Marc FURLAN

19/04/2018

DOCUMENTATION TECHNIQUE DEVELOPPEUR

Projet GALACTIC CLASH

# OBJECTIFS PRINCIPAUX DE L’APPLICATION :

L’application GALACTIC CLASH est un jeu de type Shoot’em Up au design et gameplay inspirés des classiques des années 80 et 90. Aux commandes d’un vaisseau spatial, vous devez défendre la Terre contre la menace extra-terrestre menée par le terrifiant Empereur Galactique.

Au cours de l’aventure, le joueur pourra :

* Contrôler le vaisseau sur l’écran, tirer avec plusieurs armes
* Parcourir plusieurs niveaux à la difficulté croissante et se mesurer au boss final
* Enregistrer son score et son nom dans une base de données
* Consulter les 6 meilleurs scores récupérés dans la base de données

# CONTEXTE D’UTILISATION :

Le jeu est accessible à tous les utilisateurs de par son principe très simple (on tire sur tout ce qui bouge) qui rend la prise en main très rapide.

Cependant, la difficulté assez élevée rend le jeu plus accessible aux joueurs familiarisés avec le genre et son action frénétique.

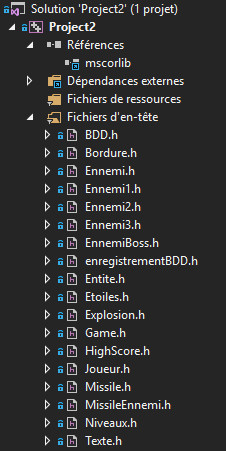
L’application a été développée pour un environnement Windows.

Le jeu est jouable au clavier ou à la manette. Un fichier .txt fourni avec l’installeur donne de plus amples détails quant aux différentes touches utilisées pour les commandes.

# MODULE, CLASSES ET LIBRAIRIES :

L’application a été développée en C++ avec Visual Studio 2015 et 2017. J’ai utilisé la librairie graphique SFML 2.4.2.

**Différentes classes utilisées :**



BDD : Classe gérant la création de la BDD

Bordure : Classe permettant de gérer les bordures de l'écran

Ennemi : Classe de création des vaisseaux ennemis

EnregistrementBDD : Cette classe permet de créer un objet qui facilite la récupération des données dans la BDD

Entite : Cette classe est la base de la plupart des objets créés dans le jeu. Elle possède un sprite et une forme

Etoiles : Permet la création des étoiles pour le background

Explosion : Gestion des explosions du joueur et de la méga bombe

Game : Cette classe gère la logique des événements du jeu et l'affichage

HighScore : Création de la table de score

Joueur : Classe de création du joueur

Missile : Projectiles du joueur (2 types, missile et canon)

MissileEnnemi : Classe de création des projectiles des ennemis

Niveaux : Classe de création des niveaux faisant office d'éditeur de niveaux

Texte : Classe de création des texte SFML à afficher à l'écran

Les classes principales sont la classe Game, qui gère les états de jeu, toute la logique et l’affichage, et la classe Niveaux qui permet de mettre en place les différents événements, l’apparition des ennemis et une partie de la logique liée à la création des niveaux.

# OUTILS ET BASE DE DONNEES :

L’installeur a été réalisé avec InnoSetup version 5.5.9.

J’ai utilisé l’outil de versionning GitHub. Voici un lien vers le dépôt qui contient le code ainsi que l’ensemble des documents demandés pour le projet 2 :

<https://github.com/mmfurlanmm/Projet2>

Pour plus de précisions quant aux différentes classes, vous pouvez consulter la documentation générée par Doxygen directement depuis le code dans le dossier « Documentation » se trouvant sur le dépôt Git. Merci de suivre le lien ci-dessus.

La base de données est une base locale réalisée avec SQLite version 3.23.1. Comme annoncé dans le dossier d’analyse, la base de données est très simple car ne sert qu’à enregistrer le nom des joueurs (en 3 caractères) et leur score. Voici le schéma relationnel :

