ISET de Sousse

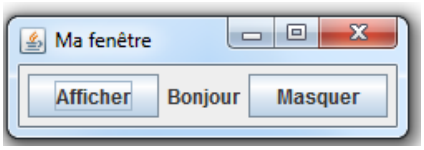
Atelier Programmation Objet Avancée
(Initiation à Swing)

Objectif

Réaliser une application graphique simple qui réagit aux actions de l'utilisateur.

Exercice 1

1. Compléter le programme suivant permet de créer une fenêtre contenant : deux boutons et un label au milieu qui contient le texte « Bonjour » :




```
import javax.swing.*;
public class Fenetre  extends …… {
       …… show, hide;
       JLabel message;
       public Fenetre (){
              setTitle("Ma fenêtre");
              setLocationRelativeTo(null);
              setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
              show = new JButton("Afficher");
              message = new ……("Bonjour");
              hide = new JButton("Masquer");
              JPanel pano = new ……();  // creation d'un panneau d'affichage
              pano.add(show);
              pano.add(message);
              pano.add(hide);
              this. ……(pano); // ajouter le panneau à la fenêtre
              pack();       // Ajuster la taille des composants
              this. ……(true);// rendre la fenêtre visible
       }
       public static void main(String[] args){
              …… Fenetre();  // création de la fenêtre
       }
}
```

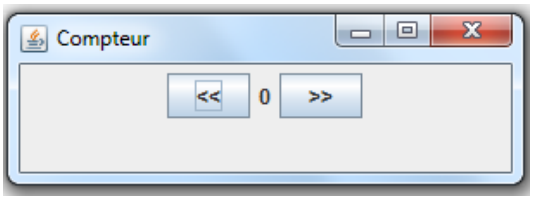
2. Dans le programme précédent, un clic sur les boutons n'a aucun effet, pour activer ces boutons de façon qu'ils affichent ou masquent respectivement le message « Bonjour », il faut associer un écouteur (**ActionListener**) à chaque bouton et développer une méthhode « actionPerformed » qui gère les événements appropriés.

Compléter puis tester le programme suivant :

```
import java.awt.*;
import javax.swing.*;
import java.awt.event.*;

public class FenetreAvecEcouteur extends JFrame implements ActionListener {
    JButton show, hide;
    JLabel message;
    public FenetreAvecEcouteur(){
              setTitle("Ma 4ème fenêtre");
              setLocationRelativeTo(null);
              setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
              show = new JButton("Afficher");
              show.addActionListener(this);   // ajout d'un écouteur
              message = new JLabel("Bonjour");
              hide = new JButton("Masquer");
              ……                              // ajout d'un écouteur
              JPanel pano = new JPanel();
              pano.add(show);
              pano.add(message);
              pano.add(hide);
              setContentPane(pano);
              pack();
              setVisible(true);
    }

    public void actionPerformed(ActionEvent e){
       if(e.getSource() == show)
              message.setVisible(true);
      else if(e.getSource() == hide)
              ……                                    }
    }

    public static void main(String[] args){
              new FenetreAvecEcouteur();
    }
}
```

Exercice 2 (Compteur)

Ecrire un programme en Java qui affiche un compteur initialisé à zéro et dont la valeur s'incrémente après chaque clic sur le bouton *plus* (>>) et se décrémente après chaque clic sur le boutons *moins* (<<).



Annexe : Gestion des événements

Fonctionnement général

- A un composant graphique (JButton, par exemple), on attache un objet capable d'écouter les événements (ActionListener).

- Quand l'utilisateur réalise une action sur le composant (clic), un événement est généré (ActionEvent).

- La machine virtuelle reçoit tous les événements, mais seul les événements écoutés déclenchent une réaction (dont le code se trouve dans la méthode actionPerformed() de l'interface ActionListener).

Quelques types d'événement

- ActionEvent / ActionListener (déclenchés par JButton)
- ItemEvent / ItemListener (déclenchés par JCheckBox)
- MouseEvent / MouseListener (déclenchés par la souris sur un composant)
- KeyEvent / KeyListener (touche clavier)
- ...

ActionEvent / ActionListener

Le couple ActionEvent / ActionListener est le plus commun.

- ActionEvent est la classe des événements déclenchés par une action bien définie sur un composant. Elle possède notamment la méthode getSource() pour obtenir le composant ayant déclenché l'événement.

- Un objet issu d'une classe qui implémente l'interface ActionListener, doté de la méthode actionPerformed(), est attaché à un tel composant par la méthode addActionListener().