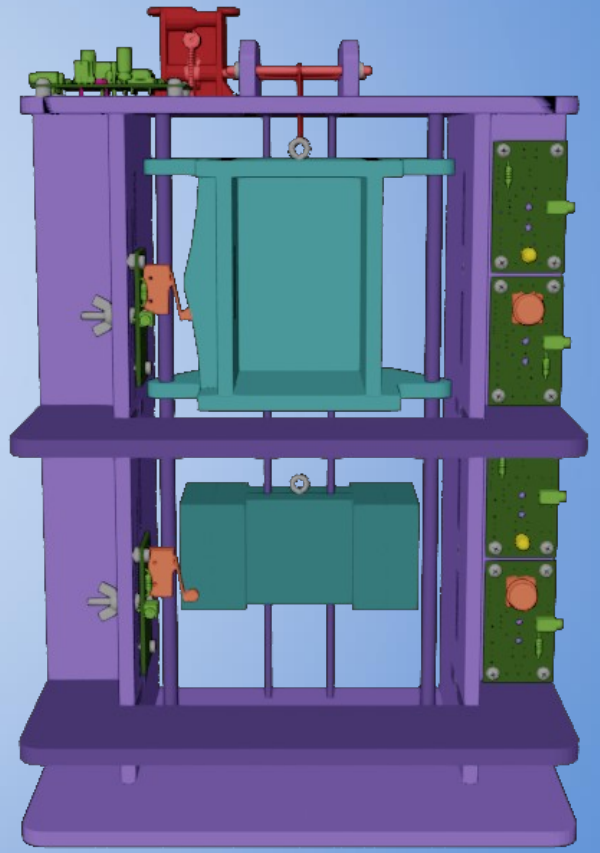
Séquence 3

Algorithme et programmation

SI

LYCÉE LÉONARD DE VINCI

Document Technique
Jumeau numérique d'un monte-charge

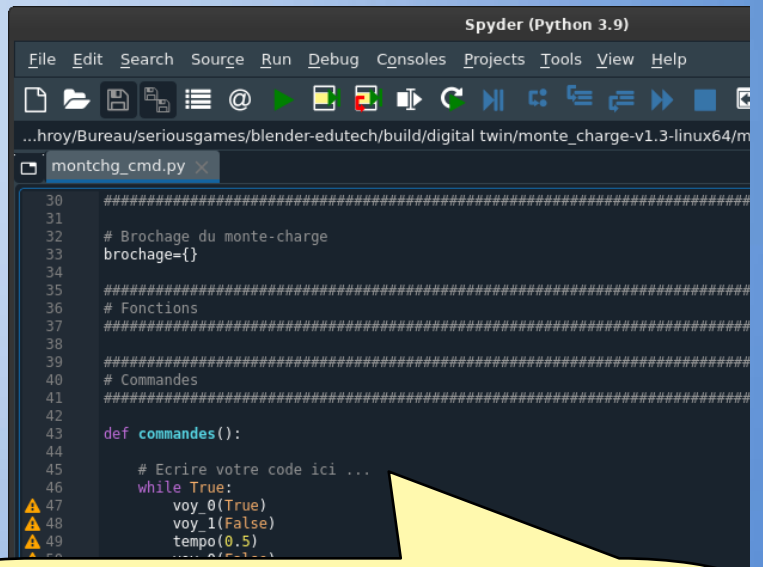


https://gitlab.com/blender-edutech/digital_twin

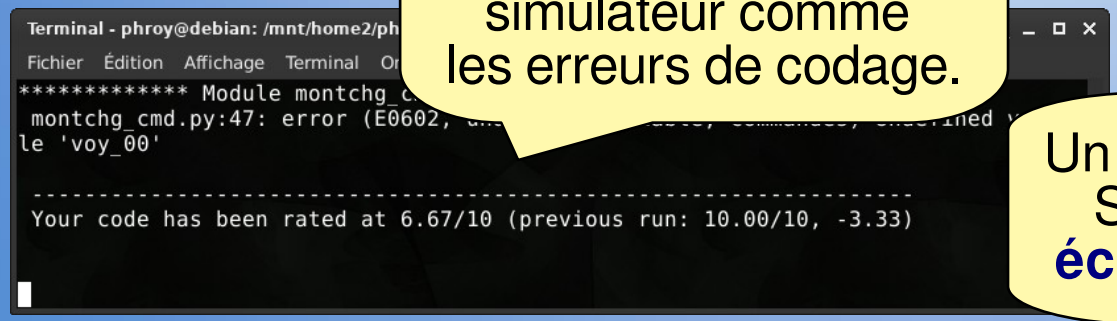
Présentation du jumeau numérique et de son environnement de programmation

Le jumeau numérique est une maquette numérique 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 système.**



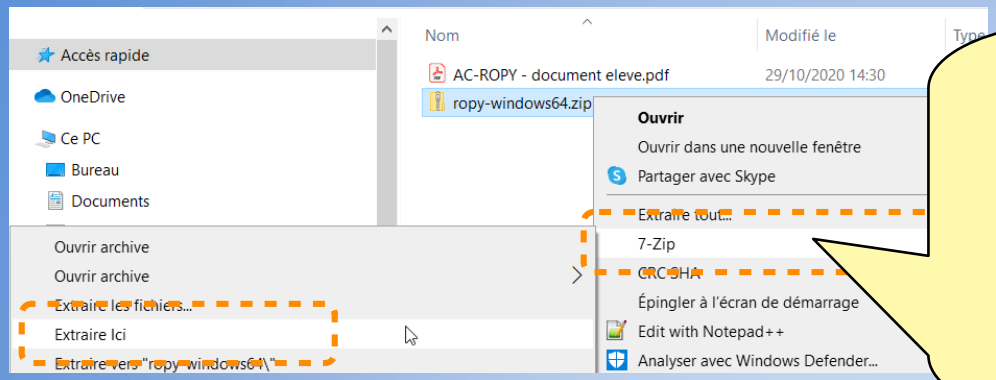
La **console** pour **visualiser les informations** du simulateur comme les erreurs de codage.



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

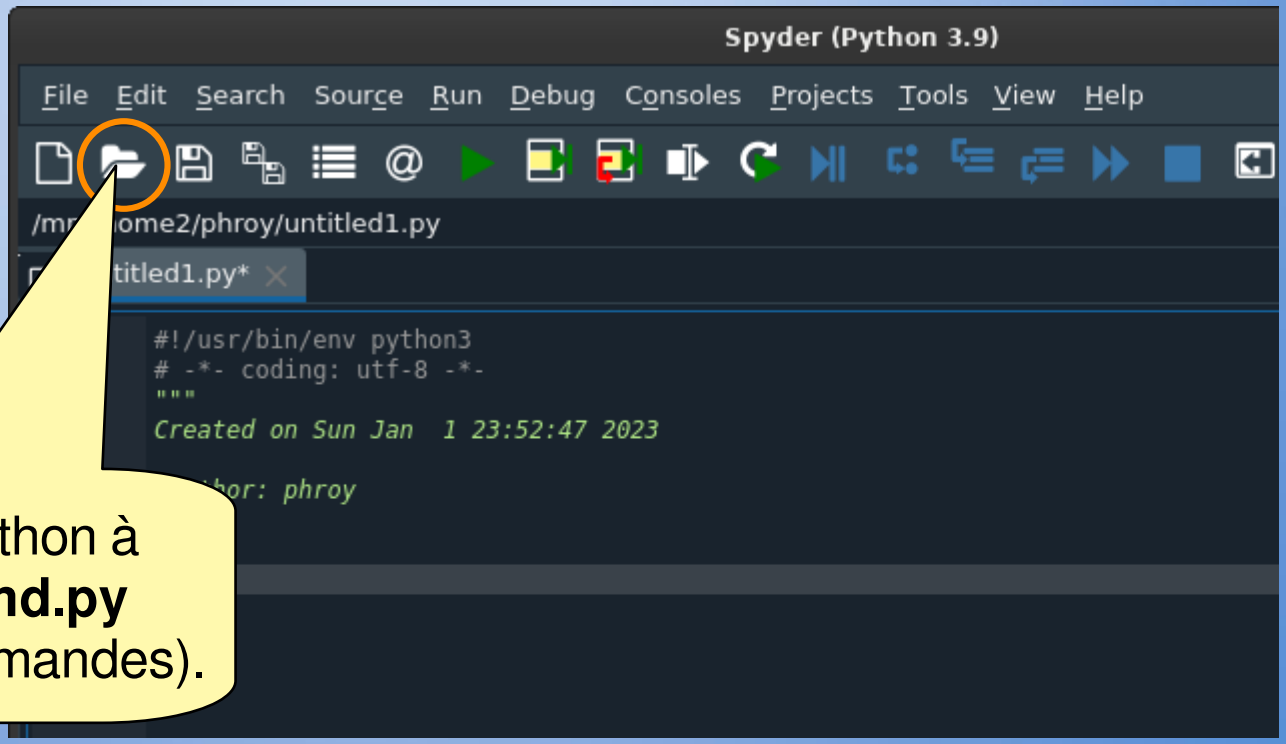
Éditer le programme avec Spyder

Ouvrir le fichier montchg_cmd.py



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

2 : Lancer le logiciel **Spyder**.



3: Ouvrir le fichier Python à éditer **montchg_cmd.py** (Portail coulissant commandes).

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 **monte_charge.bat**

Arrêter et **réinitialiser**

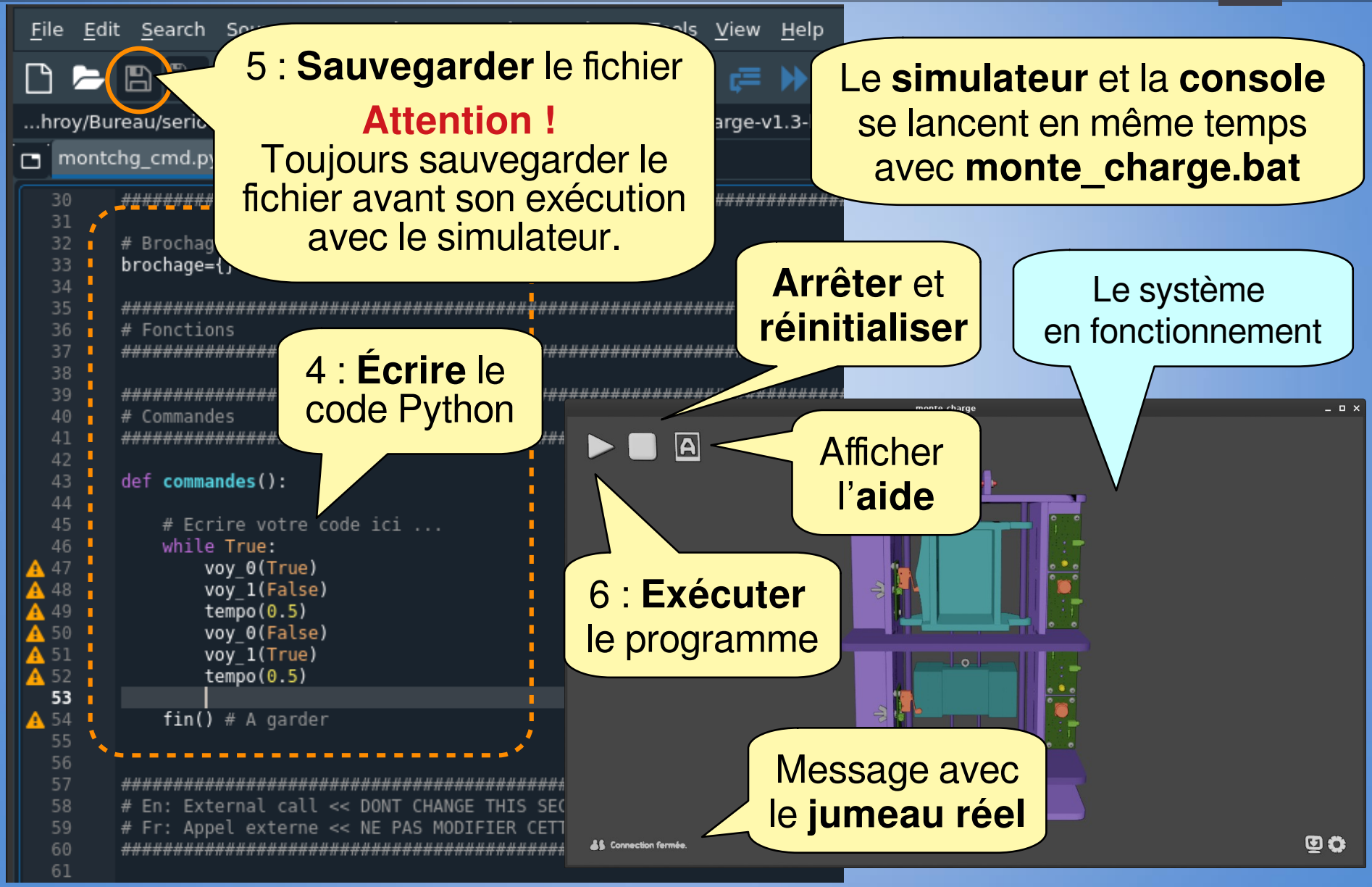
Le système en fonctionnement

4 : **Écrire** le code Python

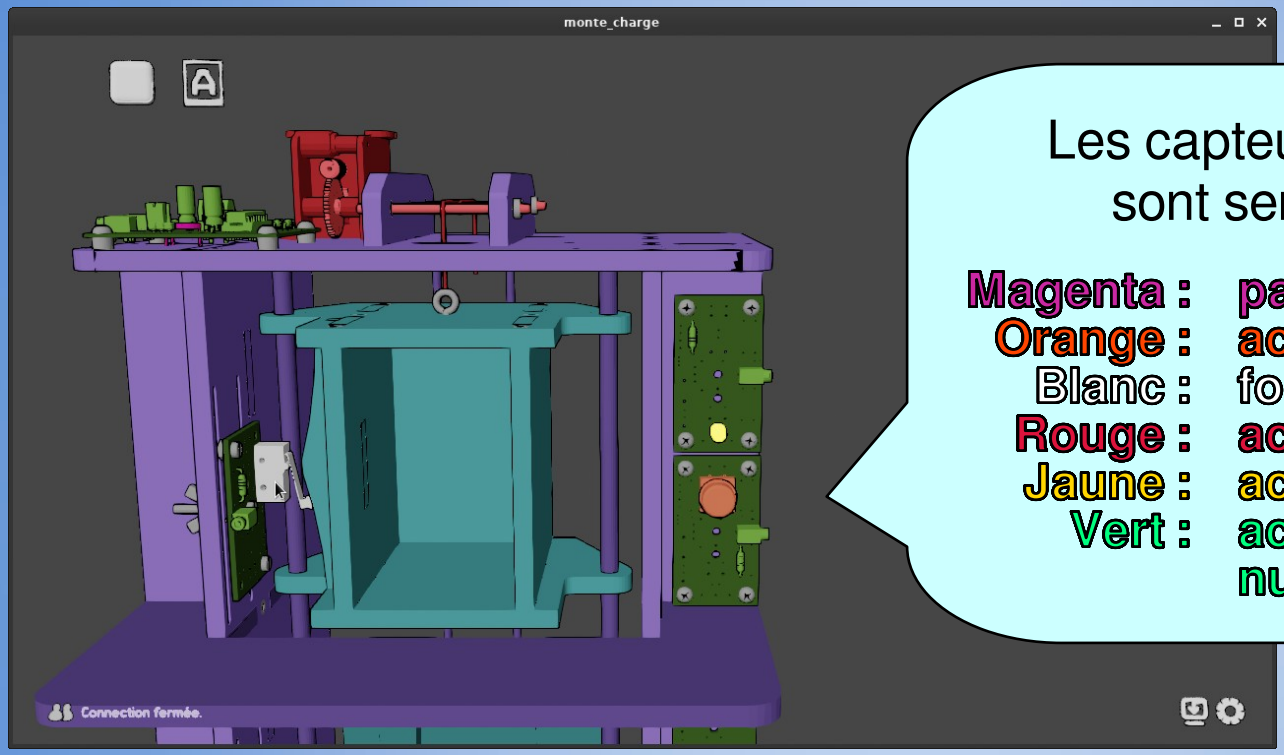
Afficher l'**aide**

6 : **Exécuter** le programme

Message avec le **jumeau réel**



Manipulation de la maquette numérique



Les capteurs et les boutons sont sensibles au clic :

Magenta :	**passif**
Orange :	**actif (activable)**
Blanc :	focus souris
Rouge :	**activé physiquement**
Jaune :	**activé numériquement**
Vert :	**activé physiquement et numériquement**

Le **bouton du centre** sert à **manipuler** le modèle 3D :

- **Clic centre** : rotation du mécanisme (orbit)
- **Clic centre + Maj** : déplacement du mécanisme (pan)
- **Clic centre + Ctrl** : zoom
- **Molette** : zoom

Jumelage et brochage

Le jumelage est basé sur le **protocole Firmata**. Il faut téléverser le programme **StandardFirmata** (IDE Arduino) vers la carte Arduino afin

- qu'elle transmette les ordres de l'ordinateur vers les actionneurs,
- qu'elle remonte les compte-rendus des capteurs vers l'ordinateur.

Pour le script **Python** le jumelage est activé par la commande `jumeau(brochage)` . `brochage` est un dictionnaire faisant le lien entre les composants numériques (objet 3D) et les composants réels :

```
brochage={'composant_num':['type',broche,'mode']}.
```

- **a** pour analogique
- **d** pour binaire (digital)

numéro de la broche (0 à 13)

- **i** pour input (entrée)
- **o** pour output (sortie)
- **p** pour pwm (sortie variable)

Par exemple :

```
brochage={'bouton':['d',2,'i'],'led':['d',3,'o']}.
```

La broche associée au composant 3D **bouton** est la **2** en mode **entrée**

La broche associée au composant 3D `led` est la **3** en mode **sortie**

Carte de référence du monte-charge

Actionneur :
- Moteur : monter la cabine : `mot_m(ordre)`
- Moteur : descendre la cabine : `mot_d(ordre)`

Capteurs de fin de course :
- Capteur de présence cabine niveau 0 : `pc_0()`
- Capteur de présence cabine niveau 1 : `pc_1()`

Pupitre :
- Bouton poussoir d'appel niveau 0 : `ba_0()`
- Bouton poussoir d'appel niveau 1 : `ba_1()`
- Voyant d'appel niveau 0 : `voy_0()`
- Voyant d'appel niveau 1 : `voy_1()`

Brochage (composants numériques) **:**
`'ba_0'`, `'ba_1'`, `'pc_0'`, `'pc_1'`, `'mot_m'`, `'mot_d'`, `'voy_0'` et `'voy_1'`.

Valeur retournée par les capteurs et les boutons
- `True` : actif
- `False` : inactif

Ordre pour les actionneurs
- `True` : activer
- `False` : désactiver