

CÉGEP STE-FOY – HIVER 2022
PROGRAMMATION DE JEUX VIDÉO 1– 420-W42-SF

Travail Pratique 2

28 avril 2022

Préparé par
Pierre Poulin

1 Résumé

Pour ce travail pratique, nous allons concevoir un petit jeu de type « space shooter » inspiré de la fin du jeu [The Guardian Legend](#) sorti en 1988 sur la NES.

2 Conditions de réalisation

Valeur de la note finale	Type	Durée	Nombre de remises
30 %	En équipe de 2	3 semaines	1

3 Objectifs du travail


Vous devez réaliser un petit jeu vidéo dans lequel un joueur contrôle un vaisseau et affronte des ennemis ayant des comportements différents. Votre jeu doit inclure des animations, plusieurs scènes qui se partagent à l'occasion des informations, de la sérialisation et un système de réutilisation de mémoire. Votre jeu doit aussi utiliser le patron de conception Publisher-Subscriber et supporter le clavier et le joystick comme méthodes d'entrée.

4 Spécifications

4.1 Récupération du projet de départ

- 1 Récupérez le projet fourni au départ. Il contient du code permettant de créer une fenêtre graphique ainsi que du code précédemment fourni pendant la session lors d'exercices.

4.2 Le menu principal

- 1 Votre menu principal doit permettre de démarrer le jeu en appuyant sur n'importe quelle touche du clavier ou sur n'importe quel bouton du joystick. Il est à noter que si le joueur appuie sur  le jeu doit se terminer.
- 2 Le menu doit avoir une musique. La musique peut se poursuivre (ou non – à votre discrétion) lors du démarrage de la partie.
- 3 Le choix du visuel du menu est laissé à votre discrétion. Assurez-vous seulement que les consignes données au joueur sont claires et valides.

4.3 La partie







Votre jeu doit minimalement contenir les éléments suivants :

- 1 L'arrière-plan du jeu
L'arrière-plan du jeu est une texture qui défile verticalement.
- 2 Musique d'ambiance
Il doit y avoir une musique pendant la partie

3 HUD

Une zone est réservée au bas de l'écran pour indiquer le pointage, le nombre de points de vie ainsi que le temps restant au joueur dans l'état « bonus arme » (4 projectiles au lieu de 2).

4 Le personnage principal

- a) Le personnage principal doit pouvoir se déplacer à sa guise dans l'écran au complet sauf une petite partie au bas de l'écran où sont affichées les informations du jeu.
- b) Le personnage doit être animé.
- c) Le personnage doit pouvoir être contrôlé à l'aide des     et tirer des projectiles à l'aide . Si un joystick est présent, le personnage doit être contrôlé par le « thumbstick » de gauche et lancer des projectiles à l'aide du bouton 0 ().
- d) Lorsqu'il reçoit un projectile le héros est invincible pendant un moment (la durée est à votre discrétion). Une rétroaction visuelle est alors présente.

5 L'ennemi de type 1

- a) L'ennemi de type 1 défile de haut en bas.
- b) Il doit être animé.
- c) Il tire des projectiles lorsque ses « canons » sont ouverts.
- d) Il peut être détruit par 5 tirs du héros ou une collision avec ce dernier.
- e) S'il n'est pas détruit lors d'un passage vertical complet dans l'écran, il doit réapparaître progressivement en haut de l'écran.
- f) Lorsqu'il est détruit, il doit faire apparaître **occasionnellement** un bonus à l'endroit de sa destruction. Le type de bonus (voir plus loin) est aléatoire. Le rythme d'apparition est laissé à votre discrétion.
- g) Un son doit être joué lorsqu'un ennemi de type 1 est détruit.
- h) L'ennemi de type 1 est celui présent dans la première phase du jeu. Vous devez vous assurer qu'il y en a toujours au moins un certain nombre (ex. 3) sauf évidemment si le nombre maximal est atteint (ex. 30).
- i) Les ennemis de type 1 doivent être préalloués au début du jeu et recyclés tout au long de la partie. Vous pouvez néanmoins utiliser une stratégie similaire à celle des solutionnaires fournis et allouer exceptionnellement un ennemi pendant le jeu si tous les ennemis existants sont actifs et qu'aucun ne peut être recyclé au moment souhaité.

6 L'ennemi de type 2






- a) L'ennemi de type 2 apparaît progressivement en haut de l'écran.
- b) Il doit être animé.
- c) Il tire des projectiles lorsque ses « bras » sont ouverts.
- d) Il peut être détruit par plusieurs tirs du héros. Une barre de progression indique la vie restante au-dessus de l'ennemi. Le nombre de tirs requis est laissé à votre discrétion.
- e) Un contact avec le joueur ne le détruit pas mais détruit le joueur.
- f) Il tente de s'aligner avec le joueur en se déplaçant de gauche à droite.
- g) Il apparaît après que tous les ennemis de type 1 aient été détruits.

7 Les projectiles

- a) Le joueur et les ennemis doivent pouvoir lancer des projectiles.
- b) Les tirs « amis » n'endommagent pas (ex. les ennemis ne peuvent pas se détruire entre eux).
- c) Les projectiles doivent être préalloués au début du jeu et recyclés tout au long de la partie.
- d) Un son différent pour le joueur et l'ennemi doit jouer lorsqu'un projectile est lancé.

- e) Inspirez-vous du projet démo fourni pour la manière de lancer « un » projectile, notamment pour le personnage principal qui en lance toujours 2 (ou 4 s'il dispose du bonus).
- 8 Les bonus
- a) Un bonus doit apparaître aléatoirement lorsqu'un ennemi est détruit. Le rythme d'apparition des bonus est laissé à votre discrétion.
 - b) Il apparaît à l'endroit où a été détruit l'ennemi.
 - c) Il y a deux types de bonus : vie et arme.
 - a. Un bonus de type vie ajoute des points de vie au personnage principal tandis qu'un bonus de type arme ajoute un nombre de secondes en état de « bonus arme » au joueur.
 - b. Lorsqu'un joueur est en état de « bonus arme », il tire 4 projectiles au lieu de 2.
 - d) Lorsqu'un joueur est en état de « bonus arme » et qu'il reçoit un projectile ennemi, il ne perd pas de points de vie mais perd toutes ses secondes en état « bonus arme ».
 - e) La récupération d'un bonus ajoute des points au personnage principal.
 - f) Un son est joué lorsque le joueur récupère un bonus. Le son est différent en fonction du type de bonus.
- 9 Le patron Publisher-Subscriber
- a) La partie doit intégrer le patron Publisher-Subscriber.
 - b) La partie doit « écouter » la mort d'un ennemi ainsi que la récupération d'un bonus.
 - c) La gestion du pointage et du son joué se fait dans le **notify** de la partie.

4.4 La scène de *leaderboard*

- 1 Une fois la partie terminée vous devez faire apparaître une fenêtre de *leaderboard*.
- 2 Cette scène doit lire et sauvegarder les informations des 5 meilleures parties jouées. Les informations sont affichées en ordre décroissant à l'écran (du meilleur au moins bon).
- 3 Les informations requises sont :
 - a) Le nom du joueur. Comme dans le temps des arcades classiques, nous allons limiter le nombre de lettres permises à 3.
Note : portez attention au caractère de fin de chaîne (`\0`)...
 - b) Le pointage du joueur.
- 4 Lorsque le *leaderboard* est chargé, le contenu du fichier est amené en mémoire. Vous devez alors récupérer le pointage réalisé par le joueur lors de sa partie. S'il fait partie des 5 meilleurs pointages vous devez permettre au joueur d'entrer son « nom ».
 - a) 3 lettres **obligatoires**
 - b) Lors de la saisie le nom est inscrit avec une autre couleur pour mettre l'emphasis sur le nom entré.
 - c) Le joueur peut appuyer sur  pour effacer une lettre ou sur  pour confirmer le nom lorsque les 3 lettres requises auront été entrées.
 - d) Vous n'êtes pas obligés de supporter le joystick pour cette opération. Si vous le faites je serai impressionné.
 - e) Vous pouvez quitter la scène et retourner au menu principal en appuyant sur  ou  après que le nom a été saisi. Assurez-vous d'ajuster vos instructions au joueur en conséquence.
Note : vous **ne** pouvez **pas** retourner au menu principal tant que les 3 lettres du nom n'ont pas été saisies **ET** que le joueur n'a pas confirmé en appuyant sur .

Vous pouvez référer à l'**exécutable fourni** pour avoir une idée du résultat à produire.

Vous pouvez utiliser sans limite tout le code que vous pourriez trouver dans les démonstrations et corrigés associés au cours. Tout autre bloc de code trouvé sur le Web peut également être utilisé, mais sa provenance doit être indiquée en commentaire.

5 Contraintes et recommandations de développement

Vous devez respecter les standards usuels du cours :

- 1 Dans les méthodes, toujours passer les objets par référence, sauf si vous avez vraiment besoin d'une copie.
- 2 Utilisation systématique de **const**, à la fin des méthodes qui ne modifie pas l'objet courant (dont les **get** et le **draw**) et **const** sur les paramètres en entrée que l'on ne veut pas modifier (donc sur la grande majorité des set).
- 3 L'utilisation d'un **ContentManager** est obligatoire. Ne chargez les textures qu'une seule fois. Idem pour les sons.
- 4 Au niveau des scènes maintenez la séparation du code dans les sections classiques du développement de jeu : initialisation / entrées / logique / affichage.
- 5 Favorisez les **constantes** plutôt que les valeurs codées en dur. Vous serez pénalisés pour chaque valeur inscrite directement dans le code.
- 6 Pas besoin de tout commenter. Favorisez un bon nommage (classes, méthodes, attributs et variables) au lieu d'un commentaire. Si vous utilisez un algorithme
- 7 Portez attention à l'utilisation d'allocation dynamique. Si vous utilisez l'opérateur **new**, assurez-vous d'utiliser adéquatement l'opérateur **delete**. L'utilisation de VLD peut s'avérer essentielle.
- 8 N'oubliez pas que vous devez utiliser un concept de *pool* d'objets réutilisables pour les ennemis, les bonus et les projectiles.
- 9 Assurez-vous de **compiler sans avertissement** tant en **Debug** qu'en **Release**. Vous serez pénalisés s'il reste des avertissements dans votre remise finale.

6 Contexte de réalisation et démarche de développement

Ce travail pratique doit être réalisé en **équipe de deux personnes**.

Vous devez vous créer un dépôt GitLab et me le partager (@ppoulincsf). Comme pour tous les travaux en équipe la contribution des deux coéquipiers sera considérée. Faites des *commits (push)* réguliers associés d'un bon commentaire.

Biens livrables :

- 1 Projet avec code source de l'application.

Conditions de remise :

- 1 L'application doit compiler sans avertissement et être fonctionnelle.
- 2 Remise via LEA dans un **.zip** duquel vous aurez retiré les dossiers **.vs**, **Debug**, **Release** et **extern** (si applicable)
- 3 Date de remise : **17 mai 2022 à 8h00**

7 Modalités d'évaluation

Tous les biens livrables devront être remis à temps et selon les modalités spécifiées.

Dans l'éventualité où vous récupérez du code existant ailleurs (internet, MSDN, etc), vous devez clairement indiquer la source ainsi que la section de code en question. Tout travail plagié ou tout code récupéré d'une source externe et non mentionnée peut entraîner la note zéro (0) pour l'ensemble du travail.

La grille d'évaluation utilisée pour ce travail est jointe à cet énoncé.