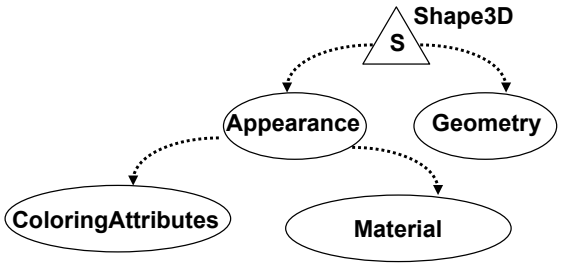
Java 3D

Iluminação

Cor

- A coloração de um objecto pode ser obtida de várias formas:
  - Appearance
    - ColoringAttributes: cor fixa e independente da iluminação.
    - Material: cor obtida por cálculo de iluminação; a cor final depende das fontes de luz adicionadas à cena.
  - Geometry Color
    - Cor definida em cada vértice juntamente com a geometria.



Prevalência

Cor obtida quando existe iluminação e o objecto tem um material definido:

Cor por vértice (geometria)	ColoringAttributes	Resultado
N	N	Material color
S	N	Geometry color
N	S	Material color
S	S	Geometry color

Cor obtida na ausência de iluminação (material não está definido):

Cor por vértice (geometria)	ColoringAttributes	Resultado
N	N	Branco (flat)
S	N	Geometry color
N	S	ColoringAttributes
S	S	Geometry color

Iluminação em Java3D

- Modelo de Iluminação: Modelo de Phong
  - Iluminação Ambiente: ilumina todos os vértices com o mesmo valor de intensidade. Efeito "*flat*".
  - Iluminação difusa: a iluminação num ponto depende apenas do ângulo entre a normal à superfície e a fonte de luz.
  - Iluminação especular: a iluminação num ponto depende do ângulo entre a posição do observador e a fonte de luz.

- Influência entre objectos
  - Não é considerada a influência entre os vários objectos. Não existem sombras.
  - A iluminação é calculada individualmente para cada objecto.

Iluminação em Java3D

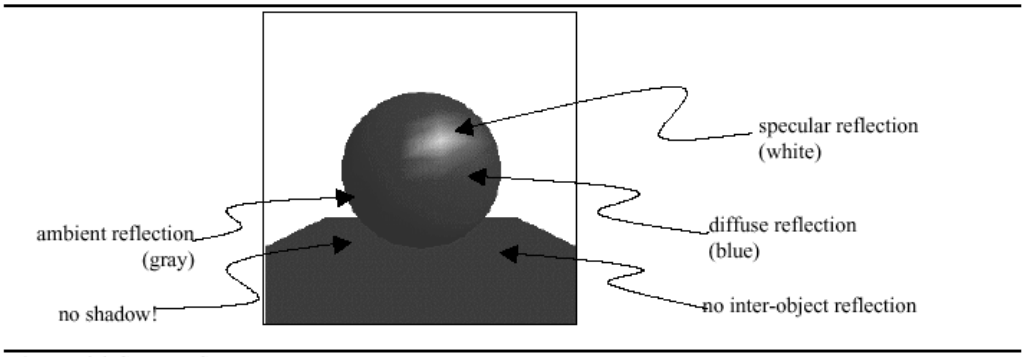


Figure 6-2 Shaded Sphere and Plane

- Modelo de Cor: o cálculo de iluminação é efectuado separadamente para cada uma das componentes RGB.

Ex:

Um objecto de cor Vermelha (1,0,0) na presença de uma fonte de luz Azul (0,0,1) não é visível.

Iluminação em Java3D

```
Material(Color3f ambienteColor, Color3f
  emissiveColor, Color3f diffuseColor, Color3f
  specularColor, float shininess)
```

```
Color3f black = new Color3f(0.0f, 0.0f, 0.0f);
Color3f deepRed = new Color3f(0.9f, 0.2f, 0.1f);
Color3f royalBlue = new Color3f(0.1f, 0.3f, 0.9f);
Color3f white = new Color3f(1.0f, 1.0f, 1.0f);
Color3f yellow = new Color3f(1.0f, 1.0f, 0.0f);

App = new Appearance();
App.setMaterial(new Material(Material(deepRed, black, deepRed, black, 1.0f);
```

Iluminação em Java3D

- Região de Influência
  - Uma fonte de luz só é usada se estiver definida a sua região de influência.
  - Objectivo: optimização dos cálculos necessários para efectuar *rendering*. São apenas consideradas em cada momento as fontes de luz cuja região de influência abrange a região visível.

```
setInfluencingBounds(Bounds region)
```

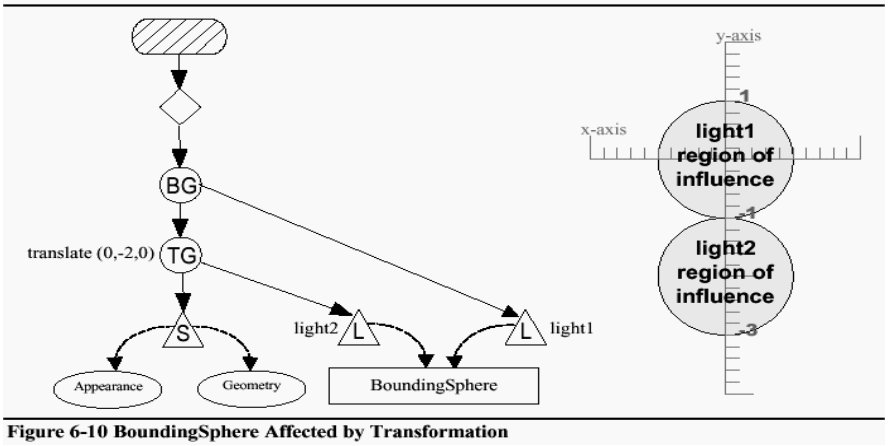


Figure 6-10 BoundingSphere Affected by Transformation

Passos necessários para definir uma fonte de luz

Definição da fonte de luz

- Região de influência
- Adicionar ao *Scene Graph*

Objecto(s) a iluminar:

- Definir as normais em cada vértice
- Definir objecto(s) Material

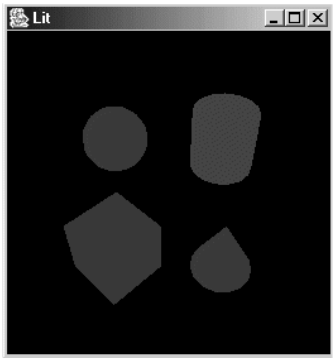
Fontes de luz:

- **AmbientLight** — `AmbienteLight(Color3f c)`
- **DirectionLight** — `DirectionLight(Color3f c, vector3f d)`
- **PointLight**
- **SpotLight**

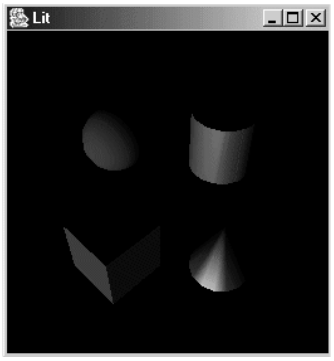
Contribuem apenas para a componente Difusa e Especular do modelo de iluminação.

Luz Ambiente e direccional

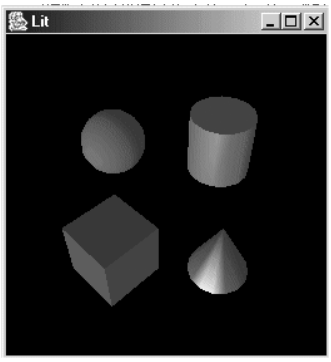
Luz ambiente



Luz direccional



Ambas

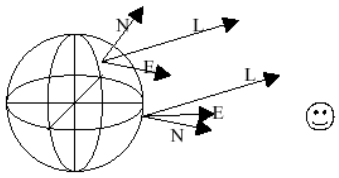



```
// Fonte global, ambient light
Color3f alColor = new Color3f(0.8f, 0.8f, 0.8f);
AmbientLight aLgt = new AmbientLight(alColor);
aLgt.setInfluencingBounds(bounds);
graphRoot.addChild(aLgt);
```

```
// Fonte direccional (infinite)
Color3f  Color1 = new Color3f(0.9f, 0.9f, 0.9f);
Vector3f  Dir1  = new Vector3f(1.0f, 1.0f, -1.0f);
DirectionalLight lgt1 = new DirectionalLight(Color1, Dir1);
lgt1.setInfluencingBounds(bounds);
 graphRoot.addChild(lgt1);
```

Fontes de luz

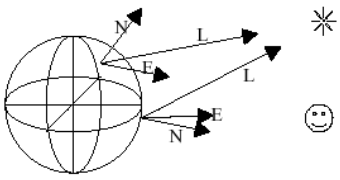
DirectionLight



Características:

- Uma única direcção dos raios luminosos.
- Não há atenuação de intensidade da luz no espaço; considera-se a fonte no infinito.

PointLight



Características:

- Tem uma localização no espaço.
- A intensidade da luz diminui com a distância; simula a lâmpada convencional.

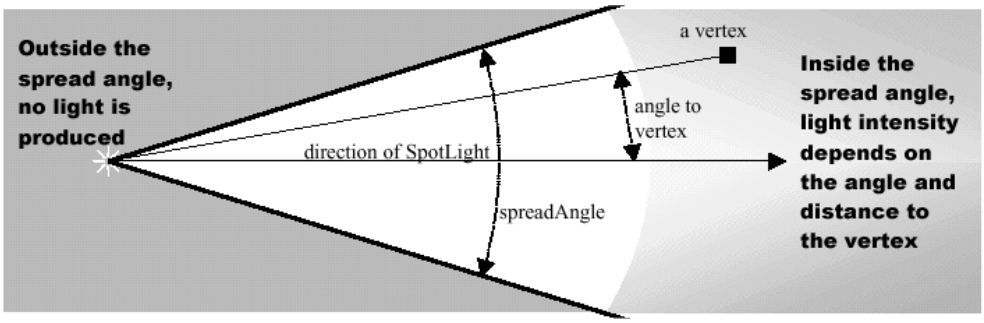
`PointLight(Color3f c, Point3f position, Point3f attenuation)`

(c, l, q)

$$\text{Atenuação} = \frac{1}{C + l * \text{distância} + q * \text{distância}^2}$$

Fonte de luz

SpotLight



Características:

- Atenuação no espaço – idêntico à PointLight
- Permite iluminar parcialmente um objecto (não se verifica nas outras fontes de luz).
- Intensidade diminui com o aumento do ângulo entre a direcção da luz e o vértice iluminado.

Interacção e Animação

Interacção: a acção ocorre em resposta a estímulos provocados pelo utilizador

Animação: a acção ocorre pela passagem do tempo

Behavior class: classe abstracta que fornece os mecanismos necessários para responder a eventos possibilitando a alteração do grafo em *run time*

Interacção e Animação

Exemplos de estímulos: teclado, rato, colisão de objectos, tempo, combinação de vários eventos,...

Exemplos de Acções: adicionar/remover objectos da cena, mudar atributos de objectos, lançar *Threads*,...

As subclasses de Behavior têm de definir:

- Método **`initialize() -`** define o evento que activa esse behavior
- Método **`processStimulus(Enumeration c)`** – método invocado pelo sistema quando ocorre o evento correspondente. A última instrução deve definir novamente a nova condição de activação.
- *Scheduling Region* : especifica a região do espaço onde o *behavior* é válido. Restringe a região onde são verificadas as condições de activação. Melhora o desempenho do sistema.

Exemplo: Tempo decorrido

```
public class SimpleBehavior extends Behavior{

        private TransformGroup targetTG;
        private Transform3D rotation = new Transform3D();

        // create SimpleBehavior
        SimpleBehavior(){
            this.targetTG = null;
        }

        // initialize the Behavior,  set initial wakeup condition
        //     called when behavior beacomes live
        public void initialize(){
            // set initial wakeup condition
            this.wakeupOn(new WakeupOnElapsedTime(5000));
        }

        // behave called by Java 3D when appropriate stimulus occures
        public void processStimulus(Enumeration criteria){
            // decode event

            // do what is necessary
            System.out.println("Passaram 5 segundos.");

            this.wakeupOn(new WakeupOnElapsedTime(5000));
        }

} // end of class SimpleBehavior
```

Exemplo: Tempo decorrido

No programa principal:

```
....

BranchGroup objRoot = new BranchGroup();

SimpleBehavior myBehavior = new SimpleBehavior();
myBehavior.setSchedulingBounds(new BoundingSphere());
objRoot.addChild(myBehavior);

....
```