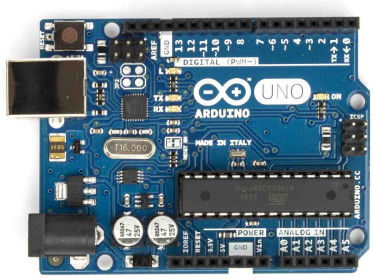
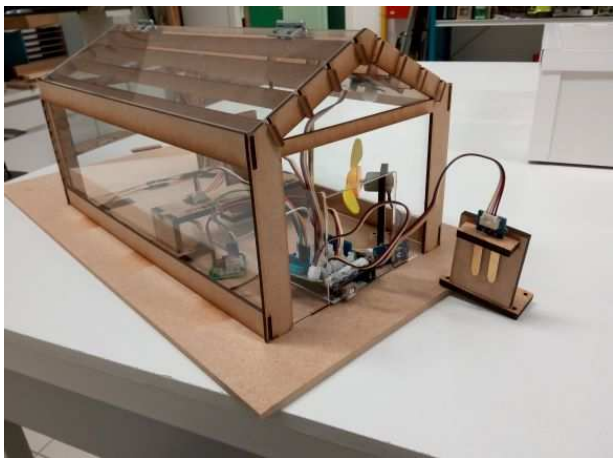
Thème : numérique

Faire de la physique avec ARDUINO – 6

Laurent FAURÉ, Nicolas HERVE, Gilles ESPINASSE (ENSFEA)



Ce sixième article va nous permettre de faire connaissance avec la technologie GROVE, bien pratique quand on veut développer des projets plus complexes avec Arduino.
Pour cela, on va prendre l'exemple d'un projet de serre automatisée.



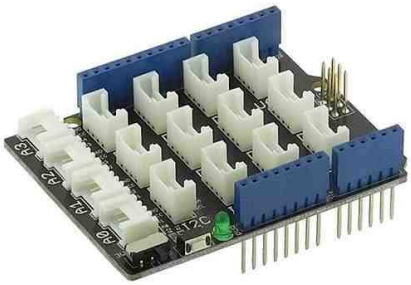
Le contexte de la serre automatisée

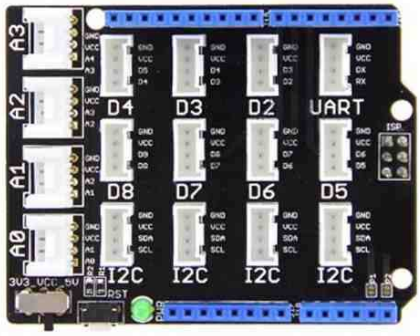
Ce projet a été initié par Cyril Dagorne (EPLEFPA de la Germinière au Mans) lors d'un stage de formation continue à l'ENSFEA, encadré par Laurent Fauré, sur la mise en place d'un fablab dans un établissement de l'enseignement agricole. Nous l'avons ensuite finalisé.
Il peut être amélioré bien sûr autant du point de vue de la programmation que du matériel utilisé.
La serre peut en effet être agrandie afin d'y déposer réellement des plantes.

Nous avons simulé l'éclairage, le chauffage, l'irrigation par des diodes électroluminescentes. On pourrait donc mettre des relais à la place des DEL pour commander « pour de vrai » tous les besoins des plantes.
Nous avons utilisé seulement le port série ici pour récupérer les mesures des capteurs, on aurait donc pu ajouter un écran LCD pour afficher directement les valeurs des différents capteurs.
Ce que nous présentons ici est donc largement perfectible, mais constitue une base solide pour aller plus loin.

Le système GROVE

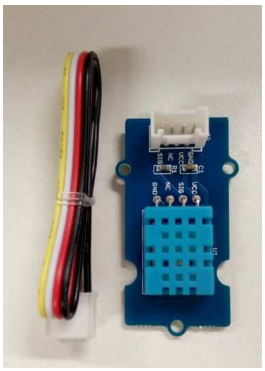
Le principal élément du système GROVE est le ***Shield***, qui est une carte d'interface, se clipsant sur une carte arduino d'un côté et raccordant facilement, rapidement et sans soudure des capteurs et des actionneurs.





Le gros avantage du système GROVE est la fin de la « choucroute » de fils (petits, nombreux et fragiles) : chaque capteur GROVE se branche avec un câble unique.

Exemple sur ce capteur de température :



Une fois le capteur connecté au shield grove, le fil noir est relié à la masse (ground), le rouge à l'alimentation positive (Vcc), le jaune à l'entrée ou à la sortie du capteur et le blanc n'est pas toujours connecté.
Le prix de ces modules est plus cher que les modules standards arduino, mais ils sont plus stables et plus costauds, notamment parce qu'il y a souvent un peu d'électronique permettant une plus grande stabilité du signal électrique.
Cette robustesse a cependant un « prix » en programmation : l'ajout d'électronique fait qu'un module GROVE est plus complexe et qu'une bibliothèque est souvent à appeler.

Les différentes fonctions de notre serre automatisée

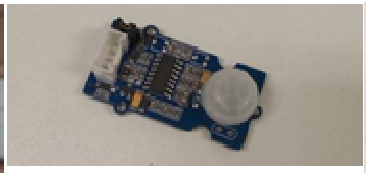
Nous souhaitons donc pour notre serre :

- Commander l'éclairage automatique de la serre si la présence d'une personne est détectée ;
- Contrôler l'humidité de la terre des plantes ;
- Commander le moteur de la pompe d'irrigation ;
- Commander la mise en route d'une ventilation selon la valeur de l'hygrométrie ;
- Commander l'allumage du chauffage selon la valeur de la température dans la serre ;
- Commander l'ouverture d'un volet ;

Montage et programme associés à chaque fonction

Commande de l'éclairage automatique :

Fixé au plafond de la serre il y a un détecteur de présence PIR et une DEL qui simule l'éclairage lors de la présence d'une personne.

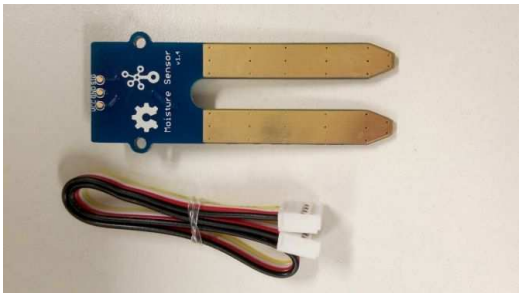



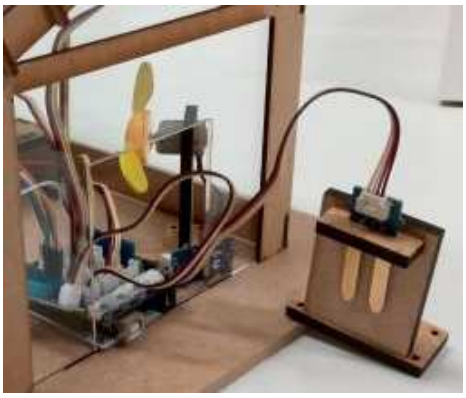

Programme :

```
int lumiere = 7;   //la LED d'éclairage (ou relais de l'éclairage) branchée sur la pin 7
int dectecteurPIR = 3;   //le détecteur de présence (PIR) est branché sur la pin 3

void setup() {
   pinMode(lumiere, OUTPUT);   //la pin de la lumière est une sortie
   pinMode(dectecteurPIR, INPUT);  //la pin du détecteur de présence est une entrée
}

void loop() {
  if (digitalRead(dectecteurPIR)) // si le détecteur de présence est activé
      {digitalWrite(lumiere, HIGH);}// allumer la lumière
  else
      {digitalWrite(lumiere, LOW);}//sinon éteindre la lumière
}
```

Contrôle de l'humidité de la terre et commande de l'irrigation :

On utilise le capteur d'humidité du sol de marque SEEED de référence 101020008 pour contrôler l'humidité de la terre (il est ici utilisé à vide, nous le mettons en contact avec un chiffon mouillé pour simuler que l'arrosage a été effectué).
Nous utilisons une DEL qui simule l'état de marche de la pompe d'irrigation.



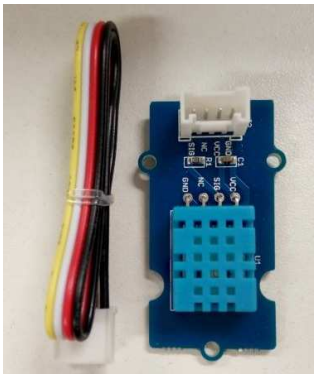


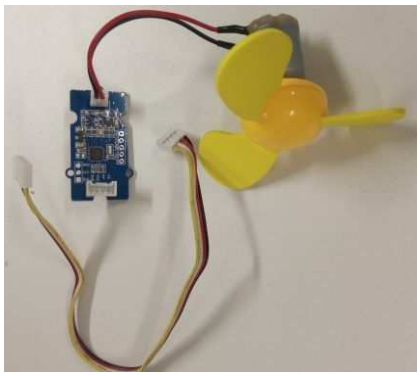
Programme :

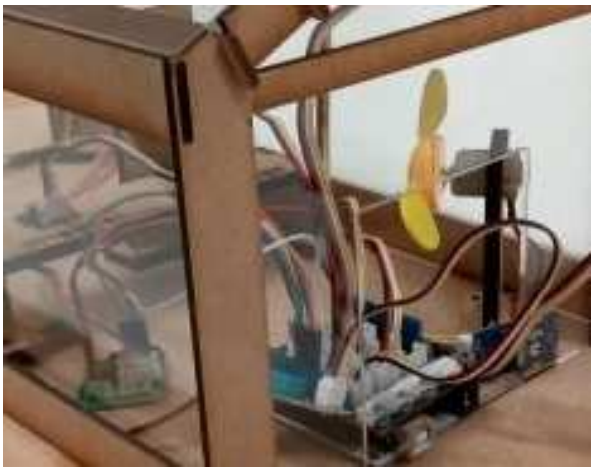
```
int irrigation = 2;  //la LED irrigation (ou relais de l'électrovanne) branchée sur la pin 2
int Capteur_Terre = A0;  //le capteur humidité de la terre est branché sur la pin A0
int Valeur_Terre;  // variable de stockage état du capteur terre
int PourcentTerre;   //variable de stockage état du capteur terre en pourcentage

void setup() {
   pinMode(irrigation, OUTPUT);  //la pin irrigation est une sortie
   pinMode(Capteur_Terre, INPUT);  //la pin du capteur d'humidité de la terre est une entrée
   Serial.begin(9600);   //initialisation du port série
}

void loop() {
  Valeur_Terre = analogRead(Capteur_Terre);//on met la valeur du capteur d'humidité de la terre (comprise entre 0 et 1023) dans la variable "Valeur_Terre"
  PourcentTerre = map(Valeur_Terre, 0, 1023, 0, 100);//on change d'échelle pour mettre de 0 à 100 dans la variable "PourcentTerre"
  Serial.print("Humidite Terre: "); //on écrit sur le port série
  Serial.print(PourcentTerre); //puis la valeur de la variable s'affiche sur le port série
  Serial.print(" %         ");
  if (PourcentTerre<50) // si la valeur de "PourcentTerre" est inférieur à 50
          {digitalWrite(irrigation, HIGH);}//on met l'irrigation au niveau haut
  else
           {digitalWrite(irrigation, LOW);}//sinon au niveau bas
  delay(1000);//delai de 1 seconde entre les mesures car la lecture prend 250ms
```

Commande de la mise en route de la ventilation selon l'hygrométrie :

Nous utilisons le capteur de température et d'humidité de l'air, le DHT11 (nous le mettons dans nos mains pour simuler l'augmentation de température et l'hygrométrie). Ce capteur fonctionne à l'aide d'une librairie (ici DHT.h). Selon les librairies que vous installez, il se peut que l'écriture change légèrement pour faire appel à l'humidité ou la température.
Le ventilateur (avec son driver) est utilisé pour abaisser l'hygrométrie.





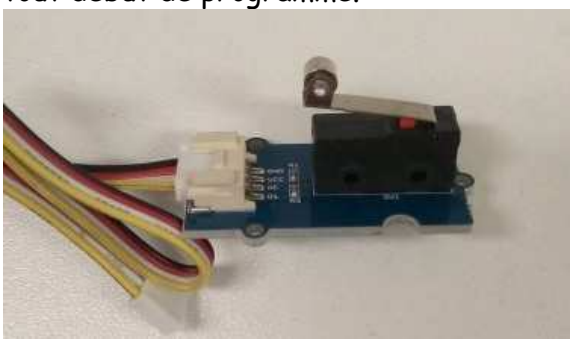


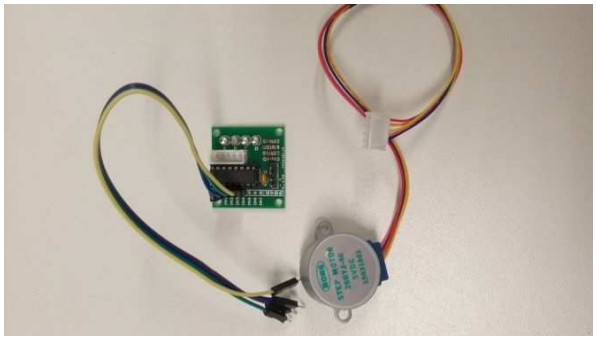
Programme :

```
#include <DHT.h>   // on charge la librairie des capteurs DHT température
#define DHTPIN 4  //on définit la pin sur laquelle est branchée la sortie du DHT11
int ventilo = 8;  //le ventilateur est branché sur la pin 8

//Décommenter la ligne qui correspond à votre capteur de température
#define DHTTYPE    DHT11     // DHT 11
//#define DHTTYPE    DHT22     // si vous utilisez le capteur DHT 22 (AM2302)
//#define DHTTYPE    DHT21     // si vous utilisez le capteur DHT 21 (AM2301)

DHT dht(DHTPIN, DHTTYPE);   //on crée l'objet du capteur de température

void setup() {
    pinMode(ventilo, OUTPUT); // la pin ventilateur est une sortie
    Serial.begin(9600);   //initialisation du port série
    dht.begin(); // démarrage du capteur de température et humidité de l'air
}

void loop() {
  float h = dht.readHumidity();   // Lecture du taux d'humidité mis dans la variable "h"
  //Stoppe le programme et renvoie un message d'erreur si le capteur ne renvoie aucune mesure
  if (isnan(h) || isnan(t))
            {Serial.println("le capteur de température ne fonctionne pas");
            return;}
  Serial.print("Humidite: "); //on écrit sur le port série
  Serial.print(h);
  Serial.print(" %         ");
  if (h>70) //si l'humidité est supérieure à 70%
     {digitalWrite(ventilo, HIGH);}// on met le ventilo à l'état haut
  else digitalWrite(ventilo, LOW); // sinon à l'état bas
}
```

Commande d'initialisation du volet :

Un capteur de fin de course permet de détecter que le volet de la serre est fermé. Il sert à initialiser le moteur pas à pas (avec son driver), qui commande ce volet à l'aide d'un bras, en tout début de programme.





Programme :

```
#include <Stepper.h>  // charge la librairie qui pilote le moteur pas à pas

int CapteurFinCourse = 5;  //le capteur de fin de course du toit est branché sur la pin 5
int FinCourse;  //variable de l'état du capteur de fin de course

//cablage moteur pas à pas 28BYJ-48
//Pin9 au IN1 du driver ULN2003
//Pin10 au IN2 du driver ULN2003
//Pin11 au IN3 du driver ULN2003
//Pin12 au IN4 du driver ULN2003
const int Pas_par_tours = 32*64;   //nombre de pas du moteur (on peut mettre 64*64 pour plus de précision)
Stepper monmoteur = Stepper (Pas_par_tours, 10, 12, 11, 9);      //création de l'objet 'monmoteur' (nombre de pas,1N2,1N4,1N3,1N1)

void setup() {
   pinMode(CapteurFinCourse, INPUT); //la pin du capteur de fin de course est une entrée

   FinCourse=digitalRead(CapteurFinCourse);// lecture de l'état du capteur de fin de course
   while (FinCourse == HIGH)  //tant que le bouton de fin de course n'est pas appuyé
       {
        monmoteur.setSpeed(10); //vitesse du moteur
        monmoteur.step(-1);  //on fait tourner le moteur pour fermer
        FinCourse= digitalRead(CapteurFinCourse);//on relit état du capteur fin de course pour sortir de la boucle
       }
}
```

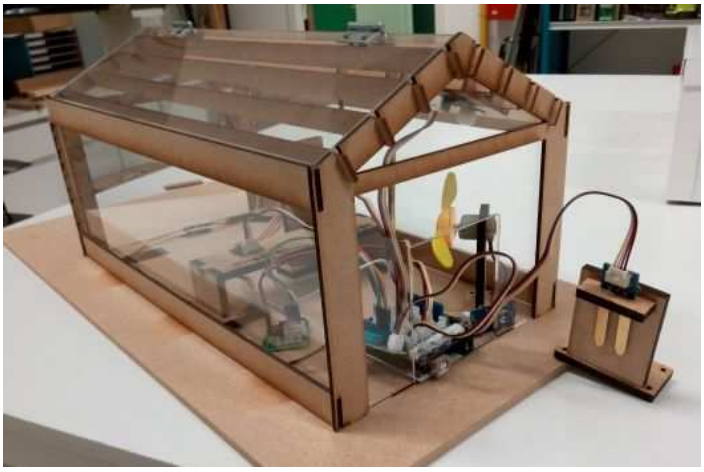
Contrôle de la température et commande du chauffage et du volet:

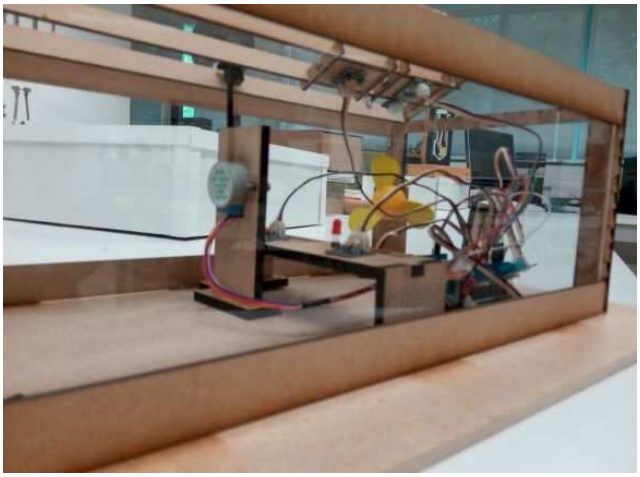
Nous utilisons le DHT11 ainsi qu'une DEL qui simule l'état de marche du chauffage, mais aussi le moteur pas à pas (avec son driver) vu précédemment.

Programme :

```
#include <DHT.h>  // on charge la librairie des capteurs DHT température
#include <Stepper.h>  // charge la librairie qui pilote le moteur pas à pas

#define DHTPIN 4  //on définit la pin sur laquelle est branchée la sortie du DHT11
int chauffage = 6;  //la LED chauffage (ou relais) branchée sur la pin 6

//Décommenter la ligne qui correspond à votre capteur de température
#define DHTTYPE    DHT11     // DHT 11
//#define DHTTYPE    DHT22     // si vous utilisez le capteur DHT 22 (AM2302)
//#define DHTTYPE    DHT21     // si vous utilisez le capteur DHT 21 (AM2301)

DHT dht(DHTPIN, DHTTYPE);   //on crée l'objet du capteur de température

//cablage moteur pas à pas 28BYJ-48
//Pin9 au IN1 du driver ULN2003
//Pin10 au IN2 du driver ULN2003
//Pin11 au IN3 du driver ULN2003
//Pin12 au IN4 du driver ULN2003
const int Pas_par_tours = 32*64;   //nombre de pas du moteur(on peut mettre 64*64 pour plus de précision)
Stepper monmoteur = Stepper(Pas_par_tours, 10, 12, 11, 9);      //création de l'objet 'monmoteur' (nombre de pas,1N2,1N4,1N3,1N1)
boolean volet_ferme = true;   //on met l'état de la variable volet_ferme sur vrai

void setup() {
   pinMode(chauffage, OUTPUT);  //la pin du chauffage est une sortie

   Serial.begin(9600);   //initialisation du port série
   dht.begin(); // démarrage du capteur de température et humidité de l'air
}

void loop() {
  float t = dht.readTemperature();   // Lecture de la température en Celcius mis dans la variable "t"
  //Stoppe le programme et renvoie un message d'erreur si le capteur ne renvoie aucune mesure
  if (isnan(h) || isnan(t))
          {Serial.println("le capteur de température ne fonctionne pas");
          return;}
  Serial.print("Temperature: ");
  Serial.print(t);
  Serial.println(" *C ");

  if (t<23 && volet_ferme == true)//si la température est trop froide et que les volets sont fermés
       {digitalWrite(chauffage, HIGH);}//on met le chauffage à l'état haut
  else {digitalWrite(chauffage, LOW);}// sinon à l'état bas

  monmoteur.setSpeed(10); //vitesse du moteur pas à pas
  if (t<24 && volet_ferme == false)//si température fraiche et volet ouvert
       {monmoteur.step(-512); //le moteur se déplace 1/4 de tour pour fermer,ce réglage est en fonction de votre montage
       volet_ferme = !volet_ferme;}// on inverse l'état de la variable "volet_ferme"
  if (t<24 && volet_ferme == true || t>24 && volet_ferme == false)
       {monmoteur.step(0);} // on ne bouge pas le moteur
  if (t>24 && volet_ferme == true)//si température chaude et volet fermé
       {monmoteur.step(512); // le moteur se déplace 1/4 de tour pour ouvrir
       volet_ferme = !volet_ferme;}// on inverse l'état de la variable "volet_ferme"
   }
```

Assemblage de toutes les fonctions





Programme :

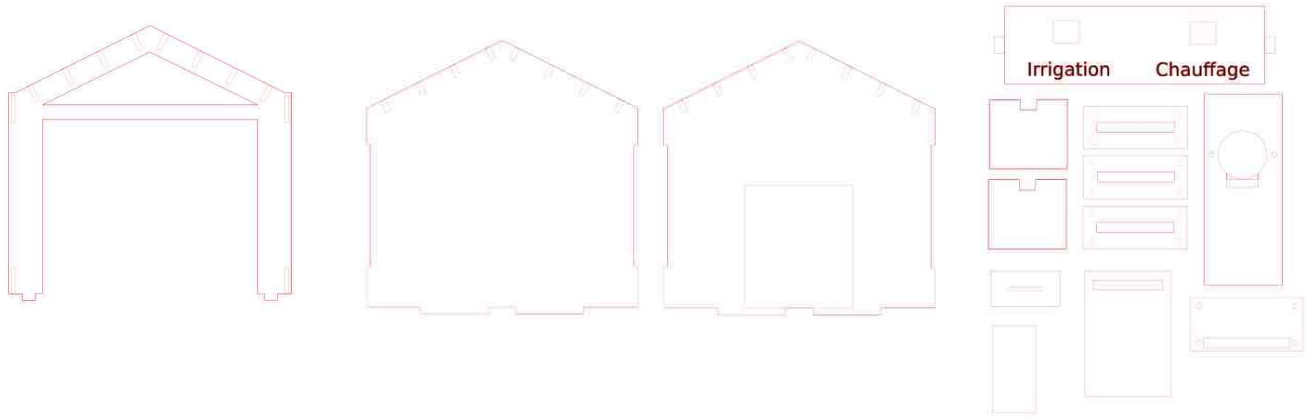
```
#include <DHT.h>  // on charge la librairie des capteurs DHT température
#include <Stepper.h>  // charge la librairie qui pilote le moteur pas à pas

#define DHTPIN 4  //on définit la pin sur laquelle est branchée la sortie du DHT11
int ventilo = 8;  //le ventilateur est branché sur la pin 8
int irrigation = 2;  //la LED irrigation (ou relais de l'électrovanne) branché sur la pin 2
int Capteur_Terre = A0;  //la capteur humidité de la terre est banché sur la pin A0
int Valeur_Terre;  // variable de stockage état du capteur terre
int PourcentTerre;  //variable de stockage état du capteur terre en pourcentage
int CapteurFinCourse = 5;  //le capteur de fin de course du toit est branché sur la pin 5
int FinCourse;  //variable de l'état du capteur de fin de course
int lumiere = 7;  //la LED d'éclairage (ou relais de l'éclairage) branché sur la pin 7
int dectecteurPIR = 3;  //le détecteur de présence (PIR) est branché sur la pin 3
int chauffage = 6;  //la LED chauffage (ou relais) branché sur la pin 6

//Décommenter la ligne qui correspond à votre capteur de température
#define DHTTYPE    DHT11    // DHT 11
//#define DHTTYPE    DHT22    // si vous utilisez le capteur DHT 22 (AM2302)
//#define DHTTYPE    DHT21    // si vous utilisez le capteur DHT 21 (AM2301)

DHT dht(DHTPIN, DHTTYPE);  //on crée l'objet du capteur de température

//cablage moteur pas à pas 28BYJ-48
//Pin9 au IN1 du driver ULN2003
//Pin10 au IN2 du driver ULN2003
//Pin11 au IN3 du driver ULN2003
//Pin12 au IN4 du driver ULN2003
const int Pas_par_tours = 32*64;  //nombre de pas du moteur (on peut mettre 64*64 pour plus de précision)
boolean volet_ferme = true;  //on met l'état de la variable volet_ferme sur vrai
Stepper monmoteur = Stepper (Pas_par_tours, 10, 12, 11, 9);  //création de l'objet 'monmoteur' (nombre de
pas,1N2,1N4,1N3,1N1)


void setup() {
   pinMode(ventilo, OUTPUT); // la pin ventilateur est une sortie
   pinMode(irrigation, OUTPUT);  //la pin irrigation est une sortie
   pinMode(Capteur_Terre, INPUT);  //la pin du capteur d'humidité de la terre est une entrée
   pinMode(CapteurFinCourse, INPUT); //la pin du capteur de fin de course est une entrée
   pinMode(lumiere, OUTPUT);  //la pin de la lumière est une sortie
   pinMode(dectecteurPIR, INPUT);  //la pin du détecteur de présence est une entrée
```

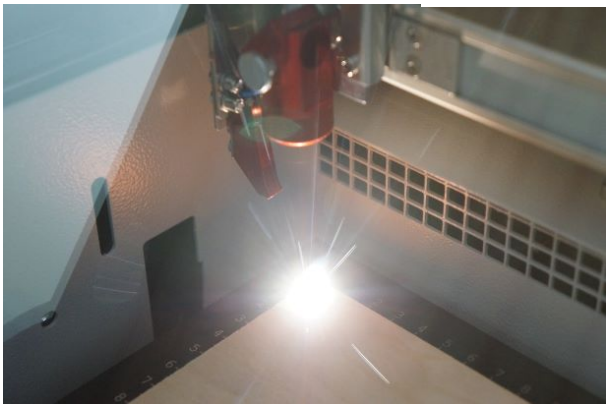
```
   pinMode(chauffage, OUTPUT);  //la pin du chauffage est une sortie

   Serial.begin(9600);   //initialisation du port série
   dht.begin(); // démarrage du capteur de température et humidité de l'air

   FinCourse=digitalRead(CapteurFinCourse);// lecture de l'état du capteur de fin de course
   while (FinCourse == HIGH)  //tant que le bouton de fin de course n'est pas appuyé
       {
        monmoteur.setSpeed(10); //vitesse du moteur
        monmoteur.step(-1);  //on fait tourner le moteur pour fermer
        FinCourse= digitalRead(CapteurFinCourse);//on relit l'état du capteur fin de course pour sortir de la boucle
       }
}

void loop() {

  if (digitalRead(dectecteurPIR)) // si le détecteur de présence est activé
      {digitalWrite(lumiere, HIGH);}// allumer la lumière
  else
      {digitalWrite(lumiere, LOW);}//sinon éteindre la lumière
      ////////////////////////////////////////////////////////////////////////
  Valeur_Terre = analogRead(Capteur_Terre);//on met la valeur du capteur d'humidité de la terre (comprise entre 0 et 1023)
dans la variable "Valeur_Terre"
  PourcentTerre = map(Valeur_Terre, 0, 1023, 0, 100);//on change d'échelle pour mettre de 0 à 100 dans la variable
"PourcentTerre"
  Serial.print("Humidite Terre: "); //on écrit sur le port série
  Serial.print(PourcentTerre); //puis la valeur de la variable s'affiche sur le port série
  Serial.print(" %          ");
  if (PourcentTerre<50) // si la valeur de "PourcentTerre" est inférieur à 50
          {digitalWrite(irrigation, HIGH);}//on met l'irrigation au niveau haut
  else
           {digitalWrite(irrigation, LOW);}//sinon au niveau bas
  delay(1000);//delais de 1 secondes entre les mesures car la lecture prend 250ms
  ////////////////////////////////////////////////////////////////////////////////
  float h = dht.readHumidity();   // Lecture du taux d'humidité mis dans la variable "h"
  float t = dht.readTemperature();   // Lecture de la température en Celcius mis dans la variable "t"
  //Stoppe le programme et renvoie un message d'erreur si le capteur ne renvoie aucune mesure
  if (isnan(h) || isnan(t))
          {Serial.println("le capteur de température ne fonctionne pas");
          return;}
  Serial.print("Humidite: "); //on écrit sur le port série
  Serial.print(h);
  Serial.print(" %         ");
  Serial.print("Temperature: ");
  Serial.print(t);
  Serial.println(" *C ");

  if (h>70) //si l'humidité est supérieure à 70%
     {digitalWrite(ventilo, HIGH);}// on met le ventilo à l'état haut
  else digitalWrite(ventilo, LOW); // sinon à l'état bas

  if (t<23 && volet_ferme == true)//si la température est trop froide et que les volets sont fermés
       {digitalWrite(chauffage, HIGH);}//on met le chauffage à l'état haut
  else {digitalWrite(chauffage, LOW);}// sinon à l'état bas

  monmoteur.setSpeed(10); //vitesse du moteur pas à pas
  if (t<24 && volet_ferme == false)//si température fraiche et volet ouvert
       {monmoteur.step(-512); //le moteur se déplace 1/4 de tour pour fermer, ce réglage est en fonction de votre montage
       volet_ferme = !volet_ferme;}// on inverse l'état de la variable "volet_ferme"

  if (t<24 && volet_ferme == true || t>24 && volet_ferme == false)
       {monmoteur.step(0);} // on ne bouge pas le moteur
```

```
  if (t>24 && volet_ferme == true)//si température chaude et volet fermé
        {monmoteur.step(512); // le moteur se déplace 1/4 de tour pour ouvrir
        volet_ferme = !volet_ferme;}// on inverse l'état de la variable "volet_ferme"

}
```

Construction de la serre et finalisation (provisoire) du projet

Lors de son stage Cyril a beaucoup utilisé notre découpe laser pour construire l'armature de la serre en bois et plexiglass. C'est un appareil couteux, mais que vous pouvez trouver dans un fablab d'une grande ville proche de chez vous. Il a tout d'abord créé sur le logiciel d'image vectorielle **Inkscape** les dessins, qu'il a ensuite découpé avec la découpe laser.





Vous avez aussi la possibilité d'acheter une « serre » déjà construite comme celle d'IKEA par exemple :

Il a aussi utilisé une imprimante 3D pour imprimer les petites biellettes qui relient le moteur pas à pas au volet pour soulever celui-ci.

Prix du projet

Liste des composants électroniques et tarifs TTC (acheté chez Gotronic) :
Arduino Uno 3 (19,50€)
Shield grove (4,50€)
3 LED grove (3 x 2,30€ = 6,90€)
Détecteur PIR grove (5,50€)
Ventilateur grove (10,60€)
Capteur de température et humidité DHT11 grove (5,70€)
Capteur d'humidité du sol grove (3,35€)
Capteur fin de course grove (2,20€)
Moteur pas à pas 28BYJ-48 et son driver basé sur un ULN2003A (7,90€)
Total : 66,15€ TTC (cela peut être beaucoup moins cher si vous achetez une carte Uno compatible, et les capteurs actionneurs non modèles « grove »).

Bibliographie

DUMONT, A. & YERNAUX, B. (2017). *50 montages pédagogiques avec Arduino*. Dijon : Educagri édition.
VAN DREUMEL, W. (2016). *36 expériences de physique avec Arduino pour la maison et l'école: Newton a rendez-vous avec l'électronique*. Publitronic – Elektor.