Gameboj : Rapport de bonus

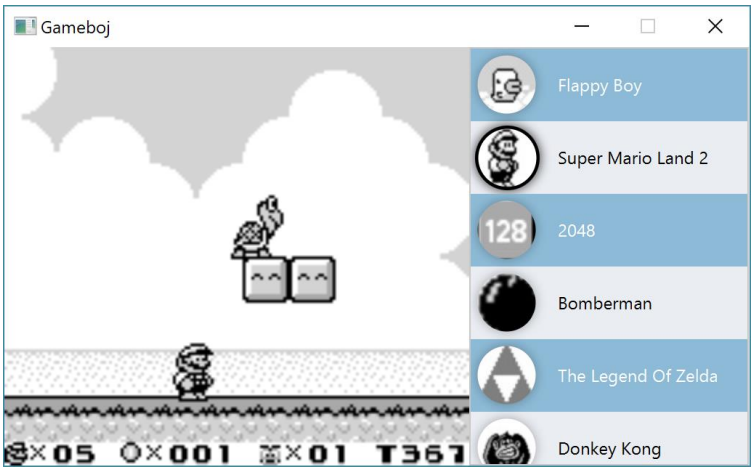
Projet de pratique de la Programmation Orientée Objet

Présentation des améliorations

Notre bonus a pour but de simplifier l'expérience de l'utilisateur. Nous nous sommes concentrés sur deux aspects : une meilleure interface graphique ainsi que la sauvegarde des jeux.

Lorsque l'utilisateur utilise l'émulateur, un élégant menu de sélection lui permet de choisir parmi une liste de nombreux jeux. Les jeux qui peuvent être sauvegardés (MBC1) le sont automatiquement comme s'il s'agissait d'une vraie cartouche.

La sauvegarde s'effectue lors du changement de jeu ou à l'arrêt du programme. Au redémarrage, l'interface s'occupe de charger la sauvegarde correcte pour chaque jeu.



Interface graphique

Dans le paquetage `ch.epfl.gameboj`, nous avons créé une classe utilitaire `Games`, qui contient une liste de différents `GameItem`. Un `GameItem`, défini dans `ch.epfl.gameboj.gui`, représente un jeu Gameboy. Chaque jeu a un identificateur unique, un nom, un fichier ROM, un fichier de sauvegarde ainsi qu'une icône. `GameItem` possède également une représentation graphique (un `Pane`) disponible via la méthode `asPane`.

Tous les `GameItem` sont placés dans le panneau latéral `GameList`, représenté graphiquement par un `ScrollPane`. Le panneau latéral possède une propriété indiquant le jeu sélectionné. Le programme principal `Main` observe cette propriété et change le jeu en fonction de sa valeur.

Pour gérer le clavier, nous avons créé une classe `KeyboardHandler`, qui prend un `ImageView` sur lequel il attache des EventHandler. On peut y attacher un joypad avec la méthode `attachJoypad`. Ceci permet de changer de Gameboy sans détruire l'écran et les `EventHandler`.

L'écran de la Gameboy est placé dans un conteneur `GBScreen`, qui gère l'affichage et le clavier. Lorsqu'on change de jeu, on appelle sa méthode `attachGameboy`, qui change de Gameboy et l'attache au `KeyboardHandler`.

L'interface est décorée en utilisant une combinaison de CSS (défini dans `ch.epfl.gameboj.gui.css`) et de styles Java FX (comme dans `GameItem`).

Sauvegarde des fichiers

La sauvegarde se fait grâce à une nouvelle classe nommée `GBSaver`, du paquetage `ch.epfl.gameboy.component.cartridge`,. Elle possède une méthode `load`, qui prend en argument une cartouche ainsi qu'un fichier optionnel qui sera utilisé pour la sauvegarde de ce jeu. Si le fichier existe, le contenu est chargé sur la cartouche. La représentation du fichier de sauvegarde du jeu est stockée dans l'instance de `GameItem` qui lui correspond. Son autre méthode, `save`, sauve l'état de la cartouche dans le fichier à chaque appel. Ces deux méthodes sont utilisées dans le listener de `GameList` et dans la méthode `stop` de `Main`.

Certains jeux ne peuvent pas être sauvegardés, puisque leur cartouche n'a pas de mémoire vive. Il faut donc déterminer cela en fonction du type de banque de mémoire de la cartouche.

Ce problème est réglé par une nouvelle classe abstraite dans le paquetage `ch.epfl.gameboy.component.cartridge` : `MBC`. Cette classe unifie les différents types de banque de mémoire. Elle devient la super classe de `MBC0` et `MBC1`.

L'API de cartridge a été augmentée par des méthodes qui permettent :

1. D'écrire l'état de la RAM dans un fichier et vice-versa
2. De savoir si un jeu peut être sauvegardé, grâce à sa cartouche ou son fichier ROM

Ces dernières sont utiles dans la classe `GBSaver` ainsi que pour créer le fichier de sauvegarde dans `GameItem`