Rapport jeu C++

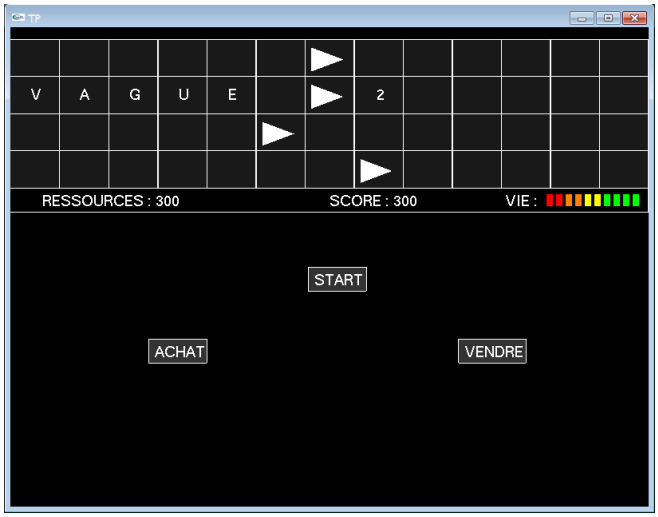
Meersman Rudy

M1 Info

1 Présentation du Projet

1.1 Le jeu

Ce programme est un jeu dans lequel il faut survivre et faire le plus grand score possible. Le joueur possède une barre de vie et des tours, pour se défendre, qu'il peut acheter avec des ressources. Les météorites arrivent par vagues, peuvent infliger des dommages aux tours et, lorsqu'elles sont détruites, font gagner des points et des ressources au joueur. Plus le jeu avance, plus les vagues deviennent agressives mais plus de tours peuvent être placées.



1.2 Comment jouer

Vous devez, en début de jeu, poser 4 tours et une première vague apparaitra. A la suite de celle-ci, vous aurez des ressources utilisables pour acheter d'autres tours dans le menu achat ou encore vendre vos tours à moitié prix. Pour cela, il suffit d'appuyer sur achat, le bouton sous la tour souhaité et cliquer sur le plateau de jeu pour la placer. Vous êtes limité à 10 000 de ressources et n'en gagnerez plus une seul tant que vous êtes au maximum.

Entre chaque vague, vous pouvez vous réorganisez en achetant des tours ou en vendant les vôtres. Pour passer à la vague suivante, il suffit d'appuyer sur le bouton Start.

Pour les tours, vous pouvez en placer que 8 en début de partie et toutes les 10 vagues, vous pourrez en mettre 4 de plus. Chaque tour a ses propres caractéristiques et prix.

Les météorites seront de plus en plus difficile à détruire mais rapporterons plus de ressource et de points.

Si une météorite atteint le bord droit du plateau, vous perdrez un point de vie. Si vous n'avez plus de points de vie, vous perdez la partie.

2 Choix de Conception

2.1 Les Tours

Pour cette partie du code, j'ai décidé de faire 4 tours différentes et qui hérites d'une classe mère appeler « abstractTower » afin de pouvoir mettre les tours dans une seul et même liste ou évité une duplication de code (chaque tour tire ou encore chaque tour peut être détruite). Chaque tour possède une liste « d'abstract shot » et d'une fréquence. De plus, toutes les class-fille possède un attribut « environnement » afin de pouvoir accéder à cette dernière afin de pourvoir gérer les collisions avec les tours. De plus, chaque fois qu'elle prend un coup par un météore, une frame d'invulnérabilité s'active pour un temps donné.

2.2 Les Shots

Les Shots sont des classes qui hérite de « MovingEntities » afin qu'elles aient une vitesse et une position. La différence des trois sous classe et pour différencier les différents types de tir. Chaque tir est inclus dans une tour afin d'être dessiner et déplaçable par appel de cette dernière. Cela évite d'avoir trop de liste dans la classe « Environment ».

2.3 Les Meteorites

Comme pour les Tours et pour les « Shots », il y'a une classe mère pour les météores. Ce sont aussi des « MovingEntities ». Chaque météore a une certaine quantité d'argent, score et de points de vie. Les deux premiers seront appelés par « l'environment » lorsqu'elle sera détruite par le tir d'une tour.

2.4 AbstractBouton

Ne voulant pas utiliser les touches du clavier, j'ai créé une classe de bouton afin de naviguer dans le menu d'achat ou lancer une vague avec la souris. Pour cela, il y'a une classe « abstractBouton » qui aura la position du bouton afin de pouvoir savoir quand on appuis dessus (cette détection est faite dans « l'environment »).

Leurs états est présent afin de pouvoir les différencier de lorsqu'ils sont utilisable ou non grâce à la différence de transparence. Les classes filles possèdent l'action représentée par ce bouton et pour « Start » une action de désactivation afin de changer le texte dans le bouton afin de changer le texte.

2.5 Player

Le joueur est représenté par cette classe que ce soit ses ressources, le score ainsi que la vie qui déclenchera le Game Over dans « l'environment » lorsque celle-ci atteindra 0.

2.6 MenuDraw

Cette classe n'a qu'un seul but : dessiner les informations sur le joueur ainsi que la vague actuel du jeu. Elle est appelée par « environment » pour être dessiné.

2.7 Board

Le « board » va contenir les « Dalles » qui contiennent les tours. Ce « board » est initialisé par une certaine taille et largeur, cependant les « Dalles » possède une taille fixe afin de facilité les calculs des placements des tours et des cliques sur les dalles.

2.8 Environment

Le moteur du proiet. Elle va contenir le « board », une liste de météorites, le joueur et des boutons.

Dans cette classe on trouve la fonction général « Draw() » qui va dessiner à chaque tic tout ce qui est demandé. Le « draw » affichera le menu d'achat si on a cliquer sur le bouton « achat » grâce au boulean modifié par « act() » de « BuyBouton ». Il affichera aussi les tours tout le temps, leurs tires (draw appeler dans « l'abstractTower ») et les vague lorsque le « timer » l'autorisera à le faire. Le timer sera modifié à la fin de chaque vague ou lorsque le bouton « Start » sera activé.

Lorsqu'il est à « False », le « idle() » modifiera les positions de tous les « MovingEntities » via leurs « move() » grâce aux tours et à la liste de météorites.

Le « idle() » lancera automatiquement la première vague du jeu lorsque le nombre de tour posé sera atteint (cela est connus grâce à « nbTower » au lieu de la fonction « sizeTower() » de « Board » afin d'évité un problème si jamais le joueur perd des tours et se retrouve à 4 tours ou moins.). Le « MouseCallBack() » est la fonction permettant de cliquer sur un bouton. On peut savoir grâce à la fonction « clickable() » si on a cliqué sur un bouton ou non en modifiant les coordonnées de la position de la souris en float (la taille de la fenêtre étant fixe grâce à la ligne `glutReshapeWindow(800, 600);` nous pouvons savoir la position en pourcentage par rapport aux dimensions.).

La vague est initialisée après chaque fin de la vague précédente grâce à une boucle qui va placer les météorites en fonctions du niveau (les vagues 4, 8 et 15 apportent de nouveau météores pour la difficulté du jeu).

L'achat de tour se fait au placement de cette dernière grâce au « type » qui change en fonction du bouton sur lequel on a appuyé (type 1 : basique, type 2 : lourde, type 3 : défensives,

type 4 : sniper). Il est détecté au moment de la pose si le joueur a assez d'argent. Si ce n'est pas le cas, la tour ne sera pas posée et l'argent ne sera pas débité du joueur.

Le Game Over sera lui activé lorsque les Point de vie du joueur seront à 0. Le boolean « gameOver » permet de le savoir et empêchera l'affichage de tout menu. La seule chose qui sera affiché sera Game Over et le HUD et le plateau du jeu.