JAVA 3D API

Creating and manipulating 3D geometric objects in JAVA

(Chapter 1)

Library and Documentation

- Tutorial:
  - http://java.sun.com/developer/onlineTraining/java3d/
- Documentation
  - http://download.java.net/media/java3d/javadoc/1.5.2/index.html
- Download
  - https://java3d.dev.java.net/binary-builds.html

On my machine

- setenv CLASSPATH /home/tran1/tson/public_html/classes/fall08-476/Java3D/lib/ext/j3dcore.jar:/home/tran1/tson/public_html/classes/fall08-76/Java3D/lib/ext/j3dutils.jar:/home/tran1/tson/public_html/classes/fall08-476/Java3D/lib/ext/vecmath.jar:.

- setenv LD_LIBRARY_PATH /home/tran1/tson/public_html/classes/fall08-476/Java3D/lib/amd64

Scene Graph

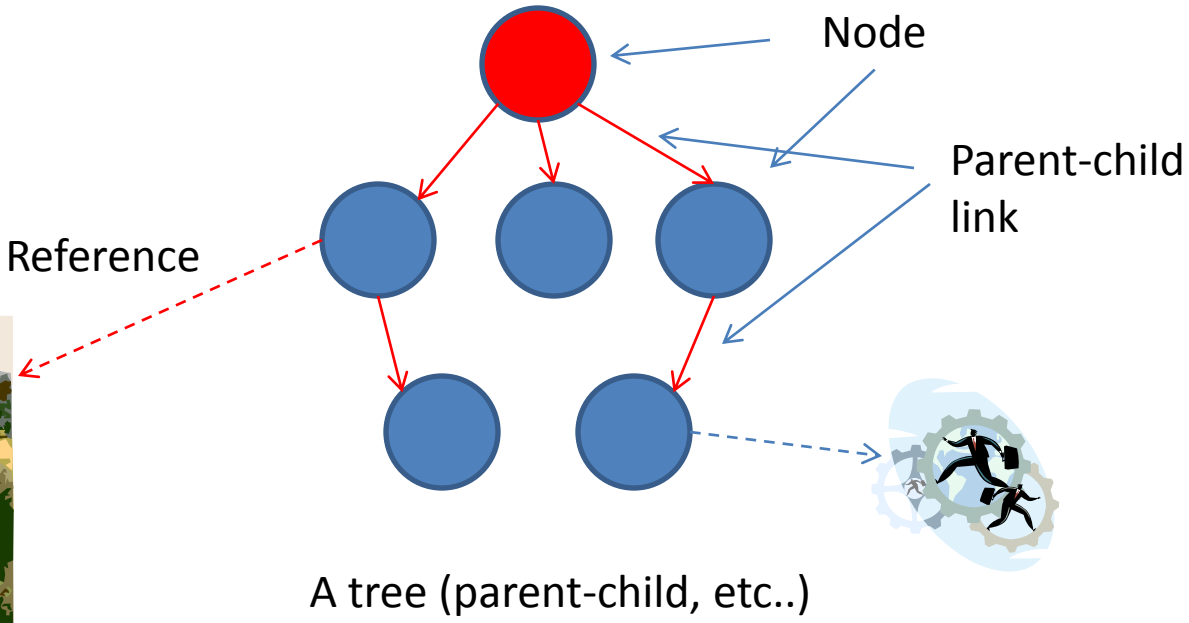
- Arrangement of Java 3D Objects in a tree structure that that completely specifies
  - the content of a virtual universe, and
  - how it is to be rendered

Basic Objects

- Core classes javax.media.j3d
- Utility com.sun.j3d.utils
  - content loaders
  - scene graph construction aids
  - geometry classes, and
  - convenience utilities.
- Every program uses
  - java.awt (abstract window for rendering) and
  - javax.vecmath (vector math classes for points, vectors, matrices, and other mathematical objects)

Building a Scene Graph

- Using instances of Java 3D classes:
  - objects (geometry, sound, lights, location, orientation, and appearance of visual and audio objects.
- Tree structure



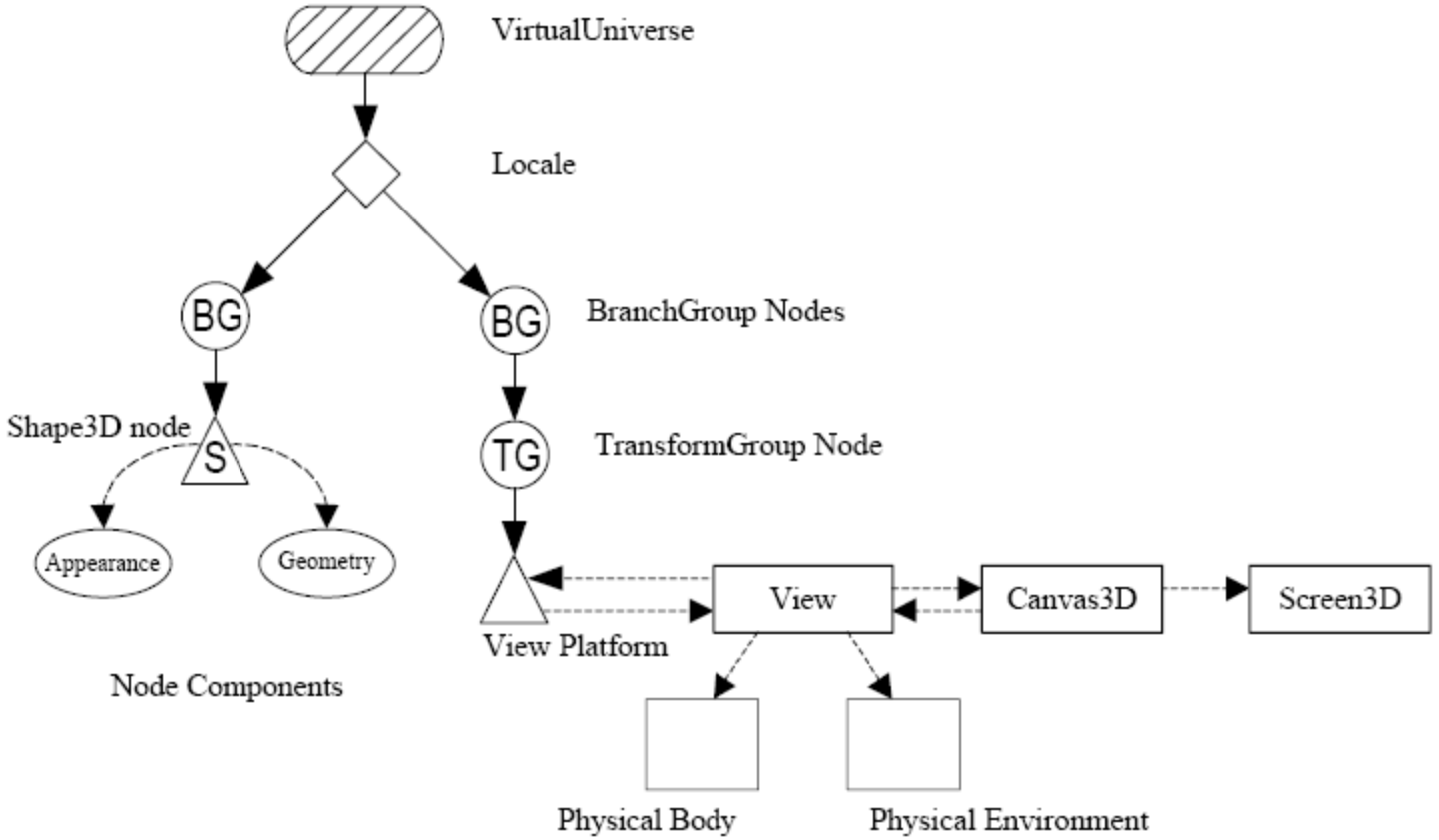


Figure 1-2 First Scene Graph Example

Nodes in Scene Graph

- A VirtualUniverse (List of Locale objects)
- Locale object: a reference point in the virtual universe (a landmark used to determine the location of visual objects in the virtual universe), usually: one per virtual universe
- Each Locale object may serve as the root of multiple subgraphs of the scene graph.
- A BranchGroup object is the root of a subgraph, or branch graph. There are two different categories of scene subgraph (tells the renderer what to do) :
  - the *view branch graph* (specifies the viewing parameters such as the viewing location and direction)
  - the *content branch graph* (the *contents of the virtual universe - geometry, appearance, behavior, location, sound, and lights.*

NodeComponent Class

The NodeComponent class is the superclass used in specifying the geometry, appearance, texture, and material properties of a Shape3D (Leaf) node. NodeComponents are not part of the scene graph, but are referenced by it. A NodeComponent may be referenced by more than one Shape3D object.

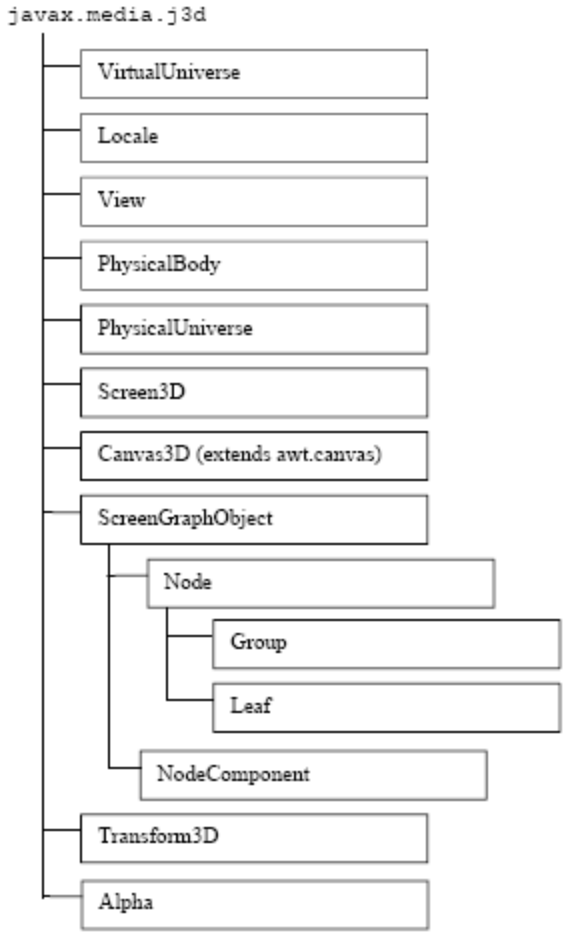


Figure 1-5 An Overview of the Java 3D API Class Hierarchy

Recipe for Writing Java 3D Program

1. Create a Canvas3D object
2. Create a VirtualUniverse object
3. Create a Locale object, attaching it to the VirtualUniverse object
4. Construct a view branch graph
   a) Create a View object
   b) Create a ViewPlatform object
   c) Create a PhysicalBody object
   d) Create a PhysicalEnvironment object
   e) Attach ViewPlatform, PhysicalBody, PhysicalEnvironment, and Canvas3D objects to View object
5. Construct content branch graph(s)
6. Compile branch graph(s)
7. Insert subgraphs into the Locale

Simple Recipe (Simplified Recipe)

1. Create a Canvas3D object
2. SimpleUniverse (Steps 2, 3, 4 in the recipe) which references the Canvas3D
   a) Customized the SimpleUniverse objects
3. Construct content branch graph(s)
4. Compile branch graph(s)
5. Insert subgraphs into the Locale

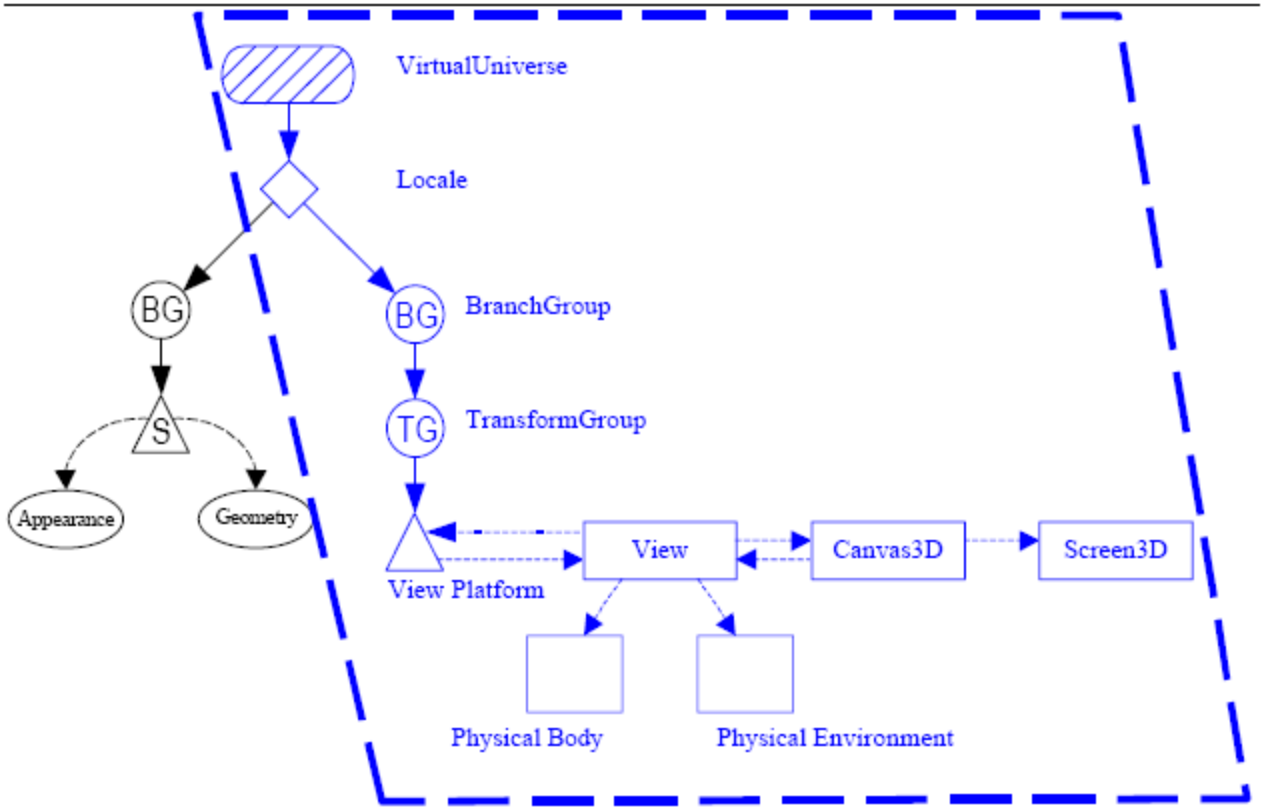


Figure 1-7 A SimpleUniverse Object Provides a Minimal Virtual Universe, Indicated by the Dashed Line.

Using a SimpleUniverse object makes the basic recipe even easier. Figure 1-8 presents the simple recipe, which is the basic recipe modified to use a SimpleUniverse object. Steps 2, 3, and 4 of the basic recipe are replaced by step 2 of the simple recipe.

SimpleUniverse

- Constructors:
  - **SimpleUniverse()**

    (Constructs a simple virtual universe)
  - **SimpleUniverse(Canvas3D canvas3D)**

    (Construct as simple universe with a reference to the named Canvas3D object)

SimpleUniverse

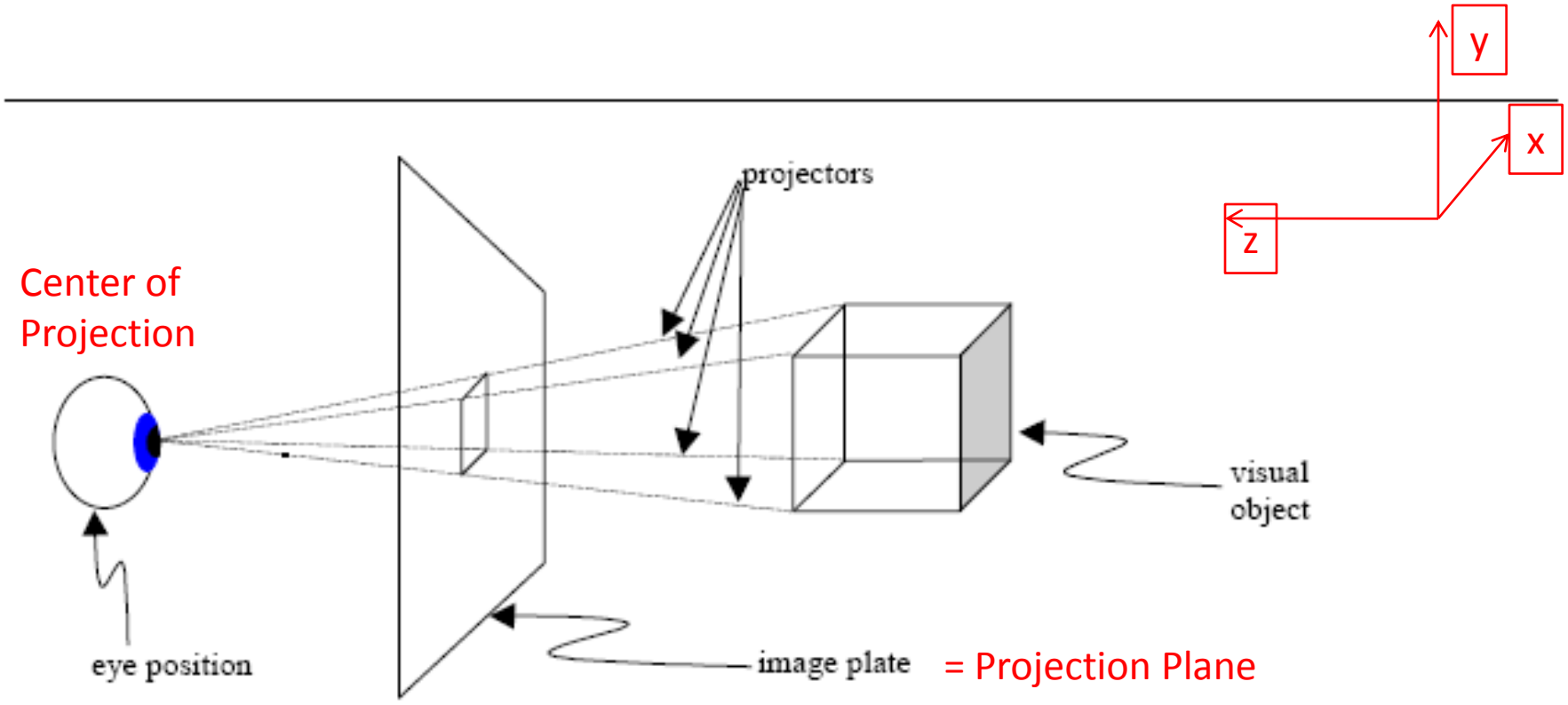


Figure 1-9 Conceptual Drawing of Image Plate and Eye Position in a Virtual Universe.

SimpleUniverse Methods (partial list)

- Package: com.sun.j3d.utils.universe

- **void addBranchGraph(BranchGroup bg)**

  Used to add Nodes to the Locale object of the scene graph created by the SimpleUniverse. This is used to add a content branch graph to the virtual universe.

- **ViewingPlatform getViewingPlatform()**

  Used to retrieve the ViewingPlatform object the SimpleUniverse instantiated. This method is used with the setNominalViewingTransform() method of ViewingPlatform to adjust the location of the view position.

ViewingPlatform

- Package: com.sun.j3d.utils.universe
- **setNominalViewingTransform() Method**
  Set the COP at (0,0,2.41) looking at negative z-direction towards the origin.
- At this viewing distance and the default field of view, objects of height and width of 2 meters generally fit on the projection plane (image plate).

Simple Recipe (Live and Compiled)

1. Create a Canvas3D object
2. SimpleUniverse (Steps 2, 3, 4 in the recipe) which references the Canvas3D
   a) Customized the SimpleUniverse objects
3. Construct content branch graph(s)
4. Compile branch graph(s) COMPILED [creating objects that can be more efficiently rendered]
5. Insert subgraphs into the Locale LIVE [objects become live, i.e., will be rendered]

Example

- HelloJava3Da.java Nominal view
- HelloJava3Db.java Transformation (Rotate)
- HelloJava3Dc.java Spin
- HelloJava3Dd.java Rotate + Spin

(see code in the examples.zip)