Rapport: Projet Design Pattern – Pacman

FINNISS Oswald Billy
M1 Informatique

2023/2024

Design Pattern utilisé:

MVC:

Le modèle gère les données du jeu (le labyrinthe, les fantômes, les pacmans) et implémentent la logique du jeu, comme les mouvements et les interactions entre les entités.

La vue c'est ce que l'utilisateur voit et avec quoi il interagit. ViewCommand et ViewPacmanGame sont des vues. Elles affichent les éléments graphiques du jeu (le labyrinthe, les fantômes, les pacmans, les boutons) à l'utilisateur. Elles se mettent à jour en fonction des changements dans le modèle et transmettent les actions de l'utilisateur au contrôleur.

Le contrôleur c'est l'intermédiaire entre la vue et le modèle. AbstracController et ControllerPacmanGame sont des contrôleurs. Il reçoit les actions de l'utilisateur depuis la vue, interagit avec le modèle pour effectuer les changements nécessaires (par exemple, changer la direction des pacmans ou des fantômes), puis met à jour la vue en conséquence.

Observateur:

Le design pattern Observateur est un modèle qui permet à un objet, appelé sujet ou observable, de transmettre automatiquement des changements à un groupe d'objets dépendants, appelés observateurs.

PacmanGame implémente l'interface Observable. Elle garde une liste d'observateurs et fournit des méthodes pour les enregistrer, les supprimer et les informer lors d'événements importants.

L'interface Observateur définit une méthode, actualiser, que les observateurs doivent implémenter. Le sujet (PacmanGame ici) appelle cette méthode pour informer les observateurs des changements en leur fournissant les détails pertinents.

Lorsqu'un changement survient dans (PacmanGame), on actualise l'état interne du jeu. Ensuite, notifierObservateurs() est invoquée pour parcourir la liste des observateurs enregistrés et appeler leur méthode actualiser avec les informations actualisées.

Dans la classe ViewPacmanGame, qui est un observateur, la méthode actualiser est implémentée pour mettre à jour l'interface graphique en fonction des changements dans le jeu. Elle récupère les nouvelles positions des Pacmans et Fantômes, ajuste l'affichage en conséquence, active ou désactive des éléments graphiques en fonction de l'état du jeu (comme l'état des fantômes lorsqu'une capsule est active) et actualise le score affiché.

Ce modèle facilite la communication entre le sujet (le jeu Pacman) et les observateurs (comme l'interface graphique). Il favorise une séparation claire

entre le jeu PacmanGame et la présentation graphique du jeu dans ViewPacmanGame.

Etat:

Le design pattern "Etat" permet à un objet de changer son comportement lorsque son état interne change. Dans ce cas précis, l'interface EtatBouton et la classe EtatEnCours utilisent ce pattern pour gérer les différents états des boutons dans une interface utilisateur. Chaque méthode dans l'interface représente un état spécifique des boutons, déterminant comment ils réagissent dans cet état (comme "Lancer", "Pause", "Pas", "Recommencer").

La classe ViewCommand utilise ces états pour contrôler le comportement des boutons en fonction de l'état actuel du programme. Plutôt que de modifier directement les boutons lors d'interactions, elle délègue cette responsabilité à l'objet représentant l'état courant des boutons (EtatBouton). Cela rend le code plus modulaire et flexible, facilitant l'ajout de nouveaux états ou la modification du comportement des boutons sans altérer directement la classe des boutons eux-mêmes.

Stratégie:

Le design pattern Stratégie est utilisé pour définir une famille d'algorithmes, les encapsuler et les rendre interchangeables. Dans ton code, tu as implémenté ce pattern avec l'interface Stratégie et la classe StrategyRandom.

Interface Stratégie:

Elle définit un comportement commun typeAction() pour toutes les stratégies possibles.

Toute classe implémentant cette interface doit fournir une implémentation spécifique pour cette méthode

Classe StratégieRandom:

Elle implémente l'interface Stratégie et fournissant ainsi une implémentation de typeAction().

La méthode typeAction() génère aléatoirement une action pour les fantômes dans le jeu Pacman.

Cette stratégie est utilisée par les fantômes pour décider de leurs mouvements dans le jeu

Classe StratégieKillPacman:

La stratégie StratégieKillPacman se base sur l'utilisation d'un algorithme de recherche en largeur (BFS) pour déterminer le chemin le plus court entre un fantôme et un pacman dans le jeu Pacman.

Recherche du plus court chemin: l'algorithme BFS explore les cases adjacentes dans le labyrinthe depuis la position actuelle du fantôme jusqu'à ce qu'il trouve le chemin menant au pacman

Choix de l'action: une fois le chemin le plus court trouvé, la stratégie détermine la prochaine action du fantôme en se basant sur la première étape de ce chemin. Elle calcule la direction vers la prochaine case du chemin afin que le fantôme puisse se rapprocher au maximum du pacman.

Les différentes fonctionnalités:

→ Il faut lancer la classe *Test* pour lancer le jeu

- ➔ Menu avec deux options au démarrage : Charger un labyrinthe et Quitter
- ➔ Les commandes Run, Restart, Step et Pause fonctionnent correctement.
- ➔ Système de scoring lorsque le pacman mange des Foods sont score augmente
- ➔ Chaque agent pacman ou fantôme se déplace d'une case à chaque tour dans la direction de son choix mais ne peut pas aller sur une case mur.
- ➔ Si un agent pacman se trouve sur la même case qu'un agent fantôme il est éliminé et disparaît du plateau.
- ➔ Des capsules spéciales sont disposées sur le plateau. Une fois mangée par un pacman, elles rendent les fantômes vulnérables pendant 20 périodes (dans le jeu j'ai juste mis une période de 5 afin de ne pas attendre trop longtemps) au cours desquelles les Pacmans peuvent les manger Fantomes.
- ➔ Lorsqu'un pacman mange une capsule les fantomes deviennent effrayés ainsi les pacmans peuvent manger les fantômes pendant cette période. En dehors cette période si un pacman rencontre un fantome le pacman meurt donc « Game Over ». Si il mange tous les Pac-Gommes il a gagné.
- ➔ On peut controler le pacman avec les directions du clavier en cliquant une première fois sur l'interface du jeu afin d'activer le focus.
- ➔ Les fantomes ou les pacmans peuvent avoir une strétégie random ou une où ils se déplacent aléatoirement sur le labyrinthe. Les fantomes peuvent avoir une StrétégieKillPacman, ainsi les fantomes adopteront le plus court chemin afin de tuer le pacman (dès fois cette StratègieKillPacman bug et ne fonctionne pas).