P\_OO - Shoot me up !

Une image contenant texte, dessin humoristique, illustration

Le contenu généré par l’IA peut être incorrect.

Kiril Bormin

ETML

27.08.2025/31.10.2025

Xavier Carrel

Table des matières

[1 Introduction 3](#_Toc212218276)

[1.1 Titre 3](#_Toc212218277)

[1.2 Description 3](#_Toc212218278)

[2 Planification 3](#_Toc212218279)

[2.1 Chronologie du projet 3](#_Toc212218280)

[2.2 Outils 3](#_Toc212218281)

[2.3 Tableau de planification 3](#_Toc212218282)

[3 Analyse fonctionnelle 3](#_Toc212218283)

[3.1 Objectifs du jeu 3](#_Toc212218284)

[3.2 Contrôle du joueur 3](#_Toc212218285)

[3.3 Ennemies 5](#_Toc212218286)

[3.4 Obstacles 5](#_Toc212218287)

[3.5 Système de score 5](#_Toc212218288)

[4 Conception 6](#_Toc212218289)

[4.1 Description 6](#_Toc212218290)

[4.2 UML 6](#_Toc212218291)

[4.3 Explication du diagramme 7](#_Toc212218292)

# Introduction

## Titre

Shoot me up !

## Description

Ce projet, réalisé dans le cadre du module P\_OO , consiste à développer un jeu 2D de type "Shoot me up". L'objectif principal est de créer une application modulaire et fonctionnelle en C#, en respectant le cahier des charges. Le thème est une bataille aérienne où le joueur pilote un avion, doit détruire des ennemis et éviter des obstacles pour obtenir le meilleur score.

# Planification

## Chronologie du projet

Le projet se déroulera sur le premier trimestre et comprendra un total de 32 périodes (soit 45 X 32 = 24 heures de travail)

## Outils

* J’ai tenu un journal de travail, utilisant **un modèle en Excel.**
* J’ai utilisé **Microsoft Visual Studio** : L'environnement de développement principal pour écrire le code C#.
* J’ai utilisé le système de versioning **git** pour sauvegarder les versions et diminuer les risques de perte de données.
* J’ai utilisé **GitHub Project** pour créer et gérer mes user stories.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Semaine 1 | Semaine 2 | Semaine 3 | Semaine 4 | Semaine 5 | Semaine 6 | Semaine 7 | Semaine 8 |
| Semaine d’accueil | Préparation de maquette, environnement Git et création des User’s Storys | Choix des texture et déplacement du joueur | Lancement des missiles | Score et vie du joueur | Collision des objets | Création d’obstacles | Rendu du projet |

## Tableau de planification

# Analyse fonctionnelle

## Objectifs du jeu

"Shoot Me Up" est un jeu de tir spatial en 2D amusant et rapide. Le but du jeu est de détruire les ennemis, éviter les obstacles et gagner le plus de points possibles en touchant les cibles.

## Contrôle du joueur

* Le joueur contrôle un avion à l’aide des flèches directionnelles ou avec les

touches « WAD », quand l’avion tourne dans une sens, il y pivote légèrement. Quand on relâche le bouton, l’avion s’aligne droit

Une image contenant croquis, dessin, art

Le contenu généré par l’IA peut être incorrect.Une image contenant croquis, dessin, skier

Le contenu généré par l’IA peut être incorrect.

* Le joueur peut tirer des missiles en appuyant sur l’espace, il a une charge maximale possible de 5 missiles, mais s’il en a moins de 5, il en recharge avec la vitesse de 1 missile chaque 5 secondes.

Une image contenant croquis, dessin, art, conception

Le contenu généré par l’IA peut être incorrect.

## Ennemies

Les ennemis (F16) apparaissent en haut de l'écran et se déplacent vers le bas.

* Points de vie : Chaque ennemi a 3 points de vie (\_hp = 3 ).
* Collision : Un missile du joueur enlève 1 point de vie. Si le joueur entre en collision avec un ennemi, les deux sont détruits.Une image contenant croquis, dessin, art

  Le contenu généré par l’IA peut être incorrect.

## Obstacles

Les obstacles (Tour cellulaire) descendent également vers le bas, mais plus lentement que les ennemis (avec la même vitesse que le fond, que donne l’impression qu’ils ne bougent pas )

* Points de vie : Ils ont 3 points de vie (\_hp = 3 ) et peuvent être détruits par les missiles du joueur.
* Collision : Une collision avec un obstacle est fatale et détruit le joueur

## Système de score

Le joueur gagne 1 point pour chaque avion ennemi détruit. Le score actuel est affiché en temps réel en haut à gauche de l'écran.

Une image contenant croquis, dessin, art

Le contenu généré par l’IA peut être incorrect.

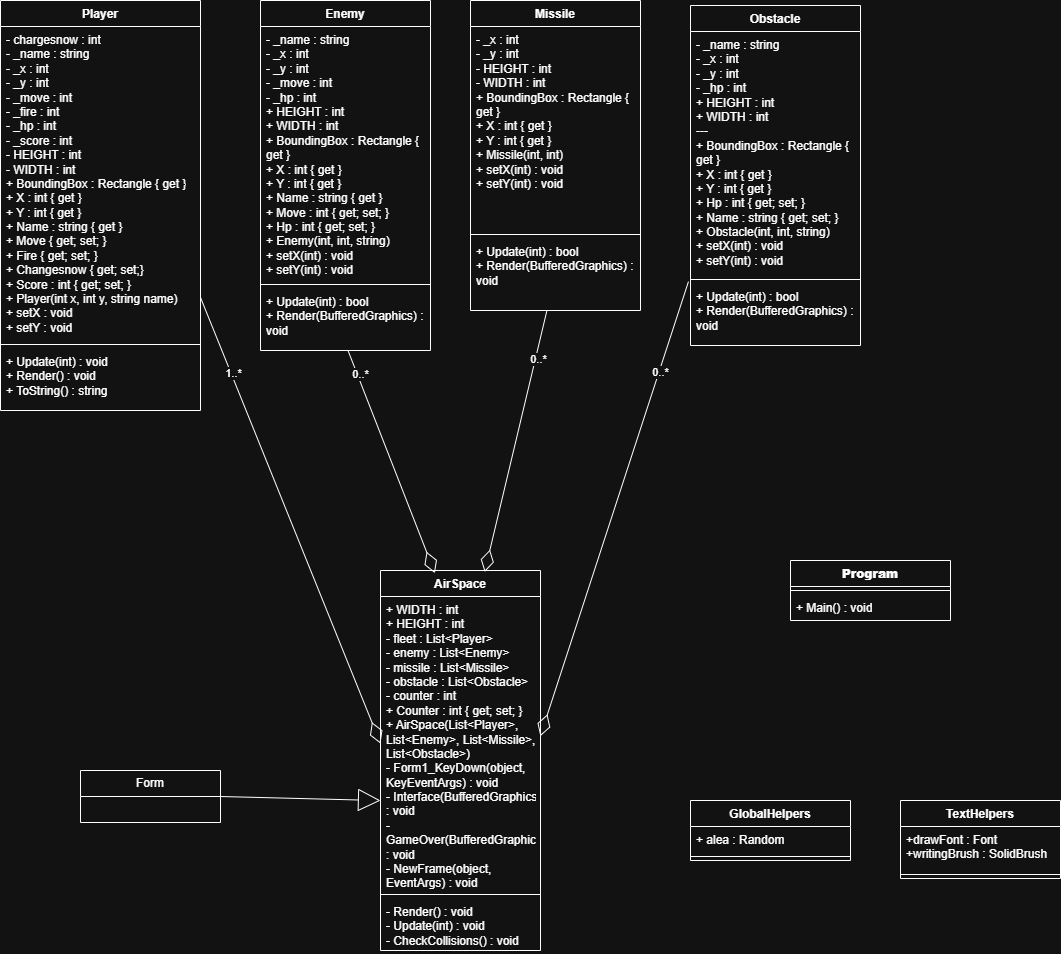
# Conception

## Description

Cette section explique la structure technique du projet, comme demandé par le cahier des charges. La conception est basée sur les principes de la programmation orientée objet (POO) pour créer un code modulaire, organisé et facile à maintenir.

## UML

Le diagramme UML ci-dessous montre les différentes classes du projet et leurs relations.



## Explication du diagramme

* AirSpace : C'est la classe la plus importante, (un peu comme le cerveau du jeu). Il hérite de Form (la flèche triangle ). C'est lui qui tourne en boucle et qui gère la logique principale, comme Render et Update de tous les objets.
* Relation d'Agrégation : AirSpace a des listes pour garder le Player, les Enemy, les Missile et les Obstacle. Sur le diagramme, c'est la ligne avec le losange. Ça veut dire que Airspace utilise ces objets.
* Player, Enemy, Missile, Obstacle : Ce sont les modèles pour les objets du jeu. Chaque classe a ses propres attributs comme \_hp, \_x, \_y, et ses méthodes Update, Render. Update() de Missile le fait bouger en haut , mais Update() de Enemy le fait bouger en bas.
* Program : Cette classe commence tout. Sa méthode Main() crée la fenêtre Airspace et les premiers objets pour lancer le jeu .
* « GlobalHelpers » et « TextHelpers » : Ce sont des classes "utilitaires". Elles sont static parce qu'on n'a pas besoin de créer des objets. GlobalHelpers donne un nombre aléatoire et TextHelpers nous aide à écrire sur l'écran.