Python : Projet Card Game

Introduction

Le but du projet est d'améliorer le jeu de cartes commencé en TP, autours de 3 axes : profondeur de jeu, jouabilité, graphisme.

Le projet est à réaliser par groupes de 4 (si vous souhaitez faire des groupes d'une autre taille, faites moi en la requête par courriel à l'adresse benjamin.raynal@rizze.com).

Notation

La notation se fera de la façon suivante :
Juste la base (ce qui était mentionné en TP) : 5/20
Base + 1 axe complètement développé : 9/20
Base + 2 axes complètement développés : 13/20
Base + 3 axes complètement développés : 19/20

Des points supplémentaires sont attribués si certains critères sont respectés.

Axe Profondeur de Jeu

Cet axe consiste a rajouter des possibilités dans le mécanisme de jeu en lui-même : avoir autre chose en main que des serviteurs n'ayant que des points de vie et d'attaque.

Cet axe vous permet de développer vos aptitudes à concevoir une structure de donnée élaborée et les mécanismes liés à son bon fonctionnement.

Vous trouverez ci-dessous des voies d'amélioration de cet axe. Il est recommander d'implémenter un maximum de ces voies, et/ou d'y ajouter vos idées personnelles.

Ajout de propriétés aux cartes serviteurs

En plus des points de vie et d'attaque des serviteurs, vous pouvez leur ajouter des propriétés spéciales, comme par exemple :

Provocation : le joueur et vos serviteurs n'ayant pas provocation ne peuvent être attaqués tant que vous avez un serviteur avec provocation sur le champs de bataille

Bouclier : un bouclier qui absorbe un certain nombre d'attaques ou de points d'attaque, avant que la vie ne soit entamée.

Camouflage : le serviteur ne peut être attaqué avant d'avoir lui-même porter une première attaque

Couleur/élément : vous pouvez ajouter un système de couleurs, qui fait qu'un serviteur sera plus fort/résistant contre les serviteurs de telle couleur, ou au contraire plus faible. Le principe est celui du pierre-ciseau-papier : La pierre est forte contre le ciseau mais faible contre le papier, le papier est faible contre le ciseau, etc...

Ajouts d’actions aux cartes serviteurs

Vous pouvez également rajouter des actions aux cartes serviteurs, comme par exemple :
- invoquer un serviteur
- infliger des dégâts
- redonner des points de vie
- augmenter le nombre de points de vie max
- ajouter/enlever une propriété
- pioche/défausse de carte

Ces actions peuvent être appliquées à :
- l’ensemble des serviteurs alliés/ennemis/alliés et ennemis
- un serviteur allié/ennemi/(allié ou ennemi) choisi aléatoirement/par le joueur/selon un critère

Ces actions peuvent être déclenchées :
- lors de la pose de la carte
- quand on pose un autre serviteur allié/ennemi/quelconque
- lors ce que le serviteur est attaqué
- lorsque un serviteur allié/ennemi/quelconque est attaqué
- lors de la mort du serviteur
- lors de la mort d’un serviteur quelconque/allié/ennemi
- au début/fin de tour

Voici quelques exemples d’actions :
- à chaque début de tour, redonne 1 point de vie à chaque serviteur
- quand le serviteur meurt, invoque deux serviteurs lambda
- quand on pose le serviteur, l’adversaire se défausse d’une carte

Ajout de cartes sorts

En plus des serviteurs, vous pouvez ajouter des cartes “Sort”, qui marchent un peu comme les actions vu ci-dessus, mais qui sont déclenchées lorsque la carte est utilisée (en échange d’un nombre de points de mana).

Des exemples de sorts sont :
- double les points de vie d’un serviteur
- invoque autant de serviteurs de type A qu’il y a d’adversaire sur le plateau
- inflige 3 points de dégats à tous les adversaires
- rend 2 points de vie au joueur
- donne la propriété provocation et +2 en points d’attaque à un serviteur
- tue un serviteur ennemi qui a plus de 3 points d’attaque

Création/gestion de decks

Vous pouvez également améliorer le système de pioche, en permettant aux joueurs de créer leur propre pioche (appelée Deck) : parmi les cartes qu’ils possèdent, ils en choisissent un certain nombre, qui seront classées aléatoirement pour servir de pioche.

Chaque joueur peut ainsi se créer plusieurs Decks, et choisir lequel utiliser en début de partie.

Vous pouvez enfin gérer le gain/déblocage de nouvelles cartes, ainsi qu’un système de gain de points d’expérience au fil des parties.

Axe graphique

Cet axe est destiné à ceux qui souhaitent s'amuser avec les interfaces graphiques en python.
Vous pouvez utiliser la librairie de votre choix, mais je vous recommande pygame, qui est relativement simple d'utilisation. Un tutorial est disponible à cette adresse :
http://fr.openclassrooms.com/informatique/cours/interface-graphique-pygame-pour-python

Le but ici est de remplacer le déroulement d'une partie en mode console par un affichage du plateau de jeu, avec toutes les informations nécessaires :

- les points de vie des deux joueurs
- vos points de mana disponible
- vos cartes en main
- le nombre de cartes en main de l'adversaire
- vos cartes sur le plateau, et celles de votre adversaire, avec leurs caractéristique
- le nombre de cartes dans la pioche (ou les pioches)
- les messages informatifs du déroulement de la partie
- l'historique des actions effectuées

Vous pouvez vous inspirer de HearthStone pour le type d'interface. Ne passez pas des heures a faire les images des serviteurs, ça ne rentrera pas en compte dans la notation. Utilisez plutôt des images (même dépareillées) trouvées via Google par exemple.

Dans un second temps, vous pouvez rajouter des animations lors des attaques et déclenchements d'actions.

Vous pouvez également faire une interface pour le menu principal, choisir le type de partie, et le panneau d'options.

Axe jouabilité

Cet axe est destiné à améliorer la jouabilité. Par défaut, vous devez jouer à 2 sur une même machine, à tour de rôle. Deux voies d'améliorations sont possibles dans cet axe.

Intelligence Artificielle

La première voie d'amélioration consiste à implémenter une IA basique pour remplacer le second joueur. Il ne vous est pas demandé de faire une IA complexe, mais juste un algorithme qui remplace un des deux joueurs:

- en posant des serviteurs aléatoirement
- en attaquant des serviteurs adverses aléatoirement

Partie en réseau local

La seconde voie d'amélioration consiste à faire communiquer par le réseau local votre jeu, un des deux joueurs faisant office de serveur, et le second de client.

Le déroulement du lancement d'(une partie en réseau pourra être :

1. le joueur "serveur" lance le mode serveur de son jeu
2. le joueur "client" lance le mode connexion et rentre l'adresse IP (et le port) du joueur serveur
3. la partie démarre alors, dirigée par le serveur.

Vous pouvez utiliser les sockets python pour la communication :
http://www.tutorialspoint.com/python/python_networking.htm

Points Bonus

Voici des choses qui seront également prises en comptent dans la note de votre projet

Documentation :
Bien renseigner les doc strings de vos classes et fonctions.

Modularité des fichiers :
Séparer le code en modules/fichiers. Par exemple, en faire un par classe, et un pour chaque fonctionnalité : gameEngine, graphic, network, etc...

Rapport :
Faire un rapport, succin mais bien rédigé, qui décrit ce qui à été fait, l'architecture globale, les fonctionnalités, etc… Ne mettez pas de code (ou très peu) dans le rapport.

Archive du projet :
Envoyez moi le projet sous forme d'archive, contenant

- les sources
- un readme disant commant lancer le jeu
- le rapport

L'archive contient le nom des membres du groupe.
L'intitulé du mail doit être "Projet Python <votre promotion> <noms des membre du groupe>"