

VSB Engineering College, Karur-639111
Department of Computer Science And Engineering
Assignment-4

Name: JEGAN M

Question-1:

Write code and connections in wokwi for ultrasonic sensor. Whenever distance is less 100 cms send "alert" to ibm cloud and display in device recent events.

Solution:

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

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

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

#define ORG "za7x6f"//IBM ORGANITION ID
#define DEVICE_TYPE "rj46"//Device type mentioned in ibm watson IOT Platform
#define DEVICE_ID "raj46"//Device ID mentioned in ibm watson IOT Platform
#define TOKEN "R0Q4uhcOCCD0hnom)K"
//Token String data3; float dist;

//----- 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/test/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
```

```
int LED = 4;
```

```
int trig = 5;
```

```
int echo = 18;
```

```
void setup()
```

```
{
```

```
Serial.begin(115200);
```

```
pinMode(trig,OUTPUT);
```

```
pinMode(echo,INPUT);
```

```
pinMode(LED, OUTPUT);
```

```
delay(10); wificonnect();
```

```
mqttconnect();
```

```
}
```

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

```
{
```

```
digitalWrite(trig,LOW);
```

```
digitalWrite(trig,HIGH);
```

```
delayMicroseconds(10);
```

```
digitalWrite(trig,LOW); float
```

```
dur = pulseIn(echo,HIGH); float
```

```
dist = (dur * 0.0343)/2;
```

```
Serial.print ("Distancein cm");
```

```
Serial.println(dist);
```

```
PublishData(dist);
```

```
delay(1000); if
```

```
(!client.loop()) {
```

```
mqttconnect();
```

```

    }
}

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

void PublishData(float dist) {
    mqttconnect();//function call for connecting to ibm

    /*    creating the String in in form JSon to update the data to ibm
    cloud
    */ String
    object; if
    (dist < 100)
    {
        digitalWrite(LED, HIGH);
        Serial.println("object is near");
        object = "Near";
    }
    else
    {
        digitalWrite(LED, LOW);
        Serial.println("no object found");
        object = "No";
    }

    String payload = "{\"distance\":";
    payload += dist; payload += ",";
    payload += "\"object\":\""; payload +=
    object; 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];
    }
    data3="";
}

```

Reference:

<https://wokwi.com/projects/347322163482591827>

