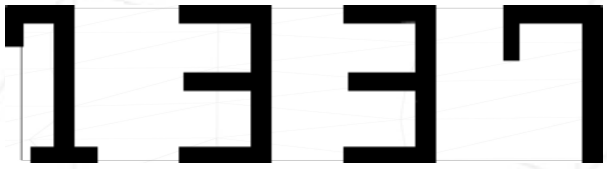


Introduction à la robotique

Ce projet a pour but de vous apprendre à utiliser le langage Python pour pouvoir commander un robot.

Table des matières

Introduction

1. Description

EV3 est un robot intelligent et programmable de Lego. Il a les caractéristiques suivantes :

- EV3 est équipé d'un moteur pour actionner ses roues, ses bras et ses autres éléments animés.
- Il est équipé de capteurs pour obtenir des informations sur son environnement.
- Il est équipé d'un petit ordinateur programmable pour contrôler ses mouvements.

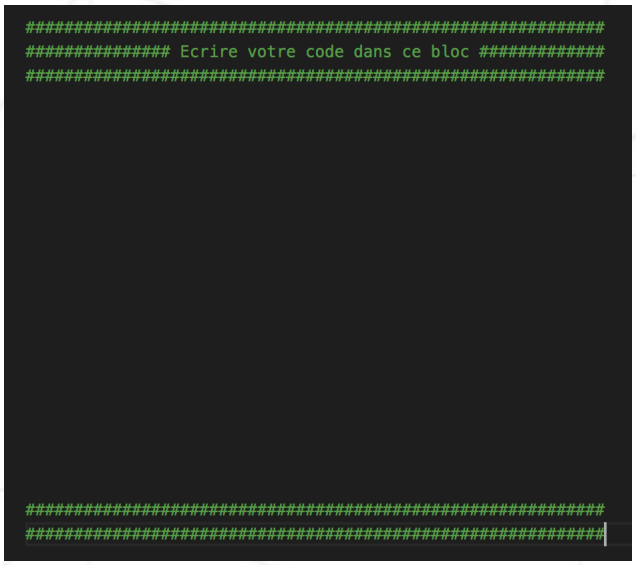
2. Programmation du robot

Pour programmer le robot, vous allez utiliser *Python*, le langage de programmation le plus populaire en 2018.

Faisons connaissance

1. *Vos premiers pas*

Sur Visual Studio Code, naviguez vers le dossier *Documents* puis ouvrez le dossier Robotics.
Dans le fichier *exercice1.py*, placez votre curseur dans la partie réservée à votre code:



Pour faire bouger le robot, vous utiliserez la fonction suivante :

ms.on_for_rotations(steering, speed, rotations)

Cette fonction permet au robot d'avancer. Le mouvement est contrôlé par trois paramètres :

- **steering :** indique la direction du robot ;
- **speed :** indique la vitesse du robot (de 1 à 100) ;
- **rotations :** indique la distance à parcourir, fixée par le nombre de rotations de roues.

Exemples :

ms.on_for_rotations(0, 20, 2) : fait avancer le robot en ligne droite (paramètre ***steering*** = 0), à 20% de sa vitesse maximale (paramètre ***speed*** = 20), à une distance équivalente à deux rotations de roues (paramètre ***rotations*** = 2).

ms.on_for_rotations(50, 20, 1) : fait tourner le robot d'environ 90 degrés vers la droite (paramètre steering = 50) à 20% de sa vitesse maximale (paramètre speed = 20).

Ci-dessous les valeurs du paramètre ***steering*** suivant la direction souhaitée du robot :

Exercice 1

Partie I :

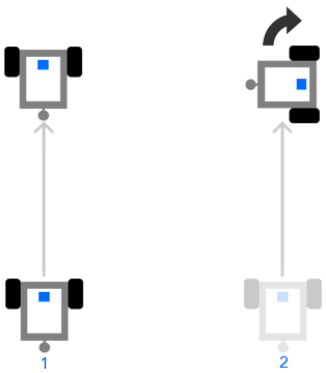
Dans le fichier *exercice1.py*, utilisez la fonction ***ms.on_for_rotations*** comme dans l'exemple ci-haut pour écrire un programme qui permet au robot d'avancer en ligne droite avec une vitesse = 20 et en faisant deux rotations de roues.

Partie II :

Votre ligne de code fait rouler le robot en ligne droite. Maintenant, ajoutez une nouvelle ligne en utilisant la fonction ***ms.on_for_rotations*** avec les paramètres suivants :

- steering = 50
- speed = 20
- rotations = 1

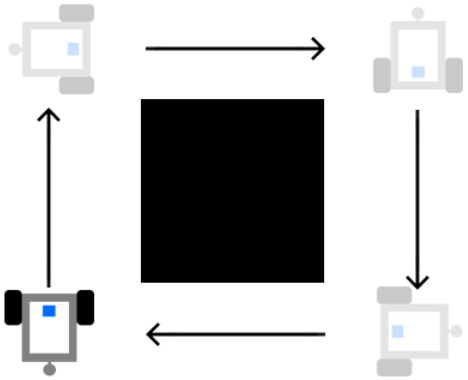
Les lignes de code seront exécutées l'une après l'autre. Le mouvement de votre robot doit ressembler à l'illustration ci-dessous :



Le robot commence par rouler en ligne droite, puis fait un quart de tour vers la droite.

2. Faisons un tour

Dans cette partie, vous allez apprendre à avoir plus de contrôle sur les mouvements du robot. En utilisant toujours la fonction ***ms.on_for_rotations***, l'objectif est de faire tourner le robot autour du carré tracé sur la table comme suit :



Ne vous en faites pas si vous n'arrivez pas à faire un trajet rectangulaire parfait.

Exercice 2 - *Partie I*

- Ouvrez le fichier *exercice2.py*.
- En vous inspirant de l'exercice 1, faites tourner le robot un tour complet autour du carré tracé sur la table.

Python vous permet de répéter des commandes avec des boucles, ça pourrait être utile, hein ?

La boucle for:

La boucle ***for*** permet de répéter des instructions un nombre de fois donné. Le nombre de répétitions est indiqué dans la fonction ***range(nombre de répétition).***

Par exemple, la boucle ci-dessous vous permet de faire rouler le robot en ligne droite, puis le faire tourner un quart de tour vers la gauche. Ces deux mouvements seront répétés 2 fois comme indiqué dans la fonction range(**2**) :

```
for i in range(2):
    ms.on_for_rotations(0,20,1)
    ms.on_for_rotations(-50,20,1)
```

Notez que toutes les lignes de code à l'intérieur de la boucle ***for*** doivent être précédées d'une tabulation.

Bouton tabulation

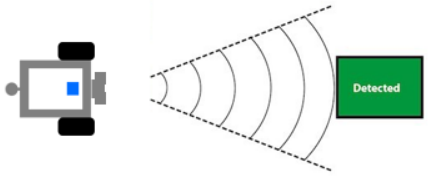
Le code que vous avez écrit dans la partie 1 est fonctionnel, mais vous avez dû remarquer la répétition des deux mêmes instructions plusieurs fois.

Exercice 2 - *Partie II*

Toujours dans le fichier *exercice2.py*, modifiez le code écrit dans la partie 1 en utilisant la boucle ***for*** pour éviter les répétitions et rendre votre code plus élégant.

Évitons les obstacles

Dans cette partie, vous utiliserez un capteur ultrasonore pour calculer les distances devant votre robot et l'arrêter avant d'heurter un obstacle :



Comme pour les moteurs, vous disposez de fonctions qui vous permettent d'exploiter les informations retournées par le capteur ultrasonore. Dans ce chapitre, vous utiliserez la fonction suivante:

us.distance_centimeters_continuous : cette fonction donne la distance entre le robot et l'objet devant lui.

Vous utiliserez une autre boucle dans cette partie : la boucle ***while***. Cette dernière permet de répéter des lignes de code tant qu'une condition est réalisée.

Par exemple: tant que la distance est supérieure à 10 cm entre le robot et un obstacle, continuez tout droit :

```
while us.distance_centimeters_continuous > 10 :
    ms.on(0,20)
```

- Une fois la condition n'est plus réalisée, le code à l'intérieur de la boucle ne sera plus exécuté.
- La fonction ***ms.on(steering, speed)*** fera avancer le robot tant que la boucle est exécutée.
- Comme pour la boucle ***for***, le code à l'intérieur doit être précédé d'une tabulation.

Exercice 3 – *Partie I*

- Ouvrez le fichier *exercice3.py*.
- En utilisant la fonction ***ms.on*** et la boucle ***while*** comme dans l'exemple précédent, faites en sorte que le robot s'arrête lorsqu'il détecte un obstacle à 20 cm de distance.

Vous avez remarqué que la boucle ***while*** exécute du code tant que la condition donnée est réalisée. Pour créer une boucle infinie, on utilise le mot clé ***True*** comme condition:

```
while True :
    ms.on(0,20)
```

Cette boucle continue à s'exécuter jusqu'à ce que vous arrêtez le programme. En effet, le robot continu à rouler même lorsqu'il heurte un obstacle.

Testez le code mentionné dans l'exemple ci-dessus. Pour arrêter votre programme, cliquez sur Stop en haut de Visual Studio Code (carré rouge) :

Au lieu d'arrêter le robot lorsqu'il détecte un obstacle, cette fois-ci vous le ferez tourner à chaque fois qu'il est face à un objet afin de l'éviter et de continuer à rouler.

En plus de la boucle ***while***, vous aurez besoin de conditions.

Les conditions ***if/else*** permettent d'exécuter une ou plusieurs instructions **si** une condition est vérifiée, d'autres instructions **sinon**. Par exemple, faire une action donnée si une valeur est positive, une autre action si cette valeur est négative.

Dans l'exemple suivant, **si** la distance devant le capteur ultrasonore est supérieure à 10 cm, le robot continue à avancer. **Sinon**, le robot fait un quart de tour vers la droite :

```
if us.distance_centimeters_continuous > 10:
    ms.on(0,20)
else:
    ms.on_for_rotations(50,20,1)
```

Exercice 3 – *Partie II*

En combinant la boucle infinie (while True:) et les conditions (if/else) comme dans les exemples précédents, écrivez un programme qui permet au robot de continuer à rouler en évitant les obstacles à 20 cm devant lui.