TP 3 Java sur Eclipse

Des robots « footballeurs » et des robots « cueilleurs »

partie 1

Objectifs du TP

- Utiliser la programmation événementielle pour gérer une balle avec le groupe de robots, ou « cueillir » des objets dans l'environnement.
- Utiliser deux nouveaux types d'agents : BallAgent et CherryAgent

Travail à réaliser

Création du package tp3

- Copier-coller le package java du tp2 (en faisant attention au fichier de configuration) et Renommer le tp3.

Création d'un agent 'balle'

- On va créer un second type d'agent qui a la particularité de réagir aux chocs grâce à un simulateur physique intégré à java3D
- Il faut tout d'abord activer ce simulateur physique dans la simulation qui calculera les forces au moment des chocs entre le robot et la balle.
  Ceci est réalisé par l'instruction suivante (où doit-on la placer?):

  ```
  setUsePhysics(true);
  ```
- On peut maintenant mettre en scène un agent **Balle** avec l'instruction de création anonyme suivante dans l'environnement (**add** est une procédure qui ajoute l'objet créé à l'environnement comme on l'a fait pour les robots) :

  ```
  add(new BallAgent(new Vector3d(x, 0, z), "ball", c,0.25f,0.25f));
  ```

  où dans l'ordre les paramètres sont :
  - x et z sont les coordonnées spatiales pour la balle dans un vecteur 3D (utiliser le générateur Math.random() pour tirer ces coordonnées au hasard),
  - 'c' est la couleur donnée à la balle (voir ci-dessous),
    ex :
    ```
    Color3f c = new Color3f(0.6f,0.5f,0.3f);
    ```

    qui prend en paramètres les 3 valeurs de couleurs RVB par une proportion entre 0f et 1f (de type float),
  - « ball » est le nom donné à la balle,
  - l'avant-dernier paramètre correspond au rayon de la balle
  - le dernier paramètre correspond à la masse de la balle

Pour avoir des compléments sur la classe BallAgent, consulter son code directement dans le package simbad.sim (il faut avoir installé les sources) où par clic droit sur le nom de la classe en lançant la commande 'Open Declaration (F3)'.

Tester le code ainsi généré avec un agent balle et plusieurs robots. Que se passe-t-il ?

Transformer ce code pour introduire plusieurs balles dans l'environnement, de couleurs différentes. Faire en sorte de le prévoir aussi dans votre fichier de configuration.

NB - Bogue : ne pas utiliser les objets Arch dans cet environnement car le simulateur physique semble ne pas les gérer (l'application crache dès qu'une balle touche une arche).

Création d'un agent 'Cherry' (cerise)

On se propose ici d'utiliser de nouvelles méthodes de détection de contact avec le type d'agent 'Cherry'. Il existe une classe CherryAgent dans Simbad dont vous pouvez consulter le code source dans le package simbad.sim (lorsque vous avez installé un projet comportant les sources de simbad).

NB : Vous devez éteindre le simulateur physique pour utiliser ces agents comme défini ci-dessous, du coup on ne peut pas vraiment utiliser des agents 'Ball' et des agents 'Cherry' dans la même simulation au risque de voir les uns ou les autres ne pas fonctionner :

```
setUsePhysics(false);
```

- Création d'un CherryAgent anonyme et son ajout dans l'environnement :

  ```
  add(new CherryAgent(new Vector3d(x, 0, z), "cherry", 0.15f));
  ```

  Comme précédemment, x et z désignent les deux coordonnées horizontales nécessaires pour positionner l'agent (utiliser le générateur Math.random() pour tirer ces coordonnées au hasard), suivi de son nom, puis de sa taille (son rayon).

- Détection d'un agent proche par contact et récupération de cet agent :

  ```
  if (anOtherAgentIsVeryNear()){
      SimpleAgent agent = getVeryNearAgent();
  ```

  où doit-on placer ce code et le code qui va suivre ?
  Puis identification du type de l'agent pour le faire disparaître (avec la méthode detach() de l'agent) si c'est une cerise (on simule la consommation de la cerise...) :

  ```
   if (agent instanceof CherryAgent){
  agent.detach();
  System.out.println("cerise cueillie !");
   }}
  ```

  ou bien à la place de faire disparaître l'objet cerise, on peut changer sa couleur (tester ce code) :

  ```
  ((CherryAgent) agent).setColor(new Color3f(0.9f,0.7f,0.1f));
  ```

- De même que précédemment, tester ce code en multipliant le nombre de cerises placées dans l'environnement.
- Intégrer un attribut dans chaque robot permettant de compter le nombre de cerises récupérées...
  Générer le code correspondant.

Astuce : on peut également utiliser la méthode 'detach()' sur un agent de type BallAgent ce qui en fait une cerise potentielle...