

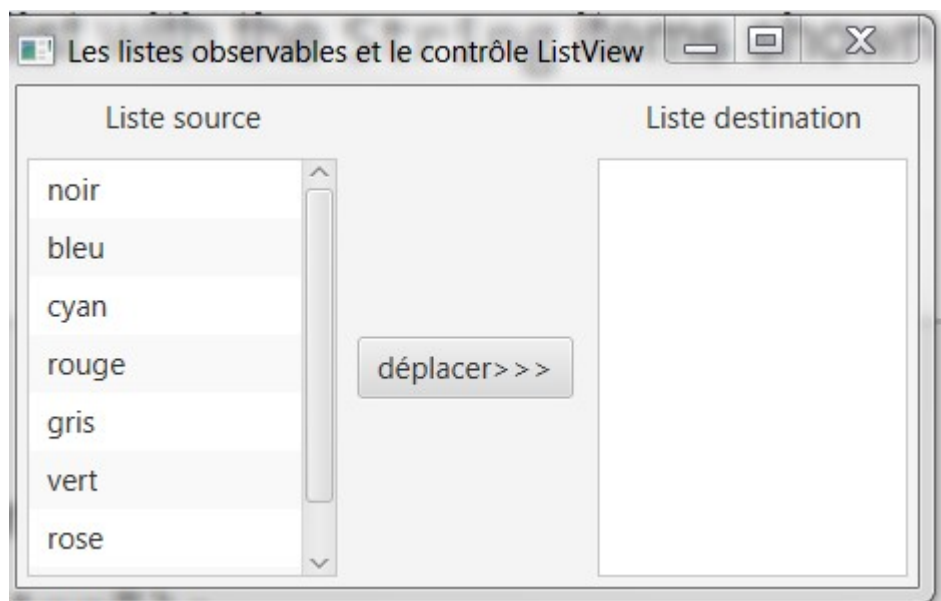
Laboratoire 6
Les contrôles ListView, TableView et la collection ObservableList
Les propriétés JavaFX

Objectifs d'apprentissage

- Utiliser le contrôle *ListView*
- Utiliser le contrôle *TableView*
- Utiliser la collection *ObservableList* pour peupler les contrôles *ListView* et *TableView*
- Comprendre et utiliser les propriétés JavaFX

I- Le contrôle ListView

Créez une classe *UtilisationDeListView*, sous-classe de *Application* pour reproduire l'interface graphique suivante :



Votre interface doit répondre aux spécifications suivantes :

- Le panneau racine (root) est une un *GridPane* de trois colonnes et deux lignes (voir plus loin pourquoi)
- L'espace entre les nœuds et les bordures du panneau est de 5 pixels.
- Les espaces horizontal et vertical entre les nœuds du panneau sont de 10 pixels.
- Les trois colonnes sont définies comme suit :

```
ColumnConstraints colonne1 = new ColumnConstraints(150, 150, Double.MAX_VALUE);
ColumnConstraints colonne2 = new ColumnConstraints(110);
ColumnConstraints colonne3 = new ColumnConstraints(150, 150, Double.MAX_VALUE);
gridPane.getColumnConstraints().addAll(colonne1,colonne2, colonne3);
```

- Bien comprendre l'effet de ces contraintes (gridPane est le panneau de type GridPane).
- Autorisez le redimensionnement en largeur des colonnes 1 et 3. Utiliser la méthode *setHGrow* de la classe *ColumnConstraints*
- Ajoutez l'étiquette « Liste source » à la position (0,0) du GridPane(1^{ère} colonne, 1^{ère} ligne). Notez que cette étiquette est au centre de la cellule. Utilisez la méthode *static setHalignment* de la classe *GridPane*.
- Ajoutez l'étiquette « Listedestination » à la position (2,0) du GridPane (3^{ième} colonne, 1^{ère} ligne). Notez que cette étiquette est au centre de la cellule.
- Créez une liste observable dont les éléments seront insérés dans la liste source comme suit :

```
//Liste de départ
final ObservableList<String> lstCouleurs = FXCollections.observableArrayList("noir",
    "bleu",
    "cyan",
    "rouge",
    "gris",
    "vert",
    "rose",
    "jaune"
);

final ListView<String> listeSourceListView = new ListView<>(lstCouleurs);
```

- Ajoutez le contrôle *listViewSource* à la position (0,1) du *gridPane*.
- Ajoutez le bouton « Déplacer » à la position (1,1) du *gridPane*
- Ajoutez un contrôle *ListView*, pour la liste destination, à la position (2,1). Ce contrôle est initialisé à l'aide d'une liste observable vide.
- Programmez le clic du bouton « Déplacer » qui permet de déplacer la couleur sélectionnée de la liste source à la liste destination. Pour récupérer la couleur sélectionnée, utilisez l'instruction :

```
String couleur = listeSourceListView.getSelectionModel().getSelectedItem();
```

- Enlevez la couleur sélectionnée de la liste observable (le modèle), l'ajouter à la liste observable (modèle) de destination. Ne pas oublier d'annuler la sélection.

```
listeSourceListView.getSelectionModel().clearSelection();
```

X

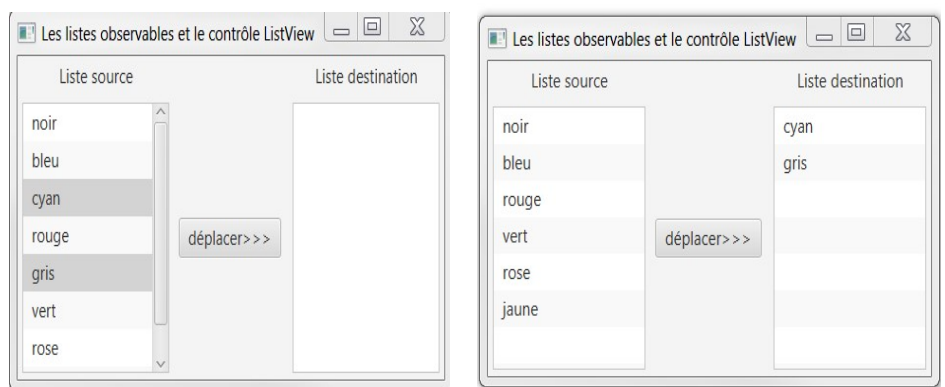
- Exécutez. Notez que les modifications des listes observables (source et destination) se répercutent dans les contrôles *ListView* qui affichent les données correspondantes.
- Mettez en commentaire, le gestionnaire du clic du bouton « Déplacer », le remplacer par un gestionnaire où la sélection multiple est autorisée.

```
// autoriser la sélection multiple.
listeSourceListView.getSelectionModel().setSelectionMode(SelectionMode.MULTIPLE);
```

- Dans ce cas, pour récupérer la liste de couleurs sélectionnée, utiliser l'instruction :

```
ObservableList<String> listeSelectionnee = listeSourceListView.getSelectionModel().getSelectedItems();
```

- Ajoutez les couleurs sélectionnée à la liste observable destination et l'enlever de la liste observable source(utilisez les méthodes `addAll` et `removeAll`) de la classe `ObservableList`)
- Voici un exemple d'exécution :



II- Les propriétés JavaFX

Créez une classe *Etudiant* avec des propriétés JavaFX comme suit :

```
import javafx.beans.property.SimpleDoubleProperty;
import javafx.beans.property.SimpleIntegerProperty;
import javafx.beans.property.SimpleStringProperty;

public class Etudiant {

    private final SimpleIntegerProperty numDA;
    private final SimpleStringProperty nom;
    private final SimpleStringProperty prenom;
    private final SimpleDoubleProperty moyenne;

    public Etudiant(int numDA, String nom, String prenom, double moyenne) {

        this.numDA = new SimpleIntegerProperty(numDA);
        this.nom = new SimpleStringProperty (nom);
        this.prenom = new SimpleStringProperty(prenom);
        this.moyenne = new SimpleDoubleProperty(moyenne);

    }
}
```

Cliquez droit dans la fenêtre de code de la classe choisir: source/Generate JavaFX Getters and Setters. Sélectionnez toutes les propriétés.

- Bien comprendre les méthodes générées. Faire des tests au besoin.

III- Le contrôle TableView

1- Création d'une table vide à 4 colonnes :



Créez une classe *UtilisationDeTableView*, sous-classe de *Application*, avec les spécifications suivantes :

- La racine est un nœud de type *Group*
- Créez un panneau *VBox* qui contient le texte « *Exemple d'utilisation d'une table* » de police « *Arial* » et de taille 20. Ce *VBox* contient aussi une table construite comme suit :
 - Ajoutez la variable d'instance suivante à votre classe :

```
private final TableView<Etudiant> table = new TableView<Etudiant>();
```
 - Créez les colonnes à l'aide des instructions suivantes :

```
TableColumn<Etudiant, Integer> colonneNumDA = new TableColumn<Etudiant, Integer> ("Numéro de DA");
TableColumn<Etudiant, String> colonnePrenom = new TableColumn<Etudiant, String> ("Prenom");
TableColumn<Etudiant, String> colonneNom = new TableColumn<Etudiant, String> ("Nom");
TableColumn<Etudiant, Double> colonneMoyenne = new TableColumn<Etudiant, Double> ("Moyenne");
table.getColumns().addAll(colonneNumDA, colonneNom, colonnePrenom, colonneMoyenne);
```
 - Fixez la largeur préférée de toutes les colonnes à 120.
 - Fixez la largeur maximum de la colonne moyenne à sa largeur préférée. (ceci empêchera son redimensionnement)

2- Association de données aux colonnes en utilisant des propriétés définies pour chaque donnée.

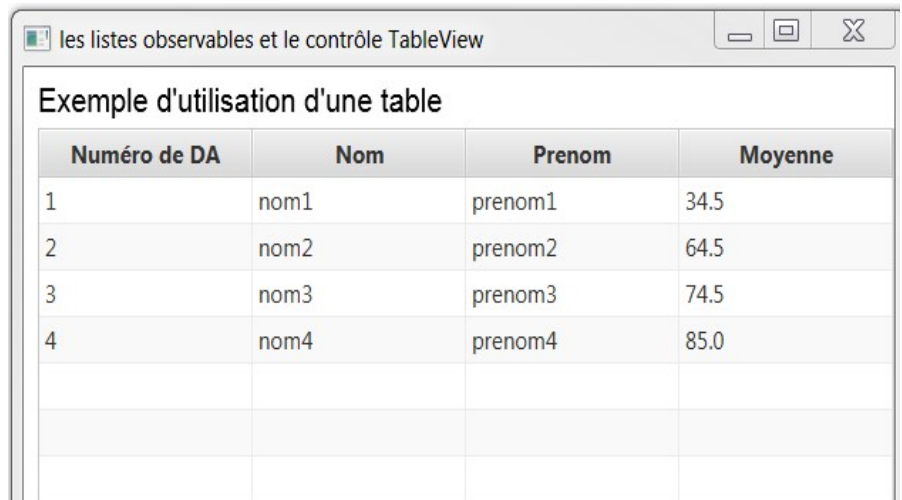
- Ajouter la variable d'instance suivante à votre classe :

```
private final ObservableList<Etudiant> donnees =  
FXCollections.observableArrayList(new Etudiant(1, "nom1", "prenom1", 34.5),  
    new Etudiant(2, "nom2", "prenom2", 74.5), new Etudiant(3, "nom3", "prenom3", 64.5),  
    new Etudiant(4, "nom4", "prenom4", 85.0));
```

- Voici les instructions pour insérer les données dans le table

```
colonneNumDA.setCellValueFactory(new PropertyValueFactory<>("numDA"));  
colonneNom.setCellValueFactory(new PropertyValueFactory<>("nom"));  
colonnePrenom.setCellValueFactory(new PropertyValueFactory<>("prenom"));  
colonneMoyenne.setCellValueFactory(new PropertyValueFactory<>("moyenne"));  
// spécifier les colonnes et les données  
table.setItems(donnees);
```

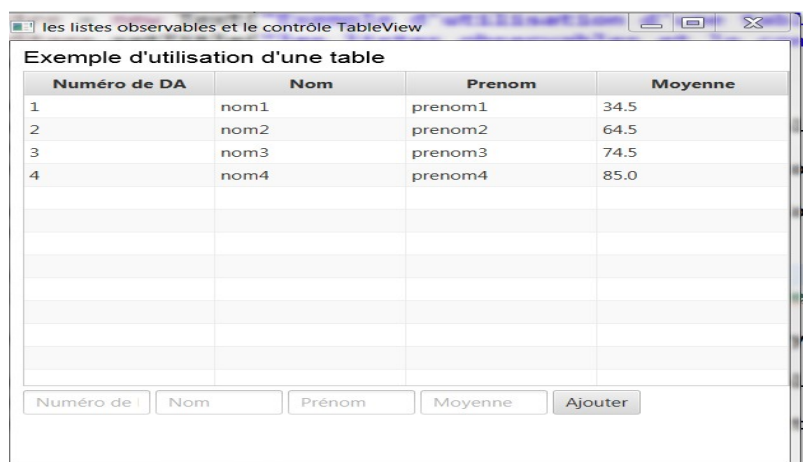
- Exécutez



Numéro de DA	Nom	Prenom	Moyenne
1	nom1	prenom1	34.5
2	nom2	prenom2	64.5
3	nom3	prenom3	74.5
4	nom4	prenom4	85.0

3- Ajout de données dans la table

- Ajoutez 4 *TextField* et un bouton à votre interface :



Numéro de DA	Nom	Prenom	Moyenne
1	nom1	prenom1	34.5
2	nom2	prenom2	64.5
3	nom3	prenom3	74.5
4	nom4	prenom4	85.0

- Programmez le gestionnaire du clic du bouton *Ajouter* comme suit :

```
btnAjout.setOnAction((ActionEvent e)-> {
    donnees.add(new Etudiant(Integer.parseInt(txtNumDA.getText().trim()),
        txtNom.getText(), txtPrenom.getText(), Double.parseDouble(txtMoyenne.getText().trim())));
    txtNumDA.clear();
    txtNom.clear();
    txtPrenom.clear();
    txtMoyenne.clear();
});
```

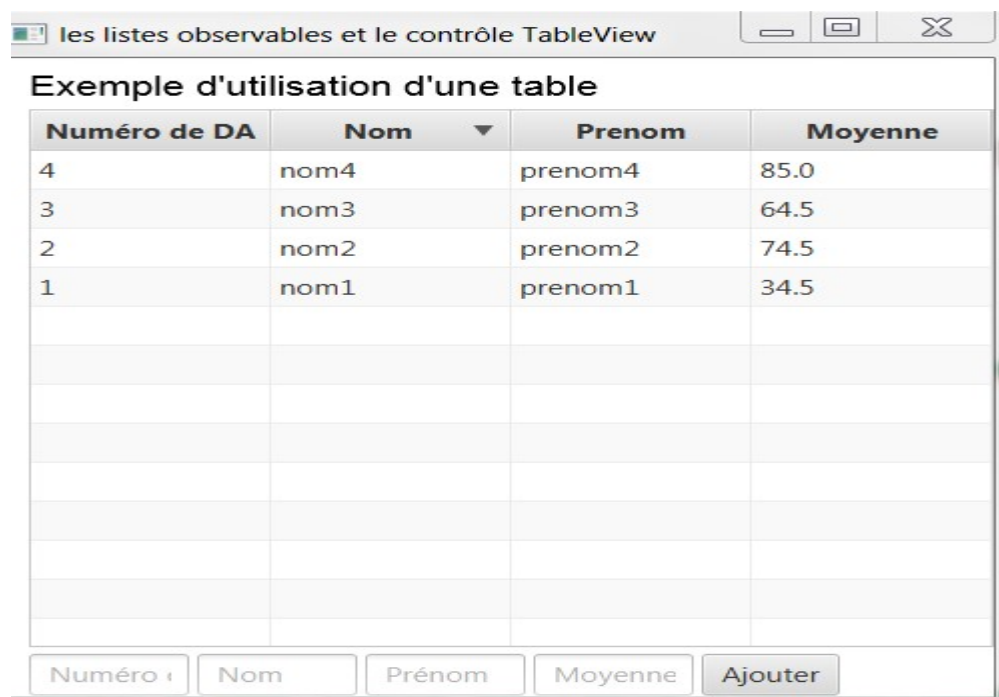
Notez l'utilisation d'une expression lambda. Vous pouvez aussi utiliser une classe anonyme.

Il est important de noter que les données ont été ajoutées dans le modèle (la structure de données) et que la mise à jour de la table est automatique

- Exécutez

4- Tri des données affichées

Le contrôle *TableView* fournit des tris pour les données d'une colonne : le premier clic active le tri en ordre croissant, le second en ordre décroissant et le troisième clic désactive le tri. Faire des tests. Voici un exemple de tri décroissant par nom.



Exemple d'utilisation d'une table

Numéro de DA	Nom ▼	Prenom	Moyenne
4	nom4	prenom4	85.0
3	nom3	prenom3	64.5
2	nom2	prenom2	74.5
1	nom1	prenom1	34.5

Numéro : Nom Prenom Moyenne Ajouter

On peut aussi trier selon plusieurs colonnes. Utilisez Shift+ les colonnes à trier.