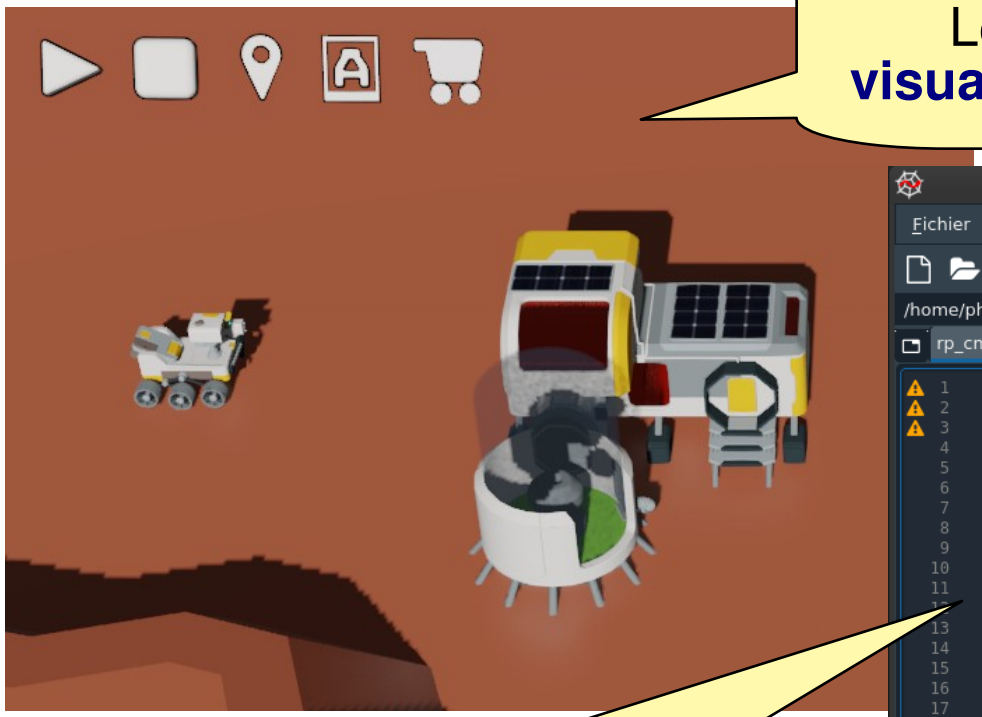
SNT

LYCÉE LÉONARD DE VINCI

Jumelage numérique de Ropy avec Maqueen

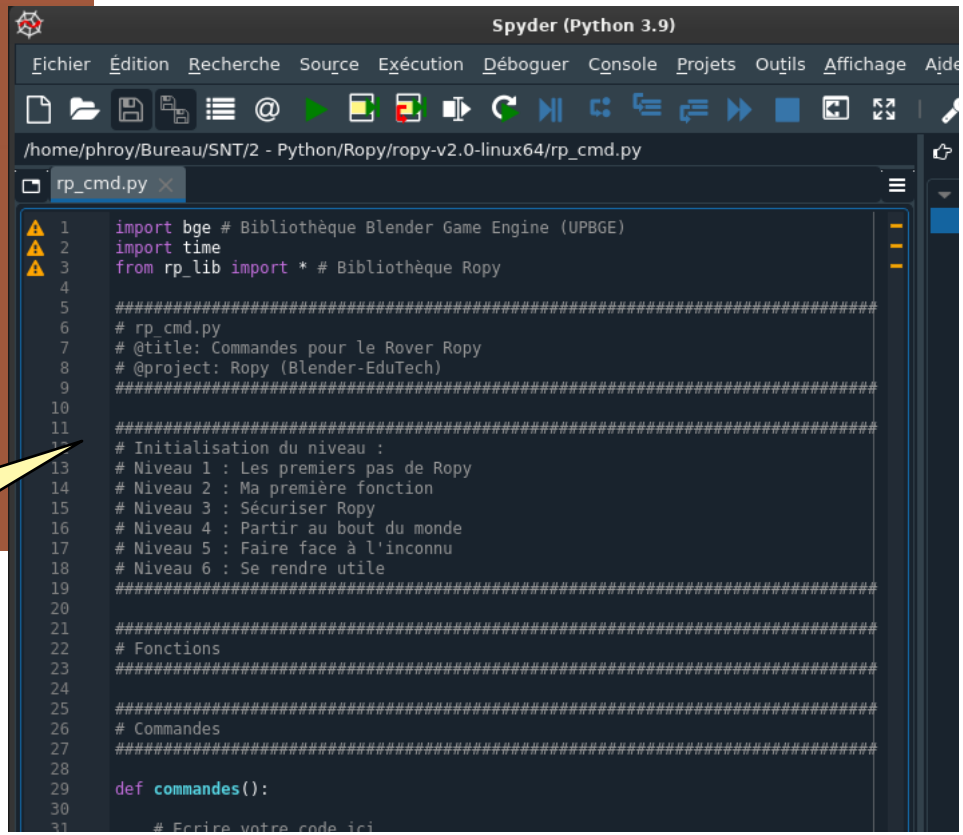
Présentation de Ropy et de son environnement de programmation

Ropy est un rover martien qui se commande grâce au langage **Python**. L'interface de programmation se décompose en 2 fenêtres : un éditeur de texte et le simulateur.



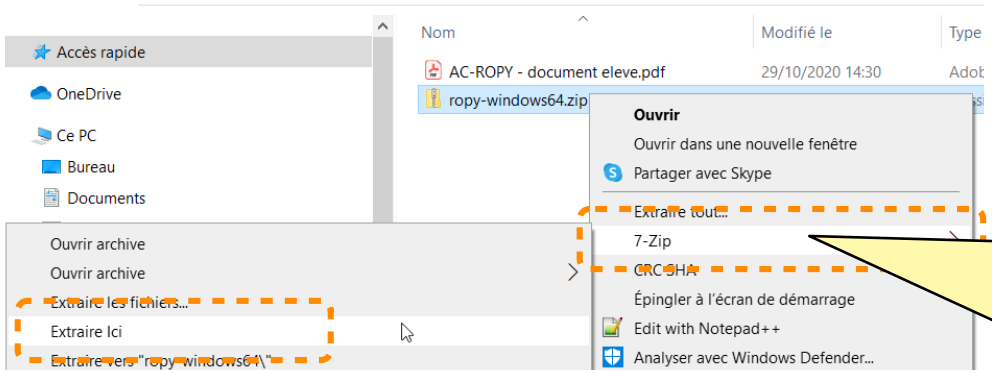
Le **simulateur** permet de **visualiser l'évolution du Rover**.

Un **éditeur de texte** (Notepad++, Spyder, Atom, Emacs, ...) pour **écrire le programme** en **Python**.



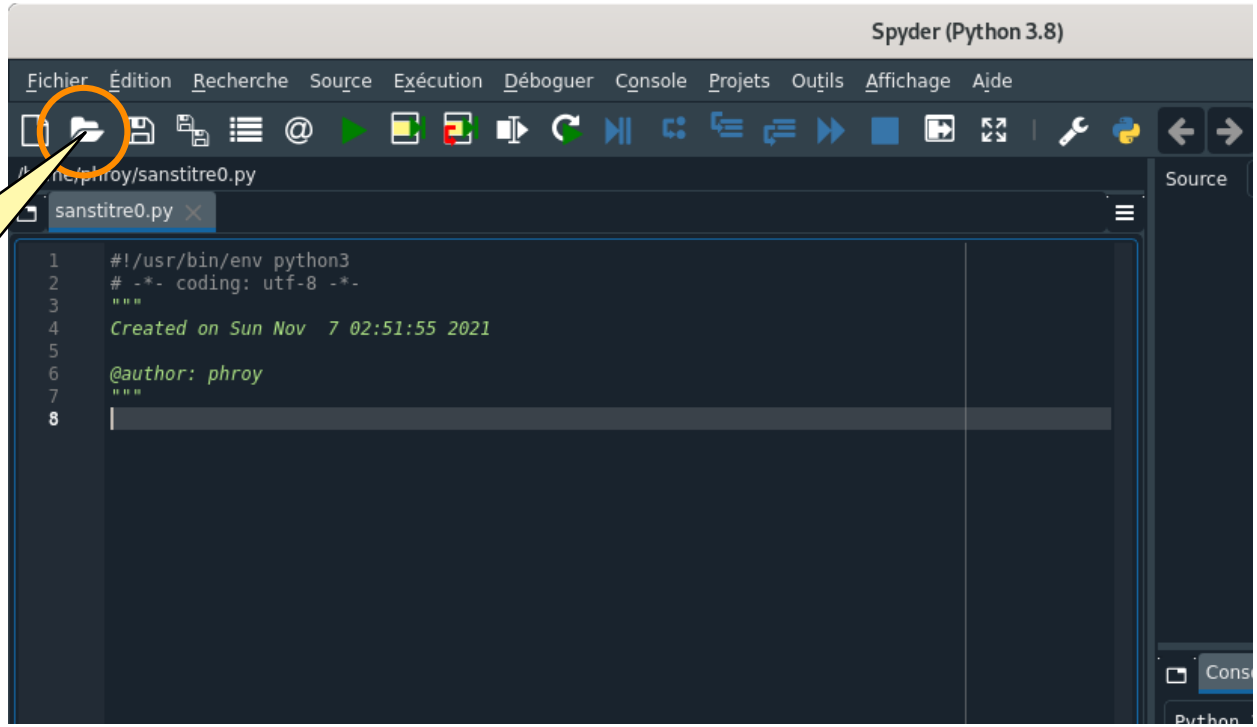
Éditer le programme avec Spyder

Ouvrir le fichier rp_cmd.py



1 : Récupérer l'archive **ropy-windows64.zip** et la décompresser avec **7-Zip** dans votre répertoire. L'extraction va créer le répertoire ropy

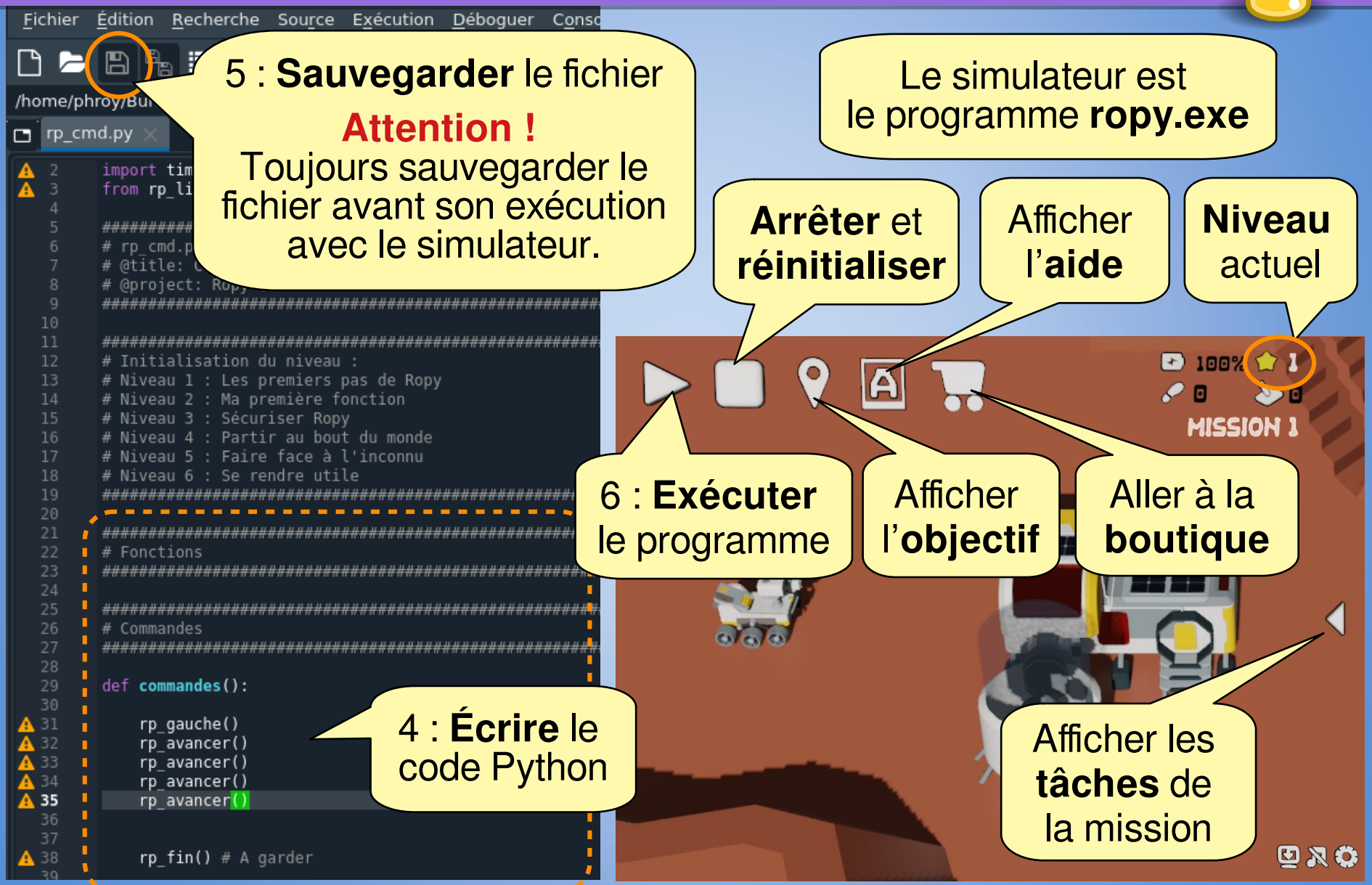
2 : Lancer le Logiciel **Spyder**.



3: Ouvrir le fichier Python à éditer **rp_cmd.py** (Ropy commandes) présent dans le répertoire ropy.

Éditer le programme avec Spyder

Exécution du programme



Contenu du fichier rp_cmd.py

Le fichier **rp_cmd.py** comporte 4 sections.

```
import bge # Bibliothèque Blender Game Engine (UPBGE)
import time
from rp_lib import * # Bibliothèque Ropy

##############################################################################
# rp_cmd.py
# @title: Commandes pour le Rover Ropy
# @project: Ropy (Blender-EduTech)
##############################################################################

##############################################################################
# Initialisation du niveau :
# Niveau 1 : Les premiers pas de Ropy
# Niveau 2 : Ma première fonction
# Niveau 3 : Sécuriser Ropy
# Niveau 4 : Partir au bout du monde
# Niveau 5 : Faire face à l'inconnu
# Niveau 6 : Se rendre utile
##############################################################################

##############################################################################
# Fonctions
##############################################################################



##############################################################################
# Commandes
##############################################################################

def commandes():

    rp_gauche()
    rp_avancer()
    rp_avancer()
    rp_avancer()
    rp_avancer()

    rp_fin() # A garder


##############################################################################
# En: Externals calls << DONT CHANGE THIS SECTION >>
# Fr: Appels externes << NE PAS MODIFIER CETTE SECTION >>
##############################################################################

if __name__=='start':
    thread_cmd_start(commandes)
if __name__=='stop':
    thread_cmd_stop()
```

Le code doit être indenté (décalé sur la droite) avec la touche Tab

Import des bibliothèques
Ne pas modifier cette section

Fonctions : section pour le codage de **vos fonctions**

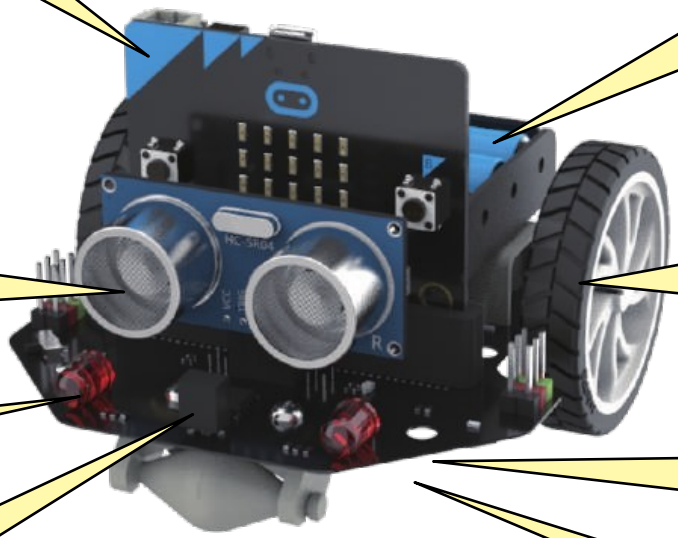
Commandes : section pour le codage des commandes du robot

La commande `rp_fin()` est à conserver.

Appels du simulateur
(Blender Game Engine)
Ne pas modifier cette section

Présentation du robot Maqueen

Maqueen est un robot mobile qui peut se piloter avec une carte **micro:bit**.



Carte micro:bit

Alimentation 3 piles AAA

Capteur Ultrason (capteur de distance)

2 roues motorisées

2 Leds rouge

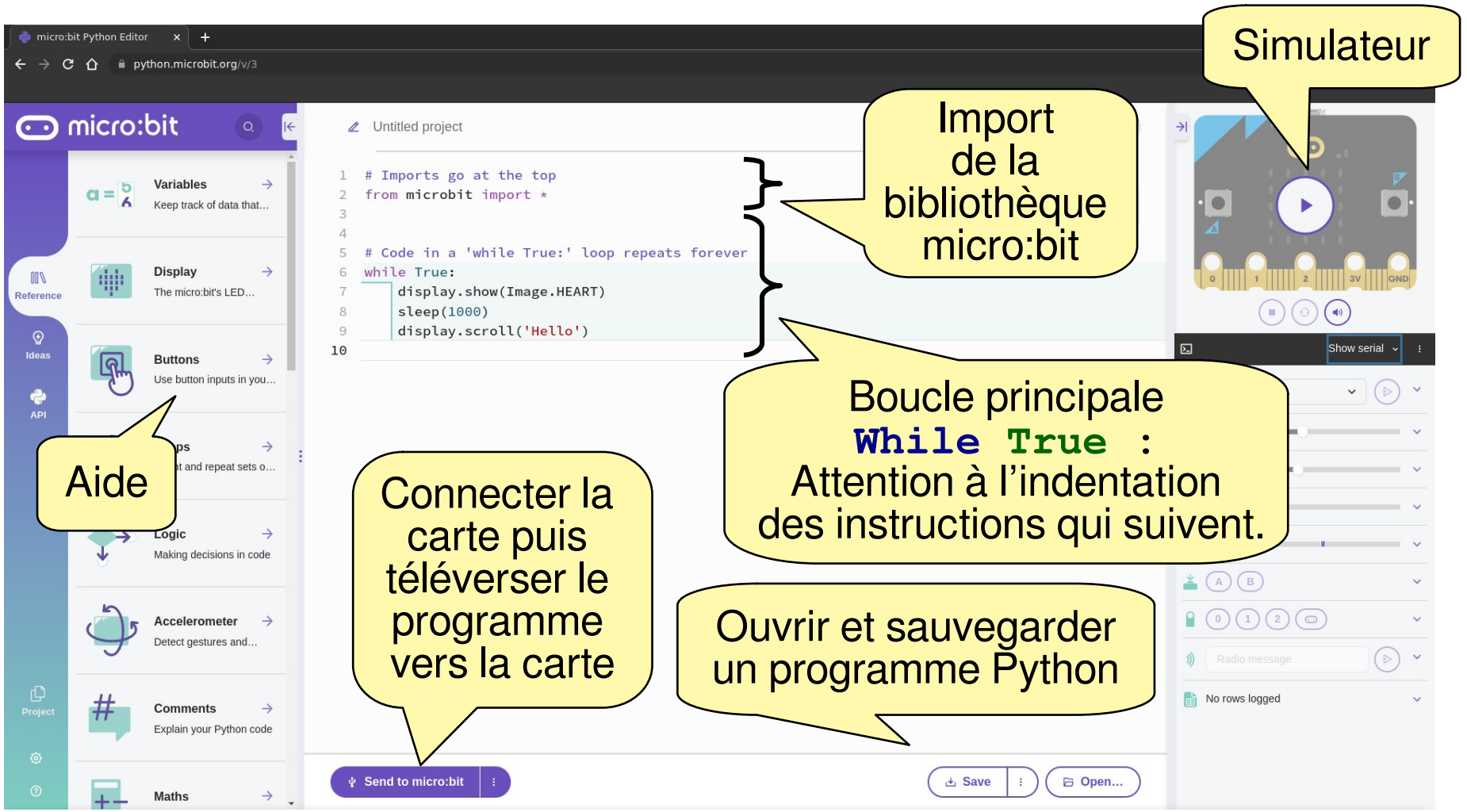
4 Leds RGB

Capteur infrarouge (pour une télécommande)

2 capteurs de suiveur de ligne

Éditer un programme Python pour les cartes micro:bit

L'édition du programme Python va se faire avec l'éditeur en ligne du site : https://python.microbit.org/ avec le navigateur Chrome.



Mission 1 : Mettre en place le jumeau réel Maqueen

Tout d'abords il faut mettre place le jumeau réel, c'est à dire il faut que le robot **Maqueen** soit à l'écoute des ordres émis par **Ropy** pour ensuite les exécuter.

Ropy

Dans le programme Python **rp_cmd.py,** écrire au tout début l'établissement de la communication série:

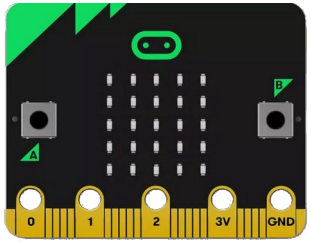
```
rp_jumeau('COM1',115200)
```

COM1 est le port de communication (voir page suivante)

115200 est la vitesse de communication

Liaison série (USB)

Carte micro:bit relais



Charger le programme **rp_maqueen-relay.py** (répertoire twins) dans la carte.

Liaison radio

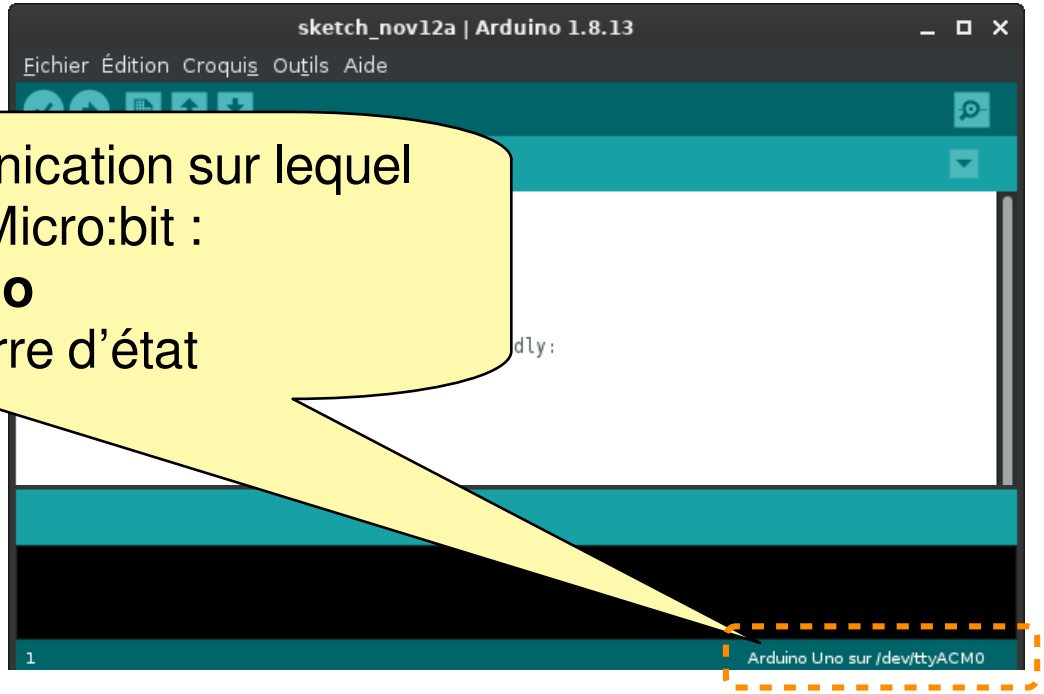
Carte micro:bit robot

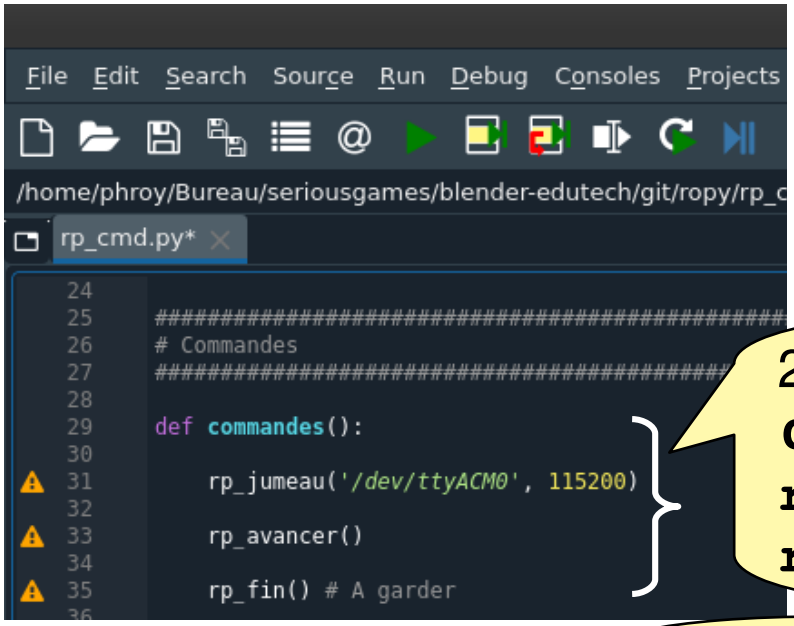
Charger le programme **rp_maqueen-robot.py** (répertoire twins) dans la carte.

Mission 1 : Mettre en place le jumeau réel
Programme rp_cmd.py

1 : **Identifier** le port de communication sur lequel est branché la carte Micro:bit :

- lancer le **programme Arduino**
- lire le **port** détecté dans la barre d'état





2 : Dans le programme **rp_cmd.py** (Spyder) **coder** le jumelage et un mouvement simple :

```
rp_jumeau('/dev/ttyACM0',115200)
rp_avancer()
```

3 : **Tester** le programme en observant si le robot **Maqueen** exécute bien le mouvement programmé.

Mission 2 : Pilotage manuel du robot Maqueen

En utilisant le **même protocole de communication lié au jumelage numérique**, on vous demande de piloter le robot à partir d'une carte micro:bit utilisée comme **télécommande**.

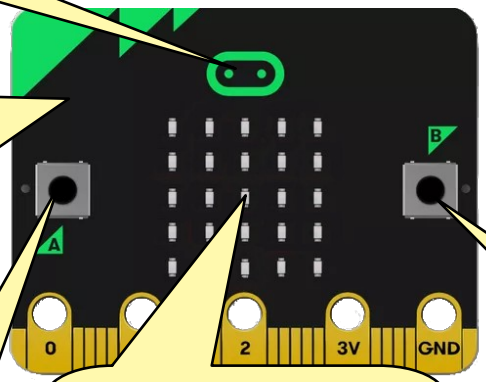
Carte micro:bit télécommande

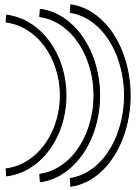
Liaison radio

Carte micro:bit robot

Marque la case (carte v2)

- **Avance** si la carte est penché en avant
- **Recule** si la carte est penchée en arrière





Garder le programme **rp_maqueen-robot.py** (répertoire twins) dans la carte.

Affiche le mouvement à exécuter

Tourne à droite 90°

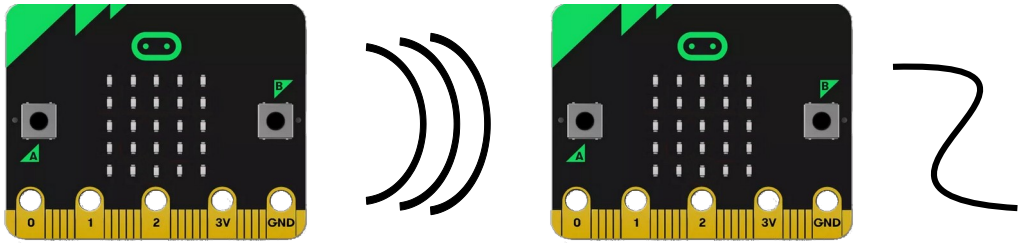
Tourne à gauche 90°

Créer le programme **rp_maqueen-tlcmd.py**

Mission 3 : Pilotage manuel du rover Ropy

En gardant votre programme de la télécommande (précédemment réalisé), on vous demande de piloter manuellement le rover **Ropy** à partir de celle-ci.

Carte micro:bit télécommande — **Liaison radio** — **Carte micro:bit relais** — **Liaison Série (USB)** — **Ropy**



Garder le programme **rp_maqueen-tlcmd.py**

Modifier le programme **rp_maqueen-relay.py** afin de placer le relais en écoute puis à transmettre les ordres reçus.

Éditer le programme Python **rp_cmd.py,** afin de placer **Ropy** en écoute puis à exécuter les ordres demandés par la télécommande.