

Assignment -4

Python Programming

Assignment Date	31 October 2022
Student Name	Janani k
Student Roll Number	210819106018

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
- Variables:** ORG, DEVICE_TYPE, DEVICE_ID, TOKEN, data3, dist, server[], publishTopic[], subscribeTopic[], authMethod[], token[], clientId[], wifiClient, LED, trig, echo.
- Code Snippet:** The code includes #include<WiFi.h>, #include<PubSubClient.h>, and various defines and setup functions for connecting to an IBM Watson Platform instance and handling ultrasonic sensor data.

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. The top bar includes navigation icons, a URL (wokwi.com/projects/3465622603455752), and a save/share button. The main area displays the sketch code:

```

sketch.ino • diagram.json libraries.txt • Library Manager
123 for(int i=0; i< payload.length; i++){
124     //Serial.print((char)payload[i]);
125     data3 +=(char)payload[i];
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 }

```

The right side of the interface features a sidebar with tabs like 'Simulator' and 'Logs'. A vertical panel on the far right is titled 'Connecting to...'.

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 top navigation bar includes 'Browse', 'Actions', 'Device Types', 'Interfaces', and an 'Add Device' button. The main area displays the device's status as 'Disconnected' and its last update time as 'Oct 20, 2022 9:46 AM'. Below this, there are tabs for 'Identity', 'Device Information', 'Recent Events', 'State', and 'Logs'. The 'Recent Events' tab is selected, showing 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, 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 are also '→' and '...' buttons. The main area has tabs for 'Identity', 'Device Information', 'Recent Events' (which is currently selected), 'State', and 'Logs'. The 'Recent Events' table has columns for 'Event', 'Value', 'Format', and 'Last Received'. The table contains five rows of data:

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

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

When object is detected in ultrasonic detector

The screenshot shows the Wokwi simulation environment for an ESP32. The top navigation bar includes back, forward, and search icons, a URL field 'wokwi.com/projects/346572482591851092', and a purple 'SAVE' button. On the right, there are 'Docs' and 'V' buttons. The main area is divided into 'sketch' and 'Simulation' tabs. The sketch tab shows a breadboard diagram with an HC-SR04 ultrasonic sensor connected to digital pins D1 and D2, and an LED connected to digital pin D13. The simulation tab shows the physical components: an ESP32 module and an HC-SR04 sensor. Below the components is the code:

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