


# Projet PFA

2020-2021

## Guide du jeu

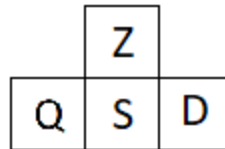
Afin de lancer le jeu, il faut exécuter la commande 'dune build' dans le répertoire racine. Si tout s'est bien passé, le répertoire '\_build' est apparu. Il faut ensuite lancer 'index.html' avec le navigateur web Firefox. Normalement, le jeu se lance une fois la page ouverte.

Notre jeu s'inspire du roguelike « The Binding of Isaac ». Le joueur  peut effectuer diverses actions au cours du jeu :



---

Déplacement



Tirer

SPACE

Lorsque le joueur arrive dans une pièce avec des ennemis, il peut en changer grâce aux portes situées aux extrémités de cette dernière. Il est libre d'en changer quand il le souhaite, peu importe qu'il y ait encore des ennemis à l'intérieur. Néanmoins, pour passer au niveau/étage suivant, il doit battre tous les ennemis qui s'y trouvent. Chaque étage se compose de 5 pièces comportant plus ou moins d'ennemis. Une fois tous les ennemis d'un étage vaincu, le joueur passe automatiquement à l'étage suivant. Chaque étage est plus dangereux que le précédent, le nombre d'ennemis et leurs points de vie augmentent .

Il y a trois type d'ennemis :



Le goblin : attaque au corps à corps, rapide, beaucoup de point de vie, court vers le joueur



---

L'araignée : attaque au corps à corps, rapide, peu de point de vie, pose des pièges dans la salle, rebondit sur les murs.



Le squelette : attaque au corps à corps et distance, lent, point de vie moyen, court vers le joueur

Lorsque le joueur est touché par une attaque, il perd un point de vie (symbolisé par les cœurs en haut à gauche) mais il a la possibilité de regagner ses points de vie (PV). En effet, certains ennemis, une fois mort, lâcheront aléatoirement un cœur qui lui permettra de regagner des PV. Lorsqu'il est touché, il est invulnérable quelques secondes le temps de récupérer mais s'il perd tous ses PV, c'est la fin de la partie.



De plus, il gagne un objet qui lui donne un bonus lorsqu'il complète un étage (dégâts augmentés, meilleure vitesse de déplacement, vitesse des projectiles augmentée).

---

Il y a 9 objets réparti en 3 catégories :

Les bottes : augmente la vitesse de déplacement de 15%, 30% et 45%.



Les arcs : augmente les dégâts de 100%, 300% et 400%.



les carquois : augmente la vitesse des projectiles de 50% , 100% et 200%.



## Implémentation

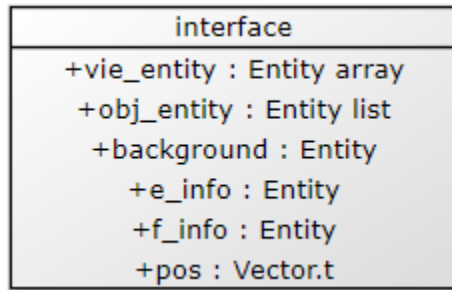
Le jeu est développé selon le modèle ECS. On a utilisé le squelette de base du cours et on y a rajouté des composants. Dans notre dossier game, on retrouve toutes les entités de notre jeu ainsi que le fichier game\_state.ml qui va contenir les modèles de notre jeu (Niveau, interface, statut du joueur, inventaire).

L'état du jeu :

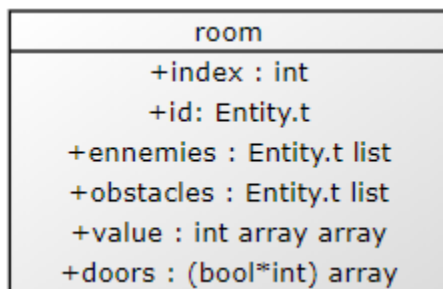
room
+index : int
+id: Entity.t
+ennemies : Entity.t list
+obstacles : Entity.t list
+value : int array array
+doors : (bool*int) array

---

L'interface :



Une salle :



Les composants rajouté en plus des composants du squelette de base sont :

BoxCollider : qui, contrairement à box qui va donner les tailles des sprites, va donner la taille de la hitbox.

Priority : il donne l'ordre de priorité dans laquelle l'entité doit être affiché à l'écran exemple le background doit être affiché en premier pour ne pas superposer le joueur.

Teleport : utilisé par les portes, il donne la position où doit être placé le joueur après être entré en collision avec une porte.

Surface : le composant surface à été modifié pour y avoir des animations et des tilemap (Un fond créé à partir d'un tableau entier et une palette de couleur).

InvulnerableFrame : contient le nombre de frames dont le joueur ne peut pas être touché.

Orientation : Dans quelle direction le joueur regarde.

Health : Point de vie de l'entité.

Active : Boolean qui représente si une entité est active dans le jeu.

Statistics : représente les statistiques de l'entité (Force , vitesse de déplacement et vitesse des projectile).

---

TextD : Permet au entité de l'interface d'avoir un texte descriptif.

Cpt : Compteur qui réalise une action toutes les x secondes.

Pour les systèmes, il n'y a qu'un seul système en plus des systèmes de base vue en cours.

Cleaning\_system : Si une entité est inactive alors on enlève tous ses composants.

Pour ce qui est des inputs en plus du gestionnaire de base, il nous fallait un table qui les garde en mémoire lorsque plusieurs étaient appuyés en même temps sinon le mouvement pouvait être saccadé.

Il y a pas eu de répartition du travail particulière, même si elle c'est faite selon nos envies et surtout selon nos niveaux en programmation.