Metode avansate de programare

Informatică Româna, 2017-2018, Curs

Curs 7

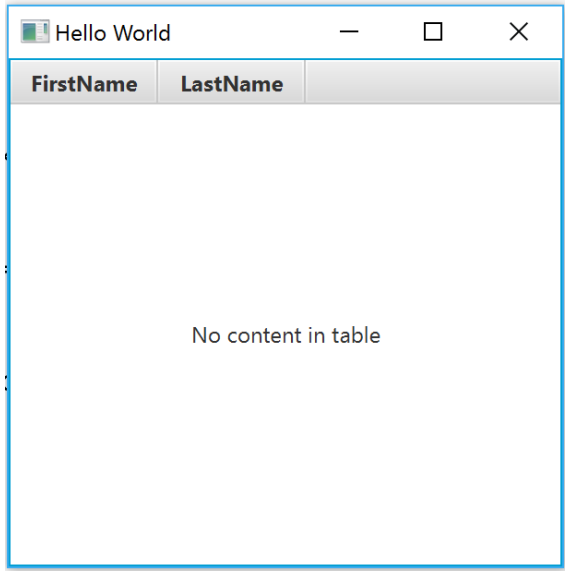
Continuare curs anterior

FXML

Continuare curs anterior TableView

- Creare

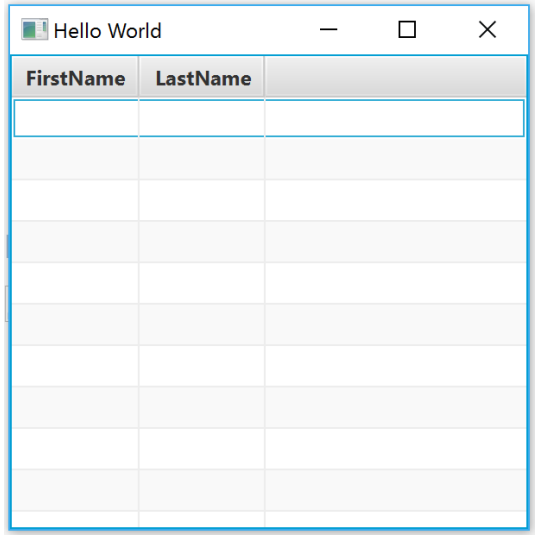
```
TableView<Student> tableView=new TableView<Student>();

TableColumn<Student,String> columnName=new TableColumn<>("FirstName");
TableColumn<Student,String> columnLastName=new TableColumn<>("LastName");


tableView.getColumns().addAll(columnName,columnLastName);
```



TableView

- Binding data

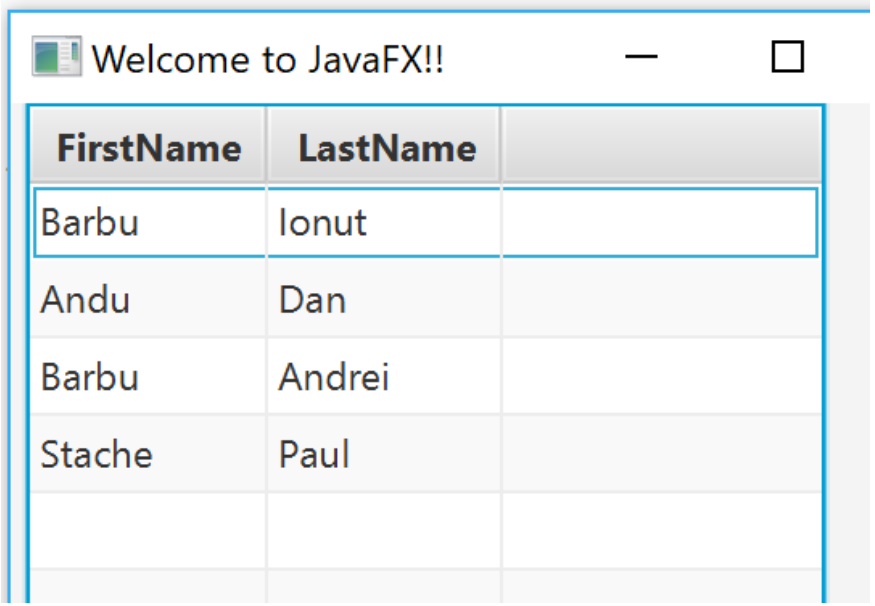
```
List<Student> l=new ArrayList<Student>();
l.add(new Student("Barbu","Ionut","aaa@yahoo.com"));
l.add(new Student("Andu","Dan","dd@yahoo.com"));
l.add(new Student("Stan","Apostol","dsss@yahoo.com"));
l.add(new Student("Stache","Paul","aads@yahoo.com"));

ObservableList<Student> students = FXCollections.observableArrayList(l);

TableView<Student> tableView=new TableView<Student>();

TableColumn<Student,String> columnName=new TableColumn<>("FirstName");
TableColumn<Student,String> columnLastName=new TableColumn<>("LastName");
tableView.getColumns().addAll(columnName,columnLastName);

tableView.setItems(students);
```



TableView

- setCellValueFactory method (render data)

```
columnName.setCellValueFactory(new PropertyValueFactory<Student, String>("firstName"));
columnLastName.setCellValueFactory(new PropertyValueFactory<Student, String>("lastName"));
```



TableView

- Listen for table selection changes

```
tableView.getSelectionModel().selectedItemProperty().addListener(new
ChangeListener<Student>() {
    @Override
    public void changed(ObservableValue<? extends Student> observable, Student
oldValue, Student newValue) {
        System.out.println("A fost selectat"+ newValue.toString());
    }
});
```

Adnotari Java – foarte pe scurt

- Au fost introduse începând cu versiunea 1.5
- Adnotările sunt o formă de a încorpora metadata în codul Java. Ele nu au impact asupra codului.
- Scop:
  - Oferă imformații suplimentare compilatorului (pot fi folosite de compilator pt a detecta erori);
  - Procesare automata din timpul compilării sau deploymentului. Instrumente soft specializate pot folosi adnotarile pentru a genera automat cod, fișiere XML, etc.
  - Procesare în timpul execuției. Unele adnotări sunt disponibile pentru a fi examinate în timpul execuției codului.
- http://www.vogella.com/tutorials/JavaAnnotations/article.html

Exemplu adnotări predefinite

```
@FunctionalInterface
interface A{
    void f();
    void g();
}
```

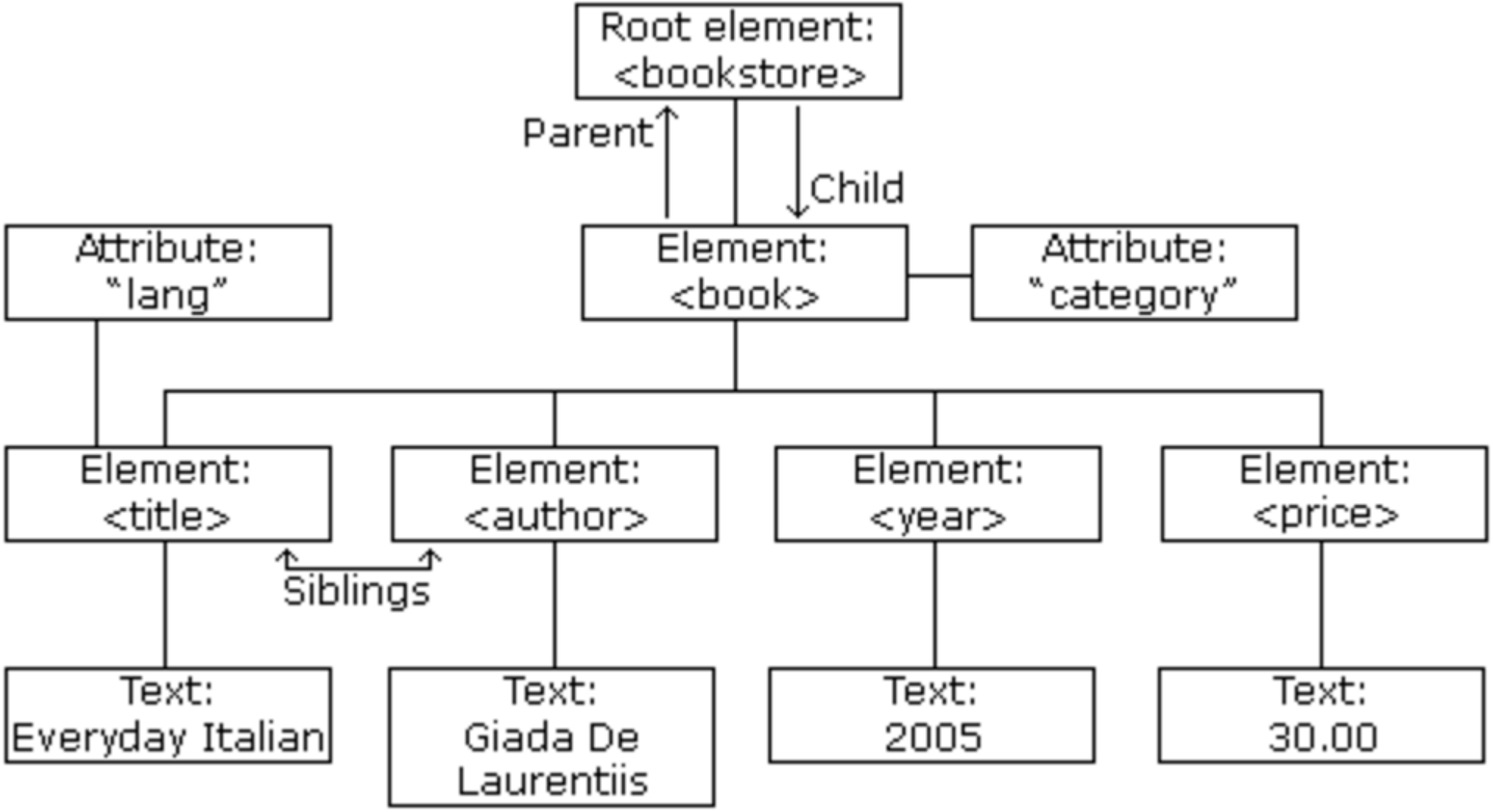
- Eroare la compilare – două metode abstracte într-o interfață ce ne dorim a fi de tipul SAM (Single Abstract Method)

XML

- XML stands for EXtensible Markup Language – limbaj descriptiv
- XML a fost conceput pentru a stoca și a transporta date.

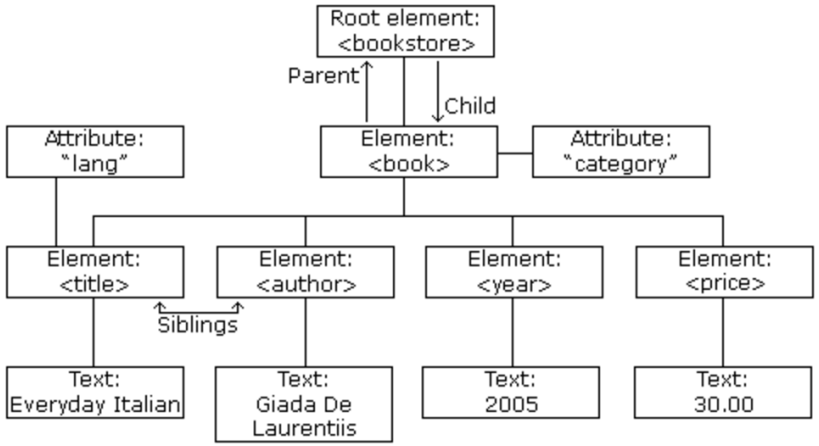
```
<note>
  <to>Tove</to>
  <from>Jani</from>
  <heading>Reminder</heading>
  <body>Don't forget me this
weekend!</body>
</note>
```

XML Tree Structure



Exemplu

```xml
<?xml version="1.0" encoding="UTF-8"?>
<bookstore>
  <book category="cooking">
    <title lang="en">Everyday Italian</title>
    <author>Giada De Laurentiis</author>
    <year>2005</year>
    <price>30.00</price>
  </book>
  <book category="children">
    <title lang="en">Harry Potter</title>
    <author>J K. Rowling</author>
    <year>2005</year>
    <price>29.99</price>
  </book>
  <book category="web">
    <title lang="en">Learning XML</title>
    <author>Erik T. Ray</author>
    <year>2003</year>
    <price>39.95</price>
  </book>
</bookstore>
```



FXML

- FXML este un limbaj declarativ de adnotare bazat pe XML prin intermediul căruia pot fi dezvoltate interfeţe grafice cu utilizatorul, fără a fi necesar ca aplicaţia să fie recompilată de fiecare dată când sunt modificate elemente din cadrul acesteia.
- În acest mod se realizează o separare între nivelul de prezentare și nivelul de logică a aplicației.
- SceneBuilder permite construirea interfeţei în mod vizual, generând automat şi documentul FXML asociat, acesta putând fi integrat apoi în orice mediu de dezvoltare.
- Astfel, nu mai este necesară decât implementarea mecanismelor de tratare a evenimentelor corespunzătoare diferitelor controale (elemente din cadrul interfeței grafice);

Programatic vs. Declarativ

- *Programatic*

```
BorderPane border = new BorderPane();
Label top = new Label("Page Title");
border.setTop(top);
Label center = new Label ("Some data here");
border.setCenter(center);
```

- *Declarativ*

```
<BorderPane>
    <top>
        <Label text="Page Title"/>
    </top>
    <center>
        <Label text="Some data here"/>
    </center>
</BorderPane>
```

View definit ca fișier FXML

- *Exemplu fereastra de autentificare (login)*

Definim un GridPane pe care il vom adăuga unui AnchorPane

Exemplu Login FXML

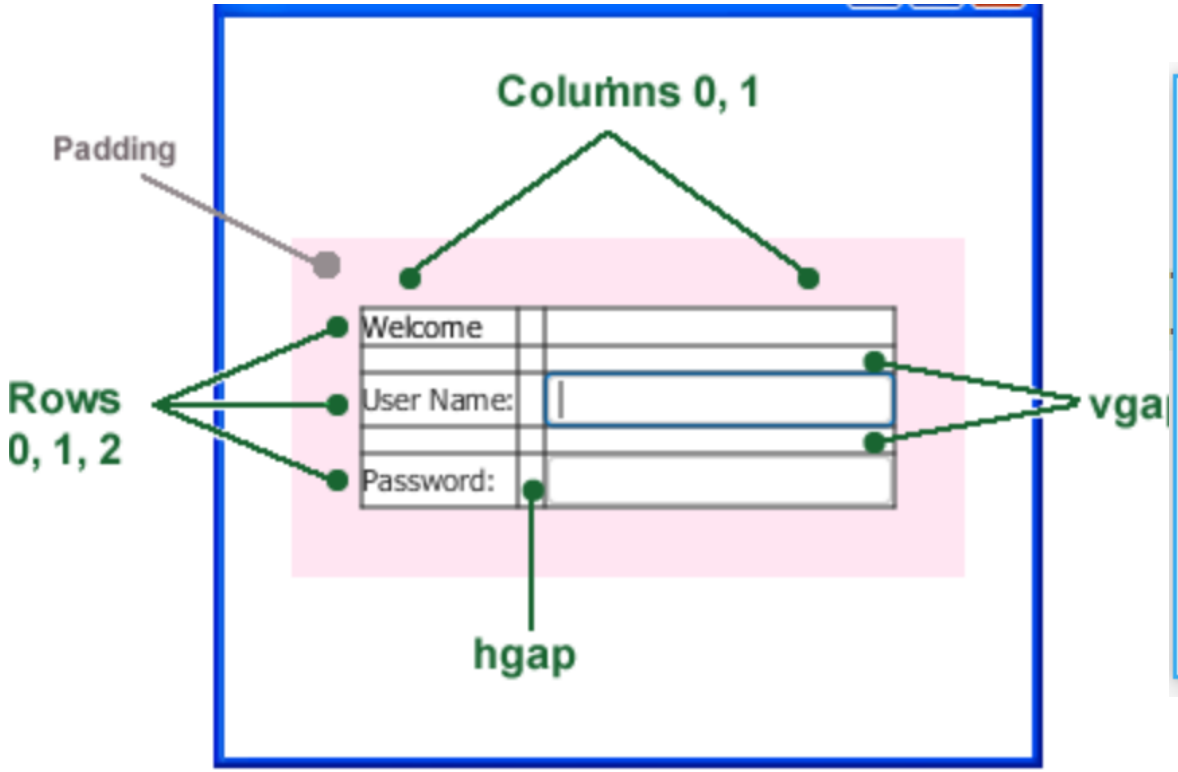
- Exemplu fereastra de autentificare (login)

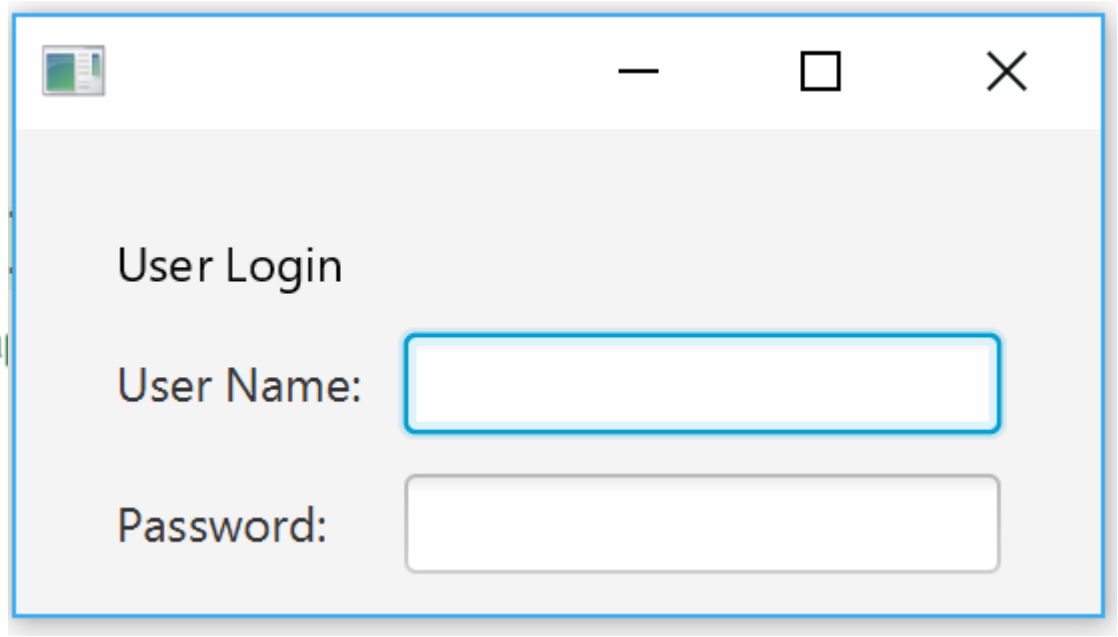
```
<GridPane xmlns:fx="http://javafx.com/fxml" alignment="center" hgap="10" vgap="10">
    <padding><Insets top="25" right="25" bottom="10" left="25"/></padding>

    <Text text="User Login "
          GridPane.columnIndex="0" GridPane.rowIndex="0"
          GridPane.columnSpan="2"/>

    <Label text="User Name:"
           GridPane.columnIndex="0" GridPane.rowIndex="1"/>

    <TextField
            GridPane.columnIndex="1" GridPane.rowIndex="1"/>

    <Label text="Password:"
           GridPane.columnIndex="0" GridPane.rowIndex="2"/>

    <PasswordField GridPane.columnIndex="1" GridPane.rowIndex="2"/>
</GridPane>
```

```
//cod JavaFX
GridPane gr=new GridPane();
//alignment="center" hgap="10" vgap="10"
gr.setAlignment(Pos.CENTER);
gr.setHgap(10);
gr.setVgap(10);
Text t=new Text("User Login ");
gr.add(t,0,0);

Label l=new Label("User Login ");
gr.add(l,0,1);
```

Exemplu Login FXML

```
<padding><Insets top="25" right="25" bottom="10" left="25"/></padding>
```





Exemplu Login FXML

- *Adaugare Buton*

```
<HBox spacing="10" alignment="bottom_right"
      GridPane.columnIndex="1" GridPane.rowIndex="4">
    <Button text="Sign In"/>
</HBox>
```

FXML Loader

```
public class Main extends Application {
    public static void main(String[] args) {
        launch(args);
    }
    @Override
    public void start(Stage primaryStage) {
        try {
            //Load root layout from fxml file.
            FXMLLoader loader=new FXMLLoader();
            loader.setLocation(getClass().getResource("LoginExample.fxml")); //URL
            GridPane rootLayout= (GridPane) loader.load();

            // Show the scene containing the root layout.
            Scene scene = new Scene(rootLayout);
            primaryStage.setScene(scene);
            primaryStage.show();

        } catch (IOException e) {
            e.printStackTrace();
        }
    }
}
```

FXML - Controller

```
<GridPane fx:controller="Exemplu.LoginExampleController“ xmlns:fx="http://javafx.com/fxml" alignment="center" hgap="10" vgap="10">
```

- In fisierul XXX.fxml ne definim view-l
- Actiunile utilizator (evenimentele) le tratam intr-un fisier Controller
- Cum?
  - Definim un fisier java, de exemplu cu numele XXXController.java
  - Specificam legatura cu fisierul XXX.fxml:

    ```
    <GridPane  fx:controller="Exemplu.LoginExampleController“>
    ```

  - Definim metode handlere in `XXXController.java` pentru tratarea evenimentelor

Obtinerea unui obiect de tip controller

```
public class Main1 extends Application {
    public static void main(String[] args) {
        launch(args);
    }
    @Override
    public void start(Stage primaryStage) {
        try {
            //Load root layout from fxml file.
            FXMLLoader loader=new FXMLLoader();
            loader.setLocation(getClass().getResource("LoginExample.fxml")); //URL
            GridPane rootLayout= (GridPane) loader.load();
            LoginExampleController controller=loader.getController();

            // Show the scene containing the root layout.
            Scene scene = new Scene(rootLayout);
            primaryStage.setScene(scene);
            primaryStage.show();
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
}
```

Tratarea evenimentelor

- *Handle Event via Controller class*

```
<HBox spacing="10" alignment="bottom_right“ GridPane.columnIndex="1" GridPane.rowIndex="4">
        <Button text="Sign In“ onAction="#handleSubmitButtonAction"/>
</HBox>
<Text GridPane.columnIndex="1" GridPane.rowIndex="6"/>
```

```
public class LoginExampleController {


    @FXML
    public void handleSubmitButtonAction(ActionEvent actionEvent) {
        System.out.println("Login button was pressed!");
    }
}
```

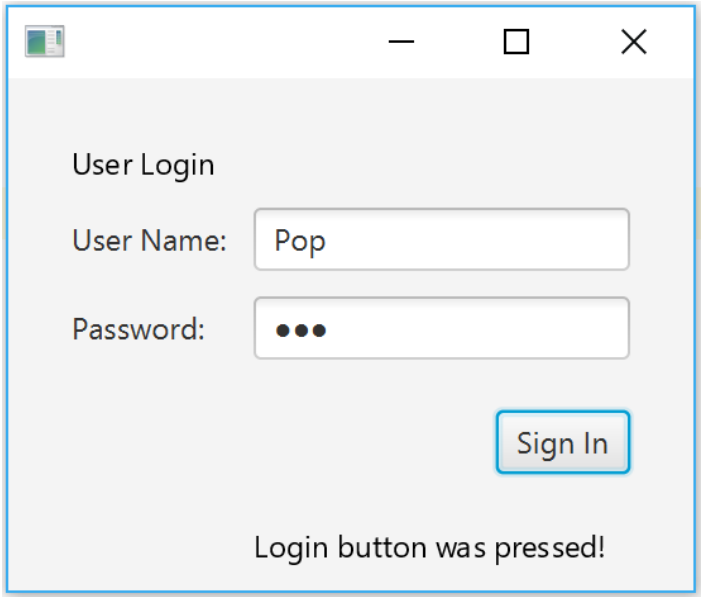
FXML – Controller initialize

```
public class LoginExampleController {
/**
     * Initializes the controller class. This method is automatically called
     * after the fxml file has been loaded.
     */
    @FXML
    public void initialize() {

    }
```

Adnotarea FXML a elementelor din view

```
<TextField fx:id="usernameField"
        GridPane.columnIndex="1" GridPane.rowIndex="1"/>

<PasswordField fx:id="passwordField" GridPane.columnIndex="1" GridPane.rowIndex="2"/>

<HBox spacing="10" alignment="bottom_right“ GridPane.columnIndex="1" GridPane.rowIndex="4">
         <Button text="Sign In“ onAction="#handleSubmitButtonAction"/>
</HBox>
<Text fx:id="textResponse“  GridPane.columnIndex="1" GridPane.rowIndex="6"/>
```

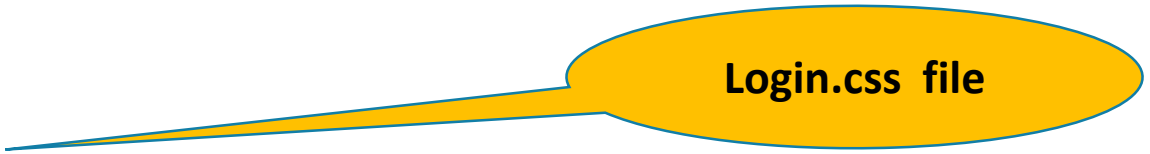
```
public class LoginExampleController {
    @FXML
    private Text textResponse;
    @FXML
    private TextField usernameField;
    @FXML
    private PasswordField passwordField;
    @FXML
    public void handleSubmitButtonAction(ActionEvent actionEvent) {
        textResponse.setText("Login button was pressed!");
        User u=new User(usernameField.getText(),passwordField.getText());
    }
}
```



CSS

```
<GridPane stylesheets="@login.css"  fx:controller="Exemplu.LoginExampleController"
          xmlns:fx="http://javafx.com/fxml" alignment="center" hgap="10" vgap="10">
```

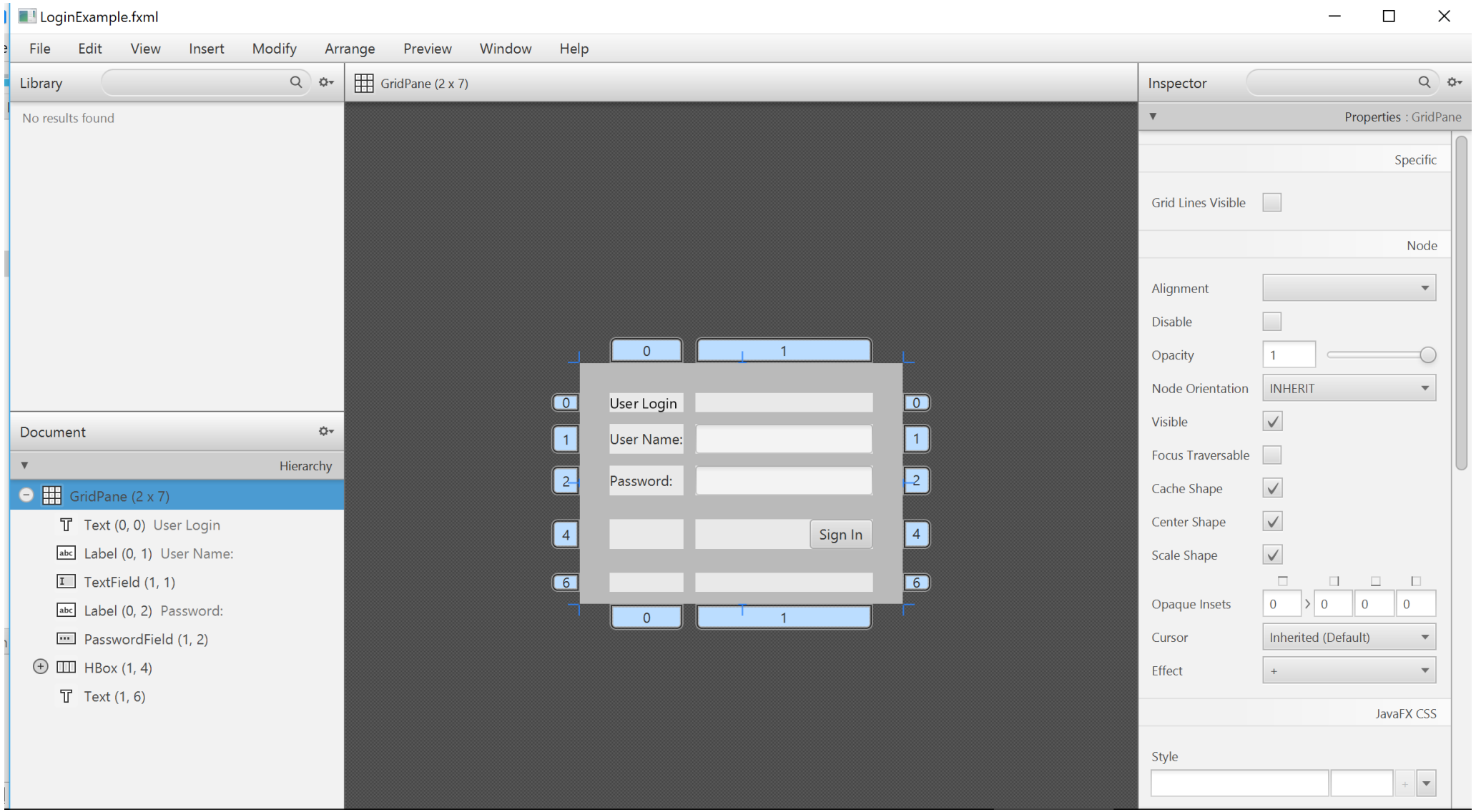
```
.root {
    -fx-background-image: url("logo.gif");
}
.button {
    -fx-text-fill: white;
    -fx-font-family: "Arial Narrow";
    -fx-font-weight: bold;
    -fx-background-color: linear-gradient(#61a2b1, #2A5058);
    -fx-effect: dropshadow( three-pass-box , rgba(0,0,0,0.6) , 5, 0.0 , 0 , 1 );
}
.label {
    -fx-font-size: 12px;
    -fx-font-weight: bold;
    -fx-text-fill: #2A5058;
    -fx-effect: dropshadow( gaussian , rgba(214, 66, 20, 0.5), 0,0,0,1 );
}
#logintext{
    -fx-font-size: 32px;
    -fx-font-family: "Arial Black";
    -fx-fill: #2A5058;
}
#textResponse {
    -fx-fill: FIREBRICK;
    -fx-font-weight: bold;
    -fx-effect: dropshadow( gaussian , rgba(255,255,255,0.5) , 0,0,0,1 );
}
```



http://www.w3schools.com/css/

FXML and Scene Builder

- http://www.oracle.com/technetwork/java/javase/downloads/sb2download-2177776.html
- http://gluonhq.com/labs/scene-builder/

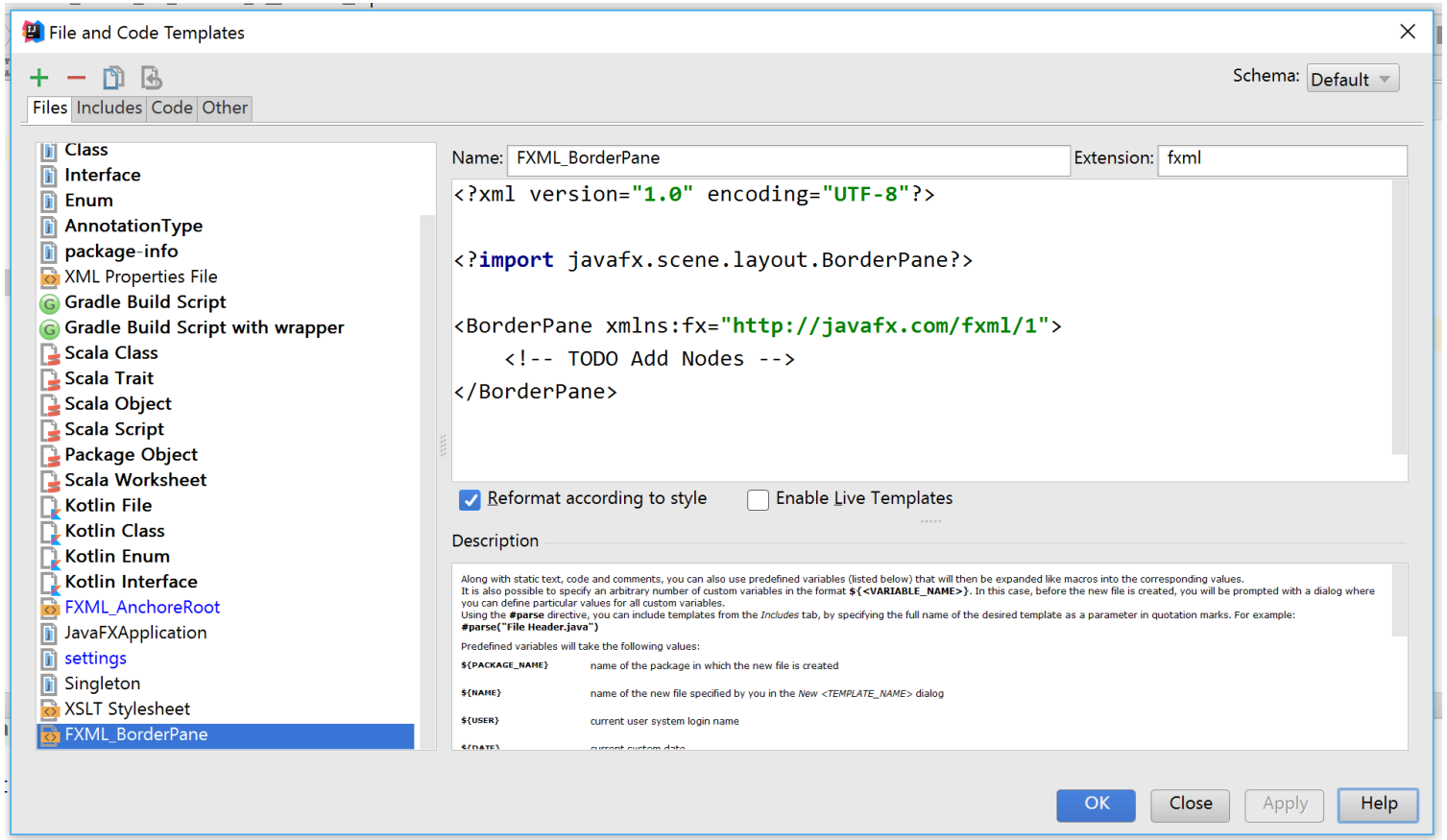


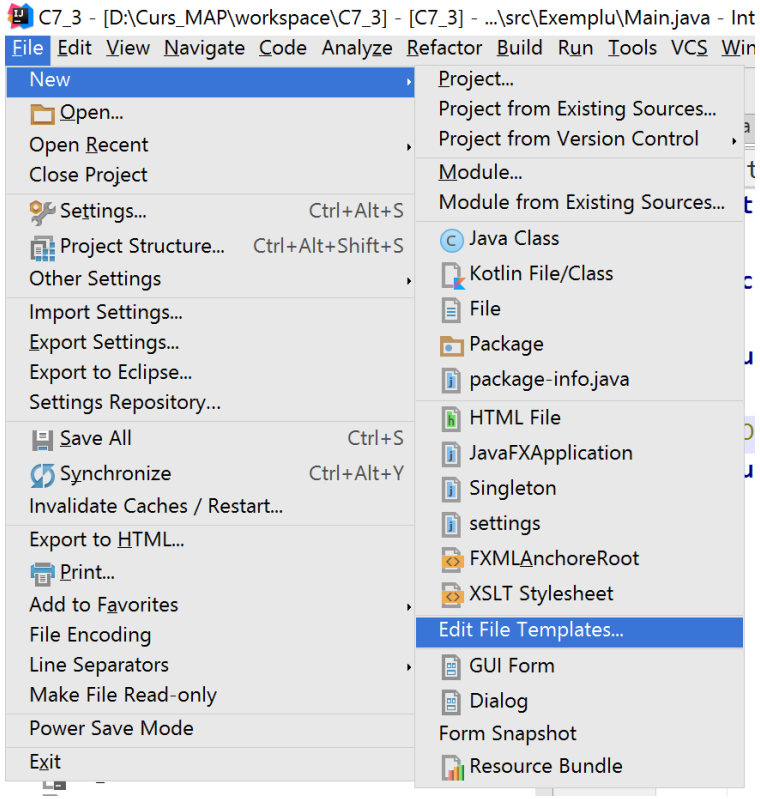
Scene Builder

- Specifying the path to the JavaFX Scene Builder executable:
- In Ecplise:
- Window -> Preferences ->Scene Builder
- In InteliJ
- File->Settings-Languages and Frameworks->Java FX
- Scene Builder download:
- http://docs.oracle.com/javafx/scenebuilder/1/use_java_ides/sb-with-eclipse.htm
- http://gluonhq.com/labs/scene-builder/#download

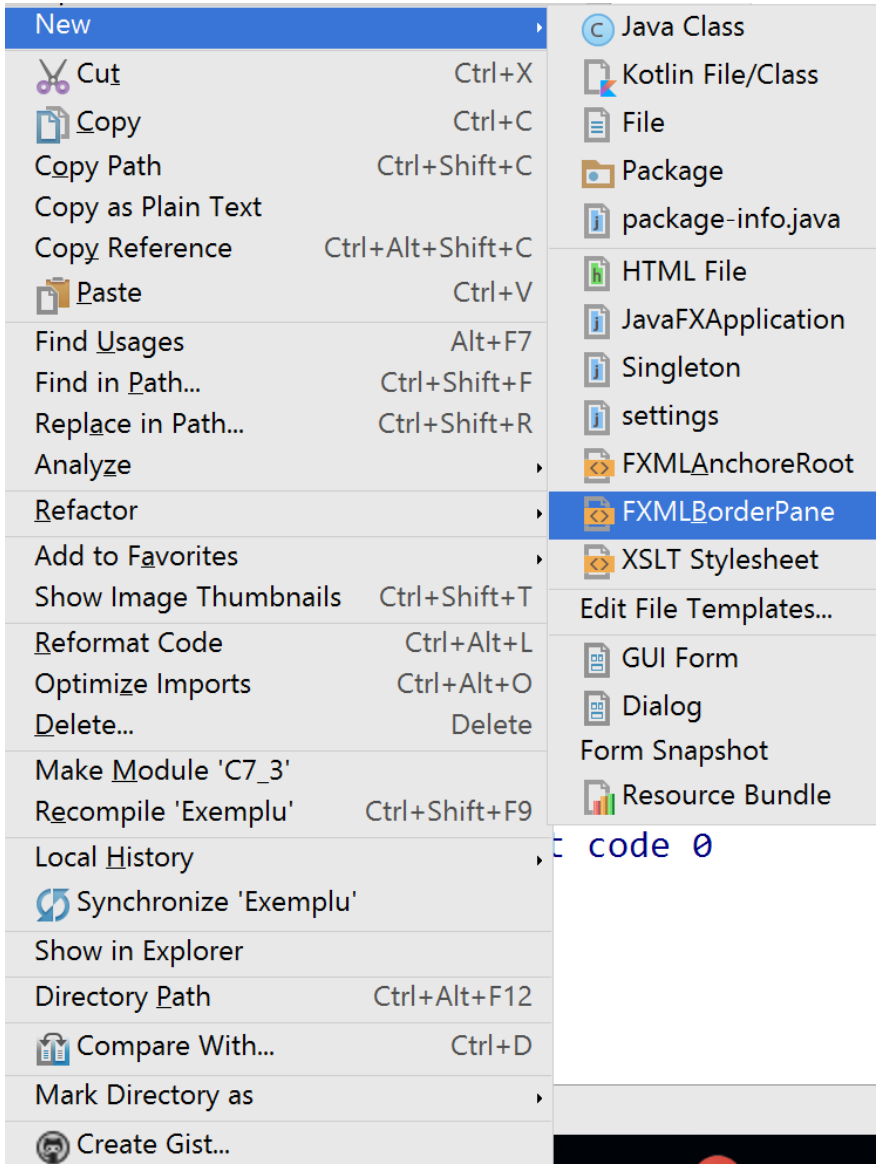
FXML File templates

- In Eclipse exista predefinite
- In InteliJ definim noi



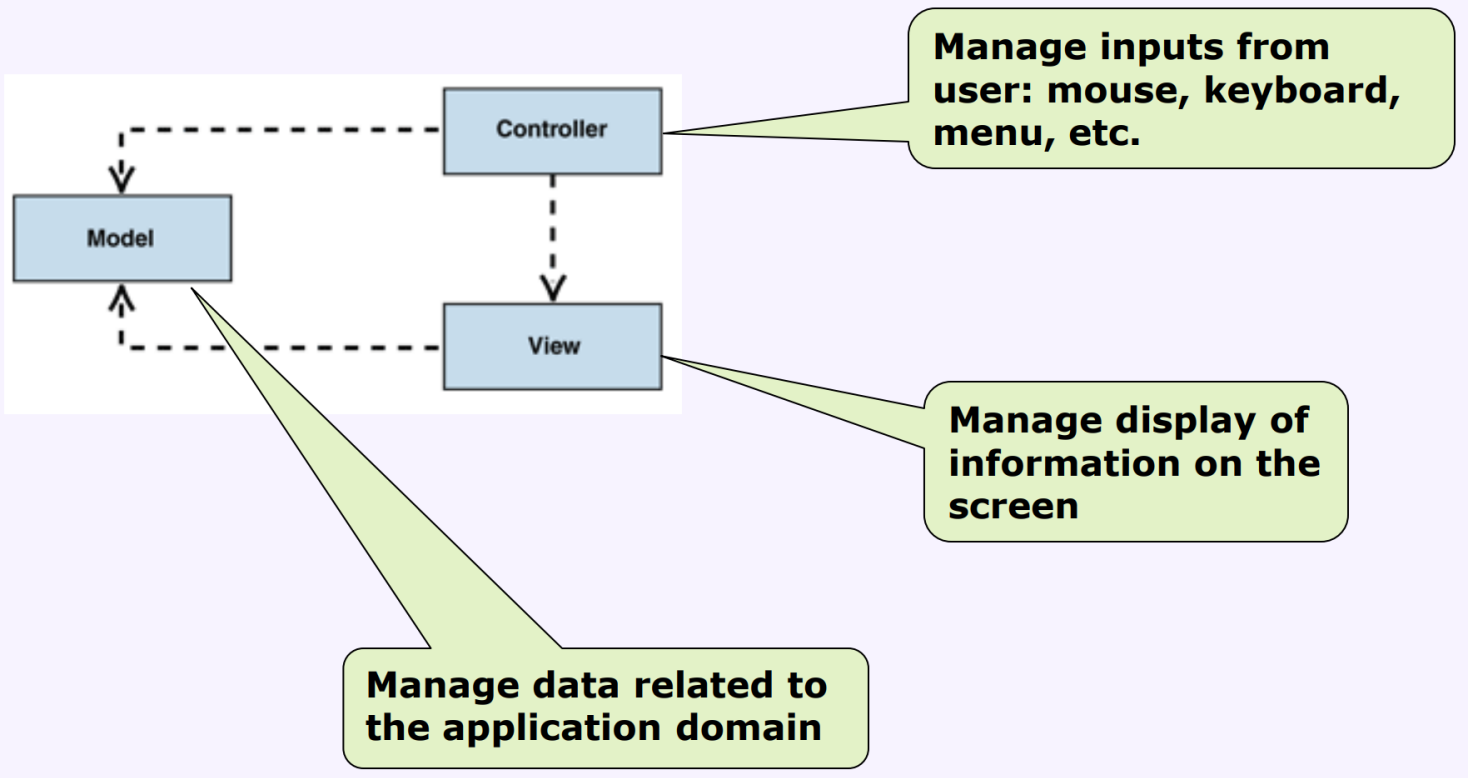


FXML File templates

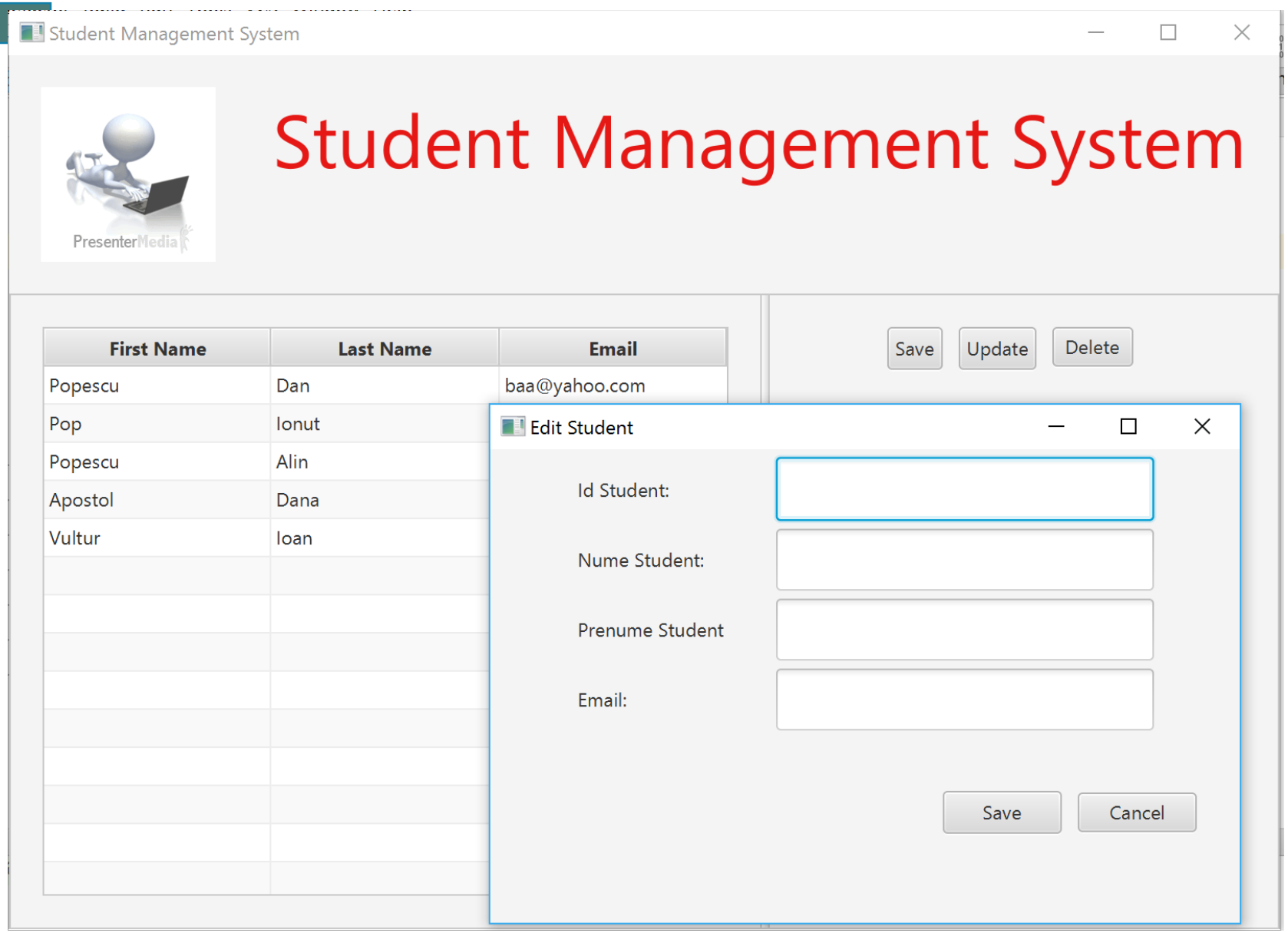


Model View Controller (MVC)

- JavaFX este dezvoltata dupa filozofia **Model View Controller** (MVC) separand partea de logica de partea de vizualizare si manipulare.



MVC FXML Exemplu



View - FXML

- Exemplu: C7_3

Swing sau JavaFX?

- **Pro Swing**
- Maturitate
- Biblioteci de componente, Framework-uri
- Documentatie si resurse bogate
- **Pro JavaFX**
- Tehnologie moderna, nu doar pentru desktop (RIA)
- Orientata spre "spectaculos"
- Suport pentru animatii, grafica 2D, 3D