Université de Bouira
Département d'Informatique
Module : Les Interactions Homme Machine
Niveau 3 info

Série TP 7

Autres Composants Swing - JRadioButton, JSlider, JComboBox, JProgressBar, JList, JScrollPane, JTabbedPane, JTextArea, et JTextPane.

Etapes à suivre

I. Top-Level Containers - JFrame

- Créer un nouveau projet Java sous le nom *IHMTP7*.
- Créer dans celui-ci une classe *MyJFrame* héritant de JFrame (extends JFrame).
- Pour configurer l'état initial de la fenêtre créée, ajouter une méthode *initJFrame* et appeler celle-ci depuis un constructeur. Utiliser les méthodes suivantes pour l'initialisation :
  - ✓ setTitle(String title)
  - ✓ setSize(int width, int height)
  - ✓ setLocationRelativeTo(null)
  - ✓ setResizable(booblean resizable)
  - ✓ setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE)
- Ajouter une classe main nommée *TestJFrame*. Créer dans cette dernière une instance de la classe *MyJFrame* qui devrait s'exécuter dans l'Event Dispatch Thread (en utilisant SwingUtilities).
- Pour afficher cette instance, utiliser la méthode *setVisible*(true).

II. Les listes déroulantes

En Swing, il existe 2 sortes de listes déroulantes :

- **JList**, liste déroulante avancée qui permet de sélectionner plusieurs éléments à la fois.
- **JComboBox**, c'est une liste de choix à l'utilisateur déroulante normale, de celle que vous voyez partout dans les applications. Un seul élément est sélectionné à la fois.

JComboBox

- Créer un JPanel, *panel*, et l'appliquer comme ContentPane de la fenêtre.
- Créer un objet-composant de type JComboBox appelé *listeBox* comme suit :

```
String itemsListeBox[]={"Facebook","Twitter","Instagram","Soundcloud","Medium"};
JComboBox listeBox=new JComboBox(itemsListe);
panel.add(listeBox);
```

- Pour remplir *listeBox*, un tableau de String *itemsListeBox* est donné au constructeur du JComboBox qui va se charger de remplir la liste automatiquement avec les chaines de caractères du tableau.
- Ajouter *listeBox* à *panel* puis afficher votre fenêtre.

Pour manipuler un JComboBox, ces méthodes peuvent être utilisées :

- *addItem*(Objet item), cette méthode va ajouter un nouvel objet à la liste ;
- *removeItem*(Objet item), cette méthode va enlever l'objet de la liste ;
- *removeAllItems*(), cette méthode va vider la liste déroulante ;
- *getSelectedItem*(), cette méthode retourne l'objet qui est actuellement sélectionné.

Pour réagir aux événements (élément sélectionné) sur les JComboBox (ItemEvent), des écouteurs de type **ItemListener** peuvent être ajoutés.

- Ajouter à *panel* un composant JLabel nommé *displayItem*.
- Ajouter à *listeBox* un écouteur de type ItemListener en utilisant la méthode *addItemListener*. Utiliser une classe interne anonyme pour définir la méthode *itemStateChange*(ItemEvent).

```
JLabel displayItem = new JLabel("Select your choice");
panel.add(displayItem);

listeBox.addItemListener(new ItemListener() {
           @Override
           public void itemStateChanged(ItemEvent e) {

           }

});
```

- Modifier *itemStateChange* pour que le JLabel *displayItem* affiche à chaque fois l'élément/item de *listeBox* sélectionné par l'utilisateur comme suit :

```
public void itemStateChanged(ItemEvent e) {
     if (e.getStateChange() == ItemEvent.SELECTED) {
            displayItem.setText(e.getItem().toString());
     }
}
```

- Afficher la nouvelle fenêtre et tester.

JList

- Créer un objet-composant de type **JList** appelé *listeFonts.*
- Comme pour le JComboBox, pour remplir *listeFonts*, créer un tableau de String *itemsListeFonts* et le passer en argument au constructeur du JList qui va se charger de remplir la liste automatiquement avec les chaines de caractères du tableau.
- Pour cet exemple, *listeFonts* contiendra la liste de tous les fonts disponibles dans le système :

```
GraphicsEnvironment ge =
               GraphicsEnvironment.getLocalGraphicsEnvironment();
String[] itemsListeFonts = ge.getAvailableFontFamilyNames();

JList <String> listeFonts = new JList<String>(itemListeFonts);
```

- JList peut avoir plusieurs éléments, physiquement impossible à afficher sur la fenêtre. Un composant **JScrollPane** peut être utilisé pour la faire défiler :

```
JScrollPane spane = new JScrollPane(listeFonts);
spane.setPreferredSize(new Dimension(250, 180));
panel.add(spane);
```

Pour réagir aux événements sur les JList (éléments sélectionnés par l'utilisateur - ListSelectionEvent), des écouteurs de type **ListSelectionListener** peuvent être ajoutés.

- Ajouter à *panel* un composant JLabel nommé *displayFonts*.
- Ajouter à *listeFonts* un écouteur de type ListSelectionListener en utilisant la méthode *addListSelectionListener*. Utiliser une classe interne anonyme pour définir la méthode *valueChanged*(ListSelectionEvent).

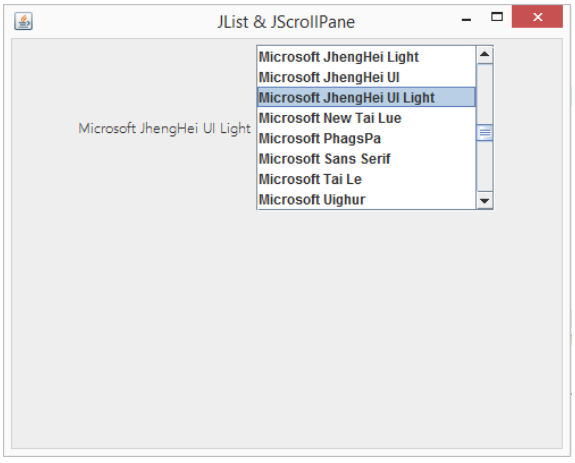
```
JLabel displayFonts = new JLabel("Fonts");
panel.add(displayFonts);

listeFonts.addListSelectionListener(new ListSelectionListener() {
            @Override
            public void valueChanged (ListSelectionEvent e) {
            }

});
```

- Modifier *valueChanged* pour que *displayFonts* affiche l'élément de *listeFonts* sélectionné par l'utilisateur avec le font choisi.

```
public void valueChanged(ListSelectionEvent e) {
     if (!e.getValueIsAdjusting()) {
            String name = (String) listFonts.getSelectedValue();
            displayFonts.setText(name) ;
            Font font = new Font(name, Font.PLAIN, 12);
            displayFonts.setFont(font);
      }
}
```

Les événements dans ListSelection sont groupés. Ils sont reçus en cas de sélection et de désélection d'un élément. Pour filtrer ceux issus de la sélection seulement, la méthode *getValueIsAdjusting* a était utilisée.

Résultat JList:



III. JRadioButton

Un objet de type **JRadioButton** représente un bouton radio d'un groupe de boutons. A un instant donné, un seul des boutons radio associés à un même groupe peut être sélectionné. [1]
Un bouton radio possède un libellé et éventuellement une icône qui peut être précisée, pour chacun des états du bouton, en utilisant les méthodes *setIcon*(), *setSelectedIcon*() et *setPressedIcon*().
La méthode isSelected() permet de savoir si le bouton est sélectionné ou non.

Un groupe de boutons radio est encapsulé dans un objet de type **ButtonGroup**. Il faut ajouter tous les JRadioButton du groupe en utilisant la méthode *add*() de la classe ButtonGroup. Lors de la sélection d'un bouton, c'est l'objet de type ButtonGroup qui se charge de déselectionner le bouton précédemment sélectionné dans le groupe. Un groupe n'a pas l'obligation d'avoir un bouton sélectionné.
Lors de la sélection d'un bouton du groupe, il y a plusieurs événements qui peuvent être émis :

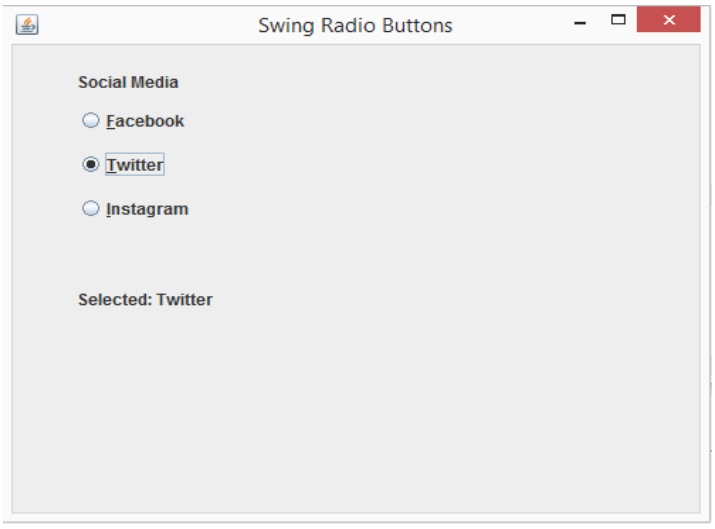
- Un événement de type ActionEvent ;
- Un événement de type ItemEvent.SELECTED émis par le bouton sélectionné ;
- Un événement de type ItemEvent.DESELCTED émis par le bouton désélectionné.

Exemple : Un JRadioButton

```
ButtonGroup group = new ButtonGroup( );
JRadioButton rb1 = new JRadioButton("Exemple");
group.add(rb1);
rb1.addItemListener(this);
panel.add(rb1);
```

Exercice : Créer et afficher la fenêtre ci-dessous. (Utiliser *setMnemonic(KeyEvent.VK _Val)* pour les raccourci clavier et BoxLayout comme layout manager)

```
//Exemple setMnemonic pour le deuxième JRadioButton Twitter
//Pour le sélectionner : Alt + T
rb2.setMnemonic(KeyEvent.VK_T);
```



[1] https://www.jmdoudoux.fr/java/dej/chap-swing.htm

Code :

```
JPanel panel = new JPanel();
panel.setLayout(new BoxLayout(panel, BoxLayout.Y_AXIS));

JLabel lbl = new JLabel("Social Media");
panel.add(Box.createRigidArea(new Dimension(20, 20)));
panel.add(lbl);

ButtonGroup group = new ButtonGroup();
JRadioButton rb1 = new JRadioButton("Facebook", true);
rb1.setMnemonic(KeyEvent.VK_F);
JRadioButton rb2 = new JRadioButton("Twitter");
rb2.setMnemonic(KeyEvent.VK_T);
JRadioButton rb3 = new JRadioButton("Instagram");
rb3.setMnemonic(KeyEvent.VK_I);

group.add(rb1);
group.add(rb2);
group.add(rb3);

rb1.addItemListener(this);
rb2.addItemListener(this);
rb3.addItemListener(this);

panel.add(Box.createRigidArea(new Dimension(20, 10)));
panel.add(rb1);
panel.add(Box.createRigidArea(new Dimension(20, 10)));
panel.add(rb2);
panel.add(Box.createRigidArea(new Dimension(20, 10)));
panel.add(rb3);
panel.add(Box.createRigidArea(new Dimension(20, 50)));

JLabel sbar = new JLabel("Selected: Facebook");
panel.add(sbar);

this.setContentPane(panel);
      ____________________________________________________

@Override
public void itemStateChanged(ItemEvent e) {
     if ((e.getStateChange() == ItemEvent.SELECTED)) {
         JRadioButton button = (JRadioButton) e.getSource();
         String text = button.getText();
         sbar.setText("Selected : " + text);
     }

}
```