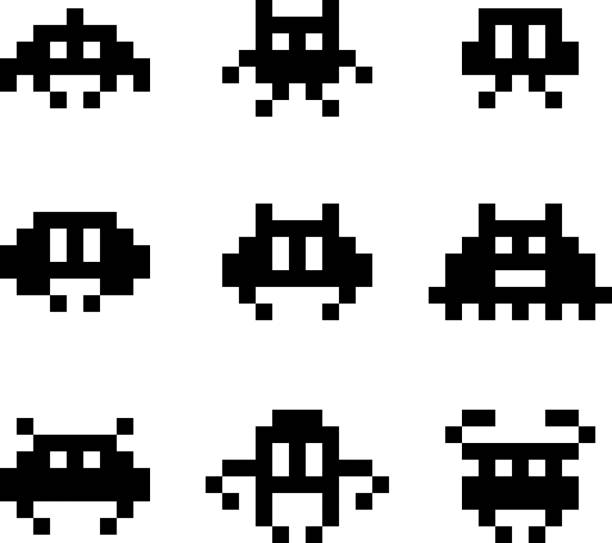
Jeux Type Spac Invaders



Thomas Reynaud

Mid 2A

ETML

28.10.2025

Introduction

Durant ce projet, nous avions le choix de créer un jeu basé sur le concept de Space Invaders.  
J’ai donc décidé de réaliser un jeu dans le même esprit, où des Yoshi nous attaquent et où le but est de les empêcher soit de nous toucher, soit d’atteindre le bas de l’écran. Si cela se produit, c’est Game Over.

Dans le jeu, le joueur peut poser jusqu’à 5 obstacles maximum. Lorsqu’un ennemi entre en contact avec l’un de ces obstacles, il est immédiatement éliminé.  
Chaque Yoshi détruit rapporte 10 points supplémentaires au score : l’objectif est donc d’en éliminer le plus possible.

Le jeu est entièrement codé en C# à l’aide du framework MonoGame, afin de simplifier le développement et d’optimiser les performances.  
J’utiliserai les technologies et notions apprises en cours, en les adaptant à l’environnement MonoGame.  
Enfin, je veillerai à respecter le cahier des charges tout au long du projet.

Plannification & Analyse Fontionnelle

Il n’y a pas eu de planification particulière, car tout est déjà présent sur le projet GitHub, avec les user stories, les maquettes, et les autres éléments nécessaires.

C’est pourquoi je ne mets rien de spécifique à ce sujet dans le rapport.

Conception

Une image contenant texte, diagramme, capture d’écran, Plan

Le contenu généré par l’IA peut être incorrect.

Durant le projet, j’ai réalisé un code qui fonctionne correctement.

Tout d’abord, j’ai créé les personnages principaux, à savoir le joueur et les ennemis. Le joueur peut se déplacer horizontalement, de gauche à droite, en bas de l’écran, tandis que les ennemis apparaissent en haut et descendent verticalement.

J’ai ensuite ajouté plusieurs fonctionnalités, comme le tir du personnage principal, qui lui permet d’éliminer les ennemis représentés par des Yoshi.

J’ai également créé une classe EnnemyManager chargée de gérer le spawn massif des ennemis, afin d’augmenter la difficulté du jeu.

Puis, j’ai ajouté une classe Obstacle, permettant au joueur de poser jusqu’à cinq obstacles. Ces obstacles détruisent automatiquement un ennemi en cas de collision.

Enfin, j’ai intégré quelques fonctionnalités supplémentaires : le joueur dispose de 3 vies, un score augmente de 10 points pour chaque ennemi éliminé, et la partie se termine (Game Over) si un ennemi touche le joueur ou atteint le bas de l’écran.

IA

Durant ce projet, j’ai eu recours à l’IA à plusieurs reprises.

Étant donné que j’ai utilisé un framework tout nouveau pour moi (MonoGame), ChatGPT m’a beaucoup aidé à en comprendre les bases, car la documentation officielle est assez vague.

L’IA m’a également permis de détecter et comprendre certaines erreurs que je ne parvenais pas à résoudre seul.

À plusieurs moments, j’ai rencontré des bugs et, grâce à l’IA, j’ai pu les analyser et les corriger plus facilement.

Je m’en suis aussi servi pour trouver de l’inspiration et obtenir des idées d’améliorations à apporter au jeu.

Elle m’a particulièrement aidé pour la création de la classe EnnemyManager, car cette partie du code me paraissait assez floue au début.

En résumé, l’IA m’a servi comme un véritable coéquipier : elle m’a aidé à corriger mes fautes d’orthographe dans ce rapport, et à apporter une petite touche supplémentaire à mon projet.