|  |  |
| --- | --- |
|  | De La Salle University  Computer Technology Department  LBY-IOT Laboratory Activity 4  My First IoT Node using Message Queues |

# Introduction

In this laboratory activity, the objective is to improve your understanding with the use of Message Queues, as well as to become familiar with the using them on the electronics platforms themselves, thus communication through the network become vital, no longer through the serial connection. This activity explores the use of the publish/subscribe architecture on the electronics platform, as well as to understand how to use them. For the purposes of this laboratory activity, a Gizduino IoT-644 is advisable, or an Arduino board with an Ethernet Shield may be used. This activity requires the use of Ethernet connectivity.

# Equipment and Software Needed

# Hardware

|  |  |  |
| --- | --- | --- |
| **Quantity** | **Equipment** | **Description** |
| 4pcs | Light-Emitting Diode (LED) | A semiconductor that emits light as current flows through it |
| 4pcs | 220 ohm Resistor | Limits the current through the LED |
| 1pc | DH22 Temperature Sensor | Digital temperature and humidity sensor that using a capacitive humidity sensor and thermistor to measure its surrounding air |
| 1unit | Gizduino IoT-644 development board | Arduino clone with built-in Ethernet port and SD Card slot |
| 1pc | USB-A to USB Mini-B cable | USB cable for serial communication |
| 1pc | Straight LAN Cable | A network cable that has similar termination arrangement on both RJ-45 connectors. The cable used is usually an unshielded twisted pair cable, commonly used for computer networks. |

# Software

For the following activity, the libraries have been uploaded to Animo Space, indicated as the [Laboratory Activity 4 Libraries.zip](https://dlsu.instructure.com/files/399949/download?download_frd=1). Download the compressed file and extract the libraries at the “libraries” folder inside the Arduino IDE home folder. Once copied, re-open your Arduino IDE in order to load the libraries. Be sure to use the admin account as much as possible, as well as turn off the Windows Firewall to ensure that there is no problem in the network connectivity.

|  |  |
| --- | --- |
| **Software** | **Source** |
| Gizduino IoT-644 Ethernet and SD Library | [http://www.e-gizmo.net/oc/kits%20documents/ Gizduino%20IOT-644/libraries.zip](http://www.e-gizmo.net/oc/kits%20documents/Gizduino%20IOT-644/libraries.zip) |
| Arduino PubSubClient Library | <https://github.com/knolleary/pubsubclient> |

# Procedure - Arduino

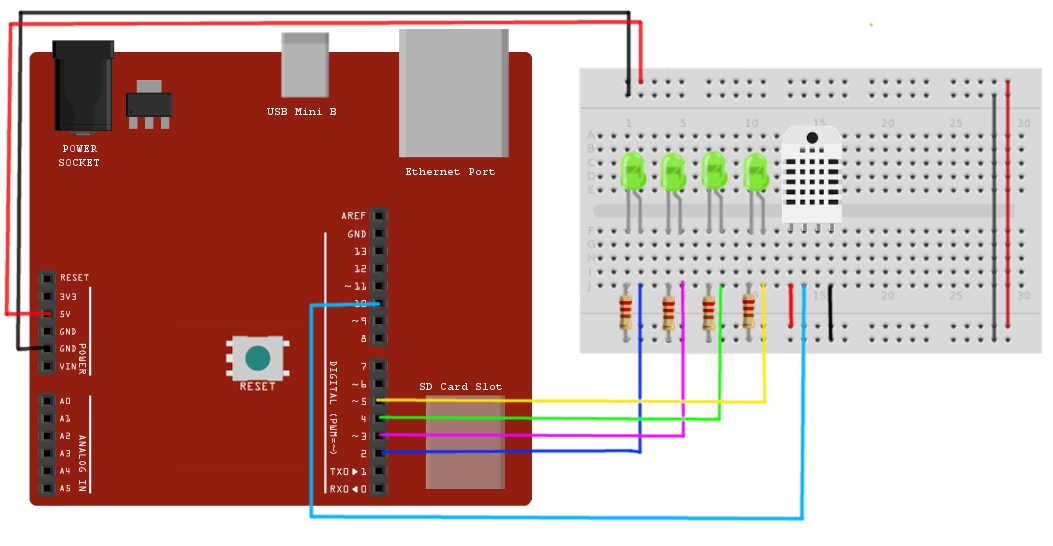


Figure 1. Laboratory Activity 4 Setup

For this activity, since your electronics platform would now be connected via Ethernet connection, remember to plug your electronics platform to the network. If you need a network or LAN cable, please ask the lab technicians.

For the following section of the laboratory activity, try to run the following programs and write the output as well as your observations on the corresponding section.

|  |
| --- |
| **Schematic 4-1** |
| #include <SPI.h>  #include <Ethernet.h>  #include <PubSubClient.h>  // MAC Address of the Gizduino-IOT, replace the '?' with your group number  byte mac[] = {0xDE,0xED,0xBA,0xFE,0xFE,**0x0?**};  // IP Address of the Gizduino-IOT, replace the '?' with your group number  IPAddress ip(10, 200, 180, **10?**);    // IP Address of your Broker Computer, replace the '??' with the last octet  IPAddress server(10, 200, 180, **??**);  void callback(char\* topic, byte\* payload, unsigned int length) {  Serial.print("Message arrived [");  Serial.print(topic);  Serial.print("] ");  for (int nCtr=0; nCtr < length; nCtr++) {  Serial.print((char)payload[nCtr]);  }  Serial.println();  }    EthernetClient ethClient;  PubSubClient client(ethClient);    void reconnect() {  // Loop until connection is completed  while (!client.connected()) {  Serial.print("Attempting MQTT connection...");  if (client.connect("arduinoClient")) {  Serial.println("Connected");  client.publish("outTopic", "Hello world!");  client.subscribe("inTopic");  } else {  Serial.print("Failed, rc=");  Serial.print(client.state());  Serial.println(" try again in 3 seconds");  delay(3000);  }  }  }  void setup(){  Serial.begin(9600);  client.setClient(ethClient);  client.setServer(server, 1883); // 1883 is the default port of MQTT  client.setCallback(callback);    Ethernet.begin(mac, ip);  delay(2000);  }  void loop(){  if(!client.connected()){  reconnect();  }  client.loop();  } |
| Open the serial monitor of your electronics platform. If the connection continuously fails, check if the network cable of your electronics platform is connected properly. Wait until the connection is successful, then indicate what the output was as well as any difficulties you have experienced in the space below. | |
|  | |
| When we opened the serial monitor, the connection kept notifying that it was failing outputting “Failed, rc= 2 try again in 3 seconds”. Then after we ran mosquitto and turned off the firewall, it connected with the output of “Connected”. | |

|  |
| --- |
| Using your terminal or command prompt, run a mosquitto\_sub program, subscribing to the topic **"outTopic"** then reset the electronics platform. Wait until the electronics platform prints the **"Connected"** message on the serial monitor (which would mean that the connection was successful), then observe the output of the terminal or command prompt. What do you think happened and why? |
|  |
| After it connected and we ran the mosquitto\_sub program subscribing to outTopic, the output of “Hello world!” came out. We observed that the code of “client.publish("outTopic", "Hello world!");” published “Hello world!” once it got connected subscribing to outTopic. |

|  |
| --- |
| While the electronics platform and the terminal from the previous question is running, run Schematic 4-2 then reset the electronics platform once again, and observe the output. |

|  |  |
| --- | --- |
| **Schematic 4-2: PYTHON** | **Observation** |
| import paho.mqtt.client as mqtt    def on\_connect(client, userdata, rc):  print ("Connected with code"+ str(rc))  def on\_message(client, userdata, msg):  print ("Topic:", str(msg.topic))  print ("Message:", str(msg.payload))  client = mqtt.Client()  client.on\_connect = on\_connect  client.on\_message = on\_message  client.connect ("**Broker IP**",1883,60)  client.subscribe("**outTopic**")  client.loop\_forever()  print("Running...") | Topic: outTopic  Message: b’Hello world!’ |

|  |
| --- |
| Was the both the terminal and that of Schematic 4-2 able to receive the message from the electronics platform? Why do you think so? What do you think happens if more devices are subscribed to the same topic? |
| Yes it was, it was because once it connected, the message from the code showed “Hello World”. Once more devices are subscribed, we assume that it will output the message as much as how many devices there are. |

|  |
| --- |
| This time using your terminal or command prompt, run a mosquitto\_pub program, publishing to the topic **"inTopic"** a message of **"Hi!"** to the same broker. Assuming that the electronics platform is still running, observe the output of the serial monitor. What was the result? Were you able to see the message **"Hi!"** in the serial monitor? Which function was responsible for handling the messages sent to the electronics platform? |
|  |
| Once connected, the serial monitor displayed “Message arrived (inTopic) Hi!”. The function of client.subscribe(“inTopic”) got the message then the callback function displayed the message. |
|  |
| **Schematic 4-3: Arduino** |
| #include <SPI.h>  #include <Ethernet.h>  #include <PubSubClient.h>  #include <DHT.h>    #define DHTPIN 10  #define DHTTYPE DHT22  DHT dht(DHTPIN, DHTTYPE);    // MAC Address of the Gizduino-IOT, replace the '?' with your group number  byte mac[] = {0xDE,0xED,0xBA,0xFE,0xFE,**0x0?**};  // IP Address of your Broker Computer, replace the '??' with the last octet  IPAddress server(10, 200, 180, **??**);  // MQTT topic on the broker  const char mqtt\_topic[] = "/sensor/temp";  float ftemp = 0;  String stemp = "";  char ctemp[10];    EthernetClient ethClient;  PubSubClient client(ethClient);  void reconnect() {  // Loop until connection is completed  while (!client.connected()) {  Serial.print("Attempting MQTT connection...");  if (client.connect("arduinoClient")) {  Serial.println("Connected");  client.publish("outTopic", "Hello world!");  client.subscribe("inTopic");  } else {  Serial.print("Failed, rc=");  Serial.print(client.state());  Serial.println(" try again in 3 seconds");  delay(3000);  }  }  }  void print\_local\_IP(){  Serial.print("My IP address: ");  for (byte nCtr = 0; nCtr < 4; nCtr++){  Serial.print(Ethernet.localIP()[nCtr], DEC);  Serial.print(".");  }  Serial.println();  }  float read\_DHT\_temp(){  float t = dht.readTemperature();  if (isnan(t)){  t = 0;  }  return (t);  }    String float\_to\_string(float x){  float y;  String s;  y = fmod(x,1);  y = y \* 100;  s = String(int(x)) + "." + String(int(y));  return (s);  }  void setup(){  Serial.begin(9600);  client.setServer(server, 1883);  Ethernet.begin(mac);  delay(2000);  print\_local\_IP();  }  void loop(){  if(!client.connected()){  reconnect();  }  ftemp = read\_DHT\_temp();  stemp = float\_to\_string(ftemp);  Serial.println(stemp);  stemp.toCharArray(ctemp, stemp.length() + 1);  client.publish(mqtt\_topic, ctemp);  delay(2000);  } |

|  |
| --- |
| Open the serial monitor of your electronics platform. If the connection continuously fails, check if the network cable of your electronics platform is connected properly. Wait until the connection is successful, then indicate what the output of the serial monitor is, as well as any difficulties you have experienced in the space below. |
|  |
| It displayed “My IP Address: 10.200.180.10 Attempting MQTT connection…Connected 0.0”. |

|  |
| --- |
| How do you think was the network configuration different from Schematic 4-1, compared to that of Schematic 4-3? Which one was statically configured and which one was dynamically configured? |
| Schematic 4-1 was statically configured while the Schematic 4-3 was dynamically configured. |

|  |
| --- |
| What do you think is the advantage or disadvantage of having a statically connected electronics platform? Does having a static or dynamically configured electronics platform affect the usage of a Message Queue technology? |
|  |
|  |

|  |
| --- |
| Using your terminal or command prompt, run a mosquitto\_sub program, subscribing to the topic **"/sensor/temp"**, then observe the output of the terminal or command prompt. Are you receiving the message? |
|  |
| When subscribing to the topic “/sensor/temp”, the output of the command prompt did not show anything. |
| Based on Schematic 4-3, why do you think that you have to convert the floating point return value of the DHT22 to a string first then to a character array, before sending it to the broker? |
|  |
| Yes because the publisher only accepted character pointers. |

|  |
| --- |
| What would happen if you directly sent a floating point value to the broker instead? What do you think would happen? Can you try it out and indicate on the space below, if you encountered any errors? |
|  |
| The function only accepts character pointer in publish therefore when we tried to put a floating point value, it outputted an error. |

# Challenge Activity

Understanding that the use of MQTT and a network connectivity using your electronics platforms, allow you to send messages via network instead of serial, frees your electronics platform from using the serial as a medium for communication, thus it is possible to that your electronics platform is no longer connected to a computer, but only to a battery and the network, and it should still work.

To simulate this scenario, create a schematic that would allow your electronics platform to send temperature data through the topic of “**/sensor/temp**” and humidity data through the topic of “**/sensor/humidity**” through the network, while being connected to one the workstations on your table. Send the temperature data, every 3 seconds, and the humidity data, every 5 seconds.

In the workstation of your table, write a Python program, that would take an input or argument from the user, enabling him or her to select if he or she would like to subscribe to “**/sensor/temp**” or “**/sensor/humidity**”. With this run two (2) instances of that Python program, one subscribing to the temperature, while the other one, subscribing to the humidity.

Further improve the setup by having improving the Python program, and creating another option in the argument or input, to now be a publisher on the topic of “**/actuator/value**” which continuously asks the user for a value from 0-4. Now, improve the schematic of your electronics platform to receive the value from the topic of “**/actuator/value**” where the value received from that topic would indicate the number of LED lights that would be turning on.

Note that the maximum delay in your program should only use “**delay(10)**”.

When you are done, notify your instructor to have your activity credited.