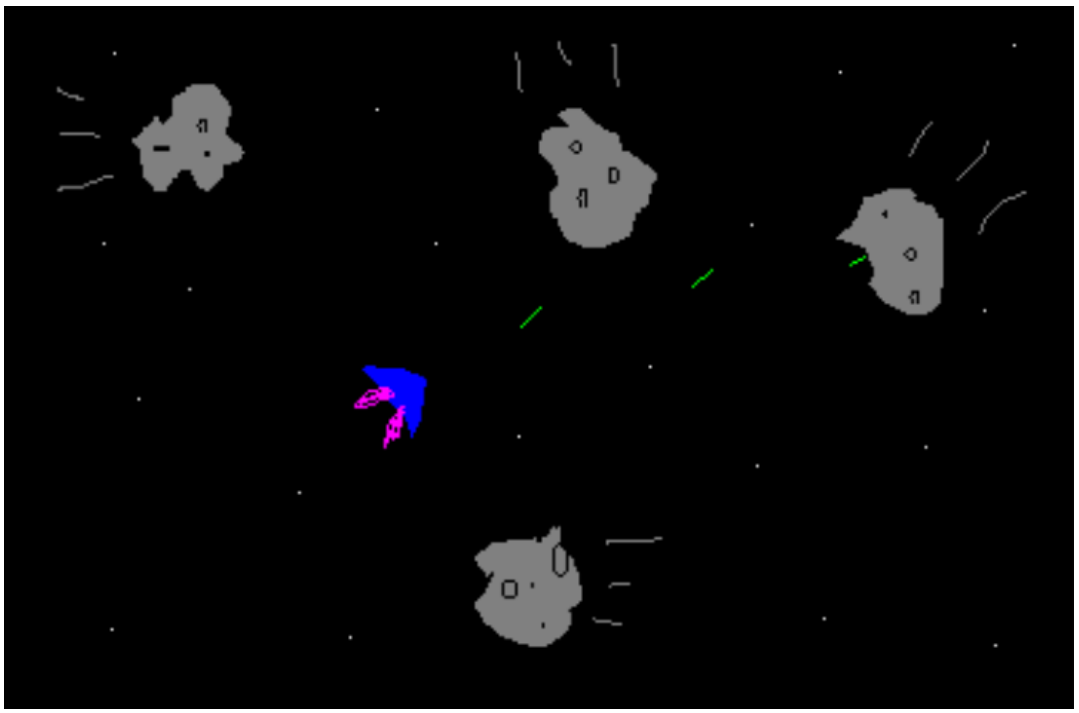
CAHIER DES CHARGES - *STARESIOD*



vue conceptuelle du jeu (vue de haut)

1. Présentation du projet

→ **Contexte :** Dans le cadre de l'UE de LIFAPCDA, il est nécessaire de réaliser un projet de conception et de développement d'application. Ce projet représentera 50% de la note finale de l'UE, et différents aspects de ce projet seront évalués à différentes occasions.

→ **Client et historique :** le corps professoral de LIFAPCDA, qui a l'habitude de noter des projets de programmation d'étudiants. La moindre tentative de triche ou d'évitement d'obstacle sera donc systématiquement dénichée et sanctionnée : pas le droit à l'erreur.

→ **Rémunération :** un check et un pouce en l'air.

2. Description

→ **Résultats visés :** Coder le mini-jeu *Staresiod*.

Le joueur contrôle un vaisseau spatial, et doit rester en vie le plus longtemps possible en évitant des obstacles (astéroïdes) qui se déplacent dans l'écran en venant des bords (on pourra ajouter des warnings avant l'apparition des astéroïdes, par exemple un point d'exclamation sur le bord de l'écran). Le vaisseau peut tirer des projectiles, qui peuvent détruire les astéroïdes en plus ou moins de coups selon la taille de l'astéroïde.

→ **Mécaniques de bases :**

Contrôle du vaisseau	Génération d'astéroïde
Avancer Reculer Tourner à droite Tourner à gauche Tirer un projectile	Par défaut, trajet rectiligne Génération d'un couple vitesse*taille avec vitesse + taille = constante.

Staresiod a un système de score, qui augmente proportionnellement au temps passé en vie. On conservera le high score dans l'application, entre deux exécutions séparées.

Fonctionnalités supplémentaires
Système de power-ups Mode multijoueur : - Co-op / Versus - J2 / IA

3. Contraintes

→ **Date limite de rendu :** 01/05/2023
→ **Budget :** 0

4. Déroulement - liste des tâches

(R : responsable de tâche, P: participe à la réalisation de la tâche.
Ugo : U, Joey : J) Ex : Ugo responsable, Joey Participe → URJP.

1. Elaboration du cahier des charges - JRUP

→ Présentation globale
→ Diagramme des classes
→ Diagramme de Gantt

→ Réalisée quand le cahier des charges complet aura été rendu pour le lundi 06/03/2023;

2. Préparation de la documentation - JRUP

→ Tout au long du projet, rédaction des commentaires Doxygen appropriés dans les *.h*.

3. Mise en place des classes primaires - Vec2 -URJP

→ Classe Vec2: accesseurs, mutateurs, opérateurs et fonction de test

→ Réalisée quand la fonction de test utilisera toutes les fonctions de la classe sans erreur

4. Mise en place des classes restantes - Vaisseau, Terrain, Jeu URJR
→ Classe Vaisseau: accesseur, mutateur, fonction de test
→ Classe Terrain: définition de la taille du terrain
→ Classe Jeu: procédures d'initialisation, de mise à jour, de réinitialisation et de stockage du score

→ Réalisée quand les fonctions de test ne renverront pas d'erreur et que les fonctions renverront les résultats voulus

5. Construction des main et du makefile UR
→ main: un mainTest pour tester toutes les fonctions, un main pour mettre en place le jeu en texte et un main pour mettre en place le jeu en SDL2
→ makefile: crée tous les fichiers objets et les exécutables

→ Réalisée quand le makefile compilera les exécutables sans erreur

6. Affichage texte URJP
→ Fonctions spécifiques à l'affichage text et un main dédié au jeu en text

7. Debuggage et résolution de problèmes avec affichage text URJP
→ Réalisée lorsque le jeu entier est fonctionnel en affichage text

8. Mise en place de l'interface graphique avec SDL2 - JRUP
→ Fonctions spécifiques à SDL2 et un main dédié au jeu en version SDL2

9. Debuggage et résolution de problème avec SDL2 - JRUP
→ Réalisée lorsque le jeu entier est fonctionnel avec un affichage SDL2

10. Implémentation du système de power ups - UR
→ Ajout des classes PowerUp et PowerUps

→ Réalisée lorsque toutes les x secondes, un power-up apparaît à une position aléatoire sur le jeu, et tous les power-ups sont fonctionnels

11. Ajout d'un mode multijoueur en coopération - URJP
→ Ajout d'un choix entre solo et multijoueur, avec un joueur qui utilise la partie droite du clavier et l'autre la partie gauche

→ Réalisée lorsque les 2 utilisateurs peuvent jouer sur la carte

12. Ajout du mode versus en multijoueur - URJP
→ Ajout d'un choix dans multijoueur entre versus et coopération

→ Réalisée lorsque le code compile avec les ajouts

13. Debuggage du mode multijoueur - URJR

→ Réalisée lorsque la totalité des fonctions multijoueurs sont fonctionnelles

14. Mise en place d'une "IA" qui contrôle le deuxième joueur en versus - JRUP

→ Ajout d'un mode "IA" dans le menu à côté de solo et multijoueur : dans ce cas il y a un score

→ l'IA essaie de se débrouiller

15. Debuggage et résolution de problème avec le système d'IA - URJR

→ Réalisée lorsque le joueur peut jouer avec une IA au niveau adapté

16. Dessin et implémentation des sprites - JR

→ Pixel art des textures du jeu : vaisseau, astéroïdes, projectile, fond.

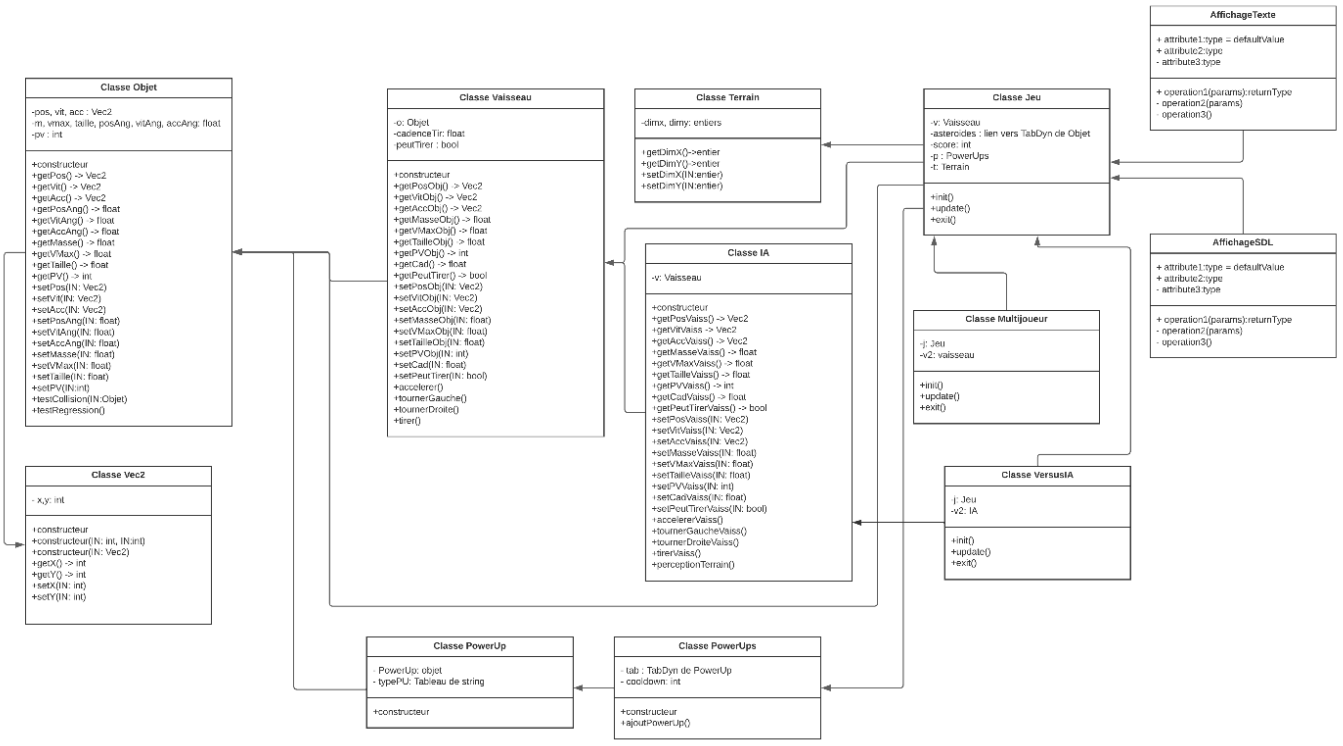
→ Réalisée lorsque le jeu ne fait plus mal aux yeux.

17. Compression de l'archive et envoi sur tomuss - cette étape a l'air compliquée, personne ne veut en endosser la responsabilité

→ Réalisée lorsque l'archive est comprimée et envoyée sur tomuss

5. Diagramme des Classes

Diagramme des Classes



6. Diagramme de Gantt

Diagramme de Gantt