Računarska grafika

JavaFX – atributi

Boja (1)

- Atribut koji se primenjuje na linije i na popunjavanje
- Klasa `Paint` je osnovna klasa za
  - boje
  - prelaze (gradijente)
  - uzorke bojenja na osnovu slike
- Apstrakcija boje – klasa `Color` izvedena iz klase `Paint`
- Klase su u paketu `javafx.scene.paint`
- Objekat boje se stvara
  - konstruktorom: `Color(double c, double z, double p, double nep)`
  - statičkim metodima klase `Color`:
    `static Color color(double c, double z, double p)`
    `static Color color(double c, double z, double p, double nep)`

Boja (2)

- Parametri konstruktora i metoda
  - `c` - crvena (*red*), `z` - zelena (*green*) i `p` - plava (*blue*)
    - osnovne komponenate boje u (aditivnom) RGB sistemu boja
    - u opsegu od 0.0 do 1.0
      - 0.0 odsustvo komponente
      - 1.0 puno prisustvo date komponente boje
    - odsustvo sve tri komponente predstavlja crnu boju
    - puno prisustvo sve tri kompoenete predstavlja belu boju
  - `nep` - neprozirnost (eng. *opacity*), tzv. alfa-vrednost
    - u opsegu od 0.0 do 1.0
      - 0.0 potpuna prozirnost (*transparent*)
      - 1.0 potpuna neprozirnost (*opaque*)

Boja (3)

- U klasi `Color` – statička polja, reference na objekte boja:
  - `Color.BLACK`, `Color.WHITE`
  - `Color.RED`, `Color.GREEN`, `Color.BLUE`,
  - `Color.YELLOW`, `Color.CYAN`, `Color.MAGENTA`
- `Color.TRANSPARENT` označava potpunu prozirnost
- Podrazumevana boja
  - za linijske primitive – crna
  - za oivičenja – `null`
- Boja linije se postavlja metodom: `setStroke(Color boja)`
- Boja popunjavanja se postavlja metodom: `setFill(Color boja)`

Boja - primer

```
Line l1 = new Line(10,5,170,5);
Line l2 = new Line(10,15,170,15);
l2.setStroke(Color.BLUE);
Line l3 = new Line(10,25,170,25);
l3.setStroke(new Color(1,0,0,1));
Line l4 = new Line(10,35,170,35);
l4.setStroke(Color.color(0,1,0));
Line l5 = new Line(10,45,170,45);
l5.setStroke(Color.color(0,1,0,0.2));

Rectangle p1 = new Rectangle(10,50,40,40);
p1.setStroke(Color.RED);
Rectangle p2 = new Rectangle(60,50,40,40);
p2.setFill(Color.TRANSPARENT);
p2.setStroke(Color.GREEN);
Rectangle p3 = new Rectangle(110,50,40,40);
p3.setFill(Color.TRANSPARENT);
```

```
MoveTo pDo1 = new MoveTo(10,100);
LineTo lDo1 = new LineTo(90,110);
LineTo lDo2 = new LineTo(170,100);
MoveTo pDo2 = new MoveTo(170,120);
LineTo lDo3 = new LineTo(10,120);
Path p = new Path();
p.getElements().addAll(
   pDo1,lDo1,lDo2,pDo2,lDo3);
p.setStroke(Color.YELLOW);
```

Atributi linije

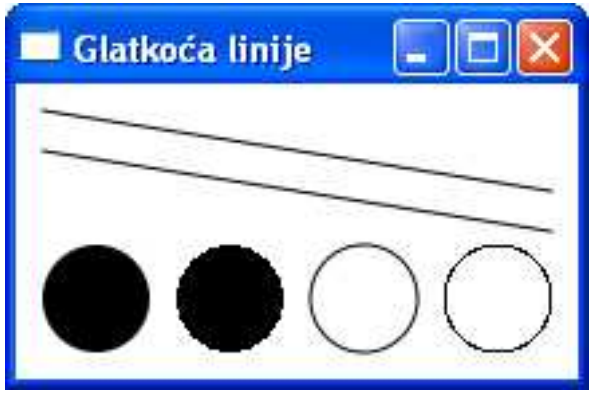
- Atributi linije se primenjuju na:
  - linijske primitive (linija, izlomljena linija)
  - oivičenja
- Atributi su:
  - boja
  - glatkost
  - debljina
  - tip kraja
  - stil isprekidanosti
  - tip oivičenja
  - tip spoja

Glatkost

- Glatkost (*smoothness*) je atribut, logičko svojstvo (*boolean property*) kako linijske primitive tako i ivice figure
- Rasterska priroda linija
  - na kosim linijama zapaža se efekat "nazupčenosti" (eng. *aliasing*)
  - struktura koju čine obojeni pikseli kroz koje prolazi matematički precizna prava linija je diskretna
- Algoritmi za uklanjanje efekta nazupčenosti (eng. *antialiasing*)
  - efekat se može vizuelno ublažiti i dobiti bolji utisak glatke linije
- Postavljanje glatkosti linije
  - metod (nasleđen iz klase `Shape`) `setSmooth(Boolean glatkost)`
  - vrednost `true` - vrši se neutralizacija efekta nazupčenosti
- Dohvatanje glatkosti linije
  - metod `isSmooth()`

Glatkost - primer

```
Line l1 = new Line(10,10,200,40);
l1.setSmooth(true);
Line l2 = new Line(10,25,200,55);
l2.setSmooth(false);
Circle k1 = new Circle(30,80,20);
Circle k2 = new Circle(80,80,20);
k2.setSmooth(false);
Circle k3 = new Circle(130,80,20);
k3.setFill(Color.TRANSPARENT);
k3.setStroke(Color.BLACK);
Circle k4 = new Circle(180,80,20);
k4.setFill(Color.TRANSPARENT);
k4.setStroke(Color.BLACK);
k4.setSmooth(false);
```



Debljina linije

- Debljina linije je realan broj i može biti 0.0 i veća
- Linija debljine 0.0 se naziva "debljinom dlake", (*hairline*)
  - linija se ne prikazuje
- Ako je debljina manja od 0.0, primenjuje se debljina 0.0
- Podrazumevana vrednost je 1.0
- Praktična razlika između 0.1, 0.5, 1.0 i 2.0
  - samo u nijansi (sive), sve su jednake debljine
- Debljina se postavlja metodom
  `setStrokeWidth(double debljina)`

Debljina linije – primer

```
//Horizontalne linije
Line l1 = new Line(10,5,180,5);
l1.setStrokeWidth(0.1);
// l2-l7: debljine 0.5, 1, 1.5, 2, 3, 4

//Kose linije
Line l8 = new Line(10,75,180,95);
l8.setStrokeWidth(0.1);
// l9-l4: debljine 0.5, 1, 1.5, 2, 3, 4
```

Tip kraja linije (1)

- Tipovi kraja linije su konstante tipa nabrajanja `StrokeLineCap`
- Tipovi kraja:
  - striktno odsečen u krajnjoj tački linije `StrokeLineCap.BUTT`
  - zaobljeno produžen od krajnje tačke `StrokeLineCap.ROUND`
  - kvadratno produžen `StrokeLineCap.SQUARE`
- Produžetak je jednak polovini širine linije

BUTT ROUND SQUARE

Tip kraja linije (2)

- Tip kraja linije se takođe primenjuje
  - na krajeve putanja
  - na segmente isprekidanih linija
- Podrazumevani tip: `StrokeLineCap.SQUARE`
- Postavljanje: `setStrokeLineCap(StrokeLineCap tip)`
- Primer:

1. Linija: BUTT
2. Linija: ROUND
3. Linija: SQUARE
4. 3 para (H+V) linija: BUTT, ROUND, SQUARE
5. Putanja: ROUND

Stil isprekidanosti (1)

- Podešavanje dva elementa:
  - uzorak (*pattern*) – koji definiše dužinu crtica i razmaka na liniji
  - pomeraj (*offset*) u tom uzorku od kojeg počinje uzorak da se primenjuje
- Uzorak se zadaje na sledeći način:
  - metodom `getStrokeDashArray()` se najpre dohvati objekat uzorka
  - zatim se metodom `addAll()` posledi niz brojeva koji predstavljaju dužine crtica, odnosno razmaka u pikselima
- Na primer, za liniju: `Line l`
  - uzorak tačkaste linije sa tačkicama i razmacima po 2 piksela:

```
l.getStrokeDashArray().addAll(2);
```

  - uzorak sa naizmeničnim crticama od 25 i 5 i razmacima po 20 piksela:

```
l.getStrokeDashArray().addAll(25, 20, 5, 20);
```

Stil isprekidanosti (2)

- Uzorak se primenjuje uz ciklično ponavljanje
- Prazan niz brojeva u uzorku – puna linija
- Neparan broj brojeva u uzorku – ponovljen zadati niz brojeva

- Pomeraj u uzorku
  - redni broj piksela uzorka od kojeg se kreće pri iscrtavanju linije
- Pomeraj se zadaje metodom:

```
setStrokeDashOffset(double pomeraj)
```

Stil isprekidanosti (3)

- Linija crtana uz definisani uzorak [25, 20, 5, 20]
  - najpre bez pomeraja
  - zatim sa pomerajem od 45 (iscrtavanje počinje od kraće crtice)

- Mehanizam: olovkom upravlja bitska maska definisana uzorkom
  - u uzorku crtica – u masci su jedinice i olovka je "spuštena na papir"
  - u uzorku razmak – u masci su nule i olovka je podignuta

Stil isprekidanosti (4)

- Svaka crtica počinje i završava odgovarajućim stilom kraja linije
- Primer – zaobljen popunjen pravougaonik, isprekidana ivica:
  - uzorak: [25, 20, 5, 20]
  - krajevi: BUTT, SQUARE, ROUND

1. Linija: uzorak [5], BUTT
2. Linija: uzorak [25,20,5,20], ROUND
3. Linija: uzorak [25,20,5,20], pomeraj 10, ROUND
4. Prvougaonik: uzorak [25,20,5,20], ROUND
5. Putanja: uzorak [25,20,5,20], ROUND

Tip oivičenja

- Tipovi oivičenja su konstante tipa nabrajanja `StrokeType`
  - određuju relativan odnos crtane ivice u odnosu na ivicu geometrijskog oblika
- Crtanje oivičenja može biti
  - sa unutrašnje strane geometrisjkog oblika `StrokeType.INSIDE`
  - centrirano po ivici geometrijskog oblika `StrokeType.CENTERED`
  - sa spoljne strane geometrijskog oblika `StrokeType.OUTSIDE`

bez oivičenja `INSIDE CENTERED OUTSIDE`

- Podrazumevani tip: `StrokeType.CENTERED`
- Postavlja se metodom: `setStrokeType(StrokeType tip)`

Tip spoja segmenata putanje

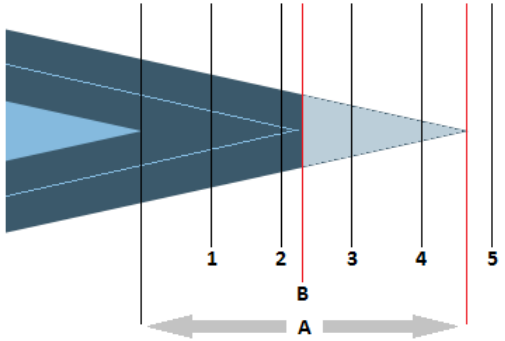
- Tipovi spojeva su konstante tipa nabrajanja `StrokeLineJoin`
- Spoj može biti:
  - sa špicom `StrokeLineJoin.MITER`
  - sa odsečenim vrhom `StrokeLineJoin.BEVEL`
  - sa zaobljenim vrhom `StrokeLineJoin.ROUND`

`MITER` `BEVEL` `ROUND`

- Podrazumevani tip: `StrokeLineJoin.MITER`
- Postavlja se metodom:
  `setStrokeLineJoin(StrokeLineJoin tip)`

Ograničenje špica

- Dužina špica A: rastojanje između
  - najisturenije i najuvučenije tačke spoja
- Ako je dužina špica A veća od navedenog ograničenja
  - špic se odseca ivicom geometrijskog oblika (u tački B)
- Za primer sa slike - vrednost ograničenja manja od A=4.65



- Podrazumevano ograničenje: 10.0
- Postavlja se metodom: `setStrokeMiterLimit(double limit)`

Atributi popunjavanja

- Atribut popunjavanja se primenjuje na zatvorene oblike
  - pravougaonike, mnogougle, krugove, elipse, lukove, krive, putanje
- Atribut je određen načinom popunjavanja
  - zadaje se metodom `setFill(Paint način)`
- Klasa `Paint` – osnovna klasa za načine popunjavanja
- Načini popunjavanja:
  - kontinualna boja
  - linearni prelaz
  - radijalni prelaz
  - uzorak popunjavanja

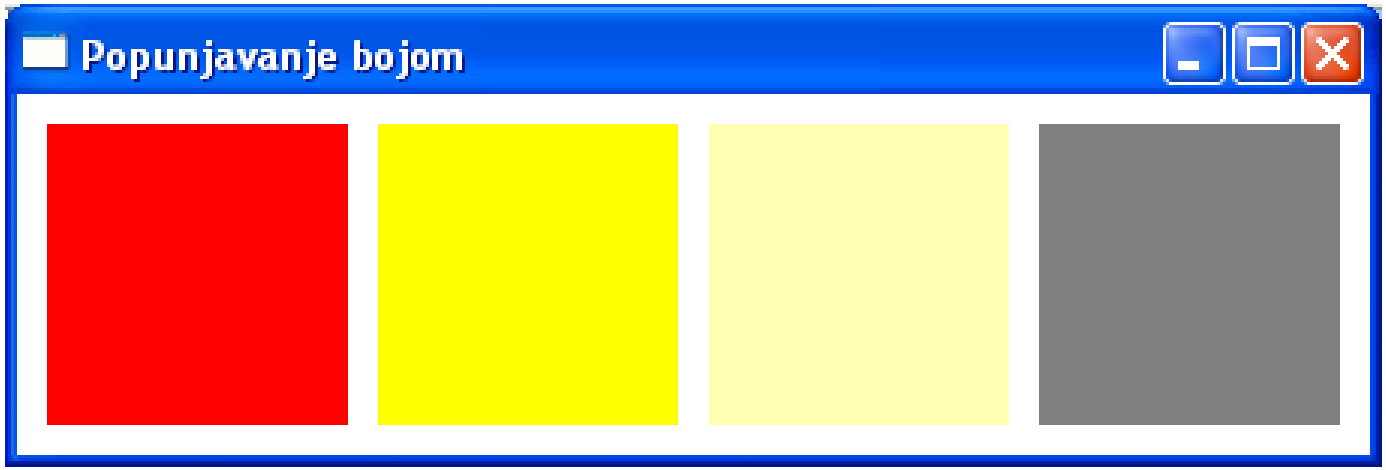
Popunjavanje kontinualnom bojom

- Prosleđuje se objekat tipa `Color` metodu `setFill()`
- Bibliotečki objekti – već formirane boje (npr. `Color.RED`)
- Više načina za stvaranje objekta boje:
  - konstruktorima sa parametrima – komponentama boje u RGB sistemu
    - (crvena, zelena, plava), realni (double) brojevi u opsegu 0.0-1.0
    - dodatnim 4. parametrom – neprozirnošću u opsegu 0.0-1.0
  - statatičkim metodima klase `Color`
    - svaki od metoda ima po dve varijante, sa 3 i 4 parametra
    - 4. parametar – neprozirnost
    - `Color.color()` – parametri su RGB komponente boje u opsegu 0.0- 1.0
    - `Color.rgb()` – parametri su RGB komponente, u opsegu 0-255
    - `Color.hsb()` – parametri su HSB komponente, u opsegu 0-255
    - `Color.web()` – parametar je niska – HTML/CSS (hex) zapis RGB boje
      - na primer, niska "0x0000FF" predstavlja čistu plavu boju.

Primer popunjavanja bojom (1)

```
Color boja1 = Color.RED;
Rectangle p1 = new Rectangle(10, 10, 100, 100);
p1.setFill(boja1);
Color boja2 = new Color(1, 1, 0, 1.0);
Rectangle p2 = new Rectangle(120, 10, 100, 100);
p2.setFill(boja2);
Color boja3 = Color.color(1, 1, 0, 0.3);
Rectangle p3 = new Rectangle(230, 10, 100, 100);
p3.setFill(boja3);
Color boja4 = Color.color(0.5, 0.5, 0.5);
Rectangle p4 = new Rectangle(340, 10, 100, 100);
p4.setFill(boja4);
```

Primer popunjavanja bojom (2)



Popunjavanje linearnim prelazom

- Prosleđuje se objekat tipa `LinearGradient` metodu `setFill()`
- Konstruktor klase `LinearGradient`:

```
public LinearGradient(double x1, double y1,
                      double x2, double y2, boolean relativno,
                      CycleMethod metod, Stop... stanice)
```

  - `x1`, `y1` – koordinate početne tačke pravca interpolacije
  - `x2`, `y2` – koordinate završne tačke pravca interpolacije
  - `relativno` – način računanja koordinata
    - relativno (`true`) – u opsegu 0-1, u okvirima geometrijskog oblika
    - apsolutno (`false`) – u koordinatnom sistemu oblika, u pikselima
  - `metod` – način ponavljanja interpoliranih nijansi boje
  - `stanice` – sekvenca tačaka na pravcu interpolacije sa bojama

Metod ponavljanja boje

- Metod ponavljanja boje je tip nabrajanja `CycleMethod`
  - određuje boju pre početne i posle završne tačke na pravcu interpolacije
- Vrednosti
  - `CycleMethod.NO_CYCLE` – boja se ne ponavlja
    - početna boja po pravcu interpolacije ispred početne tačke
    - završna boja po pravcu interpolacije posle završne tačke
  - `CycleMethod.REPEAT` – boja se ciklično ponavlja
    - prva tačka posle završne ima boju početne tačke
    - prva tačka pre početne boju završne tačke
  - `CycleMethod.REFLECT` – boja se naizmenično reflektovano ponavlja
    - pre početne tačke se ponavlja kao slika boje u ogledalu postavljenom u početnu tačku
    - posle završne tačke kao slika boje u ogledalu postavljenom u završnu tačku

Stanice

- Niz stanica definiše segmente interpolacije na pravcu interpolacije
- Svaki element niza je tipa `Stop` i sadrži
  - relativno rastojanje tačake stanice od početne tačke, mereno u opsegu od 0.0 do 1.0, na pravcu interpolacije
  - pridruženu boju

- Primer:
  - interpolacija od crvene do zelene na prvoj polovini puta po pravcu interpolacije, zatim interpolacija od zelene do plave

```
Stop[] tacke = new Stop[] { new Stop(0.0, Color.RED),
                            new Stop(0.5, Color.GREEN),
                            new Stop(1.0, Color.BLUE)
                          };
```

Primer linearnog prelaza (1)

```
Stop[] tacke = new Stop[] { new Stop(0, Color.BLACK),
                            new Stop(1, Color.RED)};
LinearGradient lg1 = new LinearGradient(10, 0, 110, 0,
                        false, CycleMethod.NO_CYCLE, tacke);
Rectangle p1 = new Rectangle(10, 10, 100, 100);  p1.setFill(lg1);
LinearGradient lg2 = new LinearGradient(0, 0, 1, 1,
                        true, CycleMethod.NO_CYCLE, tacke);
Rectangle p2 = new Rectangle(120, 10, 100, 100); p2.setFill(lg2);
LinearGradient lg3 = new LinearGradient(0, 0, 0.5, 0.5,
                        true, CycleMethod.REPEAT, tacke);
Rectangle p3 = new Rectangle(230, 10, 100, 100); p3.setFill(lg3);
LinearGradient lg4 = new LinearGradient(0, 0, 0.25, 0.25,
                        true, CycleMethod.REFLECT, tacke);
Rectangle p4 = new Rectangle(340, 10, 100, 100); p4.setFill(lg4);
```

Primer linearnog prelaza (2)

Popunjavanje radijalnim prelazom

- Prosleđuje se objekta tipa `RadialGradient` metodu `setFill()`
- Konstruktor `RadialGradient`:

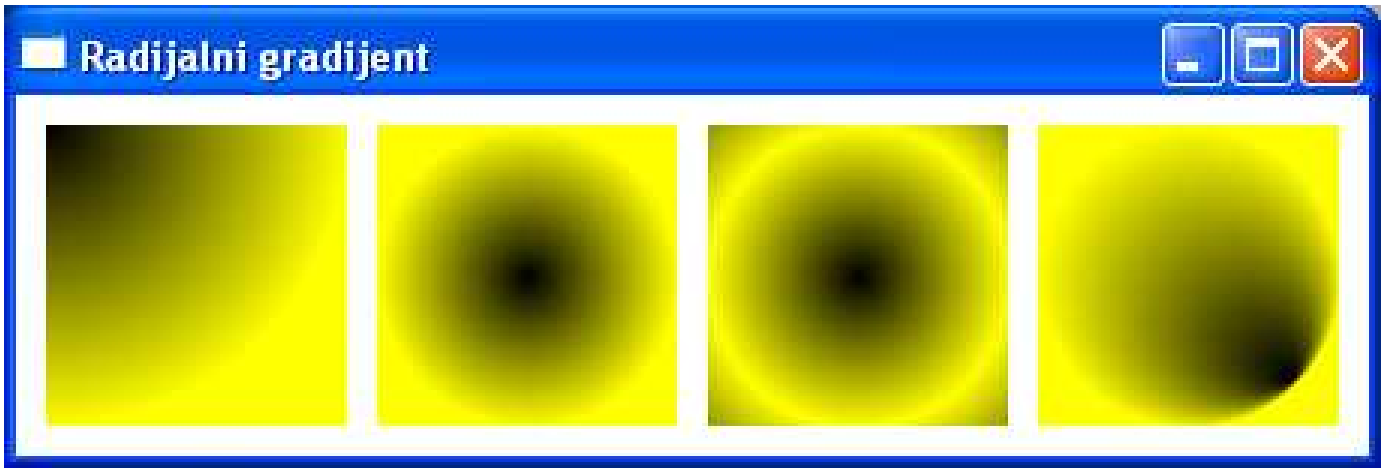
```
public RadialGradient(double uF, double dF,
  double x, double y, double r,
  boolean rel, CycleMethod metod, Stop... stanice)
```

  - `uF` i `dF` predstavljaju ugao i rastojanje tačke fokusa od centra kruga
    - tačka fokusa – početna tačka interpolacije, mora biti unutar kruga
    - ugao fokusa se meri u smeru kazaljke na časovniku, zadaje se u stepenima
    - rastojanje fokusa se meri od centra kruga, relativno u odnosu na poluprečnik
  - `x` i `y` predstavljaju položaj centra kruga
  - `r` poluprečnik kruga
  - ostali parametri su definisani kao kod linearnog prelaza
    - `rel` se ne odnosi na tačku fokusa,
      ona se uvek zadaje relativno u odnosu na krug

Primer radijalnog prelaza (1)

```
Stop[] tacke = new Stop[] { new Stop(0, Color.BLACK),
                            new Stop(1, Color.YELLOW)};
RadialGradient rg1 = new RadialGradient(0, 0, 0, 0, 1,
                         true, CycleMethod.NO_CYCLE, tacke);
Rectangle p1 = new Rectangle(10, 10, 100, 100); p1.setFill(rg1);
RadialGradient rg2 = new RadialGradient(0, 0, 0.5, 0.5, 0.5,
                                true, CycleMethod.NO_CYCLE, tacke);
Rectangle p2 = new Rectangle(120, 10, 100, 100); p2.setFill(rg2);
RadialGradient rg3 = new RadialGradient(0, 0, 0.5, 0.5, 0.5,
                                true, CycleMethod.REFLECT, tacke);
Rectangle p3 = new Rectangle(230, 10, 100, 100); p3.setFill(rg3);
RadialGradient rg4 = new RadialGradient(45, 1, 0.5, 0.5, 0.5,
                                true, CycleMethod.NO_CYCLE, tacke);
Rectangle p4 = new Rectangle(340, 10, 100, 100); p4.setFill(rg4);
```

Primer radijalnog prelaza (2)



Popunjavanje uzorkom slike

- Prosleđuje se objekat tipa `ImagePattern` metodu `setFill()`
- Konstruktor klase `ImagePattern`:

```
public ImagePattern(Image slika,double x,double y,
  double širina,double visina,boolean relativno)
```

  - `slika` – referenca na objekat slike koja se koristi za uzorak
  - `x` i `y` – koordinate za pozicioniranje gornjeg levog ugla slike
  - `širina` i `visina` – dimenzije pravougaonika u koji se uklapa slika
  - `relativno` – način računanja koordinata, širine i visine
- Ciklično ponavljanje uzorka u granicama geometrijskog oblika
- Stvaranje objekta slike na osnovu koje se formira uzorak
  - konstruktorom čiji je jedini parametar niska koja može predstavljati:
  - URL fajla slike
  - lokalnu putanju u sistemu fajlova

Primer uzorka slike (1)

```
Image sl1 = new Image("drvo.jpg");
ImagePattern us1 = new ImagePattern(sl1,0,0,1,1,true);
Rectangle p1 = new Rectangle(10,10,100,100);
p1.setFill(us1);
Image sl2 = new Image("zid.jpg");
ImagePattern us2 = new ImagePattern(sl2,120,10,50,50,false);
Rectangle p2 = new Rectangle(120,10,100,100);
p2.setFill(us2);
Image sl3 = new Image("lopta.png");
ImagePattern us3 = new ImagePattern(sl3,0.25,0.25,0.5,0.5,true);
Rectangle p3 = new Rectangle(230,10,100,50);
p3.setFill(us3);
ImagePattern us4 = new ImagePattern(sl3,340,10,50,50,false);
Rectangle p4 = new Rectangle(340,10,100,100);
p4.setFill(us4);
```

Primer uzorka slike (2)