Choix techniques

PROKOPOWICZ - BOIS

Le langage et les bibliothèques qui vont servir au développement

Quel langage et bibliothèques allez-vous utiliser ? Pourquoi ?

Nous allons utiliser Python pour sa simplicité de prototypage ainsi que son écosystème de librairies et de packages. Nous faisons également ce choix pour des raisons de familiarité avec le langage.

Nous avons prévu d'utiliser de nombreuses librairies :

LIBRAIRIES STANDARD :

- *__future__* pour l'annotation de notre code.
- *random* pour la génération procédurale de notre carte, les déplacements des monstres…
- *itertools* pour gérer l'itération sur les données
- *logging* pour gérer les retours du programme sur le terminal, dans un fichier de log…
- *math* pour utiliser des fonctions mathématiques de base
- *typing* pour réaliser un code propre et typer le code
- *collection.deque* pour utiliser les structures de base de pile/file

LIBRAIRIES EXTERNES :

- *pygame* pour la partie graphique/interactivité. Nous utilisons ce module car c'est un module python réputé pour la de création de jeux lors de projets (étudiants ou non).
- *pilow* pour réaliser une analyse des tilemaps, générer des tuiles et gérer les images affichables dans les sprites plus tard
- *rich* pour réaliser des affichages plus propres dans le terminal lors du cycle de développement

D'autres modules pourront être ajoutés à cette liste non exhaustive. Ces choix seront réalisés durant le cycle de développement, et seront justifiés par un besoin standard ou déjà implémenté sur ces mêmes modules.

Décrivez la structure de données qui va être mises en place pour modéliser votre plateau. Vous pouvez donner un exemple si nécessaire.

Notre plateau de jeu sera modélisé par le biais d'une liste de listes. Chaque élément de ces listes internes représentera une case du plateau, implémentée dans une classe particulière.

Ce plateau pourra également être conteneurisé dans une classe, afin de lui ajouter des méthodes courantes, tel que la recherche d'élément ou bien la recherche des cases voisines d'une case. Cette classe sera étroitement lié à la génération procédurale des cartes. Nous implémenterons donc probablement des méthodes de classe pour simplifier l'initialisation à partir d'une double liste d'URL vers des textures.

Décrivez la structure de données choisie pour représenter un joueur

Les joueurs seront représentés par des classes contenant un identifiant et une liste de cartes pouvant être jouées par ce joueur.

Les cartes seront représentées par une méta-classe, ainsi qu'éventuellement des sous-classes implémentant des fonctionnalités spécifiques si besoin est.

Chaque pion sur le terrain (pour rappel, tous les pions sont contrôlés par tous les joueurs) sera représenté par une classe spécialisée (concept de pion « élémentaire ») héritant d'une classe de base Pion.

Décrivez la structure de données choisie pour représenter les ennemis

Les ennemis seront représentés par une classe principale Ennemy. Cette classe définira des interactions de base, qu'il sera possible de personnaliser à l'aide de méthodes diverses ou lors de l'initialisation de l'ennemi. Cette classe possédera des propriétés, dont une « point de vie » qui, à l'aide de son setter pourra déclencher automatiquement la récompense pour l'élimination du monstre.

Décrivez la structure de données choisie pour représenter les objets

Les objets seront représentés par plusieurs classes :

- Une classe « MoveCard » qui représente une carte de déplacement. C'est une des récompenses principales du jeu.
- Des classes spécifiques pour de potentiels objets uniques, si le développement est assez avancé et le temps le permet.
- Une classe générique : on peut très bien considérer un objet comme une simple récompense changeant une variable dans le code. Cette classe générique stockera une fonction dans sa mémoire qui sera exécutée au ramassage de l'objet.

Tant qu'ils seront représentés sur la zone de jeu, les objets seront de simples sprites. Une classe sera dédiée aux éléments «ramassables », qui gardera en mémoire le sprite et se chargera d'instancier les classes susnommées appropriées.

Donner une liste des sous-programmes ou des méthodes que vous pensez utiliser (préciser leur utilité, si ce n'est pas clair)

Nous pensons fragmenter notre code dans de nombreux packages (sous-programmes segmentés dans des dossiers différents) :

- Un package sera dédié aux composants graphiques personnalisés que nous auront créé (principalement, le menu principal, le menu « pause » …)
- Un autre sera dédié à la gestion des sprites sur la zone de jeu
- Nous isolerons le code principal dans une classe Game qui se chargera de la logique centrale et de la cohésion du jeu.
- Un module important de notre code sera le module wave_collapse.py qui possédera tout le nécessaire pour créer/générer des cartes originales à chaque partie à l'aide d'une implémentation de la wave function collapse. Nous aurions souhaité nommer ce module « wave.py », mais un module standard porte ce nom (traitement de fichiers audio).

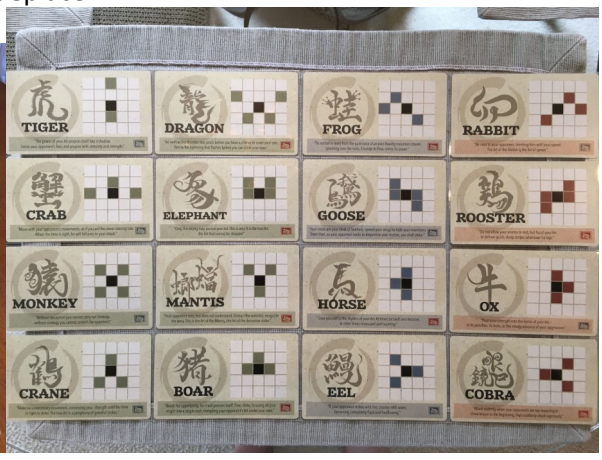
ANNEXE : références graphiques et captures d'écran

À ce stade du document, certains éléments restent flous ou peu compréhensibles. Nous allons donc indiquer ci-dessous certaines références desquelles nous nous sommes inspirés pour créer notre concept de jeu original.

Tout d'abord, le système de jeu coopératif est inspiré du jeu de plateau Magic Maze :

Les joueurs doivent dans ce jeu coopérer pour se déplacer dans un labyrinthe, sans avoir le droit de parler, en temps réel. Un sablier délimite le temps de jeu, et l'exploration est également un facteur clé.

Notre système de cartes est quand à lui inspiré du jeu de société Onitama, semblable à un jeu d'échecs simplifié mais avec des cartes pour se déplacer :



Les points noirs au centre représentent le pion que l'on souhaite déplacer, tandis que les cases colorées représentent un case sur laquelle on peut se déplacer. Un jeu mobile gratuit a d'ailleurs été créé par l'éditeur, Asmodee. Nous vous invitons à tester ce jeu pour mieux comprendre ces déplacements. Nous allons plus ou moins recréer ce style de déplacements, en représentant les cases dans des doubles listes de 5 par 5, tout comme Onitama.