTT client loop
3. เช็คเวลาปัจจุบัน ถ้าเกิน 10 วินาทีจาก lastMsg
   - อัปเดต lastMsg
   - อ่านค่าความชื้นจาก sensor
   - แสดงผลค่า "Soil moisture: " และ humid ในSerial Monitor
   - สร้างข้อความ JSON "{\"humid\": %d}"
   - ส่ง JSON ไปยัง MQTT broker