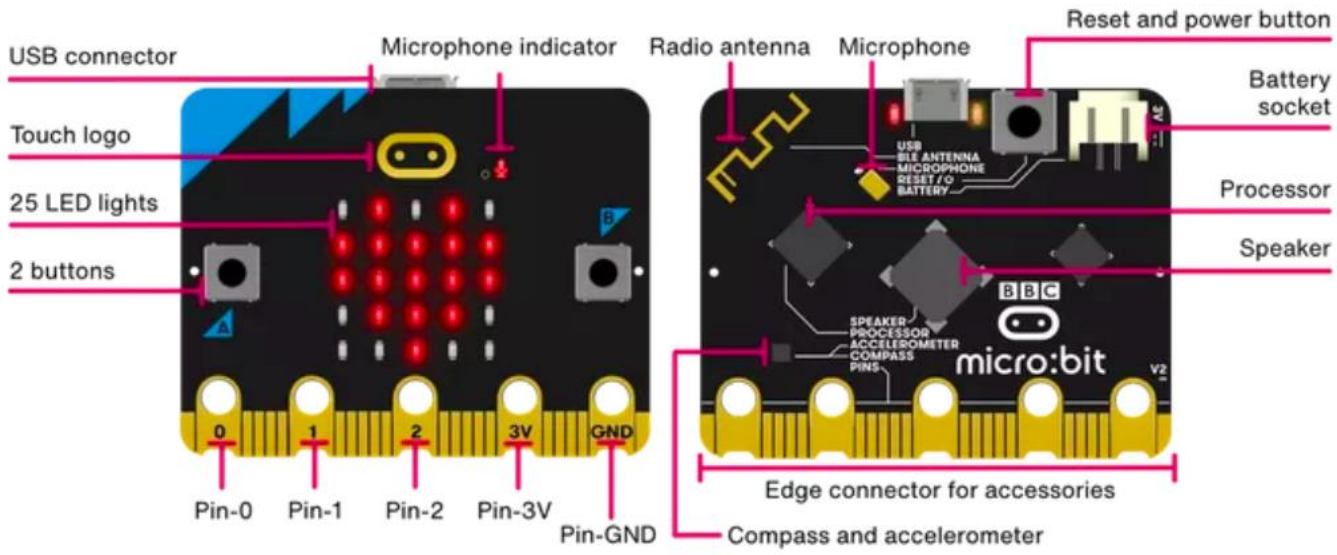
TP MICROBIT



Connecter votre carte via la prise USB à l'ordinateur.
Dans Thonny, aller dans Outils/Options/Interpréteur.
Choisir MicroPython (BBC micro:bit) et laisser le port se sélectionner automatiquement ou trouver la carte BBC micro:bit dans la sélection.
La première fois que vous enregistrez votre script appuyez sur « enregistrer sous », puis sélectionner micro:bit et sauvegarder le fichier sous le nom **main.py**.
Les fois suivantes, vous arrêterez le script avec l'icône rouge **stop**, puis vous pourrez appuyer sur l'icône verte **Run** afin de l'enregistrer automatiquement.

TP1 : Prise en main de la carte

Avant de l'utiliser, n'oubliez pas d'importer la bibliothèque à chaque début de code !

```
from microbit import *
```

Code à tester :

```
from microbit import *

# Code dans 'while True:' la boucle se répète infiniment
while True:
    display.show(Image.HEART)
    sleep(1000)
    display.scroll('Hello')
```

I. L'écran

```
display.show(Image.HEART)
```

Vous avez accès à différent affichage préprogrammé en remplaçant HEART par :
HEART_SMALL, HAPPY, SMILE, SAD, CONFUSED, ANGRY, ASLEEP, SURPRISED, SILLY, FABULOUS, YES, NO, MEH, DUCK, GIRAFE, PACMAN, SKULL, GHOST.

```
display.scroll('Vive Python', delay=150, loop=True, wait=False)
```

Permet d'afficher un texte qui défilera. Le second paramètre permet de jouer sur la vitesse de défilement.
Le paramètre `loop`, permet une lecture en boucle ou non, et `wait` permet d'éxécuter ou non la ligne de code suivante pendant le défilement.

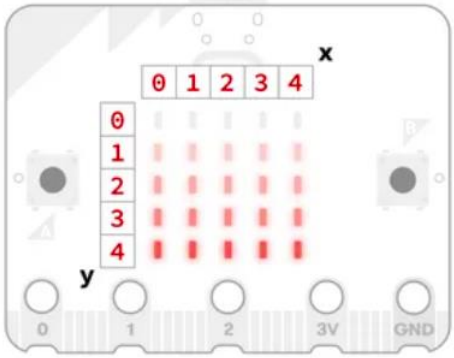
```
display.show(9876543210, delay=400, loop=True)
```

Ici, pas de défilement, mais un affichage caractère par caractère.

```
display.show(Image('00300:'
                   '03630:'
                   '36963:'
                   '03630:'
                   '00300'))
```
Vous avez accès à l'affichage de chacune des leds… À vous de créer votre motif…

```
display.clear()
```
Je pense que c'est assez clair ;)

```
display.set_pixel(0,0,9)
```
Allume la led de coordonnées (0 ; 0) avec une intensité de 9.




```
for y in range(5):
        for x in range(5):
            display.set_pixel(x,y,9)
            sleep(50)
```

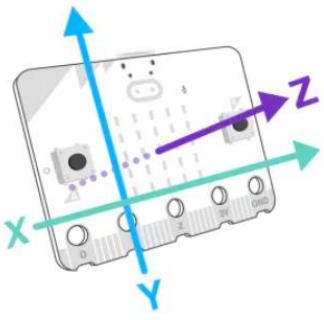
Réalisez maintenant un motif de votre imagination.

II. Les boutons

```
while True:
        if button_a.is_pressed() and button_b.is_pressed():
            display.scroll('A and B')
        elif button_a.is_pressed():
            display.scroll('A')
        elif button_b.is_pressed():
            display.scroll('B')
        sleep(100)
```
Avec ce script, le `get_presses()` permet de compter combien de fois le bouton A a été pressé :
```
display.scroll('Press A')
        sleep(3000)
        display.scroll(button_a.get_presses())
```

III. L'accéléromètre

```
while True:
    if accelerometer.was_gesture('shake'):
        display.show(Image.CONFUSED)
```



Un exemple de code:
```
from microbit import *
import random
while True:
    if accelerometer.was_gesture('shake'):
        display.show(random.randint(1, 6))
```

Vous pouvez remplacer le parametre `shake` par :
logo up, logo down, face up, face down, left, right.
Et le `was_gesture` par `is_gesture` pour un mouvement actuel.

Mesure de l'accélération selon les 3 axes *x*, *y* et *z* :
```
x_strength = accelerometer.get_x()
        display.scroll(x_strength)
```

IV. Capteur de luminosité (`display.read_light_level()`)

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

Exemple:

```
while True:
    if display.read_light_level() < 50:
        display.show(Image.HEART)
    else:
        display.clear()
```

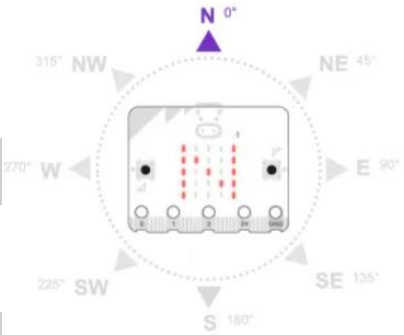
Le retour est un nombre compris entre 0 (noir) et 255 (très clair).

V. Capteur de température

```
display.scroll(temperature())
```

VI. Le compas

Pensez à la calibrer en premier ;)




```
compass.calibrate()
display.scroll(compass.heading())
```

VII. Les sons

Importer le module music:

```
import music
music.play(music. BA_DING)
```

Choix de sons:

BA_DING, BADDY, BIRTHDAY, BLUES, CHASE, DADADUM, ENTERTAINER, FUNERAL, FUNK, JUMP_UP, JUMP_DOWN, NYAN, ODE, POWER_UP, POWER_DOWN, PRELUDE, PUNCHLINE, PYTHON, RINGTONE, WAWAWAWAA, WEDDING.

Réglage du tempo de jeu (en bpm) :

```
music.set_tempo(bpm=120)
```

Jouer une note (La=A, Si=B, DO=C, …)

```
music.play(['c', 'd', 'e', 'c'])
```

Par défaut, les notes sont jouées sur la 3ème octave. Vous pouvez modifier l'octave jouée et aussi la durée des notes qui par défaut est de 4.

```
for x in range(2):
    music.play(['C4:4', 'D4:4', 'E4', 'C4'])
for x in range(2):
    music.play(['E3:4', 'F3:4', 'G3:8'])
```

Jouer une fréquence (le La est en 440 Hz)

```
music.pitch(440)
sleep(1000)
music.stop()
```

Dire un texte:

```
import speech
speech.say('Hello, world. How are you?')
```

Remarque : il se débrouille mieux en anglais… La présence de virgule, point ou tiret permet d'articuler la phrase…

Son divers

```
audio.play(Sound. GIGGLE)
```

Paramètres :

GIGGLE, HAPPY, HELLO, MYSTERIOUS, SAD, SLIDE, SOARING, SPRING, TWINKLE, YAWN

Volume:

```
set_volume(128)
```

On et Off:

```
speaker.on()
speaker.off()
```

VIII. Le micro

La carte est équipée d'un micro au dos, ainsi que d'une led qui s'allume quand le micro fonctionne.

```
while True:
    if microphone.current_event() == SoundEvent.LOUD:
        display.show(Image.HAPPY)
    elif microphone.current_event() == SoundEvent.QUIET:
        display.show(Image.ASLEEP)
```

Paramétrage des niveaux sonores LOUD et QUIET :

```
microphone.set_threshold(SoundEvent.LOUD, 200)
microphone.set_threshold(SoundEvent.QUIET, 1)
```

Mesurer le niveau sonore (de 0 à 255) :

```
display.scroll(microphone.sound_level())
```

IX. Le logo TOUCH

Le logo TOUCH doré vous permet d'avoir accès à un 3ème bouton.

```
while True:
    if pin_logo.is_touched():
        display.show(Image.HAPPY)
```

X. Retour console

Vous pouvez avoir un retour console sur Thonny avec print.

```
while True:
    print(display.read_light_level())
    sleep(1000)
```

XI. Projet

Vous avez maintenant de quoi coder votre premier projet…
À vous de jouer !