



**JavaFX**

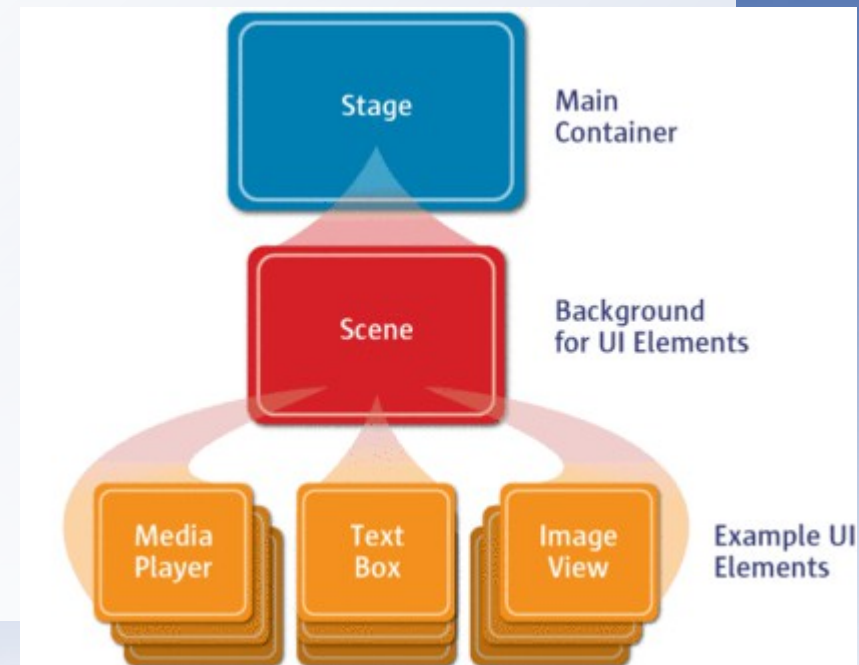
# Uvod u JavaFX

- Skup grafičkih i multimedijских paketa za razvoj aplikacija s bogatim sučeljem
- Zamjena za Swing skup alata
- Prva verzija objavljena 10/2008, a u 08/2012 objavljena verzija 2.2
- S Javom 8 se preimenovala u JavaFX 8 i postaje standardni dio Jave
- Dizajniranje grafičkog sučelja je omogućeno kroz FXML koji se generira pomoću Scene Builder razvojnog okruženja
- Podržava korištenje CSS-a, HTML5 i JavaScripta

# Glavna klasa

- Klasa koja služi za pokretanje JavaFX aplikacije mora nasljeđivati klasu "javafx.application.Application"

```
public class Main extends Application {  
    @Override  
    public void start(Stage primaryStage) {  
        try {  
            BorderPane root = new BorderPane();  
            Scene scene = new Scene(root, 500, 250);  
            scene.getStylesheets().add(getClass().getResource(  
                "application.css").toExternalForm());  
            primaryStage.setScene(scene);  
            primaryStage.show();  
        } catch (Exception e) {  
            e.printStackTrace();  
        }  
    }  
  
    public static void main(String[] args) {  
        launch(args);  
    }  
}
```



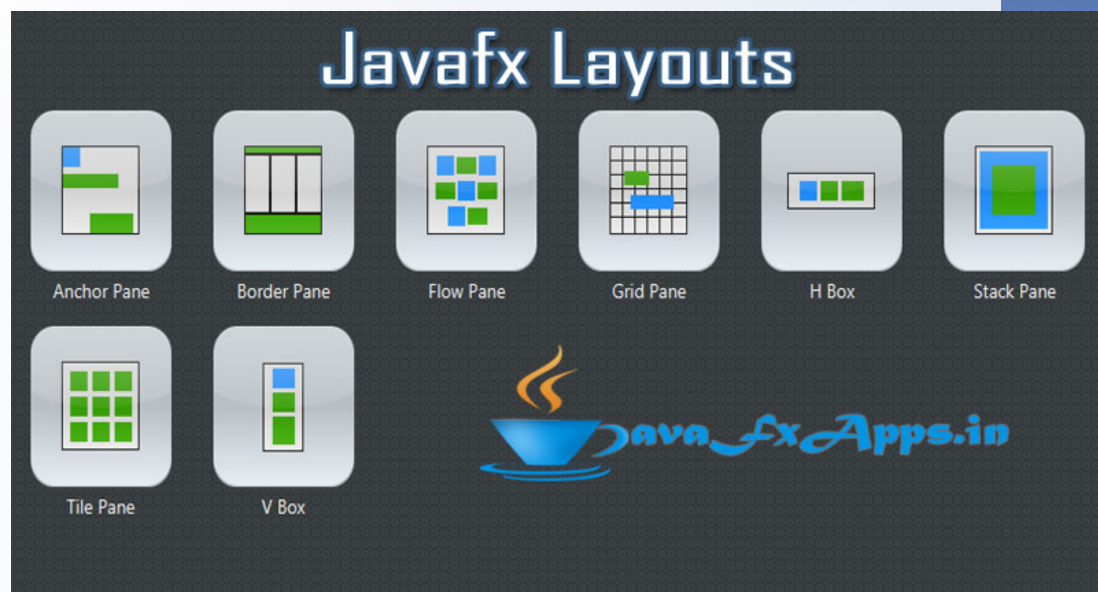
# JavaFX komponente





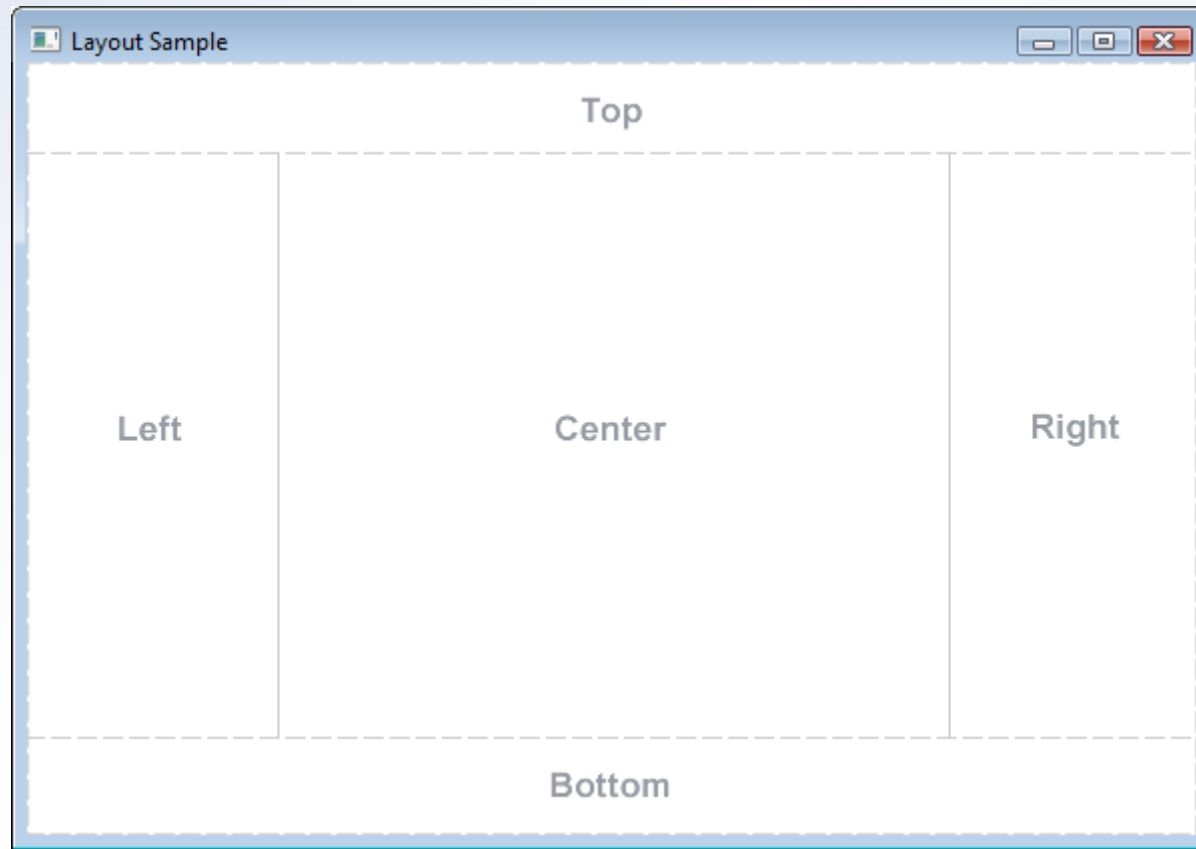
# Kreiranje grafičkog sučelja pomoću JavaFX-a

- Pomoću JavaFX spremnika dodaju se grafičke komponente na ekran
- JavaFX omogućava korištenje ugrađenih panela kao što je "BorderPane", pri čemu su mnogi ekvivalentni Swingovim inačicama organizatora rasporeda komponenti
- Najčešći JavaFX paneli su:
  - BorderPane
  - HBox i VBox
  - StackPane
  - GridPane
  - FlowPane
  - AnchorPane



## BorderPane (1/2)

- Ekvivalentan "BorderLayout" organizatoru rasporeda komponenti:



## BorderPane (2/2)

- Nakon kreiranja objekta klase "BorderPane" mogu se dodavati ostali elementi korištenjem odgovarajućih metoda za lociranje područja
- Primjer korištenja "BorderPane" panela:

```
BorderPane border = new BorderPane();  
HBox hbox = addHBox();  
border.setTop(hbox);  
border.setLeft(addVBox());  
addStackPane(hbox); // Add stack to HBox in top region  
  
border.setCenter(addGridPane());  
border.setRight(addFlowPane());
```

# HBox

- Organizira komponente u jedan horizontalni redak
- Primjer izgleda rasporeda komponenti:



```
public HBox addHBox() {
    HBox hbox = new HBox();
    hbox.setPadding(new Insets(15, 12, 15, 12));
    hbox.setSpacing(10);
    hbox.setStyle("-fx-background-color: #336699;");

    Button buttonCurrent = new Button("Current");
    buttonCurrent.setPrefSize(100, 20);

    Button buttonProjected = new Button("Projected");
    buttonProjected.setPrefSize(100, 20);
    hbox.getChildren().addAll(buttonCurrent, buttonProjected);

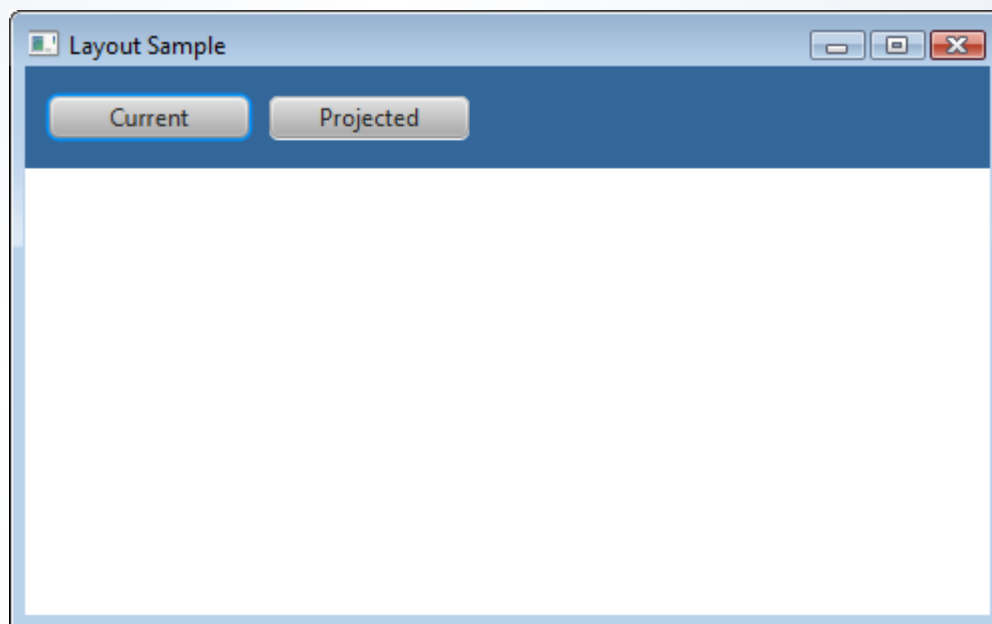
    return hbox;
}
```



## Dodavanje HBox komponente u BorderPane

- Izvršavanjem programskog koda koji dodaje "HBox" komponenti u "TOP" područje "BorderPane" panela moguće je dobiti sljedeći izgled sučelja:

```
BorderPane border = new BorderPane();  
HBox hbox = addHBox();  
border.setTop(hbox);
```



# VBox

- Organizira grafičko sučelje na način da su komponente smještene u jedan stupac:

```
public VBox addVBox() {  
    VBox vbox = new VBox();  
    vbox.setPadding(new Insets(10));  
    vbox.setSpacing(8);  
  
    Text title = new Text("Data");  
    title.setFont(Font.font("Arial", FontWeight.BOLD, 14));  
    vbox.getChildren().add(title);  
  
    Hyperlink options[] = new Hyperlink[] {  
        new Hyperlink("Sales"),  
        new Hyperlink("Marketing"),  
        new Hyperlink("Distribution"),  
        new Hyperlink("Costs")};  
  
    for (int i=0; i<4; i++) {  
        VBox.setMargin(options[i], new Insets(0, 0, 0, 8));  
        vbox.getChildren().add(options[i]);  
    }  
  
    return vbox;  
}
```

## Data

Sales

Marketing

Distribution

Costs

## StackPane (1/2)

- Organizira komponente po principu stoga, kod čega se svaka nova komponenta dodaje "na vrh" (omogućava se preklapanje komponenti)
- Primjer korištenja:

```
public void addStackPane(HBox hb) {  
    StackPane stack = new StackPane();  
    Rectangle helpIcon = new Rectangle(30.0, 25.0);  
    helpIcon.setFill(new LinearGradient(0,0,0,1, true, CycleMethod.NO_CYCLE,  
        new Stop[]{  
            new Stop(0,Color.web("#4977A3")),  
            new Stop(0.5, Color.web("#B0C6DA")),  
            new Stop(1,Color.web("#9CB6CF")),}));  
    helpIcon.setStroke(Color.web("#D0E6FA"));  
    helpIcon.setArcHeight(3.5);  
    helpIcon.setArcWidth(3.5);  
}
```

## StackPane (2/2)

```
Text helpText = new Text("?");
helpText.setFont(Font.font("Verdana", FontWeight.BOLD, 18));
helpText.setFill(Color.WHITE);
helpText.setStroke(Color.web("#7080A0"));

stack.getChildren().addAll(helpIcon, helpText);
stack.setAlignment(Pos.CENTER_RIGHT); // Right-justify nodes in stack
StackPane.setMargin(helpText, new Insets(0, 10, 0, 0)); // Center "?"


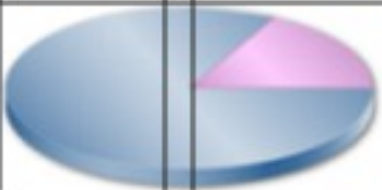
hb.getChildren().add(stack); // Add to HBox from Example 1-2
HBox.setHgrow(stack, Priority.ALWAYS); // Give stack any extra space
}
```

- Izgled nove "Help" ikone na grafičkom sučelju:



## GridPane (1/3)

- Smješta komponente u područja definirana tablicom, pri čemu komponente mogu zauzimati više redaka i stupaca (slično kao "GridBagLayout") kod Swinga

|   |  |  |                 |
|---|--|--|-----------------|
|  | <b>Sales: Current Year</b>   |  |                 |
|   | Goods and Services   |  |                 |
|   |  |  | Services<br>20% |
| Goods<br>80%  |  |  |                 |

- Primjer korištenja:

```
public GridPane addGridPane() {  
    GridPane grid = new GridPane();  
    grid.setHgap(10);  
    grid.setVgap(10);  
    grid.setPadding(new Insets(0, 10, 0, 10));  
}
```



## GridPane (2/3)

```
// Category in column 2, row 1
Text category = new Text("Sales:");
category.setFont(Font.font("Arial", FontWeight.BOLD, 20));
grid.add(category, 1, 0);

// Title in column 3, row 1
Text chartTitle = new Text("Current Year");
chartTitle.setFont(Font.font("Arial", FontWeight.BOLD, 20));
grid.add(chartTitle, 2, 0);

// Subtitle in columns 2-3, row 2
Text chartSubtitle = new Text("Goods and Services");
grid.add(chartSubtitle, 1, 1, 2, 1);

// House icon in column 1, rows 1-2
ImageView imageHouse = new ImageView(
    new
Image(LayoutSample.class.getResourceAsStream("graphics/house.png")));
grid.add(imageHouse, 0, 0, 1, 2);

// Left label in column 1 (bottom), row 3
Text goodsPercent = new Text("Goods\n80%");
GridPane.setValignment(goodsPercent, VPos.BOTTOM);
grid.add(goodsPercent, 0, 2);
```

## GridPane (3/3)

```
// Chart in columns 2-3, row 3
    ImageView imageChart = new ImageView(
        new
Image(LayoutSample.class.getResourceAsStream("graphics/piechart.png")));
    grid.add(imageChart, 1, 2, 2, 1);

    // Right label in column 4 (top), row 3
    Text servicesPercent = new Text("Services\n20%");
    GridPane.setValignment(servicesPercent, VPos.TOP);
    grid.add(servicesPercent, 3, 2);

    return grid;
}
```

# FlowPane

- Ekvivalentan "FlowLayout" iz Swinga
- Primjer izgleda sučelja:

```
public FlowPane addFlowPane() {  
    FlowPane flow = new FlowPane();  
    flow.setPadding(new Insets(5, 0, 5, 0));  
    flow.setVgap(4);  
    flow.setHgap(4);  
    flow.setPrefWrapLength(170);  
    flow.setStyle("-fx-background-color: DAE6F3;");  
  
    ImageView pages[] = new ImageView[8];  
    for (int i=0; i<8; i++) {  
        pages[i] = new ImageView(  
            new Image(LayoutSample.class.getResourceAsStream(  
                "graphics/chart_" + (i+1) + ".png")));  
        flow.getChildren().add(pages[i]);  
    }  
  
    return flow;  
}
```



# AnchorPane

- Omogućava "usidravanje" komponenti na određenim pozicijama na ekranu

```
public AnchorPane addAnchorPane(GridPane grid) {  
    AnchorPane anchorpane = new AnchorPane();  
    Button buttonSave = new Button("Save");  
    Button buttonCancel = new Button("Cancel");  
  
    HBox hb = new HBox();  
    hb.setPadding(new Insets(0, 10, 10, 10));  
    hb.setSpacing(10);  
    hb.getChildren().addAll(buttonSave, buttonCancel);  
  
    anchorpane.getChildren().addAll(grid, hb);  
    AnchorPane.setBottomAnchor(hb, 8.0);  
    AnchorPane.setRightAnchor(hb, 5.0);  
    AnchorPane.setTopAnchor(grid, 10.0);  
  
    return anchorpane;  
}
```



# Korištenje više različitih panela i komponenti na jednom ekranu

- Kao i u slučaju Swing aplikacija, JavaFX također omogućava kombiniranje različitih panela i komponenti na jednom grafičkom sučelju:





# Korištenje CSS-a u JavaFX aplikacijama (1/2)

- Izgled komponenti na grafičkim sučeljima JavaFX aplikacijama moguće je mijenjati i pomoću CSS datoteka:



## Korištenje CSS-a u JavaFX aplikacijama (2/2)

- Osim podrazumijevane "caspiian.css" datoteke, moguće je definirati i vlastite CSS datoteke i stilove, npr.

```
Scene scene = new Scene(new Group(), 500, 400);  
scene.getStylesheets().add("path/styleSheet.css")  
;
```

- Unutar samih CSS datoteka stilovi, bolje i ostali parametri mogu se definirati na sljedeći način:

```
.custom-button {  
    -fx-font: 16px "Serif";  
    -fx-padding: 10;  
    -fx-background-color: #CCFF99;  
}  
  
.button1{  
    -fx-text-fill: #006464;  
    -fx-background-color: #DFB951;  
    -fx-border-radius: 20;  
    -fx-background-radius: 20;  
    -fx-padding: 5;  
}
```

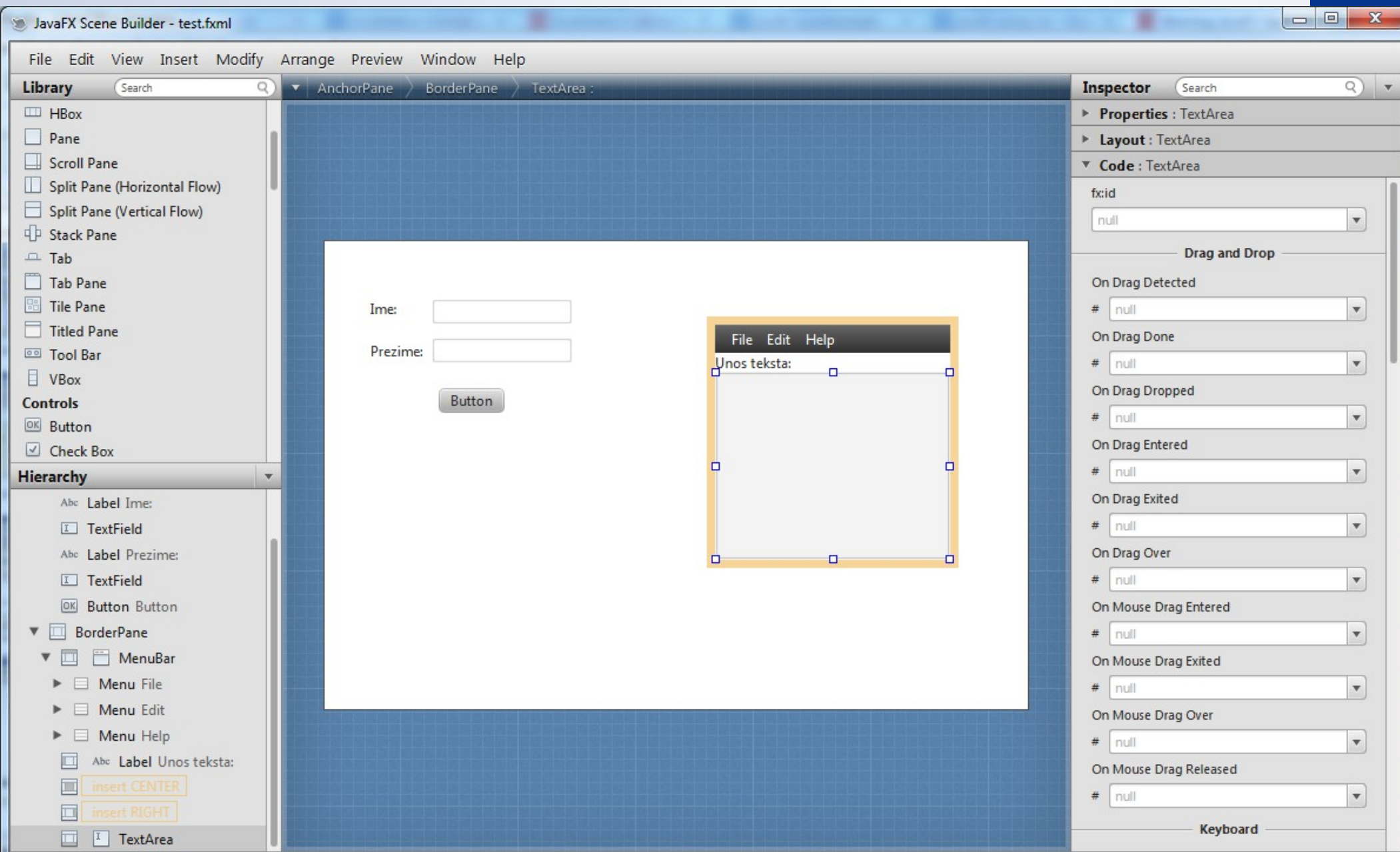
- Korištenje definiranih stilova je moguće na ovaj način:

```
Button buttonAccept = new Button("Accept");  
buttonAccept.getStyleClass().add("button1");
```

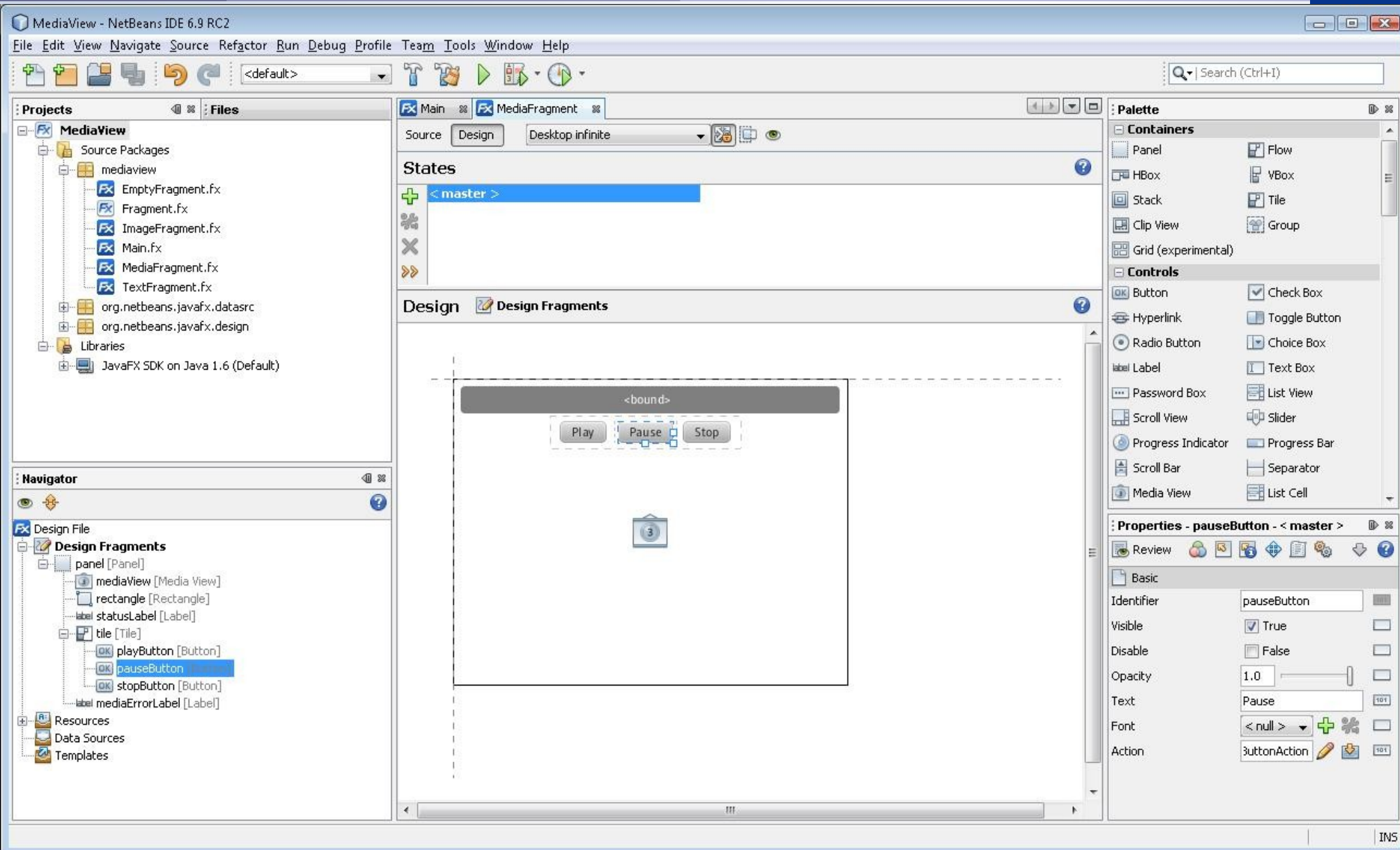
# JavaFX dizajneri grafičkih sučelja

- Postoji nekoliko dizajnera grafičkih sučelja za JavaFX aplikacije:
  - JavaFX Scene Builder
  - NetBeans JavaFX Composer
  - e(fx)clipse

# JavaFX Scene Builder



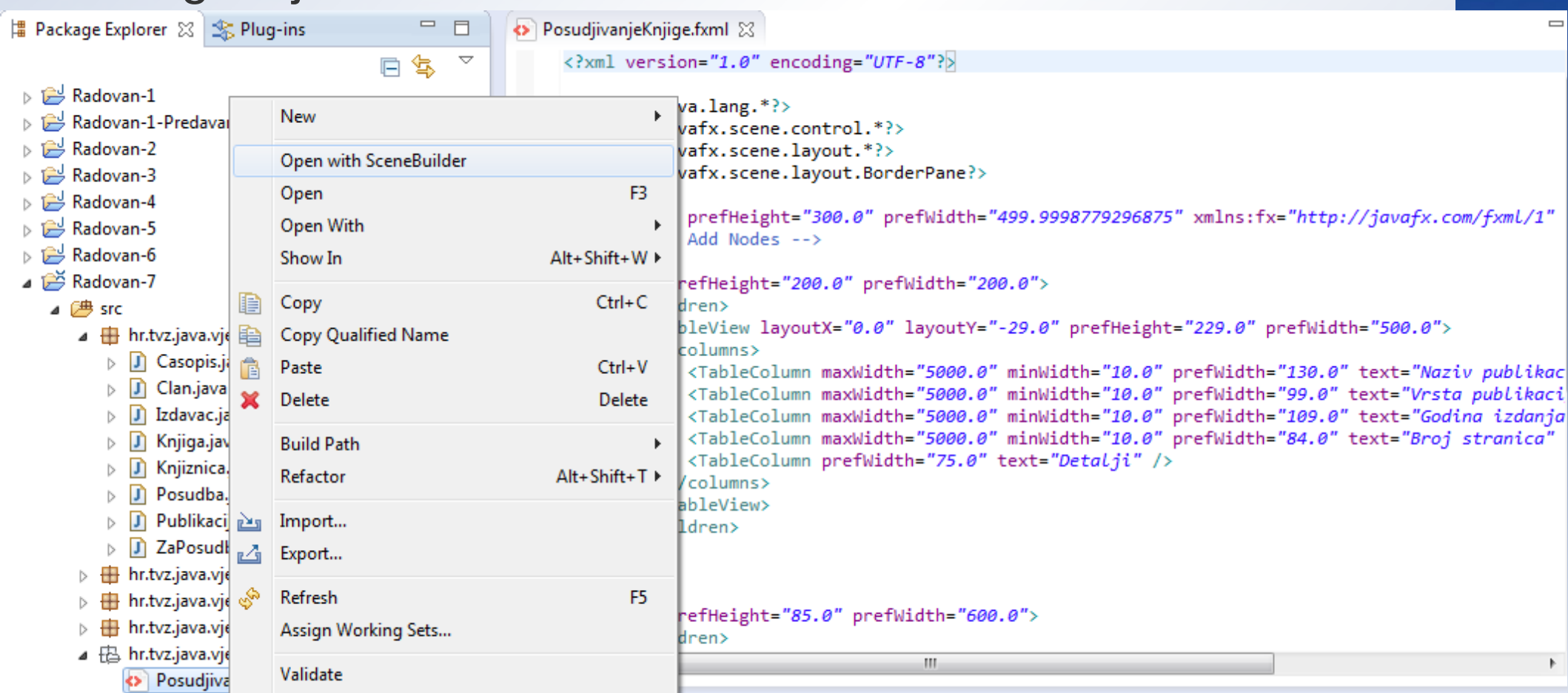
# NetBeans JavaX Composer



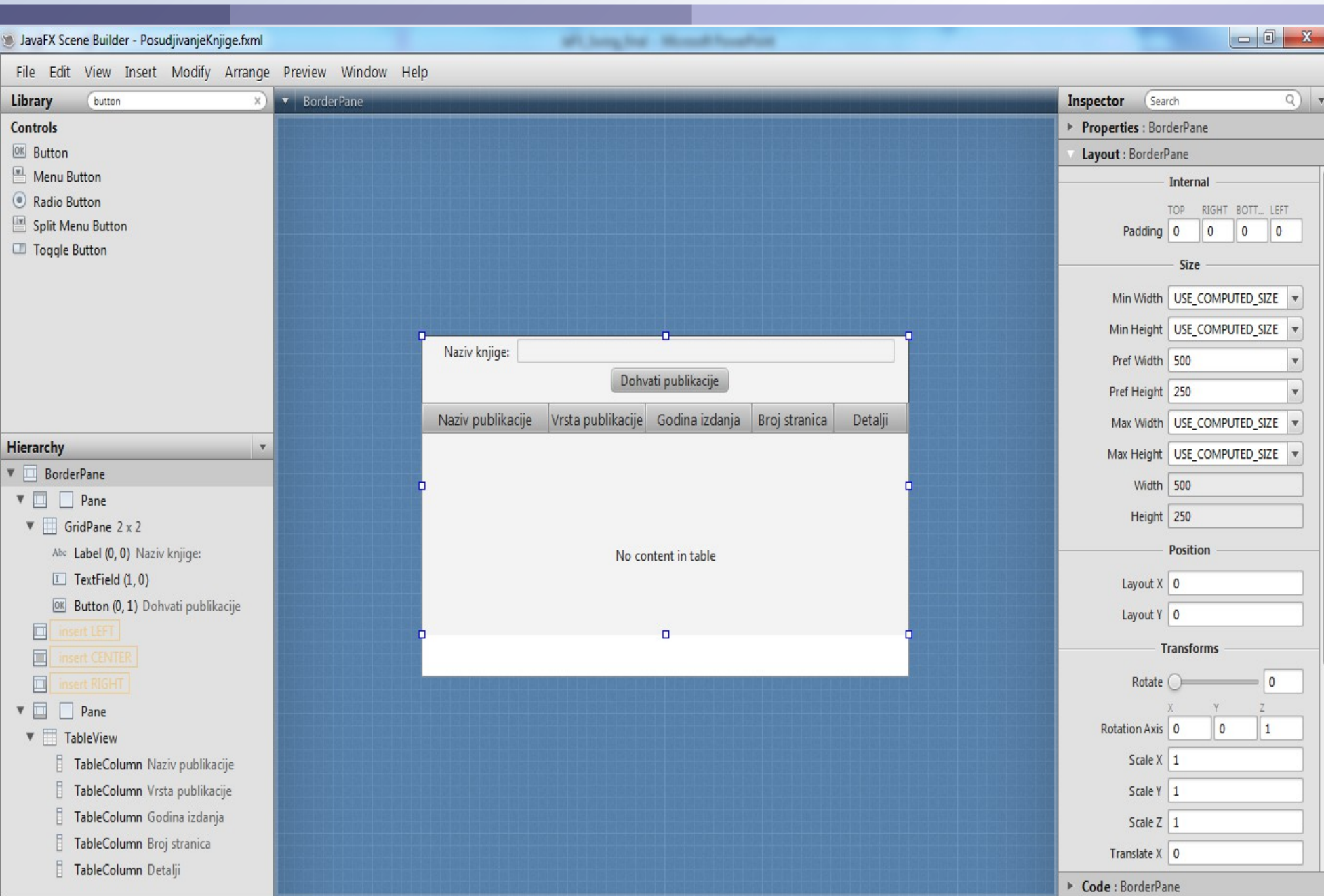


# e(fx)clipse

- Skupina *pluginova* za Eclipse
- Moguće preuzeti prekonfiguriranu inačicu Eclipsea
- Integracija sa Scene Builderom



# Dizajniranje pomoću Scene Buildera



# Generirana "fxml" datoteka

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

<?import java.lang.*?>
<?import javafx.scene.control.*?>
<?import javafx.scene.layout.*?>
<?import javafx.scene.layout.BorderPane?>

<BorderPane prefHeight="300.0" prefWidth="500.0" xmlns:fx="http://javafx.com/fxml/1" xmlns="http://javafx.com/javafx/2.2">
  <bottom>
    <Pane prefHeight="200.0" prefWidth="200.0">
      <children>
        <TableView layoutX="0.0" layoutY="-29.0" prefHeight="229.0" prefWidth="500.0">
          <columns>
            <TableColumn maxWidth="5000.0" minWidth="10.0" prefWidth="130.0" text="Naziv publikacije" />
            <TableColumn maxWidth="5000.0" minWidth="10.0" prefWidth="99.0" text="Vrsta publikacije" />
            <TableColumn maxWidth="5000.0" minWidth="10.0" prefWidth="109.0" text="Godina izdanja" />
            <TableColumn maxWidth="5000.0" minWidth="10.0" prefWidth="84.0" text="Broj stranica" />
            <TableColumn prefWidth="75.0" text="Detalji" />
          </columns>
        </TableView>
      </children>
    </Pane>
  </bottom>
  <top>
    <Pane prefHeight="85.0" prefWidth="600.0">
      <children>
        <GridPane alignment="CENTER" layoutX="23.0" layoutY="0.0" prefHeight="50.0" prefWidth="463.0">
          <children>
            <Label alignment="BASELINE_RIGHT" contentDisplay="RIGHT" graphicTextGap="4.0" text="Naziv knjige:" underline="false"
visible="true" GridPane.columnIndex="0" GridPane.rowIndex="0" />
            <TextField prefWidth="200.0" GridPane.columnIndex="1" GridPane.rowIndex="0" />
            <Button mnemonicParsing="false" text="Dohvati publikacije" GridPane.columnIndex="0" GridPane.columnSpan="2147483647"
GridPane.halignment="CENTER" GridPane.rowIndex="1" />
          </children>
          <columnConstraints>
            <ColumnConstraints hgrow="SOMETIMES" maxWidth="232.0" minWidth="10.0" prefWidth="74.0" />
            <ColumnConstraints hgrow="SOMETIMES" maxWidth="389.0" minWidth="10.0" prefWidth="389.0" />
          </columnConstraints>
          ...
        </GridPane>
      </children>
    </Pane>
  </top>
</BorderPane>
```

# Korištenje "fxml" datoteke

- Nakon što se generira konfiguracijska XML datoteka pomoću Scene Buildera, mora se uključiti u dizajn aplikacije unutar Java koda:

```
private BorderPane root;  
FXMLLoader loader = new FXMLLoader(  
Main.class.getResource("../javafx/PosudjivanjeKnjige.fxml"));  
root = (BorderPane) loader.load();  
Scene scene = new Scene(root, 500, 250);
```

- Pomoću metode "getResource" dohvaća se lokacija "fxml" datoteke i uz pomoć nje učitava "BorderPane" ekran koji je definiran kroz Scene Builder

# JavaFX zbirke

- JavaFX za potrebe rada s podacima na grafičkom sučelju uvodi sljedeća sučelja za zbirke:
  - **ObservableList**: lista koja omogućava praćenje promjena na ekranu korištenjem "listenera"
  - **ObservableMap**: mapa koja omogućava praćenje promjena na ekranu korištenjem "listenera"
  - **ListChangeListener** i **MapChangeListener**: služi za primanje notifikacija koje generira instanca klase "ObservableList" "ObservableMap"
- JavaFX također uključuje i sljedeće klase:
  - **FXCollections**: sadrži istovjetne statičke metode kao i klasa "java.util.Collections"
  - **ListChangeListener.Change** i **MapChangeListener.Change**: predstavljaju promjene nad "ObservableList" "ObservableMap"



# Primjer korištenja ObservableList

```
// Kreiranje liste koja će se pratiti
List<String> list = new ArrayList<String>();

// Kreiranje instance ObservableList iz liste
ObservableList<String> observableList = FXCollections.observableList(list);
observableList.addListener(new ListChangeListener() {

    @Override
    public void onChanged(ListChangeListener.Change change) {
        System.out.println("Detektirana promjena! ");
    }
});

// Generiranje promjene na listi koja poziva metodu "onChanged"
// This line will print out "Detected a change!"
observableList.add("item one");

// Ova promjena ne generira poziv metode "onChanged"
list.add("item two");

System.out.println("Veličina: " + observableList.size());
```

# JavaFX i *Controller* klase

- JavaFX omogućava dijeljenje aplikacije na nekoliko slojeva:
  - Podatkovni sloj (**Model**)
  - Prezentacijski sloj (**View** definiran kroz FXML datoteku)
  - Upravljački sloj (**Controller** – služi za povezivanje Viewa i poslovne logike aplikacije)
- Korištenjem @FXML anotacije unutar Controller klase moguće je izravno povezivanje s komponentama grafičkog sučelja preko njihovog ID-a, npr:

```
public class PosudjivanjeKnjigeController {  
  
    @FXML  
    private TableView<Publikacija> publikacijeTable;  
  
    ...  
}
```

## @FXML i inicijalizacija Controller klase

- Ako je potrebno obaviti inicijalizaciju vrijednosti na grafičkom sučelju, npr. povezati kolone tablice s varijablama u klasama koje predstavljaju podatke, takve metode je također potrebno označiti anotacijom @FXML:

@FXML

```
private TableColumn<Publikacija, String> nazivColumn;
```

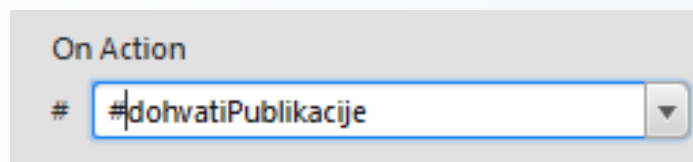
@FXML

```
public void initialize() {  
    nazivColumn.setCellValueFactory(  
        new PropertyValueFactory<Publikacija, String>("naziv"));  
}
```

# Povezivanje gumbiju s akcijama u Controller klasi

- Unutar FXML-a kroz Scene Builder moguće je definirati što će se iz Controllera izvesti npr. na pritisak gumba
- Ako se u Controlleru definira metoda koja dohvaća podatke i vraća ih u obliku "ObservableList" instance, tu akciju je preko Scene Buildera moguće povezati s gumbom definiranjem parametra "On Action" i navođenjem naziva metode:

```
@FXML ObservableList<Publikacija> dohvatiPublikacije() {  
    ...  
}
```



# Literatura

- <http://wiki.eclipse.org/Efxclipse/Tutorials>
- <http://code.makery.ch/java/javafx-2-tutorial-intro/>
- <http://javantura.com/predavanja/#javafx>
- <http://www.javafxapps.in/>
- <http://docs.oracle.com/javafx/2/collections/jfxpub-collections.htm>
- <http://docs.oracle.com/javafx/2/api/javafx/scene/doc-files/cssref.html>
- [http://docs.oracle.com/javafx/2/ui\\_controls/overview.htm](http://docs.oracle.com/javafx/2/ui_controls/overview.htm)
- <http://www.oracle.com/technetwork/articles/java/casa-1919152.html>