

ELECTRONICS 1

ELECTRONICS FOR INTERACTIVE MEDIA DESIGN

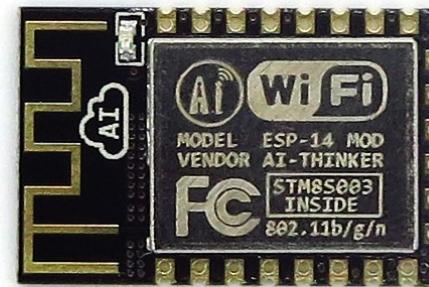
EMMA PARESCHI

WIFI MODULE: ESP8266

DIFFERENT VERSIONS OF THE SAME MODULE (CHIP):



ESP-12E MODULE



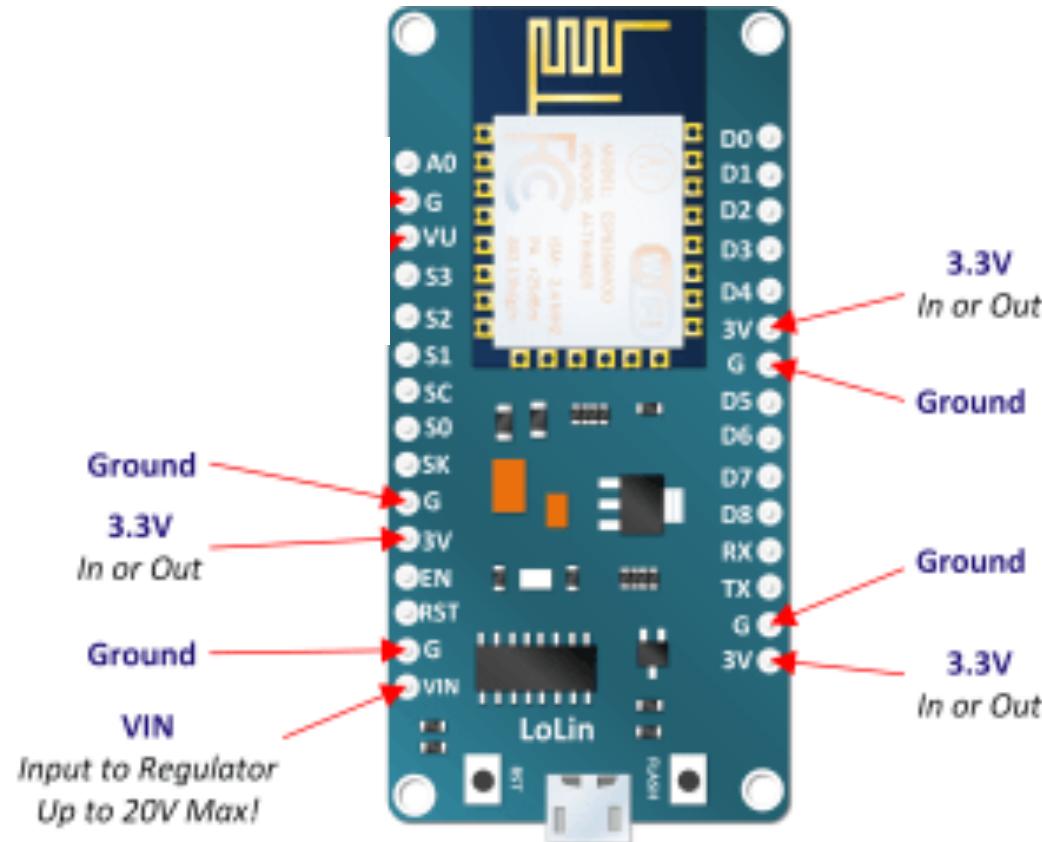
- Open-source firmware
- Arduino IDE compatible
- Programmable
- Low cost (1.15€)
- Simple
- Smart
- WI-FI enabled

NODEMCU
VERSION2



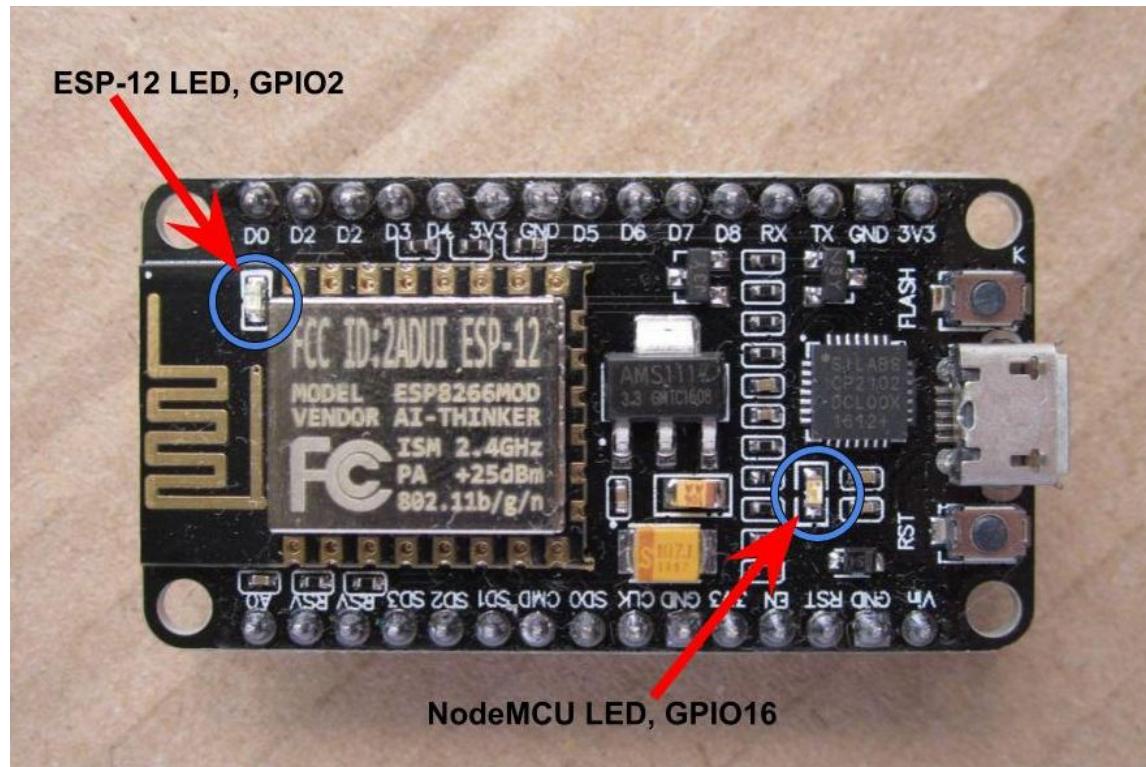
NODEMCU IS A DEVELOPMENT KIT,
WITH OPEN-SOURCE FIRMWARE, THAT
HELPS YOU TO PROTOTYPE YOUR IOT
PRODUCT.

NODEMCU v2 - POWER PIN



ESP8266 IS A 3.3V MODULE!!!!

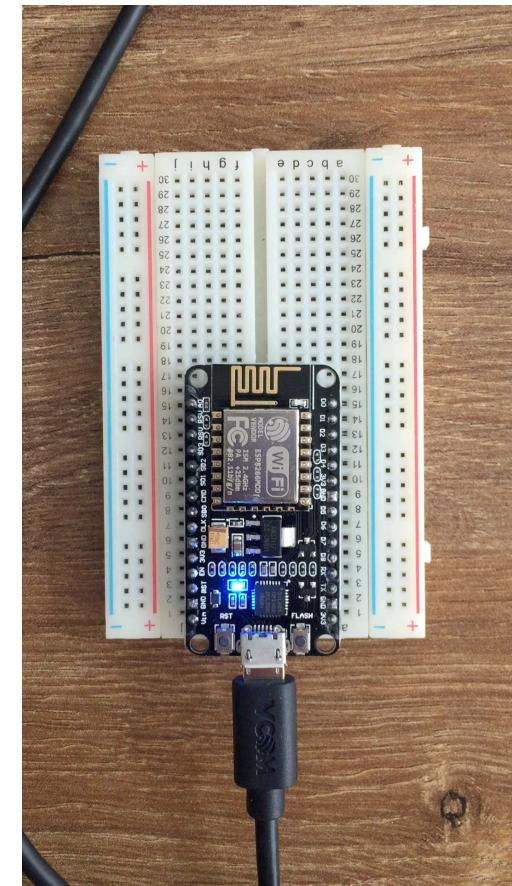
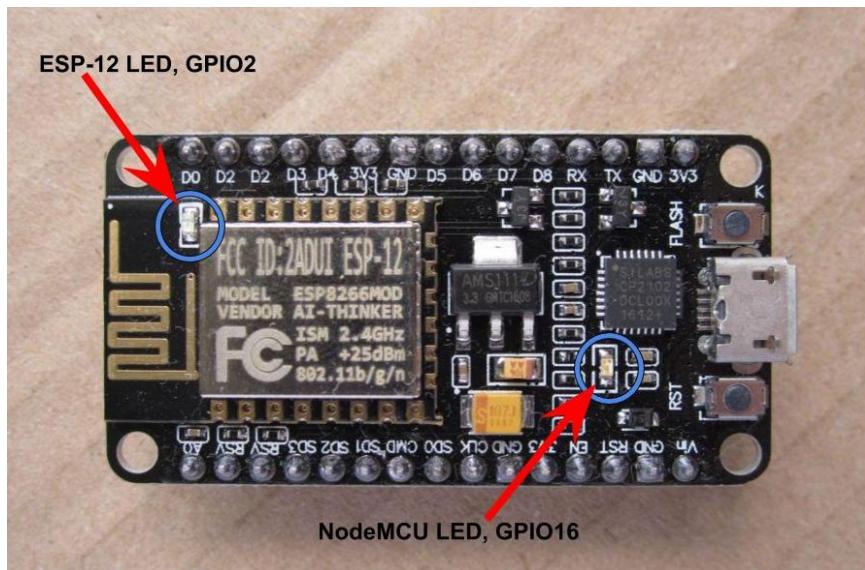
NODEMCU v2 - ONBOARD LEDs



THESE TWO LEDs ARE HANDY WHEN
USED AS FEEDBACK.

EX: EVERYTIME YOU SEND A MESSAGE TO THE CLOUD =>
=> THE LED BLINKS

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

NODEMCU v2 - ARDUINO IDE

Download and Install Drivers

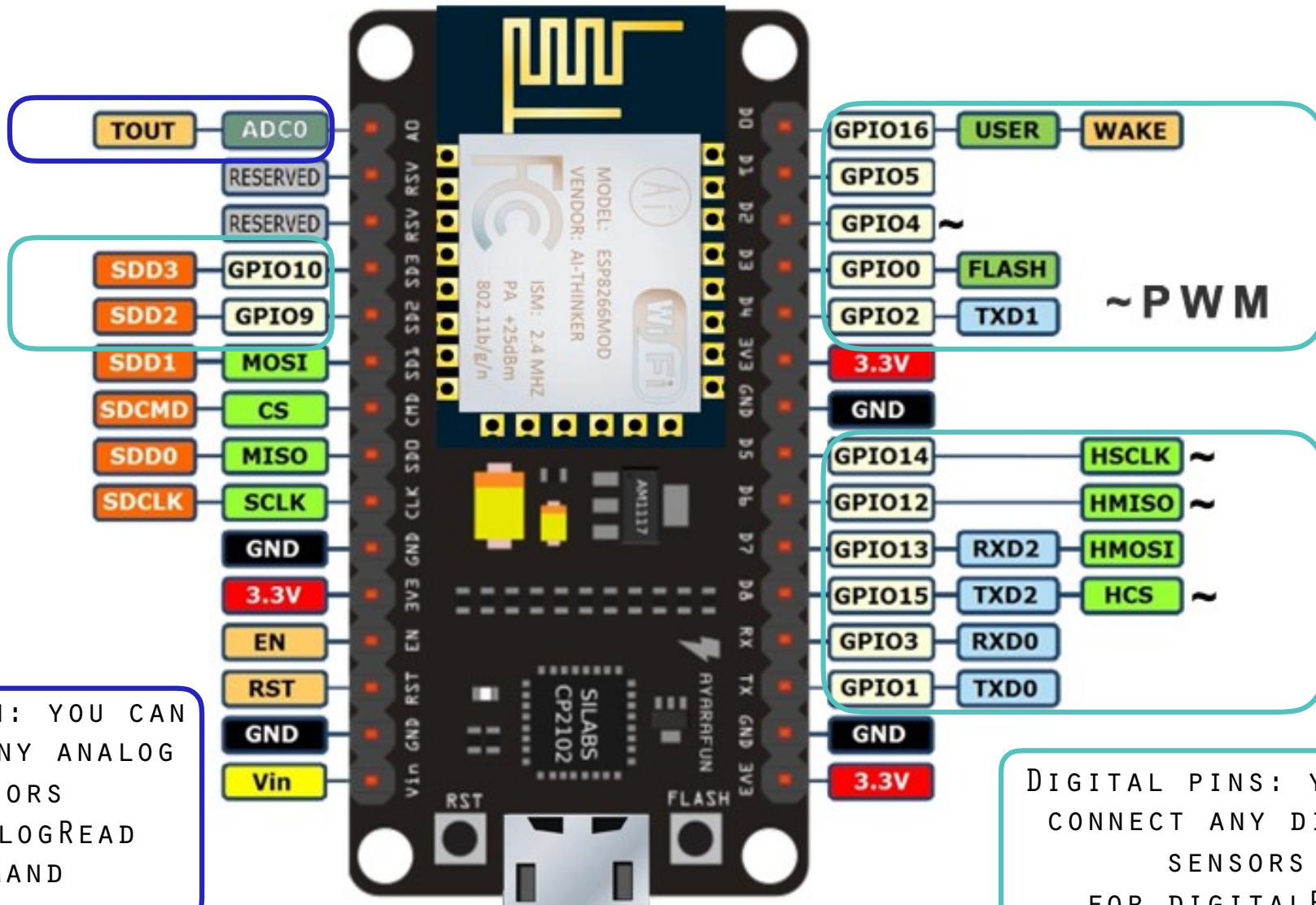
<https://www.silabs.com/products/development-tools/software/usb-to-uart-bridge-vcp-drivers>

Install the board in Arduino IDE:

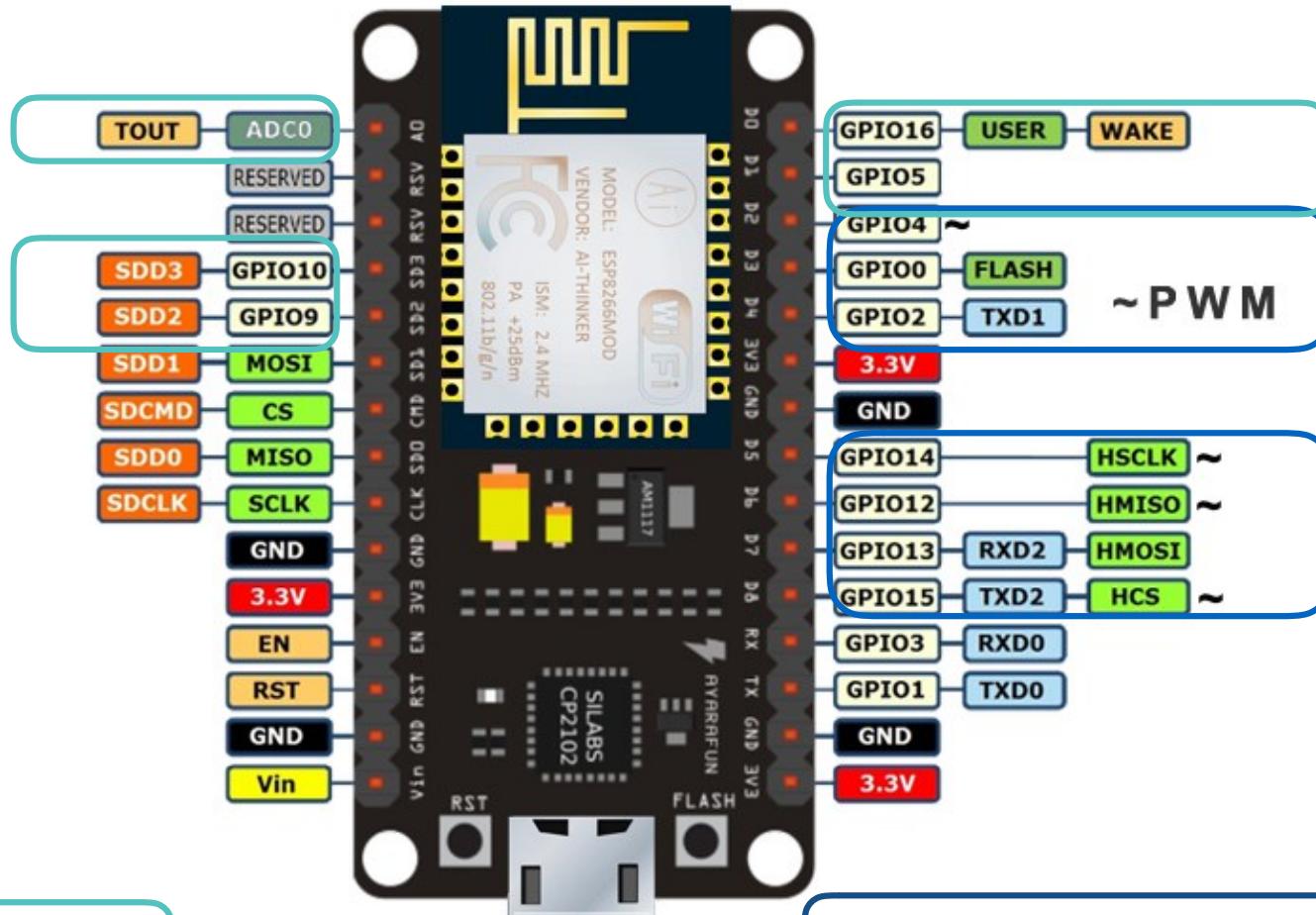
- > in preference window: http://arduino.esp8266.com/stable/package_esp8266com_index.json
- > from boards manager: esp8266



NODEMCU v2 - IO PINS - INPUT



NODEMCU v2 - IO PINS - OUTPUT



DIGITAL OUTPUT PINS:
YOU CAN CONNECT ANY
DIGITAL OUTPUT
FOR DIGITALWRITE
COMMAND

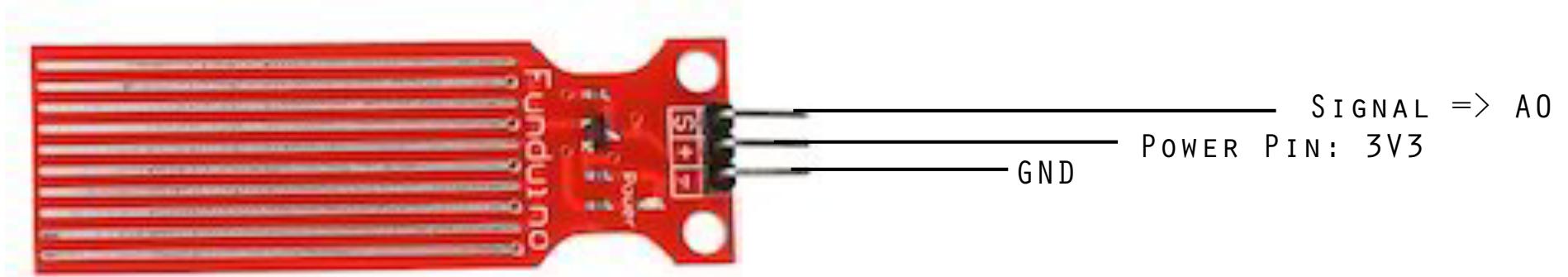
ANALOG OUTPUT PINS (PWM): YOU
CAN CONNECT ANY ANALOG
SENSORS
FOR ANALOGWRITE COMMAND (EX:
CONTROL INTENSITY OF A LED)

SENSORS

Kind of sensors:

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

WATER LEVEL SENSOR (ANALOG)



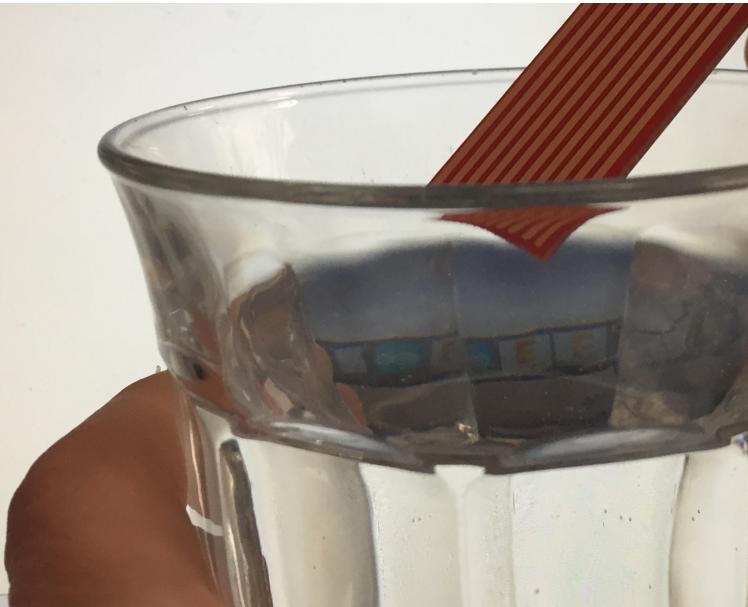
Power supply: 3.3V - 5V

Output: analog

WATER LEVEL SENSOR

Code: nodemcu_waterlevel.ino

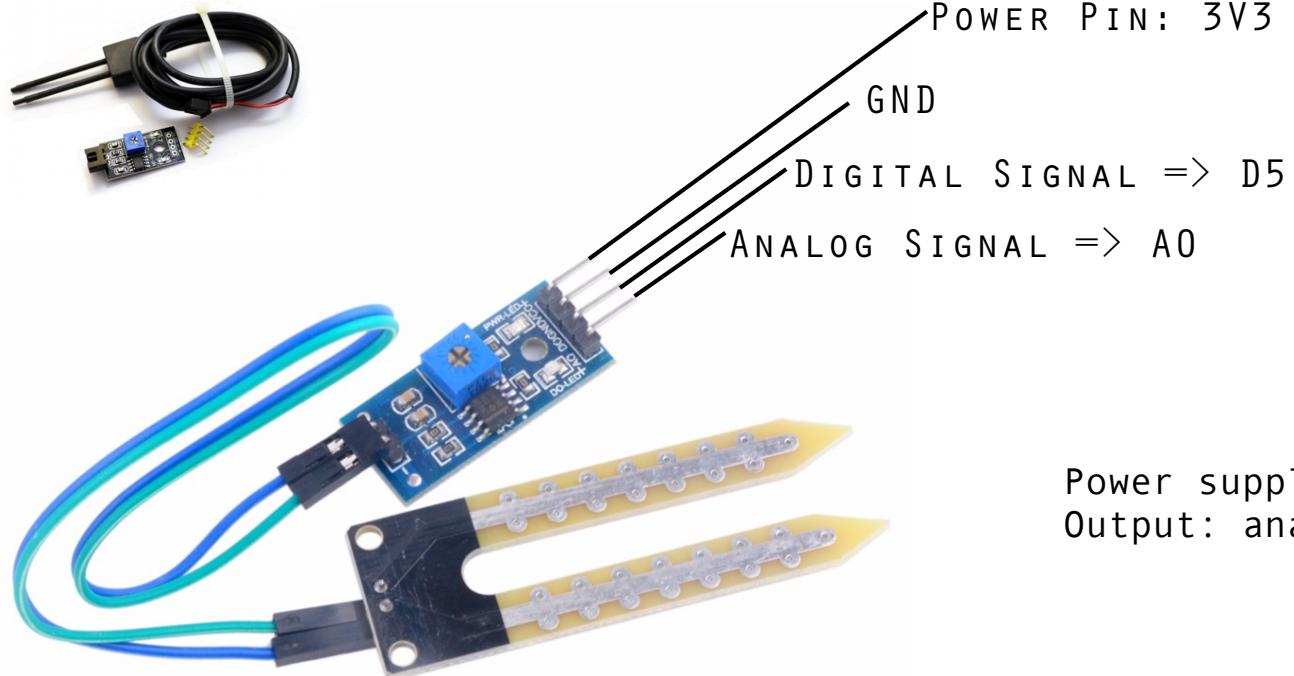
```
Water level is: 13  
Water level is: 5  
Water level is: 9  
Water level is: 9  
Water level is: 14  
Water level is: 5  
Water level is: 5  
Water level is: 14  
Water level is: 13  
Water level is: 5  
Water level is: 5  
Water level is: 5  
Water level is: 14  
Water level is: 14  
Water level is: 14  
Water level is: 5  
Water level is: 13  
Water level is: 14  
Water level is: 13  
Water level is: 13  
Water level is: 14  
Water level is: 12  
Water level is: 13  
Water level is: 13  
Water level is: 13  
Water level is: 5  
Water level is: 5  
Water level is: 9
```



```
Water level is: 13  
Water level is: 5  
Water level is: 5  
Water level is: 5  
Water level is: 14  
Water level is: 14  
Water level is: 5  
Water level is: 13  
Water level is: 14  
Water level is: 13  
Water level is: 13  
Water level is: 13  
Water level is: 14  
Water level is: 12  
Water level is: 13  
Water level is: 13  
Water level is: 5  
Water level is: 5  
Water level is: 9  
Water level is: 13  
Water level is: 87  
Water level is: 106  
Water level is: 126  
Water level is: 130
```



SOIL MOISTURE SENSOR (ANALOG/DIGITAL)



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

With Analog signal you check the level

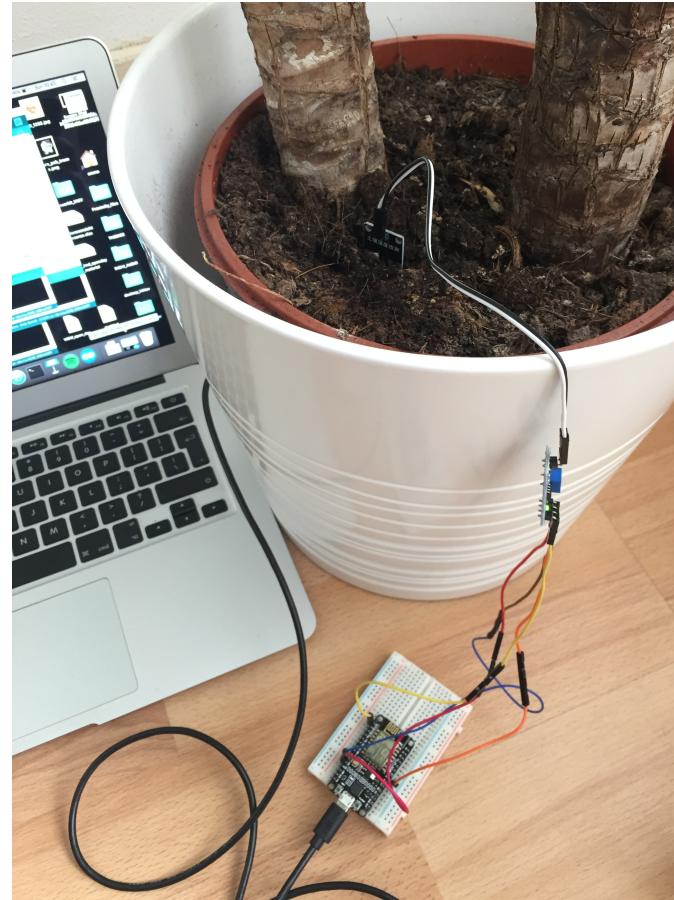
With Signal you check if moisture is higher or lower of a fixed threshold.
The threshold is fixed by the potentiometer on board.

SOIL MOISTURE SENSOR

Code: nodemcu_moisturelevel.ino

```
Moisture level is: 749  
Moisture threshold: 1023  
Moisture level is: 759  
Moisture threshold: 1023  
Moisture level is: 759  
Moisture threshold: 1023  
Moisture level is: 762  
Moisture threshold: 1023  
Moisture level is: 762  
Moisture threshold: 1023  
Moisture level is: 762  
Moisture threshold: 1023  
Moisture level is: 758  
Moisture threshold: 1023  
Moisture level is: 761  
Moisture threshold: 1023
```

Autoscroll



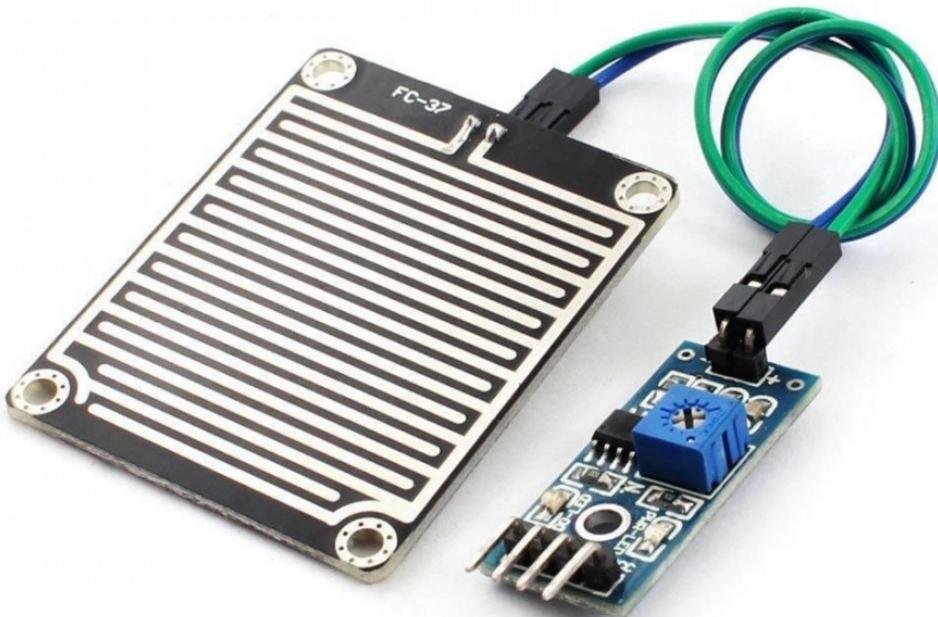
HIGH READING => DRY

```
Moisture level is: 63/  
Moisture threshold: 1023  
Moisture level is: 345  
Moisture threshold: 0  
Moisture level is: 366  
Moisture threshold: 0  
Moisture level is: 358  
Moisture threshold: 0  
Moisture level is: 367  
Moisture threshold: 0  
Moisture level is: 370  
Moisture threshold: 0  
Moisture level is: 387  
Moisture threshold: 0  
Moisture level is: 387  
Moisture threshold: 0
```

Autoscroll

LOW READING => MOIST

RAIN SENSOR (ANALOG/DIGITAL)



It works like the Moisture Sensor

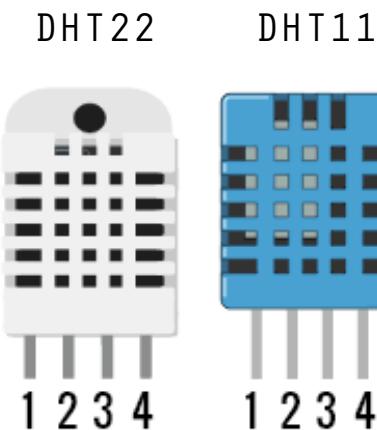
Power supply: 3.3V - 5V

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)

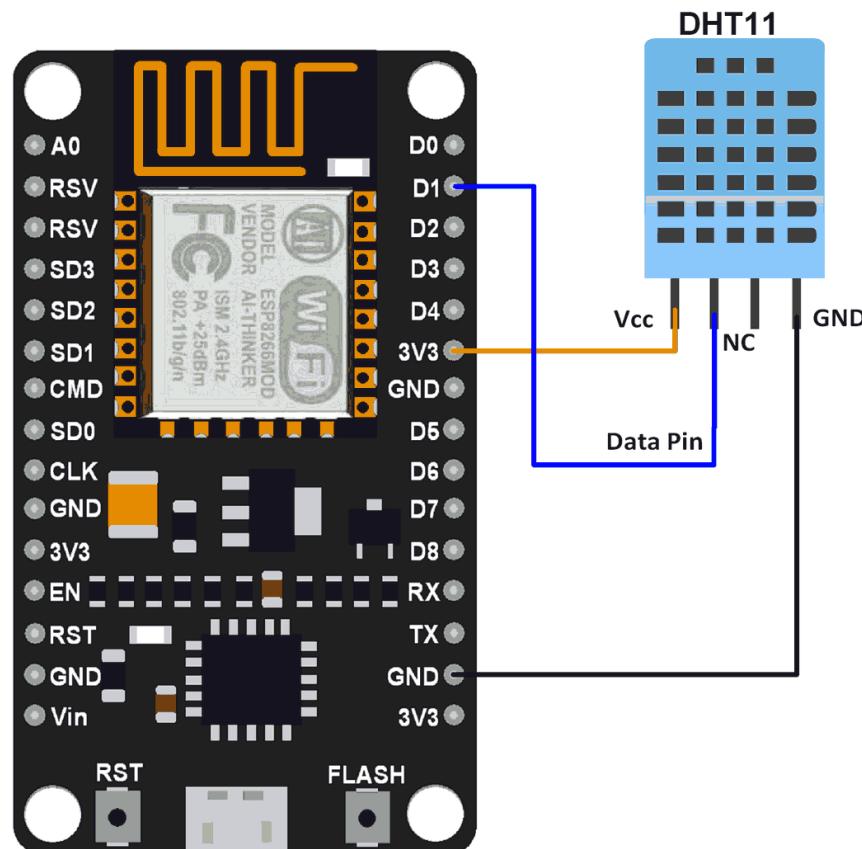


1 = VCC
2 = DATA
3 = NC
4 = GND

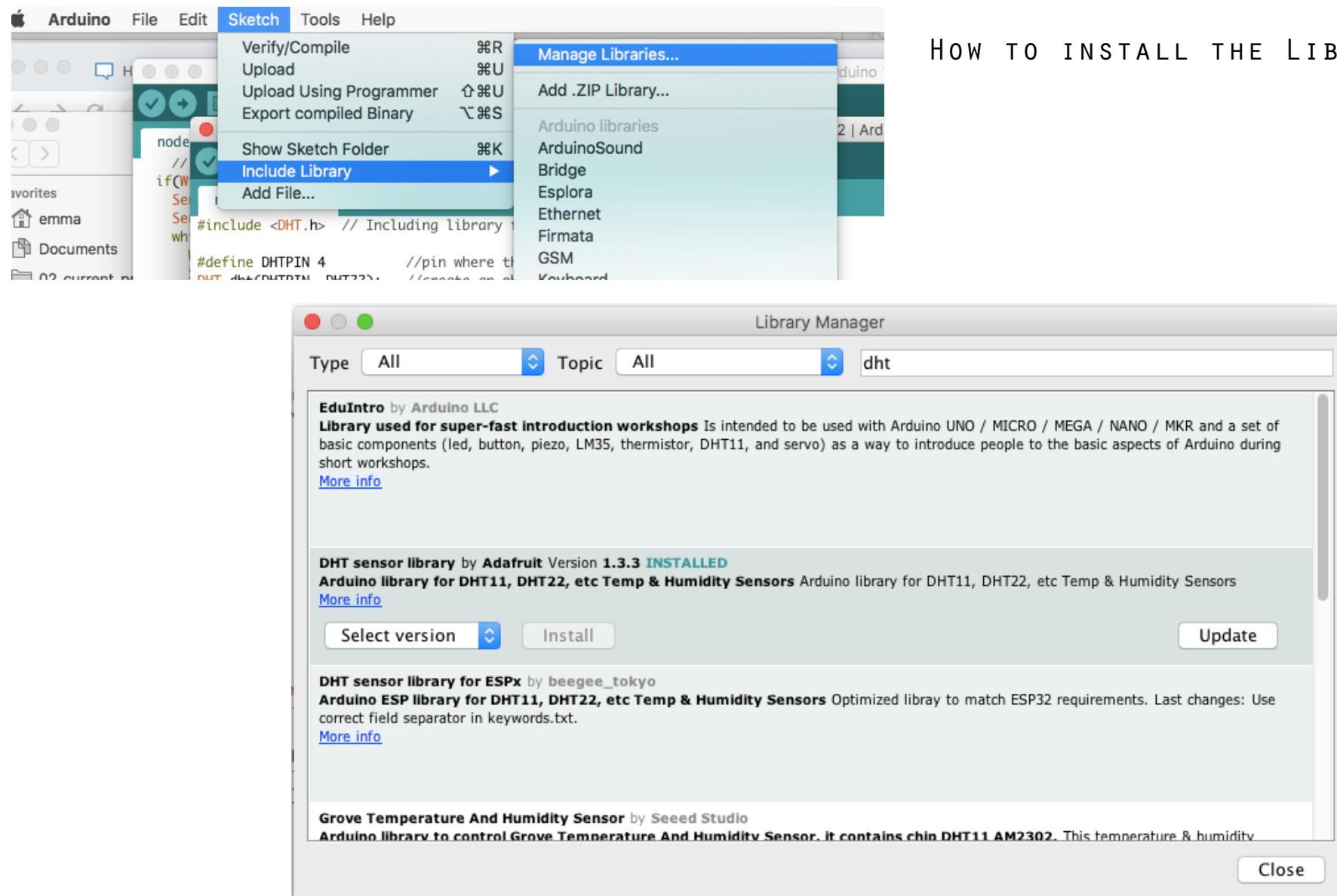
DHT22
VCC 3-5V
Temperature Range-40~80 C
Temp. Accuracy+/- 0.5C
Humidity Range0 ~ 100%
Humidity Accuracy+/- 2%

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

The screenshot shows the Arduino IDE interface with two main windows. The left window is titled "nodemcu_dht | Arduino 1.8.5" and displays the following C++ code:

```
#include <DHT.h> // Including library for dht

#define DHTPIN 5      //pin where the dht11 is connected
DHT dht(DHTPIN, DHT11); //create an object dht, definin pin and model

float temp = 0; //variable where to save temperture
float hum = 0; //variable where to save humidity

int wait = 10000;

void setup() {
    Serial.begin(9600); // Initialize serial
    dht.begin(); //Initialize DHT
}

void loop() {
    temp = dht.readTemperature(); //save the temperature in the variable temp
    hum = dht.readHumidity(); //save the humidity in the variable hum

    Serial.print("Temperature: "); Serial.print(temp); Serial.println(" °C"); //print on serial temperature
    Serial.print("Humidity: "); Serial.print(hum); Serial.println(" %"); //print on serial humidity
    Serial.println("");

    delay(wait);
}
```

The right window is a terminal window titled "/dev/cu.SLAB_USBtoUART" showing the output of the serial print statements. The output is as follows:

```
Humidity: 44.00 %
Temperature: 20.90 °C
Humidity: 44.00 %

Temperature: 20.90 °C
Humidity: 44.00 %

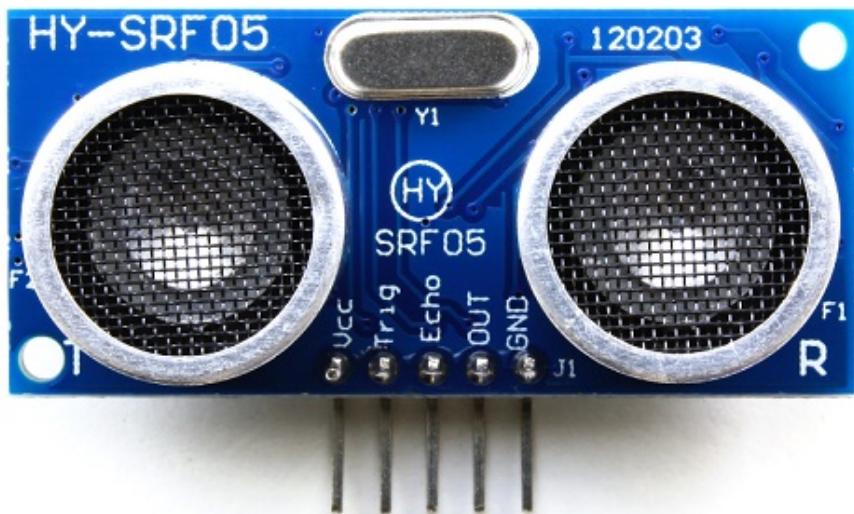
Temperature: 21.00 °C
Humidity: 67.00 %

Temperature: 21.00 °C
Humidity: 52.00 %

Autoscroll Newline 9600 baud Clear output
```

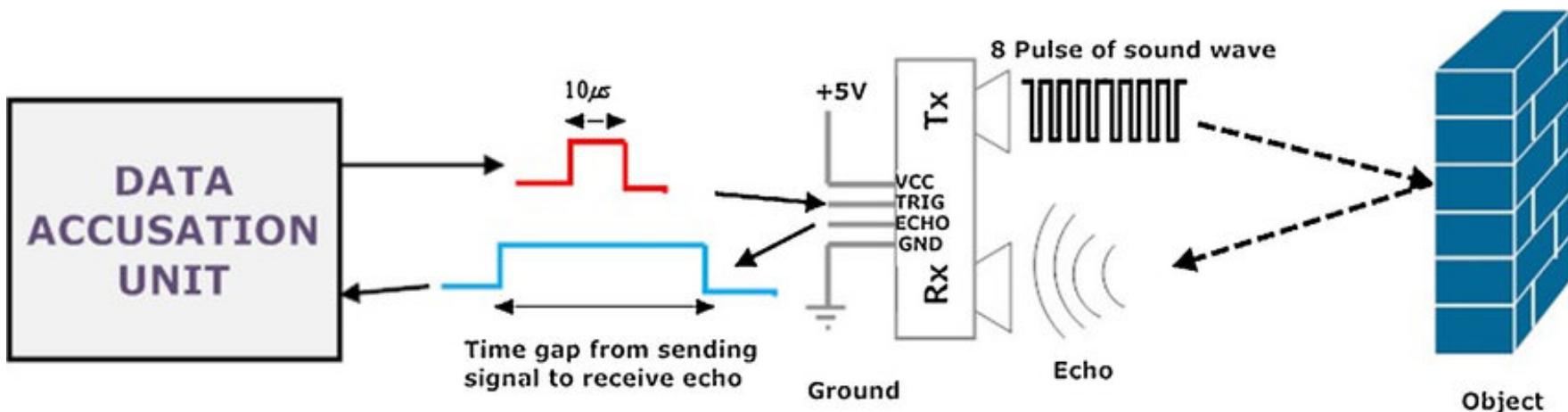
At the bottom of the terminal window, there is a message: "Done Saving." followed by "closing bootloader".

DISTANCE SENSOR -

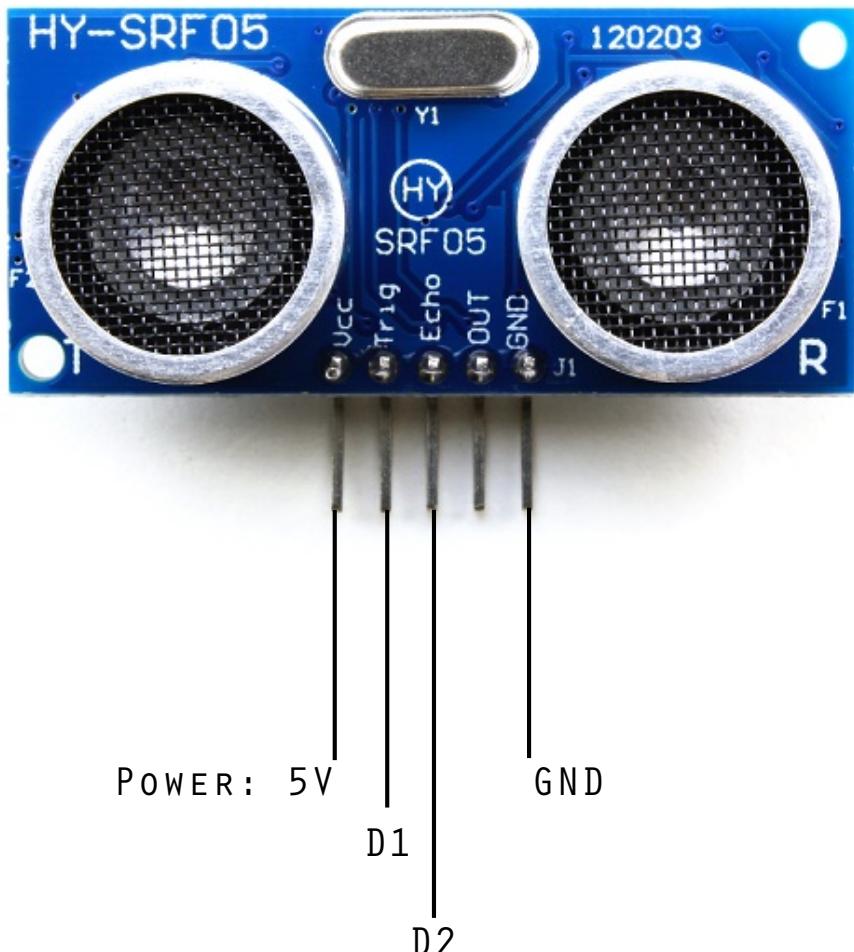


Specificaties:

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



DISTANCE SENSOR -

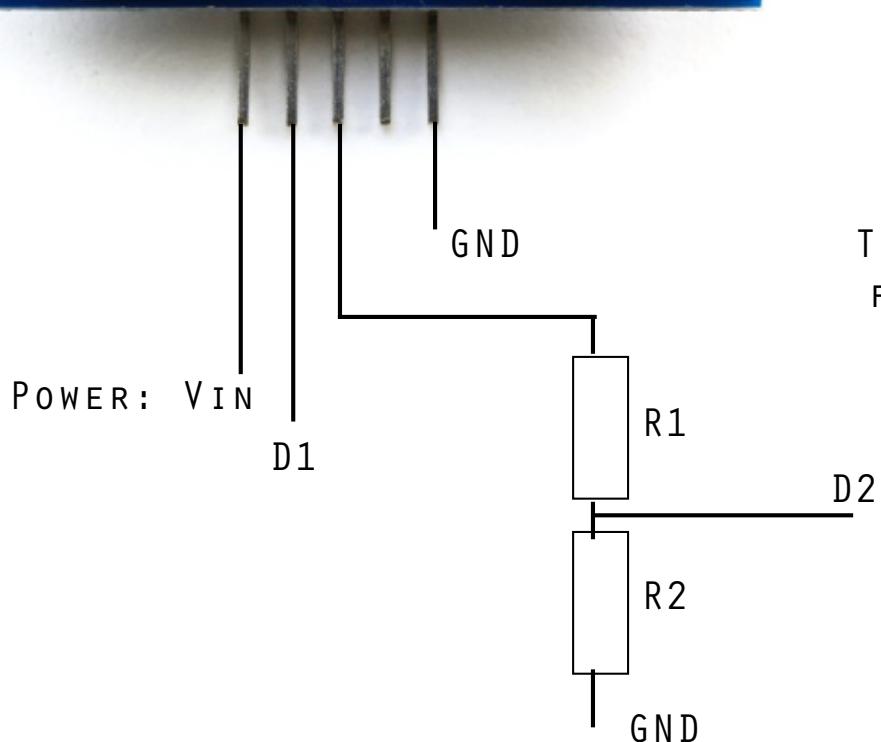


THE SIGNAL ON PIN ECHO IS 5V =>
NOT COMPATIBLE WITH ESP!!!!!!!

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

$$R_2 = 2 * R_1$$
$$R_1 = 10K$$

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);

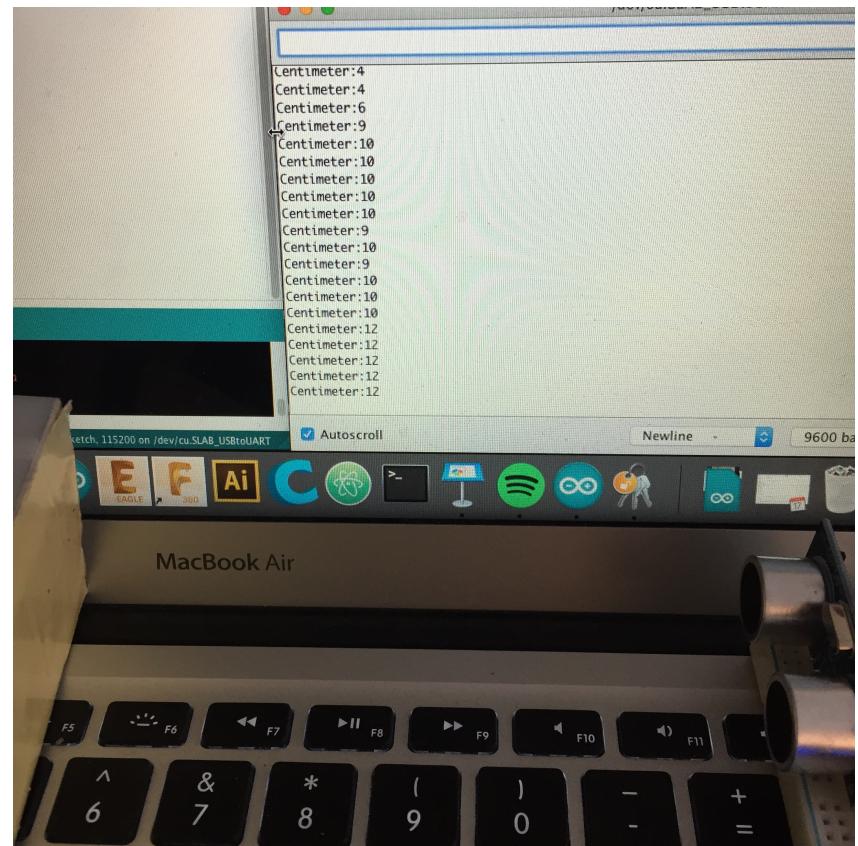
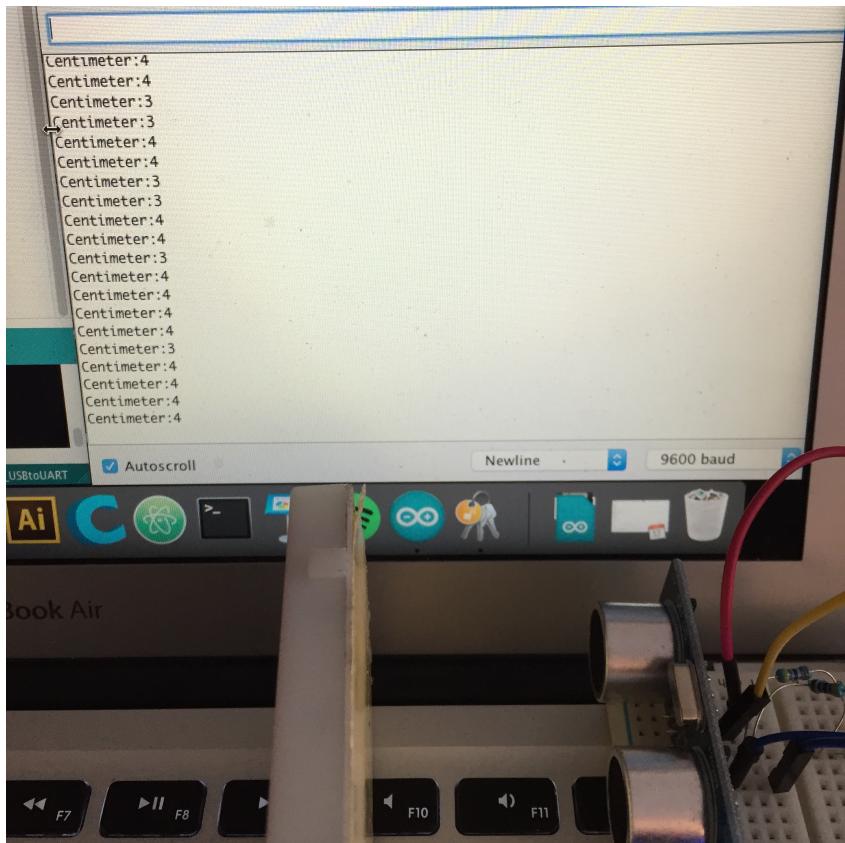
    digitalWrite(TRIGGER, HIGH);
    delayMicroseconds(10);

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

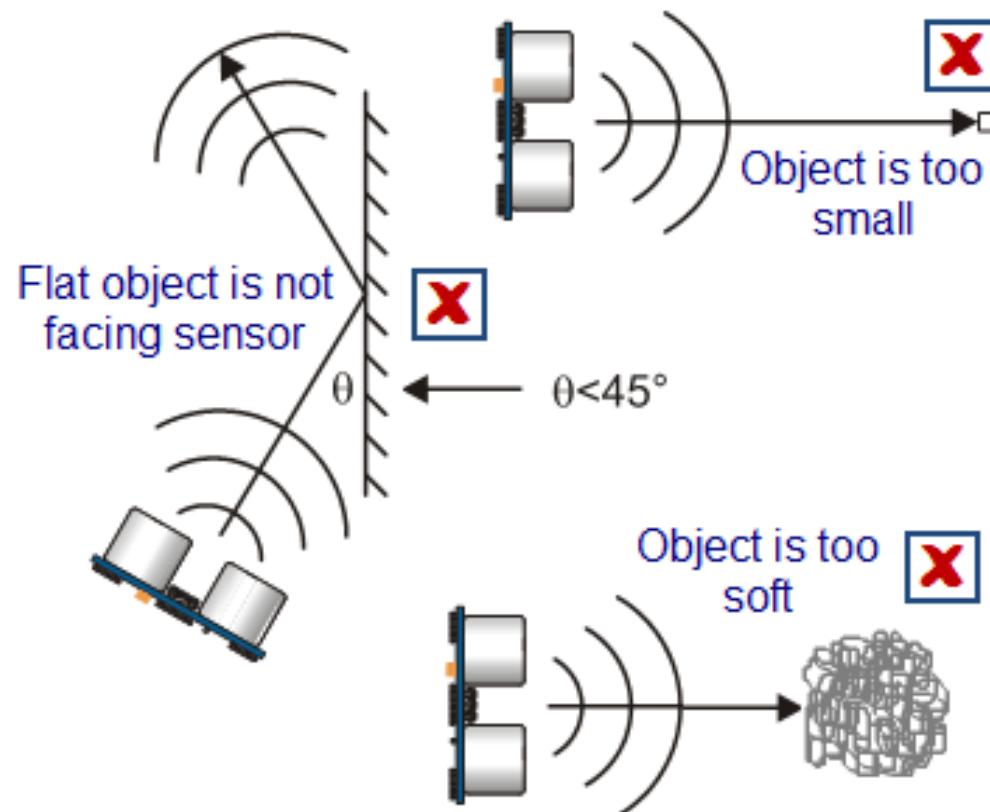
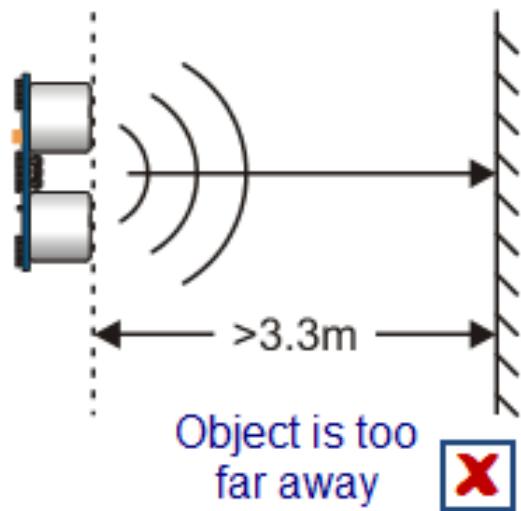
    Serial.print("Centimeter:");
    Serial.println(distance);
    delay(1000);
}
```

DISTANCE SENSOR

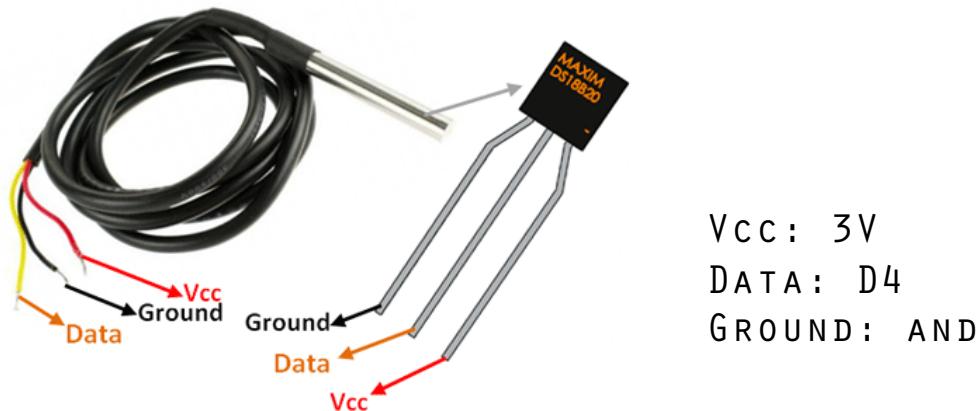
Code: nodemcu_distance.ino



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: $\pm 0.5^\circ\text{C}$
- 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

TEMPERATURE SENSOR

The screenshot shows the Arduino IDE interface with a sketch titled "nodemcu_ds18b20". The code uses the OneWire and DallasTemperature libraries to read temperatures from a DS18B20 sensor connected to port D4 (GPIO02) of a NodeMCU v2 module. The sketch initializes the serial port at 9600 bps, starts the OneWire library, and then enters a loop where it sends a command to request temperatures and prints the result. The serial monitor window shows repeated temperature readings of 20.37°C and 20.31°C.

```
// Include the libraries we need
#include <OneWire.h>
#include <DallasTemperature.h>

// Data wire is plugged into port D4 of NodeMCU v2 (GPIO02)
#define ONE_WIRE_BUS 2

// Setup a oneWire instance to communicate with any OneWire devices
OneWire oneWire(ONE_WIRE_BUS);

// Pass our oneWire reference to Dallas Temperature.
DallasTemperature sensors(&oneWire);

void setup(void)
{
    // start serial port
    Serial.begin(9600);

    // Start up the library
    sensors.begin();
}

void loop(void)
{
    // Send the command to get temperatures
    sensors.requestTemperatures();

    float tempC = sensors.getTempCByIndex(0);

    Serial.print("Temperature: ");
    Serial.println(tempC);

    delay(2000);
}
```

Done Saving.
espcomm_send_command: receiving 2 bytes of data
closing bootloader

Temperature: 20.37
Temperature: 20.31
Temperature: 20.31
Temperature: 20.31

Autoscroll

FROM SERIAL MONITOR To CLOUD

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

The image shows a comparison between a Serial Monitor window and a ThingSpeak Cloud dashboard.

Serial Monitor Window: On the left, a Mac OS X terminal window titled "/dev/cu.SLAB_USBtoUART" displays a series of "Centimeter" values ranging from 50 to 59. It includes a "Send" button and a scroll bar. At the bottom, there are buttons for "Autoscroll", "Newline", "9600 baud", and "Clear output".

ThingSpeak Cloud Dashboard: On the right, the ThingSpeak website is shown. The top navigation bar includes "ThingSpeak™", "Channels", "Apps", "Community", "Support", "Commercial Use", "How to Buy", "Account", and "Sign Out".

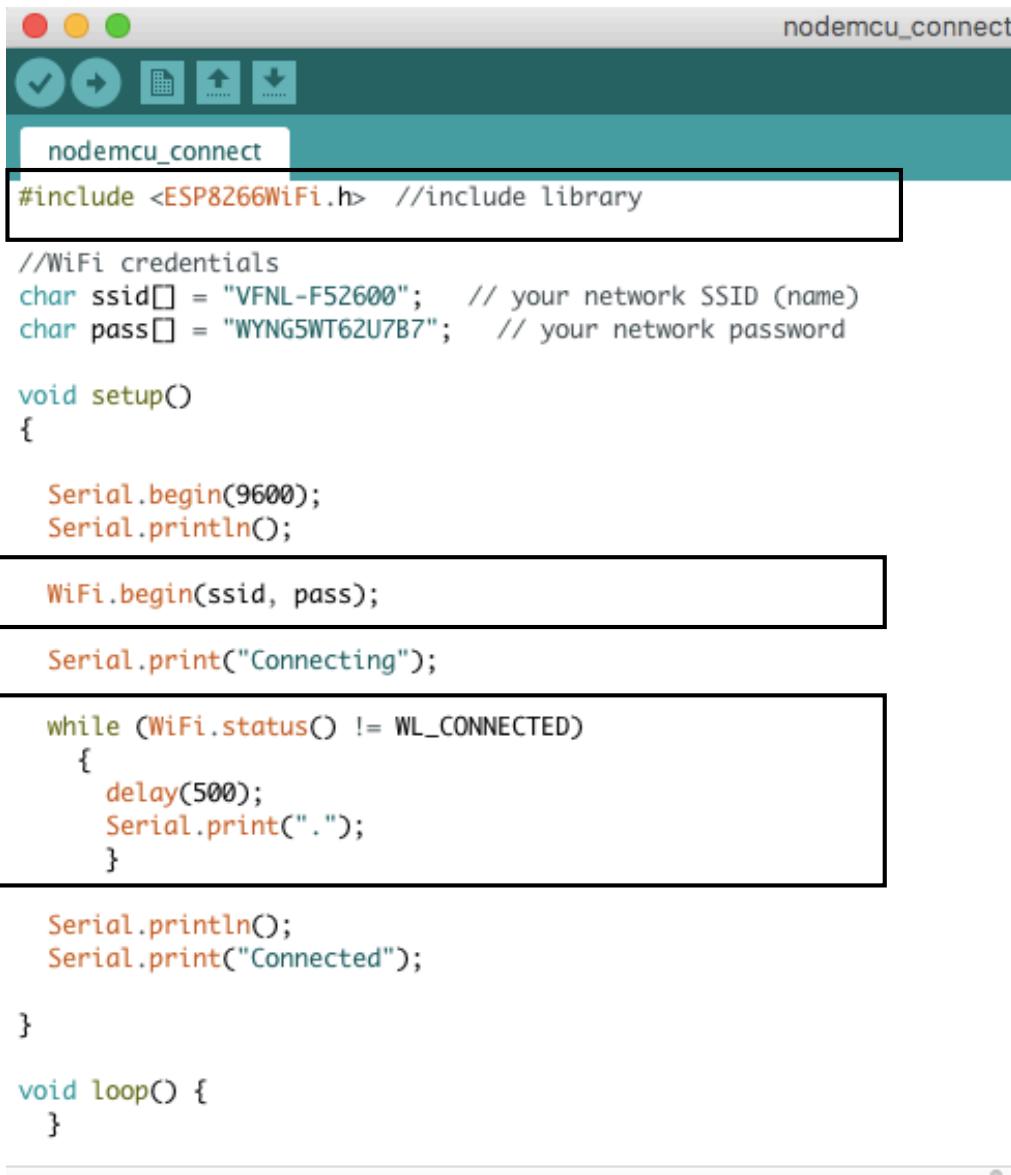
The dashboard features four main components:

- Field 1 Chart:** A line chart titled "Temperature (%)". The Y-axis ranges from 0 to 40. The X-axis shows dates from 14:25 to 14:40. The chart displays a constant value of approximately 21.40.
- Field 2 Chart:** A line chart titled "Humidity (%)". The Y-axis ranges from 0 to 100. The X-axis shows dates from 14:25 to 14:40. The chart displays a constant value of approximately 33.
- Temp (C):** A large digital display showing "21.40" with a timestamp "a month ago" below it.
- Hum (%):** A large digital display showing "33" with a timestamp "a month ago" below it.

Nodemcu v2

- CONNECT TO WIFI
- CONFIGURE THINGSPEAK CHANNEL
- SEND FIX DATA TO THINGSPEAK CHANNEL

NODEMCU v2 - CONNECT TO WIFI



The screenshot shows the NodeMCU Connect IDE interface. The title bar says "nodemcu_connect". The code editor window has a tab labeled "nodemcu_connect". The code itself is as follows:

```
#include <ESP8266WiFi.h> //include library

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

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.print("Connected");
}

void loop() { }
```

Code: nodemcu_distance.ino

```
#INCLUDE <ESP8266WiFi.h>

WIFI.BEGIN(NETWORK, PASSWORD)

WIFI.STATUS()
```

CONFIGURE THINGSPEAK CHANNEL

The screenshot shows a web browser window with the URL <https://thingspeak.com/channels>. The page title is "My Channels". The header includes a "New Channel" button, a search bar with a magnifying glass icon, and a table listing two channels: "Home Room 1" and "Hello World 1". Each channel row has buttons for "Private", "Public", "Settings", "Sharing", "API Keys", and "Data Import / Export".

Name	Created	Updated
🔒 Home Room 1	2019-03-10	2019-03-10 11:34
🔒 Hello World 1	2019-03-24	2019-03-24 15:10

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
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 measurement  
    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

LICENCE

EXCEPT WHERE OTHERWISE NOTED, THIS WORK IS LICENSED UNDER:

[HTTPS://CREATIVECOMMONS.ORG/LICENSES/BY/4.0/](https://creativecommons.org/licenses/by/4.0/)

