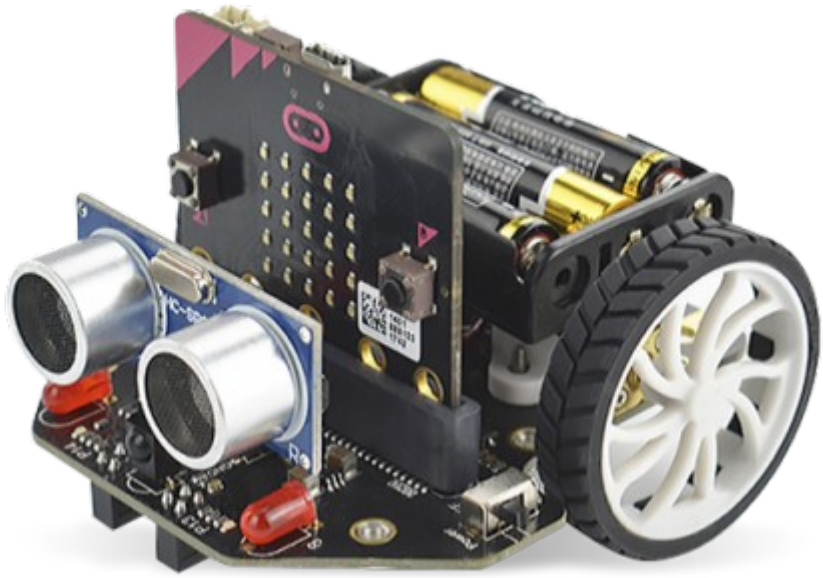
Utiliser le robot Maqueen sur Python

Le robot Maqueen micro:bit est un robot très bon marché (autour de 20€) contrôlé par la carte micro:bit. Il est petit, maniable et facile d'utilisation. Il possède beaucoup de fonctionnalités :

- capteurs de suivi de ligne
- LEDs
- 4 LED RVB neopixel pour éclairage d'ambiance
- capteur de distance ultrason
- buzzer pour effets sonores
- moteurs à engrenage contrôlables séparément par i2c
- alimentation par pack de 3 piles AAA
- capteur infrarouge permettant au robot d'être télécommandé

A l'origine, ce robot se programme par blocs. J'ai développé un module python permettant de le programmer facilement sous Python également.



Installation du module pilote maqueen sous Python

Pour installer le module sur *Mu-editor* :

téléchargez le module maqueen.zip [zip] et décompressez-le dans votre dossier mu_code

Dans Mu, cliquez sur l'icône 

Si vous avez copié le fichier maqueen.py dans le dossier *mu_code*, vous devriez le voir apparaître coté ordinateur. glissez-le sur la carte micro:bit. Vous devriez obtenir quelque chose du genre : maqueenFichier.png

Méthodes fournies par le module

- **avance(vitesse)** : avance en ligne droite. *vitesse* est un nombre entre 0 et 100. Ce paramètre est optionnel. Si non spécifié, c'est la dernière vitesse spécifiée lors de *avance()* ou *setVitesse()* qui sera utilisée.
- **recule()** : fait marche arrière.
- **stop()** : stoppe les moteurs
- **moteurDroit(vitesse)** : fait tourner la roue droite.
- **moteurGauche(vitesse)** : fait tourner la roue gauche.
- **getVitesse()** : renvoie la vitesse paramétrée par *setVitesse()* ou *avance()*
- **setVitesse()** : change la valeur de la vitesse utilisée par *avance*, *recule*, *moteur**
- **distance()** : renvoie la distance (en cm) lue par le capteur ultrason
- **son_r2d2()** et **son_bip()** : effets sonores

Accès aux autres fonctions du robot

Sur le circuit imprimé du robot figurent les adresses des broches pour les LEDs et capteurs de ligne. les voici pour rappel :

- LEDs rouges : 8 (gauche) et 12 (droite). Ex : pin8.write_digital(1)
- Neopixel : pin15
- capteurs de ligne : pin13 (gauche) et pin14 (droite). ex : pin13.read_digital()
- infrarouge : pin16

CTRL+C pour copier, CTRL+V pour coller

```
# Exemple d'éclairage d'ambiance vert avec les neopixels

from microbit import *
from neopixel import NeoPixel

np=NeoPixel(pin15,4)
for i in range(4):
    np[i]=(0,255,0)
np.show()

# np.clear() pour eteindre les neopixels
```


```
# Exemple d'éclairage d'ambiance vert avec les neopixels
from microbit import *
from neopixel import NeoPixel
np=NeoPixel(pin15,4)
for i in range(4):
    np[i]=(0,255,0)
np.show()

# np.clear() pour eteindre les neopixels
```

Utilisation du module

Voici un exemple concret vous permettant de tester le module depuis le REPL afin de vous assurer que tout fonctionne. Je vous conseille de tester cela avec le robot sur le dos pour éviter qu'il parte avec un fil à la patte !

```
>>> from maqueen import Maqueen

>>> mq=Maqueen()

>>> mq.distance() # permet de vérifier le module ultrason

>>> mq.avance(10)

>>> mq.stop()

>>> mq.moteurDroit()

>>> mq.moteurGauche(-10)

>>> mq.stop()
```

Exemple de mini-projet

Il est temps maintenant de passer à la réalisation d'un véritable projet. Il y a les incontournables classiques comme le robot suiveur de ligne, le robot télécommandé par infrarouge ou le robot éviteur d'obstacles ou la course de robots A vous de jouer !

Listing du module maqueen

```
# The MIT License (MIT)
# Copyright (c) 2016 British Broadcasting Corporation.
# This software is provided by Lancaster University by arrangement with the BBC.
# Permission is hereby granted, free of charge, to any person obtaining a
# copy of this software and associated documentation files (the "Software"),
# to deal in the Software without restriction, including without limitation
# the rights to use, copy, modify, merge, publish, distribute, sublicense,
# and/or sell copies of the Software, and to permit persons to whom the
# Software is furnished to do so, subject to the following conditions:
# The above copyright notice and this permission notice shall be included in
# all copies or substantial portions of the Software.
# THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
# IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
# FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL
# THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
# LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING
# FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER
# DEALINGS IN THE SOFTWARE.
# Olivier Lecluse
# Avril 2019

import microbit
import time
import machine
```

```
import music
class Maqueen():
    def __init__(self,addr=0x10):
        """Initiaisation robot
        addr : adresse i2c. 0x10 par defaut"""
        self.addr=addr
       self._vitesse=0 # vitesse entre 0 et 100

    def getVitesse(self):
        return self._vitesse

    def setVitesse(self, v):
        self._vitesse=v

    def moteurDroit(self, v=None):
        if v==None:
            v=self._vitesse
           sens=0
       if v>=0 else 1 # sens moteur
        vit=abs(v)*255//100   # vitesse moteur 0..255
        microbit.i2c.write(self.addr,bytearray([0,sens, vit]))

    def moteurGauche(self, v=None):
        if v==None:
            v=self._vitesse
        sens=0
       if v>=0 else 1 # sens moteur
        vit=abs(v)*255//100   # vitesse moteur 0..255
        microbit.i2c.write(self.addr,bytearray([2,sens, vit]))

    def avance(self,v=None):
        if v != None:
            self._vitesse=v
        self.moteurDroit()
        self.moteurGauche()

    def recule(self):
        self.moteurDroit(-self._vitesse)
        self.moteurGauche(-self._vitesse)

    def stop(self):
        microbit.i2c.write(self.addr,bytearray([0,0,0]))
        microbit.sleep(1)
        microbit.i2c.write(self.addr,bytearray([2,0,0]))

    def distance(self):
        """Calcule la distance à l'obstacle en cm
        pin1 : Trig
        pin2 : Echo"""
        microbit.pin1.write_digital(1)
        time.sleep_ms(10)
        microbit.pin1.write_digital(0)
        microbit.pin2.read_digital()
        t2 = machine.time_pulse_us(microbit.pin2, 1)
        d = 340 * t2 / 20000
        return d

    def son_r2d2(self):
```

```
        tune=["A7:0", "G7:0",
"E7:0","C7:0","D7:0","B7:0","F7:0","C8:0","A7:0","G7:0","E7:0","C7:0","D7:0
","B7:0","F7:0","C8:0"]
        music.play(tune)

    def son_bip(self):
        for i in range(2):
            freq=2000
            while freq>1000:
                music.pitch(int(freq),10)
                freq*=0.95
            freq=1000
            while freq<3000:
                music.pitch(int(freq),10)
                freq*=1.05

# The MI
```