Projet Prog 2 : Tower Defense
Compte Rendu 1

Raphaël LE BIHAN, Rida LALI

Février 2020

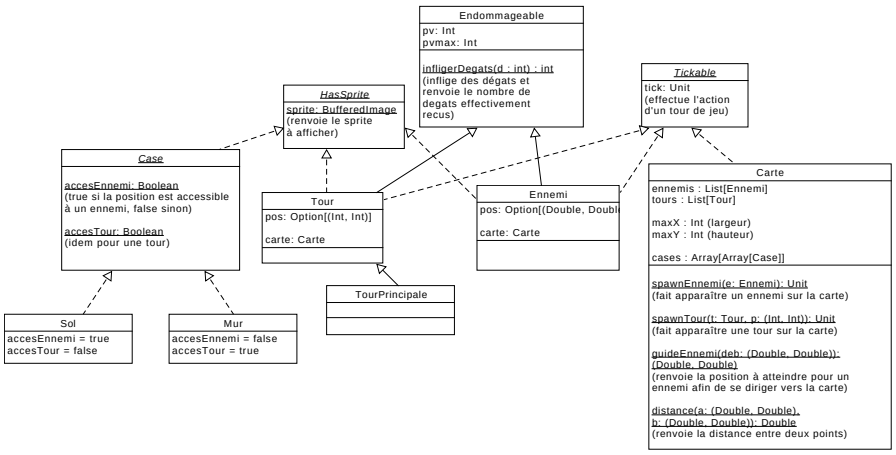
1 Présentation du projet

1.1 Présentation générale

L'objectif du projet est de réaliser un jeu de Tower Defense.
Le jeu présente une carte 2D où est située une tour appelée tour principale, reliée au bord de l'écran par un chemin. Lors de différentes manches, des ennemis arrivent du bord de l'écran et tentent de prendre d'assaut la tour principale.
L'objectif du joueur est de défendre la tour principale de l'assaut ennemi. Pour cela il peut disposer différentes tours de défense sur la carte, qui attaqueront les ennemis à proximité.

1.2 Organisation du code

Le code source s'organisent en différentes classes et traits, selon le diagramme suivant :



1.3 Moteur du jeu

Le déroulement d'une manche se déroule en une succession d'actions élémentaires appelées tick pour chaque élément placé sur la carte. Tous les éléments susceptibles d'effectuer un tick héritent du trait Tickable.

L'appel de chaque tick pour les éléments de la carte se déroule de la manière suivante :

- À un intervalle de temps (réel) régulier la fonction main appelle la méthode tick de la carte du jeu.
- Cette carte parcourt d'abord les listes des ennemis et des tours pour déterminer lesquels sont morts, et les supprimer de la liste des éléments sur la carte,
- Puis, pour chaque tour et chaque ennemi encore en vie, elle appelle la méthode tick de cette tour/cet ennemi, correspondant à un tour de jeu élémentaire

1.4 Interaction entre les classes et objets

Les différentes classes et objets définis possèdent chacuns des variables référençant les autres objets avec lesquels ils sont concernés directement. De plus les objets peuvent communiquer entre eux par échange de messages : par exemple une tour de défense (Tour) inflige des dégâts à un ennemi (Ennemi) en appelant sa méthode infligerDegats (définie dans Endommageable)

Attaque Tour -¿ Ennemi Lors de chaque attaque (déclenchée par la méthode tick), la tour :

- obtient la liste des ennemis sur la carte (message à l'objet de type Carte pointé par sa référence),
- parcourt cette liste et trouve les ennemis qui sont à sa portée (message à chaque ennemi pour connaître sa position),
- attaque un des ennemis qui est à portée de tir

Attaque Ennemi -¿ TourPrincipale Les ennemis se déplacent vers la tour principale et l'attaquent lorsque celle-ci est à portée de tir. L'attaque se fait de la même manière que l'attaque Tour -¿ Ennemi.

Déplacement d'un Ennemi Les déplacements des ennemis sont guidés par la carte. Pour se déplacer, un ennemi :

- vérifie s'il est à portée de tir de la TourPrincipale (si oui il ne bouge pas, il attaque juste)

- sinon, il consulte la carte pour connaître l'emplacement vers lequel il devrait se diriger (message guideEnnemi en envoyant sa position actuelle en paramètre)
- puis il modifie sa position (la distance parcourue dépend de sa variable vitesse)

2 Avancement du projet et difficultés rencontrées

Étant débutants en programmation orienté objet, avons rencontré quelques difficultés concernant l'utilisation des classes, traits et leurs interactions.

2.1 Etat actuel du code

A l'état actuel la partie correspondant à l'interface graphique du jeu ne compile pas. En effet nous avons commencé le développement de l'interface graphique (un peu tard) et avons rencontré des difficultés d'utilisation de la librairie Swing (avec des problèmes d'accès ralenti à la salle C411 même par SSH pour recompiler le projet).

La partie du code correspondant aux interactions entre les différents objets du jeu a été développée et débuggée (et elle fonctionne correctement).

Nous avons également fait le choix de privilégier en premier lieu le bon fonctionnement global des différents objets, classes et traits, et n'avons pas encore abordé l'utilisation plus détaillées des classes (variables et valeurs privées, classes abstraites, méthodes statiques etc..).
De même plusieurs algorithmes ou structures de données utilisées ne présentent pas une complexité optimale, et des améliorations sont possibles et prévues.

Il reste enfin certains éléments du moteur de jeu (comme la fonction main se appelant la méthode Carte.tick en temps réel) non développés.

2.2 Prochaines étapes de développement

Les prochaines étapes prévues pour le développement sont :

- la finalisation de l'interface graphique (même simple, car elle est surtout utile à des fins de débuggage),
- la complétion du moteur de jeu, ainsi que des parties laissées en suspens (comme l'implémentation de la méthode tick pour les tours)
- l'utilisation de notions détaillées (lorsque cela est plus pertinent) concernant la programmation orienté objet, avec au besoin une modification des membres des différentes classes
- (puis si nous avons suffisamment de temps) le développement de mécaniques de jeu plus intéressantes, comme un temps de rechargement pour les

tours/ennemis, des améliorations pour les tours, de l'argent pour l'achat ou l'amélioration de tours
Nous nous concentrerons en premier lieu sur les trois premiers points.