

**Name:** Maheshsriram T (737819ITR045)

**Code:**

```
#include <WiFi.h>

#include <WiFiClient.h>

#include <PubSubClient.h>

const char* ssid = "Wokwi-GUEST";

const char* password = "";

#define ORG "z69c1z"

#define DEVICE_TYPE "mahesh"

#define DEVICE_ID "ass4"

#define TOKEN "esp32mahesh"

char server[] = ORG ".messaging.internetofthings.ibmcloud.com";

char authMethod[] = "use-token-auth";

char token[] = TOKEN;

char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID;

char pubTopic1[] = "iot-2/evt/status1/fmt/json";

WiFiClient wifiClient;

PubSubClient client(server, 1883, NULL, wifiClient);

const int trigPin = 13;

const int echoPin = 12;

long lastMsg = 0;

void setup() {

  Serial.begin(115200); // Starts the serial communication

  pinMode(trigPin, OUTPUT); // Sets the trigPin as an Output

  pinMode(echoPin, INPUT); // Sets the echoPin as an Input

  Serial.println();

  Serial.print("Connecting to ");

  Serial.print(ssid);

  WiFi.begin(ssid, password);

  while (WiFi.status() != WL_CONNECTED) {

    delay(500);
```

```

    Serial.print(".");
}

Serial.println("");

Serial.print("WiFi connected, IP address: ");

Serial.println(WiFi.localIP());

if (!client.connected()) {

    Serial.print("Reconnecting client to ");

    Serial.println(server);

    while (!client.connect(clientId, authMethod, token)) {

        Serial.print(".");

        delay(500);

    }

    Serial.println("Bluemix connected");

}

}

void loop() {

    // Clears the trigPin
    digitalWrite(trigPin, LOW);

    delayMicroseconds(2);

    // Sets the trigPin on HIGH state for 10 micro seconds
    digitalWrite(trigPin, HIGH);

    delayMicroseconds(10);

    digitalWrite(trigPin, LOW);

    // Reads the echoPin, returns the sound wave travel time in microseconds
    long duration = pulseIn(echoPin, HIGH);

    // Calculate the distance
    float distanceCm = duration * 0.034/2;

    // Prints the distance in the Serial Monitor
    Serial.print("Distance (cm): ");

    Serial.println(distanceCm);

    delay(3000);
}

```

```

client.loop();

String payload = "{\"d\":{"\"Name\": \""\" DEVICE_ID \""\"";

    payload += ", \"Distance\": ";

    payload += distanceCm;

    payload += "}}";

if (client.publish(pubTopic1, (char*) payload.c_str())) {

    Serial.println("Publish ok");

} else {

    Serial.println("Publish failed");

}

}

```

## Wokwi Output:

The image shows the Wokwi simulation environment. On the left, the code editor displays the Arduino sketch. On the right, the simulation window shows a circuit with an ESP32 microcontroller and an HC-SR04 ultrasonic sensor. The output console at the bottom right shows the serial output of the program.

**Code (esp32-dht22.ino):**

```

1 #include <WiFi.h>
2 #include <WiFiClient.h>
3 #include <PubSubClient.h>
4
5 const char* ssid = "Wokwi-GUEST";
6 const char* password = "";
7
8 #define ORG "d9rc12"
9 #define DEVICE_TYPE "mahesh"
10 #define DEVICE_ID "ast"
11 #define TOKEN "esp2mahesh"
12
13 char server[] = ORG ".messaging.internetofthings.lbccloud.com";
14 char authMethod[] = "use token auth";
15 char token[] = TOKEN;
16 char clientId[] = "d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID;
17 char pubTopic[] = "iot-2/evt/status/inf/json";
18
19 WiFiClient wifiClient;
20 PubSubClient client(server, 1883, NULL, wifiClient);
21
22 const int trigPin = 13;
23 const int echoPin = 12;
24 long lastMsg = 0;
25
26 void setup() {
27   Serial.begin(115200); // Starts the serial communication
28
29   pinMode(trigPin, OUTPUT); // Sets the trigPin as an Output
30   pinMode(echoPin, INPUT); // Sets the echoPin as an Input
31
32   Serial.println();
33   Serial.print("Connecting to ");
34   Serial.print(ssid);
35   WiFi.begin(ssid, password);
36   while (WiFi.status() != WL_CONNECTED) {
37     delay(500);
38     Serial.print(".");
39   }
40   Serial.println("");
41   Serial.print("WiFi connected, IP address: ");
42   Serial.println(WiFi.localIP());
43
44   if (!client.connected()) {
45     Serial.print("Reconnecting client to ");
46     Serial.println(server);

```

**Simulation Output:**

```

Distance (cm): 351.00
Publish ok
Distance (cm): 351.00
Publish ok
Distance (cm): 351.00
Publish ok
Distance (cm): 350.96

```

## IBM output:

The screenshot displays the IBM IoT Platform interface. On the left is a dark sidebar with navigation icons. The main content area has a top navigation bar with 'Browse', 'Action', 'Device Types', and 'Interfaces'. A search bar and an 'Add Device' button are on the right. Below this is a table of devices. The first device, 'ass4', is selected and its details are shown in a modal window. The modal has tabs for 'Identity', 'Device Information', 'Recent Events', 'State', and 'Logs'. The 'Recent Events' tab is active, showing a live stream of data. Below the events is a table with 4 columns: Event, Value, Format, and Last Received. The table contains 5 rows of status updates. At the bottom of the modal, there is a pagination bar showing '1 of 1 page' and a '0 Simulations running' status box.

Device ID	Status	Device Type	Class ID	Date Added	Descriptive Location	Added By	Device Class
ass4	Disconnected	esp32	Device	Nov 19, 2022 10:51 PM		aravintha.19n@kongu.edu	
ass4	Connected	malhash	Device	Nov 19, 2022 11:45 PM		aravintha.19n@kongu.edu	

The recent events listed show the live stream of data that is coming and going from this device.

Event	Value	Format	Last Received
status1	["d":{"Name":"","ass4":{"Distance":350.96}}]	json	a few seconds ago
status1	["d":{"Name":"","ass4":{"Distance":351}}]	json	a few seconds ago
status1	["d":{"Name":"","ass4":{"Distance":351}}]	json	a few seconds ago
status1	["d":{"Name":"","ass4":{"Distance":351}}]	json	a few seconds ago
status1	["d":{"Name":"","ass4":{"Distance":170.93}}]	json	a few seconds ago

Items per page 50 | 1-3 of 3 items

1 of 1 page

0 Simulations running

Wokwi link: <https://wokwi.com/projects/348780939335172692>