Noé Barranger Conrad Seidel FISE1 C

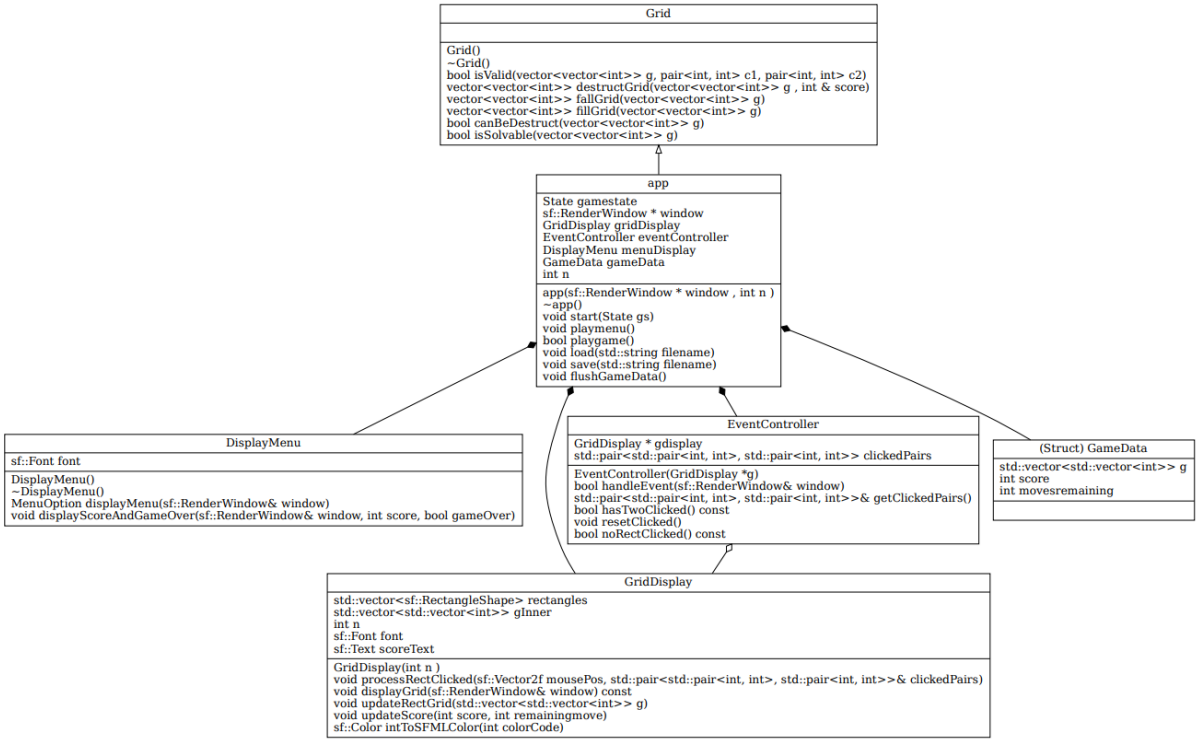
Rapport Mini-projet

Fonctionnalités :

Notre projet de jeu CandyCrush propose plusieurs fonctionnalités réparties dans deux endroits : le jeu en lui-même et le menu. Dans le menu, on a la possibilité de sauvegarder ou charger une partie, d'en créer une nouvelle ou de quitter l'application.

Dans la partie jeu, on peut jouer en appuyant sur le clic gauche de la souris pour sélectionner deux bonbons à interchanger. Si le coup est possibles les deux bonbons s'échangent et le tour se fait sinon il ne se passe rien et il faut à nouveau sélectionner deux bonbons. Sur l'écran de jeu, il y a aussi le nombre de coups restants et le score qui se mettent à jour après chaque tour. On peut appuyer sur la touche échap pour accéder au menu, et lrosque la partie est finie il y a un écran de fin avec le score et on peut appuyer sur échap pour quitter.

Diagramme de classes :



Explication des choix de classes :

On a choisi d'utiliser 5 classes qui sont Grid, app, DisplayMenu, EventController et GridDisplay.

La classe app gère la logique du jeu. C'est la classe centrale de notre projet donc c'est à elle qu'on doit lier les autres classes.

La classe grid sert à gérer le jeu en lui-même. On lie app et grid par un héritage pour pouvoir utiliser plus facilement les méthodes de grid dans la classe app quand on gère la logique du jeu.

La classe app a aussi beoin des évènements commes les clics de souris et les positions des bonbons cliqués qui sont de la classe EventController pour jouer le tour donc on les lie par une composition.

On crée une structure GameData pour clarifier et rassembler certains champs de app et mettre à part les données du jeu à changer à chaque tour donc elles sont liées par une composition.

La classe DisplayMenu est utile pour gérer la partie menu de l'application et comme on doit pouvoir accéder au menu depuis l'application on la relie par une composition à la classe app.

Enfin la classe GridDisplay gère l'affichage de la fenêtre de jeu. On la lie donc à app avec une composition pour pouvoir avoir un lien indirect avec grid qui met à jour la grille de jeu et à la fois utiliser les méthodes de app pour jouer les coups.

On a décidé de séparer les classes d'affichage de la fenêtre de menu et de la fenêtre de jeu car on voulait tous les deux pouvoir manipuler l'affichage et c'e=était plus simple pour développer en parallèle.

Déclaration commentée des classes :

Classe app :

Auteur Conrad

Cette classe sert à gérer le déroulement de la partie. Elle contient la grille actuelle, le score et le nombre de coups restants dans une struct GameData dans un de ses champs. Elle contient aussi une variable gamestate qui indique si on est en jeu ou dans le menu et les variables d'affichage. C'est cette classe qui permet de sauvegarder, créer ou charger une partie ainsi que l'initialisation et les différents tours du jeu.

Classe DisplayMenu :

Auteur Conrad

Cette classe gère tout ce qui est en rapport avec le menu que ce soit l'affichage ou les évènements à l'interieur de la fenêtre menu. Elle se sert aussi d'une enumération d'état pour gérer les différentes options du menu. Enfin elle affiche la fenêtre de fin et le score final.

Classe EventController :

Auteur Conrad

Cette classe gère les évènements créés par l'utilisateur comme les clics souris et les touches claviers notamment la touche échap pour le menu et la sélection des bonbons au clic gauche. Cette classe ne gère les évènements que dans la fenêtre de jeu.

Classe Grid :

Auteur Noé

La classe gère le jeu en lui-même. Elle sert à gérer la mise à jour de la grille au fur et à mesure des tours en faisant la destruction des bonbons, le fait qu'ils tombent, la génération aléatoire des bonbons suivants et l'incrémentation du score.

Classe GridDisplay :

Auteur Noé

La classe permet de gérer l'affichage du jeu. Elle permet de mettre à jour en temps réel la grille, le score, le nombre de coups restants, ainsi que les moments ou on sélectionne les bonbons en blanc avant de les jouer.