

# JList

## Qu'est-ce qu'un JList ?

JList est un composant qui **affiche un ensemble d'éléments** sous forme de liste et permet à l'utilisateur de **sélectionner un ou plusieurs** éléments.

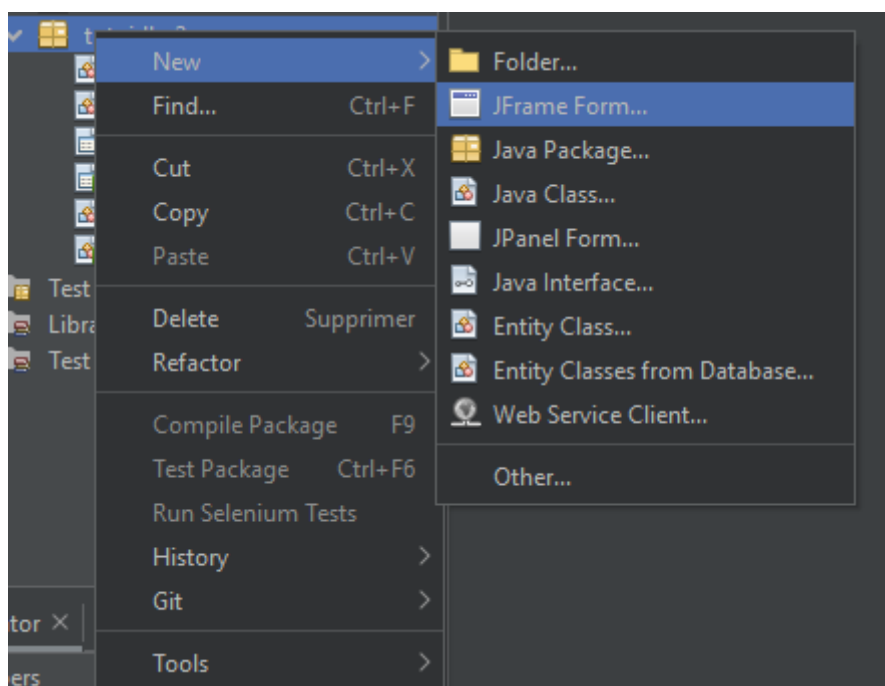
Ces éléments peuvent être de plusieurs natures, selon le besoin :

- Des valeurs de type **String** "en dur" ;
- Des valeurs de type String **dynamiquement** alimentées par programme ;
- Des valeurs de type **Object** appartenant tous à la même classe .

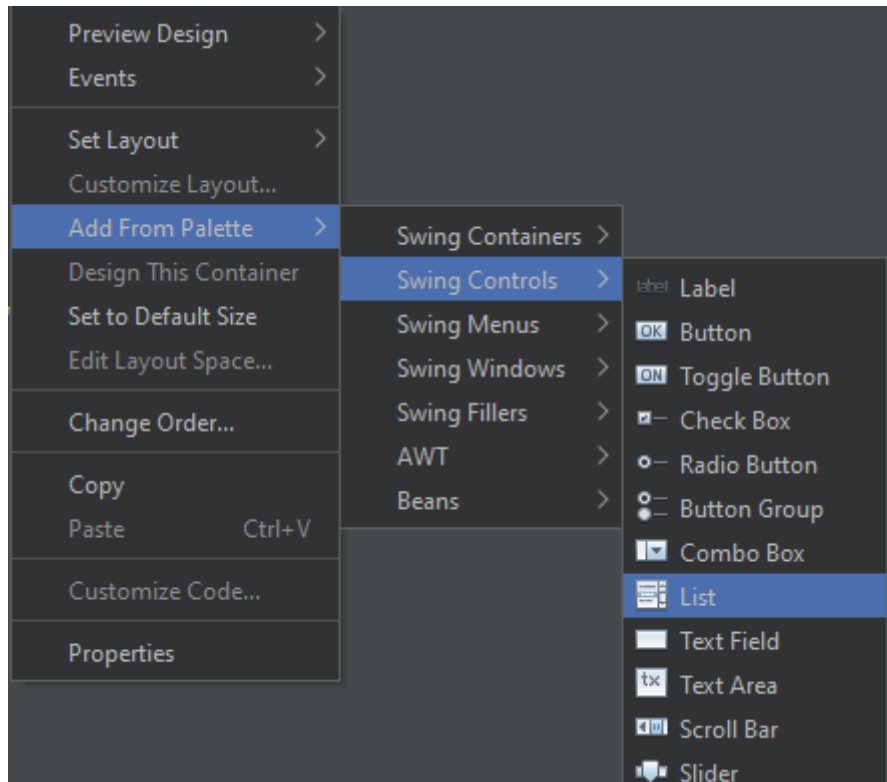
## Déposer un JList

Pour créer une Jlist il faut créer en premier lieu une fenêtre. ( **JFrame** par exemple).

### Créer une JFrame Form



### Créer une Jlist

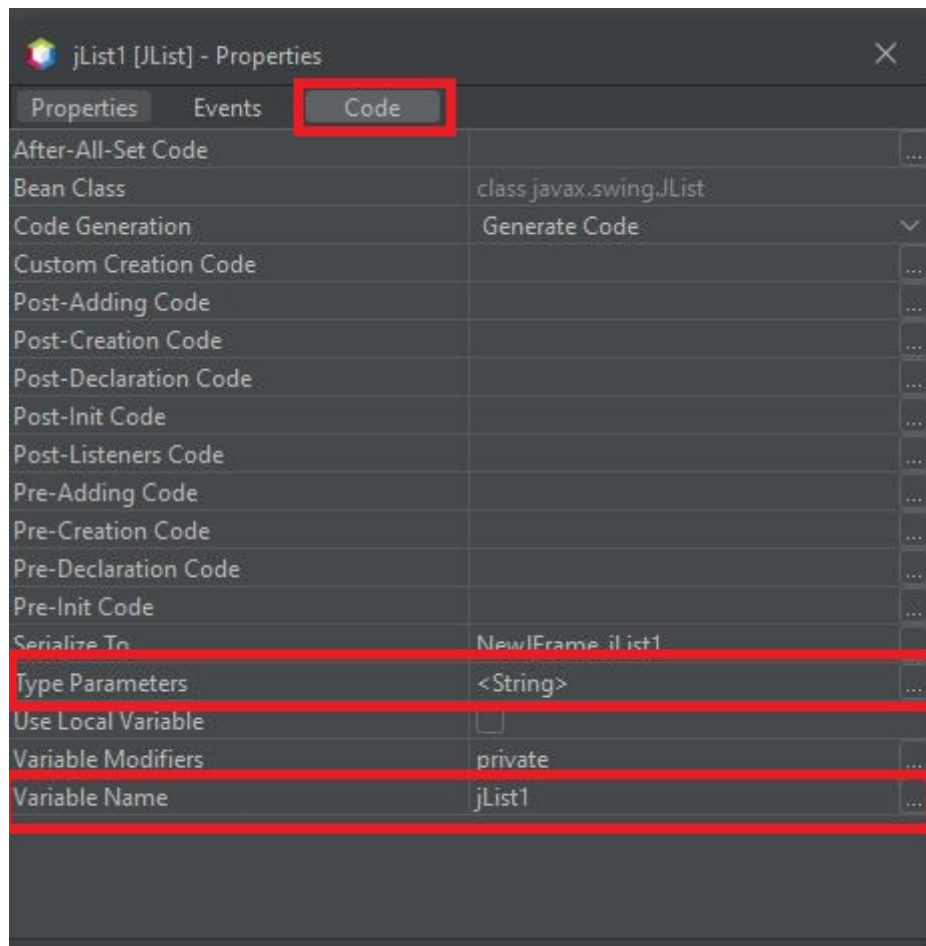


## Paramétrer le Jlist

Les **paramètres essentiels** à considérer pour personnaliser sa JList (clic droit sur la Jlist > **Propriété** ) sont :

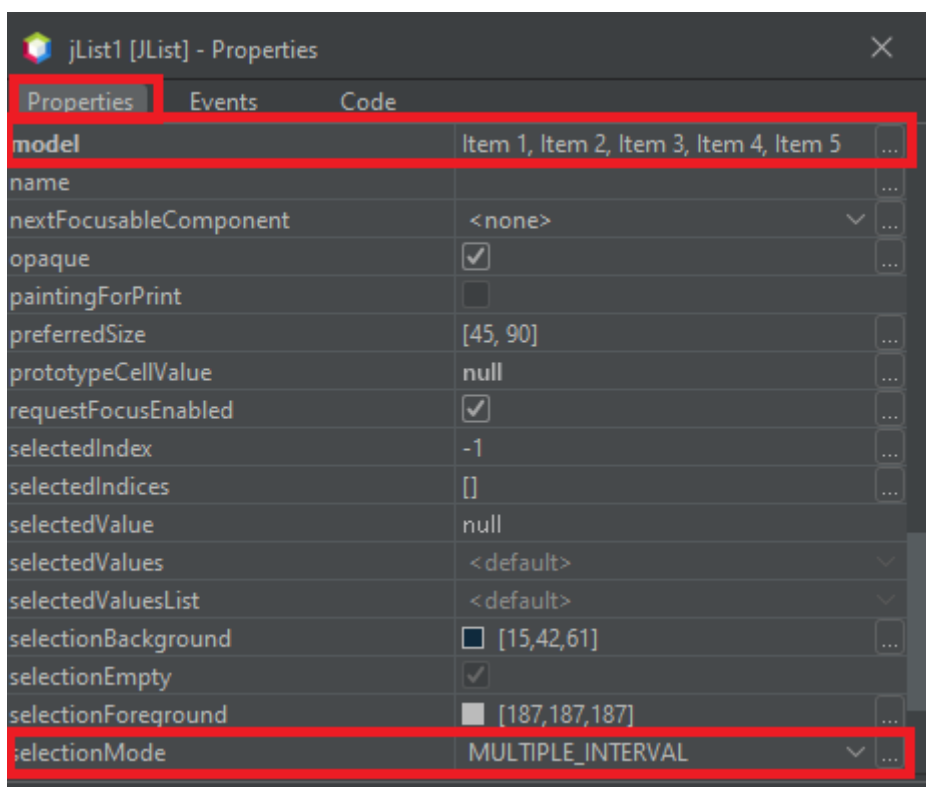
### Onglet Code

- **Variable Name** : le nom de la JList dans le code
- **Type Parameters** : le type des objets associés à la liste



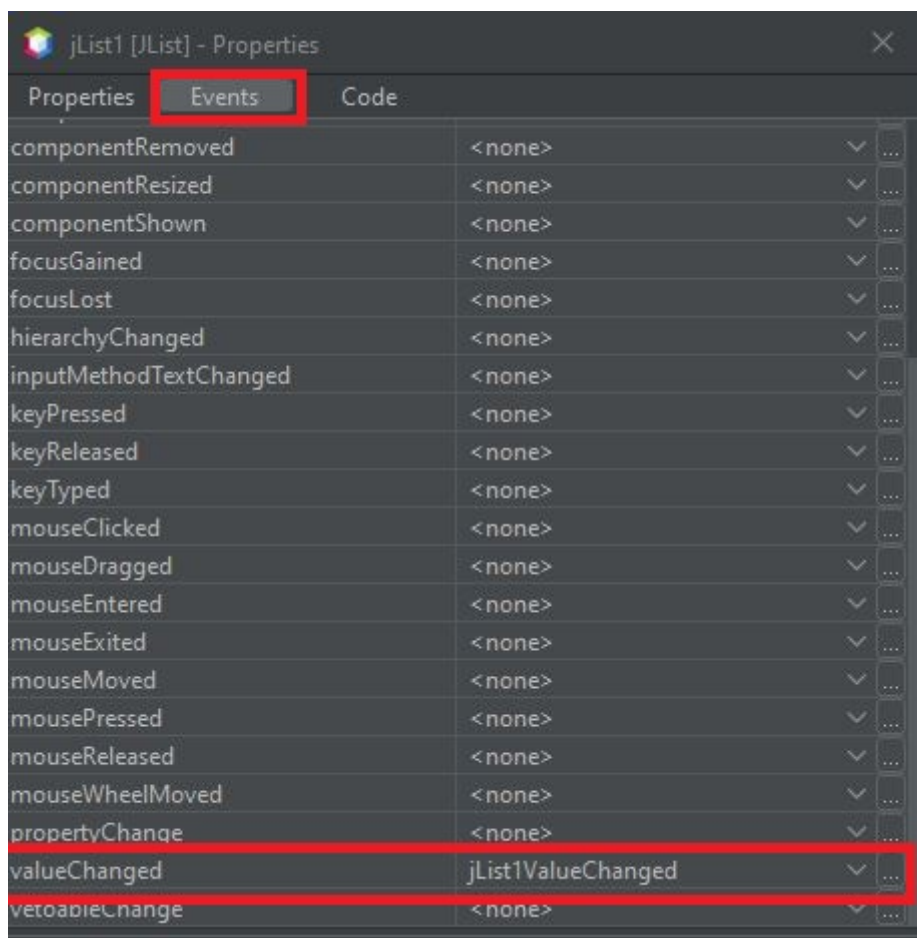
## Onglet Properties

- **selectionMode** : le mode de sélection offert à l'utilisateur
- **model** : Va définir le texte afficher



## Onglet Events

- **valueChanged** : événement qui se déclenche lorsque la valeur sélectionnée dans la liste change, du fait d'une **action utilisateur**



## Alimenter le JList avec des String

- La JList doit être de « **Type Parameter** » **<String>** (étape 2)
- **Créer** un attribut **DefaultListModel** de type **<String>**

```
public class MainWindow{

    DefaultListModel<String> listModel;
```

- **Instancier** le DefaultListModel dans le constructeur

```
public MainWindow() {
    initComponents();
    listModel = new DefaultListModel();
}
```

- Alimenter le DefaultListModel avec des **données de type String** en faisant appel à sa **méthode addElement**

```
listModel.addElement("j'ajoute un élément");
```

- **Associer le DefaultListModel** à la JList (variable name étape 2) en faisant appel au **setModel** de la JList

```
jList1.setModel(listModel);
```

## Alimenter le JList avec des Objets

- Il faut **disposer d'une classe opérante** qui représente les objets choisis. Appelons cette **classe T**
- La JList doit être de « **Type Parameter** » <T> (étape 2)
- **Créer un attribut DefaultListModel de type <T>**

```
public class MainWindow{

    DefaultListModel<T> listModel;
```

\* **Instancier** le DefaultListModel dans le constructeur

```
public MainWindow() {
    initComponents();
    listModel = new DefaultListModel();
}
```

- Si nécessaire, instancier des objets de la classe T
- **Alimenter le DefaultListModel** avec des données de type T en faisant appel à sa méthode **addElement**

```
// class T à pour constructeur 2 paramètres (String nom, String prénom)
T t = new T("Le Brun", "Titouan");
listModel.addElement(t);
```

- **Associer le DefaultListModel** à la JList (variable name étape 2) en faisant appel au **setModel** de la JList

```
jList1.setModel(listModel);
```

La **présentation des données textuelles** dépendra de la valeur renvoyée par la **méthode toString** appliquée aux objets de type T

## Exploiter la valeur sélectionnée dans la Jlist

Avec une Jlist on a la possibilité de **recupérer la donnée d'un élément** lorsqu'un utilisateur clique dessus grâce à la méthode **getSelectedValue** .

```
// getSelectedValue permet de récupérer l'objet dans son ensemble
// (libre à chacun d'utiliser des méthodes de l'objet pour
```

```
// récupérer des valeurs plus précises)
private void listCategsValueChanged(javax.swing.event.ListSelectionEvent
evt) {
    // ici affiche le résultat de la méthode toString appliquée au
    // type d'objet contenu dans la JList
    System.out.println(jList1.getSelectedValue());
}
```

From:

<https://wiki.siochaptalqper.fr/> - Wiki SIO Chaptal

Permanent link:

<https://wiki.siochaptalqper.fr/doku.php?id=bloc2:prog:poo:jlist&rev=1695394246>

Last update: **2023/09/22 16:50**

