Projet C++

Pierre Tassel

Décembre 2017

1 Introduction

Ce document ce concentrera sur les diverses étapes qui ont mené à la création du jeux de Tower Defense en C++.

2 Représentation du jeux

2.1 Le Damier

```
class Damier {
    private:
        Case** cases;
        std::queue<Asteroide*>* asteroides = new std::queue<Asteroide*>();
        int lignes;
        int colonnes;
        int vieJoueur = 3;
        int scoreJoueur = 0;
        int argentJoueur;
        int vaisseauSelectionneJoueur = 0;
        Vague* vague;
};
```

Les données du jeux sont sauvegardé dans la classe Damier, elle contient un tableau de Case représentant les cases du jeux, une file contenant les astéroides présent dans le jeux, elle gère aussi les élements relatif au joueur tel que sa vie, son score et l'argent qu'il lui reste.
La selection du type de vaisseau se faisant pas un click en haut a droite de l'écran, le damier garde en mémoire le vaisseau sélectionné par l'utilisateur.
Le damier a aussi un objet de type Vague dont il se sert pour gérer le flux de vague.
Le damier est créé à l'aide de la méthode statique creerDamier du DamierFactory

```
Damier* DamierFactory::creerDamier(int lignes, int colonnes)
```

Elle s'occupe d'initialiser les cases en les mettant à la bonne taille et aux bonnes coordonées sur l'écran en fonctions du nombre de lignes et de colonnes passé en paramètre.

2.2 Les Cases

```
class Case{
    private:
        float x;
        float y;
        float width;
        float height;
        Vaisseau *vaisseau = NULL;
};
```

Chaque case contient un pointeur vers un vaiseau (il y a maximum un vaiseau par case), si le pointeur est a NULL (par défaut) alors il n'y a pas de vaisseau.

2.3 Les Vaisseaux

```
class Vaisseau{
    private:
        float x;
        float y;
        float width;
        float height;
        int vie;
        int temps;
        int puissance;
        int vitesse;
        std::queue<Missile*>* missiles;
        int compteur = 0;
        Couleur couleur;
        int degatSubit = 0;
        float portee;
};
```

Le vaisseau a un certain nombre de vie, une puissance, une vitesse et un interval de temps entre deux missiles.
Les missiles envoyés par le vaisseau sont stocké sous forme de file.
Les vaisseaux sont crée par une factory (VaisseauFactory) qui s'occupe de générer le bon vaisseau en fonction du choix de l'utilisateur.
Le missile s'assombrit légèrement quand il subit des dégâts.

2.4 Les Missiles

```
class Missile{
    private:
        float x;
        float y;
        int degat;
        int vitesse;
        bool colision;
        Couleur couleur;
        float portee;
        float parcouru;
};
```

Les missiles ont une portée (distance parcouru avant de mourrir), une vitesse et des dégats qui sont ceux du vaisseau.

2.5 Les Asteroides

```
class Asteroide {
private:
    int vie;
    float x;
    float y;
    int degat;
    int vitesse;
    float perimetre;
    int yToLigne(float y, int lignes);
    int scoreRapportee;
    int argent;
    Couleur couleur;
};
```

Les astéroides sont stockés dans le damier sous forme file.
Les astéroides sont générés par une factory (AsteroidesFactory) à qui ont envoie la ligne où l'on veut créer l'asteroide et la difficultée de l'asteroide (de 0 à 2) :

```
Asteroide* AsteroidesFactory::creerAsteroide(int ligne, int lignes,
int difficultee);
```

La colision entre les missiles et les astéroides est géré par le damier. Il parcour la file des astéroides, récupère le numéro de ligne de l'asteroides. Puis l'on parcours toute les cases de la ligne de l'astéroide, et tous les missiles du vaisseau de la case.
Si il y a colision alors on retire de la vie a l'astéroide et on supprime le missile, sinon on regarde si le missile est toujours visible à l'écran, si non on le supprime.

On regarde aussi s'il y a une colision avec un vaisseau, si c'est le cas on réduit la vie du vaisseau et on détruit l'asteroide.

2.6 Les Vagues

```
class Vague{
private:
    int vagueNumero = 0;
    int tics = 0;
    int interval;
    int nbAsteroideRestantCreer;
    int lignes;
};
```

Les vagues sont gérés par le damier, c'est elle qui crée les astéroides au travers de la factory. Elle gère le nombre d'astéroides à créer et le type d'astéroides à créer (la difficultée).
Si il n'y a plus d'astéroides à créer et qu'il n'y a plus d'astéroides dans le damier, la vague est finie et on peut lancer une nouvelle vague en appyuant sur la touche entrée, la nouvelle vague est créé par la méthode KeyboardCallback de la classe MyControlEngine.

3 Diagramme de classe