SNT

LYCÉE LÉONARD DE VINCI

Jumelage numérique de Ropy avec Maqueen

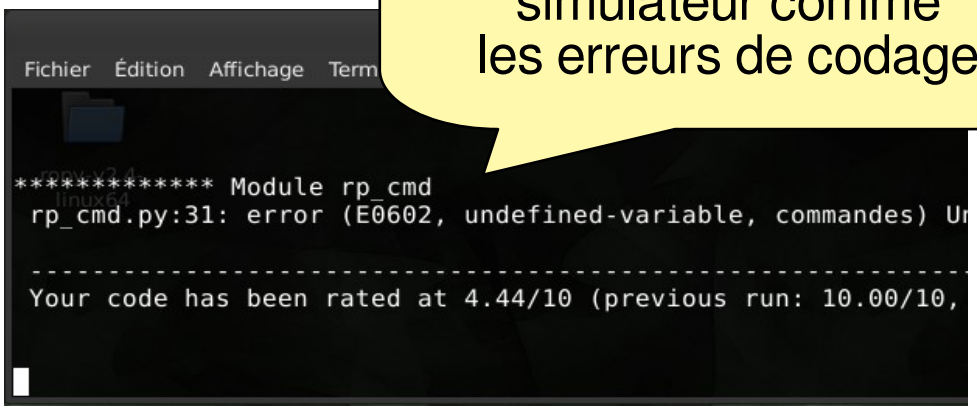
LA FORGE
des communs numériques éducatifs

Philippe Roy <philippe.roy@ac-grenoble.fr>

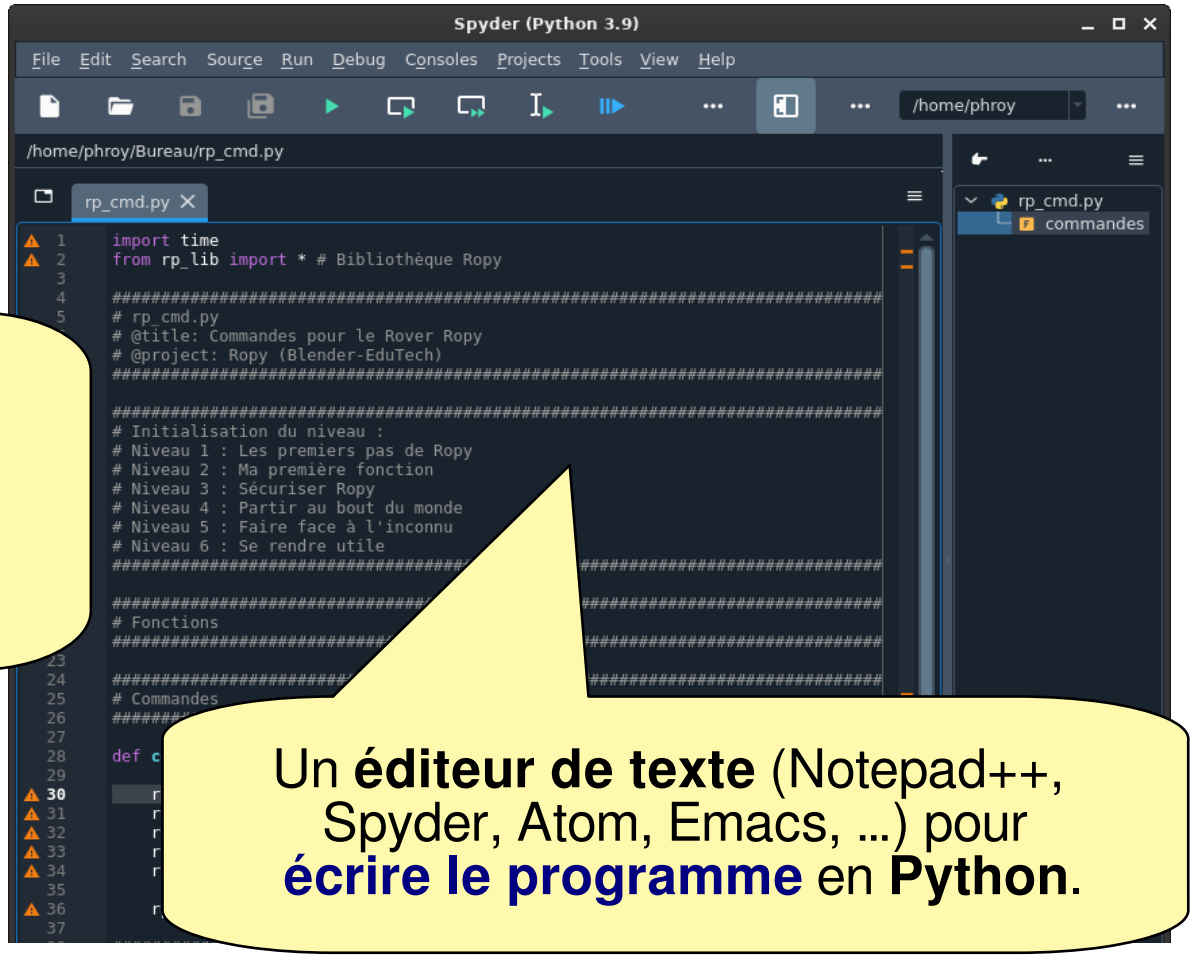
Présentation de Ropy et de son environnement de programmation

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**.

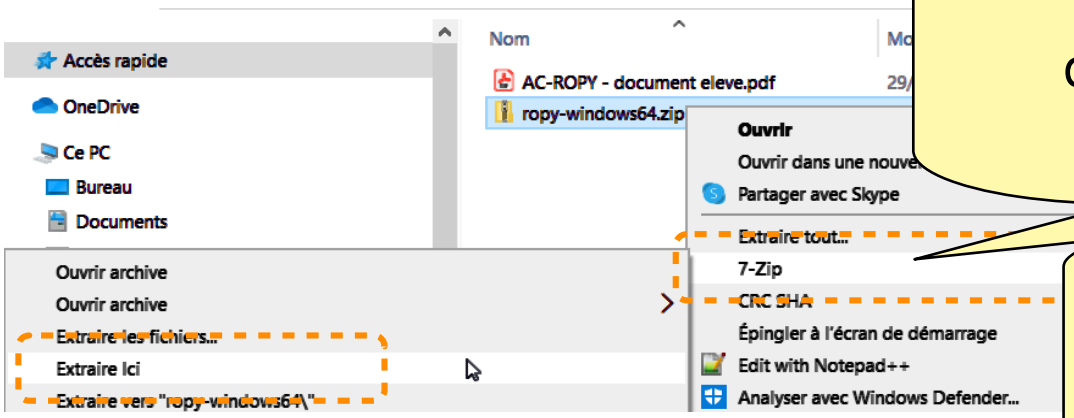


La **console** pour **visualiser les informations** du simulateur comme les erreurs de codage.



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

Mettre en place l'environnement de développement

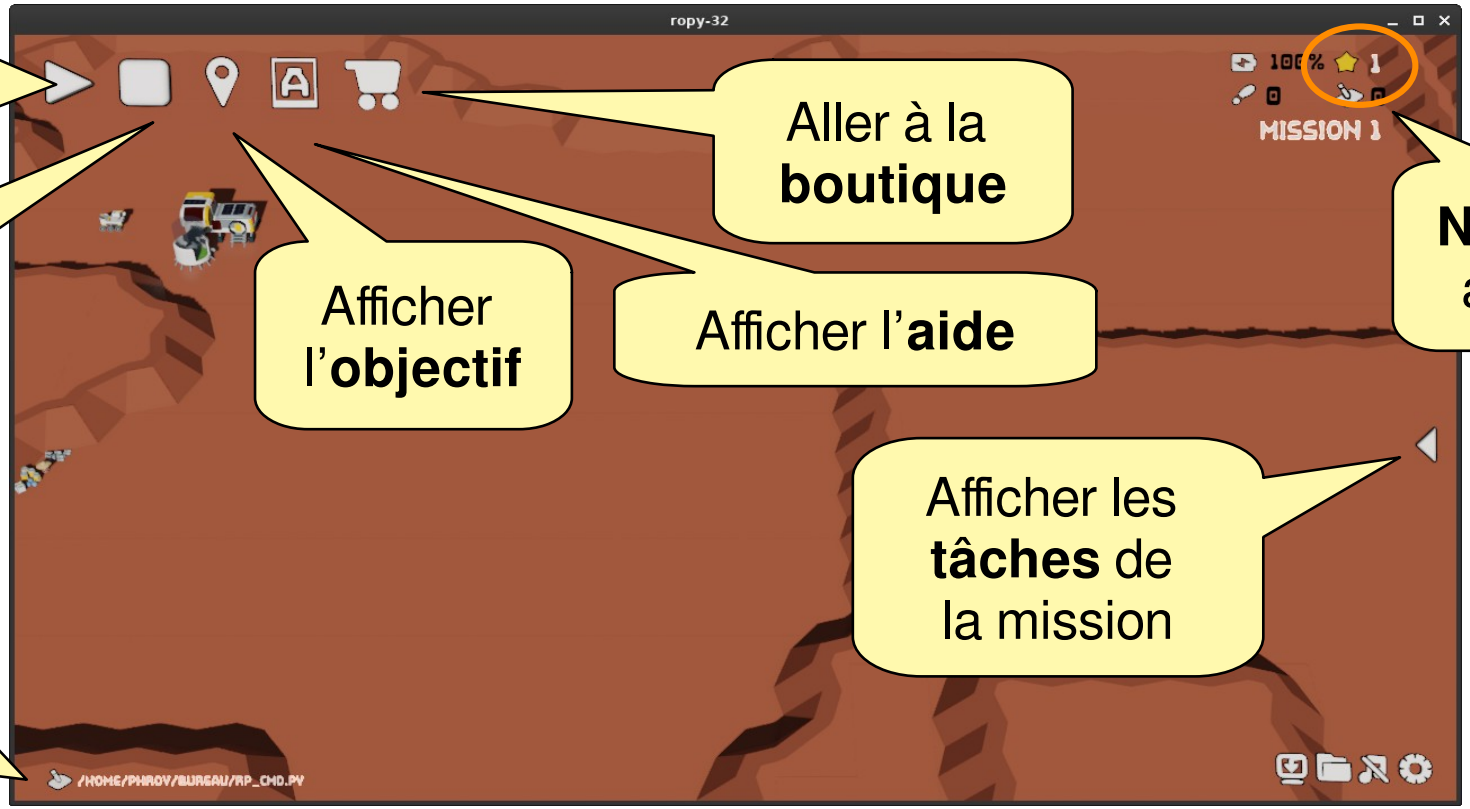


1 : Récupérer l'archive **ropy-windows64.zip** et la décompresser avec **7-Zip** sur le **bureau**. L'extraction va créer le répertoire ropy

Le **simulateur** et la **console** se lancent en même temps avec **ropy.bat** (situé dans le répertoire créé).

Exécuter le programme

Arrêter et **réinitialiser**



Aller à la **boutique**

Afficher l'**objectif**

Afficher l'**aide**

Niveau actuel

Afficher les **tâches** de la mission

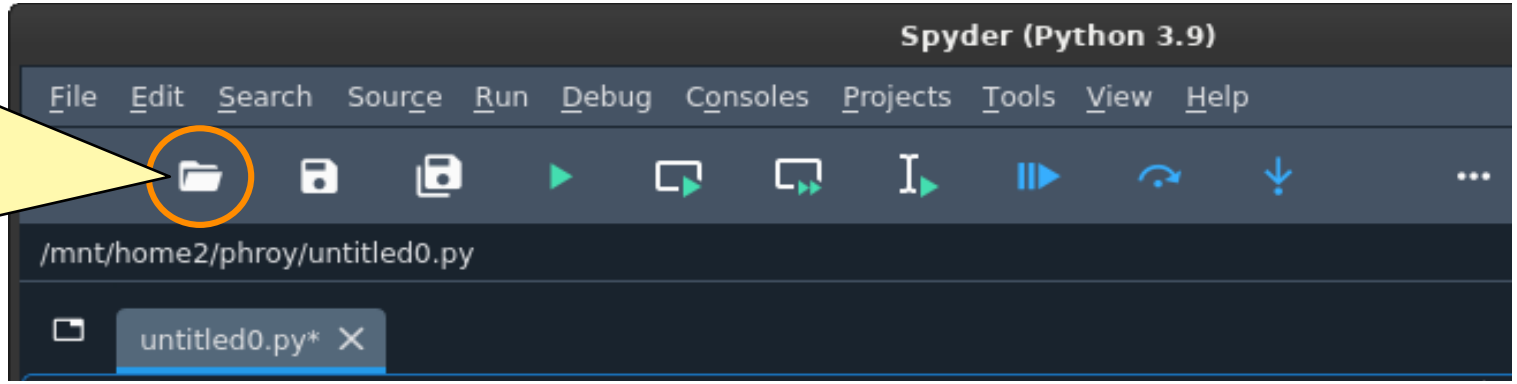
Fichier de commandes

Mettre en place l'environnement de développement

2 : Copier **dans votre répertoire** le fichier de commandes : **ropy_cmd.py** (ropy commandes).

3 : Lancer **Spyder**.

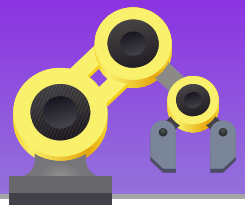
4 : Dans **Spyder** **ouvrir** le fichier de commandes qui a été précédemment copié dans votre répertoire.



5 : Dans le **simulateur**, **définir** votre fichier comme fichier de commandes.

6 : Le nom de votre fichier doit apparaître ici.

Mettre en place l'environnement de développement



8 : **Sauvegarder** le fichier
Attention !
Toujours sauvegarder le fichier avant son exécution.

File Edit Search Source Run

/home/phroy/Bureau/rp_cmd.py

rp_cmd.py

```
7   # @project: Ropy (Blender-EduTech)
8   ##################################################
9
10  ##################################################
11  # Initialisation du niveau :
12  # Niveau 1 : Les premiers pas de Ropy
13  # Niveau 2 : Ma première fonction
14  # Niveau 3 : Sécuriser Ropy
15  # Niveau 4 : Partir au bout du monde
16  # Niveau 5 : Faire face à l'inconnu
17  # Niveau 6 : Se
18  ##################
19
20  ##################
21  # Fonctions
22  ##################
23
24  ##################################################
25  # Commandes
26  ##################################################
27
28  def commandes():
29
30      rp_gauche()
31      rp_avancerr()
32      rp_avancer()
33      rp_avancer()
34      rp_avancer()
35
36      rp_fin() # A garder
37
38  ##################################
```

conda: base (Python 3.9.13)

7 : **Écrire** le code Python.

9 : **Exécuter** le programme.

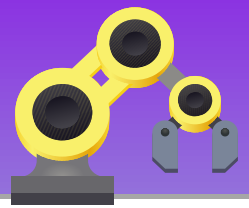
10 : Si votre code a **une erreur**, la console indique où elle se trouve. Ici la ligne 31 contient une commande indéfinie : `'rp_avancerr'`.

Fichier Édition Affichage

```
************* Module rp_cmd
rp_cmd.py:31: error (E0602, undefined-variable, commandes) Undefined variable 'rp_avancerr'

-------------------------------------------------------------------
Your code has been rated at 4.44/10 (previous run: 10.00/10, -5.56)
```

Contenu du fichier rp_cmd.py



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

```
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()


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

def cycle():
    commandes()
    rp_fin()

if __name__=='start':
    thread_cmd_start(cycle)
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**
C'est votre outillage, à garder à travers les missions !

Commandes : section pour le codage des commandes du robot

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

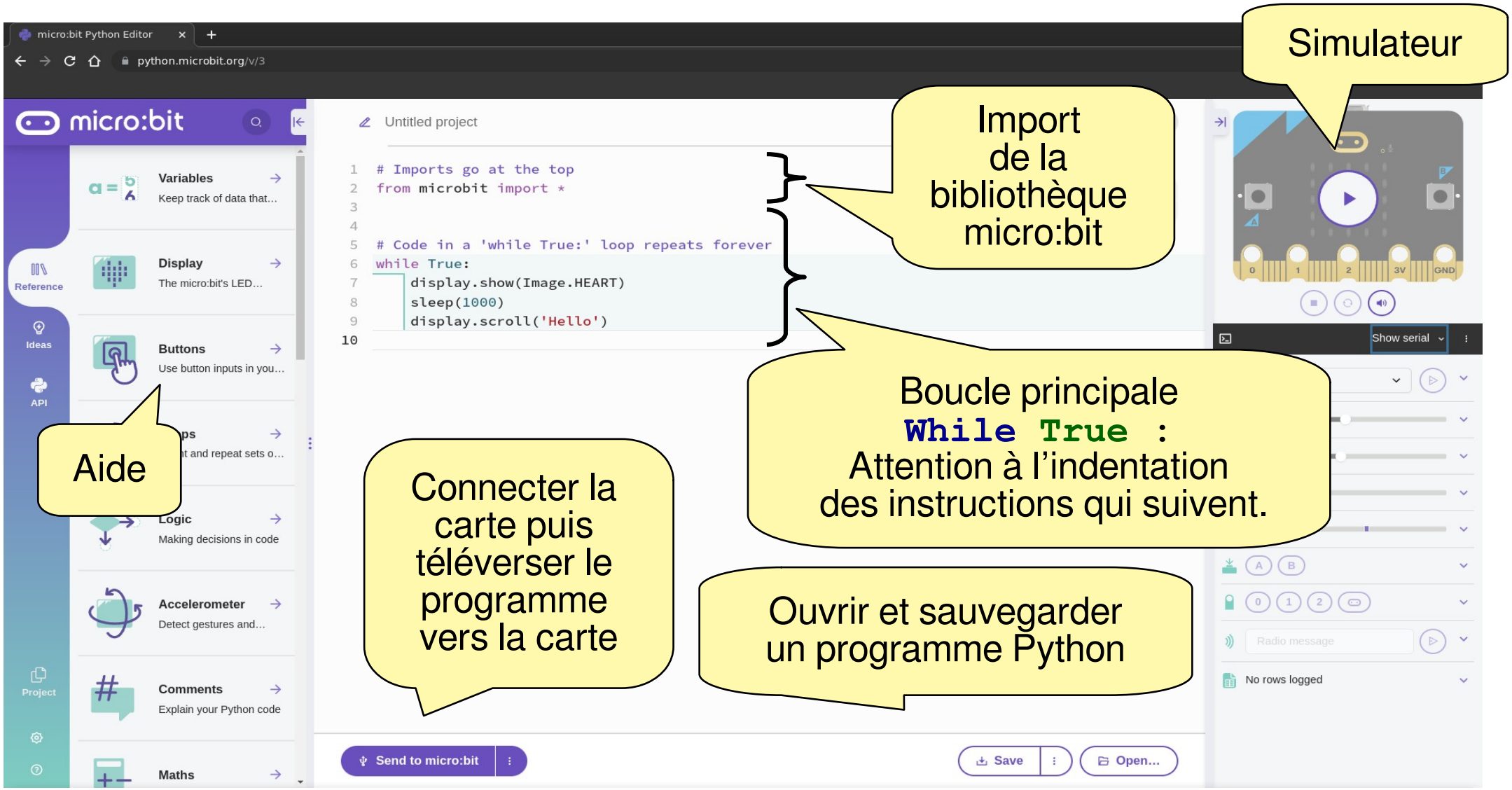
4 leds RGB

Capteur infrarouge (pour une télécommande)

2 capteurs de suiveur de ligne

Éditer un programme Python pour les cartes micro:bit

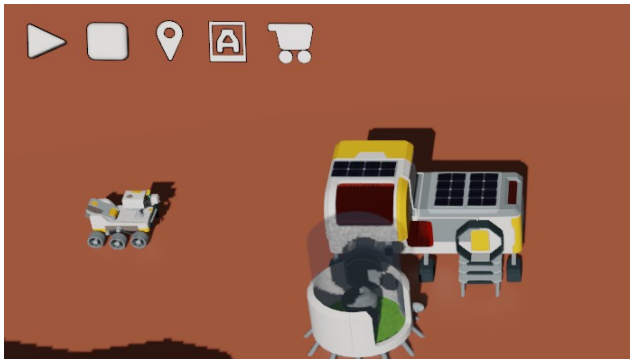
L'édition du programme Python va se faire avec l'éditeur en ligne du site : https://python.microbit.org/ avec le navigateur Chrome.



Mission 1 : Mettre en place le jumeau réel Maqueen

Tout d'abords il faut mettre place le jumeau réel, c'est à dire 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

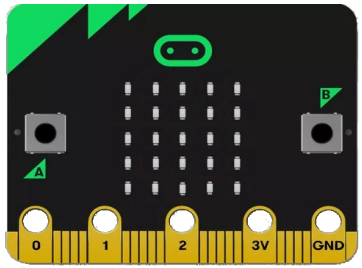


Dans le programme Python **rp_cmd.py,** **écrire** au tout début du programme le jumelage : `rp_jumeau()`

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**.

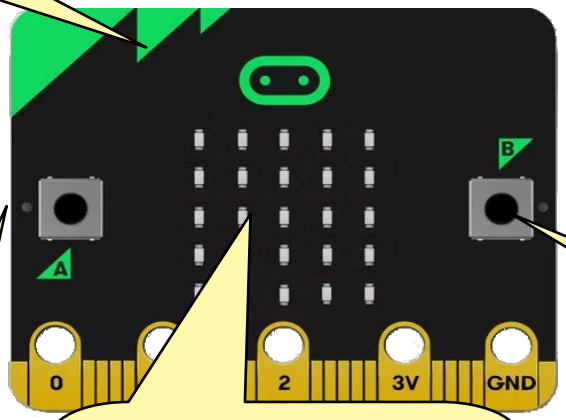
Carte micro:bit télécommande

Liaison radio

Carte micro:bit robot

Arrêter la communication (carte v2)

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



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

Tourne à droite 90°

Affiche le mouvement à exécuter

Tourne à gauche 90°

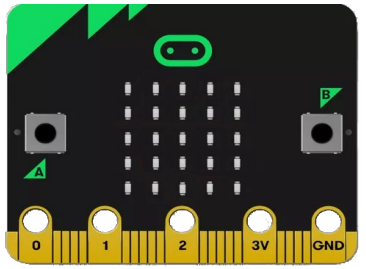
Créer le programme **rp_maqueen-tlcmd.py**

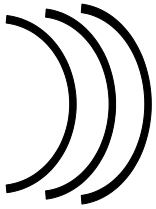
Pour les cartes v1, il faut arrêter la communication par l'appui sur les deux boutons.

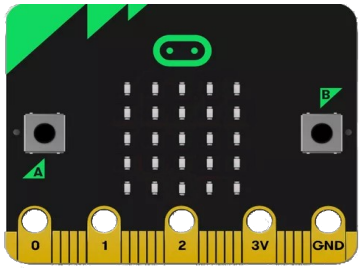
Mission 3 : Pilotage manuel du rover Ropy

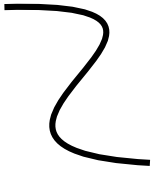
En gardant votre programme de la télécommande (précédemment réalisé), on vous demande de piloter manuellement le rover **Ropy** à partir de celle-ci.

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.