SCIENCES SORBONNE UNIVERSITÉ

Projet

Éléments de programmation par objets avec Java

Présentation

Ce projet a pour but la réalisation d'un programme mettant en œuvre la plupart des points du cours vus ce semestre. Il représente un travail devant prendre environ 5 à 6 heures. Il n'y aura pas de séance de TME dédiée au travail sur ce projet : vous devrez y travailler en dehors des heures des séances.

Le travail demandé est de proposer et d'implémenter un programme qui réalise une simulation se déroulant sur un terrain représenté sous la forme d'un tableau à 2 dimensions contenant des ressources et sur lequel des agents peuvent se déplacer et agir.

Le thème à respecter pour votre programme est **"la Nature"**. Tout programme sur ce thème est donc réalisable.

Par exemple, cela peut être la simulation de la récolte de pollen (une ressource) par une colonie d'abeilles (les agents) et sa transformation en miel (une autre ressource). D'autres exemples (non exhaustifs) de simulation : plantations et récoltes de fruits ou de légumes, ceux-ci pouvant avoir différents degré de maturité, simulation d'une chaîne de l'eau (pluie, évaporation, utilisation,...), etc.

Le programme à réaliser doit respecter le cahier des charges suivant :

- la simulation se déroule sur un terrain de ressources représenté à l'aide de la classe `Terrain` qui est fournie. *Remarque* : contrairement aux ressources, les agents ne sont pas mémorisés sur le terrain.
- la simulation gére des `Ressource` (classe qui est aussi fournie) placées sur le terrain, celles-ci ne peuvent pas être déplacées d'elles-mêmes mais peuvent être retirées du terrain, et des agents qui ont la capacité de se déplacer et de réaliser des actions sur le terrain (prendre des ressources par exemple, ou en déposer).
- votre simulation devra utiliser au moins 2 types de ressources : des ressources dont l'état interne ne change pas (leur quantité reste constante et ne peut être modifiée que par un agent) et des ressources dont l'état interne peut évoluer sans l'intervention d'un agent (*par exemple, par vieillissement (à chaque étape), la quantité peut évoluer au fil du temps, etc., ou la ressource peut se transformer en un autre type (par exemple un bourgeon devient une fleur puis devient un fruit*).

Les classes `Terrain` et `Ressource` sont fournies, accompagnées de leur documentation. Leur représentation UML client est donnée en Annexe.

Feuille de route

1. Choisir le thème de la simulation, identifier les agents et les ressources, et rédiger une description des interactions entre les agents, les ressources et le terrain (ce qui en fait une simulation). Noter qu'il existe d'autres classes que les agents et les ressources, et que des interfaces peuvent être utilisées.
2. Comprendre comment utiliser les classes `Terrain` et `Ressource` en étudiant la documentation fournie et en expérimentant avec la classe `TestTerrain` aussi fournie. Le terrain est représenté dans cette classe par un tableau à 2 dimensions de `Ressource`. Une ressource est caractérisée par un type et une quantité. Le type de la ressource est mémorisé sous la forme d'une chaîne de caractères (par exemple : "Pollen" ou "Miel") donnée lors de la création de la ressource. La quantité est donnée sous la forme d'un entier naturel.
3. Programmer la classe pour représenter des agents. Les agents ne sont pas mémorisés sur le terrain. Un agent possède deux attributs qui correspondent à sa position sur le terrain (numéros de ligne et de colonne). Cette classe doit contenir les deux méthodes suivantes (ainsi que tous les attributs et méthodes que vous jugerez utile de rajouter pour votre simulation) :
   - `distance(x,y)` qui rend la distance (euclidienne) séparant l'objet courant de la case de coordonnées (x,y) ;

- `seDeplacer(xnew,ynew)` qui change la position de l'objet courant sur le terrain et le place en (xnew, ynew) si c'est possible (case non occupée par un autre agent) sinon, le déplacement n'a pas lieu ou une autre solution est mise en œuvre.

4. Écrire la classe `Simulation` qui contient un terrain, une liste ou un tableau d'agents, et une liste ou un tableau de ressources. Cette classe possède un constructeur qui place aléatoirement m ressources sur le terrain et génère n agents. Vous pouvez ajouter dans cette classe la gestion des règles du jeu de votre simulation. Cette classe doit réaliser le travail suivant :
   (a) initialiser le terrain avec des ressources ;
   (b) réaliser une étape de la simulation qui correspond à une action de chaque agent et à la mise à jour de chaque case du terrain ;
   (c) afficher des informations sur ce qui s'est produit durant l'étape ;
   (d) recommencer à l'étape (b) jusqu'à un nombre d'étapes maximum fixés.
5. Écrire la classe `TestSimulation` contenant un `main()`, dans laquelle des essais de simulation sont effectués et qui produit un log à l'écran avec les différentes actions réalisées et leurs résultats. Remarque : de tels logs seront fournis dans le compte-rendu.

Remarque : vous pouvez ajouter d'autres attributs et méthodes aux classes demandées, vous pouvez aussi proposer d'autres classes et des interfaces.

Modalités et compte-rendu

Ce projet est à réaliser seul ou en binôme (2 étudiants max.) d'un même groupe de TD.

Remarque : dans le cas d'un travail en binôme, une seule remise sera faite sur un des 2 comptes Moodle, les noms des 2 membres du binôme doit apparaître en commentaire dans toutes les classes écrites.

Le compte-rendu sera composé d'un seul fichier archive (zip ou tar) contenant les fichiers suivants :
- un fichier PDF basé sur le fichier libreoffice `description.odt` (ou sa version word) complété avec les informations demandées. Toutes les classes écrites dans le projet seront mises (copier/coller) dans ce fichier.
- un répertoire avec tous les fichiers java écrits pour ce projet. Un répertoire `doc` avec une documentation javadoc des classes écrites sera apprécié.
- des exemples de simulations réalisées (fichier log) doivent être fourni avec le compte-rendu.

La soutenance est obligatoire et aura lieu lors de la dernière séance de TME du groupe.

Barème indicatif

La note finale (sur 20) sera déterminée de la façon suivante :

sur 1 point	:	originalité et pertinence (comment ce projet répond bien à la thématique posée)
sur 3 points	:	hiérarchie d'héritage des classes : au moins 3 niveaux, nombre de classes et d'interfaces compris entre 6 et 12. Diagramme UML fournisseur complet et correct.
sur 3 points	:	qualité de la conception, comment les classes, interfaces et leurs liens sont justifiés, respect des bonnes pratiques de POO.
sur 8 points	:	mise en œuvre des éléments vus en cours/td/tme (cf. tableau à remplir dans le fichier `description`)
sur 1 point	:	clarté et lisibilité du programme (noms des variables, méthodes, classes, présence de commentaires explicatifs)
sur 4 points	:	soutenance : réponses aux questions, présentation d'une démonstration des fonctionnalités du programme.

Remarque : la soutenance est obligatoire, un projet rendu mais pour lequel il n'y a pas de soutenance sera noté 0.

Annexe : classes fournies

Trois classes sont fournies pour ce projet. :

- la classe `Terrain` pour gérer un terrain sous la forme d'un tableau d'éléments. Cette classe est finale.
- la classe `Ressource` qui sert à définir le contenu d'une case du terrain.
- la classe `TestTerrain` qui fournit un exemple d'utilisation des 2 classes précédentes. Le source de cette classe est fourni.

Les classes `Terrain` et `Ressource` ne sont pas modifiables, pour cela elles ne sont fournies que sous forme compilée (format .class).

La documentation des classes `Terrain` et `Ressource` est fournie dans le répertoire `doc/` de l'archive `LU2IN002-projet2022.tgz`. Dans ce répertoire, il faut ouvrir le fichier `index.html` avec un navigateur internet.

Pour utiliser les 2 classes `Terrain` et `Ressource`, vous devez mettre leurs fichiers class dans le même répertoire que vos fichiers java.

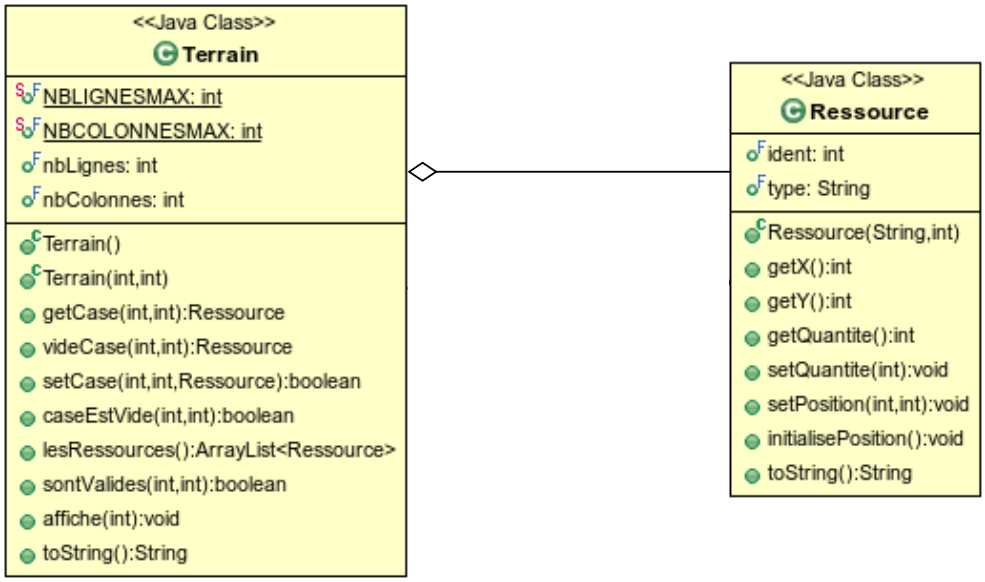

FIGURE 1 – Diagramme des classes fournies (vision client)