

# Jumelage numérique de Ropy avec Maqueen



**LA FORGE**  
des communs  
numériques  
éducatifs



# Présentation de Ropy et de son environnement de programmation

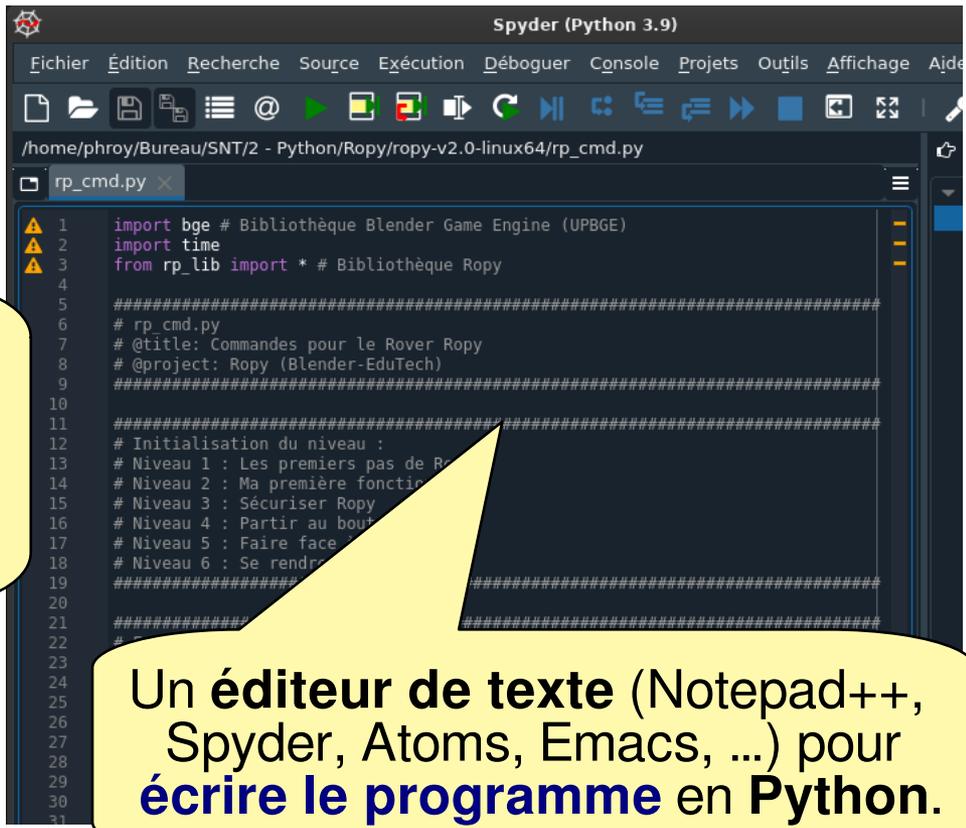
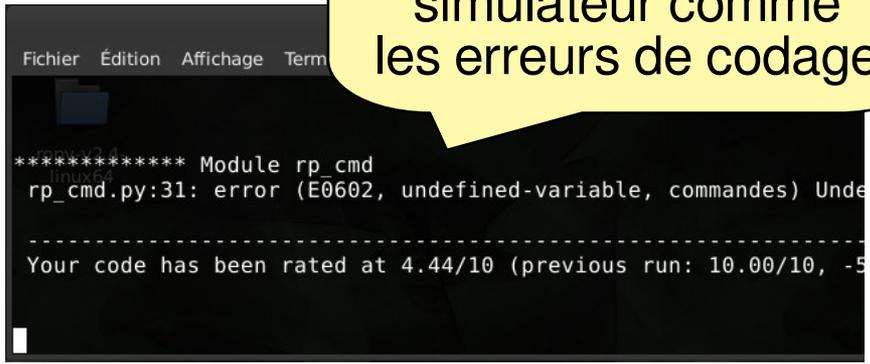


**Ropy** est un rover martien qui se commande grâce au langage **Python**. L'interface de programmation se décompose en **3 fenêtres** : un éditeur de texte, le simulateur et la console.



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

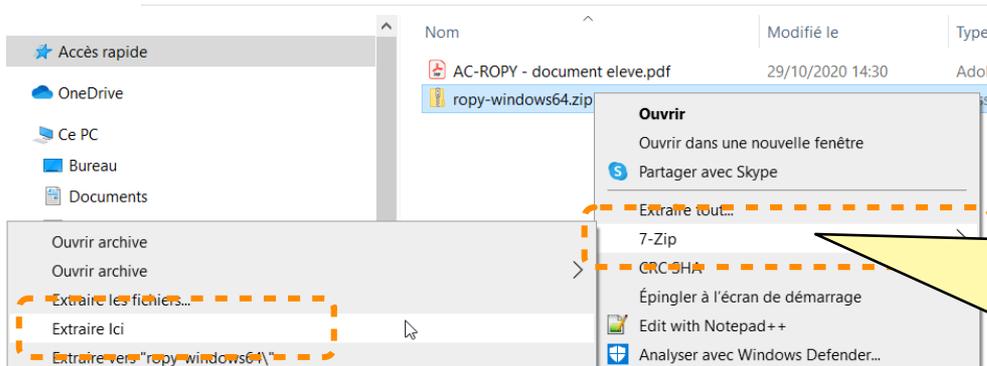
La **console** pour **visualiser les informations** du simulateur comme les erreurs de codage.



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

# Éditer le programme avec Spyder

## Ouvrir le fichier rp\_cmd.py

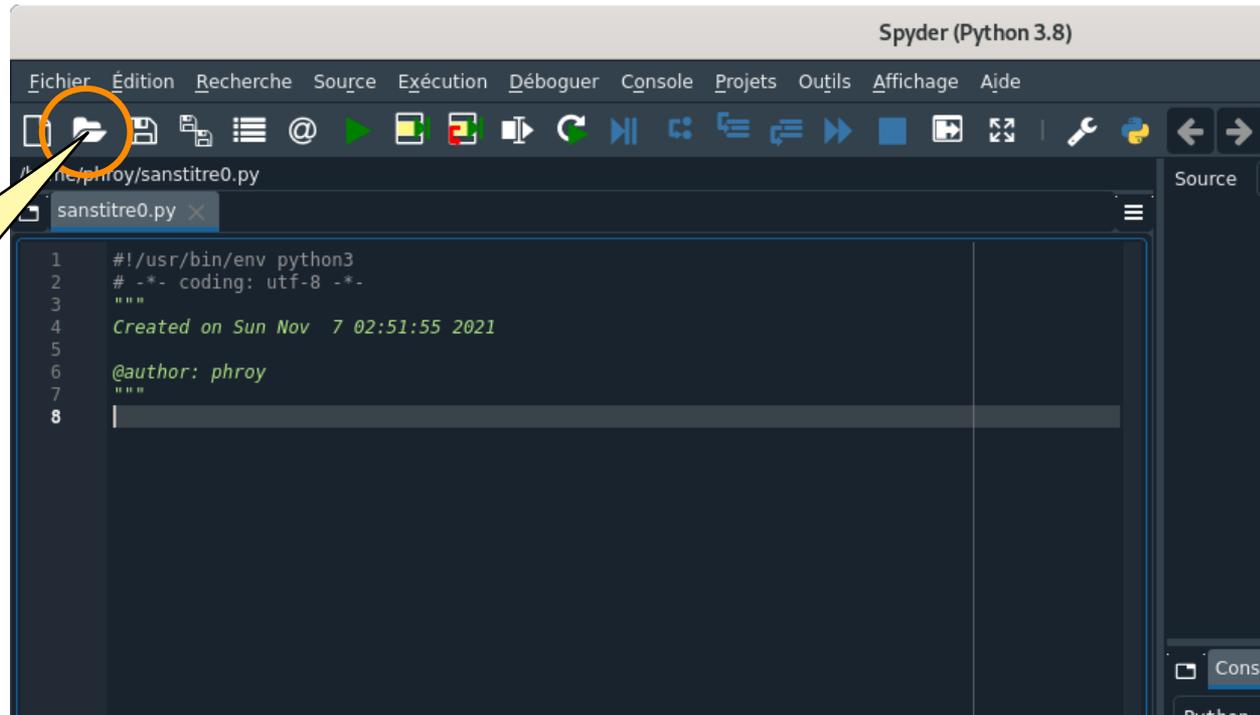


1 : Récupérer l'archive **ropy-windows64.zip** et la décompresser avec **7-Zip** dans votre répertoire. L'extraction va créer le répertoire rpy

2 : Lancer le Logiciel **Spyder**.



3: Ouvrir le fichier Python à éditer **rp\_cmd.py** (Ropy commandes) présent dans le répertoire rpy.



# Éditer le programme avec Spyder



## Exécution du programme

5 : **Sauvegarder** le fichier

**Attention !**

Toujours sauvegarder le fichier avant son exécution avec le simulateur.

Le **simulateur** et la **console** se lancent en même temps avec le programme **ropy.bat**

**Arrêter et réinitialiser**

**Afficher l'aide**

**Niveau actuel**

6 : **Exécuter** le programme

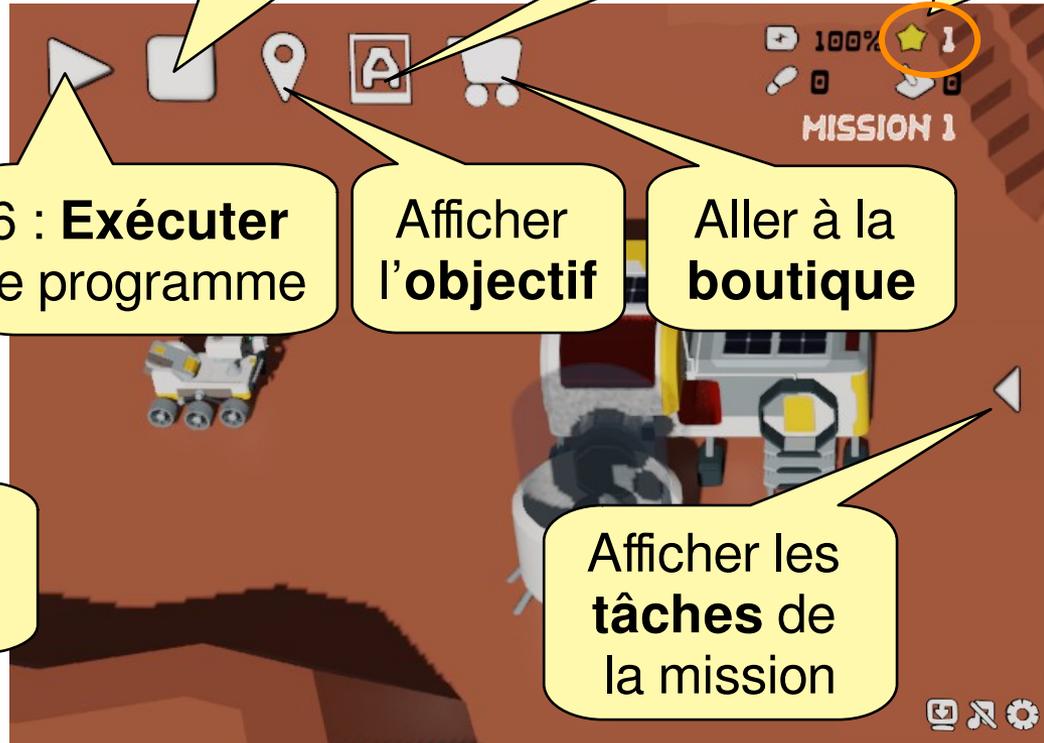
**Afficher l'objectif**

**Aller à la boutique**

4 : **Écrire** le code Python

**Afficher les tâches** de la mission

```
Fichier  Édition  Recherche  Source  Exécution  Déboguer  Console
/home/phroy/Bu
rp_cmd.py x
import tim
from rp_li
#####
# rp_cmd.p
# @title: C
# @project: Ropy
#####
#####
# 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():
31 rp_gauche()
32 rp_avancer()
33 rp_avancer()
34 rp_avancer()
35 rp_avancer()
36
37
38 rp_fin() # A garder
39
```



# 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

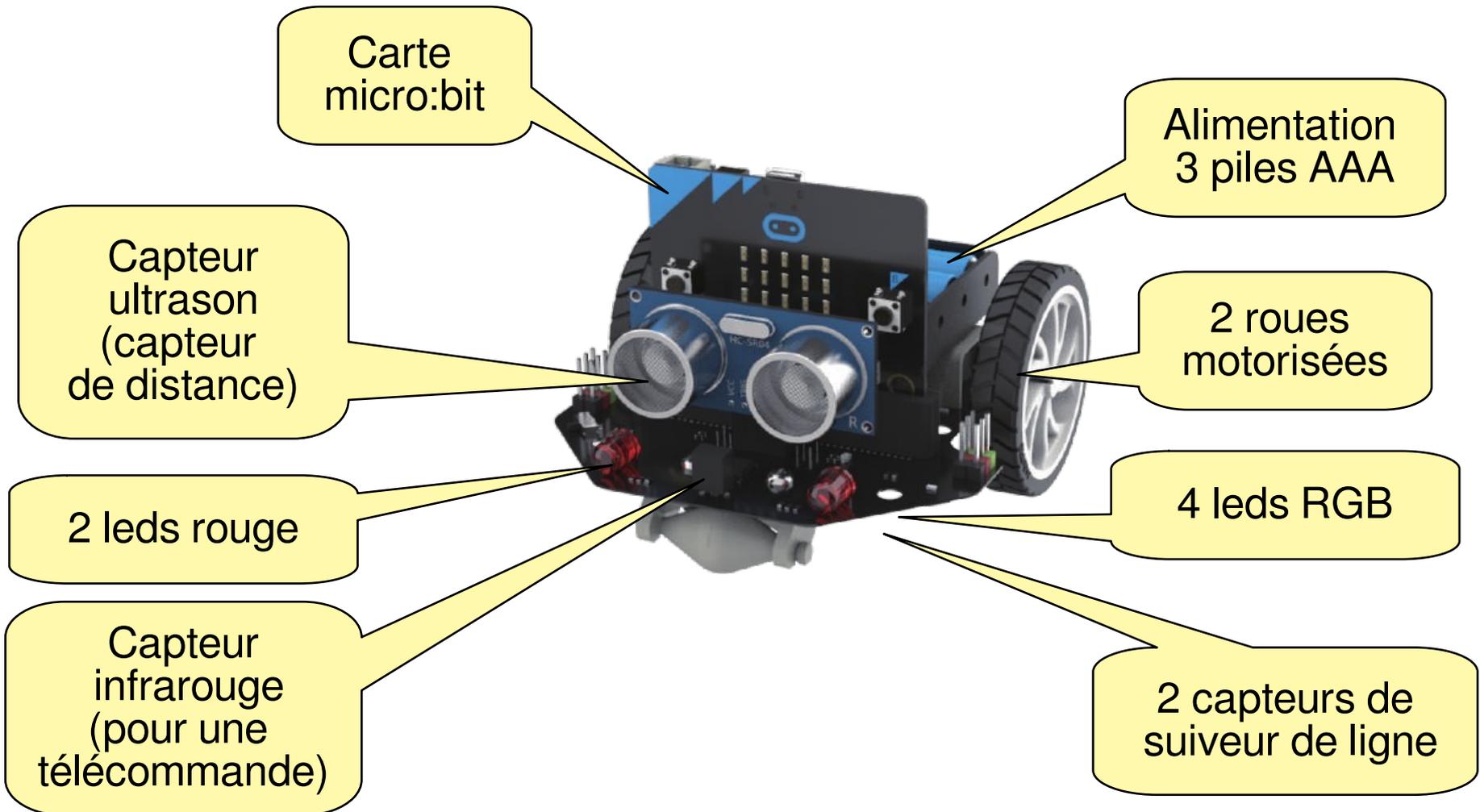
} **La commande `rp_fin()` est à conserver.**

} **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**.



# 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.

The screenshot shows the micro:bit Python Editor interface. On the left is a sidebar with various tool categories: Variables, Display, Buttons, Logic, Accelerometer, Comments, and Maths. The main area displays a Python code editor with the following code:

```
1 # Imports go at the top
2 from microbit import *
3
4
5 # Code in a 'while True:' loop repeats forever
6 while True:
7     display.show(Image.HEART)
8     sleep(1000)
9     display.scroll('Hello!')
10
```

Annotations in yellow speech bubbles provide instructions:

- Aide**: Points to the sidebar.
- Connecter la carte puis téléverser le programme vers la carte**: Points to the "Send to micro:bit" button at the bottom.
- Import de la bibliothèque micro:bit**: Points to the `from microbit import *` line.
- Boucle principale `While True` : Attention à l'indentation des instructions qui suivent.**: Points to the `while True:` loop and its indented contents.
- Ouvrir et sauvegarder un programme Python**: Points to the "Save" and "Open..." buttons at the bottom.
- Simulateur**: Points to the virtual micro:bit device on the right side of the interface.

# Mission 1 : Mettre en place le jumeau réel Maqueen



Tout d'abords il faut mettre place le jumeau réel, c'est à dire qu'il faut que le robot **Maqueen** soit à l'écoute des ordres émis par **Ropy**. Puis réaliser l'objectif de la mission 1 standard.

## Ropy

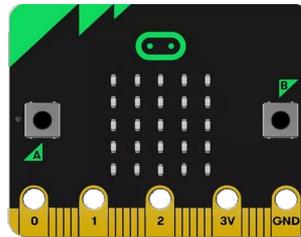


Dans le programme Python `rp_cmd.py`,  
**écrire** au tout début du programme le jumelage :  
`rp_jumeau ()`

Initialisation de la communication  
série (câble USB)

## 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 2 : Pilotage manuel du robot Maqueen



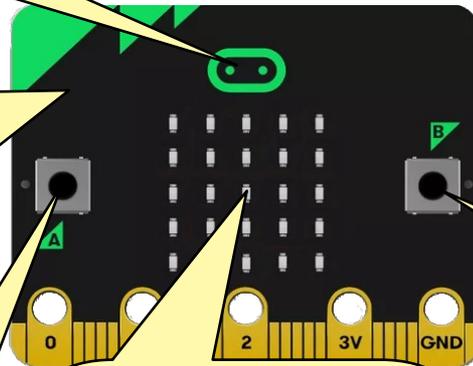
En utilisant le **même protocole de communication lié au jumelage numérique**, on vous demande de piloter le robot à partir d'une carte micro:bit utilisée comme **télécommande**.

**Arrêter** la communication (carte v2)

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

**Tourne** à gauche 90°

**Carte micro:bit télécommande**



**Affiche** le mouvement à exécuter

**Créer** le programme `rp_maqueen-rlcmd.py`

**Liaison radio**



**Tourne** à droite 90°

**Carte micro:bit robot**



**Garder** le programme `rp_maqueen-robot.py`

Pour les cartes v1, il faut arrêter la communication par l'appui sur les deux boutons.

# 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.

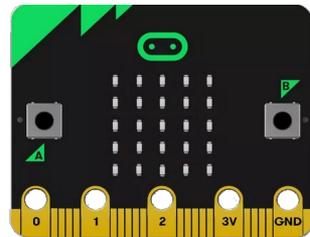
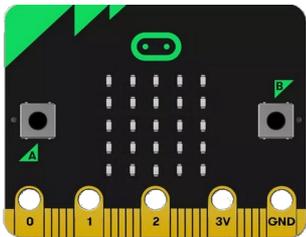
Carte  
micro:bit  
télécommande

Liaison  
radio

Carte  
micro:bit  
relais

Liaison  
Série  
(USB)

Ropy



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

**Garder** le programme  
`rp_maqueen-relay.py`

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