IUT2 Université Grenoble Alpes
Département INFO

TP01
JavaFX pour la création de scènes

JavaFX est la bibliothèque de création d'interfaces graphiques officielle du langage Java, pour toutes les sortes d'applications (applications mobiles, applications sur poste de travail, applications Web). Elle contient des composants pour les médias audio et vidéo, le graphisme 2D et 3D, la programmation Web, la programmation multi-fils etc.

L'objectif de ce TP est d'étudier 3 manières différentes de créer des interfaces JavaFX : en codant du code Java pur; en définissant l'interface dans un fichier xml; en définissant la structure de l'interface dans un fichier xml et les aspects visuels dans un fichier css.

1) Fonctionnement d'une application JavaFX

A savoir

Le point d'entrée pour les applications JavaFX est la classe Application. La fenêtre principale d'une application est représentée par un objet de type Stage qui est fourni par le système au lancement de l'application. Il faut associer à la fenêtre (Stage) une interface qui est représentée par un objet de type Scene. La scène est composée des différents éléments de l'interface graphique (composants de l'interface graphique). Cette composition est représentée par un graphe de scène (dessin à la main ou avec votre outil de dessin favori).

Chaque fois qu'une application est lancée, JavaFX exécute les actions suivantes dans l'ordre :
1. Construction d'une nouvelle instance de la classe Application
2. Appel de la méthode init () qui est appelée immédiatement après le chargement et la construction de la classe Application.
3. Appel de la méthode start (javafx.stage.Stage) qui sert à initialiser le contenu de la scène qui sera affichée dans la fenêtre Stage.
4. Attente de l'achèvement de l'application,
5. Appel de la méthode stop () qui prépare la sortie de l'application et détruit les ressources.

Les méthodes d'initialisation (init) et d'arrêt (stop) ont des implémentations concrètes qui ne font rien par défaut.

La méthode de démarrage (start) est abstraite et doit être redéfinie. Dans un premier temps, nous allons nous concentrer sur la méthode start pour créer une scène.

---

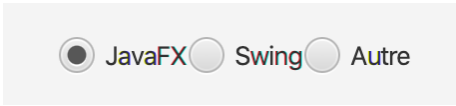
2) Création d'une première interface avec JavaFX en code Java pur

Téléchargez l'exemple de code JavaFX depuis le répertoire TP1 dans Chamilo.

Questions
1. Analysez le code de la classe exo1.java en lisant bien les commentaires.
2. Faire un dessin avec le graphe de scène correspondant à ce code.
3. Exécutez le code pour voir son fonctionnement.
4. Ajoutez entre le texte et le bouton un composant de saisie de texte, TextField
   Aide : https://openjfx.io/javadoc/18/javafx.controls/javafx/scene/control/TextField.html
5. Modifiez votre graphe de scène avec le composant de saisie de texte.

6. Ajoutez des radio boutons de manière à obtenir la présentation suivante.

Aide : pour fonctionner de manière coordonnée, les radio boutons doivent être combinés dans un groupe (ToggleGroup).
https://openjfx.io/javadoc/18/javafx.controls/javafx/scene/control/ToggleGroup.html
https://openjfx.io/javadoc/18/javafx.controls/javafx/scene/control/RadioButton.html



7. Modifiez le graphe de scène avec les composants boutons radio.

3) Création d'une première interface avec JavaFX avec Scene Builder

Pour simplifier la réalisation des interfaces, il est possible d'utiliser un outil de conception d'interfaces graphiques nommé Scene Builder. Le visuel de l'interface est créé de manière interactive. Il est encodé dans un fichier fxml qui est chargé dans la méthode start de l'Application.

Nous allons reproduire la dernière version de votre graphe de scene (interface de la partie 2) avec Scene Builder.

1. Créez un nouveau projet dans IntelliJ: nommez le projet tp01; ne modifiez pas le package. Puis sélectionnez JavaFX et prendre la **version 17** dans le panel de gauche. Dans la fenêtre relative aux librairies additionnelles, sélectionnez par défaut BootStrapFX, ControlsFX et FormsFX.

   IntelliJ a créé différents fichiers dont

   - deux fichiers HelloController.java et HelloApplication.java dans src>main>java>com.example.tp01
   - Un fichier hello-view.fxml qui contient la structure de l'interface

   Exécutez HelloApplication.

Pour l'instant, nous ne modifierons pas la classe Controller.

2. Modifiez le code de la méthode start de HelloApplication.java pour 1) empêcher le redimensionnement de la fenêtre et pour 2) modifier le nom de la fenêtre.
3. Modifiez le fichier hello-view.fxml contenu dans src>main>resources>com.example.tp01 pour réaliser la fenêtre de la partie 2. Pour faciliter le travail, vous pouvez utiliser SceneBuilder (ouvrir le fichier et sélectionner SceneBuilder sur le bas de la partie principale de la fenêtre, accepter éventuellement de télécharger Scene Builder Kit et JavaFX - attention à bien travailler en version 17). Respectez le graphe de scène de la partie 2.

   NB : la création et la modification des composants entre le .fxml et le sceneBuilder est possible dans les deux sens.

4) Exporter votre style dans un fichier séparé

Jusqu'ici, vous avez stylisé vos éléments en utilisant du code Java (partie 2) ou en utilisant Scene Builder (partie 3). Dans cette partie, vous allez exporter les styles utilisés dans un fichier css séparé.

Créer un fichier nommé « tp1.css » et placez-le dans le même package que votre fichier .fxml

Ajouter à votre scène l'instruction suivante afin qu'elle prenne en compte le fichier css créé

```
scene.getStylesheets().add(getClass().getResource("tp1.css").toExternalForm());
```

Reprenez vos styles à partir des balises de votre fichier .fxml et exportez-les .dans le fichier tp1.css en utilisant les bons sélecteurs. Par exemple, le code suivant met les labels en font Verdana en gras avec une taille de 20 :

```
Label{
    -fx-font-family: Verdana Bold;
    -fx-font-size: 20;
}
```

Il vous est recommandé d'utiliser la documentation officielle afin de trouver les différentes propriétés JavaFX (commençant par -fx) à inclure dans votre fichier css.
https://openjfx.io/javadoc/18/javafx.graphics/javafx/scene/doc-files/cssref.html