Institut supérieur des études technologiques de Mahdia

Département technologies de l'informatique

Parcours : Développement des systèmes informatiques (DSI2)

Atelier programmation orientée objet avancée

Atelier 5 : Interfaces graphiques

Elaboré par :
Mohamed Aymen Charrada
Email : mohamed_aymen_charrada@yahoo.fr

07/04/2025

L'objectif de cet atelier est de renforcer votre maîtrise de la bibliothèque Swing, en explorant de nouveaux composants graphiques plus avancés. Vous apprendrez également à manipuler les gestionnaires de disposition (*layout managers*) pour concevoir des interfaces utilisateur à la fois riches, organisées et ergonomiques. Cet atelier vous permettra de consolider vos compétences dans la création d'applications Java graphiques interactives et structurées.

Déroulement des ateliers

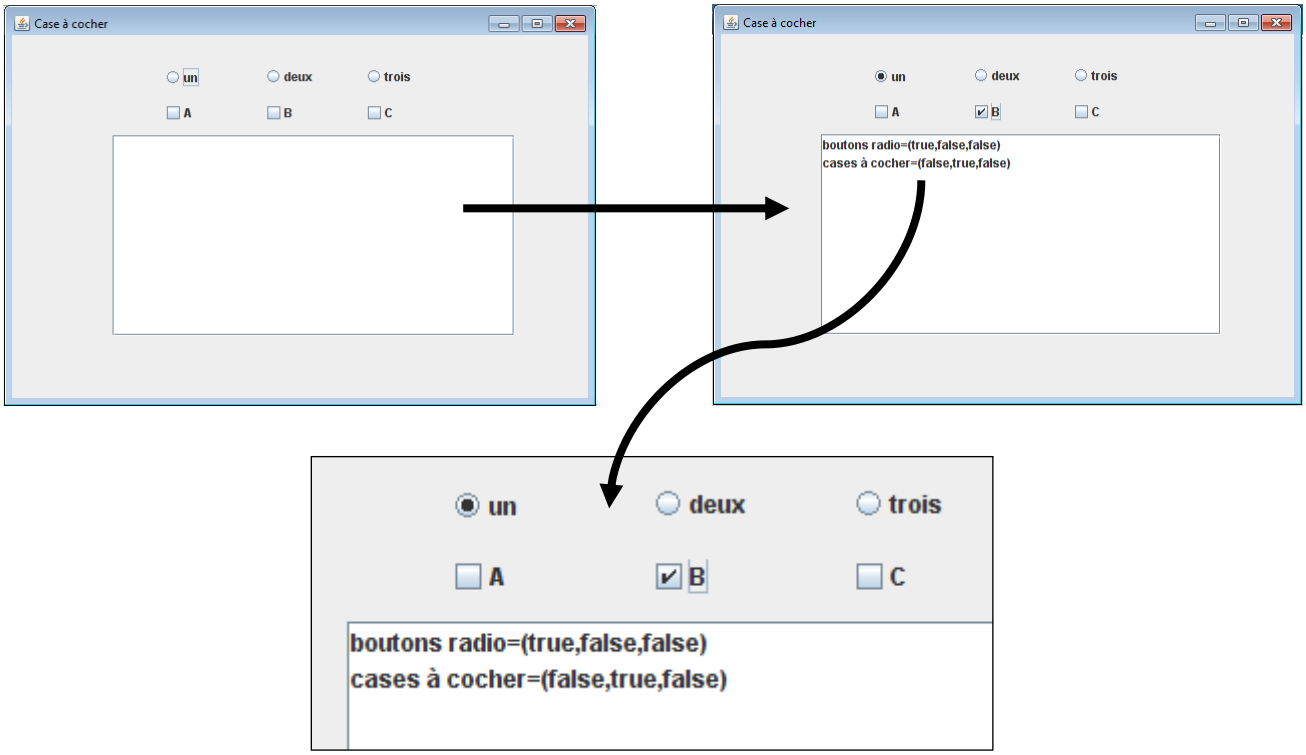
- Le travail au cours des ateliers doit être individuel ou bien en binôme.
- Chaque étudiant aura une note pour chaque séance de TP. Cette note sera calculée en fonction des critères suivants : comportement dans la salle, retard, participation et compte rendu. Une absence non justifiée signifie automatiquement un zéro.
- Un compte rendu numérique doit être envoyé à la fin de la séance à l'adresse « **boitedetestdsi3@gmail.com** », avec l'objet « **TP5_POA** », contenant votre (vos) nom(s), tous les codes sources et les réponses aux questions.

Travail demandé

Créez un nouveau projet, nommé « TP5 » et ajoutez un nouvel package « interfaces ».

Partie 1 : Gestion des composants Cases à cocher « JCheckBox », des boutons radio « JButtonRadio » et des listes « JList »

Créez la classe « **Interface1** » qui définit l'interface suivante :

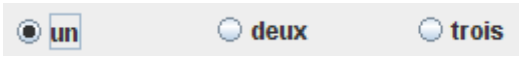


L'interface comporte les éléments graphiques suivants :

- Trois boutons radio (un, deux, trois)
- Trois cases à cocher (A, B, C)

- Une liste de texte avec ascenseur vertical

Initialement, les boutons radio et les cases à cocher ne sont pas cochés. Suite à chaque évènement où un composant est coché ou décoché, un message doit être ajouté dans la liste de texte indiquant la source de la modification et le nouvel état du composant modifié.
Par exemple : si le bouton radio « un » est coché, alors la ligne suivante s'ajoute dans la zone de texte :

Boutons radio = (True, False, False) ⇔ 

NB :

Les boutons radio doivent **obligatoirement** être insérés dans un composant invisible de type « *ButtonGroup* ». Une fois le groupe de boutons radio créé, on peut lui associer chacun des boutons radio en utilisant la méthode « *add(JRadioButton)* ».
La méthode « *Boolean isSelected()* » indique si une case à cocher ou un bouton radio est coché ou non.

Partie 2 : Gestionnaires de disposition

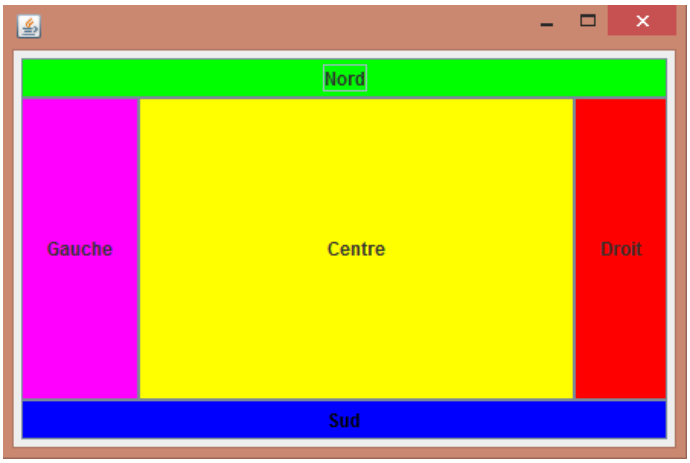
Dans cette partie, on examinera une notion fondamentale qui intervient dans toutes les interfaces graphiques : c'est **le gestionnaire de disposition**. Un gestionnaire de disposition (ou *layout manager*) est un objet invisible associé à un conteneur ; il se charge du placement et du dimensionnement de chaque composant inclus dans le conteneur.
Dans cette partie, vous allez examiner les principaux gestionnaires de disposition prédéfinis :

- **BorderLayout**: Permet de placer les composants dans cinq zones : le centre, le nord, le sud, l'est et l'ouest. Au besoin, ces composants sont étirés pour atteindre une largeur et/ou une hauteur déterminée par le gestionnaire.
- **FlowLayout**: Place les composants, sans les étirer, de la gauche vers la droite, puis du haut vers le bas.
- **GridLayout**: Place des composants, étirés si nécessaire, en matrice dont toutes les cases sont identiques, ayant un nombre de lignes et de colonnes défini à l'avance.

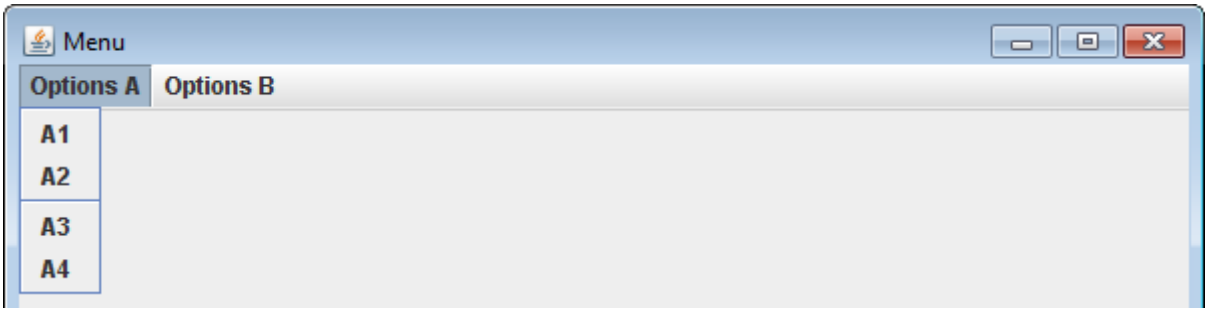
Chaque conteneur est associé à un gestionnaire de disposition par défaut. La méthode « *setLayout(LayoutManager)* » permet de créer et associer un gestionnaire de disposition à un conteneur.

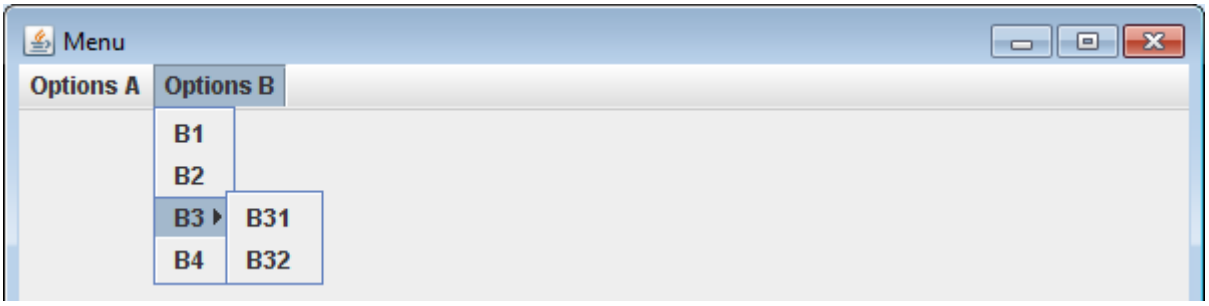
1. Ajoutez la classe « **Interface2** » qui définit une interface de taille 400 x 300 comportant cinq boutons (objets JButton) étiquetés « Nord », « Droit », « Sud », « Gauche », « Centre », placés dans les cinq zones reconnues par le gestionnaire « *BorderLayout* » *NORTH*, *EAST*, *SOUTH*, *WEST* et *CENTER* (ces identificateurs sont les noms des constantes de la classe « *BorderLayout* »). Faites en sorte que ces boutons soient de couleurs différentes. Pour donner une couleur à composant, vous aurez besoin d'utiliser la syntaxe suivante :

```
Nom_composant.setBackground(Color.nom_couleur);
```

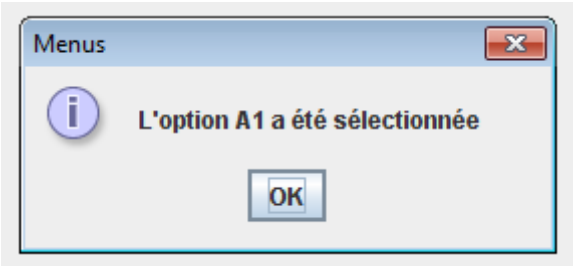


2. Vous allez créer maintenant la classe « **Interface3** » qui définit la fenêtre suivante :





Afin de simplifier les choses, nous allons limiter les actions à la simple appuie sur l'option A1 qui permet d'obtenir le message suivant :



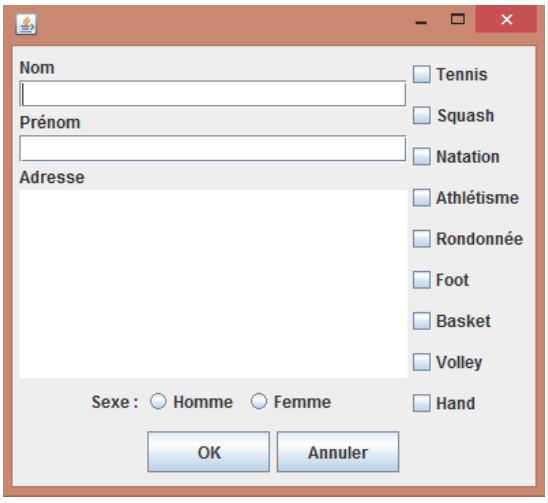
Notez bien que cette interface admet « *BorderLayout* » comme gestionnaire de disposition et que la barre de menu doit être placée dans la zone nord ("*North*") du gestionnaire. Vous trouvez, dans ce qui suit, la liste des différents composants nécessaires pour réaliser une telle interface :

```
JPanel contentPane; //conteneur principale
BorderLayout borderLayout1 = new BorderLayout();
JMenuBar jMenuBar1 = new JMenuBar();
```

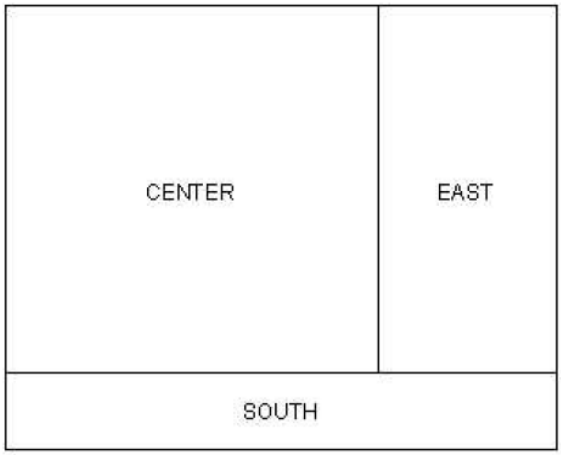
```
JMenu jMenu1 = new JMenu();
JMenuItem jMenuItem1 = new JMenuItem();
JMenuItem jMenuItem2 = new JMenuItem();
JMenuItem jMenuItem3 = new JMenuItem();
JMenuItem jMenuItem4 = new JMenuItem();
JMenu jMenu2 = new JMenu();
JMenuItem jMenuItem5 = new JMenuItem();
JMenuItem jMenuItem6 = new JMenuItem();
JMenu jMenu3 = new JMenu();
JMenuItem jMenuItem7 = new JMenuItem();
JMenuItem jMenuItem8 = new JMenuItem();
JMenuItem jMenuItem9 = new JMenuItem();
```

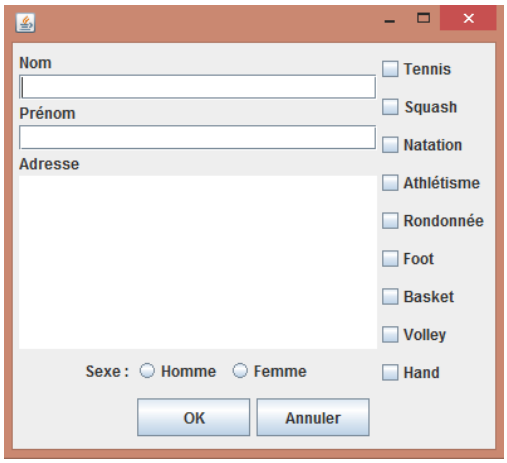
3. Réalisez la classe « **Interface4** » qui définit une interface contenant douze boutons (classe *JButton*) étiquetés « Janvier », « Février », etc. Les étiquettes des boutons seront définies par un tableau de chaînes de caractères constant. Le gestionnaire de disposition associé au panneau de contenu sera de type « *FlowLayout* » :

4. Les exercices précédents montrent comment faire fonctionner les gestionnaires de disposition basiques, mais ne donnent pas vraiment l'impression qu'avec ces gestionnaires on peut réaliser des interfaces graphiques comme on a l'habitude d'en voir dans les applications courantes. Pour vous convaincre que la chose est possible, vous allez réaliser l'interface graphique suivante, correspondant à un masque de saisie pour l'acquisition des membres d'un club où on pratique divers sports :

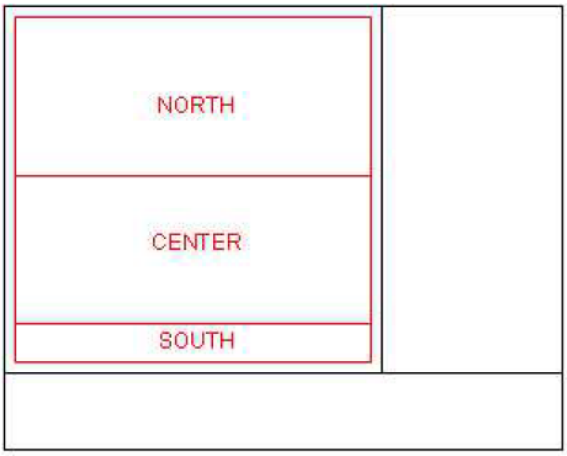


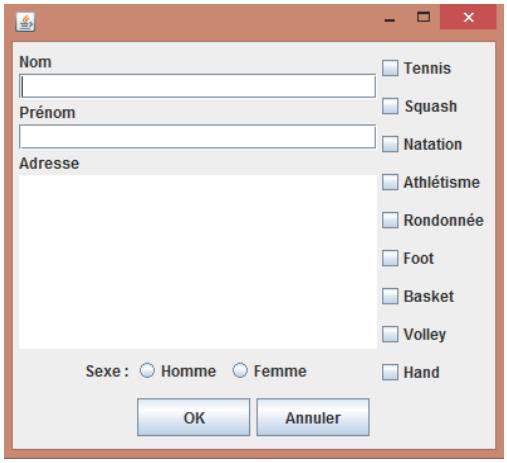
Une première organisation de notre interface nous ramène à définir trois panneaux gérés par un *BorderLayout* comme sur la figure suivante :



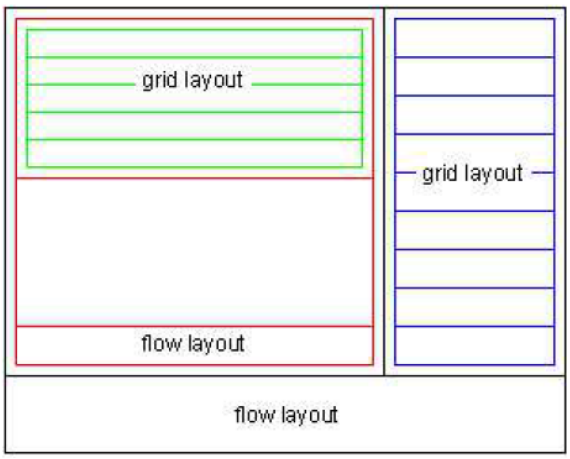


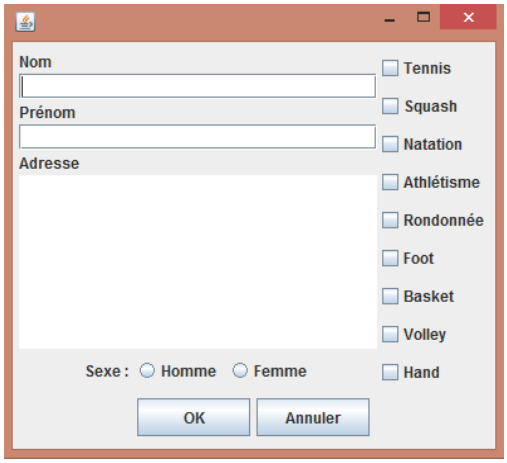
Le panneau central porte neuf composants. Les cinq premiers (l'étiquette *Nom*, un champ de texte, l'étiquette *Prénom*, un autre champ de texte et l'étiquette *Adresse*) se partagent verticalement et régulièrement un premier panneau. Les trois derniers composants (l'étiquette *Sexe* et les deux boutons-radio *Homme*, *Femme*) occupent un deuxième panneau. Finalement, le composant restant est une zone de texte. Ce qui nous amène à organiser le panneau central comme trois composants (deux panneaux et une zone de texte) gérés par un second *BorderLayout* comme sur la figure suivante :





Enfin, deux *GridLayout* et deux *FlowLayout* sont nécessaires pour gérer les composants élémentaires (cases à cocher, boutons, etc.) :





Créez la classe « **Interface5** » qui définit cette interface.