C. Recanati
Java Graphique
AIR2

Feuille de TP 6

Exercice 1

On va créer une applet qui présente une liste scrollée dans sa partie centrale et, au dessus, deux boutons alignés horizontalement dans un panneau qui permettront respectivement :

1. de détruire les items sélectionnés dans la liste
2. d'ajouter un item à la liste.

1) Compléter pour cela le squelette de programme suivant, en réalisant tout d'abord simplement l'affichage.

```
public class Test extends JApplet {
        private JList list = new JList();

        String[] items = { "item[0]", "item[1]", "item[2]", "item[3]",
                           "item[4]", "item[5]", "item[6]", "item[7]",
                           "item[8]", "item[9]" };

        public void init() {
                Container contentPane = getContentPane();
                ControlPanel controlPanel = new ControlPanel(list); //+ loin
                contentPane.add(controlPanel, BorderLayout.NORTH);
                contentPane.add(new JScrollPane(list),
                                        BorderLayout.CENTER);
                populateList();
        }
        public void populateList() {
        // construire un modèle à partir de DefaultListModel
        // l'initialiser avec le tableau d'items et l'affecter à la liste privée
        //  ...
        }
}
class ControlPanel extends JPanel {
        private JButton remove = new JButton("Supprimer la selection");
        private JButton add = new JButton("Ajouter un item");
        private JList list ;

        public ControlPanel(JList ls) {
                this.list = ls ;
```

```
            add(remove);
            add(add);

            remove.addActionListener(new ActionListener() {
                public void actionPerformed(ActionEvent e) {
                // récupérer les indices sélectionnés de la liste
                // et récupérer le modèle
                // ...
            // puis supprimer les items correspondants du modèle

                // ... on pourra présenter d'abord une boîte
                // demandant à l'utilisateur s'il souhaite vraiment
                // supprimer la sélection (utiliser une méthode de la
                // classe JOptionPane pour créer une boîte
                // de dialogue prête à l'emploi)
                    }
                }
            });
            add.addActionListener(new ActionListener() {
                public void actionPerformed(ActionEvent e) {
                    // récupérer le modèle de la liste

                    // récupérer la chaîne (l'item) à rajouter dans
                    // une boîte de dialogue modale prête à l'emploi
                    // fournie par la classe JOptionPane

                    // ajouter cette chaîne au modèle

                    SwingUtilities.invokeLater(new Runnable() {
                        public void run() {
                            list.ensureIndexIsVisible(
                                    model.getSize()-1);
                        }
                    });
                }
            });
        }
}
```

Remarque : avec `invokeLater`, la procédure `ensureIndexIsVisible` n'est effectuée qu'après que les événements de mise à jour aient été traités.

2) Encadrer joliment la liste scrollée : on placera le `JScrollPane` contenant la liste dans un `JPanel` muni d'un bord d'une certaine épaisseur et encadré. (Utiliser `createEmptyBorder` de `BorderFactory` pour avoir un bord vide, `new LineBorder(Color, int)` pour le cadre noir, et rajouter un titre comme « Liste des éléments » en affectant au `JPanel` un bord composé à partir du bord vide et du bord noir titré créé avec la méthode `createCompoundBorder` de `BorderFactory`).

3) On rajoute au panneau de contrôle du haut une `JComboBox` présentant le mode de sélection utilisée par la liste, soit : `SINGLE_SELECTION, SINGLE_INTERVAL_SELECTION,` ou `MULTIPLE_INTERVAL_SELECTION`. On écrira une procédure privée `initializeSelectionMode`() qui initialisera l'item sélectionné de la comboBox conformément au mode de sélection initial de la liste, et on ajoutera un `ItemListener` à la comboBox pour mettre à jour le mode de sélection de la liste quand un nouveau mode sera sélectionné sur la comboBox.

4) Mettre en place les réactions sur les boutons add et remove du panneau de contrôle conformément aux commentaires du squelette de programme (utiliser des boîtes de dialogue).

5) Modifier encore le panneau de contrôle `ControlPanel` pour qu'il affiche sous les boutons précédents:

1. les indices des items sélectionnés dans la liste (sous une forme comme "Index des items sélectionnés : 3, 4, 5, 9 ") ;
2. au dessous les index minimum et maximum des items sélectionnés.

Pour la mise à jour de ces labels, on écrira une méthode `updateLabel`() pour le panneau de contrôle qui sera appelée quand la sélection de la liste aura changée (utiliser un `ListSelectionListener` sur la liste).
Pour le placement des composants dans le `ControlPanel`, on pourra utiliser un `BoxLayout` vertical. En haut, un `JPanel` servira à placer les boutons et la comboBox comme antérieurement, et au milieu et en bas on placera des labels adéquats.

Eléments de correction

```
public class Test2 extends JApplet {
        private JList list = new JList();

        String[] items = { "item[0]", "item[1]", "item[2]", "item[3]",
"item[4]", "item[5]","item[6]", "item[7]", "item[8]", "item[9]" };
```

```
        public void init() {
                Container contentPane = getContentPane();
                ControlPanel controlPanel = new ControlPanel(list);
                contentPane.add(controlPanel, BorderLayout.NORTH);
                contentPane.add(new JScrollPane(list),
                                                BorderLayout.CENTER);
                populateList();
                controlPanel.update() ;

        }
        public void populateList() {
                DefaultListModel model = new DefaultListModel();
                for(int i=0; i < items.length; ++i)
                        model.addElement(items[i]);
                list.setModel(model);
        }
}

class ControlPanel extends JPanel {
        JButton remove = new JButton("supprimer la selection");
        JButton add = new JButton("ajouter un item");
        private JComboBox mode = new JComboBox();
        private JList list;

        private String single = "SINGLE_SELECTION",
                singleInterval = "SINGLE_INTERVAL_SELECTION",
                multipleInterval = "MULTIPLE_INTERVAL_SELECTION";

        private JLabel minLabel = new JLabel(),
                        maxLabel = new JLabel(),
                        selIndicesLabel = new JLabel();

        public ControlPanel(JList   list) {
                JPanel top = new JPanel(),  mid = new JPanel(),
                        bottom = new JPanel();
                this.list = list;

                setLayout(new BoxLayout(this, BoxLayout.Y_AXIS));
                setBorder(BorderFactory.createEtchedBorder());

                top.add(remove);
                top.add(add);
```

```
top.add(mode);

mode.addItem(single);
mode.addItem(singleInterval);
mode.addItem(multipleInterval);
initializeSelectionMode();

mid.add(new JLabel("Minimum:"));  mid.add(minLabel);
mid.add(new JLabel("Maximum:"));  mid.add(maxLabel);

add(top);
add(mid);
add(bottom);

bottom.add(new JLabel("Indices selectionnes:"));
bottom.add(selIndicesLabel);

remove.addActionListener(new ActionListener() {
        public void actionPerformed(ActionEvent e) {
                int[] selected = list.getSelectedIndices();
                DefaultListModel model =
                                (DefaultListModel)list.getModel();

                for(int i=0; i < selected.length; ++i) {
                        model.removeElementAt(selected[i] - i);
                }
        }
});
add.addActionListener(new ActionListener() {
        public void actionPerformed(ActionEvent e) {
                final DefaultListModel model =
                        (DefaultListModel)list.getModel();

                String s = JOptionPane.showInputDialog(
                                        list,
                                        "Entrer un item : ");
                model.addElement(s);

                SwingUtilities.invokeLater(new Runnable() {
                        public void run() {
                                list.ensureIndexIsVisible(
                                        model.getSize()-1);
                        }
```

```
                });
            }
        });
        mode.addItemListener(new ItemListener() {
            public void itemStateChanged(ItemEvent e) {
                if (e.getStateChange() == ItemEvent.SELECTED)
                    setSelectionMode((String)e.getItem());
            }
        });
        list.addListSelectionListener( new ListSelectionListener() {
            public void valueChanged(ListSelectionEvent e) {
                update();
            }
        });
}
public void update() {
        int     min = list.getMinSelectionIndex(),
                max = list.getMaxSelectionIndex() ;

        minLabel.setText(Integer.toString(min) + " / ");
        maxLabel.setText(Integer.toString(max));

        int[] selected = list.getSelectedIndices();
        String s = new String();

        for(int i = 0; i < selected.length; ++i) {
            s += Integer.toString(selected[i]);

            if(i < selected.length-1)
                s += ",";
        }
        selIndicesLabel.setText(s);
        validate();
}

private void initializeSelectionMode() {
        int m = list.getSelectionMode();

        switch(m) {
            case ListSelectionModel.SINGLE_SELECTION:
                mode.setSelectedItem(single);
                break;
```

```
                    case
        ListSelectionModel.SINGLE_INTERVAL_SELECTION:
                            mode.setSelectedItem(singleInterval); break;
                    case
        ListSelectionModel.MULTIPLE_INTERVAL_SELECTION:
                            mode.setSelectedItem(multipleInterval);
                            break;
                }
        }
        private void setSelectionMode(String s) {
                if(s.equals("SINGLE_SELECTION")) {
                   System.out.println("single selected");
                        list.setSelectionMode(
                                ListSelectionModel.SINGLE_SELECTION);
                }
                else if(s.equals("SINGLE_INTERVAL_SELECTION")) {
                        System.out.println("single interval selected");
                        list.setSelectionMode(
                     ListSelectionModel.SINGLE_INTERVAL_SELECTION);
                }
                else if(s.equals("MULTIPLE_INTERVAL_SELECTION")) {
                        System.out.println("multiple interval selected");
                        list.setSelectionMode(
                ListSelectionModel.MULTIPLE_INTERVAL_SELECTION);
                }
        }
}
//------ pour ajouter des bords
        public void init() {
        …
        JScrollPane titleListScroll  = new JScrollPane(list) ;
        titleListScroll.setBorder(
                 BorderFactory.createCompoundBorder(
                        BorderFactory.createEmptyBorder(10, 10, 10, 10)),
        BorderFactory.createTitledBorder(BorderFactory.createLineBorder
                            (Color.BLACK,1), "liste des elements")));
                …
                contentPane.add(titleListScroll,
                                            BorderLayout.CENTER);
                populateList();
                controlPanel.update() ;
        }
```