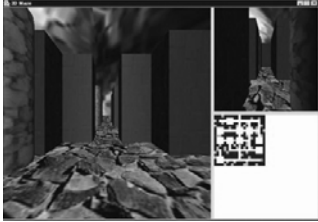


Computer Engineering
240-302, Computer Engineering Lab IV (Software)

Introduction to Java 3D (1)

Dr. Andrew Davison
Room 101, CoE
dandrew@ratree.psu.ac.th



240-302 Comp. Eng. Lab IV. Java 3D 1

Computer Engineering

Contents

- ❖ 1. Java 3D Overview
- ❖ 2. What is a Scene Graph?
- ❖ 3. A Java 3D Skeleton
- ❖ 4. Lighting a Scene
- ❖ 5. Making a Background
- ❖ 6. 3D Shapes
- ❖ 7. Shape Colour
- ❖ 8. Transformations
- ❖ 9. More Information

240-302 Comp. Eng. Lab IV. Java 3D 2

Computer Engineering

1. Java 3D Overview

- ❖ A high-level API for building interactive 3D applications and applets
 - uses a *scene graph* to model/control the 3D scene
- ❖ Fast and efficient impl. on a variety of platforms
 - built on top of OpenGL and DirectX
- ❖ Other features:
 - object behaviors
 - user interaction
 - loaders for many 3D file formats

240-302 Comp. Eng. Lab IV. Java 3D 3

Computer Engineering

Some Application Areas

- ❖ Scientific/Medical/Data visualization
- ❖ Geographical information systems (GIS)
- ❖ Computer-aided design (CAD)
- ❖ Simulation
- ❖ Computer-aided education (CAE)
- ❖ Games
 - my interest!

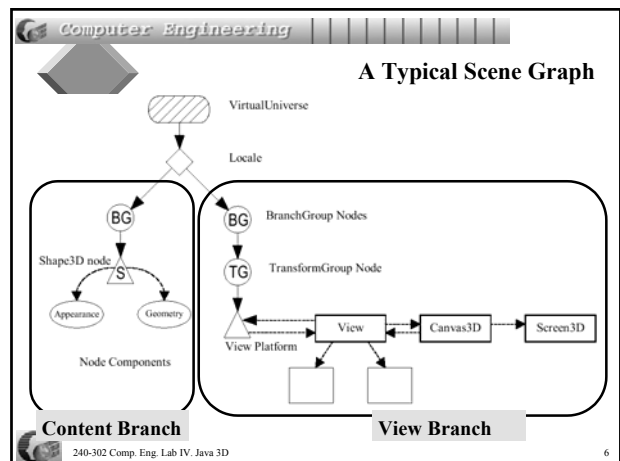
240-302 Comp. Eng. Lab IV. Java 3D 4

Computer Engineering

2. What is a Scene Graph?







- ❖ A *scene graph* is a tree-like data structure that stores, organizes, and renders 3D scene information (3D objects, materials, lights, behaviours).
- ❖ The scene graph makes 3D programming considerably easier than directly coding in OpenGL or DirectX.

240-302 Comp. Eng. Lab IV. Java 3D 5



Computer Engineering

Scene Graph Symbols

Nodes and Node Components (objects)	Arcs (object relationships)
 Group	 Parent-child link
 Leaf	 Reference
 Node Component	
 Other objects	

240-302 Comp. Eng. Lab IV. Java 3D 7

Computer Engineering

3. A Java 3D Skeleton

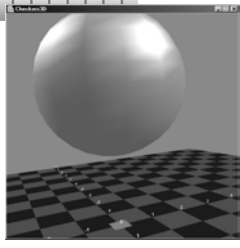
- ❖ All our Java 3D programs embed a scene inside a `JPanel`, which is part of a `JFrame`
 - this allows Swing GUI controls to be utilized along side the Java 3D panel (if necessary)
- ❖ We will develop an example called `Checkers3D` during the course of these slides.

240-302 Comp. Eng. Lab IV. Java 3D 8

Computer Engineering

Checkers3D

- ❖ The scene consists of
 - a dark green and blue tiled surface (and red center)
 - labels along the X and Z axes
 - a blue background
 - a floating sphere lit from two different directions
 - the user (viewer) can move through the scene by moving the mouse



240-302 Comp. Eng. Lab IV. Java 3D 9

Computer Engineering

Checkers3D.java

```

public class Checkers3D extends JFrame
{
    public Checkers3D()
    {
        super("Checkers3D");
        Container c = getContentPane();
        c.setLayout( new BorderLayout() );
        WrapCheckers3D w3d = new WrapCheckers3D();
        // panel holding the 3D scene
        c.add(w3d, BorderLayout.CENTER);
        setDefaultCloseOperation( JFrame.EXIT_ON_CLOSE );
        pack();
        setResizable(false); // fixed size display
        show();
    } // end of Checkers3D()

    public static void main(String[] args)
    {
        new Checkers3D();
    } // end of Checkers3D class
}

```

240-302 Comp. Eng. Lab IV. Java 3D 10

Computer Engineering

Scene Graph for Checkers3D

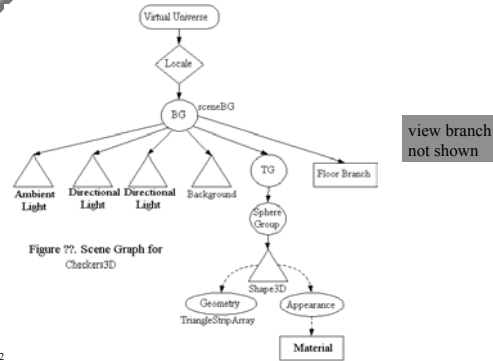


Figure ?? Scene Graph for Checkers3D

view branch not shown

240-302 11

Computer Engineering

Building the Scene Graph

- ❖ The `VirtualUniverse`, `Locale`, and `view` branch graph often have the same structure across different applications
 - programmers usually create them with the `SimpleUniverse` utility class

240-302 Comp. Eng. Lab IV. Java 3D 12

Computer Engineering

WrapChecker3D Constructor

```

private SimpleUniverse su;
private BranchGroup sceneBG; // for content branch
private BoundingSphere bounds;
:

public WrapCheckers3D()
{ setLayout( new BorderLayout() );
  setOpaque( false );
  setPreferredSize( new Dimension(PWIDTH, PHEIGHT));

  GraphicsConfiguration config =
    SimpleUniverse.getPreferredConfiguration();
  Canvas3D canvas3D = new Canvas3D(config);
  add("Center", canvas3D);
  canvas3D.setFocusable(true);
  canvas3D.requestFocus();

  su = new SimpleUniverse(canvas3D);

```

240-302 Comp. Eng. Lab IV. Java 3D 13

Computer Engineering

```

createSceneGraph();
initUserPosition(); // set user's viewpoint

orbitControls(canvas3D);
// controls for moving the viewpoint

su.addBranchGraph( sceneBG );

} // end of WrapCheckers3D()

```

240-302 Comp. Eng. Lab IV. Java 3D 14

Computer Engineering

createSceneGraph()

```

private void createSceneGraph()
// initialise the scene below sceneBG
{
  sceneBG = new BranchGroup();
  bounds = new BoundingSphere(new Point3d(0,0,0),
    BOUNDSIZE);

  lightScene(); // add the light
  addBackground(); // add the sky
  sceneBG.addChild( new CheckerFloor().getBG() );
  // add the floor

  floatingSphere(); // add the floating sphere

  sceneBG.compile(); // fix the scene
} // end of createSceneGraph()

```

240-302 Comp. Eng. Lab IV. Java 3D 15

Computer Engineering

4. Lighting a Scene

- ❖ There are four types of lighting:
 - ambient light
 - directional light
 - point light
 - spotlight
- ❖ Any number of these can be added to a scene.
- ❖ A shape must be set up to reflect light (see later).

240-302 Comp. Eng. Lab IV. Java 3D 16

Computer Engineering

4.1. Ambient Light

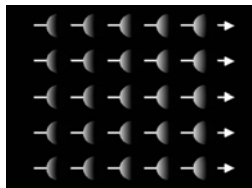
- ❖ Ambient light has the same intensity in all locations and directions
 - used as a kind of 'background' lighting in many scenes

240-302 Comp. Eng. Lab IV. Java 3D 17

Computer Engineering

4.2. Directional Light

- ❖ Provides light shining in one direction
 - often used to approximate the sun

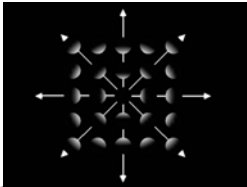


240-302 Comp. Eng. Lab IV. Java 3D 18

Computer Engineering

4.3. Point Light

- ❖ Light emitted from a point
 - the intensity changes with distance
 - often used to approximate light bulbs, candles, etc.



240-302 Comp. Eng. Lab IV. Java 3D 19

Computer Engineering

4.4. SpotLight

- ❖ Adds direction and concentration to the PointLight



240-302 Comp. Eng. Lab IV. Java 3D 20

Computer Engineering

4.5. lightScene() in WrapChecker3D

```
private void lightScene()
/* One ambient light, 2 directional lights */
{
    Color3f white = new Color3f(1.0f, 1.0f, 1.0f);
    // Red, Green, Blue values in 0-1 range

    // Set up the ambient light
    AmbientLight ambientLightNode =
        new AmbientLight(white);
    ambientLightNode.setInfluencingBounds(bounds);

    sceneBG.addChild(ambientLightNode);
    :
}
```

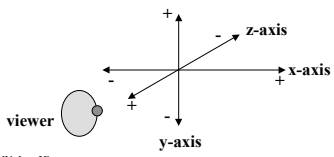
Lights must be given bounds in which to operate.

240-302 Comp. Eng. Lab IV. Java 3D 21

Computer Engineering

```
// Set up the directional lights
Vector3f light1Direction =
    new Vector3f(-1.0f, -1.0f, -1.0f);
    // (x,y,z) left, down, backwards

Vector3f light2Direction =
    new Vector3f(1.0f, -1.0f, 1.0f);
    // (x,y,z) right, down, forwards
:
```



240-302 Comp. Eng. Lab IV. Java 3D 22

Computer Engineering

```
DirectionalLight light1 =
    new DirectionalLight(white, light1Direction);
light1.setInfluencingBounds(bounds);
sceneBG.addChild(light1);

DirectionalLight light2 =
    new DirectionalLight(white, light2Direction);
light2.setInfluencingBounds(bounds);
sceneBG.addChild(light2);

} // end of lightScene()
```

240-302 Comp. Eng. Lab IV. Java 3D 23

Computer Engineering

5. Making a Background

```
private void addBackground()
// A blue sky
{ Background back = new Background();
  back.setApplicationBounds(bounds);
  back.setColor(0.17f, 0.65f, 0.92f);
  // sky colour
  sceneBG.addChild(back);
} // end of addBackground()
```

A background can be a constant colour (as here), an image, or a geometry.

240-302 Comp. Eng. Lab IV. Java 3D 24

Computer Engineering

6. 3D Shapes

- ❖ A 3D shape is defined as a `Shape3D` object.
- ❖ Each `Shape3D` object has two node components:
 - Geometry
 - ◆ made up of coordinates (vertices)
 - Appearance
 - ◆ e.g. colour, texture, transparency, material

240-302 Comp. Eng. Lab IV. Java 3D continued 25

Computer Engineering

- ❖ There are several predefined shape classes in `com.sun.j3d.utils.geometry`:
 - Box, Sphere, Cone, Cylinder
 - ◆ we use Sphere in Checkers3D
- ❖ Usually these classes are insufficient, and a shape's geometry must be built by connecting vertices.

240-302 Comp. Eng. Lab IV. Java 3D 26

Computer Engineering

Building Geometry

- ❖ The `GeometryArray` class is the parent for several useful geometry subclasses

240-302 Comp. Eng. Lab IV. Java 3D 27

Computer Engineering

7. Shape Colour

- ❖ You can specify a shape's colour in three ways:
 - in the shape's material
 - ◆ used when the scene is illuminated (as here)
 - in the shape's colouring attributes
 - ◆ used when the shape is unreflecting
 - in the vertices of the shape's geometry
 - ◆ also for unreflecting shapes

240-302 Comp. Eng. Lab IV. Java 3D 28

Computer Engineering

Illuminating a Shape

- ❖ A shape will reflect light when:
 - the scene has lights
 - ◆ set up in `lightScene()`
 - the shape has material information
 - ◆ see the next few slides
 - the shape has a geometry containing normals
 - ◆ done automatically in predefined shapes (Sphere, Box, etc.)

240-302 Comp. Eng. Lab IV. Java 3D 29

Computer Engineering

Types of Shape Reflection

240-302 Comp. Eng. Lab IV. Java 3D 30

Computer Engineering

Material

- ❖ The `Material` is part of a shape's `Appearance` node component.
- ❖ The `Material` object specifies ambient, diffuse, specular, and emissive colors and a shininess value
 - emissive colour is a kind of glowing effect, but it does not illuminate other shapes

240-302 Comp. Eng. Lab IV. Java 3D 31

Computer Engineering

Creating a Material

```
private void floatingSphere()
// A shiny blue sphere located at (0,4,0)
{
    // Create the blue appearance node
    Color3f black = new Color3f(0.0f, 0.0f, 0.0f);
    Color3f blue = new Color3f(0.3f, 0.3f, 0.8f);
    Color3f specular = new Color3f(0.9f, 0.9f, 0.9f);

    Material blueMat =
    new Material(blue, black, blue, specular, 25.0f);
    // ambient, emissive, diffuse, specular, shininess
    blueMat.setLightingEnable( true );

    Appearance blueApp = new Appearance();
    blueApp.setMaterial(blueMat);
    // position the sphere: see later
}
```

240-302 Comp. Eng. Lab IV. Java 3D 32

Computer Engineering

8. Transformations

- ❖ A shape is transformed using a `TransformGroup` object.
- ❖ The three basic transformations:
 - *translation*: move the shape to a new location
 - *rotation*: rotate the shape around the X, Y, Z axes
 - *scaling*: resize the shape
- ❖ A `TransformGroup` object is initialised using `Transform3D` objects.

240-302 Comp. Eng. Lab IV. Java 3D 33

Computer Engineering

Translation

- ❖ Build a shape


```
Shape3D myShape = new Shape3D(myGeo, myApp);
```
- ❖ Translation


```
Transform3D t3d = new Transform3D();
t3d.set( new Vector3d(2.0, 1.0, -2.0) );
```
- ❖ Create a transform group, set transform, add the shape


```
TransformGroup tg = new TransformGroup(t3d);
tg.addChild(myShape);
```

240-302 Comp. Eng. Lab IV. Java 3D 34

Computer Engineering

Rotation

- ❖ Build shape


```
Shape3D myShape = new Shape3D(myGeo, myApp);
```
- ❖ Rotation (around z-axis)


```
Transform3D t3d = new Transform3D();
t3d.rotZ(0.785); // rotate by 45 degrees
```
- ❖ Create a transform group, set transform, add the shape


```
TransformGroup tg = new TransformGroup(t3d);
tg.addChild(myShape);
```

240-302 Comp. Eng. Lab IV. Java 3D 35

Computer Engineering

Scaling

- ❖ Build shape


```
Shape3D myShape = new Shape3D(myGeo, myApp);
```
- ❖ Scaling by 1.75 in X, Y, and Z


```
Transform3D t3d = new Transform3D();
t3d.set(1.75);
```
- ❖ Create transform group, set transform, add the shape


```
TransformGroup tg = new TransformGroup();
tg.setTransform(t3d); // another way
tg.addChild(myShape);
```

240-302 Comp. Eng. Lab IV. Java 3D 36

Computer Engineering

Composite Transformations

- ❖ Combine translations, rotations, and scaling.

Methods:

```
void setTranslation(
    Vector3d vectorTranslation );
void setRotation( AxisAngle4d axisangleAxis );
void setRotation( Matrix3d matrixRotation );
void setScale( double scaleFactor );
```

240-302 Comp. Eng. Lab IV. Java 3D 37

Computer Engineering

Positioning the Sphere

```
private void floatingSphere()
// blue sphere located at (0,4,0)
{
    : // set up the blue material

// position the sphere
Transform3D t3d = new Transform3D();
t3d.set( new Vector3f(0,4,0) );
TransformGroup tg = new TransformGroup( t3d );
tg.addChild( new Sphere(2.0f, blueApp) ; );
// set its radius and appearance

sceneBG.addChild(tg);
} // end of floatingSphere()
```

240-302 Comp. Eng. Lab IV. Java 3D 38

Computer Engineering

9. More Information

- ❖ *First* install the latest version of Java:
 - <http://java.sun.com/j2se/downloads.html>
 - get the SDK (full) version
 - should be installed before Java 3D
- ❖ Also get Java's API docs and tutorial:
 - <http://java.sun.com/docs/>

240-302 Comp. Eng. Lab IV. Java 3D 39

Computer Engineering

Java 3D Software and Docs

- ❖ Main site:
 - <http://www.javasoft.com/products/java-media/3D/>
- ❖ Download page
 - <http://java.sun.com/products/java-media/3D/download.html>
- ❖ There are lots of choices between different platforms and OpenGL/DirectX
 - get a SDK (full) version
 - the OpenGL version is reportedly more stable

240-302 Comp. Eng. Lab IV. Java 3D *continued* 40

Computer Engineering

- ❖ Also get the "implementation documentation"
 - it describes the core API and the utility classes
- ❖ After installation, you will find:
 - <JAVA_HOME>\demo\java3d\
 - contains about 40 small-to-medium examples
 - a great help
 - test the installation:
 - ♦ cd to <JAVA_HOME>\demo\java3d\HelloUniverse
 - ♦ type `java HelloUniverse`
 - ♦ you should see a rotating coloured cube

240-302 Comp. Eng. Lab IV. Java 3D *continued* 41

Computer Engineering

- ❖ Download the Java 3D tutorial
 - <http://java.sun.com/products/java-media/3D/collateral/>
 - also get the example code (zipped JAR) that goes with the tutorial text

240-302 Comp. Eng. Lab IV. Java 3D 42

Computer Engineering

Resources here at CoE

- ❖ CoE Students can get Java and Java 3D from Aj. Somchai's excellent Java pages:
<http://java.coe.psu.ac.th/>
- ❖ Java
<http://java.coe.psu.ac.th/RefImp.html#J2SE>
- ❖ Java 3D
<http://java.coe.psu.ac.th/RefImp.html#Java3D>

240-302 Comp. Eng. Lab IV. Java 3D 43

Computer Engineering

Java 3D Books

Both of these are in the CoE library, or available from me.

- ❖ *Java 3D API Jump-Start*
– Aaron E. Walsh, Doug Gehringer
Prentice Hall; 0-1303-4076-6, 2001
- ❖ *Java 3D Programming*
– Daniel Selman
Manning Pub.; 1-9301-1035-9; 2002
<http://www.manning.com/selman/>

240-302 Comp. Eng. Lab IV. Java 3D *continued* 44

Computer Engineering

- ❖ The Java 3D chapters from my online book:
<http://fivedots.coe.psu.ac.th/~ad/jg/>
- ❖ The complete Checkers3D application is explained in chapter 8:
<http://fivedots.coe.psu.ac.th/~ad/jg/ch8/>

240-302 Comp. Eng. Lab IV. Java 3D 45

Computer Engineering

Java 3D Help

- ❖ Ask me!
- ❖ The Java 3D interest mailing list
<http://archives.java.sun.com/archives/java3d-interest.html>
or
<http://www.mail-archive.com/java3d-interest@java.sun.com/>
- ❖ comp.lang.java.3d newsgroup
<http://groups.google.com/groups?group=comp.lang.java.3d>

240-302 Comp. Eng. Lab IV. Java 3D *continued* 46

Computer Engineering

- ❖ Java 3D at Sun
<http://java.sun.com/products/java-media/3D/>
- ❖ j3d.org
<http://www.j3d.org>
- ❖ The java.net (formerly JavaGaming.org) forum on Java 3D:
<http://www.javagaming.org/cgi-bin/JGNetForums/YaBB.cgi?board=3D>

240-302 Comp. Eng. Lab IV. Java 3D 47