

# Rapport du projet : Rogue-Like

## Manuel Utilisateur

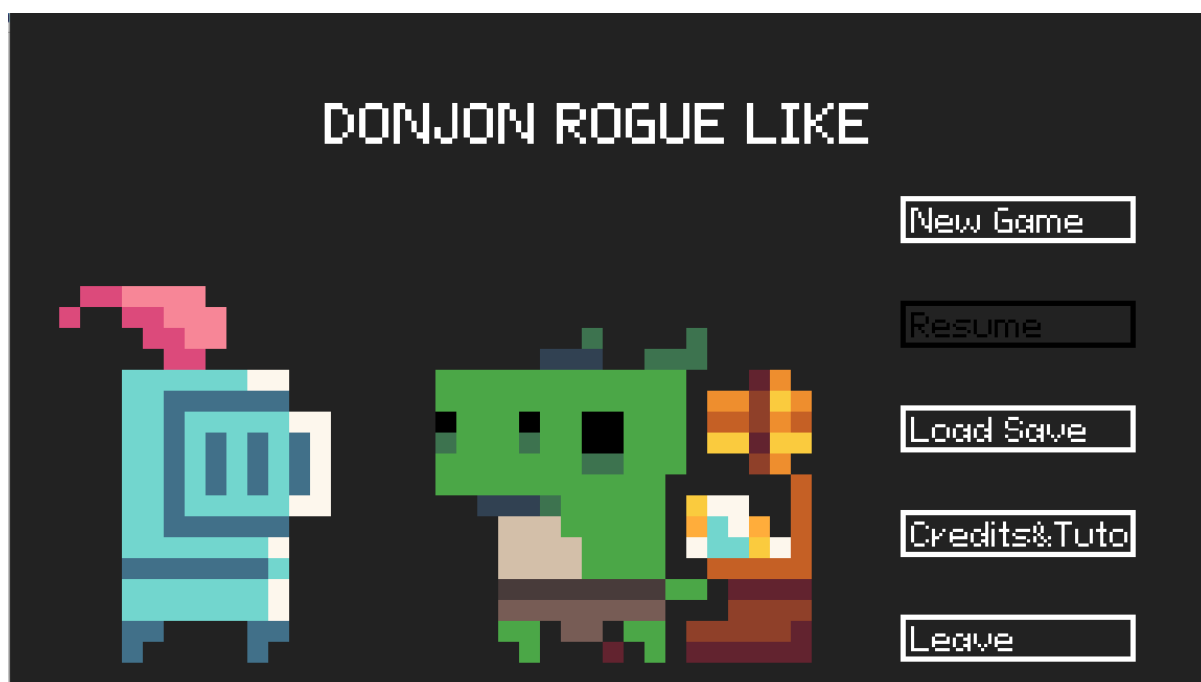
### But du jeu :

Ce jeu reprend le principe des jeux type « Rogue-like », des jeux en tour par tour demandant au joueur de progresser au fur et à mesure dans différents d'un donjon au sein duquel les éléments (monstres, objets, escaliers...) sont placés aléatoirement et où la difficulté augmente progressivement. Le joueur doit avancer au sein du donjon le plus loin possible tout en survivant, à l'aide de son équipement et de son ingéniosité, aux assauts des monstres qui l'habite.

### Comment jouer au jeu :

#### Le Menu :

Au démarrage, l'utilisateur se retrouve sur la page de menu lui proposant différentes possibilités :



- New Game : Lance une nouvelle partie avec un nouveau donjon généré aléatoirement.
- Resume : Permet de relancer une partie après que celle ci ai été mis en pause.
- Load : Permet de charger la sauvegarde d'une précédente partie.
- Credits&Tuto : Affiche les crédits ainsi que les différentes touches d'action que le joueur peut presser.
- Leave : Ferme la fenêtre de jeu et stop le programme.

## Déroulement de la partie :



Le joueur est représenté par un soldat en armure. Au lancement de la partie, celui ci se retrouve au centre du premier étage du donjon.

Un coffre se situe sur une des cases qui lui sont adjacentes et contient aléatoirement 2 objets afin de l'aider à débiter son aventure.

### Interface :

L'interface de l'utilisateur se situe à droite et indique toutes les informations importantes concernant le personnage :

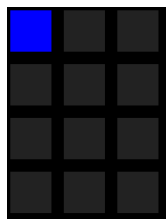
-Les points de vie : HP 100/100

-Les points de magie : MP 50/50

-Les points d'expérience : EXP 0/350

Le mode d'attaque : E : mode = 1

Et enfin l'inventaire :



-Les statistiques : Atk 10  
Def 10  
Int 10  
Lvl 0

Deux autres informations sont également affichées à l'écran :

Le numéro de l'étage en bas à gauche de l'écran, et la position relative de l'escalier menant à l'étage suivant en bas au centre de l'écran.

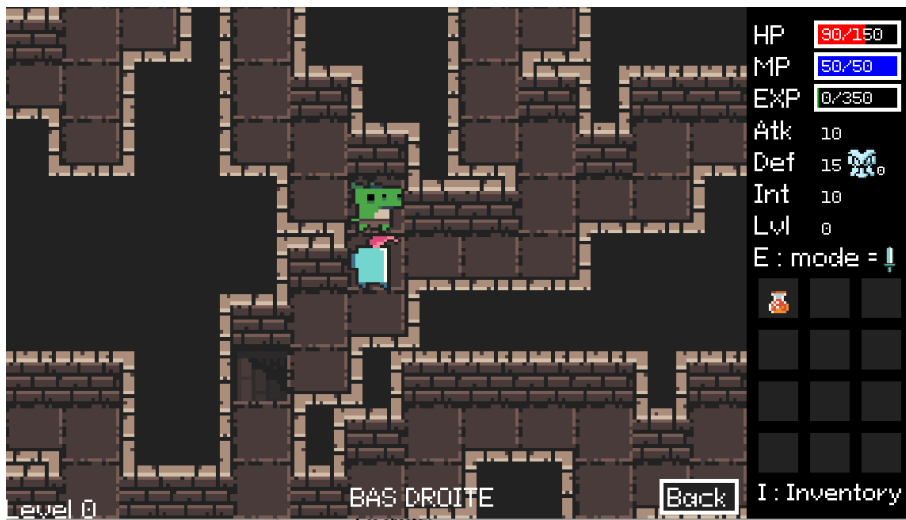
### Les touches :

Le joueur peut se déplacer à l'aide des flèches directionnelles ou des touches Z,Q,S,D (respectivement haut,gauche,bas,droite) .

Il peut également parcourir son inventaire en pressant la touche I puis s'y déplacer à l'aide des flèches directionnelles ou en cliquant directement sur les objets pour les sélectionner. Il peut alors interagir avec l'objet à l'aide de la touche E ou s'en débarrasser à l'aide de la touche T.

Enfin, il peut presser la touche Echap pour ouvrir le menu ou sortir de l'inventaire si il s'y trouve

### Les combats :



Les monstres sont représentés par des dragons sur 2 pattes. Lorsque le joueur se retrouve sur une case adjacente à ceux-ci, le monstre l'attaque et le joueur subit des dégâts faisant diminuer ses points de vie. Pour l'attaquer à son tour, le joueur doit presser la touche directionnelle correspond à la position du monstre ( ici par exemple le joueur doit presser la touche « ↑ » ou Z pour attaquer). Une fois le monstre vaincu, celui lâche un trésor et le joueur d'expérience. Si ce dernier a assez de points d'expérience, il augmente alors d'un niveau augmentant ainsi ses statistiques.



### Les objets :

Les objets peuvent être récupérés à l'intérieur des coffres (en marchant sur ces derniers) et sont alors placés dans l'inventaire.

Il en existe différentes variétés ayant leur effets propres :

- Les potions : elles peuvent être bues à n'importe quel moment par le joueur et leur effets dépendent de la couleur de celles-ci :



→ restaure 10 % des points de vie.



→ restaure 20 points de vie et 10 points de magie chaque tour pendant 30 tours.




→ augmente le nombre de points d'expérience obtenus de 30 % pendant 30 tours.





→ restaure les points de magie de 10 %.



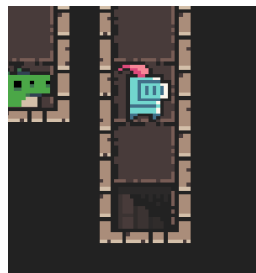
→ augmente les chances de coups critiques pendant 30 tours.

- Les armures  : confère un bonus de défense au joueur

- Les épées  : confère un bonus d'attaque au joueur

- Les baguettes  : confère un bonus d'intelligence au joueur

### Le changement d'étage :



En se rendant sur un escalier, le joueur est capable de changer d'étage.

S'il s'agit de l'escalier au centre de l'étage, le joueur remonte d'un étage (si il ne se trouve pas à l'étage 0).

S'il s'agit de l'escalier en coin d'étage, le joueur descend d'un étage et se retrouve au milieu de l'étage suivant.

# Manuel Développeur

## **Modularité :**

Un fichier makefile est a disposition pour pouvoir compiler le projet.

Après avoir fait make il suffira de lancer la commande ./Donjon dans le terminal pour jouer au meilleur jeu rogue like de la génération.

La modularité du projet a suivi respectivement l'énoncé. Ainsi le rendu contient les dossiers suivant :

-bin qui contient les fichiers générés par la compilation.

-include qui contient les headers des fichiers.c.

-src qui contient bien évidemment les fichiers sources

-save qui contient les différents fichiers de sauvegardes nécessaires pour le jeu

-sprites qui contient les différentes ressources graphiques nécessaires pour le jeu

Et enfin le rapport qui contient le manuel utilisateur et développeur ainsi que le dev\_log sont présents dans le rendu final.

## **Interface Graphique :**

Le projet utilise la merveilleuse interface graphique MLV. Nous n'avons pas eu de problème à l'utiliser car maintenant que nous sommes en L3 nous la maîtrisons relativement. Tout est géré dans le module graphique.h

Cependant à cause d'un souci de mémoire et d'optimisation nous avons décidé de créer une (énorme) variable globale « spriteimage » qui contient tout les sprites du jeu.

En effet nous avons remarqué (en tout cas sur nos machines) qu'il fallait mieux les charger qu'une seule fois ; de plus lorsque nous recadrons une image, celle-ci se décalait d'un pixel en taille et en position. Ce qui devient très gênant quand à chaque tour de jeu nous faisons un affichage qui actualise tout les sprites.

En jeu la fenêtre est donc composé du plateau qui affiche les (13x9) cases autour du personnage. Sur la droite un affichage est prévu pour les statistiques du personnage ainsi que son inventaire. A chaque tour de jeu ces trois fonctions sont appelés pour rafraîchir l'affichage de jeu. Des fonctions pour naviguer dans les interfaces et utiliser les touches et la souris sont également appelées.

## **Structure du programme :**

le programme est structuré en plusieurs modules. En général un module a été crée pour chaque grand type que nous manipulerons dans le programme. Le module action est le module « principale » qui réunit toutes les actions pour faire réagir les différents types entre eux. Par exemple la fonction deplacePerso(Position \*position, Plateau \*etage, Direction \*direction) qui a besoin des modules position.h, plateau.h et direction.h pour pouvoir faire bouger la position du joueur dans l'étage du donjon selon la direction.

La partie graphique est géré par MLV est utilisé uniquement dans le module prévu « graphique.h » à cet effet.

Il y a également un module fichier.h qui importe tout les modules types. Il s'occupe de gerer les fichier de sauvegarde en écriture et en lecture. Ces fichiers sont générés spécialement dans le fichier « save » conçu pour l'occasion. Les sauvegardes sont écrites en binaires, les informations utiles sont alors écrites de manière brute à l'aide de manipulation de bit dans des suites de unsigned long long.

Note :

- Par manque de temps nous n'avons pas eu le temps de vérifier si une sauvegarde est legit ou non. Au lancement du programme l'exécutable considérait que la sauvegarde est bonne.

- A l'ouverture du fichier l'utilisateur verra une suite de très gros nombres, il ne pourra modifier les fichiers de sauvegardes sans trop comprendre ce qu'il fait et aura 99 % de chance d'avoir une sauvegarde bugé.

Voici donc la liste des différents modules orienté type :

- item.h qui contient la définition des différents types d'objets, qui sont eux mêmes définit dans différentes structures.

- inventaire.h qui est en réalité un tableau de 12 items qui peuvent être vide ou présent dans celui-ci.

- perso.h qui est une structure des différents attributs qui caractérise le héros.

- monstre.h qui est également une structure et qui fonctionne a peu près comme perso.h

- position.h et direction.h sont des structures qui permettent d'avoir les informations nécessaires sur la position du joueur, son étage, ect.

- terrain.h qui est un gros module qui contient des cases et un tableau à double dimension de case qui forme un étage. Ainsi un tableau de terrain initialisé dans le main permet de former le donjon tout entier.

Pour plus de détail veuillez vous rediriger vers les headers des modules en questions ou chaque fonctions sont documentés.

## **Note et difficulté rencontré :**

Par manque de temps nous n'avons pas eu le temps d'équilibrer le jeu, même si il reste largement jouable. Nous aurions voulu également faire des améliorations, comme par exemple faire des animations, des étages spéciaux ou alors des monstres et items encore différents.

La graine aléatoire est basé sur le lancement du programme une save chargée sera alors différentes dans le temps si elle est lancée plusieurs fois.

Le joueur peut aller dans le menu, reprendre sa partie ou en charger une autre à tout moment. Le partie se sauvegarde automatiquement et seulement quand on quitte le jeu par le menu principal.

Le projet a été faisable dans son ensemble même si la manipulation de la mémoire avec la réallocation des étages par exemple n'a pas été de tout repos. Heureusement l'aide de Monsieur Nadime Francis nous a été précieuse.