Université Claude Bernard Lyon 1

Projet Java : PacMan

Sommaire

Table des matières

Fonctionnalités

Pour la répartition des tâches, Liam aura le code couleur vert et Lucas aura le code couleur rouge

Gestion du Jeu

- Déplacer le personnage PacMan de case en case. Le code était fourni, aucun temps de travail.

- Manger des « PacGums » (petites boules jaunes) qui rapportent des points. Le jeu se finit lorsqu'il n'y en a plus. Temps consacré : environ 1h.

- Il y a des « PacGums » spéciales, qui confère des pouvoirs au PacMan.
  3 pouvoirs différents :
  - La gum verte : En la mangeant il peut passer à travers les murs
  - La gum bleue : En la mangeant, il est « invisible » et ne peut pas se faire manger par les fantômes
  - La gum rouge : En la mangeant, il devient « mangeur » de fantômes

  Les pouvoirs durent 8 secondes à partir du moment où le PacMan mange la gum. Ces gums apparaissent sur la grille toutes les 10 secondes, aléatoirement positionnées. Un thread est dédié au 8 secondes du pouvoir du PacMan, et un autre à la création/destruction des « PacGums » spéciales. Temps consacré : ~20h.

- Effet de « wrap arround » pour le personnage PacMan : lorsqu'il sort de la grille il réapparait de l'autre côté s'il n'y a pas de murs. Temps consacré : ~2h.

- Implémentation d'un timer, qui pose une limite de temps. Lorsque le timer arrive à 0, la partie se termine, et le joueur a perdu. Temps consacré : ~2h.

- Système de nombre de vie : Lorsque le PacMan se fait manger par un fantôme, il perd une vie, et le joueur perd la partie quand il ne lui reste plus de vie. Temps consacré : ~2h

- Système de score : Le score augmente de 10 pour chaque PacGum mangée. Les PacGums spéciales ne valent pas de point. Temps consacré : ~30min

- Système de paramètre : Les paramètres sont enregistrés dans un fichier settings.txt lorsqu'ils sont modifiés. Les paramètres modifiables sont le timer, le nombre de vie, le nombre de fantômes et la vitesse du jeu. Temps consacré : ~4h

Graphisme

- Toutes les infos sont affichés au-dessus de la grille : Score, Timer, et Vies.
    - Le timer devient rouge lorsqu'il ne reste que 15 secondes. Temps consacré : ~30min
    - Les vies sont représentées par les icones PacMan, qui s'effacent tour à tour à chaque fois que le joueur perd une vie. Temps consacré : ~2h

- Ajout des murs et des « PacGums » : Une classe Objet mère a été créée, dont dérive la classe Murs et la classe PacGum. Une grille pour les objets a également été créée dans la classe Jeu, pareillement à la grille des entités. Temps consacré : ~3h

- Utilisation d'un tableau de 4 icones pour représenter le PacMan, les 4 icones sont les mêmes mais avec une orientation différente. Cette fonctionnalité sera expliquée en détails dans la partie consacrée au choix de conception. Temps consacré : ~ 30min.

- Chaque PacGums spéciale mangées confère sa couleur au PacMan, et un pouvoir comme expliqué plus haut. Temps consacré : ~30min.

Gestion des Fenêtres

L'application comporte 4 fenêtres en tout : la fenêtre du menu, la fenêtre des paramètres, la fenêtre de jeu, et la fenêtre de fin de partie.

- Fenêtre du menu : C'est une simple fenêtre avec une image en fond prise sur Internet, avec 2 boutons transparents qui se placent sur le texte « Start » et sur le texte « Controls » de l'image. « Start » démarre la fenêtre de jeu, « Controls » démarre la fenêtre de paramètres. Les paramètres du jeu sont chargés par la fenêtre du Menu depuis un fichier texte puis passés en paramètres pour chaque création de nouvelle fenêtre. Voir Annexe (1). Temps consacré : ~2h.

- Fenêtre de paramètres : Les paramètres du jeu (nombre de fantôme, durée du timer, nombre de vie et transmis par la fenêtre de menu sont réutilisés pour afficher les paramètres actuels. Lorsque le joueur appuie sur le bouton sauvegarder, les nouveaux paramètres sont enregistrés dans le fichier, puis la fenêtre se ferme et la fenêtre du menu apparait. Voir Annexe (2). Temps consacré : ~5h

- Fenêtre de jeu : La fenêtre de jeu comporte ce qui a précédemment été dit dans la partie Graphisme : la première partie, située tout en haut de la fenêtre indique le score, le timer et le nombre de vie. La deuxième partie juste en dessous comporte la grille avec le jeu. Lorsque la partie est terminée, la fenêtre de fin de partie se lance. Voir Annexe (4). Temps consacré : ~2h.

- Fenêtre de fin de partie : Cette fenêtre ne comporte pas grand-chose, seulement un texte qui annonce si l'on a perdu ou gagné (booléen passé en paramètre lors de la création de la fenêtre). Elle comporte également 3 boutons : le 1er envoie vers la fenêtre de paramètres directement, le 2ème envoie vers le menu, et le 3ème ferme la fenêtre et quitte le jeu. Voir Annexe (3). Temps consacré : ~2h.

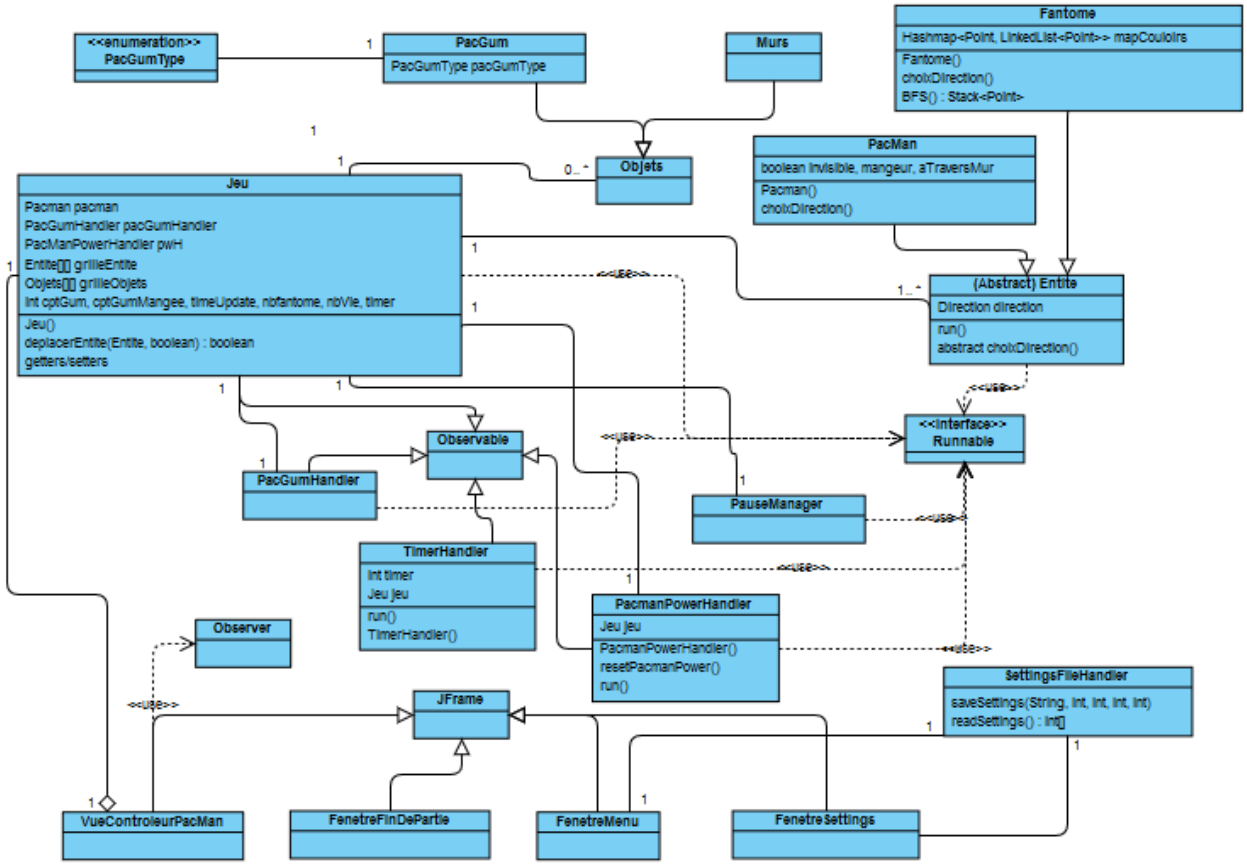


Diagramme de classe de l'application

Dans un souci de place et de lisibilité, les méthodes des classes n'ont pas été représentées dans leur totalité.

Pour ce projet, le code de départ était conçu sur la base du pattern MVC (Modèle, Vue et Controleur). Nous avons donc décidé de rester sur ce choix et de construire l'application sur ce modèle. Le Controleur et la Vue sont cependant dans le meme fichier étant donné que le Controleur sers uniquement pour récuperer les entrées claviers afin de diriger le PacMan.

Pour ce qui est de l'organisation des données du modèle concernant la grille, ses composants et les entités. Le code fourni avait modélisé les entités et leur position dans une un tableau deux dimensions qui permettait de retrouver les entités à partir de leur position. Pour ce qui est des objets, on a décidé de créer une classe mère Objets à l'instar de la classe Entite. Deux classes héritent d'Objets : Murs et PacGum. Sur le même principe que la grille des Entite, un tableau 2 dimensions « grilleObjets » a également été créé afin de connaitre l'objet en question à partir de la position de la grille.

Toutes les classes qui implémentent l'interface « Runnable » sont des classes qui représentent un thread particulier. La classe Jeu représente le thread principal du jeu qui gère les données, récupèrent les directions choisis par les threads des Entités puis demande à la Vue de rafraichir l'affichage. Les threads PacGumHandler et TimerHandler sont lancés en début de partie et tournent en boucle tout le long. PacGumHandler gère la création des PacGums spéciales pendant toute la partie, et TimerHandler gère le timer (il attend une seconde, change la variable timer dans Jeu et demande à la vue de rafraîchir l'affichage). Le thread PacManPowerHandler est lancé à chaque fois que le PacMan mange une PacGums spéciale (il attend 10 secondes et change l'état du PacMan).

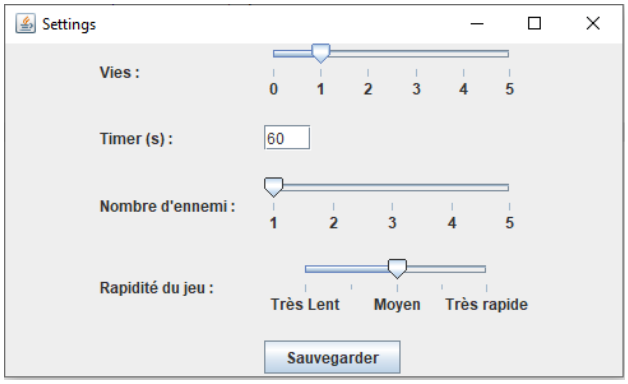
Pour l'affichage du PacMan, nous avons fait le choix d'utiliser une image par direction du PacMan, au lieu de d'effectuer une rotation sur l'icône à chaque fois. Si le PacMan va vers le haut, la vue charge l'icône « PacMan_haut.png ». La même technique a été utilisé pour chaque pouvoir que le PacMan peut avoir en mangeant les PacGums spéciales (un dossier pour chaque pouvoir, contenant les 4 icones du PacMan pour les 4 directions), et chaque pouvoir colorie le Pacman de la couleur de la gum spéciale associée.

Nous avons également fait le choix de passer les paramètres du jeu (nombre de vie, durée du timer, nombre de Fantôme et rapidité du jeu) en tant que paramètre de méthode lors de la création de chaque fenêtre. Les paramètres sont lus dans le fichier texte lorsque la fenêtre du Menu s'affiche et c'est elle qui passe les paramètres lus à la fenêtre de jeu ou à la fenêtre des paramètres selon le choix de l'utilisateur.

Plusieurs extensions ont été ajoutés :

- 3 PacGums spéciales avec 3 pouvoirs différents présentés dans les fonctionnalités.
- Le « wrap-arround » qui permet au PacMan de passer d'un coté à l'autre de la grille
- Système de paramètre, avec une fenêtre dédiée et une sauvegarde dans un fichier texte
- Système de « Pause » lorsque le PacMan perd une vie : Il est repositionné dans le coin gauche en haut de la grille (position (0,0)), et clignote pendant 3 secondes afin de laisser le temps au joueur de choisir dans quelle direction en fonction de la position des Fantômes. Le thread représenté par la classe PauseManager() se charge de ça.
- Implémentation d'un Timer, qui termine la partie lorsqu'il arrive à 0.

(1) Fenêtre de Menu



(2) Fenêtre de paramètres