

Assignment-4

Write a code and make connections in wokwi for ultrasonic sensor. When ever distance is less than 100cm send "alert" to ibm cloud and display in device recent events.

Solution:

```
#include <WiFi.h> //library for wifi
#include <PubSubClient.h> //library for MQTT
#define EchoPIN 4 // what pin we're connected to
#define TrigPIN 2
#define LED 5

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

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

#define ORG "8p2x1c" //IBM ORGANITION ID
#define DEVICE_TYPE "ABCDE" //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;

//----- 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
float dist,dur;
String data;
//.....-
```

```
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);
    pinMode(TrigPIN, OUTPUT);
    digitalWrite(TrigPIN, LOW);
    pinMode(EchoPIN, INPUT);
    pinMode(LED,OUTPUT);
    delay(10);
    Serial.println();
    wificonnect();
    mqttconnect();
}
```

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

```
    digitalWrite(TrigPIN, HIGH);
    delayMicroseconds(10);
    digitalWrite(TrigPIN, LOW);
```

```
    dur = pulseIn(EchoPIN,HIGH);
```

```
    dist= dur *0.034 / 2;
```

```
if(dist<100)
```

```
{
    data="alert";
    digitalWrite(LED,HIGH);
```

```
}
```

```
else{
    data="safe";
    digitalWrite(LED,LOW);
```

```
}
```

```
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 payload = "{\"distance\":\"";
    payload += dist;
    payload += "\", \"msg\":\"";
    payload += data;
    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 ");

```

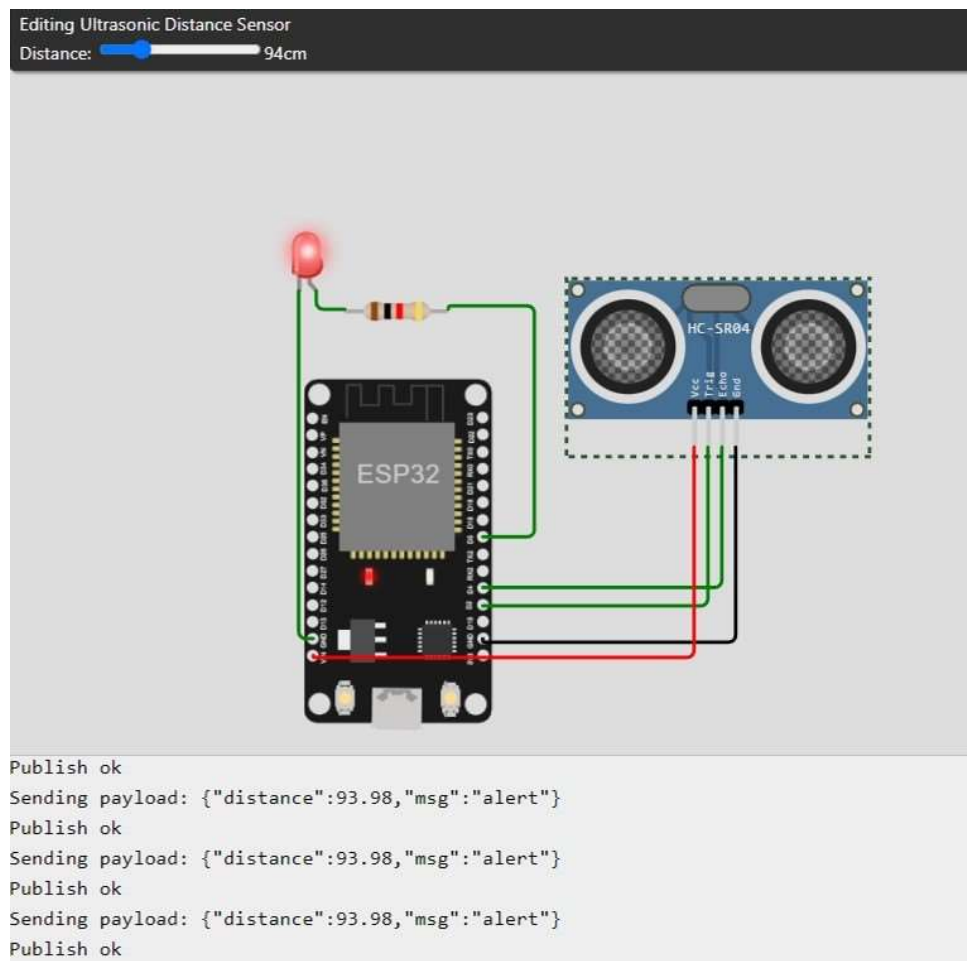
```

    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)
{
}

```



```

1  #include <WiFi.h>//library for wifi
2  #include <PubSubClient.h>//library for MQTT
3  #define EchoPIN 4    // what pin we're connected to
4  #define TrigPIN 2
5  #define LED 5
6  //DHT dht (DHTPIN, DHTTYPE);// creating the instance by passing pin and typr of dht connected
7
8  void callback(char* subscribetopic, byte* payload, unsigned int payloadLength);
9
10 //-----credentials of IBM Accounts-----
11
12 #define ORG "8p2xlc"//IBM ORGANITION ID
13 #define DEVICE_TYPE "ABCDE"//Device type mentioned in ibm watson IOT Platform
14 #define DEVICE_ID "1234"//Device ID mentioned in ibm watson IOT Platform
15 #define TOKEN "12345678"    //Token
16 String data3;
17
18
19
20 //----- Customise the above values -----
21 char server[] = ORG ".messaging.internetofthings.ibmcloud.com";// Server Name
22 char publishTopic[] = "iot-2/evt/Data/fmt/json";// topic name and type of event perform and format in which data
23 char subscribetopic[] = "iot-2/cmd/command/fmt/String";// cmd REPRESENT command type AND COMMAND IS TEST OF FORM
24 char authMethod[] = "use-token-auth";// authentication method
25 char token[] = TOKEN;
26 char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID;//client id
27 float dist,dur;
28 String data;
29 //-----
30 WiFiClient wificlient; // creating the instance for wificlient
31 PubSubClient client(server, 1883, callback ,wificlient); //calling the predefined client id by passing parameter
32
33
34 void setup()// configureing the ESP32
35 {
36     Serial.begin(115200);
37     pinMode(TrigPIN, OUTPUT);
38     digitalWrite(TrigPIN, LOW);
39     pinMode(EchoPIN, INPUT);
40     pinMode(LED,OUTPUT);
41     delay(10);
42     Serial.println();
43     wificlient.begin();

```

WokwiLink: <https://wokwi.com/projects/347662137833816659>

IBM Cloud

Device Recent Events

	Device ID	Status	Device Type	Class ID	Date Added	Descriptive Location
▼ 	1234	 Connected	ABCDE	Device	Nov 7, 2022 6:13 PM	
Identity Device Information <u>Recent Events</u> 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	<code>{"distance":93.99,"msg":"alert"}</code>	json	a few seconds ago
Data	<code>{"distance":93.98,"msg":"alert"}</code>	json	a few seconds ago
Data	<code>{"distance":93.98,"msg":"alert"}</code>	json	a few seconds ago