

Assignment-4

Write code and connections in wokwi for the ultrasonic sensor.

Whenever the distance is less than 100 cms send an "alert" to the IBM cloud and display in

the device recent events.

Upload document with wokwi share link and images of IBM cloud

Program:

```
#include <WiFi.h>

#include <PubSubClient.h>
WiFiClient wifiClient;
String data3;
#define ORG "z7l8rv"
#define DEVICE_TYPE "hari"
#define DEVICE_ID "hari1234"
#define TOKEN "HC8raxG34J)5+FtVFC"
#define speed 0.034
#define led 14
char server[] = ORG ".messaging.internetofthings.ibmcloud.com";
char publishTopic[] = "iot-2/evt/hari/fmt/json";
char topic[] = "iot-2/cmd/home/fmt/String";
char authMethod[] = "use-token-auth";
char token[] = TOKEN;
char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID;
PubSubClient client(server, 1883, wifiClient);
void publishData();

const int trigpin=5;
const int echopin=18;
String command;
String data="";

long duration;
float dist;

void setup()
{
  Serial.begin(115200);
  pinMode(led, OUTPUT);
  pinMode(trigpin, OUTPUT);
  pinMode(echopin, INPUT);
  wifiConnect();
```

```

    mqttConnect();
}

void loop() {
    bool isNearby = dist < 100;
    digitalWrite(led, isNearby);

    publishData();
    delay(500);

    if (!client.loop()) {
        mqttConnect();
    }
}

void wifiConnect() {
    Serial.print("Connecting to "); Serial.print("Wifi");
    WiFi.begin("Wokwi-GUEST", "", 6);
    while (WiFi.status() != WL_CONNECTED) {
        delay(500);
        Serial.print(".");
    }
    Serial.print("WiFi connected, IP address: ");
    Serial.println(WiFi.localIP());
}

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

void initManagedDevice() {
    if (client.subscribe(topic)) {
        // Serial.println(client.subscribe(topic));
        Serial.println("IBM subscribe to cmd OK");
    } else {
        Serial.println("subscribe to cmd FAILED");
    }
}

void publishData()
{
    digitalWrite(trigpin, LOW);
    digitalWrite(trigpin, HIGH);
    delayMicroseconds(10);
    digitalWrite(trigpin, LOW);
}

```

```

duration=pulseIn(echopin,HIGH);
dist=duration*speed/2;
if(dist<100){
    String payload = "{\"Alert Distance\":\"";
    payload += dist;
    payload += "}";

    Serial.print("\n");
    Serial.print("Sending payload: ");
    Serial.println(payload);
    if(client.publish(publishTopic, (char*) payload.c_str())) {
        Serial.println("Warning crosses 110cm -- it automaticaly of the loop");
        digitalWrite(led,HIGH);
    }
}

if(dist>101 && dist<111){
    String payload = "{\"Normal Distance\":\"";
    payload += dist;
    payload += "}";

    Serial.print("\n");
    Serial.print("Sending payload: ");
    Serial.println(payload);

}

}

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++){
        dist += (char)payload[i];
    }
    Serial.println("data:"+ data3);
    if(data3=="lighton"){
        Serial.println(data3);
        digitalWrite(led,HIGH);
    }
    data3="";
}

```

SIMULATION:

The screenshot shows the Wokwi IoT simulator interface. On the left, the 'sketch.ino' file is open, displaying the following code:

```
1 #include <WiFi.h>
2 #include <PubSubClient.h>
3 WiFiClient wifiClient;
4 String data3;
5 #define ORG "z7l8rv"
6 #define DEVICE_TYPE "hari"
7 #define DEVICE_ID "hari1234"
8 #define TOKEN "HC8raxG34J)5+FtVFC"
9 #define speed 0.034
10 #define led 14
11 char server[] = ORG ".messaging.internetofthings.ibmcloud.com";
12 char publishTopic[] = "iot-2/evt/hari/fmt/json";
13 char topic[] = "iot-2/cmd/home/fmt/String";
14 char authMethod[] = "use-token-auth";
15 char token[] = TOKEN;
16 char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID;
17 PubSubClient client(server, 1883, wifiClient);
18 void publishData();
19
20
21 const int trigpin=5;
22 const int echopin=18;
23 String command;
24 String data="";
25
26 long duration;
27 float dist;
```

On the right, the 'Simulation' window shows a visual representation of the ESP32 board and a blue sensor module. Below the simulation, a text area displays the following log output:

```
Connecting to
Wifi.....
.....WiFi connected, IP address:
10.10.0.2
Reconnecting MQTT client to
z7l8rv.messaging.internetofthings.ibmcloud.com
.....
```

MESSAGE IN IBM CLOUD

The screenshot shows the IBM Watson IoT Platform dashboard. The top navigation bar includes 'Browse', 'Action', 'Device Types', and 'Interfaces'. A sidebar on the left contains various icons for navigation. The main content area displays details for a device named 'hari_1', which is 'Connected'. The 'Recent Events' tab is selected, showing a table of events:

Event	Value	Format	Last Received
event_1	{"Alert distance":130}	json	a few seconds ago
event_1	{"Alert distance":110}	json	a few seconds ago
event_1	{"Alert distance":149}	json	a few seconds ago
event_1	{"Alert distance":147}	json	a few seconds ago
event_1	{"Alert distance":146}		

At the bottom of the events table, it indicates '1 Simulation running'.