Intégration avec Swing

Intégrer JavaFX

Dans une application Swing

- Possible via le composant Swing JFXPanel

- Utilisation côté Swing

```
//dans l'EDT (thread Swing)
JFrame frame = new JFrame("Zenika");
final JFXPanel fxPanel = new JFXPanel();
frame.add(fxPanel);
frame.setSize(300, 200);
frame.setVisible(true);
frame.setDefaultCloseOperation(JFrame.EXIT_O
N_CLOSE);
```

Intégrer JavaFX

Dans une application Swing

- Utilisation côté JavaFX

```
//initialiser la scène dans l'AT JavaFX
Platform.runLater(new Runnable() {

        @Override
        public void run() {

                //créer le SceneGraph
                Group root = new  Group();
                Scene scene = new  Scene(root, Color.WHITE);

                Text text = new  Text();
                text.setText("Formation JavaFX!");
                root.getChildren().add(text);

                //ajouter la scène dans Swing
                fxPanel.setScene(scene);
        }
});
```

JavaFX et Swing

Des threads et des problèmes

- Attention : Swing et JavaFX ont tous les deux leurs threads dédiés
  - Chacun s'attend à ce que le code touchant à l'IHM soit exécuté dans son thread
  - Envelopper les appels d'un GUI Thread à un autre via des instances d'objet Runnable soumises aux méthodes dédiées de chaque framework
- Les interactions entre Swing et JavaFX apportent donc beaucoup de "boiler-plate code"

JavaFX et Swing

Manipuler les threads de chacun

- EDT pour Swing (non vérifié à l'exécution)

```
SwingUtilities.invokeLater(new Runnable() {
	public void run() {
		//...mon traitement qui impacte Swing
	}
};
```

- AT pour JavaFX (vérifié à l'exécution)

```
Platform.runLater(new Runnable() {
	public void run() {
		//...mon traitement qui impacte JavaFX
	}
};
```

Swing dans JavaFX

Introduction au SwingNode

- Depuis Java SE 8, on peut faire l'inverse et introduire des composants Swing dans une partie du SceneGraph
- SwingNode est une Node dans laquelle on peut afficher un JComponent, via la méthode setContent(JComponent swing)
- De la même manière, attention au choix du thread
  - EDT Swing ou AT JavaFX suivant les composants
  - setContent de SwingNode peut être invoqué de l'EDT comme de l'AT est représente une exception

SwingNode

Exemple

- Il suffit d'ajouter ce composant JavaFX au Scene Graph
- Celui-ci accepte des composants enfants de type Swing

```
final SwingNode swingNode = new SwingNode();
swingNode.setContent(new JLabel("Swing !"));

StackPane pane = new StackPane();
pane.getChildren().add(swingNode);

stage.setScene(new Scene(pane, 250, 150));
stage.show();
```

JavaFX + Swing

Les avantages

- Utilisation de Swing côté JavaFX
  - Permet de réutiliser des composants personnalisés existants en attendant de les migrer vers JavaFX
  - Certaines librairies Java ne fonctionnent qu'avec Swing et n'ont pas encore d'équivalent en JavaFX
- Utilisation de JavaFX côté Swing
  - Permet d'utiliser de nouveaux composants intéressants comme WebView ou Canvas
  - Initier une migration vers JavaFX en douceur

JavaFX + Swing

Les inconvénients

- La bonne gestion des threads est ardue et verbeuse
  - L'EDT et l'AT en Java FX 8 (livré avec Java SE 8) on été fusionnés pour plus de facilité
- Architecture hétérogène pouvant entrainer une perte de performance
- Est-ce durable ? Cet assemblage fonctionnera t-il dans les prochaines versions de Java / JavaFX / Swing / JVM ?

JavaFX + Swing

Fusion des threads

- Encore expérimental
- Activable via une propriété JVM

```
-Djavafx.embed.singleThread=true
```