Računarska grafika

JavaFX – 3D grafika:
scena i objekti

Uvod

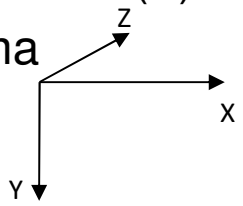
- 3D grafika uvodi treću dimenziju, tačka ima 3 koordinate
- Prikazana scena je realističnija od scene u 2D grafici
- Prikaz scene – prikazna ravan omeđena prikaznim prozorom
- JavaFX podržava 3D scenu sa 3D oblicima, svetlima i kamerom
- Na prikaznu ravan se projektuju objekti 3D scene realnog sveta
  - objekti → oblici (*shapes*) u JavaFX terminologiji
- Centar projekcije je tačka u kojoj se nalazi kamera
- Izgled scene određuje:
  - geometrija objekata, njihove pozicije i orijentacije
  - materijal objekata, način prikaza (žični ili model čvrstog tela, vidljivost površi)
  - vrsta, boja, pozicija svetala
  - pozicija, orijentacija i vidni ugao kamere
- U JavaFX grafici kamera i svetla su, takođe, čvorovi 3D scene

Posao JavaFX programera

- Geometrijsko modelovanje 3D objekata
  - JavaFX nudi samo 3 osnovna modela 3D tela: kvadar, valjak i lopta
  - postoji i proizvoljna trougaona mreža
- Definisanje osobina materijala površi objekata
  - boja, tekstura, reljef, refleksivnost, sopstveno osvetljenje
- Pozicioniranje i orijentisanje 3D objekata u 3D sceni
- Osvetljavanje objekata
  - izborom vrste i pozicioniranjem svetala u 3D sceni
- Prikaz objekata
  - izborom vrste i vidnog ugla, pozicioniranjem i orijentisanjem kamere

Koordinatni sistem

- Tačka u 3D koordinatnom sistemu realnog sveta:
  - ima tri koordinate (x, y, z).
  - koordinate su realni brojevi (`double`)
- U 3D grafici JavaFX koristi desni pravougli koordinatni sistem:
  - kada prsti desne pesnice prikazuju smer od prve ose (X) prema drugoj osi (Y), ispruženi palac pokazuje smer treće ose (Z)
  - ako se krene od levog prvouglog 2D koordinatnog sistema smer treće (Z) ose je od posmatrača



- 3D objekat u lokalnom koordinatnom sistemu ima tri dimenzije:
  - širinu (u x-pravcu), visinu (u y-pravcu) i dubinu (u z-pravcu).

3D scena

- U 3D sceni neki (neprozirni) objekti zaklanjaju druge
  - zaklanjanje može biti u potpunosti ili delimično
  - i stranice neprozirnih objekata zaklanjaju druge stranice istih objekata
- Kada se prikazuje 3D scena u ravni potrebno je odrediti vidljivost svakog piksela površi svakog neprozirnog objekta
- U JavaFX 2D grafici podrazumevani redosled iscrtavanja objekata:
  - redosled kojim su oni dodavani u graf scene
- U 3D grafici potrebno je sofističnije upravljanje redosledom crtanja objekata u sceni

Z-bafer (bafer dubine)

- Jednostavan mehanizam uklanjanja nevidljivih tačaka scene
  - hardverski implementiran u savremenim grafičkim kontrolerima
- Svakom pikselu u baferu ekrana odgovara jedna lokacija u z-baferu
- Z-bafer se inicijalizuje na najveću moguću Z vrednost u sceni
- Kada se crta piksel sa koordinatama ekrana (x,y):
  - računa se z-koordinata tačke na crtanom poligonu iz scene
  - provera: $z<z_{bafer}$ za (x,y) ➔ da li je tačka bliža posmatraču?
  - jeste: crta se piksel (x,y), izračunato z se upisuje u z-bafer;
    tačka u sceni je bliža posmatraču od ranije nacrtane tačke (x,y)
  - nije: ne crta se piksel, ne menja se z-bafer

Z-bafer u JavaFX sceni

- U JavaFX grafici, prilikom stvaranja scene:
  - zadaje se kao 4. argument koji određuje da li scena koristi z-bafer

```
boolean zBafer = true;
Scene scena = new Scene(korenGrafaScene,
                        širina, visina, zBafer);
```

- Bez `zBafer` argumenta stvara se scena bez z-bafera
- Z-bafer nije moguće naknadno dodeliti sceni
- Moguće je proveriti da li scena ima pridružen z-bafer metodom:

```
scena.isDepthBuffer()
```

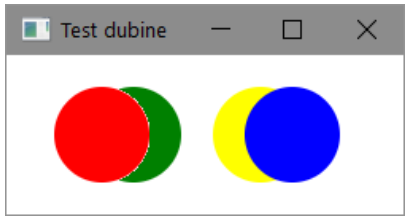
Postavljanje testa Z-bafera

- Da bi se vršila provera dubine (test z-bafera) prilikom crtanja čvora, potrebno je čvoru postaviti test dubine
- Metod: `setDepthTest(DepthTest testDubine)`
- Tip `javafx.scene.DepthTest` ima vrednosti:
  - `ENABLE` – vrednosti z-koordinate se uzimaju u obzir pri crtanju čvora
  - `DISABLE` – čvor se crta po redosledu dodavanja u graf scene
  - `INHERIT` – svojstvo `deptTest` čvora se nasleđuje od roditelja; ako nema roditelja, isto je značenje kao `ENABLE`
- Podrazumevana vrednost svojstva je `DepthTest.INHERIT`

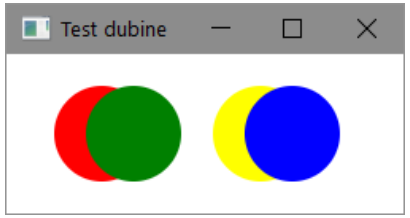
Primer

```
import javafx.scene.DepthTest;
//...
  Circle crveni = new Circle(60, 50, 30, Color.RED);
  Circle zeleni = new Circle(80, 50, 30, Color.GREEN);
  zeleni.setTranslateZ(10);
  Circle zuti = new Circle(160, 50, 30, Color.YELLOW);
  Circle plavi = new Circle(180, 50, 30, Color.BLUE);
  plavi.setTranslateZ(10);
  zuti.setDepthTest(DepthTest.DISABLE);
  plavi.setDepthTest(DepthTest.DISABLE);
  Group koren = new Group();
  koren.getChildren().addAll(crveni,zeleni, zuti, plavi);
  zBafer=true;
  Scene scena = new Scene(koren, 250, 100, zBafer);
```

zBafer=true;



zBafer=false;



Neutralizacija nazupčenosti

- Kod iscrtavanja 3D oblika kojima je uključen test dubine zapaža se efekat nazupčenosti (eng. *aliasing*)
- Rešavanje ovog problema čini iscrtavanje manje efikasnim, te se podrazumevano ne otklanja nazupčenost
- JavaFX podržava scenu sa balansiranim otklanjanjem nazupčenosti
  - ivice 3D oblika su više glatke, a efikasnost crtanja prihvatljiva
- Konstruktor scene sa parametrom koji određuje da li se primenjuje tehnika za otklanjanje nazupčenosti:

```
Scene(Parent koren, double šir, double vis,
    boolean z-bafer, SceneAntialiasing nenazupčen)
```

Vrednosti parametra nazupčenosti

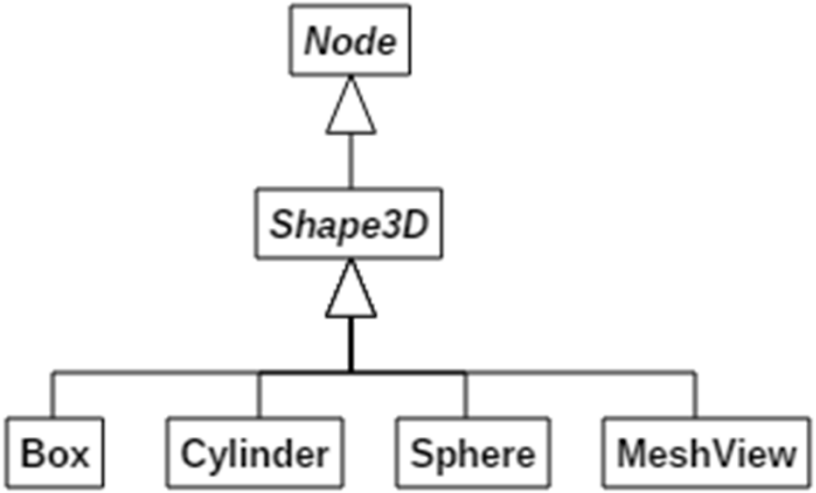
- Klasa:
  `javafx.scene.SceneAntialiasing`
- Dve imenovane konstante:
  - `DISABLED` – onemogućen mehanizam ukljanjanja nazupčenosti
  - `BALANCED` – optimizovan balans između kvaliteta i performansi
- Podrazumevana vrednost – `DISABLED`

3D oblici

- JavaFX podržava 3D oblike – objekte u sceni
- Postoji nekoliko već definisanih oblika:
  - kvadar (klasa `Box`)
  - valjak (klasa `Cylinder`)
  - lopta (klasa `Sphere`)
- Može da se definiše proizvoljna mreža trouglova (*triangular mesh*)
  - proizvoljan 3D oblik koji može biti zatvoren ili otvoren (samo površ)
  - klasa `MeshView` – apstrakcija proizvoljnog 3D oblika
- Nalaze se u paketu: `javafx.scene.shape`
- Centar 3D oblika je u koordinatnom početku njegovog lokalnog koordinatnog sistema

Hijerarhija klasa oblika

- Sve navedene klase su potklase apstraktne klase `Shape3D`
- Klasa `Shape3D` je izvedena iz apstraktne klase `Node`



Svojstva 3D oblika

- Svaki 3D objekt (oblik) je objekat potomaka klase `Shape3D` i ima:
  - geometriju – mrežu trouglova koji čine njegovu površ
  - svojstva
- Svojstva su:
  - način crtanja (*draw mode*),
    - opisan tipom nabrajanja `DrawModel`, koji definiše način crtanja trouglova mreže oblika: popunjeni trouglovi ili samo njihove ivice
  - izbor površi za sakrivanje (*face culling*)
    - opisan tipom nabrajanja `CullFace`
  - materijal
    - opisan klasom `Material`, koji (uz svetlo) određuje boju piksela

Kvadar

- Stvara se podrazumevanim i sledećim konstruktorom:

```
Box(double širina, double visina, double dubina)
```

- Centar u koordinatnom početku
- Ivice su paralelne osama koordinatnog sistema
- Dimenzije
  - širina u pravcu X-ose
  - visina u pravcu Y-ose
  - dubina u pravcu Z-ose
- Mrežu čine pravougaonici podeljeni dijagonalom
- Podrazumevane dimenzije su 2x2x2

Valjak

- Stvara se podrazumevanim i sledećim konstruktorima:

```
Cylinder(double polupr, double visina)
Cylinder(double polupr, double visina, int podela)
```

- Osnova je paralelna XoZ ravni, visina u pravcu Y-ose
- Centar valjka je u koordinatnom početku
- Podrazumevani poluprečnik je 1, a visina 2
- Podelom se dobija mreža trouglova koji se crtaju
  - broj podela je broj ravnomerno raspoređenih tačaka na osnovama, tako da omotač čine trouglovi čija su temena u datim tačkama osnova
  - broj odeljaka je najmanje 3, a podrazumevano je 15

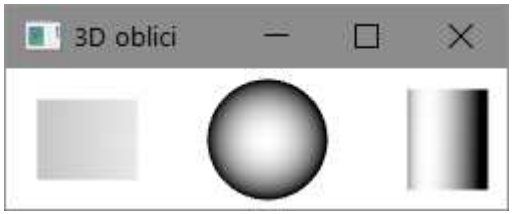
Lopta

- Stvara se podrazumevanim i sledećim konstruktorima:

```
Sphere(double poluprečnik)
Sphere(double poluprečnik, int podela)
```

- Podrazumevani poluprečnik je 1
- Aproksimira se trougaonom mrežom, broj podela određuje finoću
  - veći broj – više trouglova u mreži, pa je oblik bliži geometrijskoj lopti
  - najmanje 1, podrazumevano 24
- Podele su po koord. osama ravnima koje su normalne na osu
  - presečne tačke dobijenih kružnica su temena trouglova u mreži
- Broj podela 1 (po svakoj od osa) presecanje koordinatnim ravnima
  - 6 presečnih tačaka → dve četvorostrane piramide spojene osnovama

Primer 3D oblika




```
Box kvadar = new Box(50, 40, 10);
kvadar.setTranslateX(40);  kvadar.setTranslateY(35);
kvadar.setTranslateZ(10);
Cylinder valjak = new Cylinder(20,50);
valjak.setTranslateX(220);  valjak.setTranslateY(35);
valjak.setTranslateZ(30);
Sphere lopta = new Sphere(30);
lopta.setTranslateX(130);  lopta.setTranslateY(35);
lopta.setTranslateZ(20);
Group koren = new Group();
koren.getChildren().addAll(kvadar, lopta, valjak);
SceneAntialiasing nazupčenost = SceneAntialiasing.BALANCED;
Scene scena = new Scene(koren, 250, 70, true, nazupčenost);
```

Način crtanja mreže trouglova

- Način crtanja mreže trouglova:
  - popunjeni trouglovi (model čvrstog tela, *solid model*)
  - samo ivice trouglova (žični model, *wireframe model*)
- Tip nabrajanja `javafx.scene.shape.DrawMode` – konstante:
  - `LINE` (ivice trouglova, žični model)
  - `FILL` (popunjeni trouglova, model čvrstog tela)
- Način crtanja je svojstvo oblika koje se postavlja metodom:

```
setDrawMode(DrawMode model)
```

- Podrazumevani način crtanja: `FILL`

Vidljive/sakrivene površi

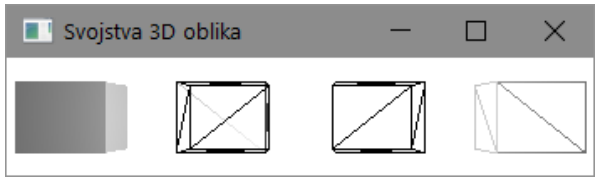
- Efikasnost crtanja oblika
  - određena je brojem trouglova u mreži kojom se predstavlja oblik
- Trouglove okrenute naličjem nema smisla crtati za neprozirno telo
- Izbor trouglova mreže koji se neće crtati (*face culling*)
  - svojstvo oblika koje se postavlja metodom:

`setCullFace(CullFace sakiveno)`

- Tip nabrajanja `javafx.scene.CullFace` definiše konstante:
  - `BACK` – neće se crtati trouglovi koji su okrenuti naličjem prema kameri
  - `FRONT` – neće se crtati trouglovi koji su okrenuti licem prema kameri
  - `NONE` – svi se crtaju
- Podrazumevano sakrivanje površi: `BACK`

Primer svojstava 3D oblika

```
Box kvadar1 = new Box(50, 40, 10);
kvadar1.setDrawMode(DrawMode.FILL); //podrazumevano
kvadar1.setTranslateX(40); kvadar1.setTranslateY(35);
Box kvadar2 = new Box(50, 40, 10);;
kvadar2.setDrawMode(DrawMode.LINE);
kvadar2.setCullFace(CullFace.NONE);
kvadar2.setTranslateX(130); kvadar2.setTranslateY(35);
Box kvadar3 = new Box(50, 40, 10);
kvadar3.setDrawMode(DrawMode.LINE);
kvadar3.setCullFace(CullFace.FRONT);
kvadar3.setTranslateX(220); kvadar3.setTranslateY(35);
Box kvadar4 = new Box(50, 40, 10);
kvadar4.setDrawMode(DrawMode.LINE);
kvadar4.setCullFace(CullFace.BACK); //podrazumevano
kvadar4.setTranslateX(310); kvadar4.setTranslateY(35);
```



Podscena

- U 3D sceni može postojati
  - proizvoljan broj čvorova 3D objekata i izvora svetala
  - samo jedna kamera
- Ukoliko se želi više kamera (više pogleda na scenu)
  - potrebno je uvesti podscene
  - svakoj podsceni pridružiti njenu kameru
- Podscena je, kao i scena, kontejner grafa scene
- Opisana je klasom `SubScene`
  - izvedenom iz klase `Node`
  - definisanom u paketu `javafx.scene`

Parametri podscene

- Podscena ima svojstva i atribute kao i scena:
  - graf scene
  - visinu i širinu
  - boju pozadine
  - z-bafer
  - glatkoću ivica objekata (*antialiasing*)
  - vlastitu kameru
- Konstruktor podscene ima iste parametre kao i konstruktor scene
- Ako se ne postavi kamera podscene – podrazumevana paralelna
- Ako postoje 3D objekti u podsceni – podrazumevano svetlo
  - tačkasti izvor belog svetla na kameri (*headlight*)

Korišćenje podscene

- `SubScene` je izvedena iz `Node`
  - primerci podscene mogu da se koriste kao čvorovi u sceni
- Graf scene može sadržati više podscena
  - podscene se kombinuju u jednu scenu
- Kompozicija scene – i graf podscene može sadržati podscene
- Podscene se mogu koristiti da bi se kombinovala 2D i 3D grafika
  - neke podscene mogu biti 3D, dok druge mogu biti 2D
- Podscene imaju podrazumevano transparentnu pozadinu
  - moguće je vizuelno preklopiti njihov prikaz i dobiti jedinstvenu sliku

Primer podscene (1)

- Dve podscene od kojih se obrazuje jedna scena
- U prvoj podsceni je kvadar:
  - prikazan perspektivnom kamerom
    - rotiranom oko X ose za -30°
  - osvetljen odozgo svetlom boje CYAN
- U drugoj podsceni je lopta:
  - prikazana podrazumevanom paralelnom kamerom
  - osvetljena sleva svetlom boje MAGENTA

Primer podscene (2)

```
...
import javafx.scene.Scene;
import javafx.scene.SubScene;
import javafx.scene.SceneAntialiasing;
import javafx.scene.Group;
public class Podscena extends Application {
  @Override public void start(Stage prozor) {
    // definisanje i pozicioniranje objekata kvadar i lopta
    // definisanje i poz. svetala svetloKva i svetloLop
    // definisanje, poz. i orjent. kamera kamKva i kamLop
```

Primer podscene (3)

```
Group grKva = new Group(kvadar, svetloKva);
SceneAntialiasing g=SceneAntialiasing.BALANCED;
SubScene scenaKva = new SubScene(grKva,230,70,true,g);
scenaKva.setCamera(kamKva);
Group grLop = new Group(lopta, svetloLop);
SubScene scenaLop = new SubScene(grLop,230,70,true,g);
scenaLop.setCamera(kamLop);
Group koren = new Group();
koren.getChildren().addAll(scenaKva,scenaLop);
Scene scena = new Scene(koren, 230, 70, true, glatko);
...
```