

Date	10 November 2022
Team ID	PNT2022TMID20484
Project Title	Industry-Specific Intelligent Fire Management System

Display the temperature values:

Submitted by

Muthu Lakshmi P – 49621911089

Shirlin Jeno Wincy J – 49621911103

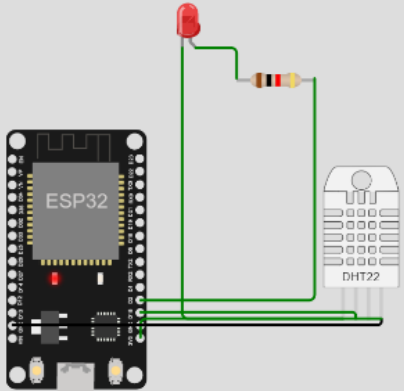
Mohamed Anas.M– 49621911009

Nishar Ahamed T - 4962191109

Wokwi link:<https://wokwi.com/projects/348737733541233235>

Simulation

01:20.097 99%



```
Publish ok
temperature:24.00
Sending payload: {"temperature":24.00}
Publish ok
temperature:24.00
Sending payload: {"temperature":24.00}
Publish ok
```

ENG 11:55 AM

CODING:

```
#include <WiFi.h> //library for wifi
#include <PubSubClient.h> //library for MQTT
#include "DHT.h" // Library for dht11
#define DHTPIN 15 // what pin we're connected to
#define DHTTYPE DHT22 // define type of sensor DHT 11
#define LED 2
DHT dht (DHTPIN, DHTTYPE);
void callback(char* topic, byte* payload, unsigned int
```

```

payloadLength);
//-----credentials of IBM Accounts-----
#define ORG "yzs5sj" //IBM ORGANITION ID
#define DEVICE_TYPE "fire_IoT" //Device type mentioned in ibm watson IOT
Platform
#define DEVICE_ID "17082001" //Device ID mentioned in ibm watson IOT Platform
#define TOKEN "1911089abcdefgh" //Token
String data3;
float t;
//----- Customise the above values -----
char server[] = ORG ".messaging.internetofthings.ibmcloud.com"; // Server
char publishTopic[] = "iot-2/evt/Data/fmt/json"; // topic name and type ofevent
perform and format in which data to be send
char subscribetopic[] = "iot-2/cmd/command/fmt/String"; // cmd REPRESENTcommand
type AND COMMAND IS TEST OF FORMAT STRING
char authMethod[] = "use-token-auth"; // authentication method
char token[] = TOKEN;
char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID; //client id
//-----
WiFiClient wifiClient; // creating the instance for wifi client
PubSubClient client(server, 1883, callback ,wifiClient); //calling
thepredefined client id by passing parameter like server id,portand
wifi credential
void setup() // configureing the ESP32
{
  Serial.begin(115200);
  dht.begin();
  pinMode(LED, OUTPUT);
  delay(10);
  Serial.println();
  wifi connect();
  mqttconnect();
}
void loop() // Recursive Function
{
  t = dht.readTemperature();
  Serial.print("temperature: ");

  Serial.println(t);
  PublishData(t);
  delay(1000);
  if (!client.loop()) {
    mqttconnect();
  }
}

```

```

/*.....retrieving to
Cloud.....*/
void PublishData(float temp) {
  mqttconnect();//function call for connecting to ibm
  /*
  creating the String in in form JSON to update the data to ibm cloud
  */
  String payload = "{\"temperature\": ";
  payload += temp;
  payload += "}";
  Serial.print("Sending payload: ");
  Serial.println(payload);
  if (client.publish(publishTopic, (char*) payload.c_str())) {
    Serial.println("Publish ok");// if it successfully upload data on the cloud
    then it will print publish ok in Serial monitor or else it will print publish
    failed } else {
      Serial.println("Publish failed");
    }
  }
}

void mqttconnect() {
  if (!client.connected()) {
    Serial.print("Reconnecting client to ");
    Serial.println(server);
    while (!client.connect(clientId, authMethod, token)) {
      Serial.print(".");
      delay(500);
    }

    initManagedDevice();
    Serial.println();
  }
}

void wifi connect() //function definition for wifi connect
{
  Serial.println();
  Serial.print("Connecting to ");
  WiFi.begin("Wokwi-GUEST", "", 6);//passing the wifi credentials to establish
  the connection

  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  Serial.println("");
  Serial.println("Wi-Fi connected");
}

```

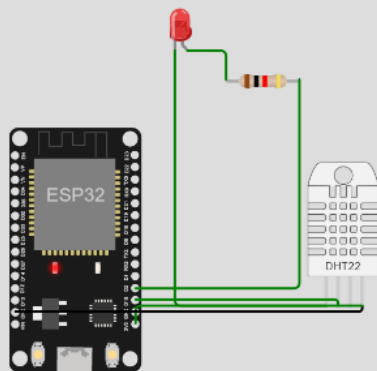
```

    Serial.println("IP address: ");
    Serial.println(WiFi.localIP());
}
void initManagedDevice() {
    if (client.subscribe(subscribetopic)) {
        Serial.println((subscribetopic));
        Serial.println("subscribe to cmd OK");
    } else {
        Serial.println("subscribe to cmd FAILED");
    }
}
void callback(char* subscribetopic, byte* payload, unsigned int payloadLength)
{
    Serial.print("callback invoked for topic: ");
    Serial.println(subscribetopic);
    for (int i = 0; i < payloadLength; i++) {
        //Serial.print((char)payload[i]);
        data3 += (char)payload[i];
    }
    Serial.println("data: " + data3);
    if(data3=="lighton")
    {
        Serial.println(data3);
        digitalWrite(LED, HIGH);
    }
    else
    {
        Serial.println(data3);
        digitalWrite(LED, LOW);
    }
    data3="";
}

```

Simulation

01:20.097 99%



Publish ok

temperature:24.00

Sending payload: {"temperature":24.00}

Publish ok

temperature:24.00

Sending payload: {"temperature":24.00}

Publish ok



Events 1

New event type +

Event type name

event_1

Send

Schedule

20

Every Minute

Payload

Specify the event payload in the editor window or by uploading a [CSV file](#).

```

0 {
1   "Gas": random(0, 100),
2   "Flame": random(0, 1)
3 }

```

Displaying gas sensor & flame sensor values:

Inbox x

IBM C x

Down x

IBM-E x

Docu x

Down x

You x

IBM x

IBM-E x

IBM V x

IBM x

IBM-E x

W sketch x

+

yzs5j.internetofthings.ibmcloud.com/dashboard/devices/browse

Gmail

YouTube

Maps

IBM Watson IoT Platform

1911009@nec.edu.in
ID: yzs5aj

Browse

Action

Device Types

Interfaces

Add Device +

Identity

Device Information

Recent Events

State

Logs

The recent events listed show the live stream of data that is coming and going from this device.

Event	Value	Format	Last Received
Data	{"temperature":24}	json	a few seconds ago
Data	{"temperature":24}	json	a few seconds ago
Data	{"temperature":24}	json	a few seconds ago
Data	{"temperature":24}	json	a few seconds ago
Data	{"temperature":24}	json	a few seconds ago

Items per page 50

1-1 of 1 item

1 of 1 page

<

1

>

Type here to search

30°C Haze

ENG IN

11:55 AM

11/19/2022