

**Projet :** Création d’un jeu vidéo 2D Snake

**Version du document :** 3

**Propriétaire du document :** Equipe 22

**Date de Création :** 12/11/2023

**Document de Spécification du Moteur Physique, Graphique et Noyau du Jeu Snake**

**Introduction**

Ce projet consiste en le développement du jeu vidéo ‘Snake’, mettant en place les moteurs physique et graphique. Ces moteurs permettent de simuler le déplacement des objets du jeu et d'afficher les éléments visuels du jeu, en utilisant des principes physiques tels que la position, la vélocité, et l'accélération.

**1.1.** **Objectif :**

Ce document a pour but de définir les spécifications détaillées du moteur physique et graphique du jeu. Il servira de référence pour le développement du moteur.

**1.2.** **Intended Audience :**

Ce document est destiné aux développeurs, concepteurs de jeux, testeurs et toute autre personne impliquée dans le développement du jeu vidéo.

**1.3.**  **Intended Use :**

Les informations contenues dans ce document seront utilisées pour guider le développement du moteur physique et graphique du jeu.

**1.4.**  **Product Scope :**

Le produit couvert par ce document est le moteur physique et graphique du jeu, qui est conçu pour gérer les objets et l'affichage graphique du jeu.

**2.**     **Moteur physique :**

Un moteur physique dans le développement du jeu Snake est un composant logiciel qui simule les lois de la physique dans le contexte du jeu. Pour gérer le déplacement du serpent, les collisions et les réactions, en tenant compte de principes physiques tels que la vélocité, l'accélération et les forces.

**2.1.**  **Espace du jeu :** Dans le jeu Snake, l'espace du jeu est représenté par une grille en deux dimensions, où le serpent évolue en suivant les règles relatives à sa croissance, son mouvement et les collisions. Cet espace de jeu est essentiel pour déterminer comment la partie se déroule. En d'autres termes, c'est la zone de jeu où le serpent interagit avec les règles du jeu pour créer une expérience de jeu captivante.

**2.2.**  **Objets :** Dans ce jeu, nous aurons trois types d'objets, un objet représentant le serpent qui est le personnage principal, un autre représentant un fruit qui sert à faire grandir le serpent à chaque fois que ce dernier le mange et le dernier représentant les mûres qui sont des obstacles.

**Serpent :**

C’est le personnage contrôlé par le joueur. Il est composé d'une tête et d'un corps, qui est lui-même composé de segments.

 La tête du serpent est utilisée pour se déplacer et collecter des points.

 Les segments du corps du serpent sont les parties du corps du serpent qui suivent la tête en formant une chaîne derrière elle. Ils sont générés en grandissant à chaque fois que le serpent mange une pomme.

La queue du serpent

**Les fruits :**

Ce sont les objets que le serpent doit manger pour grandir**.** Les fruits devraient apparaître de manière aléatoire sur la grille du jeu.

**Dynamique des objets:**

Le moteur doit permettre la simulation du mouvement du serpent. Chaque objet dans un espace bidimensionnel est caractérisé par:

**Position :** L'objet a une position dimensionnelle (x, y) dans l'espace du jeu, représentée par un vecteur.

**Vitesse :** Il possède également une vitesse (x, y), présentée par un vecteur.

**Accélération** : L'accélération (x, y) est utilisée pour modifier la vitesse de l'objet et est stockée dans un vecteur.

**Dimensions :** L'objet a une largeur et une hauteur qui définissent sa taille dans l'espace du jeu.

Le déplacement du serpent s'effectue en suivant un processus simple :

Le serpent se déplace en fonction des commandes du joueur. L'objectif est de déplacer le serpent le serpent en mangeant des fruits tout en évitant les collisions avec les bords de la grille et avec son propre corps.

**Mise à Jour de la Vitesse du serpent :**

Une méthode ajuste la vitesse de l'objet en fonction de l'accélération. Elle ajoute la composante x de l'accélération à la vitesse x de l'objet et la composante y de l'accélération à la vitesse y de l'objet. La vitesse doit augmenter à chaque fois que le serpent mange un fruit.

  Mathématiquement : `velocityX += accelerationX` et `velocity.y += acceleration.y`.

**Mise à Jour de la Position du serpent:**

La méthode met à jour la position de l'objet en fonction de sa vitesse. Elle ajoute la composante x de la vitesse à la position x de l'objet et la composante y de la vitesse à la position y de l'objet.

*Mathématiquement :* position.x += velocity.x et position.y += velocity.y.

Cela signifie que l'accélération influe sur la vitesse, qui à son tour influe la position. En d'autres termes, l'accélération représente le changement de vitesse au fil du temps. Ce processus est itératif, ce qui signifie qu'il se répète à chaque itération du jeu, permettant ainsi au serpent de se déplacer dans l'espace 2D.

**Mise à Jour de la longueur du serpent:**

Lorsque le serpent mange un fruit, sa longueur augmente car un nouveau segment est ajouté à son corps. Les segments du corps du serpent sont mis à jour pour suivre la position de la tête, chaque segment prenant la position du segment précédent.

La partie se termine si la tête du serpent entre en collision avec un mur ou si elle se heurte à son propre corps.

Contrairement au serpent qui se déplace, les fruits sont statiques, c'est-à-dire, la vitesse et l'accélération sont fixées à zéro. Quant aux positions, il faudra générer des coordonnées aléatoires pour positionner chaque fruit à des emplacements différents à chaque apparition, et faire en sorte que ses positions ne soient pas les mêmes avec celles du serpent.

**Interaction avec le serpent :**

Lorsque la tête du serpent atteint la position d'un fruit, cela déclenche un événement de collision qui augmente le score du joueur, fait grandir le serpent (Ajout des segments), et régénère un nouveau fruit à un emplacement aléatoire différent de celui du serpent..

**Les murs :**

Dans le jeu Snake, les murs jouent un rôle essentiel en créant des limites et des obstacles. Les bords de la grille définissent les limites de l'espace de jeu. En plus, les murs au milieu de la grille sont des obstacles internes que le serpent doit éviter ou surmonter.

**2.5.**  **Gestion de Collision :**

Le moteur doit gérer efficacement les collisions entre objets. Lorsque deux entités se superposent ou entrent en contact :

Détection de Collision :Le moteur doit être capable de détecter les collisions en vérifiant les positions des objets dans la grille:

Avec les fruits en comparant sa position avec les positions de tous les segments du corps.

Avec les bords de la grille. Cela se produit lorsque la position x ou y de la tête dépasse les limites de la grille.

Réponse aux Collisions : Le moteur doit fournir des mécanismes pour gérer la réaction aux collisions, y compris la résolution de la collision, la gestion des dégâts.

Lorsque la tête du serpent atteint la position d'un fruit, le moteur doit ajouter un segment au corps du serpent pour simuler sa croissance, régénérer un nouveau fruit à une position aléatoire qui est différente de celle du serpent.

Si la tête du serpent atteint un mur de la grille (Bords de la grille) ou se trouve sur la même case qu'un segment de son propre corps, le jeu se termine.

**Moteur graphique du Jeu  :**

Ce composant du jeu est responsable de l'affichage des éléments graphiques, y compris les personnages.

**3.1. Affichage :** Le moteur se charge de l’affichage des différents objets.

**3.2. Animations :** Le moteur doit prendre en charge des systèmes pour la création d'effets spéciaux.  Elles sont utilisées dans une variété de contextes, tels que le mouvement du serpent, les actions spéciales, les réactions aux événements.

**Noyau  :**

Le noyau est responsable de la gestion des aspects essentiels du jeu, la physique, le graphique et la gestion des entrées du jeu pour:

Fournir une infrastructure solide pour la création de serpents.

Faciliter l'intégration de la logique du jeu, des interactions utilisateur et des graphismes.

**Composants:**

Le noyau du jeu de serpent est composé des éléments suivants :

Moteur physique

Moteur graphique

**InputHandler :** Un gestionnaire d'entrées pour gérer les actions de l'utilisateur.

**GameObject :** Une liste d'objets du jeu, y compris le serpent et la nourriture.

**Fonctionnalités:**

Le noyau permet :

La création et la gestion des objets du jeu, y compris le serpent et la nourriture.

La mise à jour de l'état du jeu.

D'assurer la gestion des entrées utilisateur, incluant la prise en charge des commandes pour déplacer le serpent.

De mettre en pause et de reprendre le jeu en modifiant l'échelle temporelle.