

Mesurer la température / l'humidité avec ESPHome

WORK IN PROGRESS/

Mettre en place une mesure de température en passant par un capteur type **DS18B20**

De quoi on parle ?



Le capteur de température DS18B20 est un capteur très utilisé dans les montages électronique. Il s'alimente de **3 à 5 Volts** et a une plage de mesure de **-55° à +125°**

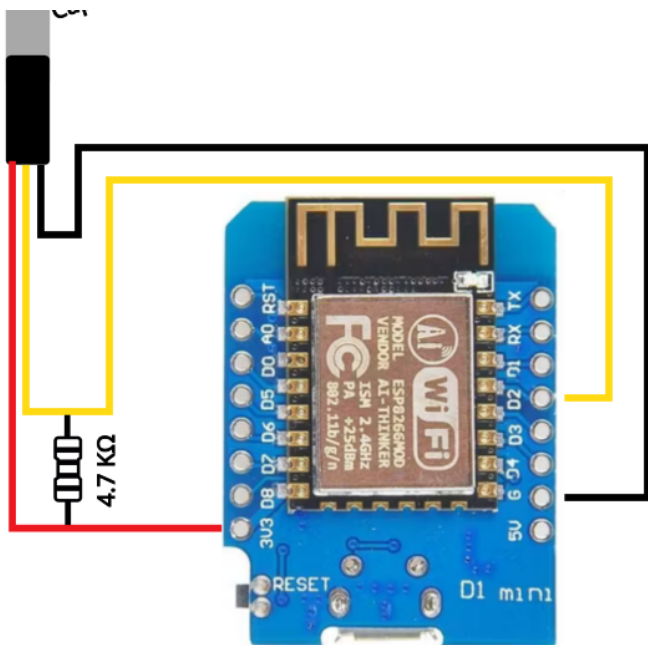
Même dans la serre, à plus de 50° : il résiste !

De plus, dans sa forme encapsulée (comme sur la photo), il est **étanche**.

Notre objectif sera donc de l'interfacer avec un micro contrôleur ESP (D1_mini dans notre exemple) pour remonter ses données vers notre serveur Home Assistant, en passant par ESP Home.

Voir [Page "Mon premier micro-contrôleur avec ESP Home"](#)

La soudure :



ESP8266 D1_Mini

Une fois équipé de son fer à souder préféré, on

soudera le fil rouge (la phase du capteur) sur la broche 3V3 de l'ESP, le fil de donnée (le jaune en général) sur, par exemple, la PIN D2. Le fil noir (la masse du capteur) sera quand à lui soudé sur une des Pins GND (Ground)

On soudera également, entre le fil de donnée et la phase, une résistance de 4.7Kohm (pour stabiliser la tension/le passage des données).

[plan_soudure_DS18B20.svg](#)

Programmer l'ESP et remonter les Données

Une fois les composants soudés, on va passer par l'interface d'ESPHome pour charger du code dans le micro-contrôleur.

Si notre micro-contrôleur est déjà reconnu et en capacité de communiquer en Wifi (voir module précédent) , on aura qu'à rajouter un bloc de code dans le fichier YAML pour communiquer avec le capteur de température.

```
one_wire:                #précise qu'on utilise le protocole OneWire
  - platform: gpio
    pin: GPIO4            #La pin D2 du micro-contrôleur, correspondant à GPIO04 : voir plan du
                          premier module

sensor:
  - platform: dallas_temp          # On précise utiliser un capteur type
    dallas_temp
      name: "Temperature DS18B20"  # On lui choisi un nom
      update_interval: 10s        # On lui demande de remonter la temperature vers le
                                  serveur toutes les 10 secondes
```

Plusieurs capteurs de température sur une carte?

```
[15:56:30.581][C][logger:219]: Initial Level: DEBUG
[15:56:30.582][C][logger:226]: Log Baud Rate: 115200
[15:56:30.582][C][logger:226]: Hardware UART: UART0
[15:56:30.582][C][gpio.one_wire:021]: GPIO 1-wire bus:
[15:56:30.583][C][gpio.one_wire:152]: Pin: GPIO4
[15:56:30.583][C][gpio.one_wire:085]: Found devices:
[15:56:30.583][C][gpio.one_wire:088]: 0x6b03176085c6ff28 (DS18B20)
[15:56:30.584][C][dallas.temp.sensor:029]: Dallas Temperature Sensor:
[15:56:30.584][C][dallas.temp.sensor:034]: Address: 0x6b03176085c6ff28 (DS18B20)
[15:56:30.584][C][dallas.temp.sensor:035]: Resolution: 12 bits
[15:56:30.584][C][dallas.temp.sensor:451]: Update Interval: 10.0s
[15:56:30.616][C][captive_portal:134]: Captive Portal:
[15:56:30.617][C][wifi:1505]: WiFi:
[15:56:30.617][C][wifi:1505]: Local MAC: F8:B3:B7:8C:2F:E0
[15:56:30.617][C][wifi:1505]: Connected: YES
[15:56:30.617][C][wifi:1216]: IP Address: 192.168.1.28
[15:56:30.617][C][wifi:1227]: SSID: 'fablab-lannion'
[15:56:30.617][C][wifi:1227]: BSSID: AC:15:A2:84:5A:76
[15:56:30.617][C][wifi:1227]: Hostname: 'capteur-temp-serre'
[15:56:30.617][C][wifi:1227]: Signal strength: -60 dB
[15:56:30.617][C][wifi:1227]: Channel: 11
[15:56:30.617][C][wifi:1227]: Subnet: 255.255.255.0
[15:56:30.617][C][wifi:1227]: Gateway: 192.168.1.1
```

Le protocole 1-Wire (<https://fr.wikipedia.org/wiki/1-Wire>) permettant de connecter plusieurs capteurs en même temps, il conviendra de récupérer l'adresse de chaque capteur connecté (dans les Logs) pour ne pas les mélanger.

Il n'y aura plus qu'à ajouter un bloc de code par capteur dans le YAML correspondant

```
sensor:
```

```
- platform: dallas_temp
```

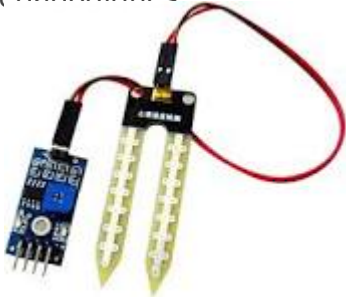
```
  address: 0x123456789ABCDEF #adresse numérique que l'on a récupéré dans les Logs
```

```
  name: "Temperature Salon"
```

Utiliser un capteur d'humidité type HW-080

De quoi on parle ?

Le capteur HW-080 est un grand classique dans la famille des capteurs utilisés dans les kits pédagogiques



Il s'agit d'une résistance qui varie en fonction de la conductivité du sol.

Le HW-080 possède une sortie analogique qui va lire une tension entre 0V et 3.3V et la convertir en valeur numérique.

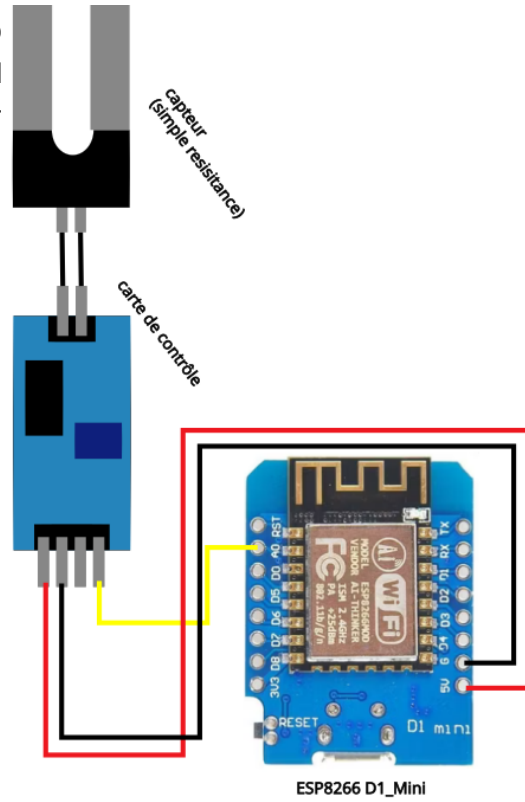
- **Plus la terre est sèche**, plus la résistance est forte, plus la tension de sortie (**AO**) est **proche de VCC** (tension max).
- **Plus la terre est mouillée**, plus l'eau conduit l'électricité, plus la résistance baisse, et plus la tension **descend vers 0V**.

La soudure

La partie soudure est assez classique : un +, un - et fil de données !

- => Le fil rouge (couleur par conv)
- => Le fil noir raccordera lui les G
- => Le fil jaune quand a lui, servir

our aller au VCC



une entrée analogique de l'ESP,

_A0 dans notre cas.

Ne pas hésiter à se référer au Pin-out de l'ESP !

[plan soudure HW 080.svg](#)

Le code YAML

L'étape suivante va donc être de rajouter du code à un ESP communiquant déjà en wifi avec notre interface ESPHome.

```
sensor:
  - platform: adc
    pin: A0
    name: "Humidité du sol - Valeur brute"
    id: humidite_sol_brute
    unit_of_measurement: "" # On enlève le "V" qui induit en erreur
    accuracy_decimals: 0
    update_interval: 5s      # 5 secondes pour que ce soit réactif en atelier

  filters:
    # Étape indispensable sur ESP8266 pour récupérer la valeur brute de 0 à 1023
    - multiply: 1023.0

    # Calibration "Tout ou rien" à ajuster avec vos tests :
    - calibrate_linear:
        # Format : valeur_brute -> pourcentage_voulu
        - 850 -> 0.0      # À l'air libre (Sec)
        - 250 -> 100.0   # Dans un verre d'eau (Humide)

  - platform: template
    name: "Humidité du sol"
    unit_of_measurement: "%"
    icon: "mdi:water-percent"
    accuracy_decimals: 1
    lambda: |-
      return id(humidite_sol_brute).state;

sensor:
  - platform: adc
    pin: A0
    name: "Humidité du sol - Valeur brute"
    id: humidite_sol_brute
```

```
unit_of_measurement: "" # On enlève le "V" qui induit en erreur
accuracy_decimals: 0
update_interval: 5s      # 5 secondes pour que ce soit réactif en atelier

filters:
  # Étape indispensable sur ESP8266 pour récupérer la valeur brute de 0 à 1023
  - multiply: 1023.0

  # Calibration "Tout ou rien" à ajuster avec vos tests :
  - calibrate_linear:
      # Format : valeur_brute -> pourcentage_voulu
      - 850 -> 0.0    # À l'air libre (Sec)
      - 250 -> 100.0 # Dans un verre d'eau (Humide)

- platform: template
  name: "Humidité du sol"
  unit_of_measurement: "%"
  icon: "mdi:water-percent"
  accuracy_decimals: 1
  lambda: |-
    return id(humidite_sol_brute).state;
```

Révision #9

Créé 2026-05-20 13:57:40 UTC par Florian

Mis à jour 2026-05-30 07:24:38 UTC par Florian