

Introduction aux listes graphiques (JList)

Jusqu'à présent vous avez utilisé des `ArrayList` qui permettent de stocker plusieurs objets sous forme d'une liste. Pour afficher une telle liste d'objets dans un programme avec une interface graphique, il vous faut cependant un nouveau composant.

L'image suivante montre l'exemple d'une interface qui permet de saisir et gérer les noms et les notes des participants à un examen.

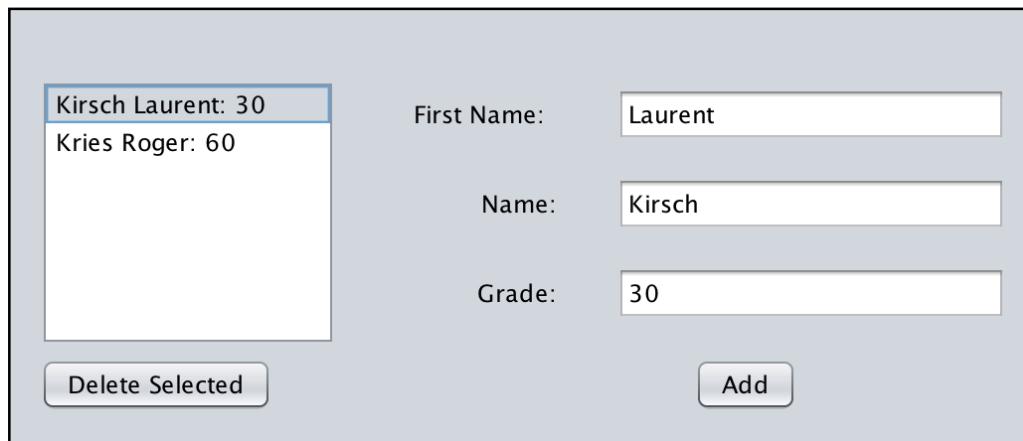


Figure 1 - Interface graphique finale

Le nouveau composant à gauche est une liste graphique qui permet d'afficher tous les participants à l'examen. La classe Java qui représente une telle liste graphique s'appelle `JList`.



Une `JList` affiche les objets d'une liste (vue) tandis qu'une `ArrayList` stocke ces objets (modèle).

1. Copiez le modèle **JList_Intro_M** du répertoire de classe vers votre répertoire personnel.
2. Après avoir ouvert le projet dans NetBeans, vous constatez qu'il y a déjà trois classes Java :
 - Une classe `Exam` qui permet de gérer une liste de participants à un examen. Elle permet d'ajouter des participants à la liste et d'en supprimer.
 - Une classe `Participant` qui représente un participant avec son prénom, son nom de famille et sa note obtenue à l'examen.
 - Une classe `MainFrame` dont le code n'a pas encore été modifié, mais dont l'interface graphique contient déjà les composants graphiques connus avec des noms de variables adaptés.

Configuration de la `JList` à travers l'interface graphique de NetBeans

- La première étape consiste à ajouter une liste graphique à l'interface. Pour cela il faut choisir l'élément correspondant et le poser sur la fiche par glisser-déposer.

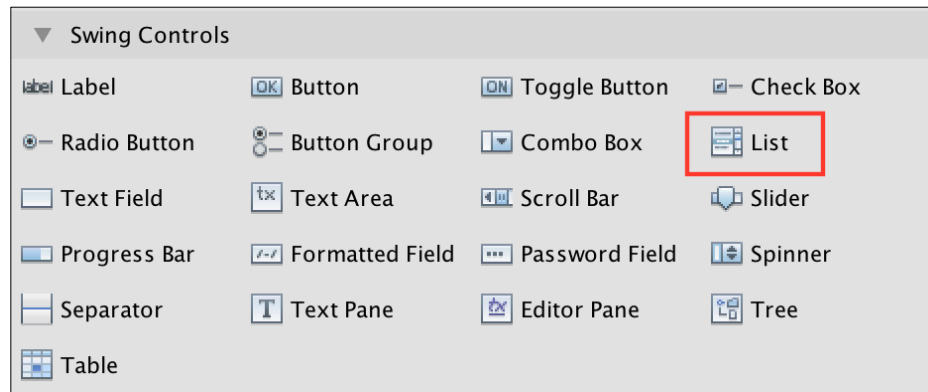


Figure 2 – Choisissez une liste et ajoutez-la à la fiche.

Après avoir adapté les dimensions de la liste, votre interface graphique ressemblera à l'image suivante :

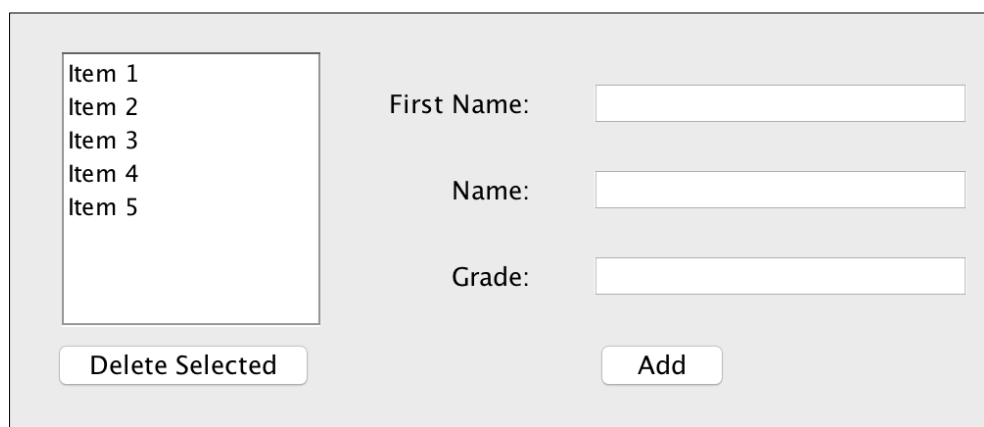


Figure 3 - Interface après l'ajout de la liste

- Vous allez tout d'abord configurer quelques propriétés de la liste. En premier vous allez modifier la propriété **selectionMode** de façon à ce qu'on ne puisse sélectionner qu'un seul élément de la liste. Pour cela la propriété doit avoir la valeur **SINGLE**.

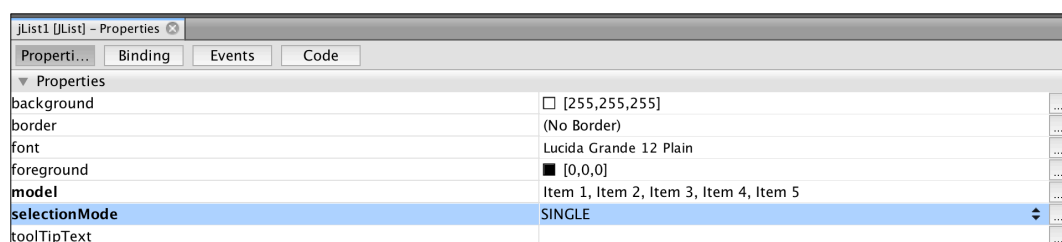


Figure 4 - Cliquez sur **Properties**, puis choisissez la valeur **SINGLE** pour la propriété **selectionMode**

5. Votre liste est censée afficher des objets plus complexes que de simples chaînes de caractères. Elle doit afficher des objets de la classe `Participant`. C'est pourquoi après avoir activé l'onglet **Code** vous devez enlever la valeur actuelle de **Type Parameters**.

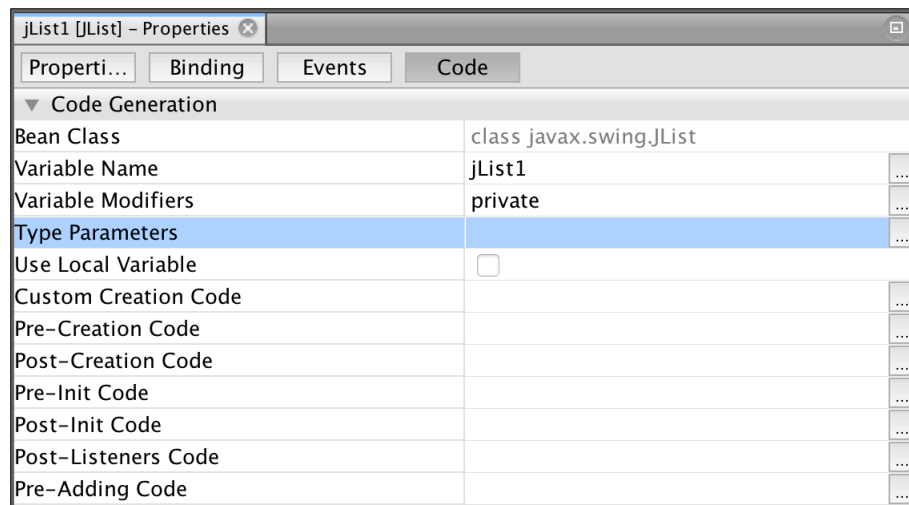


Figure 5 - Enlevez la valeur de **Type Parameters**

6. Profitez de l'occasion pour changer le nom de variable de votre liste. Vous pourriez aussi procéder comme pour les autres éléments à travers le menu contextuel (clic droit sur la liste graphique), mais vous pouvez aussi renommer la variable à travers l'onglet **Code**. Notez que ceci fonctionne aussi pour les autres éléments graphiques.

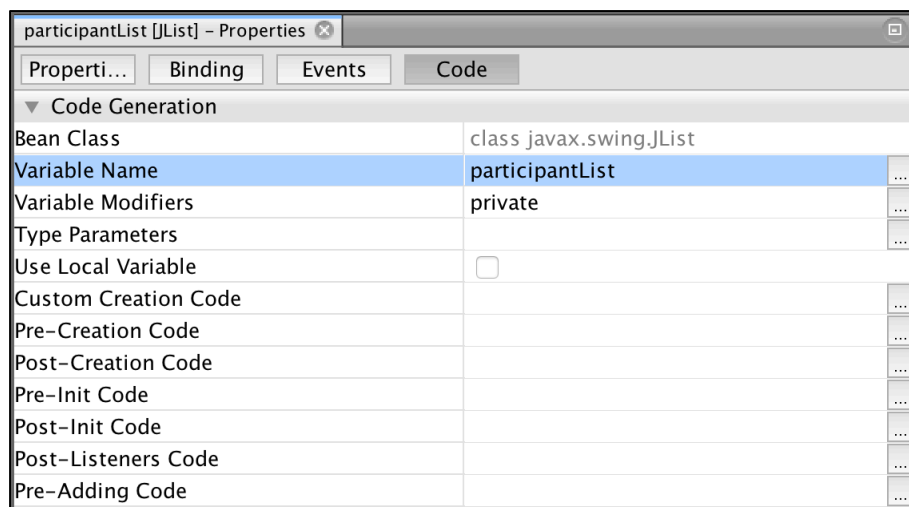


Figure 6 – Changez le nom de variable

A présent votre liste est entièrement configurée et vous pouvez entamer l'écriture du code Java.

Code de la classe `MainFrame` - Initialisations

7. Comme pour tous les programmes MVC il faut d'abord créer un lien entre la vue et le modèle, pour que la vue puisse communiquer avec le modèle. Ajoutez donc un attribut appelé `exam` à la classe `MainFrame` et initialisez-le avec un objet de la classe `Exam`.
8. Au démarrage de votre programme la liste contient les éléments « Item 1 » à « Item 5 », ce qui n'est pas ce que vous voulez afficher. Vous devez indiquer dans le code que la liste graphique `participantList` doit afficher les participants stockés dans votre modèle.

Pour cela vous devez tout d'abord ajouter la méthode suivante à la classe `Exam`.

```
public Object [] toArray(){
    return alParticipants.toArray();
}
```

Figure 7 – La méthode à ajouter

Ne la tapez pas manuellement, mais générez-la comme suit :

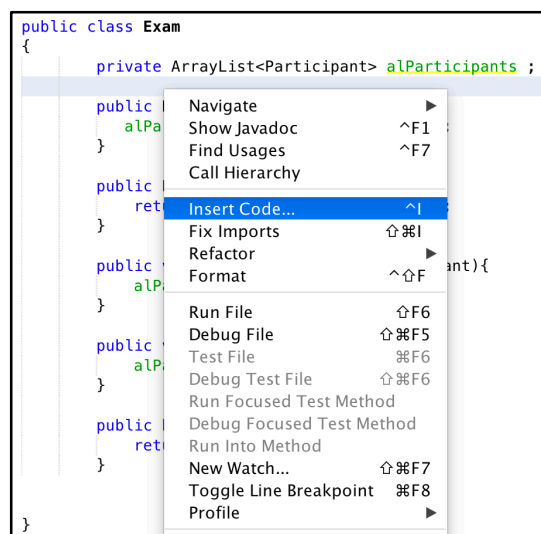


Figure 8 - Effectuez un clic droit sur la fenêtre affichant le code, puis choisissez **Insert Code...**

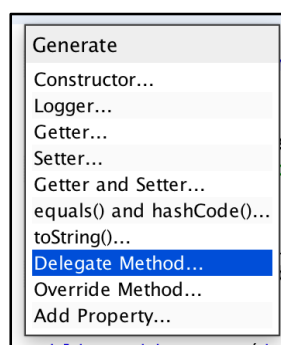


Figure 9 - Choisissez **Delegate Method...**

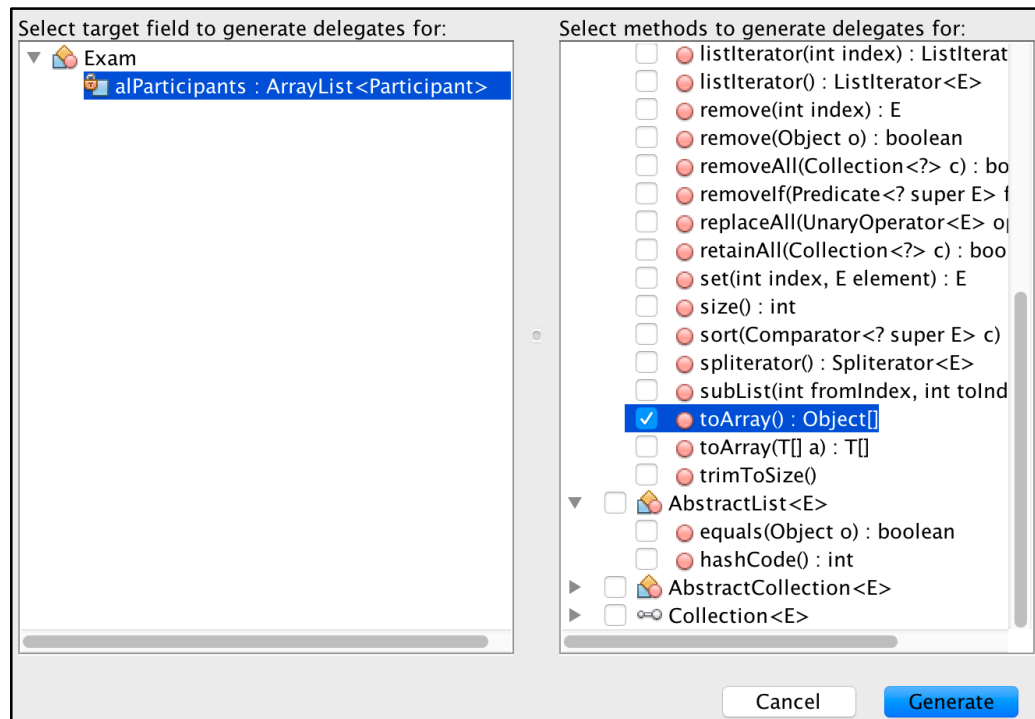


Figure 10 - Sélectionnez **alParticipants** (à gauche), cochez la méthode **toArray** (à droite) et appuyez sur le bouton **Generate**

La méthode `toArray` retourne les objets de la liste sous forme d'un tableau (array), comme une `JList` requiert les données sous forme d'un tableau pour fonctionner correctement. Vous pourrez générer la méthode `toArray` de la même façon pour tous vos programmes.

9. Ajoutez la deuxième instruction de l'image au constructeur de la classe `MainFrame` :

```
public MainFrame() {
    initComponents();
    participantList.setListData(exam.toArray());
}
```

Figure 11 - Passez les données du modèle sous forme d'un tableau à la `JList`

Cette instruction assure que la `JList` du nom `participantList` affiche les objets de la liste `alParticipants` de l'objet `exam` dès le démarrage de votre programme.

Code de la classe `MainFrame` – Le comportement des boutons

10. Quand le bouton **Add** est appuyé, il faut accéder aux données des trois champs de texte, créer un nouveau participant, l'ajouter à la liste du modèle, puis actualiser la liste graphique.

L'instruction...

```
participantList.setListData(exam.toArray());
```

... doit être exécutée chaque fois que le contenu affiché par la liste graphique doit être actualisé.



11. Quand le bouton **Delete** est appuyé, il faut accéder à l'index sélectionné dans la liste graphique, vérifier qu'il y a bien un index sélectionné et supprimer le participant correspondant de la liste du modèle, s'il y a un index sélectionné.

```
participantList.getSelectedIndex()
```

retourne soit l'index du participant sélectionné, soit -1 si aucun participant de la liste n'est sélectionné.



Code de la classe `MainFrame` – Réagir aux changements de sélection

12. Après chaque changement de sélection dans la liste graphique les champs de texte doivent être remplis avec les informations du participant qui vient d'être sélectionné dans la liste. Il faut donc détecter et réagir aux changements de sélection.

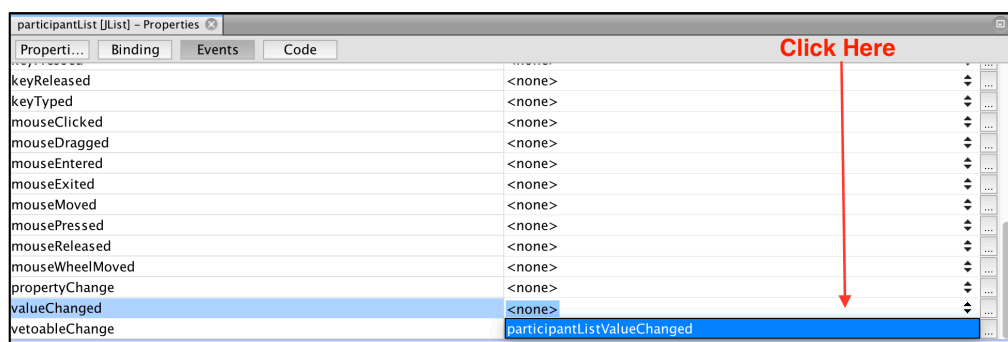


Figure 12 – Ajoutez la méthode qui permet de réagir aux changements de sélection

Sélectionnez la liste `participantList` et activez l'onglet **Events**. Cherchez l'événement **valueChanged** et cliquez sur l'espace blanc - comme indiqué sur l'image ci-dessus - avant de cliquer sur le nom de la méthode qui vient d'apparaître.

La méthode vide que vous voyez après avoir effectué les étapes ci-dessus doit être complétée comme suit : Il faut accéder à l'index sélectionné de la liste graphique, vérifier s'il y a bien un index sélectionné et uniquement s'il y en a un, il faut accéder au participant correspondant et afficher ses données dans les trois champs de texte.