Détection et analyse environnementale - Préface

Voici la deuxième partie de la ressource sur l'utilisation du Micro:bit en classe de sciences. Elle a été préparée pour être utilisée par les enseignants qui dispensent le cours de sciences SNC1W décloisonné en Ontario, Canada. Elle peut être adaptée pour répondre aux besoins des autres cours de sciences également. Les enseignants sont libres de modifier ces notes selon leurs besoins. Ces ressources sont disponibles sur csintegration.ca

La première partie aborde les bases du microcontrôleur Micro:bit et du langage de programmation Micro Python. Elle explique également comment utiliser l'éditeur Micro Python, le capteur de température et les activités de journalisation de température.

1. Capteur de lumière

La lumière est un paramètre que nous souhaitons mesurer et contrôler. La plupart des bâtiments intelligents utilisent un contrôle automatique de la lumière qui ouvre et ferme les stores afin de réduire les coûts énergétiques liés à l'éclairage. La lumière joue également un rôle dans la croissance des plantes. Dans ce laboratoire, nous mesurerons l'intensité lumineuse sur une échelle relative. Tout d'abord, nous obtiendrons les niveaux de lumière à partir du simulateur, puis nous connecterons une cellule photoélectrique pour mesurer la lumière ambiante dans une pièce ou un laboratoire.

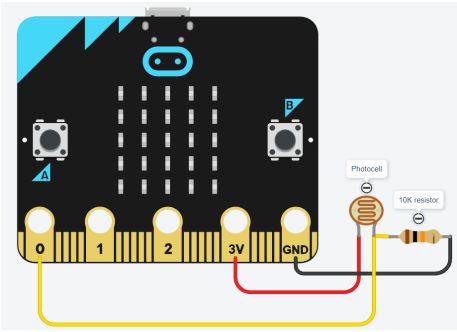
1.1 Utilisation du simulateur Micro Python

Lancez l'éditeur MicroPython. Cliquez sur "Play" pour voir l'icône en forme de cœur et le message "Hello". Cliquez sur "Light Level" dans le panneau de gauche et faites glisser et déposez le bout de code d'une ligne dans la zone de l'éditeur. Si vous appuyez sur "Play", vous verrez un niveau de lumière de 127 sur la LED défilante. Ce niveau peut être varié de 0 (sombre) à 255 (le plus lumineux) en déplaçant le curseur sous le simulateur.

Si ce code est envoyé à un Micro:bit physique, nous pouvons mesurer le niveau de lumière ambiante dans une pièce. Cela sera un nombre compris entre 0 et 255. Les 25 LED sur le Micro:bit sont utilisées comme capteur de lumière pour fournir une meilleure estimation de la lumière ambiante.

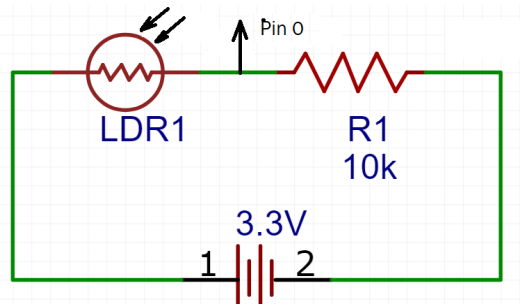
.

```
# Les imports doivent être placés en haut.
from microbit import *

# Le code dans une boucle 'while True:' se répète indéfiniment.
while True:
    display.show(Image.HEART)
    sleep(1000)
    display.scroll('Hello')

display.scroll(display.read_light_level())
```



1.2 Utilisation d'une cellule photoélectrique

Une cellule photoélectrique ou une résistance photo-dépendante (LDR) est un type de résistance. Comme une résistance, elle n'a pas de polarité, c'est-à-dire qu'il n'y a pas de '+' ou '-' à ses extrémités. Elle peut donc être connectée dans un circuit dans n'importe quel sens. Sa résistance sera d'environ 2000 ohms en pleine lumière et peut atteindre jusqu'à 10000 ohms dans des conditions sombres. Ces valeurs peuvent varier d'une LDR à l'autre.



Le sulfure de cadmium (CdS) est le composé chimique présent dans une LDR qui modifie sa résistance lorsqu'il est exposé à la lumière. Les schémas de câblage et les diagrammes schématiques sont présentés. Il s'agit d'un circuit en série connecté à 3,3 V à partir du Micro:bit. En tant qu'exercice supplémentaire, cela peut être enseigné dans le cadre du cours "Physique D - Caractéristiques et applications de l'électricité", attente spécifique D2.6.

> *"construire des circuits en série et en parallèle pour comparer le courant électrique, la différence de potentiel et la résistance dans ces types de circuits."*

Sous une lumière vive : Supposons que la résistance de la LDR soit de 2000 ohms. La résistance totale dans le circuit en série est de 2000 + 10000 = 12000 ohms. En utilisant la loi d'Ohm, le courant dans le circuit en série est de : 3,3 V / 12000 ohms = 0,000275 A. Ce courant générera une tension à travers R1 = 10000 * 0,000275 = 2,75 V.

La broche 0 du Micro:bit lira cette tension et la convertira en un nombre de 0 à 1023. Le Micro:bit est un dispositif numérique qui fonctionne en binaire (0, 1). Il utilise 10 chiffres pour représenter les tensions de 0 V à 3,3 V. 0 V correspondra à 0000000000 (dix zéros) et 3,3 V correspondra à 1111111111 (10 uns) ou 1023. 2,75 V deviendra 853 ou 1101010101 en binaire. C'est-à-dire (1023 / 3,3) * 2,75 = 853.

Sous une faible lumière : La résistance de la photocellule sera de 10000 ohms. La résistance totale dans le circuit en série est maintenant de 10000 + 10000 = 20000 ohms. En utilisant la loi d'Ohm, le courant dans le circuit en série est de : 3,3 V / 20000 ohms = 0,000165 A. Ce courant générera une tension à travers R1 = 10000 * 0,000165 = 1,65 V. Le Micro:bit sur la broche 0 convertira cette tension en un nombre de 512 en utilisant le calcul indiqué ci-dessus.

Ici, 853 est pour une lumière vive et 512 est pour une faible lumière. Ces nombres changeront en fonction de la lumière ambiante et du type de photocellule utilisée. Le bout de code ci-dessous affichera la valeur de la lumière sur le moniteur série situé

sous la zone du simulateur : appuyez sur *"Show Serial"* pour ouvrir le moniteur série. Si nécessaire, ces valeurs peuvent être converties sur une échelle de 0 à 100 en les multipliant par (100 / 1023).

```
print(pin0.read_analog()*(100 / 1023))
```

853 et 512 deviendront respectivement 83 % et 50 %. Si vous éclairez le capteur avec la lampe de poche d'un téléphone portable, la luminosité augmentera encore.

```
from microbit import *
while True:
    print(pin0.read_analog())
    sleep(1000)
```

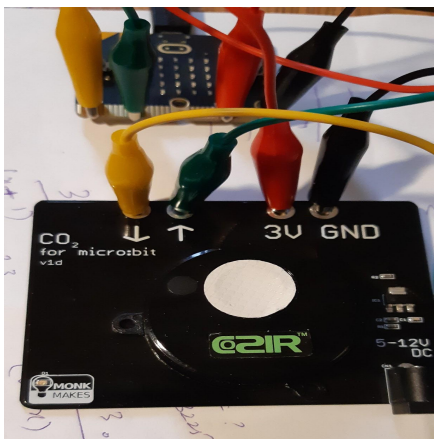
2. Capteur de CO2

Ce capteur provenant de https://monkmakes.com/mb_co2.html, du projet "Let Us Talk Science Living", peut fournir des mesures de CO2, de température et d'humidité. Ce site propose des liens pour acheter ce capteur et obtenir plus d'informations à son sujet.

Dans ce laboratoire, nous allons mesurer les niveaux de CO2. Cette mesure peut être nécessaire dans des espaces de bureau, industriels et de serre. Ce laboratoire répond à l'attente spécifique B2.3.

"*décrire et comparer les processus de la respiration cellulaire et de la photosynthèse, et expliquer l'importance de leur relation de complémentarité pour l'équilibre dynamique des écosystèmes.*" de la compétence B. Durabilité des écosystèmes et changements climatiques.

Connectez l'alimentation 3V et la masse (GND) du capteur depuis le Micro:bit. Connectez la broche 0 du Micro:bit à la flèche 'IN' et la broche 1 à la flèche 'OUT' sur le capteur.



L'USB alimente le Micro:bit, qui alimente à son tour le capteur de CO2. Si l'USB n'est pas connecté, alimentez le capteur de CO2 à l'aide du connecteur d'alimentation 6V ou 9V, qui alimente ensuite le Micro:bit. Ce capteur risque de ne pas fonctionner correctement s'il est alimenté par le pack de piles 3V du Micro:bit. Il est important de le noter lorsque vous enregistrez les valeurs de CO2 en mode autonome.

De plus, l'exemple de code sur le site Monk ne fonctionne probablement pas en raison d'un changement de version. Utilisez la version modifiée du code donnée ci-dessous.

Ce capteur est un capteur intelligent, c'est-à-dire qu'il peut fournir les valeurs directement, contrairement au capteur de lumière où nous devons lire les tensions et les convertir en niveaux de lumière. Le Micro:bit va "demander" au capteur de fournir les valeurs de CO2, de température ou d'humidité. Et le capteur répondra avec les valeurs qui peuvent être affichées directement sur la LED. Le moniteur série ne peut pas être utilisé lorsque le capteur de CO2 est connecté. Il est important de le noter lors de l'utilisation de ce capteur de CO2.

À l'extérieur, la lecture sera d'environ 400, dans une pièce avec quelques personnes elle sera d'environ 1000. Le tableau ci-dessous montre des valeurs typiques des niveaux de CO2. Les valeurs sont en ppm ou parties par million.

Les niveaux de CO2 dans l'air et les problèmes de santé potentiels sont les suivants :

400 ppm	**Niveau moyen de l'air extérieur.**
400–1,000 ppm	**Niveau typique trouvé dans les espaces occupés avec un bon renouvellement d'air.**
1,000–2,000 ppm	**Associé à des plaintes de somnolence et d'air vicié.**
2,000–5,000 ppm	**Associé à des maux de tête, de la somnolence et de l'air stagnant, vicié. Une mauvaise concentration, une perte d'attention, une augmentation du rythme cardiaque et une légère nausée peuvent également être présentes.**
5,000 ppm	**Cela indique des conditions d'air inhabituelles où des niveaux élevés d'autres gaz pourraient également être présents. Une toxicité ou une privation d'oxygène pourrait survenir. Il s'agit de la limite d'exposition admissible pour les expositions quotidiennes sur le lieu de travail.**
40,000 ppm	**Ce niveau est immédiatement nocif en raison d'une privation d'oxygène.**

Adapté de : Département des services de santé du Wisconsin

Pour étalonner le capteur, laissez-le alimenté pendant 24 heures. Il s'étalonnera automatiquement. Vous pouvez également utiliser le code d'étalonnage ci-dessous. Il n'y a aucun mal à effectuer un étalonnage occasionnel ! Vous trouverez plus d'informations sur ce capteur COZIR sur le lien suivant :

https://www.airtest.ca/support/datasheet/COZIRSerialInterface.pdf

2.1 Étalonnage

Si vous exécutez le programme suivant, le capteur sera étalonné et vous verrez une valeur d'environ 30000 sur les LEDs. J'ai obtenu une valeur de 30701. Après l'étalonnage, de l'air frais aura une mesure d'environ 400 ppm de CO2. Si vous obtenez xxx à la place d'un nombre sur l'affichage, cela signifie que le capteur ne communique pas avec le Micro:bit. Vérifiez les connexions du câblage et des pinces crocodile.

Les commentaires dans le code (le caractère # commence un commentaire) expliquent ce que fait le code. Ils ne sont pas nécessaires pour exécuter le programme.

```
from microbit import *
uart.init(tx=pin0, rx=pin1)

while True:
    response = ""
    # demande au capteur de passer en mode d'étalonnage
    uart.write('G\r\n')
    # attendre 0.1 seconde
    sleep(100)
    if uart.any():
        # lire ce que le capteur indique
        response = str(uart.read())
        # le capteur devrait afficher au moins 8 caractères
        # sinon, afficher xxx sur les LEDs
        if len(response) < 8:
            response = "xxx"
        # extract the numbers from response
        response = response[:len(response)-5]
        startChar = response.find(" ")
        response = response[startChar+2:]
    display.scroll(response)
    sleep(500)
```

2.2 Mesure de CO2

Pour mesurer les niveaux de CO2, utilisez le code suivant. Ce code est exactement le même que celui utilisé pour l'étalonnage, sauf que nous envoyons "Z" au lieu de "G" pour demander au capteur de nous envoyer les valeurs de CO2 cette fois-ci. Mes lectures étaient de 00402 après l'étalonnage, elles sont montées à 01093 après que j'ai expiré dans le capteur pendant une minute, puis elles sont descendues à 00453 après quelques minutes. Laissez au capteur quelques minutes pour enregistrer de nouvelles valeurs.

```
from microbit import *
uart.init(tx=pin0, rx=pin1)

while True:
    response = ""
    uart.write('Z\r\n')
    sleep(100)
    if uart.any():
        response = str(uart.read())
        if len(response) < 8:
            response = "xxx"
        response = response[:len(response)-5]
        startChar = response.find(" ")
        response = response[startChar+2:]
    display.scroll(response)
    sleep(500)
```

Le code de l'annexe de ce document peut être utilisé pour mesurer les valeurs de CO2, de température et d'humidité. Le CO2 sera toujours affiché sur les LEDs. En appuyant sur le bouton A (bouton de gauche), vous obtiendrez la température, et en appuyant sur le bouton B (bouton de droite), vous obtiendrez les valeurs d'humidité.

2.3 Enregistrement de CO2

Cette activité couvre les attentes spécifiques suivantes : A. Habiletés liées aux STIM, carrières, et liens connexes

A1.1: appliquer une démarche de recherche et les habiletés connexes pour effectuer des recherches afin d'établir des liens entre celles-ci et les concepts scientifiques à l'étude.

A1.2: appliquer une démarche expérimentale et les habiletés connexes pour effectuer des expériences afin d'établir des liens entre ses observations et conclusions et les concepts scientifiques à l'étude.

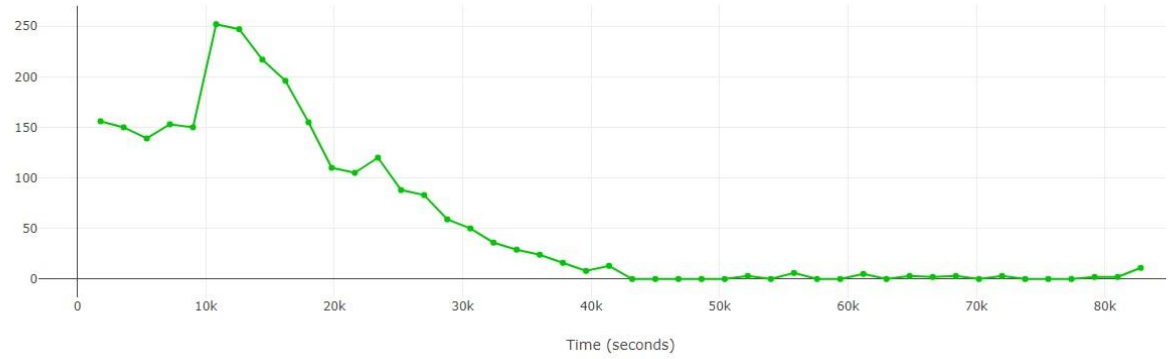
A1.3: appliquer un processus de design en ingénierie et les habiletés connexes pour concevoir, construire et tester des dispositifs, des modèles, des structures et/ou des systèmes.

A1.4: appliquer des habiletés en codage pour examiner et modéliser des concepts scientifiques et des relations connexes.

Cette activité peut être enrichie pour expliquer les changements climatiques et la nécessité de réduire les niveaux de CO2 existants de 500 à 350 ppm afin de ralentir le réchauffement climatique.

Méthodologie : Le Micro:bit et le capteur de CO2 ont été connectés à un port USB sur un ordinateur de bureau dans une salle de classe de 20 pieds sur 12 pieds. Il a été programmé pour enregistrer les niveaux de CO2 toutes les 30 minutes de 9h à 21h, capturant ainsi 48 points de données. Il y avait deux tours de culture avec des légumes-feuilles : de la laitue et du chou frisé. Veuillez voir les photos. Les lumières de croissance LED étaient allumées pendant toute la durée de l'expérience.

Le graphique montre les niveaux de CO2 à intervalles de 3 heures. Le temps sur l'axe des abscisses est exprimé en secondes écoulées. L'axe des ordonnées représente les niveaux de CO2 en ppm. Il est important de noter que le niveau de CO2 était proche de zéro après 21h. Le fichier brut des données téléchargé depuis le Micro:bit est disponible ici.



Le code ci-dessous utilisé pour enregistrer les niveaux de CO2 est similaire à celui que nous avons utilisé précédemment pour mesurer les niveaux de CO2. Le panneau de référence à gauche de l'éditeur MicroPython contient toute la documentation nécessaire sur la façon de faire un enregistrement de données. Quelques lignes supplémentaires ont été ajoutées pour enregistrer les données de CO2 toutes les 30 minutes.

La capture des données commence lorsque le Micro:bit est alimenté et se termine lorsqu'il est éteint ou débranché de l'ordinateur. Après avoir débranché le Micro:bit, connectez-le à un ordinateur et il apparaîtra comme un lecteur USB externe sur l'ordinateur. Si nous ouvrons les fichiers, ils contiennent les données et le graphique indiqués ci-dessus. Nous pouvons également télécharger les données brutes au format CSV (valeurs séparées par des virgules) pour une analyse ultérieure.

```
from microbit import *
import log

# cette étiquette CO2 doit correspondre à la clé, nom de la paire
de valeurs ci-dessous
# secondes signifie secondes écoulées PAS secondes dans le temps

log.set_labels('co2',timestamp=log.SECONDS)

uart.init(tx=pin0, rx=pin1)
co2 = 0
response = ''

# log every 30 minutes

@run_every(min=30)
def log_data():
    log.add({
```

```
       'co2' : get_co2(),

    })
def check_incoming_messages():
 global co2, response
 response = ''
 if uart.any():
     response = str(uart.read())
     if len(response) < 8:
         return
     response = response[5:10]

def get_co2():
     global response
     uart.write('Z\r\n')
     sleep(100)
     check_incoming_messages()
     co2 = response
     return co2


while True:
    sleep(100000)
```

3. Conseils pour prendre soin de votre Micro:bit

- ★ Le connecteur Micro USB sur le Micro:bit est délicat. Les branchements répétés peuvent endommager le connecteur. Il est recommandé de laisser le câble USB connecté à l'extrémité du Micro:bit et de simplement le brancher et le débrancher du PC au besoin.
- ★ Le connecteur de la batterie est également délicat. Il est recommandé de laisser le câble de la batterie connecté au Micro:bit et de simplement retirer les 2 piles AAA lorsque l'appareil n'est pas utilisé.
- ★ Rappelez aux étudiants de télécharger le programme par défaut avant de rendre le Micro:bit. De cette manière, les enseignants pourront vérifier l'état du Micro:bit.

Annexes

A.1 Composants et liens utiles

Components are from abra-electronics.com, an electronics retailer based in Montreal.

- ★ micro:bit v2 Go Paquet
- ★ LM35DZ CI, Capteur de Température
- ★ CDS004-5001 Lumière Photocellule
- ★ 10K Ohm Résistance
- ★ Fils sur Bobines 22 AWG Solide Cela vient dans une petite boîte avec six bobines de fils de différentes couleurs pour une distribution facile.
- ★ Pince-Dénudeur réglable 10-30 AWG Ceci est un outil pratique pour dénuder les fils. Utilisez l'encoche pour maintenir fermement le fil et tirez pour dénuder le fil. Il nécessite la bonne quantité de pression, une pression excessive risque de couper le fil.
- ★ CAPTEUR DE CO2 pour MICRO:BIT (MONKMAKES)

Lines utiles

- ★ BBC Micro:bit
- ★ Micro:bit pour les enseignants
- ★ Éditeur Micro Python
- ★ https://microbit-micropython.readthedocs.io/en/v2-docs/umentation

A.2 Capteur de CO2 pour mesurer le CO2, la température et l'humidité

La valeur de CO2 sera toujours affichée. En appuyant sur le bouton A, la température sera affichée avec la lettre "T", tandis qu'en appuyant sur le bouton B, l'humidité relative sera affichée avec "HR=".

```
from microbit import *
uart.init(tx=pin0, rx=pin1)
co2 = 0
tempC = 0
rh = 0
response = ""

def check_incoming_messages():
 global co2, tempC, rh, response
 if uart.any():
     response = str(uart.read())
     if len(response) < 8:
         return
     # Supprimer les caractères \r\n de la réponse
     # Vous aurez Z, T ou H espace #####
     # où ##### est la valeur renvoyée par le capteur
     response = response[:len(response)-5]
     # Les valeurs sont renvoyées après l'espace
     startChar = response.find(" ")
     response = response[startChar+2:]
     display.scroll(response,100)

def get_co2():
     global response
     uart.write('Z\r\n')
     sleep(100)
     check_incoming_messages()
     co2 = response
     return co2


def get_tempC():
     global response
     uart.write('T\r\n')
     sleep(100)
     check_incoming_messages()
     tempC = (int(response)-1000)/10
     return tempC

def get_rh():
     global response
     uart.write('H\r\n')
```

```
    sleep(100)
    check_incoming_messages()
    rh = int(response) / 10
    return rh


while True:
 # La valeur de CO2 sera toujours affichée
 display.scroll(get_co2(),100)
 sleep(500)
 if button_a.was_pressed():
    display.scroll('T=' + str(get_tempC()),100)
 sleep(500)
 if button_b.was_pressed():
    display.scroll('RH=' + str(get_rh()),100)
 sleep(500)
```