COURS – TD 7: ARBRES

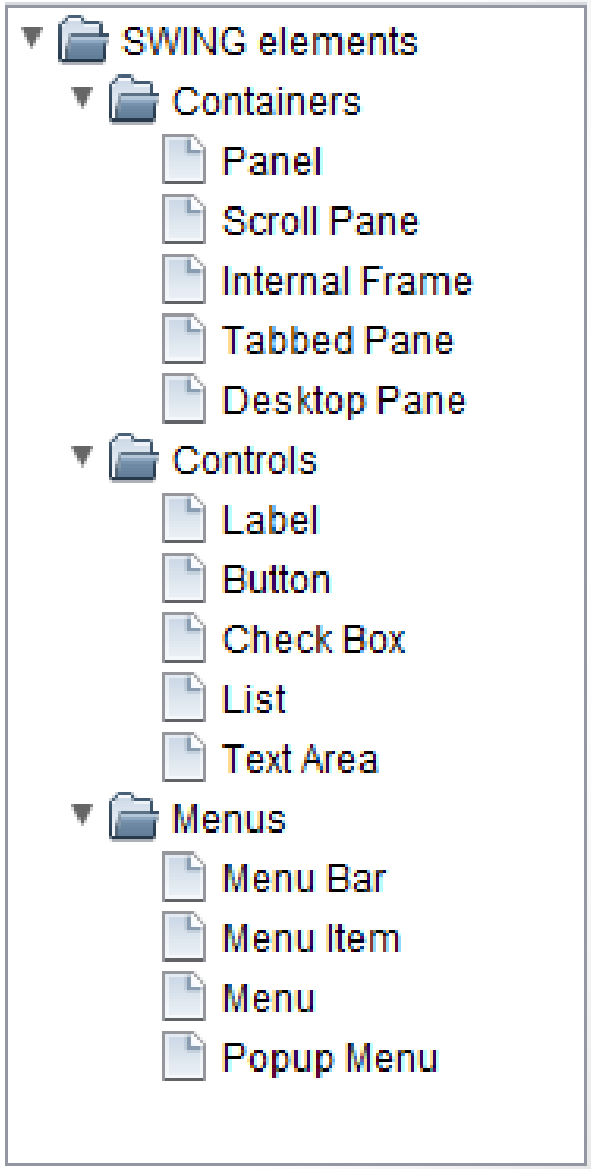
Arbre: utilité et composition

- Visualisation d'informations
- Hiérarchie

- Composition d'un arbre
  - Racine
  - Branches
  - Feuilles

SWING - JTree

- Racine - Root
- Noeud - Node
  - Branche – Branch node
  - Feuille – Leaf node



Création simple d'un JTree

- **Arbre - JTree**

Constructeur

JTree public JTree(TreeNode root)

- **Noeuds/Données - DefaultMutableTreeNode**

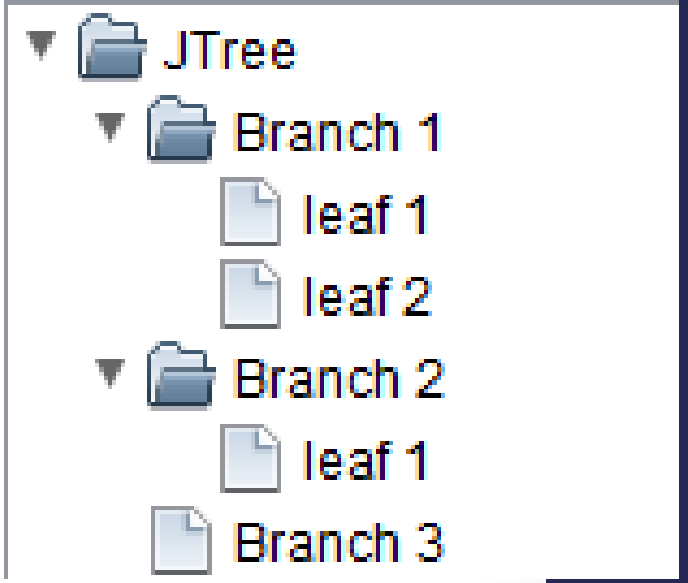
Constructeur

public DefaultMutableTreeNode(Object userObject)

Ajout d'un sous-noeud

public void add(MutableTreeNode newChild)

Création simple d'un JTree - exemple



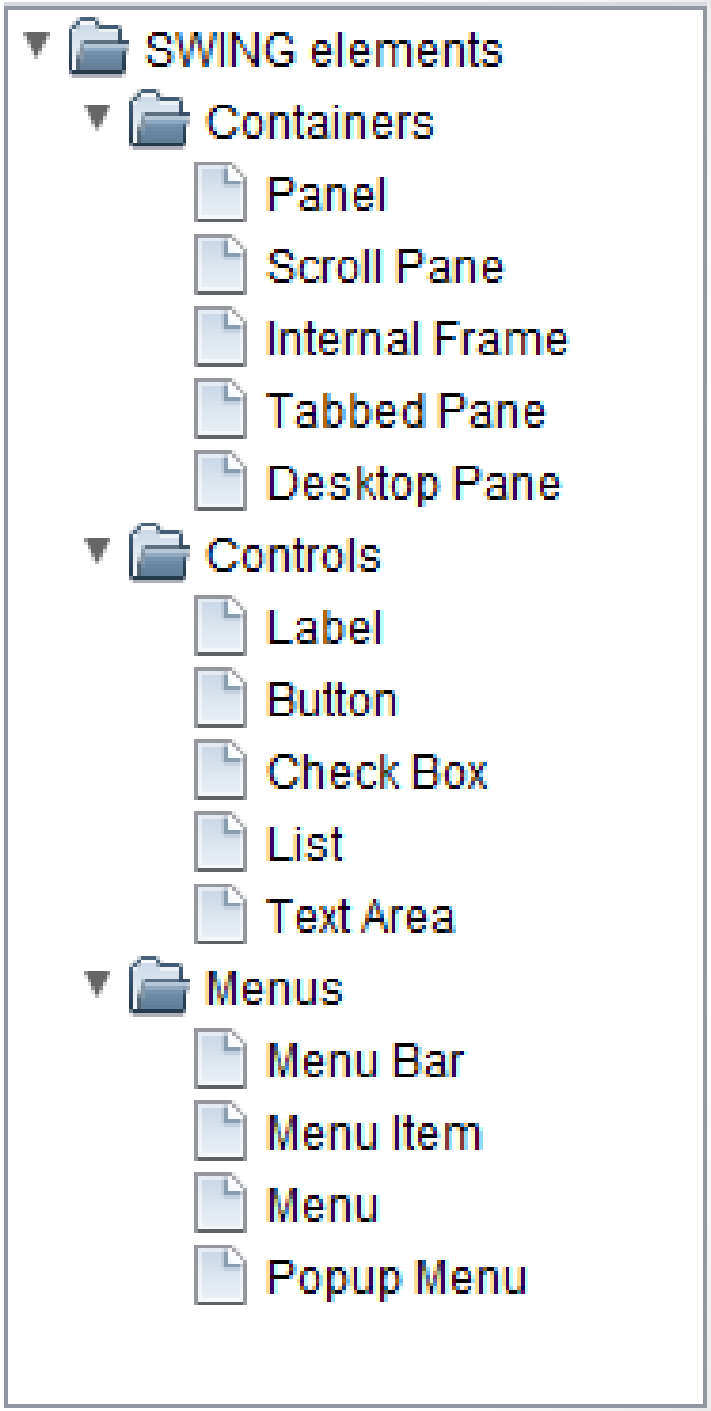

```
DefaultMutableTreeNode treeNode1 = null;
DefaultMutableTreeNode treeNode2 = null;
DefaultMutableTreeNode  treeNode3 = null;

treeNode1 = new DefaultMutableTreeNode("JTree");
treeNode2 = new DefaultMutableTreeNode("Branch 1");
treeNode3 = new DefaultMutableTreeNode("leaf 1");

//Branche “Branch 1”
treeNode2.add(treeNode3);
treeNode3 = new DefaultMutableTreeNode("leaf 2");
treeNode2.add(treeNode3);
treeNode1.add(treeNode2);
…
JTree jtree = new JTree(treeNode1);
```

Exercice 1

- Ecrire la portion de code permettant de créer l'arbre ci-contre



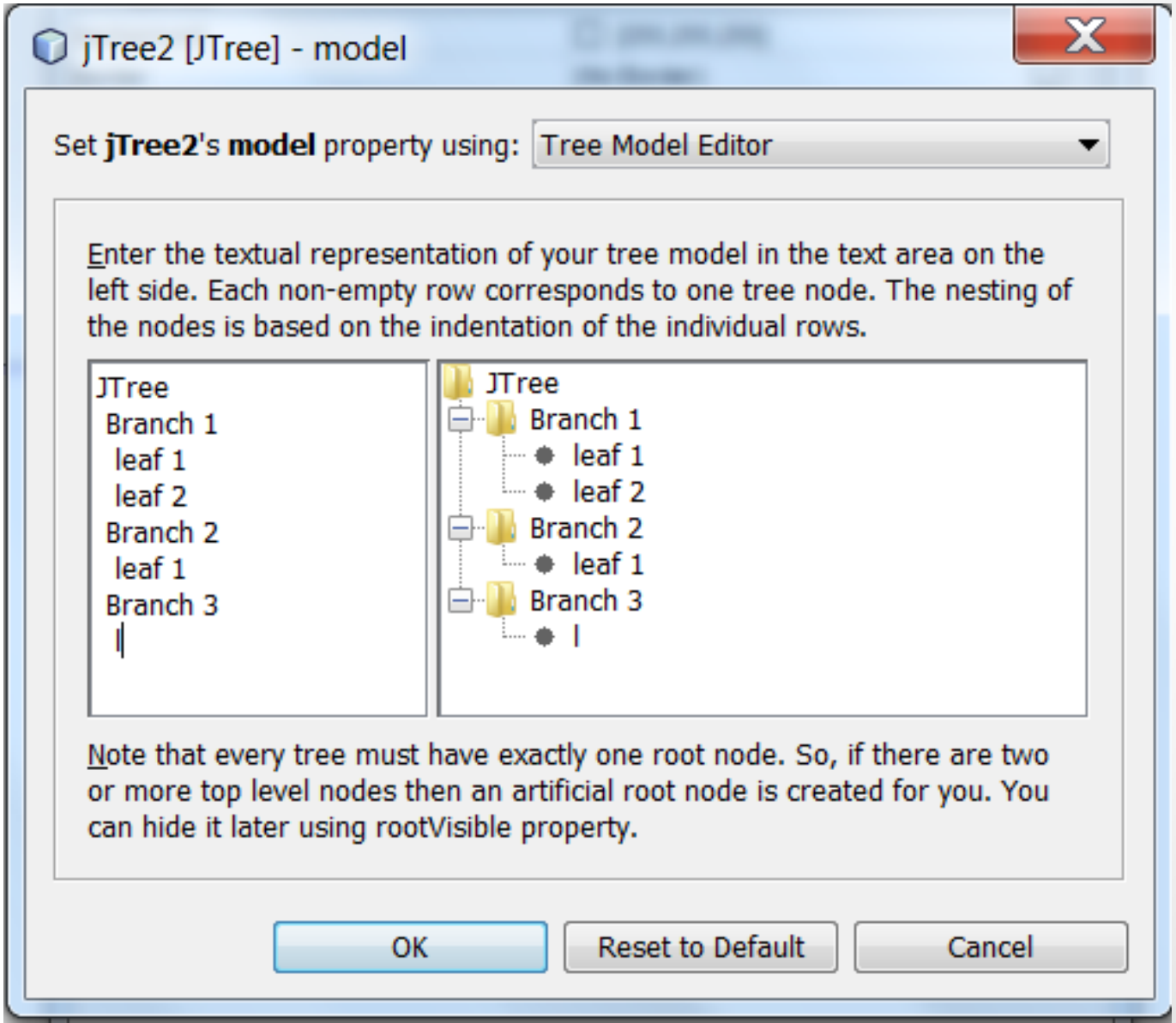
Container - remarque

- JScrollPane conseillé
  - Nombre de données à visualiser vs surface disponible
  - Même problématique que la Jtable
  - Avec Netbeans: automatiquement ajouté lors d'un glisser-déposer d'un JTree dans la vue de conception

Avec Netbeans

- Dans la fenêtre de propriétés du Jtree

Editeur de données et code généré automatique-ment



Personnalisation du rendu (simple)

- Affichage de l'élément racine

public void **setRootVisible(boolean rootVisible)**

- Affichage des "poignées de dépliage/repliage"

public void **setShowsRootHandles(boolean newValue)**

Evènements (sémantiques)

- Sélection d'un noeud
  - TreeSelectionEvent
  - TreeSelectionListener

    void valueChanged(TreeSelectionEvent e)
- Dépliage/repliage d'un noeud
  - TreeExpansionEvent
  - TreeExpansionListener
    - void treeExpanded(TreeExpansionEvent event)
    - void treeCollapsed(TreeExpansionEvent event)

Informations sur la sélection courante (un seul noeud)

- Par la classe JTree: dernier noeud sélectionné

  public Object getLastSelectedPathComponent()

- Par le biais de l'évènement reçu (TreeSelectionEvent)

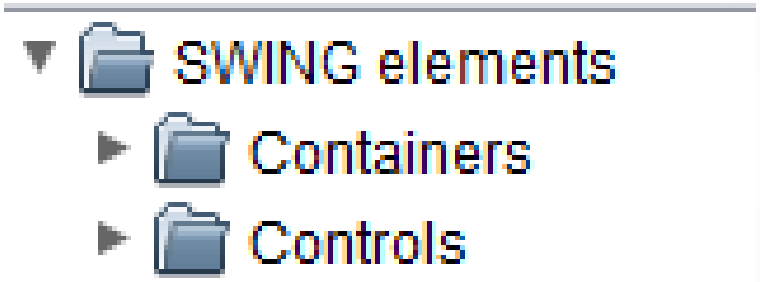
  public **TreePath** getPath()

  Classe TreePath:

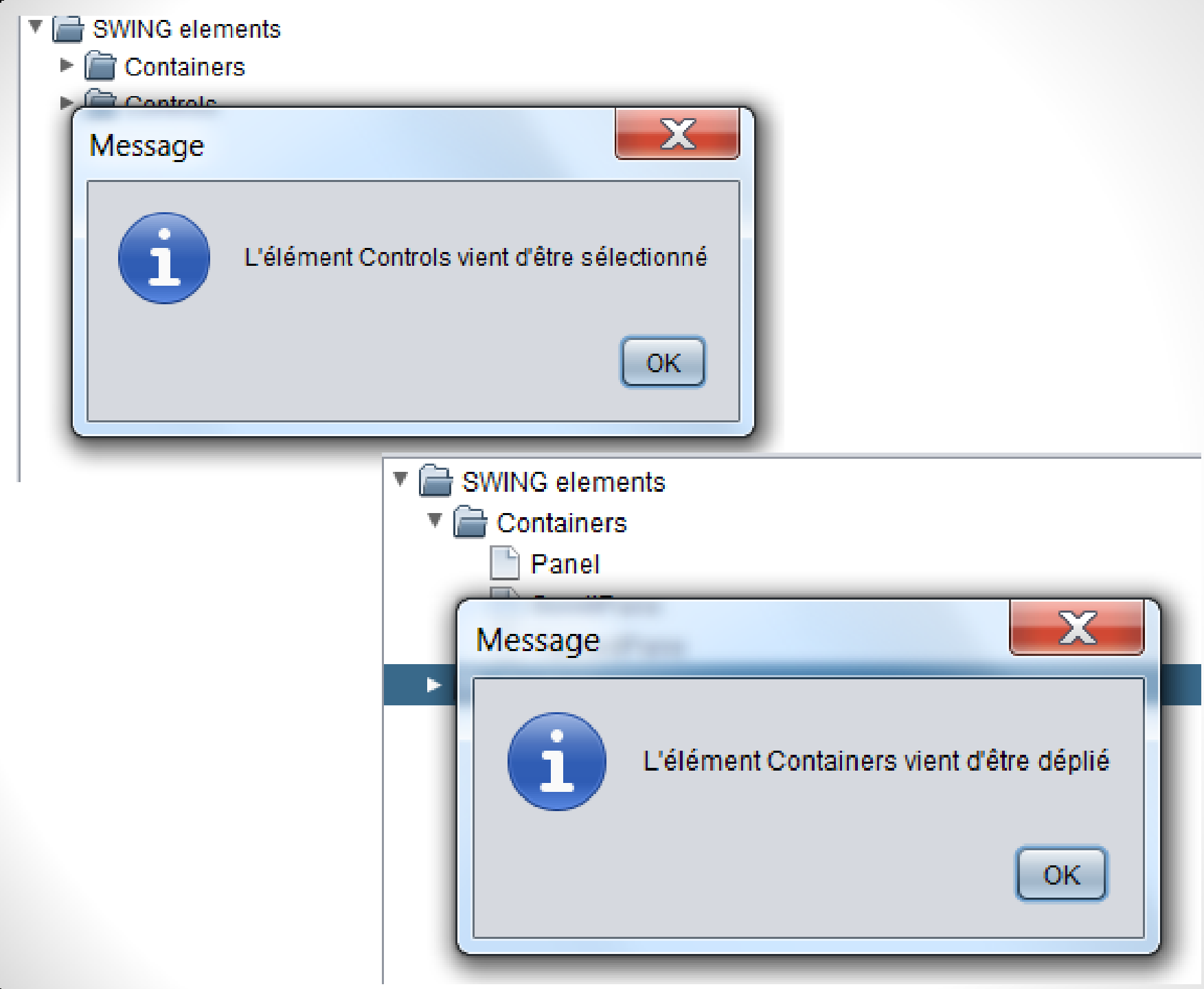
  public Object getLastPathComponent()

Exercice 2

- Ecrire la portion de code permettant d'associer un listener au JTree



ainsi que l'event handler déclenchant les affichages suivants:


SWING elements
Containers
Controls
Message
L'élément Controls vient d'être sélectionné
OK
SWING elements
Containers
Panel
Message
L'élément Containers vient d'être déplié
OK

Sélection (suite)

- **Mode de sélection**

  CONTIGUOUS_TREE_SELECTION (ensemble continu)

  DISCONTIGUOUS_TREE_SELECTION (ensemble discontinu)

  SINGLE_TREE_SELECTION (un seul noeud, par défaut)

- **Mise en oeuvre**

Classe JTree:

public void setSelectionModel(TreeSelectionModel selectionModel)

public TreeSelectionModel getSelectionModel()

Classe TreeSelectionModel:

public void setSelectionMode (int mode)

Informations sur la sélection (plusieurs noeuds)

APIs JTree

- Liste des éléments sélectionnés

public TreePath[] getSelectionPaths()

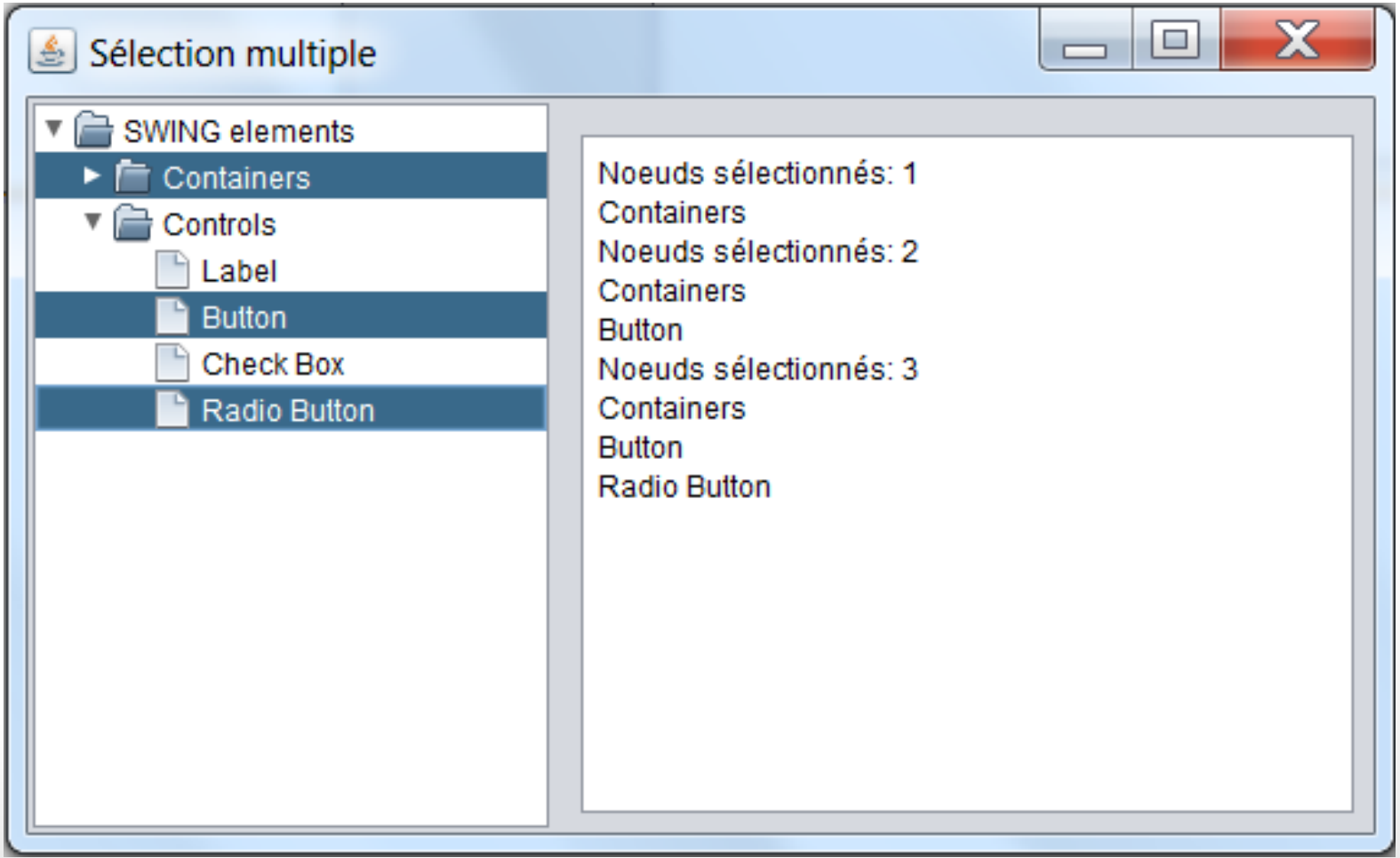
- Index des éléments sélectionnés

public int getSelectionCount()

public int[] getSelectionRows()

Exercice 3

- Ecrire la portion de code de l'event handler d'une sélection multiple permettant d'afficher la sélection courante dans une zone de texte:



Gestion avancée d'un JTree

- Model - Modèle de données
- Rendu visuel - Renderer

Création d'un JTree associé à un modèle de données

- Constructeur JTree

  JTree(TreeModel newModel)

- DefaultTreeModel (associé par défaut à une instance de classe JTree)

  DefaultTreeModel(TreeNode root)

Création d'un JTree associé à un modèle de données (2)

- Implémentation de l'interface TreeModel
  - Si DefaultTreeModel ne correspond pas aux besoins de structuration hiérarchique
  - Exemple: 2 types de hiérarchie à visualiser pour une même structure de données

Personnalisation du rendu

- Reconfiguration du renderer associé par défaut
  - DefaultTreeCellRenderer

    public DefaultTreeCellRenderer()

  ou
- Extension de la classe DefaultTreeCellRenderer

  ou
- Implémentation de l'interface de rendu (et extension d'un JComponent – cf cours JTable)
  - TreeCellRenderer

Association d'un renderer à un JTree

public void
    setCellRenderer(TreeCellRenderer x)

DefaultTreeCellRenderer

- Icône associée aux feuilles

  void setLeafIcon(Icon newIcon)

- Icône associée noeuds repliés

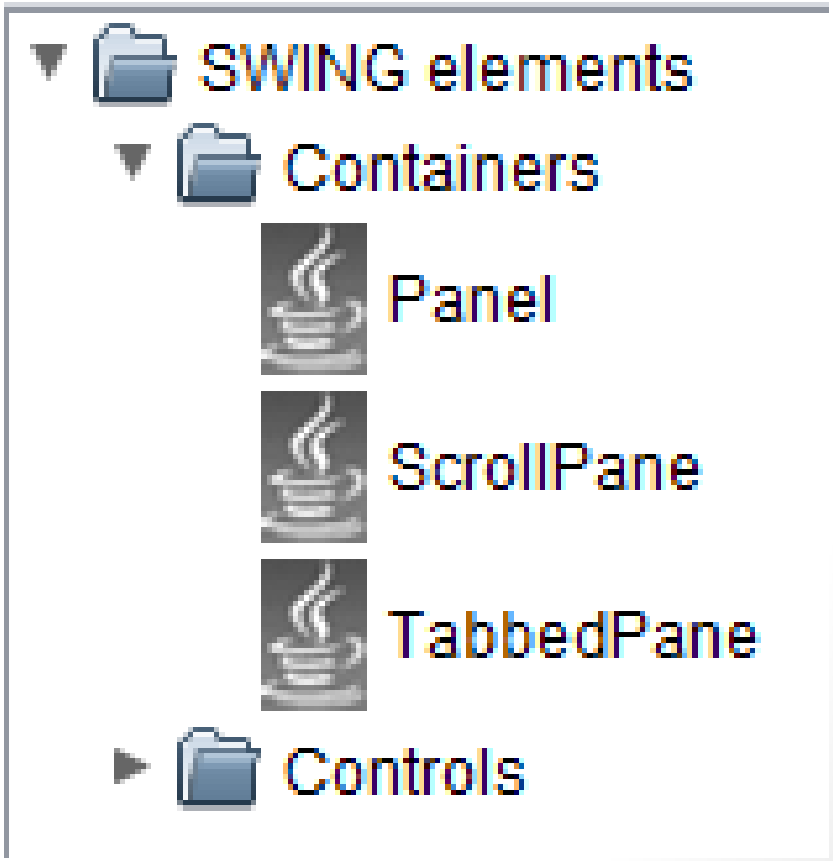
  void setClosedIcon(Icon newIcon)

- Icône associée aux noeuds dépliés

  void setOpenIcon(Icon newIcon)

Exercice 4

- Ecrire la portion de code permettant d'associer une nouvelle icône aux feuilles d'un arbre, tel qu'exposé ci-après:



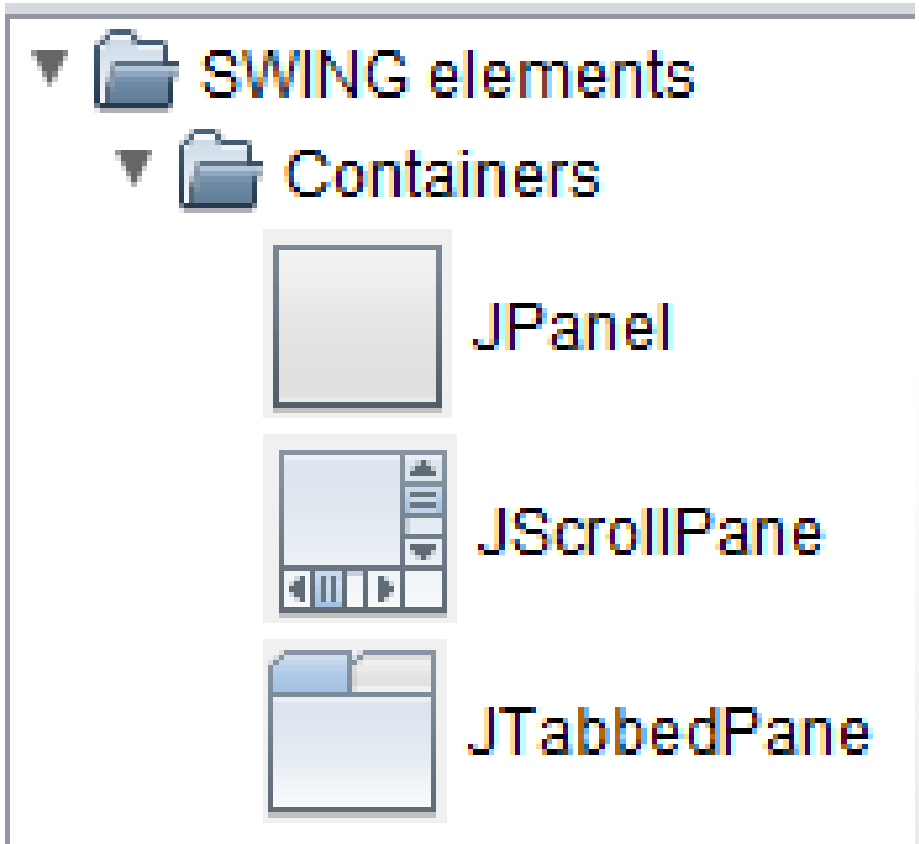
Mise en oeuvre d'une classe de rendu

- Extension de la classe DefaultTreeCellRenderer
- Surcharge de la méthode

```
public Component
        getTreeCellRendererComponent(JTree tree,
                Object value,
                boolean sel,
                boolean expanded,
                boolean leaf,
                int row,
                boolean hasFocus)
```

Exercice 5

- Programmer un renderer permettant de personnaliser l'icône d'une feuille en fonction de son type



Autres possibilités: exemples

- Réorganisation des éléments de l'arbre en glissant/déposant les éléments d'un endroit à un autre
  - Gestion des évènements souris
- Edition des éléments
  - Arbre non éditable par défaut
  - DefaultTreeCellEditor, TreeCellEditor