Compte rendu projet Python

# Fonctionnalité

Notre jeu permet une gestion des deck. En effet, il est possible de sauvegarder un deck et de le réutiliser pour une prochaine partie. Un nom est demandé lors de la sauvegarde du deck pour différencier les différents deck sauvegardés. Lors d’une nouvelle partie le joueur a la possibilité de charger un deck précédemment sauvegardé dans ce fichier.

Le joueur a aussi la possibilité de crée un deck en choisissant successivement sur le nombre de carte total dans le deck, 1 carte parmi 3. Et comme précédemment, le deck formé est enregistrable et changeable à chaque début de partie.

L’interface laisse aussi le choix à l’utilisateur de crée son deck complètement aléatoirement, toujours avec les fonctions de sauvegarde et chargement du deck.

Pour permettre aux joueurs de jouer à la console, il lui est demander d’entrer un certain input selon une liste d’input afficher. C’est de cette manière qu’ils font les divers actions du jeu, comme poser des cartes, regarder le plateau, regarder ses cartes, poser une carte, attaquer, etc….

# Architecture

Nous avons utilisé la programmation orientée objet pour notre projet.

En effet, nous avons découpé notre jeu en plusieurs classes.

## Game

La classe Game est le coeur de notre jeu. Elle créer les deux joueurs et les configures pour une partie. La configuration d’un joueur signifie la création de son deck. Son deck peut ètre créé de plusieurs façon, la première est la récupération d’un deck existant dans le fichier de configuration. La seconde possibilité est de laisser le jeu créer un deck totalement aléatoire. La dernière façon est de créer un deck de toute pièce en choisissant chaque carte qui le composera. Cependant le joueur ne selectionne pas directement les cartes qui composeront sont deck. En effet, un deck est composé de N carte (ce paramètre est modifiable dans la classe Player), pour chaque carte composant le deck un choix de trois carte est proposé au joueur, le joueur choisit alors parmit ces trois carte.

## Player

La class player contient toutes les informations relative à un joueur. C’est à dire ses points de vie, son deck, sa main, son plateau pour poser ses cartes.

Cette classe s’occupe de la gestion du deck, c’est à dire, le tirage de carte que ce soit au lancement de la partie (tirage de 3 cartes) ou au fils des tours où on pioche une carte par tour.

Elle permet le posage ou utilisation de cartes, ainsi qui leur destructions. Par exemple, lorsque l’on pose un servant, utilise un item, équipé une arme, quand une arme se casse, quand un servant meurt, etc…

.

Elle fait la transition entre cartes dans le deck, cartes dans la main et cartes sur le plateau de jeu.

Et sur le plateau de jeu, il y a un espace spécifique pour poser les serviteurs, et un espace spécifique pour poser tout ce qui n’est pas serviteur (Item et Weapon)

C’est aussi cette classe qui s’occupe de la gestion du mana du joueur pour poser les carte

## Card

Card est la class mère des troix classes filles Servant, Weapon et Item.

En effet, les 3 classes cités ont toutes des attributs commun qui sont :

* un id
* un nom
* une description
* un coût en mana pour poser
* un objet Stat

## Stat

Cette class est importante car elle se retrouve sur toutes les cartes. Elle est l’une des bases de notre gameplay car notre jeu de carte est inspiré du genre RPG.

La classe comporte les attributs suivants :

* point de vie
* force
* intelligence
* précision
* vitesse
* défense
* résistance
* critique

C’est à partir de ça que l’on détermine une partie de l’issue d’un combat entre 2 serviteurs, qu’on affecte les effets d’un objet ou d’une arme. Car les objets et armes ne font que varier les stats d’un servant.

## Servant

La classe servant est une grosse classe car les servants sont les acteurs principaux, aussi important que les Players. C’est avec eux que les joueurs jouent et c’est grâce à eux que l’on gagne une partie principalement. Cette classe est volumineuse en raison de la gestion des combats car le système n’est pas des plus simples.

D’ailleurs, pour alléger les codes, des enums ont été utilisés pour éviter des faires des tonnes de conditions.

C’est dans cette classe que les stats d’un Servant sont transcrites pour avoir les statistiques lors d’un combat entre serviteur. Il y a aussi la prise en compte du type d’arme utilisé par le serviteur pour affecter les bonus ou malus d’attaque et précision, si un serviteur inflige des dégâts selon sa force ou intelligence, si il se protège selon sa défense ou sa résistance, ajout des stats d’une arme s’il y en a une d’équipé sur le serviteur.

## Weapon

Cette class représente un équipement que l’on peut affecter à un serviteur. Elle possède un type particulé. Et c’est seul un serviteur prouvant équiper ce type d’arme pourra équiper un carte weapon ayant ce type.

L’attribut spécial de cette classe est la durabilité. Car une arme s’use à chaque utilisation et se casse au bout d’un moment pour finir au cimetière.

## Item

Classe n’ayant qu’une fonction qui ne se trouve pas dans la classe Card (useItem). Cette classe item est fait pour l’utilisation d’objet directement dur un serviteur, contrairement à une Weapon, qui reste sur le terrain tant que l’arme ne se casse pas ou si un serviteur ayant une arme sur lui ne meurt pas.

Nous avons aussi des classes de support dans le package utils

## Enum

Class qui est la class nous ayant permi d’implémenter les enum en python

Config

Config est utilisé pour l’utilisation du json, en lecture lors d’un chargement de carte/deck ou écriture lors d’une sauvergader de carte

Les cartes sont toutes stockées dans un fichier .json. Il est donc facile de toucher le fichier pour ajouter une carte, modifier ses valeurs ou bien supprimer une carte

ClearScreen

Cette classe n’est utilise que pour aérer l’affichage des informations sur la console lors d’une partie