Programmation en Java — Projet

ESIPE – IR 1 – printemps 2019

Sujet : Réaliser le jeu Hanabi en Java

Le but de ce projet est de réaliser une version PC offline d'un jeu de société : Hanabi. Il s'agit d'un jeu de stratégie où plusieurs joueurs doivent collaborer pour réaliser les plus beaux feux d'artifice possibles.

Le jeu

La règle détaillée du jeu Hanabi est disponible en ligne. Nous vous invitons à la lire avec attention, puisqu'il s'agit du jeu que vous allez devoir programmer.

Le programme à réaliser

Vous allez réaliser votre jeu en 3 phases. Chaque phase devra être terminée et fonctionnelle avant de passer à la phase suivante.

Phase 1 : La base

Dans un premier temps, vous devez réaliser une version très simplifiée du jeu, avec uniquement un mode de jeu permettant à deux joueurs humains de collaborer, et un affichage **en ligne de commande**.
Dans cette version très simplifiée, les joueurs trichent et voient leurs propres cartes. Par conséquent :

- il ne sera pas nécessaire de donner d'indice à son partenaire, et chaque joueur pourra, s'il le souhaite, défausser une de ses cartes sans se préoccuper du nombre de jetons bleus disponibles ;
- en revanche, le joueur devra toujours avoir la possibilité de jouer comme un crétin et de faire une erreur en décidant de jouer une carte qu'il n'aurait pas dû jouer : on devra donc gérer correctement le nombre de jetons rouges disponibles.

Phase 2 : Le jeu complet

Dans un deuxième temps, vous devez réaliser une version complète du jeu, **toujours en ligne de commande**, incluant un écran de démarrage où l'on indique combien il y aura de joueurs. Tous les joueurs seront humains.
Vous devrez notamment trouver un moyen adapté de faire apparaître, en ligne de commande :

- les cartes situées dans la main du partenaire ;
- les cartes déjà posées, et qui forment un commencement de feu d'artifice ;
- la carte située au-dessus de la défausse ;
- le nombre de jetons bleus et rouges qui peuvent encore être utilisés.

Vous devrez aussi faire en sorte que le joueur puisse indiquer quelle action il souhaite exécuter et, le cas échéant, à quel partenaire il souhaite donner un indice et quelle information (couleur ou valeur) il souhaite lui donner. Vous devrez aussi trouver un moyen adapté de signaler à un joueur quelles informations ses partenaires lui ont transmises.
Par ailleurs, il faudra bien sûr vous assurer que, quand le tour d'un joueur commence, la ligne de commande n'affiche plus les informations destinées au joueur précédent : pas question qu'un joueur puisse voir les cartes dont il dispose sous prétexte qu'elles ont été précédemment affichées à destination de ses partenaires !

Phase 3 : Améliorations

Une fois la phase 2 terminée, on pourra, par exemple :

- ajouter la présence de joueurs simulés par l'ordinateur : nous laissons à chaque groupe le soin de mettre au point des stratégies pour les joueurs simulés ; dans ce cas, il pourra être bon de lancer plusieurs parties où tous les joueurs sont simulés par l'ordinateur, pour tester l'efficacité de telle ou telle stratégie ;
- se reposer sur l'utilisation d'un fichier de paramètres ; ces paramètres devront être enregistrés dans un fichier de manière lisible par un humain, et vous devrez décider si vous souhaitez qu'on puisse les modifier uniquement en éditant directement le fichier en question, ou bien si vous prévoyez également un menu "Paramètres" accessible depuis l'écran de démarrage ; ces paramètres pourront inclure le nombre de joueurs autorisés et le nombre de cartes par joueur, le nombre de couleurs disponibles (et les teintes associées), le nombre de valeurs disponibles dans chaque famille (et le nombre de cartes pour chaque valeur) ;
- mettre en place une interface graphique (voir ci-dessous) ; dans ce cas, vous devrez trouver une manière adaptée de laisser à l'utilisateur le choix entre l'interface graphique et l'interface en ligne de commande.

Toute autre amélioration sera considérée négativement tant que les améliorations ci-dessus n'ont pas toutes été mises en place.

L'interface graphique

Il est possible de réaliser entièrement l'affichage du jeu au terminal. Néanmoins, pour des questions d'esthétique et de jouabilité, vous pouvez, durant la phase 3, réaliser une interface graphique simple :

- Dans ce cas, vous devrez utiliser la bibliothèque d'interface graphique `zen` fournie avec ce sujet (fichier `zen5.jar`).
  Pour ajouter un jar à un projet sous Eclipse, il faut :
  - Rajouter un dossier `lib` dans le répertoire du projet et y placer le fichier `.jar`.
  - Dans Eclipse, faire un clic droit sur le fichier `.jar` et choisir `Build Path > Add to Build Path`.
- On vous fournit également un mini-exemple (très incomplet) de code utilisant cette bibliothèque et suivant un modèle de développement classique appelé MVC (Modèle-Vue-Contrôleur) que vous devrez appliquer dans votre code (même si vous faites l'affichage au terminal) :
  - une classe `SimpleGameData` est utilisée pour gérer les données du jeu (le modèle) ainsi que toutes les actions possibles, suivant les règles du jeu ;
  - une classe `SimpleGameView` permet de gérer l'affichage graphique (la vue).
  - le contrôleur `SimpleGameController` implante la boucle de jeu et la gestion des événements utilisateur (clics, touches et déplacements).

Conseils

- Le but de ce projet est de nous montrer que vous savez programmer "objet". Vous serez donc pénalisés si vous n'exploitez pas suffisamment les notions vues en cours.
- Lisez bien les règles du jeu et prenez bien le temps de réfléchier à la meilleure façon de mettre en place tel ou tel mécanisme du jeu avant de commencer à coder. Si vous commencez à coder avant d'avoir suffisamment réfléchi, vous serez, en pratique, obligés de revenir en arrière, ce qui vous demandera beaucoup plus de temps et d'efforts.
- Le sujet est évolutif : respectez bien les phases de réalisation, mais gardez à l'esprit ce que vous devrez faire dans les phases suivantes lorsque vous faîtes des choix d'implantation !

Consignes de rendu

- Ce projet est à faire en **binôme**.
- Sur e-learning, vous déposerez une archive contenant :
  - Les sources (`.java`) de votre projet (dossier `src` sous Eclipse), correctement commentés. Il vous sera demandé de fournir la javadoc de l'ensemble des classes et méthodes `public` de votre code ; pour les autres classes et méthodes, fournir une javadoc n'est pas nécessaire, mais on doit pouvoir comprendre aisément votre code grâce à son organisation, les noms de méthodes choisies et vos commentaires.
  - Un fichier `readme.pdf` qui explique
    - ce qui a été implémenté,
    - l'organisation du programme,
    - les choix techniques,
    - les éventuels problèmes rencontrés.

    Vous pouvez y ajouter tous les commentaires ou remarques que vous jugez nécessaires à la compréhension de votre code.
- Vous devrez effectuer deux rendus :
  - La phase 1 doit être terminée et déposée sur e-learning avant le 13 mai 2019 à 23h55. Il n'est pas exigé d'inclure une javadoc et un fichier `readme.pdf` pour cette version, mais c'est néanmoins recommandé.
  - La date limite de rendu final est le 11 juin 2019 à 23h55. Passé ce délai, la zone de rendu sera fermée et la note de votre projet sera 0.
- Une soutenance sera organisée le jeudi 13 juin 2019. Pendant cette soutenance, vous ferez une démonstration sur une machine des salles de TP et serez interrogés sur le projet.
- Vous perdrez des points si vous ne respectez pas ces consignes.
- Si le code de la version finale ne compile pas, la note de votre projet sera 0.