2018

Lab 4 Event Hub

IOT IT515R CLASS

CJ CORNEL

Contents

I. Objective

Building off of Lab 3, code and create an event hub that connects the LEDs, Distance Sensor, and a Switch (this will simulate the garage door opening and closing).

a. Red – Perfect
b. Yellow – Close
c. Green – Too far
d. Flashing Green and Yellow – Too close
e. Off – Garage door closed
f. On – Garage door open

II. Materials

- Raspberry Pi
- 3 Wemos D1 Mini ESP-8266EX Board
- 1 Distance Sensor HC-SR04
- 1 Reed Switch
- 3 LEDs (Red, Green, and Yellow)
- 3 220 Ω Resisters
- 6 headers
- 10 Male to male wires
- 2 alligator cables

III. Procedure

This procedure is assuming that all previous labs have been completed beforehand.

A. Understanding How it Works

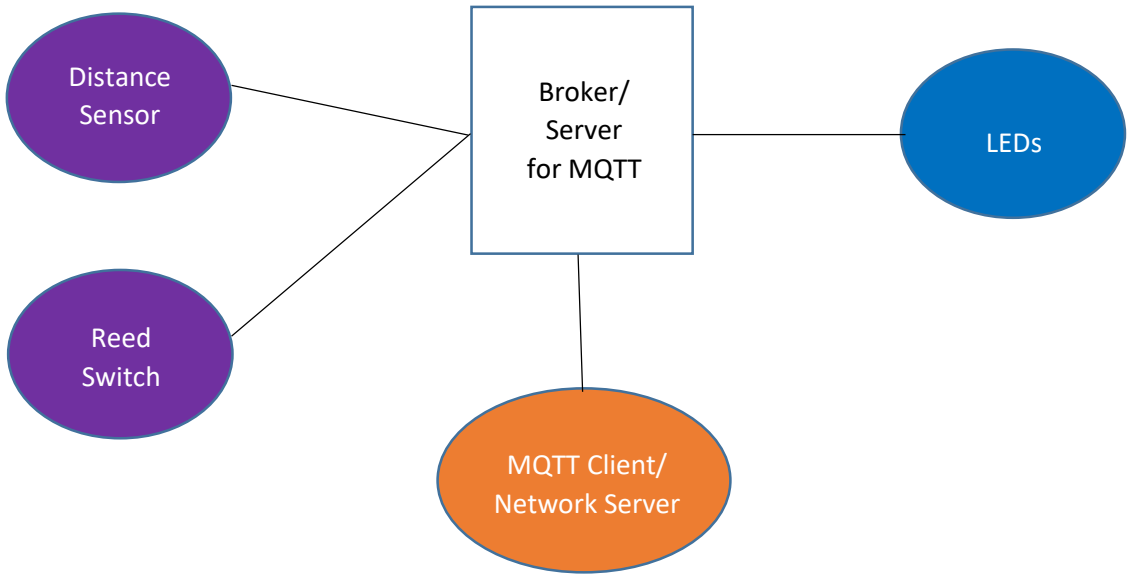

Figure 1: How everything is connected.

1. Distance Sensor – Tracks how far the car is from parking at the correct spot
2. Reed Switch – Acts as a simulation of the garage door opening and closing
3. Broker/Server for MQTT – Accepts client messages and sends them to the specified channel
4. MQTT Client/Network Server – Interprets messages and decides what the other clients will do
5. LEDs – Listens for messages and acts upon them (e.g. light up the red LED)

B. Channels – Subscribers & Publishers

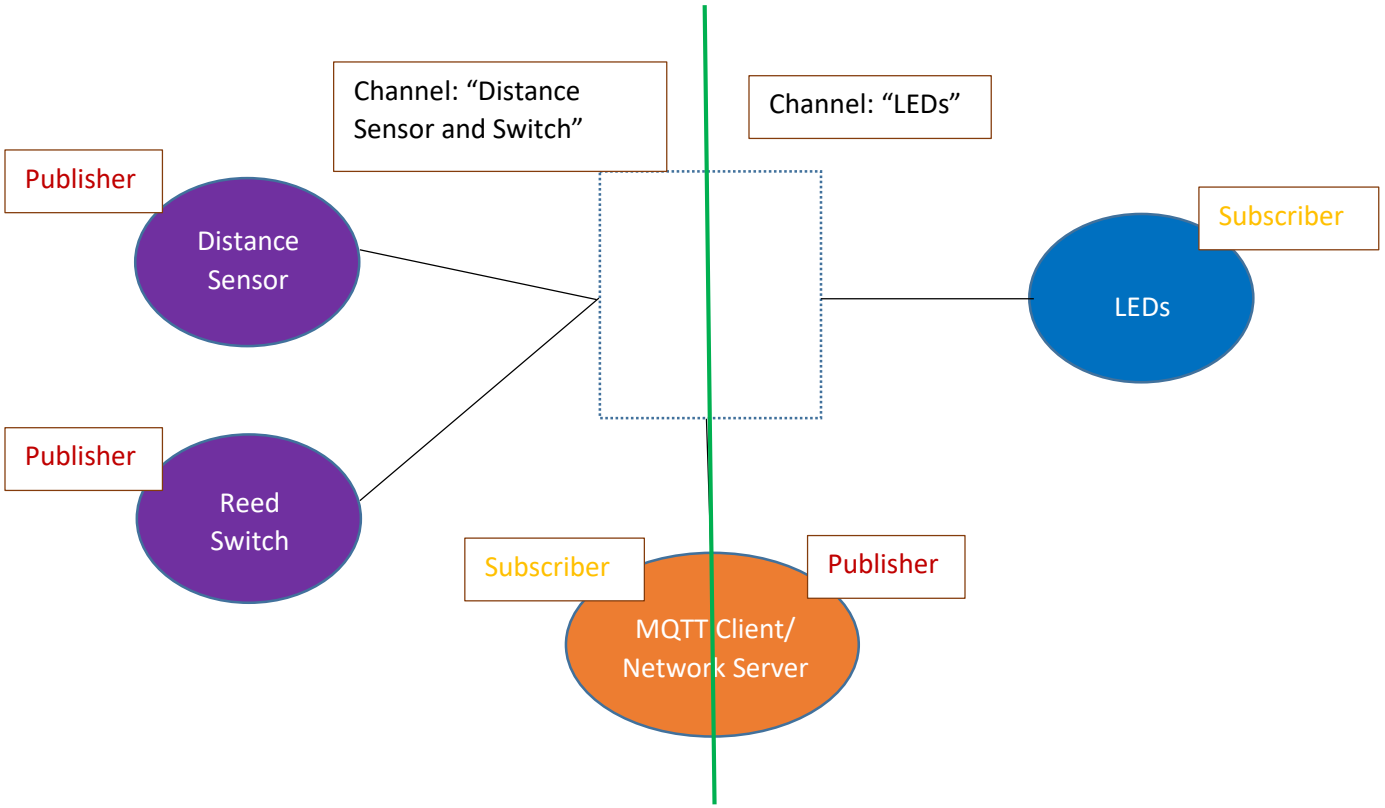

Figure 2: There are two channels being made: “Distance Sensor and Switch” and “LEDs.” Each channel has at least one subscriber and one publisher. MQTT Server does not need to be on any channel.

B. MQTT Broker Client & Server

1. Follow the tutorial (Step 2 only) to create the MQTT Broker using your Raspberry Pi
    a. https://www.instructables.com/id/How-to-Use-MQTT-With-the-Raspberry-Pi-and-ESP8266/
2. Do Step 7 in the before mentioned tutorial to create the MQTT Client (/Network Server)
    a. Ensure that the channels are named as programmed throughout the rest of the clients
    b. Set as a subscriber to the “Distance Sensor and Switch” channel
    c. Set as publisher to the “LEDs” channel

C. Reed Switch

1. Connect a Weemos to a Reed Switch to simulate the garage door opening and closing

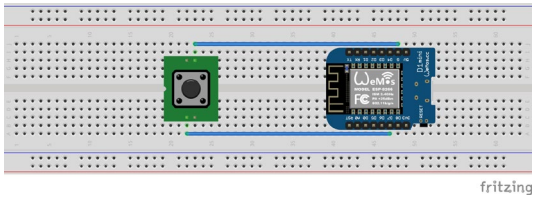

Figure 3: Reed Switch Configuration. The switch pictured is not a reed switch due to unavailability of switch on the program used.

2. Setup the connection with the MQTT server
3. Create a function for what to do when connected to the server and a payload is received
    a. I just had mine print that a message was received
4. Subscribe to the "Distance Sensor and Switch" channel
5. Describe what to do if the Reed switch is connected and not connected and then publish this to the channel.

D. Distance Sensor

1. Setup the connection with the MQTT server
2. Create a function for what to do when connected to the server and a payload is received
    a. I just had mine print that a message was received
3. Subscribe to the "Distance Sensor and Switch" channel
4. Describe what to do at certain distances and then publish it to the channel.

E. LEDs

1. Setup the connection with the MQTT server
2. Create a function for what to do when connected to the server and a payload is received
    a. Read the payload to decipher what command was published to the channel
3. Subscribe to the "LEDs" channel
4. Describe what to do at certain distances and then call the corresponding function
    a. The functions are already defined in this program

IV. Links

- Github: https://github.com/Sh0rtyTheCircuit/MQTT
- Blog: https://ideatethecreate.blogspot.com/2018/11/iot-lab-4-event-hub.html

V. Thought Questions

1. How does the communication between devices change with the shift to using an event hub? How does this facilitate greater scalability?
   Communication is now managed through one connection instead of having need for a website. This facilitates scalability because MQTT has made the process easy to implement new devices. The communication is through a central hub rather than direct connections.
2. What are strengths and weaknesses of the direct communication between the sensor and actuator? What are strengths and weaknesses of the event hub? Which do you feel is better for this application?

   Direct Communication
   Strengths: Fast communication
   Weaknesses: Multiple connections needed (depending on how many devices there are)

   Event Hub
   Strengths: Central commander (MQTT) – every device talks to it, scalable.

Weaknesses: Requires more devices, takes a long time to setup (compared to Direct communication)

I think that the Event hub is better because this guy sounds like he wants to add more devices and changes in the future (I'm assuming this because there are more labs).

3. What was the biggest challenge you overcame in this lab?
   Troubleshooting. Could not figure out why my code was not working until it was found that two of my clients were named the same thing. As a result, one would be disconnected when the other came on.

4. Please estimate the total time you spent on this lab and report.
   11

VI. Sources

- https://www.instructables.com/id/How-to-Use-MQTT-With-the-Raspberry-Pi-and-ESP8266/
- http://fritzing.org/home/
- Andrew Thomas, friend, M.S. candidate

VII. Appendix A

A. MQTT_Client.py

```
#import os
import paho.mqtt.client as mqtt
import time

#Channel Topic
sensors = "Distance Sensor and Reed Switch"
LED="LED"

#ip of localhost
mqtt_broker="192.168.43.40"
mqtt_port = 1883

#Set up MQTT Client object (access to MQTT functions in the library)
```

```
#client = mqtt.Client()
#print "MQTT client object is set up"

#Define functions

def msg_rcv(sensors, user_data, msg):  #Interpret Msgs (Loops)
    print "Payload is " + str(msg.payload)
    if (str(msg.payload) == "on"):
        LED_color = "on"
    elif (str(msg.payload) == "off"):
        LED_color = "off"
    elif (str(msg.payload) == "red"): # or str(msg.payload) == "green" or str(msg.payload) == "yellow"):
        LED_color = "red"
#         LED_color = str(msg.payload)
    elif (str(msg.payload) == "yellow"):
        LED_color = "yellow"
    else:
        LED_color = "green"
    sensors.publish (LED, payload = LED_color, qos=0, retain=False) #(channel, msg to publish)
    print (LED_color)

def run_broker(client, user_data, flags, rc):              #Subscribe to topics (Once)
    print "In the broker function"
    client.subscribe(sensors)              #Listen to the Sensors channel
    print "Subscribed to "
    print (sensors)

client = mqtt.Client()
```

```
#When message is received, run msg_rcv function
client.on_message = msg_rcv

#When connected to Broker, run run_broker function
client.on_connect = run_broker

#Begin connection to MQTT Broker
client.connect(mqtt_broker,mqtt_port)
print "connection to broker started"

#while not client.connect(mqtt_broker,mqtt_port):
#     print "Finding a connection"
#     time.sleep(.5)

#Predefined functions
client.loop_forever()  #Client will keep itself alive
client.disconnect()    #Disconnect before dying (cntrl C or kill)
```

B. LEDs.ino

```
#include <PubSubClient.h>
#include <ESP8266WiFi.h>
#include <ESP8266mDNS.h>
#include <WiFiClient.h>



//#### LED Pin Setup ####//
int GREEN=D6;
int YELLOW=D7;
```

```
int RED=D6;

//###Sensor Variables sent ###//
char* LED_pwr = "On";
char* LED_color = "Green";

//#####MQTT Server connection Setup - Raspberry Pi Broker #####//
char* mqtt_server = "192.168.43.40";
int mqtt_port =1883;

WiFiClient Wifi;        //Setup Wifi object
PubSubClient client(Wifi); //Object that gives you all the MQTT functionality, access objects in PubSubClient Library

//######Wifi Connection Setup #####//
char WifiName[] = "Verizon-SM-G935V";        //SSID
char Password[] = "password";

//#####LED Functions Setup #####//
void TurnGREEN(){
 AllClear();
 digitalWrite(GREEN,HIGH);
 Serial.println("GREEN");
}

void Flash(){
 digitalWrite(GREEN,HIGH);
 delay(200);
 digitalWrite(GREEN,LOW);
 delay(200);
```

```
 digitalWrite(GREEN,HIGH);
 delay(200);
 digitalWrite(GREEN,LOW);
 delay(200);
 digitalWrite(GREEN,HIGH);
 delay(200);
 digitalWrite(GREEN,LOW);
 delay(200);
}

void TurnYELLOW(){
 //TurnOFF();
 AllClear();
 digitalWrite(YELLOW,HIGH);
 Serial.println("YELLOW");
}

void TurnRED(){
 AllClear();
 digitalWrite(RED,HIGH);
 Serial.println("RED");
}

void AllClear(){
 digitalWrite(GREEN,LOW);
 digitalWrite(YELLOW,LOW);
 digitalWrite(RED,LOW);
}
```

```
// ### Look through payload (Message Sent) for LED instructions ### //

void Msg_rcv(char* topic, byte* payload, unsigned int length){    //Unsigned int = Positive numbers (more range)
 Serial.println((char)payload[1]);
 if ((char) payload[0] == 'o'){
  if ((char) payload[1] == 'n'){
   LED_pwr = "On";
  }
  else{
   LED_pwr = "Off";
   AllClear();
  }
 }
 else if ((char) payload[0] == 'g' && LED_pwr == "On"){
  //LED_color = "Green";
  TurnGREEN();
 }
 else if ((char) payload[0] == 'y' && LED_pwr == "On"){
  //LED_color = "Yellow";
  TurnYELLOW();
 }
 else if ((char) payload[0] == 'r' && LED_pwr == "On"){
  LED_color = "Red";
  TurnRED();
 }
 //Turn_color();
}

// ### Call the functions according to the payload and LED power being on or off ### //
```

```
void Turn_color(){
 if (LED_pwr == "On"){
  if (LED_color == "Green"){
   TurnGREEN();
  }
  else if (LED_color == "Red"){
   TurnRED();
  }
  else if (LED_color == "Yellow"){
   TurnYELLOW();
  }
  Serial.print (LED_color);
  Serial.println (" is on");
  }
  else{
  AllClear();
  Serial.println ("Power is off");
  }
}

void setup() {
 // put your setup code here, to run once:
 pinMode(GREEN,OUTPUT);
 pinMode(YELLOW,OUTPUT);
 pinMode(RED,OUTPUT);
 Serial.begin(9600);              //Starts the Serial Monitor (Input printed on screen)

 client.setServer(mqtt_server, mqtt_port);
```

```
 client.setCallback(Msg_rcv);              //Send payload to function (Msg_rcv)

 WiFi.begin(WifiName,Password);
 while (WiFi.status() !=WL_CONNECTED){      //If not connected to Wifi, delay until connected
  delay (2000);
  Serial.println("Finding a Connection...");
 }

 Serial.println("Connection Started");      //Begin Connection to Wifi
 Serial.print("IP Address: ");
 Serial.println(WiFi.localIP());           //IP assigned to Server by host wifi

 while(!client.connect("LED_board")){       //LED_board is name of Wemos/arduino connected to code. Waiting to connect to Broker.
  Serial.println("Finding a Connection...");
 }
 client.subscribe("LED");
 Serial.println("Subscribed to LED");
}

void loop() {                    // put your main code here, to run repeatedly:
 client.loop();                  // pre-made function. Connection to MQTT run continuosly (Constantly listening)
 if(!client.connected()){
  client.connect("LED_board");
 }
 //Turn_color();
}

///## SOURCES ##///
//https://www.instructables.com/id/How-to-Use-MQTT-With-the-Raspberry-Pi-and-ESP8266/
```

```
//Andrew Thomas, friend, M.S. candidate
```

C. MotionSensor.ino

```
#include <ESP8266WiFi.h>
#include <ESP8266mDNS.h>
#include <ESP8266HTTPClient.h>
#include <WiFiClient.h>
#include <PubSubClient.h>


const int Ping = D7; //trigger - sends sound out to measure distance
const int Receive = D5; //Echo - Receives sound from trigger to measure travel time (microsec)
long TimeTravel;  //2 bit integer
int distance;    //1 bit integer


//##### MQTT Server connection Setup - Raspberry Pi Broker #####//
char* mqtt_server = "192.168.43.40";
int mqtt_port = 1883;
char* topic = "Distance Sensor and Reed Switch";

WiFiClient Wifi;        //Setup Wifi object
PubSubClient client(Wifi);  //Object that gives you all the MQTT functionality, access objects in PubSubClient Library


//###### Wifi Connection Setup #####//
char WifiName[] = "Verizon-SM-G935V";        //SSID
char Password[] = "password";
```

```
void Msg_rcv(char* topic, byte* payload, unsigned int length){   //Unsigned int = Positive numbers (more range)
 Serial.println ("Message Received");
}

void setup() {
 // put your setup code here, to run once:
 pinMode (Ping, OUTPUT);
 pinMode(Receive, INPUT);
 Serial.begin(9600);     //Begin serial monitor

 client.setServer(mqtt_server, mqtt_port);
 client.setCallback(Msg_rcv);              //Send payload to function (Msg_rcv)

 // ### Begin Connection to Wifi ### //
 WiFi.begin(WifiName,Password);
 while (WiFi.status() != WL_CONNECTED){      //If not connected to Wifi, delay until connected
  delay (2000);
  Serial.println("Finding a Connection...");
 }

 Serial.println("Connection Started");      //Begin Connection to Wifi
 Serial.print("IP Address: ");
 Serial.println(WiFi.localIP());          //IP assigned to Server by host wifi

 while(!client.connect("Motion_Sensor")){      //LED_board is name of Wemos/arduino connected to code. Waiting to connect to
Broker.
  Serial.println("Finding a Connection...");
 }
```

```
 //client.subscribe(topic);
 Serial.println(topic);
}

void loop() {
 // put your main code here, to run repeatedly:
 client.loop();

 if(!client.connected()){
  client.connect("Motion_Sensor");
  client.publish(topic, "Reconnected");
 }

 // ### Activate Sensor ### //

 digitalWrite(Ping,LOW);   //Clear
 delayMicroseconds(2);
 digitalWrite(Ping,HIGH);   //Send out sound
 delayMicroseconds(10);
 digitalWrite(Ping,LOW);    //Clear again

 // ### Convert time to cm ### //

 TimeTravel = pulseIn (Receive,HIGH);     //Get the input of Receive
 distance = (TimeTravel*0.034)/2;       //converts time to cm. If want a more precise reading, do an average of 3 readings.
 if (distance > 0){                 // in cm
  if (distance > 10){
   Serial.println("Green – Too far");
   client.publish(topic,"green"); //Server is 192.168.43.177. Setup GET request to send to this
```

```
    delay(100);
  }
  else if (distance<10 && distance>5){
    Serial.println("Yellow - Getting Close");
    client.publish(topic,"yellow");
    delay(100);
  }
  else if (distance==5) {
    Serial.println ("Red - Perfect");
    client.publish(topic,"red");
    delay(100);
  }
  else if (distance<5){
    Serial.println ("Flash Green and Yellow - Too close");
    client.publish(topic,"green");
    delay(300);
    client.publish(topic,"yellow");
    delay(300);
    //client.publish(topic,"green");
    //delay(100);
  }
 }
 //delay(500);
}

/// ## SOURCES ## ///
//https://www.instructables.com/id/How-to-Use-MQTT-With-the-Raspberry-Pi-and-ESP8266/
//Andrew Thomas, friend, M.S. candidate
```

D. Switch.ino

```
#include <ESP8266WiFi.h>
#include <PubSubClient.h>

//#### LED Pin Setup ####//
int Switch = D7;
int State = 0;

//#### MQTT Server connection Setup – Raspberry Pi Broker ####//
char* mqtt_server = "192.168.43.40";
int mqtt_port = 1883;
char* topic = "Distance Sensor and Reed Switch";

WiFiClient Wifi;        //Setup Wifi object
PubSubClient client(Wifi); //Object that gives you all the MQTT functionality, access objects in PubSubClient Library

//###### Wifi Connection Setup ####//
char WifiName[] = "Verizon-SM-G935V";        //SSID
char Password[] = "password";

void Msg_rcv(char* topic, byte* payload, unsigned int length){    //Unsigned int = Positive numbers (more range)
 Serial.println ("Message Received");
}

void setup() {
 // put your setup code here, to run once:
 pinMode (Switch, INPUT_PULLUP);          //INPUT_PULLUP, Pin is high (3.3V) until low
 Serial.begin(9600);     //Begin serial monitor
```

```
 client.setServer(mqtt_server, mqtt_port);
 client.setCallback(Msg_rcv);              //Send payload to function (Msg_rcv)

 // ### Begin Connection to Wifi ### //
 WiFi.begin(WifiName,Password);
 while (WiFi.status() != WL_CONNECTED){      //If not connected to Wifi, delay until connected
  delay (2000);
  Serial.println("Finding a Connection...");
 }

 Serial.println("Connection Started");      //Begin Connection to Wifi
 Serial.print("IP Address: ");
 Serial.println(WiFi.localIP());           //IP assigned to Server by host wifi

 while(!client.connect("Switch")){      //LED_board is name of Wemos/arduino connected to code. Waiting to connect to Broker.
  Serial.println("Finding a Connection...");
 }
 client.subscribe(topic);
 Serial.println(topic);
}

void loop() {
 // put your main code here, to run repeatedly:
 client.loop();
 State = digitalRead(Switch);
 if (State == 1){
  client.publish(topic,"on");
 }
```

```
  else if (State == 0){
    client.publish(topic,"off");
  }
  Serial.print(topic);
  Serial.print(" ");
  Serial.println(State);
  delay (100);
}
```