

MACE ROBOTICS

Documentation du robot MR-25

Mise à jour : 25/09/25

Table des matières

1) Présentation du robot MR-25.....	3
2) Logiciels & matériels nécessaire.....	4
3) Mise en marche du robot.....	5
4) Rechargement de la batterie.....	8
5) Indicateur de l'état de la batterie.....	8
6) Avancer le robot.....	9
7) Les déplacements du robot.....	10
8) Lecture des encodeurs.....	11
9) Lecture de la tension/courant de la batterie.....	12
10) Lecture de la version du firmware.....	13
11) Lecture d'un capteur de proximité.....	13
12) Utilisation du buzzer.....	15
13) Utilisation de la led RGB.....	15
14) Contrôle des moteurs.....	16
15) Position du robot.....	17
16) Branchement de la batterie.....	17
17) Mise à jour du firmware du microcontrôleur Pico.....	19
18) API.....	21

1) Présentation du robot MR-25

MR-25 est un petit robot mobile basé sur une carte Raspberry Pi. Il est très facile à programmer et personnalisable. Une taille de 130 mm de diamètre vous permet de facilement le programmer sur une table de bureau.

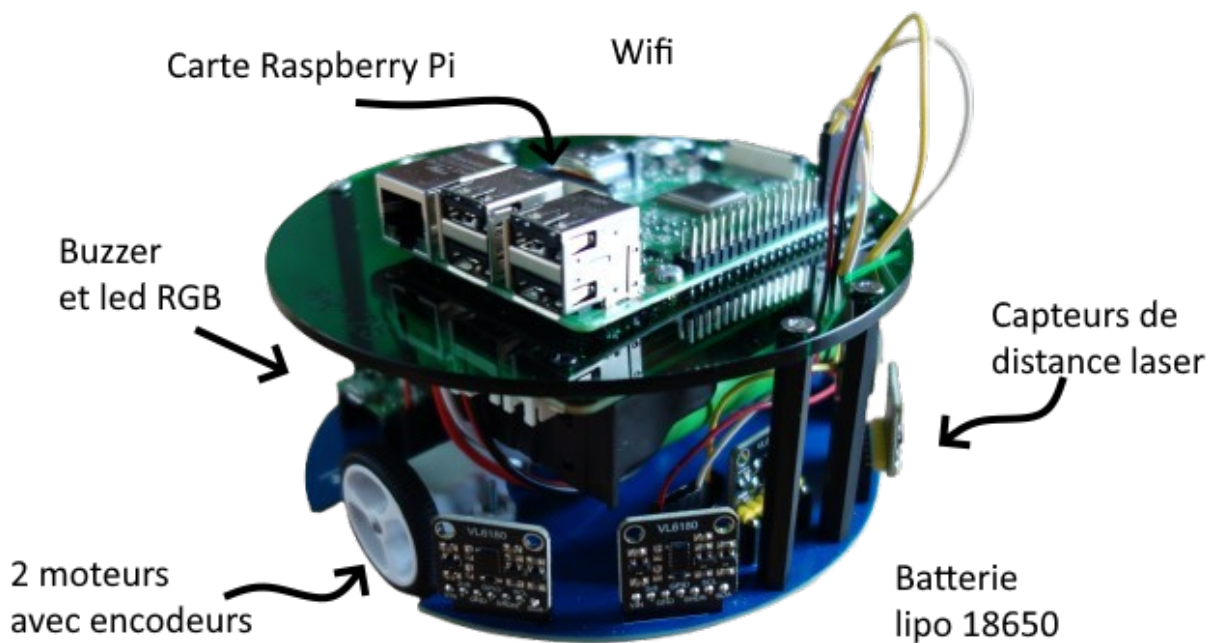


Figure 1: Robot MR-25

Caractéristiques techniques :

- Dimensions :
 - Diamètre de 130 mm
 - Hauteur : 85 mm
- Carte Raspberry Pi 3, 2W ou 5
- Une Led RGB
- Microcontrôleur : Raspberry Pi Pico
- Un buzzer
- Un capteur de courant, mesure tension batterie (INA219)
- Deux encodeurs magnétiques
- Deux moteurs à courant continu 100 RPM
- 5 capteurs de distances Time-of-Flight VL6180X.
- Batterie 18650 (3S)

Le microcontrôleur Raspberry Pico communique via une liaison série avec la carte Raspberry Pi.

2) Logiciels & matériels nécessaire

- Putty : <https://www.putty.org/>

3) Mise en marche du robot

Activez l'interrupteur ON/OFF sur la position ON. Le robot doit émettre un petit bip après la mise en marche.

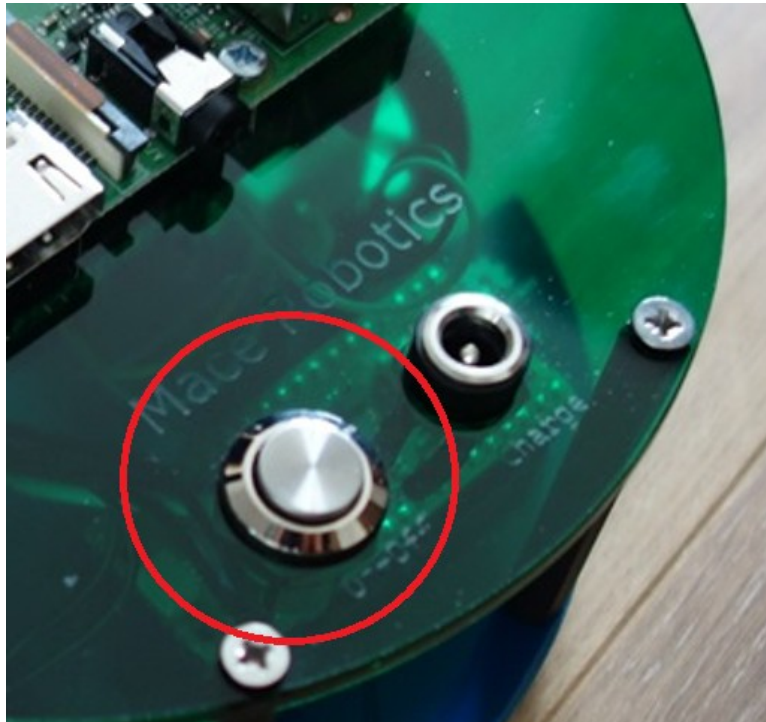


Figure 2: Bouton marche/arret

Connexion au point d'accès wifi du robot :

- Nom du point d'accès : **ROBOT_MR25**
- Mot de passe : **mrobot2025**

MACE ROBOTICS

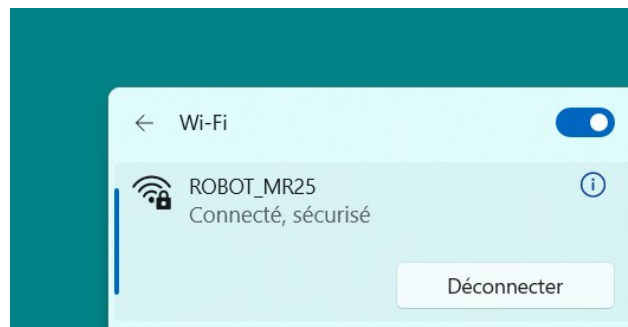
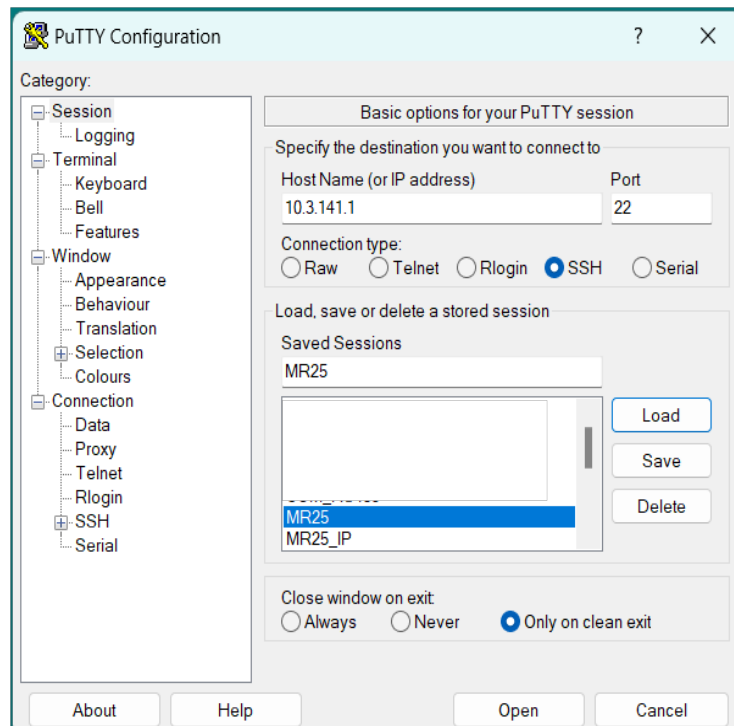


Figure 3: Point accès Wifi du Robot

- **Adresse IP** : 10.3.141.1
- **Port** : 22



MACE ROBOTICS

Connexion avec putty :

- login : **pi**
- mot de passe : **pi**

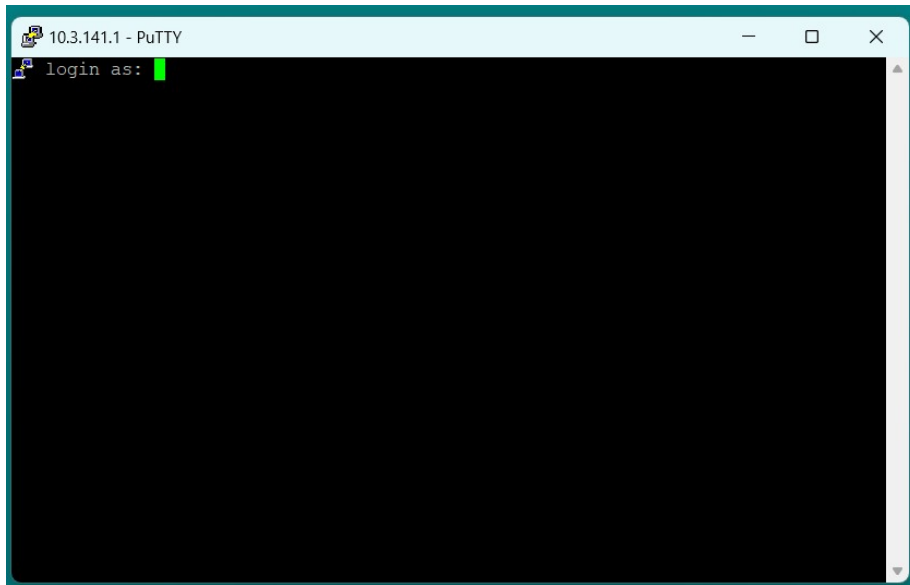


Figure 4: Connexion avec Putty

Dossier MR25 pour les exemples de programmes Python.

Exemple : python3 main.py

4) Rechargement de la batterie

Pour recharger la batterie du robot MR-25 vous avez besoin :

- Adaptateur secteur 230V – 12VDC fourni avec le robot.

Ensuite pour recharger la batterie du robot :

1. Insérer la prise jack mâle dans le connecteur jack femelle du robot
2. La batterie est rechargée lorsque la led est de couleur verte sur l'adaptateur secteur

5) Indicateur de l'état de la batterie

La led rouge (LOW BATT) s'allume lorsque la batterie du robot est trop basse.



Figure 5: Led "Low Batt"

6) Avancer le robot

Un programme simple en langage Python pour faire avancer le robot MR-25 :

```
$ python3 main.py
```

```
#!/usr/bin/python
import MR25
import time, sys

print("Exemple robot MR-25")

while True:
    print("Forward")
    MR25.forward(25)
    time.sleep(2)
    MR25.stop()
    time.sleep(2)
#End of file
```

- Vitesse de 25 %
- Arrêt du robot pendant 2 secondes

7) Les déplacements du robot

Un programme pour faire avancer, reculer et tourner à droite et gauche le robot MR-25:

```
#!/usr/bin/python
import MR25
import time, sys

while True:
    MR25.forward(25)
    time.sleep(2)

    MR25.back(25)
    time.sleep(2)

    MR25.turnRight(50)
    time.sleep(2)

    MR25.turnLeft(50)
    time.sleep(2)

    MR25.stop()
    time.sleep(2)
#End of file
```

8) Les déplacements précis du robot

Un programme pour faire avancer et tourner le robot de manière précise :

- **forwardmm()** : avancer ou reculer d'une distance en millimètre
 - distance positive : avancer
 - distance négative: reculer
- **turnAngle()** : tourné d'un angle en degré
 - angle positive : tourner à droite
 - angle négative: tourner à gauche

```
#!/usr/bin/python
import MR25
import time, sys

while True:
    MR25.forwardmm(-100)# reculer de 100 mm
    time.sleep(2)

    MR25.turnAngle(90)# tourner de 90°
    time.sleep(1)

#End of file
```

9) Lecture des encodeurs

Un exemple de lecture des deux encodeurs du robot. Le robot est équipé de deux encodeurs magnétiques sur chaque moteur à courant continu :

```
#!/usr/bin/python
import MR25
import time, sys

while True :
    coderRight = MR25.encoderRight()
    coderLeft = MR25.encoderLeft()

    print("Encoder Right =", coderRight)
    print("Encoder Left =", coderLeft)
    time.sleep(2)

#End of file
```

Exemple pour remettre à zéro le compteur des encodeurs :

```
#!/usr/bin/python
import MR25
import time, sys

while True:
    print("Reset encodeur : ")
    MR25.encoderReset()

#End of file
```

Résolution des encodeurs :

- 1000 ticks pour un tour de roue
- Tick positive si la roue avance
- Tick négatif si la roue recule

10) Lecture de la tension/courant de la batterie

Un exemple de lecture de la tension de la batterie et du courant :

- L'unité est en volt et mA
- La tension nominal de la batterie est de 11,1V
- Utilisation du capteur INA219 (INA219.py)

```
#!/usr/bin/python
import MR25
import time, sys

while True:
    voltage = MR25.battery()
    print("Tension batterie =", voltage)
    current = MR25.batteryCurrent()
    print("Courant batterie en mA =", current)
    time.sleep(2)

#End of file
```

11) Lecture de la version du firmware

Un exemple de lecture de la version du firmware du microcontrôleur Pico :

```
#!/usr/bin/python
import MR25
import time, sys

while True:
    print("Lecture version firmware du Pico")
    version = MR25.firmwareVersion()
    print("Version firmware = ", version)
    time.sleep(2)

#End of file
```

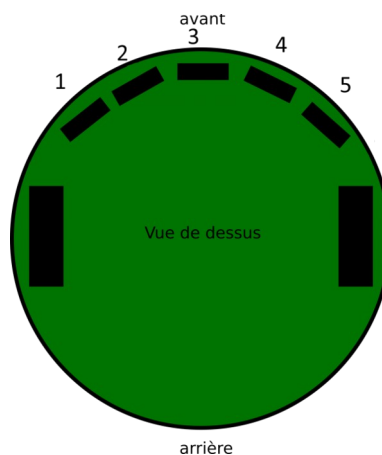
12) Lecture d'un capteur de proximité

Le robot MR-25 est équipé de cinq connecteurs afin de recevoir des capteurs de proximité de type :

- [VL6180X](#)

Le robot MR-25 peut avoir cinq capteurs maximum.

Voici une vue de dessus du robot et le placement des cinq capteurs à l'avant du robot :



MACE ROBOTICS

Lecture du capteur sur l'emplacement n°1 :

```
#!/usr/bin/python
import MR25
import time, sys

while True :
    distance1 = MR25.proxSensor(1)
    print("Distance capteur 1 =", distance1)
    time.sleep(2)

#End of file
```

Caractéristiques du capteur :

- Sortie en millimètres
- De 0 à 90 mm (dépend de la lumière et de la surface de réflexion)
 - De préférence réflexion sur une surface blanche.
- Si la distance est supérieure à 90mm, la sortie = 255

13) Utilisation du buzzer

Exemple d'utilisation du buzzer du robot :

```
#!/usr/bin/python
import MR25
import time, sys

while True :
    MR25.buzzer(600) # 600 Hz
    time.sleep(2)

    MR25.buzzer(800) # 800 Hz
    time.sleep(2)

    MR25.buzzer(2000) # 2000 Hz
    time.sleep(2)

    MR25.buzzerStop()
    time.sleep(5)

#End of file
```

14) Utilisation de la led RGB

Utilisation de led RGB du robot :

```
#!/usr/bin/python
import MR25
import time, sys

while True :
    MR25.ledRGB("100") # rouge
    time.sleep(2)

    MR25.ledRGB("010") # vert
    time.sleep(2)

    MR25.ledRGB("001") # bleu
    time.sleep(2)

#End of file
```

15) Contrôle des moteurs

Exemple, pour faire tourner le robot vers la droite avec la gestion des moteurs à deux vitesses différentes :

- motorRight(Direction, vitesse)
 - Direction : 0 ou 1
 - Vitesse : 0 à 100

```
#!/usr/bin/python
import MR25
import time, sys

while True:
    print("Forward")
    MR25.motorRight(1,25) # contrôle de la roue droite
    time.sleep(2)
    MR25.motorRight(0,25)
    time.sleep(2)
    MR25.stop()
    time.sleep(4)

#End of file
```

Deuxième exemple :

```
#!/usr/bin/python
import MR25
import time, sys

while True:
    print("Forward")
    MR25.motorLeft(1, 50) # contrôle de la roue gauche
    time.sleep(2)
    MR25.motorRight(0,25)
    time.sleep(2)
    MR25.motorLeft(0,25)
    MR25.motorRight(0,25)
    time.sleep(2)
    MR25.stop()
    time.sleep(4)

#End of file
```

16) Position du robot

Trois fonctions permettent de connaître la position et l'orientation du robot sur un repère orthonormé :

- **orientation()** : renvoie l'orientation du robot en degré
- **positionX()** : renvoie la position sur l'axe X du robot, en millimètre
- **positionY()** : renvoie la position sur l'axe Y du robot, en millimètre

Un exemple simple de lecture de la position du robot :

```
#!/usr/bin/python
import MR25
import time, sys

while True :
    print("Orientation =", MR25.orientation())
    print("Orientation =", MR25.positionX())
    print("position Y du robot en mm =", MR25.positionY())
    time.sleep(2)

#End of file
```

17) Branchement de la batterie

Voici le branchement de la batterie sur la carte châssis :

- Fil rouge vers l'avant du robot

MACE ROBOTICS

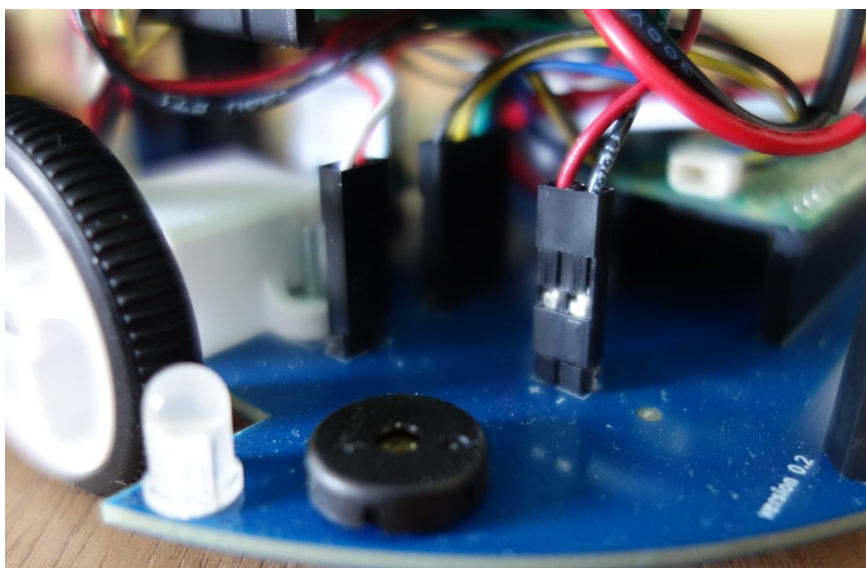


Figure 6: Branchement de la batterie

- Fil noir vers l'arrière

18) Mise à jour du firmware du microcontrôleur Pico

Le logiciel Thonny est nécessaire : <https://thonny.org/>

- Allumer le robot MR-25
- Branchez un câble micro-USB sur le connecteur de la carte Raspberry Pico et votre PC
- Ouvrir le logiciel Thonny

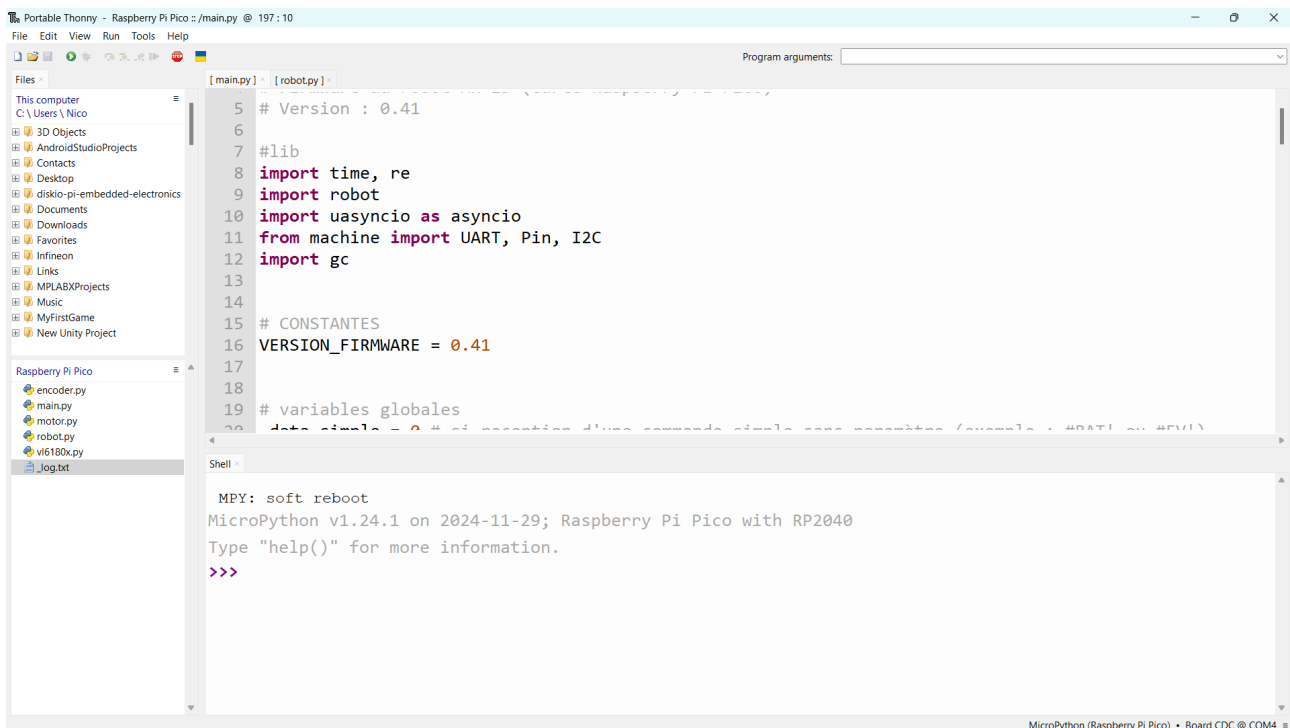
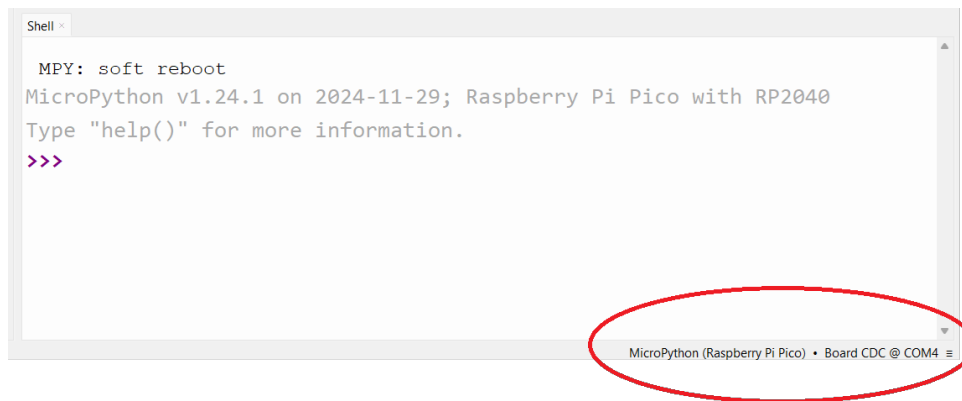


Figure 7: Logiciel Thonny

MACE ROBOTICS

Faire un clic gauche sur le menu ci-dessous pour choisir la carte Pico :



19) API

- **battery()** : lecture de la tension batterie
- **batteryCurrent()** : lecture du courant de la batterie
- **proxRead()** : lecture d'un capteur de proximité
- **proxReadAll()** : lecture des 5 capteurs de proximité
- **encoderRight()** : lecture encodeur droit
- **encoderLeft()** : lecture encodeur gauche
- **encoderReset()** : reset encodeurs
- **ledRgb()** : gestion de la led RGB
- **buzzer()** : gestion du buzzer
- **buzzerStop()** : arrêt du buzzer
- **motorRight()** : contrôle moteur droit
- **motorLeft()** : contrôle moteur gauche
- **forward()** : avancer
- **back()** : reculer
- **turnRight()** : tourner à droite
- **turnLeft()** : tourner à gauche
- **stop()** : stop robot