1 FXML

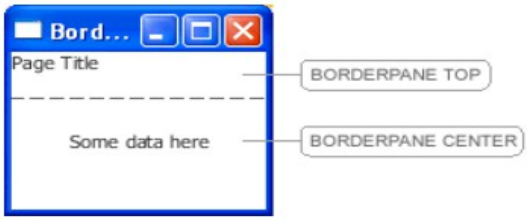
FXML je zasnovan na jeziku XML i obezbeđuje strukturu za izgradnju grafičkog korisničkog interfejsa odvojeno od aplikacione logike koda.

FXML nema shemu poput XML-a, ali ima osnovnu, unapred definisanu strukturu.
FXML je direktno povezan sa Javom.

Iako FXML može da se koristi za kreiranje bilo kakvog grafičkog korisničkog interfejsa, posebno je koristan za kreiranje grafičkih korisničkih interfejsa koji imaju velike, kompleksne grafove scena, forme, unos podataka ili složene animacije.

FXML je pogodan za definisanje statičkih rasporeda kao što su forme, kontrole i tabele, ili za dinamičke sadržaje koji uključuju skripte.

Primer grafičkog korisničkog interfejsa koji sadrži `BorderPane` i u poljima na vrhu i u centru ima labele.



Java kod

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

FXML

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

Dobre strane upotrebe FXML-a

Graf scene se jednostavnije sagleda u FXML-u, pa je jednostavnije napraviti i održavati grafički korisnički interfejs kroz testiranja.
Kod pisan u FXML-u ne mora da se kompajlira da bi se videle promene.

FXML može da se upotrebljava sa bilo kojim Java Virtual Machine (JVM) jezikom, kao što su Java, Scala ili Clojure.
U FXML-u mogu da se koriste JavaScript i drugi skript jezici.

Korisni linkovi
http://docs.oracle.com/javafx/2/api/javafx/fxml/doc-files/introduction_to_fxml.html

2 FXML i SceneBuilder

Scene Builder generiše izvorni kod FXML dokumenta kako se definiše grafički korisnički interfejs za JavaFX aplikaciju.

Scene Builder omogućava da se brzo i jednostavno napravi prototip interaktivne aplikacije koja povezuje vizuelne komponente sa logikom aplikacije.

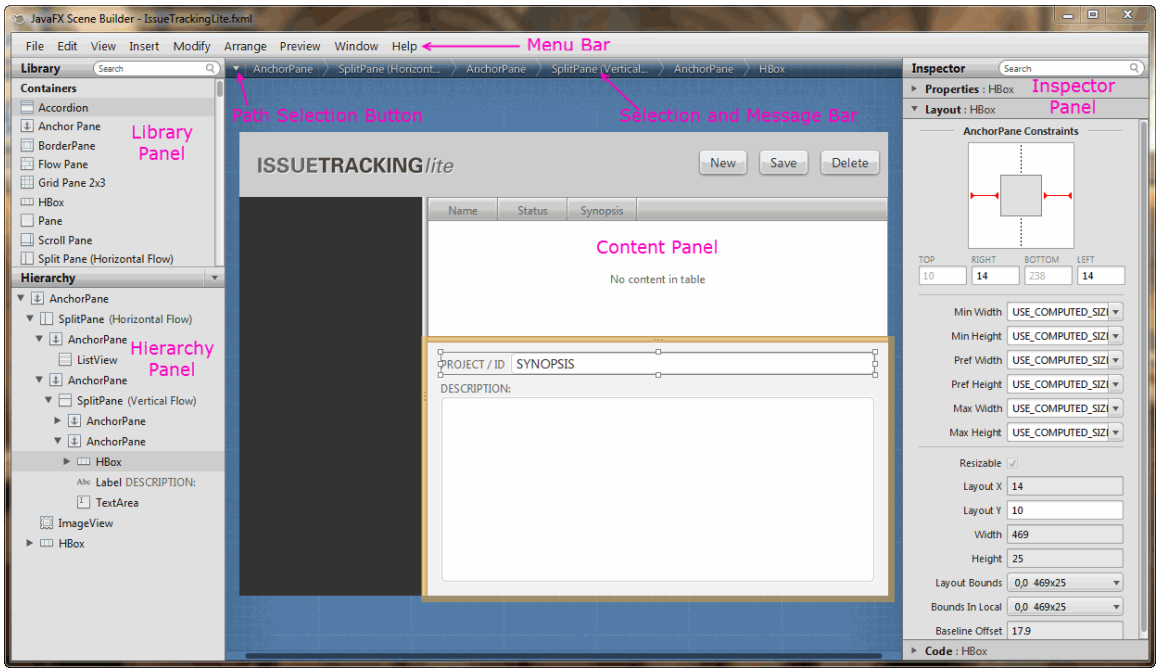
JavaFX Scene Builder je alat za dizajniranje JavaFX aplikacija.

Omogućava jednostavno prevlačenje (drag-and-drop) i pozicioniranje komponenata grafičkog korisničkog interfejsa na JavaFX scenu.

Kako se gradi scena, tako se automatski generiše odgovarajući FXML dokument.

3 Osnovni prozor JavaFX Scene Builder-a

Osnovni prozor sastoji se iz nekoliko panela koji su označeni na slici:



Library Panel

Omogućava izbor, prevlačenje i pozicioniranje komponente u centralnu radnu površinu (Content Panel).
Komponente su organizovane u grupe:

- Containers
- Controls
- Popup Controls
- Menu Content
- Shapes
- Charts
- Misc

U polje za pretragu se može uneti naziv (dovoljno je i početno slovo naziva) potrebne komponente, čime se olakšava izbor iste.

Hierarchy Panel

Sadrži hijerarhijski prikaz čvorova grafa scene. Klikom na strelicu sa desne strane može se izabrati način prikaza hijerarhije:

- Show Info (podrazumevani prikaz)
- Show fx:id
- Show Node

Moguće je u panelu promeniti prikazane informacije ili fx:id za odgovarajući čvor. Dvostrukim klikom na prikazani podatak, dobija se mogućnost editovanja.

Inspector Panel

Podeljen je na tri sekcije:

- Properties
- Layout
- Code

4 Primer: testSceneBuilder

Kreiranje FXML dokumenta

U odgovarajućem paketu (testSceneBuilder) napraviti FXML dokument:

```
New -> Other... -> JavaFX -> New FXML Document
```

- u polje **Name** uneti ime dokumenta (`TestSB`)
- **Source folder** treba da sadrži naziv odgovarajućeg JavaFX projekta, promeniti po potrebi
- polje **Package** sadrži ime odgovarajućeg paketa, ili je prazno ako se sadržaj nalazi u podrazumevanom paketu
- postaviti **Root Element** (koreni čvor grafa scene) - podrazumevano je ponuđen AnchorPane, izabrati BorderPane
- ostaviti **DynamicRoot** neizabran, jer se opcija koristi za naprednija podešavanja
- **Finish**

Otvaranje FXML dokumenta pomoću Scene Builder-a

Iz kontekstnog menija za TestSB.fxml fajl (nakon desnog klika) izabrati opciju:

```
Open with SceneBuilder
```

Ako se koristi e(fx)clipse, nakon izbora ove opcije, automatski se otvara SceneBuilder.

U slučaju da koristi eclipse Juno kome je dodata podrška za JavaFX, potrebno je izvršiti neka podešavanja, da bi SceneBuilder mogao da se pokrene:

- Window → Preferences
- klik na JavaFX (ne na plusić ispred)
- u polje *SceneBuilder executable* traba da stoji putanja do SceneBuilder-a
- klik na Browse...
- izabrati SceneBuilder.exe iz odgovarajućeg direktorijuma, na primer, pod Windows-om, uobičajena putanja je:
  C:\Program Files\Oracle\JavaFX Scene Builder 1.1\ JavaFX Scene Builder 1.1.exe

Druga varijanta je da se omogući da se, kada se dva puta klikne na .fxml datoteku, ona odmah otvori u SceneBuilder-u:

- Window → Preferences
- klik na plusić ispred General
- klik na plusić ispred Editors
- izabrati FileAssociations
- U polju File types: pronaći i selektovati *.fxml
- u polje Associated editors: treba dodati SceneBuilder
- klik na Add...
- u otvorenom prozoru selektovati opciju External programs
- klik na Browse... i izabrati SceneBuilder sa odgovarajuće lokacije
- nakon toga, SceneBuilder će biti dodat u polje sa editorima
- kliknuti na dugme Default sa desne strane, da bi se SceneBuilder proglasio za podrazumevani editor .fxml dokumenata
- klik na OK

Pravljenje grafa scene pomoću SceneBuilder-a

Po potrebi povećati Content panel.

BorderPane

U Hierarchy panelu selektovati BorderPane, a u Inspector panelu izabrati sekciju Layout i postaviti:

- Pref Width: 300, Pref Height: 150
- Padding: 10 (sve vrednosti)

VBox

Izabrati VBox i prevući ga u centralno područje.

Selektovati Vbox u Hierarchy panelu.
U sekciji Layout Inspector panela postaviti:

- Margin: 10 (sve vrednosti)
- Spacing: 20

a u sekciji Properties postaviti:

- Alignment: CENTER (komponente koje će biti dodate u VBox biće centrirane)

Button

Izabrati Button i prevući ga u VBox.

Selektovati Button u Hierarchy panelu.
U sekciji Properties Inspector panela postaviti:

- Text: Zdravo
- Font: klik na strelicu
  - family: Bookman Old Style
  - style: Bold Italic
  - size: 14
- Text Fill: web: #990093 (ili bilo koju drugu boju)
-
- čekirati Underline

Label

Izabrati Label i prevući je u VBox.

Selektovati Label u Hierachy panelu.
U sekciji Properties Inspector panela postaviti:

- Text: Poruka za kraj
- Font: klik na strelicu
  - family: Antique Olive
  - style: Italic
  - size: 14
- Text Fill: web: #e9ffcc (ili bilo koju drugu boju)

U sekciji Layout Inspector panela postaviti:

- Pref Width: 80, Pref Height: 30

Pregled generisanih komponenti

```
Preview → Show Preview in Window (Ctrl+P)
```

Sačuvati izmene i potom pogledati sadržaj FXML dokumenta u Eclipse okruženju.
Može se uočiti kod koji je automatski generisan, postepeno, tokom izgradnje samog korisničkog interfjesa u Scene Builder-u.

Pridruživanje CSS datoteke sa stilovima

U odgovarajućem paketu (testSceneBuilder) napraviti CSS datoteku:

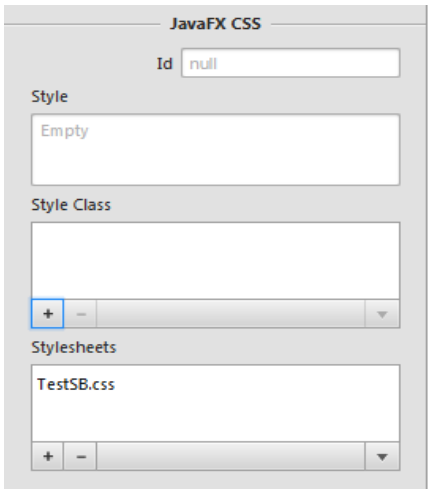
```
New → Other... → Web → CSS
```

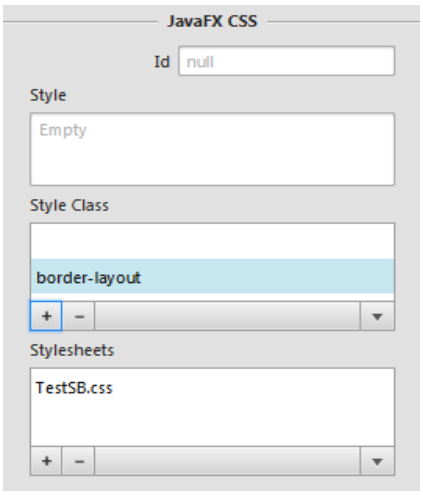
i definisati potrebne stilove.
Na primer:

```
.border-pane {
        -fx-background-color: radial-gradient(radius 100%, white, skyblue);
        -fx-border-color: blueviolet;
}
```

U Hierarchy panelu izabrati komponentu kojoj se pridružuje CSS stil, za početak BorderPane.

Izabrati sekciju Properties u Inspector panelu, kliknuti na dugme sa znakom plus (+) u Stylesheets i izabrati .css dokument (TestSB.css) sa podacima o stilu.
Potom iz StyleClass klikom na dugme za znakom plus (+) izabrati odgovarajuću klasu koja će se primeniti nad komponentom (border-pane).





Slično uraditi i za Button:

- u StyleClass dodati klase: button i font

Ako se pogleda kako izgledaju svojstva Font i Text Fill, može se uočiti da su primenjene css stilova promenile prethodno postavljene vrednosti. Sa desne strane od izabranog fonta, odnosno boje, sada se nalazi dugme sa labelom „CSS“ i ako se klikne na njega, u editoru se otvara odgovarajući CSS dokument.
Odgovarajući stilovi primenjeni na dugme:

```
.font
{
        -fx-text-fill: darkcyan;
        -fx-font-style: oblique, italic;
        -fx-font-size: 16;
        -fx-font-family: "Book Antiqua";
}

.dugme
{
        -fx-border-color: blueviolet;
        -fx-background-color: radial-gradient(radius 100%, gray, white);
}
```

Ručno postavljanje stilova u SceneBuilder-u

Stilovi se mogu i ručno postaviti za komponente direktno u SceneBuilder-u:
- u polje Style zadaje se odgovarajuća vrednost

Na primer, za dugme *Zdravo* bi se uradilo sledeće:
- selektovati dugme u Hierarchy panelu
- izabrati sekciju Properties u Inspector panelu
- u polje Style uneti:

```
-fx-border-color: blueviolet;
-fx-background-color: radial-gradient(radius 100%, gray, white
```

ili zadati neko drugo svojstvo za dugme

Direktno pridruživanje CSS dokumenta sceni (ako se ne koristi SceneBuilder)

Moguće je pridružiti sceni .css dokument i direktno:

```
scene.getStylesheets().add(getClass().
            getResource("../TestSB.css").toExternalForm());
```

String sadrži relativnu putanju .css dokumenta u odnosu na .class fajl.

Metod **setStyle()** se koristi za definisanje izgleda pojedinačnih komponenti.
Primer:

```
dugme.setStyle(
  "-fx-background-color: radial-gradient(radius 100%, gray, red)");
```

Controller class

Dalje je potrebno pridružiti komponentama neke akcije (događaje) i napisati kod za obradu događaja.

Za početak, akcije su jednostavne:
- kada se klikne na dugme:
  - u labeli treba da se prikaže poruka: „POZDRAV ZA KRAJ“ crvenom bojom
  - na standardni izlaz se ispisuje poruka: ZDRAVO!!!
- kada se miš nalazi na površini dugmeta, kursor ima oblik ruke, a kada se pomeri sa površine dugmeta dobija svoj podrazumevani izgled

Dakle, na osnovu ovih zahteva, zaključuje se da će u obradi događaja biti korišćene obe komponente, i dugme i labela.

Definicija controller klase

U odgovarajućem paketu (testSceneBuilder) generisati Java klasu koja implementira interfejs `javafx.fxml.Initializable` (postaviti ovo svojstvo prilikom pravljenja klase).

U klasi obezbediti instancne članice za one komponente koje su potrebne u kodu za obradu događaja (u primeru su to dugme i labela).
Obezbediti i metode za obradu događaja u skladu sa zahtevima. U primeru postoje tri metoda:

- `handleButtonAction(ActionEvent)`
- `handleOnMouseEntered(MouseEvent)`
- `handleOnMouseExited(MouseEvent)`

Šta predstavlja controller klasa?

Controller class je kompajlirana klasa koja implementira „kod u pozadini“ (događaje u pozadini) hijerarhije objekata koja je definisana FXML dokumentom.

Kontroleri se koriste za implementaciju Event Handler-a koji služe za rukovanje događajima vezanih za elemente grafičkog korisničkog interfejsa koji su definisani u FXML dokumentu.

Atribut **fx:controller** (koji će automatski biti dodat u FXML dokument nakon prethodnog izbora kontroler klase) omogućava povezivanje FXML dokumenta i klase za kontrolu.

Postavljanje Contoller klase u SceneBuilder-u

Potrebno je u Scene Builder-u postaviti klasu za kontrolu korenom elementu.
Na taj način se Scene Builder-u omogućuje da pristupi nazivima Event Handler-a i deklarisanih instancnih promenljivih.

Za BorderPane:

- Inspector panel → Code → **Controller class**
  postaviti odgovarajuće ime klase za kontrolu
  (testSceneBuilder.TestSBController)
  pošto je klasa već definisana, klikom na dugme sa strelicom desno, u padajućoj listi biće ponuđena odgovarajuća klasa, koju treba izabrati.

Takođe, za dugme i labelu treba postaviti vrednost atributa **fx:id** na ime odgovarajuće instancne promenljive iz klase za kontrolu koja se na njega/nju odnosi.

- Inspector panel → Code → **fx:id**
  - fx:id : dugme (za dugme)
  - fx:id : labela (za labelu)

```
<Button fx:id="dugme" mnemonicParsing="false"
        onAction="#handleButtonAction"
        onMouseEntered="#handleOnMouseEntered"
        onMouseExited="#handleOnMouseExited"
        prefHeight="30.0"
        prefWidth="80.0" styleClass="dugme, font"
        text="Zdravo"
        textAlignment="LEFT"
        textFill="#990093"
        underline="true"
        wrapText="false">
  <font>
        <Font name="Bookman Old Style Bold Italic" size="14.0" />
  </font>
</Button>
```

```
public class TestSBController implements Initializable {
```

```
@FXML
private Button dugme;

@FXML
private void handleButtonAction(ActionEvent event) {
        System.out.println("ZDRAVO!!!");
        labela.setText("POZDRAV ZA KRAJ!");
        labela.setStyle("-fx-text-fill: red");
}
@FXML
private void handleOnMouseEntered(MouseEvent event) {
        dugme.setCursor(Cursor.HAND);
}
@FXML
private void handleOnMouseExited(MouseEvent event) {
        dugme.setCursor(Cursor.DEFAULT);
}
```

Metod initialize()

Najčešće je dovoljno da se rukovanje događajima definiše na predstavljeni način. Ako je potrebno da se postigne veća kontrola nad ponašanjem kontrolera i elemenata kojima on upravlja, može da se definiše metod:

```
public void initialize();
```

koji se poziva jednom, prilikom implementacije kontrolera, kada su sadržaji FXML dokumenata koji su mu pridruženi u potpunosti učitani, čime se omogućava implementiranoj klasi da izvrši post-procesiranje nad sadržajem, da pristupi resursima koji su korišćeni pri učitavanju dokumenata ili lokacijama koje su korišćene za rešavanje relativnih putanja u dokumentu.

```
<Button fx:id="dugme" mnemonicParsing="false"
        prefHeight="30.0"
        prefWidth="80.0" styleClass="dugme, font"
        text="Zdravo"
        textAlignment="LEFT"
        textFill="#990093"
        underline="true"
        wrapText="false">
  <font>
        <Font name="Bookman Old Style Bold Italic" size="14.0" />
  </font>
</Button>
```

```
public class TestSBController implements Initializable {

@FXML
private Button dugme;

@Override
    public void initialize(URL url, ResourceBundle rb) {
        dugme.setOnAction(new EventHandler<ActionEvent>() {
                @Override
                public void handle(ActionEvent event) {
                     System.out.println("ZDRAVO!!!");
                     labela.setText("POZDRAV ZA KRAJ!");
                     labela.setStyle("-fx-text-fill: red");
```

```
                }
        });

        dugme.setOnMouseEntered(new EventHandler<MouseEvent>() {
                public void handle(MouseEvent event) {
                        dugme.setCursor(Cursor.HAND);
                }
        });

        dugme.setOnMouseExited(new EventHandler<MouseEvent>() {
                public void handle(MouseEvent event) {
                        dugme.setCursor(Cursor.DEFAULT);
                }
        });
    }
}
```

@FXML

Ukoliko se u klasi za kontrolu polja članovi i metodi za upravljanje događajima označe kao **private** ili **protected**, neophodno je označiti ih obeležjem **javafx.fxml.FXML** (`@FXML`), kako bi FXML dokument mogao da im pristupi ili da ih menja.

FXML Loader

Klasa FXMLLoader je odgovorna za učitavanje FXML izvornog dokumenta i vraćanje dobijenog grafa scene:

```
@Override
public void start(Stage primaryStage) throws IOException {
      Parent root = FXMLLoader.load(
                      getClass().getResource("TestSB.fxml"));

      Scene scene = SceneBuilder.create()
                      .root(root)
                      .build();

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

5 Dodatne napomene

Često je potrebno poništiti ono što je urađeno: **Menu → Edit → Undo**.

Moguće je promeniti veličinu komponente da bude jednaka veličini roditelja:
Modify → Fit to Parent Ili **Ctrl+K**.

Moguće je veći broj komponenti organizovati na određen način:

- označiti veći broj komponenti: obeležiti prvu od njih klikom mišem, a potom držeći pritisnut Ctrl, označiti i ostale

- **Menu → Arrange → Wrap in HBox** (primera radi)

Za pregled grafičkog korisničkog interfejsa: **Menu → Preview → Preview in Window**.

Ako se dodaje više istih komponenti, nakon što se doda prva, postupak se može ponoviti izborom opcije: **Edit** → **Duplicate**
Inspector panel → Layout:

- **Vgrow** ili **Hgrow** su postavljeni na **ALWAYS**:
  - element se uvećava/smanjuje po vertikali ili horizontali kada se roditeljski kontejner uvećava/smanjuje

- **MinWidth** je postavljen na **USE_PREF_SIZE**:
  - element će i dalje biti vidljiv čak i kada se veličina prozora aplikacije toliko smanji da se ne mogu videti svi UI elementi