Université Claude Bernard Lyon 1
Département Informatique - Polytech Lyon

4A Apprentis, 2017-2018

ISI3

TP 4. *Refactoring* de code & MVC

Victor Lequay, Laëtitia Matignon, Huan Vu

L'objectif du TP est de réaliser un refactoring de code, d'appliquer le patron d'architecture MVC et différents patrons de conception, et de réviser la programmation d'interfaces graphiques en Java avec Swing.

Vous trouverez sur le site du cours :
- un rappel du patron MVC `rappelMVC.pdf`
- un document d'introduction à Java Swing

1 Refactoring du code et MVC

On vous fournit une archive logoInit.zip contenant trois classes dont un programme principal dans la classe *SimpleLogo*.

Question 1 *Commencer par lire, comprendre et tester ce code, puis réaliser une retro-conception de l'application (construction du diagrammes de classes).*

Le code contenu dans l'archive est un package relativement fouillis où les couches graphique et métier ne sont pas séparées. On souhaite tout d'abord **retravailler le code** de cette application pour améliorer sa souplesse et modularité mais **sans y ajouter de fonctionnalités**.

On souhaite obtenir une version simplifiée, qui permet simplement de commander une Tortue dans ses déplacements (avant, arrière, droite, gauche) avec les boutons de l'IHM mais **sans affichage de la trace laissée au fur et à mesure du parcours** (cette fonctionnalité pourra être supprimée lors du *refactoring*).

Question 2 *Proposer un refactoring du code de cette application de sorte à appliquer l'architecture MVC en trois couches : Modèle, Vue, Contrôleur (pas de fusion du contrôleur et de la vue). Pour cela, vous devez réorganiser les éléments de l'application en trois package. Les changements dans le modèle métier (déplacements de la tortue) devront être répercutés dans l'affichage. L'utilisation de l'IHM (boutons) entrainera des changements dans le modèle métier en passant par le contrôleur.* ***La couche Modèle***

devra être indépendante de tout aspect graphique. Votre conception devra aussi permettre d'ajouter facilement d'autres formes pour les tortues (cercle, polygone, ...).

Le diagramme de classes obtenu après refactoring devra être mis dans votre compte-rendu. Ce diagramme précisera en particulier l'organisation des classes en packages et les dépendances entre package.

Question 3 *Modifier le code des classes pour implémenter votre refactoring.*

2 Extension de l'application

2.1 Gestion de plusieurs tortues

On souhaite pouvoir visualiser plusieurs tortues dans la feuille de dessin. L'ajout d'une nouvelle tortue se fera par l'intermédiaire d'un bouton de l'IHM ; la nouvelle tortue aura la couleur sélectionnée dans le JTextField. Un clic souris sur une des tortues la sélectionnera comme la tortue courante. La tortue courante est la tortue commandée par les boutons de déplacement de l'IHM.

Question 4 *Améliorer l'application de sorte à pouvoir ajouter différentes tortues et sélectionner par un clic de souris la tortue courante.*

2.2 Tortues autonomes

On souhaite maintenant au lancement de l'application, proposer le choix de lancer :

- soit une fenêtre avec plusieurs tortues contrôlées avec les boutons de l'IHM (fait dans la section précédente)
- soit une fenêtre avec un ensemble de tortues qui se déplacent aléatoirement (direction et vitesses aléatoires), dans un environnement toroïdal.

Question 5 *Ajouter ces fonctionnalités à votre application.*

2.3 Comportement de *flocking*

On souhaite enfin utiliser l'application avec les tortues pour simuler le comportement de *flocking* que l'on peut observer chez certains animaux comme les poissons ou les oiseaux. Une tortue "intelligente" a une distance de vue et un angle de vue qui définissent son champ de vision. Son comportement est le suivant :

- Essaye d'aller à la vitesse moyenne des tortues qui sont dans son champ de vision

— Va dans la direction moyenne des tortues qui sont dans son champ de vision
— Se rapproche des tortues qui sont dans son champ de vision tout en maintenant une distance minimale avec elles.

(Ces concepts sont une légère variante des règles proposées par Craig Reynolds, ingénieur chez Sony qui a inventé le modèle de Boid (Bird-like Object) qui fait référence dans le domaine des simulations de vie artificielle (cf. `http://www.red3d.com/cwr/boids/`))

Question 6 *Améliorer l'application de sorte à pouvoir visualiser un comportement de flocking avec les tortues. Au lancement de l'application, on proposera les trois choix (tortues contrôlées, tortues autonomes, tortues en mode flocking).*