TP01 – Introduction aux classes et objets

Ce TP vise à vous permettre de créer et de manipuler des classes et des objets en Python. En plus d'utiliser VS Code, vous travaillerez avec CoppeliaSim, qui vous permet d'afficher des objets (formes géométriques, robots) sur une interface pour observer les effets de vos modifications sur les propriétés de ces objets.

Remarques :

- Installez/Utilisez la version 4.5.1 EDU de CoppeliaSim depuis (https://www.coppeliarobotics.com/downloads)
- Attention : dans CoppeliaSim, avant de redémarrer les simulations (cad de lancer un programme python), vous devrez créer de nouvelles scènes (File -> New scene). Vous pouvez également supprimer d'anciennes scènes (File -> Close scene).

Exercice 1 – Définition d'une classe Element

1 – Préparation avant la séance de TP

1. Définissez sur papier une classe "Elément" avec les attributs suivants :

   `nom` : nom unique
   `type` : type d'élément (1 = plane ; 2 = disc ; 3 = cuboid ; 4 = spheroid ; 5 = cylinder)
   `dimension` taille de l'élément, une liste python différentes selon le type
   `couleur` – couleur de l'élément, une liste python de 3 valeurs (taux de RVB, entre 0 et 255 pour chaque valeur)
   `position` - position dans l'espace, une liste python de 3 valeurs (x,y,z)

   Le constructeur reçoit uniquement le nom, le type et la taille, et initialise couleur et position comme des listes vides.

2 – Réalisation en séance

2. Téléchargez le fichier zip "TP1_info.zip" depuis moodle (cours POO) et extrayez-le. Ouvrez VS Code et accédez au répertoire que vous venez d'extraire (File->Open folder). Nous travaillerons avec le fichier "TP1_coppelia.py". Enfin, lancez CoppeliaSim.

   Dans le terminal de VS code installer zmq et cbor qui permettent au script python de communiquer avec CoppeliaSim :

   ```
   pip install pyzmq
   pip install cbor
   ```

3. Implémentez dans TP1_coppelia.py la classe définie dans la question 1, et créer une méthode `afficher_console` qui affiche le nom, la position et la couleur de l'élément sur la console

4. Ecrire dans le programme principal une instance de la classe `Element`, de type boîte (cuboid) avec les dimensions [3,2,1] (pour ce type cette liste consiste en [longueur, largeur, hauteur]). Puis, changez sa couleur en rouge [250,0,0] et positionnez la boite à xyz = [2,0,0.5] en modifiant les attributs `couleur` et `position` Appelez la méthode afficher_console. Testez et faites valider.

5. Nous allons maintenant visualiser la scène dans CoppeliaSim. Pour cela il faut ajouter des méthodes et attributs à la classe Element. Ne modifiez pas la scène depuis CoppeliaSim ; tout doit être fait en Python :

   Il faut d'abord ajouter un attribut `handle` que l'on initialise dans le constructeur de la classe Element avec :

   ```
   self.handle = send_object_coppelia(nom,type,size)
   ```

   Ensuite pour les modifications de position et de couleur, il vous est demandé de créer 2 méthodes `set_position` et `set_couleur`. Ces 2 méthodes doivent mettre à jour les attributs `position` et `couleur` de la classe `Element`, puis faire appel aux méthodes ci-dessous (pour que Coppelia se mette à jour)

   Changer la position d'un objet dans la scène :

   ```
   set_object_pos_coppelia(self,position)
   ```

   Changer la couleur d'un objet dans la scène :

   ```
   set_object_color_coppelia(self,value)
   ```

   Relancer le programme de création d'une boite, testez et faites valider.

Exercice 2 – Extension et utilisation de Element

1 – Préparation avant la séance de TP

Pas de préparation.

2 – Réalisation en séance

1. Modifiez la méthode `set_couleur` pour garantir qu'aucune valeur ne dépasse 255 (si une valeur excède 255 alors mettre la valeur à 255).

2. Dans le programme principal, créez avec une boucle `for`, 5 boîtes avec des dimensions [1,1,1] et positionnez-les de manière équidistante entre la position [-4,0,1] et la position [4,0,1]. Au sein de cette boucle, utilisez des `if` pour que les boucles avec une coordonnée x négative soient rouges, celles avec une coordonnée x nulle soient vertes et celles avec une coordonnée x positive soient bleues.

3. Exécutez le programme pour observer l'apparition des objets, faites valider

---

Exercice 3 – Définition d'une classe Robot

1 – Préparation avant la séance de TP

Pas de préparation.

2 – Réalisation en séance

1. Définissez une classe "Robot" avec les attributs suivants :
   a. - nom unique
   b. - type de robot (SCARA ou 3DDL)
   c. - position base
   d. - handle (pour Coppeliasim)

   Pour le constructeur, seuls les attributs nom et type sont initialisés.

2. Dans le programme principal créez un robot SCARA, un robot 3DDL et une boîte avec des dimensions [0.5,1,0.25]. Positionnez la boîte à xyz = [-1,0,0.25/2]. Positionnez le robot SCARA à [-1,-0.2,0.4] et le robot 3DDL à [-1,0.2,0.28].

   Pour ajouter des robots dans CoppeliaSim, utilisez les fonctions suivantes dans les méthodes de la classe :

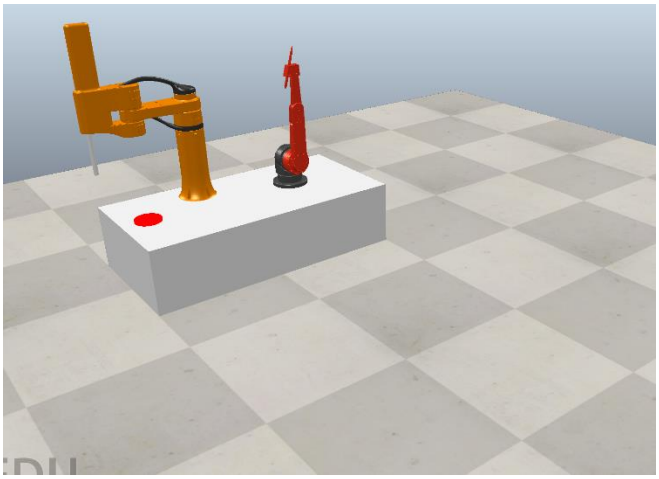
   Ajouter un robot dans la scène :

```
self.handle = send_robot_coppelia(nom, type)
```

   Changer la position de la base d'un robot dans la scène :

```
set_object_pos_coppelia(self, position):
```

3. Lancez la simulation pour observer tous les éléments affichés dans la scène. Et faites valider.
   Résultat attendu :



4. Faites en sorte que le robot SCARA se positionne sur l'objet, le saisisse et le rapproche du robot 3DDL. Pour le faire, ajoutez des méthodes `saisir_objet`, `relacher_objet`, et `move`, dans la classe robot qui doivent contenir les fonctionnes suivantes :

   Pour permettre aux robots de déplacer leur effecteur final (end effector) à la position souhaitée, vous pouvez utiliser la fonction :

```
move_robot_coppelia(self,ref)
```

   Avec ref = "left" pour se déplacer à gauche, et ref = "middle" pour se déplacer au milieu de la table.

   Pour permettre aux robots d'attraper un objet :

```
attach_object_EE_coppelia(self,object)
```

   Pour permettre aux robots de relâcher un objet :

```
release_object_EE_coppelia(self):
```