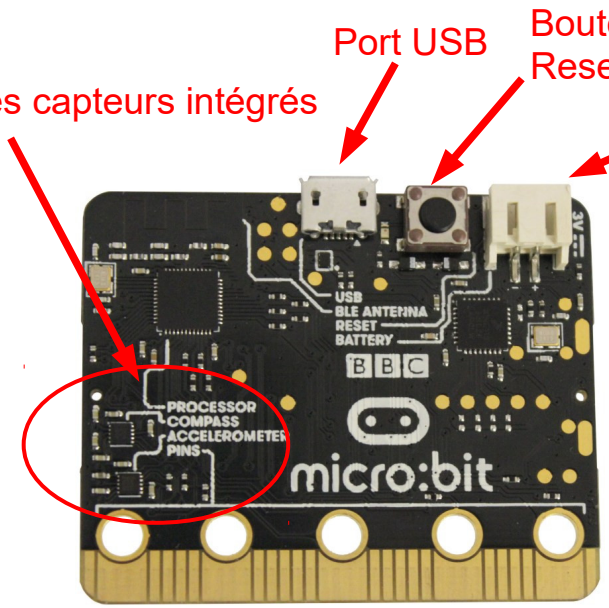
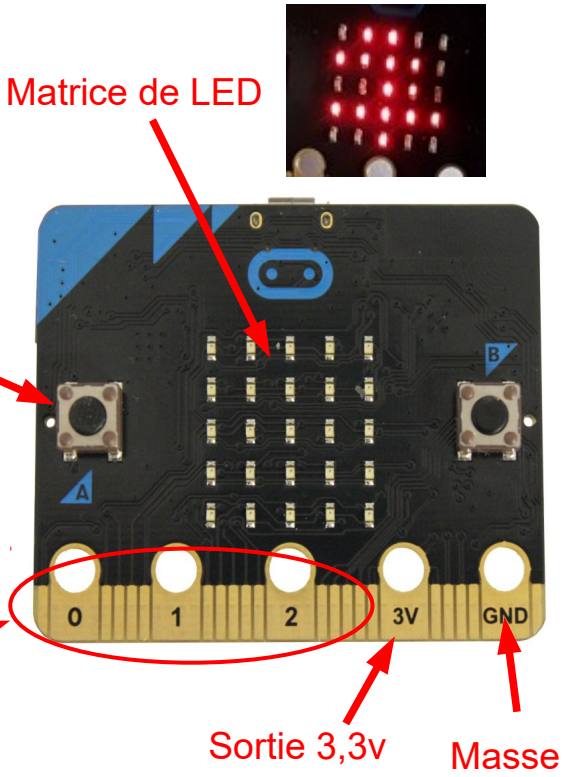
La carte micro:bit

https://microbit.org/fr/

La carte **micro:bit** est une carte électronique (nano-ordinateur) créée par la BBC en 2016 pour promouvoir l'apprentissage du codage auprès des élèves. C'est une carte micro-contrôleur, programmable, ayant des capteurs et actionneurs intégrés. Elle est plus puissante que la carte **Arduino Uno** .



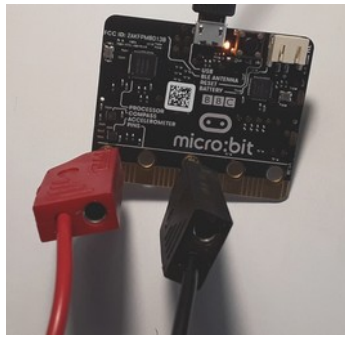
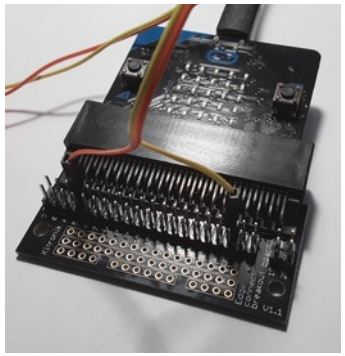
La carte comprend :
- une connexion **Bluetooth**,
- 2 **boutons poussoirs**,
- un **magnétomètre 3D**,
- un **accéléromètre 3D,**
- un **thermomètre**,
- une **LED**
- une **matrice 5 par 5 de LED.**



La carte peut fonctionner de manière autonome ou elle peut rester connectée en USB à un ordinateur. Elle peut alimenter des capteurs en 3,3V. Quand on la branche à un ordinateur, elle est détectée comme une carte SD ou une clé USB : il n'y a donc pas de drivers à installer (sous win10) et il suffit simplement de déposer le micro-programme (fichier .hex) dans sa mémoire. La carte exécute ensuite ce programme.

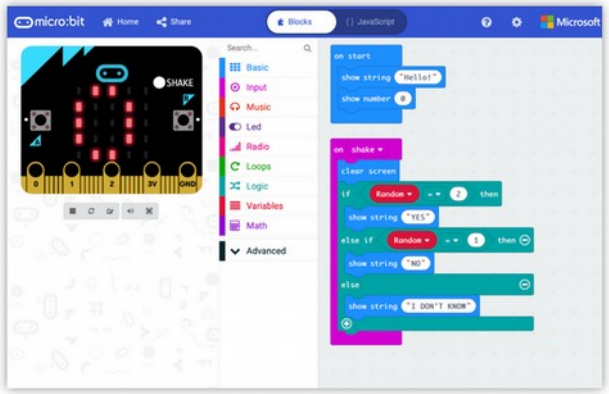
La carte peut-être programmée dans un langage dérivé de **Python**, mais très proche : le **Micropython**. Toutefois, la bibliothèque **matplotlib** et certains modules comme **numpy** ne sont pas fonctionnels avec la carte Micro:bit.

Les capteurs peuvent se brancher de différentes manières :

Par pinces crocodiles (pour 3 entrée ou sorties seulement)		Par fiches bananes 4mm (pour 3 entrée ou sorties seulement)
	Platine spéciale (shield) Il existe de nombreux modèles disponibles comme le système Grove par exemple.	

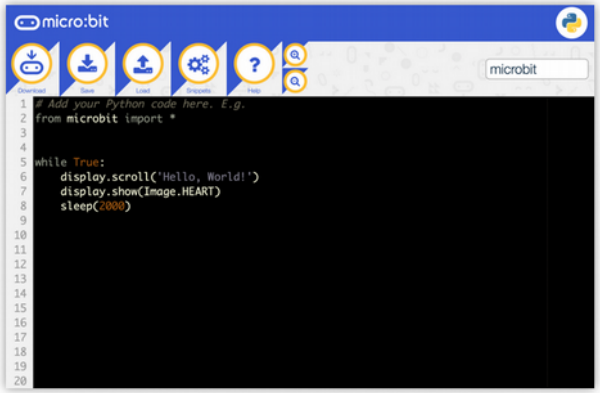
La carte peut être pilotée de différentes manières :

MakeCode



https://makecode.microbit.org/#lang=fr
Éditeur en ligne, en langage bloc.

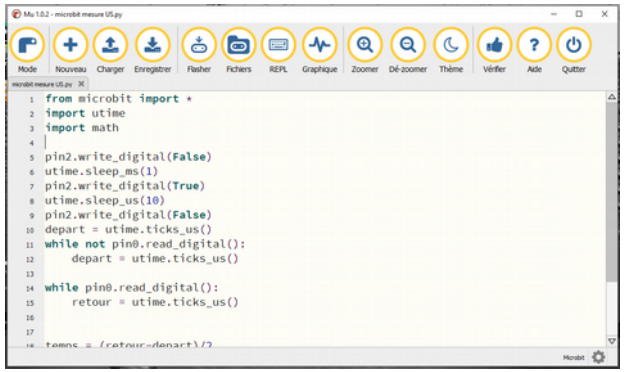
Python Editor



```
# Add your Python code here. E.g.
from microbit import *


while True:
    display.scroll('Hello, World!')
    display.show(Image.HEART)
    sleep(2000)
```

https://python.microbit.org/v/1.1
Éditeur en ligne, en langage python.
Cette solution ne permet pas d'afficher de valeurs mesurées.

MU Editor



```
from microbit import *
import utime
import math

pin2.write_digital(False)
utime.sleep_ms(1)
pin2.write_digital(True)
utime.sleep_us(10)
pin2.write_digital(False)
depart = utime.ticks_us()
while not pin0.read_digital():
    depart = utime.ticks_us()

while pin0.read_digital():
    retour = utime.ticks_us()
```

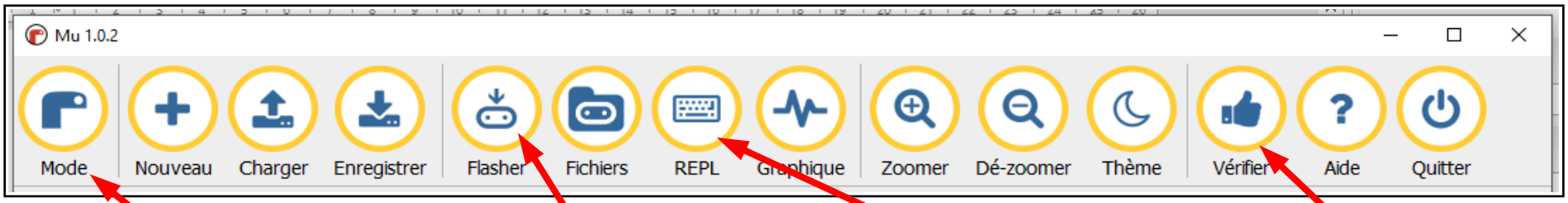
https://codewith.mu/
Logiciel à installer, programmation en langage python.

Utilisation du Logiciel MU Editor

MU editor est un logiciel permettant de déposer directement le microprogramme sur la carte, sans avoir à passer par l'étape manuelle de dépôt du fichier .HEX et il permet également de recevoir et d'envoyer des données en temps réel à la carte (on appelle cela la **console REPL**). Aller sur le site https://codewith.mu/ et suivre les instructions pour l'installation. Ouvrir **MU editor.**

(Pour les versions inférieur à windows10, il faut également installer le Windows Serial driver :

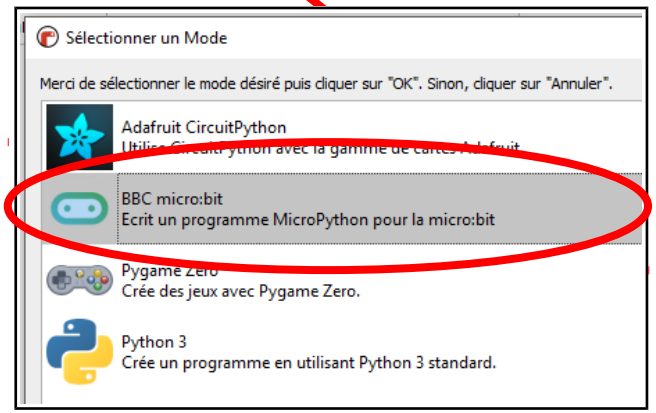
https://os.mbed.com/docs/mbed-os/v5.7/tutorials/windows-serial-driver.html)



Dépose le microprogramme sur la carte.

Ouvre la console pour afficher les mesures (console)

Permet de vérifier les erreurs de codage (débuggage).



1) Choisir "**mode**" puis "**BBC micro:bit**".
2) Faire "**nouveau**", puis "**enregistrer**" pour sauvegarder.
3) Taper le code.
4) Faire "**vérifier**" et suivre les conseils données en cas d'erreurs ou de problèmes de mises en forme du code.
5) Déposer le microprogramme sur la carte : **"flasher"**
Le programme démarre, faire "**REPL**" pour afficher la console si besoin. Dans ce cas il faut appuyer sur le bouton RESET de la carte pour relancer le programme et l'affichage.

Remarque : Pour chaque nouvelle modification du programme, il faut fermer la console « REPL », et « flasher » de nouveau pour déposer le programme modifié sur la carte.

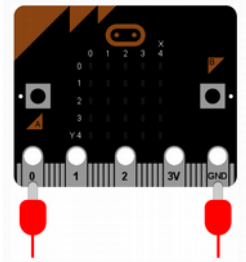
Exemples de programmes

Les 3 exemples suivants sont très succinctement expliqués. Les principales commandes « **micropython** » pour **micro:bit** se trouvent en ligne, ne pas hésiter à consulter : https://microbit-micropython.readthedocs.io/en/v1.0.1/index.html Les commandes **Python** génériques ne sont pas non plus détaillées. https://docs.python.org/fr/3/index.html

Programme 1 :

Affichage de texte et clignotement d'une LED

Une LED est branchée entre les bornes « GND » et 0. Elle va clignoter 10 fois et le texte « bonjour » va s'afficher dans la console et défiler sur la matrice de LED.



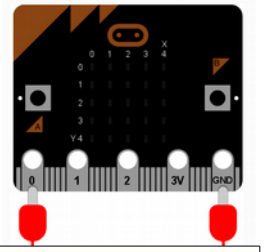
```
from microbit import *

while True:
    print("bonjour")
    display.scroll("bonjour")
    for i in range(0,10):
        pin0.write_digital(1)
        sleep(100)
        pin0.write_digital(0)
        sleep(100)

    sleep(1000)
```

print() affiche le résultat dans la console (repl)
display.scroll() fait défiler le texte sur la matrice de led.
pin0.write_digital(1) impose 3V sur la sortie 0.
pin0.write_digital(0) abaisse la tension à 0V.

Programme 2 :

Générer des sons de différentes fréquences

Le programme va générer 10 sons de durée 2s de la fréquence 100 Hz, 200 Hz, 300Hz… jusqu'à 10000 Hz.
Un tweeter est branché entre les bornes « gnd » et 0



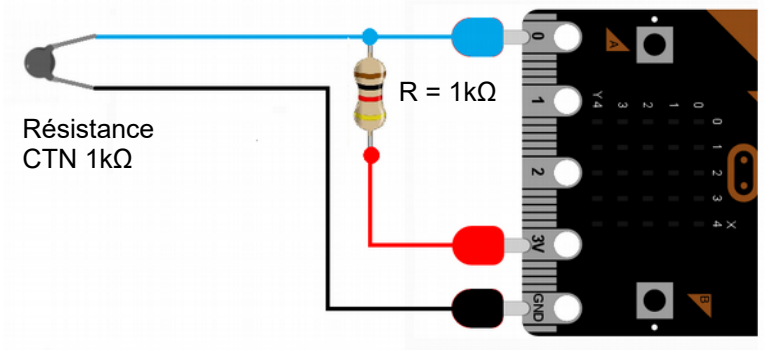
```
from microbit import *
import music

for i in range(1, 11):
    music.pitch(100*i, 2000, pin0)
    print('son de fréquence', i*100, ' Hz')
```

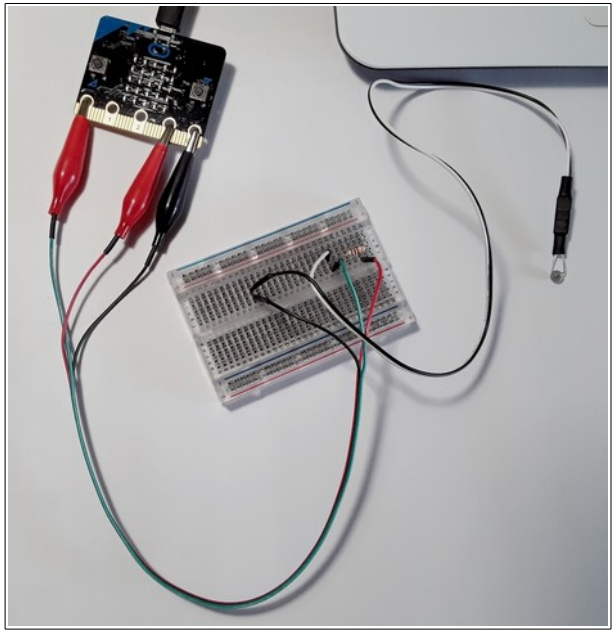
music.pitch(*fréquence, durée en ms, numéro de sortie)* permet de générer des sons simples de fréquences maximales 10 kHz.

Programme 3 :

Mesure d'une température par une résistance CTN et comparaison au capteur interne de la carte.

```
from microbit import *
import math

def temp(rctn):
    return math.ceil(-27*math.log(rctn)+215)

while True:
    Umes = pin0.read_analog()
    Rctn = 1000*Umes/(1023-Umes)
    TemperatureCTN = temp(Rctn)
    print(TemperatureCTN)
    print(temperature())
    sleep(1000)
```





math.ceil() renvoie la valeur de l'entier supérieur.
math.log() calcule le logarithme népérien ln !!!!

pin0.read_analog() mesure la tension de la borne 0 et renvoie une valeur comprise entre 0 et 1023 qui correspond à une valeur comprise entre 0 et 3,3V.

La fonction **temperature()** renvoie la valeur la température mesurée par la capteur interne de la carte.