7. Sedma laboratorijska vježba

7.1. JAVAFX

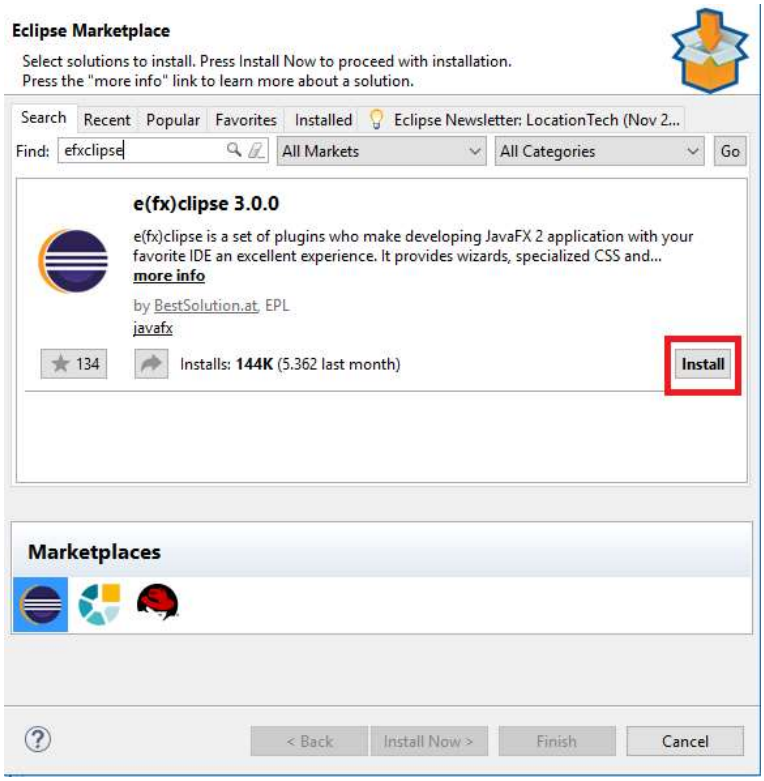
Svrha laboratorijske vježbe je dizajniranje ekrana korisničkog sučelja pomoću JavaFX tehnologije. Aplikacija mora podržavati funkcionalnosti koje su navedene u ovom videu:

https://www.youtube.com/watch?v=GIvPF4IKkO0

7.2. ZADATAK ZA PRIPREMU

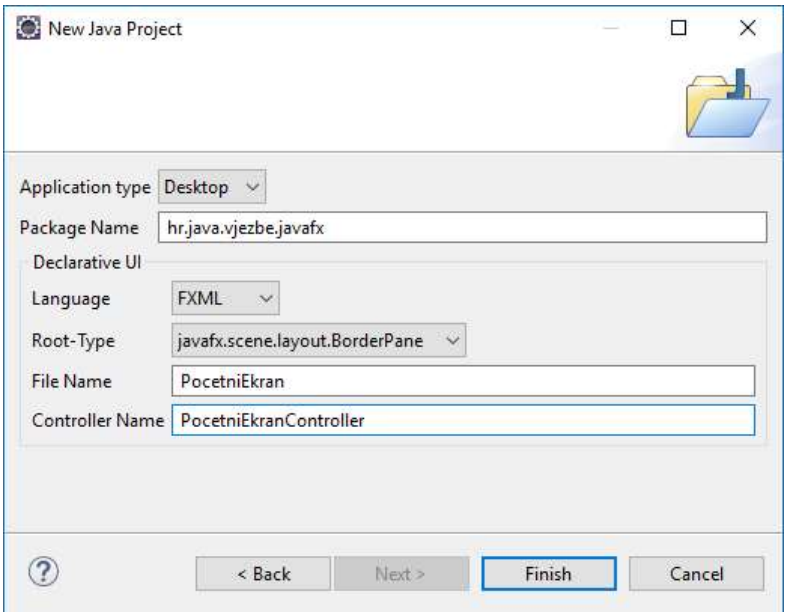
Nastaviti razvoj aplikacije iz šeste laboratorijske vježbe i oblikovati ekrane grafičkog sučelja aplikacije prema koracima u nastavku.

1. Sa stranice http://gluonhq.com/labs/scene-builder/ preuzeti „Scene Builder 2.0" i instalirati ga (na laboratorijskim vježbama je to već obavljeno).
2. Instalirati dodatak „e(fx)clipse" u razvojno okruženje Eclipse korištenjem opcije „Help->Eclipse Marketplace…", upisom „efxclipse"u „Find:" polje i pritiskom na gumb „Install". Nakon dijaloga na slici 1. pojavljuje se ekran za potvrdu instalacije, odabrati prihvaćanje licence te pričekati da proces instalacije završi i potvrditi ponovno pokretanje Eclipsea.



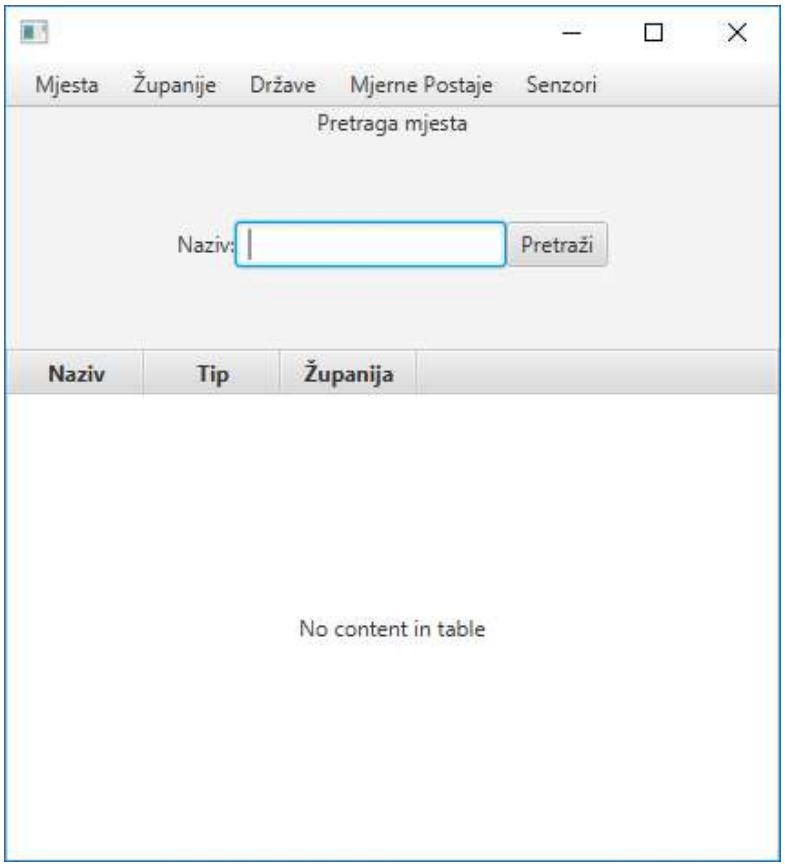
Slika 1. Dijalog za instalaciju dodatka „efxclipse"

3. Kreirati novi „JavaFX" projekt korištenjem opcije „File->New->JavaFX" i nazvati ga prema vlastitom prezimenu sa sufiksom „7". Pritiskom na tipku „Next" (dva puta) prikazuje se dijalog na kojem je potrebno definirati naziv paketa i vrstu aplikacije prema slici 2:



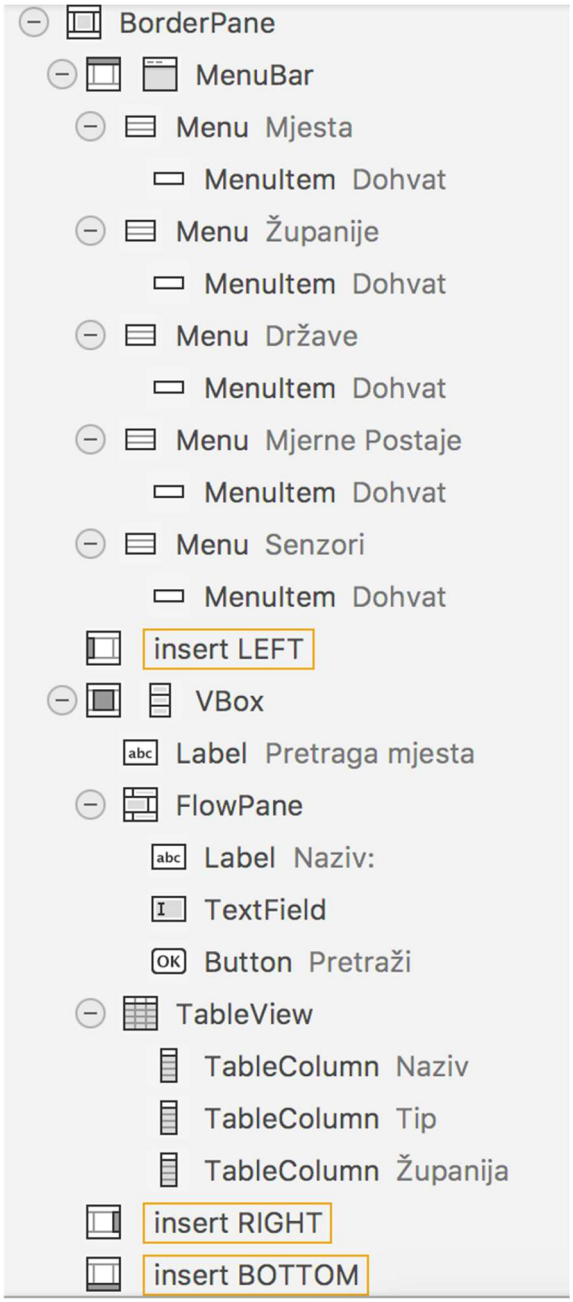
Slika 2. Dijalog za definiranje parametara JavaFX projekta

4. Pretvoriti JavaFX projekt u „Maven Project", prebaciti „pom.xml" datoteku i sve ostale resurse kako bi se mogli učitavati podaci iz datoteke i prikazivati na ekranu.
5. Korištenjem Scene Buildera otvoriti „FXML" datoteku (korištenjem desne tipke miša i odabirom opcije „Open with Scene Builder") dizajnirati ekran aplikacije na način da izgleda ovako:



Slika 3. Izgled ekrana aplikacije

Prijedlog rasporeda i hijerarhije izgleda ovako:



Slika 4. Hijerarhija komponenti

6. Iz prethodne vježbe je potrebno kopirati cijeli paket koji sadrži klase koje predstavljaju entitete u paket za ovu laboratorijsku vježbu.
7. U klasu „Main" potrebno je iz prošle laboratorijske vježbe kopirati metode za dohvat podataka iz datoteka, postaviti im vidljivost na „public" i po potrebi promijeniti kako bi vraćale liste podataka.
8. U klasi „PocetniEkranController" implementirati dohvat podataka o mjestima iz datoteke „mjesta.txt" te podatke popuniti u „TableView" komponentu. Unos teksta u polje „Naziv" i pritisak na gumb „Pretraži" u tablicu mora popunjavati same ona mjesta koji se podudaraju po dijelu ili cijelom nazivu mjesta. To je moguće postići korištenjem sljedećeg programskog isječka unutar „Controller" klase:

```
    private List<Mjesto> listaMjesta;
```

```
	private List<Drzava> listaDrzava;
	private List<Zupanija> listaZupanija;
	private List<Senzor> listaSenzora;
	private List<MjernaPostaja> listaPostaja;

	@FXML
	private TextField mjestaFilterTextField;

	@FXML
	private TableView<Mjesto> mjestaTableView;

	@FXML
	private TableColumn<Mjesto, String> nazivColumn;

	@FXML
	private TableColumn<Mjesto, String> tipColumn;

	@FXML
	private TableColumn<Mjesto, String> zupanijaColumn;

	@FXML
	public void initialize() {
		nazivColumn.setCellValueFactory(new
PropertyValueFactory<Mjesto, String>("naziv"));

		tipColumn
				.setCellValueFactory(
						new
Callback<TableColumn.CellDataFeatures<Mjesto, String>,
ObservableValue<String>>() {

			@Override
			public ObservableValue<String> call(
					CellDataFeatures<Mjesto, String> param) {
			  return new ReadOnlyObjectWrapper<String>
(param.getValue().getVrstaMjesta().toString());
		}});

		zupanijaColumn
		.setCellValueFactory(new
Callback<TableColumn.CellDataFeatures<Mjesto, String>,
ObservableValue<String>>() {

			@Override
			public ObservableValue<String> call(
					CellDataFeatures<Mjesto, String> param) {
			  return new ReadOnlyObjectWrapper<String>
(param.getValue().getZupanija().getNaziv());
		}});
```

```
            listaDrzava = Main.dohvatiDrzave();
            listaZupanija = Main.dohvatiZupanije();
            listaMjesta = Main.dohvatiMjesta();
            listaSenzora = Main.dohvatiSenzore();
            listaPostaja = Main.dohvatiPostaje();
      }

      public void prikaziMjesta() {
            List<Mjesto> filtriranaMjesta = new ArrayList<Mjesto>();
            if (mjestaFilterTextField.getText().isEmpty() == false) {
                  filtriranaMjesta = listaMjesta.stream().filter(m ->
m.getNaziv().contains(mjestaFilterTextField.getText()))
                  .collect(Collectors.toList());
            } else {
                  filtriranaMjesta = listaMjesta;
            }
            ObservableList<Mjesto> listaMjesta = FXCollections
                        .observableArrayList(filtriranaMjesta);
            mjestaTableView.setItems(listaMjesta);
      }
```

U slučaju da korisnik ne unese tekst u polje za definiranje mjesta, potrebno je dohvatiti sva mjesta iz datoteke.

9. Potrebno je obratiti pozornost prilikom „importanja" da sve klase budu iz paketa „javafx".

10. Implementirati izbornik koji će sadržavati opcije za dohvat mjesta, županija, država, mjernih postaja i senzora. Svaki glavni izbornik mora imati jedan podizbornik pod nazivom „Dohvat" koji prikazuje ekran za dohvat podataka o pojedinim entitetima. Ekran je potrebno prikazivati unutar postojećeg ekrana na način da se glavnoj klasi postavi „Pane" objekt koji sadrži elemente za pretragu entiteta. To je moguće implementirati na način da se u glavnoj klasi modificira metoda „start" te doda metoda za prikaz centralnog okvira „setCenterPane":

```
      private static BorderPane root;
      private Stage primaryStage;

      @Override
      public void start(Stage stage) {
            primaryStage = stage;
            try {
                  root =
(BorderPane)FXMLLoader.load(getClass().getResource(
"PocetniEkran.fxml"));
                  Scene scene = new Scene(root,600,400);
                  scene.getStylesheets().add(getClass().getResource(
      "application.css").toExternalForm());
                  primaryStage.setScene(scene);
                  primaryStage.show();
            } catch(Exception e) {
                  e.printStackTrace();
```

```
            }
      }
public static void setCenterPane(BorderPane centerPane) {
            root.setCenter(centerPane);
      }
```

Nakon toga se ta „setCenterPane“ metoda može pozvati u drugim metodama koje prikazuju preostale ekrane za dohvat ostalih entiteta na sljedeći način:

```
public void prikaziEkranZupanije() {
            try {
                  BorderPane zupanijePane = FXMLLoader.load(Main.class
                              .getResource("Zupanije.fxml"));
                  Main.setCenterPane(zupanijePane);
            } catch (IOException e) {
                  e.printStackTrace();
            }
      }
```

11. Na sličan način kao što je opisano, potrebno je implementirati ekrane za pretragu i prikaz ostalih entiteta kreiranjem odgovarajućih „FXML“ i „Controller“ klasa prema YouTube videu navedenom na početku opisa vježbe.

NAPOMENE:

1. Nakon svake promjene u „Scene Builder“ okruženju napraviti „Refresh“ projekta unutar Eclipse razvojnog okruženja.