LAST GUARDIAN

Projet fait par :

Bastien Luck
École d'ingénieurs 2ème année
informatique
luck.e1604990@etud.univ-ubs.fr

Gameplay actuel :

Le jeu est actuellement jouable et nécessite au moins l'utilisation de trois armes différentes pour vaincre le boss final (surement possible sans le poison mais beaucoup trop difficile pour que je le réussisse sans), petit bémol pour le projectile lourd qui étant prévu pour transpercer alors que cette mécanique n'est pas faite, il est donc presque entièrement inutile.

Le projectile basique sans upgrade suffit pour finir le premier niveau et le second niveau s'il est entièrement amélioré.

Les ennemis semblent équilibré, le mushroom étant peut-être inférieur (mécanique de déplacement d'hitbox lors du saut absente étant peut-être une partie de la raison), faire qu'à sa mort il créer un nuage qui bloque les balles (et ralenti les ennemis pour ne pas être trop puissant) pourrait le rendre plus intéressant.

Les mécaniques des projectiles semblent suffisamment compréhensible (hors explosion qui peut paraitre capricieuse lorsque l'on ne sais pas comment sont calculés les hitbox et que l'on n'appui pas directement sur pause à l'impact pour voir qui n'était pas dans le range à la détonation ^^).

Ce que l'on devait faire :

- La documentation sur les headers et sources sont présente ainsi que un dossier avec la documentation en format html (doxygen).
- READ ME et exécutable fait, seul les instructions de compilation ne sont pas faites.
- Le projet possède plus de trois classes et utilise plus d'un héritage maison. Il totalise en tout 28 header pour 29 sources (Main compris) pour un total de 4612 lignes de code (1064 pour les header, 3548 pour les sources), commentaires, lignes d'espacement vide et accolade compris dans le calcul.
- La surdéfinition d'opérateur est utilisé pour la perte de point de vie de la classe Base et son utilisation se trouve ligne 158 du fichier ennemis.cpp.
- La tentative d'utilisation d'un try catch (inutile) se trouve ligne 604-615 du fichier frame.cpp (de ce que j'ai compris, hors cas où on utilise un fichier où l'on n'a pas la main dessus, l'utilisation d'un try/catch ne sert à rien d'autre que faire un if/else qui doit quand même marcher car sinon le programme bug quand même…).

- La librairie STL QVector est utilisé presque partout.
- Le jeu offre la possibilité de jouer avec plusieurs armes contre un spawn aléatoire d'ennemis mais manque de plusieurs features qui étaient prévues et est donc loin d'être fini (le jeu étant gagnable sans être trop facile non plus).
- La vidéo est simple et un peu trop longue (bug de son à l'enregistrement poussant à supprimer celui-ci lors du montage et logiciel utilisé peu pratique malgré tout). Il aurait été pratique de connaitre quel logiciel de capture/montage vidéo utiliser pour éviter les pubs inutile de ces logiciels.

Ce qui était prévu mais n'a pas été fait :

Les fonctionnalités implémenté correspondent en grandes parties à ce que le calendrier prévoyait, au détail près des actions spéciales (les sprites sont prêt (ou presque) mais leurs implémentations n'est pas faite (même si elle n'a pas de créneaux dans le calendrier ^^')).

De plus, la partie storyboard n'a pas été faite, il n'y a donc pas de cinématique lors de l'appui sur le bouton « jouer » ni d'explication de l'histoire in game.

Les informations prévues dans la barre inférieure de l'écran ne sont pas implémenté (le texte présent dans l'écran de combat n'est là que pour dire que le déblocage d'arme n'est pas implémenté et ne devrait pas être un texte pour expliquer comment changer d'arme) : barre de vie de la base, armes possédé + surbrillance de l'arme actuellement équipée, compétences spéciales, cooldown de l'arme, upgrade en cours.

Les actions spéciales des ennemis ne se résume qu'à celle du boss alors que les mobs devaient en avoir (le mob à bouclier n'a pas été fait par exemple).

Le paramètre transperce des projectiles n'est pas prit en compte (problème avec le fait que le projectile touchait plusieurs fois le même ennemi car les deux sprites restent en contact plus d'une frame).

Ce qu'il resterait à faire/améliorer :

Pour le guardian : le décongeler ainsi que lui donner une autre option offensive que son tir principal.

Pour les ennemis : rendre générique la méthode de spawn du boss summoner pour qu'elle soit réutilisable ainsi qu'implémenter plusieurs sous-type d'ennemi (bouclier, mouvement vertical, healer, buffeur, etc).

Pour les barres de vies en haut de l'écran : mettre le nom du boss associé, améliorer le visuel de la barre, afficher un élément graphique indiquant le nombre de barre restante. Améliorer le contraste en la barre de vie actuelle et les barres de vies supplémentaire.

Pour le son : utiliser les sound effect qui sont prêt.

Pour les inputs : ajouter la possibilité de changer d'arme de façon arme suivante/arme précédente.

Pour le menu option : faire un menu plus grand et l'organiser en sous fenêtre, ajouter un slider pour la sélection du volume sonore du jeu (en plus du mute), donner la possibilité de rebinder ses touches, choisir la résolution de l'écran. Trouver un moyen de lier un texte à une image pour faciliter l'utilisation des checkbox.

Pour les crédits : retrouver les liens http perdus, inviter le joueur à appuyer sur une touche pour revenir au menu principal.

Pour le gameplay : ne plus sectionner le jeu par niveau mais faire un seul long niveau avec une meilleur gestion du spawn rate et que le menu upgrade soit accessible par un bouton pour gérer de façon dynamique les améliorations des armes. Faire une sortie du menu d'upgrade avec une courte pause pour permettre au joueur de se remettre en situation (faire comme le 3-2-1-0 des vieux films).

Pour le menu d'upgrade : faire un vrai arbre d'upgrade et non donner la possibilité d'améliorer que les dégats/la cadence de tir et le nombre de tir, ajouter un descriptif de l'upgrade en bas de l'écran lorsque l'on passe sur un upgrade (temps nécessaire à sa complétion, effets). Implémenter la manière de débloquer une arme (upgrade dans un arbre d'une arme inférieure ?). Il faudrait peut-être revoir les upgrades de cadence de tir à la hausse car hors projectile explosif où l'amélioration est visible, elle est invisible pour les autres.

Pour le CPU : modifier la méthode de simulation de framerate avec lecture d'input car cela fait tourner un cœur à 100% sans arrêt (donc par exemple, sur un quadcore on tourne à 25% permanent mais un des processeur prend toute la charge sur lui même).

Pour la frame : mieux séparer ses actions en méthodes.

Pour la fenêtre : ne pas forcer une taille fixe, autoriser le fenêtré et le fullscreen, en fenêtré ne pas supprimer le contour haut de la fenêtre.

Pour la base : trouver un sprite à afficher pour rendre plus logique le fait que les ennemis attaquent une barrière et non le vent.

Pour le niveau : avoir son constructeur qui prend en compte le remplissage de sont QVector spawn_order.

Pour les projectiles : rendre leurs apparitions plus fluide et non le sprite entier directement pour les gros projectiles, faire des sous classes de Projectiles pour les types des projectiles pour ne pas gérer des attributs true/false/==0.

Pour les armes : faire des sous classes d'armes pour le type de tir à la place de tout mettre dans la classe Armes (comme pour Projectiles). Faire d'autres type de tir : munition double côte à côte, rafale trois tir, surchauffe, laser (un laser est un projectile qui fait la taille de l'écran, ne dure qu'une frame car sort de l'écran ensuite et ne fait peut être pas toujours mal pour éviter le burst trop important (ce n'est donc pas un effet spécial d'un projectile ^^)).

Pour la victoire et la défaite : ne pas s'afficher instantanément et améliorer leurs visuels.

Pour l'équilibrage : le premier niveau sans changer d'arme (qui est censé être normal) est beaucoup plus compliqué que prévu, ralentir les slimes pourrait être la solution (ou ajouter un ennemi faible

moins rapide pour éviter que le slime soit l'ennemi le moins fort). Voir pour mettre un réglage de la difficulté au lancement du jeu (arbitraire ou avec choix des modifications par le joueur). Faire un new game + si cela rend le début du jeu impossible avec l'arme de base.

Ce qu'il faudrait voir pour l'année prochaine :

- Faire un partenariat avec une école de graphisme pour que les élèves aient accès à un panel graphique pratique (imaginons qu'un devoir de l'année soit de faire une animation plus ou moins complète d'un personnage, faire un décor, etc. Demander l'accès à l'école cible de ces images et les sauvegarder pour les années suivantes pour que petit à petit les élèves aient accès à une grande variété d'images).
- Cibler des fonctionnalités qui doivent être présente dans le programme (pousse les élèves à en faire plus et égalise plus facilement le niveau final des compétences apprises par les élèves) comme gestion fichier ou autres.
- Garder le passage d'un moment de vacance dans la timeline du projet malgré le décalage du moment du projet (si cela a lieu) car certes peu de groupe utilise ce temps mais cela reste très utile pour d'autres (voir même pour rajouter des semaines de délais en rendu (avantage à ne pas finir le cours à la fin du s3) pour que les retardataires ne soient pas submergé).
- Peut-être faire une annexe/td/tp pour la 3D car beaucoup étaient intéressé mais peu l'on finalement choisi (problème téléchargement, peur de la complexité, temps disponible pour apprendre un nouveau concept, etc).
- Vérifier que les élèves ont téléchargé **correctement** QtCreator avant le premier jour de TP ☺.