Java 3D

Criação de Geometria

Criação da Geometria do Objecto

- Todos os objectos da cena são visualizados com base na representação das coordenadas dos seus vértices.

Métodos da classe Shape3D:

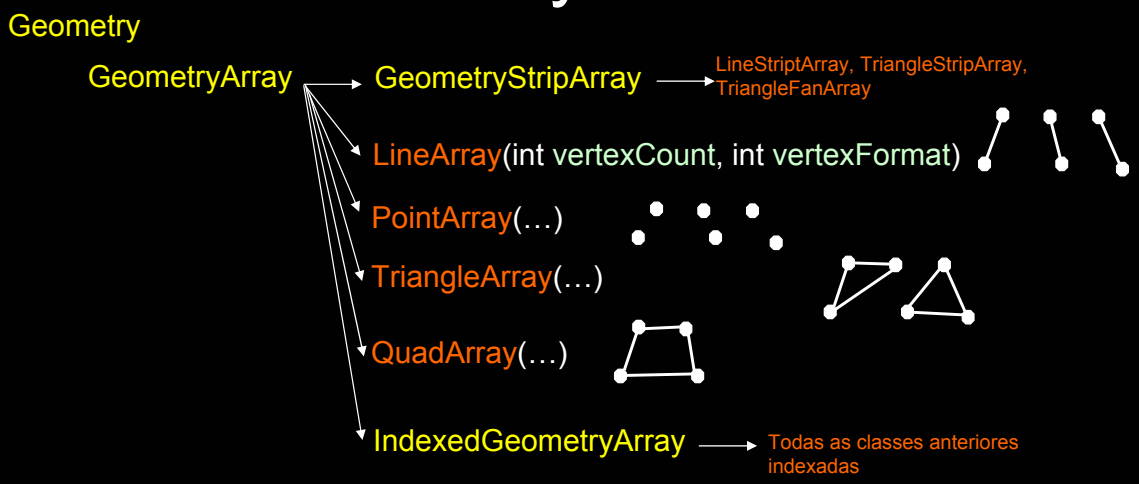
```
Void addGeometry(Geometry geometry)
Void SetAppearance(Appearance appearance)
```

Geometry: superclasse de um conjunto de classes usadas para representar a geometria de objectos usando as coordenadas dos vértices.

Especificação de Geometria

- Geometry Utility Classes
  - com.sun.j3d.utils.geomery.*
    - Box()
    - Cone()
    - Cylinder()
    - Sphere()
    - ColorCube()
- Geometry Classes [pág. 2-22]
  - javax.media.j3d.*;

Geometry Classes



Cada vértice pode especificar até 4 parâmetros (indicado por vertexFormat):

- Coordenadas
- Cor
- Normais à superfície (necessário para calculo de iluminação)
- Coordenadas de textura

Geometria

- A principal diferença entre as classes de especificação de geometria está no número de vértices guardados.
- Nos *Arrays* básicos o mesmo vértice pode aparecer mais do que uma vez. Maior consumo de memória mas obtém-se *rendering* mais eficiente.
- Com *Arrays* indexados cada vértice aparece apenas uma vez. Um nível mais de redireccionamento => maior complexidade no *rendering.*

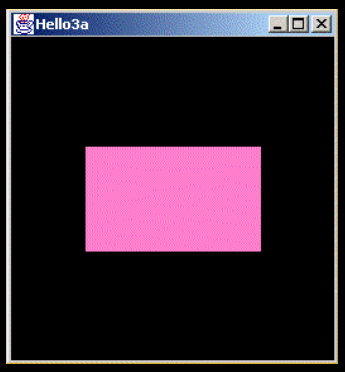
Recomendação: usar StripArrays sempre que possível

(ver exemplos: AxisApp.java, YoyoApp.java e Axis.java)

com.sun.j3d.utils.geometry. *

Classes para representar primitivas geométricas:

- Box
- Cone
- Cylinder
- Sphere
- ColorCube



Exemplo:

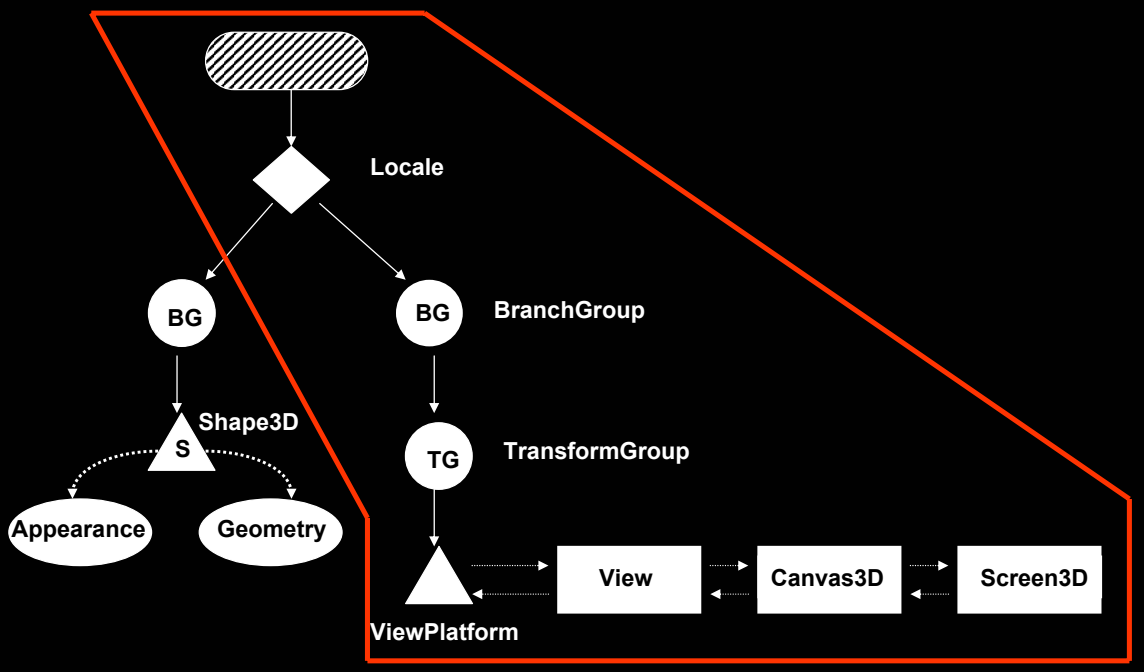
```
Appearance app = new Appearance();
app.setColoringAttributes(new
        ColoringAttributes(1.f,0.5f,0.8f,ColoringAttributes.FASTEST));
objRoot.addChild(new Box(0.5f,0.3f,0.2f, app));

// ver método getShape() e setAppearance() de Shape3D para atribuír cores diferentes às faces
```

(ver exemplo: ConeYoyoApp.java e Axis.java)

YoyoLineApp.java

```
private Geometry yoyoGeometry() {

        TriangleFanArray tfa;
        int     N = 17;
        int     totalN = 4*(N+1);
        Point3f coords[] = new Point3f[totalN];
        int     stripCounts[] = {N+1, N+1, N+1, N+1};
        float   r = 0.6f;
        float   w = 0.4f;
        int     n;
        double  a;
        float   x, y;

        // set the central points for four triangle fan strips
        coords[0*(N+1)] = new Point3f(0.0f, 0.0f, w);
        coords[1*(N+1)] = new Point3f(0.0f, 0.0f, 0.0f);
        coords[2*(N+1)] = new Point3f(0.0f, 0.0f, 0.0f);
        coords[3*(N+1)] = new Point3f(0.0f, 0.0f, -w);

        for(a = 0,n = 0; n < N; a = 2.0*Math.PI/(N-1) * ++n){
                x = (float) (r * Math.cos(a));
                y = (float) (r * Math.sin(a));
                coords[0*(N+1)+n+1] = new Point3f(x, y, w);
                coords[1*(N+1)+N-n] = new Point3f(x, y, w);
                coords[2*(N+1)+n+1] = new Point3f(x, y, -w);
                coords[3*(N+1)+N-n] = new Point3f(x, y, -w);
        }

           tfa = new TriangleFanArray (totalN,
                                       TriangleFanArray.COORDINATES,
                                       stripCounts);

        tfa.setCoordinates(0, coords);

        return tfa;

} // end of method yoyoGeometry in class Yoyo
```


SimpleUniverse



Posição da Câmara

```
TransformGroup tfcamara;

tfcamara = sUniv.getViewingPlatform().getViewPlatformTransform();

t3d = new Transform3D();
t3d.setTranslation(new Vector3f(0.0f, 0.0f, 1.2f));
Transform3D t3droty = new Transform3D();
t3droty.rotY(-Math.PI/4);
Transform3D t3drotx = new Transform3D();
t3drotx.rotX(Math.PI/4);
t3droty.mul(t3drotx);

t3droty.mul(t3d);
tfcamara.setTransform(t3drot);
```