Tower Defense

Projet réalisé en C++

Tran Thanh Long - 25 décembre 2017

Read Me

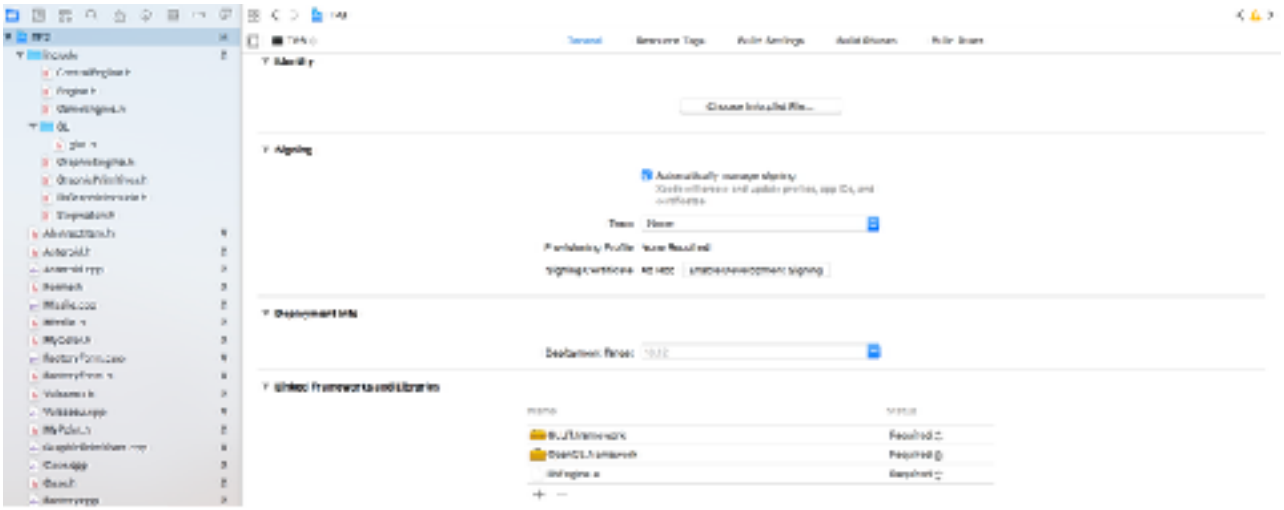
C'est un projet avec Xcode sous Mac:

- dézipper
- cliquer sur TP3.xcodeproj pour ouvrir le fichier dans l'environnement XCode
- lancer

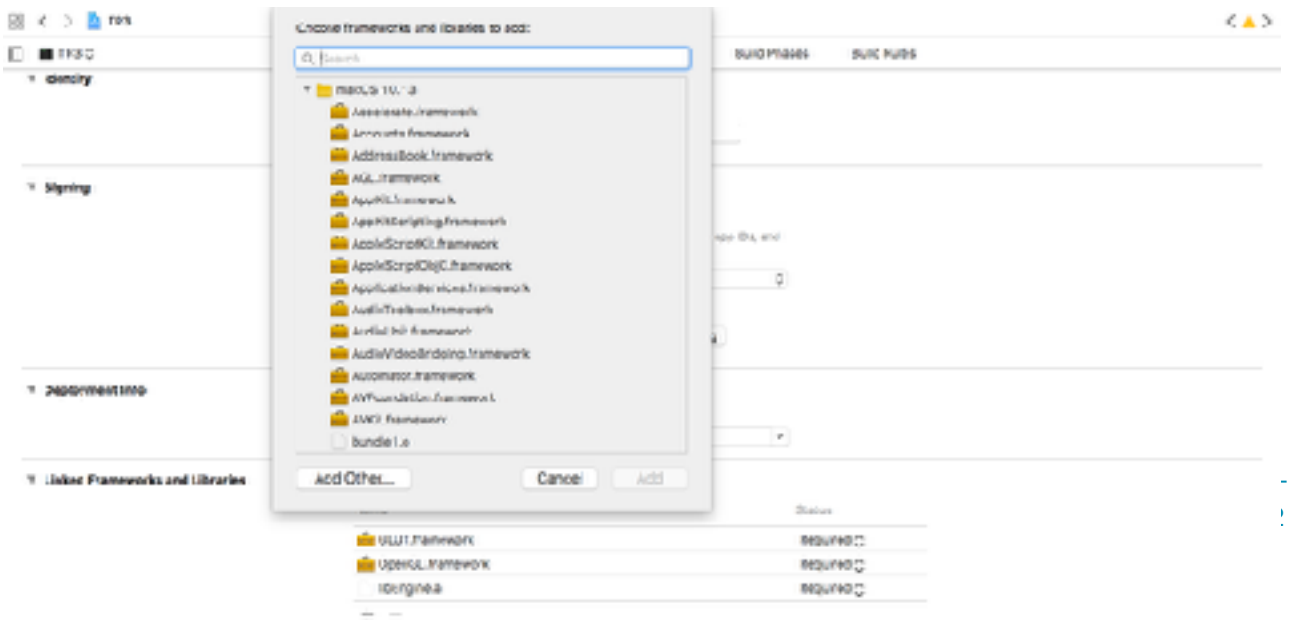
Si cela ne se lance pas, suivez les étapes:

Les dossiers « include » et « libMac » se trouvent dans le zip, même emplacement que le dossier « src ».

1) Cliquer sur TP3 dans la barre à gauche, puis « General », vous allez retrouvé l'onglet ci-dessus, supprimer les liens dans Linked Frameworks, et ajouter de nouveau les fichiers. Les fichiers GLUT et OPENGL sont déjà dans Xcode.



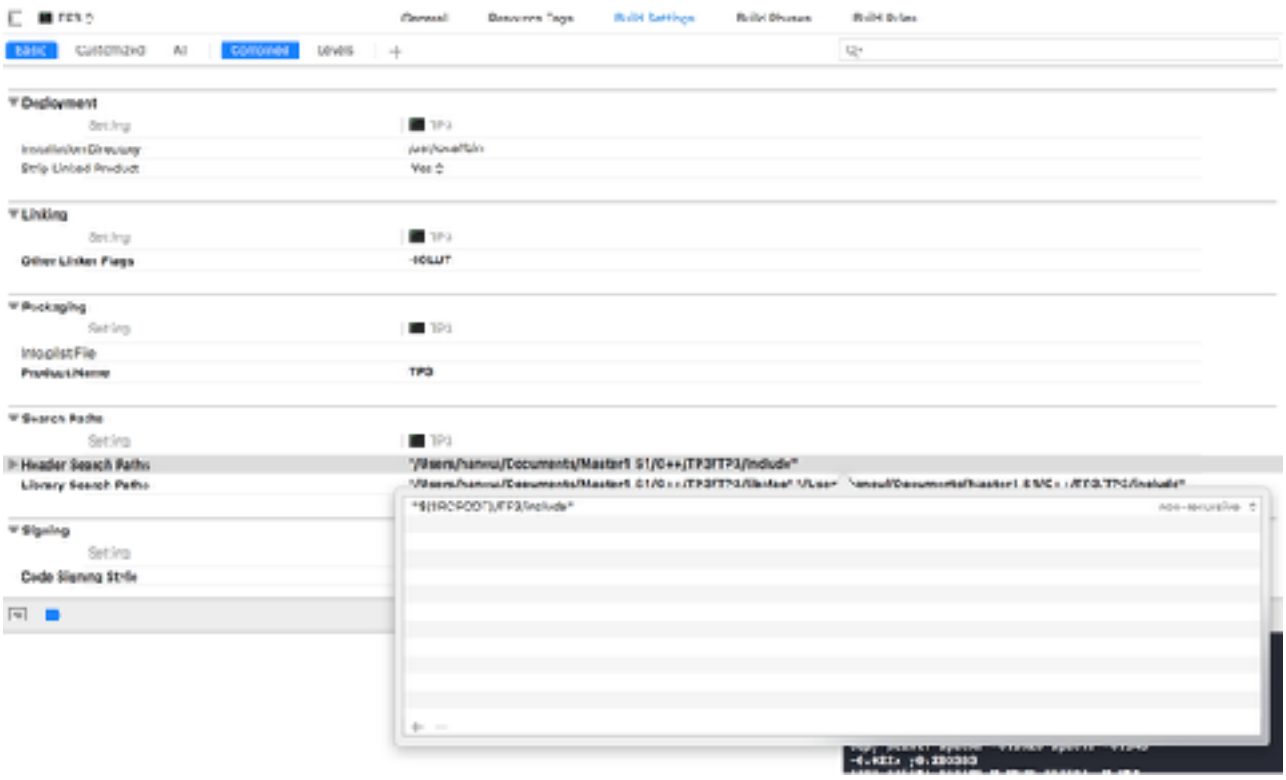
2) Cliquer sur + et vous obtenez cette nouvelle fenêtre. Rechercher ces fichiers dans la barre et les ajouter. « Add other » pour ajouter libEngine qui se trouve dans le dossier libMac.



Ensuite, cliquer sur « BuildSettings » et chercher ou défiler jusqu'à « search paths ».

Cliquer sur Header Search Paths et supprimer le(s) liens . Glisser le dossier include dans cette même fenêtre.

Cliquer sur Library Search Paths et supprimer le(s) liens . Glisser le dossier « include » et « libMac » dans la même fenêtre.



Lancer à nouveau.

Information sur le jeu

Appuyer sur commencer pour lancer le jeu. Choisissez et placez un des tours sur le quadrillage.

- si la tour touche le bord du quadrillage, vous perdez 1 vie.
- Vous gagner de l'argent en tuant les astéroïdes.
- Vous perdez de l'argent en choisissant un tour.

Il existe 3 niveau de tour:

Niveau 1: fréquence = 50; compteur = 10; missile niveau 1 (puissance = 1, vitesse = 0.02f;);
Niveau 2: fréquence = 100; compteur = 5; missile niveau 2 (puissance = 3, vitesse = 0.002f);
Niveau 3: fréquence = 150; compteur = 3; missile niveau 3 (puissance = 5, vitesse = 0.001f);

La fréquence correspond au décompte du tick(), on tire lorsque f = 0;
Compteur = nombre de missiles;
Puissance = fait perdre la vie d'un astéroïde.
La vitesse correspond le pas de translation.

Il existe 3 niveau d'astéroïde:

Niveau 1: vie = 5; peut se dédoubler = non, vitesse = 0.001f;
Niveau 2: vie = 8; peut se dédoubler = vertical, vitesse = 0.0005f;
Niveau 3: vie = 5; peut se dédoubler = horizontale, vitesse = 0.005f;

Destruction de chaque objet:

Tour: lorsque l'astéroïde lui touche (l'astéroïde est lui même détruit).
Astéroide: lorsqu'il touche le mur, ou les missiles qui lui fait perdre la vie.
Missile: lorsqu'il touche le mur ou l'astéroïde.

Emplacement des tours

On ne peut placer qu'un tour par case et par ligne

Choix de conception

UML

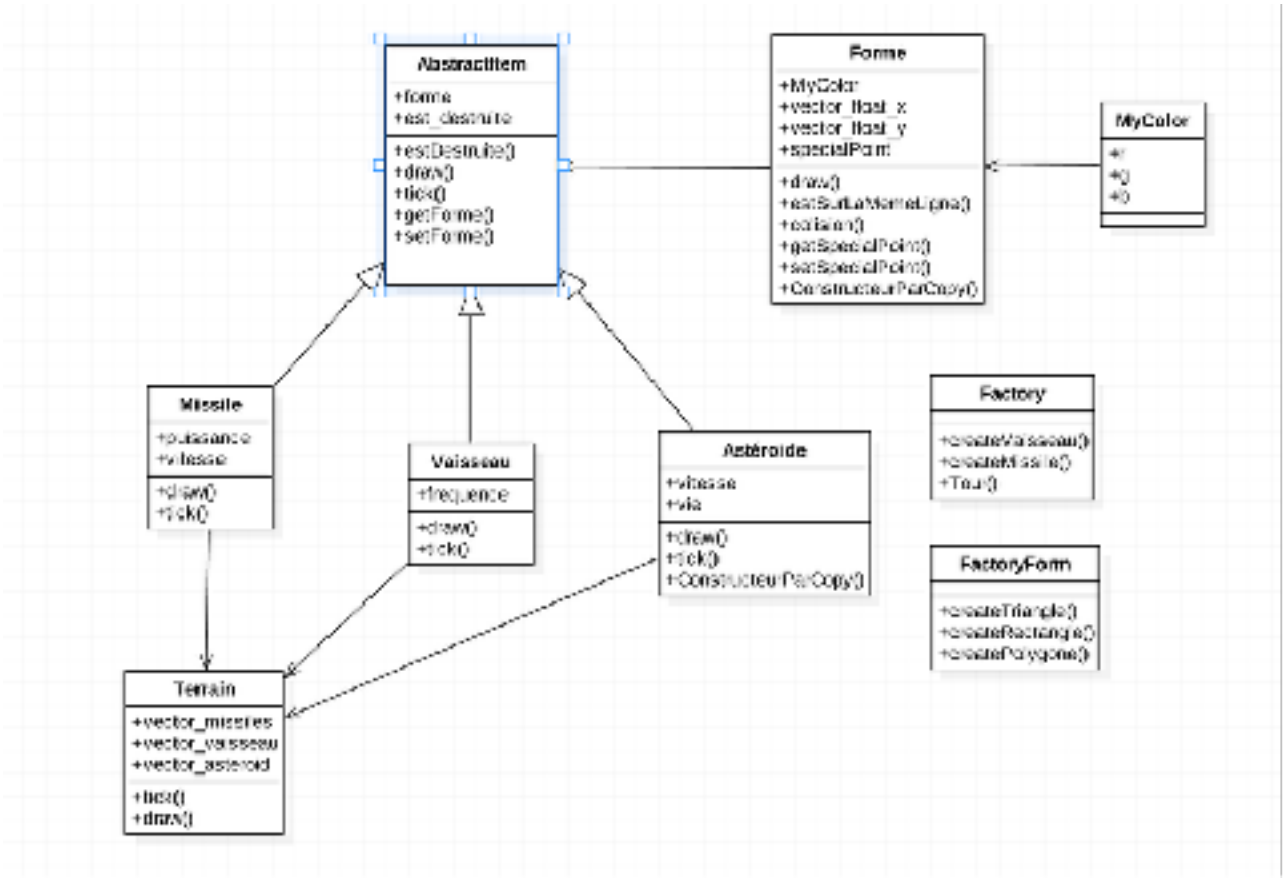


Diagramme UML non complet mais essentiel.

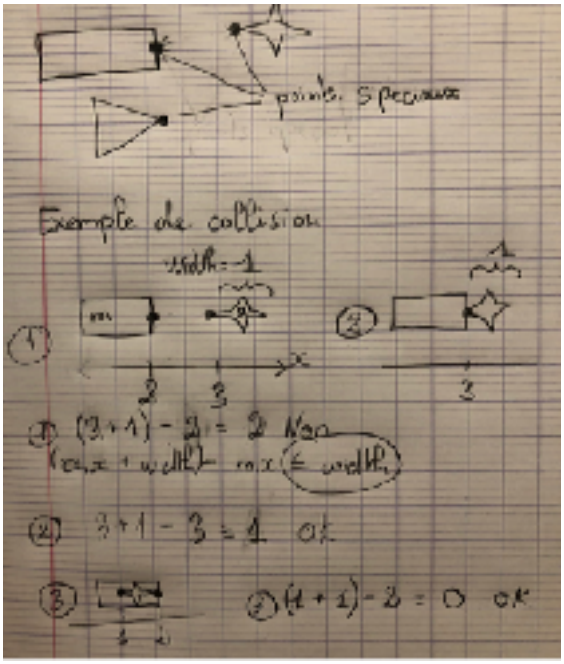
Classe Terrain

contient un vecteur de Missile/Vaisseau/Asteroid.
Il s'occupe de :

- exécuter tous les ticks/draws.
- faire le ménage des items détruits.
- Mettre à true les items détruits lors du collision: cette vérification se fait avec la fonction collision de la classe Forme. On vérifie s'il sont sur la même ligne, et s'il les points spéciaux sont très très proche.

Soit m missile, a astéroïde, width d'astéroïde: la collision se fait lorsque:

(A.x + width) - m.x <= width



Cette façon de gérer la collision gère aussi le cas 3 comme dans l'image, ça arrive quand le joueur de décide de placer un tour ou il y missile ou astéroïde, ou l'astéroïde qui apparait à l'endroit ou il y a missile.

- Gérer l'emplacement des vaisseaux: on vérifie avec sa forme (fonction estSurLaMêmeLigne()) s'il n'y a aucun vaisseau. Et on le place.
- Générer les astéroïdes: si c'est un astéroïde de niveau 1 ou 3 on le place à la ligne qu'on veut sur la dernière colonne. Si c'est niveau 2, on le place entre les lignes [2-9] et si en haut et en bas de la ligne généré ne contient pas d'astéroïde, comme ça on ne gère pas le doublement pendant le jeux.
- Gérer l'argent.
- Gérer la vie du joueur.
- Dédoubler les astéroïdes.

Classe Missile

- détruit s'il y a collision avec le mur
- Translate sa forme

Classe Vaisseau

- tire les missiles, et se détruit s'il y en a plus.

Classe Asteroid

- détruit s'il y a collision avec le mur
- Translate sa forme
- Se dédoubler: pour se dédoublé on utilise le constructeur par copie et puis on utilise les fonctions moveUP/moveDown/moveLeft pour la copie.

Classe Factory/FactoryForm

Ce sont des singletons utilisants le pattern Factory pour la fabrication des objets.

Une question de pointeur…et de cours…

Tout le jeu est gérer de façon dynamique, tout passe par pointeur, et donc new/delete systématiquement. Dans le constructeur par copie, on oublie pas de copier aussi les pointeurs autrement ça fait des segment fault…Pour éviter les copies, en utilise le passage par référence…