Documentation Technique

Projet C# – Jeu *Doodle Jump*

Auteur : Etienne Caulier

Date : 16 septembre 2025

CFPT Informatique

Table des matières

1 *Doodle Jump* en C#

Le projet consiste à développer une version du jeu **Doodle Jump** en C#, en utilisant une approche orientée objet et un moteur graphique 2D.

Le joueur contrôle un personnage qui saute automatiquement et dont l'objectif est de monter le plus haut possible en utilisant des plateformes.

1.1 Objectifs du projet

— Mettre en pratique la programmation orientée objet en C#.
— Gérer la **physique de base** (gravité, collisions, déplacements).
— Implémenter une **boucle de jeu fluide** (mises à jour et rendu en temps réel).
— Développer une **interface simple et intuitive** avec un score affiché à l'écran.
— Offrir une **expérience de jeu amusante et progressive** inspirée de l'original Doodle Jump.
— **Sauvegarde** de la partie en locale avec du *JSON*.

2 Technologies utilisées

— **Framework :** MonoGame.
— **Langage de programmation :** C#.
— **Environnement de développement :** Visual Studio.
— **Outils de versionnalisation :** GitHub.

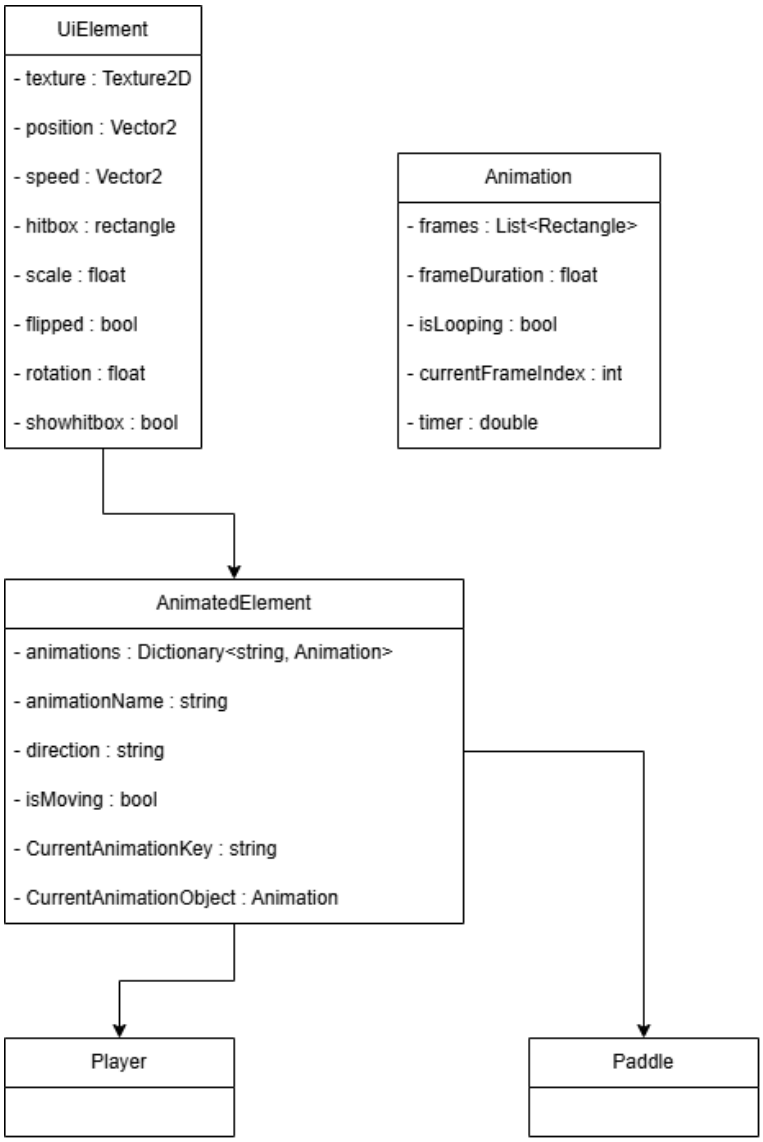
3 Analyse et conception

3.1 Description des fonctionnalités

Le jeu repose sur plusieurs mécanismes clés :
— **Gravité et sauts :** le personnage subit une gravité constante et rebondit lorsqu'il touche une plateforme.
— **Sprites 2D :** les éléments visuels (personnage, plateformes, arrière-plan, bonus, ennemis) sont représentés par des sprites 2D.
— **Défilement vertical infini :** le décor se génère de manière continue pour donner une impression de progression infinie vers le haut.
— **Interactions :**
  — plateformes fixes ou mobiles,
  — bonus (ressorts, jetpacks, etc.) **si j'ai le temps !**,
  — obstacles et ennemis à éviter ou éliminer.
— **Score :** le joueur accumule des points en fonction de la hauteur atteinte.

3.2 Diagramme UML

Le diagramme UML suivant représente la structure des principales classes du jeu :
— **Player :** Gère les actions du joueur.
— **Enemy :** Gère les comportements des ennemis.
— **Projectile :** Représente les tirs des personnages.
— **GameState :** Gère les états du jeu (menu, jeu en cours, fin de partie)


UiElement
- texture : Texture2D
- position : Vector2
- speed : Vector2
- hitbox : rectangle
- scale : float
- flipped : bool
- rotation : float
- showhitbox : bool
Animation
- frames : List<Rectangle>
- frameDuration : float
- isLooping : bool
- currentFrameIndex : int
- timer : double
AnimatedElement
- animations : Dictionary<string, Animation>
- animationName : string
- direction : string
- isMoving : bool
- CurrentAnimationKey : string
- CurrentAnimationObject : Animation
Player
Paddle