

## Practical Sheet N° 5 (Node-Red)

Objectives	Duration
<p>At the end of the training, you will be able to:</p> <ul style="list-style-type: none"> <li>Develop a device driver for an Edge device that allows you to interface digital (GPIO) or analog with a sensor/actuator.</li> <li>Develop firmware for an Edge that communicates through an MQTT broker with an application server and thus update a dashboard that aggregates the information of the Edges developed by all the groups.</li> </ul>	60min.

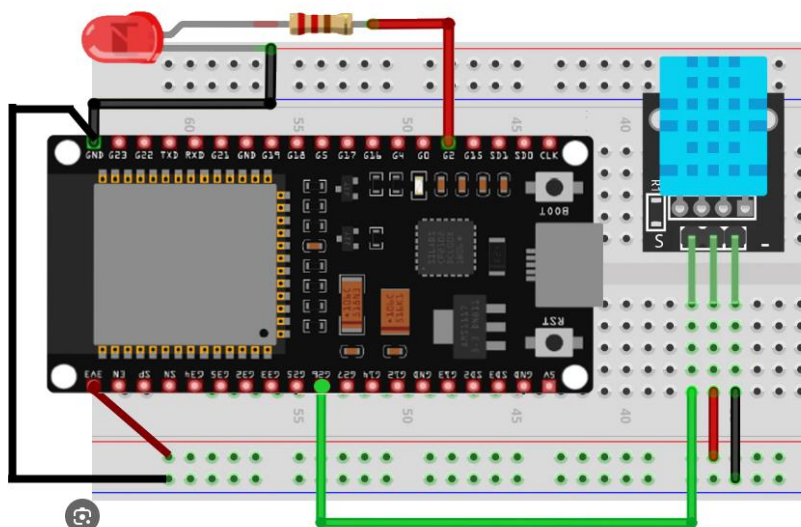
### Part 1 – Device driver

In this work it is intended that each group develop a device capable of interfacing a sensor/actuator. The sensor used is the Temperature humidity sensor DHT11, and the actuator will be simulated by a LED, but can be replaced by a relay, a buzzer or any other actuator.



Temperature range	0 to 50 °C $\pm 2^{\circ}\text{C}$
Humidity range	20 to 90% $\pm 5\%$
Resolution	Humidity: 1% Temperature: $1^{\circ}\text{C}$
Operating voltage	3 – 5.5 V DC
Current supply	0.5 – 2.5 mA
Sampling period	1 second

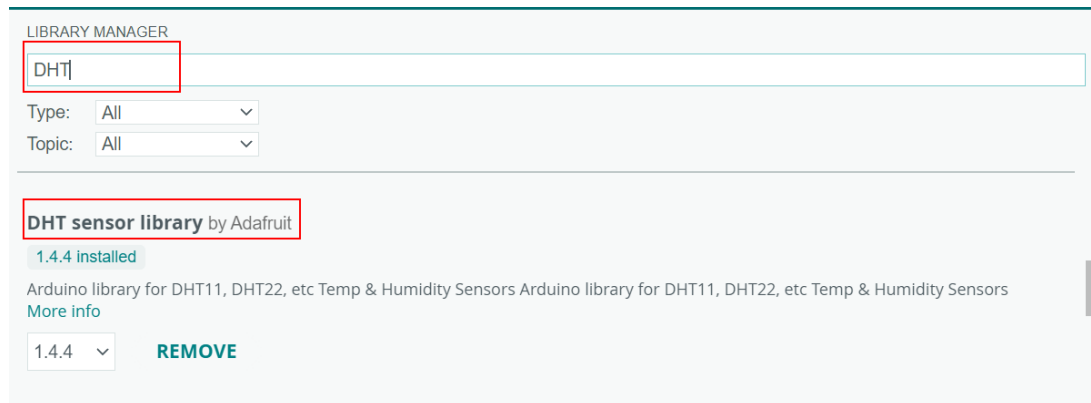
1. Connect your DHT11 sensor, ESP32Dev board and the LED in the same way as you can see in the picture above:



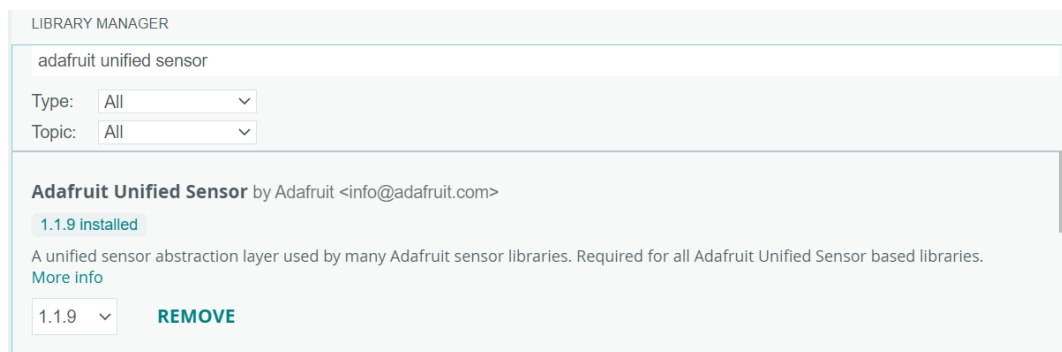
2. To read from the DHT sensor, we will use the DHT library from Adafruit. To use this library, you also need to install the Adafruit Unified Sensor library. Follow the next steps to install those libraries.

Open your Arduino IDE and go to Sketch > Include Library > Manage Libraries. The Library Manager should open.

Search for “DHT” on the Search box and install the DHT library from Adafruit.



After installing the DHT library from Adafruit, type “**Adafruit Unified Sensor**” in the search box. Scroll all the way down to find the library and install it.



3. Create a new Sketch and paste the following code:

```
#include "DHT.h"

#define DHTPIN 4
#define DHTTYPE DHT11

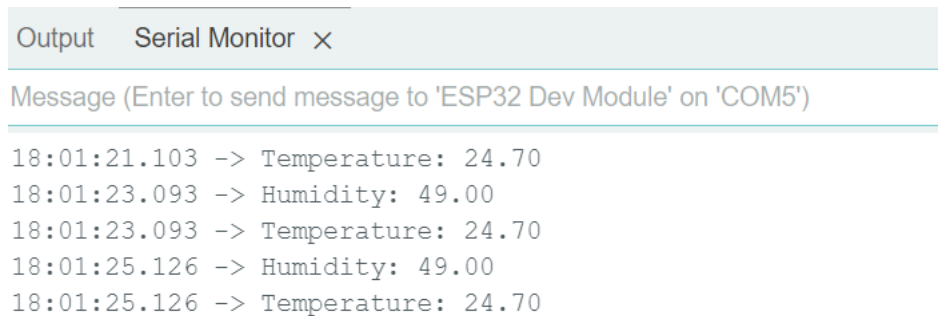
DHT dht(DHTPIN, DHTTYPE);

void setup() {
  Serial.begin(9600);
  Serial.println(F("DHTxx test!"));

  dht.begin();
```

```
}  
  
void loop() {  
    delay(2000);  
  
    float h = dht.readHumidity();  
    float t = dht.readTemperature();  
  
    if (isnan(h) || isnan(t)) {  
        Serial.println(F("Failed to read from DHT sensor!"));  
        return;  
    }  
  
    Serial.print(F("Humidity: "));  
    Serial.println(h);  
    Serial.print(F("Temperature: "));  
    Serial.println(t);  
}
```

After uploading the code to the ESP32 , you should see the following results on the serial port:



```
18:01:21.103 -> Temperature: 24.70  
18:01:23.093 -> Humidity: 49.00  
18:01:23.093 -> Temperature: 24.70  
18:01:25.126 -> Humidity: 49.00  
18:01:25.126 -> Temperature: 24.70
```

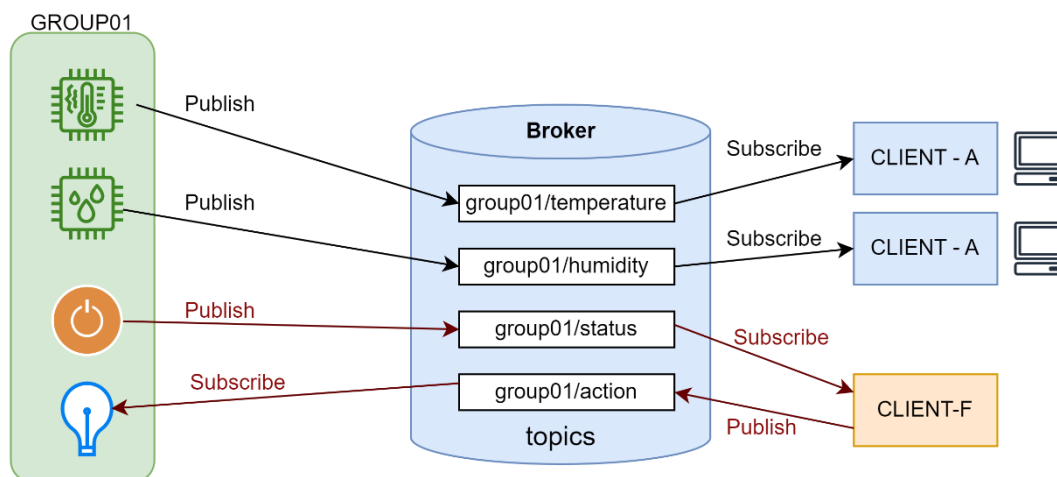
## Part 2 – Edge device firmware development

In this part we intend to develop the Edge device firmware. This should communicate through an MQTT broker with an application server and thus update a dashboard that aggregates the information from all the Edges developed by all the groups.

The sensor devices should be configured to operate in MQTT Push mode, communicating data to the server whenever necessary, and the actuators should be configured in MQTT Pull mode, i.e., the actuator status should be communicated back to the device that caused the status change in the actuator. Make the device publish the temperature and humidity every 10 seconds.

Group 1 for example should publish the temperature and relative humidity read by the sensor and publish the LED status. It also subscribes to an action that should turn **OFF** the LED if it is **ON** or vice versa.

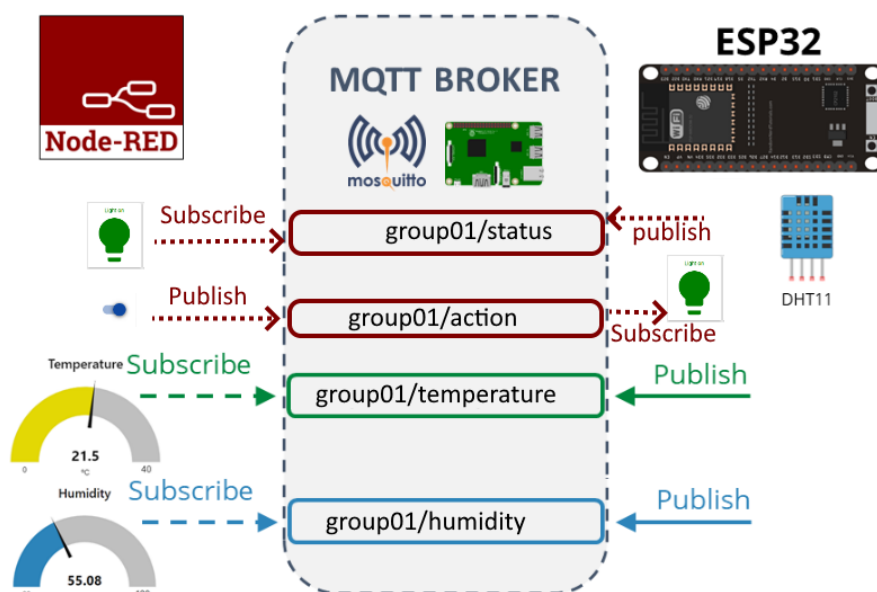
Using the MQTT broker implement the firmware to publish and subscribe to the topics described in the above figure.



### Part 3 – GUI Dashboard

The dashboard's graphical interface will be developed using Node-red. Besides the readings, it is possible to insert command buttons and the graphical representation of a light bulb.

The following diagram shows a high-level overview of the project we will build.



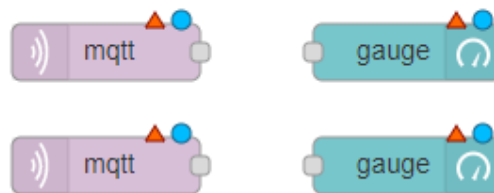
The ESP32 is publishing temperature readings every 10 seconds on the **group01t/temperature** and **group01/humidity** topics. Now, you can use any dashboard that supports MQTT or any other device that supports MQTT to subscribe to those topics and receive the readings.

As an example, we will create a simple flow using Node-RED to subscribe to those topics and display the readings on gauges.

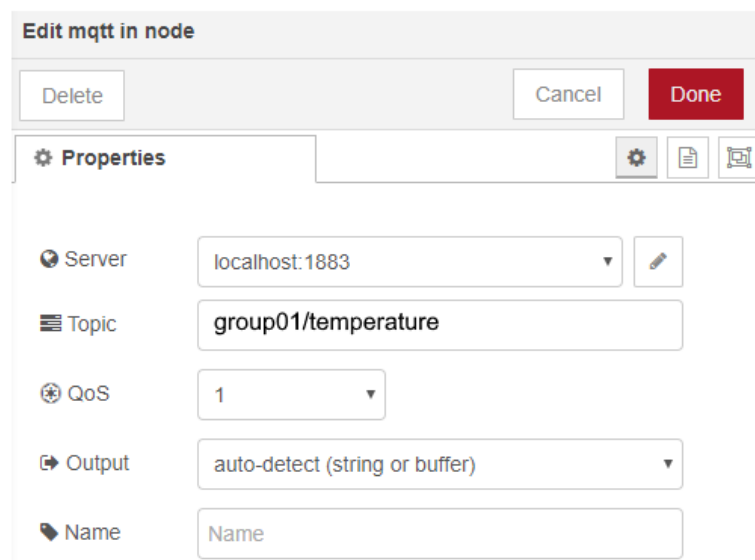
1. Install node-red on your machine. Documentation found [here](#).

Having Node-RED running on your machine go to **localhost :1880**.

2. Drag two MQTT in Nodes and two gauge nodes on the flow.



3. Click in the MQTT node and edit the properties.



Insert the topic you want to be subscribed to and the QoS. This previous MQTT node is subscribed to the **group01/temperature** topic.

Click on the other MQTT in node and edit its properties with the same server, but for the other topic: **group01/humidity**.

Click on the gauge nodes and edit its properties for each reading. The following node is set for the temperature readings. Edit the other chart node for the humidity readings.

**Edit gauge node**

Delete Cancel Done

**Properties**

Group [Home] DHT

Size auto

Type Gauge

Label Temperature

Value format {{value}}

Units °C

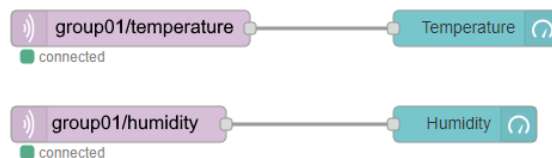
Range min 0 max 40

Colour gradient

Sectors 0 ... optional ... optional ... 40

Name

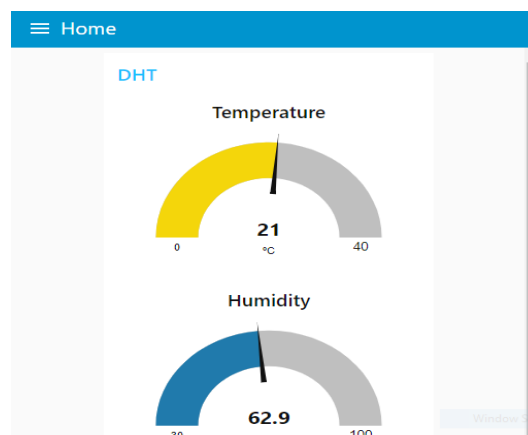
4. Wire your nodes as shown below:



5. Finally deploy your follow (press the button on the upper right corner)



6. Open a browser on <http://localhost:1883/ui>  
You should get access to the current DHT temperature and humidity readings on the Dashboard.



7. Add one more MQTT subscriber and one publisher, a Led and a toggle button and implement the rest of the dashboard to turn on and off the LED. Try to switch ON/OFF the other group LEDs.