Application “Mine de diamants”

Cette application est un jeu de la famille des “bejeweled” : le joueur doit échanger des pièces adjacentes pour en aligner au moins trois de même couleur. Les pièces alignées disparaissent et sont remplacées par d’autres. Le nom initial de “bejeweled” est “Diamond Mine”. Ce jeu existe dans de très nombreuses versions. On se contente ici d’une version pédagogique …

L’application devra être développée en utilisant la bibliothèque Swing en respectant le modèle architectural MVC. Les questions doivent être traitées dans l’ordre, sans vouloir aller plus vite que ce qui est demandé. Ceux qui ont déjà une certaine expérience de la bibliothèque Swing trouveront largement de quoi s’occuper et profiteront de l’occasion pour développer avec plus de rigueur.

1. Dessiner l’arborescence des composants

Voici une capture d’écran du jeu. Dessiner l’arborescence des composants Swing (la racine de cette arborescence est **JFrame**) et déposer ce dessin sur arche.

Le plateau de jeu est affiché au centre dans un **JPanel** de **JButton**, géré par un **GridLayout**.

La partie droite de la fenêtre est un **JPanel** géré par un **BoxLayout**. Les deux **JPanel** qu’il contient contiennent eux-mêmes des **JLabel,** gérés en **GridLayout**. Le **JPanel** en bas contient des **JButton**.
Les **JPanel** sont dotés de bordures avec titres, créés par une fabrique de bordures (**BorderFactory**).

2. Dessiner le diagramme de classes

À partir de l’arborescence, dessiner le diagramme de classes de l’application, sans oublier les fonctions. Le modèle peut être restreint à une seule classe **Jeu** qui mémorise les pièces dans un tableau d’entiers. Déposer ce diagramme sur Arche à la fin de la première séance.

3. Implanter le modèle

Construire la classe **diamant.Jeu** en respectant le diagramme de classes. En particulier, cette classe doit gérer une collection de vues. Il faut donc également créer l'interface **diamant.Vue** avec la fonction ***void mettreAJour()*.**
Pour simplifier la création d'un jeu, cette classe définit quatre constantes entières statiques publiques : **SAPHIR**, **RUBIS**, **EMERAUDE** et **DIAMANT** et propose un constructeur avec un tableau d'entiers en paramètre.

4. Implanter la classe qui visualise le plateau du jeu

Construire la classe **diamant.VuePlateau** qui visualise le plateau de jeu, en respectant l'arborescence des composants. Pour éviter une perte de temps inutile, les images à utiliser vous sont fournies sur Arche. Vous pourrez en choisir d'autres plus tard.

Écrire la classe principale **diamant.LancerJeu** qui hérite de **JFrame** ; elle crée un jeu et affiche un plateau de jeu.

On pourra utiliser le tableau suivant pour la création d'un exemple de jeu.

```
int[ ][ ] val = { {Jeu.RUBIS, Jeu.EMERAUDE, Jeu.RUBIS, Jeu.SAPHIR, Jeu.SAPHIR, Jeu.EMERAUDE},
                  {Jeu.SAPHIR, Jeu.RUBIS, Jeu.EMERAUDE, Jeu.SAPHIR, Jeu.DIAMANT, Jeu.RUBIS},
                  {Jeu.DIAMANT, Jeu.DIAMANT, Jeu.EMERAUDE, Jeu.EMERAUDE, Jeu.RUBIS, Jeu.SAPHIR},
                  {Jeu.DIAMANT, Jeu.EMERAUDE, Jeu.DIAMANT ,Jeu.EMERAUDE, Jeu.SAPHIR, Jeu.EMERAUDE},
                  {Jeu.EMERAUDE, Jeu.DIAMANT, Jeu.SAPHIR, Jeu.SAPHIR, Jeu.RUBIS, Jeu.SAPHIR},
                  {Jeu.EMERAUDE, Jeu.SAPHIR, Jeu.EMERAUDE, Jeu.RUBIS, Jeu.RUBIS, Jeu.SAPHIR}} ;
```

5. Rendre les boutons réactifs

Un clic sur un bouton du plateau doit :

- ➢ afficher les coordonnées de la case sélectionnée dans le **JPanel** en haut à droite.
- ➢ mettre un cadre de couleur autour de la case sélectionnée

Que se passe-t-il dans l'application si on clique une seconde fois ?

6. Compléter la réaction des boutons

Deux clics successifs sur deux cases voisines sont possibles (on tient compte uniquement des quatre voisins Nord, Sud, Est, Ouest). Ils permettent de mémoriser les coordonnées des deux cases sélectionnées, soit dans les écouteurs soit dans le modèle.
Deux clics successifs sur la même case ou sur deux cases non voisines n'ont aucun effet (en dehors de désélectionner les cases).

7. Échanger les deux cases sélectionnées

Écrire l'écouteur du bouton qui permet d'échanger les deux cases sélectionnées. Ce bouton est inactif (grisé) en l'absence de cases sélectionnées.

Une fois l'échange fait, les cases doivent être désélectionnées.

8. Compter le nombre d'échanges effectués

Compter le nombre d'échanges réalisés depuis le début du jeu. Compléter le **JPanel** à droite avec le nombre d'échanges effectués.

9. Supprimer les alignements

Écrire l'écouteur du bouton qui permet de supprimer les alignements. Les cases correspondantes sont vides. Ce bouton reste toujours actif. En l'absence d'alignement, il n'a aucun effet.

Compléter le **JPanel** à droite avec le score (+10 à chaque alignement, par exemple).

10. Faire descendre les pierres

Écrire l'écouteur du bouton qui permet de faire descendre les pierres dans les cases vides, colonne par colonne. Ce bouton est inactif en l'absence de cases vides.

11. Générer de nouvelles pierres

Écrire l'écouteur du bouton qui permet de générer de nouvelles pierres dans les colonnes incomplètes. Ce bouton est inactif en l'absence de colonnes incomplètes.

12. Accélérer le jeu

Écrire l'écouteur du bouton **Rapide** qui permet, en un seul clic, de réaliser les opérations précédentes : échanger, supprimer les alignements, faire descendre les pierres, générer de nouvelles pierres. Toutes les étapes, sauf l'échange, doivent être réalisées tant que de nouveaux alignements apparaissent.

13. Recommencer une nouvelle partie

Ajouter un menu (**JMenuBar**/**JMenu**/**JMenuItem**) et une barre d'outils avec icônes (**JToolBar**/**JButton**) permettant de :

- ➢ quitter proprement l'application
- ➢ rejouer
- ➢ générer un plateau de jeu aléatoire
- ➢ changer la taille du plateau

14. Améliorer le look&feel de l'application

Remplacer le double-clic de **JButton** par un drag&drop. Ajouter un bouton qui permet de passer du mode "debug" au mode "jeu" :

- ➢ Mode "debug" : on garde tous les boutons qui permettent de suivre les étapes intermédiaires du jeu,
- ➢ Mode "jeu" : on allège l'interface de tout ce qui sert plus à rien pour le jeu.

Ajouter des tooltips sur les boutons de la **ToolBar**.