2020

Rapport de travail
Cancre Simulator

Bovay Louis, Amstutz Thomas, Goudron Mathieu

DIVTEC – PORTES-OUVERTES 2020

05/10/2020

Table des matières

1. Introduction 1
2. Le jeu 2
   2.1 Le moteur 2
   2.2 Planter le décor 2
   2.3 Le joueur 2
   2.4 Le tir 2
   2.5 Les cibles 2
   2.6 Le HUD 2
3. L'application smartphone 3
   3.1 Environnement de développement 3
   3.2 Fonctionnement 3
   3.3 Technologie 3
4. Unreal Engine 3
   4.1 Le moteur 3
   4.2 Avancement du projet 3
   4.3 Abandon de Unreal Engine 3
5. Unity 4
   5.1 Renaissance de l'espoir 4
   5.2 Scripts 4
6. Gestion des scores 5
   6.1 Début de partie 5
   6.2 En cours de partie 5
   6.3 Fin de partie 5
7. La connexion 5
   7.1 Serveur TCP Unity 5
   7.2 Connexion depuis un smartphone 5
   7.3 Problèmes 5
8. Installation 6
   8.1 Matériel 6
   8.2 Logiciels requis 6
   8.3 Package requis 6
   8.4 Fichiers du projet 6
   8.5 Installation 6

8.6 Le dossier du projet 7
9. Déroulement du jeu 7
10. Ajouts de dernière minute 8
10.1 Un bord d'écran 8
10.2 De meilleures couleurs 8
11. Améliorations possibles 8
11.1 Ajouts de scènes 8
11.2 Tableau des scores 8
11.3 Faire du jeu un jeu de survie 8
11.4 Smartphones à la place de manettes 9
11.5 Geler le curseur 9
12. Conclusion 9
13. Sources 10

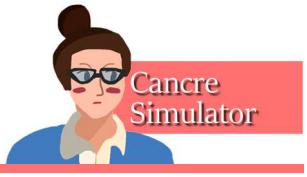
1. Introduction

L'EMT nous a donné l'occasion de créer un projet libre autour de l'informatique à présenter lors de la semaine des portes ouvertes.

Nous sommes partit sur un jeu mêlant plusieurs périphériques : un ordinateur (qui gère le jeu) et un téléphone (qui sert de contrôleur).

Le but : marquer le plus de points dans un temps imparti, attention toutefois à ne pas tirer lorsqu'un prof peut voir le joueur, sinon il perdra des points !

Figure 1 La salle de classe et ses trois professeurs



2. Le jeu

2.1 Le moteur

Nous avons utilisé Unity qui va de pair avec Visual Studio pour la partie programmation.

Le moteur est assez simple à prendre en main, et les utilisateurs sont actifs sur internet et mettent en place un grand nombre d'aides/tutoriel

2.2 Planter le décor

Le décor est une succession d'images faites mains par Yvan Naef, un ami à nous, qui représentent fidèlement une salle de classe ainsi que ses employés.

2.3 Le joueur

Le joueur est représenté par un pointeur à la couleur unique servant de viseur.

2.4 Le tir

Par simple action d'un bouton (par défaut A), si le pointeur du joueur est sur une cible, la cible est détruite.

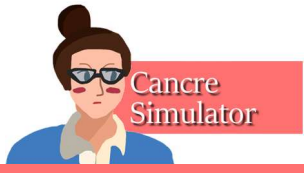
2.5 Les cibles

Les cibles bougement horizontalement, verticalement à une vitesse aléatoire ou ne bougent pas. Elles apparaissent aussi avec une taille aléatoire qu'elles conservent jusqu'à leur destruction.
Une cible dorée à 2% de chances d'apparaître, si détruite par un joueur, elle lui confère un grand nombre de points.

2.6 Le HUD

Le HUD contient deux éléments par joueur :

- Le pointeur de couleur qui sert de viseur pour le joueur
- Le nombre de points du joueur qui a la même couleur que son pointeur



3. L'application smartphone

3.1 Environnement de développement

Nous avons utilisé le SDK Flutter par Google, avec Visual Studio Code comme outil de développement graphique. Le choix de Flutter s'est imposé grâce à sa modernité et sa compatibilité avec iOS et Android sans avoir besoin de modifier le code.

3.2 Fonctionnement

Après le démarrage de l'application et le passage du splash screen, un champ de texte est disponible pour entrer un nom, après avoir mis le nom, un bouton Jouer apparaît.

Pour la version de test, nous avons ajouté un menu de paramètres pour nous permettre de modifier l'IP et le port de connexion facilement.

3.3 Technologie

Le SDK Flutter utilise la langage Dart qui est ensuite converti lors de la compilation en Swift pour iOS et en Java pour Android. Les téléphones se connectent en Wi-Fi au stand et la connexion est établie avec le protocole TCP entre le serveur de jeu et l'application.

4. Unreal Engine

4.1 Le moteur

Unreal est un moteur de développement de jeu 3D géré par Epic.

Ce fut notre premier choix car nous avions déjà un peu tâté le terrain précédemment.

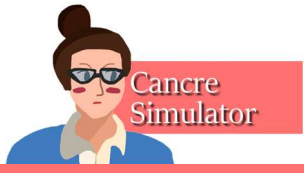
4.2 Avancement du projet

Nous avons réussi à implémenter :

- Une salle de classe
- Des cibles mouvantes
- Un HUD
- Des plans de caméra (Menu, jeu, fin de partie)

4.3 Abandon de Unreal Engine

Nous avons fini par abandonner Unreal Engine, ce moteur est trop compliqué et à peu d'aides disponible sur internet. Nous avions aussi un problème de connexion lié au serveur TCP.



5. Unity

5.1 Renaissance de l'espoir

Nous avons pris comme second choix d'outil de développement, le moteur Unity.

Beaucoup plus simple à prendre en main, une doc en ligne très complète et une communauté ouverte et prête à aider.

5.2 Scripts

Nous avons créé beaux nombres de scripts pour notre projet, certains gèrent des objets, d'autres des évènements ou les deux à la fois, voici les scripts les plus importants/intéressants que l'on a réalisé.

Le GameManager :
Nom du script : GameManager

Là où se trouve tout le fonctionnement du jeu et faisant communiquer les scripts entre eux, gère aussi le début, le milieu et la fin de partie ainsi que les scores.

Les Professeurs :
Nom des scripts : DoorTeacher, WindowTeacher, Professor

Servent à contrôler les mouvements des profs avec deux méthodes « MoveIN() » et « MoveOUT() » qui leur permettent de rentrer et sortir de la classe. On a aussi adapté ses fonctions au prof principal qui lui ne peut que se retourner.

Chaque prof contient une variable booléenne isIN qui informe si oui ou non le prof est dans la salle de classe, ce qui nous sert à savoir s'il nous voit tirer ou non.

La variable est atteignable via leur méthode « IsTeacherIn() »

La cible :
Nom du script : Target
La cible n'a comme seule fonction celle qui permet de détecter lorsqu'elle est touchée.

Elle possède aussi tout un code qui définit sa taille, ses mouvements, si elle est dorée et sa durée de vie si personne ne la touche.

On peut donc par la suite créer un objet « cible » qui contient le script Target et l'invoquer à volonté pour avoir plein de cible qui apparaissent sur le tableau.

6. Gestion des scores

6.1 Début de partie

Le score du joueur est caché de base par une fonction mathématique sans importance écrite au tableau dans le coin supérieur gauche, lorsqu'un joueur rejoint la partie, la fonction au tableau est remplacée par le score du joueur.

6.2 En cours de partie

Le score du joueur s'incrémente ou se décrémente à chaque fois qu'il marque ou perd des points, le joueur peut avoir un score négatif.

6.3 Fin de partie

Les scores des joueurs restent affichés en haut à gauche. Le meilleur joueur est au centre de l'écran.

7. La connexion

7.1 Serveur TCP Unity

Nous avons mis en place un serveur TCP dans notre projet qui permet de connecter plusieurs téléphones via Wi-Fi et d'envoyer des informations entre le jeu et l'application mobile (donnée du gyroscope, action de tir et nom du joueur).

7.2 Connexion depuis un smartphone

L'application contient l'adresse IP du serveur noté en dur ainsi que son port, seule la modification du nom est permis à l'utilisateur.

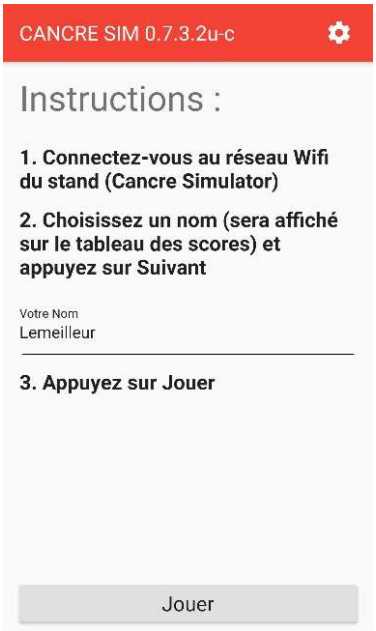


Figure 2 Ecran d'accueil de l'application

7.3 Problèmes

La connexion entre le téléphone et le jeu est fonctionnelle, mais le problème insoluble est la traduction des données reçues, que nous n'avons pas réussi en mettre en place.

Nous avons donc pris la décision de passer aux manettes Xbox par manque de temps.

8. Installation

Cette partie du document permet la reprise du projet par une personne externe au groupe et donnera tous les détails nécessaires pour y parvenir.

8.1 Matériel

Un PC avec Windows 10 Professional, un clavier, une souris, un écran, une connexion internet et une manette Xbox One.

8.2 Logiciels requis

- Unity Hub
- Unity 2020.1.4
- Visual studio (est à installer lors de l'installation de Unity)
- Github Desktop

8.3 Package requis

- Input System

8.4 Fichiers du projet

La totalité du projet est disponible sur GitHub.

8.5 Installation

1) Créez-vous un compte étudiant Unity puis téléchargez Unity Hub
   Se créer un compte - Unity
2) Installez Unity Hub
3) Lancez Unity Hub et connectez-vous
4) Depuis Unity Hub, sélectionnez la version 20.1.4 de Unity, cochez l'installation parallèle de Visual Studio et installer le tout (de préférence sur un SSD)
5) Téléchargez et installez GitHub Desktop
6) Clonez le projet depuis GitHub
7) Depuis Unity Hub, ouvrez le projet
8) Une fois sur Unity, rendez-vous dans la barre d'outils supérieure sous « Tools > Package manager »
9) Sélectionnez « Input System », installez et relancez Unity
10) Toujours dans la barre d'outils supérieure allez vers « Edit > Preferences… » et sous l'onglet « External Tools » choisissez « Visual Studio 2019 (Community) » comme Editor Attaching.

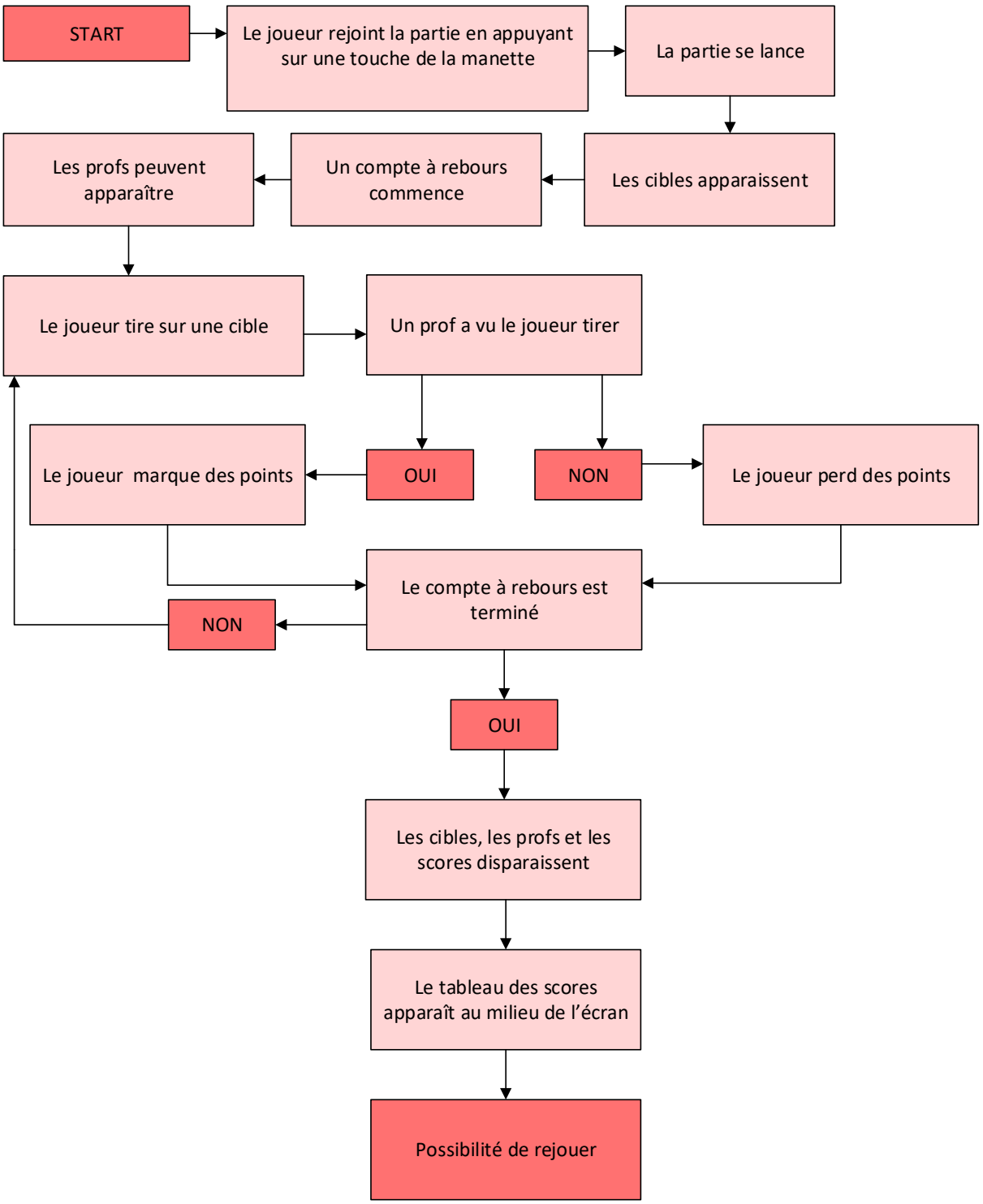
L'installation est terminée et vous êtes prêts à travailler.

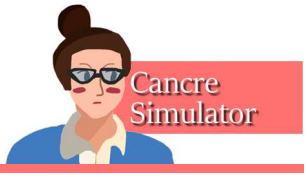
8.6 Le dossier du projet

Les fichiers du jeu ont été correctement classés par types (Sprites, Script, Scène…) dans le dossier du projet et chaque scripts/méthodes ont été commentés avec soins.

Nous avons aussi détaillé les principaux scripts/fonctionnements dans le 5.2.

9. Déroulement du jeu





10. Ajouts de dernière minute

10.1 Un bord d'écran

Après la première journée de test nous avons remarqué que beaucoup de joueurs sortent leur viseur de l'écran et le « perdent », nous avions prévu un bouton pour recentrer le curseur mais les gens ne pensent jamais à l'utiliser car ils sont trop concentrés sur le jeu.

C'est pourquoi nous avons mis une limite au bord de l'écran pour que les viseurs soient forcés de rester dans l'écran, dès que quelqu'un essaie de sortir de la zone autorisée, il est re-téléporté à l'intérieur de cette zone.

10.2 De meilleures couleurs

A cause du projecteur, les couleurs claires ne se voient que très peu, nous avons donc passé le viseur vert clair en orange et le viseur cyan en bleu foncé.

11. Améliorations possibles

11.1 Ajouts de scènes

Nous n'avons découvert que trop tard les scènes dans Unity, qui permettent de sauter d'une situation à l'autre simplement, cela nous aurait permis de mettre un écran de début et un écran de fin sans devoir à chaque fois désactiver tout ce qui se trouve dans la scène.

11.2 Tableau des scores

Un tableau des scores enregistrer dans un fichier JSON était prévu mais par manque de temps nous avons dû le laisser de côté.
Nous avions prévu de faire un tableau du top 20 des meilleurs scores, donc si durant une partie les 4 joueurs font un score excellent, leur 4 scores seraient affichés.

11.3 Faire du jeu un jeu de survie

Au lieu de simplement marquer des points en tirant sur des cibles, le joueur aurait 2 barres : une de style et une autre de colère.
En tirant sur les cibles, il gagne du style, en se faisant voir par un prof, il gagne de la colère.

Le niveau de colère ne descend jamais contrairement au style qui descend après 5 secondes d'inactivité.

Le joueur devrait survivre le plus longtemps sans que sa barre de style ne soit vide ou que sa barre de colère n'atteigne son maximum.

11.4 Smartphones à la place de manettes

Créer un serveur TCP ainsi qu'une application mobile sur Unity pour pouvoir se servir de smartphones à la place de manettes.

Le smartphone servirait à recueillir le nom du joueur, la position de son gyroscope et d'instancier un identifiant pour le joueur.

Les curseurs du jeu ne seraient donc plus contrôlés par le joystick de la manette mais par le gyroscope du smartphone du joueur.

11.5 Geler le curseur

Si un professeur voit le joueur tirer sur une cible, le joueur perd des points et son curseur est gelé ou ralenti pendant 5 secondes.

12. Conclusion

Découvrir le développement d'un jeu vidéo nous a permis de comprendre comment certaines applications étaient construites.
Nous avons aussi mis en œuvre l'organisation, le partage des tâches et le travail d'équipe.

Nous sommes plutôt fiers de notre travail, même si nous restons quelque peu frustrés de ne pas avoir pu faire certaines choses autrement.

13. Sources

Documentation Unity :	La documentation
Divers postes StackOverflow :	Stack Overflow
Le forum Unity :	Forum Unity
Chaîne YouTube officielle Unity :	Leur chaîne YouTube
GameDev pour la gestion des timers :	Leur site
Code Monkey qui propose de nombreux tutoriel Unity :	Sa chaîne YouTube
Le discord de Tom Weiland :	Le discord

Cancre Simulator 2020