

## Assignment – 4

Assignment Date	25 October 2022
Student Name	Sasikala A
Student Roll Number	142219106089
Maximum Marks	2 Marks

### Question-1:

Write code and connections in wokwi for ultrasonic sensor. Whenever distance is less than 100cms send “alert” to ibm cloud and display in device recent events. Upload document with wokwi share link and images of ibm cloud.

### Solution:

```
#include <WiFi.h> //library for wifi
#include <PubSubClient.h> //library for MQTT

#define ECHO_GPIO 12
#define TRIGGER_GPIO 13
#define MAX_DISTANCE_CM 100 // Maximum of 5 meters
#include "Ultrasonic.h"

Ultrasonic ultrasonic(13, 12);
int distance;

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

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

#define ORG "4t4nj5" //IBM ORGANITION ID
#define DEVICE_TYPE "Resh" //Device type mentioned in ibm watson IOT Platform
#define DEVICE_ID "2229" //Device ID mentioned in ibm watson IOT Platform
#define TOKEN "87654321" //Tokenn
String data3;
float h, t;

//----- 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);
    delay(10);
    Serial.println();
    wificonnect();
    mqttconnect();
}

void loop()// Recursive Function
{

    distance = ultrasonic.read(CM);
    if(distance < 100){
        Serial.print("Distance in CM: ");
        Serial.println(distance);
        PublishData(distance);
        delay(1000);
        if (!client.loop()) {
            mqttconnect();
        }

    }

    delay(1000);

}

/*.....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 = "{\"Alert Distance\":\"";
    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);
    }
    else
    {
        Serial.println(data3);
    }
    data3="";
}

```

WOKWI SAVE SHARE esp32-dht22.ino by urish Docs

esp32-dht22.ino • diagram.json • libraries.txt • Library Manager

```

29 char token[] = TOKEN;
30 char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID;//client id
31
32
33 //-----
34 WiFiClient wificlient; // creating the instance for wificlient
35 PubSubClient client(server, 1883, callback, wificlient); //calling the p
36
37 void setup()// configuring the ESP32
38 {
39   Serial.begin(115200);
40   delay(10);
41   Serial.println();
42   wificlient.connect();
43   mqttconnect();
44 }
45
46 void loop()// Recursive Function
47 {
48
49   distance = ultrasonic.read(CM);
50   if(distance < 100){
51     Serial.print("Distance in CM: ");
52     Serial.println(distance);
53     PublishData(distance);
54     delay(1000);

```

Simulation

00:46.503 101%

Connecting to ..  
WiFi connected  
IP address:  
10.10.0.2  
Reconnecting client to  
4t4nj5.messaging.internetofthings.ibmcloud.com  
.....

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	{"Alert Distance":28}	json	a few seconds ago
Data	{"Alert Distance":28}	json	a few seconds ago
Data	{"Alert Distance":28}	json	a few seconds ago
Data	{"Alert Distance":28}	json	a few seconds ago
Data	{"Alert Distance":28}	json	a few seconds ago

Items per page 50 | 1-2 of 2 items

1 of 1 page

0 Simulations running

**Wokwi share link:**

<https://wokwi.com/projects/322410731508073042>