SAEs S2 2022-2023

1 Auteurs

Arsene – Zen - Benjamin

2 Diagrammes de classe (10 points pour SAE S2.01)

(Voir pièce jointe)

Ce diagramme montre bien l'héritage avec les classes abstraite attaque – Ennemi et tour qui permettent de stoker les différents types de tour/ennemi en fonction de leurs statistiques. De même pour les attaques des tours, on a la possibilité de créer un nouveau type d'attaque (projectiles des tours) à tout moment.

On remarque aussi 2 interface : « mobile » et « objet » qui sont utiles pour les calcules de position entre les différents élément qui ont une position dans le jeu (qui ont un x et un y).L'interface mobile sert a regrouper ennemi et attaque car les ils peuvent se déplacé dans le jeu.

Les paquets tour et ennemi désigne l'ensemble des sous-classes de leurs parents, ce qui correspond à l'ensemble des tours et ennemis réel de notre jeu. Ceci nous permet de rajouter une nouvelle tour très facilement à tout moment

3 Tests Junit : (15 pointspour SAE S2.01)

Une ou deux classes bien choisies devront être accompagnées de tests Junit cohérents et raisonnablement complets.

Les tests junits sont à rendre de façon indépendante sur git (fichiers .java)

4 SAE S2.02

Structures de données (30 points)

HashMap<> : utilisation d'un dictionnaire pour :

- Stocker les différentes images de la vue dans le but de les charger qu'une seul fois et d'éviter les lags
- Retrouver facilement le nom des évolutions de chaque pokémon (tours) en fonction de leur nom de base et inversement. Utile pour ne faire un grand ifelse a chaque fois qu'un pokémon va évoluer et donc changer de nom.

IntegerProperty :

- Beaucoup utilisé pour relier une valeur du modèle qui va pertinemment changer a un élément qui doit être visible dans la vue. Utilisé par exemple pour les hp des ennemis qui sont utilisé dans un calcul puis bind a une barre de vie dans la vue

algorithmique (70 points)

1) Algorithme de déplacement dans la classe ennemi : utilisation d'un BFS pour calculer le chemin que va devoir emprunter l'ennemi.

2) La méthode attaque() de Tour (i compris les méthodes utilisées dedans) : cherche rapidement et efficacement le premier ennemi que la tour peut attaquer puis l'attaque.
3) La méthode bouge() dans Projectile : permet un déplacement simplifier et efficace de tout les projectiles des tours
4) La méthode appliquePoison() dans la classe Nidoran : permet d'infliger des dégâts sur la durée a tous les ennemi que Nidoran a toucher auparavant.

5 Document utilisateur (40 points pour SAE S2.05) :

- Voir pdf SAE_S2.05_DocumentUtilisateur_ArseneZenBenjamin.pdf sur git.