Cours : INF111 – Programmation orientée objet
Enseignant : Abdelmoumène Toudeft
Session : Hiver 2022
Groupes : 1 et 3

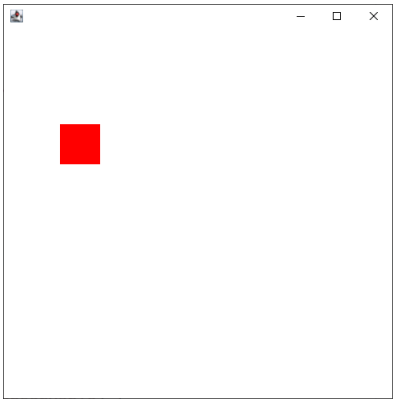
Laboratoire de révision

Objectif :

- Pratiquer la plupart des notions vues pendant la session.

Exercice 1 :

Vous devez réaliser l'application présentant l'interface graphique suivante à partir du projet de départ ***Labo_Revision_Exercice1*** fourni :



L'application permettra à l'utilisateur de déplacer le carré rouge sur le panneau en utilisant les flèches du clavier. Le carré ne doit pas sortir des limites du panneau.

Voici le travail demandé :

Attention : Vous devez respecter les principes de l'architecture MVC et répartir adéquatement les tâches entre le modèle, la vue et le contrôleur.

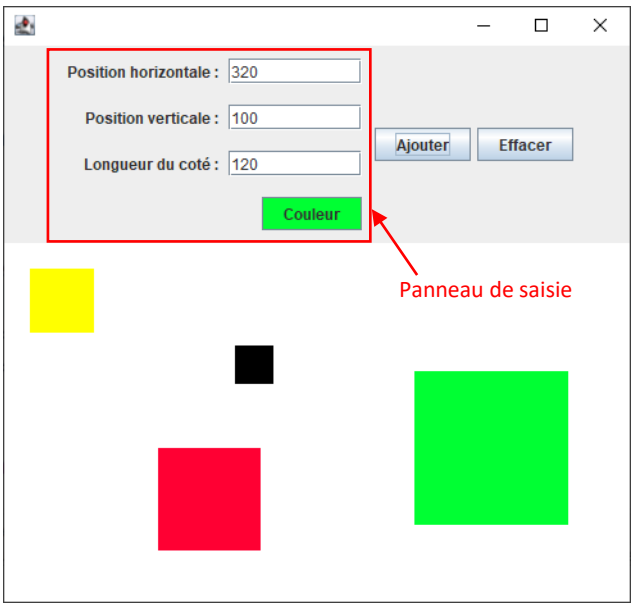
1. Vous devez créer par héritage (extension) le panneau contenant le carré. Le carré doit être dessiné chaque fois que le panneau s'affiche. La classe ***OutilDeDessin*** fournie contient une méthode permettant de dessiner le carré.
2. Utilisez le patron ***Observer*** pour que le panneau observe le carré afin de se rafraichir lorsque ce dernier se déplace (la classe ***observer.Observable*** et l'interface ***observer.Observateur*** sont fournies).
3. Ajoutez un écouteur d'événement clavier sur la fenêtre (***JFrame***) pour déplacer le carré sur la grille à l'aide des flèches du clavier. L'écouteur doit s'assurer que le carré ne sortira pas du panneau avant de valider le déplacement.
4. Vous devez compléter la classe ***Fenetre*** pour construire la fenêtre principale de l'application. Celle-ci doit créer un carré, un panneau et un écouteur d'événement clavier et mettre en place le nécessaire pour que l'application fonctionne correctement.

Cours : INF111 – Programmation orientée objet
Enseignant : Abdelmoumène Toudeft
Session : Hiver 2022
Groupes : 1 et 3

Laboratoire de révision

Exercice 2 :

Vous devez réaliser l'application présentant l'interface graphique suivante à partir du projet de départ ***Labo_Revision_Exercice2*** fourni :



L'application permettra à l'utilisateur d'ajouter des carrés au panneau en spécifiant la position et la taille de chaque carré. La couleur du carré est choisie à l'aide d'un sélecteur de couleur (***JColorChooser***).

Le bouton « ***Effacer*** » permettra d'effacer tous les carrés.

Voici le travail demandé :

Attention : Vous devez respecter les principes de l'architecture MVC et répartir adéquatement les tâches entre le modèle, la vue et le contrôleur.

1. Vous devez créer par héritage (extension) le **panneau de saisie** contenant les champs pour saisir les positions horizontale et verticale du carré et la longueur de son côté. Le panneau doit aussi contenir le bouton « ***Couleur*** » permettant de choisir la couleur du carré à l'aide d'un sélecteur de couleur. Le panneau doit fournir les méthodes permettant de récupérer les valeurs saisies dans les champs ainsi que la couleur choisie.
2. Utilisez une classe interne anonyme pour gérer l'événement d'action sur le bouton « ***Couleur*** » afin de choisir la couleur en utilisant un sélecteur de couleur.
3. Vous devez compléter la définition des méthodes de la classe ***PlanDeJeu*** pour que le plan de jeu stocke les carrés dans un vecteur (***ArrayList*** ou ***Vector***) et qu'on puisse ajouter des carrés au plan de jeu et le vider.
4. Vous devez créer par héritage (extension) le panneau contenant le plan de jeu (c'est le

Laboratoire de révision

panneau qui affiche les carrés du plan de jeu). Le plan de jeu sera fourni au constructeur du panneau. Les carrés doivent être dessinées chaque fois que le panneau s'affiche. La classe ***OutilDeDessin*** contient une méthode permettant de dessiner un carré.

5. Vous devez rendre le plan de jeu observable et le panneau observateur. Le panneau doit se rafraichir chaque fois qu'un carré est ajouté au plan de jeu ou que le plan de jeu est vidé de ses carrés.
6. Vous devez créer une classe écouteur d'événements d'action. La classe doit encapsuler un plan de jeu et un panneau de saisie (fournis au constructeur).
7. L'écouteur doit gérer l'événement d'action comme suit :
    7.1. Récupérer les valeurs saisies dans les champs du panneau de saisie.
    7.2. Valider les saisies pour s'assurer que ce sont des valeurs entières (en gérant l'exception ***NumberFormatException***). Si au moins une des saisies n'est pas correcte, il faut afficher un message en conséquence dans une **boite de dialogue de message** et ne pas faire les traitements des points ***7.3*** et ***7.4*** suivants.
    7.3. Créer un carré ayant les caractéristiques saisies par l'utilisateur.
    7.4. Ajouter le carré au plan de jeu.
8. Vous devez compléter la fenêtre de l'application pour qu'elle contienne :
    8.1. Au nord, un panneau de saisie, un bouton « **Ajouter** » et un bouton « **Vider** ».
    8.2. Au centre, un panneau affichant les carrés.

    La classe doit aussi créer un plan de jeu et un écouteur d'événement d'action (classe créée au point 6) et mettre en place les mécanismes permettant à l'application de fonctionner correctement (enregistrer écouteurs et observateurs).
9. Utilisez une classe interne anonyme pour gérer le bouton « ***Vider*** » de manière à vider le plan de jeu.

Exercice 3 :

Le projet de départ ***Labo_Revision_Exercice3*** contient les classes implémentant les listes doublement chaînées et les files simplement chaînées vues en cours.

3.1. Complétez la méthode ***inverser()*** de la classe ***FileChaineeSimple*** pour inverser l'ordre des éléments de la file (vous devez modifier le chainage et non pas permuter les données contenues dans les nœuds).

3.2. Complétez la méthode ***inverser()*** de la classe ***ListeChaineeDouble*** pour inverser l'ordre des éléments de la liste (vous devez modifier le chainage et non pas permuter les données contenues dans les nœuds).