Electronics 1

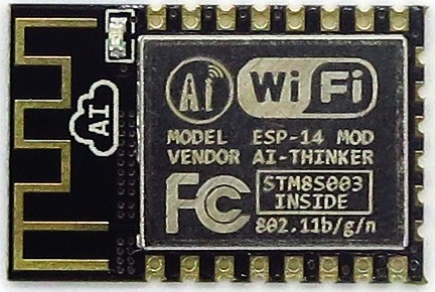
Electronics for Interactive Media Design

Emma Pareschi

WiFi Module: ESP8266

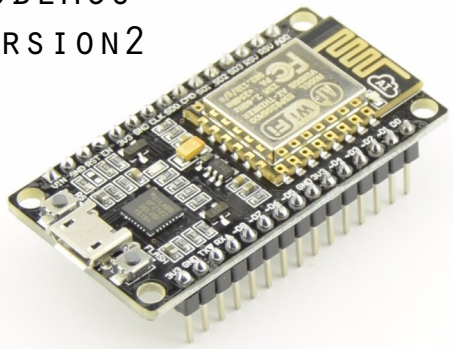
Different versions of the same module (chip):

ESP-12E module



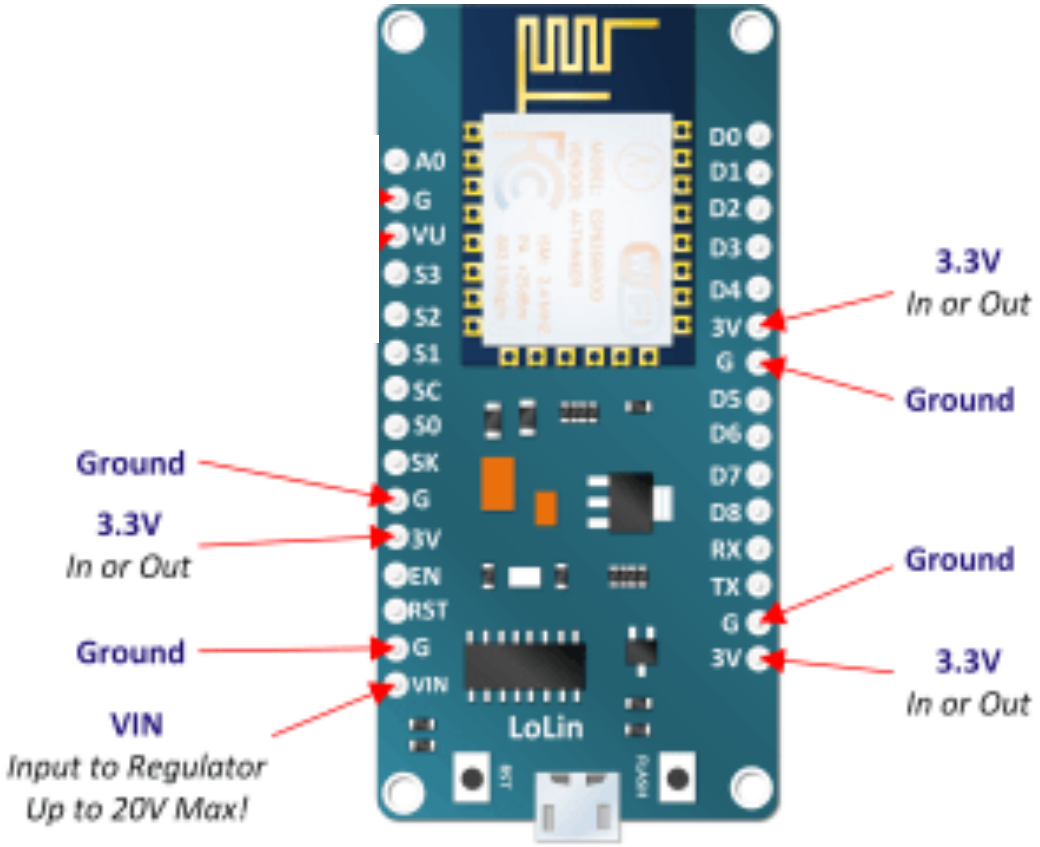
- Open-source firmware
- Arduino IDE compatible
- Programmable
- Low cost (1.15€)
- Simple
- Smart
- WI-FI enabled

NodeMCU version2



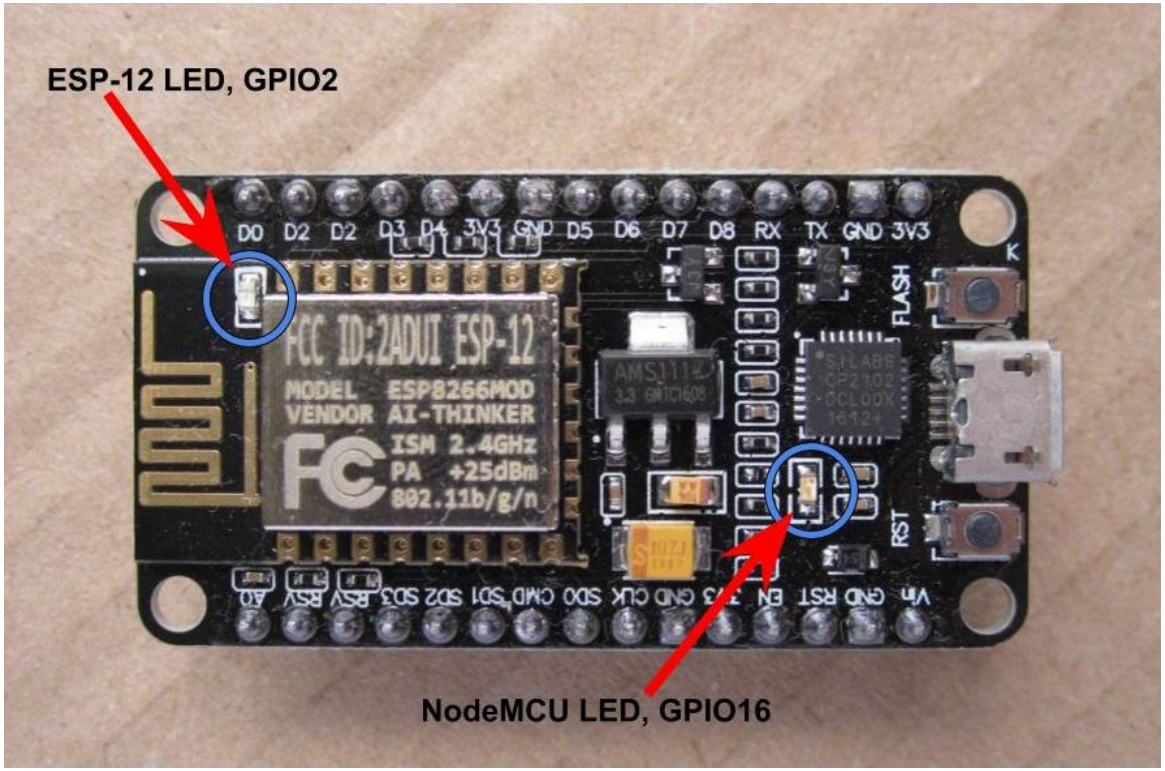
NodeMCU is a development kit, with open-source firmware, that helps you to prototype your IOT product.

NodeMCU v2 - Power Pin



ESP8266 is a 3.3V Module!!!!!

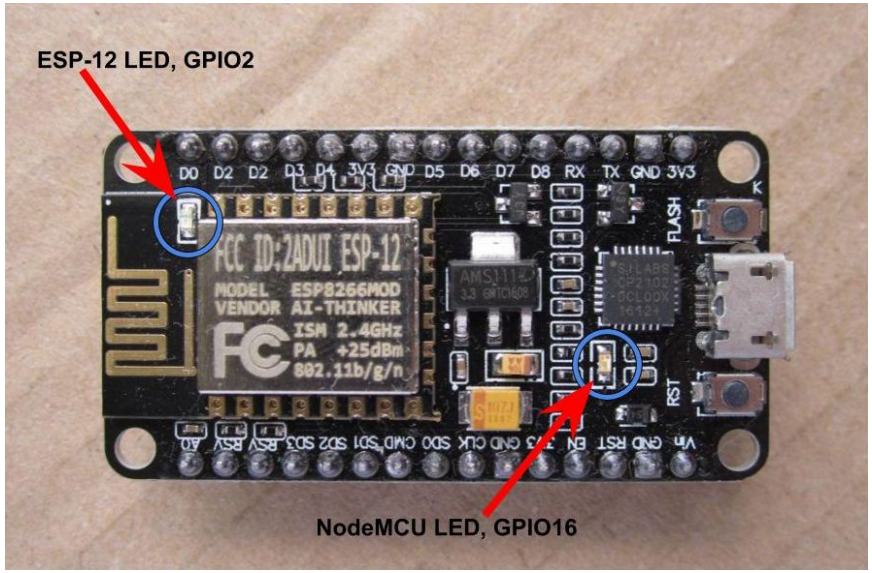
NodeMCU v2 - Onboard LEDs

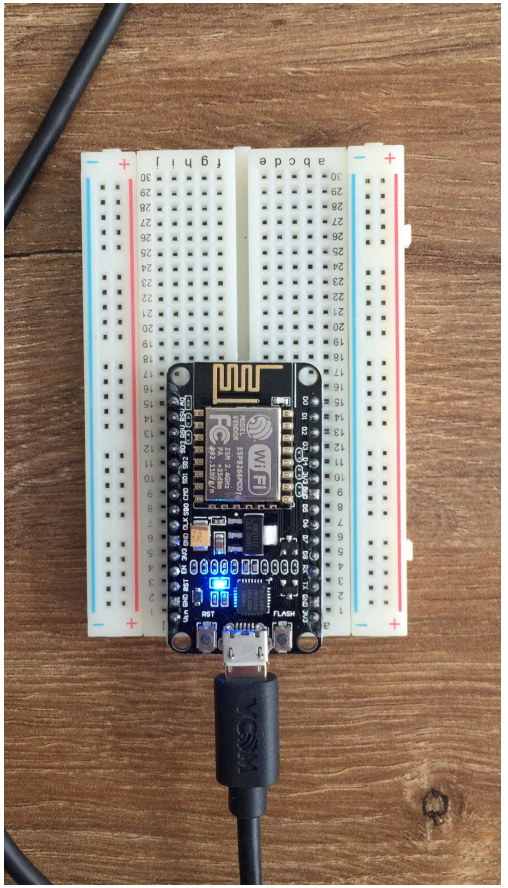


These two LEDs are handy when
Used as feedback.
Ex: everytime you send a message to the cloud =>
=> the led blinks

Image from https://lowvoltage.github.io/2017/07/09/Onboard-LEDs-NodeMCU-Got-Two

NodeMCU v2 - Hello LEDs





Overtime you work for the first time with a board,
You goes through the workflow to program it.
The most typical “hello world” code blinks the onboard LEDs.
On Arduino Uno board: one LED (pin 13 or build-in Led)
On NodeMCU: two LEDs

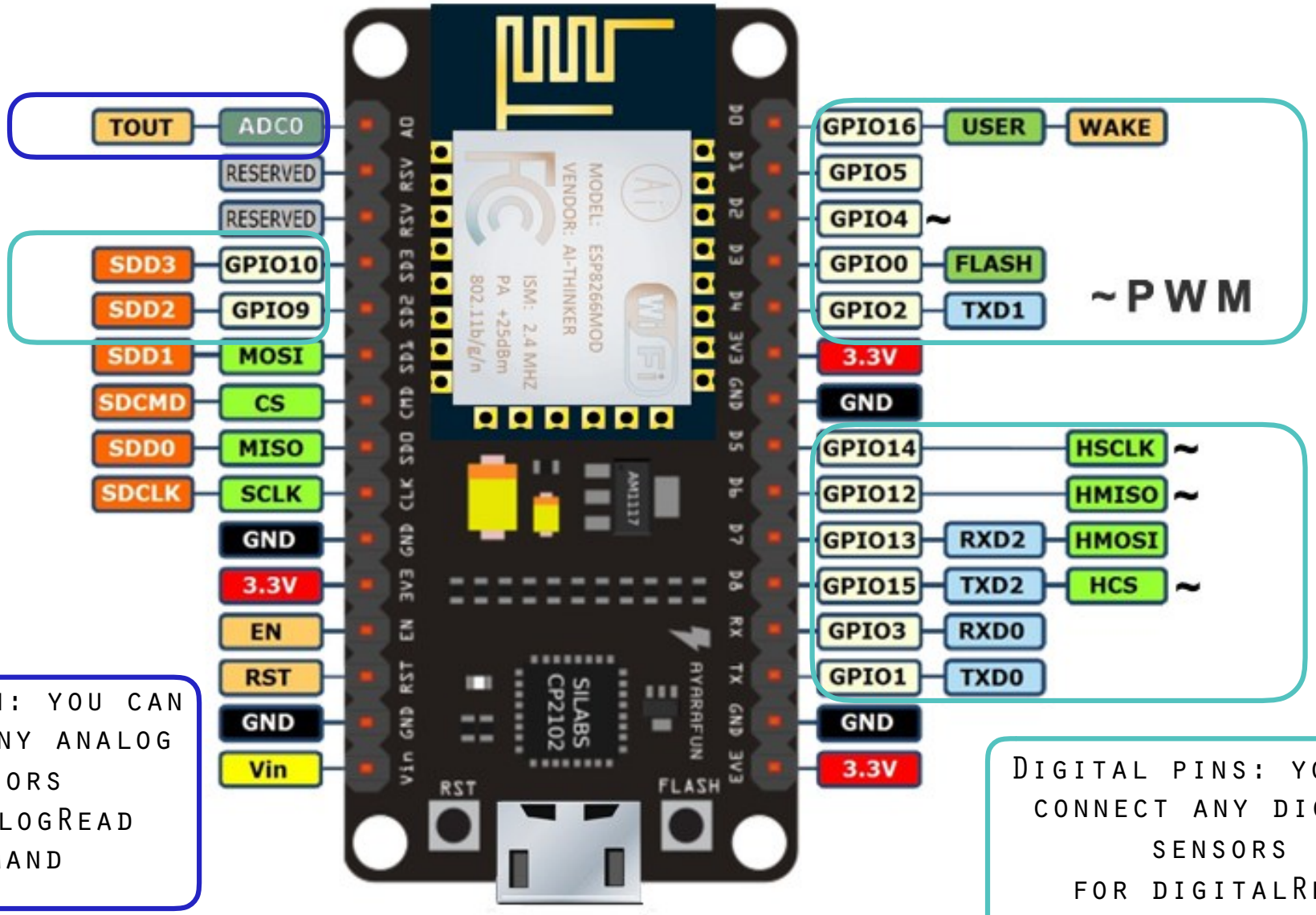
Image from https://lowvoltage.github.io/2017/07/09/Onboard-LEDs-NodeMCU-Got-Two

NodeMCU v2 - Arduino IDE

Download and Install Drivers
https://www.silabs.com/products/development-tools/software/usb-to-uart-bridge-vcp-drivers

Install the board in Arduino IDE:
—> in preference window: http://arduino.esp8266.com/stable/package_esp8266com_index.json
—> from boards manager: esp8266

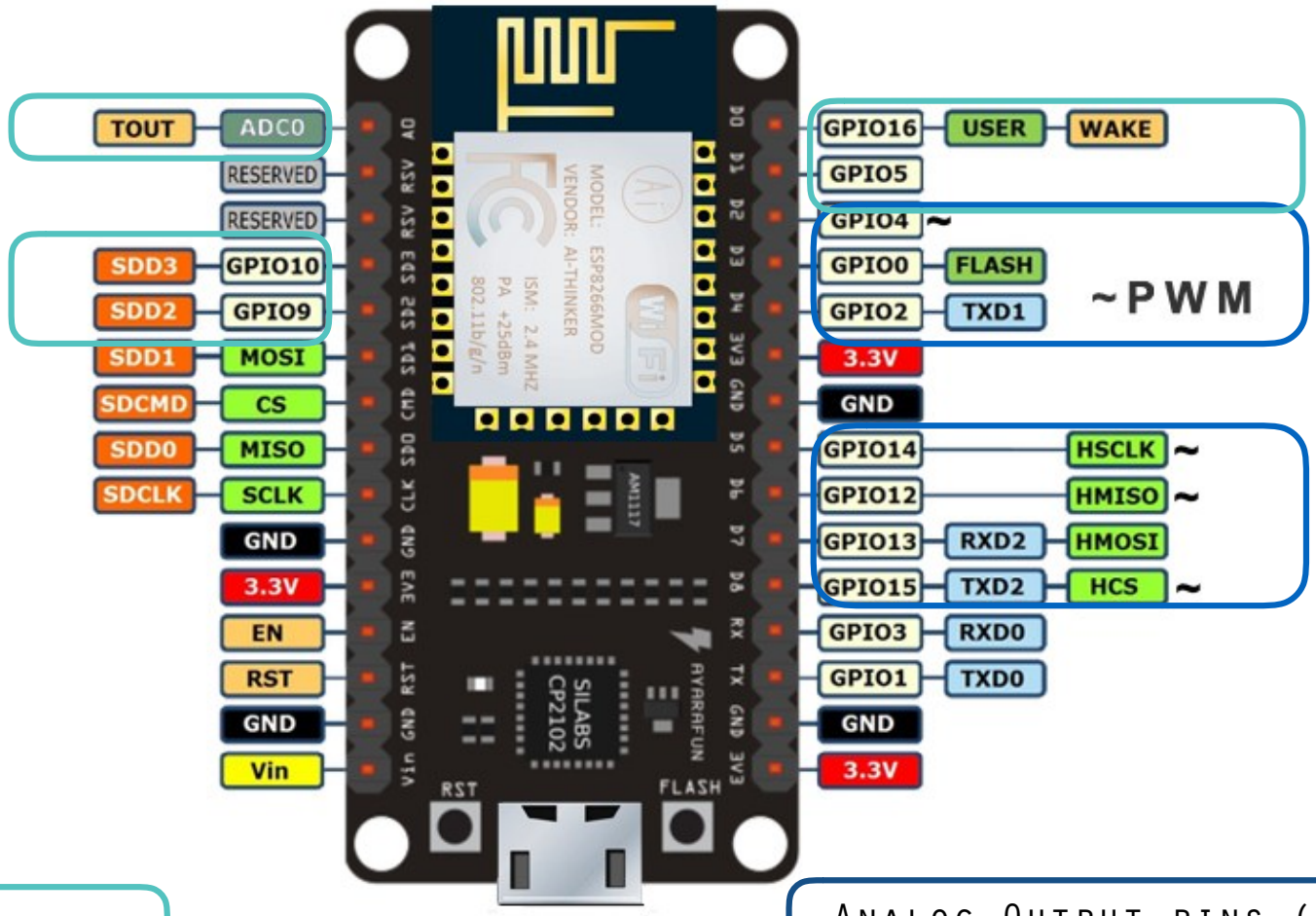
NodeMCU v2 - IO Pins - INPUT



Analog pin: you can connect any analog sensors for analogRead command

Digital pins: you can connect any digital sensors for digitalRead command

NodeMCU v2 - IO Pins - OUTPUT



Digital Output pins: you can connect any digital output for digitalWrite command

Analog Output pins (PWM): you can connect any analog sensors for analogWrite command (ex: control intensity of a Led)

Sensors

Kind of sensors:

- Digital
- Analog
- Analog/Digital
- Protocols/Data analysis

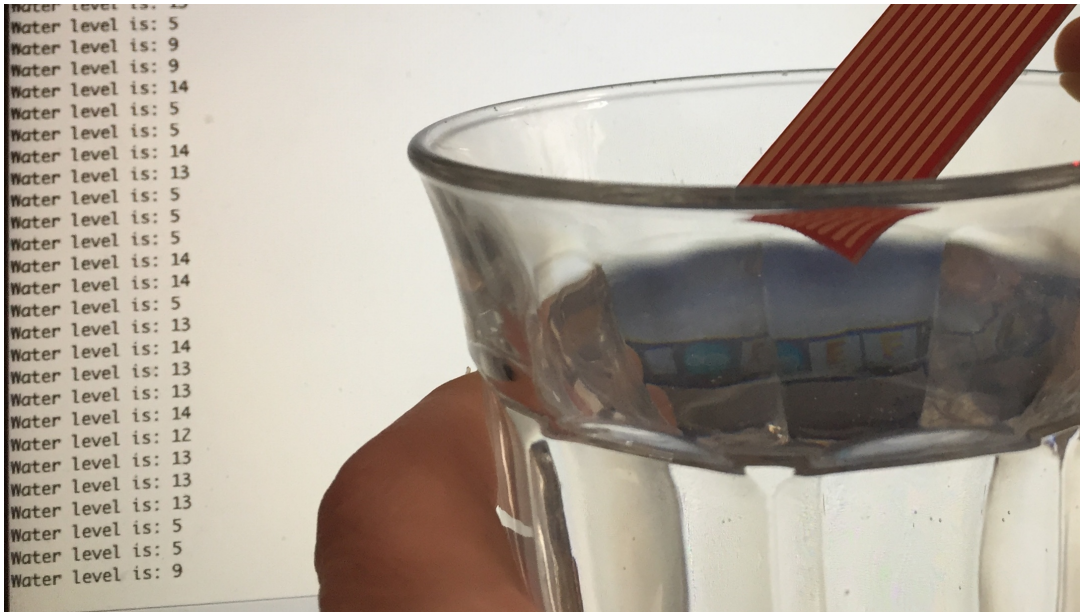
Water Level sensor (analog)

Signal => A0
Power Pin: 3V3
GND

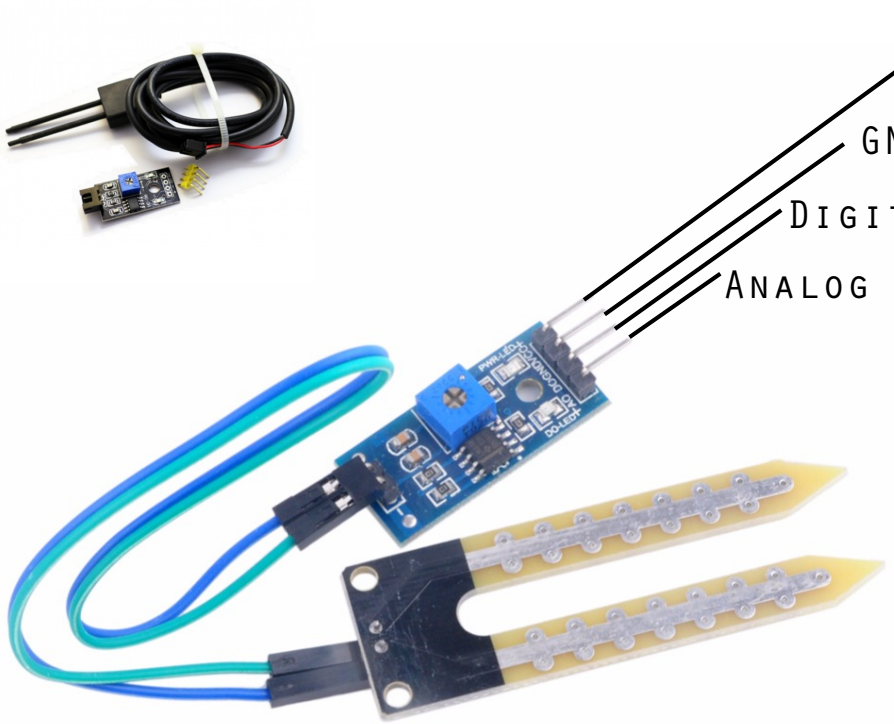
Power supply: 3.3V - 5V
Output: analog

Water Level sensor

Code: nodemcu_waterlevel.ino



Soil Moisture Sensor (analog/digital)



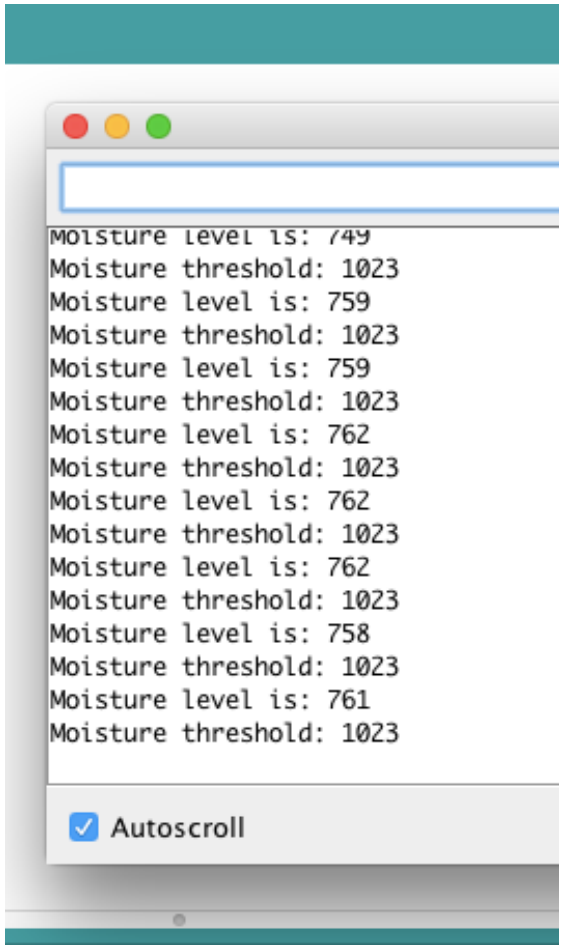
Power supply: 3.3V - 5V
Output: analog and/or digital

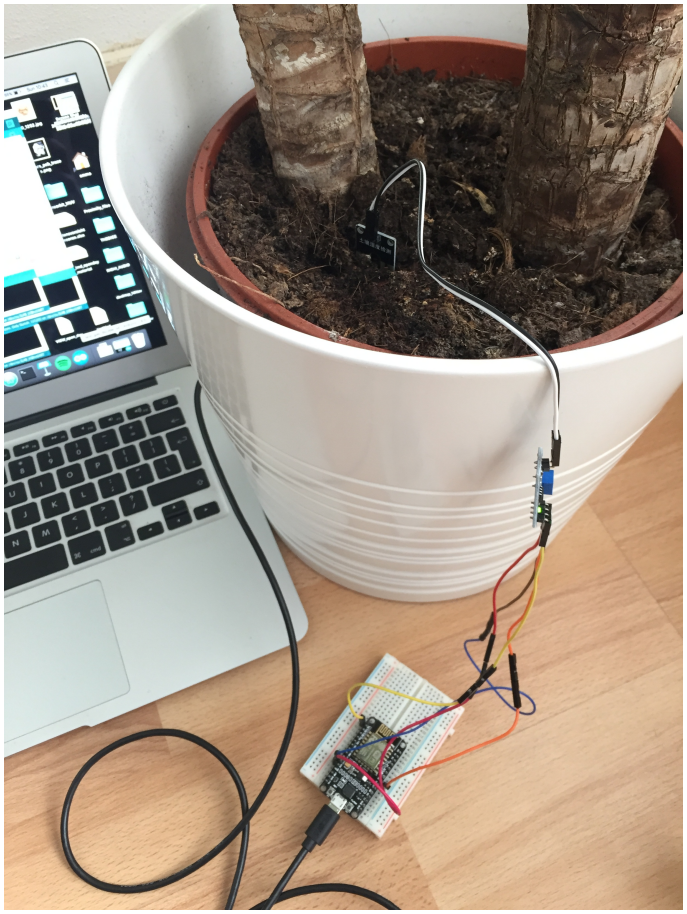
With Analog signal you check the level

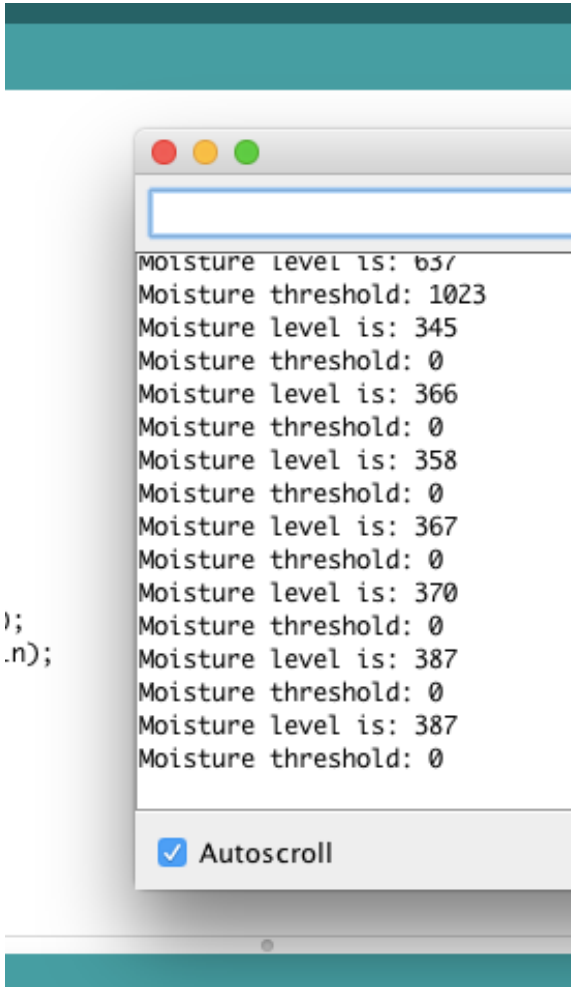
With Signal you check if moisture is higher or lower of a fixed threshold.
The threshold is fixed by the potentiometer on board.

Soil Moisture Sensor

Code: nodemcu_moisturelevel.ino



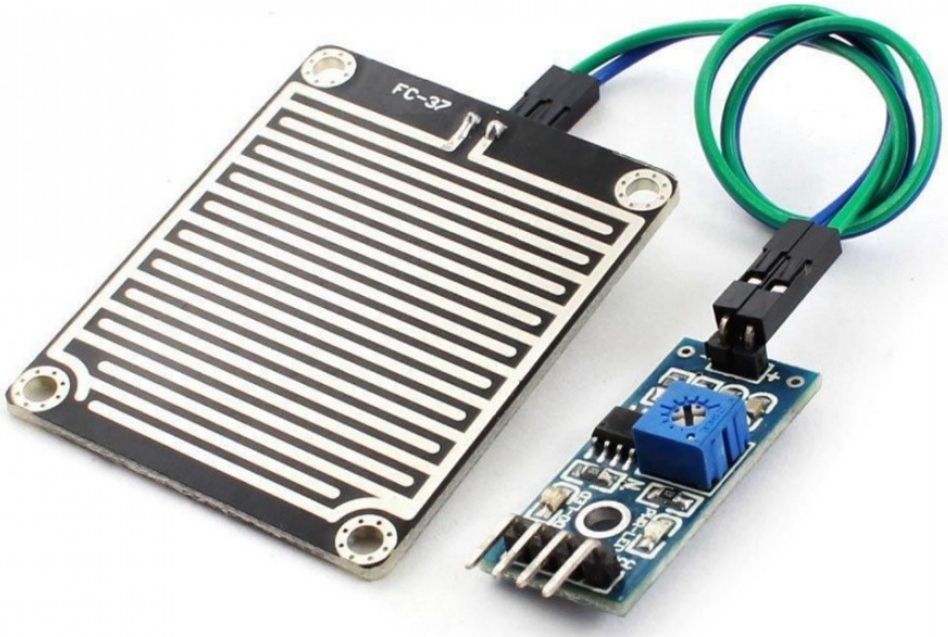




HIGH reading => Dry

LOW reading => Moist

Rain Sensor (analog/digital)



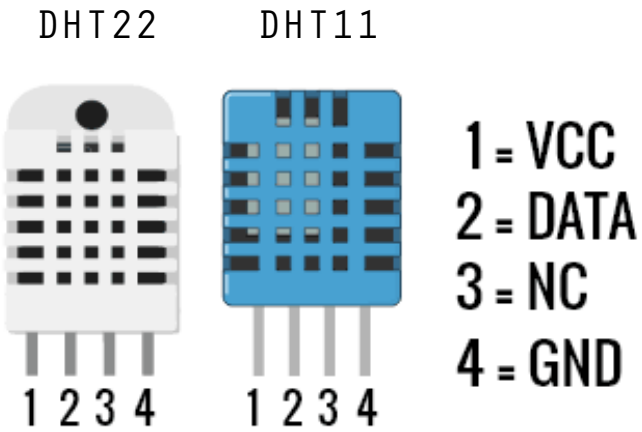
It works like the Moisture Sensor

Power supply: 3.3V - 5V
Output: analog and/or digital

With Analog signal you check the level

With Signal you check if moisture is higher or lower of a fixed threshold.
The threshold is fixed by the potentiometer on board.

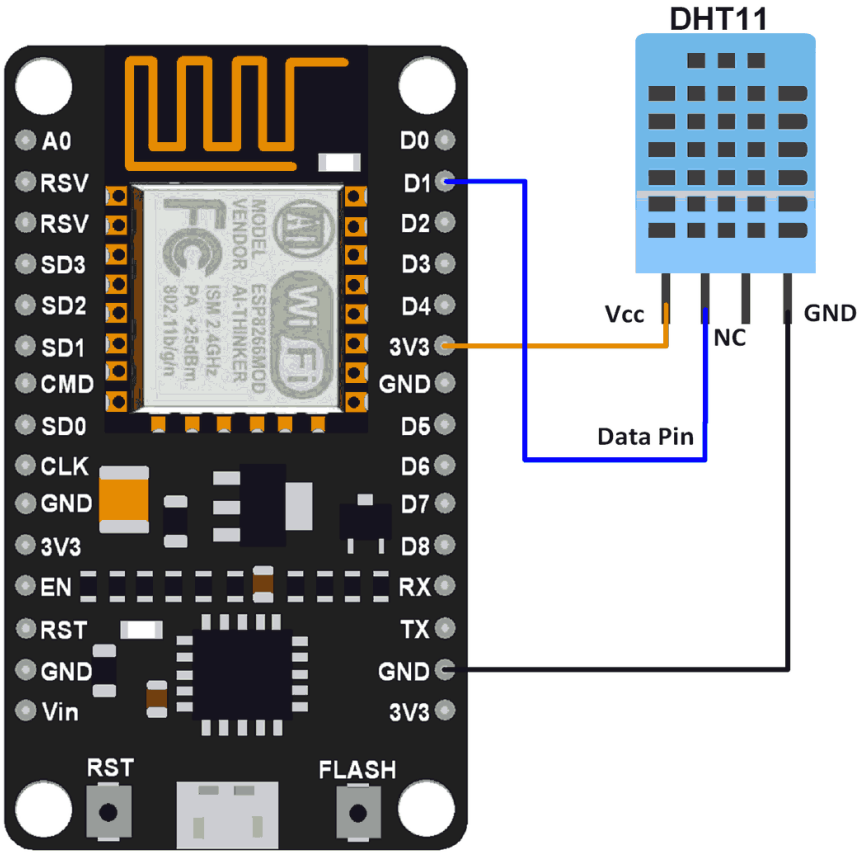
Air temperature and Humidity - DHT (protocol)



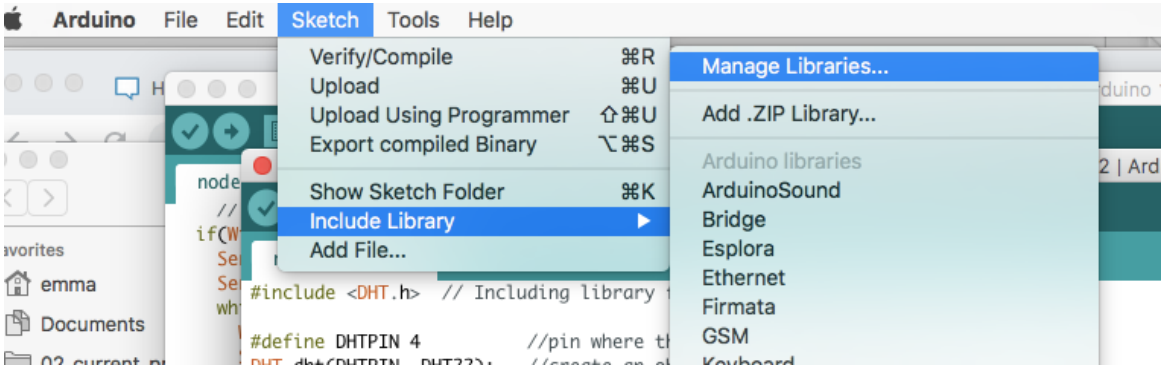
DHT22
VCC 3-5V
Temperature Range-40~80 C
Temp. Accuracy+/- 0.5C
Humidity Range0 ~ 100%
Humidity Accuracy+/- 2%

DHT11
VCC 3-5V
Temperature Range-40~80 C
Temp. Accuracy+/- 2C
Humidity Range0 ~ 100%
Humidity Accuracy+/- 4%

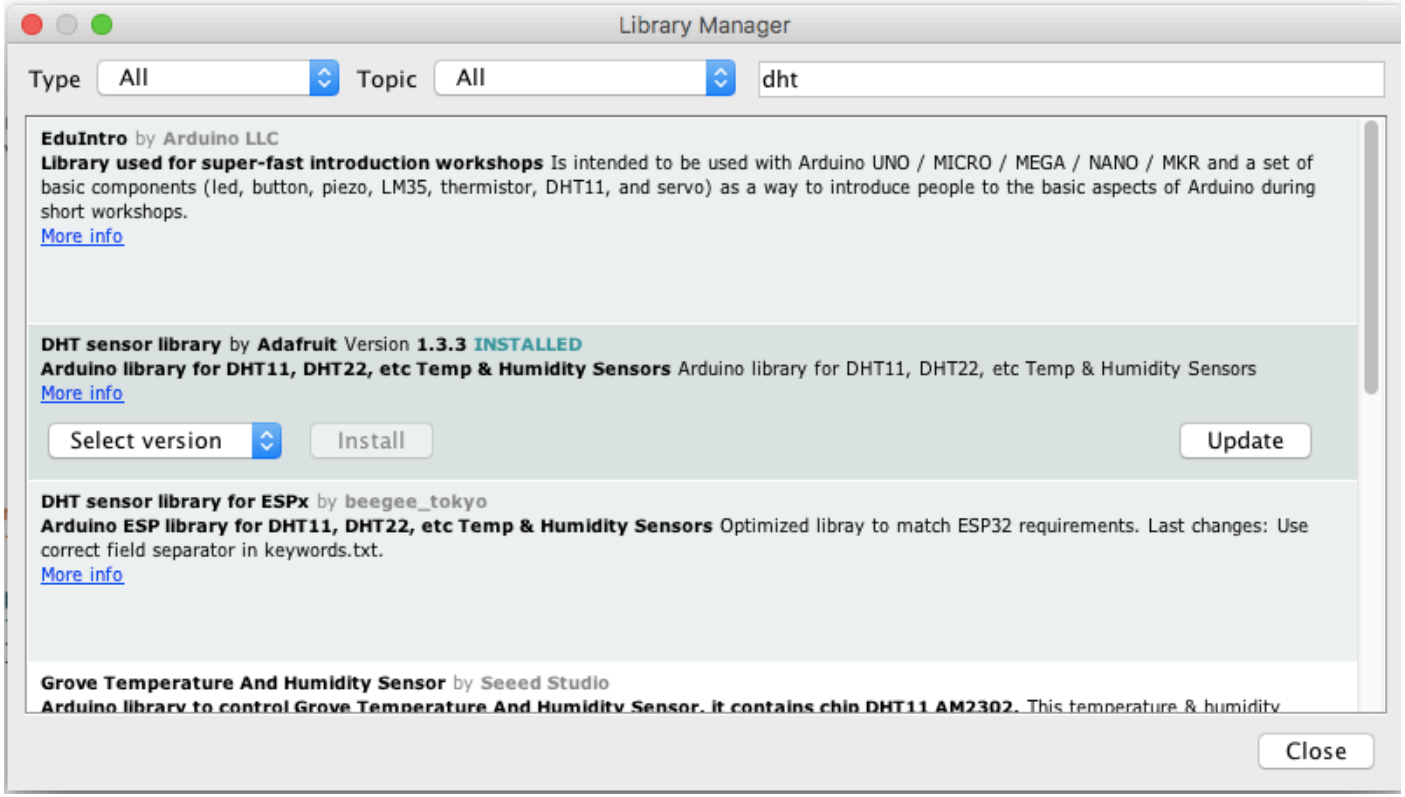
NodeMCU v2 - NodeMcu + DHT11



NodeMCU v2 - NodeMcu + DHT - Arduino IDE

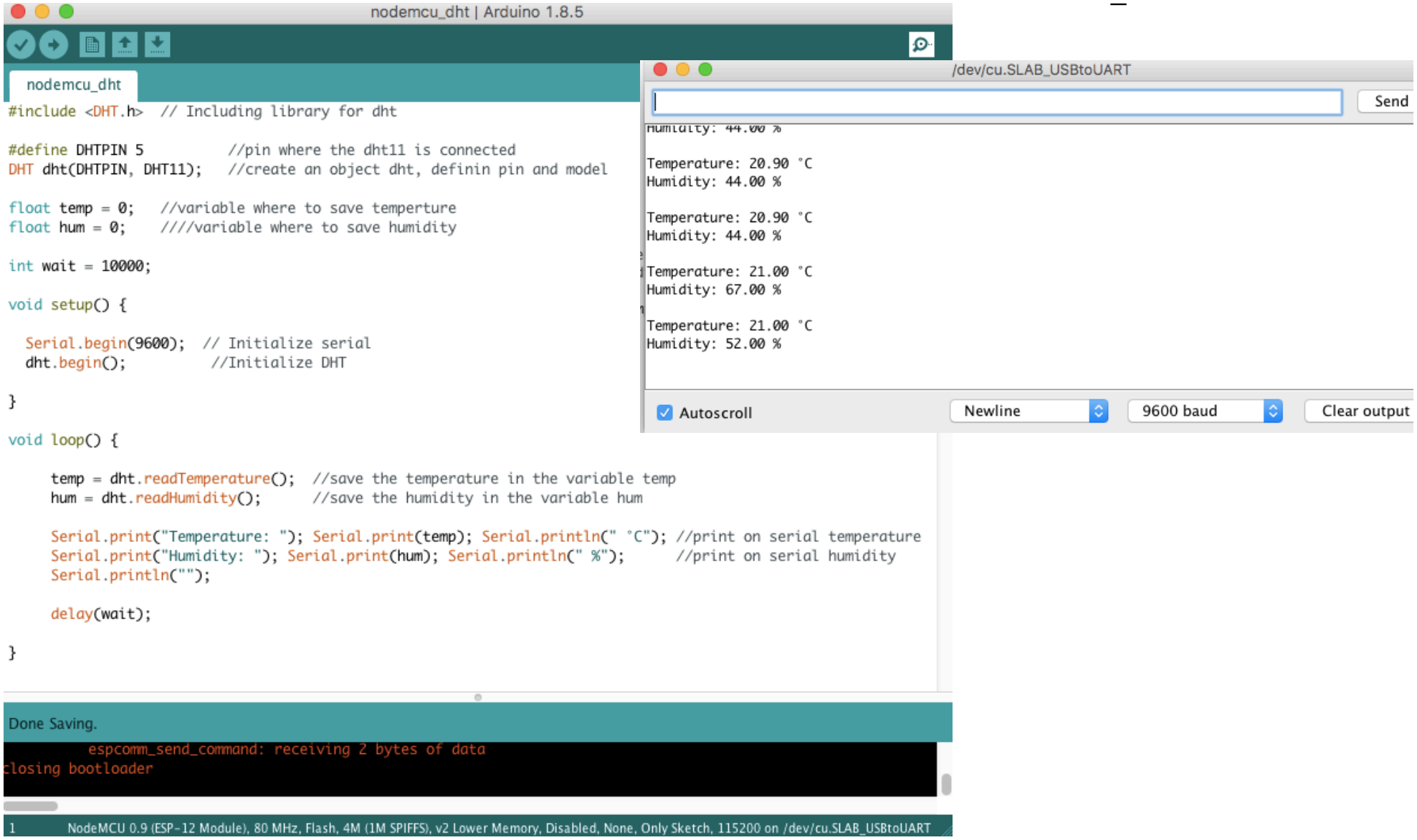


How to install the Library

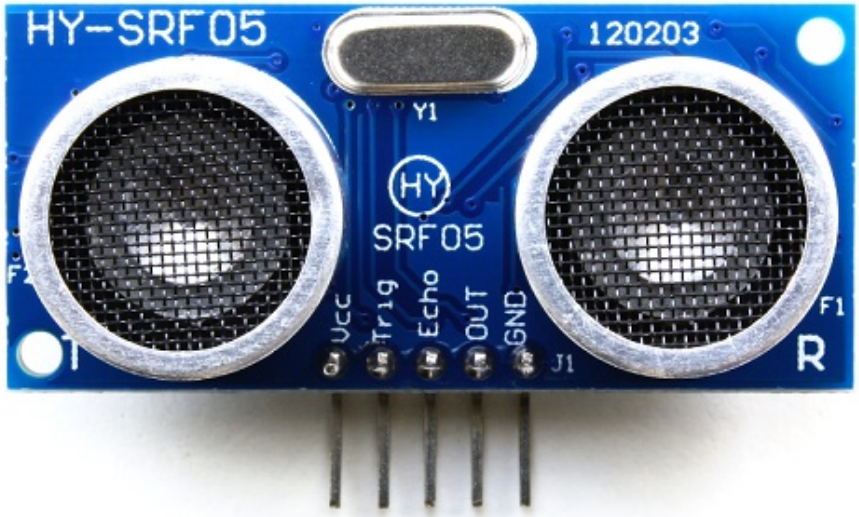


NodeMCU v2 - NodeMcu + DHT

Code: nodemcu_dht.ino

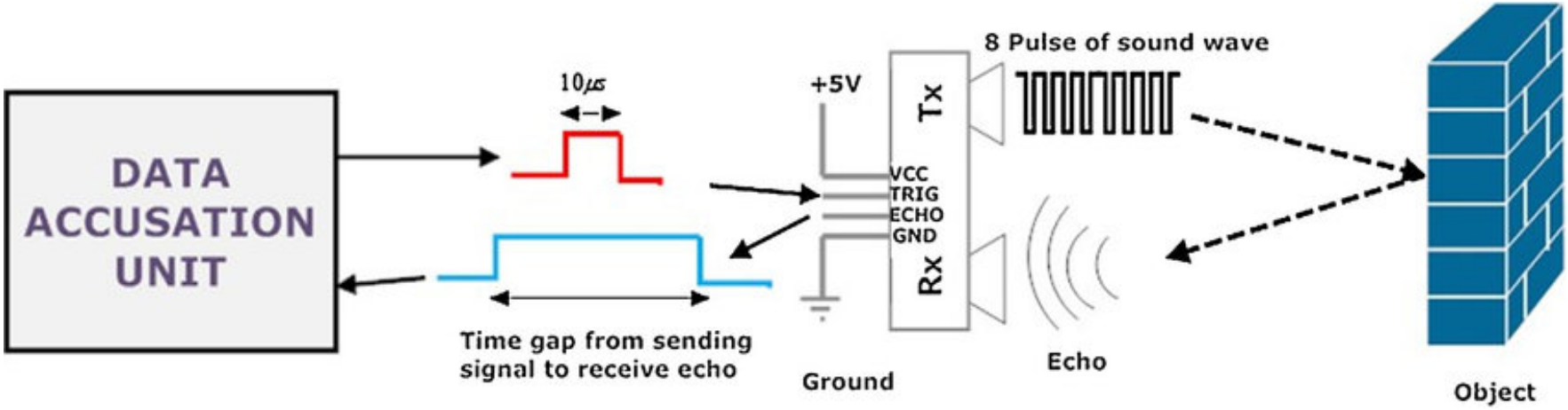


Distance Sensor -



Specificaties:

- Detectiebereik: 2-450cm
- Spanning: 5V
- Stroom: <2mA
- Resolutie: 3mm
- Sensor hoek: <15◦
- Afmetingen: 45x20mm



Distance Sensor -

Power: 5V

D1

D2

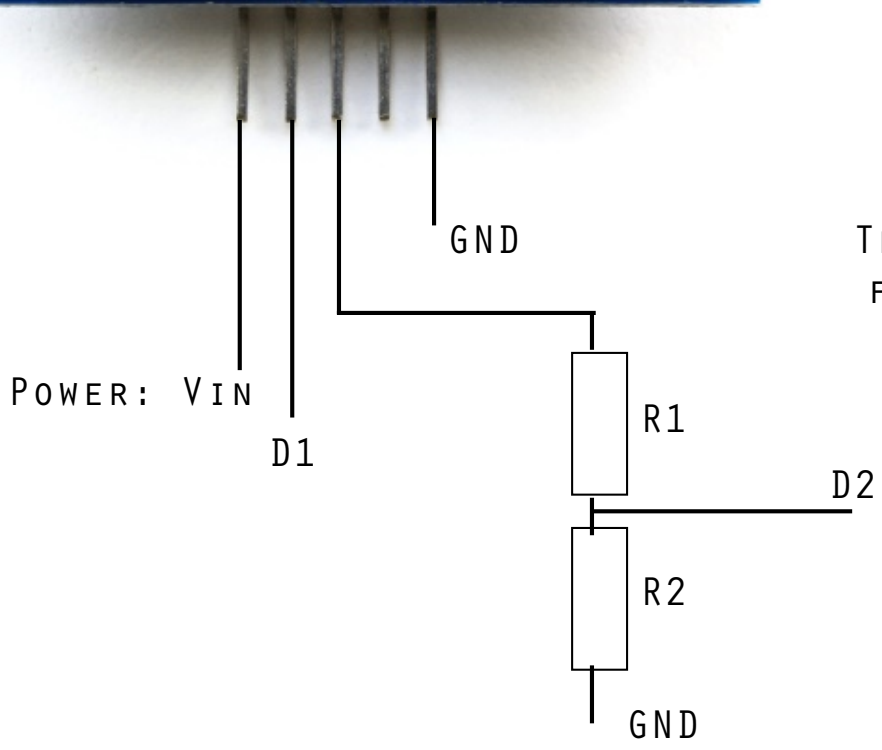
GND

The signal on pin Echo is 5V => not compatible with ESP!!!!!!!!!!

Specification:

- detect range: 2-450cm
- power supply: 5V
- consumption: <2mA
- resolution: 3mm
- Sensor angle: <15◦
- dimensions: 45x20mm

Distance Sensor



The signal on pin Echo is 5V =>
not compatible with ESP!!!!!!!!!!

We create a level shifter
using two resistors.

The level shifter change the voltage
from high voltage to lower voltage.

From 5V to 3.3V

R2 = 2*R1
R1 = 10K

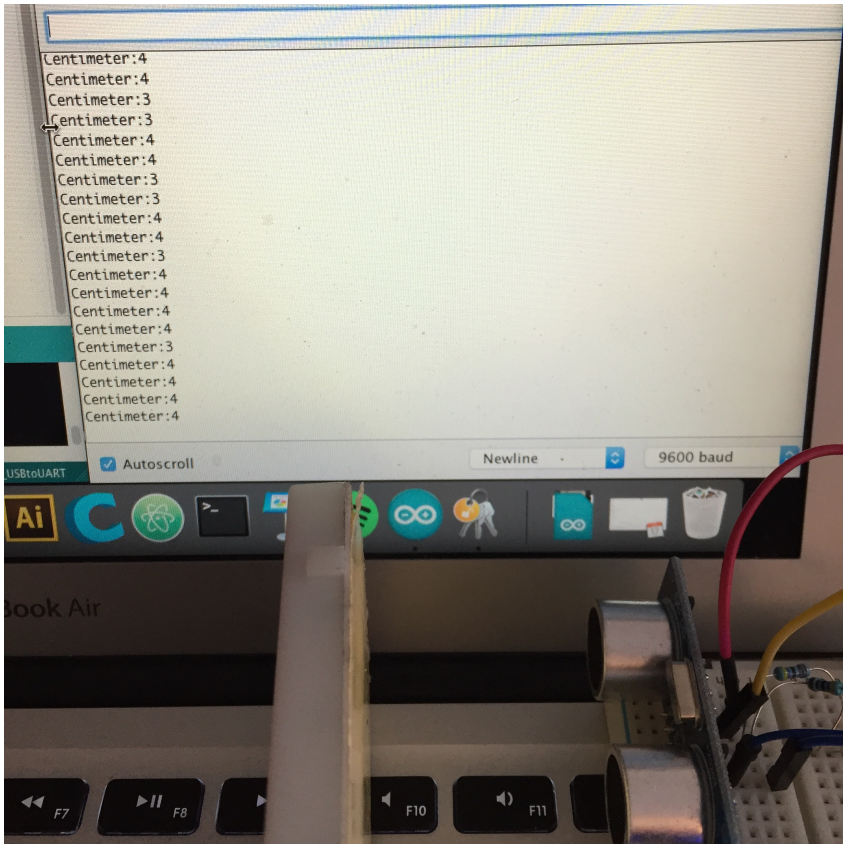
Distance Sensor

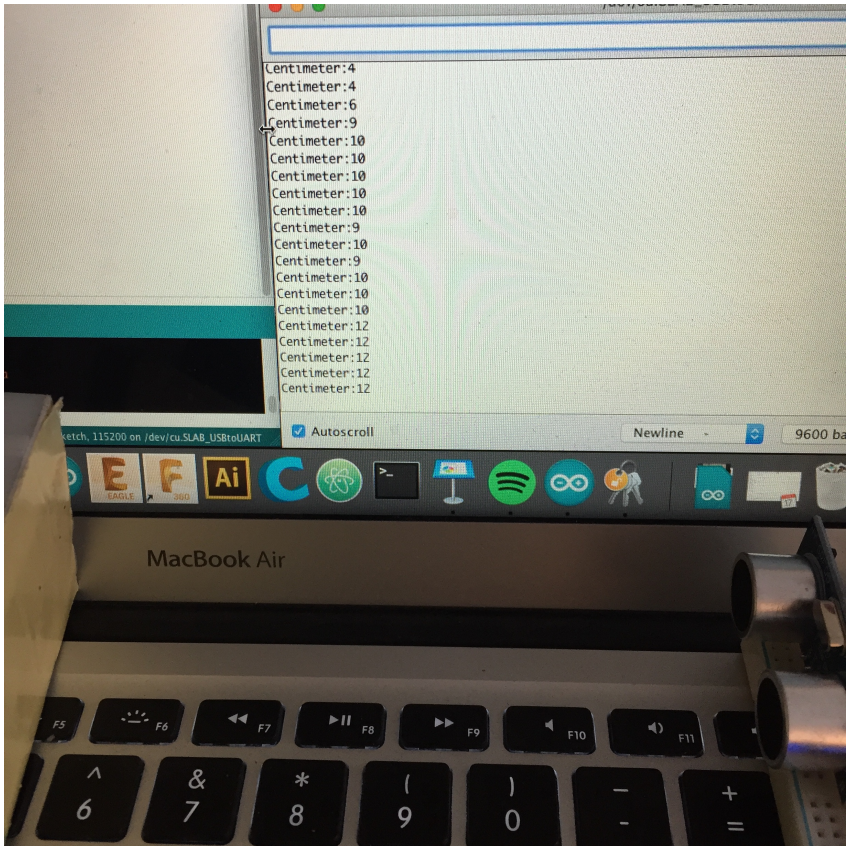
nodemcu_distance

```
// NodeMCU Pin D1 > TRIGGER | Pin D2 > ECHO
#define TRIGGER 5
#define ECHO    4

void setup() {

  Serial.begin (9600);
  pinMode(TRIGGER, OUTPUT);
  pinMode(ECHO, INPUT);
  pinMode(BUILTIN_LED, OUTPUT);
}

void loop() {

  long duration, distance;
  digitalWrite(TRIGGER, LOW);
  delayMicroseconds(2);

  digitalWrite(TRIGGER, HIGH);
  delayMicroseconds(10);

  digitalWrite(TRIGGER, LOW);
  duration = pulseIn(ECHO, HIGH);
  distance = (duration/2) / 29.1;


  Serial.print("Centimeter:");
  Serial.println(distance);
  delay(1000);
}
```

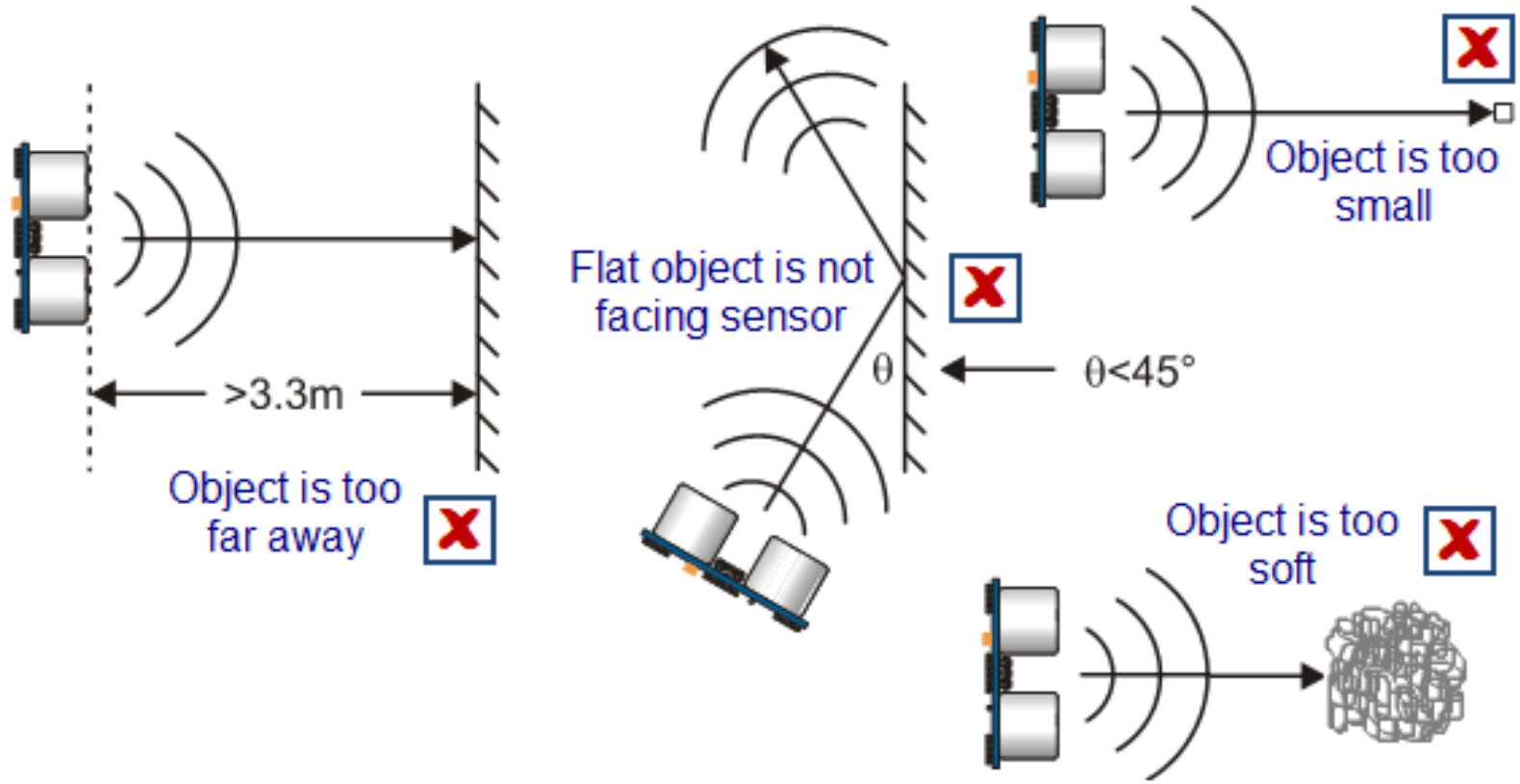
Distance Sensor

Code: nodemcu_distance.ino





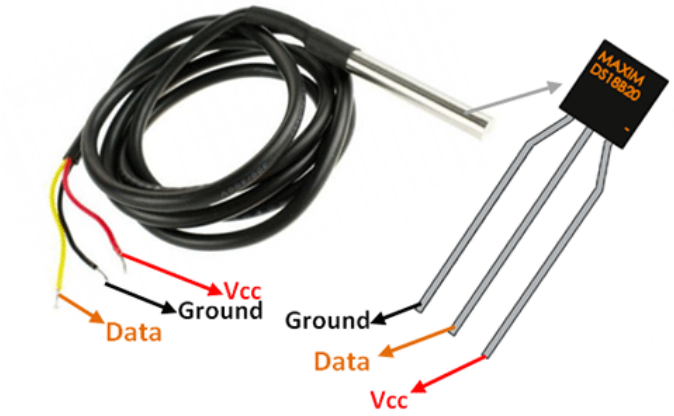
Distance Sensor



Temperature Sensor

Specifications:

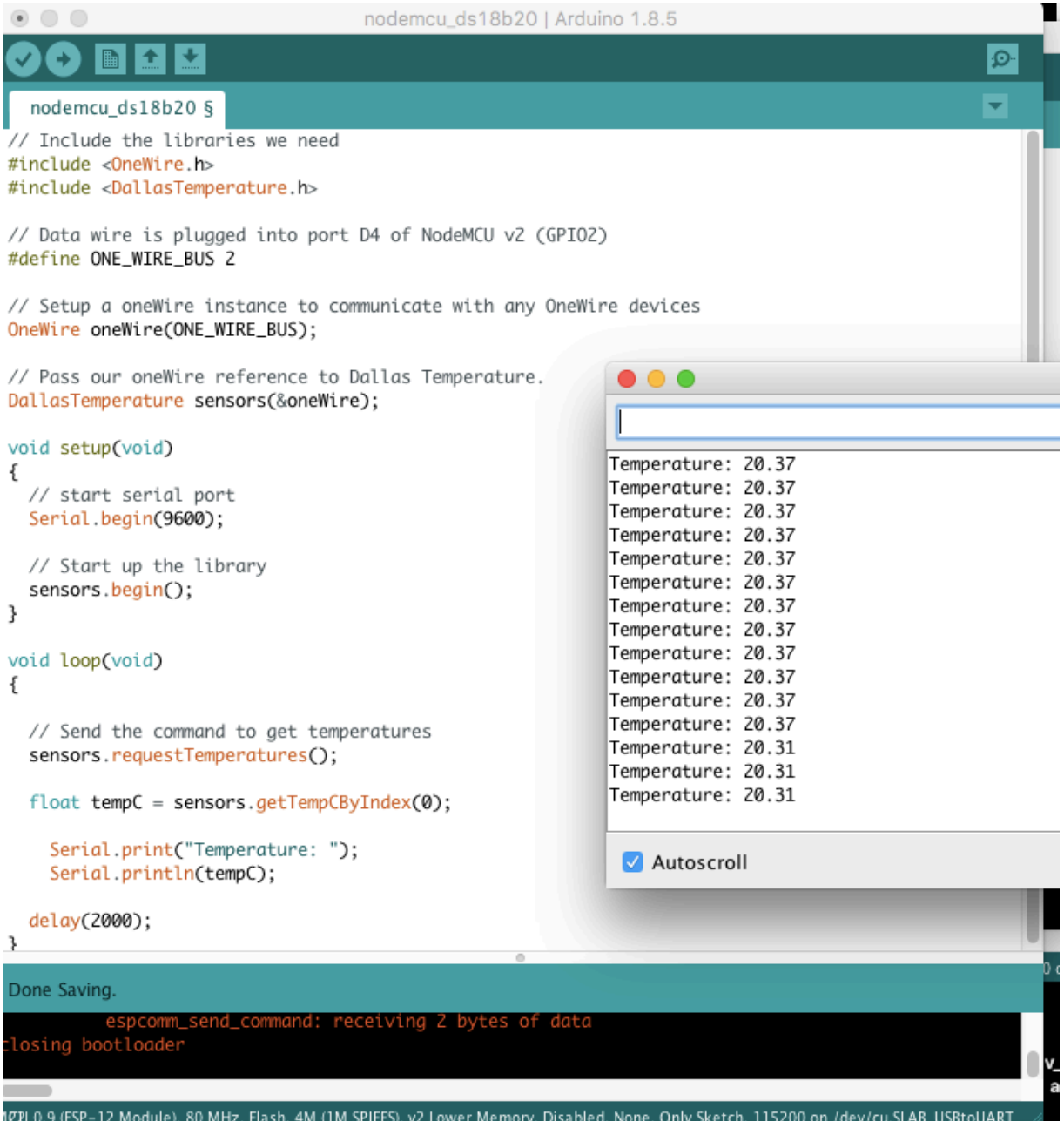
- Programmable Digital Temperature Sensor
- Communicates using 1-Wire method
- Operating voltage: 3V to 5V
- Temperature Range: -55°C to +125°C
- Accuracy: ±0.5°C
- Output Resolution: 9-bit to 12-bit (programmable)
- Unique 64-bit address enables multiplexing
- Conversion time: 750ms at 12-bit



Vcc: 3V
Data: D4
Ground: AND

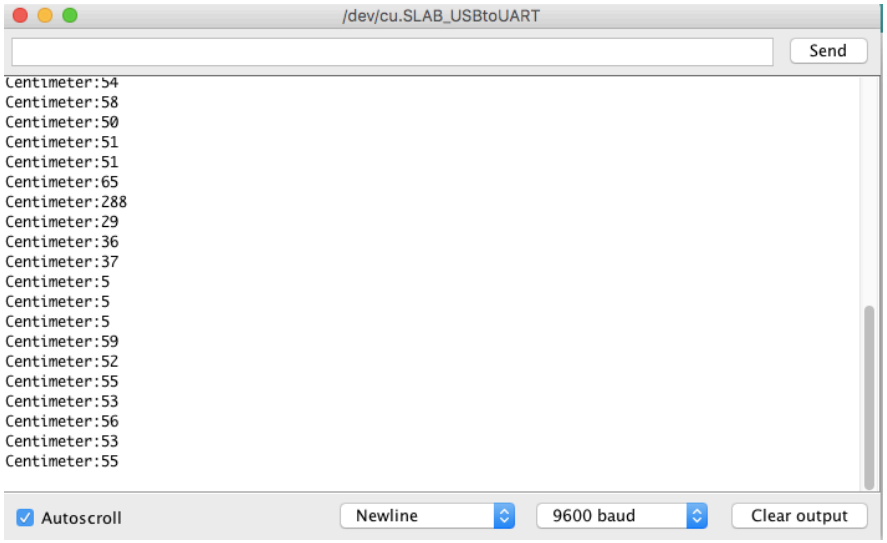
image: https://components101.com/sensors/ds18b20-temperature-sensor

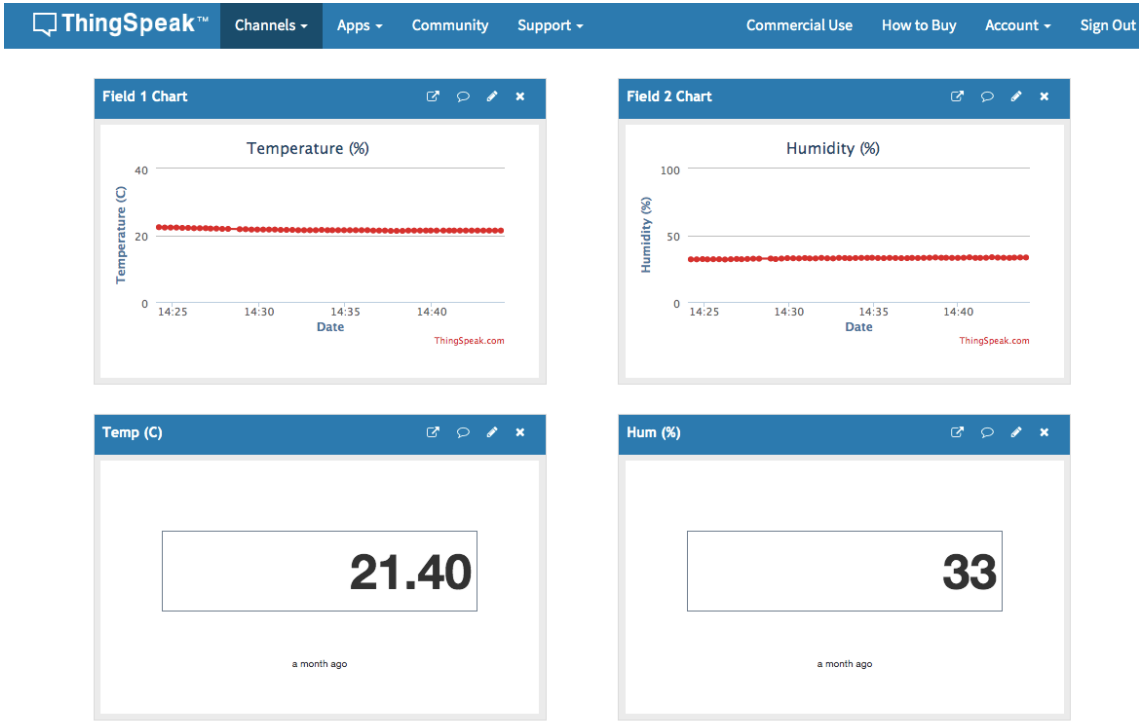
Temperature Sensor



From Serial Monitor To Cloud

On Serial Monitor ——————————————————> Cloud
temperature / humidity
water level / moisture / rain
distance





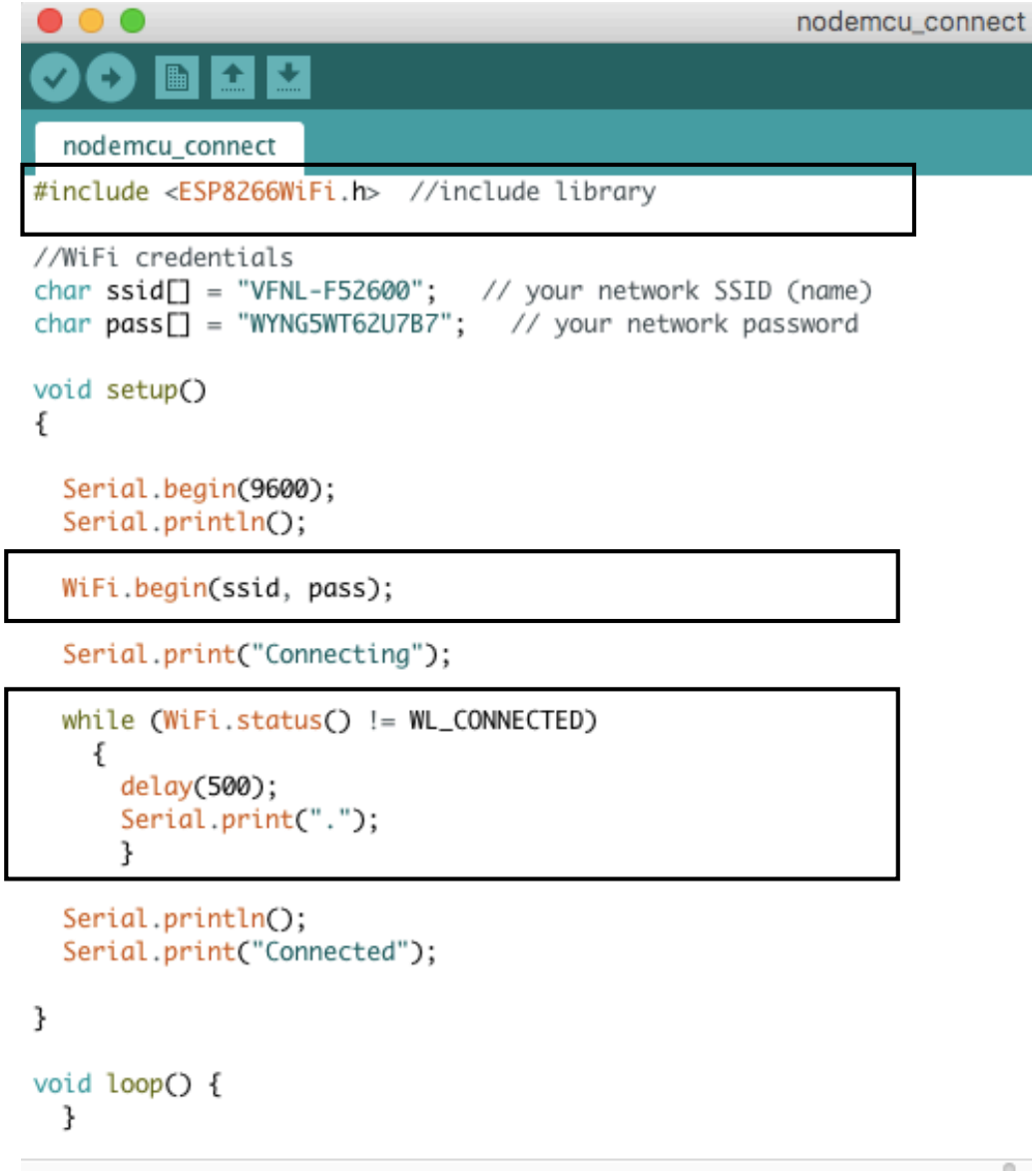
NodeMCU v2

- Connect to WiFi

- Configure ThingSpeak Channel
- Send fix data to ThingSpeak Channel

NodeMCU v2 - Connect to WiFi

Code: nodemcu_distance.ino

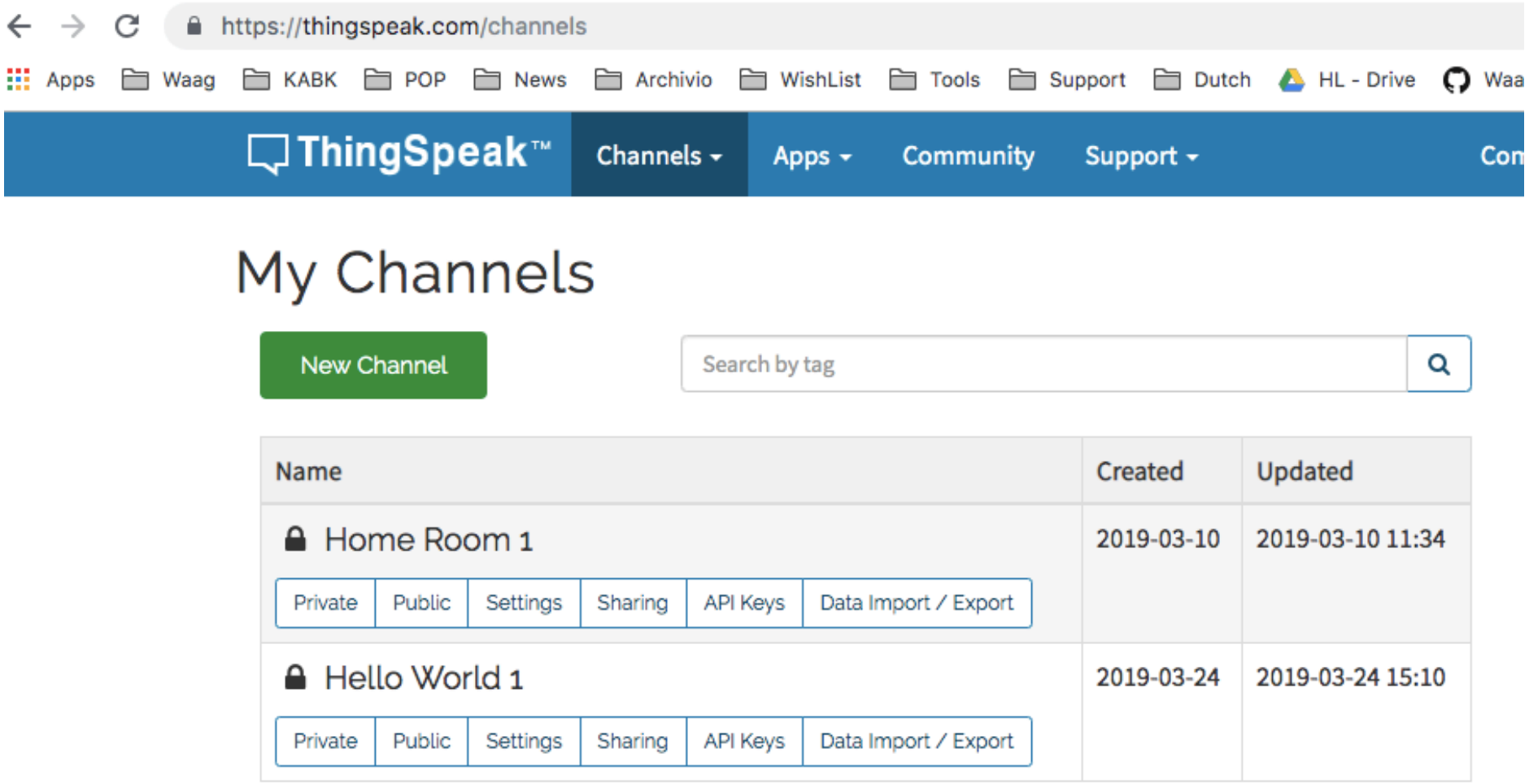


#include <ESP8266WiFi.h>

WiFi.begin(network, password)

WiFi.status()

Configure ThingSpeak Channel



NodeMCU v2 - Send fixed data to ThingSpeak Channel

nodemcu_uplink

```
#include <ESP8266WiFi.h> //Including library to use esp8266
#include "ThingSpeak.h"   //Including library to send data to thingspeak

//WiFi credentials
char ssid[] = "VFNL-F52600";   // your network SSID (name)
char pass[] = "WYNG5WT62U7B7";   // your network password

WiFiClient  client;

//ThingSpeak credentials credentials
unsigned long myChannelNumber = 739610;
char myWriteAPIKey[] = "W1ANLYD6BF60MPX2";

float data = 345;  //data that I sendf to thingspeak
int wait = 20000; // Time between two uplinks

void setup() {

  Serial.begin(9600);
  Serial.println();

  WiFi.begin(ssid, pass);

  Serial.print("Connecting");

  while (WiFi.status() != WL_CONNECTED)
    {
      delay(500);
      Serial.print(".");
      }

    Serial.println();
    Serial.println("\nConnected.");

  ThingSpeak.begin(client);  //Initialize ThingSpeak

}
```

NodeMCU v2 - Send fixed data to ThingSpeak Channel

```
void loop() {

  // set the ThingSpeak fields with the values
  ThingSpeak.setField(1, data);

  // send uplink
  ThingSpeak.writeFields(myChannelNumber, myWriteAPIKey);

  // wait before the next measurment
  delay(wait);

}
```

Done Saving.

Assignment

Share the hardware in groups of 2 people, one group of 3

Create a channel called: KABK_ex1

Read the sensor you used in class with NodeMCU and send those to channel KABK_ex1.
Send the data every minute.

Keep the sensor running in your house for one week (or few days)

Document the process in your blog:
—> schematic of the circuit
—> code (readable)
—> pictures
—> screenshots of ThingSpeak platform