

## Assignment -4

### Python Programming

Assignment Date	31 October 2022
Student Name	Latha B
Student Roll Number	210819106030

#### Question-1:

Write code and connections in wokwi for ultrasonic sensor.

Whenever distance is less than 100 cms send "alert" to ibm cloud and display in device recent events.

Upload document with wokwi share link and images of ibm cloud.

#### Solution:

The screenshot shows the Wokwi IDE interface with the following details:

- Project URL:** wokwi.com/projects/346566226034557521
- Sketch File:** sketch.ino
- Libraries:** WiFi, PubSubClient
- Code Content:**

```
1 #include<WiFi.h> //library for wifi
2 #include<PubSubClient.h> //library for MQTT
3 void callback(char* subscribetopic, byte* payload,unsigned int payloadlength);
4 //-----credentials of IBM Account-----
5 #define ORG "izyybo" // IBM ORGANIZATION ID
6 #define DEVICE_TYPE "iotdeviceproject" //DEVICE TYPE MENTIONED IN IOT WATSON PLATFORM
7 #define DEVICE_ID "129714" //DEVICE ID MENTIONED IN IOT WATSON PLATEFORM
8 #define TOKEN "24681012" //Token
9 String data3;
10 float dist;
11 //----- customize the above value -----
12 char server[]="messaging.internetofthings.ibmcloud.com"; //server name
13 char publishtopic[]="ultrasonic/evt/Data/fmt/json"; //topic name and type of event perform
14 | and format in which data to be send|
15 char subscribetopic[]="ultrasonic/cmd/test/fmt/String"; /*cmd REPRESENT Command type and
16 COMMAND IS TEST OF FORMAT STRING*/
17 char authMethod[]="use-token-auth"; //authentication method
18 char token[] =TOKEN;
19 char clientId[]="d:" ORG ":" DEVICE_TYPE ":" DEVICE_ID //CLIENT ID
20 //-----
21 WiFiClient wifiClient; // creating an instance for wifiClient
22 PubSubClient client(server, 1883 , callback , wifiClient); /*calling the predefined client id
23 by passing parameter like server id,portand wificredential*/
24 int LED =4;
25 int trig =5;
26 int echo=18;
27 void setup()
28 {
29   Serial.begin(115200);
30   pinMode(trig,OUTPUT);
```
- Right Panel:** Shows the "Sims" tab selected, with a green play button icon.

wokwi.com/projects/346566226034557523

WOKWI

sketch.ino

```
61 Serial.println("no object is near");
62 object="Near";
63 }
64 else
65 {
66 digitalWrite(LED,LOW);
67 Serial.println("no object found");
68 object="No";
69 }
70 String payload={"distance"};
71 payload +=dist;
72 payload += " " "object": "";
73 payload += object;
74 payload += "}";
75
76 Serial.print("Sending payload: ");
77 Serial.println(payload);
78 if(client.publish(publishtopic, (char*) payload.c_str())){
79   Serial.println("Publish ok");/* If its successfully upload data on the cloud then it will print
80   publish ok in serial monitor or else it will print publish failed*/
81 } else{
82   Serial.println("Publish failed");
83 }
84 }
85 void mqttconnect(){
86 if(!client.connected()){
87   Serial.print("Reconnecting client to ");
88   Serial.println(server);
89   while(!client.connect(clientId,authMethod, token)){
90     Serial.print(".");
91     delay(500);
92   }
93   initManagedDevice();
94   Serial.println();
95 }
96 }
97 void wificonnect()//function defenition for wificonnect
98 {
99   Serial.println();
100  Serial.print("Connecting to ");
101  WiFi.begin("Wokwi.GUEST", "",6); //PASSING THE WIFI CREDIDENTIALS TO ESTABLISH CONNECTION
102  while (WiFi.status() !=WL_CONNECTED){
103    delay(500);
104    Serial.print(".");
105  }
106  Serial.println("");
107  Serial.println("Wifi connected");
108  Serial.println("IP address");
109  Serial.println(WiFi.localIP());
110 }
111 void initManagedDevice(){
112  if(client.subscribe(subscribetopic)){
113    Serial.println(subscribetopic);
114    Serial.println("subscribe to cmd OK");
115  }else{
116    Serial.println("subscribe to cmd failed");
117  }
118 }
119 void callback(char* subscribetopic, byte* payload, unsigned int payloadLength)
120 {
121  Serial.print("callback invoked for topic: ");
122  Serial.println(subscribetopic);
123 }
```

wokwi.com/projects/346566226034557523

WOKWI

sketch.ino

```
92 }
93 initManagedDevice();
94 Serial.println();
95 }
96 }
97 void wificonnect()//function defenition for wificonnect
98 {
99 Serial.println();
100 Serial.print("Connecting to ");
101 WiFi.begin("Wokwi.GUEST", "",6); //PASSING THE WIFI CREDIDENTIALS TO ESTABLISH CONNECTION
102 while (WiFi.status() !=WL_CONNECTED){
103   delay(500);
104   Serial.print(".");
105 }
106 Serial.println("");
107 Serial.println("Wifi connected");
108 Serial.println("IP address");
109 Serial.println(WiFi.localIP());
110 }
111 void initManagedDevice(){
112 if(client.subscribe(subscribetopic)){
113   Serial.println(subscribetopic);
114   Serial.println("subscribe to cmd OK");
115 }else{
116   Serial.println("subscribe to cmd failed");
117 }
118 }
119 void callback(char* subscribetopic, byte* payload, unsigned int payloadLength)
120 {
121 Serial.print("callback invoked for topic: ");
122 Serial.println(subscribetopic);
123 }
```

The screenshot shows the WOKWI IDE interface. At the top, there are tabs for 'sketch.ino' (selected), 'diagram.json', 'libraries.txt', and 'Library Manager'. Below the tabs is the code editor containing the following Arduino sketch:

```

123 for(int i=0; i< payload.length; i++){
124     //Serial.print((char)payload[1]);
125     data3 +=(char)payload[1];
126 }
127 //Serial.println("data: "+ data3);
128 //if(data3=="Near"){
129 //{
130     //Serial.println(data3);
131     //digitalWrite(LED,HIGH);
132 //}
133 //else
134 //{
135     //Serial.println(data3);
136     //digitalWrite(LED,LOW);
137 //}
138 data3="";
139 }

```

To the right of the code editor is a vertical sidebar with sections like 'Connections', 'Components', 'Testing', and 'Tools'.

## OUTPUT:

DATA IS SENT TO IBM CLOUD WHEN NO OBJECT IS DETECTED

The screenshot shows the IBM Cloud Device view for a device named 'DISTANCEDECTECT'. The device status is 'Disconnected'. The interface includes tabs for 'Identity', 'Device Information', 'Recent Events', 'State', and 'Logs'. The 'Recent Events' tab is selected, displaying the following log entries:

Event	Value	Format	Last Received
Data	[{"distance": 74, "object": "Near"}]	json	a few seconds ago
Data	[{"distance": 70, "object": "Near"}]	json	a few seconds ago
Data	[{"distance": 74, "object": "Near"}]	json	a few seconds ago
Data	[{"distance": 79, "object": "Near"}]	json	a few seconds ago
Data	[{"distance": 70, "object": "Near"}]	json	a few seconds ago

At the bottom of the page, there are pagination controls: 'Items per page: 50' and '1 of 1 page'.

## When no object is detected

The screenshot shows the Wokwi Device Manager interface. At the top, there are tabs for 'Browse', 'Actions', 'Device Types', and 'Interfaces'. On the right, there is a blue 'Add Device' button. Below the tabs, the device name 'DISTANCEDECTECT' is shown, along with its status 'Disconnected', type 'ULTRASONIC', and last update time 'Oct 20, 2022 9:46 AM'. There are also 'Device Information', 'Recent Events', 'State', and 'Logs' tabs. The 'Recent Events' tab is selected, displaying a table of received data. The table has columns for 'Event', 'Value', 'Format', and 'Last Received'. The data entries are:

Event	Value	Format	Last Received
Data	{"distance": "79 cm", "object": "Near"}	json	a few seconds ago
Data	{"distance": "79 cm", "object": "Near"}	json	a few seconds ago
Data	{"distance": "79 cm", "object": "Near"}	json	a few seconds ago
Data	{"distance": "79 cm", "object": "Near"}	json	a few seconds ago
Data	{"distance": "79 cm", "object": "Near"}	json	a few seconds ago

At the bottom of the table, there are pagination controls: 'Items per page: 50' with a dropdown arrow, '1 of 2 items', and '1 of 1 page'.

## When object is detected in ultrasonic detector

The screenshot shows the Wokwi simulation environment. At the top, there is a navigation bar with back, forward, and search icons, followed by the URL 'wokwi.com/projects/346572482591851092'. To the right are 'SAVE', 'SHARE', 'Docs', and a help icon. The main area is divided into 'sketch' and 'Simulation' tabs. The sketch tab shows a circuit diagram with an ESP32 microcontroller connected to an HC-SR04 ultrasonic sensor module. A red LED is also connected to the ESP32. The simulation tab shows a live view of the hardware components. Below the circuit diagram, the serial monitor displays the following code output:

```
object is near
1 Sending payload: {"distance":97.82,"object":"Near"}
1 Publish ok
3 Distance in cm97.82
2 object is near
2 Sending payload: {"distance":97.82,"object":"Near"}
2 Publish ok
2
```