ÉLÉMENTS DE CORRECTION POUR LE PROJET CATALOGUE

Les explications sont basées sur le projet *demo_jlist_pour_catalogue*.

1. Rappel sur l'architecture MVC (Modèle - Vue - Contrôleur)

L'architecture MVC permet de structurer une application interactive en 3 partie :

- le modèle qui gère les données ;
- la vue qui présente les informations à l'utilisateur ;
- le contrôleur qui gère le comportement de l'application.

Avec Swing, les composants graphiques implémentent la vue et le contrôleur il reste à coder le modèle et à associer le modèle au composant Swing.

2. Utilisation d'une JList

L'objectif est de gérer une liste de snippets (un snippet est un morceau de code) à l'aide du composant graphique JList

La méthode de travail à suivre est la suivante :

1) Création de la classe *Snippet* pour représenter le concept de snippet. Un snippet est caractérisé par un titre, un contenu et 5 tags

2) Création de la classe *ModeleListeSnippets* destinée à être associée à la *JList.* Cette classe étend la classe *AbstractListModel* (qui elle-même implémente l'interface *ListModel*).

AbstractListModel est une classe générique, on précise donc que ce sont des objets de type *Snippet* qui sont gérés dans le modèle.

3) Création de le classe *Main* : elle est chargée de démarrer le programme : elle construit et lance la fenêtre principale.

4) Création de la classe *FenetrePrincipale qui hérite de Jframe.*
On supprime la méthode *main* (qui a été générée automatiquement).

Il faut associer le modèle *ModeleListeSnippets* et le composant *Jlist.* Pour ce faire :

- clic droit sur la JList puis item "Customize code" ;
- repérer le code générer où la méthode setModel est utilisée et mettre son propre code.

3. Gestion du double clic sur un item de la liste

En mode *design*, clic-droit sur la *JList* puis Events > Mouse > mouseClicked.
Ajouter le code (par exemple) :

```
JList liste = (JList)evt.getSource();
if (evt.getClickCount() == 2) {
    int index = liste.locationToIndex(evt.getPoint());
```

```
this.jTextField1.setText(this.modeleListeSnippets.getElementAt(index).toString()
);
        }
```

4. Ajout d'une icône

1) Création d'un paquetage nommé *images* destiné à recevoir les images de l'application.

2) Ajout du fichier contenant l'icône dans le répertoire "images".

3) Ajout dans le constructeur de la fenêtre principale du code suivant :

```
        // mise en place de l'icône de l'application
        Toolkit tk = Toolkit.getDefaultToolkit();
        Image im = tk.getImage(getClass().getResource("/images/icone.png"));
        ImageIcon icon = new ImageIcon(im);
        setIconImage(icon.getImage());
```

5. Gestion de la sélection

Plusieurs items peuvent être sélectionner dans une *JList*.

Il faut utiliser la méthode *getSelectedValuesList* pour récupérer une liste des items sélectionnés comme on peut le voir sur l'exemple suivant :

```
List<Snippet> liste = this.jList1.getSelectedValuesList();
        for(Snippet s : liste)
            this.jTextField1.setText(this.jTextField1.getText() + " " +
s.toString());
```