

ASSIGNMENT-4

WOKWI SIMULATOR AND IBM CLOUD

Assignment Date	7 November 2022
Student Name	Preethi Govindaraj
Student Roll Number	312319106123
Team ID	PNT2022TMID00340
Maximum Marks	2 Marks

QUESTION:

Write code and connections in wokwi for the ultrasonic sensor. Whenever the distance is less than 100 cm 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.

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 0.1 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 "zfc7n"//IBM ORGANITION ID
#define DEVICE_TYPE "ESP32_Controller"//Device type mentioned in ibm watson IOT Platform
#define DEVICE_ID "ibmA-4"//Device ID mentioned in ibm watson IOT Platform
#define TOKEN "GRTl4P*Zvcm4PBiOf" //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 Centimeters: ");

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 SIMULATION:

The screenshot shows the Wokwi web interface. On the left, the Arduino IDE editor displays a sketch named 'IBMAssignment4.ino'. The sketch includes libraries for WiFi and MQTT, defines pins for an ultrasonic sensor, and sets up an MQTT client to publish distance data. The main part of the sketch is a loop that reads the sensor distance and publishes it to a topic. On the right, the 'Simulation' window shows a 3D model of an ESP32 microcontroller connected to an HC-SR04 ultrasonic sensor. Below the model, a console window shows the simulation output, including the distance in centimeters and the MQTT payload being sent.

```

1 //----- IBM ASSIGNMENT-4 PREETHI GOVINDARAJ (312319106123)-----//
2 #include <WiFi.h> //library for wifi
3 #include <PubSubClient.h> //library for MQTT
4
5 #define ECHO_GPIO 12
6 #define TRIGGER_GPIO 13
7 #define MAX_DISTANCE_CM 100 // Maximum of 0.1 meters
8 #include "Ultrasonic.h"
9
10 Ultrasonic ultrasonic(13, 12);
11 int distance;
12
13 void callback(char* topic, byte* payload, unsigned int payloadLength);
14
15 //-----credentials of IBM Accounts-----
16
17 #define ORG "zfc7n" //IBM ORGANIZATION ID
18 #define DEVICE_TYPE "ESP32_Controller" //Device type mentioned in ibm watson IOT Platform
19 #define DEVICE_ID "ibmA-4" //Device ID mentioned in ibm watson IOT Platform
20 #define TOKEN "GRT14P*ZvcM4PB10ft" //Token
21 String data3;
22 float h, t;
23
24 //----- Customise the above values -----
25 char server[] = ORG ".messaging.internetofthings.ibmcloud.com"; // Server Name
26 char publishTopic[] = "iot-2/evt/Data/fmt/json"; // topic name and type of event perform
27 char subscribTopic[] = "iot-2/cmd/command/fmt/String"; // cmd REPRESENT command type AIO
28 char authMethod[] = "use-token-auth"; // authentication method
29 char token[] = TOKEN;
30 char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID; //client id
31
32 //-----
33 WiFiClient wificlient; // creating the instance for wificlient
34 PubSubClient client(server, 1883, callback, wificlient); //calling the predefined client
  
```

Simulation Output:

```

Publish ok
Distance in Centimeters: 84
Sending payload: {"Alert Distance":84.00}
Publish ok
Distance in Centimeters: 35
Sending payload: {"Alert Distance":35.00}
Publish ok
  
```

IBM Watson Platform-Device Event Log:

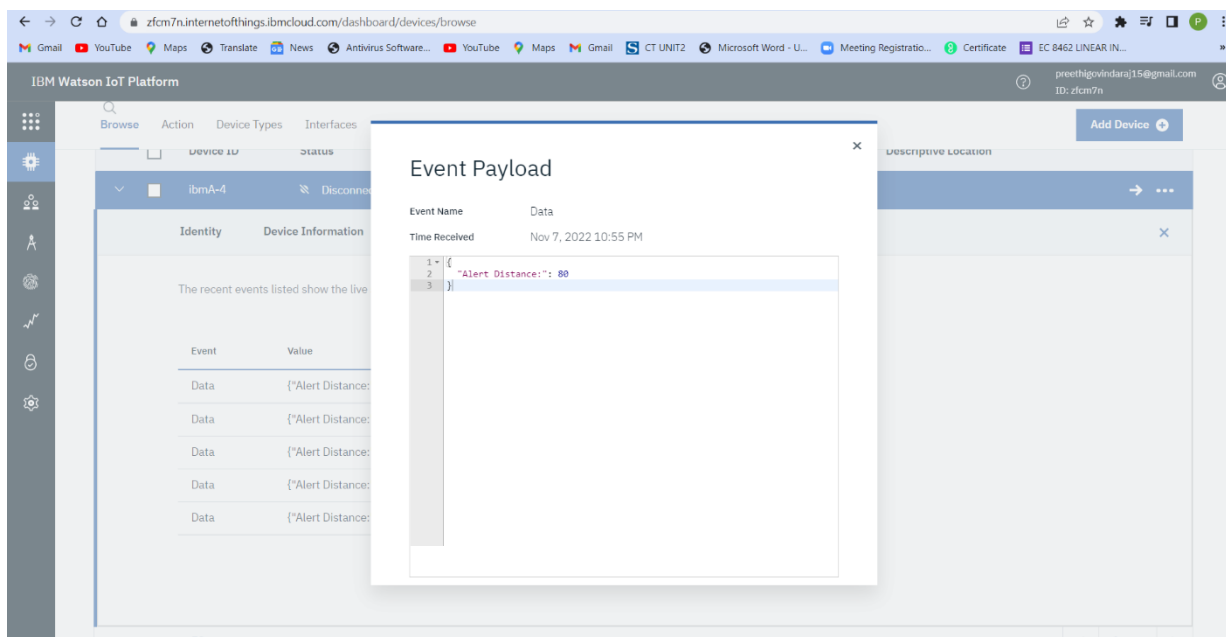
The screenshot shows the IBM Watson IoT Platform dashboard. The 'Browse' tab is selected, displaying a table of devices. The device 'ibmA-4' is highlighted, and its 'Recent Events' are shown in a table below. The events table lists the data received from the device, including the alert distance in centimeters and the time it was received.

Event	Value	Format	Last Received
Data	{"Alert Distance":76}	json	a few seconds ago
Data	{"Alert Distance":80}	json	a few seconds ago
Data	{"Alert Distance":73}	json	a few seconds ago
Data	{"Alert Distance":69}	json	a few seconds ago
Data	{"Alert Distance":68}	json	a few seconds ago

IBM Watson Platform-Board:



IBM Watson Platform-Event Payload:



WOKWI SIMULATION LINK:

<https://wokwi.com/projects/347689848615731795>