# Club Robotique collège Cycle 2 - Station de mesure (T°, H)

# Phase 1 Conception (2 séances)

**Phase 1 - Conception**

**Objectifs**

**Compétences techniques:**

- Introduction générale à la modélisation 3D
- Prendre en main le logiciel TinkerCad
- Modéliser un boîtier pour le capteur

**Savoir-être, compétences transversales:**

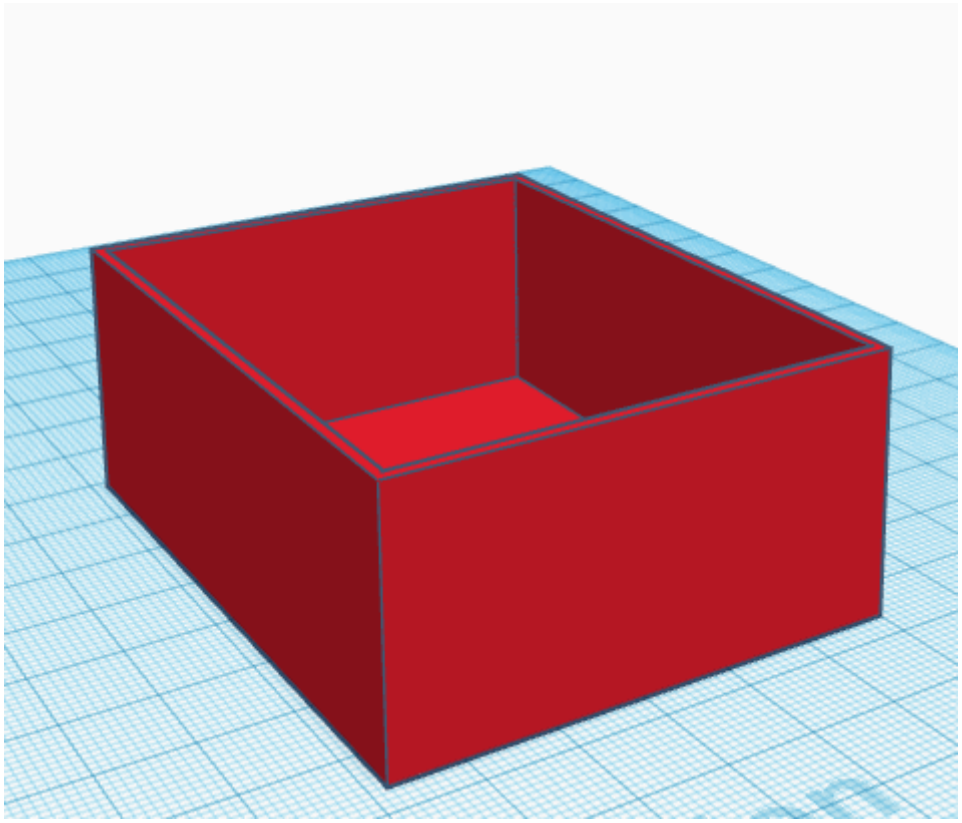
- Travail individuel
- Lire un tutoriel détaillé

**Déroulement de la séance:**

1. **Consignes: Sécurité, précautions matériel**
   - Travail individuel.
   - Rappel: Tout est fragile.
2. **Phases et méthodes d’animation** a. **Présentation de l'activité du jour**
   - Objectifs rappelés.
   - Importance de la modélisation 3D dans le projet.
3. **Prendre en main le logiciel TinkerCad**
   - Démonstration interactive.
   - Manipulation guidée des outils de base.
4. **La conception du fichier :**

**Création du Boîtier Principal:**

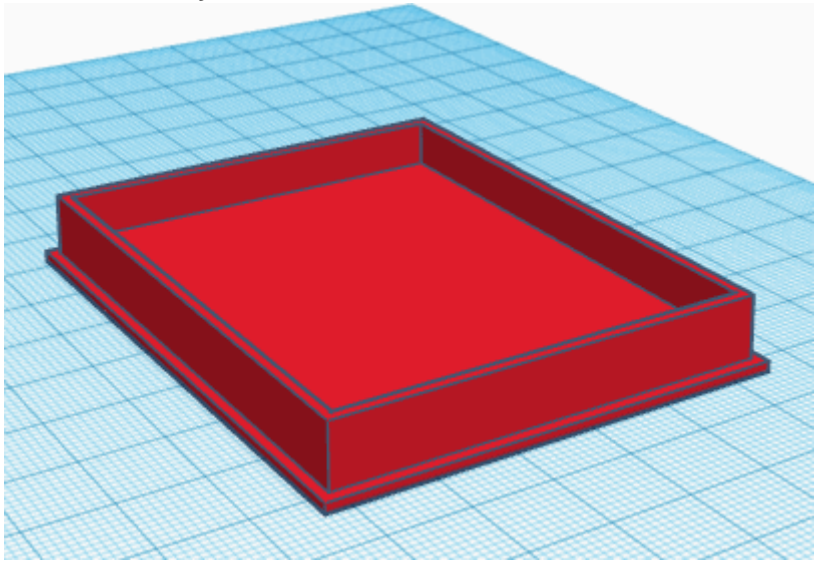
- Ouvrir TinkerCad et démarrer un nouveau projet.
- Placer un cube de base de 7.5*5.8*3 cm depuis la bibliothèque d'objets.
- Ajouter un deuxième cube de 7.3*5.6*2.8 cm.
- Venir percer le deuxième cube au premier.

- 

**Création du Passage pour le Capteur:**

- Choisir une paroi latérale du boîtier.
- Utiliser l'outil "Percage" pour créer un trou rectangulaire de 2*3 cm.
- Placer le trou à l'emplacement souhaité pour le passage du capteur.

**Conception du Capot:**

- Créer un nouveau cube de 7.3*5.6*0.9 cm au-dessus du boîtier existant.
- Positionner le cube de manière à recouvrir entièrement le boîtier.
- Ajouter un deuxième cube de 7.1*5.4*0.9 cm et venir percer ce dernier au premier grâce à l’outil "perçage".
- Puis ensuite ajouter un cube en dessous de 7.5*5.8*0.1 cm.
- 

**Ajustements Finaux:**

- Vérifier l'alignement du capot avec le boîtier principal.
- Ajuster la position du capot pour qu'il puisse se fermer correctement.
- Réviser les dimensions et l'agencement global pour assurer la cohérence du design.

**Validation et Export:**

- Vérifier visuellement la modélisation en 3D pour s'assurer de la conformité avec les spécifications.
- Exporter le modèle au format souhaité pour une utilisation ultérieure (STL, etc.).

**Conclusion / Rangement / Démontage:**

- Retour sur les notions clés de la séance.
- Q&R pour clarifier les doutes.
- Rangement en fin de séance.
- Éteindre les PC.

# Phase 2 - Impression 3D

**Phase 2 - Impression 3D**

**Objectifs :**

- Acquérir une introduction à la préparation de modèles pour l'impression 3D.
- Maîtriser l'utilisation du logiciel Cura.
- Préparer un modèle 3D pour l'impression avec les paramètres appropriés.

**Compétences techniques :**

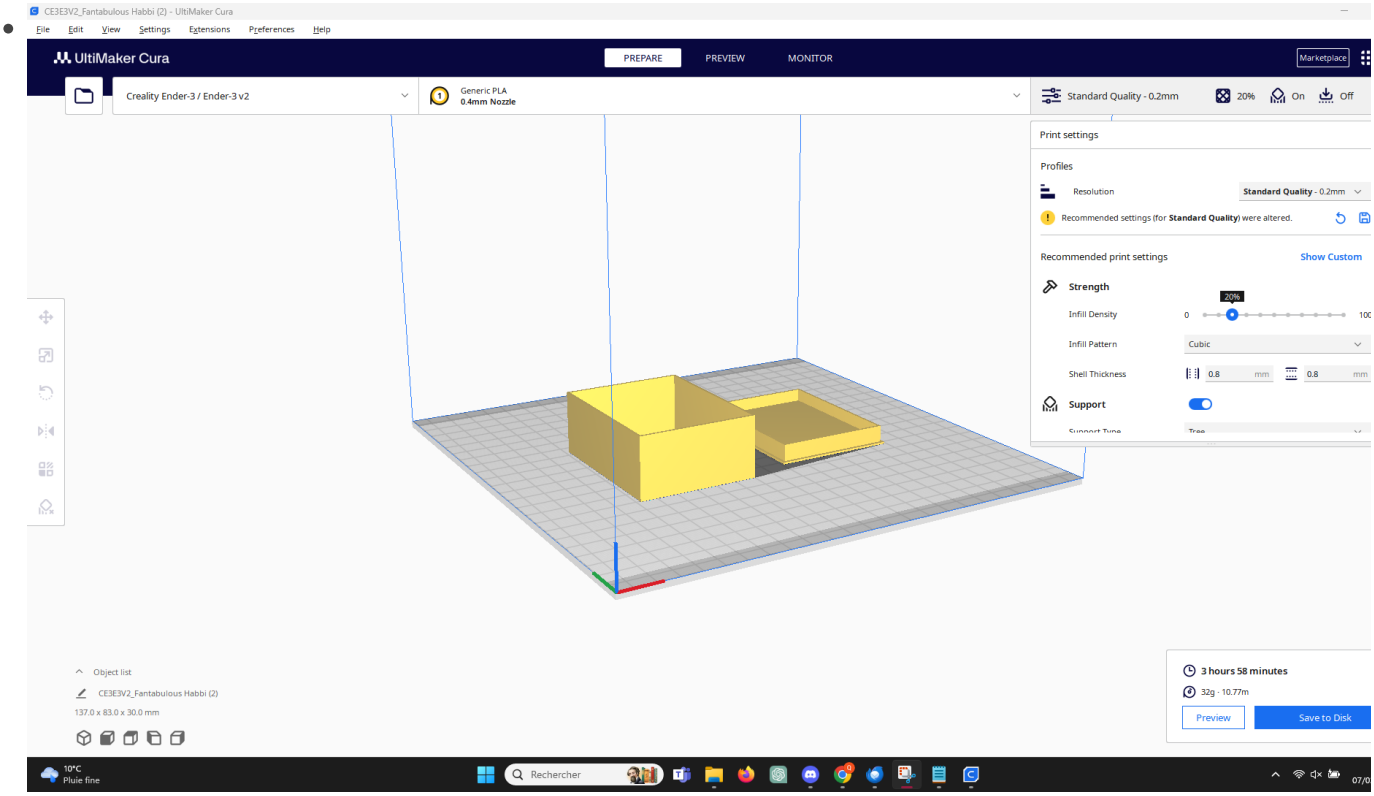
- Utilisation générale de la préparation pour l'impression 3D.
- Utilisation spécifique du logiciel Cura.

**Savoir-être et compétences transversales :**

- Capacité à suivre des instructions détaillées.

**Déroulement de la séance :**

- Consignes de sécurité et précautions matérielles.
- Travail individuel.
- Présentation de l'activité du jour, rappel des objectifs.
- Démonstration interactive de l'utilisation du logiciel Cura.
- Importation du fichier STL exporté depuis TinkerCad dans Cura.
- Configuration des paramètres d'impression :
  - Sélection d'une qualité standard.
  - Activation des supports pour les parties surplombantes.
  - Réglage des autres paramètres selon les besoins spécifiques du projet.
- Aperçu du modèle dans l'environnement d'impression.
- Vérification de la disposition du modèle sur le plateau d'impression.
- Prévisualisation des supports générés par Cura.
- Ajustements éventuels des paramètres pour optimiser l'impression.
- Génération du code G pour l'imprimante 3D.
- Exportation du fichier prêt pour l'impression.




**Conclusion / Rangement :**

- Récapitulation des points clés de la séance.
- Rangement du matériel en fin de séance.

Dans cette phase, les participants vont apprendre à préparer leur modèle 3D pour l'impression en utilisant le logiciel Cura. Ils vont importer leur fichier STL, ajuster les paramètres d'impression, générer le code pour l'imprimante 3D et exporter le fichier prêt pour l'impression. Si des questions se posent ou si vous avez besoin de plus d'informations, n'hésitez pas à demander !

# Phase 3 - Câblage et Programmation

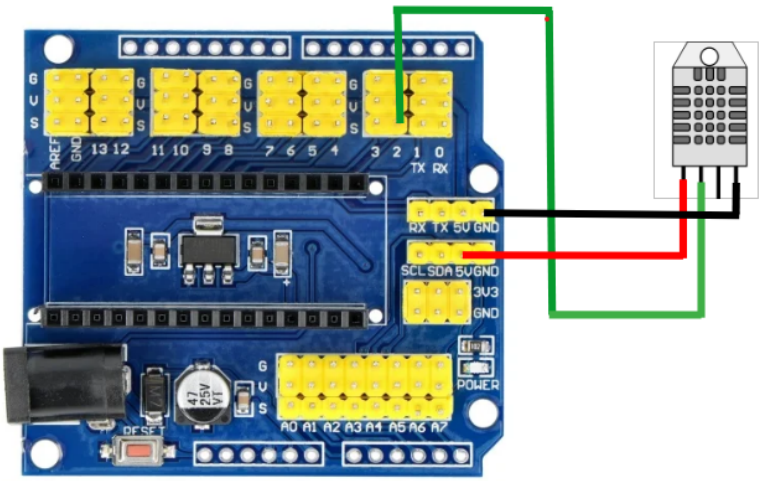
**Phase 3 - Câblage et Programmation de la station**

**Objectifs**

**Compétences techniques:**

- Compréhension du câblage d'un Arduino Nano avec un Sensor Shield et un capteur de température.
- Programmation pour stocker les valeurs de température et d'humidité dans la mémoire de l'Arduino Nano.

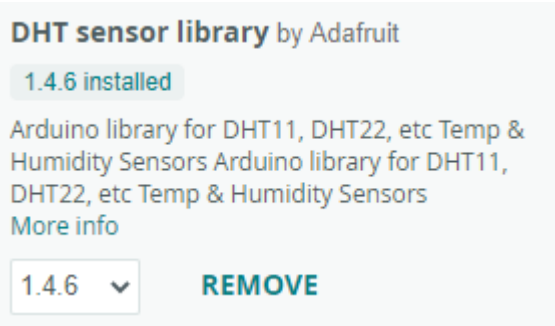
## Déroulement de la séance

1. **Introduction:**
   - Présentation des éléments nécessaires pour le câblage et la programmation.
   - Importance du câblage et de la programmation dans le projet de station météo.
2. **Câblage:**
   - Branchement de l'Arduino Nano au Sensor Shield.
   - Connexion du capteur de température (DHT-11 ou DHT-22) aux broches spécifiques :
     - VCC sur une pin 5V.
     - DAT sur la pin 2.
     - GND sur une pin GND.
   - 

3. **Programmation:**
   - Télécharger la librairie Arduino DHT de Adafruit

     https://www.arduino.cc/reference/en/libraries/dht-sensor-library/

- bien installer la dépendance Adafruit Unified Sensor




- Déclaration des librairies nécessaires et des adresses de stockage dans la mémoire EEPROM.
-
- ```
  #include <DHT.h>
  #include <EEPROM.h>

  #define DHTPIN 2        // La broche à laquelle le capteur DHT est connecté
  #define DHTTYPE DHT22   // Type de capteur DHT (DHT11 dans cet exemple)

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

  }

  DHT dht(DHTPIN, DHTTYPE);

  int adresseEEPROM = 0; // Adresse de la mémoire EEPROM pour stocker la
  température (partie entière)
  int adresseDecimaleEEPROM = 1; // Adresse suivante pour stocker la partie
  décimale
  int adresseHumiditeEEPROM = 2; // Adresse suivante pour stocker l'humidité
  (partie entière)
  int adresseDecimaleHumiditeEEPROM = 3; // Adresse suivante pour stocker la
  partie décimale de l'humidité
  ```

- Fonctionnement de la fonction loop pour mesurer la température et l'humidité.
- Stockage des valeurs dans la mémoire EEPROM avec gestion des adresses.
- Réinitialisation des adresses lorsque la mémoire est pleine.
- ```
  void loop() {
    // Mesurer la température et l'humidité
    float temperature = dht.readTemperature();
  ```

```
  float humidite = dht.readHumidity();

  // Vérifier si les mesures sont valides
  if (!isnan(temperature) && !isnan(humidite)) {
    // Stocker la température en interne (partie entière à adresse, partie
décimale à adresseDecimaleEEPROM)
    EEPROM.write(adresseEEPROM, int(temperature));
    EEPROM.write(adresseDecimaleEEPROM, int((temperature - int(temperature)) *
100));

    // Stocker l'humidité en interne (partie entière à adresseHumiditeEEPROM,
partie décimale à adresseDecimaleHumiditeEEPROM)
    EEPROM.write(adresseHumiditeEEPROM, int(humidite));
    EEPROM.write(adresseDecimaleHumiditeEEPROM, int((humidite - int(humidite)) *
100));

    // Incrémenter les adresses pour la prochaine mesure
    adresseEEPROM += 4;  // Incrémenter de 4 pour laisser un espace pour la
prochaine paire (température et humidité)
    adresseDecimaleEEPROM += 4;
    adresseHumiditeEEPROM += 4;
    adresseDecimaleHumiditeEEPROM += 4;

    // Vérifier si nous avons atteint la fin de l'espace EEPROM
    if (adresseEEPROM >= EEPROM.length()) {
      adresseEEPROM = 0; // Revenir au début de l'espace EEPROM
      adresseDecimaleEEPROM = 1;  // Commencer à l'adresse suivante pour la
partie décimale de la température
      adresseHumiditeEEPROM = 2;  // Commencer à l'adresse suivante pour la
partie entière de l'humidité
      adresseDecimaleHumiditeEEPROM = 3;  // Commencer à l'adresse suivante pour
la partie décimale de l'humidité
    }
  } else {
    Serial.println("Erreur de lecture du capteur.");
  }

  // Attendre avant la prochaine mesure
  delay(600000);  // Attendre temps de secondes entre les mesures (ajustez selon
```

```
vos besoins)
}0
```

- Maintenant il nous faut le code pour lire les valeurs stockées.

```
#include <EEPROM.h>

int adresseEEPROM = 0; // Adresse de la mémoire EEPROM pour la lecture des
données

void lireDonneesEEPROM() {
  Serial.println("Lecture des données depuis l'EEPROM:");

  while (adresseEEPROM < EEPROM.length()) {
    // Lire la partie entière de la température
    int partieEntiereTemperature = EEPROM.read(adresseEEPROM);
    // Lire la partie décimale de la température
    int partieDecimaleTemperature = EEPROM.read(adresseEEPROM + 1);

    // Lire la partie entière de l'humidité
    int partieEntiereHumidite = EEPROM.read(adresseEEPROM + 2);
    // Lire la partie décimale de l'humidité
    int partieDecimaleHumidite = EEPROM.read(adresseEEPROM + 3);

    // Afficher les valeurs lues
    Serial.print("Température: ");
    Serial.print(partieEntiereTemperature);
    Serial.print(".");
    Serial.print(partieDecimaleTemperature);
    Serial.print(" °C, Humidité: ");
    Serial.print(partieEntiereHumidite);
    Serial.print(".");
    Serial.print(partieDecimaleHumidite);
    Serial.println(" %");

    // Incrémenter l'adresse pour la prochaine paire de valeurs
    adresseEEPROM += 4;
  }
}
```

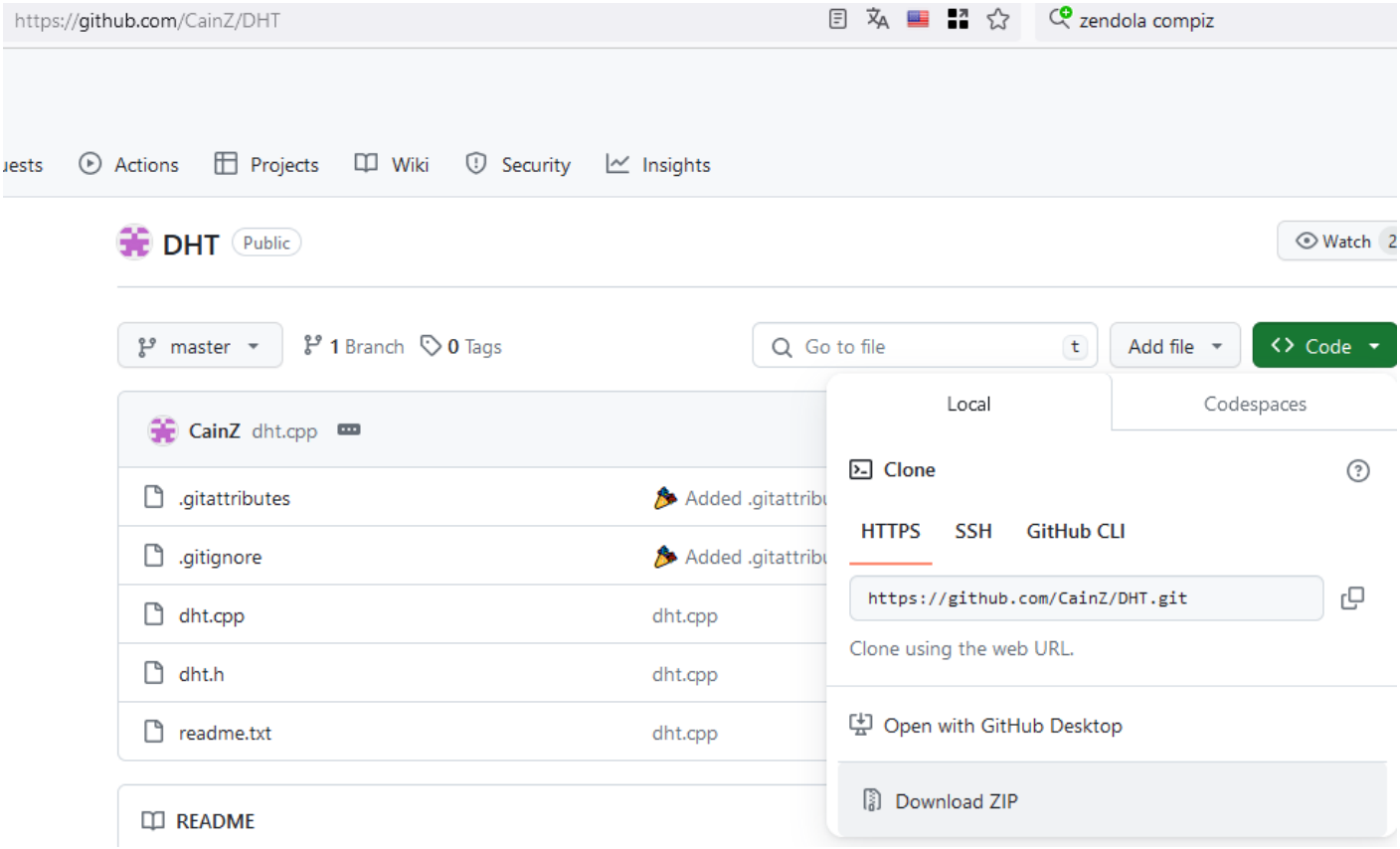
```
void setup() {
  Serial.begin(9600);
  lireDonneesEEPROM();
}

void loop() {
  // Votre code principal ici
}
```

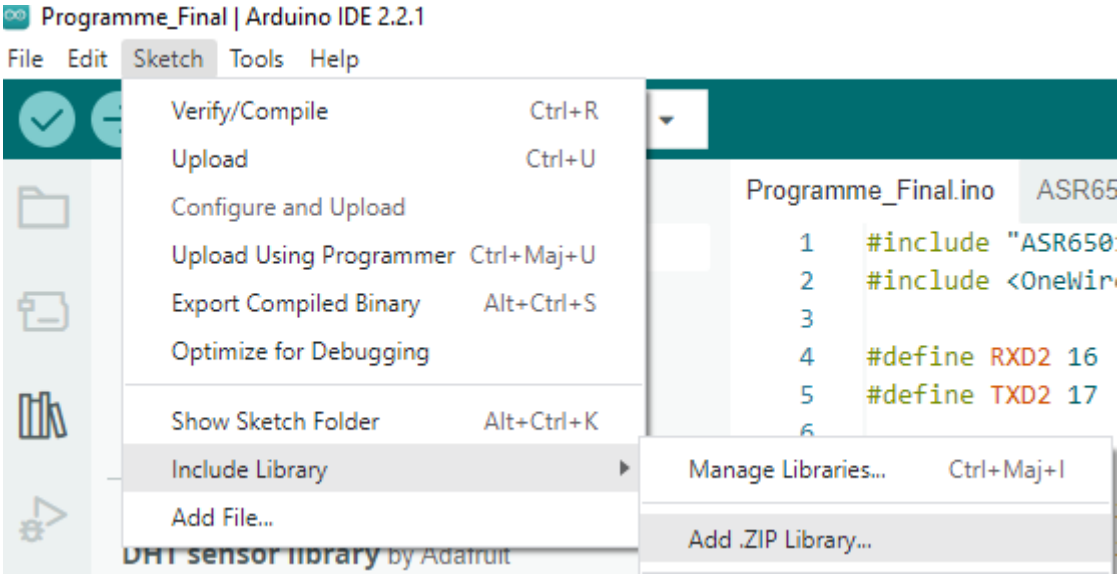
2. **Validation:**
   - Vérification du programme pour s'assurer du bon fonctionnement.
   - Test du stockage des valeurs dans la mémoire EEPROM.
3. **Conclusion:**
   - Récapitulation des points clés de la séance.
   - Réponses aux questions éventuelles.

# Résolution de problèmes

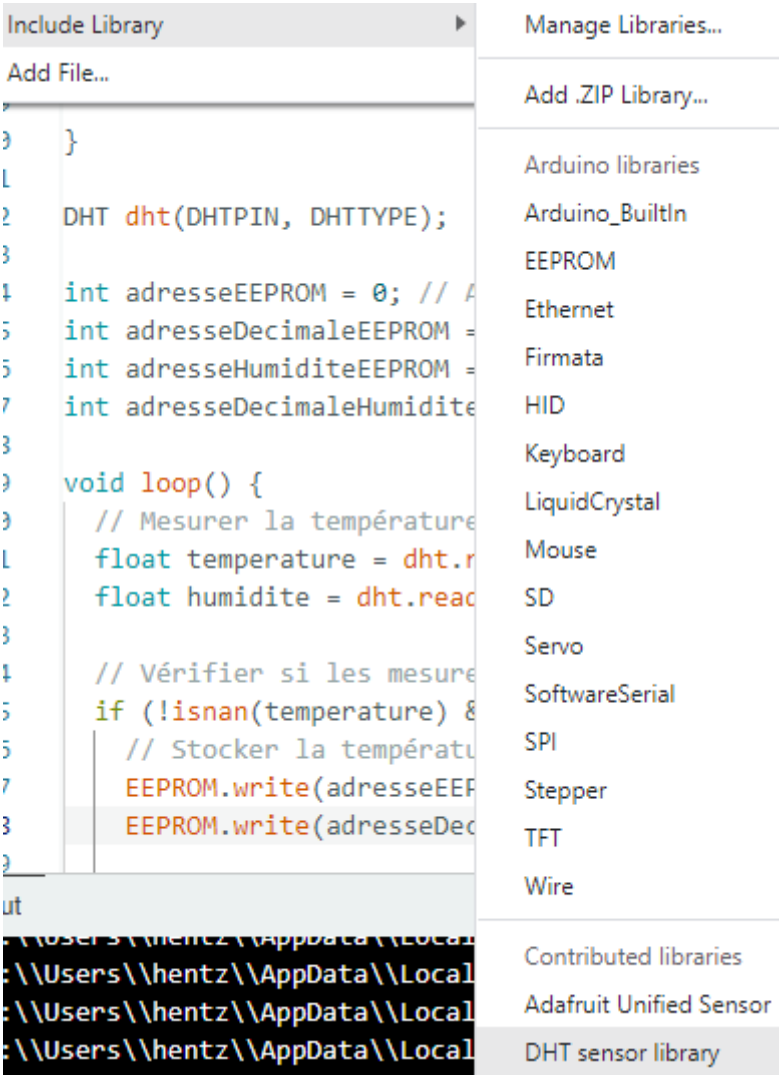
- Si `dht.h` n'est pas reconnu quand vous vérifiez/compilez le fichier, essayer une librairie adaptée à la marque de votre DHT22. Par exemple pour DFRobot :
  - https://wiki.dfrobot.com/DHT22_Temperature_and_humidity_module_SKU_SEN0137#target_2
- Cliquer sur Code > Télécharger ZIP https://github.com/CainZ/DHT




- Ajouter la librairie à la main dans ArduinoIDE




- Vérifier la bonne installation de la librairie, dans "contributed libraries"


Include Library
Add File...
}
DHT dht(DHTPIN, DHTTYPE);
int adresseEEPROM = 0; //
int adresseDecimaleEEPROM =
int adresseHumiditeEEPROM =
int adresseDecimaleHumidite
void loop() {
// Mesurer la température
float temperature = dht.r
float humidite = dht.reac
// Vérifier si les mesure
if (!isnan(temperature) &
// Stocker la températu
EEPROM.write(adresseEEP
EEPROM.write(adresseDec
ut
:\\Users\\hentz\\AppData\\Local
:\\Users\\hentz\\AppData\\Local
:\\Users\\hentz\\AppData\\Local
Manage Libraries...
Add .ZIP Library...
Arduino libraries
Arduino_BuiltIn
EEPROM
Ethernet
Firmata
HID
Keyboard
LiquidCrystal
Mouse
SD
Servo
SoftwareSerial
SPI
Stepper
TFT
Wire
Contributed libraries
Adafruit Unified Sensor
DHT sensor library