TD3 - Interface graphique

Vendredi 1 mars 2013

Exercice 1 (Exercice 1 : Architecture SWING)
Dans cet exercice, on veut créer une fenêtre, spécifier ses paramètres, et organiser son contenu en panneaux.

1. Créer une nouvelle classe héritant de la classe *JFrame*. Modifier le constructeur de cette classe afin de créer une fenêtre centrée, en précisant son titre. Modifier sa taille à 600 par 200. Que faut-il préciser pour que la fenêtre s'affiche ? Vous remarquerez que fermer cette fenêtre ne termine pas l'application Java. Y remédier, à l'aide de *setDefaultCloseOperation*.
2. Définir le conteneur principal en ajoutant un attribut de classe *JPanel*, et en utilisant la méthode *setContentPane*. Afin de pouvoir visualiser ce panneau, modifier sa couleur de fond à l'aide de la méthode *setBackGround()*.
3. Préciser le layout du conteneur en *BorderLayout* à l'aide de la méthode *setLayout*. Définir 3 panneaux, que l'on ajoutera au conteneur, respectivement au nord, au centre, et au sud. Modifier leur couleur de fond pour les distinguer sur la fenêtre.

Solution :

```
import java.awt.BorderLayout;
import java.awt.Color;
import java.awt.Container;
import java.awt.Dimension;

import javax.swing.JFrame;
import javax.swing.JPanel;

public class TD3 extends JFrame {

  public TD3(String title) {
     super(title);
     setSize(600, 200);

     // partie 2
     // JPanel main_panel = new JPanel();
     // main_panel.setBackground(new Color(255, 0, 0));
     // setContentPane(main_panel);

     // partie 3
     Container cp = this.getContentPane();
     JPanel panel1 = new JPanel();
     panel1.setBackground(new Color(255, 0, 0));
     JPanel panel2 = new JPanel();
     panel2.setBackground(new Color(0, 255, 0));
     panel2.setPreferredSize(new Dimension(600, 200));
     JPanel panel3 = new JPanel();
     panel3.setBackground(new Color(0, 0, 255));
     cp.add(panel1, BorderLayout.PAGE_START);
     cp.add(panel2, BorderLayout.CENTER);
     cp.add(panel3, BorderLayout.PAGE_END);
```

```
    setVisible(true);
    setDefaultCloseOperation(javax.swing.JFrame.EXIT_ON_CLOSE);

  }

  public static void main(String[] args) {
    JFrame frame = new TD3("new JFrame");
  }
}
```

Exercice 2 (Exercice 2 : les composants)
Le but de cet exercice est de garnir notre fenêtre de composants (boutons, champs de texte, labels...) en les ajoutants aux panneaux créés.

1. Ajouter 2 boutons (classe *JButton*) en précisant leur texte (*setText()*), sur le panneau du centre. Y ajouter également un champ de texte (*JTextField*).
2. Ajouter 2 labels (classe *JLabel*), un sur le panneau nord, un sur le panneau sud. Préciser leur texte pour vérifier qu'ils sont bien placés. Modifier le layout du panneau du centre pour observer les déplacements des boutons et du champ de texte. Essayer notamment le layout *GridLayout*, pour observer l'effet de la modification de ses 2 arguments.

Solution : *Après la définition de* panel2 *ajoutez :*

```
JPanel panel2 = new JPanel();
...
// change layout for panel 2
panel2.setLayout(new GridLayout(3, 1));

// creating buttons and a text field
JButton btn1 = new JButton();
btn1.setText("btn1");
JButton btn2 = new JButton();
btn2.setText("btn2");
JTextField txt = new JTextField();
txt.setText("a text field");

// adding them to panel2
panel2.add(btn1);
panel2.add(btn2);
panel2.add(txt);
```

Exercice 3 (Exercice : les évènements)
Le but de cet exercice est de rendre "vivante"la fenêtre que l'on vient de construire.

1. Définir 2 classes internes, héritant de la classe *ActionListener*, pour y redéfinir la méthode *actionPerformed()*. Pour le moment, laisser ces méthodes vides. Ces 2 classes correspondent chacune à un des deux boutons crées dans l'exercice précédent. Ajouter à chacun de ces boutons un auditeur (*addActionListener*) de la classe correspondante.

2. On veut ajouter un compteur, qui compte le nombre de clics sur l'un ou l'autre des boutons, et affiche le résultat dans le label nord. Ajouter en attribut à la classe fenêtre ce compteur (entier), et l'afficher sur le label nord à l'initialisation de la fenêtre. Ajouter une ligne dans les méthodes *actionPerformed* incrémentant l'attribut compteur de 1 à chaque clic. Pourquoi cela ne fonctionne-t-il pas ? Que proposez vous ?
3. On veut que le premier bouton affiche sur le label sud le texte contenu dans le champ de texte, et que le second bouton efface ce texte. On utilisera les méthodes *setText()* et *getText()*.
4. Quand le nombre de sources d'évènements (boutons, etc) devient important, il peut être judicieux de ne pas créer une classe d'auditeur par source. Un auditeur peut être abonné à plusieurs sources, et avoir des instructions différentes en fonction de ces sources. Mettre ceci en pratique en redéfinissant une unique classe interne héritant d'*ActionListener*. Définir sa méthode *actionPerformed* de façon à distinguer un clic sur le premier bouton, qu'on pourra appeler "afficher message", et le second, qu'on pourra appeler "effacer". On pourra utiliser les méthodes *getSource()* et *getText()*.

Solution : *Paries 1,2 et 3*

```
public class TD3 extends JFrame {

     // fields, if we do not keep the JLabel as a field we cannot update it
     int counter = 0;
    JLabel lb1 = new JLabel();
    JLabel lb2 = new JLabel();
    JTextField txt = new JTextField();


     class Listener1 implements ActionListener {
          public void actionPerformed(ActionEvent arg0) {
               ++counter;
               lb1.setText("count " + counter);
               lb2.setText(txt.getText());
          }
     }

     class Listener2 implements ActionListener {
          public void actionPerformed(ActionEvent arg0) {
               ++counter;
               lb1.setText("count " + counter);
               lb2.setText("");
          }
     }
  ....
  public TD3(String title) {
  ....
          btn1.addActionListener(new Listener1());
          btn2.addActionListener(new Listener2());
  }
}
```

Partie 3 : Il faut avoir les boutons accessibles (champs)

```
public class TD3 extends JFrame {
....
     JButton btn1, btn2;

     class Listener3 implements ActionListener {
          public void actionPerformed(ActionEvent e) {
               if (e.getSource() == btn1) {
                    ++counter;
                    lb1.setText("count " + counter);
                    lb2.setText(txt.getText());
               } else if (e.getSource() == btn2) {
                    ++counter;
                    lb1.setText("count " + counter);
                    lb2.setText("");
               }
          }
     }
....
  public TD3(String title) {
  ....
   Listener3 myListener = new Listener3();
          btn1.addActionListener(myListener);
          btn2.addActionListener(myListener);
  }
```

Pour éviter tous les variables globales pensez aux classes internes anonymes.