Explication détaillé sur le jeu : Maze

Introduction

Ce projet est un jeu interactif basé sur SFML (« Simple and Fast Multimedia Library »), permettant de visualiser différents algorithmes de parcours et de recherche de chemin dans une grille. Le jeu combine des concepts de programmation en C++ et une interface graphique fluide et réactive pour offrir une expérience à la fois ludique et éducative. Il met en œuvre les algorithmes BFS (Breadth-First Search), DFS (Depth-First Search), Dijkstra et A* pour explorer et trouver des chemins optimaux entre deux points donnés.

Fonctionnalités Principales

Interface Graphique

- **Grille Dynamique** : La grille est générée dynamiquement en fonction de la taille de la fenêtre.
- **Interactivité** : Les cellules peuvent être cliquées pour définir des points de départ, d'arrivée, ou des murs.
- **Animations** : Les algorithmes de parcours sont animés pour permettre une meilleure compréhension de leur fonctionnement.
- **Checklists et Boutons** : L'interface contient une liste de contrôle et des boutons pour choisir l'algorithme à utiliser et exécuter des actions.

Algorithmes Implémentés

1. **BFS (Breadth-First Search)** :
    - Algorithme de recherche.
    - Explore les nœuds couche par couche pour trouver le chemin le plus court en termes de nombre de cellules.
2. **DFS (Depth-First Search)** :
    - Algorithme de recherche.
    - Explore les nœuds en profondeur avant de revenir en arrière, ce qui peut ne pas garantir un chemin optimal.
3. **Dijkstra**:
    - Algorithme de recherche.
    - Trouve le chemin le plus court en prenant en compte le coût de chaque cellule.
4. **A*** :

- Algorithme de recherche heuristique.
- Combine Dijkstra avec une fonction heuristique de Manhattan pour optimiser les performances.

Séparation des Couches

Le projet est structuré en deux couches principales :

1. **Couche Traitement** : Contient la logique des algorithmes et la gestion des données
2. **Couche Graphique** : Gère l'affichage, les animations et les interactions avec l'utilisateur.

Cette architecture modulaire facilite la maintenance et l'extensibilité du projet.

Détails Techniques

Grille et Cellules

Chaque cellule est modélisée par une classe Cell contenant des informations comme :

- Position (index et coordonnées pixel)
- Type (vide, mur, départ, arrivée)
- Couleur (définie dynamiquement selon l'état)
- Distance et heuristique (pour les algorithmes)
- Parent (pour reconstruire le chemin optimal)

Algorithmes de Parcours

Chaque algorithme suit ces étapes générales :

1. Initialisation des distances, heuristiques, et statuts des cellules.
2. Utilisation d'une structure de données adaptée :
    - File pour BFS
    - Pile pour DFS
    - File de priorité pour Dijkstra et A*
3. Exploration des voisins et mise à jour des distances, heuristiques, et couleurs.
4. Affichage de l'animation à chaque étape pour visualiser le processus.
5. Reconstruction et affichage du chemin final.

Interactivité

- **Définition des Points** : L'utilisateur peut cliquer sur la grille pour définir les points de départ et d'arrivée ainsi que les obstacles.

- **Sélection de l'Algorithme** : L'utilisateur sélectionne un algorithme dans la checklist pour l'exécuter.
- **Visualisation Dynamique** : Chaque algorithme est animé en temps réel.

Performance et optimisation

- Les algorithmes ont été optimisés pour fonctionner efficacement sur des grilles de grande taille.
- Le système de mise à jour graphique évite les recalculs inutiles.
- Une fonction heuristique adaptée Manhattan est utilisée dans A* pour accélérer le processus.

Déploiement Web (via WebAssembly)

Bien que SFML ne soit pas directement compatible avec le web, il est possible de porter le jeu en ligne avec des outils comme :

- **Emscripten** : Permet de compiler du code C++ en WebAssembly.

Améliorations Futures

1. **Ajout d'Algorithmes** :
    - Intégrer d'autres algorithmes.
2. **Personnalisation Avancée** :
    - Permettre à l'utilisateur de modifier les coûts des cellules.
    - Ajouter des types de terrain (eau, montagnes, etc.) avec des coûts différents.
3. **Environnement 3D :**

Utiliser SFML en combinaison avec une bibliothèque 3D pour créer des grilles tridimensionnelles.

Conclusion

Le jeu offre une excellente plateforme pour apprendre et visualiser les algorithmes de recherche de chemin. Avec un déploiement stratégique et des améliorations futures, ce jeu peut devenir un outil à la fois divertissant et pédagogique.