Université Paris Cité

Projet de programmation avancé

Fancy Fencing

Fait par:
BELMIR Yaghmoracen

2022/2023

Introduction

Dans ce document vous trouvez les explication concernant les fonctionnalité presente dans le jeu, ce dernier est assez bien assisté et donc les joueurs ne seront presque jamais en état de confusion, le jeu avertit les joueurs lors d'une manoeuvre sensible, et explique tout le fonctionnement et comment s'y prendre face au menu. Plusieurs fonctionnalités on été rajoutés ainsi que plusieurs testes qui se font au fur et à mesure à fin de perfectionner le jeu. Le projet est mis sur GitHub[1].

Version terminal

Fonctionnalités

Menus

Dans le jeu, il existe deux menus, un menu principal et un menu du mode “pause”, leur fonction en python n’est autre que des affichages qui expliquent en détails ce que les joueurs peuvent faire dans ce menu et donc y aura des “inputs” clavier pour connaître la décision du joueur.
Menu principale :

Contenu	**Fonctionnalités**
Un guide	en insérant la touche “h”, les joueurs verront afficher sur l’écran une brève explication sur la façon de joueur au jeu ainsi que “key_binding”
Lancer une nouvelle partie	avec la touche “n” les joueurs accéderont à la possibilité de choisir une des scènes existante puis chargera un fichier “.ffscene” et générer la scène dans le grâce à la fonction “make_scene”
Continuer une partie	avec la touche “c” les joueurs pourront donc reprendre une partie sauvegardée avec l’extension “.ffsave” et ce qui la diffère de l’extension précédente est le fait que cette dernière possède au plus de la scène, les états des joueurs (positions et statistiques)
Quitter le jeu	avec la touche “q” le jeu fini de s’exécuter et donc faudra relancer la version terminal de nouveau

Table 1 – Fonctionnalités du menu principale.

Le tableau ci-dessus représente le contenu du menu principale du jeu version terminal.

Contenu	**Fonctionnalités**
Continuer la partie courante	en insérant la touche “c”, les joueurs peuvent continuer la partie qui à été mise en pause
Réinitialiser la partie courante	avec la touche “r” les joueurs peuvent réinitialiser la partie ce qui remet leur scores à 0 et les repositionnent au point de départ
Sauvegarder une partie	avec la touche “s” le jeu contient une quantité de sauvegarde limitée, les joueurs pourront donc soit créer une sauvegarde dans un fichier existant vide “.ffsave” soit écraser une ancienne sauvegarde si bien sur cette décision à été accorde par le joueur et donc l’ancienne sauvegarde sera remplacée par la partie courante tout en prenant en compte leurs scores et leur positions ainsi que leurs stats
Retour au menu principal	avec la touche “m” les joueurs pourront revenir au menu principale si ils acceptent d’abandonner la partie courante
Quitter le jeu	avec la touche “q” le jeu fini de s’exécuter et donc faudra relancer la version terminal de nouveau

Table 2 – Fonctionnalités du menu principale.

Le tableau ci-dessus représente le contenu du menu pause du jeu version terminal, ce menu en revanche possède directement l'affichage des "key_binding" pour voir les touches ainsi que les actions de chaque joueur (nécessite pas l'accès au guide).
dans les deux, les touches citées peuvent soit être en minuscule soit en majuscule, dans les deux cas la fonction s'exécute.
Toute fonction qui possède un lien avec les fichiers est bien gère de telle façon que le jeu ne rencontre aucun "bug", ex : si le jeu détecte un fichier manquant, l'action qui a enclenchée cette vérification ne s'exécutera pas ; Le jeu donne bien une solution qui s'affiche sur l'écran et qui n'est que "Veillez restaurer tout les fichiers". ASDJFHA]DFA DHFAFA

Génération de la scène

Selon ce qui à été décidé par les joueurs, la fonction "make_scene" récupère la parie nécessaire du fichier qui à été choisis dans le menu et crée une matrice en positionnant bien les joueurs et tout ce que la scène peut posséder comme objet, elle crée en même temps deux instances de la classe "player" qui vont représenter les deux joueurs
Cette classe possède plusieurs attribut, a savoir : positions, statistique, et leur façons dont ils doivent être affichés, de ce fait les deux joueurs peuvent s'inverser leurs place et seront toujours l'un face a l'autre, la détection d'obstacles diffère de celle que quand ils sont dans leur bon cote mais ceci n'annulera pas le bon fonctionnement du jeu.
La fonction "update_scene" permet de mettre à jour la matrice scène qui représente la scène : en repositionnant les joueurs et affiche leurs posture après toute actions, m.à.j des scores si cela à été produit, etc...
la majorité des variables/instances créés dans le programme sont globales, ceci est du au modifications multiples faites par les fonction du jeu.
La scène est afficher par une simple itération sur le contenu de la matrice scène ceci est fait avec la fonction "dispaly"

Listner/actioner

Le programme possède deux fonction fondamentale au jeu, la 1^ere est la fonction "listner" qui permet de détecter les touches du clavier, ainsi si une touche est valide elle sera immédiatement retenu en compte, l'exécution de cette dernière dépendra des statistique du joueur, exemple : si le joueur 1 effectue une attaque dans la frames 0 et que sont "attaque speed" est égale à 3, son action ne sera exécutée qu'après ces 3 frames sont écoulées et ceci est traité par la 2^eme fonction "actioner".
Le jeu ne permet pas plusieurs actions en même temps, si le joueur effectue deux mouvement durant la même frame d'affichage, seulement la 1^ere action sera prise en compte, néanmoins une action et un mouvement peuvent être exécuté en même temps (le joueur peut attaquer et avancer en même temps mais ne peut pas aller à gauche et à droite en même temps, ni bloquer et attaquer en même temps).
la fonction calcule les distances entre les joueurs et les obstacles (un joueur est considéré comme obstacle par rapport à son adversaire). Elle prends en compte aussi le faite que si le joueur 1 et au coté gauche ou droite de la scène d'où le teste "*if player1.get_weap() == 'g'*", pareil se fait pour le 2^eme jouer.
Aussi cette fonction permet d'annuler une succession de mouvement si le joueur se vois trop lent et qu'il maintenais une touche de mouvement pendant plusieurs frames et décidé de rebrousser chemin.

Concernant la 2^eme fonction "actioner", elle permet simplement de compter les frames écoulées après avoir retenu chaque action depuis le "listner".

Calcule de score

Deux fonctions bien simples qui permettent ceci, la 1^ere "hitbox", permet de savoir si un joueur, lors d'une attaque, son ennemie se trouve bien dans sa portée. Ensuite la fonction "referee" intervient, et juge la situation ; Si les deux joueurs se trouvent dans la portée de l'un de l'autre et que les deux ont attaque en même temps, rien ne se passe.
En revanche si l'un des deux q attaqué et que son adversaire est dans sa portée et surtout si ce dernier n'est pas en position de blocage, le point est attribuée a l'attaquant et repositionne les deux joueurs au point de départ.

Déroulement du jeu

Le jeu est lancé avec une boucle infinie qui fait appelle à la fonction “terminal” qui elle même fait appelle au fonctions expliqué au-dessus dans ce document et incrémente les frames, etc... Le jeu est accompagné par de la musique.

Références

[1] https://github.com/yaghmo/Projet-Programmation-Avancee.