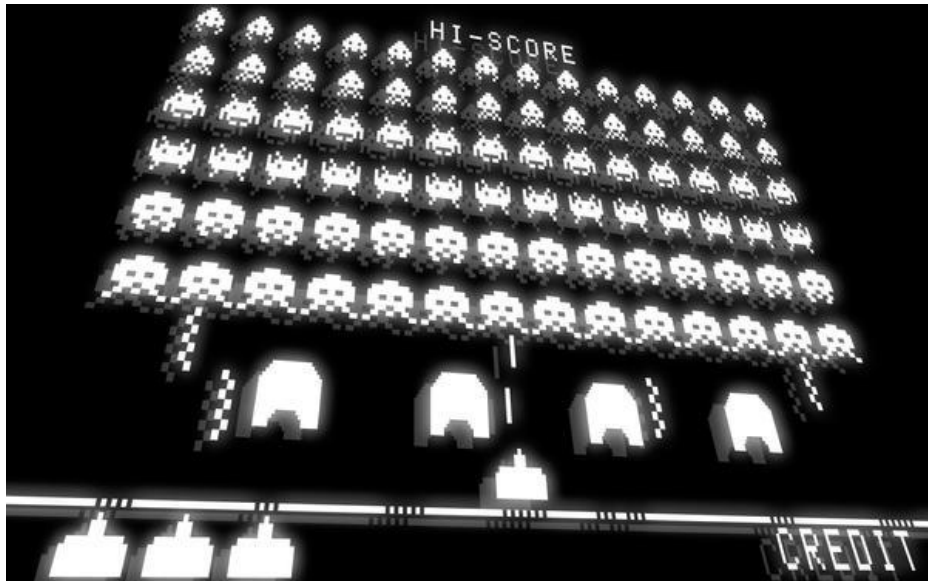


Shoot me up !



Bastien Segalen – MID2B
ETML - Vennes
80p
Curchod – Mveng - Melly

Table des matières

1	SPÉCIFICATIONS.....	3
1.1	TITRE	3
1.2	DESCRIPTION	3
2	PLANIFICATION INITIALE.....	3
3	RÉALISATION	3
3.1	DOSSIER DE RÉALISATION	3
3.1.1	Module 106 (DB) :	3
3.1.2	Module 320 (POO) :	4
3.1.3	Module 322 (UX) :	7
4	CONCLUSION.....	10
4.1	BILAN DES FONCTIONNALITÉS DEMANDÉES	10
4.2	BILAN DE LA PLANIFICATION.....	10
4.3	BILAN PERSONNEL	10
5	DIVERS.....	10
5.1	JOURNAL DE TRAVAIL	10
5.2	IA.....	10

1 SPÉCIFICATIONS

1.1 Titre

Shoot me up !

1.2 Description

Concevoir un jeu 2D modulaire de tir à la troisième personne et en réaliser une partie.

2 PLANIFICATION INITIALE

[T-106-BastienSegalen-jdt-planif-DB.xlsm](#)

[T-320-BastienSegalen-jdt-planif-POO.xlsm](#)

[T-322-BastienSegalen-jdt-planif-UX.xlsm](#)

3 RÉALISATION

3.1 Dossier de Réalisation

3.1.1 Module 106 (DB) :

Auto-évaluation DB : [auto-evaluation-DB.xlsx](#)

MCD et MLD: [MCD-MLD\mcd-shootmeup.loom](#)

Fichier de création des données : [MCD-MLD\shootmeup-donnees.sql](#)

Fichier de création de la base de données : [MCD-MLD\db_shootmeup.sql](#)

Utilisateurs :

Fichier de création des utilisateurs et des rôles :

[MCD-MLD\shootmeup_create_users.sql](#)

Administrateurs : créer, lire, mettre à jour, supprimer toutes les tables.

Droits : ALL PRIVILEGES sur toutes les tables, GRANT OPTION

Gestionnaire du jeu : lire toutes les tables et mettre à jour les highscores.

Droits : SELECT sur toute les tables, INSERT sur les tables niveau (pour l'éditeur de niveau) et highscores.

INDEX :

- 1) Index sur les clés primaires, les clés étrangères et les champs « unique »
- 2) Plus rapide pour les requêtes, créer les index prend du temps (insertion de données)
- 3) Sur le champ « score » de la table « t_highscores »

BACKUP/RESTORE :

Fichier dump de la base de données : [MCD-MLD\db_shootmeup_dump.sql](#)

- 1) Backup : Ouvrir un cmd à l'endroit où le dump doit être stocké, exécuter la commande :
`docker exec -i id_du_container mysqldump -u root -proot db_shootmeup > db_shootmeup_dump.sql`
- 2) Restore : Une fois que la base de données est supprimée, se connecter à mysql et exécuter cette commande :
`CREATE DATABASE db_shootmeup ;`
Ensuite, ouvrir un cmd à l'endroit où se trouve le dump, et exécuter cette commande :
`docker exec -i id_du_container mysql -u root -proot db_shootmeup < db_shootmeup_dump.sql`

3.1.2 Module 320 (POO) :

Description du shoot me up :

On peut bouger le vaisseau dans toutes les directions, les ennemis descendent en ligne et en tirant des lasers. Quand on tue les ennemis, on peut obtenir un power up, qui est soit un tir transperçant qui peut tuer plusieurs ennemis, soit un power up qui désactive le rechargement, on peut donc tirer très vite. Il y a des obstacles pour pouvoir se protéger des tirs ennemis.

Niveaux : Il y a deux niveaux avec un nombre fixe d'ennemis et un niveau "survie" qui est infini.

On peut tuer les ennemis avec nos tirs ou avec notre vaisseau, au coût d'une vie. Les obstacles peuvent être détruits par les tirs adverses, par nos tirs ou quand un ennemi touche un obstacle.

Description des scénarios de tests :

- Test si le missile est supprimé quand il sort de l'écran :
Arrange : On crée un niveau, sans aucun obstacle ou ennemi. Ensuite, on crée un missile qui est en dehors de l'écran et on l'ajoute dans la liste de missiles.
Act : On appelle la méthode `Timer_Tick`, qui s'occupe de supprimer les missiles quand ils sont en dehors de l'écran.
Assert : On utilise la méthode `Assert.IsFalse` pour vérifier qu'il n'y a plus aucun missile dans la liste de missiles.
On fait ce test car il est important que le missile soit supprimé, car sinon on déplace le missile inutilement, ce qui fait que le jeu devient injouable une fois qu'on a tiré beaucoup de missiles.
- Test si le joueur perd de la vie quand un ennemi sort de l'écran :
Arrange : On crée un niveau, sans aucun obstacle ou ennemi. Ensuite, on crée un ennemi en dehors de l'écran et on l'ajoute à la liste d'ennemis.
Act : On appelle la méthode `Timer_Tick`, qui s'occupe aussi de supprimer les ennemis quand ils sortent de l'écran et d'enlever une vie au joueur.
Assert : On utilise la méthode `Assert.AreEqual` pour vérifier que `shipLife` et 2 sont égaux (2 car le vaisseau commence le jeu avec 3 vies).
On fait ce test car le fait de perdre une vie quand un ennemi sort de l'écran est une des fonctionnalités du jeu, et donc on veut qu'elle fonctionne sinon le jeu ne serait pas assez difficile.
- Test si les ennemis sont supprimés quand ils sont touchés par un missile :

Arrange : Comme les deux tests précédents, on crée un niveau. Ensuite, on crée un ennemi et un missile qui ont la même position. On ajoute le missile dans la liste de missiles et l'ennemi dans la liste d'ennemis.

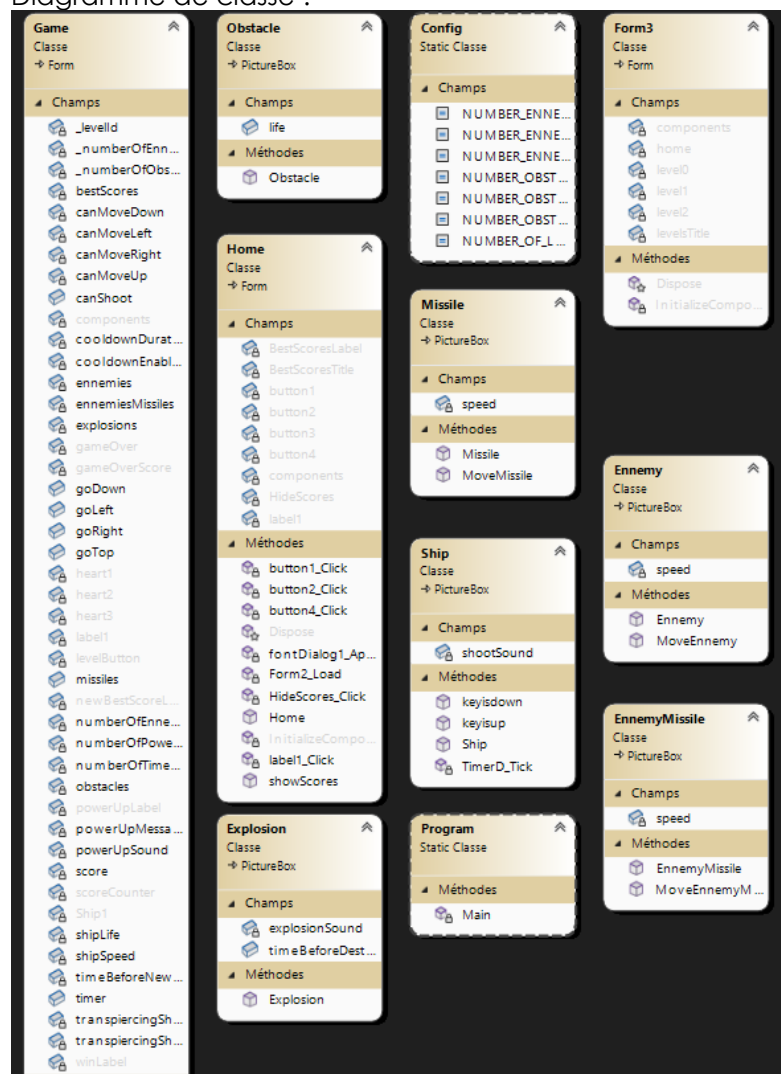
Act : On appelle la méthode Timer_Tick car elle supprime les missiles et ennemis quand ils entrent en collision.

Assert : On utilise la méthode Assert.IsFalse pour vérifier que la liste de missile ne contient pas de missile.

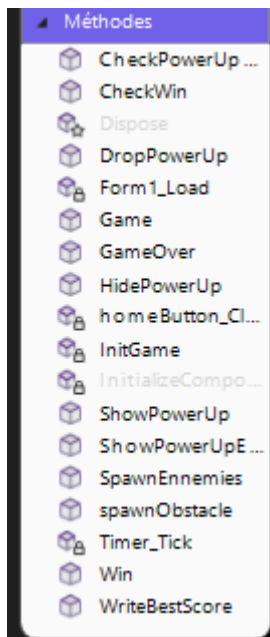
On fait ce test car si les ennemis ne sont pas supprimés quand ils sont touchés par un missile, on ne pourrait pas gagner.

- Test si les missiles sont supprimés quand ils touchent un obstacle :
 Arrange : On crée un niveau, et un missile et un obstacle qui ont la même position. On ajoute le missile dans la liste de missiles et l'obstacle dans la liste d'obstacle.
 Act : On appelle Timer_Tick, qui supprime les missiles quand ils touchent un obstacle.
 Assert : On utilise la méthode Assert.IsFalse pour vérifier que la liste de missiles ne contient pas de missile.
 On fait ce test car le fait de ne pas pouvoir tirer à travers les obstacles est une des fonctionnalités du jeu, on veut qu'elle fonctionne sinon le jeu serait trop facile.

Diagramme de classe :



Méthode de Game :



Détail d'implémentation spécifique :
Création d'un fichier de sauvegarde :

```
//create the save file if it doesnt exists
if (!File.Exists(".././../Ressources/score.txt"))
{
    bestScores = new string[Config.NUMBER_OF_LEVELS];
    //replace the default null creation value by 0
    for (int i = 0; i < Config.NUMBER_OF_LEVELS; i++)
    {
        if (bestScores[i] == null)
        {
            bestScores[i] = "0";
        }
    }
    File.AppendAllLines(".././../Ressources/score.txt", bestScores);
}
bestScores = File.ReadAllLines(".././../Ressources/score.txt");
```

On regarde si le fichier existe déjà. S'il n'existe pas, on crée un tableau dont la taille est le nombre de niveau. Ensuite on remplit le tableau de zéro pour éviter des problèmes à la fin du niveau. Ensuite on écrit le tableau dans le fichier avec la méthode `File.AppendAllLines()`, qui crée aussi le fichier s'il n'existe pas. Et enfin on stocke les scores dans un tableau pour pouvoir comparer les scores plus tard.

Gestion des collisions entre le vaisseau et les obstacles :

```

if (goRight)
{
    if (!(this.Ship1.Left > this.Width - Ship1.Width))
    {
        canMoveRight = true;
        foreach (var obstacle in obstacles)
        {
            if (obstacle.Bounds.Intersects(Ship1.Bounds))
            {
                canMoveRight = false;
                this.Ship1.Left -= shipSpeed;
                break;
            }
        }
        if (canMoveRight)
        {
            this.Ship1.Left += shipSpeed;
        }
    }
}

```

Quand on presse sur D, goRight = true.

Si le vaisseau n'est pas en dehors, on met canMoveRight à true.


Ensuite on regarde si le vaisseau est en collision avec les obstacles.

S'il n'y a pas de collision canMoveRight sera true et le vaisseau pourra bouger.

S'il y a une collision, on met canMoveRight à false, on déplace le vaisseau pour qu'il n'y ait pas de collision et on sort de la boucle.

3.1.3 Module 322 (UX) :

Personnas : [UX\personnas.txt](#)



Alex Martin

AGE 29

STATUS Marié

OCCUPATION Ingénieur en informatique

LOCATION Crissier, Vaud, Suisse

NIVEAU DE TECHNOLOGIE Haut

“ J'adore la sensation de battre un niveau difficile après des heures de pratique. Space Invaders me rappelle pourquoi j'ai commencé à jouer aux jeux vidéo.”

Personnalité

Sociable Curieux

Déterminé

Objectifs de vie

Devenir un expert reconnu dans le domaine de l'intelligence artificielle.

Voyager à travers le monde pour découvrir de nouvelles cultures et technologies.

Créer une startup innovante dans le secteur des jeux vidéo.

Motivations

- La nostalgie des jeux d'arcade classiques.
- Le défi de battre ses propres records et ceux des autres joueurs.
- La satisfaction de maîtriser un jeu difficile.

Frustrations

- Les niveaux trop faciles qui ne présentent pas de défi.
- Les bugs ou problèmes techniques qui interrompent le jeu.
- Les adversaires qui utilisent des tricheurs ou des hacks.

Applications favorites


Discord GitHub Twitch

Platform

Téléphone: iPhone 13 Pro

ordinateur: MacBook Pro M1

Emma Dupont



AGE

24

STATUS

Marriée

OCCUPATION

Étudiante en astrophysique

LOCATION

Lausanne, Vaud, Suisse

NIVEAU DE TECHNOLOGIE

moyen

“ L'univers est vaste et plein de mystères. Jouer à Space Invaders me rappelle que même les plus grands défis peuvent être surmontés avec persévérance. ”

Personnalité

Sociable

Curieuse

Déterminée

Motivation

Obtenir un doctorat en astrophysique.
Travailler pour une agence spatiale comme l'ESA ou la NASA.
Voyager dans des observatoires du monde entier pour observer les étoiles et les galaxies.




Core needs

- Défi : La difficulté croissante des niveaux la pousse à s'améliorer constamment.
- Nostalgie : Le jeu lui rappelle les classiques de l'arcade qu'elle jouait enfant.
- Compétition : Elle aime comparer ses scores avec ceux de ses amis et de la communauté en ligne.

Frustrations

- Répétitivité : Les niveaux peuvent parfois sembler répétitifs après de longues sessions de jeu.
- Difficulté : Certains niveaux peuvent être extrêmement difficiles, ce qui peut être frustrant.
- Limites techniques : Les graphismes et les mécanismes de jeu peuvent sembler datés par rapport aux standards modernes.


Applications favorites



Platform

Téléphone: Samsung Galaxy S21
ordinateur: Dell XPS 15

Lucas Moreau



AGE

27

STATUS

Célibataire

OCCUPATION

Musicien (guitariste et compositeur)

LOCATION

Nantes, France

NIVEAU DE TECHNOLOGIE

Moyen

“ J'adore la simplicité de Space Invaders. Les couleurs contrastées et le gameplay répétitif permettent de vraiment me concentrer et me plonger dans le jeu, même si je ne vois pas tout parfaitement. ”

Personnalité

Créatif

Patient

Persévérant

Objectifs de vie

Devenir un musicien reconnu, en particulier dans le domaine de la musique de film.
Organiser une tournée internationale pour jouer sa propre musique.
Créer un studio d'enregistrement accessible à tous, peu importe leurs handicaps.

Motivations




- Le plaisir de jouer à Space Invaders, qui propose une interface simple et un gameplay accessible.
- Le besoin de se détendre et de se divertir après des sessions d'enregistrement.
- La satisfaction de maîtriser des jeux rétro grâce à des adaptations spécifiques.

Frustrations

Les jeux vidéo qui ne prennent pas suffisamment en compte les besoins des malvoyants, avec des contrastes faibles ou des interfaces trop complexes.

- Les contrôles tactiles ou les systèmes de navigation non adaptés.
- Les descriptions sonores ou auditives manquantes dans les jeux vidéo, rendant certains jeux inaccessibles.

Applications favorites



Platform

Téléphone : iPhone SE avec des réglages d'accessibilité activés (agrandissement, voix-off, etc.).
Ordinateur : MacBook Air avec logiciels d'accessibilité comme VoiceOver.

Description du shoot me up :

Un mode "histoire" composée de plusieurs niveaux de +en + difficile, et un mode "survie" qui est infini, devenant plus dur + on survit

Si on gagne un niveau, on gagne de l'or qu'on pourra utiliser pour acheter des améliorations.

Éco conception : 115 bonne pratique :

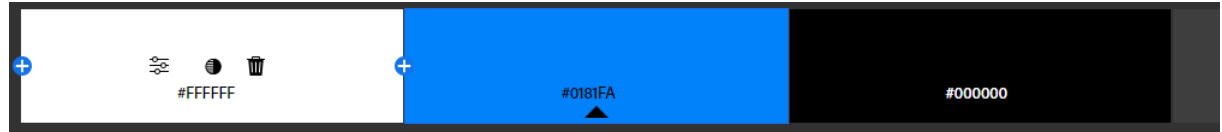
5 : Optimiser le parcours utilisateur :

Il y a peu de bouton, les interface sont simple et facile à comprendre.

Accessibilité :

Fonctionnalité d'accessibilité mise en place : contraste élevé pour les personnes malvoyantes.

Palette de couleur :



Élément original : Arbre de compétence :

Les joueurs peuvent acheter des améliorations pour leur vaisseau avec des pièces qu'ils obtiennent en gagnant des niveaux.

Pour acheter une amélioration on doit avoir acheter la précédente, et les améliorations ne sont pas cumulative, par exemple si ont à acheter +1 vie et +2 vies, on aura deux vies supplémentaires et pas trois.

Conception :

Maquettes hautes et basse fidélité : [UX\shoot-me-up.pdf](#)

Prototype cliquable Figma : [UX\shoot-me-up.fig](#)

Choix effectués :

- Écran d'accueil : Il y a un titre et quatre boutons : Jouer, score, éditeur de niveau et paramètres. L'écran est simple et l'utilisateur comprendra facilement ce qu'il peut faire.
- Écran de scores : Il y a un tableau de score avec pour chaque ligne place, nom et score. Il y a un bouton pour changer de niveau et un pour revenir à l'accueil.
- Écran jouer : L'écran est simple, il y a seulement des boutons pour sélectionner le niveau qu'on veut jouer et un pour aller sur l'écran d'amélioration.
- Écran amélioration : Il y a l'argent qu'on a et les boutons pour acheter des compétences.
- Écran de paramètres : On peut activer le mode contraste élevé pour les personnes malvoyante et on peut changer de nom.
- Écran éditeur de niveau : Sur la gauche il y a la zone des paramètres, où on peut changer les paramètres du vaisseau, des ennemis et des obstacles. La zone est divisée en plusieurs parties (joueur, obstacle, etc.) et on peut réduire les zones si on n'en a pas besoin. Si on a toutes les zones dépliées, le contenu sera trop grand pour être affiché et on pourra scroller pour accéder aux options qui sont en bas. La partie droite est la zone de jeu, où on peut voir à quoi ressemblera le niveau et on peut aussi changer la position du vaisseau et des obstacles, et leur taille. En bas à gauche il y a deux boutons : tester le niveau et publier le niveau. Quand on clique sur publier le niveau, une fenêtre s'ouvre et on peut donner un nom à notre niveau et on doit sélectionner la difficulté du niveau. Il y a aussi un bouton pour revenir à l'accueil et un autre pour fermer la zone des options.

4 CONCLUSION

4.1 Bilan des fonctionnalités demandées

Le jeu a toutes les fonctionnalités demandées, il y a 3 niveau avec un joueur, des ennemis, des obstacles, différent mode de tir. Il y a aussi un système de sauvegarde des meilleurs scores.

Pour la partie POO, il y a des explications sur les tests unitaires, le schéma des classes et deux détails d'implémentation spécifique.

Pour la partie UX, il y a les différentes maquettes demandées, trois persona dont un qui a des besoins d'accessibilité, l'écran éditeur de niveau en haute fidélité, un élément original et un prototype cliquable Figma. Dans le rapport il y un chapitre sur la palette graphique, un sur l'éco-conception et un sur l'accessibilité.

Pour la partie DB, il y a le script de création de la base de données, un dump de la base de données, un mcd, un script de création des utilisateurs et un fichier avec des données pour la base de données.

Donc toutes les fonctionnalités demandées ont été faites.

4.2 Bilan de la planification

La planification originale a pris plus de temps que prévus et le rapport a pris moins de temps que prévu.

4.3 Bilan personnel

Si c'était à refaire : prendre moins de temps sur la planification initiale.

Ce projet m'a appris Windows form, a permis de m'améliorer dans les tests unitaires.

Suite à donner : Changer le message de power up pour qu'il soit plus visible.

5 DIVERS

5.1 Journal de travail

[T-106-BastienSegalen-jdt-planif-DB.xlsm](#)

[T-320-BastienSegalen-jdt-planif-POO.xlsm](#)

[T-322-BastienSegalen-jdt-planif-UX.xlsm](#)

5.2 IA

L'IA a été utilisé pour la réalisation des personnas. (IA utilisé : Copilot et ChatGPT)
ChatGPT a été utilisé pour générer des données pour la base de données.