Wokwi code to gather data from DHT22 sensor

```
#include <WiFi.h>//library for wifi
#include <PubSubClient.h>//library for MQtt
#include "DHT.h"// Library for dht11
#include <Stepper.h>
#define DHTPIN 15     // what pin we're connected to
#define DHTTYPE DHT22   // define type of sensor DHT 11

#define DIRPIN 4
#define STEPPIN 5
#define DELAY_US 2000

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 "k1o3f1"//IBM ORGANITION ID
#define DEVICE_TYPE "abcd"//Device type mentioned in ibm watson IOT Platform
#define DEVICE_ID "1234"//Device ID mentioned in ibm watson IOT Platform
#define TOKEN "12345678"     //Token
String data3;
float h, t ,s;
;


//-------- 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 ,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(DIRPIN, OUTPUT);
  pinMode(STEPPIN, OUTPUT);
  delay(1000);

  Serial.println();
  wificonnect();
  client.setCallback(callback);

  mqttconnect();
  client.setCallback(callback);
}

void loop()// Recursive Function
{

  h = dht.readHumidity();
  t = dht.readTemperature();
  s=0;
  Serial.print("temp:");
  Serial.println(t);
  Serial.print("Humid:");
  Serial.println(h);

  PublishData(t, h,s);
  delay(1000);
  if (!client.loop()) {
    mqttconnect();
  }
}



/*....................................retrieving to
Cloud...............................*/

void PublishData(float temp, float humid,float status) {
  mqttconnect();//function call for connecting to ibm
  /*
     creating the String in in form JSon to update the data to ibm cloud
  */
  if (temp>50 && humid>60){
   status=1;
  }
  String payload = "{\"temp\":";
  payload += temp;
  payload += "," "\"Humid\":";
  payload += humid;
  payload += "," "\"Status\":";
  payload += status;
  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);
    client.setCallback(callback);
    while (!!!client.connect(clientId, authMethod, token)) {
      Serial.print(".");
      delay(500);
    }
  }
     initManagedDevice();
     client.subscribe("cmnd/command/motoron");
 //    client.subscribe("cmnd/GarageDoor/POWER2");
     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.print(client.subscribe(subscribetopic));
    callback(subscribetopic,0,1);
    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];
  }
  data3 = "motoron";
  Serial.println("data: "+ data3);
  if(data3=="motoron")
  {
Serial.println(data3);
Serial.print("Motor running");
digitalWrite(DIRPIN, HIGH);
for (int i = 0; i < 200; i++) {

    digitalWrite(STEPPIN, HIGH);
    delayMicroseconds(DELAY_US);
    digitalWrite(STEPPIN, LOW);
    delayMicroseconds(DELAY_US);
  }
  }
  else
  {
Serial.println(data3);
digitalWrite(DIRPIN, LOW);
  }
data3="";
}
```