

Assignment -4

Python Programming

Assignment Date	31 October 2022
Student Name	Poornima K
Student Roll Number	210819106050

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:** The code is written in C and includes definitions for IBM account credentials, device type, ID, and token, along with configuration for the ultrasonic sensor and WiFi connection. It uses the PubSubClient library to publish sensor data to an IBM Watson Platform topic.

```
#include<WiFi.h> //library for wifi
#include<PubSubClient.h> //library for MQTT
void callback(char* subscribetopic, byte* payload,unsigned int payloadlength);
//-----credentials of IBM Account-----
#define ORG "izybyo" // IBM ORGANIZATION ID
#define DEVICE_TYPE "iotdeviceproject" //DEVICE TYPE MENTIONED IN IOT WATSON PLATFORM
#define DEVICE_ID "129714" //DEVICE ID MENTIONED IN IOT WATSON PLATEFORM
#define TOKEN "24681012" //Token
String data3;
float dist;
//----- customize the above value -----
char server[]="messaging.internetofthings.ibmcloud.com"; //server name
char publishtopic[]="ultrasonic/evt/Data/fmt/json"; //topic name and type of event perform
// and format in which data to be send/
char subscribetopic[]="ultrasonic/cmd/test/fmt/String"; /*cmd REPRESENT Command type and
COMMAND IS TEST OR 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 an instance for wifiClient
PubSubClient client(server, 1883 , callback , wifiClient); /*calling the predefined client id
by passing parameter like server id,portand wificredential*/
int LED =4;
int trig =5;
int echo=18;
void setup()
{
  Serial.begin(115200);
  pinMode(trig,OUTPUT);
```

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', the component type 'ULTRASONIC', the device ID 'Device', and the date 'Oct 20, 2022 9:46 AM'. There is also a 'Logs' tab and a 'Recent Events' tab which is currently selected. The 'Recent Events' table has columns for 'Event', 'Value', 'Format', and 'Last Received'. The table contains five entries, all of which are 'Data' events with values like 'distance: "94.62", object: "Near"' and 'distance: "79.62", object: "Near"'. The last received time for these events is '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 'V' icon. Below the navigation bar, there are tabs for 'sketch' and 'Simulation'. The 'Simulation' tab is active, showing a circuit diagram. The circuit consists of an ESP32 microcontroller connected to an HC-SR04 ultrasonic sensor module. The sensor module has two black piezoelectric transducers and four pins connected to the ESP32. A red LED is also connected to the ESP32. In the bottom left corner of the simulation window, there is a text area displaying the serial output of the sketch. The output shows the following text:
object is near
1 Sending payload: {"distance":97.82,"object":"Near"}
1 Publish ok
1 Distance in cm97.82
2 object is near
2 Sending payload: {"distance":97.82,"object":"Near"}
2 Publish ok
2