Secteur Tertiaire Informatique

Filière « Etude et développement »

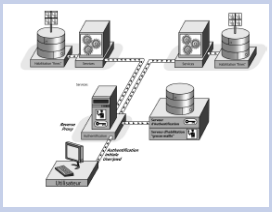
TP – Développer des interfaces graphiques

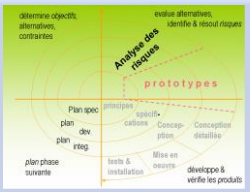
IHM Java FX – TableView

Apprentissage

Mise en situation

Evaluation





1. PROJET FXML

FXML est un format de données textuelles, dérivé du XML, permettant de décrire une interface utilisateur pour JavaFX. Il peut être considéré comme le « html du JavaFX ».

Il permet de mettre en forme tous les objets nécessaires à la construction d'interface graphiques, tels que :

- Des conteneurs (GridPane, AnchorPane, BorderPane…) ;
- Des composants graphiques ;
- Des formes ;
- Des graphiques…

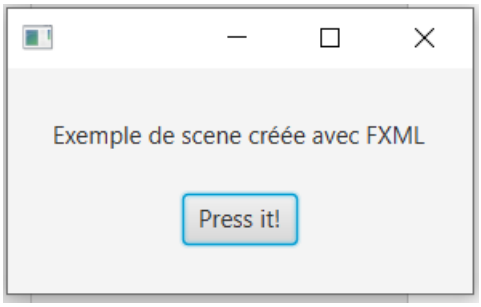
Les fichiers FXML peuvent être écrits manuellement mais peuvent rapidement devenir lourd à gérer, par exemple le code suivant n'a pour objectif que d'afficher un label et un bouton :

```
<?xml version="1.0" encoding="UTF-8"?>

<?import javafx.geometry.Insets?>
<?import javafx.scene.control.Button?>
<?import javafx.scene.control.Label?>
<?import javafx.scene.layout.VBox?>

<VBox alignment="CENTER" spacing="20.0" xmlns="http://javafx.com/javafx/18"
xmlns:fx="http://javafx.com/fxml/1"
fx:controller="fr.afpa.fxmltest.PrimaryController">
   <children>
      <Label fx:id="helloLabel" text="Exemple de scene créée avec FXML"
visible="false" />
      <Button fx:id="primaryButton" onAction="#displayLabel" text="Press it!" />
   </children>
   <padding>
      <Insets bottom="20.0" left="20.0" right="20.0" top="20.0" />
   </padding>
</VBox>
```

Ce code FXML affiche la fenêtre suivant (après un clic sur le bouton « Press it! ») :



Afin de faciliter la conception, un outil de développement a été conçu : Scene builder.

Cet outil est disponible au téléchargement ici : https://gluonhq.com/products/scene-builder/.

L'outil peut être intégré à votre IDE grâce à une extension :

- « SceneBuilder extension » sur VS Code ;
- « e(fx)clipse » sur Eclipse.

La suite du TP a pour objectif de coder un tableau affichant des informations concernant des personnes en utilisant une « TableView ».

2. CREATION DU PROJET

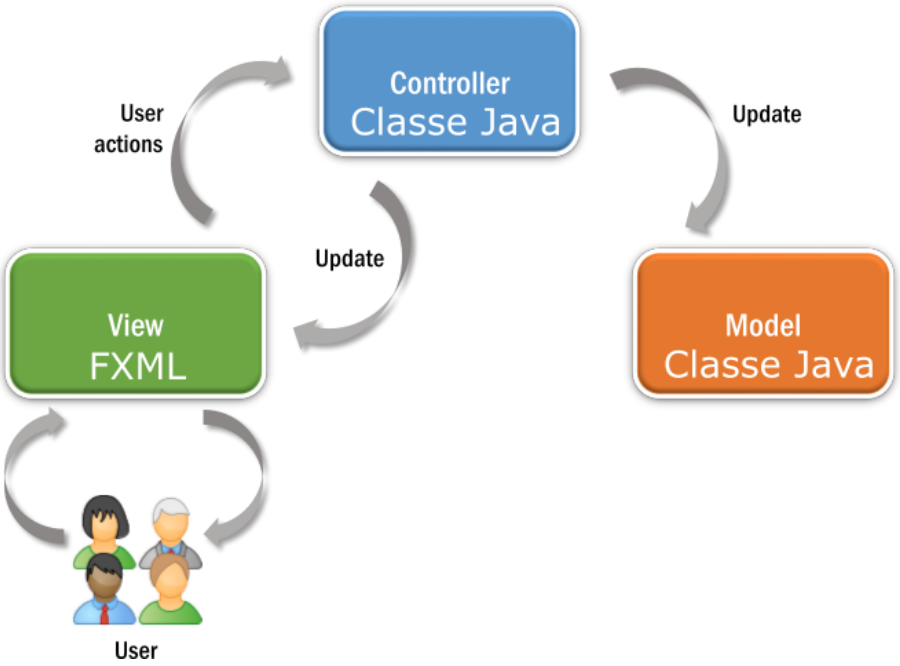
Créez un projet basé sur Maven en utilisant l'archétype « JavaFX ». Ceci va vous permettre d'obtenir un projet simple comprenant des classes Java faisant office de controller ainsi que des fichiers FXML de démonstration.

Ce projet peut servir de base au développement.

2.1 MODELE MVC (SIMPLIFIE)

Coder une application JavaFX basé sur du FXML revient à adopter une architecture logicielle en MVC.

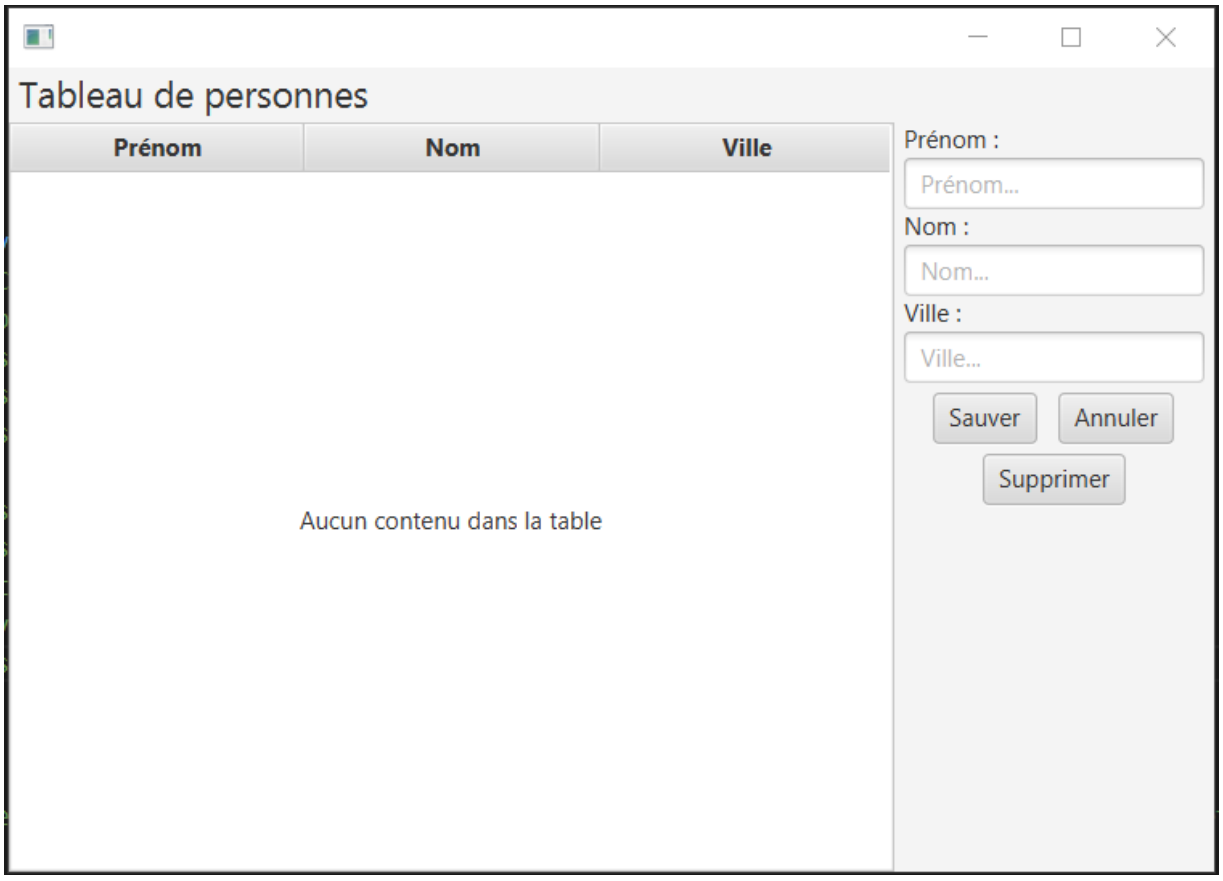
Ainsi, le modèle adapté à ce TP peut être vue de la sorte :



2.2 CREATION DE LA VUE EN FXML

Ouvrez votre vue dans SceneBuilder et ajoutez un élément `TableView`, puis glissez trois `TableColumn` dans votre `TableView`.

Ajoutez les labels et les boutons nécessaires pour obtenir la fenêtre suivante :



Composants graphiques à ajouter :

- TableView → tableau affichant les personnes
- TableColumn → colonne correspondant aux informations à afficher
- TextField (et leur Label) → formulaire pour ajouter/modifier une personne
- Button → modification du tableau/formulaire

Fonctionnalités :

- Affichage d'une liste de personnes ;
- Ajout d'une nouvelle personne au tableau en utilisant le bouton « Sauver » (les informations contenues dans les « TextField » prénom, nom et ville seront utilisée) ;

- Suppression d'une personne du tableau avec le bouton « Supprimer » ;
- Nettoyage de la saisie utilisateur du formulaire via le bouton « Annuler ».

2.3 CREATION D'UN MODELE

Nous allons écrire une classe « Person » pour charger le contenu de la tableview.

Pour notre exemple, une personne aura trois attributs :

- Nom ;
- Prénom ;
- Ville.

Pour le type des attributs nous pouvons utiliser des « String » classiques mais il est également possible (et recommandé) d'utiliser des objets de la classe « javafx.beans.property.stringProperty » permettant plusieurs choses :

- L'ajout d'un « listener » sur l'attribut en question permet de pouvoir traiter un évènement lors de sa modification ;
- Le lien (ou « binding ») de l'attribut permettant d'effectuer directement dans le modèle une modification effectuée sur l'interface grpahique en JavaFX.

Vous devrez donc créer une classe « Person » avec les attributs suivants :

```
    private StringProperty firstName;
    private StringProperty lastName;
    private StringProperty city;
```

Le constructeur de la classe pour être déclaré comme suit :

```
    public Person(String firstName, String lastName, String city)
```

A vous de trouver comment faire en sorte d'instancier un objet de la classe « StringProperty » à partir d'un objet « String ».

Il vous faudra également générer les accesseurs et mutateurs (« getters » et « setters »).

Pour ce projet de découverte de FXML vous pourrez créer la classe « Person » dans votre package principal.

A terme, pour une meilleure architecture logicielle, il sera préférable de créer un package Java par nature des fichiers sources qu'il contient.

Par exemple, pour la classe « Person » qui appartient au modèle vous pourriez créer un package « fr.afpa.model » (nom à adapter à votre contexte).

2.4 CREATION D'UN CONTROLEUR

Le contrôleur est une classe Java qui va implémenter la logique de traitement à appliquer sur le modèle en fonction des actions utilisateurs effectuée sur la vue.

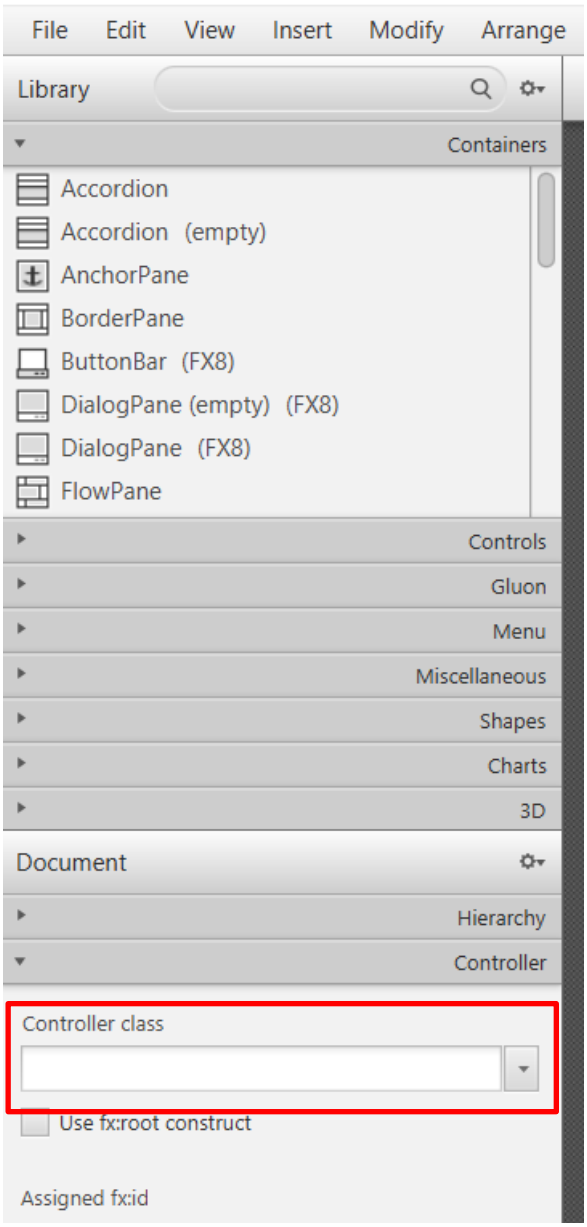
Cette classe doit comprendre des attributs correspondant aux composants graphiques mis en place dans la vue en FXML.

2.4.1 Lien entre la vue et le contrôleur

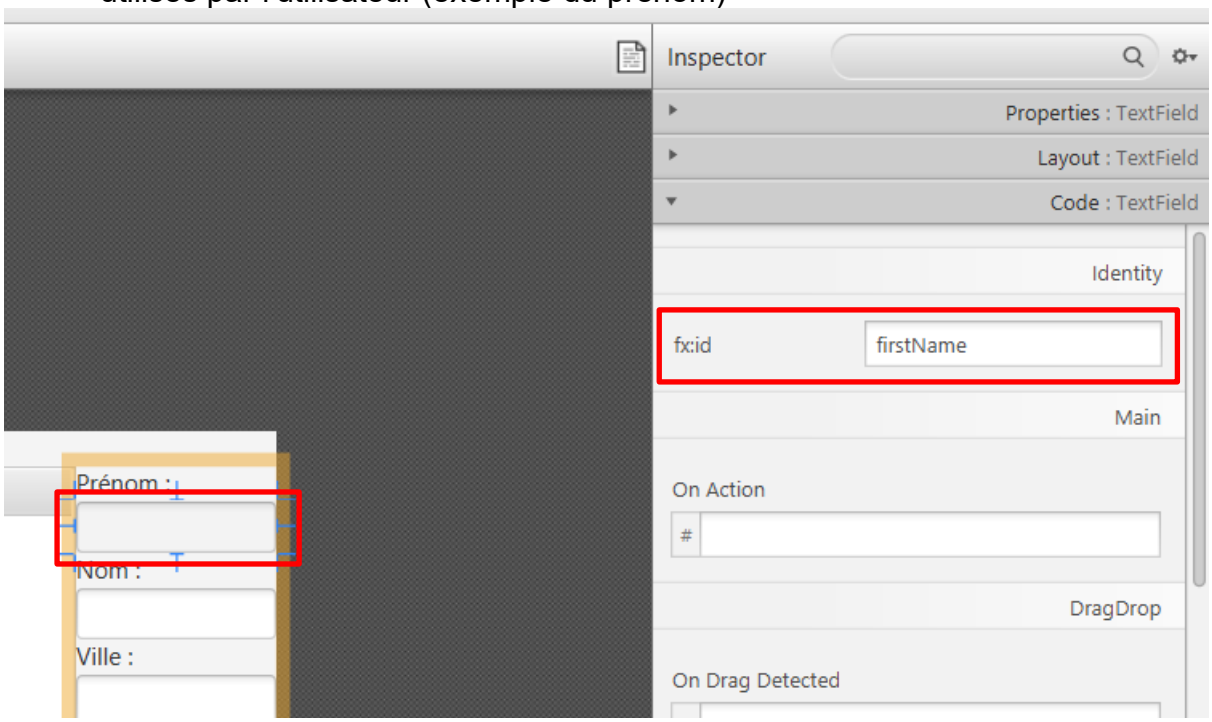
L'exemple suivant fournit la marche à suivre pour lier un composant FXML (ici le « TextField » correspondant au prénom d'une personne) avec le contrôleur :

1. Créer une classe qui sera le contrôleur de la vue. Par convention nous utiliserons un nom de classe suffixé par « Controller ». Dans notre exemple vous pourrez utiliser « **TableViewController.java** »

2. Ajouter sur Scene Builder le lien avec le contrôleur nouvellement créé :



3. Ajouter sur Scene Builder les « `fx :id` » des composants graphiques qui seront utilisés par l'utilisateur (exemple du prénom)



4. Ajouter en attribut de la classe contrôleur les composants graphiques déclarés dans le FXML :

```
@FXML
private TextField firstName;
```

Le lien en l'élément FXML se fait en utilisant l'annotation « `@FXML` ».

Ajoutez des fx :id sur les composants suivants :

- TableColumn
- TextField
- Button

2.4.2 Structure de données répertoriant les personnes

Une fois tous les attributs ajoutés il vous faudra un attribut stockant toutes les personnes à afficher. En JavaFX vous pourrez utiliser une structure de données appelée « Collection observable » (structure de donnée qui émet des évènements lorsque son contenu change).

Pour plus d'informations sur les collections observables :
https://java.developpez.com/faq/javafx?page=Collections-observables#Que-sont-les-collections-observables

Vous pouvez donc ajouter un attribut comme tel :

```
private ObservableList<Person> persons = FXCollections.observableArrayList();
```

2.4.3 Initialisation du contrôleur

Le chargement d'un fichier FXML associé à son contrôleur est effectué grâce à la classe FXMLoader et sa méthode « `load()` ».

« `load()` » effectue une instanciation du contrôleur associé au FXML et une méthode « `initialize()` » du contrôleur est appelée (https://openjfx.io/javadoc/18/javafx.fxml/javafx/fxml/Initializable.html).

Comme son nom l'indique, cette méthode permet d'initialiser les données nécessaires au fonctionnement du contrôleur (et donc de notre vue). Dans notre cas, nous cherchons à afficher une liste de personnes.

Méthode à ajouter au contrôleur (attention à l'annotation @FXML nécessaire pour qu'elle puisse appelée par FXMLoader) :

```
    @FXML
    public void initialize()
```

Il vous faudra ensuite ajouter des personnes à la collection observable.

```
        persons.add(new Person("Josh", "Homme", "Joshua Tree"));
        persons.add(new Person("Dave", "Grohl", "Warren"));
        persons.add(new Person("Robert", "Trujillo", "Santa Monica"));
```

2.4.4 Paramétrage de l'affichage

Il vous faut maintenant faire le lien entre les colonnes et vos objets de « Person ».

Voici un exemple pour **paramétrer l'affichage des cellules de la colonne** « Prénom » (ce qui est passé en paramètre est une « expression lambda », une excellente ressource sur le sujet est disponible ici) :

```
firstNameCol.setCellValueFactory(cellData -> cellData.getValue().getFirstName());
```

Ajoutez le code pour correctement afficher toutes les colonnes du tableau.

2.4.5 Implémentation des actions utilisateur

Implémentez les méthodes suivantes dans votre classe contrôleur :

```
/**
     * Sauvegarde les nouvelles données dans le modèle (création d'une nouvelle
personne et ajout à l'"ObservableList").
     *
     * @param event L'évènement à gérer.
     */
    @FXML
    private void save(ActionEvent event) {

    }

    /**
```

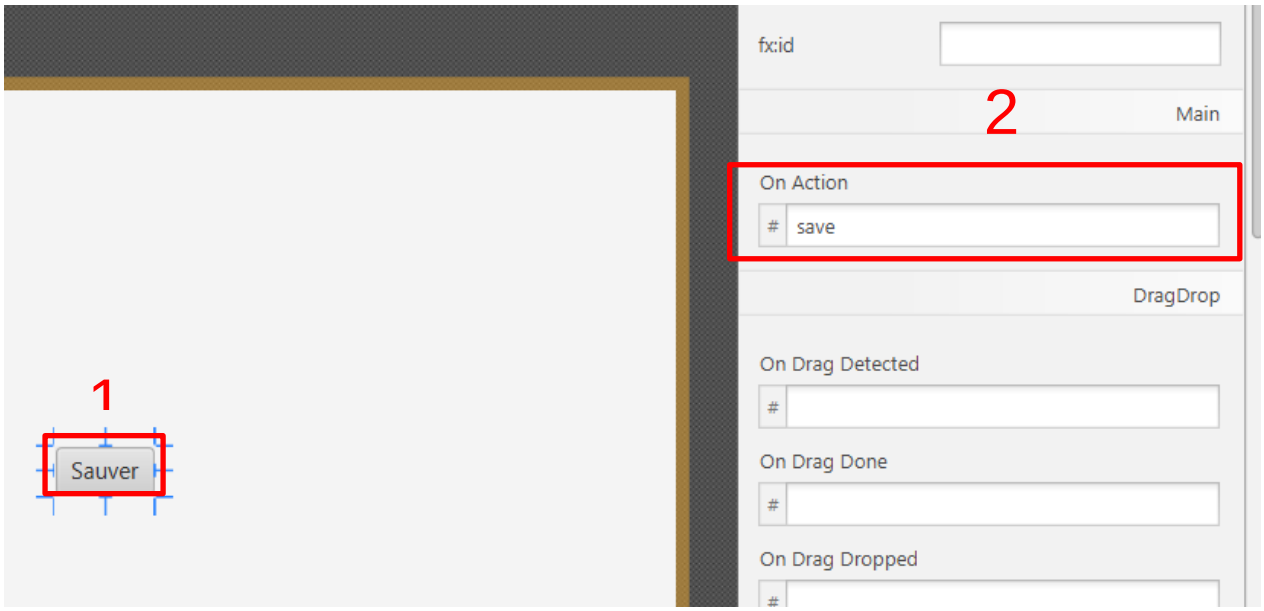
```java
     * Annuler la saisie (vide le formulaire).
     */
    @FXML
    private void cancel(ActionEvent event) {

    }

    /**
     * Suppression d'une personne du tableau.
     */
    @FXML
    private void delete(ActionEvent event) {

    }
```

Le lien entre FXML et la méthode de gestion de l'évènement du contrôleur peut se faire via Scene Builder comme suit :



CREDITS

ŒUVRE COLLECTIVE DE l'AFPA
Sous le pilotage de la DIIP et du centre d'ingénierie sectoriel Tertiaire-Services

Equipe de conception (IF, formateur, mediatiseur)
Michel Coulard – Formateur Evry
Chantal Perrachon – IF Neuilly sur Marne

Date de mise à jour : 28/02/2022