Onglet 1



**DOCUMENTATION TECHNIQUE**

**Formation Développeur de Jeux Vidéo Pro**

(Apprentissage par Projet / © Active Learning by Gaming Campus)

Sommaire

[**Introduction 3**](#_rgsg6orgjknt)

[Description du projet 3](#_3zdb0w48y92k)

[Le projet 3](#_y4stb4ejwjrs)

[Ressources 3](#_uz0msnfkkjj)

[Risques 3](#_voktt8dge6ws)

[Gestion du projet 3](#_lm04js3ajfp6)

[Description des outils techniques 3](#_tbf84l42y2kl)

[Unity 3](#_lzpzs9ka7155)

[**Doc technique du jeu 5**](#_6kqlujxjeado)

[Structure du projet 5](#_bm6zl5mfieb)

[Nomenclature 5](#_vot7e29bxp)

[Design patterns 5](#_dwhh9mser4do)

[Génération procédurale du terrain 5](#_2hlddvlurol3)

[Phases de jeu 5](#_ydjvg0m4hi2f)

[Éléments interactifs 5](#_gtwh6h1f9crb)

[Persistance de données 6](#_tpuufe6lqhg4)

[Procédure de tests 6](#_lx651h54pxwx)

[Planning de production 6](#_xucqjby5lptk)

[Historique des évolutions 6](#_ij85eh875p22)

[**Annexes 7**](#_2aon15j2j5ru)

[Game Design Document 7](#_xv15yhvvk8is)

[Comptes-rendus de réunion 7](#_tmbhbez34uh)

# Introduction

## Description du projet

Dans le cadre de notre formation, nous avons développé un jeu vidéo de type Infinite Runner sur Unity sur une période de 5 semaines de mars à avril 2025.

Le Game Design Document (GDD), nous a été fourni par Jonathan Cadiou à qui nous avons pu demander des précisions et présenter nos prototypes pour d'éventuelles remarques, lors de trois réunions d’équipes.

Le concept du jeu est simple, *Dr VAN TOPOR CRAZY Run* est un *infinite runner* inspiré des titres Temple Run ou encore Subway Surfers.

Le joueur doit gérer simultanément quatre jauges de sérum, chacune liée à une gélule spécifique. Le but est d'atteindre le score le plus élevé possible au fil de la partie et de survivre le plus longtemps possible.

La difficulté augmente avec le temps il faudra donc être rapide et agile afin que les jaunes ne se vide pas complètement et faire partie du tableau des Highscores.

### Ressources

* Développeur : Jimmy Marmin
* Game Designer : Jonathan Cadiou

### Ressources matérielles :

* Textures : ChatGPT et Photoshop
* Modèle 3D : <https://sketchfab.com>
* Animation : <https://www.mixamo.com/>
* Sons et musiques : itch.io

### Gestion du projet

Méthode de projet : Réunion environ toutes les 2 semaines pour discuter de l’avancement mais également pour demander d’éventuelles précisions sur le GDD.

Google Drive est utilisé pour échanger les documents projet.

Le versionning du projet est géré via Unity Version Control et des backup sur le poste.

Un channel Discord est également disponible afin d’échanger sur le projet.

Tâches, missions et responsabilités de chacun :

* Jimmy Marmin : Développement du jeu et tests
* Jonathan Cadiou : Rédaction du GDD et test du prototype

## Description des outils techniques

### Unity

Description Wikipedia :

**Unity** est un [moteur de jeu](https://fr.wikipedia.org/wiki/Moteur_de_jeu) [multiplateforme](https://fr.wikipedia.org/wiki/Logiciel_multiplateforme) ([smartphone](https://fr.wikipedia.org/wiki/Smartphone), [ordinateur](https://fr.wikipedia.org/wiki/Ordinateur), [consoles de jeux vidéo](https://fr.wikipedia.org/wiki/Console_de_jeux_vid%C3%A9o) et [Web](https://fr.wikipedia.org/wiki/World_Wide_Web)) développé par [Unity Technologies](https://fr.wikipedia.org/wiki/Unity_Technologies). Il a la particularité de proposer une [licence gratuite](https://fr.wikipedia.org/wiki/Freeware) dite « *Personal* » avec quelques limitations de technologie avancée au niveau de l'éditeur, mais sans limitation au niveau du moteur.

Il est l'un des deux moteurs les plus répandus dans l'industrie du [jeu vidéo](https://fr.wikipedia.org/wiki/Jeu_vid%C3%A9o), aussi bien pour les grands studios que pour les [indépendants](https://fr.wikipedia.org/wiki/Jeu_vid%C3%A9o_ind%C3%A9pendant). Par rapport à son concurrent principal, [Unreal Engine](https://fr.wikipedia.org/wiki/Unreal_Engine), Unity est considéré comme ayant une [interface utilisateur](https://fr.wikipedia.org/wiki/Interface_utilisateur) plus facile d'accès et un service plus approprié pour les créations indépendantes.

Concrètement, ce qui rend Unity très intéressant pour notre projet d’infinite runner, c’est entre autres le fait qu’il est aisé de créer et manipuler des objets 3D dans l’interface. Notre objectif principal étant l’apprentissage et l’utilisation de la programmation, Unity nous permet de ne pas passer trop de temps sur la création d’objets 3D et de proposer un prototype assez rapidement.

Utilisation de code C#, programmation orientée objet et design pattern.

# Doc technique du jeu

## Structure du projet

|-- Assets

|-- |-- Audio Contient les ressources audio

|-- |-- |-- Musics

|-- |-- |-- Sounds

|-- |-- Graphics Contient les ressources graphiques

|-- |-- |-- Materials

|-- |-- |-- Models

|-- |-- |--|-- Animations

|-- |-- |-- Textures

|-- |-- Prefabs Contient les préfabriqués

|-- |-- |-- Grounds

|-- |-- |-- Gelule

|-- |-- Scenes Contient les scènes

|-- |-- Scripts Contient les scripts

|-- |-- UI

|-- |-- Save and Load

## Nomenclature

● Fichiers : CamelCase, noms explicites (contient mat pour un matérial, text pour une texture etc)

● Fonctions : verbes d’action (LaunchGame, AddLife)

● Variables : camelCase (life, movementSpeed...)

## Design patterns

* Singleton : GameManager
* Coroutine : GeluleSpawner
* Game Loop : Start, Update, Exit
* Events : OnTriggerExit, OnColliderEnter, OnClick, etc..
* UnityEvent : OnInitLife, OnChangeLife,...
* Algorithme de la roulette : Permet de générer les Gelule en fonction de la barre de vie et d’un poids

## Génération procédurale du terrain et des objets

* Sol générer à partir d’un préfab Ground et d’un empty Spawner avec une collision box qui permet la création d’un nouveau Ground à chaque fois l’objet sort de celle-ci.
* Gelule générer aléatoirement à partir de 5 points de spawn.
* Un Scroller permet de faire avancer le tout.
* Un empty Destroyer permet de supprimer les éléments passer derrière le joueur hors de l’écran.

## Phase de jeu

* Menu (Jouer, Options, Highscore, Quit)
* Lancement du jeu (Scrolling du sol, Spawn des Gelules, Diminution des jauges)
* Gameplay en boucle jusqu’à ce qu’une jaune soit vide
* Game Over avec le score et retour au menu

## Éléments interactifs

* Gélules

## Persistance de données

● Le score est sauvegardé à chaque partie dans un JSon

## Procédure de tests

* Affichage de logs via Debug.Log
* Utilisation des breakpoint dans Visual studio
* Affichage des colliders dans Unity pour tester les collisions

## Planning de production

| **Semaine** | **Travail réalisé** |
| --- | --- |
| 1 | Apprentissage Unity |
| 2 | Apprentissage Unity |
| 3 | Premier prototype avec une capsule |
| 4 | Ajout d’un personnage animé et colliders |
| 5 | Ajout de gélules générées aléatoirement et menus |

## Historique des évolutions

| **Version** | **Détails de la version** |
| --- | --- |
| V0.1 | Création d’une route généré aléatoirement et gestion du scrolling |
| V0.2 | Ajout d’une capsule et des colliders |
| V0.3 | Remplacement de la capsule par un personnage animé |
| V0.4 | Ajout de la texture du sol et correction de bugs |
| V0.5 | Ajout de gélules générées aléatoirement et de la gestion du score |

## 

# Annexes

## Game Design Document

## Comptes-rendus de réunion

## 