Java3D

Presented By
Chris Gundlach
chris@paragenix.com

Why Java3D?

- People like rich applications (see Google Local).
- AJAX stinks -- Javascript.
- Flash stinks -- Javascript.
- Java2D/Java3D can produce highly creative and high performance UI’s.
- Java Security is the enabler.

History/Future

- First version 1998
- Development stopped 2003-2004. Now Open Source.
- Previous version is 1.3.2, which is a bug fix release (Windows, Linux, Solaris, OS X).
- Current version 1.4 3/2/06 (Windows, Linux, Solaris) adds programmable shaders.
- Version 1.5 JOGL Rendering

3D Java APIs

- JOGL – A Java based OpenGL rendering API
- LWJGL (Lightweight Java Game Library) – Low level API targeted for games
- Java3D – A Scenegraph API
- Xith3D – Scenegraph API based on JOGL or LWJGL
- jME (jMonkey Engine) – Scenegraph API based on LWJGL

OpenGL Sample

```
void renderTeapot (GLfloat x, GLfloat y, GLfloat z, GLfloat ambr, GLfloat ambg, GLfloat
ambb, GLfloat difr, GLfloat difg, GLfloat difb, GLfloat specr, GLfloat specg, GLfloat specb,
GLfloat shine) {
    GLfloat mat[4];
    glPushMatrix();
    glTranslatef (x, y, z);
    mat[0] = ambr; mat[1] = ambg; mat[2] = ambb; mat[3] = 1.0;
    glMaterialfv (GL_FRONT, GL_AMBIENT, mat);
    mat[0] = difr; mat[1] = difg; mat[2] = difb;
    glMaterialfv (GL_FRONT, GL_DIFFUSE, mat);
    mat[0] = specr; mat[1] = specg; mat[2] = specb;
    glMaterialfv (GL_FRONT, GL_SPECULAR, mat);
    glMaterialf (GL_FRONT, GL_SHININESS, shine*128.0);
    glCallList(teapotList);
    glPopMatrix(); }
```

Rendering

- Java3D renders is done with either OpenGL or Direct3D (Windows).
- Java3D will be moving to using JOGL bindings
- Hardware Acceleration == Goodness
- Works on a wide variety of hardware (a T40 ThinkPad for instance).

JOGL Rendering

- Remove the current rendering code from Java 3D, will allow the project to focus on the scenegraph.
- JOGL implementations are available for more platforms (eg. OS X).
- Will allow for a lightweight component.
- 6 to 9 months to complete

Performance

- 3 Rendering Modes

- Immediate Mode
- Retained Mode
- Compiled-Retained Mode
- Uses the set capabilities flags to notify the API, what optimizations it can make.

Capabilities

- ALLOW_READ_BOUNDS
- ALLOW_APPEARANCE_WRITE
- ALLOW_GEOMETRY_WRITE

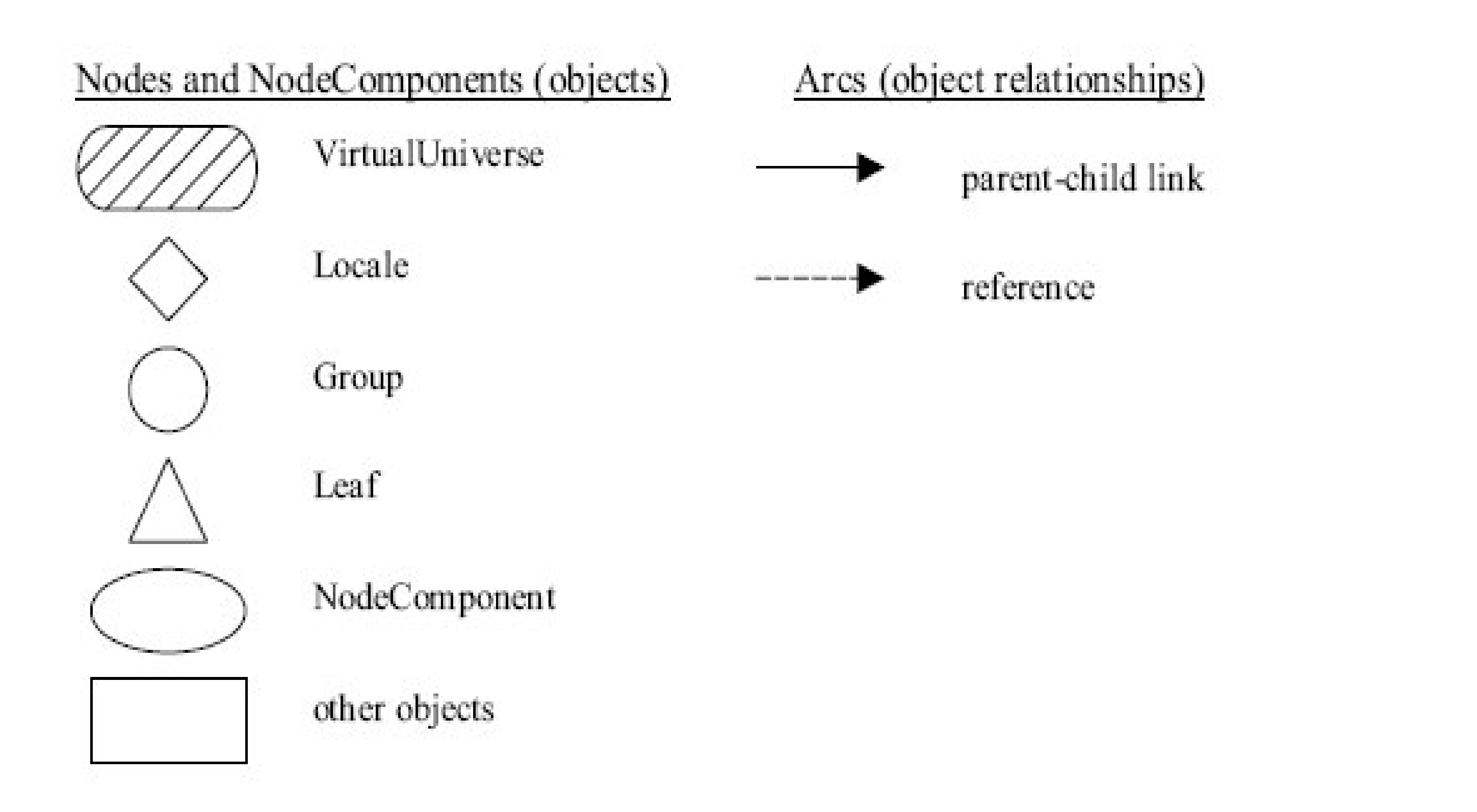
UI Clases

- Canvas3D -- is the heavyweight component where the scene is rendered.
- GraphicsConfigTemplate3D – contains instructions for rendering (for instance anti-aliasing).
- J3DGraphics2D for drawing on the Canvas3D using 2D instructions (not in compiled mode).

Component Descriptions

- VirtualUniverse - top level container of the scenegraph (SimpleUniverse)
- Locale – Hi resolution location in the Universe
- Group – Contains other Nodes
- Leaf – Shapes, Behavior, Light, Background

Scenegraph Components



Nodes and NodeComponents (objects)

▨ VirtualUniverse

◇ Locale

○ Group

△ Leaf

⬭ NodeComponent

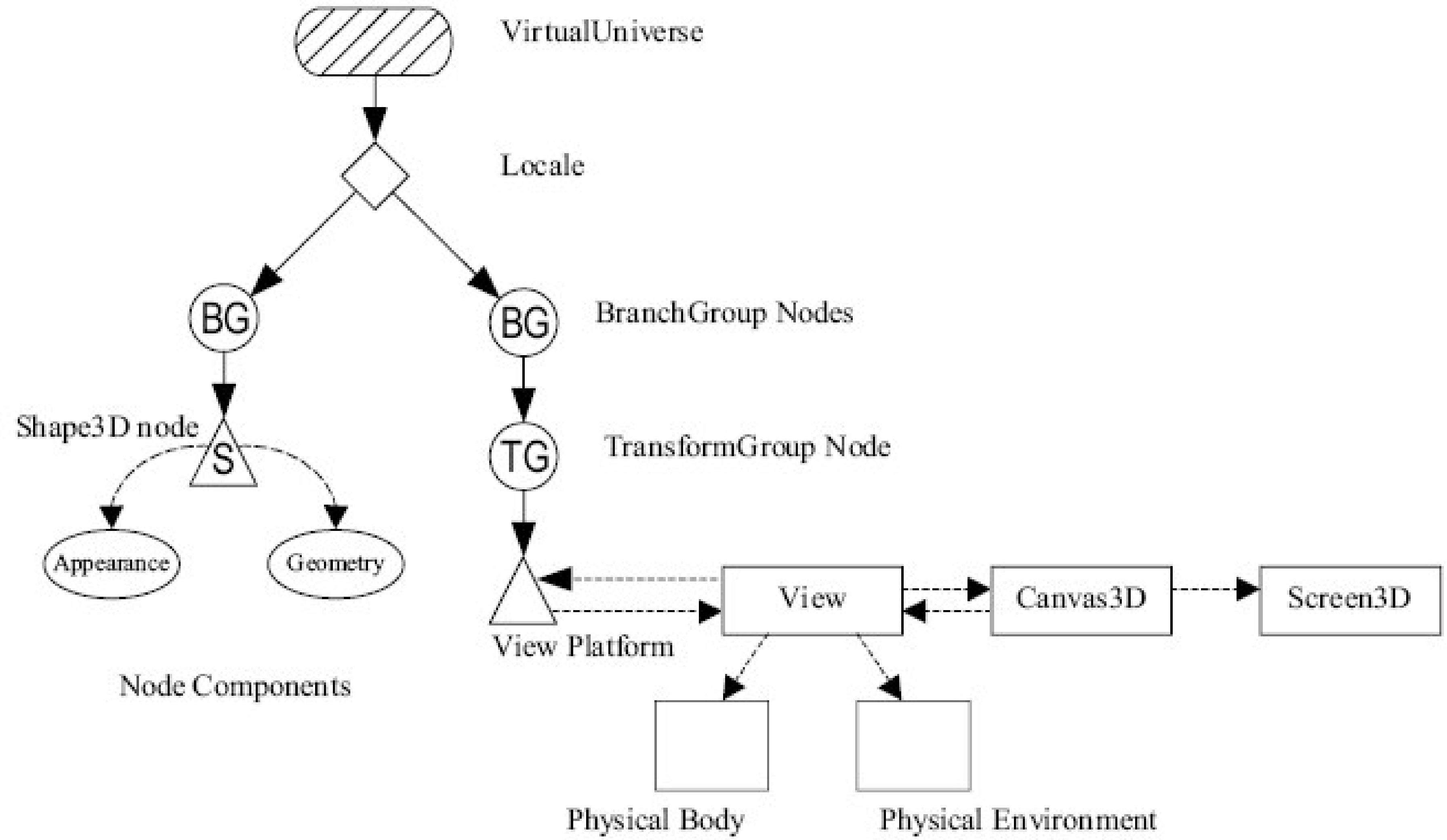
□ other objects

Arcs (object relationships)

→ parent-child link

⇢ reference

Sample Graph



Coordinates

- Default unit of measure is meters
- +x is to the right
- +y is local gravitation up
- +z is towards the viewer
- Hires Coordinates are 256 bit fixed point numbers for x,y,z

BranchGroup

- Extends Group functionality
- compile() method
  optimizes the contents of the group
- detach() method
  removes the group from the graph

Shape3D

- Geometry

  Examples -- LineArray, PointArray, QuadArray, TriangeArray, Text3D

- Appearance
  - ColoringAttributes, Texture, Material, TranparencyAttributes

Lighting

- AmbientLight – Light that affects all objects uniformly.
- DirectionalLight – Oriented light with origin at infinity.
- PointLight – Located at some coordinate and emanates in all directions.
- SpotLight – Located at some coordinate, and shines in a particular direction.

TranformationGroup

- Provides a mechanism to scale, rotate, move, etc. a subgraph.

- Transform3D
  - rotX, rotY, rotZ
  - set(scale)
  - setTranslation(Vector3d)

Behaviors/Interpolator

- Adds action to the scenegraph.

  SchedulingBounds, SchedulingInterval

- Interpolator is essentially a given transformation over time.

  Examples: PositionInterpolator, ScaleInterpolator, RotationInterpolator

Loading Models

- Instantiate your loader…
  (extends
  com.sun.j3d.loaders.LoaderBase)
  call load
  get the Scene object
  get the BranchGroup from the Scene
  add the Group to your graph

Loaders

- Lightwave and Wavefront included (com.sun.j3d.loaders…)
- 3D Studio Max
- AC3D
- VRML

Different loaders have different levels of ability to load in things like behaviors.

- http://java3d.j3d.org/utilities/loaders.html

References

- Java3D specification
- Killer Game Programming in Java
- Java3D home on java.net.
- https://java3d.dev.java.net/