PT&SCHE

Introduction of part-time and short cycle studies in Serbia

Funded by the Erasmus+ Programme of the European Union

This project has been funded with support from the European Commission. This publication [communication] reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

KI105 - JAVA 3: PROGRAMIRANJE KORISNIČKOG INTERFEJSA

Osnove JavaFX

Lekcija 03

PRIRUČNIK ZA STUDENTE

KI105 - JAVA 3: PROGRAMIRANJE KORISNIČKOG INTERFEJSA

Lekcija 03

OSNOVE JAVAFX

UVOD

Ciljevi ove lekcije

- Obajsniti razliku između JavaFX, Swing i AWT
- Napisati jednostavan JavaFX program i razumeti odnos među faza, scena i čvorova.
- Kreiranje korisničkog interfejsa korišćenjam okna (panes), kontrole korisničke interakcije, i oblika (shapes)
- Automatsko ažuriranje vrednosti svojstava preko povezivanja svojstava.
- Upotreba zajedničkih stilova svojstava i rotacija za čvorove.
- Kreiranje boja upotrebnom klase Color
- Kreiranje fontova upotrebom klase Font
- Kreiranje slika ipotrebom klase Image i kreiranje poglede slika upotrebom klase ImageView
- Čvorovi raspoređivanja upotrebom klasa Pane, StackPane, FlowPane, GridPane, BorderPane, Hbox i Vbox.
- Prikazivanje teksta upotrebom klase Text i kreiranje oblika upotrebom Line, Circle, Rectangle, Elipse, Arc, Polygon i Polyline
- Razvoju višestruko upotrebljive GUI komponente ClockPane za prikaz analognog sata

Referenca: Y. Daniel Liang, INTRODUCTION TO JAVA PROGRAMMING (COMPREHENSIVE VERSION), Tenth Edition, Pearson, ISBN 10: 0-13-376131-2, ISBN 13: 978-0-13-376131-3

Ovo je osnovni udžbenik za ovaj predmet i preporučuje se za dalju upotrebu.

Poglavlje 1

Osnovna struktura JavaFX programa

ZAŠTO JAVAFX?

Swing i AWT su zamenjene sa JavaFX platformom za razvoj aplikacija sa grafikom i GUI. JavaFX aplikacija može da isto radi i u desktop aplikaciji i na veb pretraživaču (Web browser).

JavaFX je novi radni okvir za razvoj Java GUI programa.

Na početku, Java je nudila biblioteku klasa za rad sa grafikom pod nazivom ***Abstract Windows Toolkit (AWT).*** Ove klase su omogućavale razvoj jednostavnog grafilkog korisničkog interfejsa (Graphic User Interface – GUI), ali ne i zahtevnijih aplikacija. Pored toga, ***AWT*** je zavisio od izvršne platforme, te je bio izložen i njihovim bagovima.

AWT je kasnije zamenjen sa novom robusnijom, boljom i fleksibilnijom bibliotekom tzv.***Swing*** komponenata. ***Swing*** komponente se direktno nanose i boje na "platnu" (canvas) u pozdadini upotrebom Java koda.***Swing*** komponente manje zavise od platforme na kojoj se izvršavaju, i upotrebljavaju manje sopstvenih GUI resursa. Swing je razvijen za razvoj desktop (stone) GUI aplikacija.

Swing je sada zamenjen sa potpuno novom GUI platformom koja je nazvana ***JavaFX***. ***JavaFX*** sadrži moderne GUI tehnologije koje omogućavaju razvoj "bogatih Internet aplikacija". Termin "bogata Internet aplikacija" (rich Internet aplikacija – RIA) je uveden termin koji označava veb aplikaciju koja ima ista svojstva i funkcije kao kod desktop aplikacije. ***JavaFX*** aplikacija može da isto radi i u desktop aplikaciji i na čitaču veba (***Web browser***).

JavaFX nudi i jedno novo svojstvo, a to je podrška radu uređaja koji reaguju na dodir prstiju korisnika, kao što su smart telefoni i tableti.

JavaFX podržava primenu 2D i 3D grafike i animacija, prikaz video i audio snimaka, a radi ili kao nezavisna desktop aplikacija ili korišćenjem čitača veba (***Web browser***).

Treba napomenuti i da je znatno lakše naučiti korišćenje, a lakša je i izrada GUI programa pomoću ***JavaFX*** nego sa ***Swing*** komponentama.

Swing je osuđen na "smrt" jer nije predviđena dalja njegova podrška . JavaFX je sada nova platforma za razvoj Internet aplikacija koje rade na različitim platformama, na desktop (stonim) računarima, ručnim uređajima (tableti, smart telefonima) i na vebu.

PRIMER 1 - PROGRAM MYJAVAFX

Apstraktna klasa javafx.application.Application definiše bitni radni okvir za pisanje JavaFX programa.

Ovde će se prikazati osnovna struktura jednog ***JavaFX*** programa. Svaki ***JavaFX*** program proširuje klasu ***javafx.application.Application.***

```
1 import javafx.application.Application;
 2 import javafx.scene.Scene;
 3 import javafx.scene.control.Button;
 4 import javafx.stage.Stage;
 5
 6 public class MyJavaFX extends Application {
 7   @Override // Predefinisanje metoda start() u klasi Application
 8   public void start(Stage primaryStage) {
 9   // Kreiranje scene i postavljanje dugmeta u sceni
 10  Button btOK = new Button("OK");
 11  Scene scene = new Scene(btOK, 200, 250);
 12  primaryStage.setTitle("MyJavaFX"); // Unos nayiva pozornice
 13  primaryStage.setScene(scene); // Postavljanje scene na pozornicu
 14  primaryStage.show(); // prikaz pozornice
 15   }
 16
 17   /**
 18   * Metod main je potreban samo za IDE sa ograničnom podrškom
 19   * za JavaFX. Nije potreban ako se izvršava sa komandne linije.
 20   */
 21   public static void main(String[] args) {
 22  Application.launch(args);
 23   }
 24 }
```

Ovaj program možete testirati i izvršavati bilo korišćenjem konzole računara, bili primenom IDE (Integrated Development Environment) SW alate za razvoj programa, kao što su NetBeans ili Eclipse. Izvršenjem ovog programa dobija se prozor sa OK dugmetom, na slici 1.

Slika 1.1 Program MyJavaFX program prikazuje prozor sa OK dugmetom

Metod launch()(linija 22) je metod klase ***Application*** koji aktivira izvršenje samostalnog J***avaFX*** programa. Ako aktiviranje vršite sa komandne linije, metod ***main()*** nije potreban.

Metod ***main()*** redefiniše metod ***start()*** klase ***Application*** (linija 8). Linija 10 kreira objekat ***Button*** i u objekat ***Scene***, koga kreira konstruktor ***Scene(node, width, height)***.

POZORNICA, SCENA I GLUMCI

JavaFX imenuje klase Stage i Scene po analogiji pozorišta. Pozornica je platforma koja podržava scenu i čvorove (nodes), kao glumce na sceni

Kada se program aktivira sa metodom start(), postavlja se kontrola interakcije sa korisnikom, tzv. UI kontrola u scenu, tj. ***Scene*** objekat, a on se smešta u objekat.***Stage*** , tj. na pozornicu (stage). Kao što je prikazano na slici 2.

Slika 1.2 (a) Pozornica je prozor u kome se prikazuje scena sa čvorovima (b) Moguć je prikaz više pozornica u JavaFX

Objekat Stage, tj. Pozornica je prozor. Tzv. primarni ***Stage*** objekat, tj. primarnu pozornicu JVM automatski kreira. Linija 13 u listingu **MyJavaFX** postavlja scenu (***Scene***) na primarnu pozornicu (***Stage***) (linija 14). ***JavaFX*** imenuje klase ***Stage*** i ***Scene*** po analogiji pozorišta. Pozornica je platforma koja podržava scenu i čvorove (nodes), kao glumce na sceni.

PRIMER 2 - PROGRAM SA VIŠE SCENA

Moguće je kreiratu više scena u jednom programu.

Možete kreirati dodatne pozornice (Stage objekte) u vašem ***JavaFX*** programu. Sledeći listing prikazuje slučaj sa kreiranjem dve pozornicem, prikazanih na slici 2.b. Napominjemo da je ovde izostavljen ***main()*** metod jer je identičan za svaki ***JavaFX*** program. Radi kraćeg prikaza, nadalje ćemo uvek izostavljati metod main(), iako on mora da postoji. Ako se želi da spreči da korisnik može da promeni dimenzije pozornice, poziva se: ***stage.setResizable(false***)

```
1 import javafx.application.Application;
 2 import javafx.scene.Scene;
 3 import javafx.scene.control.Button;
 4 import javafx.stage.Stage;
 5
 6 public class MultipleStageDemo extends Application {
 7   @Override // Predefinisanje metoda start u klasi Application
 8   public void start(Stage primaryStage) {
```

```
9   // Kreiranje scene i postavljanje dugmeta u nju.
10  Scene scene = new Scene(new Button("OK"), 200, 250);
11  primaryStage.setTitle("MyJavaFX"); // Unos naslova pozornice
12  primaryStage.setScene(scene); // Postavljanje scene na pozornicu
13  primaryStage.show(); //Prikaz pozornice
14
15  Stage stage = new Stage(); // Kreiranje pozornice
16  stage.setTitle("Second Stage"); // Unos naslova pozornice
17  // Postavljanje scene sa dugmetom na pozornici
18  stage.setScene(new Scene(new Button("New Stage"), 100, 100));
19  stage.show(); // Prikaz pozornice
20   }
21 }
```

ZADATAK 1

Samostalno kreiranje programa sa više scena.

- Kreirajte program koji koristi dve scene, kao u Primeru 2;
- Pored postojećih dugmadi kreirajte i dve labele kojima je opisano šta će se desiti kada kliknete na dugme sa odgovarajuće scene;
- Ponovo prevedite program i pokrenite ga.

VIDEO - PROMENA SCENA

avaFX Java GUI Tutorial - 4 - Switching Scenes (8,25 minuta)

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

Poglavlje 2

Okna, UI kontrola i oblici

NAŠE POZORIŠTE

Okna (panes), UI kontrole, i oblici (shapes) su podtipovi od Node.

Čvor (*Node*) je vizualna komponenta oblik (***Shape***), pogled slike (***ImageView***), UI kontrola, ili okno (***Pane***).

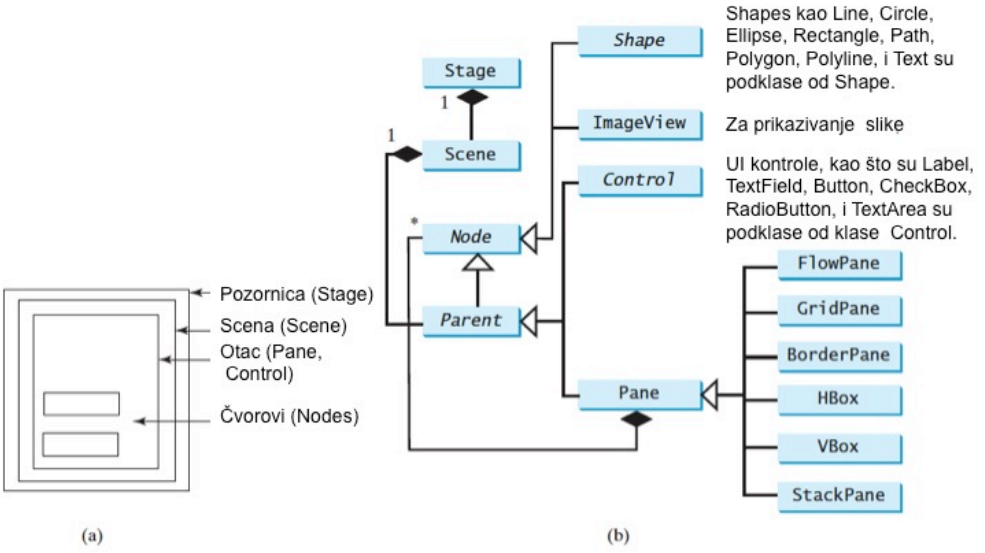
Okna (panes**)** su kontejner klase koje automatski raspoređuju čvorove na određene lokacije i sa određenim veličinama.

Oblik (shape**)** može da predstavlja tekst, liniju, krug, elipsu, luk, pravougaonik, poligon, polilinuju i dr.

UI kontrola (UO controls**)** se odnosi na nalepnicu (label), dugme (button), polje za potvrdu (check box), dugme opcoije (radio button), tekstualno polje (text field), polje za tekst u više redova (text area) i dr.

Scena (*Scene*) je prikaz na pozornici (***Stage***), kao što je prikazano na slici 1.a. UML dijagram klasa za Stage, Scene, Note, Control i Pane je dat na slici 1.b.

Jedna pozornica (Stage) može imati više scena (Scene). Scena sadrži ***Control*** ili ***Pane*** objekte, ali ne i ***Shape*** ili ***ImageView*** objekte. Okno, tj. ***Pane*** objekat, može da sadrži bilo koji podtip ***Node*** objekta. ***Scene*** objekat se kreira konstruktorima ***Scene(Parent, width, height***) ili ***Scene(Parent***). Podklase ***Node*** klase imaju konstruktore bez argumenata za kreiranje unapred definisanih elemenata.



Slika 2.1 a) Java koordinatni sistem b) Krug u centru okna i scene c) Krug nije u centru promrnom veličine okna

PRIMER 3 - DUGME U CENTRU OKNA

Program postavlja dugme u centar okna.

Sledeći listing prikazuje program koji postavlja dugme u okno.

```
1 import javafx.application.Application;
  2 import javafx.scene.Scene;
  3 import javafx.scene.control.Button;
  4 import javafx.stage.Stage;
  5 import javafx.scene.layout.StackPane;
  6
  7 public class ButtonInPane extends Application {
  8   @Override // Redefinicija metoda start() klase Application
  9   public void start(Stage primaryStage) {
 10  // Kreiranje scene i stavljanje dugmenta u scenu e
 11  StackPane pane = new StackPane();
 12  pane.getChildren().add(new Button("OK"));
 13  Scene scene = new Scene(pane, 200, 50);
 14  primaryStage.setTitle("Button in a pane"); // Unos naziva pozornice
 15  primaryStage.setScene(scene); // STavljanje scene na pozornicu
 16  primaryStage.show(); // Prikaz pozornice
 17   }
 18 }
```

Program kreira StackPane (linija 11) i dodaje dugme kao dete okvira (linija 12). Metod getChilidren() vraća primerak od javafx.collections.ObservableList. ObesrvableList se ponaša kao ArrayList koji memoriše kolekciju elemenata.

Pozivom a***dd(e)*** dodaje se element u listu.***StackPane*** postavlja čvorove u centar okna. U omom primeru se koristi samo jedan element. StackPane koristi predefinisanu veličinu čvora. Zato je dugme prikazano u predefenisanoj veličini.

Na slici 2 prikazan je dobijeni prozor izvršenjem programa ButtonInPane.

Slika 2.2 Dugme je u centru okna

PRIMER 4 - KRUG U CENTRU OKNA

Postavljanje kruga u centru okna i scene.

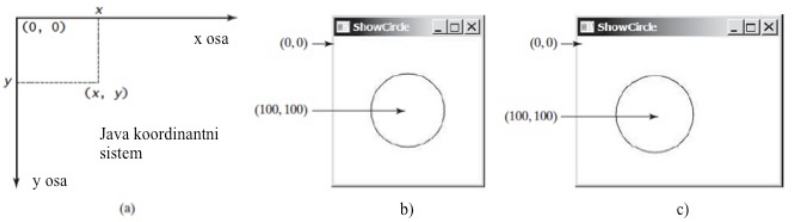
Listing pokazuje program koji postavlja krug u centar okna.

```
1 import javafx.application.Application;
 2 import javafx.scene.Scene;
 3 import javafx.scene.layout.Pane;
 4 import javafx.scene.paint.Color;
 5 import javafx.scene.shape.Circle;
 6 import javafx.stage.Stage;
 7
 8 public class ShowCircle extends Application {
 9   @Override // Predefinisanje metoda start u klasi Application
 10   public void start(Stage primaryStage) {
 11  // Kreiranje kruga i podešavanje njegovoh svojstava
 12  Circle circle = new Circle();
 13  circle.setCenterX(100);
 14  circle.setCenterY(100);
 15  circle.setRadius(50);
 16  circle.setStroke(Color.BLACK);
 17  circle.setFill(Color.WHITE);
 18
 19  // Kreiranje okna koje sadrži krug
 20  Pane pane = new Pane();
 21  pane.getChildren().add(circle);
 22
 23  // Kreiranje scene i postavljanje scene na pozornicu ge
 24  Scene scene = new Scene(pane, 200, 200);
 25  primaryStage.setTitle("ShowCircle"); // Unos naslova pozornice
 26  primaryStage.setScene(scene); // Postavljanje scene na pozornicu
 27  primaryStage.show(); // Prikaz pozornice
 28   }
 29 }
```

Program kreira krug (linija 12) i postavlja ga u centar (100,100) (linije 13-14), što je i centar scene, jer je ona kreirana sa dimenzijama 200x200 (linija 24=. Poluprečnik kruga je 50 (linija 15). Sve jedinice su u pikselima. Linija kruga je crna (linija 16), a popunjen je belom bojom (linija 17).

Program kreira okno (linija 20) i postavlja krug u okno (linija 21). . Okno ima koordinate gornjeg levog ugla (0,0). To je koordinatni početak Java koordinatnog sistema (slika 5.a). . X-kooridnata ide s leva na desno, a z-koordinata ide odozgo nadole.

Okno je postavljeno na scenu (linija 24), a scena na pozornicu (linija 26), Krug je centru pozornice (slika 5.b). Ako se promeni veličina okna, krug onda nije više u njegovom centru.



Slika 2.3 a) Java koordinatni sistem b) Krug u centru okna i scene c) Krug nije u centru promrnom veličine okna

ZADACI 2.1 - 2.4

Proverite vaše razumevanje okana, UI kontrola i oblika

1. Kako kreirate objekt Scene? Kako postavljate scenu na pozornicu? Kako postavljate krug u scenu?
2. Šta je okno? Šta je čvor? Kako postavljate čvor u okno? Da li možete direktno postavljate Shape ili ImageView u Scene?
3. Kako kreirate Circle? Kako postavljate centar kruga i poluprečnik? Kako podešavate boju kružnice i boju unutražnjosti kruga?

VIDEO - KREIRANJE OKVIRA ZA UZBUNU

JavaFX Java GUI Tutorial - 5 - Creating Alert Boxes (10,53 minuta)

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

Poglavlje 3

Povezivanje svojstava

POVEZIVANJE SVOJSTAVA CILJNOG I IZVORNOG OBJEKTA

Možete povezati ciljni i izvorni objekat. Svaka promena izvornog objekta odmah se prenosi na promenu i ciljnog objekta.

JavaFX uvodi novi koncept povezivanja svojstava koji omogućava ciljni objekat da bude povezan sa izvornim objektom. Kada se promeni vrednost izvronog objekta, promeni se i vrednost ciljnog objekta. Ciljni objekat se naziva povezaniom (***binding***) objektom, a oba objekta se nazivaju povezanim (***bindable***) objektima.

U prethodnom listing, krug nije bio više u centru kada je promenjena dimenzija okna. Ako se povežu **centerX** kruga sa dimenzijom ***width*****/2** i **centerZ** kruga sa dimenzijom ***height*****/2** okna, kao što je pokazano u sledećem listingu.

Metod ***bind()*** povezuje ciljni objekt sa izvornim objektom:

```
target.bind(source);
```

Metod b***ind()*** je definisan u interfejsu ***javafx.beans.property.*** Property . Svojstvo povezivanja je primerak o klase ***javafx.beans.property.Property.*** Izvorni objekat je primerak interfejsa***javafx.beans.value.ObservableValue.*** ***ObservableValue*** je entitet koji obuhvata vrednost i dozvoljava da se ona osmatra radi praćenja promena.

JavaFX definiše povezana svojstva za primitivne vrednosti i stringove. Za vrednost tipadouble /float /long /int /boolean , njegova povezan tip je ***DoubleProperty /FloatProperty /***

LongProperty /IntegerProperty /BooleanProperty .

Za neki string, povezano svojstvo je ***StringProperty.***

Ova svojstva su takođe potipovi od ***ObservableValue.*** Zato, mogu da budu upotrebljeni i kao izvorni objekat za povezivanje svojstava.

Svako povezano svojstvo ima svoje geter i seter metode. Na primer za ***centerX*** svojstvo, metod je ***getCenterXPropert***y().

PRIMER 5 - POVEZIVANJA SVOJSTAVA

Program kreira primerak od DoubleProperty upotrebom SimpleDoubleProperty()

Slika 1.a prikazuje konvenciju za definisanje svojstva povezivanja u nekoj klasi, a slika 1.b pokazuje primer u kome je centerX je svojstvo povezivanje tipa***DoubleProperty***.

```
public class SomeClassName {

  private PropertyType x;

  /** Value getter method */
  public propertyValueType getX() { ... }

  /** Value setter method */
  public void setX(propertyValueType value) { ... }

  /** Property getter method */
  public PropertyType
    xProperty() { ... }
}
```

(a) X je svojstvo povezivanja

```
public class Circle {

  private DoubleProperty centerX;

  /** Value getter method */
  public double getCenterX() { ... }

  /** Value setter method */
  public void setCenterX(double value) { ... }

  /** Property getter method */
  public DoubleProperty centerXProperty() { ... }
}
```

(b) centerX je svojstvo povezivanja

Slika 3.1 Svojstvo povezivanja ima geter i seter vrednosti, i geter metod svojstva

Ovde se prikazuje listing drugog primera povezivanja svojstava ***BindingDemo***. Program kreira primerak od DoubleProperty upotrebom SimpleDoubleProperty(1). Program povezuje **d1** sa **d2** (linija 8). Sada su vrednosti za d1 i d2 iste. Posle unosa vrednosti za d2 od 70.2 (linija 11), i vrednost za d1 postaje 70.2 (linija 13). Ovo je primer **unidirekcionog povezivanja**. Ako se promena svojstva prenosi u oba smera, onda je to **bidirekciono povezivanja** promenom ***metoda bindBidirectional().***

```
1 import javafx.beans.property.DoubleProperty;
 2 import javafx.beans.property.SimpleDoubleProperty;
 3
 4 public class BindingDemo {
 5   public static void main(String[] args) {
 6   DoubleProperty d1 = new SimpleDoubleProperty(1);
 7   DoubleProperty d2 = new SimpleDoubleProperty(2);
 8   d1.bind(d2);
 9   System.out.println("d1 is " + d1.getValue()
 10      + " and d2 is " + d2.getValue());
 11   d2.setValue(70.2);
 12   System.out.println("d1 is " + d1.getValue()
 13  + " and d2 is " + d2.getValue());
 14   }
 15 }
```

Dobijen rezultat:

d1 is 2.0 and d2 is 2.0

d1 is 70.2 and d2 is 70.2

ZADACI 3.1 - 3.4

Proverite da li ste dobro razumeli povezivanje svojstava

1. Šta je svojstvo povezivanja? Koji interfejs definiše svojstvo povezivanja? Koji interfejs definiše izvorni objekat? Koji su tipovi objekata povezivanja za int, long, float, double, i bollean? Da li su Integer i Double svojstva povezivanja? Da li Integer i Double se mogu upotrebiti kao izvorni objekti za povezivanja?
2. U skladu sa konvencijuom za nazive za JavaFX svojstva povezivanja, a za svojstvo povezovanja sa nazivom age tipa IntegerProperty, koji je geter metod vrednosti, ster metod vrednosti, i geter metod svojstva?
3. Da li možete kreirati objekat od IntegerPropertz upotrebom new IntegerPropretz(3)? Ako ne, koji je ispravan način njegovog kreiranja? Koji je izlaz ako se linija 8 zameni sa d1.bind(d2.multiply()) u listingu programa BindingDemo? Koji je izlaz ako se linija 8 zameni sa d1.bind(d2.add(2)) u listingu BindingDemo?
4. Šta je unidirekciono povezivanje i šta je bidirekciono povezivanje? Da li se svojstva povezivanja mogu bidirekciono povezati? Koji je iskaz za definisanje bidirekciono povezivanja svojstva d1 sa svojstvom d2?

VIDEO - KOMUNIKACIJA IZMEĐU PROZORA

JavaFX Java GUI Tutorial - 6 - Communicating Between Windows

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

Poglavlje 4

Zajednička svojstva i metodi za čvorove

KLASA NODE

Apstraktna klasa Node definiše mnoga svojstva i metode koji su zajedničke za sve čvorove.

Čvorovi dele mnoga zajednička svojstva. Ovde ćemo analizirati dva takva svojstva stila i svojstva rotacije.

JavaFX svojstva stila su slične sa CSS (cascade style sheets) koji specificiraju stilovi za HTML elemente na veb stranice. Zato, svojstva stilova u JavaFX se nazivaju JavaFX CSS. U JavaFX svojstva stilova se definišu sa prefiksom –fx. Svaki čvor ima svoje sopstvene stilove. Detaljniju informaciju možete naći na adresi: http://docs.oracle.com/javafx/2/api/javafx/scene/doc-files/cssref.html

Sintaksa za postavljanje stila je ***styleName*** vrednost. Višestruka svojstva nekog čvora se može podesiti zajedno, ali odvojeni sa (;). Na primer sledeći iskaz:

```
circle.setStyle("-fx-stroke: black; -fx-fill: red;");
```

Postavlja dva svojstva stilova za krug. Ovaj iskaz je ekvivalentan sa sledećim iskazima:

```
circle.setStroke(Color.BLACK);
circle.setFill(Color.RED);
```

Ako se JavaFX CSS nepravilno koristi, program će raditi, ali će ignorisati stilove.

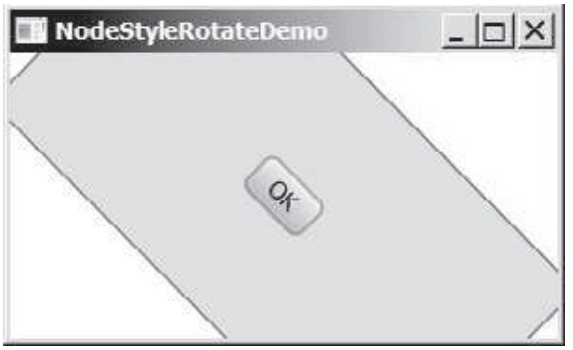
Svojstvo ***rotate*** omogućava vam da specificirate ugao u stepenima za rotaciju čvora oko centra. Pozitivna vrednost ugla označava ugao u smeru sklayaljke na satu. U suprotnom, kreće se suprotno okretanju skazaljke na satu. Na primer, sledeći iskaz rotira dugme za 80 stepeni.

```
button.setRotate(80);
```

PRIMER 5 - SVOJSTVO ROTACIJE

Primer prikazuje program koji kreira dugme, postavlja njegov stil, i stavlja ga u okno, a onda ga rotira za 45 stepeni, i postavlja crvenu boju linije na granici i boju pozadine dugmeta.

Sledeći listing prikazuje program koji kreira dugme, postavlja njegov stil, i stavlja ga u okno. Onda se rotira za 45 stepeni i postavlja stil za crvenu boju linije na granici i boju pozadine svetlo sive, kao što je pokazano na slici 1.



Slika 4.1 Postavljen stil okvira i zarotiran za 45 stepeni.

Kao što se vidi, rotacijom okna, zarotiralo se i dugme.

Klasa ***Node*** sadrži mnoge korisne metode koji se mogu primeniti na sve čvorove. Na primer, možete iskoristiti metod ***contains(double x, double y)*** za testiranje da li je tačka (x, y) unutar grsnica čvora.

```
1 import javafx.application.Application;
 2 import javafx.scene.Scene;
 3 import javafx.scene.control.Button;
 4 import javafx.stage.Stage;
 5 import javafx.scene.layout.StackPane;
 6
 7 public class NodeStyleRotateDemo extends Application {
 8   @Override // Predefinisanje metoda start u klasi Application
 9   public void start(Stage primaryStage) {
 10  // Kreiranje scene i postavljanje dugmeta u scenu
 11  StackPane pane = new StackPane();
 12  Button btOK = new Button("OK");
 13  btOK.setStyle("-fx-border-color: blue;");
 14  pane.getChildren().add(btOK);
 15
 16  pane.setRotate(45);
 17  pane.setStyle(
 18    "-fx-border-color: red; -fx-background-color: lightgray;");
 19
 20  Scene scene = new Scene(pane, 200, 250);
 21  primaryStage.setTitle("NodeStyleRotateDemo"); // Unos naziva scene
 22  primaryStage.setScene(scene); // Stavljanje scene na pozornicu
```

```
23  primaryStage.show(); // Prikaz pozornice
24   }
25 }
```

ZADACI 4.1 - 4.2

Utvrdite vaše razumevanje zajedničkih svojstava i metoda za čvorove

1. Kako postavljate stil čvora sa crvenom bojom granične linije? Promenite kod za postavljanje crvene boje teksta na dugmetu
2. Da li možete da zarotirate okvir , tekst ili dugme? Promenite kod tako da dugme zarotirate za 15 stepeni suprotno okretanju skazaljke na satu.

VIDEO - SVOJSTVA

JavaFX Java GUI Tutorial - 28 - Properties (11,16)

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

VIDEO - POVEZIVANJE SVOJSTAVA

JavaFX Java GUI Tutorial - 29 - Binding (4,35)

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

VIDEO - PRIMER POVEZIVANJE SVOJSTAVA

JavaFX Java GUI Tutorial - 30 - Binding Properties Example (5,12 minuta)

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

Poglavlje 5

Klase Color i Font

SVOJSTVA KLASE COLOR

Klasa Color služi za kreiranje boja

JavaFX koristi apstraktnu klasu ***Paint*** za bojenje nekog čvora. Klasa ***javafx.scene.paint.Color*** je konkretna podklasa klase ***Paint*** koja sadrži metode za definisanje boja (slika 1).

Klasa ***Color*** koristi tzv. ***RGBA*** model, koji definiše neku boju kombinacijom crvene (R), plave (B), zelene (G) boje i stepena prozirnosti – alfa vrednost (A). Objekat ***Color*** je nepromenljiv, posle kreiranja, jer mu se svojstva ne mogu menjati. Primenom metoda ***brighter()*** i ***darker()*** menjaju se vrednosti R, G i A, ali se vraća novi ***Color*** objekat, sa istom vrednosti prozračnosti A.

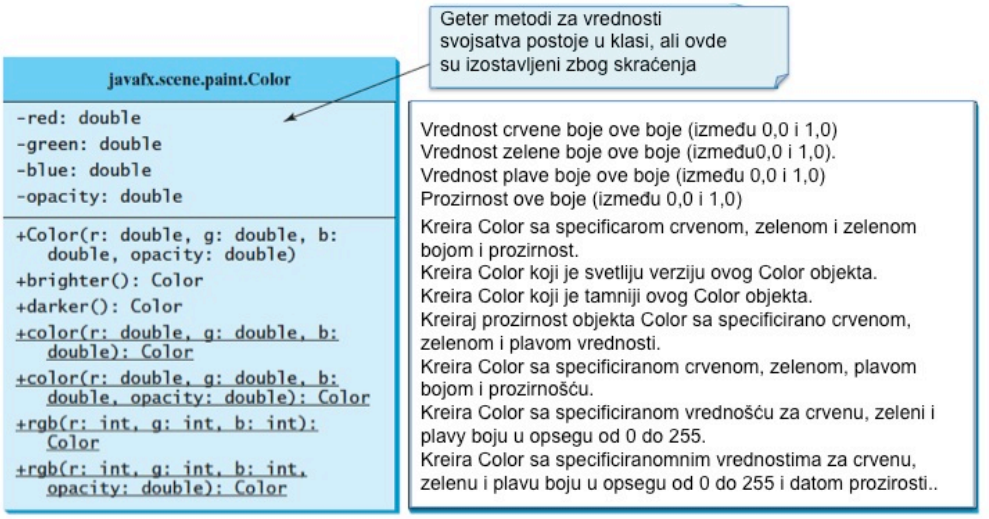
Objekat ***Color*** se može dobiti i sledećim kontruktorima***: color(r, g, b), color(r,g, b, opacity), rgb(r, g, b), and rgb(r, g, b, opacity).***Postoji i 18 unapred definisanih boja. Na primer:

```
circle.setFill(Color.RED);
```

Objekat Color se kreira primenom sledećeg konstruktora:

```
public Color(double r, double g, double b, double opacity);
```

Gde argumenti r, g i b određuju vrednost svojstva za crvenu, zelenu i plavu boju, gde vrenosti se kreću između 0.0 (najtamnija) i 1.0 (najsvetlija). Ragument opacity oy+značava prozirnost, i kreće se od 0.0 (prozirno) do 1.0 (neprozirno).



Slika 5.1 Klasa Color sadrži informacije o bojama

SVOJSTVA KLASE FONT

Klasa Font definiše ime fonta (vrste slova) i njegovu postavku i veličinu

Klasa ***javafx.scene.text.Font*** služi za kreiranje fontova. Objekat klase Font se kreira konstruktorima ili statičkim metodima. Statički metod ***getFamilies()*** daje listu raspoloživih fontova (npr. ***Ariel, Times New Roman, Courier***), u vidu objekta***ArrayList***.

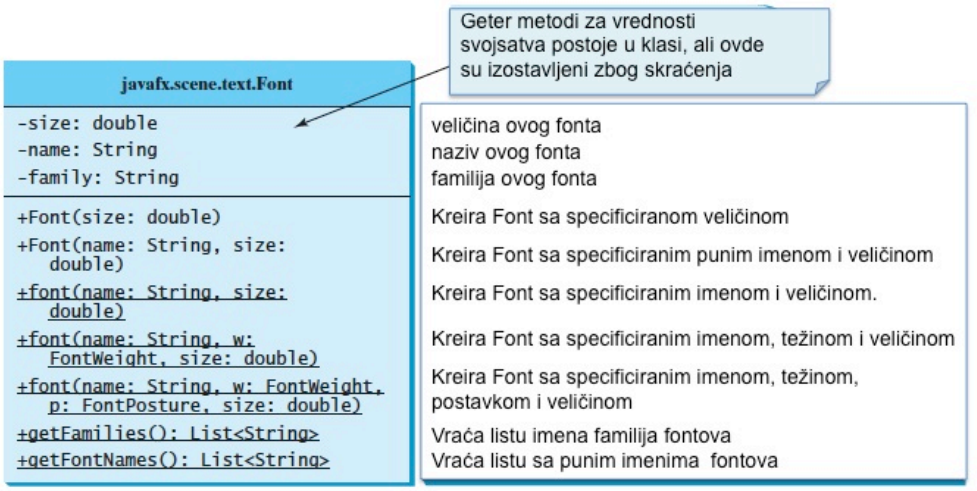
Objekat ***Font***, pored naziva fonta i njegove veličine, definiše i njegovu postavku (posture) te može biti:sa:

- *kosim slovima* ili *kurzivom* ***(italic***)
- **izraženim slovima** (***bold***) i
- normalnim slovima (***regular***)

Postavka slova se definiše izrazom: ***FontPosture.ITALIC*** ili ***FontPosture.BOLD*** ili ***FontPosture.REGULAR.***

Na primer, sledeći iskazi kreiraju dva fonta:

```
Font font1 = new Font("SansSerif", 16);
 Font font2 = Font.font("Times New Roman", FontWeight.BOLD,
      FontPosture.ITALIC, 12);
```



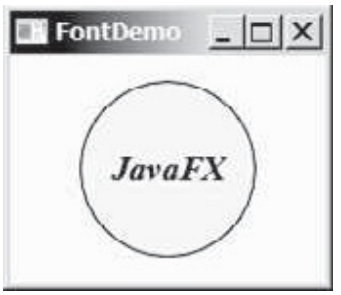
Slika 5.2 Klasa Font sadrži informacije o fontovima

PRIMER 6 - POGRAM SA COLOR I FONT KLASOM

Postavljanje kruga sa nalepnicom "JavaFX" u centar scene.

Na slici 2 prikazan je krug sa nalepnicom JavaFX koji je postavljen u centar scene. Treba koristiti sledeći font: Times New Roman, izraćena slova, kurziv, i veličina 20.

.



Slika 5.3 Nalepnica je stavljena preko kruga kojije u centru scene

Program kreira okvir StackPane (linija 14) i dodaje mu krug i nalepnicu (oznaku) (linije 21 i 27). Ta dva iskaya se mogu zameniti i sa jednim iskazom:

```
pane.getChildren().addAll(circle, label);
```

StackPlan postavlja čvorove u centar, a njih stavlja jedan na drugi. Postavlja se popuna kruga bojom (linija 20). Kreira se nalepnica i njen font (linija 25). Promenom dimenzija prozora ostaju u centru jer je ***StackPane*** automatski postavlja čvorove u centar.

Daje se listing programa koji to realizuje

```
1 import javafx.application.Application;
 2 import javafx.scene.Scene;
 3 import javafx.scene.layout.*;
```

```
4 import javafx.scene.paint.Color;
5 import javafx.scene.shape.Circle;
6 import javafx.scene.text.*;
7 import javafx.scene.control.*;
8 import javafx.stage.Stage;
9
10 public class FontDemo extends Application {
11   @Override // Predefinisanje metoda start u klasi Application
12   public void start(Stage primaryStage) {
13  // Kreiranje okna u koje se smešta krug
14  Pane pane = new StackPane();
15
16  // Kreiranje kruga i unos njegovih svojstava
17  Circle circle = new Circle();
18  circle.setRadius(50);
19  circle.setStroke(Color.BLACK);
20  circle.setFill(new Color(0.5, 0.5, 0.5, 0.1));
21  pane.getChildren().add(circle); // Dodavanje kruga u okno
22
23  // Kreiranje nalepnice i unos njenih svojstava
24  Label label = new Label("JavaFX");
25  label.setFont(Font.font("Times New Roman",
26  FontWeight.BOLD, FontPosture.ITALIC, 20));
27  pane.getChildren().add(label);
28
29  // Kreiranje scene i njeno postavljanje na pozornicu
30  Scene scene = new Scene(pane);
31  primaryStage.setTitle("FontDemo"); // Unos naziva pozornice
32  primaryStage.setScene(scene); // Stavljanje scene na pozornicu
33  primaryStage.show(); // Prikaz pozornice
34   }
35 }
```

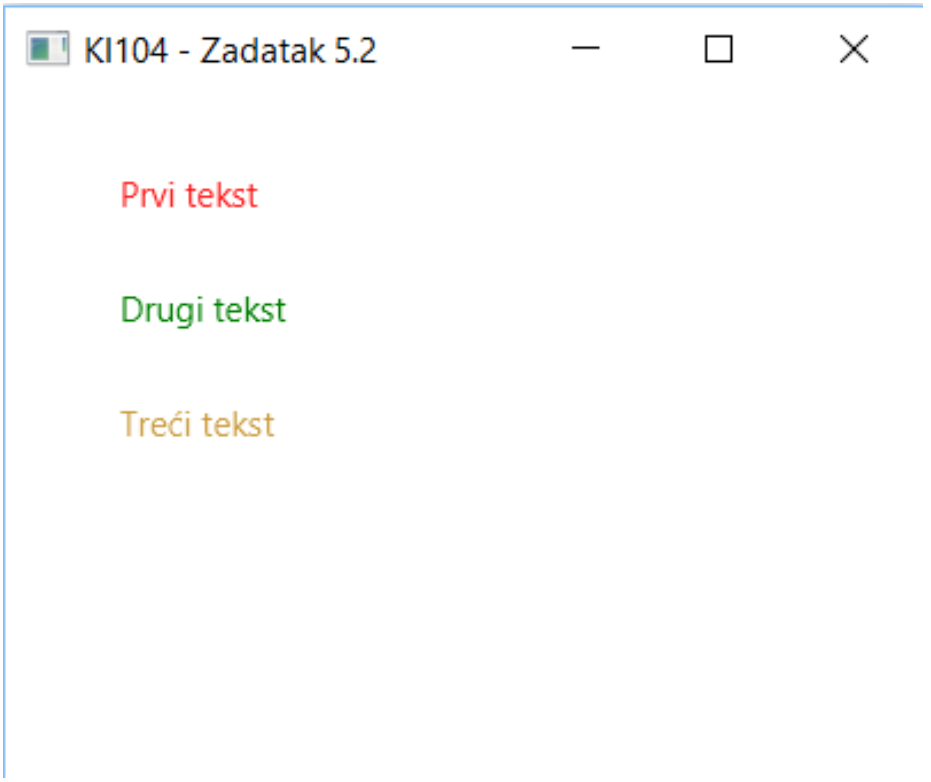
ZADACI 5.1 - 5.2

Kreiranje programa koji podešava boju i font teksta.

1. Kreirajte JavaFX program koji koristi klase Font i Color i daje izlaz prikazan sledećom slikom.

Slika 5.4 Zadati izlaz iz programa

2. Kreirajte JavaFX program koji koristi klase Font i Color i daje izlaz prikazan sledećom slikom. Tekst se razlikuje po veličini i boji fonta,



Slika 5.5 Zadati izlaz iz programa

Poglavlje 6

Klase Image i ImageView

SVOJSTVA KLASA IMAGE I IMAGEVIEW

Klasa Image predstavlja grafičku sliku, a klasa ImageView prikazuje sliku.

Klasa ***javafx.scene.image.Image*** predstavlja grafičku sliku i koristi se za unošenje slike sa specificirane datoteke ili sa URL. Primeri:

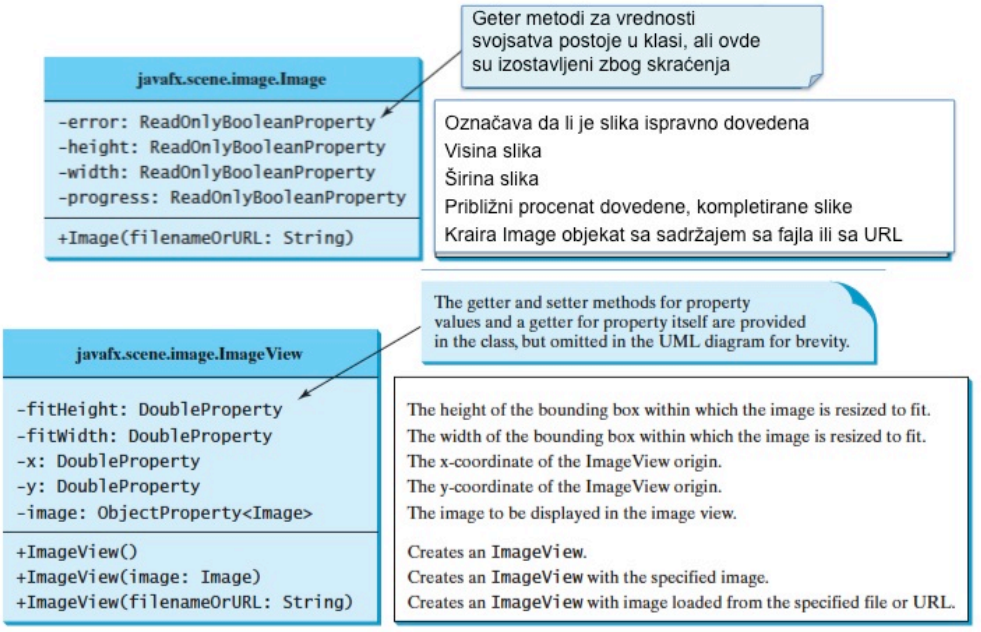
- ***Image***("image/us.gif")
- ***Image***("http://www.cs.armstrong.edu/liang/image/us.gif")

Klasa ***javafx.scene.image.ImageView*** je čvor za prikazivanje neke slike. ***ImageView*** se može kreirati iz ***Image*** objekta. Na primer:

```
Image image = new Image("image/us.gif");
 ImageView imageView = new ImageView(image);
```

Alternativno, može se kreirati ***ImageView*** direktno iz datoteke ili sa URL:

```
ImageView imageView = new ImageView("image/us.gif");
```

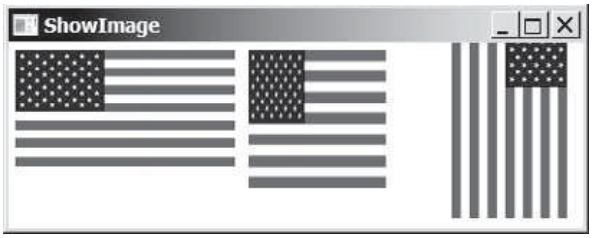


Slika 6.1 Klase Image i ImageView

PRIMER 7 - PRIKAZ SLIKE

Prikazuju se tri ImageView objekta koji prikazuju istu sliku, tj. objekat Image.

Na slici 2 prikazan je prozor sa tri slike. Prikazan listing prikazuje program koji realizuje prikaz ove tri slike.



Slika 6.2 Slika koja sadrži tri prikaza slika u oknu

Program kreira ***Hbox*** (linija 14) . To ja okno koji horizontalno postavlja čvorove u jednom redu. Program kreira ***Image*** i onda i ***ImageView*** za prikaz***Image***. I postavlja ***ImageView*** u ***Hbox*** (linija 17) Program kreira i drugi ***ImageView*** (linija 19), postavlja svojstva ***fitHight*** i ***fitWidth*** (linije 20-21) i postavljanja ***ImageView*** u ***Hbox*** (linija 22). Program kreira i treći ***ImageView*** (linija 24), rotira ga za 90 stepeni (linija 25) i postavlja ga u ***Hbox*** (linija 26). Metod ***setRotate*** je definisan u klasi ***Note*** i može se koristite za bilo koji čvor. Image se ovde deli od strane tri ***ImageView***, ali se čvor ***ImageView*** ne može da deli. Ne može se postaviti više puta.

Datoteka sa slikom mora da bude u istim direktorijumu sa class datotekom. UL mora da sadrži i http:// deo adrese.

```
1 import javafx.application.Application;
 2 import javafx.scene.Scene;
 3 import javafx.scene.layout.HBox;
 4 import javafx.scene.layout.Pane;
 5 import javafx.geometry.Insets;
 6 import javafx.stage.Stage;
 7 import javafx.scene.image.Image;
 8 import javafx.scene.image.ImageView;
 9
 10 public class ShowImage extends Application {
 11   @Override // Predefinisanje metoda start u klasi Application
 12   public void start(Stage primaryStage) {
 13  // Kreiranje okna za smeštaj slike
 14  Pane pane = new HBox(10);
 15  pane.setPadding(new Insets(5, 5, 5, 5));
 16  Image image = new Image("image/us.gif");
 17  pane.getChildren().add(new ImageView(image));
 18
 19  ImageView imageView2 = new ImageView(image);
 20  imageView2.setFitHeight(100);
```

```
21  imageView2.setFitWidth(100);
22  pane.getChildren().add(imageView2);
23
24  ImageView imageView3 = new ImageView(image);
25  imageView3.setRotate(90);
26  pane.getChildren().add(imageView3);
27
28  // Kreiranje scene i njeno postavljanje na pozornicu
29  Scene scene = new Scene(pane);
30  primaryStage.setTitle("ShowImage"); // Unos naslova pozornice
31  primaryStage.setScene(scene); // Postavljanje scene na pozornicu
32  primaryStage.show(); // Prikaz pozornice
33   }
34 }
```

ZADATAK 6

Samostalno vežbanje rada sa klasama Image i ImageView u JavaFX programu.

1. Kreirajte JavaFX program koji koristi klase Image i ImageView i daje izlaz kao što je na sledećoj slici. Koristite neke vaše slike umesto prikazane slike.

Slika 6.3 Primer očekivanog izlaza programa

VIDEO - ZATVARANJE SVOJSTAVA PROGRAMA

JavaFX Java GUI Tutorial - 7 - Closing the Program Properly (8,21 minuta)

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

Poglavlje 7

Okna rasporeda

UVOD

JavaFX obezbeđuje više okna koja automatski raspoređuju čvorove na željene lokacije sa željenom veličinom.

JavaFX obezbeđuje puno tipova okna za organizovanje čvorova u kontejneru:

- ***Pane*****:** Osnovna klasa za okna za raspoređivanje. Sadrži metod ***getChilidren*** koji vraža listu čvorova u okviru.
- ***StackPane*****:** Postavlja čvorove jedan iznad drugog u centru okvira.
- ***FlowPane*****:** Postavlja čvorove iz reda u red po horizontali ili kolona po kolona po vertikali.
- ***GridPane*****:** Postavlja čvorove u ćelije dvodimenzione rešetke (matrice).
- ***BorderPane***: Postavlja čvorove na vrh, desno, dole, levo i u centar okna.
- ***Hbox*****:** Postavlja čvorove u jedan red (horizontalu).
- ***Vbox***: Postavlja čvorove u jednu kolonu (vertikalu)

Okno (***pane***) se koristi kao platno (canvas) za prikazivanje **oblika** (shapes). ***Pane*** je osnovna klasa za sve ostale specijalizovane (pot)klase za okna. Svaki okvir sadrži listiučvorova koje sadrži. Ta lista je primerak klase ***ObservableList*** koji se dobija pozivom metoda ***getChilidren()***. Sa metodom *add(node)* mogu se čvorovi dodavati u okno, tj. u listu, a sa metodom *addAll(node1, node2, ...)* dodaje promenljivi broj čvorova u okvir.

7.1 FlowPane

KLASA FLOWPANE

FlowPane je okvir koji ređa čvorove po horizontali (red po red) ili po vertikali (kolona po kolona).

FlowPane ređa čvorove u oknu horizontalno, sleva udesno, ili vertikalno, odozgo nadole, po redosledu njihovog dodavanja. Kada se popuni jedan red ili kolona, počinje ređanje u drugom redu ili koloni. Primenom dve konstante:

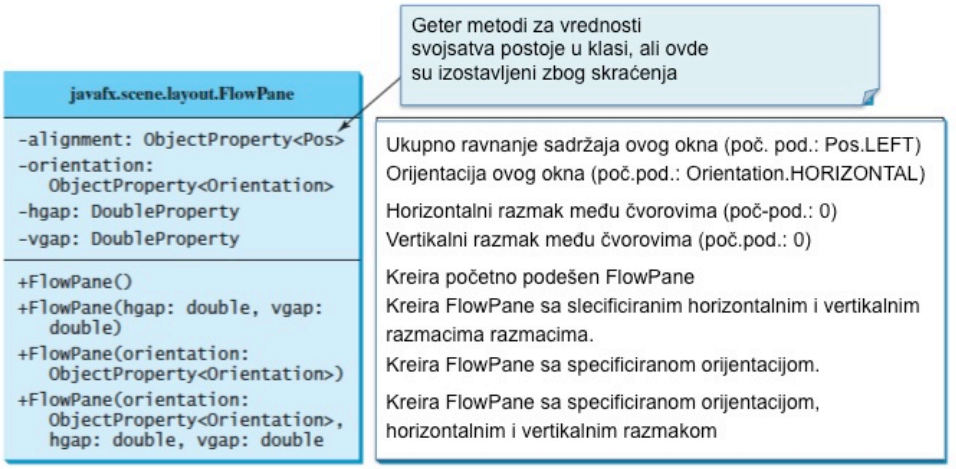
- ***Orientation.HORIZONTAL***
- ***Orientation.VERTICAL***

Određujete da li želite ređanje po horizontali ili vertikali.

Možete da specificirate i razmak između čvorova u pikselima (i po horizontali i po vertikali.

Ravnanje polja sa podacima, razmaci među njima **hgap** i **vgap**, su povezana svojstva (binding properties). Svaki od njih ima svoj getter metod, na primer, ***getHgap()*** koji vraća svoju vrednost, a setter metod, npr., ***setGap(double***) unosi željenu vrednost, a geter metod vraća

vrednost svojstva (na primer, ***hGapeProperty().*** Za polje podataka tipa ObjectProperty<T> vrednosni geter metod vraća vrednost tipa T , a geter metod za svojstva, vraća vrednost svojstva tipa ObjectProperty<T>.



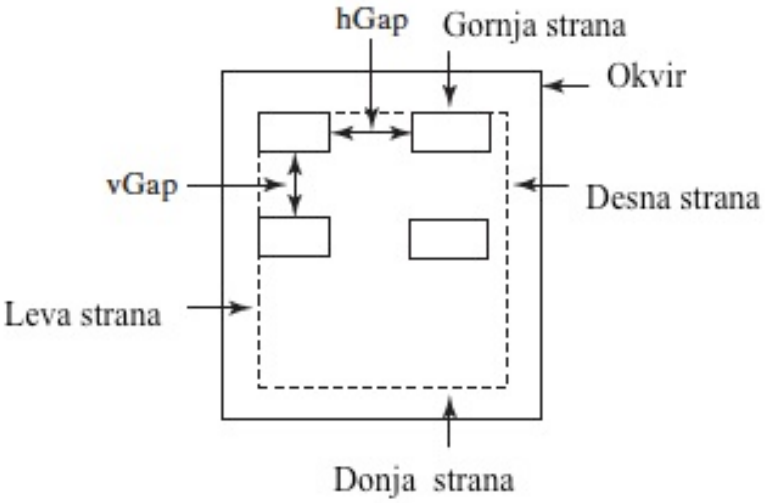
Slika 7.1.1 FlowLayout raspoređuje čvorove po horizontali ili po vertikali

NAČIN RASPOREĐIVANJA ČVOROVA U OKNU

Objekat Insets definiše veličinu granice okna. Svojstva hGap i vGap definišu razmake susednih čvorova u horizontalnom i vertikalnom pravcu.

Konstruktor za***Insets(11,12,13,14***) kreira ***Insets*** sa veličinom granice gore (11), desno (12), dole (13) i levo (14) u pikselima, kao što je pokazano na slici 3. Možete koristiri i konstruktor ***Insets***(value) koji kreira okno sa jednakom granicom na sve četiri strane. Svojstva ***hGap*** i ***vGap*** u linijama 15-16 specificiraju horizontalni i vertikalni razmak između dva čvora, kao što je pokazano na slici 3.

.



Slika 7.1.2 Razmaci hGap i vGap između čvorova.

Svaki ***FlowPane*** sadrži objekat ***ObservableList*** koji sadrži čvorove okna. Ova se lista dobija pozivom metoda ***getChilidren*** () (linija 19). Dodavanje novih čvorova se vrši metodima ***add***(node) i ***addall***(node1, node2...).Uklanjanje čvorova iz liste se vrši pozivanje metode ***remove(***Node) i ***removeAll***()..

Program dodaje natpise u okviru (linije 19-24).Pozivom metoda ***tfMi.setPrefColumnCount(1)*** postavlja jednu kolonu za ***TextField*** objekat za ***MI*** tekstualno polje (linija 22). Program deklariše eksplicitnu referencu***tfMi*** za ***TextField*** objekat ***MI***. Eksplicitna referenca je neophodna jer moramo da direktno pristupamo objektu da bi podesili ***prefColumnCount*** svojstvo.

Program dodaje okno u scenu (linija 27), postavlja scenu na pozornicu (linija 29) i prikazuje pozornicu (linija 30).

Treba primetiti da se promenom dimenzija prozora, čvorovi se automatski preuređuju da bi se uklopili u nove dimenzije okna. Na slici 2.a prvi red ima tri čvora, ali na slici 2.b prvi red ima četiri čvora, jer je povećana širina prozora.

Jedan čvor se može dodati samo u jedno okno i samo jedanput. Dodavanje nekog čvora više puta u isto okno ili u različita okna prourokovaće grešku izvršenja.

PRIMER 8 - PRIMENA FLOWPANE

Zadatak je da se dodaju natpisi i tekstualna polja u okno FlowPane

Potrebno je dodati natpise i tekstualna polja u ***FlowPane*** okno (slika 2).

Slika 7.1.3 Čvorivi se ređaju horizontalno, red po red

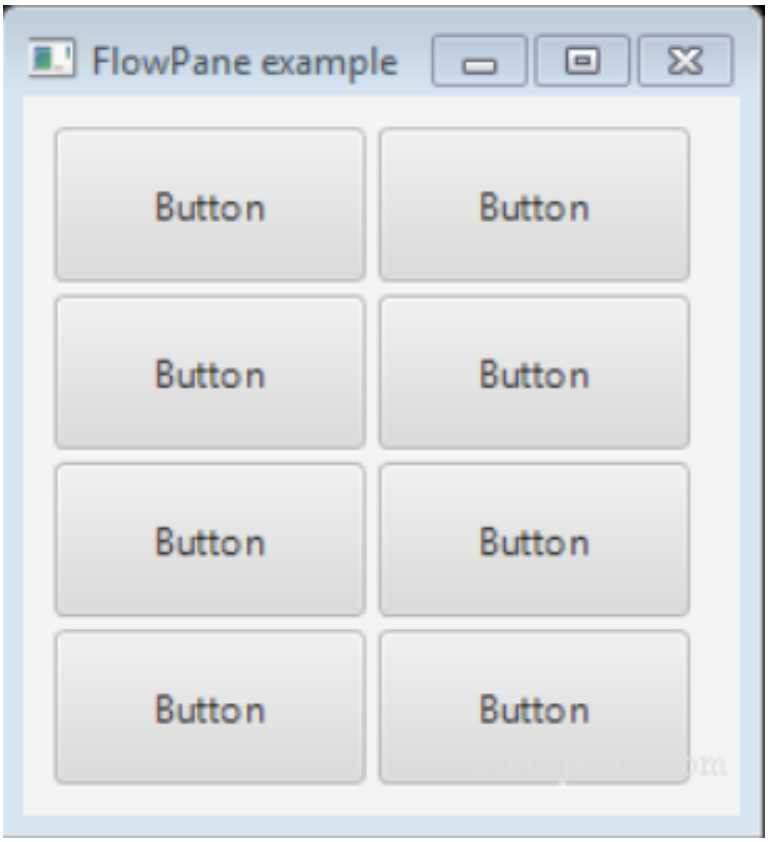
Program kreira***FlowPane*** (linija 13) i postavlja svoje svojstvo ispunjenja (padding) sa objektoma***Insets*** (linija 14). On specificira veličinu i granicu okvira.

```
1 import javafx.application.Application;
 2 import javafx.geometry.Insets;
 3 import javafx.scene.Scene;
 4 import javafx.scene.control.Label;
 5 import javafx.scene.control.TextField;
 6 import javafx.scene.layout.FlowPane;
 7 import javafx.stage.Stage;
 8
 9 public class ShowFlowPane extends Application {
 10   @Override // Redefinisanje start metoda klase Aplikacije
 11   public void start(Stage primaryStage) {
 12  // Kreiranje okvira i unos njegovih svojstava
 13  FlowPane pane = new FlowPane();
 14  pane.setPadding(new Insets(11, 12, 13, 14));
 15  pane.setHgap(5);
 16  pane.setVgap(5);
 17
 18  // Postavljanje čvorova u okviru
 19  pane.getChildren().addAll(new Label("First Name:"),
 20  new TextField(), new Label("MI:"));
 21  TextField tfMi = new TextField();
 22  tfMi.setPrefColumnCount(1);
 23  pane.getChildren().addAll(tfMi, new Label("Last Name:"),
 24  new TextField());
 25
 26  // Kreiranje scene i njeno postavljenje  na pozornici
 27  Scene scene = new Scene(pane, 200, 250);
 28  primaryStage.setTitle("ShowFlowPane"); // Unos naziva pozornice
 29  primaryStage.setScene(scene); // Postavljanje scene na pozornicu
 30  primaryStage.show(); // Prikaz pozornice
 31   }
 32 }
```

ZADACI 7.1 - 7.2

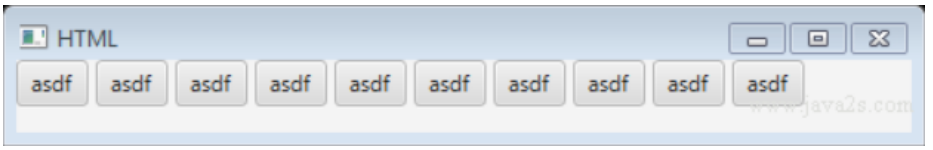
Samostalno vežbanje rada sa menadžerom FlowPane u JavaFX programu.

1. Kreirajte JavaFX program koji koristi FlowPane daje izlaz kao što je na sledećoj slici.



Slika 7.1.4 Očekivani izlaz iz programa

2. Kreirajte JavaFX program koji koristi FlowPane daje izlaz kao što je na sledećoj slici. Koristite neke vaše slike umesto prikazane slike. Koristite petlju da postavite prikazani niz dugmića.



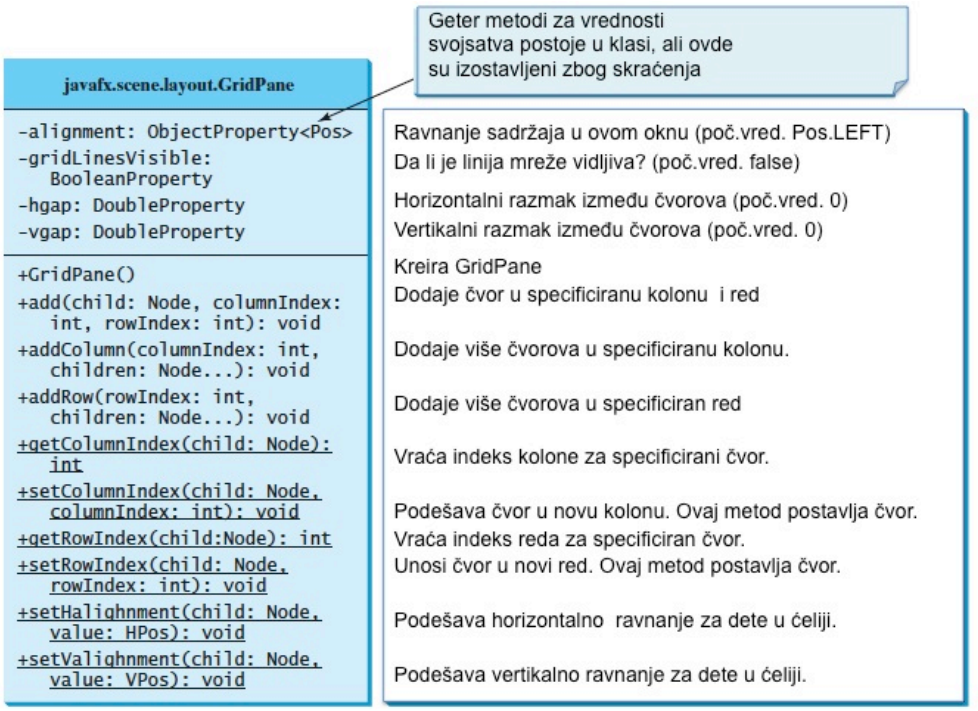
Slika 7.1.5 Očekivani izlaz iz programa

7.2 GridPane

KLASA GRIDPANE

GridPane uređuje čvorove u matričnom obliku

GridPane uređuje čvorove u matričnom obliku. Čvorovi se postavljaju u specificirane kolone i redove. UML dijagram klase GridPane je dat na slici 1.

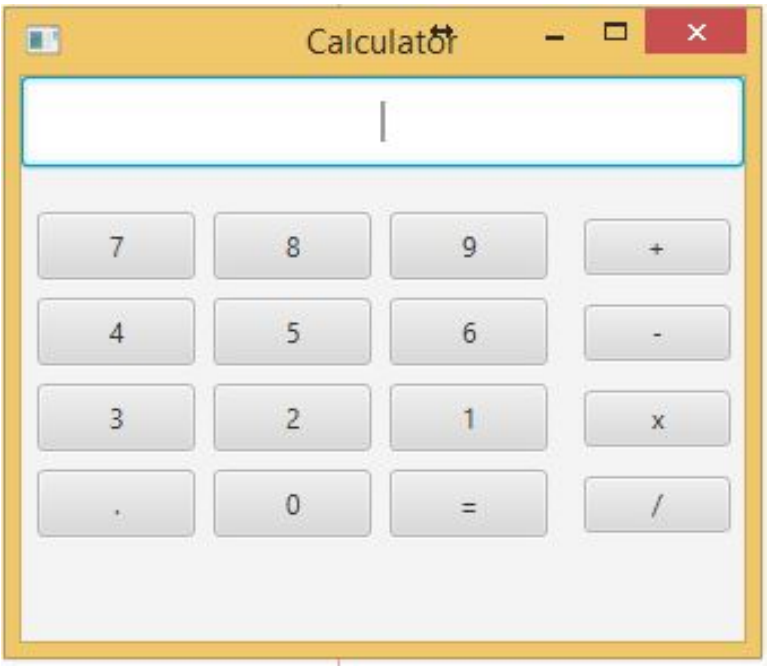


Slika 7.2.1 GridPane rsporeduje čvorove u ćelije matrice

PRIMER 9

Pravljenje digitrona koristeći odgovarajuće kontejnere iz Jave FX

Napraviti interfejs za digitron koristeći Javu FX po sledećoj slici



Slika 7.2.2 - Zadatak 1

PRIMER 9 - REŠENJE

Prikaz programskog koda za zadatak 1

Klasa CalcButton:

```
public class CalcButton extends Button {
    private String sign;

    public CalcButton() {
        super();
    }

    public CalcButton(String sign) {
        super();
        setText(sign);
        setMinSize(70, 30);
        setMaxSize(70, 30);
    }

    public String getSign() {
        return sign;
    }

    public void setSign(String sign) {
        this.sign = sign;
    }

}
```

Klasa CalculatorStage:

```
public class CalculatorStage extends Application {

    @Override
    public void start(Stage primaryStage) {
        // postavljamo VBox layout i postavljamo verikalno rastojanje izmedju
cvorova unutar njega
        VBox root = new VBox();
        root.setSpacing(20);
        String[] signArr = new String[]{"7", "8", "9",
                                        "4", "5", "6",
                                        "3", "2", "1",
                                        ".", "0", "="};
        String[] opArr = new String[]{
            "+", "-", "x", "/"
        };
        // kreiramo textField i postavljamo mu visinu i poravnanje
        TextField display = new TextField();
        display.setMinHeight(40);
        display.setMaxHeight(40);
        display.setAlignment(Pos.CENTER);
```

```
        // kreiramo gridPane container
        GridPane grid = new GridPane();
        grid.setAlignment(Pos.CENTER);
        grid.setHgap(8);
        grid.setVgap(8);
        int signInd = 0;

        // dodavanje dugmica kroz for petlju
        for (int i = 0; i < 4; i++) {
            for (int j = 0; j < 3; j++) {
                CalcButton bTmp = new CalcButton(signArr[signInd]);
                grid.add(bTmp, j, i);
                signInd++;
            }
        }
        for (int i = 0; i < opArr.length; i++) {
            Button bTmp = new Button(opArr[i]);
            bTmp.setMinSize(65, 25);
            bTmp.setMaxSize(65, 25);
            grid.add(bTmp, 4, i);
        }

        // stavljanje textField-a u root container
        root.getChildren().add(display);
        // stavljanje grid container unutar root-a
        root.getChildren().add(grid);
        // podesavamo scenu tako sto joj prosledjujemo koreni container i dimenzije
        Scene scene = new Scene(root, 320, 250);

        // podasavanje glavnog stage-a
        primaryStage.setTitle("Calculator");
        primaryStage.setScene(scene);
        primaryStage.show();
    }

    /**
    * @param args the command line arguments
    */
    public static void main(String[] args) {
        launch(args);
    }

}
```

PRIMER 9 - OBJAŠNJENJE

Cilj sekcije je da se pojasni kod zadatka 4

U ovom zadatku smo kreirali svoju klasu koja će CalcButton koja predstavlja jedno dugme na kalkulatoru. Jedini atribut koji ima ova klasa je znak.

Izdvajanje dugmeta u posebnu klasu može da bude korisno ukoliko želimo da stilizujemo dugme kalkulatora kako bi smo sav taj kod koji se odnosi na stilizovanje definisali samo jednom u okviru klase. Takođe, sve zajedničke funkcionalnosti dugmića na kalkulatoru možemo definisati u ovoj klasi.

Za prikaz dugmića kao na slici je najadekvatniji GridPane kontejner s obzirom da su dugmići razvrstani po vrstama i kolonama. Dakle, definisali smo u nizu singArr sve brojeve koje treba prikazati, a u nizu opArr sve operacije koje se nalaze na kalkulatoru. Zatim ćemo prvo kroz ugnježdenu for petlju dodati prve tri kolone koje se odnose na brojeve, a zatim proći kroz niz sa operacijama i dodati poslednju kolonu dugmića. Svako dugme je objekat klase CalcButton koju smo malopre pomenuli.

S obzirom da kalkulator sadrži i display, potrebno je definisati određenu root komponentu u koju ćemo smestiti GridPane sa dugmićima i display koji je predstavljen kao TextField objekat. Root komponenta će biti instanca VBox kontejnera i u nju ćemo dodati prvo display, zatim GridPane i nakon toga ćemo je postaviti u scenu.

PRIMER 10 - KORIŠĆENJE GRIDPANE OKNA

Program kreira dva okna a po dve kolone i četri reda primenom GridPane okna.

Na slici 1 su prikazana dva okna. Svaki sa po tri natpisa i tri tekstualna polja i sa jednim dugmetom. Listing prokazuje program koji ih kreira.

Slika 7.2.3 GridPane raspoređuje čvorove u kolone i redove

Program kreira GridPane (linija 16) i postavlja svojstva (linije 17-20). Ravnanje se postavlja u centar (linija 27), te i čvor se postavlja u centar okna. Ako promenite veličinu prozora, čvor ostaje u centru.

Program dodaje natpis u kolonu 0 i u red 0 (linija 23). Indeksi kolone i redova počinju od 0. Ne mora se svaka ćelija matrice da se popuni. Dugme se stavlja u kolonu 1 i red 3 (linija 30), a nema čvorova u kolonu 0 i redu 3. Uklanjanje čvora bi se radilo metodom pane.getChilidren().remove(node). Uklanjenje svih čvorova se radi metodom pane.getChilidren().removeAll().

Program poziva statički metod setHalignment kojim se dugme ravna u ćeliji (linija 30).

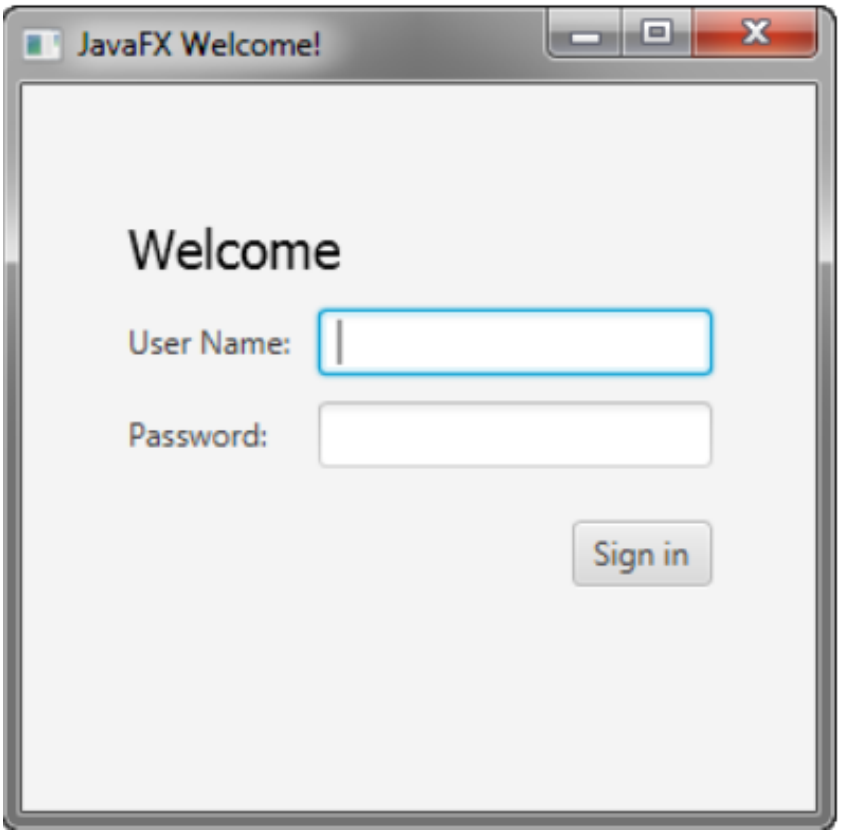
Veličina scene nije uneta (linija 34), već se automatski određuje.

```
1 import javafx.application.Application;
 2 import javafx.geometry.HPos;
 3 import javafx.geometry.Insets;
 4 import javafx.geometry.Pos;
 5 import javafx.scene.Scene;
 6 import javafx.scene.control.Button;
 7 import javafx.scene.control.Label;
 8 import javafx.scene.control.TextField;
 9 import javafx.scene.layout.GridPane;
 10 import javafx.stage.Stage;
 11
 12 public class ShowGridPane extends Application {
 13   @Override // Predefinisanje metoda start u klasi Application
 14   public void start(Stage primaryStage) {
 15  // Kreiranje okna i unos njegovih svojstava
 16  GridPane pane = new GridPane();
 17  pane.setAlignment(Pos.CENTER);
 18  pane.setPadding(new Insets(11.5, 12.5, 13.5, 14.5));
 19  pane.setHgap(5.5);
 20  pane.setVgap(5.5);
 21
 22  // Postavljanje čvorova u okno
 23  pane.add(new Label("First Name:"), 0, 0);
 24  pane.add(new TextField(), 1, 0);
 25  pane.add(new Label("MI:"), 0, 1);
 26  pane.add(new TextField(), 1, 1);
 27  pane.add(new Label("Last Name:"), 0, 2);
 28  pane.add(new TextField(), 1, 2);
 29  Button btAdd = new Button("Add Name");
 30  pane.add(btAdd, 1, 3);
 31  GridPane.setHalignment(btAdd, HPos.RIGHT);
 32
 33  // Kreiranje scene i njeno postavljanje na pozornicu
 34  Scene scene = new Scene(pane);
 35  primaryStage.setTitle("ShowGridPane"); // Unos naziva pozornice
 36  primaryStage.setScene(scene); // Postavljanje scene na pozornicu
 37  primaryStage.show(); // Prikaz pozornice
 38   }
 39 }
```

ZADATAK 7.3

Napraviti login formu koristeći GridPane kontejner.

3. Napraviti login formu koristeći GridPane kontejner. Login forma treba da ima polja UserName i password, kao i dugme za potvrdu.



Slika 7.2.4 Očekivani izlaz iz programa

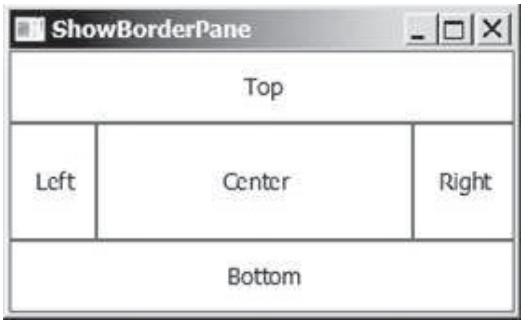
7.3 BorderPane

KLASA BORDERPANE

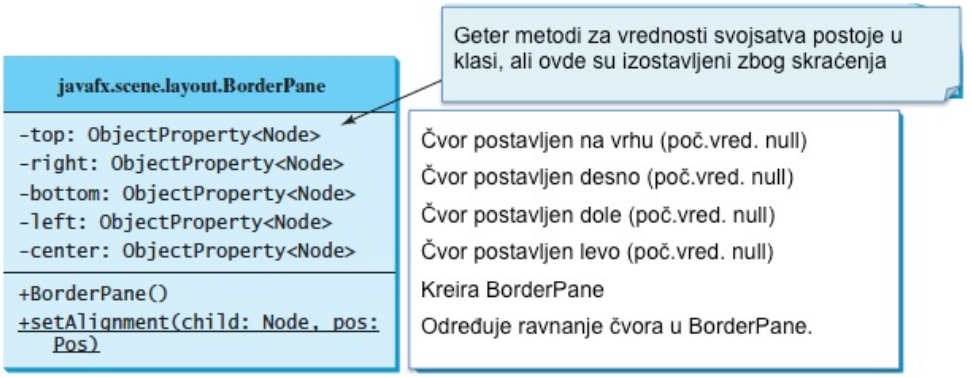
BorderPane postavlja čvorove u pet delova okna: gore, dole, levo, desno i u centar

BorderPane postavlja čvorove u pet delova okna: gore, dole, levo, desni u centar (slika 1), upotrebom metoda: ***setTop(node) , setBottom(node) , setLeft(node) , setRight(node)*** , i***setCenter(node***)

UML dijagram klase je dat na slici 2.



Slika 7.3.1 BorderPane raspoređuje čvorove gore, dole, levo, desno i u centar



Slika 7.3.2 Klasa BorderPane

PRIMER 11 - PRIMENA OKNA BORDERPANE

Listing programa ShowBorderPane pokazuje upotrebu okna BorderPane raspoređujući pet dugmeta u pet delova okna

Listing programa **ShowBorderPane** pokazuje upotrebu okna ***BorderPane*** raspoređujući pet dugemta u pet delova okna (slika 1)

Program definiše ***CustomPane*** koji proširuje***StackPane*** (linija 31) . Konstruktor za ***CustomPane*** dodaje natpis sa specificiranim nazivom (linija 33), definiše stilove za boju granice. I postavlja popunjenost (padding) upotrebom objekta Insets (linija 35).

Program kreira***BorderPane*** (line 13) i postavlja per primeraka objekta ***CustomPane*** u pet delova okna s granicom (linije 16-20). Vidi se da je okno takođe i čvor. Zato, okno se može dodati u drugo okno. Za uklanjanje čvora iz gornjeg dela, treba pozvati ***setTop(null***). Ako neki deo (region) nije zauzet, ne rezerviše se prostor za taj deo (region).

.

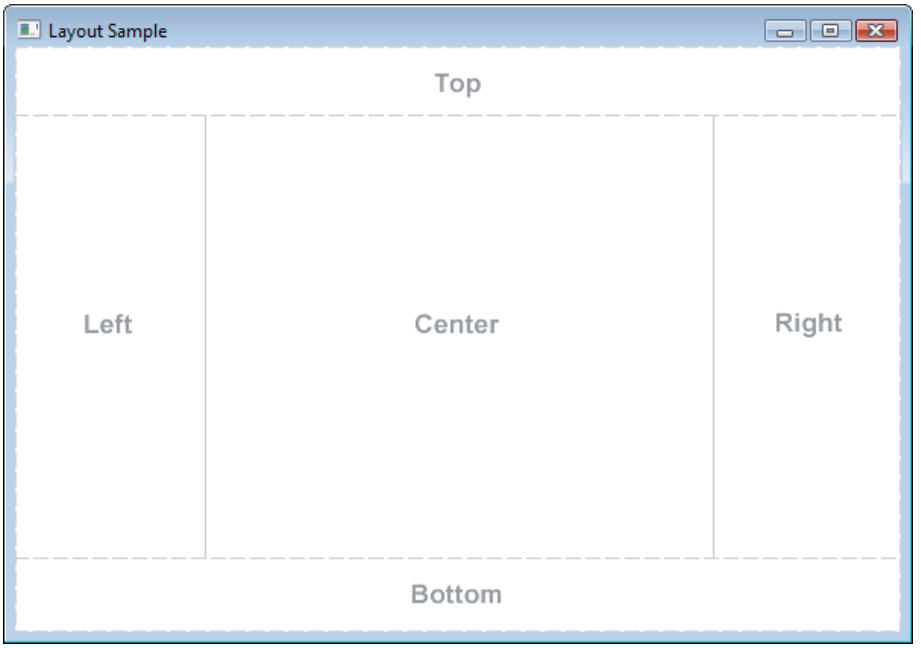
```
1 import javafx.application.Application;
 2 import javafx.geometry.Insets;
 3 import javafx.scene.Scene;
 4 import javafx.scene.control.Label;
 5 import javafx.scene.layout.BorderPane;
 6 import javafx.scene.layout.StackPane;
 7 import javafx.stage.Stage;
 8
 9 public class ShowBorderPane extends Application {
 10   @Override //Predefinisanje metoda start u klasi Application
 11   public void start(Stage primaryStage) {
 12  // Kreiranje BorderPane objekta
 13  BorderPane pane = new BorderPane();
 14
 15  // Postavlja čvor u okno
 16  pane.setTop(new CustomPane("Top"));
```

```
17  pane.setRight(new CustomPane("Right"));
18  pane.setBottom(new CustomPane("Bottom"));
19  pane.setLeft(new CustomPane("Left"));
20  pane.setCenter(new CustomPane("Center"));
21
22  // Kreiranje scene i njeno postavljanje na pozornicu
23  Scene scene = new Scene(pane);
24  primaryStage.setTitle("ShowBorderPane"); // Set the stage title
25  primaryStage.setScene(scene); // Place the scene in the stage
26  primaryStage.show(); // Display the stage
27   }
28 }
29
30 // Definisanje okna koji sadrži natpis u centru okvira.
31 class CustomPane extends StackPane {
32   public CustomPane(String title) {
33  getChildren().add(new Label(title));
34  setStyle("-fx-border-color: red");
35  setPadding(new Insets(11.5, 12.5, 13.5, 14.5));
36   }
37 }
```

ZADATAK 7.4

Napraviti GUI koristeći BorderPane kontejner.

4. Napraviti GUI koristeći BorderPane kontejner. Raspored bi trebalo da bude kao na sledećoj slici. Svako od polja sa slike zamenite kontrolom po izboru: dugme, labela, slika...



Slika 7.3.3 BorderPane šablon kao ilustracija raspoređivanja kontola

7.4 HBox i VBox

KLASE HBOX I VBOX

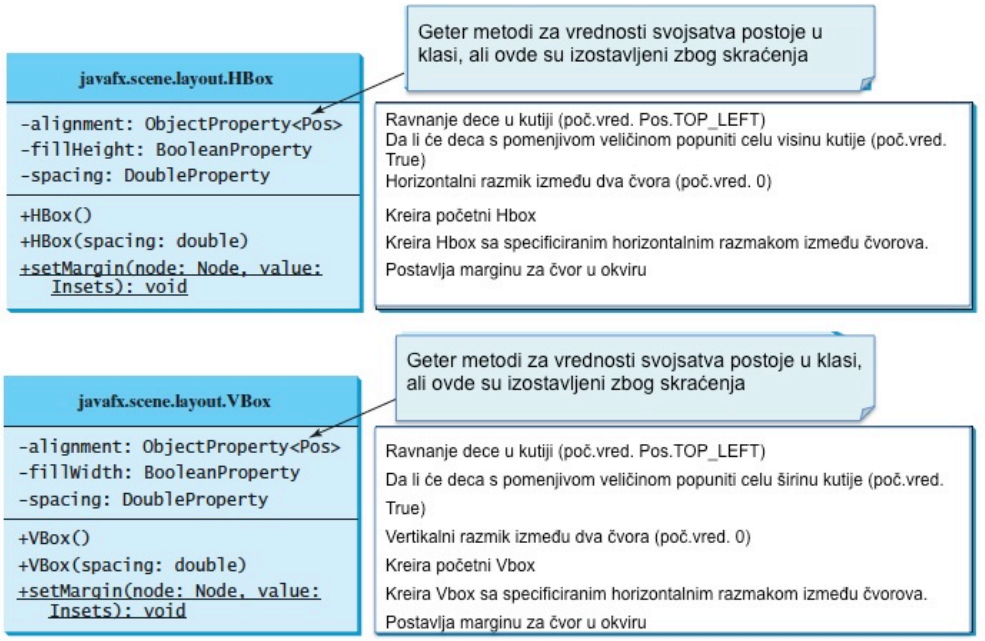
HBox i VBox sve čvorove stavljaju u jedan red, odnosno u jednu kolonu

HBox i VBox su okna koja sadrže, kao kontejneri, čvorove, koja se često nazivaju i decom, jer su vezani sa oknom vezom tipa “sadrži”.

HBox postavlja svoju decu u jedan horizontalni red.***VBox*** postavlja svoju decu i jednu vertikalni kolonu.

UML klasni dijagrami za ***HBox*** i ***VBox*** dati su na slici 1.

.



Slika 7.4.1 Klase HBox i VBox koje postavljaju čvorove u jedan red, odn. U jednu kolonu

PRIMER 12 - PRIMENA HBOX I VBOX OKANA

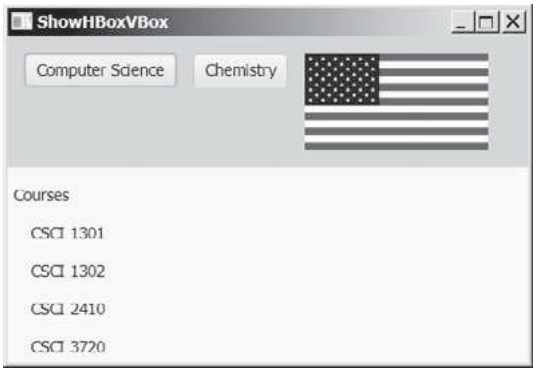
Primer dobijanja prozora sa dva okna. HBox okno treba da obezbedi dva dugmeta i prikaz zastave. Okno VBox treba da prikaže pet natpisa u jednoj koloni.

Potrebno je napraviti program čijim izvršenjem dobija se prozor sa Hbox oknom sa dva dugmeta i Vbox okno sa pet natpisa, kao što je pokazano na slici 2.

Program definiše metod getHBox() metod. On vraža Hbox koji sadrži dva dugmeta i prikaz sliku (linije 30-39). Boja pozadine okna Hbox je zlatna, a određena je upotrebom Java CSS (linija 33).

Program definiše metod getVBox() koji vraća Vbox okno koji sadrži pet natpisa (linije 41-55). Prvi natpis se dodaje u Vbox okno u liniji 44, a drugih četiri natpisa se dodaju u liniji 51.

Metod setMargin se upotrebljava za postavljenje margine čvora kada se postavlja unutar Vbox (linija 50).



Slika 7.4.2 HBox postavljaju čvorove u jedan red, a VBox - u jednu kolonu

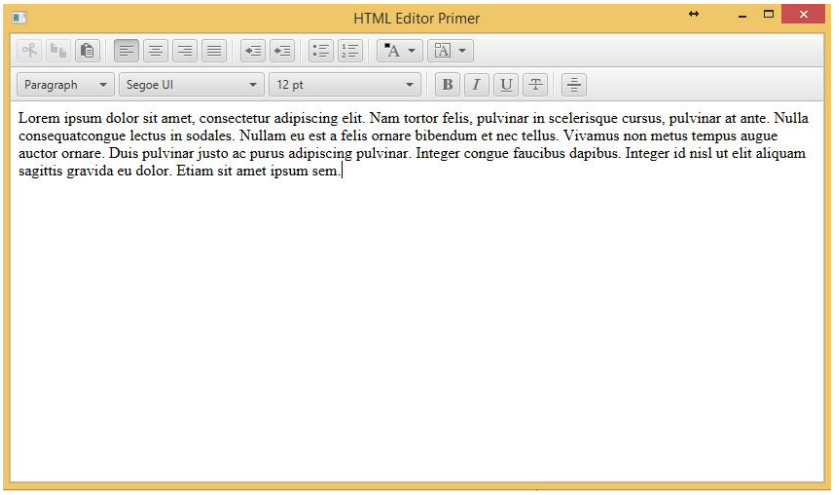
```
1 import javafx.application.Application;
 2 import javafx.geometry.Insets;
 3 import javafx.scene.Scene;
 4 import javafx.scene.control.Button;
 5 import javafx.scene.control.Label;
 6 import javafx.scene.layout.BorderPane;
 7 import javafx.scene.layout.HBox;
 8 import javafx.scene.layout.VBox;
 9 import javafx.stage.Stage;
 10 import javafx.scene.image.Image;
 11 import javafx.scene.image.ImageView;
 12
 13 public class ShowHBoxVBox extends Application {
 14   @Override // Redefiniše se start metod u klasi Application
 15   public void start(Stage primaryStage) {
 16  // Kreira okno sa granicom e
 17  BorderPane pane = new BorderPane();
 18
 19  // Postavljanje čvorova u okbu
 20  pane.setTop(getHBox());
 21  pane.setLeft(getVBox());
 22
 23  // Kreira scenu i stavlja je na pozornicu
 24  Scene scene = new Scene(pane);
 25  primaryStage.setTitle("ShowHBoxVBox"); // Unos naziva pozornice
 26  primaryStage.setScene(scene); // Postavlja scenu
 27  primaryStage.show(); // Prikazuje pozornicu
 28   }
 29
 30   private HBox getHBox() {
```

```
31   HBox hBox = new HBox(15);
32   hBox.setPadding(new Insets(15, 15, 15, 15));
33   hBox.setStyle("-fx-background-color: gold");
34   hBox.getChildren().add(new Button("Computer Science"));
35   hBox.getChildren().add(new Button("Chemistry"));
36   ImageView imageView = new ImageView(new Image("image/us.gif"));
37   hBox.getChildren().add(imageView);
38   return hBox;
39 }
40
41 private VBox getVBox() {
42 VBox vBox = new VBox(15);
43 vBox.setPadding(new Insets(15, 5, 5, 5));
44 vBox.getChildren().add(new Label("Courses"));
45
46 Label[] courses = {new Label("CSCI 1301"), new Label("CSCI 1302"),
47 new Label("CSCI 2410"), new Label("CSCI 3720")};
48
49 for (Label course: courses) {
50 VBox.setMargin(course, new Insets(0, 0, 0, 15));
51 vBox.getChildren().add(course);
52 }
53
54 return vBox;
55 }
56
```

PRIMER 13

Cilj ovog zadatka je prikaz korišćenja HTML Editor komponente u Java FX-u

Napraviti formu ko na sledećoj slici koristeći HTMLEditor komponentu u JavaFX-u.



Slika 7.4.3 - Zadatak 2

JavaFX nam obezbeđuje gotovu GUI komponentu koja se naziva HTMLEditor i koja nam omogućava osnovne funkcije manipulacije sa tekstom (formatiranje teksta, copy, cut, paste, itd).

Potrebno je samo da kreiramo instancu klase HTMLEditor, nakon čega joj možemo definisati dimenzije i inicijalni tekst. Kreiranu instancu moramo dodati na scenu. U prikazanom primeru root komponenta aplikacije je VBox kontejner u koji ćemo smestiti naš editor, a zatim ćemo VBox objekat proslediti sceni.

Klasa Main.java

```
public class Main extends Application {

    private final String TEKST_ZA_PRIKAZ = "Lorem ipsum dolor sit "
            + "amet, consectetur adipiscing elit. Nam tortor felis, pulvinar "
            + "in scelerisque cursus, pulvinar at ante. Nulla consequat"
            + "congue lectus in sodales. Nullam eu est a felis ornare "
            + "bibendum et nec tellus. Vivamus non metus tempus augue auctor "
            + "ornare. Duis pulvinar justo ac purus adipiscing pulvinar. "
            + "Integer congue faucibus dapibus. Integer id nisl ut elit "
            + "aliquam sagittis gravida eu dolor. Etiam sit amet ipsum "
            + "sem.";

    @Override
    public void start(Stage stage) {
        stage.setTitle("HTML Editor Primer");
        stage.setWidth(500);
        stage.setHeight(500);
        Scene scene = new Scene(new Group());
        VBox root = new VBox();
        final HTMLEditor htmlEditor = new HTMLEditor();
        htmlEditor.setPrefHeight(600);
        htmlEditor.setHtmlText(TEKST_ZA_PRIKAZ);
        root.getChildren().addAll(htmlEditor);
        scene.setRoot(root);
        stage.setScene(scene);
        stage.show();
    }

    /**
    * @param args the command line arguments
    */
    public static void main(String[] args) {
        launch(args);
    }

}
```

PRIMER 13 - REŠENJE

Prikaz programskog koda i objašnjenje rešenja zadatka 1

Klasa Main.java

```
public class Main extends Application {

    private final String TEKST_ZA_PRIKAZ = "Lorem ipsum dolor sit "
            + "amet, consectetur adipiscing elit. Nam tortor felis, pulvinar "
            + "in scelerisque cursus, pulvinar at ante. Nulla consequat"
            + "congue lectus in sodales. Nullam eu est a felis ornare "
            + "bibendum et nec tellus. Vivamus non metus tempus augue auctor "
            + "ornare. Duis pulvinar justo ac purus adipiscing pulvinar. "
            + "Integer congue faucibus dapibus. Integer id nisl ut elit "
            + "aliquam sagittis gravida eu dolor. Etiam sit amet ipsum "
            + "sem.";

    @Override
    public void start(Stage stage) {
        stage.setTitle("HTML Editor Primer");
        stage.setWidth(500);
        stage.setHeight(500);
        Scene scene = new Scene(new Group());
        VBox root = new VBox();
        final HTMLEditor htmlEditor = new HTMLEditor();
        htmlEditor.setPrefHeight(600);
        htmlEditor.setHtmlText(TEKST_ZA_PRIKAZ);
        root.getChildren().addAll(htmlEditor);
        scene.setRoot(root);
        stage.setScene(scene);
        stage.show();
    }

    /**
    * @param args the command line arguments
    */
    public static void main(String[] args) {
        launch(args);
    }

}
```

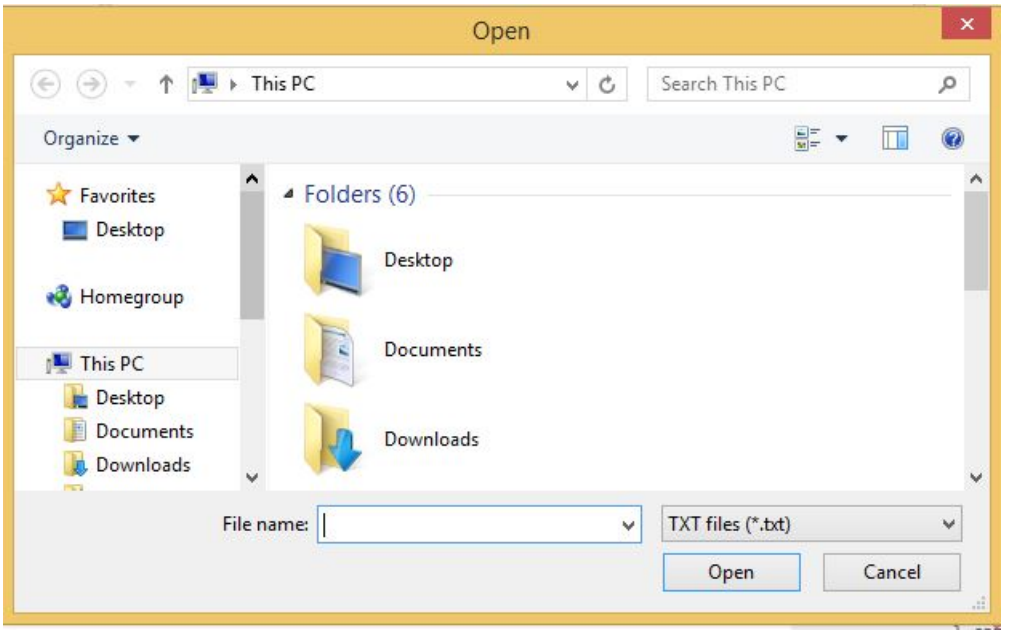
JavaFX nam obezbeđuje gotovu GUI komponentu koja se naziva HTMLEditor i koja nam omogućava osnovne funkcije manipulacije sa tekstom (formatiranje teksta, copy, cut, paste, itd).

Potrebno je samo da kreiramo instancu klase HTMLEditor, nakon čega joj možemo definisati dimenzije i inicijalni tekst. Kreiranu instancu moramo dodati na scenu. U prikazanom primeru root komponenta aplikacije je VBox kontejner u koji ćemo smestiti naš editor, a zatim ćemo VBox objekat proslediti sceni.

PRIMER 14

Korišćenje FileChooser komponente u JavaFX-u.

Napraviti aplikaciju koja otvara txt fajl po izboru korisnika a potom menja svako slovo a u slovo b unutar fajla. Kako bi ovo uradili potrebno je da koristite FileChooser komponentu JaveFx.



Slika 7.4.4 - Zadatak 5

PRIMER 14 - REŠENJE

Prikaz programskog koda zadatka 5

Klasa Main:

```
public class Main extends Application {

    private BufferedWriter bw;
    private BufferedReader br;
    private File file;

    @Override
    public void start(Stage primaryStage) {
        Group root = new Group();
        Button buttonLoad = new Button("Ucitaj fajl");
        buttonLoad.setOnAction(new EventHandler<ActionEvent>() {
            @Override
            public void handle(ActionEvent arg0) {
                FileChooser fileChooser = new FileChooser();
                FileChooser.ExtensionFilter extFilter = new
FileChooser.ExtensionFilter("TXT files (*.txt)", "*.txt");
```

```
                fileChooser.getExtensionFilters().add(extFilter);
                file = fileChooser.showOpenDialog(primaryStage);
                try {
                    br = new BufferedReader(new FileReader(file));
                    //
                    String forReplace = readFromFile();
                    bw = new BufferedWriter(new FileWriter(file));
                    forReplace = forReplace.replace("a", "b");
                    writeInFile(forReplace);
                } catch (Exception e) {
                }
            }
        });
        VBox vBox = VBoxBuilder.create()
                .children(buttonLoad)
                .build();
        root.getChildren().add(vBox);
        primaryStage.setScene(new Scene(root, 500, 400));
        primaryStage.show();
    }

    public void writeInFile(String text) throws IOException {
        bw.write(text);
        bw.flush();
        bw.close();
    }

    public String readFromFile() throws IOException {
        String line = null;
        String result = "";
        while ((line = br.readLine()) != null) {
            result += line + "\r\n";
        }
        br = new BufferedReader(new FileReader(file));
        return result;
    }

    /**
    * @param args the command line arguments
    */
    public static void main(String[] args) {
        launch(args);
    }

}
```

PRIMER 14 - OBJAŠNJENJE

Cilj sekcije je da se pojasni kod zadatka 5

JavaFX nam obezbeđuje GUI komponentu u vidu dijaloga za odabir datoteka sa računara koja se naziva FileChooser.

FileChooser sadrži brojne statičke metode koje nam pružaju mogućnost raznih podešavanja. U ovom zadatku smo iskoristili mogućnost filtriranja datoteka na osnovu ekstenzija. Konkretno, u ovom primeru smo ograničili da nam FileChooser nudi samo tekstualno datoteke tj. sa ekstenzijom txt.

Metoda showOpenDialog prikazuje FileChooser prozor koji primenjuje definisane filtere nad datotekama.

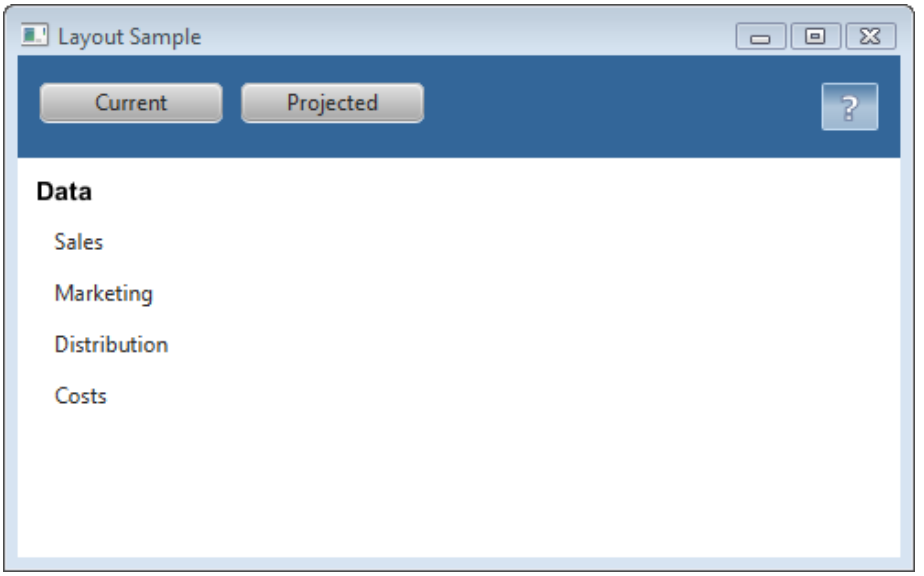
Nakon odabira datoteke, preko klase BufferedReader učitavamo čitav sadržaj odabrane datoteke, zatim vršimo traženu zamenu karaktera i preko klase BufferedWriter upisujemo nov sadržaj u datoteku.

U ovom primeru VBox root komponenta se kreira na dugačiji način, tj. preko šablona builder. To funkcioniše tako što se metodom create kreira instanca nakon čega možemo lančano pozivati druge metode koje podešavaju parametre komponente. Svaka metoda kada primeni prosleđene parametre vraća ponovo referencu na objekat kako bi se omogućilo nadovezivanje sledeće metode. Kada preko ulančanih metoda definišemo sve parametre, onda je potrebno da pozovemo metodu build koja finalizira kreairani objekat.

ZADATAK 7.5

Napraviti GUI koristeći HBox i VBox kontejnere.

5. Napraviti GUI koristeći HBox i VBox kontejnere. Pokušajte da realizujete GUI sa sledeće slike.



Slika 7.4.5 Napraviti GUI koristeći HBox i VBox kontejnere

VIDEO - RAD SA RASPOREDIMA

JavaFX Java GUI Tutorial - 8 - Embedding Layouts (6,25 minuta)

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

VIDEO - GRIDPANE

JavaFX Java GUI Tutorial - 9 - GridPane (11,15 minuta)

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

Poglavlje 8

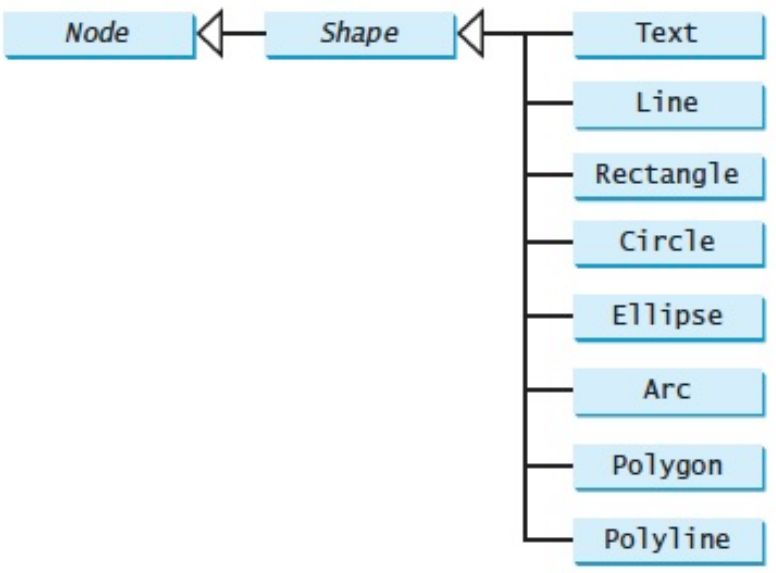
Klasa Shapes

UVOD

JavaFX obezbeđuje puno klasa oblika (shapes) koje crtaju tekstove, linije, krugove, pravougaonike, elipse, lukove, poligone i polilinije

Klasa Shape je apstrkna klasa koja definiše zajednička svojstva za sve oblike. Ta svojstva određuju boju popune površine oblika, boju i izgled granične linije. Svojstvo popune površine specificira boju koja popunava unutrašnjost oblika. Svojstvo izgleda granične linije (stroke) specificira boju linije obika. Svojstvo strokeWith specificira širinu linije oblika.

Ovde će se prikazati klase ***Text, Line, Rectangle, Circle, Elipse, Arc, Polygon i Polyline*** za crtanje tekstova i jednostavne oblike, UML dijagram ovih klasa je prikazan na slici 1. .



Slika 8.1.1 UML dijagram klasa koje definišu osnovne oblike čvorova

8.1 Klasa Text

SVOJSTVA KLASE TEXT

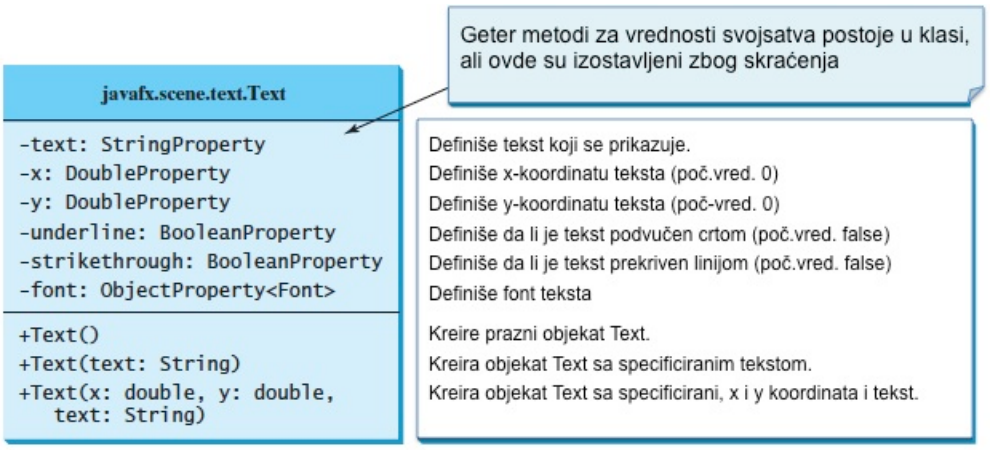
Klasa Text prikazuje željeni tekst u određenoj poziciji okna.

Klasa Text definiše čvor koji prikazuje string (niz znakova) od tačke (x,y) kao što je prikazano na slici 2.a. A čija svojstva (atributi i metodi(su prikazani na slici 1.

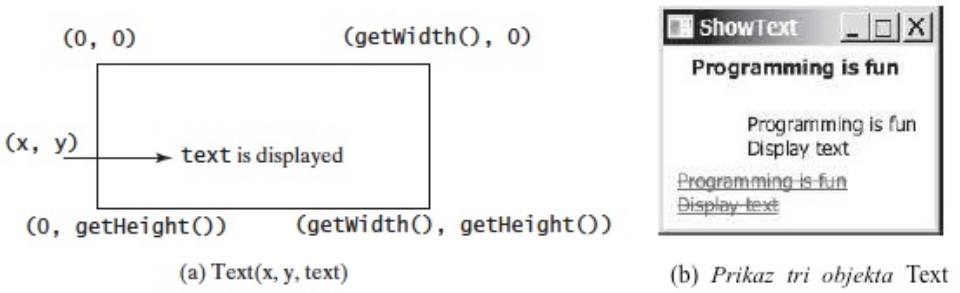
Tekst se smešta u okno koje ima koordinatni početak (0,0) u gornjem levom uglu a metodi *pane.getWith()* i *pane.getHeight()* daju informaciju o širini i visini okna, odn. O koordinati i donjeg desnog ugla okna.

Tekst može da bude prikazan u više linija koje odvaja kontrolni znak \n.

.



Slika 8.2.1 UML dijagram klase Text



Slika 8.2.2 Objekat Text prikazuje željeni tekst

PRIMER 15 - KORIŠĆENJE KLASE TEXT

Program treba da prikaže u oknu četiri teksta sa različitim svojstvima (naglašen, u više redova, podvučen, prekriven crtom).

Ovde je prikazan listing programa ShowText čijim izvršenjem se dobija prikaz tekstana slici 2.b.

Program kreira Text objekat (linija 18), određuje mu font (linija 19) i postavlja ga u okno (linija 21).

Program kreira drugi Text objekat sa više linija (linija 23) i postavlja ga u okno (linija 24).

Program kreira treći Text objekat (linija 26), određuje njegovu boju (linija 27), određuje tekst podvučen donjom crtom i tekst sa linijiom preko teksta (linje 28-29) i postavlja ga u okno (linija 30).

```
1 import javafx.application.Application;
 2 import javafx.scene.Scene;
 3 import javafx.scene.layout.Pane;
 4 import javafx.scene.paint.Color;
 5 import javafx.geometry.Insets;
 6 import javafx.stage.Stage;
 7 import javafx.scene.text.Text;
 8 import javafx.scene.text.Font;
 9 import javafx.scene.text.FontWeight;
 10 import javafx.scene.text.FontPosture;
 11
 12 public class ShowText extends Application {
 13   @Override // Predefinisanje metoda start u klasi Application
 14   public void start(Stage primaryStage) {
 15  // Kreiranje okna koji treba da sadrži tekst
 16  Pane pane = new Pane();
 17  pane.setPadding(new Insets(5, 5, 5, 5));
 18  Text text1 = new Text(20, 20, "Programming is fun");
 19  text1.setFont(Font.font("Courier", FontWeight.BOLD,
 20  FontPosture.ITALIC, 15));
 21  pane.getChildren().add(text1);
 22
 23  Text text2 = new Text(60, 60, "Programming is fun\nDisplay text");
 24  pane.getChildren().add(text2);
 25
 26  Text text3 = new Text(10, 100, "Programming is fun\nDisplay text");
 27  text3.setFill(Color.RED);
 28  text3.setUnderline(true);
 29  text3.setStrikethrough(true);
 30  pane.getChildren().add(text3);
 31
 32  // Kreiranje scene i njeno postavljene na pozornicu
 33  Scene scene = new Scene(pane);
 34  primaryStage.setTitle("ShowText"); // Unos naziva pozornice
 35  primaryStage.setScene(scene); // Postavljenje scene na pozornicu
 36  primaryStage.show(); // Prikaz pozornice
 37   }
 38 }
```

ZADATAK 8.1

Samostalno vežbanje korišćenja klase Text

1. Pokušajte da samostalno kreirate program koji će dati izlaz kao na sledećoj slici. Koristite klasu Text, ali i Font i Color.

Slika 8.2.3 Samostalno vežbanje korišćenja klase Text

VIDEO - UNOŠENJE TEKSTA

JavaFX Java GUI Tutorial - 10 - Extract and Validate Input (8,37)

Ova lekcija sadrži video materijal. Ukoliko želite da pogledate ovaj video morate da otvorite LAMS lekciju.

8.2 Klasa Line

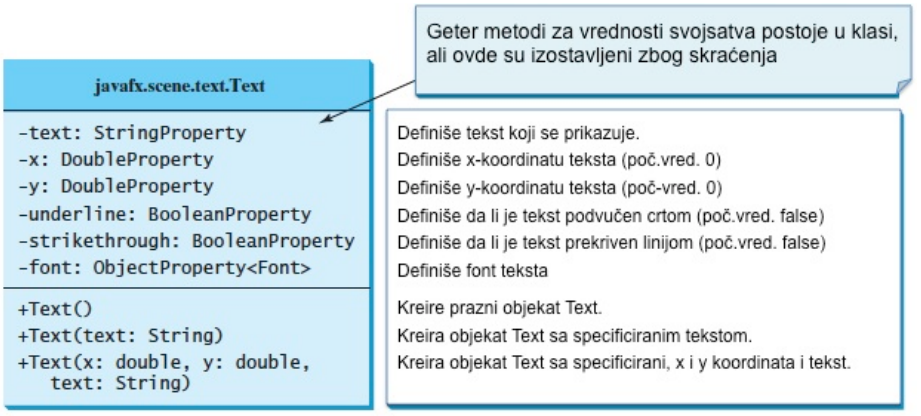
SVOJSTVA KLASE LINE

Klasa Line definiše liniju određenu početnom i krajnjom tačkom.

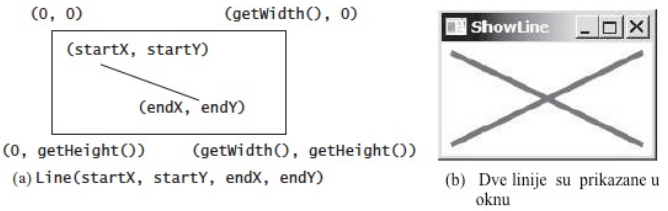
Slika 1 prikazuje UML dijagram klase ***Line*** koja definiše liniju.

Linija povezuje dve tačke koje određene parametrima ***startX , startY , endX i endY*** , kao što je prikazano na slici 2.a.

.



Slika 8.3.1 UML dijagram klase Line koja prikazuje liniju između dve tačke



Slika 8.3.2 Kreira se objekat Line radi prikaza linije

PRIMER 16 - KORIŠĆENJE KLASE LINE

Program treba da prikaže dve ukrštene linije u oknu

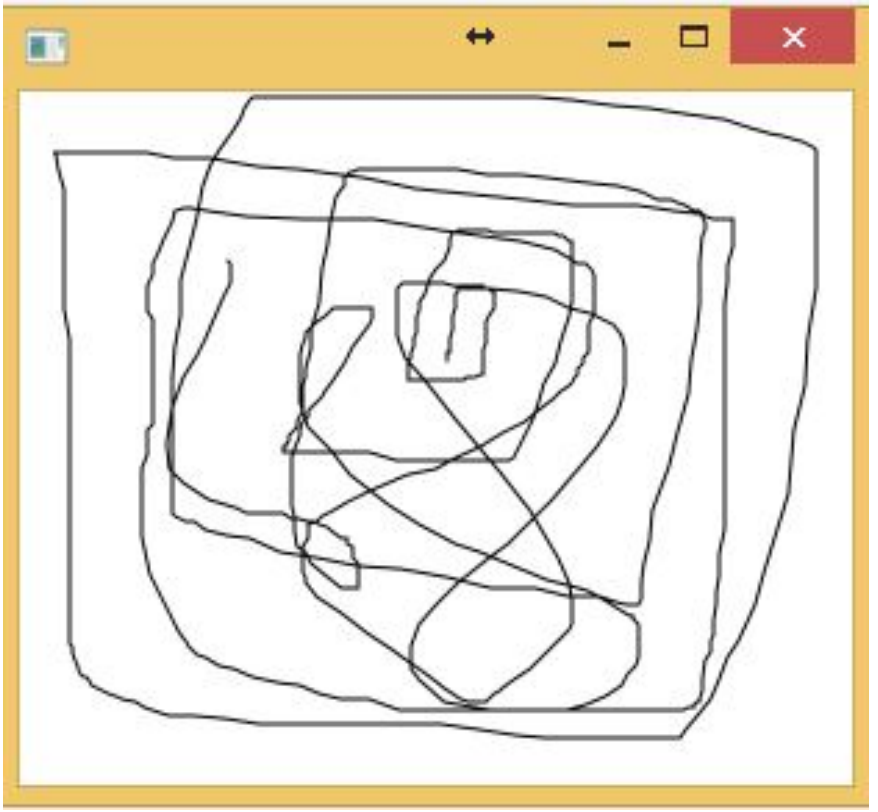
Listing prikazuje program koji pokazuje kako se dobija prikaz dve ukrštene linije na slici 2.b.

```
1 import javafx.application.Application;
 2 import javafx.scene.Scene;
 3 import javafx.scene.layout.Pane;
 4 import javafx.scene.paint.Color;
 5 import javafx.stage.Stage;
 6 import javafx.scene.shape.Line;
 7
 8 public class ShowLine extends Application {
 9   @Override // Predefinisanje metoda start u klasi Application
 10   public void start(Stage primaryStage) {
 11  // Kreira scenu i postavlja je na pozornicu
 12  Scene scene = new Scene(new LinePane(), 200, 200);
 13  primaryStage.setTitle("ShowLine"); // Unosi naziv pozornice
 14  primaryStage.setScene(scene); // Postavlja scenu na pozornicu
 15  primaryStage.show(); // Prikazuje pozornicu
 16   }
 17 }
 18
 19 class LinePane extends Pane {
 20   public LinePane() {
 21  Line line1 = new Line(10, 10, 10, 10);
 22  line1.endXProperty().bind(widthProperty().subtract(10));
 23  line1.endYProperty().bind(heightProperty().subtract(10));
 24  line1.setStrokeWidth(5);
 25  line1.setStroke(Color.GREEN);
 26  getChildren().add(line1);
 27
 28  Line line2 = new Line(10, 10, 10, 10);
 29  line2.startXProperty().bind(widthProperty().subtract(10));
 30  line2.endYProperty().bind(heightProperty().subtract(10));
 31  line2.setStrokeWidth(5);
 32  line2.setStroke(Color.GREEN);
 33  getChildren().add(line2);
 34   }
 35 }
```

PRIMER 17

Cilj ovog zadatka je da prikaže rad sa događajima miša i crtanje pomoću Path-a u Javi FX

Napraviti aplikaciju koja omogućava korisniku da crta po ekranu tako što drži levi klik i prevlači miša po ekranu. Kada korisnik hoće da poništi ono što je nacrtao treba da klikne jednom desni klik.



Slika 8.3.3 - Zadatak 6

PRIMER 17 - REŠENJE

Prikaz programskog koda zadatka 6

Klasa MainCrtanje:

```
public class MainCrtanje extends Application {

    Path path;

    @Override
    public void start(Stage primaryStage) {
```

```
        Group root = new Group();
        Scene scene = new Scene(root, 300, 250);

        path = new Path();
        path.setStrokeWidth(1);
        path.setStroke(Color.BLACK);

        scene.setOnMouseClicked(mouseHandler);
        scene.setOnMouseDragged(mouseHandler);
        scene.setOnMouseEntered(mouseHandler);
        scene.setOnMouseExited(mouseHandler);
        scene.setOnMouseMoved(mouseHandler);
        scene.setOnMousePressed(mouseHandler);
        scene.setOnMouseReleased(mouseHandler);

        root.getChildren().add(path);
        primaryStage.setScene(scene);
        primaryStage.show();
    }

    EventHandler<MouseEvent> mouseHandler = new EventHandler<MouseEvent>() {

        @Override
        public void handle(MouseEvent mouseEvent) {
            if (mouseEvent.getEventType() == MouseEvent.MOUSE_PRESSED) {
                if (mouseEvent.getButton() == MouseButton.SECONDARY) {
                    path.getElements().clear();
                }
                path.getElements()
                        .add(new MoveTo(mouseEvent.getX(), mouseEvent.getY()));
            } else if (mouseEvent.getEventType() == MouseEvent.MOUSE_DRAGGED) {
                path.getElements()
                        .add(new LineTo(mouseEvent.getX(), mouseEvent.getY()));
            }

        }

    };

    /**
    * @param args the command line arguments
    */
    public static void main(String[] args) {
        launch(args);
    }

}
```

PRIMER 17 - OBJAŠNJENJE

Cilj sekcije je da se pojasni kod zadatka 6

JavaFX nam obezbeđuje GUI komponentu u vidu dijaloga za odabir datoteka sa računara koja se naziva FileChooser.

FileChooser sadrži brojne statičke metode koje nam pružaju mogućnost raznih podešavanja. U ovom zadatku smo iskoristili mogućnost filtriranja datoteka na osnovu ekstenzija. Konkretno, u ovom primeru smo ograničili da nam FileChooser nudi samo tekstualno datoteke tj. sa ekstenzijom txt.

Metoda showOpenDialog prikazuje FileChooser prozor koji primenjuje definisane filtere nad datotekama.

Nakon odabira datoteke, preko klase BufferedReader učitavamo čitav sadržaj odabrane datoteke, zatim vršimo traženu zamenu karaktera i preko klase BufferedWriter upisujemo nov sadržaj u datoteku.

U ovom primeru VBox root komponenta se kreira na dugačiji način, tj. preko šablona builder. To funkcioniše tako što se metodom create kreira instanca nakon čega možemo lančano pozivati druge metode koje podešavaju parametre komponente. Svaka metoda kada primeni prosleđene parametre vraća ponovo referencu na objekat kako bi se omogućilo nadovezivanje sledeće metode. Kada preko ulančanih metoda definišemo sve parametre, onda je potrebno da pozovemo metodu build koja finalizira kreairani objekat.

ZADATAK 8.2

Samostalno vežbanje korišćenja klase Line

2. Detaljno izvršite analizu priloženog koda. Kakav će biti izlaz na ekranu kada se ovaj program pokrene.

```
import javafx.application.Application;
import javafx.scene.Group;
import javafx.scene.Scene;
import javafx.scene.paint.Color;
import javafx.scene.shape.Line;
import javafx.scene.shape.StrokeLineCap;
import javafx.stage.Stage;

/**
 *
 * @author erix7
 */
public class Main extends Application {

    /**
     * @param args the command line arguments
     */
    public static void main(String[] args) {
        launch(args);
    }

    @Override
```

```
    public void start(Stage primaryStage) {
        primaryStage.setTitle("Crtanje Linija");

        Group root = new Group();
        Scene scene = new Scene(root, 300, 150, Color.GRAY);

        Line redLine = new Line(10, 10, 200, 10);

        redLine.setStroke(Color.RED);
        redLine.setStrokeWidth(10);
        redLine.setStrokeLineCap(StrokeLineCap.BUTT);

        redLine.getStrokeDashArray().addAll(15d, 5d, 15d, 15d, 20d);
        redLine.setStrokeDashOffset(10);

        root.getChildren().add(redLine);

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

8.3 Klasa Rectangle

SVOJSTVA KLASE RECTANGLE

Klasa Rectangle definiše pravougaonik određen gornjim levim temenom, širinom i visinom pravougaonika.

Slika 1 prikazuje UML dijagram klase Rectangle koja definiše pravougaonik.

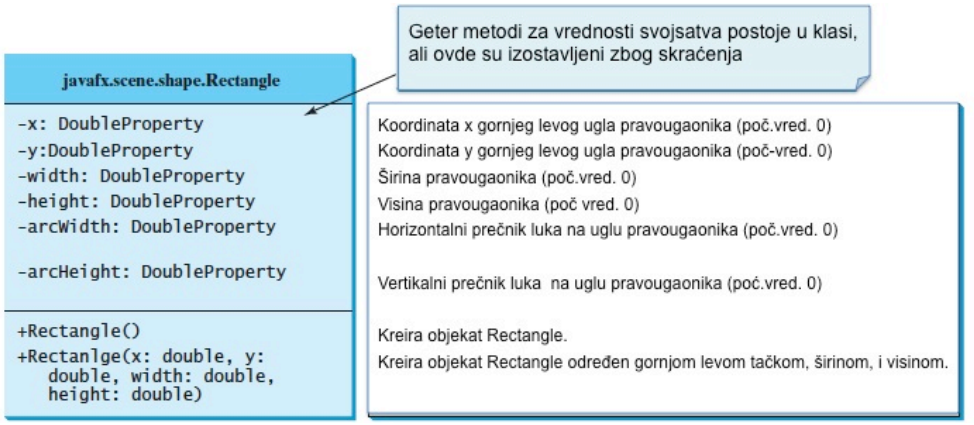
Pravougaonik je definisan parametrima x i y (koordinatama gorenjeg levog temena), širinom, visinom, širinom i visinom luka u nejgovim temenima (slika 2.a) , tj. parametrima: ***x , y , width , height , arcWidth , i arcHeight.***

Koordinate ***x*** i ***y*** određuju položaj gornjeg levog temena pravougaonika***, arcgWith*** i ***ArcHeight*** određuju horizontalni, odnosno vertikalni prečnik ugaonih lukova pravougaonika.

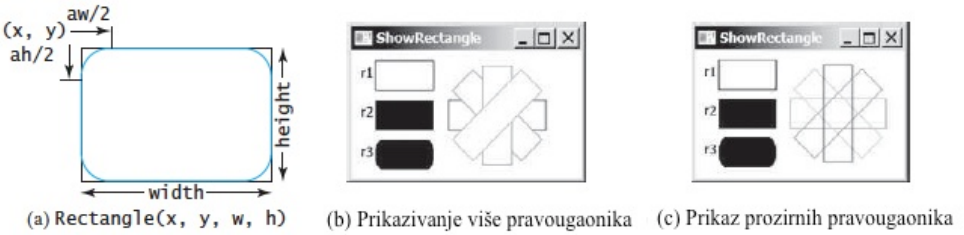
.

.

.



Slika 8.4.1 UML dijagram klase Rectangle



(a) Rectangle(x, y, w, h) (b) Prikazivanje više pravougaonika (c) Prikaz prozirnih pravougaonika

Slika 8.4.2 Objekat Rectangle prikazuje pravougaonik

PRIMER 18 - KORIŠĆENJE KLASE RECTANGLE

Program prikazuje crtanje više pravougaonika, zarotiranih oko centra scene za određeni ugao.

Listing programa ShowRectangle demonstrira crtanje pravougaonika prikazanih na slici 2.b.

Program kreira vipe pravougaonika. Njihova površina ima unapred definisanu boju, a to je crna. Isto tako, početno podešena boja okvirne linije pravougaonika je crna.

Boja okvirne linije pravougaonika r1 je crna (linija 17).

Program kreira pravougaonik r3 (linija 26) i određuje širine i visine njegovih lukova u uglovima (linije 27-28). Zato, pravougaonik r3 ima zaobljenna temena.

Program više puta ponavlja kreiranje pravougaonika (linija 33), rotira ga (linija 34), slučajno postavlja boju njegove okvirne linije (linije 35-36), boju njegove unutrašnjosti koja je bela (linija 37) i dodaje pravougaonik u okno (linija 38), ređajuži ih jedan preko drugoga.

Ako se linija 37 zameni se linjom:

r.setFill(null);

Pravougaonik neće biti popunjen bojom. Zako se dobija prikaz prozirnih pravougaonika na slici 2.c.

.

```
1 import javafx.application.Application;
 2 import javafx.scene.Scene;
 3 import javafx.scene.layout.Pane;
 4 import javafx.scene.paint.Color;
 5 import javafx.stage.Stage;
 6 import javafx.scene.text.Text;
 7 import javafx.scene.shape.Rectangle;
 8
 9 public class ShowRectangle extends Application {
 10   @Override // Predefinisanje metoda start u klasi Application
 11   public void start(Stage primaryStage) {
 12  // Kreiranje okna
 13  Pane pane = new Pane();
 14
 15  // Kreiranje pravougaonika i njihovo dodavanje u okno
 16  Rectangle r1 = new Rectangle(25, 10, 60, 30);
 17  r1.setStroke(Color.BLACK);
 18  r1.setFill(Color.WHITE);
 19  pane.getChildren().add(new Text(10, 27, "r1"));
 20  pane.getChildren().add(r1);
 21
 22  Rectangle r2 = new Rectangle(25, 50, 60, 30);
 23  pane.getChildren().add(new Text(10, 67, "r2"));
 24  pane.getChildren().add(r2);
 25
 26  Rectangle r3 = new Rectangle(25, 90, 60, 30);
 27  r3.setArcWidth(15);
 28  r3.setArcHeight(25);
 29  pane.getChildren().add(new Text(10, 107, "r3"));
 30  pane.getChildren().add(r3);
 31
 32  for (int i = 0; i < 4; i++) {
 33    Rectangle r = new Rectangle(100, 50, 100, 30);
 34    r.setRotate(i * 360 / 8);
 35    r.setStroke(Color.color(Math.random(), Math.random(),
 36    Math.random()));
 37    r.setFill(Color.WHITE);
 38    pane.getChildren().add(r);
 39  }
 40
 41  // Kreiranje scene i njeno postavljanje u pozornicu
 42  Scene scene = new Scene(pane, 250, 150);
 43  primaryStage.setTitle("ShowRectangle"); // Unos naziva pozornice
 44  primaryStage.setScene(scene); // Stavljanje scene na pozornicu
 45  primaryStage.show(); // Prikaz pozornice
 46   }
 47 }
```

ZADATAK 8.3

Samostalno vežbanje korišćenja klase Rectangle

3. Pokušajte da napravite JavaFX program koji će, koristeći petlju nacrtati nekoliko pravougaonika. Svaki naredni pravougaonik bi trebalo da bude upisan (nalazi se u) prethodno nacrtanom pravougaoniku.

8.4 Klase Circle i Ellipse

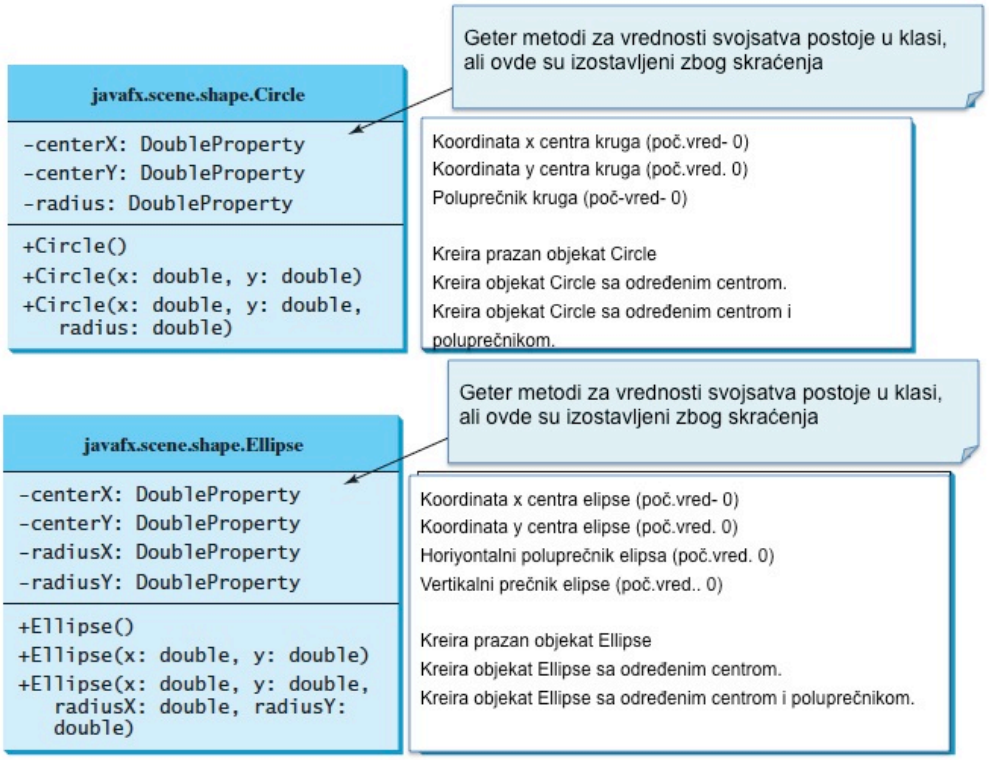
SVOJSTAVA KLASA CIRCLE I ELLIPSE

Klase Circle i Ellipse definišu krug, odnosno elipsu

Na slici 1 prikazani su UML dijagrami klasa ***Circle*** i ***Ellipse***, koje definišu krug, odn. elipsu.

Krug je definisam svojm cenrom sa parametrima ***centerX , centerY*** , polupečnikom ***(radius***) .

Elipsa je definisana svojm centro, tj. parametrima .***centerX , centerY*** , i horizontalnim i vertikalnim polupračnikom, tj. parametrima ***radiusX*** , and ***radiusY*** , kao što je prikazano na slici 2.a.



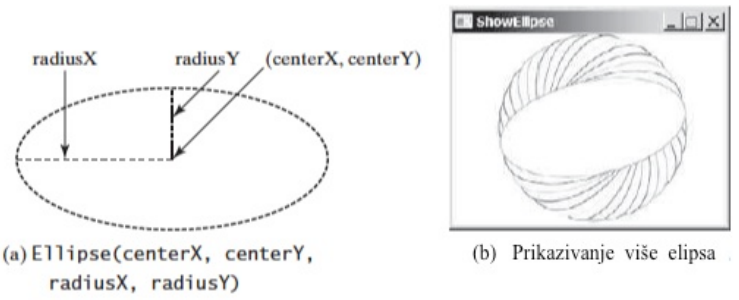
Slika 8.5.1 UML dijagrami klasa Circle i Ellipse

PRIMER 19 - KORIŠĆENJA KLASE ELLIPSE

Program treba da nacrta niz zarotiranih elipsi oko centra okna i scene, a svaka elipsa ima slučajno određena svojstva (boja granične linije i popune).

Potrebno je da se nacrta niy yarotiranih elipsa u oknu, kao što je to prikazano na slici 2.b. Listing programa ShowEllipse prikazuje progra, čijim izvrešenjem se dobija skup zarotiranih elipsi oko centra okna i scene (slika 2.b).

Program više puta ponavlja crtanje elipse (linija 16), slučajno određuje boju granične linije (linije 17-18), postavlja boju njihove popune (linija 19) i dodaje nacrtan pravougaonik u okno (linija 21).



Slika 8.5.2 (a) Elipsa (b) Prikaz više elipsa

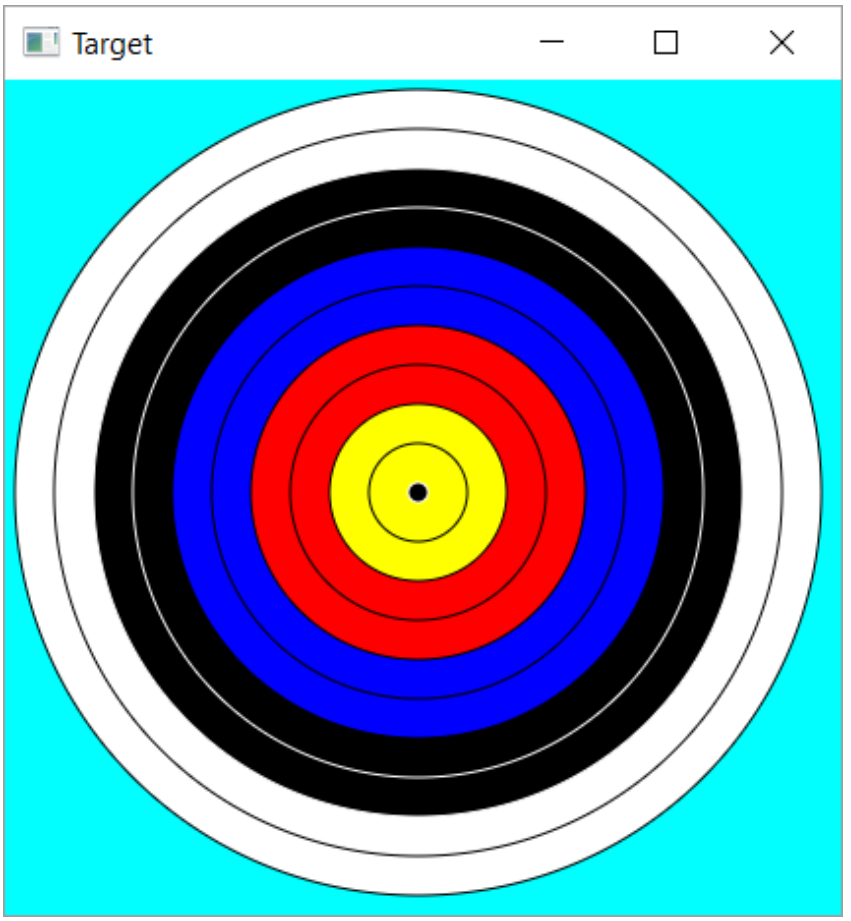
```
1 import javafx.application.Application;
 2 import javafx.scene.Scene;
 3 import javafx.scene.layout.Pane;
 4 import javafx.scene.paint.Color;
 5 import javafx.stage.Stage;
 6 import javafx.scene.shape.Ellipse;
 7
 8 public class ShowEllipse extends Application {
 9   @Override // Predefinisanje metoda start u klasi Application
 10   public void start(Stage primaryStage) {
 11  // Kreiranje okna
 12  Pane pane = new Pane();
 13
 14  for (int i = 0; i < 16; i++) {
 15    // Kreiranje elipse i njeno dodavanje u okno
 16    Ellipse e1 = new Ellipse(150, 100, 100, 50);
 17    e1.setStroke(Color.color(Math.random(), Math.random(),
 18    Math.random()));
 19    e1.setFill(Color.WHITE);
 20    e1.setRotate(i * 180 / 16);
 21    pane.getChildren().add(e1);
 22  }
 23
 24  // Kreiranje scene i njeno postavljanje na pozornicu
 25  Scene scene = new Scene(pane, 300, 200);
```

```
26  primaryStage.setTitle("ShowEllipse"); // Unos naziva pozornice
27  primaryStage.setScene(scene); // Postavljanje scene na pozornicu
28  primaryStage.show(); // Prikaz pozornice
29  }
30 }
```

ZADATAK 8.4

Samostalno vežbanje korišćenja klase Circle i Ellipse

4. Crtajući koncentrične krugove, pokušajte da napravite JavaFX program koji crta metu kao na slici 3.



Slika 8.5.3 Samostalno vežbanje korišćenja klase Circle

8.5 Klasa Arc

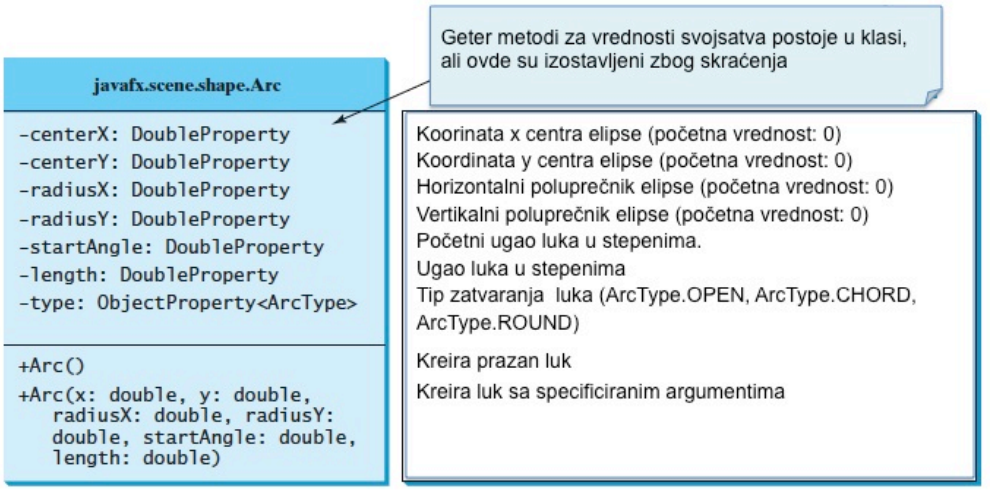
SVOJSTVA KLASE ARC

Klasa Arc definiše luk.

Klasa ***Arc*** definiše luk. UML dijagram klase***Arc*** je prikazan na slici 1.

Luk je deo elpse definisan parametrima: ***centerX , centerY , radiusX , radiusY , startAngle , length ,*** i tip luka ***type (ArcType.OPEN , ArcType.CHORD ,*** ili***ArcType.ROUND***).

Parametar ***startAngle*** je početni ugao, a ***length*** je ugao prostiranja luka (tj. Ugao obuhvaćen lukom). Uglovi se mere u stepenima i primenjuju uobičajenu matematičku knvenciju (tj. 0 stepeni je u pravcu istoka, a pozitivan ugao znači ugao u smeru suprotnom od kretanja kazaljke na satu), kao što je pokazano na slici 2.a.



Slika 8.6.1 UML dijagram klase Arc koja crta luk

DEFINISANJE UGLOVA LUKA

Ugao je negativan ako se dobija oktretanjem poluprečnika luka u smeru okretanja kazaljke na satu, počev od istočnog pravca.

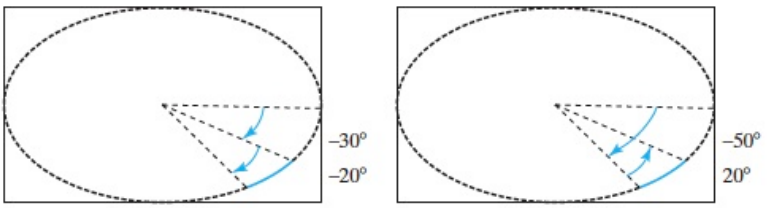
Uglovi mogu biti i negativni. To znači da se kreće u smeru kazaljke na satu, počev od istočnog pravca (slika 3). Negativni prelazni ugao označava ugao koji se formira kretanje u smeru obrtanja skazaljke na satu počev od početnog ugla. Sledeća dva iskaza prikazuju definišu isti ugao.

:

```
new Arc(x, y, radiusX, radiusY, -30, -20);
 new Arc(x, y, radiusX, radiusY, -50, 20);
```

Prvi iskaz upotrebljava početni ugao -30 i negativni prelazni ugao -20, kao što je prikazano na slici 3.a.,

Drugi iskaz upotrebljava negativni početni ugao -50 i pozitiv prelazni ugao 20, kao što je prikazano na slici 3.b.



(b) Negativni početni ugao -30° i egativni ugao prostiranja -20°

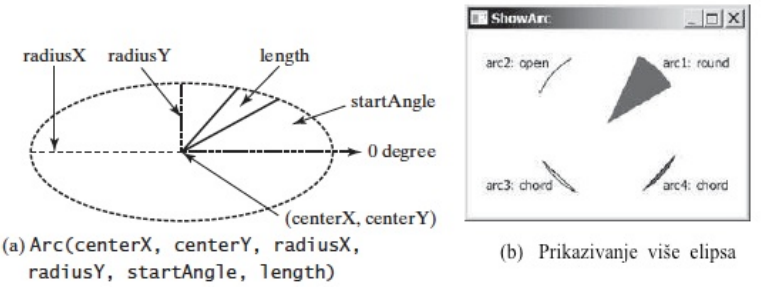
(b) Negativni početni ugao -50° i positivni ugao prostiranja 20°

Slika 8.6.2 Ugao može biti i negativan

PRIMER 20 - CRTANJE LUKOVA

Program crta četiri luka i odsečka luka koji su prikazani na slici 2b.

Na slici 2a prikazani su prikazani parametri elipse, a na slici 2b primer korišćenja klase Arc. Prikazan listing programa **ShowArc** prikazuje kod koji ostvaruje rezultat prikazan na slici 2b.



(a) `Arc(centerX, centerY, radiusX, radiusY, startAngle, length)`

(b) Prikazivanje više elipsa

Slika 8.6.3 Objekat Arc prikazuje jedan luk

Program kreira luk arc1 sa centrom (150,100) sa poluprečniko, radiusX 80 i radiusY 80. Početni ugao je 30, a prelazni ugao je 35 (linija 15). Tip luka arc1 je ArcTzpe.ROUND (linija 18). Arc1 je popunjen crvenom bojom, kao i njegova granična linija

Program kreira i luk arc3, sa centrom (150,100) poluprečnikom radiusX 80 i radiusZ 80. Početni ugao je 30 + 180 sa prelaznim uglom 35 (linija 29(, Tip luka arc3 je . ArcTzpe.CHORD (linija 31). Boja popune luka arc3 je bela i boja granične linije odesčka luka je crna.

```
1 import javafx.application.Application;
 2 import javafx.scene.Scene;
 3 import javafx.scene.layout.Pane;
 4 import javafx.scene.paint.Color;
 5 import javafx.stage.Stage;
 6 import javafx.scene.shape.Arc;
 7 import javafx.scene.shape.ArcType;
 8 import javafx.scene.text.Text;
 9
 10 public class ShowArc extends Application {
 11   @Override // Predefinisanje metoda start u klasi Application
 12   public void start(Stage primaryStage) {
 13  // Kreiranje okna
 14  Pane pane = new Pane();
```

```
15
16  Arc arc1 = new Arc(150, 100, 80, 80, 30, 35); // Kreiranje luka
17  arc1.setFill(Color.RED); // Unos boje popunjavanja segmenta luka
18  arc1.setType(ArcType.ROUND); // Unos tipa lika
19  pane.getChildren().add(new Text(210, 40, "arc1: round"));
20  pane.getChildren().add(arc1); // Dodavanje luka u okno
21
22  Arc arc2 = new Arc(150, 100, 80, 80, 30 + 90, 35);
23  arc2.setFill(Color.WHITE);
24  arc2.setType(ArcType.OPEN);
25  arc2.setStroke(Color.BLACK);
26  pane.getChildren().add(new Text(20, 40, "arc2: open"));
27  pane.getChildren().add(arc2);
28
29  Arc arc3 = new Arc(150, 100, 80, 80, 30 + 180, 35);
30  arc3.setFill(Color.WHITE);
31  arc3.setType(ArcType.CHORD);
32  arc3.setStroke(Color.BLACK);
33  pane.getChildren().add(new Text(20, 170, "arc3: chord"));
34  pane.getChildren().add(arc3);
35
36  Arc arc4 = new Arc(150, 100, 80, 80, 30 + 270, 35);
37  arc4.setFill(Color.GREEN);
38  arc4.setType(ArcType.CHORD);
39  arc4.setStroke(Color.BLACK);
40  pane.getChildren().add(new Text(210, 170, "arc4: chord"));
41  pane.getChildren().add(arc4);
42
43  // Kreiranje scene i njeno postavljanje na pozornicu
44  Scene scene = new Scene(pane, 300, 200);
45  primaryStage.setTitle("ShowArc"); // Unos naslova pozornice
46  primaryStage.setScene(scene); // postavljanje scene pozornice
47  primaryStage.show(); // Prikaz pozornice
48   }
49 }
```

ZADATAK 8.5

Koristeći klasu Arc, nacrtati figuru kao na slici.

5. Koristeći klasu Arc, nacrtati figuru kao na slici 4.

Slika 8.6.4 Koristeći klasu Arc, nacrtati figuru kao na slici

8.6 Klase Polygon i Polyline

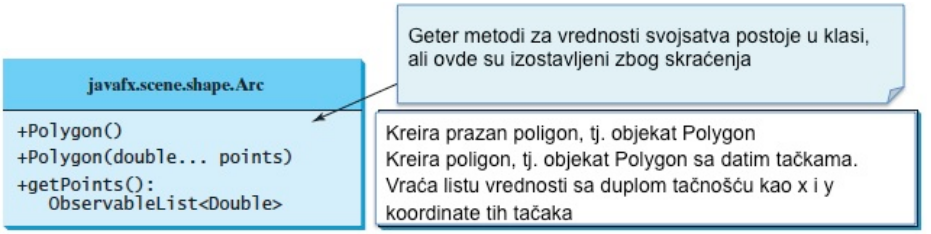
SVOJSTVA KLASE POLYGON

Klasa Polygon definiše poligon koji povezuje niz tačaka.

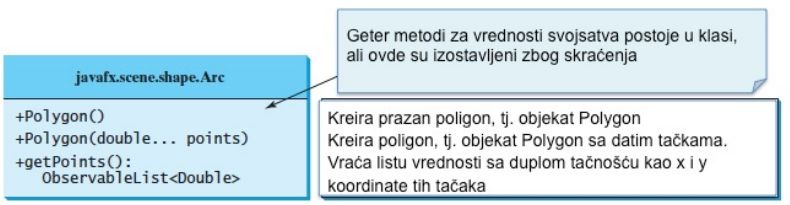
Na slici 1 prikazan je UML dijagram klase ***Polygon*** koja definiše poligon koji povezuje niz tačaka kao što je prikazano na slici 2.a.

Klasa ***Polyline*** je slična klasa klasi ***Polygon*** sem što klasa ***Polyline*** se automatski ne zatvara, kao što je prikazano na slici 2.b.

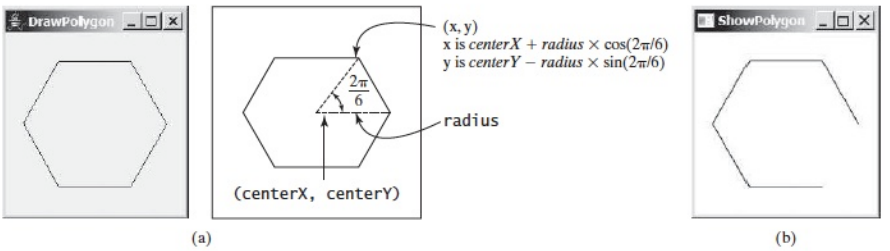
Na slici 3 su prikazani parametri klasa ***Polygon*** i ***Polyline***.



Slika 8.7.1 Klasa Polygon



Slika 8.7.2 Polygon crta zatvorenu liniju,a Polyline otvorenu liniju kroz niz tačaka



Slika 8.7.3 Parametri klasa Polygon i Polyine

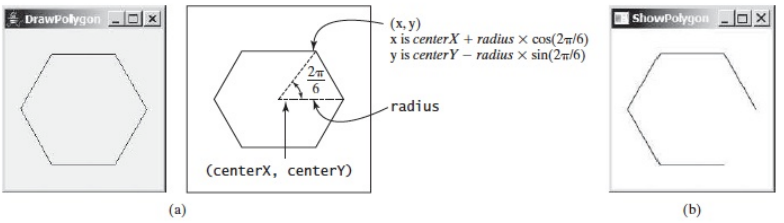
PRIMER 21 - PRIMENA KLASE POLYGON

Program crta heksagon, uz pomoć klase Polygon.

Program kreira Polygon objekat (linija 14) i dodaje ga u okvir (linija 15). Metod ***getPoints()*** vraća listu ***aObservableList<Double>*** (linija 18), koja sadrži add() metod za dodavanje elemenata u listu (linije 26-27). Parametar metoda add(vlue) mora bite duple tačnosti, tj. tipa ***Double.***

Petlja dodaje šest tačaka poligona (linije 25-28). Svaka tačka je predstavljena svojim x- i y-koordinate heksagona na slici Koordinate heksagona se računaju prema formulama datim na slici 3.a.

Ako se objekat Polygon zameni sa objektom Polyline, program onda prikazuje polilinju, kao na slici 3.b.



Slika 8.7.4 Parametri klasa Polygon i Polyine

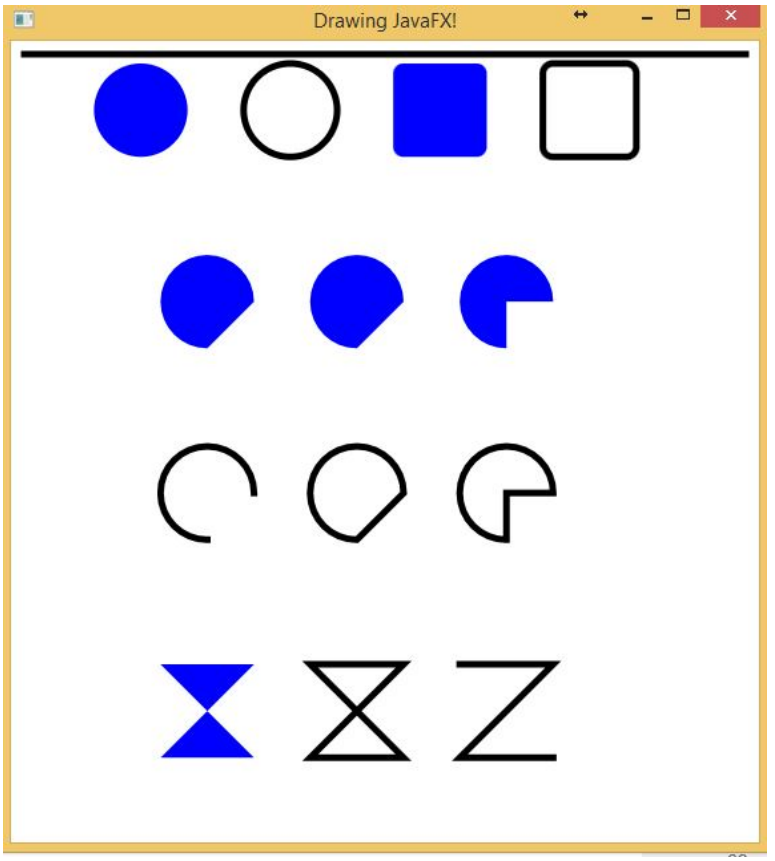
```
1 import javafx.application.Application;
 2 import javafx.collections.ObservableList;
 3 import javafx.scene.Scene;
 4 import javafx.scene.layout.Pane;
 5 import javafx.scene.paint.Color;
 6 import javafx.stage.Stage;
 7 import javafx.scene.shape.Polygon;
 8
 9   public class ShowPolygon extends Application {
 10  @Override // Redefinisanje metoda start klase Application
 11  public void start(Stage primaryStage) {
 12    // Create a pane, a polygon, and place polygon to pane
 13    Pane pane = new Pane();
```

```
14    Polygon polygon = new Polygon();
15    pane.getChildren().add(polygon);
16    polygon.setFill(Color.WHITE);
17    polygon.setStroke(Color.BLACK);
18    ObservableList<Double> list = polygon.getPoints();
19
20    final double WIDTH = 200, HEIGHT = 200;
21    double centerX = WIDTH / 2, centerY = HEIGHT / 2;
22    double radius = Math.min(WIDTH, HEIGHT) * 0.4;
23
24    // unos tačaka poligona
25    for (int i = 0; i < 6; i++) {
26    list.add(centerX + radius * Math.cos(2 * i * Math.PI / 6));
27    list.add(centerY - radius * Math.sin(2 * i * Math.PI / 6));
28   }
29
30   // Kreiranje scene i njeno postavljanje na pozornicu
31   Scene scene = new Scene(pane, WIDTH, HEIGHT);
32   primaryStage.setTitle("ShowPolygon"); // Unos naslova pozornice
33   primaryStage.setScene(scene); // Postavljanje scene na pozornicu
34   primaryStage.show(); // Prikaz pozornice
35   }
36 }
```

PRIMER 21

Cilj zadatka je da se demonstrira iscrtavanje podobjekata klase Shape pomoću GraphicsContext-a u Javi FX

Napraviti aplikaciju koja crta sve što je prikazano na sledećoj slici. Koristiti GraphicsContext za crtanje preko JaveFX



Slika 8.7.5 Zadatak 7

PRIMER 21 - REŠENJE

Cilj ovog zadatka je da prikaže korišćenje crtanja pomoću GraphicsContext-a u Javi FX

Klasa MyCanvas:

```
/**
 * Klasa MyCanvas nasladjuje klasu Canvas i sluzi za demonstraciju iscrtavanja
objekata
 * koji nasledjuju klasu Shape
 * @author Rados
 */
public class MyCanvas extends Canvas {
    // atribut graphicsContext na osnovu kog znamo da radimo sa 2D grafikom
    GraphicsContext gc2d = this.getGraphicsContext2D();

    public MyCanvas() {
        super();
        setWidth(600);
        setHeight(600);
        draw();
    }
```

```
    // metoda koja iscrtava sav sadrzaj unutar canvas-a
    public void draw() {
        gc2d.clearRect(0, 0, getWidth(), getHeight());
        gc2d.setFill(Color.BLUE);
        gc2d.setStroke(Color.BLACK);
        gc2d.setLineWidth(5);
        gc2d.strokeLine(10, 10, getWidth() - 10, 10);
        gc2d.fillOval(getWidth() / 5 - 50, getHeight() / 9 - 50, getWidth() / 8,
getWidth() / 8);
        gc2d.strokeOval(2 * getWidth() / 5 - 50, getHeight() / 9 - 50, getWidth() /
8, getWidth() / 8);
        gc2d.fillRoundRect(3 * getWidth() / 5 - 50, getHeight() / 9 - 50,
getWidth() / 8, getWidth() / 8, 15, 15);
        gc2d.strokeRoundRect(4 * getWidth() / 5 - 50, getHeight() / 9 - 50,
getWidth() / 8, getWidth() / 8, 15, 15);
        gc2d.fillArc(getWidth() / 5, 3 * getHeight() / 9 - 40, getWidth() / 8,
getWidth() / 8, 0, 270, ArcType.OPEN);
        gc2d.fillArc(2 * getWidth() / 5, 3 * getHeight() / 9 - 40, getWidth() / 8,
getWidth() / 8, 0, 270, ArcType.CHORD);
        gc2d.fillArc(3 * getWidth() / 5, 3 * getHeight() / 9 - 40, getWidth() / 8,
getWidth() / 8, 0, 270, ArcType.ROUND);
        gc2d.strokeArc(getWidth() / 5, 5 * getHeight() / 9 - 30, getWidth() / 8,
getWidth() / 8, 0, 270, ArcType.OPEN);
        gc2d.strokeArc(2 * getWidth() / 5, 5 * getHeight() / 9 - 30, getWidth() /
8, getWidth() / 8, 0, 270, ArcType.CHORD);
        gc2d.strokeArc(3 * getWidth() / 5, 5 * getHeight() / 9 - 30, getWidth() /
8, getWidth() / 8, 0, 270, ArcType.ROUND);
        gc2d.fillPolygon(new double[]{getWidth() / 5, 13 * getWidth() / 40,
getWidth() / 5, 13 * getWidth() / 40},
                    new double[]{7 * getHeight() / 9, 7 * getHeight() / 9, 7 *
getHeight() / 9 + getWidth() / 8, 7 * getHeight() / 9 + getWidth() / 8}, 4);
        gc2d.strokePolygon(new double[]{2 * getWidth() / 5, 21 * getWidth() / 40, 2
* getWidth() / 5, 21 * getWidth() / 40},
                    new double[]{7 * getHeight() / 9, 7 * getHeight() / 9, 7 *
getHeight() / 9 + getWidth() / 8, 7 * getHeight() / 9 + getWidth() / 8}, 4);
        gc2d.strokePolyline(new double[]{3 * getWidth() / 5, 29 * getWidth() / 40,
3 * getWidth() / 5, 29 * getWidth() / 40},
                        new double[]{7 * getHeight() / 9, 7 * getHeight() / 9, 7 *
getHeight() / 9 + getWidth() / 8, 7 * getHeight() / 9 + getWidth() / 8}, 4);
    }

}
```

Klasa DrawingMain:

```
public class DrawingMain extends Application {

    private MyCanvas canvas = new MyCanvas();

    @Override
    public void start(Stage primaryStage) {
        // kreiranje AnchorPane korenog container-a
        AnchorPane root = new AnchorPane();
```

```
        Scene scene = new Scene(root, 600, 600);
        // ubacivanje canvas objekta unutar root container-a
        root.getChildren().add(canvas);

        primaryStage.setTitle("Drawing JavaFX!");
        primaryStage.setScene(scene);
        primaryStage.show();

        // listener-i koji osluskuju promenu velicine scene unutar stage-a
        scene.widthProperty().addListener(new ChangeListener<Number>() {
            @Override
            public void changed(ObservableValue<? extends Number> observableValue,
Number oldSceneWidth, Number newSceneWidth) {
                canvas.setWidth((double) newSceneWidth);
                canvas.draw();
            }
        });
        scene.heightProperty().addListener(new ChangeListener<Number>() {
            @Override
            public void changed(ObservableValue<? extends Number> observableValue,
Number oldSceneHeight, Number newSceneHeight) {
                canvas.setHeight((double) newSceneHeight);
                canvas.draw();
            }
        });

    }

    /**
    * @param args the command line arguments
    */
    public static void main(String[] args) {
        launch(args);
    }

}
```

PRIMER 21 - OBJAŠNJENJE

Cilj sekcije je da se pojasni kod zadatka 7

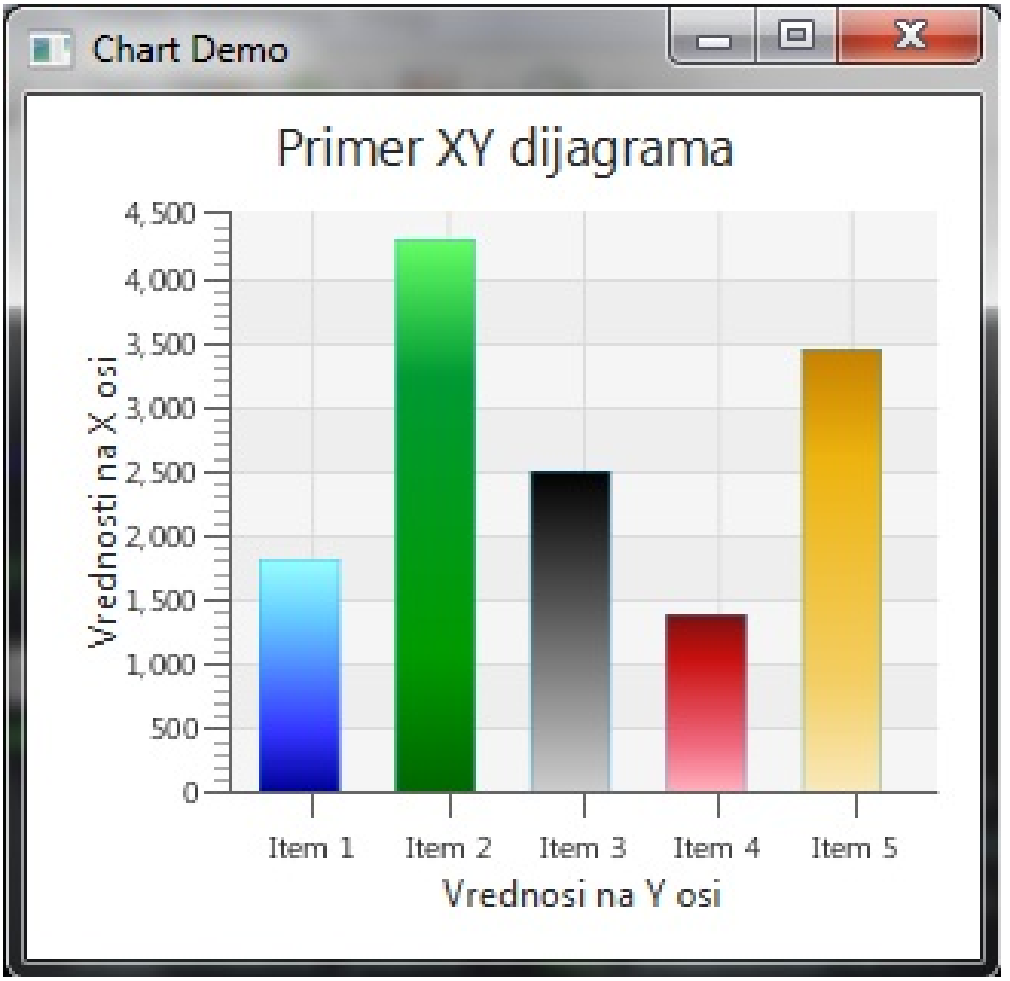
Ovaj primer demonstrira iscrtavanje raznih objekata koji pripadaju potklasama klase Shape koji su navedeni u predavanjima.

Ovde se koristi kontejner AnchorPane koji ima mogućnost da prati promene dimenzija prozora i omogućava nam da obrađujemo događaje promene visine ili širine prozora što je korisno u slučaju da želimo da pravimo prilagodljive crteže tj. crteže koji menjaju dimenzije u skladu sa promenom veličine prozora.

PRIMER 22

Cilj zadatka je da se prikaže rad sa dijagramom i korišećenje CSS-a za stilizovanje.

Koristeći JavuFX (BarChart klasu), napraviti dijagram sa X i Y osama na kojem će na X osi biti prikazani tekstualni elementi dok će se na Y osi nalaziti odgovarajuće numeričke vrednosti. Takođe koristeći CSS potrebno je stilizovati pravouganike koji predstavljaju podeoke na dijagramu. Vrednosti na X i Y osama mogu biti proizvoljne.



Slika 8.7.6 Primer XY dijagrama

PRIMER 23 - REŠENJE

Prikaz programskog koda zadatka 8

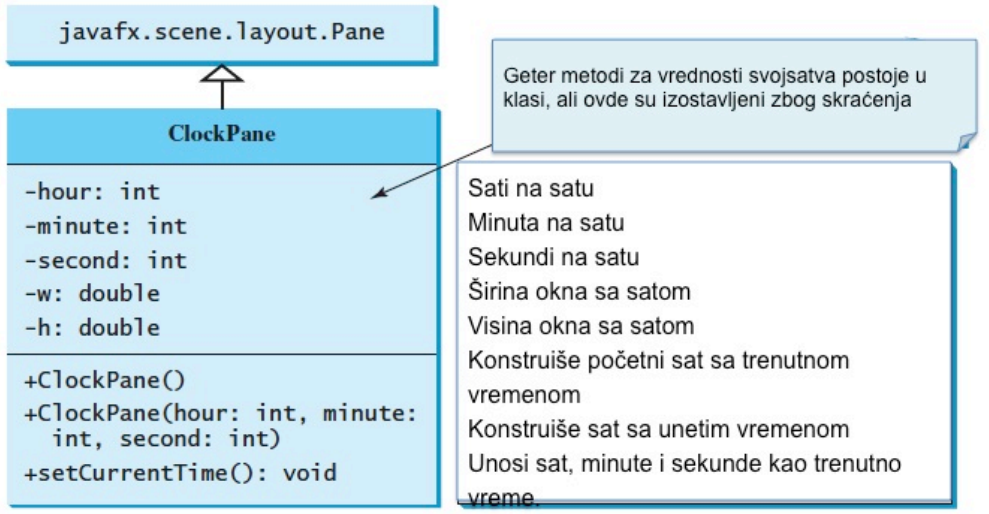
```
public class ChartMain extends Application {

    @Override
    public void start(Stage stage) {
        // definisemo da ce vrednosti na X osi biti tipa string tj. tekstulane
        CategoryAxis xAxis = new CategoryAxis();
```

```
        // definisemo da ce vrednosti na Y osi bit brojevi
        NumberAxis yAxis = new NumberAxis();
        // kreiramo objekat klase BarChart koji prikazuje koordinatni sistem sa X i
Y osama i pri kreiranju
        // navodimo kog tipa ce biti X i Y ose
        BarChart<String, Number> bc = new BarChart<>(xAxis, yAxis);
        // postavljamo naslov dijagrama kao i naziv X i Y osa
        bc.setTitle("Primer XY dijagrama");
        xAxis.setLabel("Vrednosi na Y osi");
        yAxis.setLabel("Vrednosti na X osi");
        // kriramo objekat klase XYChart.Series koji cini kolekciju parova podataka
za X i Y osu
        XYChart.Series series = new XYChart.Series();
        // dodajemo vrendnosti za X i Y osu u parovima kao objekat XYChart.Data
klase
        series.getData().add(new XYChart.Data("Item 1", 1800.58));
        series.getData().add(new XYChart.Data("Item 2", 4300.40));
        series.getData().add(new XYChart.Data("Item 3", 2500.50));
        series.getData().add(new XYChart.Data("Item 4", 1390.75));
        series.getData().add(new XYChart.Data("Item 5", 3450.89));

        Scene scene = new Scene(bc, 600, 600);
        // ucitavamo css fajl u kome cemo stilizovati dijagram

scene.getStylesheets().add(ChartMain.class.getResource("stylechars.css").toString())
;
        bc.getData().add(series);
        bc.setLegendVisible(false);
        stage.setTitle("Chart Demo");
        stage.setScene(scene);
        stage.show();
    }

    /**
    * @param args the command line arguments
    */
    public static void main(String[] args) {
        launch(args);
    }
```

PRIMER 24 - KLASA CLOCKPANE

Klasa ClockPane kreira, crta i boji sat.

UML dijagram klase ClockPane je prikazan na slici 1.



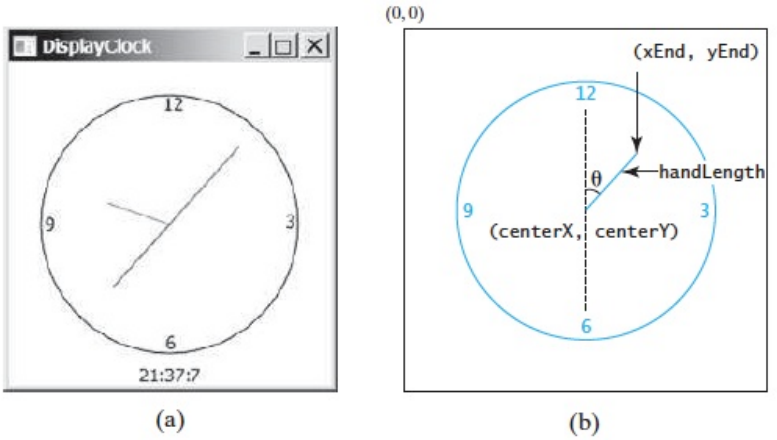
Slika 8.7.7 Klasa ClockPane prikazuje analogni sat

PRIMER 24 - LISTING PROGRAMA DISPLAYCLOCK

Program prikazuje analogni sat sa oznakama sata, minuta i sekundi.

Pod pretpostavkom da je klasa ClockPane raspoloživa (biće opisana kasnije), ovde se prikazuje listing programa za prikaz analognog sata i za upotrebu natpisa za prikaz sata, minuta i sekundi kao što je prikazano na slici 2.

.



Slika 8.7.8 Izgled sata koji prikazuje program ClockPane

Koordinate vrha skazaljke se računa izrazima:

endX = centerX + handLength × sin(θ)

endY = centerY - handLength × cos(θ)

```
1 import javafx.application.Application;
 2 import javafx.geometry.Pos;
```

```
3 import javafx.stage.Stage;
4 import javafx.scene.Scene;
5 import javafx.scene.control.Label;
6 import javafx.scene.layout.BorderPane;
7
8 public class DisplayClock extends Application {
9   @Override // Predefinisanje metoda start u klasi Application
10   public void start(Stage primaryStage) {
11  // Kreiranje sata i naslova
12  ClockPane clock = new ClockPane();
13  String timeString = clock.getHour() + ":" + clock.getMinute()
14      + ":" + clock.getSecond();
15  Label lblCurrentTime = new Label(timeString);
16
17  // Postavljanje sata i nalepnice nu okno  BorderPane
18  BorderPane pane = new BorderPane();
19  pane.setCenter(clock);
20  pane.setBottom(lblCurrentTime);
21  BorderPane.setAlignment(lblCurrentTime, Pos.TOP_CENTER);
22
23  // Kreiranje scene i postavljanje je na pozornicu
24  Scene scene = new Scene(pane, 250, 250);
25  primaryStage.setTitle("DisplayClock"); // Unos imena pozornice
26  primaryStage.setScene(scene); // STavljanje scene na pozornicu
27  primaryStage.show(); // Prikaz pozornice
28   }
29 }
```

PRIMER 24 - LISTING KLASE CLOCKPANE

Klasa ClockPane crta analogni sat u oknu

Jedan minut ima 60 sekundi, te je ugao okretanje sklazajke sa sekundama:

second × (2π/60)

Položaj skazalje sa minutima zavisi od minuta i sekundi. Vrednost minuta se određuje je: minute + second/60.

Ugao skayaljke sa minutima se određuje izrazom:

(minute + second/60) × (2π/60)

Kako u jednom krugu postoji 12 sati, ugao skazaljke sa satom se računa pomoću izraza:

(hour + minute/60 + second/(60 × 60)) × (2π/12)

Ako se zanemari uticaj na sekundi, moigu se koristiti sledeće pojednostavljeni izrazi:

$$
\begin{aligned}
secondX &= centerX + secondHandLength \times \sin(second \times (2\pi/60)) \\
secondY &= centerY - secondHandLength \times \cos(second \times (2\pi/60)) \\
minuteX &= centerX + minuteHandLength \times \sin(minute \times (2\pi/60)) \\
minuteY &= centerY - minuteHandLength \times \cos(minute \times (2\pi/60)) \\
hourX &= centerX + hourHandLength \times \sin((hour + minute/60) \times (2\pi/12)) \\
hourY &= centerY - hourHandLength \times \cos((hour + minute/60) \times (2\pi/12))
\end{aligned}
$$

Slika 8.7.9 Izrazi za obračun koordinata vrhova sklazaljki

```
1 import java.util.Calendar;
 2 import java.util.GregorianCalendar;
 3 import javafx.scene.layout.Pane;
 4 import javafx.scene.paint.Color;
 5 import javafx.scene.shape.Circle;
 6 import javafx.scene.shape.Line;
 7 import javafx.scene.text.Text;
 8
 9 public class ClockPane extends Pane {
 10   private int hour;
 11   private int minute;
 12   private int second;
 13
 14   // Širina i visina okna sata
 15   private double w = 250, h = 250;
 16
 17   /** Konstruisanje početnog sata sata sa trenutnim vremenom */
 18   public ClockPane() {
 19  setCurrentTime();
 20   }
 21
 22   /** Konstruisanje sata sa specificiranim satim, minutima i sekundama  */
 23   public ClockPane(int hour, int minute, int second) {
 24  this.hour = hour;
 25  this.minute = minute;
 26  this.second = second;
 27  paintClock();
 28   }
 29
 30   /** Vraća hour */
 31   public int getHour() {
 32  return hour;
 33   }
 34
 35   /** Unisi novu vrednost za  hour */
 36   public void setHour(int hour) {
 37  this.hour = hour;
 38  paintClock();
 39   }
 40
 41 /** vraća minute */
 42   public int getMinute() {
 43  return minute;
 44   }
```

```
45
46   /** Unosi novu vrednost za minute */
47   public void setMinute(int minute) {
48  this.minute = minute;
49  paintClock();
50   }
51
52   /** Vraća second */
53   public int getSecond() {
54  return second;
55   }
56
57   /** Unosi novu vrednost za second */
58   public void setSecond(int second) {
59  this.second = second;
60  paintClock();
61   }
62
63   /** Vraća širinu okna za sat  */
64   public double getW() {
65  return w;
66   }
67
68   /** Unosi širinu okna za sat */
69   public void setW(double w) {
70  this.w = w;
71  paintClock();
72   }
73
74   /** Vraća visinu okna za sat  */
75  public double getH() {
76  return h;
77   }
78
79   /** Podešava visinu okna sata  */
80   public void setH(double h) {
81  this.h = h;
82  paintClock();
83   }
84
85   /* Unosi sadašnje vreme u sat  */
86   public void setCurrentTime() {
87  // Konstruiše kalendar ya sadašnji datum i vreme
88  Calendar calendar = new GregorianCalendar();
89
90  // Unos trenutnog vremena sata, minuta i sekundi
91  this.hour = calendar.get(Calendar.HOUR_OF_DAY);
92  this.minute = calendar.get(Calendar.MINUTE);
93  this.second = calendar.get(Calendar.SECOND);
94
95  paintClock(); // nacrtaj sat
96   }
97
```

```
98   /** Crtanje sata  */
99   protected void paintClock() {
100     // Inicijalizacija parametra sata
101     double clockRadius = Math.min(w, h) * 0.8 * 0.5;
102     double centerX = w / 2;
103     double centerY = h / 2;
104
105     // Crtanje kruga
106     Circle circle = new Circle(centerX, centerY, clockRadius);
107     circle.setFill(Color.WHITE);
108     circle.setStroke(Color.BLACK);
109     Text t1 = new Text(centerX - 5, centerY - clockRadius + 12, "12");
110     Text t2 = new Text(centerX - clockRadius + 3, centerY + 5, "9");
111     Text t3 = new Text(centerX + clockRadius - 10, centerY + 3, "3");
112     Text t4 = new Text(centerX - 3, centerY + clockRadius - 3, "6");
113
114     // Crtanje skazaljke za prikaz sekundi
115     double sLength = clockRadius * 0.8;
116     double secondX = centerX + sLength *
117     Math.sin(second * (2 * Math.PI / 60));
118     double secondY = centerY - sLength *
119     Math.cos(second * (2 * Math.PI / 60));
120     Line sLine = new Line(centerX, centerY, secondX, secondY);
121     sLine.setStroke(Color.RED);
122
123     // Crtanje velike skazalje za prikaz minuta
124     double mLength = clockRadius * 0.65;
125     double xMinute = centerX + mLength *
126     Math.sin(minute * (2 * Math.PI / 60));
127     double minuteY = centerY - mLength *
128     Math.cos(minute * (2 * Math.PI / 60));
129     Line mLine = new Line(centerX, centerY, xMinute, minuteY);
130     mLine.setStroke(Color.BLUE);
131
132     // Crtanje sklayalje ya prikaz sati
133     double hLength = clockRadius * 0.5;
134     double hourX = centerX + hLength *
135         Math.sin((hour % 12 + minute / 60.0) * (2 * Math.PI / 12));
136     double hourY = centerY - hLength *
137         Math.cos((hour % 12 + minute / 60.0) * (2 * Math.PI / 12));
138     Line hLine = new Line(centerX, centerY, hourX, hourY);
139     hLine.setStroke(Color.GREEN);
140
141     getChildren().clear();
142     getChildren().addAll(circle, t1, t2, t3, t4, sLine, mLine, hLine);
143   }
144 }
```

PRIMER 24 - OBJAŠNJENJE LISTINGA PROGRAMA CLOCKPANE

Metod paintClock se poziva uvel kada se promeni bilo koje svojstvo sata (hour, minute, second, w i h)

Program prikazuje analogni sat sa trenutnom vremenom korišćenjem početnog konstruktora bez argumenata (linije 18-20) . Prikazuje se sat sa specificiranim satom, minutom i sekundom upotrebom konstruktora (linije 23-28). Konstruktor sa trenutnim vremenom je dat na linijama 86-96).

Klasa definiše svojstva: sat, minut i sekunda koja se memoriše kao trenutno vreme (linije 10-12). Širina i visina okvira je definisano u linijama 15). Početne vrednosti za ***w*** i ***H*** su 250. To s emože promeniti metodima ***setW*** i ***setH*** (linije 69, 80). Te veličine s ekoriste za crtanja sata u oknu sa metodom ***paintClock().***

.

.

Metod ***paintClock()*** boji i crta sat (linije 99-143). Poluprečnik sata je proporcionalan širini i visini okna (linija 101). Krug sata je postavljen u centar okna (linija 106). Tekst koji pokazuje sate: 12,, 3, 6, 9 je kreiran u linijama 109-112.

Skazaljka-sekundara, skazaljka ya minute i skazaljka ya satove su linij ekreirane u linijama 114-139. Metod ***paintClock()*** postavlja sve ove oblike u okbu upotrebom addAll metod u listu (linija 142). Metod ***paintCloick*** se poziva uvel kada se promeni bilo koje svojstvo sata (hour, minute, second, w i h) u linijama 27, 38, 49, 60, 71, i 82, i 95. Pre ubacivanja novih svojstava u sadrćaj okna, briše se prethodni (linija 141)

ZADACI 8.6 - 8.15

Proverite svoje razumevanje klasa Polygon i Polyline.

6. Kako prikazujete tekst, liniju, pravougaonik, krug,elipsu, luk, poligon i poliliniju?

7. Napišite deo koda za prikaz stringa zarotitaronoga za 45 stepeni u centru okna.

8. Napišite deo koda za prikaz debele linije sa 10 piksela iz (10,10) do (70,30).

9. Napišite deo koda za prikaz unutranjosti pravougaonika sa crvenom bojom širine 100 i visine 50 sa gornjim levim uglom u (10,10)-

10. Napišite deo koda za prikaz pravougaonika sa zaobljenim uglovima sa širinom 100, visionom 200 sa gornjim levim uglom u (10,10) , horizontalnim prečnikom 40, i vertikalnim prečnikom u uglu sa 20.

11. Napišite deo koda za prikaz elipse sa horizontalnim poluprečnikom 50 i veritikalnim poluprečnikom 100.

12. Napišite delove koda za prikaz spoljašnjeg dela gornjeg levog dela kruga sa poluprečnikom 50.

13. Napišite deo koda za prikaz donje polovine kruga sa poluprečnikom 50 i sa popunom sa crvenom bojom.

14. Napišite delove koda za prikaz poligona koji povezuje sledeće tačke: (20, 40), (30, 50), (40, 90), (90, 10), (10, 30), i popunite poligon sa zelenom bojom.

15. Napišite delove koda za prikaz polilinije koji povezuje sledeće tačke: (20, 40), (30, 50), (40, 90), (90, 10), (10, 30).

Poglavlje 9

Domaći zadaci

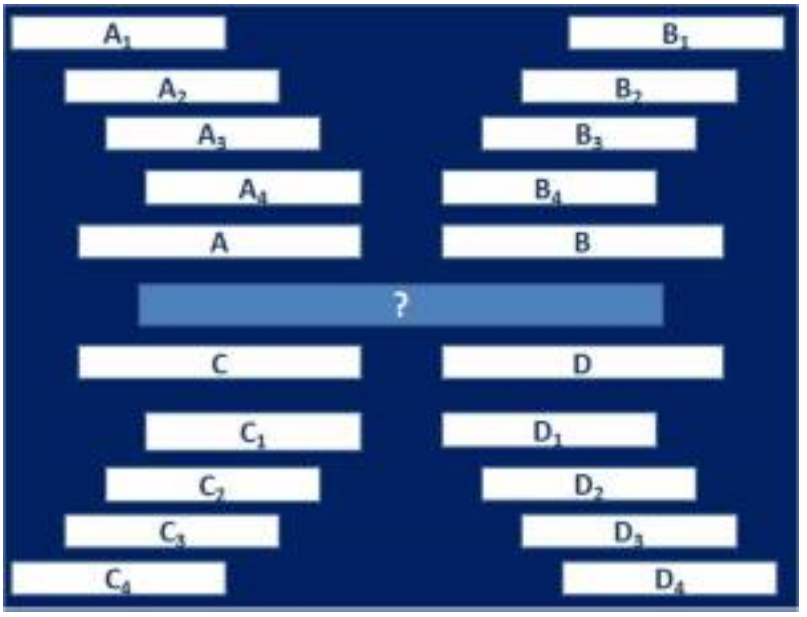
ZADACI ZA DOMAĆI RAD

Ove zadatke student treba da radi samostalno za vreme individualnih vežbi.

1. Napisati program koji iscrtava n krugova iste veličine organizovanih u strukturu matrice. Program treba da prima kao parametar broj vrsta i kolona krugova i krugovi treba da budu različitih boja.
2. Napraviti program koji kao parametar može da primi brojeve od 1 do 6 (kao na kocki za igru) i u zavisnosti od primljenih parametara na ekranu iscrtava izgled kocke za igru. Raspored krugova treba da bude kao na kocki za igru. Potruditi se da krugovi budu prilagodljivi na promenu veličine ekrana.

4. Isprobati rad sa dijagramima koristeći „Pie“ strukturu dijagrama. Može se koristiti gotova JavaFX klasa PieChart ili se realizovati sopstveno rešenje iscrtavanjem delova krugova.

5. Koristeći JavaFX napraviti strukturu dugmića u obliku X slova kao u igri asocijacija na slagalici. Dugmiće stilizovati korišćenjem CSS-a.



Zaključak

REZIME

Glavne poruke lekcije

1. JavaFX je novi radniokvir za razvoj bogate Internet aplikacije. JavaFX u celosti zamenjuje Swing i AWT.
2. Metod main() u JavaFX mora da proširi javafx.application.Application i primenjuju metod start(). Primarna pozornica se automatski kreira od strane JVM i ubacuje u metod start().
3. Pozornica je prozor za prikaz scene. Možete dodavati čvorove u scenu. Okna (panes), kontrole (controls) i oblici (shapes) su čvorovi. Okna se mogu da koriste kao kontejneri u čvorova.
4. Svojstvo povezivanja se može vezati za izvorni objekat koji se posmatra (observable). Promena u izvornom objektu će se automatski odraziti na povezano svojstvo. Svojstvo povezivanja ima vrednost u geter metodu, vrednost u seter metodu i svojstvo u geter metodu.
5. Klasa Node definiše mnoga svojstva koja su zajednička ya sve švorove. Mogu se primeniti kod okana, kontrola i oblika.
6. Možete kreirati Color objekat sa specificiranom crvenom, zelenom i plavom komponentom, a vrednostu prozirnosti.
7. Možete kreirati objekat Font, i uneti njegov naziv, veličinu, širinu i postavka.
8. Klasa javafx.scene.image.Image se koriste za dovođenje slike i njeno prikazivanje u ImageView objektu.
9. JavaFX obezbeđuje mnogo tipova okana za automatsko raspoređivanje čvorova u željenu lokaciju i veličinu. Klasa Pane je osnovna klasa ya sva okna. Sadrži metod getChilidren() koji vraća listu ObservableList. Možete upotrebiti metode add(node) i addAll(node1, node 2, ...) radi dodavanja čvorova u okno.
10. Okno FlowPane uređuje čvorove horizontalno, s leva u desno, ili vertikalno, od gore ka dole i redosledu njihovog dodavanja. Okno GridPane uređuje čvorove u formatu matrice. Čvorovi se stavljaju u određenu kolonu i red. Okno BorderPane postavlja čvorove u pet regiona, na vrhu, dole, levo, desno i u centar. Hbox stavlja decu u jedan horizonatalni red, a VBox ih postavlja vertikalno i jednu kolonu.
11. JavaFX obezbeđuje mnogo klasa oblika.