

Introduction à JavaFX

Emmanuel BRUNO

Université de Toulon, LIS UMR CNRS 7020
emmanuel.bruno@univ-tln.fr

2026-10-14

Table des matières

1 Introduction	1
2 Historique et évolution	1
3 Quickstart	1
4 Etudes de Cas pratiques	1
5 Modules	2
6 Structure d'une application JavaFX	2
7 Multi-tâche dans JavaFX	2
8 Vues (Scene)	2
9 Layouts et contrôles	2
10 Application JavaFX	3
11 JavaFX Properties	3
12 Contrôleur	3
13 Pour aller plus loin	3

1 Introduction

- JavaFX est une API pour construire des IHM en Java
- Entre Java 8 et 10 inclus dans le JDK
- A partir de Java 11 dépendance externe
- Intégration avec l'OS natif :
 - Optimisation matérielle (OpenGL, DirectX, ...)
 - Files d'évènements

2 Historique et évolution

- **2008** : Oracle annonce JavaFX comme successeur de Swing
- **2011** : Intégration dans le JDK avec Java 7
- **2014** : JavaFX 8 avec l'API Streams et Lambda

- **2018** : Séparation du JDK (OpenJFX) à partir de Java 11
- **Aujourd'hui** : Projet communautaire actif et maintenance continue

3 Quickstart

- Suivre le tutoriel officiel pour démarrer rapidement avec Maven
 - <https://openjfx.io/openjfx-docs/#maven>

4 Etudes de Cas pratiques

- Explorer le projet exemple sample-javafx-client pour mettre en pratique les concepts présentés
 - Étude du package `fxapp1` : application simple avec interface graphique simple
 - Étude du package `fxapp2` : application plus avancée utilisant FXML et le pattern MVC
- Points d'intérêt particuliers :
 - Organisation du code et séparation des responsabilités
 - Utilisation des propriétés JavaFX
 - Gestion des événements et binding
 - Comparaison des approches programmatique vs déclarative
- Lire le fichier README.md pour les instructions de compilation et d'exécution
- **Lire les explications suivantes et repérer les éléments correspondants dans le code**

5 Modules

- Dans le cas d'une application « modulaire », le plugin javafx permet de créer une distribution autonome (avec son JRE) avec :

```
./mvnw javafx:jlink
```

- Cela ne fonctionne pas si l'une des dépendances n'est pas modularisée :
 - Il faut alors utiliser les outils jdeps et jlink (cf. pom.xml de l'exemple précédent)
 - Cette commande produit un JRE réduit et une copie des modules des dépendances nécessaires dans le répertoire `target/image`
 - Exécution : `./target/image/bin/myapp`

6 Structure d'une application JavaFX

- Stage
 - Représente une fenêtre graphique
 - Contient une Scene
 - Au démarrage d'une application JavaFX, un Stage principal est automatiquement créé
- Scene
 - Conteneur racine pour tous les composants graphiques
 - Une seule Scene active par Stage à un moment donné
- Node
 - Classe abstraite dont héritent tous les composants graphiques
 - Organisés hiérarchiquement dans un graphe de scène

7 Multi-tâche dans JavaFX

- JavaFX gère au minimum deux threads distincts :
 - Thread de rendu graphique : gère l'affichage et le rendu du graphe de scène

- Thread principal de l'application : exécute la logique métier et les traitements
- Important : les modifications de l'interface graphique doivent être réalisées depuis le thread JavaFX (via `Platform.runLater()`)

8 Vues (Scene)

- Deux approches pour la création des interfaces :
 - Déclaratives en XML
 - Langage FXML
 - Écriture manuelle ou via l'éditeur graphique Scene Builder
 - Intégrable dans les IDEs comme IntelliJ IDEA
 - Programmatives en Java
 - Construction dynamique des interfaces
 - Flexibilité maximale pour les interfaces générées à l'exécution

9 Layouts et contrôles

- Layouts - conteneurs pour organiser les éléments :
 - HBox, VBox - disposition linéaire horizontale/verticale
 - FlowPane, TilePane - disposition en flux/grille
 - GridPane - disposition en cellules
 - BorderPane, StackPane, AnchorPane - dispositions spécialisées
- Contrôles - composants d'interface utilisateur :
 - Basiques : Label, Button, TextField, TextArea
 - Données : ListView, TableView, TreeView
- Conteneurs de contrôles - organisation avancée :
 - Accordion, TitledPane, TabPane
 - SplitPane, ScrollPane
- Fonctionnalités supplémentaires :
 - Multimédia : WebView, MediaView

- Graphismes : Canvas, Chart, support 2D/3D

10 Application JavaFX

- Une classe qui hérite de `javafx.application.Application`
 - `init()` : initialisation des ressources (appelée avant le thread JavaFX)
 - `start(Stage primaryStage)` : point d'entrée principal (thread JavaFX)
 - `stop()` : libération des ressources (appelée à la fermeture)
- La méthode de classe `launch(args)` démarre l'application :
 - Crée une instance de votre classe d'application
 - Initialise le toolkit JavaFX
 - Appelle séquentiellement `init()`, puis `start()` avec le Stage principal

11 JavaFX Properties

- Structure JavaFX qui implémente le design pattern Observateur/Observé
- Types de propriétés :
 - `ReadOnlyProperty` : lecture seule
 - `Property` : modifiable (lecture/écriture)
- Binding (liaison) entre propriétés :
 - Unidirectionnel ($A \rightarrow B$) : valeur de B mise à jour quand A change
 - Bidirectionnel ($A \leftrightarrow B$) : synchronisation dans les deux sens
- Cas d'usage : liaison entre modèle et vue (ex. état d'un composant graphique et variable dans le contrôleur)
- Types de propriétés disponibles :
 - Primitives : `SimpleBooleanProperty`, `SimpleDoubleProperty`, `SimpleFloatProperty`, `SimpleIntegerProperty`, `SimpleLongProperty`, `SimpleStringProperty`
 - Collections : `SimpleListProperty`, `SimpleMapProperty`, `SimpleSetProperty`, `SimpleObjectProperty`

12 Contrôleur

- Classe Java responsable de la logique et de la réactivité de l'interface
- Dans l'architecture MVC (Modèle-Vue-Contrôleur) de JavaFX :
 - Gère la communication entre le modèle de données et la vue (FXML)
 - Implémente les gestionnaires d'événements (event handlers)
 - Manipule les éléments d'interface déclarés dans le FXML
- Avec FXML, les contrôleurs sont liés via l'attribut `fx:controller`
- Les composants manipulables sont annotés avec `@FXML`
- Les gestionnaires d'événements (méthodes annotées `@FXML`) réagissent aux actions utilisateur

13 Pour aller plus loin

- Cours très complet: <https://jenkov.com/tutorials/javafx/index.html>