NodeMCU MQTT

NodeMCU MQTT

❖ MQTT 관련 라이브러리 PubSubClient

- 라이브러리 매니저
  - PubSubClient 검색 및 설치

NodeMCU MQTT

❖ ex04/app.ino

```
#include <WifiMiniCom.h>
#include <PubSubClient.h>
#include <Led.h>

const char *ssid = "Campus7_Room4_2.4GHz";
const char *password = "12345678";
const char *mqtt_server = "192.168.0.159"; // mqtt broker ip address

WifiMiniCom com;

WiFiClient espClient;
PubSubClient client(espClient);
Led led(BUILTIN_LED);

int value = 0;
```

NodeMCU MQTT

❖ ex04/app.ino

```
void callback(char *topic, byte *payload, unsigned int length) {
    char buf[128];
    memcpy(buf, payload, length);
    buf[length] = '\0';

    Serial.print("Message arrived [");
    Serial.print(topic);
    Serial.print("] ");
    Serial.println(buf);

    com.print(0, topic);
    com.print(1, buf);

    if (buf[0] == '1') {
        led.setValue(LOW);
    } else {
        led.setValue(HIGH);
    }
}
```

NodeMCU MQTT

❖ ex04/app.ino

```
void reconnect() {
    while (!client.connected())     {
        Serial.print("Attempting MQTT connection...");
        if (client.connect("ESP8266Client")) { // 클라이언트 ID 중복 주의
            Serial.println("connected");

            client.publish("outTopic", "hello world");
            client.subscribe("inTopic");    // subscribe할 토픽 등록
        }
        else { // 연결실패한 경우 5초 후 재시도
            Serial.print("failed, rc=");
            Serial.print(client.state());
            Serial.println(" try again in 5 seconds");
            delay(5000);
        }
    }
}
```

NodeMCU MQTT

❖ ex04/app.ino

```
void publish() {
    char msg[50];
    ++value;
    sprintf(msg, "hello world #%ld", value);
    Serial.print("Publish message: ");
    Serial.println(msg);
    client.publish("outTopic", msg);
}

void setup() {
    com.init(ssid, password);
    com.setInterval(2000, publish);
    client.setServer(mqtt_server, 1883);
    client.setCallback(callback); // 토픽 수신 시 호출할 함수 등록
}

void loop() {
    com.run();
    if (!client.connected()) {
        reconnect();
    }
    client.loop();
}
```

NodeMCU mqtt

❖ publish 확인

- mosquitto_sub -v -h localhost -t outTopic

❖ subscribe 확인

- mosquitto_pub -h localhost -t inTopic -m 1
  - LED ON

- mosquitto_pub -h localhost -t inTopic -m 0
  - LED OFF

NodeMCU MQTT

❖ MqttCom 클래스

- Mqtt 프로토콜 지원 클래스
- WiFiMiniCom을 상속

- 주요 메서드
  - MqttCom(int serial_bps=115200, int lcd_addr=0x27)
    - 생성자
  - void init(const char *ssid, const char *password, int no_lcd=false);
    - 네트워크 초기화 및 lcd 사용 여부 지정
  - void setServer(const char *server, const char *topic = NULL,
    MQTT_CALLBACK_SIGNATURE = NULL);
    - Mqtt 관련 정보 설정
  - void reconnect();
    - Mqtt 서버 재연결
  - void run();
    - 기본 운영 및 Mqtt 이벤트 처리
  - void publish(…);
    - 지정한 토픽으로 메시지 publish

NodeMCU MQTT

❖ MqttCom.h

```
#pragma once

#include <WifiMiniCom.h>
#include <PubSubClient.h>

class MqttCom: public WifiMiniCom  {
protected:
    const char *server;       // MQTT 브로커 IP 주소
    String client_id;         // 클라이언트(NodeMCU의 ID)
    WiFiClient espClient;
    PubSubClient client;

    const char *topic;        // subscribe 토픽명
    // void (*callback)(char*, uint8_t*, unsigned int);
    MQTT_CALLBACK_SIGNATURE;  // subscribe 콜백 함수 포인터, 변수명은 callback
```

NodeMCU MQTT

❖ MqttCom.h

```
public :
    MqttCom(int serial_bps=115200, int lcd_addr=0x27);
    void init(const char *ssid, const char *password, int no_lcd=false);
    void setServer(const char *server, const char *topic = NULL,
                   MQTT_CALLBACK_SIGNATURE = NULL);
    void reconnect();
    void run();

    void publish(const char *topic, const char *value);
    void publish(const char *topic, int value);
    void publish(const char *topic, float value);
};
```

NodeMCU MQTT

❖ MqttCom.cpp

```
#include "MqttCom.h"

MqttCom::MqttCom(int serial_bps, int lcd_addr) :
    WifiMiniCom(serial_bps, lcd_addr), client(espClient) {
    topic = NULL;
    callback = NULL;
    server = NULL;

    // 랜덤하게 클라이언트 ID 배정
    randomSeed(analogRead(0));
    int r = random(300);
    client_id = String("ESP8266Client") + r;
}

void MqttCom::init(const char *ssid, const char *password, int no_lcd) {
    WifiMiniCom::init(ssid, password);
    if(no_lcd) {
        WifiMiniCom::setNoLcd();
    }
}
```

NodeMCU MQTT

❖ MqttCom.cpp

```
void MqttCom::setServer(const char *server, const char *topic,
                        MQTT_CALLBACK_SIGNATURE) {
    this->server = server;
    this->callback = callback;
    this->topic = topic;

    client.setServer(server, 1883);
    if(callback != NULL) {
        client.setCallback(callback);
    }
}
```

NodeMCU MQTT

❖ MqttCom.cpp

```
void MqttCom::reconnect() {
    while (!client.connected()) {
        Serial.print("Attempting MQTT connection...");
        print(0, "try MQTT con...");
        if (client.connect(client_id.c_str())) {    // 클라이언트 ID 중복 주의
            Serial.println("connected");
            print(0, "MQTT connected");
            if(topic != NULL) {
                client.subscribe(topic);
            }
        } else {
            char buf[17];
            sprintf(buf, "failed, rc=%d", client.state());
            Serial.print(buf);
            print(0, buf);
            Serial.println(" try again in 5 seconds");
            print(1, "try again in 5 sec");
            delay(5000);
        }
    }
}
```

NodeMCU MQTT

❖ MqttCom.cpp

```
void MqttCom::run() {
    MiniCom::run();
    if (!client.connected()) {
        reconnect();
    }
    client.loop();
}

void MqttCom::publish(const char *topic, const char *value) {
    client.publish(topic, value);
}

void MqttCom::publish(const char *topic, int value) {
    char msg[10];
    sprintf(msg, "%d", value);
    client.publish(topic, msg);
}

void MqttCom::publish(const char *topic, float value) {
    String msg = "";
    msg += value;
    client.publish(topic, msg.c_str());
}
```

NodeMCU MQTT

❖ ex05/app.ino

```
#include <MqttCom.h>
#include <Led.h>

const char *ssid = "Campus7_Room4_2.4GHz";
const char *password = NULL;
const char *mqtt_server = "192.168.0.159"; // mqtt broker ip address

MqttCom com;
Led led(BUILTIN_LED);

int value = 0;
```

NodeMCU MQTT

❖ ex05/app.ino

```
void callback(char *topic, byte *payload, unsigned int length) {
    char buf[128];
    memcpy(buf, payload, length);
    buf[length] = '\0';

    com.print(0, topic);
    com.print(1, buf);

    if (buf[0] == '1') {
        led.setValue(LOW);
    } else {
        led.setValue(HIGH);
    }
}
```

NodeMCU MQTT

❖ ex05/app.ino

```
void publish() {
    char msg[50];
    ++value;
    sprintf(msg, "hello world #%ld", value);
    com.publish("outTopic", msg);
}

void setup() {
    com.init(ssid, password);
    com.setServer(mqtt_server, "inTopic", callback);
    com.setInterval(2000, publish);
}

void loop() {
    com.run();
}
```

NodeMCU mqtt

❖ publish 확인

- mosquitto_sub -v -h localhost -t outTopic

❖ subscribe 확인

- mosquitto_pub -h localhost -t inTopic -m 1
  - LED ON

- mosquitto_pub -h localhost -t inTopic -m 0
  - LED OFF