University of Lleida

Master's Degree in Informatics Engineering

Higher Polythecnic School

Data Producer 1

Ubiquitous Computing and Embedded Systems

Francesc Contreras

Albert Pérez

Marc Visa

November 10, 2021

Table of contents

List of Figures

1 Introduction

The purpose of this document is to explain the Data Producer 1 development for the Sprint 1 of the project.

We shall introduce all the steps and software we have used when developing the it and showing images as a real example for detailed explanation.

2 Environment

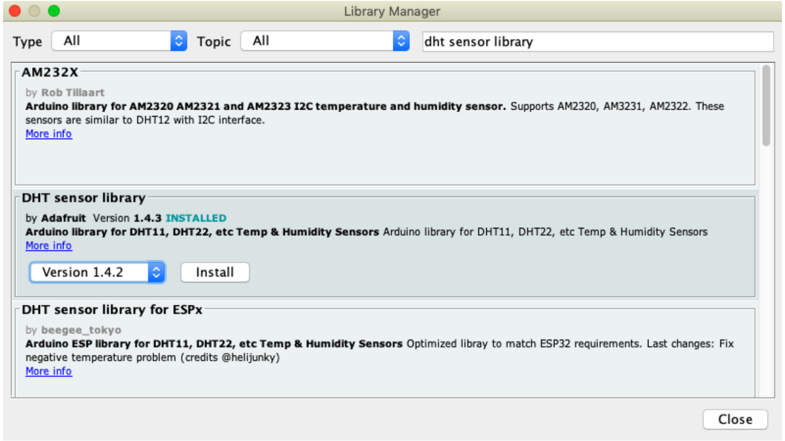
At first, these are the steps for installing the libraries and setting up the Arduino IDE to get started for the Data Producer 1 development:

1. Install Arduino IDE v.1.8.16.
2. Install driver for USB adapter to ESP-01 and switch the interrupter to PROG mode.
3. Install ESP8266 Board from Boards Manager from Arduino IDE.
4. Install the following libraries from the Manage Libraries of Arduino IDE:
   (a) DHT [1] library by Adafruit.
   (b) Adafruit Unified Sensor by Adafruit.
5. Install the following libraries from GitHub: (These libraries should be added to the Arduino/libraries folder):
   (a) PubSubClient[2]
6. Connect the ESP-01[3] along with the USB adapter to your machine.
7. Code the program for the ESP-01 and DHT11.

[1] Temperature and humidity sensor

[2] MQTT Library

[3] Low-cost WiFi microchip with built-in TCP/IP networking software, and microcontroller capability



Library Manager

Type: All Topic: All dht sensor library

AM232X

by Rob Tillaart

Arduino library for AM2320 AM2321 and AM2323 I2C temperature and humidity sensor. Supports AM2320, AM3231, AM2322. These sensors are similar to DHT12 with I2C interface.

More info

DHT sensor library

by **Adafruit** Version **1.4.3** **INSTALLED**

Arduino library for DHT11, DHT22, etc Temp & Humidity Sensors Arduino library for DHT11, DHT22, etc Temp & Humidity Sensors

More info

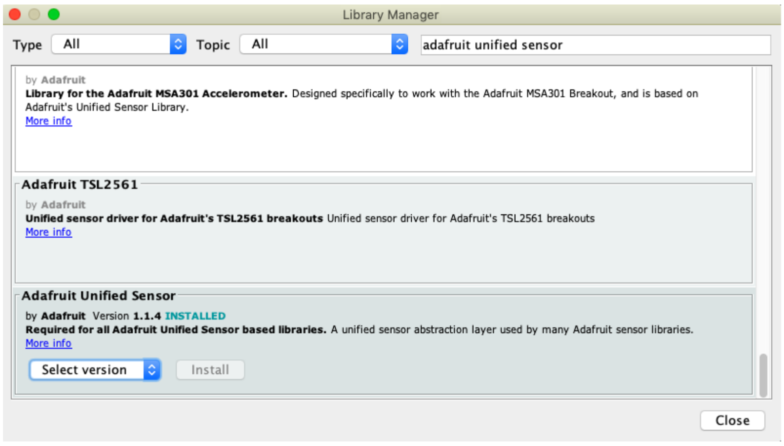
DHT sensor library for ESPx

by beegee_tokyo

Arduino ESP library for DHT11, DHT22, etc Temp & Humidity Sensors Optimized libray to match ESP32 requirements. Last changes: Fix negative temperature problem (credits @helijunky)

More info

Figure 1: Installing the DHT sensor library for ESPx from the Arduino Manage Libraries.



Library Manager

Type: All Topic: All adafruit unified sensor

by Adafruit

Library for the Adafruit MSA301 Accelerometer. Designed specifically to work with the Adafruit MSA301 Breakout, and is based on Adafruit's Unified Sensor Library.

More info

Adafruit TSL2561

by Adafruit

Unified sensor driver for Adafruit's TSL2561 breakouts Unified sensor driver for Adafruit's TSL2561 breakouts

More info

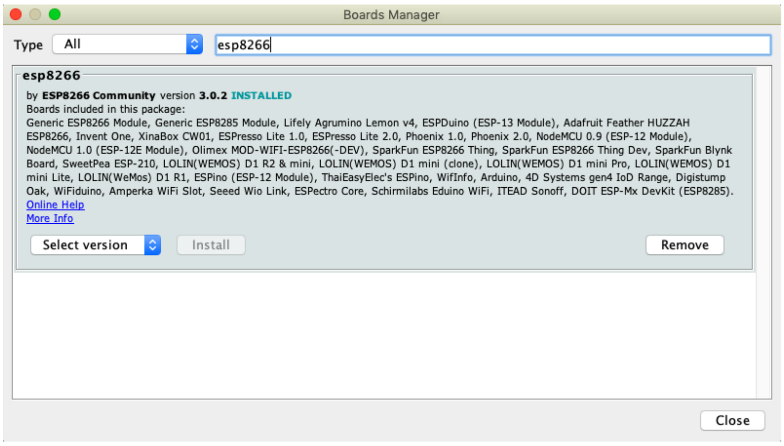
Adafruit Unified Sensor

by **Adafruit** Version **1.1.4** **INSTALLED**

Required for all Adafruit Unified Sensor based libraries. A unified sensor abstraction layer used by many Adafruit sensor libraries.

More info

Figure 2: Installing the Adafruit Unified Sensor from the Arduino Manage Libraries.



Boards Manager

Type: All esp8266

esp8266

by **ESP8266 Community** version **3.0.2** **INSTALLED**

Boards included in this package:

Generic ESP8266 Module, Generic ESP8285 Module, Lifely Agrumino Lemon v4, ESPDuino (ESP-13 Module), Adafruit Feather HUZZAH ESP8266, Invent One, XinaBox CW01, ESPresso Lite 1.0, ESPresso Lite 2.0, Phoenix 1.0, Phoenix 2.0, NodeMCU 0.9 (ESP-12 Module), NodeMCU 1.0 (ESP-12E Module), Olimex MOD-WIFI-ESP8266(-DEV), SparkFun ESP8266 Thing, SparkFun ESP8266 Thing Dev, SparkFun Blynk Board, SweetPea ESP-210, LOLIN(WEMOS) D1 R2 & mini, LOLIN(WEMOS) D1 mini (clone), LOLIN(WEMOS) D1 mini Pro, LOLIN(WEMOS) D1 mini Lite, LOLIN(WeMos) D1 R1, ESPino (ESP-12 Module), ThaiEasyElec's ESPino, WifInfo, Arduino, 4D Systems gen4 IoD Range, Digistump Oak, WiFiduino, Amperka WiFi Slot, Seeed Wio Link, ESPectro Core, Schirmilabs Eduino WiFi, ITEAD Sonoff, DOIT ESP-Mx DevKit (ESP8285).

Online Help

More Info

Figure 3: Installing the ESP8266 Board from the Arduino Boards Manager.

2.1 Material required

Subsequently, you will find all the required components for the Data Producer 1, and a short explanatory description for those not so common.

- **ESP-01 (3.3 V)**: is a Wi-Fi module that allows microcontrollers access to a Wi-Fi network.
- **Breadboard**: A board, having a matrix of small holes to which components may be attached without solder.
- **LCD Screen (5V)**: is a flat-panel display or other electronically modulated optical device that uses the light-modulating properties of liquid crystals combined with polarizers.
- **DHT11 (3.3V)**: is a basic, ultra low-cost digital temperature and humidity sensor.
- **VCC Protoboard adapter (3.3V, 5V)**
- **ESP Programmer module**
- **Wires**

3 Development

Regards the development, this is based on two parts, the first one realated to work directly with the electronic components, and the second by programming the arduino sketch.

The following sections shows you how the electronic schema is in real, as well as how finally we got the visualization of the temperature and the humidity by two different ways.

Moreover, there is the code shown below, but also you may consult the github repository of the project, the one is specified subsequently.

3.1 Hardware

3.1.1 Diagram

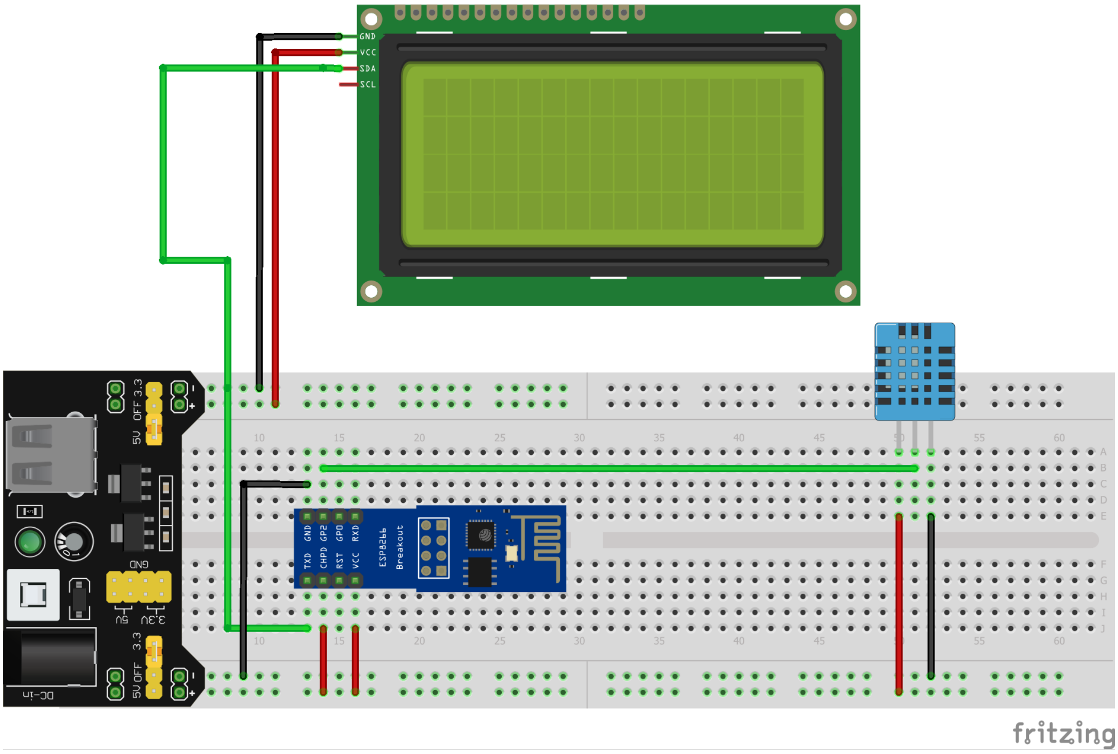


Figure 4: Data Producer 1 Diagram

Pay Attention!, in our LCD Screen we don't have SDA and SCL connections, we have RX and TX, then our RX is SDA and TX is SCL in the Figure 4.

3.1.2 Assembly

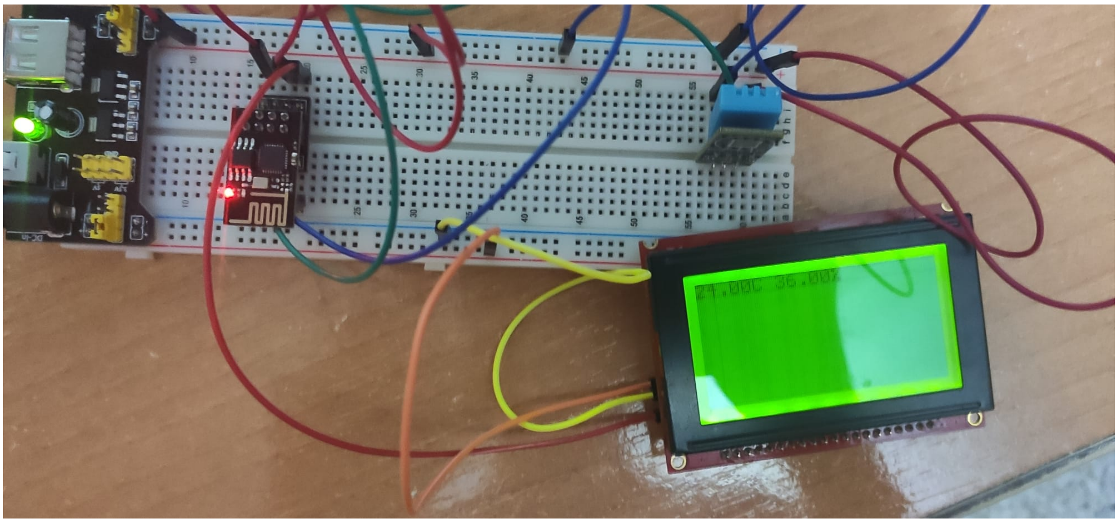


Figure 5: Data Producer 1 Assembly

3.2 Software

```
// Import required libraries
#include <Arduino.h>
#include <ESP8266WiFi.h>

#include <Adafruit_Sensor.h>
#include <DHT.h>

//MQTT Pub
#include <PubSubClient.h>

// Replace with your network credentials
const char* ssid = "-";
const char* password = "-";

// DHT11
#define DHTPIN 2   // Digital pin connected to the DHT sensor
#define DHTTYPE  DHT11    // DHT 11

DHT dht(DHTPIN, DHTTYPE);

float t = 0.0;
float h = 0.0;

// MQTT

const char* mqtt_server = "192.168.1.163"; // MQTT BROKER IP
WiFiClient espClient;
```

```
PubSubClient client(espClient);

#define MSG_BUFFER_SIZE (50)
char msg[MSG_BUFFER_SIZE];

void callback(char* topic, byte* payload, unsigned int length) {
  /*Serial.print("Message arrived [");
  Serial.print(topic);
  Serial.print("] ");
  for (int i=0;i<length;i++) {
    Serial.print((char)payload[i]);
  }
  Serial.println();*/
}

void reconnect() {
  // Loop until we’re reconnected
  while (!client.connected()) {
    Serial.print("Attempting MQTT connection...");
    // Create a random client ID
    String clientId = "DataProducer1";
    clientId += String(random(0xffff), HEX);
    // Attempt to connect
    if (client.connect(clientId.c_str())) {
      Serial.println("connected");
      // Once connected, publish an announcement...
      client.publish("outTopic", "connected");
      // ... and resubscribe
      client.subscribe("#");
    } else {
      Serial.print("failed, rc=");
      Serial.print(client.state());
      Serial.println(" try again in 5 seconds");
      // Wait 5 seconds before retrying
      delay(5000);
    }
  }
}

void setup(){

  Serial.begin(115200);

  // Connect to Wi-Fi
  WiFi.begin(ssid, password);
  Serial.println("Connecting to WiFi");

  while (WiFi.status() != WL_CONNECTED) {
    delay(1000);
    Serial.println(".");
  }

  // Print ESP8266 Local IP Address
  Serial.println(WiFi.localIP());

  dht.begin();

  client.setServer(mqtt_server, 1883);
  client.setCallback(callback);
}

void clean_screen()
{
  Serial.write(0x7C);
  Serial.write(0x00);
}

void loop(){
    clean_screen();

    // MQTT
      if (!client.connected()) {
        reconnect();
      }
      client.loop();

    // Temperature
    float newT = dht.readTemperature();
    if (isnan(newT)) {
      Serial.println("Failed to read from DHT sensor!");
    }
    else {
      t = newT;
      String outStringT = (String) t+"C ";
      Serial.print(outStringT);
      snprintf (msg, MSG_BUFFER_SIZE, "%f C", t);
      client.publish("/temperature", msg);
    }

    // Humidity
    float newH = dht.readHumidity();
    // if humidity read failed, don't change h value
    if (isnan(newH)) {
      Serial.println("Failed to read from DHT sensor!");
    }
    else {
      h = newH;
      String outStringH = (String) h+"%";
      Serial.print(outStringH);
      snprintf (msg, MSG_BUFFER_SIZE, "%f ", h);
      client.publish("/humidity", msg);
    }

    delay(5000);
}
```

4 Code repository

Wind Turbine Generator Github repository

References

[1] ESP8266 DHT11/DHT22 Temperature and Humidity Web Server with Arduino IDE