Space Invaders

PROJET C#

AUTEURS : ALEXANDRE MANETA

30/11/2019

Table des matières

Introduction

Ce projet de C# consiste à réaliser une réplique du jeu shoot'em up **Space Invader**. Nous avons eu le choix entre trois types de modèle : POO guidé, POO libre et Entité-composant-système. J'ai choisi de réaliser le projet en utilisant l'ECS.

Structure du programme

La solution C# est constituée de deux projets différents. L'un des projets correspond au framework ECS que j'ai développé. Le second projet correspond au Space Invader. Pour construire la solution puis la démarrer, il faut commencer par générer l'EXE du framework puis ensuite l'EXE du jeu. Une fois cette étape faite, il faut choisir le projet **Space Invader** comme projet de démarrage. Enfin, il suffit de lancer la solution.

Les projets utilisent des librairies externes pour fonctionner que l'on retrouve dans les fichiers « package ».

Le projet Space Invader est organisé en plusieurs dossiers : **components**, **systems**, **nodes**, **utils**. Cette organisation des fichiers est ainsi faite car elle permet de suivre les différents éléments que l'on retrouve dans le framework ECS.

Enfin, l'ensemble du projet est rassemblé dans une classe « Game » qui fait le lien entre le « form windows » et le jeu.

Le dossier « util » met à disposition des classes réutilisables pour la construction du jeu.

L'utilité des éléments du framework est décrite dans le code source.

Les diagrammes de classes sont disponibles dans le code source.

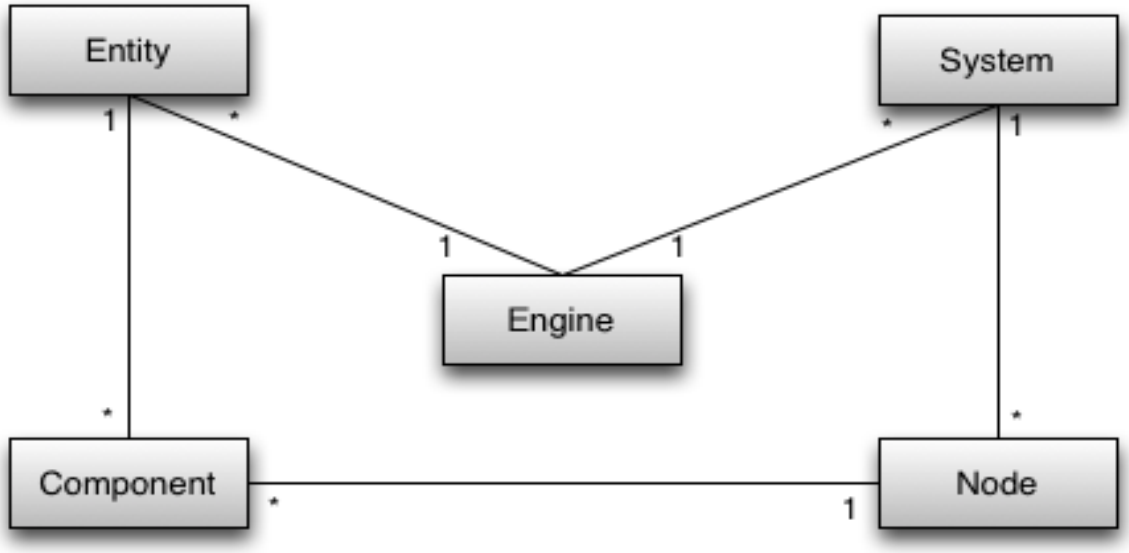

Figure 1 : Illustration du diagramme de classes de l'ECS (lien de l'illustration : https://www.richardlord.net/blog/ecs/why-use-an-entity-framework.html)

Informations complémentaires

Le jeu comporte une infinité de level ne se différenciant que par leur difficulté croissante. Tout d'abord, les différents types d'ennemis occasionnent une quantité de dégâts différente (rangée basse → 1, rangée du milieu → 2, rangée du haut → 3). A chaque niveau supérieur, 1 point de dégât leur est ajouté. Leur vitesse augmente également avec les niveaux.

J'ai également implémenté plusieurs types de bonus. La plupart d'entre eux ne nécessitent aucune action de l'utilisateur sauf celui du « missile téléguidé ». Ce missile se dirige à l'aide des touches « z,q,s,d ».

La gestion du shoot a été altérée afin que le jeu soit plus agréable à utiliser. L'attente pour pouvoir tirer n'est pas adéquat avec le système de niveau mis en place. En effet, la difficulté croit après chaque niveau. Si la contrainte avait été gardée, il aurait été impossible de pouvoir jouer après le niveau 3. Par conséquent. Il est donc possible de tirer plusieurs missiles à la fois.

Problèmes rencontrés

Tout au long du projet, j'ai fait face à plusieurs problèmes. Ils prenaient en grande partie source dans le framework ECS :

- Problème de référence sur objet → *Solution :* effectuer des copies profondes d'objet
- Passage de classe en paramètre compliqué → *Solution partielle* : utilisation d'ID permettant d'identifier chaque classe. Il suffit alors de passer une instance de la classe pour que l'on puisse effectuer le traitement adéquat.
- Pas de PriorityList → *Solution :* utilisation d'un « OrderedBag » provenant d'une source externe. Ainsi, il est possible d'ordonnancer l'exécution des différents systèmes.
- Impossibilité de passer l'objet graphique dans les champs d'autres classes → *Solution* : passage du graphics par paramètre de méthodes
- Debug difficile avec le framework ECS → *Pas de solution*
- Gestion des conversions parfois capricieuse → *Solution* : être vigilant et utiliser les outils de debug que Visual Studio propose (breakpoints, exécution pas à pas)
- Timer animation → *Solution* : les animations sont pour la plupart du temps utilisées pour la mort des entités. Quand une entité est morte elle est détruite du l'engine. Or, pour que l'animation se termine, il faut ajouter le composant « d'animation » puis que le système d'animation se lance. Il faut donc mettre en place un système de priorité qui fasse en sorte de lancer les animations puis ensuite détruire les entités.