C. Recanati
IG Java
INFO 2, 2014

TP n° 5bis

Exercice 1

On va implanter un programme analogue à ceux vus en cours sur les différents LayoutManager, et qui affichaient 6 boutons. Ici, les boutons créés utiliseront 2 fontes de taille différentes (une assez grosse et une petite). En outre, les textes de ces boutons seront de différentes longueurs.
On va tester ici le LayoutManager de classe GridLayout, pour afficher 12 boutons sur 3 lignes et 4 colonnes. On observera attentivement ce qui se passe si on agrandit ou rétrécit la fenêtre.

Exercice 2 (*suite du programme Sketcher, logiciel de dessin*).

Rappel : On a créé une zone de texte sous une barre de menus qui comporte trois menus : `Fichier`, `Elements` et `Aide`. Dans cet exercice, on va se servir de la zone texte pour afficher des messages indiquant l'origine des événements se produisant quand on actionne les items des menus.
1) Implémenter l'interface `ActionListener` dans la fenêtre cadre (JFrame) pour traiter l'affichage des événements se produisant lors de la sélection des items du menu `Fichier.`
☞ On pourra utiliser la procédure suivante pour l'affichage des informations (cf. TP1):

```
// Returne juste le nom de la classe – sans les packages
protected String getClassName(Object o) {
      String classString = o.getClass().getName();
      int dotIndex = classString.lastIndexOf(".");
      return classString.substring(dotIndex+1);
}
```

2) Même question en rajoutant l'implémentation de l'interface `ItemListener` pour traiter la sélection des items du sous-menu « `couleurs` » de `Elements.`

Exercice 3

On veut réaliser une boîte de dialogue présentant des champs de textes étiquetés pour que l'utilisateur entre des données. Il y a plusieurs manières de réaliser l'alignement des étiquettes et des champs (et vous pouvez aussi essayer d'implémenter celle à laquelle vous pensez). Une manière néanmoins un peu générale de procéder

serait d'utiliser un `GridBagLayout`, et d'ajouter un tableau de labels et un tableau de textFields de même longueur à un conteneur géré par un GridBagLayout. Le prototype de la procédure d'ajout des labels et des textField serait alors par exemple :

```
void  addLabelTextRows(JLabel[] labels,
                       JTextField[] textFields,
                       GridBagLayout gridbag,
                       Container container)
```

Implémentez cette méthode générique, de sorte qu'elle aligne les étiquettes du tableau labels[ ] et les champs de textes (donnés dans le tableau textFields[ ]) grâce au `GridBagLayout`.
☞ utiliser la constante `GridBagConstraints.REMAINDER` pour marquer les éléments de fin de lignes (cf. corrigé plus loin).

Eléments de correction

Exercice 3

```
private void addLabelTextRows(JLabel[] labels,
                              JTextField[] textFields,
                              GridBagLayout gridbag,
                              Container container) {
/** suppose qu'il y a le même nb de JTextField que de JLabels
et que le gridbag est le gestionnaire d'aff. associé au container */
     GridBagConstraints c = new GridBagConstraints();
     c.anchor = GridBagConstraints.EAST;
     int nitem = labels.length;     // nombre de lignes

     for (int i = 0; i < nitem; i++) {
          c.gridwidth = GridBagConstraints.RELATIVE;
                                //suit le dernier entré
          c.fill = GridBagConstraints.NONE;
                                // ne sera pas redimensionné
          c.weightx = 0.0;    // les labels ne s'élargiront pas
          gridbag.setConstraints(labels[i], c);
          container.add(labels[i]);
          c.gridwidth = GridBagConstraints.REMAINDER ;
                                //dernier de ligne
          c.fill = GridBagConstraints.HORIZONTAL;
          c.weightx = 1.0;   // absorbe tout l'espace
          container.add(textFields[i]);
     }
}
```

Exercice 2 :

// fichier SketcherFrame.java : **lignes en gras pour les evenements**

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

```
import javax.swing.*;

public class SketcherFrame extends JFrame
             implements ActionListener,ItemListener {
     protected JTextArea output;          // la zone de texte
     protected JScrollPane scrollPane;
     private JMenuBar menuBar;
     String newline = "\n";

     public SketcherFrame(String titre) {

          JMenu menu, submenu;
          JMenuItem menuItem;
          JRadioButtonMenuItem rbMenuItem;
          JCheckBoxMenuItem cbMenuItem;

          setTitle(titre);
          setDefaultCloseOperation(EXIT_ON_CLOSE);

          // Créer le texte
          output = new JTextArea(10, 80);
          output.setEditable(false);
          scrollPane = new JScrollPane(output);

          //Ajouter les composants
          Container contentPane = getContentPane();
          contentPane.add(toolBar, BorderLayout.NORTH);
          contentPane.add(scrollPane, BorderLayout.CENTER);

          //Ajouter une barre de menu
          menuBar = new JMenuBar();
          setJMenuBar(menuBar);

          //Construire le menu Fichier
          menu = new JMenu("Fichier");
          menu.setMnemonic(KeyEvent.VK_F);

          //les items de Fichier
          menuItem = new JMenuItem("Ouvrir",
                                   KeyEvent.VK_O);
          menuItem.setAccelerator(KeyStroke.getKeyStroke(
                                   KeyEvent.VK_1,
                                   ActionEvent.ALT_MASK));
          menuItem.addActionListener(this);
          menu.add(menuItem);

          menuItem = new JMenuItem
                    ("A la fois texte et icone",
                    new ImageIcon("images/middle.gif"));
          menuItem.setMnemonic(KeyEvent.VK_A);
          menuItem.addActionListener(this);
```

```
        menu.add(menuItem);

        menuItem = new JMenuItem(new
                        ImageIcon("images/middle.gif"));
        menuItem.setMnemonic(KeyEvent.VK_B);
        menuItem.addActionListener(this);
        menu.add(menuItem);

        menu.addSeparator();            // le séparateur

        menuItem = new JMenuItem("Enregistrer",
                                 KeyEvent.VK_E);
        menuItem.setAccelerator(KeyStroke.getKeyStroke(
                                  KeyEvent.VK_2,
                                  ActionEvent.ALT_MASK));
        menuItem.addActionListener(this);
        menu.add(menuItem);

        // Construire le second menu : Elements
        menu = new JMenu("Elements");
        menu.setMnemonic(KeyEvent.VK_E);
        menuBar.add(menu);

        //un groupe pour ses boutons radio
        ButtonGroup group = new ButtonGroup();

        rbMenuItem = new JRadioButtonMenuItem("Ligne",
                                 true);
        rbMenuItem.setMnemonic(KeyEvent.VK_L);
        rbMenuItem.addActionListener(this);
        group.add(rbMenuItem);
        menu.add(rbMenuItem);

        rbMenuItem = new JRadioButtonMenuItem
                                  ("Rectangle",  false);
        rbMenuItem.setMnemonic(KeyEvent.VK_R);
        rbMenuItem.addActionListener(this);
        group.add(rbMenuItem);
        menu.add(rbMenuItem);

        // … ibid pour CERCLE, etc.
        menu.addSeparator();

        // le sous-menu des cases couleurs
        submenu = new JMenu("Couleur");
        submenu.setMnemonic(KeyEvent.VK_C);
        ButtonGroup group2 = new ButtonGroup();

    cbMenuItem = new JCheckBoxMenuItem("Rouge", true) ;
        cbMenuItem.setMnemonic(KeyEvent.VK_R);
        cbMenuItem.addItemListener(this);
```

```
        group2.add(cbMenuItem);
        submenu.add(cbMenuItem);

        cbMenuItem = new JCheckBoxMenuItem("Bleu",
                                 false) ;
        cbMenuItem.setMnemonic(KeyEvent.VK_B);
        cbMenuItem.addItemListener(this);
        group2.add(cbMenuItem);
        submenu.add(cbMenuItem);
                                 // ... vert, etc.
        menu = new JMenu("Aide");
        menu.setMnemonic(KeyEvent.VK_A);
        menuBar.add(Box.createHorizontalGlue());
        menuBar.add(menu);

    } // fin du constructeur

    public void actionPerformed(ActionEvent e) {
        JMenuItem source
                  = (JMenuItem)(e.getSource());
        String s = "Detection d'ActionEvent."
                  + newline
                  + "    source de l'ev.: "
                  + source.getText()
                  + " (instance de "
                  + getClassName(source) + ")";
        output.append(s + newline);
    }

    public void itemStateChanged(ItemEvent e) {
        JMenuItem source
                  = (JMenuItem)(e.getSource());
        String s = "Detection d'ItemEvent."
                  + newline
                  + "    source de l'ev.: "
                  + source.getText()
                  + " (instance de "
                  + getClassName(source) + ")"
                  + newline
                  + "    Nouvel Etat: "
                  + ((e.getStateChange()==
                            ItemEvent.SELECTED)?
                            "selected":"unselected");
        output.append(s + newline);
    }

    protected String getClassName(Object o) {
        String classString = o.getClass().getName();
        int dotIndex = classString.lastIndexOf(".");
        return classString.substring(dotIndex+1);
    }
}
```