



4.2

MQTT in ESP32 SIMULATOR

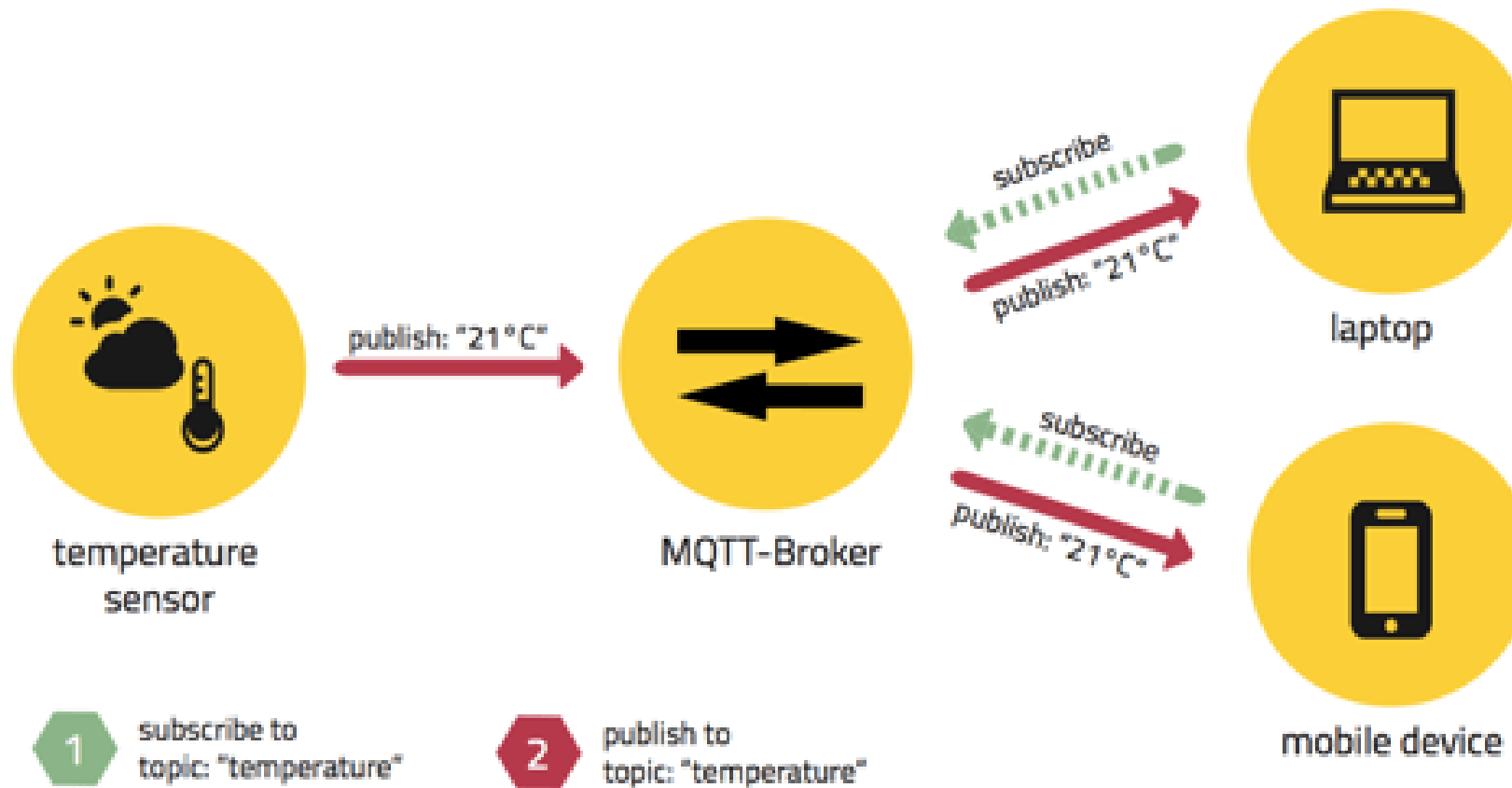
The background features a light gray circuit board pattern with various IoT-related icons in circular frames. These include a radio tower, a water drop, a thermometer, a lightbulb, a microchip, a laptop, and server racks. A large central circle contains a Wi-Fi symbol and the text 'MQTT'.

WHAT IS MQTT?



- **MQTT (Message Queuing Telemetry Transport)** is a publish-subscribe network protocol that transports messages between devices.

HOW DOES MQTT WORK?



broker.mqtt-
dashboard.com



test.mosquitto.org



broker.hivemq.com



MQTT BROKER

The background features a light gray circuit board pattern with various IoT-related icons in circular nodes. These include a radio tower, a water drop, a thermometer, a lightbulb, a microchip, a laptop, and server racks. A large central circle contains a Wi-Fi symbol. The text "MQTT" is faintly visible behind the main title.

Send message to MQTT

1. Add *PubSubClient* Library
2. Include lib in program

```
#include <WiFi.h>
#include "PubSubClient.h" ←
const char* ssid = "Wokwi-GUEST";
const char* password = "";
const char* mqttServer = "test.mosquitto.org";
int port = 1883;

WiFiClient espClient;
PubSubClient client(espClient);
```

```
void wifiConnect() {
  WiFi.begin(ssid, password);
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  Serial.println(" Connected!");
}

void setup() {
  Serial.begin(9600);
  Serial.print("Connecting to WiFi");

  wifiConnect();

  client.setServer(mqttServer, port);
}
```

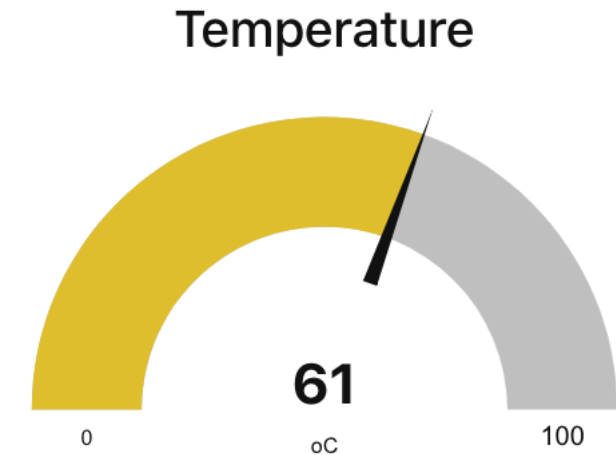
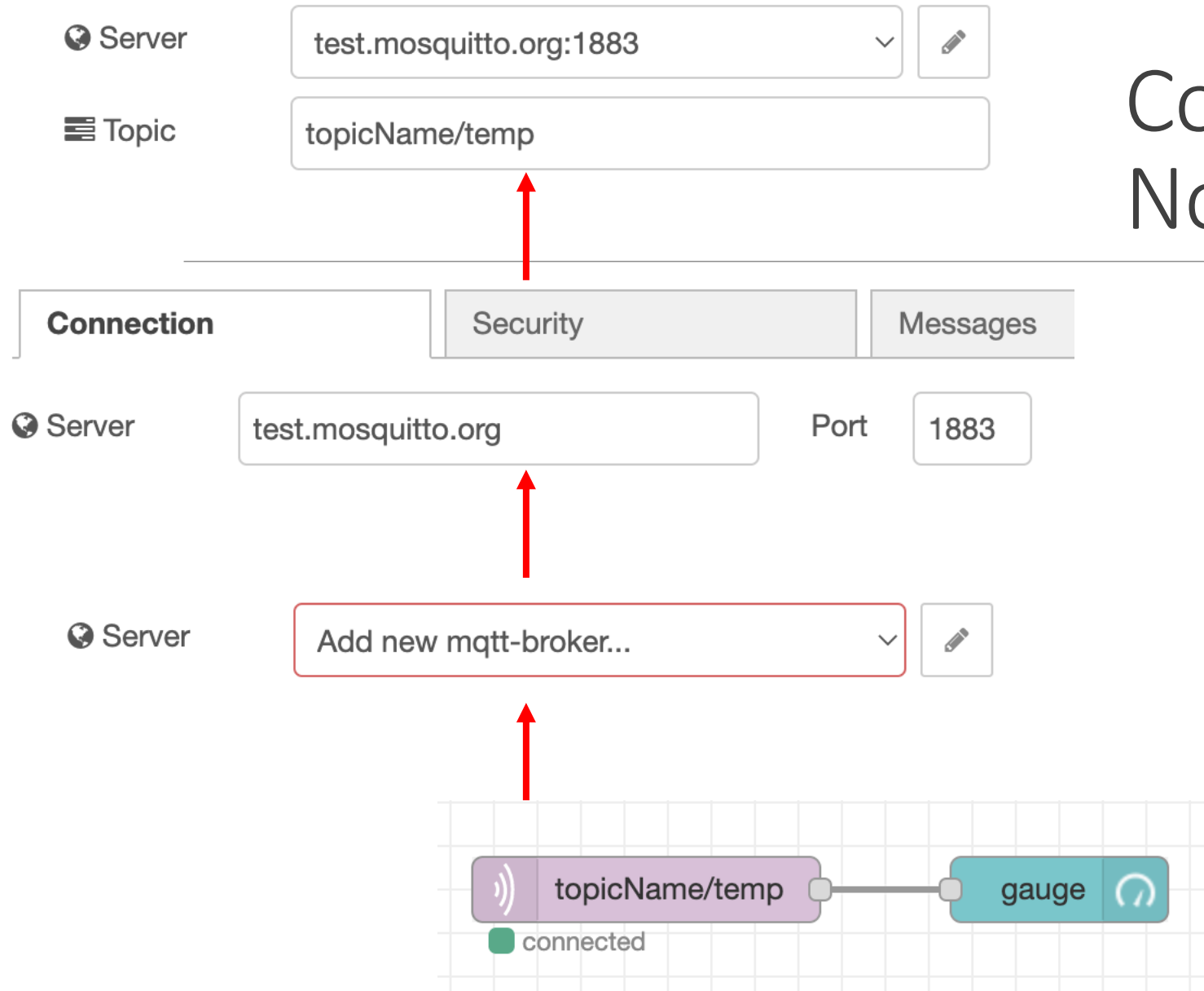
Set unique-id
your device



```
void mqttReconnect() {  
  while (!client.connected()) {  
    Serial.print("Attempting MQTT connection...");  
    if (client.connect("12345678")) {  
      Serial.println(" connected");  
    } else {  
      Serial.println(" try again in 5 seconds");  
      delay(5000);  
    }  
  }  
}
```

```
void loop() {  
  if (!client.connected()) {  
    mqttReconnect();  
  }  
  client.loop();  
  
  int temp = random(0, 100);  
  char buffer[50];  
  sprintf(buffer, "%d", temp);  
  client.publish("topicName/temp", buffer);  
  delay(5000);  
}
```


Config “MQTT in” in Node-RED





Send real
temperature
from DHT22
to Node-RED

JSON

```
{  
  "temperature": 40,  
  "humidity": 10  
}
```

Label

Temperature

Value format

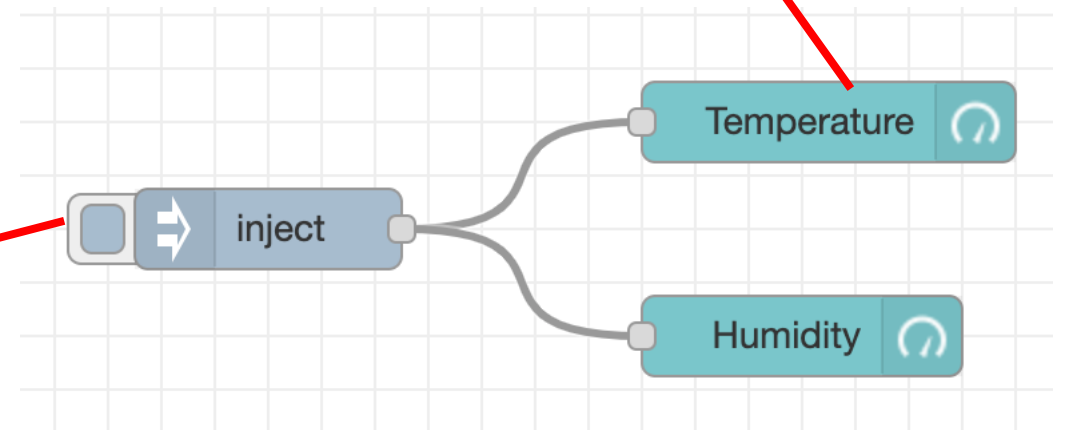
{{msg.payload.temperature}}

Units

oC

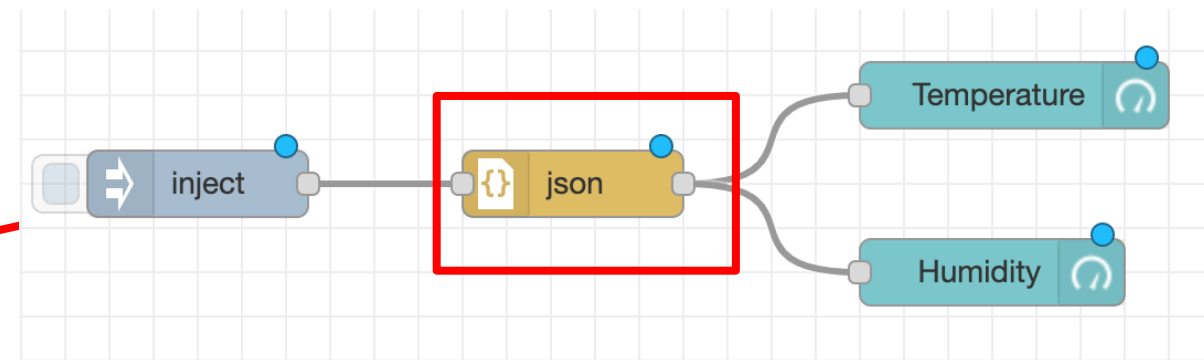
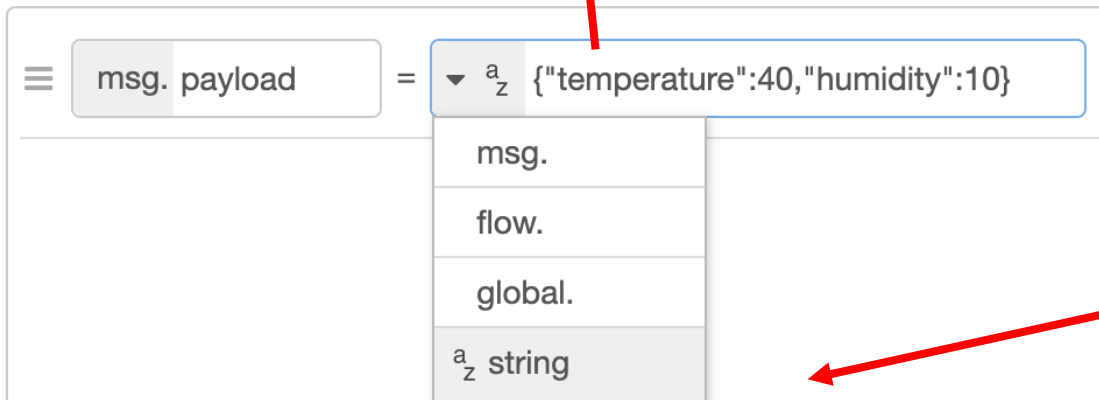
msg. payload = `{ "temperature":40,"humidity":10 }`

- msg.
- flow.
- global.
- `az` string
- `09` number
- boolean
- { } JSON**



Parse JSON string

`{"temperature":40,"humidity":10}`





Send both
temperature
and humidity
from DHT22
to Node-RED

```
char buffer[50];  
sprintf(buffer, "{\"temperature\":%d,\"humidity\":%d}", temp, humidity);  
client.publish("topicName/temp", buffer);
```


The background features a light gray circuit-like pattern with several circular icons: a radio tower, a water drop, a thermometer, a lightbulb, a microchip, a laptop, and server racks. A large central circle contains a Wi-Fi symbol and the text 'MQTT'.

Receive message from MQTT

Subscribe topic

```
void mqttReconnect() {  
  while (!client.connected()) {  
    Serial.print("Attempting MQTT connection...");  
    if (client.connect("12345678")) {  
      Serial.println(" connected");  
      client.subscribe("topicName/led");  
    } else {  
      Serial.println(" try again in 5 seconds");  
      delay(5000);  
    }  
  }  
}
```

```
void setup() {  
    Serial.begin(9600);  
    Serial.print("Connecting to WiFi");  
  
    wifiConnect();  
  
    client.setServer(mqttServer, port);  
    client.setCallback(callback);  
}
```



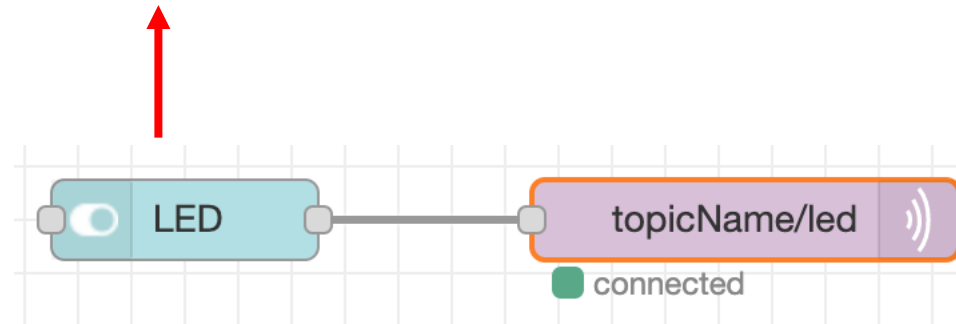
```
void callback(char* topic, byte* message, unsigned int length) {  
    Serial.print(topic);  
    String stMessage;  
    for (int i = 0; i < length; i++) {  
        stMessage += (char)message[i];  
    }  
    Serial.println(stMessage);  
}
```


Config “MQTT out” in Node-RED

☒ When clicked, send:

On Payload ▼ a_z on

Off Payload ▼ a_z off



```
void callback(char* topic, byte* message, unsigned int length) {  
  Serial.println(topic);  
  String stMessage;  
  for (int i = 0; i < length; i++) {  
    stMessage += (char)message[i];  
  }  
  Serial.println(stMessage);  
}
```

Diagram illustrating the MQTT configuration for the LED node:

- The `topic` parameter in the `Serial.println(topic);` line is linked to the `topicName/led` field in the MQTT configuration.
- The `stMessage` variable in the `Serial.println(stMessage);` line is linked to the `on` and `off` payload options in the MQTT configuration.

Server

test.mosquitto.org:1883

Topic

topicName/led



Turn LED
on/off from
Node-RED
