
Ressources pour la carte Micro:bit

Version 1

IREM Marseille

mai 09, 2018

Projets par blocs

1	Projets à réaliser	3
1.1	Boîte fermée	3
1.2	Pierrot et Simon	5
1.3	Températures	7
1.4	Coffre fort	12
1.5	Planche de Galton	14
2	Index et page de recherche	17
Index		19

par le groupe InEFLP de l'IREM de Marseille

Contenu du site

Vous trouverez dans ce document des ressources permettant de se former à Micro :bit.

Qui sommes-nous ?

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

Notre groupe, *Innovation, Expérimentation et Formation en Lycée Professionnel* (InEFLP) a une partie de son travail consacrée l'enseignement de l'algorithme en classes de lycée professionnel. Dans le cadre de cette recherche, nous explorons les objets connectés tels que Arduino ou Microbit.

Site du groupe InEFLP.



Table des matières du document

CHAPITRE 1

Projets à réaliser

1.1 Boîte fermée

1.1.1 Description

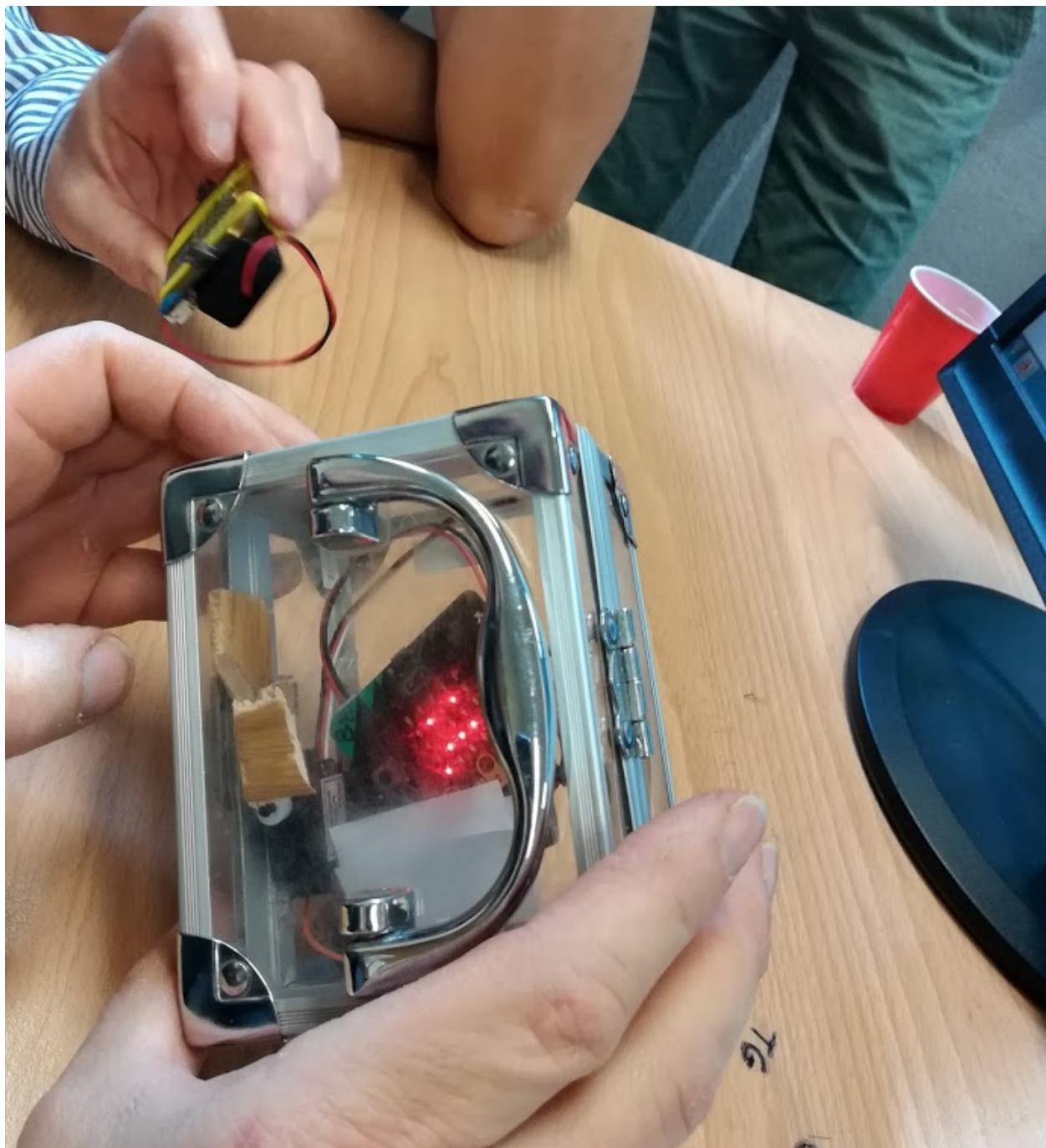
À faire : capture d'écran / gif animée

Exemple(s) d'utilisation

Escape game

Nous avons utilisé le projet *Boîte fermée* pour un escape game proposé en stage.

- diaporama d'accueil : <http://url.univ-irem.fr/boite>
- page de formation : <http://url.univ-irem.fr/algo1718-boite>



1.1.2 Réalisation

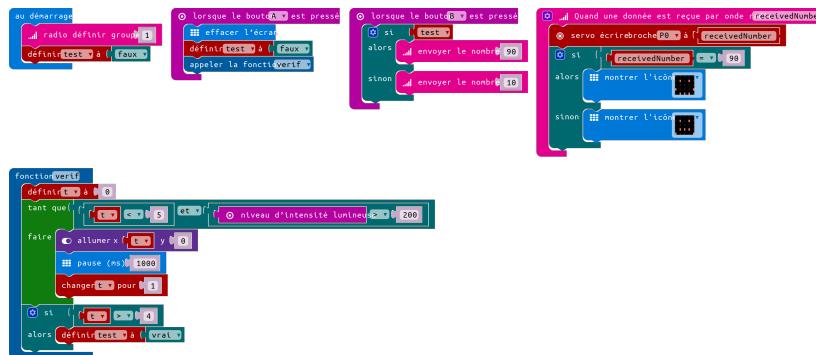
Fabriquer

Nous détaillons ici comment fabriquer et assembler le matériel nécessaire à la réalisation du projet *Boîte fermée*.

À faire : tout faire.

Coder

Nous détaillons ici le code nécessaire à la réalisation du projet *Boîte fermée*.



1.2 Pierrot et Simon

1.2.1 Description

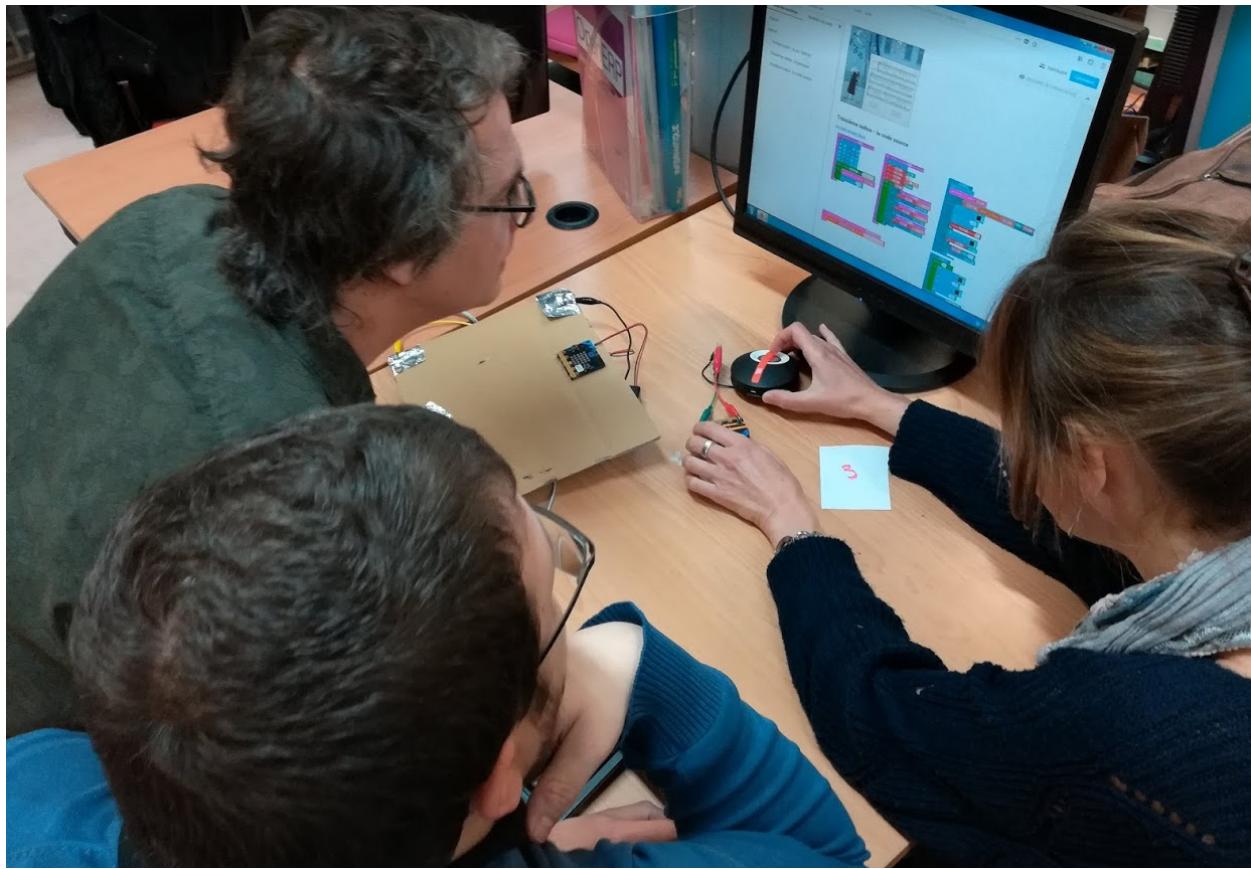
À faire : capture d'écran / gif animée

Exemple(s) d'utilisation

Escape game

Nous avons utilisé le projet *Pierrot et Simon* pour un escape game proposé en stage.

- diaporama d'accueil : <http://url.univ-irem.fr/pierrot>
- page de formation : <http://url.univ-irem.fr/algo1718-pierrot>



1.2.2 Réalisation

Fabriquer

Nous détaillons ici comment fabriquer et assembler le matériel nécessaire à la réalisation du projet *Pierrot et Simon*.

À faire : tout faire.

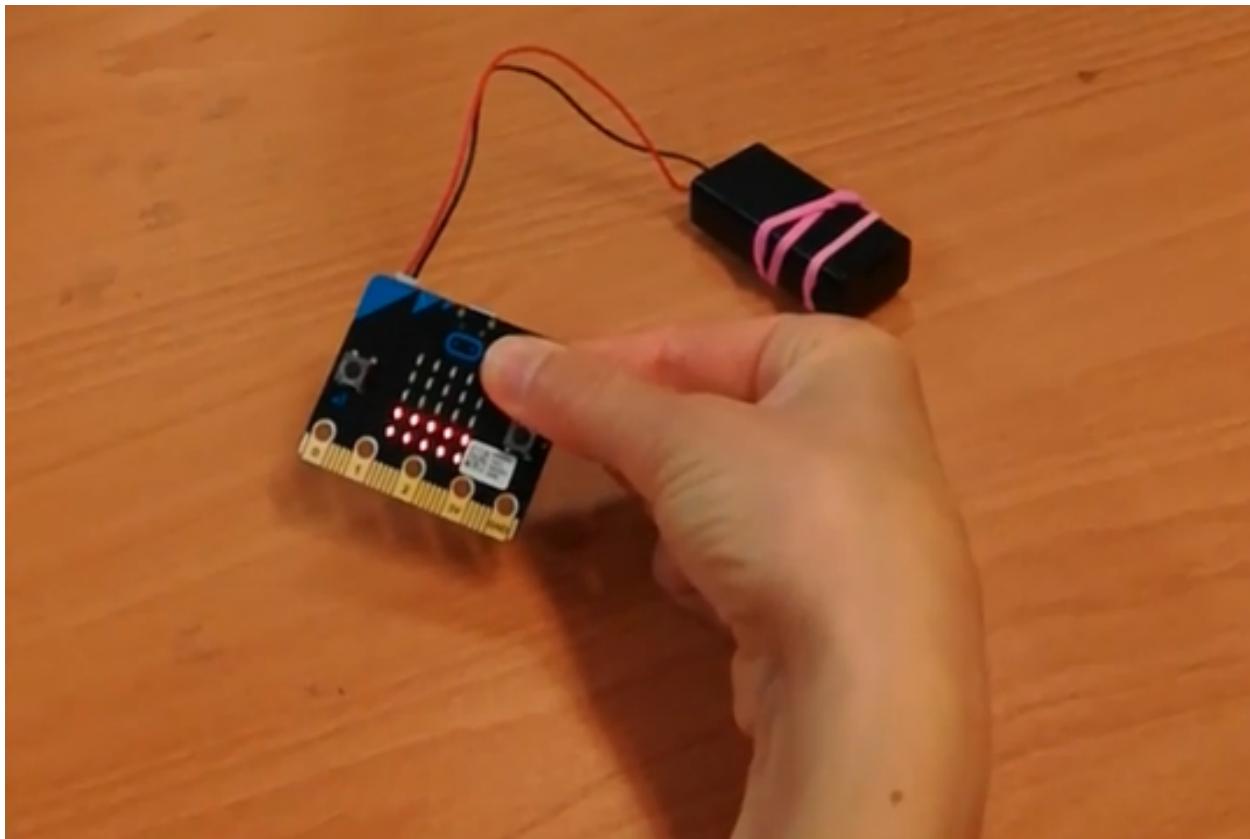
Coder

Nous détaillons ici le code nécessaire à la réalisation du projet *Pierrot et Simon*.

À faire : tout à faire !

1.3 Températures

1.3.1 Description



Le code téléchargé dans le Micro :bit permet d'afficher un **code secret**. Pour cela, il faut que la température augmente et dépasse les 34°C. L'écran affiche une jauge qui se remplit.

Une fois la température atteinte, le code secret s'affiche après une petite animation.

Exemple(s) d'utilisation

Escape game

Nous avons utilisé le projet *Températures* pour un escape game proposé en stage.

- diaporama d'accueil : <http://url.univ-irem.fr/temp>
- page de formation : <http://url.univ-irem.fr/algo1718-temp>

Le but était d'afficher un code secret permettant d'accéder à l'énigme suivante.



Les stagiaires devaient donc prendre connaissance du diaporama d'accueil qui les renvoiaient vers des indices et le code source téléchargé dans le micro:bit.

À l'aide de l'étude du code source, des indices et la bidouille, les stagiaires devaient effectuer les manipulations nécessaires à l'affichage du code source.

De très bons moments pour tous !

1.3.2 Réalisation

Fabriquer

Pour le projet *Températures*, il n'y a besoin de presque rien :

- carte micro:bit;
- alimentation électrique (pile par exemple).

Coder

Le code nécessaire à la réalisation du projet *Températures* a été écrit en micropython. Vous trouverez ci-dessous :

- [*Le code, étape par étape*](#)
- [*Le code final*](#)

Le code, étape par étape

1. Incluons la bibliothèque Micro:bit

```
from microbit import *
```

2. Créons les images qui nous servirons à animer l'écran. La luminosité d'une diode varie de 0 (éteinte) à 9 (maximale).

Lorsque la température augmente, l'affichage passe progressivement de image1 à image 5.

```
image1 = Image(
    '00000:'
    '00000:'
    '00000:'
    '00000:'
    '99999')
image2 = Image(
    '00000:'
    '00000:'
    '00000:'
    '99999:'
    '77777')
image3 = Image(
    '00000:'
    '00000:'
    '99999:'
    '77777:'
    '77777')
image4 = Image(
    '00000:'
    '99999:'
    '77777:'
    '77777:'
    '77777')
image5 = Image(
    '99999:'
    '77777:'
    '77777:'
    '77777:'
    '77777')
```

3. Il y aura deux états dans le jeu :

- La variable victoire est vrai et l'écran affiche le code secret.
- la variable victoire est fausse et l'écran affiche l'énigme.

Au début, la variable est donc fausse.

```
victoire = False
```

4. La phase de configuration est terminée. Passons maintenant à la boucle qui... tourne en boucle.

Tout ce qui suivra cette codee sera donc indenté (tabulation).

```
while True:
```

5. Nous envisageons trois actions possibles :

- (a) le jeu se réinitialise grâce au bouton A ;

```
if button_a.is_pressed():
    victoire = False
```

- b) le jeu est gagné et l'écran affiche le code final (après une petite animation) ;

```
if victoire:  
    # petite image joyeuse  
    display.show(Image.HAPPY)  
    sleep(500)  
    # code secret à afficher...  
    display.scroll("XXXXXX")
```

(c) le jeu est en cours et l'écran affiche les images.

```
if not victoire:
```

Lire la température

```
temp = temperature()
```

Plus la température augmente, plus les images affichées remplissent l'écran

```
if temp < 29:  
    display.clear()  
elif 29 <= temp < 30:  
    display.show(image1)  
elif 30 <= temp < 31:  
    display.show(image2)  
elif 31 <= temp < 32:  
    display.show(image3)  
elif 32 <= temp < 33:  
    display.show(image4)  
elif 33 <= temp < 34:  
    display.show(image5)
```

et enfin, si la température dépasse 34°C, alors là on passe en mode victoire vrai. On ajoute une petite animation pour montrer que la victoire approche.

```
elif 34 <= temp:  
    victoire = True  
    # petite animation  
    for i in range(2):  
        display.show(Image.SQUARE_SMALL)  
        sleep(100)  
        display.show(Image.SQUARE)  
        sleep(100)
```

Pour finir, une pause syndicale de 500ms.

```
sleep(500)
```

Le code final

```
1  # -*- coding: utf-8-*# Encoding cookie added by Mu Editor  
2  from microbit import *  
3  
4  # définir mes images perso  
5  # pour les lignes qui se colorent  
6  image1 = Image(  
7      '00000:'  
8      '00000:'
```

(suite sur la page suivante)

(suite de la page précédente)

```

9      '00000:'  
10     '00000:'  
11     '99999')  
12 image2 = Image(  
13     '00000:'  
14     '00000:'  
15     '00000:'  
16     '99999:'  
17     '77777')  
18 image3 = Image(  
19     '00000:'  
20     '00000:'  
21     '99999:'  
22     '77777:'  
23     '77777')  
24 image4 = Image(  
25     '00000:'  
26     '99999:'  
27     '77777:'  
28     '77777:'  
29     '77777')  
30 image5 = Image(  
31     '99999:'  
32     '77777:'  
33     '77777:'  
34     '77777:'  
35     '77777')  
36  
37 # booléen pour savoir si l'énigme est réussie  
38 victoire = False  
39  
40 # à faire toujours et toujours...  
41 while True:  
42     # utiliser le bouton A pour réinitialiser  
43     if button_a.is_pressed():  
44         victoire = False  
45  
46     # si l'énigme est résolue  
47     if victoire:  
48         # petite image joyeuse  
49         display.show(Image.HAPPY)  
50         sleep(500)  
51         # code secret à afficher...  
52         display.scroll("XXXXXX")  
53  
54     # si l'énigme n'a pas été résolue  
55     if not victoire:  
56         # lire la température (en °C)  
57         temp = temperature()  
58         # affichage des images en fonction  
         # de temp  
59         if temp < 29:  
60             display.clear()  
61         elif 29 <= temp < 30:  
62             display.show(image1)  
63         elif 30 <= temp < 31:  
64             display.show(image2)

```

(suite sur la page suivante)

(suite de la page précédente)

```
66     elif 31 <= temp < 32:
67         display.show(image3)
68     elif 32 <= temp < 33:
69         display.show(image4)
70     elif 33 <= temp < 34:
71         display.show(image5)
72     # victoire !
73     elif 34 <= temp:
74         victoire = True
75         # petite animation
76         for i in range(2):
77             display.show(Image.SQUARE_SMALL)
78             sleep(100)
79             display.show(Image.SQUARE)
80             sleep(100)
81         sleep(500)
```

1.4 Coffre fort

1.4.1 Description

À faire : capture d'écran / gif animée

Exemple(s) d'utilisation

Escape game

Nous avons utilisé le projet *Coffre fort* pour un escape game proposé en stage.

- diaporama d'accueil : <http://url.univ-irem.fr/coffre>
- page de formation : <http://url.univ-irem.fr/algo1718-coffre>



1.4.2 Réalisation

Fabriquer

Nous détaillons ici comment fabriquer et assembler le matériel nécessaire à la réalisation du projet *Coffre fort*.

À faire : tout faire.

Coder

Nous détaillons ici le code nécessaire à la réalisation du projet *Coffre fort*.

À faire : tout à faire !

1.5 Planche de Galton

1.5.1 Description

À faire : capture d'écran / gif animée

Exemple(s) d'utilisation

1.5.2 Réalisation

Fabriquer

Nous détaillons ici comment fabriquer et assembler le matériel nécessaire à la réalisation du projet *Planche de Galton*.

À faire : tout faire.

Coder

Nous détaillons ici le code nécessaire à la réalisation du projet *Planche de Galton*.

```
from microbit import *
from random import random, seed

seed(300)                      # la graine de hasard ???
n = [0, 0, 0, 0, 0]             # le tableau contenant les compteurs

def aff(n, m):                  # la fonction affichant le graph
    q = n // 9                  # nombre de led éclairé totalement
    r = n % 9                   # portion de la dernière led éclairé
    for i in range(0, q):
        display.set_pixel(m, 4-i, 9)
    display.set_pixel(m, 4-q, r)

def chute(t):                   # fonction affichant la chute
    display.clear()
    y, x = 0, 0
    display.set_pixel(x, y, 9)
    sleep(t)
    while y < 4:
        display.clear()
        if round(random()):      # si arrondi de alea est vrai (différent de 0)
            y = y + 1            # on augmente y de 1
        else:
            x = x + 1
            y = y + 1
        display.set_pixel(x, y, 9)
        sleep(t)
    n[x] = n[x]+1               # incrementation du compteur de la position x
```

(suite sur la page suivante)

(suite de la page précédente)

```
display.set_pixel(x, y, 1)

while True:
    if button_a.is_pressed():
        chute(500)

    elif button_b.get_presses():
        n = [0, 0, 0, 0, 0]
        for k in range(80):
            chute(round(500 / (1.05**k)))
            for j in range(5):
                aff(n[j], j)
                sleep(200)
        print(n)
```


CHAPITRE 2

Index et page de recherche

- genindex
 - search
-

Index

A

accéléromètre, 13
audio, 6

G

galton, 14

L

luminosité, 4

M

micropython, 6, 8, 13, 14

P

programmation par blocs, 4

S

servo, 4

T

température, 8