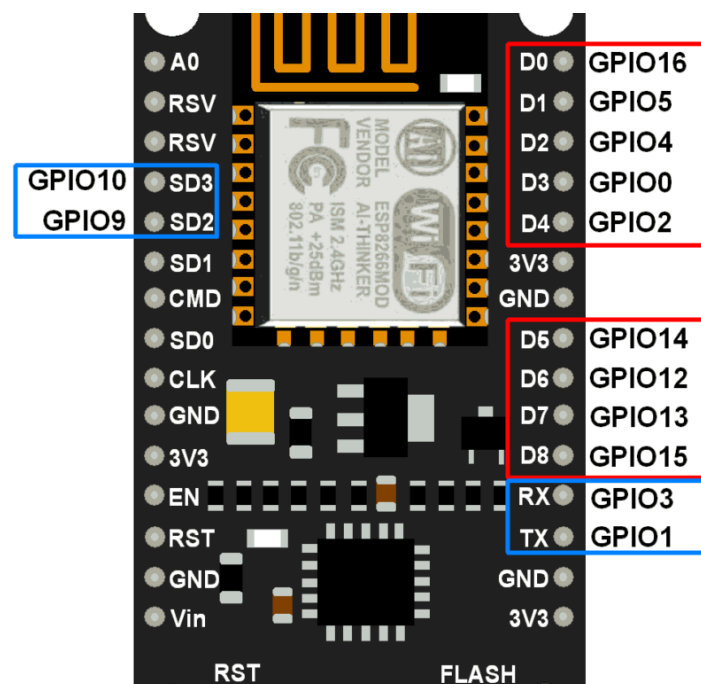


Rapport de la séance 2

Capteur Température et humidité

- Cette séance était dédiée au capteur Temp&Humi.
- Tout d'abord nous avons fait le point sur les capteurs à utiliser.
- Puis j'ai voulu commencer à programmer.
- J'ai rencontré alors plusieurs problèmes:
 - En programmant sur l'ESP8266 je me suis rendue compte qu'il n'y avait la possibilité de traiter les informations d'un seul capteur
 - Je me suis donc dirigé vers un ESP8266 NodeMcu ayant plusieurs entrées « data » (D0, D1...)
 - Pour définir le port lié au capteur j'ai fait face à un autre problème: le nom en effet le D4 est appelé par DHTPIN 2

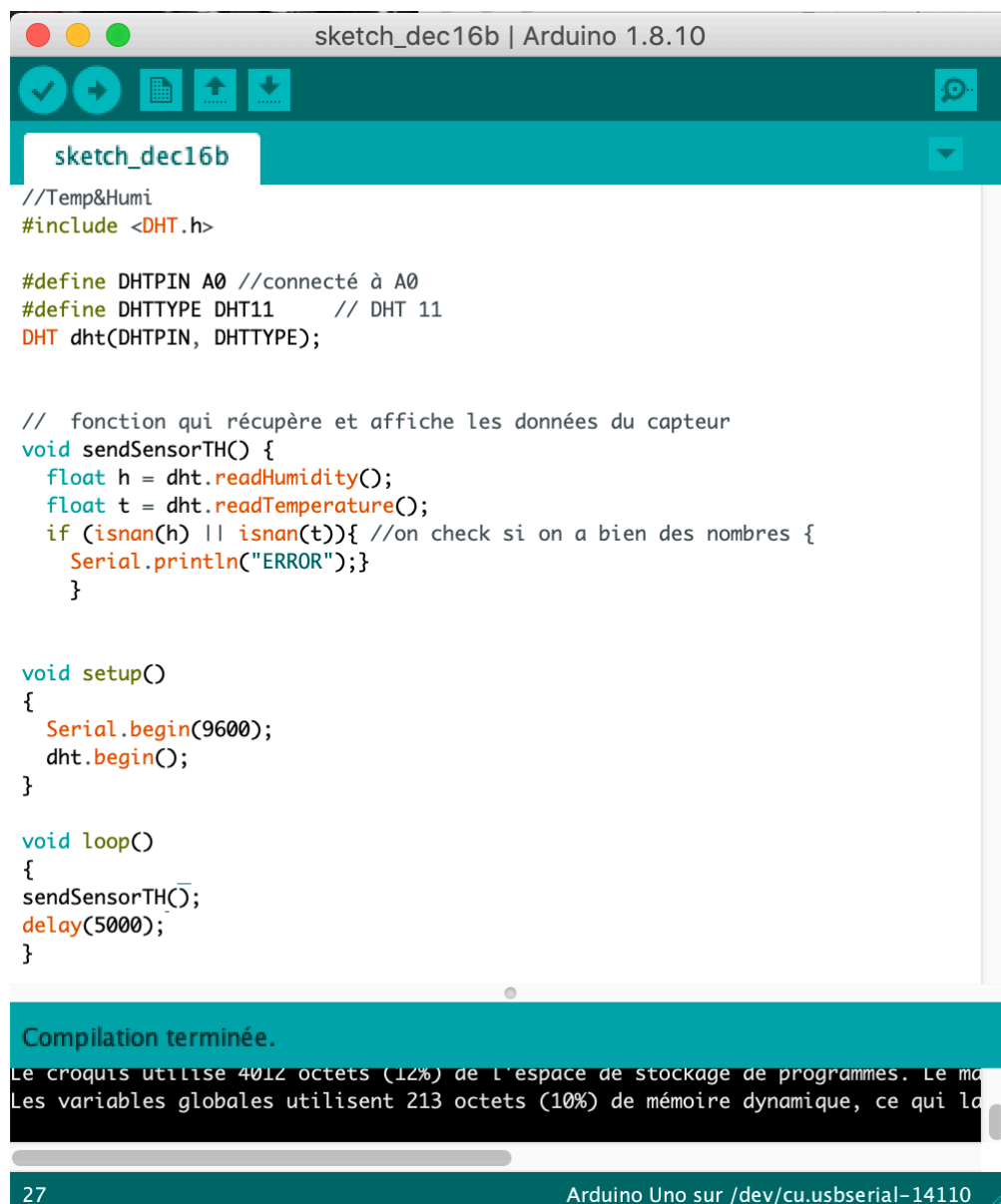


- Puis le problème de librairies sur lequel je passe actuellement ma soirée; les documents qui ne sont pas trouvés, de nouveaux à télécharger qui requiert le téléchargement de nouvelles libraires...

Je n'ai donc pas réussi encore à faire fonctionner mon programme mais nous avons avancé et cela ira plus rapidement pour les prochains.

EDIT: Finalement j'ai revu ma chaîne de traitement de l'information afin de n'avoir plus qu'une seule interaction (série) du module wifi avec la carte Arduino (et non plus avec les capteurs comme j'ai voulu essayer). Cela me permet de n'avoir plus besoin des bibliothèques complexes.

ARDUINO <==> DHT11



```
sketch_dec16b | Arduino 1.8.10

//Temp&Humi
#include <DHT.h>

#define DHTPIN A0 //connecté à A0
#define DHTTYPE DHT11 // DHT 11
DHT dht(DHTPIN, DHTTYPE);

// fonction qui récupère et affiche les données du capteur
void sendSensorTHC() {
  float h = dht.readHumidity();
  float t = dht.readTemperature();
  if (isnan(h) || isnan(t)){ //on check si on a bien des nombres {
    Serial.println("ERROR");}
}

void setup()
{
  Serial.begin(9600);
  dht.begin();
}

void loop()
{
  sendSensorTHC();
  delay(5000);
}
```

Compilation terminée.

Le croquis utilise 4012 octets (12%) de l'espace de stockage de programmes. Le mo
Les variables globales utilisent 213 octets (10%) de mémoire dynamique, ce qui la

27 Arduino Uno sur /dev/cu.usbserial-14110

La fonction sera enrichie : ce ne sera que lorsqu'ArduinoUno recevra un message du module wifi que la fonction tournera. Celui-ci recevant l'action de l'application Blynk. Les mesures ne sont pas affichées sur le moniteur série car celles-ci seront

envoyées sur l'application. Cependant afin de s'assurer du bon fonctionnement nous rajoutons

```
Serial.print("La température est: ");
```

```
Serial.print(t);
```

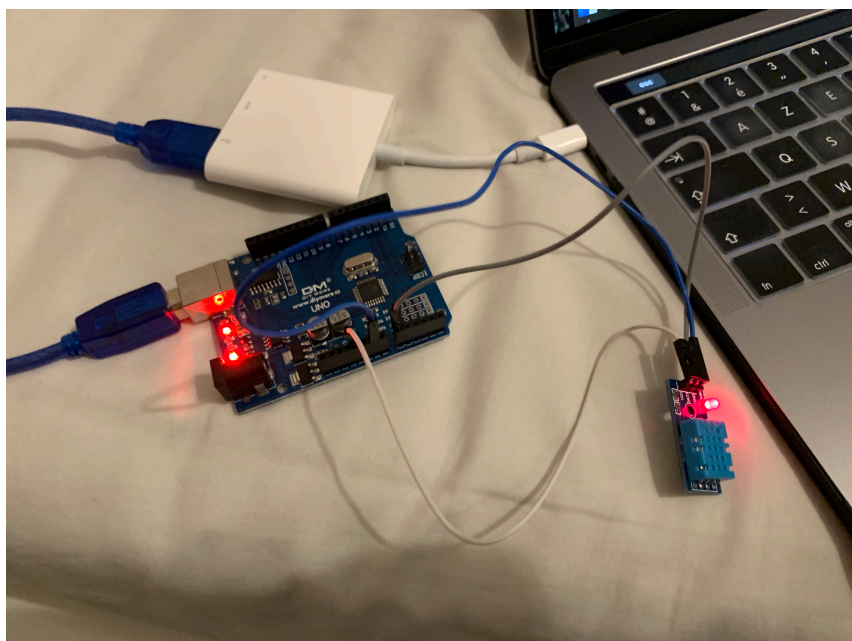
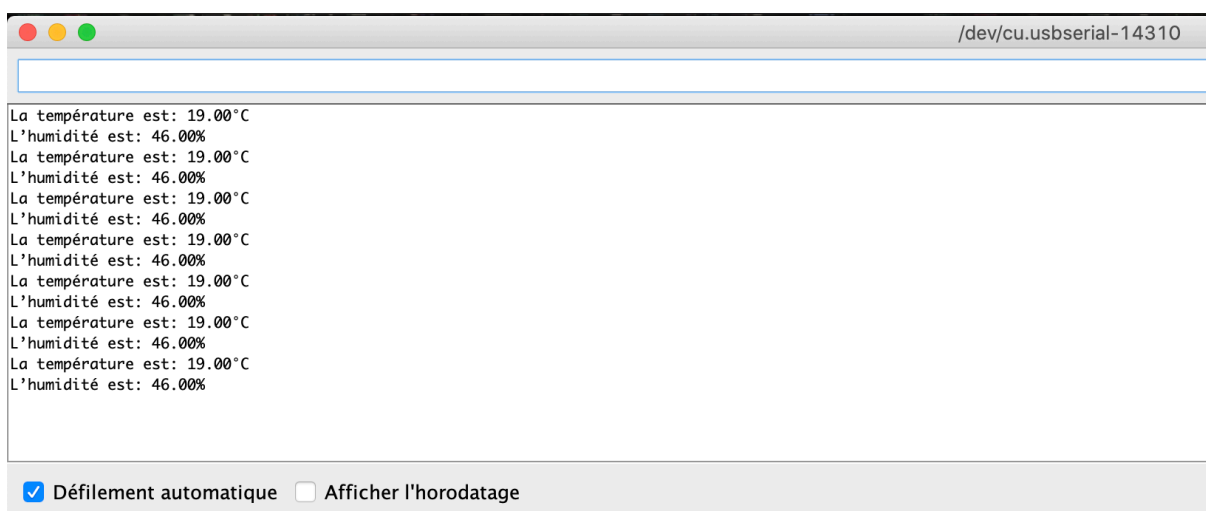
```
Serial.println("°C");
```

```
Serial.print("L'humidité est: ");
```

```
Serial.print(h);
```

```
Serial.println("%");
```

Nous obtenons alors sur le moniteur série :



lundi 16 décembre 2019