Mémo Micro:bit (et python)

Méthode (Essentiel)

Importer toutes les fonctions — `from microbit import *`

Faire une pause (en ms) — `sleep(...)`

Méthode (Écrire dans un terminal)

Écrire du **texte** (string) — `print()`

ex. ✎ `print('Bouton A : 5 fois')`

Aligner avec des **tabulations** — `'\t '`

ex. ✎ `print('Bouton \tA \t :\t 5 fois')`

Sauter des lignes — `'\n '`

ex. ✎ `print('Bouton A :\n 5 fois\n ')`

Convertir en chaînes (string) — `str()`

ex. ✎ `print('Bouton A :' + str(5) +' fois')`

Remarque

Activer la **communication série**

Méthode (LED Micro:bit)

Afficher texte/image — `display.scroll()`

ex. ✎ `display.scroll("G")`

ex. ✎ `display.scroll(Image.HEART)`

Faire **défiler** texte/image — `display.show()`

ex. ✎ `display.show("Gagne !")`

ex. ✎ `display.show(Image.HAPPY)`

Créer des **animations** — `delay=...`

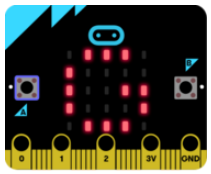
ex. ✎ `display.show(Image.ALL_CLOCKS, delay=200)`

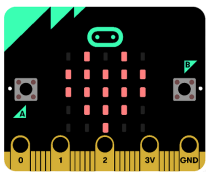
Modifier **pixel** — `display.set_pixel(x,y,lum)`

(lum = luminosité [0;9])

ex. ✎ `display.set_pixel(0,4,9)`

Effacer l'écran — `display.clear()`





Remarque

Pour créer/utiliser des **animations**,
⟹ **tableaux d'images**.

ex. ✎ `Image.ALL_CLOCKS` (lignes)

ex. ✎ `Image.ALL_ARROWS` (flèches)

```
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.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
Image.GHOST
Image.SWORD
Image.GIRAFFE
Image.SKULL
Image.UMBRELLA
Image.SNAKE
```

Méthode (Les boutons)

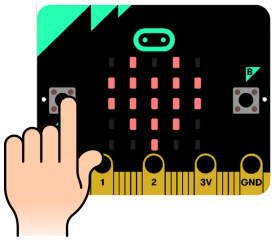
Accéder aux **boutons** A ou B
```
button_a. ...
button_b. ...
```

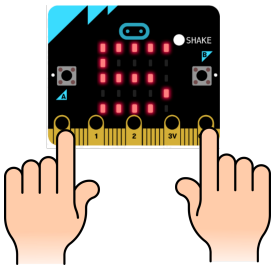
Nombre d'appuis (**cumul**)
```
button_a.get_presses()
button_b.get_presses()
```
ex. ✎ `display.scroll(str(button_a.get_presses()))`

Tester si bouton **est** pressé
```
button_a.is_pressed()
button_b.is_pressed()
```
ex. ✎ `if (button_b.is_pressed()):print("Bouton B")`

Tester si bouton **a été** pressé
(appeler 2 × la fonction)
```
button_a.was_pressed()
button_b.was_pressed()
```
ex. ✎ `if (button_b.was_pressed()):display.show(Image.HAPPY)`

Utiliser les **broches**
```
pin0.is_touched()
pin1.is_touched()
pin2.is_touched()
```





Remarque

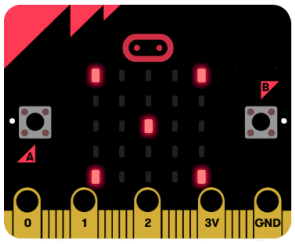
Détecter un **évènement**, avec une **boucle infinie**.
ex. ✎ `while True:`
Pour **sortir d'une boucle** lorsque l'évènement est détecté :
ex. ✎ `break`

Méthode (Aléa)

Activer le module **hasard**
```
import random
```

Nombre **aléatoire** sur [0 ; 1]
```
random.random()
```
ex. ✎ `display.show(round(random.random(),3))`

Nombre **entier** aléatoire sur [a ; b]
```
random.randint(a,b)
```
ex. ✎ `display.show(random.randint(1,6))`



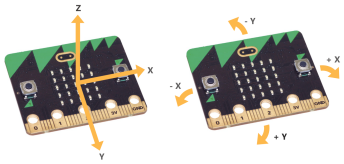
Méthode (Mouvement)

1 coord. vect. accélération
```
accelerometer.get_x()
accelerometer.get_y()
accelerometer.get_z()
```
ex. ✎ `display.show(accelerometer.get_x())`

3 coord. vect. (tuple)
```
accelerometer.get_values()
```
ex. ✎ `print(accelerometer.get_values())`

Historique **gestes** (tuple)
(appeler 2 × la fonction)
```
accelerometer.get_gestures()
```
ex. ✎ `print(accelerometer.get_gestures())`

Tester geste **actuel**
```
accelerometer.is_gesture('...')
```
Tester geste **passé**
(appeler 2 × la fonction)
```
accelerometer.was_gesture('...')
```



Gestes reconnus

'up'	'freefall'
'down'	'3g'
'left'	'6g'
'right'	'8g'
'face up'	'shake'
'face down'	

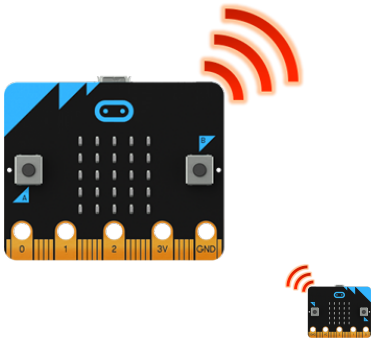
Méthode (Radio)

Activer le **module** radio
```
import radio
```
Mettre en route la radio
```
radio.on()
```
Envoyer un message
```
radio.send('...')
```
ex. ✎ `radio.send('pile')`

Message reçu (string) *(= `None` si vide)*
```
radio.receive()
```
ex. ✎ `if (radio.receive()=='pile'): print('Pile')`

Canal perso (⩽ 100)
```
radio.config(channel=..)
```
ex. ✎ `radio.config(channel=21)`

Puissance perso (⩽ 7)
```
radio.config(power=..)
```
ex. ✎ `radio.config(power=7)`

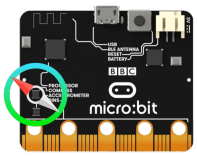


Méthode (Boussole)

1 coord. vect. champ magnétique
```
compass.get_x()
compass.get_y()
compass.get_z()
```
ex. ✎ `display.show(compass.get_x())`

Angle (en °) avec le nord
```
compass.heading()
```
ex. ✎ `aiguille=((15-compass.heading())//30)%12`

Calibrer la boussole
```
compass.calibrate()
```



Remarque

Toujours calibrer la boussole.

Mesures/valeurs **fluctuantes**
(présence de métal, d'aimants, etc.)

À propos de cette publication

Pourquoi les objets connectés ?

Alors que dans certaines disciplines le temps commence à manquer pour traiter l'ensemble du programme, certains évoquent déjà l'idée d'en faire plus !

En effet, les enseignants utilisent déjà les outils numériques. Par exemple, dans les classes de mathématiques, l'utilité du tableur et de GeoGebra n'est plus à démontrer. Jusqu'à l'introduction de l'algorithmique, ces deux logiciels efficaces et maîtrisés par les enseignants étaient amplement suffisants. Est-ce donc juste un effet de mode de faire cours avec les robots (Thymio, Mbot), les objets programmables et connectés (Arduino, `Micro:bit`, STM education, Raspberry Pi) ou est-ce une nouvelle façon d'aborder notre enseignement ? Ces nouvelles possibilités technologiques, forcément chronophages, nous permettront-elles de traiter un contenu disciplinaire exigeant dans un cadre institutionnel contraignant ?

Nous n'avons bien sûr pas toutes les réponses à ces questions mais nous pensons que lorsqu'il est accompagné de certains de ces outils, notre enseignement a beaucoup à y gagner.

L'introduction de l'algorithmique en lycée professionnel nous interroge. Longtemps il nous a semblé impensable et inenvisageable d'avoir à enseigner un langage de programmation comme Python auprès d'un public d'élèves globalement en difficulté avec les mathématiques. Fort de ce constat, nous avons cherché les moyens de lier les mathématiques à la logique et au raisonnement algorithmique. C'est pourquoi nous avons exploré les potentialités des objets connectés.

Notre postulat est double. Nous pensons que :

- grâce à des situations réelles et concrètes, les objets connectés facilitent la mise en activité de tous les élèves ;
- grâce à des activités simples mais évolutives centrées autour de réalisations matérielles, la dimension affective du travail est valorisée. Soyons fous et espérons que l'élève tisse une histoire personnelle avec l'activité, qu'il soit fier du travail accompli et qu'il prenne également du plaisir à expliquer et à montrer ses réalisations.

En devenant de plus en plus simples, accessibles et facilement utilisables, les objets connectés permettent d'aborder des contenus disciplinaires et de développer des compétences transversales essentielles pour l'élève.

En travaillant à partir des objets connectés, la situation de départ est plus concrète et l'objectif à atteindre suffisamment clair pour l'élève. Plus ou moins guidé selon son niveau d'expertise technique, il est alors libre dans sa démarche. Avec des interfaces de programmation accompagnées parfois de simulateurs, la démarche par essais et erreurs a ici toute sa place. Par ailleurs, l'élève devra clarifier sa pensée avant de verbaliser ses idées en langage naturel. Il pourra ainsi proposer et élaborer un modèle acceptable par la machine pour enfin traduire son algorithme en se pliant à la rigueur du langage de programmation.

Effectuant régulièrement des va-et-vient entre abstraction et réalité, cherchant à valider son algorithme à partir d'un visuel ou d'une exploitation des résultats, l'élève entre progressivement dans la modélisation.

Les scénarios proposés dans cette brochure permettent tout cela : une approche des mathématiques et des sciences qui laisse la place à l'expérimentation : manipulation, programmation et auto-validation.

Qui sommes-nous ?

Nous sommes des enseignants de maths/sciences regroupés au sein d'un groupe de recherche de l'IREM de Marseille.

groupe Lycée Professionnel innovation, expérimentation et formation de l'IREM d'Aix-Marseille

Notre groupe, Innovation, Expérimentation et Formation en Lycée Professionnel (InEFLP) consacre une partie de son travail à l'enseignement de l'algorithmique en classes de lycée professionnel. Dans le cadre de cette recherche, nous explorons les objets connectés tels que Arduino, `Micro:bit`, `STM32 Éducation` ou mbot.

Liens utiles

Page du groupe InEFLP
`http://url.univ-irem.fr/ineflp`

IREM de Marseille Site académique de l'IREM de Marseille
`http://url.univ-irem.fr/mars`

Portail des IREM Site national des IREM
`http://www.univ-irem.fr/`

Formation à l'algorithmique LP et SEGPA Padlet de utilisé lors de nos formations académiques
`http://url.univ-irem.fr/stage-algo`

Collecte de ressources pour `Micro:bit` Padlet sur `Micro:bit` utilisé en formation
`http://url.univ-irem.fr/algo2017-microbit`

Brochure sur `Micro:bit` Publication de la C2i TICE pour une prise en main de `Micro:bit`
`http://url.univ-irem.fr/c2it-mb-t1-pdf`

Description `Micro:bit` Fiche sommaire de description de `Micro:bit`
`http://url.univ-irem.fr/ineflp-microbit`

Site IREM dédié à `Micro:bit` Site de ressources sur `Micro:bit` du groupe
`http://url.univ-irem.fr/o`

Un extrait de la brochure

Les objets connectés pour enseigner l'algorithmique en lycée professionnel

< version du 31 janvier 2020 >