



Browse Action Device Types Interfaces

Add Device

---

868292
 Connected
Sandhiyasivakumar
Device
Nov 8, 2022 10:31 AM
 ...

---

Identity Device Information Recent Events State Logs
X

---

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

Event	Value	Format	Last Received
Data	{"Alert!!":399.96}	json	a few seconds ago
Data	{"Alert!!":399.96}	json	a few seconds ago
Data	{"Alert!!":399.92}	json	a few seconds ago

1 Simulation running

url: <https://wokwi.com/projects/new/esp32>

<https://hk4u3i.internetofthings.ibmcloud.com/dashboard/devices/browse>

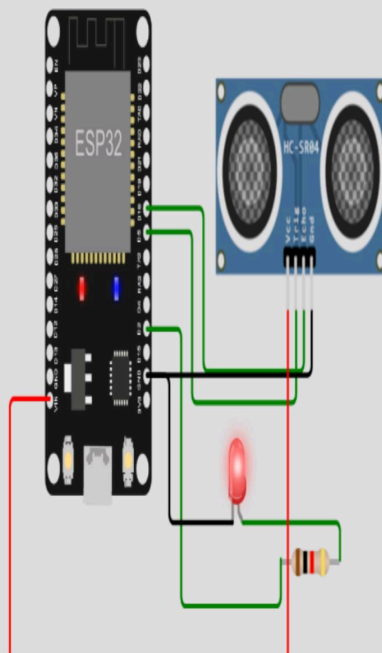
WOKWI SAVE SHARE Docs

sketch.ino diagram.json libraries.txt Library Manager

```
1 #include<WiFi.h> //library for wifi
2 #include<PubSubClient.h> //library for MQTT
3
4
5 void callback(char* subscribetopic, byte* payload, unsigned int payloadLength);
6 // credentials of IBM Accounts
7 #define ORG "hk4u3i"
8
9 #define DEVICE_TYPE "VishaliElango" //Device type mentioned in ibm watson IOT Platform
10 #define DEVICE_ID "1902" //Device ID mentioned in ibm watson IOT Platform
11 #define TOKEN "@0uX1V3f4ffreK47h6" //Token
12 String data3;
13
14 // Customise the above values
15 char server[] = ORG ".messaging.internetofthings.ibmcloud.com"; // Server Name
16 char publishTopic[] = "iot-2/evt/data/fmt/json"; // topic name and type of event
17 char subscribetopic[] = "iot-2/cmd/test/fmt/String"; // cmd REPRESENT command type
18 char authMethod[] = "use-token-auth"; // authentication method
19 char token[] = TOKEN;
20 char clientId[] = "d:"ORG ":"DEVICE_TYPE ":"DEVICE_ID; // client id
21 WiFiClient wificlient; // creating the instance for wificlient
22 PubSubClient client(server, 1883, callback, wificlient); //calling the predefined
23 const int trigpin = 5;
24 const int echopin = 18;
25 const int ledpin = 2;
26
27 long duration ;
28 float distance;
29 #define sound_speed 0.034
30 void setup()
31 {
32   Serial.begin(115200);
33   pinMode(trigpin, OUTPUT);
34   pinMode(echopin, OUTPUT);
35   pinMode(ledpin, OUTPUT);
```

Simulation

03:22:644 99%



Publish ok  
Alert !!  
399.96  
Sending payload : {"Alert !! ": 399.96}  
Publish ok  
Reconnecting client to hk4u3i.messaging.internetofthings.ibmcloud.com  
....

## CODE:

```
#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 "hk4u3i"

#define DEVICE_TYPE "VishaliElango">//Device type mentioned in ibm watson IOT
Platform
#define DEVICE_ID "1902">//Device ID mentioned in ibm watson IOT Platform
#define TOKEN "@OWX1VJf4ffreK47h6">//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
char subscribetopic[] = "iot-2/cmd/test/fmt/String";// cmd REPRESENT command
type
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
const int trigpin = 5;
const int echopin = 18;
const int ledpin = 2;

long duration ;
float distance;
#define sound_speed 0.034
void setup()
{
    Serial.begin(115200);
    pinMode(trigpin, OUTPUT);
    pinMode(echopin, OUTPUT);
    pinMode(ledpin, OUTPUT);
    wificonnect();
    mqttconnect();
}
void loop()
{
    digitalWrite(trigpin, LOW);
```

```

digitalWrite(trigpin, HIGH);
delayMicroseconds(10);
digitalWrite(trigpin, LOW);
duration= pulseIn(echopin,HIGH);
distance = duration * sound_speed /2;
if(distance>=100)
{
PublishData(distance);
delay(1000);
if(!client.loop())
{
mqttconnect();
}
digitalWrite(ledpin, HIGH);
Serial.println("Alert !!");
Serial.println(distance);
}
else
{
digitalWrite(ledpin, LOW);
}
delay(10); // this speeds up the simulation
}
// Retrieving to Cloud
void PublishData(float distance)
{
mqttconnect();// Function call for connecting to ibm
// creating the String in in form JSon to update the data to ibm cloud
String payload = "{\"Alert !! \": ";
payload += distance;
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
}
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
    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="";
}
```