Rapport de projet

Introduction

Plongeons dans l'univers passionnant de la création du jeu Malloc World, une aventure formidable orchestrée par les esprits brillants d'Aymen BENTALHA et Joackim SEGOR. Au fil de cette quête, ces étudiants exceptionnels ont entrepris de suivre scrupuleusement plusieurs étapes cruciales, bravant ainsi les dangers du monde du C. Découvrons ensemble la magie de leur collaboration et les défis surmontés pour donner vie à ce projet captivant.

Répartition

Aymen s'est donc chargé des déplacements du joueur, des collisions, des changements de zone, de la collecte, du levelling et des combats.

Joackim s'est chargé de la map, du joueur, du pnj, des objets, des crafts, des réparations, de la sauvegarde et restauration du jeu.

Analyses

Dans ce projet nous avons donc plusieurs structures de données :

Coordonnée :

```
struct Coordonnee{
    int x;
    int y;
};
```

Object :

```
typedef struct Object
{
    int id;
    int durability;
    int quantity;
    int damage;
} Object;
```

Player :

```c
typedef struct Player
{
    int hp;
    int maxHp;
    int xp;
    int xpNext;
    int level;
    Object* inventory;
} Player;
```

Npc :

```c
typedef struct Npc{
    Craft* crafts;
    Chest* chest;
}Npc;
```

Chest :

```c
typedef struct Chest{
    Object object;
    struct Chest* next;
}Chest;
```

Craft :

```c
typedef struct Craft{
    int id;
    Object* composent;
}Craft;
```

Monster :

```c
struct Monster{
    char* name;
    int pv;
    int experience;
};
```

Les fonctions les plus utiles sont :

```c
int*** initMap()
void drawMap(int** map[])
```

Pour générer et afficher la map.

```c
Player initPlayer()
Npc initNpc()
Player initChargePlayer()
```

```
int addInventory(Object* inv,Object o)
void showInventory(Player p)
void withdrawOfChest(Player p,Npc npc)
void storeInChest(Player p, Npc npc)
void withdrawInventory(Player p, Object o)
void craftObject(Craft* c,Player p)
void repair(Object* inv)
```

Pour initialiser le Player et le Npc puis ajouter/retirer des Object à l'inventaire ou au coffre.

```
Object initObject(int id,int q)
int stackable(Object* o,Object* ob)
```

Pour initialiser un Object et pour vérifier si il est empilable.

```
int initMonster(int id, Player p)
```

Pour initialiser les monstres.

```
int checkMovement(int resultTab, Player p, int zone, Npc npc)
```

Pour gérer les collisions.

Installation

Rendez-vous sur le GitHub du projet :
https://github.com/TheJoker971/MallocWorld

Et télécharger le fichier « ***app.exe*** ».

Mettez celui-ci dans un dossier vide puis ajouter un dossier « ***saves*** ».

Attention ! : Si vous sauvegardez dans un même fichier celui-ci sera écraser vérifiez bien que les noms des fichiers soient différents pour éviter les mauvaises surprises.

Utilisation

Après avoir effectué l'installation ci-dessus vous pourrez exécuter le fichier « ***app.exe*** » et profiter de l'expérience du jeu Malloc World.

La limite de caractère lors d'une sauvegarde est de 29 et vous n'avez pas besoin de rajouter l'extension du fichier cela se fait automatiquement.

Exemple :

```
Entre le nom de la sauvegarde du player : PlayerSave1
```

Celle-ci se retrouvera dans le dossier « ***saves*** » créer préalablement et sous le nom « ***PlayerSave1*** » avec l'extension « ***.txt*** ».

Pareillement lors du chargement d'une sauvegarde.

Exemple :

```
Entre le nom de la sauvegarde du player a charger : PlayerSave1
```

Pour se déplacer il faut entrer les touches suivantes « z » pour monter, « s » pour descendre, « q » pour aller à gauche et « d » pour aller à droite lors de la saisie de l'une de ces touches appuyer sur « entrée » et pour quitter le jeu la touche sera « ; ».

Bilan

Nous n'avons pu réaliser la partie VI du projet ni ajouter les fonctionnalités permettant de réduire les dégâts avec le plastron.

Difficultés :

Pour ce projet nous avons eu du mal à nous organiser puis à nous familiariser avec GitHub. Par la suite nous avons rencontrer des petits bugs qui se manifestait que sur le Mac et pas sur le MSI.