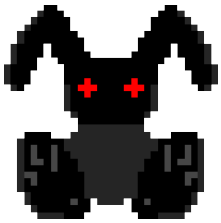


TekDoom

Jeu de tir à la première personne

Accessible uniquement sur excellent résultat au Wolf3D

- Le Laboratoire aux Lapins Noirs -
lapinsnoirs@epitech.eu

Ce projet consiste à construire un programme affichant une perspective interne d'un labyrinthe. Le labyrinthe devra présenter des variations de hauteur au plafond, au sol ainsi que des murs formant des angles différents de 90°.

Aucune ressource technique ne sera à votre disposition pour ce projet du coté de l'e-learning: vous devez rechercher vous-même un moyen de construire cette projection 3D.
Une piste pour démarrer : comment Doom a-t-il été réalisé ?

Nom du dépôt de rendu : tekdoom

Index

1 – Détails administratifs

Votre dépôt de rendu doit s'appeller tekdoom.

Votre programme devra être compilé avec un Makefile.

Votre binaire devra être nommé « tekdoom ».

Vous devez respecter la norme.

Votre rendu ne devra pas comporter votre programme compilé, un fichier .o ou un fichier tampon type "#*" ou "*~" sous peine d'avoir 1,5.

Votre programme doit compiler avec -W -Wall -Werror.

Les lignes de liaison avec les bibliothèques doit être les suivantes :
-I/home/${USER}/.froot/include
-L/home/${USER}/.froot/lib
-llapin -lsfml-audio -lsfml-graphics -lsfml-window -lsfml-system -lstdc++ -ldl -lm

Votre programme **devra** s'exécuter en 1024*768 ou au-delà.

La première ligne de code votre main, directement après les déclarations de variables, **doit** être un appel à la fonction bunny_set_maximum_ram. La quantité **maximale** de RAM autorisé pour ce projet est de **30Mo**.
La quantité de RAM est susceptible d'être modifié durant l'évaluation.

Vous **devez** appeler la fonction bunny_set_memory_check en lui passant true, cela avant de quitter votre programme ou encore plus tôt.

La taille **totale** de votre dépôt ne **doit pas** excéder **100Mo**.

Prenez garde à régler les droits d'accès de votre dépôt et de vos dossiers et fichiers.

2 – Fonctions autorisées

Ci-dessous, la liste des fonctions systèmes autorisées pour réaliser ce projet :

open, close, read, write
alloca, setjmp, longjmp, assert
srand, rand, time,
opendir, readdir, closedir

L'intégralité des fonctions de la libdl
L'intégralité des fonctions nécessaires à la mise en place d'un réseau.
L'intégralité de la lib pthread et d'OpenCL.
Ainsi que l'intégralité de la LibMath.

En cas de jeu en réseau, il n'y a aucune restriction de fonction coté serveur.

Ainsi que les fonctions de la LibLapin présente dans lapin/basic.h

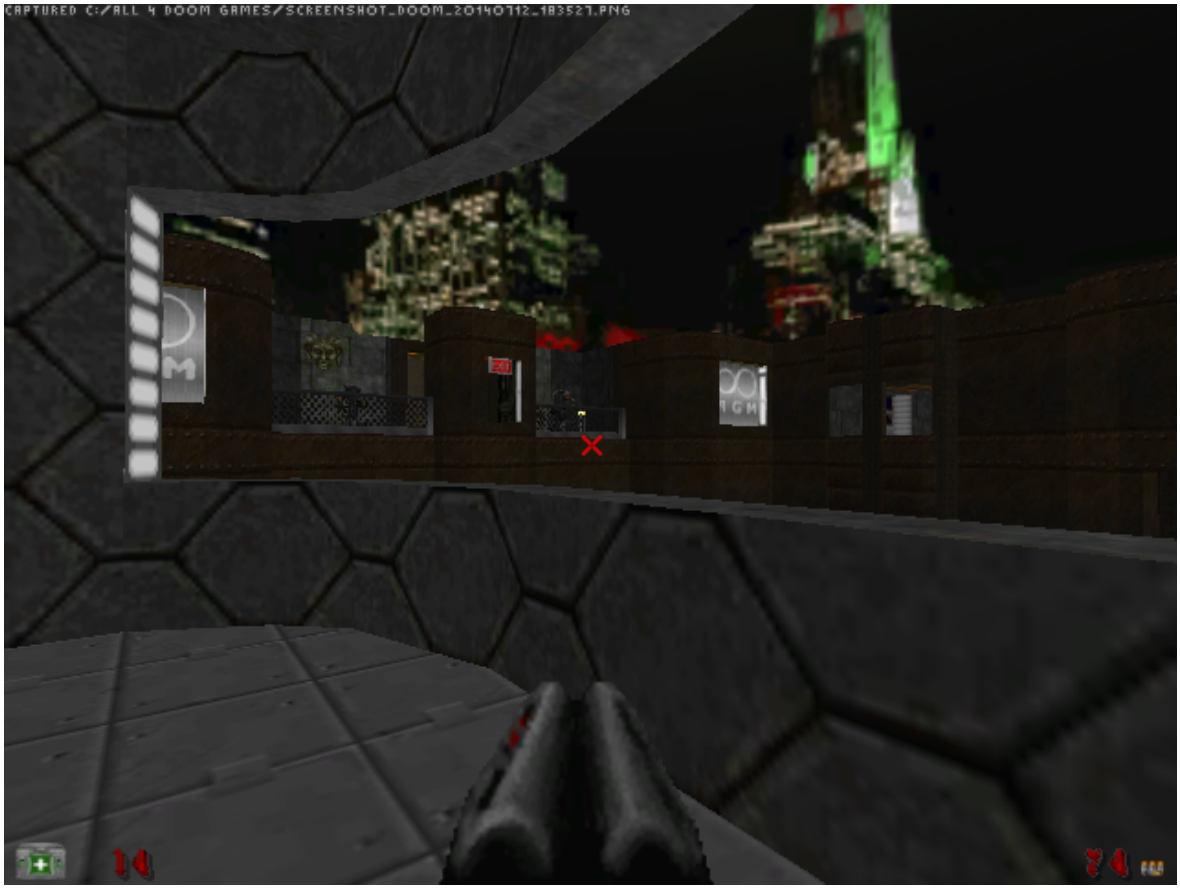
Toute fonction permettant d'allouer de la mémoire en dehors de bunny_*alloc, alloca ainsi que tous moyens de précipiter la fin du programme (exit...) sont **interdits**.

Les fonctions présentes dans lapin/advanced.h sont soumise à condition :

Toutes les fonctions **associées à un pointeur sur fonction** portant un nom type "gl_bunny_my*" doivent avoir été reprogrammée parfaitement. Une fonction disponible dans lapin/self_test.h vous aidera à vérifier leur fonctionnement. Les autres sont autorisées.

Si la fonction de test vous indique que votre fonction n'est pas une copie conforme, vous ne pouvez **pas** utiliser la fonction.

3 – Sujet



Ce projet consiste à intégrer à votre Wolf3D les évolutions qui démarquent ce jeu de son célèbre successeur Doom.

Vous devrez prendre en compte un élément dimensionnel supplémentaire : la hauteur. Effectivement, dans Wolf3D, l'intégralité du jeu partage le même plancher des vaches et le même plafond. Dans Doom, il est possible de construire des escaliers, des tours, des trous…

Votre jeu devra comporter des murs pouvant former des angles autrement qu'à 90°.

Vous devez intégrer des textures, autant pour les murs que pour le sol et le ciel/plafond.

Vous devez intégrer une physique minimale : il est possible de tomber et d'être projeté, on ne rentre pas dans les murs, on glisse contre eux ou alors on est arrêté. Le personnage dispose d'une certaine inertie.

Vous devez intégrer un système de lumière : plus les objets sont loin, plus ils sont sombres, plus ils sont distants de points lumineux (définie dans la carte), plus ils sont sombres.

Vous devez intégrer, en dernier lieu, des boutons, des portes, des objets à ramasser ou immobile et bloquant, des monstres, des armes de distance, une arme de corps à corps ainsi que des ascenseurs.

Votre programme devra comporter un menu. Il doit être possible de passer en plein-écran.

4 – Bonus

La liste des bonus ci-dessous n'est pas exhaustive :

Des textures sont animées
Certains objets sont animés

Antialiasing

Les cartes présentent des plans inclinés

On peut lever et baisser la tête
On peut bouger tourner l'œil à la souris
Le personnage peut sauter

Vue à la troisième personne
Mode sniper (Zoom)
Vision à 360°

Éléments de jeux issu d'une bibliothèque dynamique

Mini-carte
Boussole

Édition des touches via le jeu lui-même
Cinématique d'entrée
Écran de game over
Écran de transition entre les différents menus ou étape du jeu

Jeu en réseau
Jeu en écran partagé

« Campagne » (Au moins 5 niveaux)

Génération procédurale des niveaux (Procédurale, pas aléatoire)

Différents modes de jeu (Classique – Course – Capture de drapeau (Si multijoueur))

Effets sonores
Musique adaptés aux situations/niveau

Établissement d'un format de carte partagée par plusieurs équipes et/ou d'un format de modules d'extensions (.so) partagé par plusieurs équipes dont il existe une documentation complète. (Documentation et logins de ceux qui partagent ce(s) format(s) exigés)

5 – Interface de correction automatique

Il n'y a pas d'interface de correction automatique pour ce projet.