Rapport de projet – 320-POO

Stone Sling



Luca Premat – FID1

ETML – Vennes

32 périodes

P\_320

M. P. Chenaux

Table des matières

[1. Description : 2](#_Toc192812547)

[2. Règles du jeu : 2](#_Toc192812548)

[3. Déroulement du jeu : 2](#_Toc192812549)

[4. Code : 2](#_Toc192812550)

[4.1. Classes : 2](#_Toc192812551)

[4.1.1. Joueurs : 2](#_Toc192812552)

[4.1.2. Tours : 2](#_Toc192812553)

[4.1.3. Projectile : 2](#_Toc192812554)

[4.1.4. AimPoints : 2](#_Toc192812555)

# Introduction :

Le projet est un jeu de type application console, programmé en C# programmation orientée objet. Le jeu est basé sur le jeu « Stone Sling » sorti en 1981 sur la console Videopac.

But de ce projet :

* Maitrise de la programmation orientée objet
* Savoir programmer un jeu avec des mécanique plus poussée tel que de la détection de collision, de la physique simple, du calcul d’angles ainsi que d’autres éléments interactifs.

# Description :

Le jeu Stone Sling est un jeu 2v2 où les joueurs tirent tour par tour un projectile en direction de l’adversaire. Le but du jeu est de toucher le joueur adverse 3 fois pour gagner, il y a aussi un système de points, et une tour devant chaque joueur pour les protéger. Toucher une tour aura pour conséquence de détruire petit à petit cette dernière et donne des points.

# Règles du jeu :

Si un projectile touche la tour adverse, celle-ci s’abime et diminue en taille, et accorde 10 points au joueur. Si un joueur touche ça propre tour, celle-ci s'abime de la même façon, mais lui fera perdre 10 points.

Si un projectile touche un joueur, celui-ci perd un de ces 3 points vies (HP). Toucher un joueur accorde aussi 50 points, se toucher soi-même aura pour conséquence de se faire perdre une vie et -50 points.

# Déroulement du jeu :

Chaque joueur joue chacun son tour, un tour se déroule en deux étapes : d’abord un point défile devant le joueur, ce point permet de sélectionner l’angle, quand le joueur presse la barre espace l’angle est verrouillé. Vien ensuite la phase de la sélection de la force : Une barre qui se charge est décharge en boucle apparait en dessous du cadre contenant le score et la vie du joueur, cette barre permet la sélection de la force, plus la barre est remplie plus le projectile sera tiré fort.

Quand un joueur arrive à court de vies la partie se termine et le joueur adverse a gagné.

# Démarche initiale :

Quand nous avons commencé le projet j’ai d’abord réfléchi à comment démarrer la programmation, Comme je n’avais aucune idée de comment faire toute la partie technique (Physique, collisions, changement du score et des HP en fonction des collisions) j’ai décidé de commencer par faire le visuel de l’application : j’ai d’abord fait l’affichage des joueur, de simples stickmen, ensuite j’ai dessiné les deux tours, et finalement je me suis penché sur le design de l’HUD.

Grâce à ce squelette J’ai pu plus facilement imaginer comme faire fonctionner les autres fonctionnalités du programme.

# Conception :

Mon code est divisé en huit classes afin d’avoir un projet mieux organisé et plus simple à naviguer lors de la programmation voici les huit classes :

**Program** : Classe principale du projet, contient le main qui gère le bon déroulement de tout le jeu, on y trouve aussi la gestion des tours et la génération des angles.

**Player** : Cette classe s’occupe d’afficher les joueurs ainsi que de gérer les collisions avec ceux-ci.

**HUD**: Affichage du rectangle contenant les informations sur les joueurs tel que : joueur 1 ou 2, HP et score.

**AimPoints** : Classe qui affiche et fait se déplacer le point de sélection de l’angle

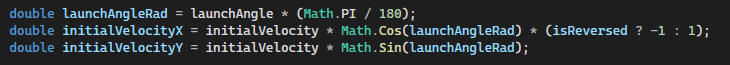
**Towers** : Contient l’affichage des différents états des tours, la gestion des collisions et le changement d’état (tour de plus en plus détruite).

**StrengthBar** : gestion de la barre de force et détection si la barre espace est pressée.

**Projectile**: S’occupe de tout ce qui du projectile après qu’il soit tiré, on y trouve le calcul de la physique, les collisions et l’attribution ou déductions des points et HP.

## Gestion de la physique et des collisions :

Tout le bon fonctionnement du jeu se repose sur la physique du projectile et des collisions.

Au départ je ne savais pas du tout comment m’y prendre pour reproduire la physique du projectile donc en suivant le conseil de M. Chenaux j’ai demandé à GitHub copilot de me créer un programme qui simulerait une balle lancée d’un coter a l’autre en respectant une physique réaliste en fonction de la force et de l’angle. J’ai ensuite fait quelque expérimentation avec ce programme pour comprendre son fonctionnement, j’ai donc pu reprendre la formule mathématique qui permettait la gestion de la physique, formule qui est la suivante :  


Ainsi que les formules permettant le calcul de la position future du projectile :   
A computer screen shot of code

AI-generated content may be incorrect.

Grace à cela j’ai pu me débloquer et continuer la suite du projet

Pour la gestion des collision j’ai aussi suivi le conseil du professeur et ait utilisé les positions X et Y des tours et des joueurs en les comparants a la position X et Y du projectile afin de détecter si celui-ci entre en collision avec un objet.

## Gestion de l’angle :

Pour déterminer l’angle de tir j’ai fait une boucle qui compte le temps passé et fait correspondre le temps a la position en degré du points qui se déplace pour choisir l’angle.

A screen shot of a computer code

AI-generated content may be incorrect.

Ensuite l’angle est enregistré pour le tir

## Gestion de la force :

A black background with white text

AI-generated content may be incorrect.A computer screen shot of a code

AI-generated content may be incorrect.Après que l’angle ait été sélectionné vient le choix de la force le fonctionnement est assez simple, on incrémente juste une variable i sur la longueur de la barre (20 caractères) et la position finale quand on appuie sur espace définit la valeur de la force. On a donc une valeur comprise entre 0 et 20 qui sera ensuite multipliée par trois.