DEVOIR DE CONTROLE CONTINU

MIM4A1

A RENDRE POUR LE 6 AVRIL 2020

UNIVERSITÉ CAEN NORMANDIE

Rappel : le plagiat et l'échange de code sont interdits et passibles de sanctions.

1. Organisation

Ce devoir de Contrôle Continu est à réaliser en groupes de 4 étudiants. En cas de nécessité, des groupes de 3 étudiants pourront être acceptés.

Une partie du temps de travail de TP (voire l'intégralité des 4 derniers TP) pourra y être consacrée, selon les indications de vos enseignants respectifs, mais du travail hors des heures de TP sera indispensable pour mener à bien ce devoir.

Un dépôt devra être créé sous subversion pour chaque groupe (obligatoirement comme un sous-projet du projet "TP Compléments Objet L2"). Les noms court et long du dépôt devront comporter les noms des quatre membres du groupe, selon l'exemple suivant :

- nom court "durand-dupont-smith-doe"
- nom long "Génie Logiciel Durand, Dupont, Smith, Doe".

D'autre part, Céline Alec, Christophe Charrier, Pascal Fougeray, Olivier Ranaivoson et Yann Mathet devront être ajoutés comme **managers** du dépôt. Si ces points ne sont pas respectés, le devoir ne sera pas corrigé (note de 0).

Le 6 avril au soir (minuit), le code de chaque groupe sera extrait de son dépôt pour correction : un répertoire nommé *livraison* devra être présent à la racine et contenir :

- un répertoire *src* contenant le code commenté
- un répertoire *doc* contenant la Javadoc
- un répertoire *dist* contenant l'exécutable (un fichier .jar, et d'éventuelles ressources nécessaires à l'exécution)
- Et enfin un *répertoire* rapport contenant un mini rapport au format PDF contenant toute information utile à la compréhension de votre conception logicielle. Par exemple quelques diagrammes de classes précisant votre mise en oeuvre de MVC, le principe de vos algorithmes (mais pas directement le code), etc. Ce rapport pourra faire aux environs de 5 à 7 pages bien illustrées.

2. Critères d'évaluation

En premier lieu, nous prendrons en compte la qualité de l'architecture logicielle et de votre code. Nous regarderons notamment si votre conception est :

- facile à comprendre (répartition en packages et en classes cohérente, noms de classes et de méthodes éloquents, commentaires dans le code lorsque c'est utile)
- facile à maintenir et à faire évoluer (pas de code spaghetti, pas de code redondant, mise en oeuvre de MVC)
- robuste (tests effectués)

Bien sûr, une bonne ergonomie, un design agréable, des options supplémentaires seront appréciés mais ne constitueront pas l'essentiel de la note. En particulier, une application qui ferait ce qui est demandé dans le sujet mais qui ne répondrait pas aux critères d'évaluation exposés ci-dessus n'obtiendrait assurément pas la moyenne.

3. Sujet : Jeu de Puzzle à glissières (Taquin)

Le but de ce devoir est de réaliser une application de jeu, dotée d'une interface graphique, (mais pouvant être utilisé sans l'interface graphique) qui consiste en un puzzle à glissière comme illustré ci-dessous :

Une grille de n lignes sur m colonnes (3 sur 3 dans l'illustration) comporte n*m-1 éléments (c'est-à-dire qu'une case est vide).
Le joueur peut faire glisser l'un des éléments contigus à la case vide vers cette dernière (l'élément du haut, du bas, de gauche ou de droite).
À tout moment, il y a donc 2, 3 ou 4 éléments déplaçables.
Remarque : dans l'illustration ci-dessus, une pièce est en cours de déplacement.
Le but du jeu est de reconstituer l'image du puzzle par déplacements successifs. La partie est finie lorsque le puzzle est reconstitué.

Nous souhaitons une réalisation intégralement MVC, avec un modèle totalement indépendant de l'interface graphique. En particulier, le jeu devra être jouable en ligne de commande (par affichage sous forme de System.out.println(...) et contrôle au clavier).

La partie interface devra donc, comme toujours en MVC, qu'une partie Vue-Contrôleur supplémentaire venant se greffer sur le modèle.

Dans la partie modèle de cette application, le puzzle sera constitué de chiffres en ordre croissant sous la forme suivante (en configuration 3x3) :

1 2 3
4 5 6
7 8

Ce sera notamment le cas dans une utilisation en ligne de commande, vu qu'il n'est pas possible d'afficher d'image dans ce contexte.

Idéalement, les étudiants les plus avancés pourront faire en sorte que la version avec interface graphique soit capable de découper une image en n sur m éléments (et retirer le coin inférieur droit) afin de proposer une version du jeu correspondant à la première illustration. La partie modèle sera toujours la même (c'est-à-dire avec des chiffres), et la partie graphique pourra faire une correspondance entre un chiffre et un morceau d'image de puzzle.

Avant qu'une partie soit lancée, le puzzle sera bien sûr mélangé. Attention, un mélange quelconque peut mener, bien souvent, à une configuration sans solution (i.e. impossible à résoudre sans démonter le jeu). Il faudra donc mélanger en faisant glissez aléatoirement les pièces un nombre de fois suffisant.

Dans l'interface graphique, le contrôle sera fait de deux façons simultanées :

- un clic sur l'élément que l'on souhait déplacer (s'il est déplaçable). Un plus sera de mettre en exergue (par exemple en l'encadrant) un élément survolé par la souris lorsque celui-ci est déplaçable, ce qui permet à l'utilisateur de mieux voir ce qu'il peut faire.
- Un appui sur l'une des 4 flèches du clavier (par ex. la flèche de gauche signifie que l'on veut déplacer l'élément situé à gauche)