



---

## TP1 (Annexe)

---

Ce TP est une introduction au développement d'application graphique en Java à l'aide de la bibliothèque Swing.

**Java Swing** est une bibliothèque de composants graphiques pour la création d'interfaces utilisateur (**GUI**) pour les applications Java. Elle est basée sur la bibliothèque **AWT (Abstract Window Toolkit)**, mais elle offre une plus grande flexibilité et des performances améliorées.

- **Swing** a été conçu plus spécifiquement pour créer des applications pour ordinateur (desktop environment).
- Il est composé d'un ensemble de classes qui se trouvent dans le package *javax.swing* ainsi que dans ses sous-packages
- Depuis **Java 8**, Swing n'est plus la bibliothèque de référence pour créer des applications graphiques : elle a été remplacée par **JavaFX**. Néanmoins, **Swing** reste encore largement connu et donc utilisé par les développeurs.
- **Swing** a lui-même remplacé une bibliothèque Java plus ancienne nommée **AWT (Abstract Window Toolkit)**. Cette dernière offrait un accès en Java à l'API graphique du système alors que Swing offre des composants graphiques totalement implémentés en Java (et donc avec un comportement et un rendu identiques quel que soit le système sous-jacent). AWT est toujours présent dans la bibliothèque standard de Java et ses classes se trouvent dans le package *java.awt*. Swing utilise une partie des classes fournies par AWT.
- Pour utiliser les classes de la bibliothèque AWT, vous devez les importer depuis ce chemin : *java.awt*.
- Pour utiliser les classes de la bibliothèque Swing, vous devez les importer depuis ce chemin : *javax.swing*.

```
import java.awt  
import javax.swing
```

nous pouvons diviser les classes de la bibliothèque Swing en deux parties : **conteneurs** et **composants**.

**Les conteneurs** sont des composants qui peuvent contenir d'autres composants. Ils sont utilisés pour organiser les composants et créer des structures de présentation complexes. Les conteneurs Swing les plus courants sont :

- **JFrame** : une fenêtre qui peut contenir d'autres composants.
- **JPanel** : un panneau simple qui peut contenir d'autres composants.
- **JDialog** : une boîte de dialogue qui peut contenir d'autres composants.
- **JSplitPane** : un panneau qui peut être divisé en deux parties.
- **JTabbedPane** : un panneau qui peut contenir plusieurs onglets.
- **JScrollPane** : un panneau qui peut contenir un composant et permet de le faire défiler.

**Les composants** sont des éléments de l'interface graphique qui peuvent interagir avec l'utilisateur. Ils sont utilisés pour afficher des informations, collecter des données ou permettre à l'utilisateur d'effectuer des actions. Les composants Swing les plus courants sont :

- **JButton** : un bouton qui peut être cliqué par l'utilisateur.
- **JLabel** : une étiquette qui peut afficher du texte ou une image.
- **TextField** : une zone de texte qui permet à l'utilisateur de saisir du texte.

- **JTextArea** : une zone de texte étendue qui permet à l'utilisateur de saisir une grande quantité de texte.
- **JCheckBox** : une case à cocher qui peut être cochée ou décochée par l'utilisateur.
- **JRadioButton** : un bouton radio qui peut être sélectionné par l'utilisateur.
- **JComboBox** : une liste déroulante qui permet à l'utilisateur de sélectionner une option.
- **JList** : une liste qui permet à l'utilisateur de sélectionner une ou plusieurs options.
- **JMenuBar, JMenu et JMenuItem** : Composants pour créer des barres de menus et des éléments de menu.
- **JFileChooser** : Composant pour sélectionner des fichiers ou des répertoires.
- **JColorChooser** : Composant pour choisir une couleur.
- **JTree** : Composant pour afficher des données hiérarchiques sous forme d'arborescence.



- **Les containers dans Java Swing** Les conteneurs font partie intégrante des composants GUI SWING. Un conteneur fournit un espace où un composant peut être localisé. Deux types:

- a) **Heavyweight Containers (Top-Level):** JDialog, JWindow, et JFrame.
  - b) **Lightweight Containers:** Par défaut (ContentPane), JPanel, etc.
- 

## L'objet JFrame

Voici les méthodes de la composant JFrame de Swing :

- **Méthodes d'initialisation**
  - `JFrame()` : crée une nouvelle fenêtre sans titre, sans taille et sans position.
  - `JFrame(String title)` : crée une nouvelle fenêtre avec le titre spécifié.
  - `JFrame(String title, int width, int height)` : crée une nouvelle fenêtre avec le titre et la taille spécifiés.
  - `JFrame(String title, int width, int height, int x, int y)` : crée une nouvelle fenêtre avec le titre, la taille et la position spécifiés.
- **Méthodes d'affichage**
  - `setVisible(boolean visible)` : rend la fenêtre visible ou invisible.
  - `pack()` : ajuste la taille de la fenêtre pour s'adapter à ses composants.
  - `setLocationRelativeTo(Component c)` : centre la fenêtre par rapport au composant spécifié.
  - `setLocation(int x, int y)` : définit la position de la fenêtre.
  - `setSize(int width, int height)` : définit la taille de la fenêtre.
  - `setIconImage(Image image)` : définit l'icône de la fenêtre.
- **Méthodes de gestion des événements**
  - `add(Component c)` : ajoute un composant à la fenêtre.
  - `remove(Component c)` : supprime un composant de la fenêtre.
  - `setFocusable(boolean focusable)` : définit si la fenêtre est centrée sur l'écran lors de son affichage.
  - `setFocusableWindowState(boolean focusable)` : définit si la fenêtre peut recevoir le focus.
  - `toFront()` : ramène la fenêtre au premier plan.
  - `toBack()` : envoie la fenêtre en arrière-plan.
- **Méthodes d'accès aux propriétés**
  - `getTitle()` : retourne le titre de la fenêtre.
  - `getWidth()` : retourne la largeur de la fenêtre.
  - `getHeight()` : retourne la hauteur de la fenêtre.
  - `getX()` : retourne la position horizontale de la fenêtre.
  - `getY()` : retourne la position verticale de la fenêtre.
  - `getIconImage()` : retourne l'icône de la fenêtre.
  - `isVisible()` : retourne si la fenêtre est visible.
  - `isFocusable()` : retourne si la fenêtre est centrée sur l'écran lors de son affichage.
  - `isFocusableWindowState()` : retourne si la fenêtre peut recevoir le focus.

Voici un exemple d'utilisation des méthodes de JFrame :

- Pour définir le titre et la taille de la fenêtre

```
JFrame frame = new JFrame();
frame.setTitle("Mon application");
frame.setSize(300, 200);
```

## L'objet JButton

Voici les méthodes de la composant JButton de Swing :

- **Méthodes d'initialisation**
  - `JButton()` : crée un nouveau bouton sans texte ni icône.
  - `JButton(String text)` : crée un nouveau bouton avec le texte spécifié.
  - `JButton(Icon icon)` : crée un nouveau bouton avec l'icône spécifiée.
  - `JButton(String text, Icon icon)` : crée un nouveau bouton avec le texte et l'icône spécifiés.
- **Méthodes d'affichage**
  - `setText(String text)` : définit le texte du bouton.
  - `setIcon(Icon icon)` : définit l'icône du bouton.
  - `setFont(Font font)` : définit la police du texte du bouton.
  - `setForeground(Color fg)` : définit la couleur du texte du bouton.
  - `setBackground(Color bg)` : définit la couleur de fond du bouton.
  - `setBorder(Border border)` : définit la bordure du bouton.
  - `setRolloverEnabled(boolean rolloverEnabled)` : définit si le bouton doit afficher une bordure lorsque la souris est survolée.
  - `setRolloverIcon(Icon rolloverIcon)` : définit l'icône du bouton lorsque la souris est survolée.
  - `setSelected(boolean selected)` : définit si le bouton est sélectionné.
  - `setMnemonic(int mnemonic)` : définit la touche de raccourci du bouton.
  - `setAccelerator(KeyStroke accelerator)` : définit la touche de raccourci du bouton.
- **Méthodes de gestion des événements**
  - `addActionListener(ActionListener listener)` : ajoute un écouteur d'événements au bouton.
  - `removeActionListener(ActionListener listener)` : supprime un écouteur d'événements du bouton.
- **Méthodes d'accès aux propriétés**
  - `getText()` : retourne le texte du bouton.
  - `getIcon()` : retourne l'icône du bouton.
  - `getFont()` : retourne la police du texte du bouton.
  - `getForeground()` : retourne la couleur du texte du bouton.
  - `getBackground()` : retourne la couleur de fond du bouton.
  - `getBorder()` : retourne la bordure du bouton.
  - `isSelected()` : retourne si le bouton est sélectionné.
  - `getMnemonic()` : retourne la touche de raccourci du bouton.
  - `getAccelerator()` : retourne la touche de raccourci du bouton.

Voici un exemple d'utilisation des méthodes de JFrame :

- Pour définir le texte et l'icône du bouton

```
JButton bouton = new JButton("Mon bouton");
bouton.setText("Nouveau texte");
bouton.setIcon(new ImageIcon("mon_icone.png"));
```