# 3D grafika, materijali, tranformacije, projekcije, izvori svetlosti

Skripta objašnjenja primera za gradivo 9. nedelje:

- **3D grafika**,
- **materijali,**
- **transformacije,**
- **projekcije i izvori svetlosti**

U Visual Studio projektu Primer_2.2, prvo će biti objašnjeni sadržaji prozora **Basics.xaml** (prikazan na **slici 1**) i **CullingExample.xaml** (prikazan na **slici 2**).



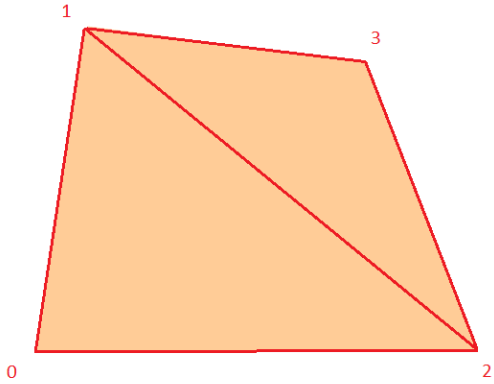
**Slika 1 – Sadržaj prozora Basics.xaml**



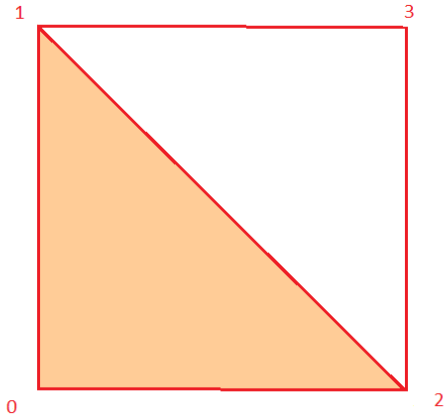
**Slika 2 – Sadržaj prozora CullingExample.xaml**

Na oba prozora je prikazan po jedan *mesh* model, sastavljen od po dva trougla. *Mesh* model se iscrtava kao objekat klase ***MeshGeometry3D***, kome se dodeljuju koordinate tačaka i, naspram indeksa navedenih tačaka, definišu trouglovi iz kojih će se sastojati model. Dodeljivanje tačaka se vrši u *property*-u ***Positions***. Kako se radi o trodimenzionom iscrtavanju, potrebno je za svaku tačku navesti X, Y i Z koordinate. Pojedinačne koordinate tačke se razdvajaju zarezima, dok se tačke razdvajaju razmakom.

*Property* ***TriangleIndices*** određuje trouglove *mesh* modela, potrebno je navesti indekse tačaka iz ***Posiotions*** *property-a*. Ovde je bitno obratiti pažnju na redosled navođenja tačaka trougla. Ukoliko se tačke navedu u smeru suprotnom od smera kazaljke na satu, trougao će se iscrtati licem prema kameri, obrnutno će se iscrtati naličjem prema kameri.




**Slika 3 – Trouglovi *mesh* modela u Basic.xaml prozoru**




**Slika 4 – Trouglovi mesh modela u CullingExample.xaml prozoru**

Na **slici 3** su tačke oba trougla navede u smeru suprotnom od smera kazaljke na satu:

- trougao 1 – 0, 2, 1 (lice),
- trougao 2 – 1, 2, 3 (lice).

Na **slici 4** je drugačija situacija:

- trougao 1 – 0, 2, 1 (lice),
- trougao 2 – 1, 3, 2 (naličje).

Kako je iscrtano naličje drugog trougla na **slici 4**, on se ne prikazuje na prozoru.

Objekat klase ***MeshGeometry3D*** se uvek mora inicijalizovati unutar ***GeometryModel3D*** klase, gde je moguće, pored same geometrije, podesiti i materijal iscrtanog modela. Materijal se opisuje preko svojih refleksivnih karakteristika. Materijal se opisuje preko tri svetlosne komponente: difuzne, emisione i spekularne.

**Difuzno svetlo** osvetljava objekte iz nekog konkretnog pravca. Svetlosni zraci se prelamaju i rasipaju na površini objekta. Ova komponenta svetla najintenzivnije obasjava poligone čija je normala uperena u poziciju svetla.

**Emisiona komponenta** predstavlja boju koju materijal isijava.

**Spekularno svetlo**, slično kao i difuzno, dolazi iz nekog pravca, ali je refleksija pod mnogo oštrijim uglom i nema raspisanja. Ova svetlost kreira veoma jako osvetljenje površine na objektu – odsjaj.

Na **listingu 1** je prikazan primer definisanja geometrije i modela *mesh* modela. Materijal se definiše dinisanjem boja pojedinačnih svetlosnih komponenti materijala.

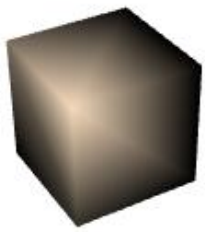
```
<GeometryModel3D>
    <GeometryModel3D.Geometry>
        <MeshGeometry3D Positions = "0,0,0 0,8,0 8,0,0 8,8,0" TriangleIndices = "0,2,1 1,3,2"/>
    </GeometryModel3D.Geometry>

    <GeometryModel3D.Material>
        <DiffuseMaterial Brush = "Bisque" />
    </GeometryModel3D.Material>

</GeometryModel3D>
```

**Listing 1 – XAML definicija *mesh* modela**

Na **slici 5** je prikazan sadržaj prozora **3Dgraphics.xaml**.



**Slika 5 - Sadržaj prozora 3Dgraphics.xaml**

Na prozoru je prikazana kocka napravljena korišćenjem ***MeshGeometry3D*** klase. U *property*-u ***Positions*** je definisano osam tačaka koje predstavljaju temena kocke, dok je u *property*-u ***TriangleIndices*** definisano dvanaest trouglova, po dva za svaku stranicu kocke. Za materijal kocke je podešena difuzna svetlosna komponenta korišćenjem objekta klase ***SolidColorBrush***, sa bojom postavljenom na vrednost *Bisque*.

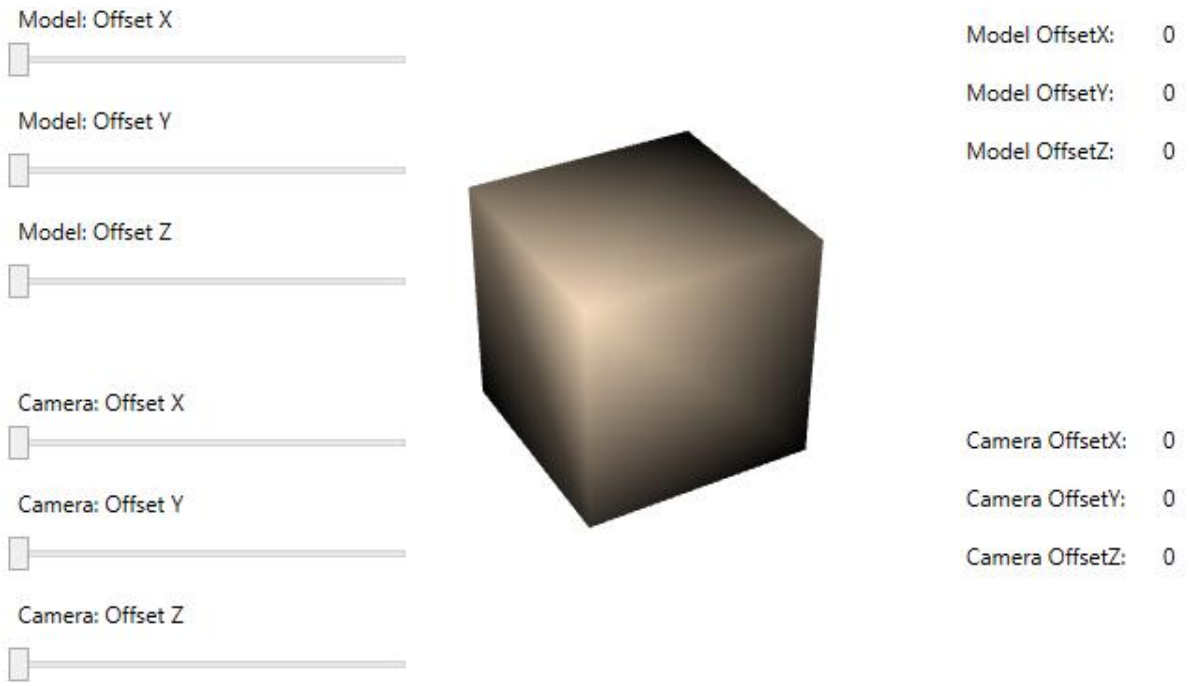
Analogno 2D grafici, **3D transfomacije** služe za rotiranje, skaliranje i transliranje objekata, odnosno *mesh* modela. Ovde je bitno napomenuti da u 3D grafici ne postoji opcija transformacije zakrivljivanja, budući da se ona uglavnom koristi za simulaciju 3D dubine u 2D modelima.

Primer korišćenja transformacija je prikazan u prozoru **3Dtransformacije.xaml**.

Transformacija rotacije se modeluje pomoću klase ***RotateTransform3D***. Potrebno je definisati ugao pod kojim će se objekat rotirati i osu oko koje će se se vršiti rotacija. Ugao se definiše pomoću *property*-a ***Angle***, dok se osa rotacije definiše uz pomoć ***Axis*** *property*-a.

Transformacija skaliranja se modeluje pomoću klase ***ScaleTransform3D***. Analogno skaliranju u 2D grafici, očekuje povećanja po osama, u ovom slučaju X, Y i Z. Vrednosti koje se tu prosleđuju odgovaraju vrednostima koliko puta treba uvećati/smanjiti objekat po toj osi. Podešavanje se vrši pomoću ***ScaleX, ScaleY*** i ***ScaleZ***. Ukoliko je potrebno da se objekat uveća 2 puta po X osi, za vrednost ***ScaleX*** će se postaviti 2, a ako je potrebno smanjiti objekat na polovinu trenutne veličine po istoj osi, dodeliće se vrednost 0.5.

Transformacija translacije (pomeranja) je prikazana u prozoru **Positioning.xaml**, koji je prikazan na **slici 6**.




**Slika 6 - Sadržaj prozora Positioning.xaml**

Na prozoru je prikazana kocka modelovana identično objektu sa prethodnog primera. Sa leve strane se nalazi 6 *Slider*-a, 3 za podešavanje pomeraja (translacije) samog objekta, dok su ostala 3 zadužena za podešavanje kamere, o kojoj će biti reči kasnije u tekstu.

Transformacija translacije se modeluje pomoću klase ***TranslateTransform3D***. Očekuje se da joj se zadaju pomeraji po svakoj osi (X, Y i Z) putem *property*-a, respektivno, ***OffsetX***, ***OffsetY*** i ***OffsetZ***. Dodeljivanjem pozitivne vrednosti se objekat pomera u pozitivnom smeru zadate ose, dok se zadavanjem negativne vrednosti objekat pomera u negativnom smeru iste ose (pravilo tri prsta desne šake).

Na prozoru je prikazano kako se koristi transformacija translacije, gde se na promene vrednosti slajdera reaguje događajima, i zavisno od izabrane vrednosti primenjuje data ransformacija. Moguće je primeniti više transformacija nad istim objektom kada se sve transformacije stave u grupu koja se dodeljuje konkretnom UI elementu.

Na **Listingu 2** je prikazana definicija translacije kocke na **Slici 6**. Početne vrednosti pomeraja kocke po sve tri ose su podešene na vrednost 0.

```
<ModelVisual3D.Transform>
    <Transform3DGroup>
        <TranslateTransform3D x:Name="translate" OffsetX="0" OffsetY="0" OffsetZ="0"/>
    </Transform3DGroup>
</ModelVisual3D.Transform>
```

**Listing 2 - XAML definicija transformacije**

Na **Listingu 3** je prikazano povezivanje trenutne vrednosti pojedinačnih *Slider*-a sa odgovarajućim osama. Korisničkim pomeranjem *Slider*-a se njegova vrednost dodeljuje pomeraju po odgovarajućoj osi i tranformacija se automatski izvršava.

```
<Slider Height = "23" HorizontalAlignment = "Left"
    Margin = "10,36,0,0" Name = "slider1"
    VerticalAlignment = "Top" Width = "217"
    Maximum = "3"
    Value = "{Binding ElementName = translate, Path=OffsetX}" />
<Slider Height = "23" HorizontalAlignment = "Left"
    Margin = "10,95,0,0" x:Name = "slider1_Copy"
    VerticalAlignment = "Top" Width = "217"
    Maximum = "3"
    Value = "{Binding OffsetY, ElementName=translate}" />
<Slider Height = "23" HorizontalAlignment = "Left"
    Margin = "10,154,0,0" x:Name = "slider1_Copy1"
    VerticalAlignment = "Top" Width = "217"
    Maximum = "3"
    Value = "{Binding OffsetZ, ElementName=translate}" />
```

**Listing 3 - povezivanje *Slider*-a sa translacijom**

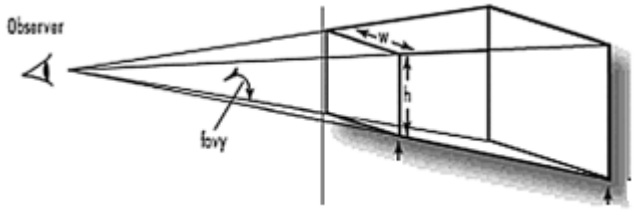
Kamera i projekcije će biti objašnjene na prozoru **Camera.xaml**. Da bi se objasnili rad sa kamerom i vrste projekcija, potrebno je uvesti pojam **Viewport.** Definisanjem *Viewport*-a se u stvari definiše kako će se skalirati prozor pri promeni veličine prozora od strane korisnika. Tada se u stvari vrši mapiranje logičkih koordinata u fizičke koordinate, odnosno mapiranje kliping (*clipping*) prozor u fizički prozor aplikacije. Podešavanje *Viewport*-a vrši se uz pomoć objekta klase ***Viewport3D***. Potrebno je podesiti mu dimenzije (*property* ***widht, height***), potrebno je i staviti vrednost ***ClipToBounds*** *property*-a na *True*. Ovim se određuje mapiranje kliping prozora na celokupan prozor aplikacije.

Nakon definisanja *Viewport*-a, potrebno je definisati kameru koja će taj prozor prikazivati. Ovde su ponuđena dva tipa kamera, kamera sa ortogonalnom projekcijom (***OrthographicCamera***) i kamera sa perspektivnom projekcijom (***PerspectiveCamera***).

Kod **kamere sa ortogonalnom projekcijom** je bitno napomenuti da se svi objekti koji su istih dimenzija prikazuju u istoj veličini, bez obzira na to gde se nalaze. Ova projekcija se najčešće koristi u CAD i arhitektonskom dizajnu. Prilikom podešavanja ove kamere potrebno je podesiti poziciju kamere zadavanjem koordinata (*property* ***Position***), smer u kome je kamera okrenuta se podesava zadavanjem 3D vektora (property ***LookDirection***). Pored pozicije i smera gledanja kamere, potrebno je odrediti i minimalnu i maksimalnu razdaljinu od kamere koju ona snima. Odnosno, potrebno je

odrediti tačku u prostoru smera gledanja od koje počinje snimanje i tačku u kojoj ovo snimanje prestaje (*property* ***NearPlaneDistance*** i *property* ***FarPlaneDistance***).

Kod **kamere sa perspektivnom projekcijom** svi objekti koji su dalji od posmatrača se prikazuju manjim, dok se svi objekti bliži posmatraču prikazuju većim. Ova projekcija omogućava stvaranje realističnog utiska sveta. Kao i kod ortogonalne kamere, i perspektivna kamera zahteva podešavanje **Position** i **LookDirection** *property*-a. Pored toga, potrebno je još odrediti ugao posmatranja kamere (**Slika 7**), odnosno dubinu vidnog polja (eng. *field of view*).

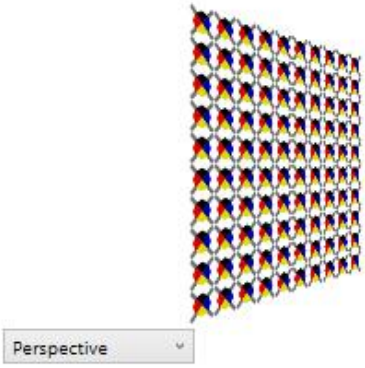



**Slika 7 - Perspektivna projekcija**

Za ugao posmatranja kamere je zadužen *property* ***FieldOfView***. Ugao se zapisuje u stepenima, dok je predefinisana vrednost ugla 45 stepeni.

Na **slici 8** i **slici 9**, su prikazani primeri ortognonalne projekcije i projekcije perspektive koji su prikazani na prozoru **Camera.xaml**.


**Slika 8 - Ortogonalna projekcija**




**Slika 9 - Perspektiva**

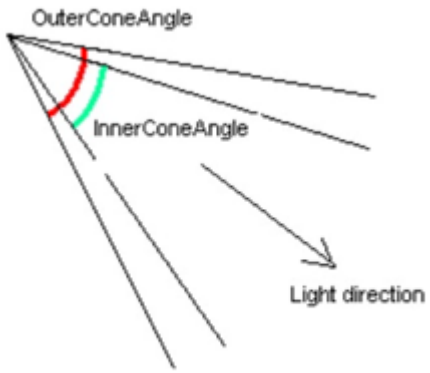
Pri otvaranju prozora postavljena je kamera sa projekcijom perspektive, podešenom u *XAML*-u. Odabirom jedne od dve opcije u *ComboBox*-u poziva se *Listener* koji trenutnu kameru uništava i zatim definiše novi, odabrani tip kamere.

Svetlosni izvori će biti objasnjeni na primeru prozora **LightDifferences.xaml**. Osvetljenje je esencijalno u postizanju realizma modelovanje scene. Takođe, svojstva materijala dolaze do izražaja tek kada je u sveni prisutno osvetljenje. Svetlosni izvori se dele u dve velike grupe: **pozicioni** i **direkcioni**.

**Direkcioni svetlosni izvor** podrazumeva postojanje paralelnih zraka i predstavlja svetlost koja je veoma udaljena i kod koje nema slabljenja inteziteta. Ovaj tip svetlostnog izvora simula sunčeve zrake. U WPF grafici se realizuje uz pomoć ***DirectionalLight*** klase, unutar koje je potrebno definisati smer osvetljaja (3D vektor) putem *property*-a ***Direction*** i boju svetla (*property* ***Color***). Za razliku od ostalih tipova svetlosnih izvora, direkcioni svetlosni izvor nema poziciju u prostoru.

Pozicioni izvori svetlosti se dalje dele na **reflektorske** i **tačkaste**.

**Reflektorski svetlosni izvor** podrazumeva postojanje snopa zraka svetlosti. Za ovaj tip izvora moguće je definisati dve vrste slabljenja. Prvo predstavlja slabljenje koje zavisi od udaljenosti od objekata, a drugo podrazumeva slabljenje od središta snopa ka njegovom obodu. Ova svetlosti izvor je moguće opisati primerom automobilskog fara (**Slika 10**).




**Slika 10 - Reflektorsko osvetljenje**

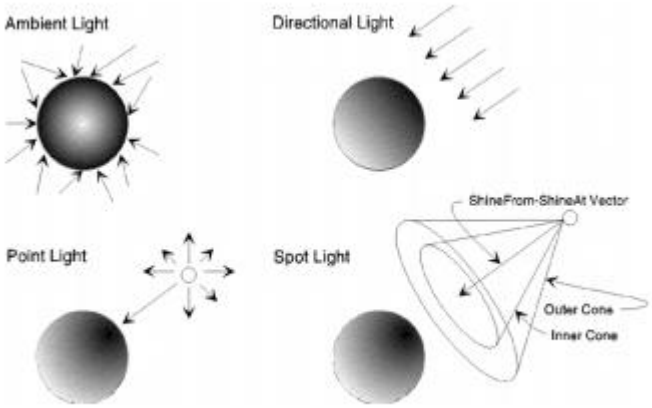
Za realizaciju ovog svetlosnog izvora koristi se klasa ***SpotLight***. Analogno direkcionom izvoru svetlosti, i ovde je potrebno odrediti smer osvetljenja i boju svetlosti. Pozicija reflektorskog izvora svetlosti podešava se uz pomoć *property*-a ***Position***, u koji je potrebno upisati koordinate X, Y i Z pozicije na koju se želi postaviti izvor svetlosti. Omogućeno je i podešavanje *property*-a ***InnerConeAngle*** i ***OuterConeAngle***. Unutrašnji ugao snopa svetlosti predstavlja onaj ugao pod kojim je svetlo najjače, odnosno nema rasipanja. Rasipanje svetlosti započinje kada se iz unutrašnjeg ugla snopa prelazi u spoljašnji ugao snopa svetlosti.

**Tačkasti izvor svetlosti** karakterišepostojanje svetlosnih zraka, koji se rasejavaju na sve strane. Primer za njega jeste sijalica, koja nije zaklonjena preprekom i rasejava svetlosne zrake na sve strane. Za njega je moguće definisati način slabljenja u zavisnosti od udaljenosti objekata. I za tačkasti i za reflektorski izvor svetlosti je moguće odrediti razdaljinu od svetlosnog izvora nakon kojeg on više nema uticaja. Odnosno, nakon te distance više nema osvetljenja. I kod ovog izvora je potrebno podesiti boju i poziciju. Podešavanje slabljenja u odnosu na udaljenost objekata može se odrediti pomoći sledećih *property*-a:

- ***ConstantAttenuation*** – zadavanjem konstante udaljenosti nakon koje vise nema osvetljenja,
- ***LinearAttenuation*** – zadavanjem vrednosti koja određuje linearno umanjenje inteziteta svetlosti naspram udaljenosti,
- ***QuadraticAttenuation*** – zadavanjem vrednosti koja određuje kvadratno umanjenje inteziteta svetlosti naspram udaljenosti.

Pored navedenih tipova svetlosnih izvora, moguće je navesti i globano **ambijentalno osvetljenje**. Ovaj tip osvetljenja služi za aproksimaciju efekta svetlosti u celoj sceni, koje se raspršuje reflektovanjem od višestrukih difuznih površina. Zraci ambijentalnog osvetljenja obuhvataju sve površine iz svih pravaca, zbog čega ne postoji senčenje. Potrebno je podešavanje samo boje svetla.

**Slika 11** predstavlja ilustraciju navedenih tipova svetlosnih izvora.




**Slika 11 - Tipovi svetlosnih izvora**