10/29/2020

Rescue Princess

Asalhi Adel
BASNET Devashish

Contents

Circonstance

Dans le cadre de la création de sa plateforme de jeu, Monsieur J. Joestar nous demande de programmer un mini-jeu de type réflexion/puzzle. Celui-ci devra être simple à prendre en main et ludique. Afin de faciliter le lien entre le jeu et la plateforme, Monsieur Joestar demande que le jeu soit programmé en python. L'interface graphique sera simple et dans un style plutôt rétro afin de correspondre avec le thème général de la plateforme du client.

Présentation du projet

Le but de notre projet est de créée un jeu, qui sera un mini-jeu jouable en console par ligne de commande. Ce jeu est un petit labyrinthe représenté par une grille d'une taille définie avec des obstacles représentés par un caractère spéciale choisis, qui seront ajoutés dans la grille de manière aléatoire, le caractère aura pour but en se déplaçant d'atteindre la case finale pour valider le niveau.

Intervenants

Intervenant	**Client**	**Développeurs**
Nom	Joseph Joestar	D.BASNET, A.salhi
Tél	0466666666	0483121066
Mail	j.joestar@jojo.jp	python@hotmail.com
Ville	Bruxelles	Waterloo

Cible - Utilisateurs

Les cibles de ce jeu sont les joueurs de jeux vidéo. Mais aussi tout types de personnes voulant simplement passer du temps. En effet les jeux présents sont des jeux rapides et simple qui permettent de raccourcir les trop longs moments d'ennui.

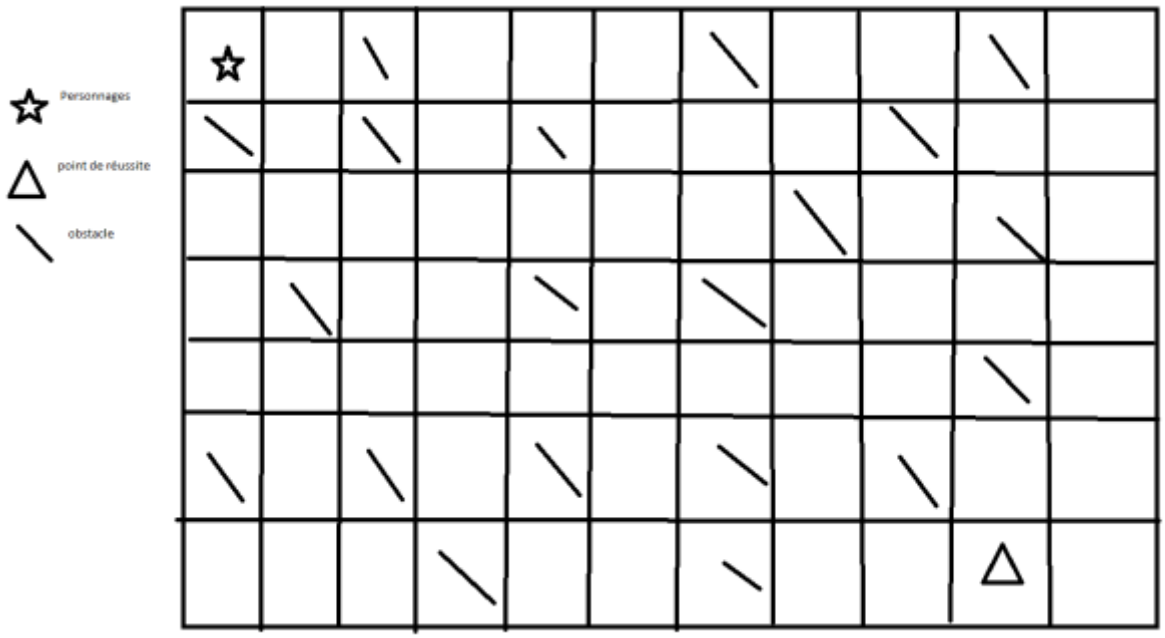
Charte Graphique et Ergonomique

L'interface sera composée d'une grille représentant un labyrinthe. Le personnage contrôlé, les obstacles et la case finale seront représentés par différents caractères définis.

Un tableau récapitulatif sera mis en place dans le but de légender les caractères utilisés. Cet affichage sera visible en console.

Cette interface sera visible en console, il y aura également une interface graphique assez simple de type « rétro ».

Interface



Interface : l'affichage en console

Planification et Organisation

Le projet doit être rendu et fonctionnel pour la fin du mois de décembre 2020.

Des démos et réunions seront planifiés chaque mercredi matin vers 9h30. Ces réunions serviront à montrer l'état d'avancement du projet et de discuter les éventuels problèmes. Ces réunions s'effectueront durant tout le projet.

Planification

Plan technique du projet (26 octobre 2020 – 30 octobre 2020)

- Création du cahier des charges
- Création du MVP
- Création du diagramme UML

Codage (2 novembre 2020 – 6 Novembre 2020)

- Codage de l'ensemble du projet en Python

Test (9 novembre 2020 – 13 novembre 2020)

- Vérification des fonctionnalités du programme
- Vérification de l'ergonomie du programme

Démo Finale du Programme (date inconnue)

- Présentation du programme et défense du projet.
- Éventuels changements de dernières minutes sur demande du client.

Contraintes

- Sur demande du client, le langage de programmation sera le Python 3.8.
- Le programme devra fonctionner sans erreurs, les commandes de l'utilisateurs seront à rentrer sous forme d'input en console.
- L'interface graphique sera simple et de type rétro.
- Le jeu doit être simple à prendre en main, il doit être fluide.

Demandes Fonctionnelles

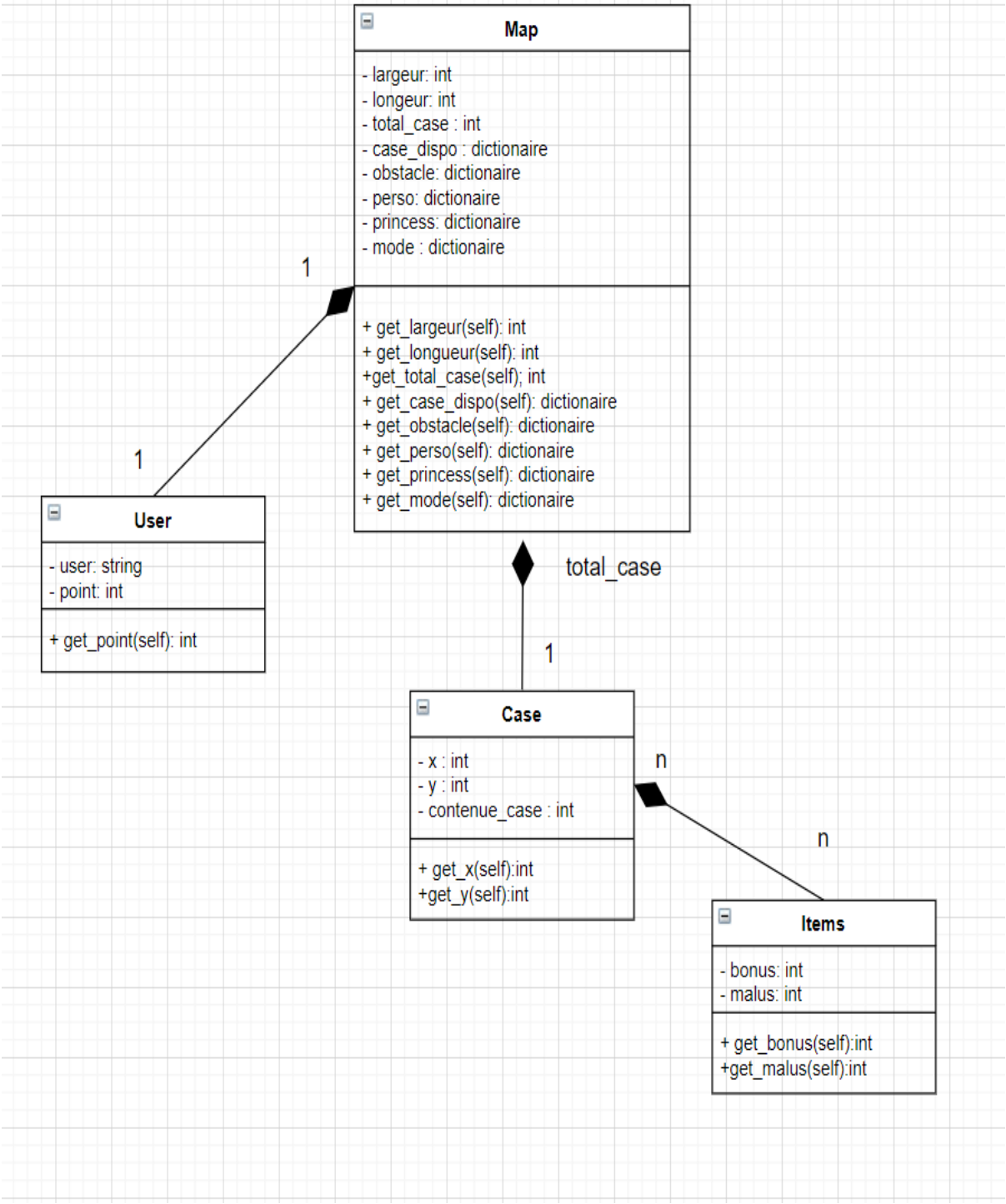
- Une map composé de cases dont la largeur et la longueur seront définis par l'utilisateur via un input.
- Des obstacles placés aléatoirement sur la map, définis en fonction du nombre de case totale (exemple : si total case < 20 && >10 → 7 obstacles, si total case < 30 && >20 → 10)
- Un personnage qui aura la possibilité de se déplacer (haut, bas, droite, gauche).
- Un point de fin de niveau immobile qui sera représenté par la princesse.
- Des bonus placés sur des positions aléatoires au chargement de la map qui ajoute 1 à 5 points au score. (En fonction du nombre de case totale)

- Des malus placés sur des positions aléatoires au chargement de la map qui retire 1 à 5 points au score. (En fonction du nombre de case totale)
- Enregistrement de pseudo dans une base de données à la fin de la partie.
- Affichage des scores trié

Fonctionnalités

- Possibilité d'enregistrer son score et son pseudo
- Création d'un mode de jeu (différent du principale)
- Ajout d'item (bonus/malus)

Diagramme de classe UML :
Princess Rescue



Map :

La classe map correspond à la génération de la map. L'utilisateur introduira la taille de la map via un input, il définira la largeur et la longueur qui sont les attributs de la map de plus que le mode souhaité, ensuite les obstacles, la princesse et le personnage seront généré aléatoirement sur la map.

User :

Une fois l'utilisateur ayant atteint la princesse, il aura fini la partie, il verra son score ainsi que la taille de la map et un input demandant d'entrer son pseudo.

(Son pseudo ainsi que son score et la taille de la map seront enregistrés dans une base de données)

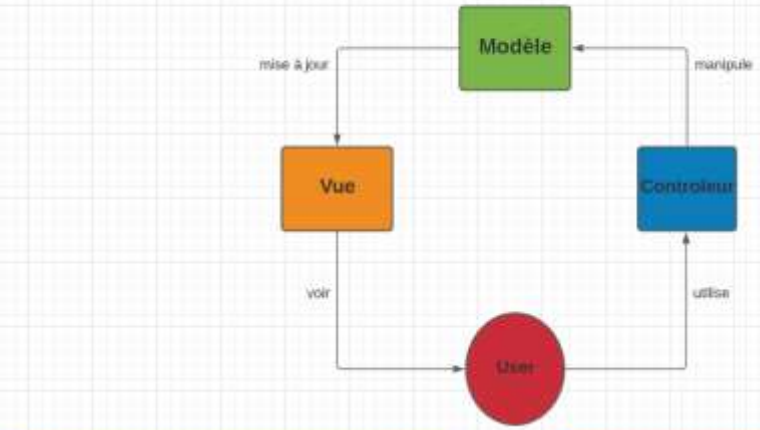
Case :

X et y sont les coordonnées de chaque case, contenue case définit le contenue de la case (par exemple : 0 = case dispo, 1 = obstacle, 2=princesse)

Items :

Bonus sont des additions au score de l'utilisateur à l'inverse malus qui sont des soustractions.

Schémas architecture Princess Rescue



Modèle : représente le jeu, le labyrinthe constitué de ses différents caractères

Vue : représente la console que l'user verra dans la console affiché par des printf

Controleur : sa responsabilité est de traiter les inputs et de les renvoyer

User : a pour but d'influencer les déplacements

Justification : nous avons choisit ce format de cycle, car il respresente le mieux notre projet dont le fonctionnement est répétitif et basique. On observe bien dans le diagramme que chaque éléments dépend du précédent pour continuer le cycle.