ENSET
ECOLE NORMALE SUPÉRIEURE DE L'ENSEIGNEMENT TECHNIQUE DE MOHAMMEDIA
UNIVERSITÉ HASSAN II DE CASABLANCA

Programmation Orientée Objet

Les interfaces graphiques avec JavaFX

Mme Hassna BENSAG
Email: **h.bensag@gmail.com**

JavaFX

- JavaFx est la dernière version de bibliothèques qui permet de créer des interfaces graphiques de qualité pour les applications Java Desktob, Mobile et Web.
- Avec l'apparition de Java 8 en mars 2014, JavaFX devient la bibliothèque de création de l'interface graphique officielle du langage Java .
- Le développement de son précédent SWING étant abandonné

Structure d’une application JavaFX

- Les éléments de base d’une application JavaFX:
  - **Stage**: réprésente la fenêtre principale de l’application.
  - **Scene**: le composant qui permet d’afficher tout ce qui devrait apparaitre dans l’application.
  - **Node**: les éléments ou bien les composantes graphiques de la Scene.

Lancement d’une fenêtre

- la classe principale est toujours une application
- La méthode **start(),** est automatiquement appelée par le framework JavaFX lorsque l’application démarre:
  1) l’objet de type Application est instancié par l’environnement
  2) L’environnement JavaFX fournie une fenêtre de type Stage à la méthode start(Stage primaryStage)

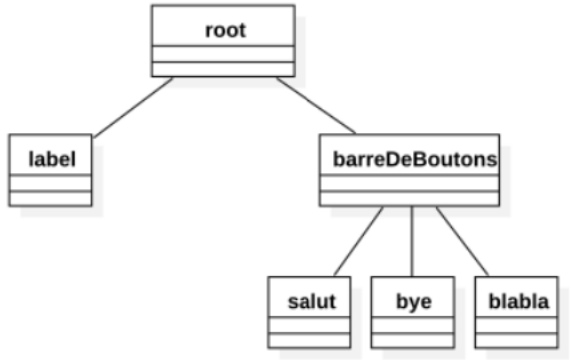
GuiApplication.java

```
1  import javafx.application.Application;
2  import javafx.stage.Stage;
3
4  public class GuiApplication extends Application {
5
6      @Override
7      public void start(Stage stage) throws Exception {
8          stage.setTitle("Liste Etudiants");
9          stage.show();
10     }
11
12     public static void main(String[] args) {
13         Application.launch(args);
14     }
15 }
```

L’arbre de Scene

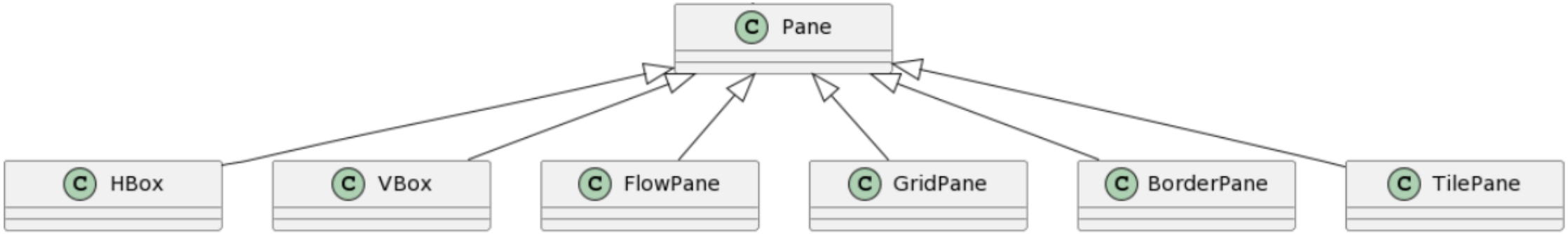
- Les éléments de scène sont organisés sous forme d’un arbre:
  - Un objet de type Node est désigné comme la **racine**
  - Des objets Node **fils intermédiaires :** des conteneurs regroupant plusieurs composants
  - Des objets Node **feuilles** qui n’ont aucun descendant: boutons, champs de saisie ou texte, formes graphiques

```
public void start(Stage stage) throws Exception {
    BorderPane root = new BorderPane();
    Label label = new Label( s: "Salut le monde !");
    HBox bareDeBoutons = new HBox();
    Button salut = new Button( s: "Salut");
    Button bye = new Button( s: "Bye");
    Button blabla = new Button( s: "Raconte une histoire");
    bareDeBoutons.getChildren().addAll(salut, bye, blabla);
    root.setLeft(label);
    root.setRight(bareDeBoutons);
    Scene scene = new Scene(root,  v: 420,  v1: 100);
    stage.setTitle("Ma fenetre!");
    stage.setScene(scene);
    stage.show();
}
```



Conteneurs

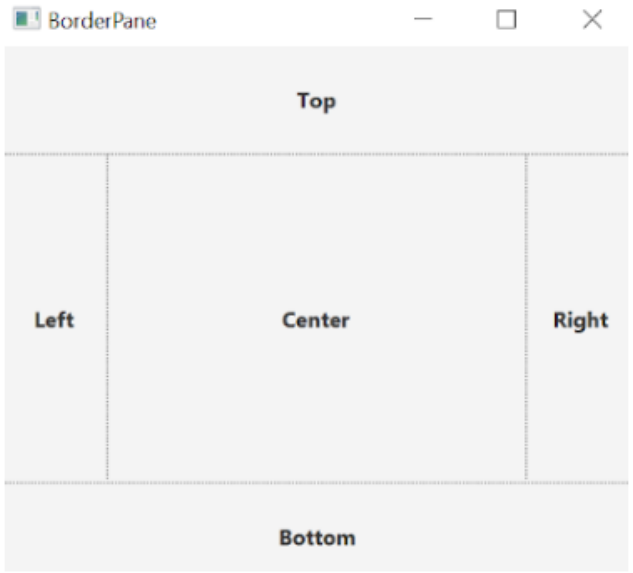
- Les conteneurs(Layout) sont des nœuds qui permettent d'indiquer l'organisation des composants sur la scène.



Conteneurs - BorderPane

```java
@Override
public void start(Stage primaryStage) throws Exception {
    BorderPane root= new BorderPane();
    Button top=new Button( s: "Top");
    top.setMaxWidth(Double.MAX_VALUE);
    root.setTop(top);
    Button bottom=new Button( s: "Bottom");
    bottom.setMaxWidth(Double.MAX_VALUE);//pour que le boutton occupe toute la largeur
    root.setBottom(bottom);
    Button left= new Button( s: "Left");
    left.setMaxHeight(Double.MAX_VALUE);//pour que le boutton occupe toute l'hauteur
    root.setLeft(left);
    Button right=new Button( s: "Right");
    right.setMaxHeight(Double.MAX_VALUE);
    root.setRight(right);
    Button center=new Button( s: "Center");
    center.setMaxHeight(Double.MAX_VALUE);
    center.setMaxWidth(Double.MAX_VALUE);
    root.setCenter(center);
    Scene scene= new Scene(root, v: 300, v1: 300);
    primaryStage.setScene(scene);
    primaryStage.setTitle("BorderPane");
    primaryStage.show();
}
```

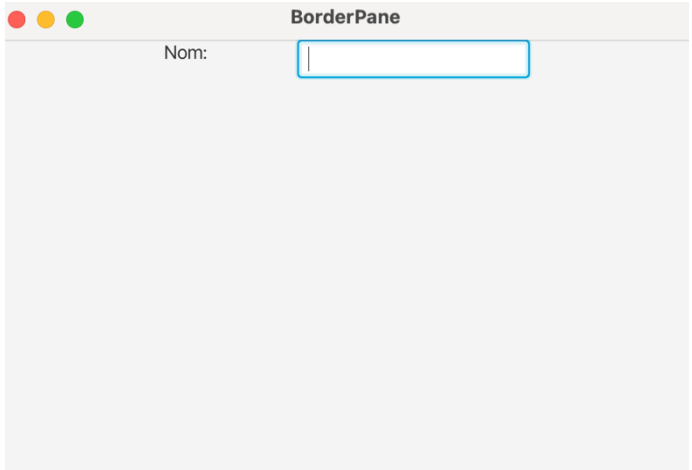
- Un conteneur divisé en 5 zones :Top, Bottom, Left, Right, Center
- Chaque zone peut contenir un seul nœud



Conteneurs – Hbox et VBox

- Hbox
  - placement des composants sur une ligne horizontale, de gauche à droite
  - des fonctions permettent d'adapter le conteneur : setAlignment(), setMinWidth(), setSpacing(), etc.
- Vbox: idem que Hbox mais en vertical

```
        HBox hbox=new HBox();
        hbox.setSpacing(60);
        hbox.setAlignment(Pos.TOP_CENTER);
        Label nom= new Label( s: "Nom: ");
        TextField textname= new TextField();
        hbox.getChildren().addAll(nom,textname);
        Scene scene= new Scene(hbox, v: 300, v1: 300);
        primaryStage.setScene(scene);
        primaryStage.setTitle("BorderPane");
        primaryStage.show();
    }
```

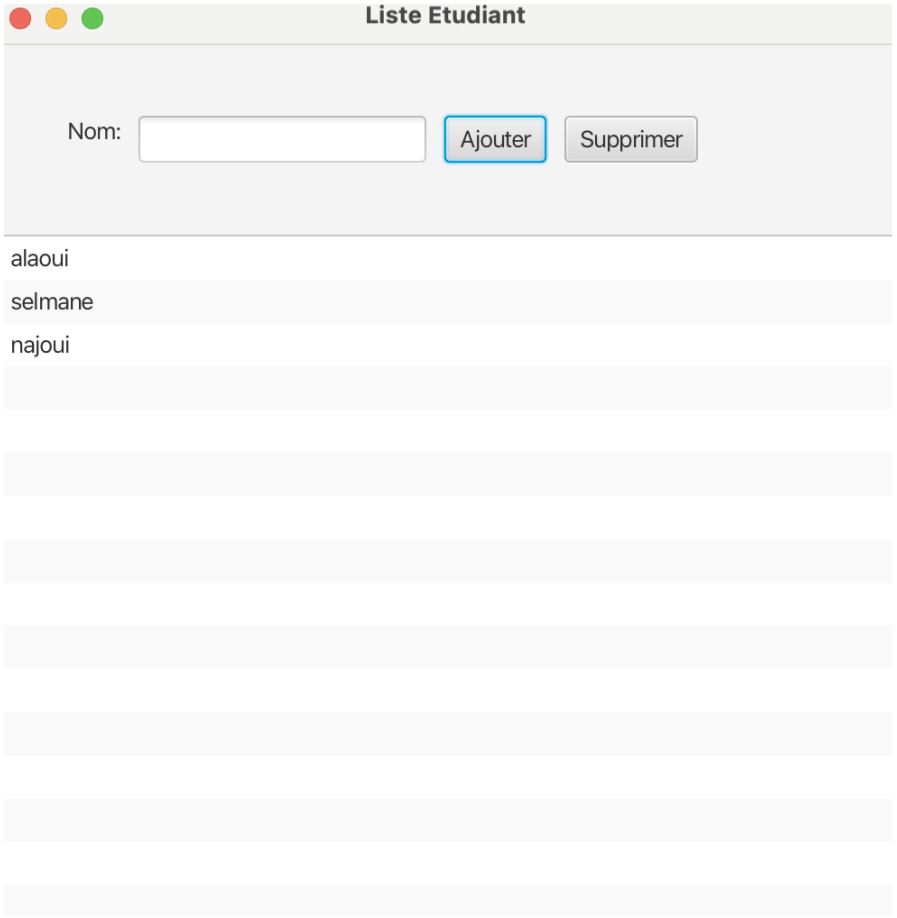


Conteneurs

- GridPane: permet de créer une grille d'élément organisés en lignes et colonnes
- TilePane: organise les éléments dans une grille. Contrairement à GridPane où vous devez spécifier des lignes et des colonnes précises, le TilePane dispose ses éléments automatiquement dans des "tuiles" (tiles) qui ont la même taille.
- FlowPane:Les éléments sont disposés sur une ligne (horizontale ou verticale) lorsque il n'y a plus assez de place disponible, on passe à la ligne suivante
- StackPane: est un conteneur , qui permet de ranger les éléments de façon à ce que chaque nouvel élément inséré apparaisse au-dessus de tous les autres.

Exemple – Liste d'étudiants

- Un exemple d'application JavaFx, qui permet:
  - d'ajouter les noms des étudiants à une **ListView**
  - Supprimer le nom selectionné depuis **la ListView**



GUI à travers FXML

- Le FXML est un langage basé sur XML pour construire des interfaces graphiques JavaFX.
- Un fichier .fxml décrit la structuration du graphe de scène, les propriétés graphiques de chaque composant (taille, police, couleur etc.)
- Le code FXML est modifiable indépendamment du code métier de l'application
- Pour programmer les réponses aux événements produits dans les composants, on crée un Contrôleur associé à la vue FXML.
- Il est préconisé d'utiliser des outils interactifs de création d'interfaces graphiques FXML: SceneBuilder

Exemple – Liste d'étudiants avec FXML

- Créer la view list.fxml à travers SceneBuilder

list-view.fxml

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

<?import javafx.geometry.*?>
<?import javafx.scene.control.*?>
<?import javafx.scene.layout.*?>

<BorderPane maxHeight="1.7976931348623157E308" maxWidth="1.7976931348623157E308" prefHeight="400.0" prefWidth="600.0"
            xmlns="http://javafx.com/javafx/17.0.12" xmlns:fx="http://javafx.com/fxml/1" fx:controller="com.example.demo.ListEtudiantController">
   <top>
      <HBox prefHeight="100.0" prefWidth="200.0" spacing="20.0" BorderPane.alignment="CENTER">
         <children>
            <Label text="Nom:" />
            <TextField fx:id="textFieldName" />
            <Button mnemonicParsing="true" onAction="#OnAddClick" text="Ajouter" />
            <Button mnemonicParsing="false" onAction="#OnDeleteClick" text="Supprimer" />

         </children>
         <padding>
            <Insets bottom="30.0" left="30.0" right="30.0" top="30.0" />
         </padding>
      </HBox>
   </top>
   <center>
      <ListView fx:id="listEtudiant" prefHeight="200.0" prefWidth="200.0" BorderPane.alignment="CENTER" />
   </center>
</BorderPane>
```

Exemple – Liste d'étudiants avec FXML

- Créer un Controller ListEtudiantController:

ListEtudiantController.java

```java
package com.example.demo;

import ...

public class ListEtudiantController {
    @FXML
    private ListView listEtudiant;
    @FXML
    private TextField textFieldName;
    @FXML
    protected void OnAddClick(){
        String nom=textFieldName.getText().trim();
        if(!nom.isEmpty()){
            listEtudiant.getItems().add(textFieldName.getText());
            textFieldName.clear();}
        else{
            Alert alert=new Alert(Alert.AlertType.WARNING);
            alert.setTitle("Champs Vide");
            alert.setContentText("Veuillez saisir un nom avant d'ajouter");
            alert.showAndWait();
        }
    }
    @FXML
    protected void OnDeleteClick(){
        int index=listEtudiant.getSelectionModel().getSelectedIndex();
        if(index>=0){
            listEtudiant.getItems().remove(index);
            listEtudiant.getSelectionModel().clearSelection();
        }else{
            Alert alert=new Alert(Alert.AlertType.WARNING);
            alert.setTitle("Champs Non Selectionné");
            alert.setContentText("Veuillez selectioner un nom avant de supprimer");
            alert.showAndWait();
        }
    }
```

Exemple – Liste d’étudiants avec FXML

- Créer l’application JAVA

ListEtudiantApp.java

```java
package com.example.demo;

import javafx.application.Application;
import javafx.fxml.FXMLLoader;
import javafx.scene.Scene;
import javafx.stage.Stage;

public class ListEtudiantApp extends Application {
    @Override
    public void start(Stage stage) throws Exception {
        FXMLLoader fxmlloader= new FXMLLoader(ListEtudiantApp.class.getResource( name: "list-view.fxml"));
        Scene scene=new Scene(fxmlloader.load(), v: 600, v1: 600);
        stage.setScene(scene);
        stage.setTitle("List Etudiants");
        stage.show();
    }

    public static void main(String[] args) {
        launch();
    }
}
```