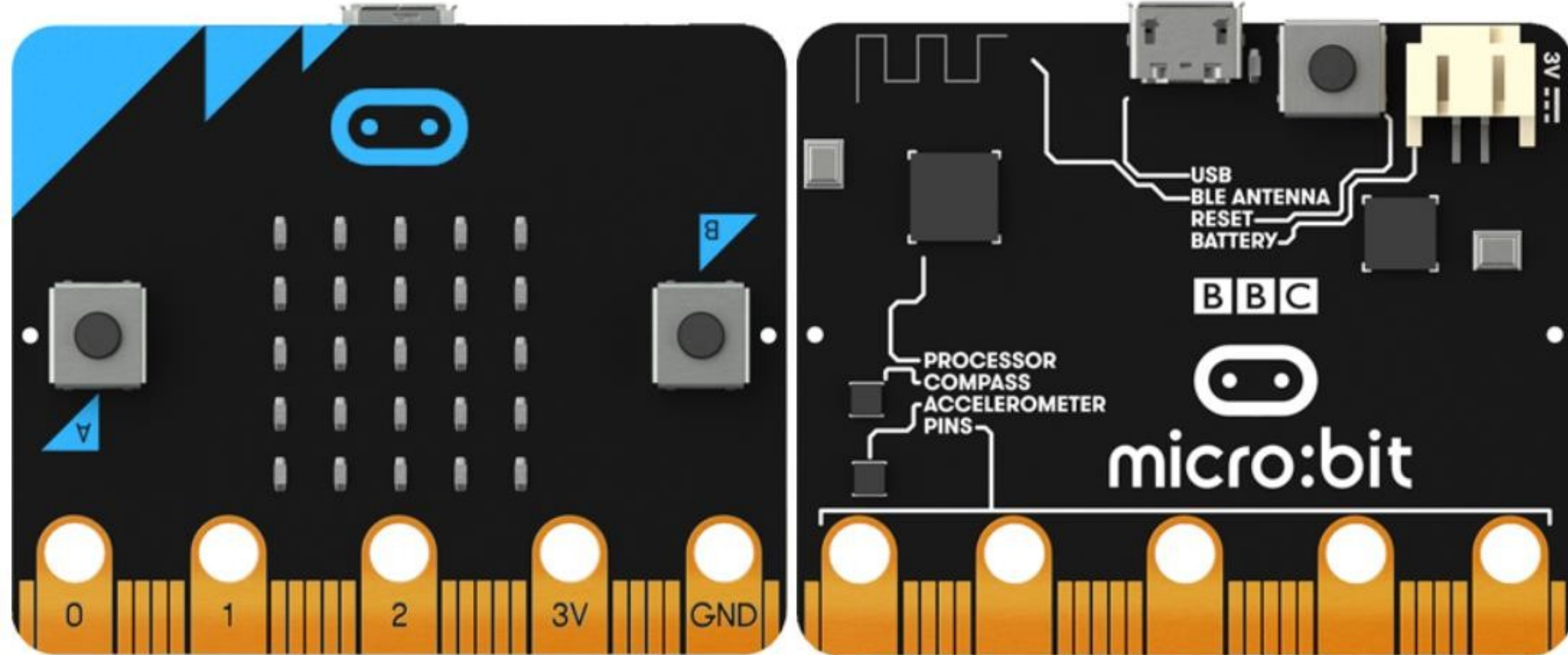


Présentation carte Micro:bit



La carte contient de nombreux capteurs :

- Deux boutons programmables, désignés « A » et « B » sur la carte
- afficheur digital carré de 25 LED (5 × 5) rouges programmables pouvant servir d'affichage, notamment pour des motifs animés, du texte alphanumérique déroulant

Micro:bit , les autres capteurs

Un bouton de réinitialisation, désigné « R » pour « Reset »

Un accéléromètre pour détecter les mouvements de l'appareil (détection des actions comme secouer, pencher et la chute libre)

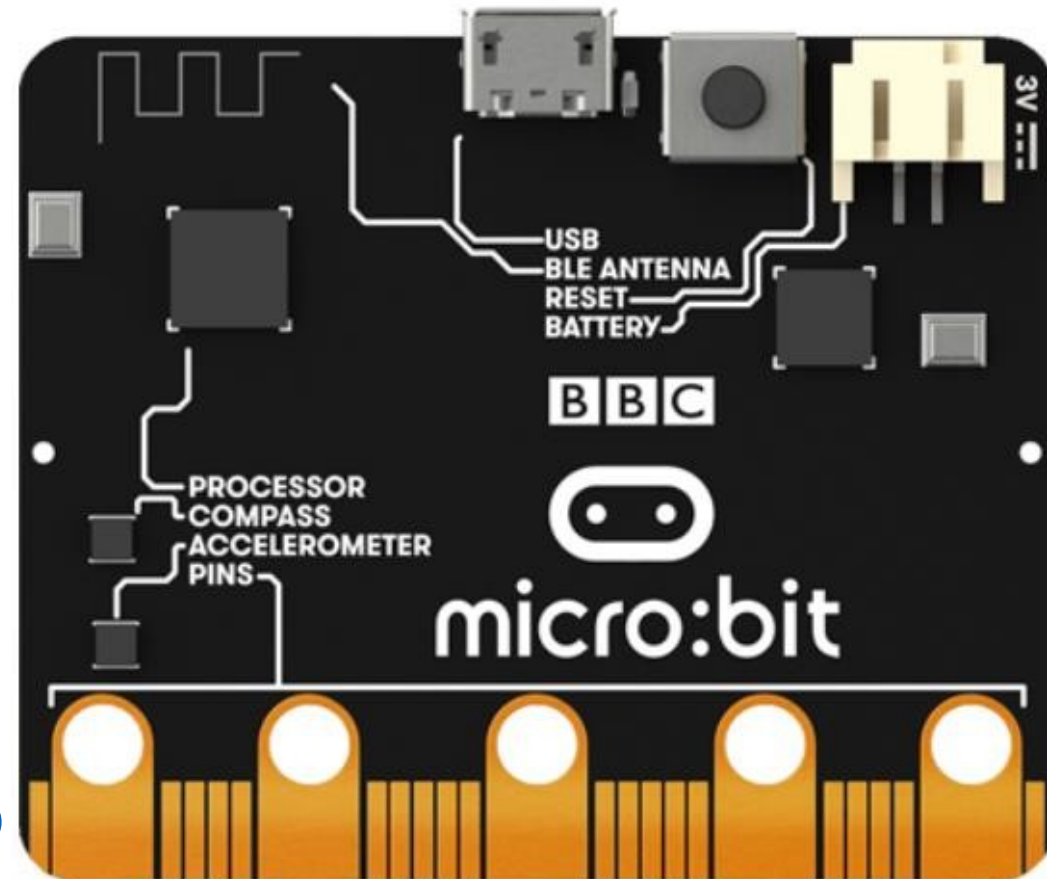
Une boussole magnétique 3D pour détecter les champs magnétiques : l'orientation de la carte.

Un capteur de température (sur le processeur)

Un capteur de luminosité lié aux LED

Une connectique bluetooth 4.0 basse énergie/2.4 GHz maître/esclave

Un connecteur micro-usb pour le programmer, connecteur alimentation 3,3V



Micro:bit approfondissement

Nous allons utiliser les autres capteurs
Micro:bit avec le site Vittascience en ligne

<https://fr.vittascience.com>

Ce site permet de travailler avec des blocs
ou en python sur Microbit et il dispose
d'un simulateur intégré

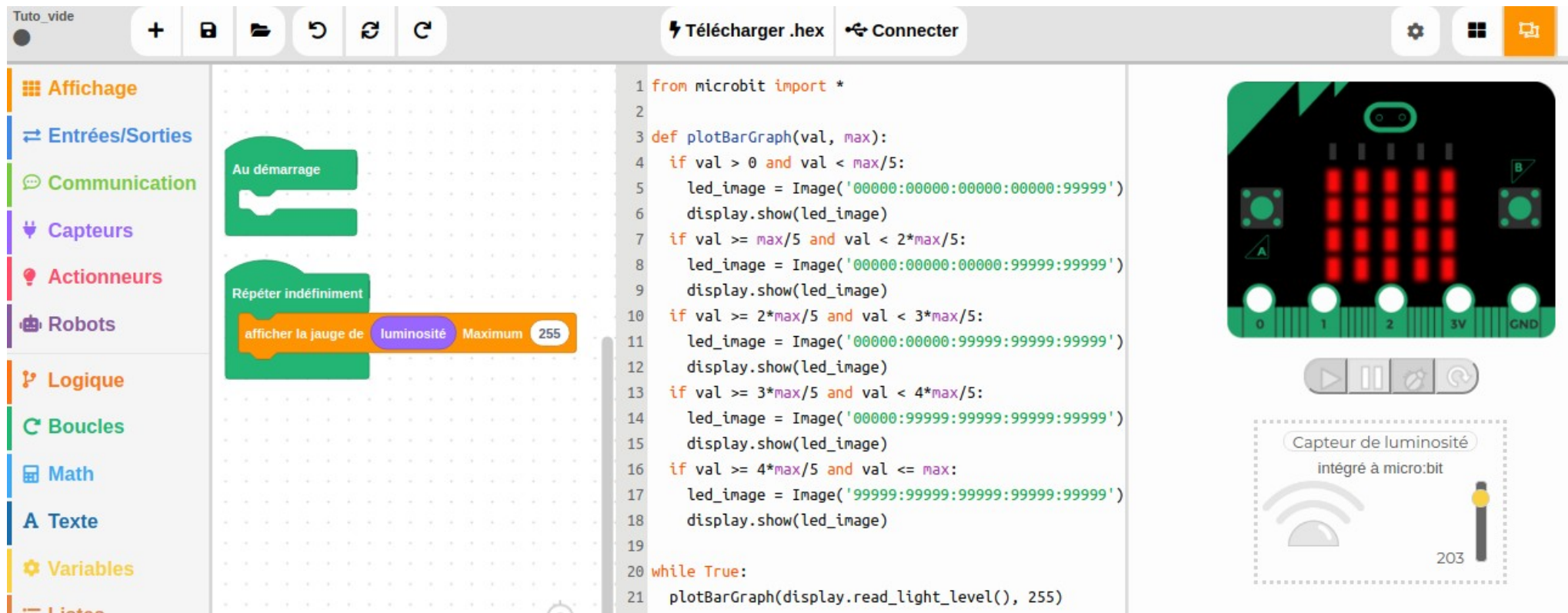
Ouvrez le Site



Vittascience Micro:bit Exo11

Un capteur de luminosité. Faîtes varier la valeur.
Notez la plage de valeur, à quoi correspond elle ?

Expliquez la jauge et les différents seuils.



The screenshot displays the Micro:bit IDE interface. On the left, a sidebar lists categories: Affichage, Entrées/Sorties, Communication, Capteurs, Actionneurs, Robots, Logique, Boucles, Math, Texte, and Variables. The main workspace contains a Python script and a visual representation of the Micro:bit device.

Python Script:

```
1 from microbit import *
2
3 def plotBarGraph(val, max):
4     if val > 0 and val < max/5:
5         led_image = Image('00000:00000:00000:00000:99999')
6         display.show(led_image)
7     if val >= max/5 and val < 2*max/5:
8         led_image = Image('00000:00000:00000:99999:99999')
9         display.show(led_image)
10    if val >= 2*max/5 and val < 3*max/5:
11        led_image = Image('00000:00000:99999:99999:99999')
12        display.show(led_image)
13    if val >= 3*max/5 and val < 4*max/5:
14        led_image = Image('00000:99999:99999:99999:99999')
15        display.show(led_image)
16    if val >= 4*max/5 and val <= max:
17        led_image = Image('99999:99999:99999:99999:99999')
18        display.show(led_image)
19
20 while True:
21     plotBarGraph(display.read_light_level(), 255)
```

Visual Representation:

The right side of the IDE shows a visual representation of the Micro:bit device. It features a 5x5 LED matrix displaying a bar graph with red LEDs. Below the device, a control panel includes a play button, a stop button, a refresh button, and a slider for the light sensor value, currently set to 203. The text "Capteur de luminosité intégré à micro:bit" is displayed above the slider.

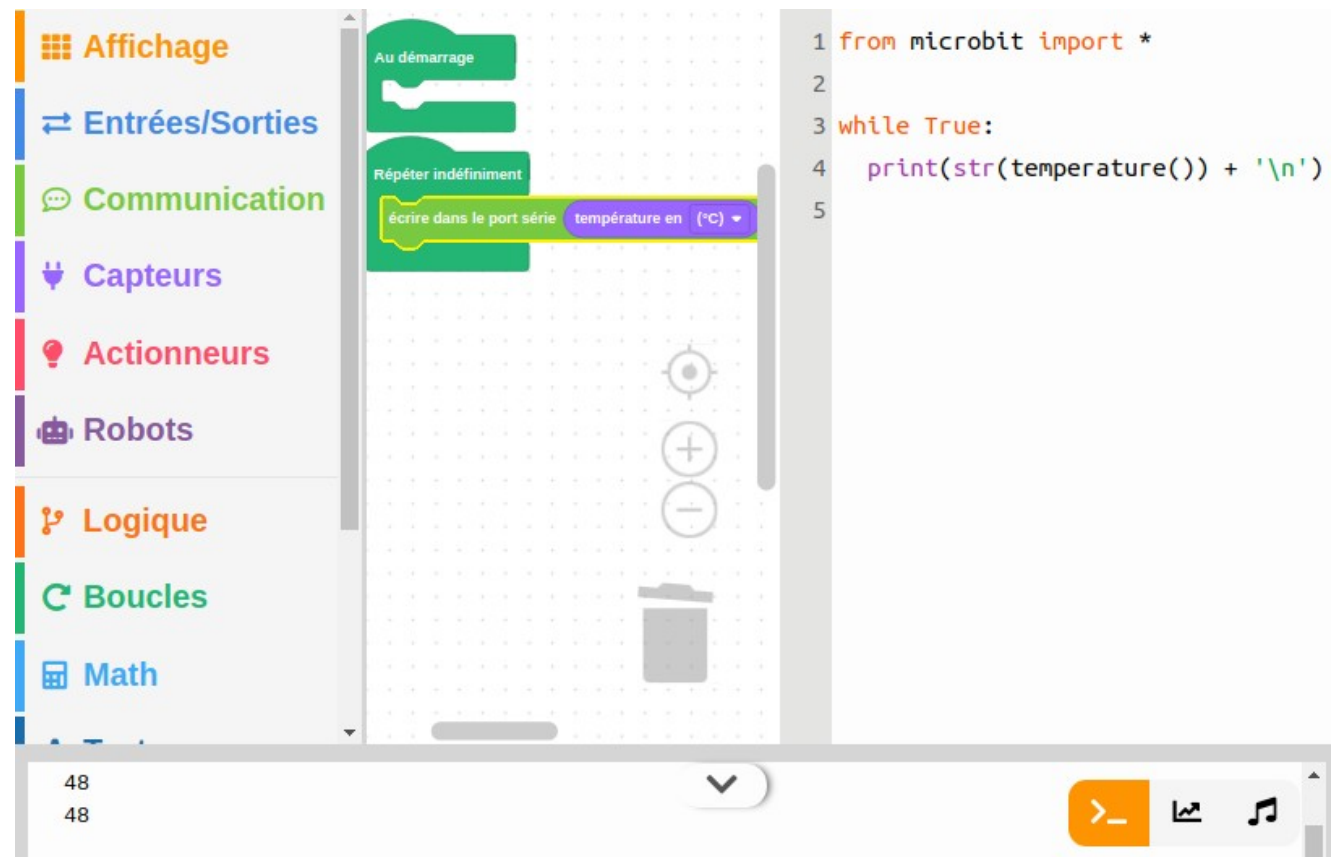
Vittascience Micro:bit Exo12

Capteur de Température .

Attention, il faut le navigateur Chrome, sinon utiliser
Le résultat est dans la console. Faîtes varier la valeur.
Notez la plage de valeur, à quoi correspond elle ?

afficher le nombre 0

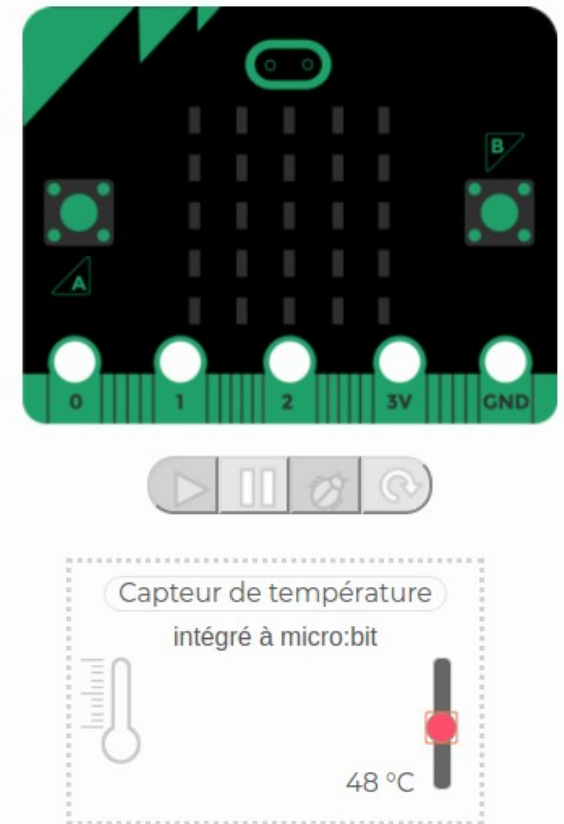
0



The screenshot shows the Micro:bit IDE interface. On the left, a sidebar contains categories: Affichage, Entrées/Sorties, Communication, Capteurs, Actionneurs, Robots, Logique, Boucles, and Math. The main workspace contains a script starting with 'Au démarrage' followed by a 'Répéter indéfiniment' loop. Inside the loop is a block 'écrire dans le port série' with the text 'température en (°C)'. To the right, a code editor shows the following Python code:

```
1 from microbit import *
2
3 while True:
4     print(str(temperature()) + '\n')
5
```

At the bottom, there are two lines of text '48' and a dropdown menu showing a downward arrow.



Vittascience Micro:bit Exo13

Capteur_acc...

+ [Icons] Télécharger .hex [Icon] Connecter [Icon] [Icon]

Affichage
Entrées/Sorties
Communication
Capteurs
Actionneurs
Robots
Logique
Boucles
Math
Texte
Variables

Au démarrage

Répéter indéfiniment

fixer mesure à accélération (mg) x

fixer val à valeur absolue mesure

fixer x à reste de val ÷ 5

fixer y à val ÷ 5

contrôler la led x x y état HAUT

pause 250 milliseconde(s)

contrôler la led x x y état BAS

si mesure > val

faire contrôler la led x 2 y 4 état HAUT

sinon

faire contrôler la led x 2 y 4 état BAS

```
1 from microbit import *
2 import math
3
4 while True:
5     mesure = accelerometer.get_x()
6     val = math.fabs(mesure)
7     x = val % 5
8     y = val / 5
9     display.set_pixel(x,y,9)
10    sleep(250)
11    display.set_pixel(x,y,0)
12    if mesure != val:
13        display.set_pixel(2,4,9)
14    else:
15        display.set_pixel(2,4,0)
16
```

Micro:bit Display

0 1 2 3V GND

Accéléromètre : Axe x
intégré à micro:bit

13 m/s-2

Accéléromètre. Appuyez sur la carte vous verrez la valeur changer.
Vous pouvez changer l'axe x ou y ou z.
Expliquez l'affichage et quelle est la LED qui indique le signe ?

Vittascience Micro:bit Exo14

Boussole magnétique.

Terminer le programme

Tuto_Boussole

Télécharger .hex

Connecter

Affichage

Entrées/Sorties

Communication

Capteurs

Actionneurs

Robots

Logique

Boucles

Math

A Texte

Variables

Listes

Fonctions

Au démarrage

Répéter indéfiniment

fixer Direction à direction du compas (°)

si Direction > 337.5 ou Direction ≤ 22.5

faire afficher la flèche Nord

sinon si Direction > 22.5 et Direction ≤ 67.5

faire afficher la flèche Nord-Ouest

sinon si

faire

sinon si

faire

sinon si

faire

sinon si

faire

sinon si

faire

sinon

faire

pause 1 seconde(s)

```
1 from microbit import *
2
3 while True:
4     Direction = compass.heading()
5     if Direction > 337.5 or Direction <= 22.5:
6         display.show(Image.ARROW_N)
7     elif Direction > 22.5 and Direction <= 67.5:
8         display.show(Image.ARROW_NW)
9     elif False:
10        pass
11    elif False:
12        pass
13    elif False:
14        pass
15    elif False:
16        pass
17    elif False:
18        pass
19    elif False and False:
20        display.show(Image.ARROW_NE)
21    else:
22        pass
23    sleep(0)
24
```

Boussole

intégré à micro:bit

0°

Regardez bien cette structure à choix multiples
if choix1 :
elif choix2 :
...
else :

Vittascience Micro:bit Exo14

Musique.

- Pas de simulation pour l'instant, voici un exemple mais il faut une carte et un casque pour tester.
- Remarquez la notation anglaise des notes en python.

Tuto_OpenLo...

+ [Icons]

Télécharger .hex Connecter

Affichage

Entrées/Sorties

Communication

Capteurs

Actionneurs

Robots

Logique

Boucles

Math

Au démarrage

Répéter indéfiniment

[Musique] jouer les notes sur la broche P0

note Ré à l'octave 4 durée 1

note Fa# à l'octave 4 durée 1

note Sol à l'octave 3 durée 1

```
1 from microbit import *
2 import music
3
4 while True:
5     music.play(['d', 'f#', 'g3'], pin=pin0)
6
```


Vittascience Micro:bit Exo14 suite

- Avec la musique vous pourrez aller plus loin.
- Vous pourrez jouer une note, des notes ou une mélodie ...

Musique

[Musique] jouer la mélodie **Beethoven - Dadadadum** une fois sur la broche **P0**

[Musique] jouer les notes sur la broche **P0**

- note **Ré** à l'octave **4** durée **1**
- note **Fa#** à l'octave **4** durée **1**
- note **Sol** à l'octave **4** durée **1**

[Musique] jouer la fréquence **440** pendant **500** (ms) sur la broche **P0**

[Musique] arrêter la musique sur la broche **P0**

[Musique] définir la pulsation **4** et le tempo **120**

[Musique] obtenir le tempo de la musique

Vittascience Micro:bit

- La partie Vittascience et la simulation se termine avec ces derniers exemples.
- Dans Vittascience, découvrez des activités dans Ressources puis dans support Micro:bit

