

Interface graphique JavaFX

POO

Introduction

- Le JavaFX est un langage de programmation orienté interface graphique. Cette technologie est en faveur des applications internet riches qui se comportent sur des multiples plateformes.
- JavaFX évolue vers le multimédia à travers le traitement des images, la création de media Player, et l'intégration audio et vidéo. Ces applications ont l'avantage d'être exécuté sur le web, sur le bureau, (Desktop) ainsi que sur mobile, et même sur télévision.
- La nouvelle version JavaFX donne la possibilité de développer en deux modes :
 - Mode procédural : à travers les classes java.
 - Mode déclaratif : en utilisant la syntaxe XML.

Structure application JavaFX

- La structure des composants dans une application JavaFX est basée sur la métaphore du théâtre. Le conteneur de niveau le plus élevé est le stage, à l'intérieur du stage se trouve la scène, cette scène contient le graphe de scène, c'est-à-dire la structure hiérarchique des éléments graphiques dans l'interface.

```
Public class Main extends Application
```

Stage est la fenêtre principale indispensable pour l'affichage. C'est l'équivalent de `java.awt.Frame` en Java.

`primaryStage.show();` //afficher la fenêtre principale.

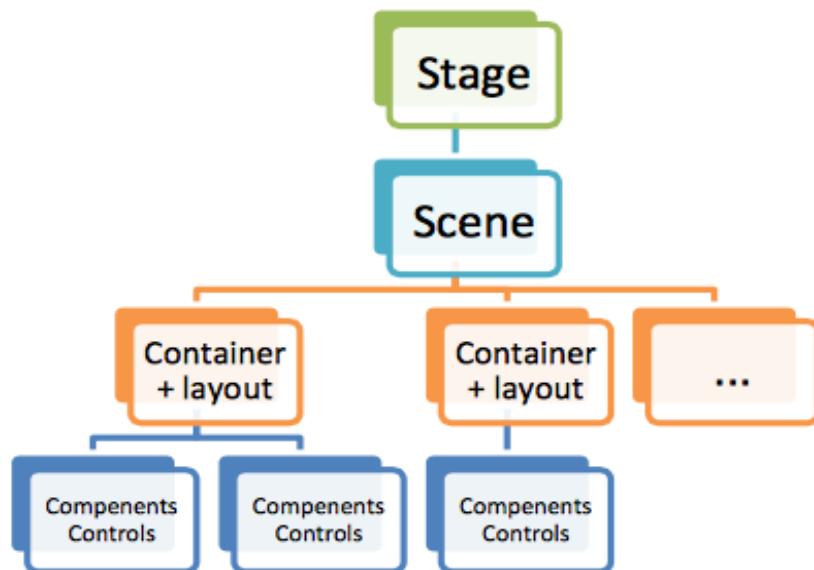
Scene contient tous les éléments de notre interface graphique. Elle compose une page dans l'application.

```
Scene scene=new Scene(root,500,400);
```

Une Scène est associée par la suite à la fenêtre principale **Stage** :

```
primaryStage.setScene(scene);
```

Nodes : ce sont les éléments visuels de la scène : les **containers** et les **composants** de JavaFX.

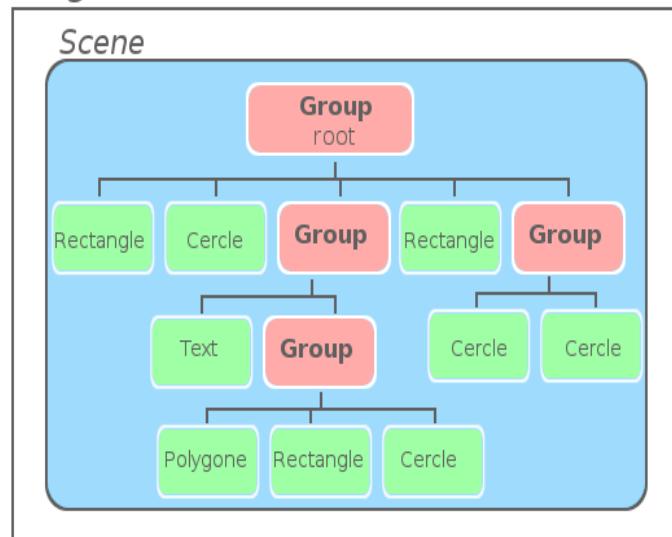


Structure application JavaFX

Une application JavaFX est un peu constituée comme une scène de théâtre.

1. Le premier élément dont elle est constituée est un objet Stage, c'est l'équivalent de notre théâtre, c'est dans cet objet que tout se passe, il représente la fenêtre de notre application.
2. A l'intérieur de cet objet Stage il y a un objet Scene, l'équivalent de la scène du théâtre. Tout ce qui apparaîtra dans notre application devra y être inséré.
3. Enfin, l'objet Scene contient des nœuds graphiques, l'équivalent des acteurs de notre pièce de théâtre. Ces nœuds graphiques sont des objets qui peuvent être de différents types : des cercles, des rectangles, des images... Ils peuvent même être des groupes de plusieurs objets graphiques.

Stage



Exemple 1

```
1 package test;
2
3 import javafx.application.Application;
4 import javafx.event.ActionEvent;
5 import javafx.event.EventHandler;
6 import javafx.scene.Group;
7 import javafx.scene.Scene;
8 import javafx.scene.control.Button;
9 import javafx.scene.paint.Color;
10 import javafx.stage.Stage;
11
12 public class Test extends Application {
13
14     public static void main(String[] args) {
15         Application.launch(Test.class, args);
16     }
17
18     @Override
19     public void start(Stage primaryStage) {
20         primaryStage.setTitle("Hello World");
21         Group root = new Group();
22         Scene scene = new Scene(root, 300, 250, Color.LIGHTGREEN);
23         Button btn = new Button();
24         btn.setLayoutX(100);
25         btn.setLayoutY(80);
26         btn.setText("Hello World");
27         btn.setOnAction(new EventHandler<ActionEvent>() {
28
29             public void handle(ActionEvent event) {
30                 System.out.println("Hello World");
31             }
32         });
33         root.getChildren().add(btn);
34         primaryStage.setScene(scene);
35         primaryStage.setVisible(true);
36     }
37 }
38 }
```

- La classe Test hérite de la classe **Application**, c'est la classe principale de notre application,
- **La fonction *main()*:** Elle appelle la fonction *launch()* qui lancera le reste du programme. C'est la seule instruction que doit contenir la fonction main().
- **La fonction *start()* :** Cette fonction est déclenchée par la fonction, elle prend en argument un objet de type Stage.

Exemple 2

```
1 public class Test extends Application {  
2  
3     public static void main(String[] args) {  
4         Application.launch(Test.class, args);  
5     }  
6  
7     @Override  
8     public void start(Stage primaryStage) {  
9  
10        primaryStage.setVisible(true);  
11    }  
12 }  
13 }
```

- Une fenêtre transparente, il n'y a même pas de fond. C'est à ça que se résume notre objet *Stage* : une fenêtre absolument vide.

Exemple 3

```
1 @Override
2     public void start(Stage primaryStage) {
3         Group root = new Group();
4         Scene scene = new Scene(root, 800, 600, Color.LIGHTBLUE);
5         primaryStage.setScene(scene);
6
7         primaryStage.setVisible(true);
8     }
9 }
```

- 1.On crée **le groupe root**, celui qui contiendra tous les objets graphiques
- 2.On crée l'objet **Scene** qui contiendra le groupe root.
- 3.On ajoute l'objet **Scene** à l'objet Stage

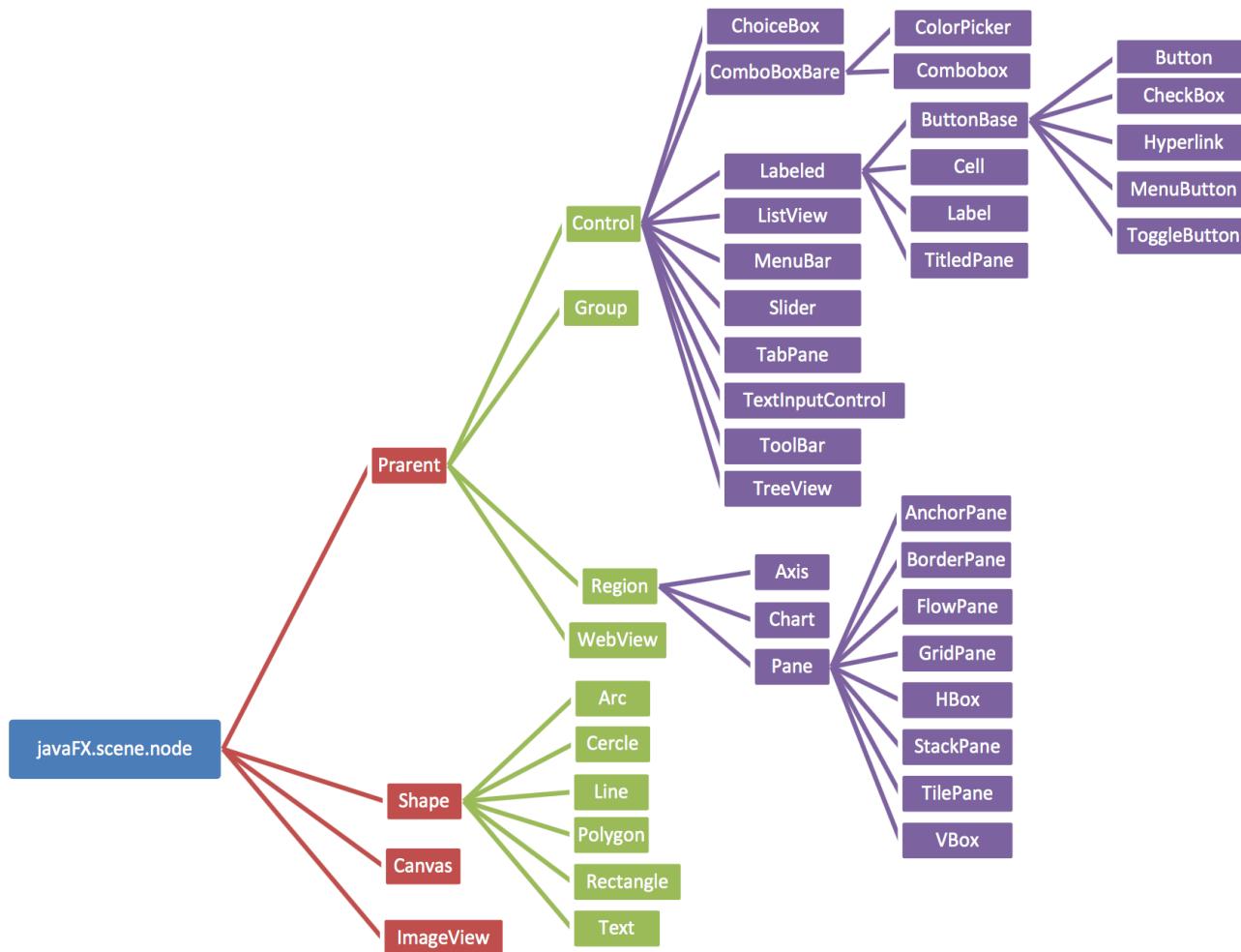
Exemple 4

```
1 @Override
2 public void start(Stage primaryStage) {
3     Group root = new Group();
4     Scene scene = new Scene(root, 800, 600, Color.LIGHTBLUE);
5     primaryStage.setScene(scene);
6
7     Circle cercle = new Circle();
8     cercle.setCenterX(300);
9     cercle.setCenterY(200);
10    cercle.setRadius(100);
11    cercle.setFill(Color.YELLOW);
12    cercle.setStroke(Color.ORANGE);
13    cercle.setStrokeWidth(5);
14
15    Rectangle rectangle = new Rectangle();
16    rectangle.setX(300);
17    rectangle.setY(200);
18    rectangle.setWidth(300);
19    rectangle.setHeight(200);
20    rectangle.setFill(Color.GREEN);
21    rectangle.setStroke(Color.DARKGREEN);
22    rectangle.setStrokeWidth(5);
23    rectangle.setArcHeight(30);
24    rectangle.setArcWidth(30);
25
26    root.getChildren().add(cercle);
27    root.getChildren().add(rectangle); //On ajoute dabord le rectangle
28    primaryStage.setVisible(true);
29 }
30 }
```

```
1 root.getChildren().add(rectangle); //On ajoute dabord le rectangle
2 root.getChildren().add(cercle); //puis le cercle
3
```

- 1.On déclare et on construit un objet de type ***Circle*** et un objet de type ***Rectangle***
- 2.On règle les paramètres de cet objet comme on le souhaite (sa couleur, sa taille, etc...),
- 3.On ajoute les objets au groupe **root** de notre objet **Scene**.

Graphe de scène

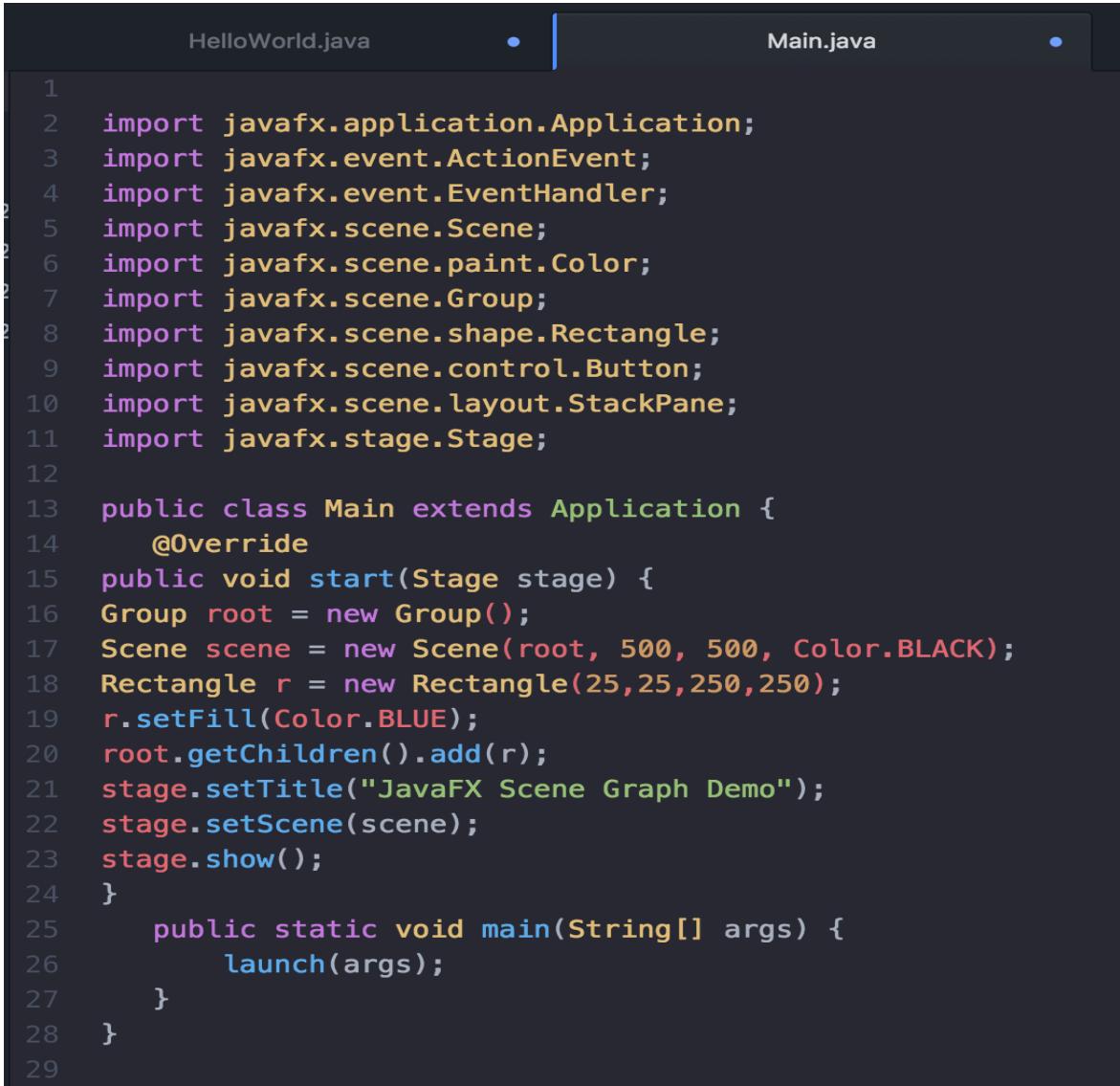


Exemple 5

The screenshot shows a code editor with three tabs: `HelloWorld.java`, `Main.java`, and `Test.java`. The `Test.java` tab is active, displaying the following Java code:

```
1 import javafx.application.Application;
2 import javafx.scene.Group;
3 import javafx.scene.Scene;
4 import javafx.scene.paint.Color;
5 import javafx.scene.text.Text;
6 import javafx.stage.Stage;
7 /*from www.javax2.com*/
8 public class Test extends Application {
9     public static void main(String[] args) {
10         Application.launch(args);
11     }
12
13     @Override
14     public void start(Stage primaryStage) {
15         primaryStage.setTitle("Drawing Text");
16         Group root = new Group();
17         Scene scene = new Scene(root, 300, 250, Color.WHITE);
18         int x = 100;
19         int y = 100;
20         int red = 30;
21         int green = 40;
22         int blue = 50;
23
24         Text text = new Text(x, y, "JavaFX 2.0");
25
26         text.setFill(Color.rgb(red, green, blue, .99));
27         text.setRotate(60);
28         root.getChildren().add(text);
29
30
31         primaryStage.setScene(scene);
32         primaryStage.show();
33     }
34 }"
```

Exemple 6



```
1 import javafx.application.Application;
2 import javafx.event.ActionEvent;
3 import javafx.event.EventHandler;
4 import javafx.scene.Scene;
5 import javafx.scene.paint.Color;
6 import javafx.scene.Group;
7 import javafx.scene.shape.Rectangle;
8 import javafx.scene.control.Button;
9 import javafx.scene.layout.StackPane;
10 import javafx.stage.Stage;
11
12
13 public class Main extends Application {
14     @Override
15     public void start(Stage stage) {
16         Group root = new Group();
17         Scene scene = new Scene(root, 500, 500, Color.BLACK);
18         Rectangle r = new Rectangle(25,25,250,250);
19         r.setFill(Color.BLUE);
20         root.getChildren().add(r);
21         stage.setTitle("JavaFX Scene Graph Demo");
22         stage.setScene(scene);
23         stage.show();
24     }
25     public static void main(String[] args) {
26         launch(args);
27     }
28 }
29
```

Exercice

- Créer une interface graphique qui contient des formes colorées superposées.