

ESP32로 만드는 IoT 월드 with 아두이노 IDE 2

프로젝트명 Ch6.MQTT

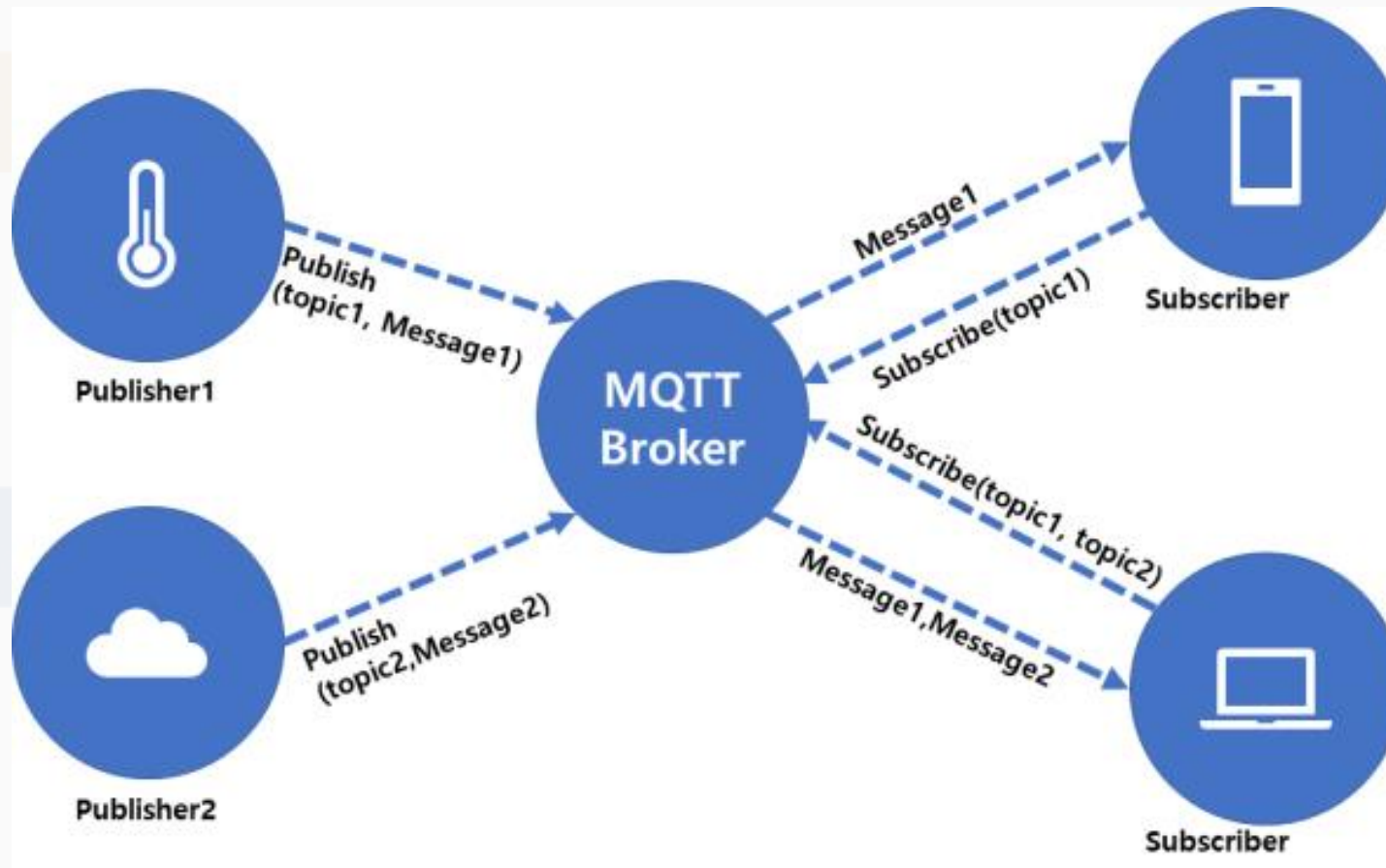
난이도 ★★★★★



학습요약

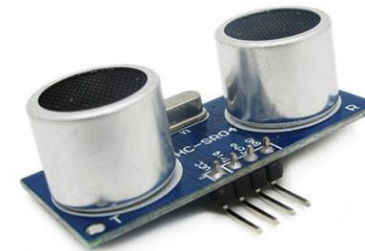
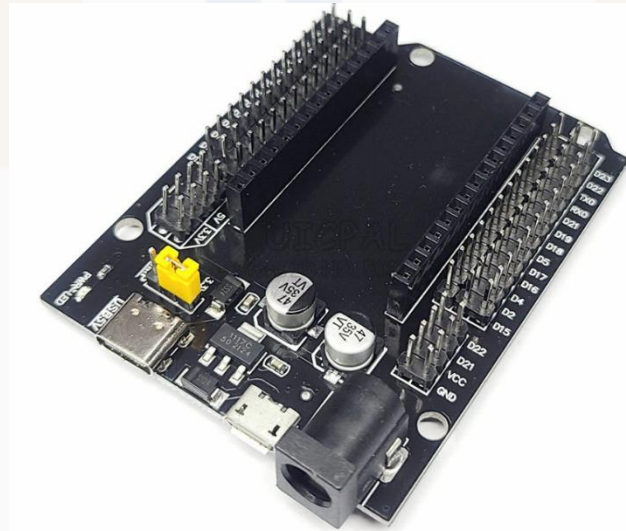
학습목표	MQTT를 활용한 IOT 서비스를 구성해 봅시다.
핵심 키워드	ESP32, IoT, 사물인터넷, MTQQ, 초음파센서, 능동부저
준비물	ESP32, ESP32 확장 실드, 초음파센서, 능동부저
학습 시간	2시간
학습 난이도	★★★★☆☆

MQTT는 Message Queuing Telemetry Transport의 약자로, Machine-to-Machine(M2M) 및 Internet of Things(IoT) 응용 프로그램을 위해 설계된 경량 프로토콜입니다.

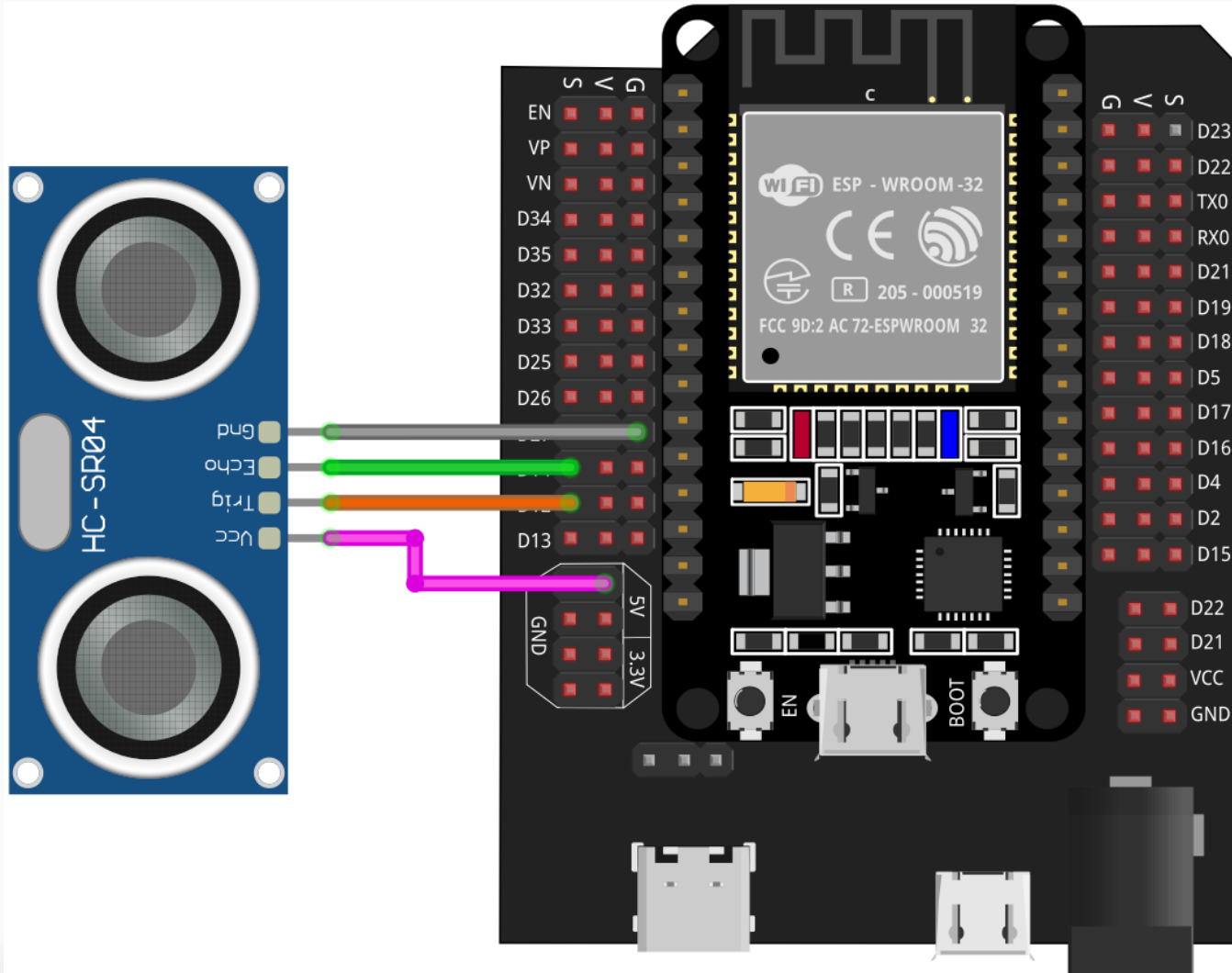


초음파 센서는 음파 파동을 사용하여 거리를 측정하는 센서입니다.
초음파 센서를 ESP32에 연결해 거리를 측정해 보겠습니다.

준비물 : ESP32, ESP32 확장 실드, 초음파 센서



회로 연결하기



ESP32 쉴드	초음파센서
S : D12	Trig
S : D14	Echo
5V	Vcc
G	Echo


```
1  const int trigPin = 12; // 초음파 센서의 트리거 핀
2  const int echoPin = 14; // 초음파 센서의 에코 핀

8  void setup() {
9      Serial.begin(115200); // 시리얼 통신 시작 (통신 속도: 115200bps)
10     pinMode(trigPin, OUTPUT); // 트리거 핀을 출력으로 설정
11     pinMode(echoPin, INPUT); // 에코 핀을 입력으로 설정
12 }

15 digitalWrite(trigPin, LOW); // 트리거 핀 LOW로 초기화
16 delayMicroseconds(2); // 2 마이크로 초 대기
17 digitalWrite(trigPin, HIGH); // 트리거 핀 HIGH 로 설정하여 초음파 송신
18 delayMicroseconds(10); // 10 마이크로 초 동안
19 digitalWrite(trigPin, LOW); // 초음파 송신 종료
20
21 duration = pulseIn(echoPin, HIGH); // 에코 핀에서 초음파의 왕복 시간 측정
22
23 distanceCm = duration * 0.034/2; // CM거리 계산
24
25 distanceInch = distanceCm * 0.393701; // 인치로 변환
```

※ 초음파 센서 Triger 핀은 Output, Echo 핀은 Input으로 설정해 줍니다.

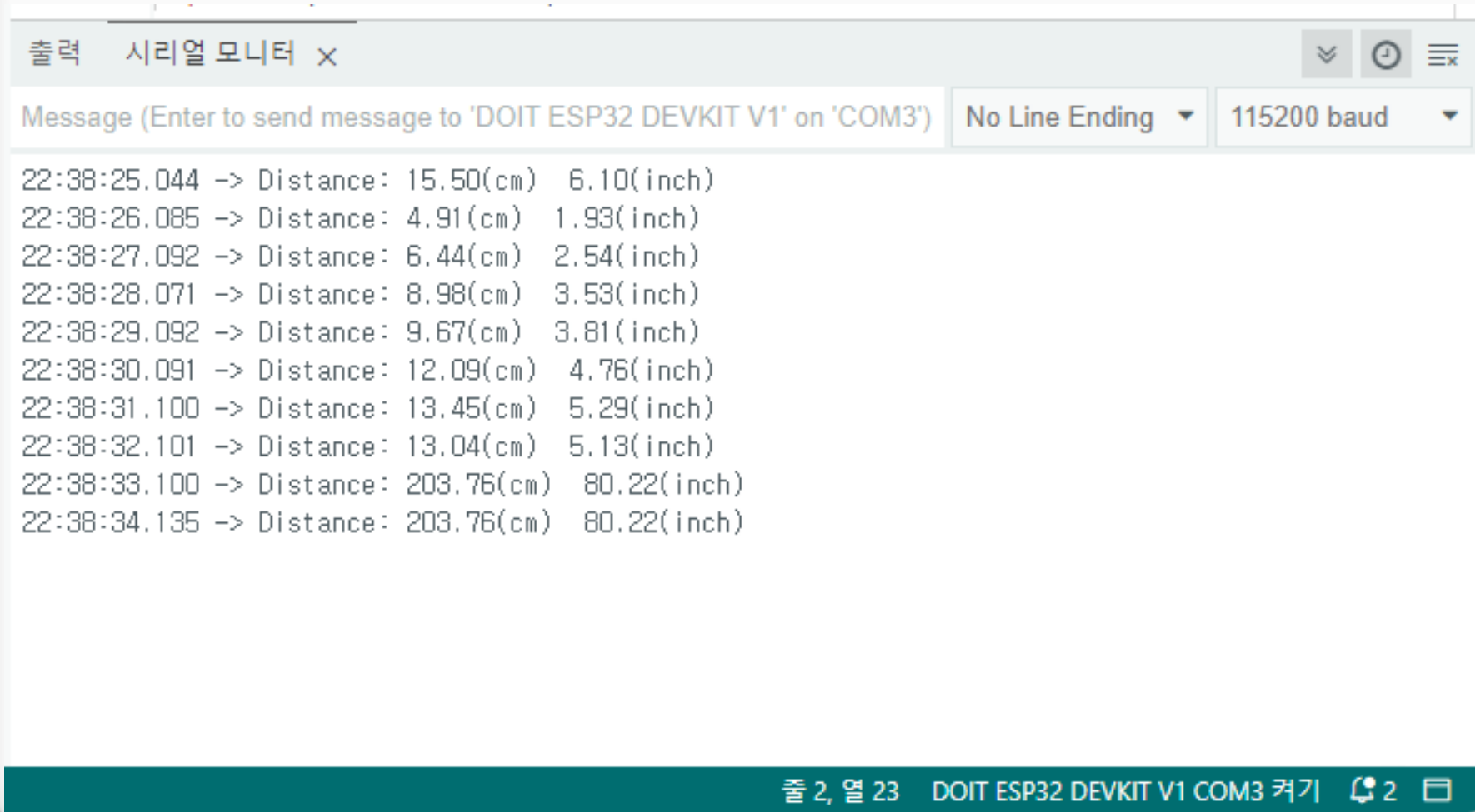
※ Triger 핀으로 초음파를 송신

※ echo 핀에 수신 된 초음파 시간을 측정

※ 시간에 따른 거리 계산

실행 결과

1초 간격으로 초음파센서와 물체와의 거리가 측정되어 출력



The screenshot shows a serial monitor window titled "출력 시리얼 모니터 x". The window displays a series of distance measurements in centimeters and inches, taken at 1-second intervals. The measurements show the sensor detecting a nearby object for the first 6 seconds, then a much further object for the last 2 seconds. The status bar at the bottom indicates the connection is to "DOIT ESP32 DEVKIT V1 COM3" at a baud rate of 115200.

```
Message (Enter to send message to 'DOIT ESP32 DEVKIT V1' on 'COM3') No Line Ending 115200 baud
```

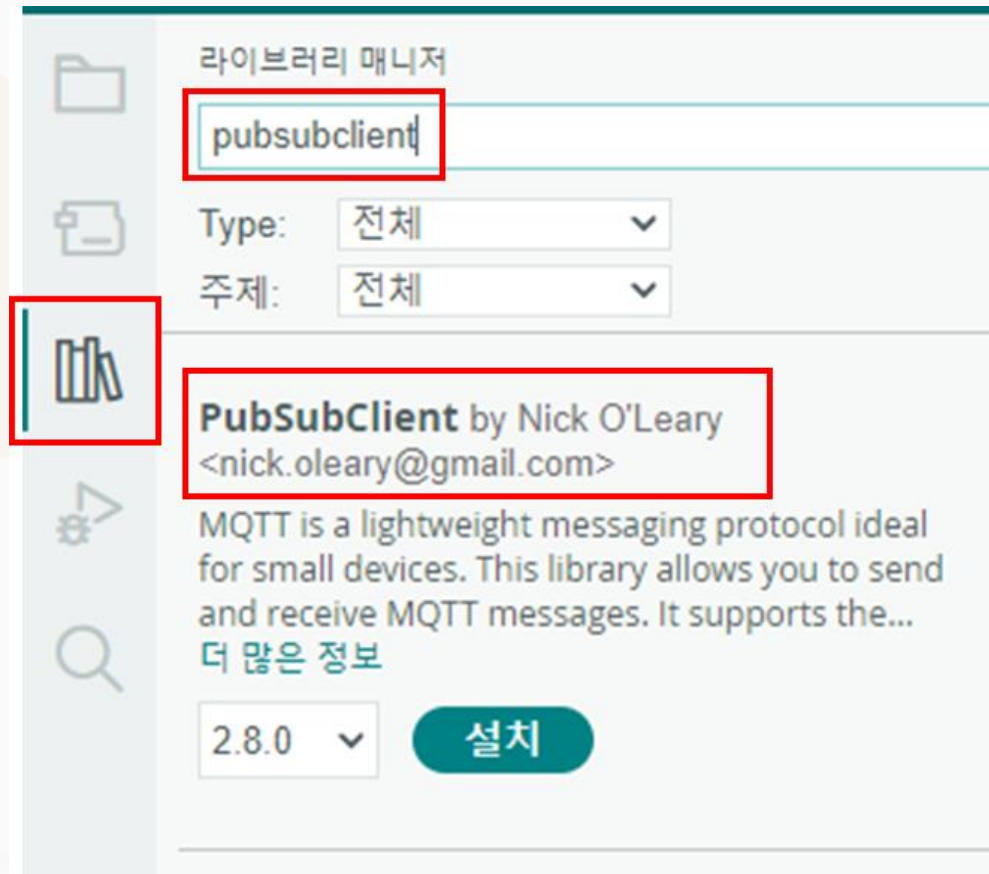
```
22:38:25.044 -> Distance: 15.50(cm) 6.10(inch)
22:38:26.085 -> Distance: 4.91(cm) 1.93(inch)
22:38:27.092 -> Distance: 6.44(cm) 2.54(inch)
22:38:28.071 -> Distance: 8.98(cm) 3.53(inch)
22:38:29.092 -> Distance: 9.67(cm) 3.81(inch)
22:38:30.091 -> Distance: 12.09(cm) 4.76(inch)
22:38:31.100 -> Distance: 13.45(cm) 5.29(inch)
22:38:32.101 -> Distance: 13.04(cm) 5.13(inch)
22:38:33.100 -> Distance: 203.76(cm) 80.22(inch)
22:38:34.135 -> Distance: 203.76(cm) 80.22(inch)
```

줄 2, 열 23 DOIT ESP32 DEVKIT V1 COM3 켜기 2

초음파 센서 데이터를 발행하는 Publisher를 만들어 보겠습니다

라이브러리 설치

"PubSubClient by Nick O'Leary"를 검색하고 설치합니다.




```

1  #include <WiFi.h>           //Wi-Fi 연결 관련 라이브러리
2  #include <PubSubClient.h>   //MQTT 프로토콜을 사용하기 위한 라이브러리
3
4  // 다음 변수들을 당신의 SSID와 비밀번호로 대체하세요.
5  const char* ssid = "여기에_당신의_SSID_입력";
6  const char* password = "여기에_당신의_비밀번호_입력";
    
```

※ 실습환경에 맞게 SSID와 PASSWORD를 수정해 줍니다.

```

11 // MQTT 브로커 주소를 여기에 입력하세요.
12 const char* mqtt_server = "test.mosquitto.org";
13
14 WiFiClient espClient;
15 PubSubClient client(espClient);
    
```

※ MQTT 서버 주소를 설정해 줍니다.

```

21  setup_wifi();
22  client.setServer(mqtt_server, 1883); // mqtt 서버 설정 (주소, 포트)
    
```

※ MQTT client객체를 선언하고 wifi가 연결되면 서버와 포트를 클라이언트에 지정해 줍니다.

```
73 void loop() {  
74   if (!client.connected()) {  
75     reconnect();  
76   }  
77   // 클라이언트가 메시지를 처리하고 서버와 연결 유지  
78   client.loop();  
79  
80   long now = millis();  
81   if (now - lastMsgTime > 1000) { //1초 간격  
82     lastMsgTime = now;  
83     // 초음파 센서 값을 읽어옵니다.  
84     float sensorValue = readUltrasonicSensor();  
85     char sensorString[8];  
86     dtostrf(sensorValue, 1, 2, sensorString);  
87     client.publish("user1/esp32/ultra", sensorString);  
88   }  
89 }  
51 void reconnect() {  
52   // 연결이 될 때까지 반복  
53   while (!client.connected()) {  
54     Serial.print("MQTT 연결 시도 중...");  
55     // 랜덤 클라이언트 ID 생성  
56     String clientId = "ESP32Client-";  
57     clientId += String(random(0xffff), HEX);  
58     Serial.print("클라이언트 ID: ");
```

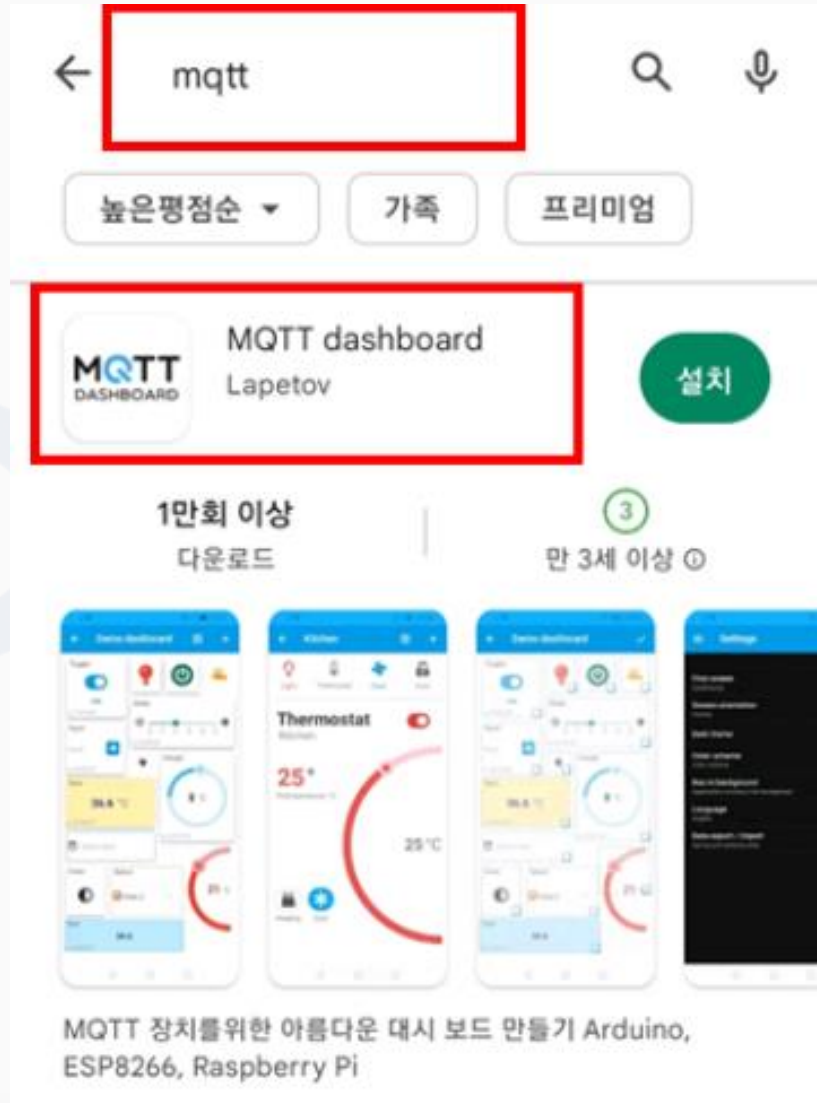
※ MQTT 서버에 연결을 시도합니다.

※ 초음파 센서 값을 "user1/esp32/ultra" 토픽으로 Publish합니다.

※ MQTT서버 연결시에 고유한 ClientID를 생성하여 줍니다.

실행 결과

MQTT dashboard를 앱 설치



실행 결과

"초음파 센서" 대시 보드 추가

Dashboards
+

Demo dashboard
Demo widgets
Kitchen
Kitchen control

←
New dashboard
✓

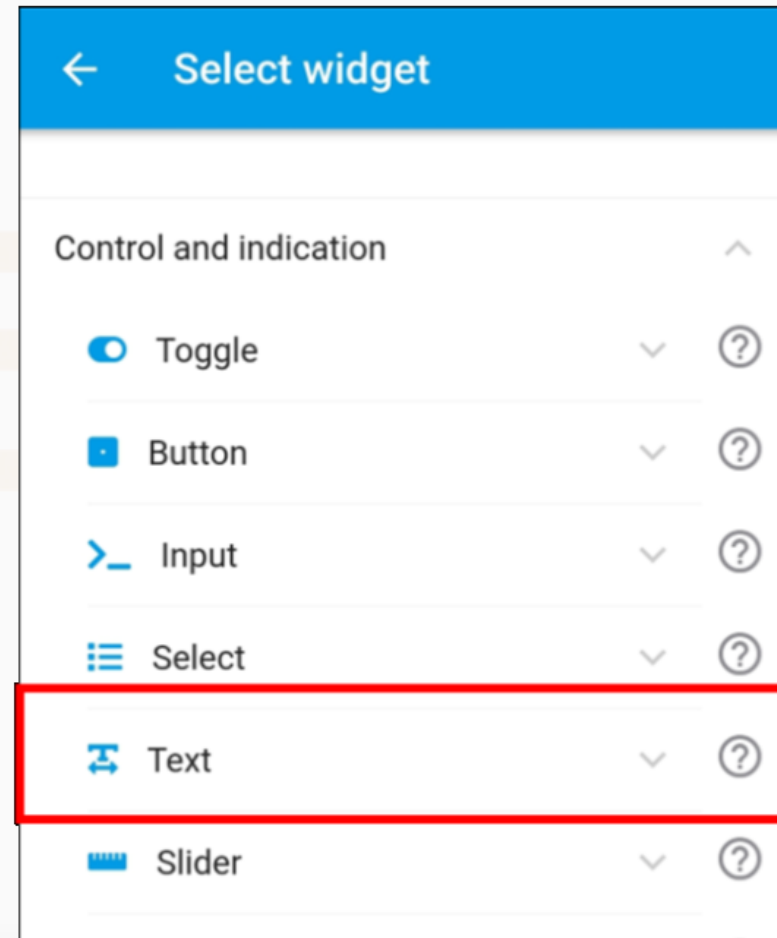
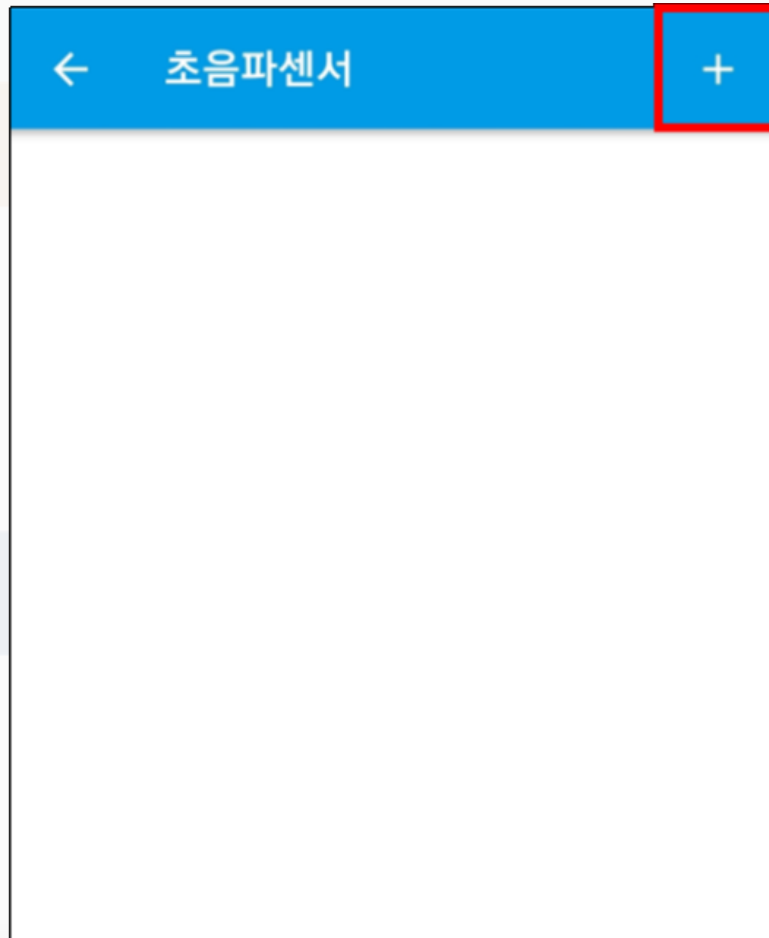
Name*
초음파센서
Parent dashboard
Not set
Description

Dashboards
+

Demo dashboard
Demo widgets
Kitchen
Kitchen control
초음파센서

실행 결과

"초음파 센서" 대시 보드에 텍스트 위젯 추가



실행 결과

텍스트 위젯 설정

← New widget ✓

Name*
초음파센서값

MQTT

MQTT enable ☒

MQTT connection*
Connection 1

Subscribe to topic
user1/esp32/ultra

Payload (subscribe) is JSON (?) ☐

← Edit widget ✓

Name*
초음파센서값

MQTT

Design

Text value

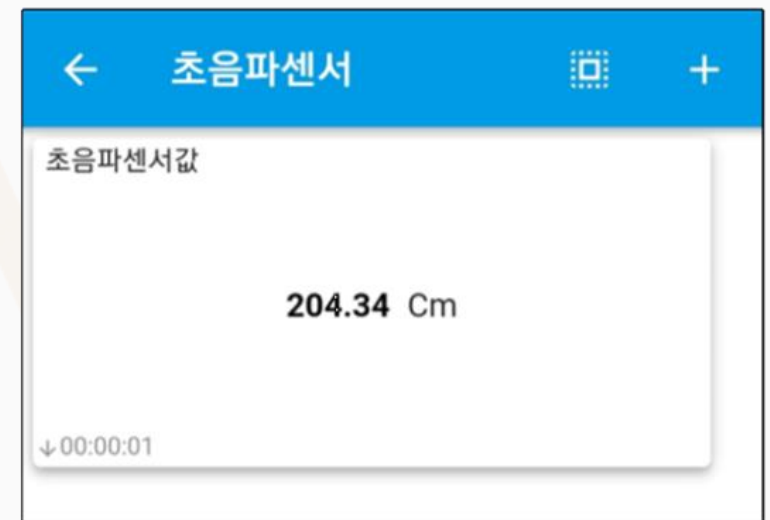
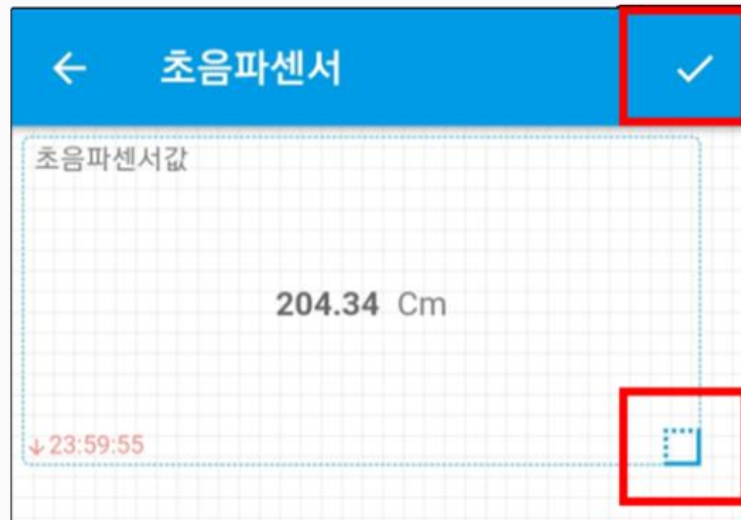
Unit

Show unit ☒

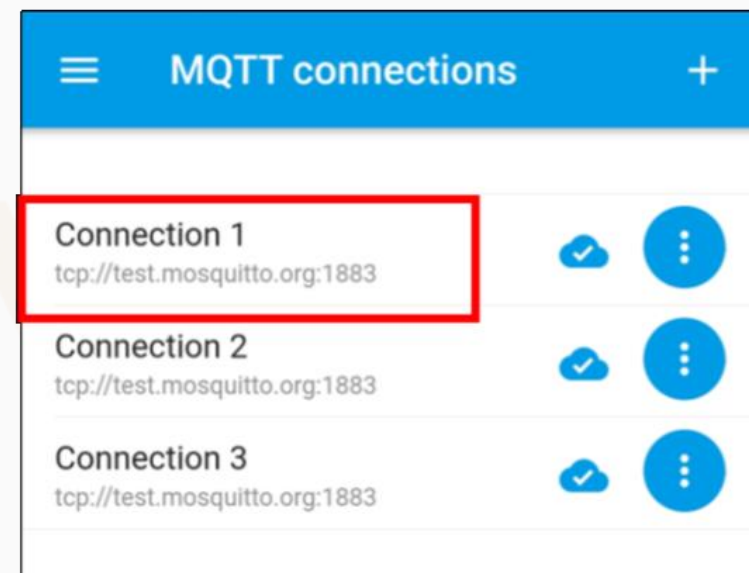
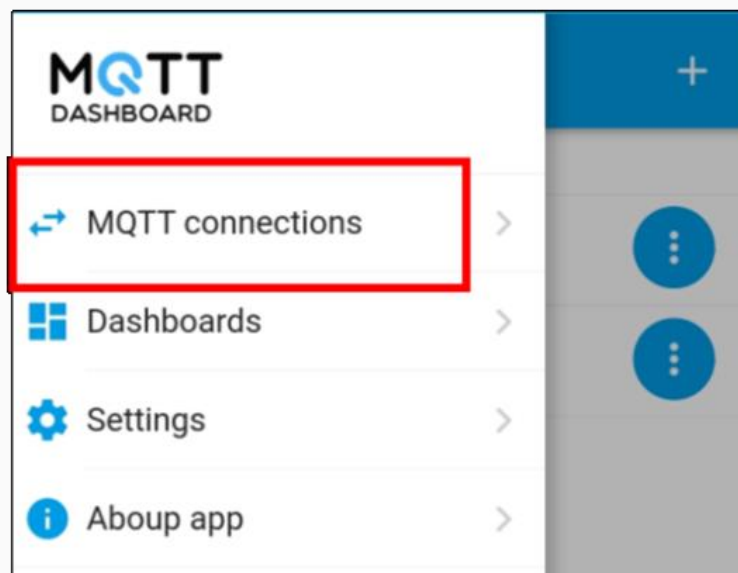
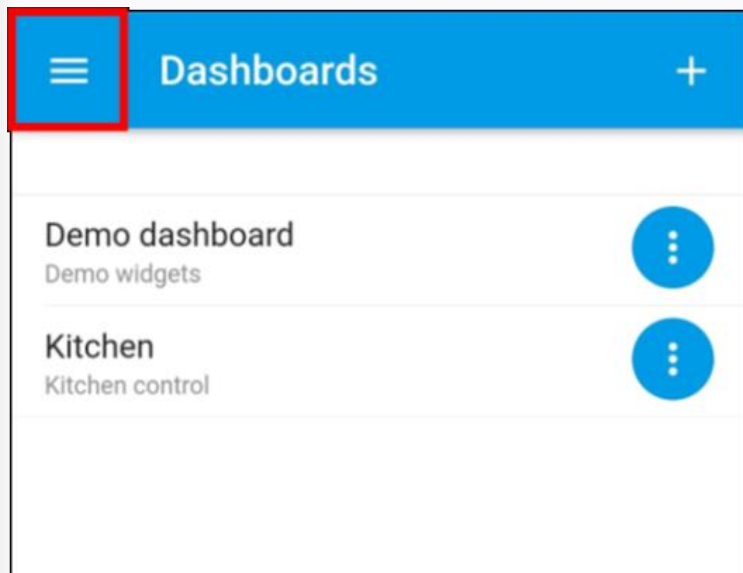
Unit
Cm

실행 결과

위젯 크기 및 위치 설정

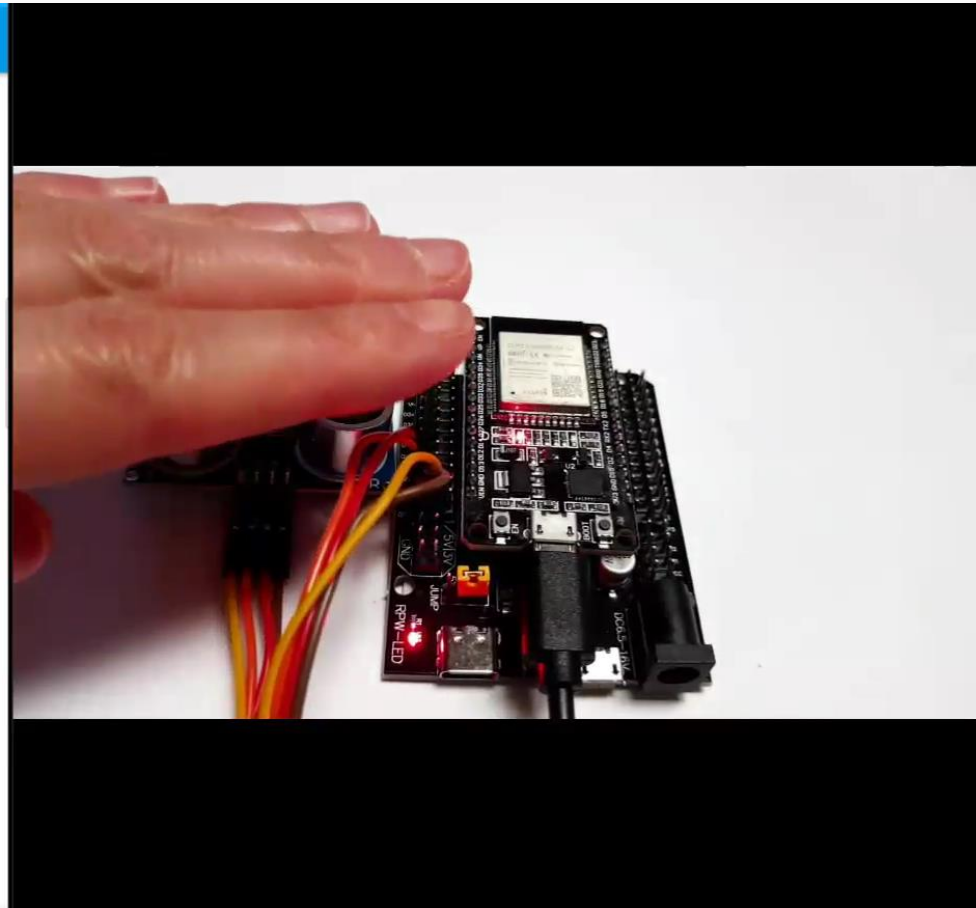
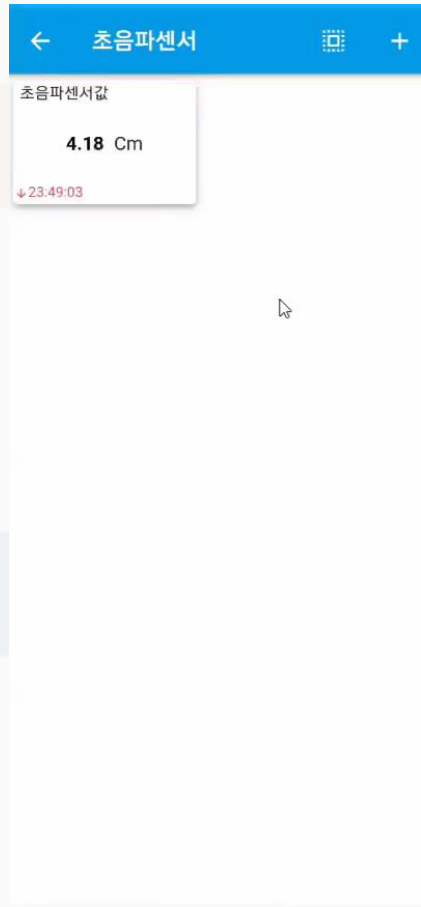


실행 결과
커넥션 정보 확인



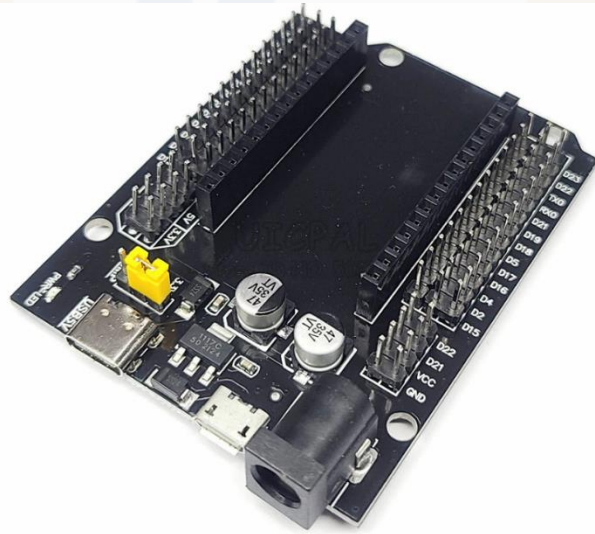
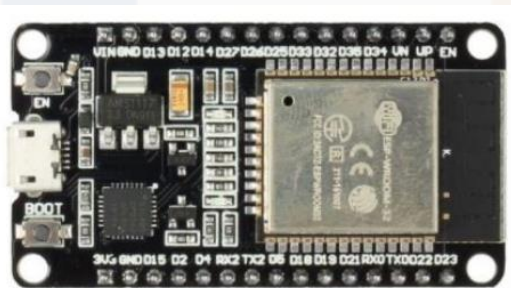
실행 결과

초음파 센서 거리 값 변화를 MQTT Dash보드 앱에서 확인하기

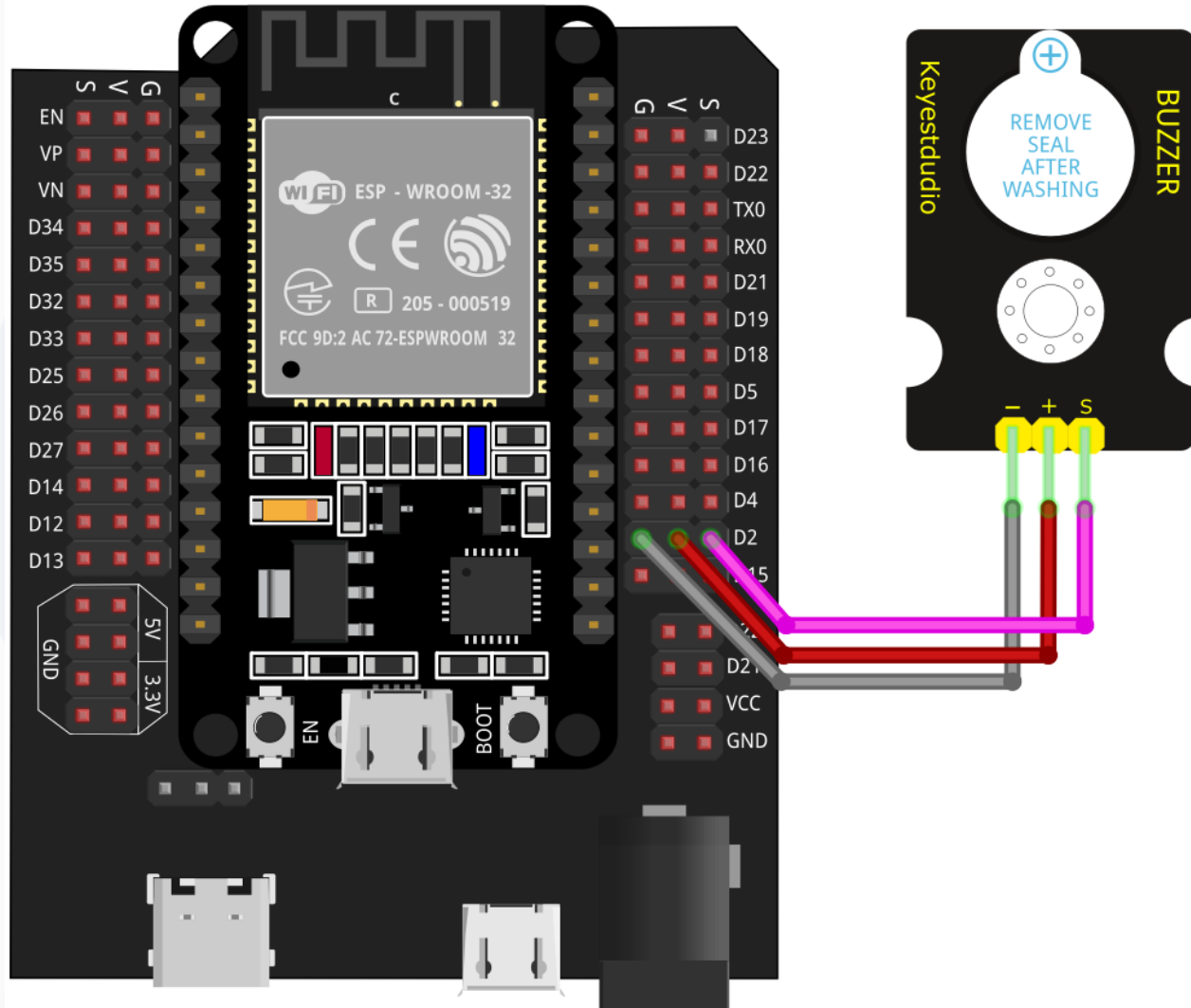


능동 부저는 전기 신호를 가지고 직접 소리를 발생시키는 부저입니다.
ESP로 부저를 켜고 끄는 간단한 예제를 실행해 보겠습니다.

준비물 : ESP32, ESP32 확장 실드, 능동부저



회로 연결하기



ESP32 실드	능동 부저
S : D2	S
V	V
G	G

```
1  const int buzzerPin = 2; //led 핀 번호 설정
2
3  void setup (){
4  pinMode (buzzerPin,OUTPUT );//buzzerPin 을 출력으로 설정
5  }
6
7  void loop (){
8  digitalWrite (buzzerPin, HIGH ); //buzzerPin 에 HIGH 값 쓰기
9  delay (500); // 0.5 초 기다리기
10 digitalWrite (buzzerPin, LOW ); //buzzerPin 에 LOW 값 쓰기
11 delay (1000 ); // 1 초 기다리기
12 }
```

※ 능동 부저는 자체적으로 진동체를 가지고 있어 전원만 넣어 주면 소리가 납니다.

"user1/esp32/buzzer"토픽을 구독하고, 해당 토픽으로 전송된 메시지를 수신하여 부저를 켜고 끄는 Subscriber를 실습해 보겠습니다.

```

9  const char* mqtt_server = "test.mosquitto.org";
10 const int mqttPort = 1883;
11 const char* mqttTopic = "user1/esp32/buzzer"; // 사용자에게 맞게 변경
12 const int buzzerPin = 2;                      // 부저에 연결된 GPIO 핀 번호
    
```

※ MQTT 서버 및 포트, 토픽 정보 설정

```

43 client.setServer(mqtt_server, mqttPort); // Broker 정보 설정 (주소, 포트)
44 client.setCallback(callback);           // 메시지 수신시 동작으로 callback
    
```

※ client에 연결할 서버 정보 넘겨주고, 메시지 수신시 동작할 Callback 함수 등록

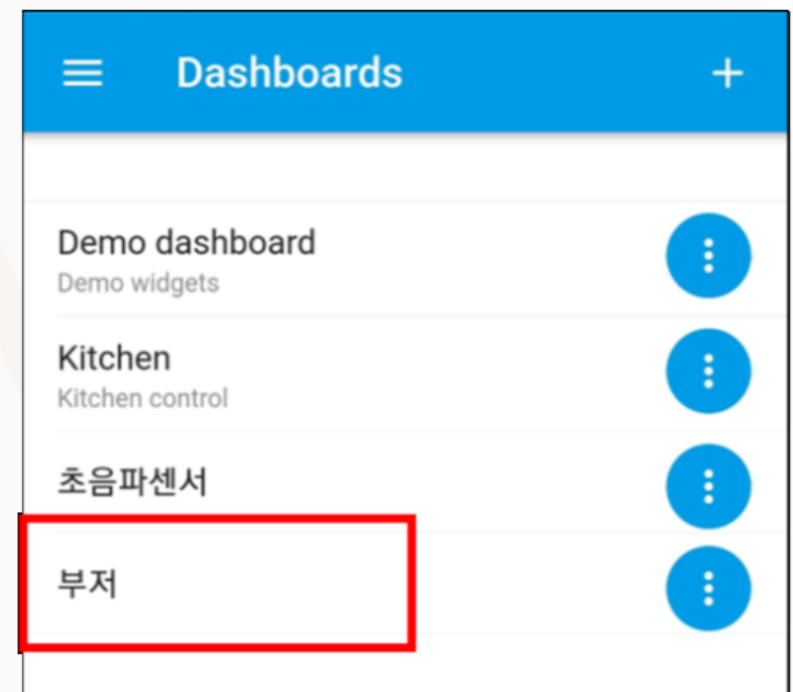
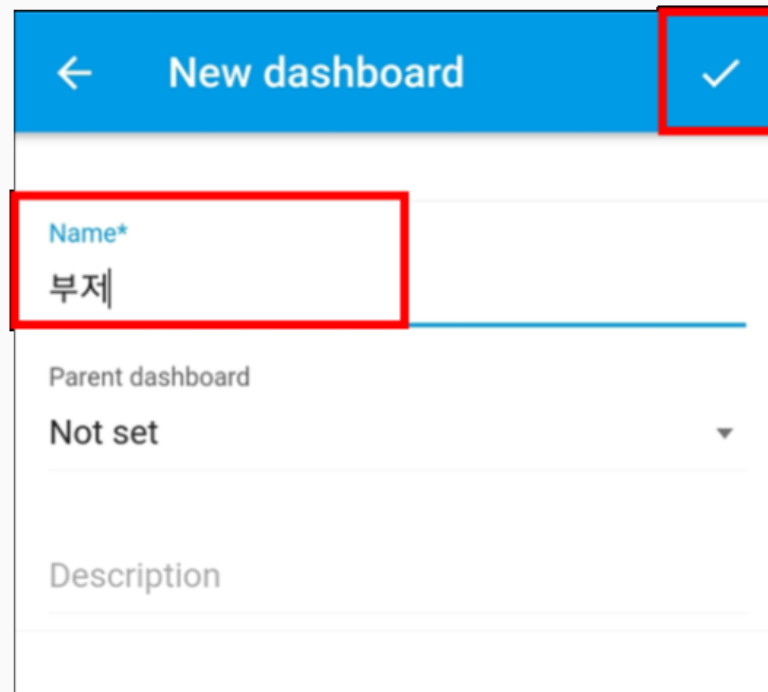
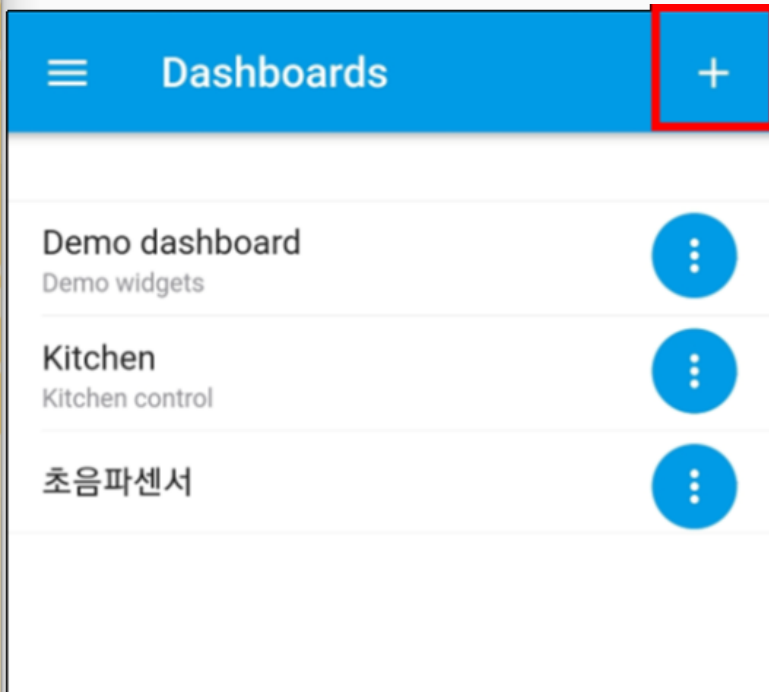
```

56 // 메시지 수신시 호출되어 동작함
57 void callback(char* topic, byte* payload, unsigned int length) {
66     if (strcmp(topic, mqttTopic) == 0) {
67         if (payload[0] == '1') {
68             // 부저를 켜는 코드 작성
69             digitalWrite(buzzerPin, HIGH);
70         } else if (payload[0] == '0') {
71             // 부저를 끄는 코드 작성
72             digitalWrite(buzzerPin, LOW);
73         }
74     }
75 }
    
```

※ 메시지 수신시 부저를 켜고 끄는 동작 정의하는 콜백 함수

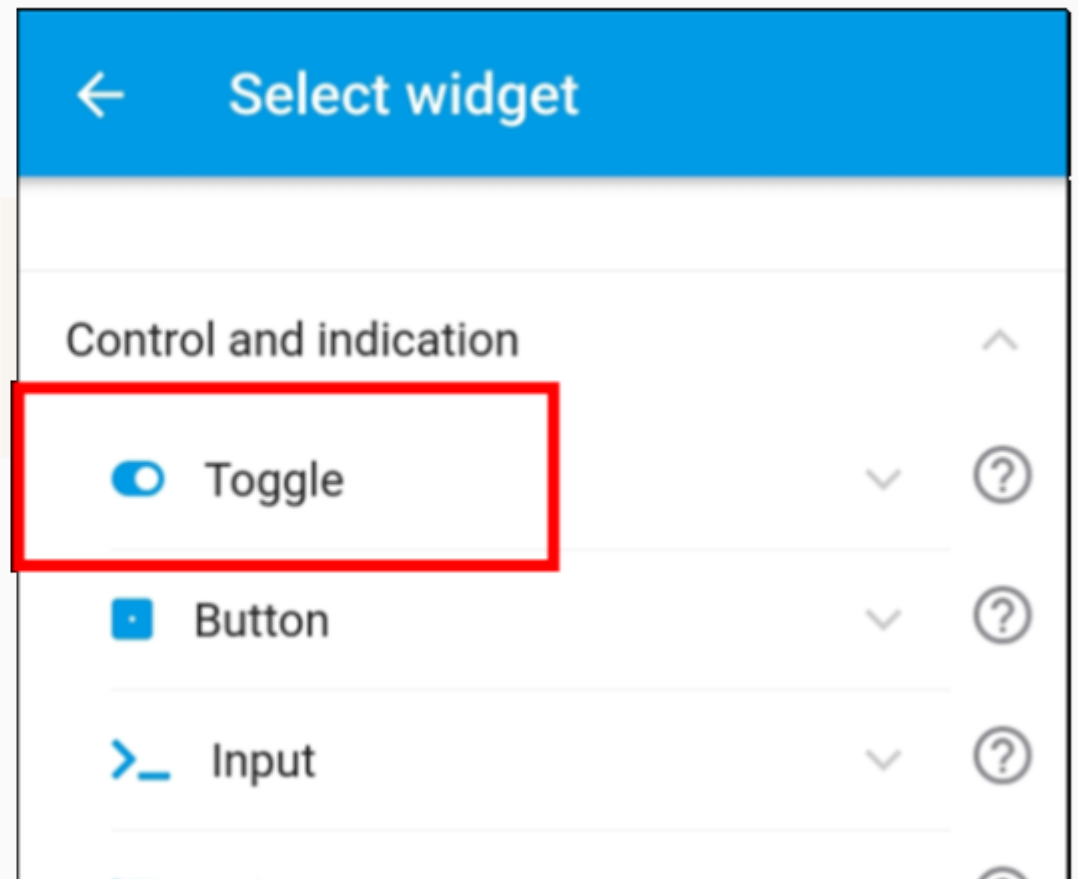
실행 결과

"부저" 대시 보드 생성



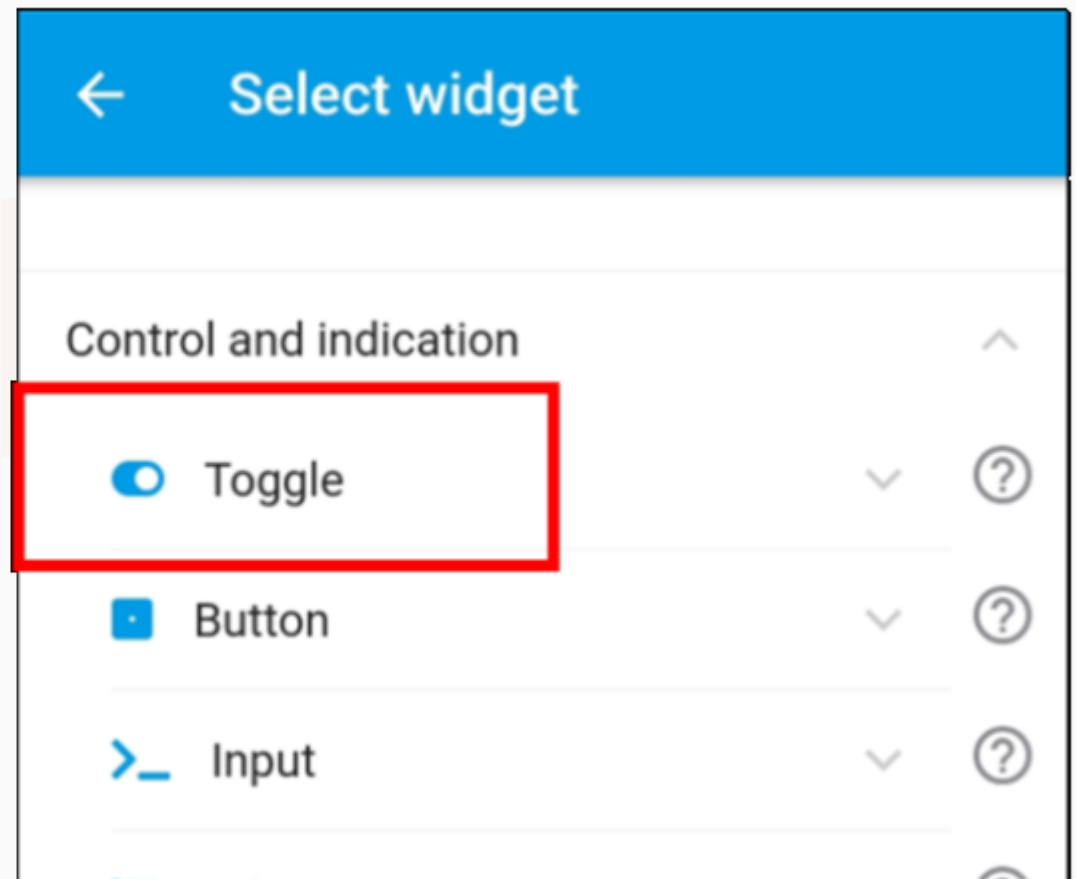
실행 결과

"부저" 대신 보드에 토글 위젯 추가



실행 결과

"부저" 대신 보드에 토글 위젯 추가



실행 결과
토글 위젯 설정

← New widget ✓

Name*
부저 토글

MQTT

MQTT enable

MQTT connection*
Connection 1

Payload (subscribe) is JSON ?

Qos for subscribe
0

Topic for publish
user1/esp32/buzzer

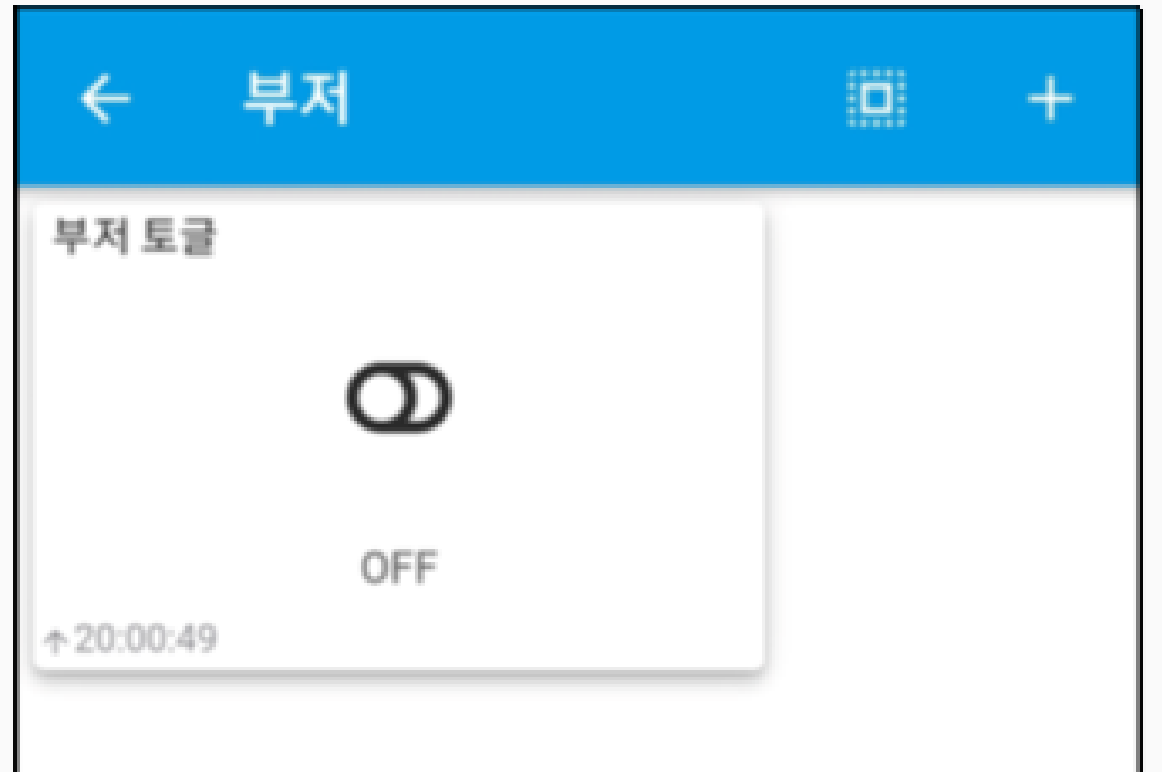
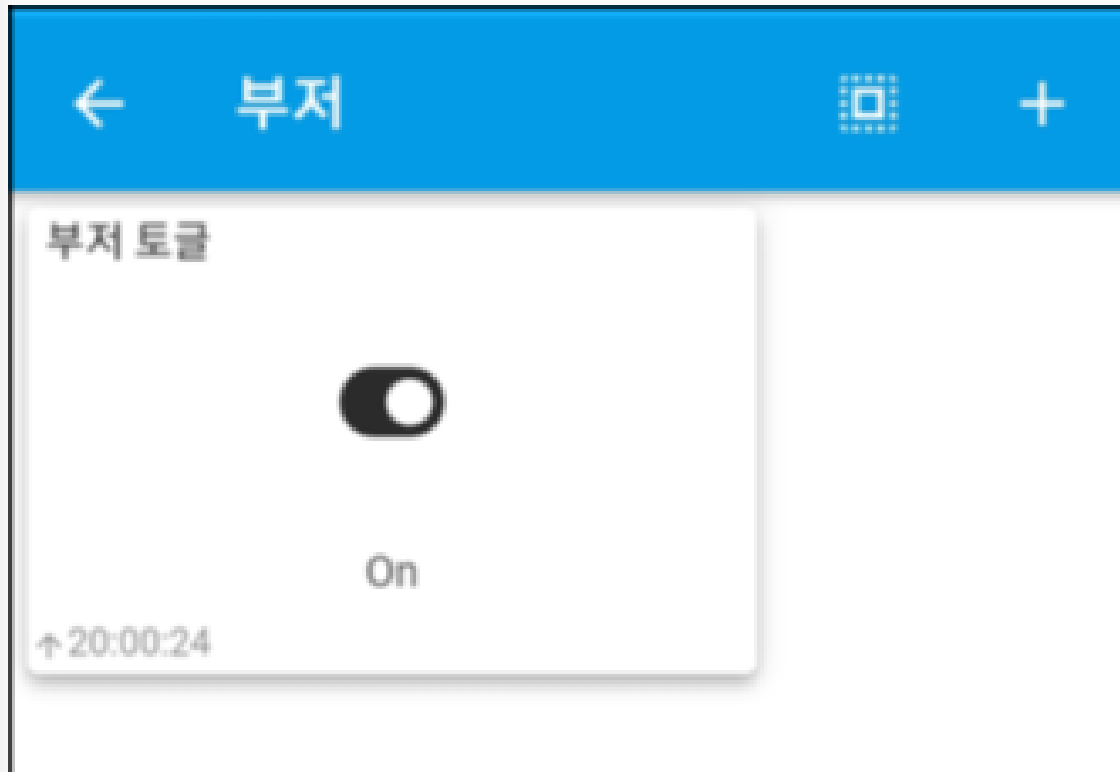
☒ Retain Qos for publish
0

Confirm send

Design

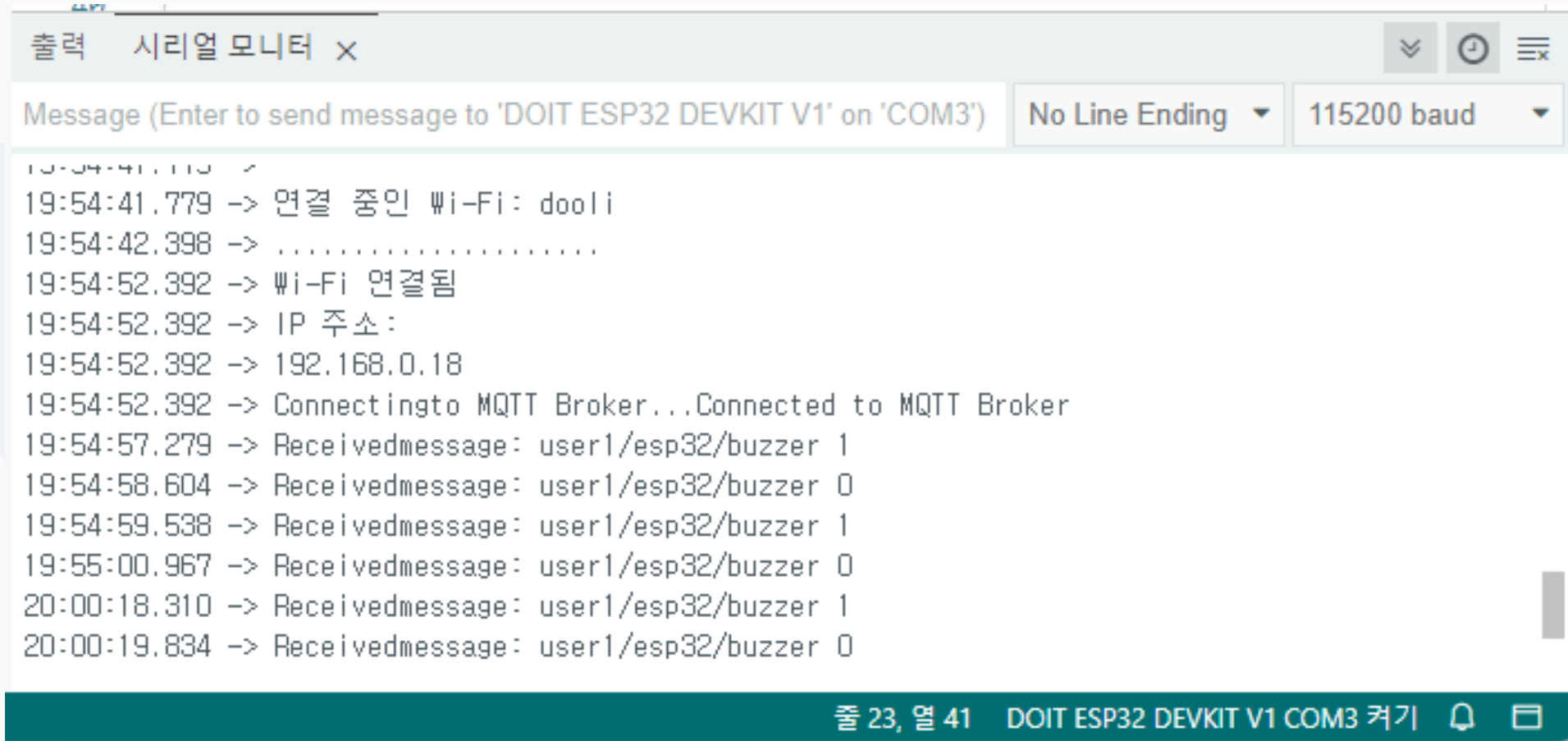
실행 결과

대시 보드 앱에서 토글 상태 변경 하기



실행 결과

토글 상태 변경에 따라 메세지 전송되며, 부저 소리 확인



```

출력 시리얼 모니터 x
Message (Enter to send message to 'DOIT ESP32 DEVKIT V1' on 'COM3') No Line Ending 115200 baud
19:54:41.113 ->
19:54:41.779 -> 연결 중인 Wi-Fi: dooli
19:54:42.398 -> .....
19:54:52.392 -> Wi-Fi 연결됨
19:54:52.392 -> IP 주소:
19:54:52.392 -> 192.168.0.18
19:54:52.392 -> Connecting to MQTT Broker...Connected to MQTT Broker
19:54:57.279 -> Receivedmessage: user1/esp32/buzzer 1
19:54:58.604 -> Receivedmessage: user1/esp32/buzzer 0
19:54:59.538 -> Receivedmessage: user1/esp32/buzzer 1
19:55:00.967 -> Receivedmessage: user1/esp32/buzzer 0
20:00:18.310 -> Receivedmessage: user1/esp32/buzzer 1
20:00:19.834 -> Receivedmessage: user1/esp32/buzzer 0
    
```

HTTP, MQTT 비교

MQTT

특성	HTTP	MQTT
사용 사례	웹 애플리케이션, API, 서버-클라이언트 모델	IoT, 실시간 메시징, M2M 통신 등
프로토콜	요청-응답	게시-구독
통신 방식	클라이언트-서버 간 동기 통신	게시자-구독자 간 비동기 통신
연결 유지	요청 후 연결 닫힘	연결 유지 가능
대역폭 및 리소스	높은 대역폭 및 리소스 사용량	낮은 대역폭 및 리소스 사용량
메시지 크기 제한	큰 메시지 전송 가능	작은 메시지 전송에 최적화
확장성	서버의 확장성 제한	수백만 개의 클라이언트 처리 가능
신뢰성	데이터 손실 가능	다양한 QoS 레벨 제공으로 데이터 신뢰성 보장
보안	추가 보안 기능 필요	SSL/TLS를 통한 데이터 암호화



[저작권 안내]

*본 콘텐츠는 아이씨뱅크(ICBANQ)에 소유권이 있습니다. 소유권자의 허가를 받지 않고 무단으로 수정, 삭제, 배포, 상업적 사용을 할 수 없으며 위반시 민형사적 법적 처벌을 받을 수도 있습니다.