

```

#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 BUZZER_PIN 2
int BUZZER_CHANNEL = 0;
DHT dht (DHTPIN, DHTTYPE); // creating the instance by passing pin and typr of dht connected

void callback(char* subscribetopic, byte* payload, unsigned int payloadLength);

//-----credentials of IBM Accounts-----

#define ORG "lryrya" //IBM ORGANITION ID
#define DEVICE_TYPE "Temp" //Device type mentioned in ibm watson IOT Platform
#define DEVICE_ID "T1" //Device ID mentioned in ibm watson IOT Platform
#define TOKEN "123456789" //Token
String data3;
float Temperature;

//----- Customise the above values -----
char server[] = ORG ".messaging.internetofthings.ibmcloud.com"; // Server Name
char publishTopic[] = "iot-2/evt/Data/fmt/json"; // topic name and type of event perform and
format in which data to be send
char subscribetopic[] = "iot-2/cmd/command/fmt/String"; // cmd REPRESENT command 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 wificlient
PubSubClient client(server, 1883, callback ,wifiClient); //calling the predefined client id by
passing parameter like server id,portand wificredential
void setup() // configureing the ESP32
{
  Serial.begin(115200);
  dht.begin();
  pinMode(BUZZER_PIN,OUTPUT);
  ledcAttachPin(BUZZER_PIN, BUZZER_CHANNEL);
  delay(10);
  Serial.println();

```

```
wificonnect();
mqttconnect();
}
```

```
void loop()// Recursive Function
{
```

```
    Temperature = dht.readTemperature();
    Serial.print("temperature:");
    Serial.println(Temperature);
```

```
    PublishData(Temperature);
    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 sucessfully 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 wificonnect() //function defination for wificonnect
{
  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("WiFi 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(BUZZER_PIN,HIGH);  
    ledcWriteNote(BUZZER_CHANNEL, (note_t)NOTE_D, 8);  
}  
  
else  
{  
    Serial.println(data3);  
    digitalWrite(BUZZER_PIN,LOW);  
    ledcWriteNote(BUZZER_CHANNEL, (note_t)NOTE_D, 0);  
}  
data3="";  
  
}
```