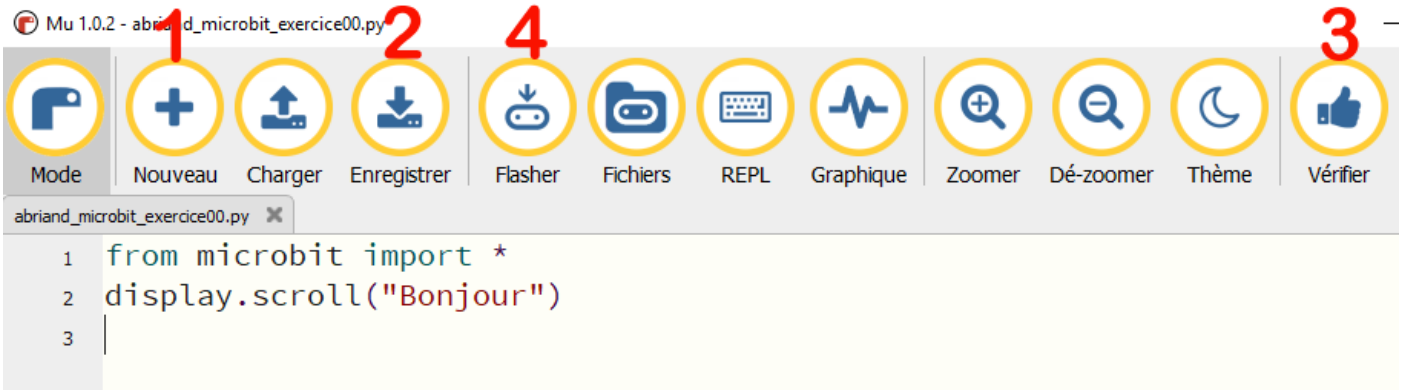
Activité Micro:bit

Activité 0

Ouvrez l'éditeur Mu.

1. Cliquez sur le bouton "Nouveau"
2. Cliquez sur "Enregistrer"; Choisir votre dossier personnel et enregistrez ce premier fichier Python sous le nom abriand_microbit_exercice00.py si vous vous appelez Aristide Briand.
3. Tapez ce qui suit dans l'éditeur Mu puis cliquez sur le bouton "Vérifier". S'il y a des erreurs, corrigez-les.
4. Quand il n'y a plus d'erreur, cliquez sur le bouton "Flasher"; le programme est alors transféré à la carte Microbit dont la led jaune clignote. Aussitôt après le transfert, la carte Microbit commence à exécuter le programme qui vient d'être transféré.

⚠ Cette procédure (commencer par nouveau puis enregistrer etc…) devra être respectée au début de chaque nouvel exercice. La raison est que Mu enregistre automatiquement toutes les modifications. Donc si on prend un ancien programme pour le modifier pour en faire un nouveau, l'ancien programme est automatiquement détruit.



La première ligne de ce programme importe la bibliothèque de fonctions micro:bit.
La deuxième ligne fait défiler un message sur la matrice de leds. Cela n'est exécuté qu'une fois. Pour exécuter à nouveau le programme, appuyez sur le bouton reset de la carte.

Le programme suivant utilise une boucle `while` pour faire défiler le message de manière répétée sur l'écran.
L'instruction `sleep` provoque la pause du micro:bit pendant un nombre défini de millisecondes.

```
from microbit import *
while True:
    display.scroll("Bonjour")
    sleep(5000)
```

Les boucles `While` se répètent tant que la condition spécifiée est vraie. Dans le cas précédent, nous avons dit que la condition est vraie. Cela crée une boucle infinie. Le code qui doit être répété est en retrait. On dit que ces lignes sont indentées.

Dans le code suivant, nous utilisons l'instruction `break` pour sortir de la boucle infinie lorsque le bouton A est enfoncé (un visage joyeux est alors affiché sur la matrice de leds).

```
from microbit import *
while True:
      if button_a.is_pressed():
            break
      display.scroll("Bonjour")
      sleep(5000)
display.show(Image.HAPPY)
```

Exercice 1

Vous pouvez activer et désactiver les leds de la matrice en utilisant leurs coordonnées. Le code suivant définit chacune des leds (pixels) de la rangée supérieure (y = 0) avec une luminosité maximum (9).

```
from microbit import *

for x in range(5):
    display.set_pixel(x,0,9)
    sleep(500)
```

La méthode `set_pixel` est définie comme suit :
`display.set_pixel (x, y, b)`
Il y a 3 arguments. Les deux premiers arguments spécifient la colonne (x) et la ligne (y). La dernière valeur est la luminosité du pixel et doit être un nombre compris entre 0 et 9.

Adaptez le programme de sorte que la rangée médiane de pixels soit activée au lieu de la rangée supérieure.

Exercice 2

Modifiez le code de l'exercice 1 pour qu'il éclaire les pixels de la colonne 0 au lieu de la ligne 0.

Exercice 3

Utilisez 2 boucles, une pour x et une pour y. Allumez chaque pixel, un à la fois, ligne par ligne.

Exercice 4

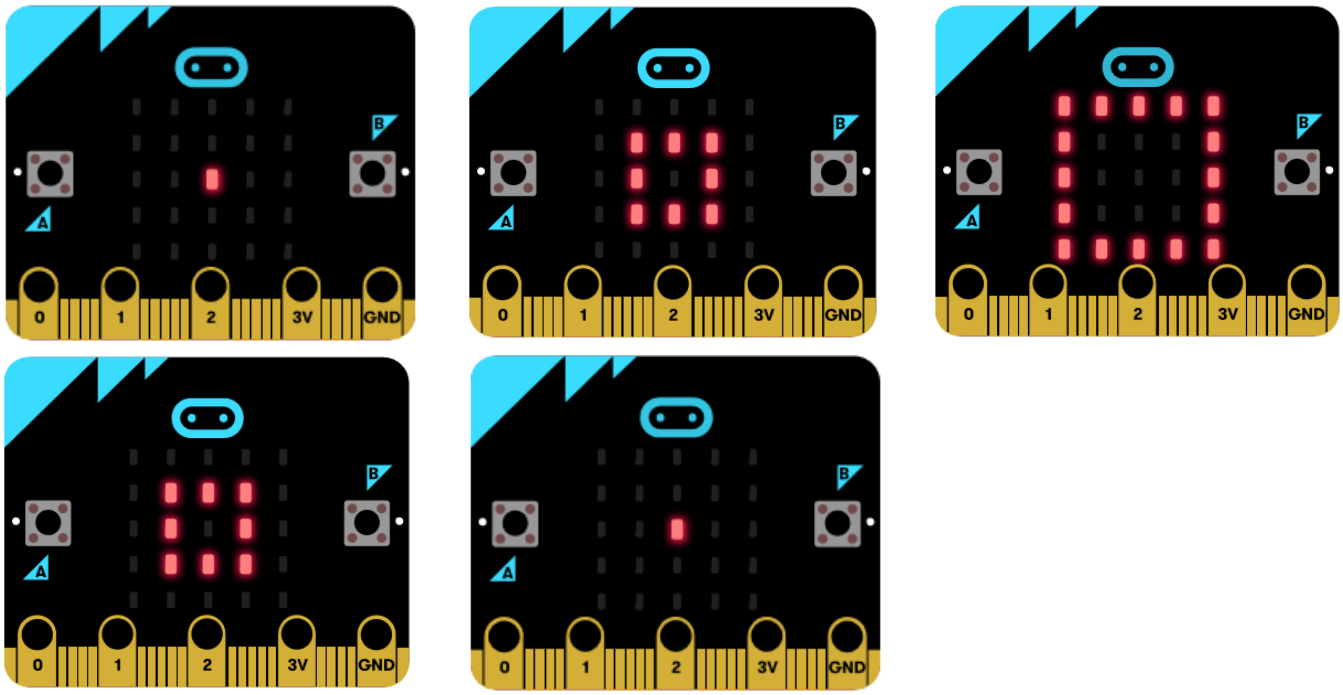
La même chose que l'exercice précédente, mais allumez les pixels colonne par colonne.

Exercice 5

En réglant la luminosité d'un pixel sur 0, vous le désactivez. Copiez votre code de l'exercice 4 et ajoutez du code à la suite afin de désactiver les pixels dans l'ordre inverse. Vous aurez besoin d'une boucle pour compter en arrière. Exemple : for x in range(4,-1,-1):

Exercice 6

Réalisez un programme pour afficher la séquence suivante :



Remarque : pour éteindre toute la matrice, il possible d'utiliser la méthode `display.clear()`

Exercice 7

La méthode `display.show ()` est utilisée pour afficher un caractère.
Si vous voulez afficher une valeur entière, vous devez la convertir en chaîne : `display.show (str (x))` Utilisez cette méthode pour que le micro:bit réalise un compteur de 0 à 9, avec un écart d'une seconde entre chaque chiffre affiché.
Utilisez une boucle for qui compte de 0 à 9.

Exercice 8

Réalisez un compte à rebours de 9 à 0.

Exercice 9

Etudiez le code suivant :

```
from microbit import *


num = 0
while True:
    if button_a.was_pressed():
        display.show(str(num))
        sleep(250)
        display.clear()
    sleep(50)
```

Adaptez le programme de manière à ce que le chiffre affiché s'incrémente de 1 chaque fois que vous appuyez sur le bouton A. S'il est égal (==) à 10, réinitialiser à 0.

Exercice 10

Ecrivez un programme qui utilise les deux boutons. Faites en sorte que l'utilisateur puisse augmenter le nombre en appuyant sur le bouton B et le diminuer en appuyant sur le bouton A. Assurez-vous que le nombre ne soit jamais inférieur à 0 et jamais supérieur à 9.

Exercice 11

Ecrivez un programme qui utilise les deux boutons. Faites en sorte que l'utilisateur puisse augmenter le nombre en appuyant sur le bouton B et le diminuer en appuyant sur le bouton A. Assurez-vous que le nombre ne soit jamais inférieur à 0 et jamais supérieur à 9.

Exercice 12

Ecrivez un programme qui utilise les données de l'accéléromètre. Faites en sorte que l'utilisateur puisse déplacer le pixel allumé vers la gauche en inclinant vers la gauche la carte et vers la droite en inclinant vers la droite. Utilisez l'instruction `dx = accelerometer.get_x ()`.

- Si dx est supérieur à 300, ajoutez 1 à x.
- Si dx est inférieur à -300, soustrayez 1 de x.