

## Assignment 4

Assignment Date	01 <sup>st</sup> November 2022
Student Name	B.Mejalin Arno
Roll number	2019504037
Maximum Marks	2 Marks

### Question

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.

### Code

```
#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 "herflx" //IBM ORGANITION ID
#define DEVICE_TYPE "ESP32" //Device type mentioned in ibm watson IOT Platform
#define DEVICE_ID "2019504037" //Device ID mentioned in ibm watson IOT Platform
#define TOKEN "s6maAQdyA6wh(jEpY0" //Token
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="";
}

```

# Simulation output

```
1 #include <WiFi.h> //library for wifi
2 #include <PubSubClient.h> //library for MQTT
3 #define ECHO_GPIO 12
4 #define TRIGGER_GPIO 13
5 #define MAX_DISTANCE_CM 100 // Maximum of 5 meters
6 #include "Ultrasonic.h"
7 Ultrasonic ultrasonic(13, 12);
8 int distance;
9 void callback(char* subscribetopic, byte* payload, unsigned int payloadlength);
10 //-----credentials of IBM Accounts-----
11 #define ORG "herflx" //IBM ORGANITION ID
12 #define DEVICE_TYPE "ESP32" //Device type mentioned in ibm watson IOT Platform
13 #define DEVICE_ID "2019504037" //Device ID mentioned in ibm watson IOT Platform
14 #define TOKEN "s6maAQdyA6wh(JEpV0" //Token
15 String data;
16 float h, t;
17 //----- Customise the above values -----
18 char server[] = ORG ".messaging.internetofthings.ibmcloud.com"; // Server Name
19 char publishTopic[] = "iot-2/evt/data/fmt/json"; // topic name and type of event perform a
20 char subscribetopic[] = "iot-2/cmd/command/fmt/string"; // cmd REPRESENT command type AND
21 char authMethod[] = "use-token-auth"; // authentication method
22 char token[] = TOKEN;
23 char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID; //client id
24 //-----
25 WiFiClient wifiClient; // creating the instance for wifiClient
26 PubSubClient client(server, 1883, callback, wifiClient); //calling the predefined client
27 void setup() // configureing the ESP32
28 {
29   Serial.begin(115200);
30   delay(10);
31   Serial.println();
32   wifiConnect();
33   mqttConnect();
34 }
```

Simulation

Editing Ultrasonic Distance Sensor  
Distance: 28cm

Distance in CM: 29  
Sending payload: {"Alert Distance":29.00}  
Publish ok  
Distance in CM: 29  
Sending payload: {"Alert Distance":29.00}  
Publish ok

[← Back](#)

## Device Drilldown - 2019504037

Connection Information

Recent Events

State

Device Information

Metadata

Diagnostics

Connection Logs

Device Actions

### Recent Events

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":29}	json	a few seconds ago
Data	{"Alert Distance":29}	json	a few seconds ago
Data	{"Alert Distance":29}	json	a few seconds ago
Data	{"Alert Distance":29}	json	a few seconds ago
Data	{"Alert Distance":29}	json	a few seconds ago

### State

## Project link

<https://wokwi.com/projects/347108355090678356>