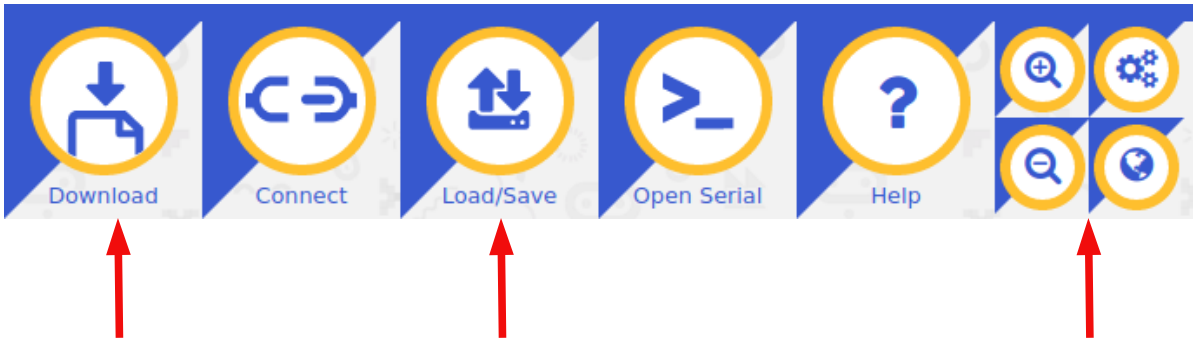
Partie 1 : Comment fonctionne la carte BBC micro:bit. ?

La carte BBC micro:bit a été développée au Royaume-Uni afin de permettre l'apprentissage du codage aux jeunes écoliers. La carte est équipée d'un microcontroleur ARM Cortex-M0, d'un accéléromètre 3D, une boussole numériques 3D, d'une matrice de LEDs 5x5, d'une connexion USB et Bluetooth. Son design prévoit d'y adjoindre des cartes d'extensions pour augmenter ses possibilités d'interaction avec le réel.

Manipulation 1

Cette première manipulation a pour objectif de s'approprier la chaîne de développement afin d'écrire son propre code à l'intérieur de la carte. Pour l'activité le langage Python sera utilisé. Pour cela, nous utiliserons l'éditeur de code en ligne. Brancher la carte micro:bit sur l'ordinateur à l'aide du câble USB.

Dans le navigateur internet, aller l'adresse https://python.microbit.org pour ouvrir l'éditeur en ligne.



Transférer le code dans la carte micro:bit

Sauvegarde ou Charge le code sur le disque dur

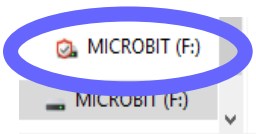
Vider la zone de d'édition puis taper le code suivant :

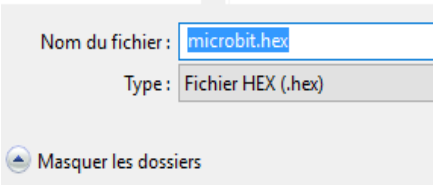
```
from microbit import *

display.show('Hello SNT ', delay=300, loop=True)
```

Transférer le code dans la carte (on dit parfois aussi 'flasher') en cliquant sur Download

L'éditeur vous propose de sauvegarder le fichier **microbit.hex**
Pour le transférer directement dans la mémoire programme (ROM) de la carte choisir comme emplacement le **lecteur MICROBIT** qui doit normalement apparaître dans l'arborescence si la connexion s'est bien faite. **Inutile de changer le nom du fichier.**





La LED orange de la carte clignote pendant le transfert du programme. L'exécution démarre immédiatement après.

1. Observer la matrice de LED pour constater le résultat du code exécuté.
2. Modifier le code précédent afin de :

- personnaliser le message
- régler différemment la valeur du delay,
- remplaçer le True par un False.

Suivre à chaque fois, la procédure précédente pour observer le changement sur la carte micro:bit.

Manipulation 2 : Affichage des symboles prédéfinies

Des symboles prédéfinies peuvent être affichés sur la matrice de LEDs. Le nom des symboles sont
`Image.HEART, Image.HEART_SMALL, Image.HAPPY, Image.SMILE, Image.SAD, Image.CONFUSED, Image.ANGRY, Image.ASLEEP, Image.SURPRISED, Image.SILLY, Image.FABULOUS, Image.MEH, Image.YES, Image.NO, Image.CLOCK12, Image.CLOCK11, Image.CLOCK10, Image.CLOCK9, Image.CLOCK8, Image.CLOCK7, Image.CLOCK6, Image.CLOCK5, Image.CLOCK4, Image.CLOCK3, Image.CLOCK2, Image.CLOCK1, Image.ARROW_N, Image.ARROW_NE, Image.ARROW_E, Image.ARROW_SE, Image.ARROW_S, Image.ARROW_SW, Image.ARROW_W, Image.ARROW_NW, Image.TRIANGLE, Image.TRIANGLE_LEFT, Image.CHESSBOARD, Image.DIAMOND, Image.DIAMOND_SMALL, Image.SQUARE, Image.SQUARE_SMALL, Image.RABBIT, Image.COW, Image.MUSIC_CROTCHET, Image.MUSIC_QUAVER, Image.MUSIC_QUAVERS, Image.PITCHFORK, Image.XMAS, Image.PACMAN, Image.TARGET, Image.TSHIRT, Image.ROLLERSKATE, Image.DUCK, Image.HOUSE, Image.TORTOISE, Image.BUTTERFLY, Image.STICKFIGURE.`

La ligne de code permettant d'afficher un symbole est :
```
display.show(Image.SMILE) # pour afficher le smiley par exemple
```

Un exemple de code pour afficher 3 symboles pourrait être :
```
from microbit import *

while True :
     display.show(Image.HEART) ;
     sleep(500)
     display.show(Image.HEART_SMALL) ;
     sleep(500)
     display.show(Image.BUTTERFLY)
     sleep(500)
```

Taper le code permettant d'afficher plusieurs symboles, les uns après les autres. Un temps de repos **sleep(500)** devra être inséré entre chaque affichage afin d'observer chacune des figures.

3. Réaliser un compte à rebours à partir de 10

L'algorithme (succession de procédures) est le suivant :
```
    Debut compte_a_rebour :
    Faire infiniment :
         Pour i de 10 à 0 par pas de -1 :
              afficher(i, pendant 1s)
    Fin compte_a_rebours
```
Remarques
- La fonction d'affichage possède des paramètres qui permettent de régler le temps d'affichage, et l'extinction des LEDs entre 2 affichages, la syntaxe est la suivante :
```
display.show(str(i) , delay = 1000, loop= False, clear = True)
```

- Une boucle **POUR** décomptant de 5 en 5 en partant de 100 en python s'écrit :
```
for i in range(100, 0, -5) :
     print(i)
```

Manipulation 3 : Utilisation des boutons

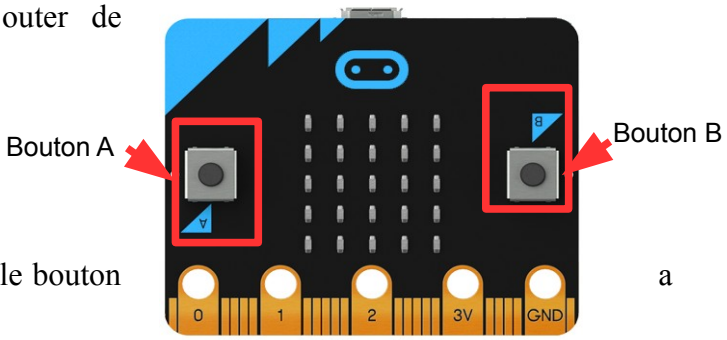
2 boutons sont présents sur la carte afin de rajouter de l'interactivité dans les programmes.

Les fonctions associées sont :

button_a.is_pressed() renvoie True si le bouton est effectivement appuyé

button_a.was_pressed() renvoie True si le bouton a été appuyé

button_a.get_pressed() renvoie le nombre de fois que le bouton a été appuyé.



Les mêmes fonctions existent pour le bouton B en remplaçant la lettre a dans le nom par b.

Voici un exemple de code utilisant la première fonction :

```
from microbit import *

while True :
    if button_a.is_pressed() :
        display.show(Image.HAPPY)
    else :
        display.show(Image.SAD)
```

4. Décrire ce que réalise ce morceau de code

5. Réaliser un dé électronique à 6 faces. L'appui sur le bouton A provoque l'affichage d'un nombre choisi au hasard entre 1 et 6 par la microbit.

Bonus : simuler le roulage du dé, en faisant une petite animation avant d'afficher le nombre.

Compter le nombre d'appuis

Il est parfois intéressant de compter le nombre d'appuis qu'il y a eu sur le bouton pendant une période donnée. La fonction **button_a.get_presses()** permet de le réaliser.

Taper le code suivant et le flasher dans la carte

```
from microbit import
    while True :
        sleep(2000)
        count = button_a.get_presses()
        display.scroll(str(count))
```

6. Décrire ce que réalise ce morceau de code.

Remarque

la valeur renvoyée par la fonction button_a.get_presses() est un nombre entier. Il est nécessaire de la convertir en chaîne de caractères (string) avant de l'afficher. C'est ce que fait le code **str(count)** sur la dernière ligne.

Mini-projet Jeu de réflexe Consonne ou Voyelle

L'objectif est de réaliser un jeu de réflexe. La micro :bit affiche une lettre qui est soit une consonne, soit une voyelle. Le joueur doit appuyer sur le bouton A si c'est une consonne, sur le bouton B si c'est une voyelle. La réponse doit être choisie en moins d'une seconde. Si le mauvais bouton est appuyé, la microbit affiche une image négative (à choisir) et le jeu est fini affichant le meilleur score du joueur. Si le bon bouton a été pressé, la microbit affiche une image positive et rajoute 1 point au score du joueur, et présente alors un nouveau tour de jeu.

Une solution à compléter est proposée en annexe, après l'avoir lue et analysée (avec le professeur) , rajouter les lignes de code manquantes et tester votre solution.

Annexe solution mini-projet à compléter

```
from microbit import *
import random

# les voyelles sont sur les positions paires,
# les consonnes sur les positions impaires
lettres = ['a', 'b', 'e', 'm', 'i', 'k', 'o', 'f', 'u', 'z', 'y']

while True:
    # On initialise le score
    score = 0
    # Decomptage avant de commencer
    for i in range(5, 0, -1):
        display.show(str(i), delay=500, clear=True)
    while True:
        display.clear()
        # tirer une lettres au hasard
                                            # 11 lettres dans la liste
        # on affiche la valeur choisie
        display.show(lettres[tirage])
        # on lance la temporisation de 1s

        # on teste quel bouton a été pressé
        if button_a.was_pressed() and tirage % 2 == 1:  # bonne réponse consonne
                                                    # Afficher image bonne réponse
                                                # augmenter le score de 1
        elif button_b.was_pressed() and tirage % 2 == 0:  # bonne réponse voyelle
                                                        # Afficher image bonne réponse
                                        # Afficher image bonne réponse
        else:   #mauvaise réponse
            while True:
                                                            # mauvaise réponse afficher image mauvais
                                                    # affichage score et FIN
```