Adventures in 3D with Eclipse ICE and JavaFX

Tony McCrary
Robert Smith
What is Eclipse ICE?

• Eclipse ICE is an Eclipse instance featuring a workbench for scientists.

• It integrates scientific codes such as MOOSE and VIBE, providing full workflows within a single, user-friendly environment.

• Also offers functionality for visualization of output files.
How does ICE use JavaFX modeling?

- The Geometry Editor is a Constructive Solid Geometry based editor for creating 3D shapes.
- The Mesh Editor is for 2D polygonal mesh creation, for use with tools such as Nek5000 fluid dynamics simulator.
- These tools, along with third party technologies like VisIt, are being split into an independent project, the Eclipse Advanced Visualization Project (EAVP)
Lawyers, IP Issues, and Professional Open Source

- The Eclipse Foundation requires thorough IP checks to ensure that its software is truly open and to protect itself and participants from litigation.

- ICE’s original rendering library, jMonkeyEngine, was unable to meet this high standard due to dependencies on libraries such as the Lightweight Java Game Library.

- JavaFX’s 3D modeling code, due to being integrated with the JDK as of Java 8, provides a convenient open source alternative whose ownership and licensing are not in question.
Adventures in 3D with Eclipse ICE and JavaFX

Architecture

Geometry Editor
Mesh Editor

JavaFX SceneGraph

Windowing Toolkit

Prism

OpenGL

Direct3D

JRE
Mesh Editor

This editor can construct one or two dimensional meshes.

Mode: Toggle HUD Toggle Axis Delete

Camera center (x, y): (0.0, 0.0) Cursor position (x,y): NA

Mesh

Polygon 2

<table>
<thead>
<tr>
<th>Translate</th>
<th>0.0</th>
<th>0.0</th>
<th>35.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rotation</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>Scale</td>
<td>1.0</td>
<td>1.0</td>
<td>0.1</td>
</tr>
</tbody>
</table>

Edge 5

Type: None

Values: 0.0

Thermal Boundary Condition
Mesh Editor

- Allows for the creation, editing, and display of 2D meshes.
- Polygons can be added through clicking the grid to specify the vertices.
- The visual editor can be used to edit vertices by dragging them around the screen.
- Alternatively, they can be set to exact coordinates using the Properties View.
- Edges are associated with Nek5000 boundary conditions, also editable through the Properties View.
Geometry Editor

This tool allows you to create and edit a 3D geometry.
Geometry Editor

• The ICE Geometry Editor displays pre-defined 3D shapes. (e.g. spheres, cylinders, tubes, etc.)

• Shapes can be added, removed, or modified by the user.

• User interaction is handled through a tree menu, with buttons to add, remove, or copy shapes, including multiple copies with user input offsets between them.
Geometry Editor

• The scene is modeled as a Constructive Solid Geometry tree.

• Individual simple shapes are combined into more complex shapes.

• Shapes can be modified through the transformation view.
  – A shape’s rotation, translation, and scale in the three axis directions can be set.

• Modifying a parent shape will apply the same change to its children, as if they were a single composite object.
Working with the JavaFX 3D Scene Graph

• JavaFX uses a scene graph, with each transformation applied to an object being relative to its ancestors’ transformations.
  – An object translated 5 units up which is the child of a node that is translated 1 unit to the right will be displayed 1 unit to the right and 5 up from the origin.

• Groups are the internal nodes of the tree, having graphical objects (Shape3D and MeshView) or other Groups as children.

• A camera is just another node on the graph.
Working with the JavaFX 3D Scene Graph

- Shape3D is a superclass for simple 3D shapes (box, cylinder, sphere) that JavaFX can render out of the box.

- ShapeViews are 3D shapes which are based on TriangleMeshes.

- TriangleMesh specifies a shape by:
  - An array of 3D points.
  - An array of 2D coordinates on a texture
  - An array of triangular faces as indexes into the array of points and array of texture coordinates.
  - An array of smoothing groups, where two faces are in the same smoothing group if their indices in this array have the same numerical value
  - Optionally, an array of normal vectors, with each face also having an index into this normal vector array for each of its points.
Example TriangleMesh

- Points: [0, 0, 0, 1, 0, 0, 0, 1, 0]
- Texcords: [0, 1, 0, 1, 0, 0, 1, 1, 0]
- Faces: [0, 2, 1, 1, 2, 0]
- Smoothinggroups: [0]
Working with the JavaFX 3D Scene Graph

• User interaction within a 3D environment is done with a technique called “picking”.
• During 3D scene rendering, geometry is projected from Model View space into screen space (i.e. 2D screen coordinates).
• Picking is performed by creating a ray that is unprojected from a location in 2D screen space back into 3D Model View space.
• The ray is then cast into the 3D frustum and used with ray-triangle intersection testing to determine what was on screen.
Working with the JavaFX 3D Scene Graph
• User interaction events like MouseEvent and DragEvent are extended to return PickResults.

• A PickResult contains the point, node, face, and texture coordinate intersected by the user action, as well as the distance between the intersected point and the camera.

• Arbitrary Java objects can be placed in a node’s properties.

• Picking a Java class can be done by simply retrieving it from the JavaFX node’s properties map.
Node Composition

• JavaFX’s Nodes are built on a classic Java hierarchical design.

• While being easy to understand for Java programmers, this type of design can be limiting for complex 3D applications.

• ICE uses a variant of the Composition Pattern for adding functionality to scene nodes.

• This is commonly called an Entity Component System in other industries, particularly the game industry.
Node Composition

Node

3D Shape Node

Core Channel
Node Composition
Leveraging E(fx)clipse

• E(fx)clipse is an Eclipse plug-in which handles integration between Eclipse and JavaFX.

• JavaFX’s source is not part of the standard JDK src. zip, but e(fx)clipse allows it to be stepped through while debugging.

• Equinox will not handle JavaFX packages by default, so e(fx)clipse provides an Equinox Adapter Hook to allow OSGI services to make use of JavaFX classes.
JavaFX 3D Eclipse Interop

• FXCanvas is a SWT Canvas with a JavaFX Scene embedded inside.
• JavaFX will draw its scene and place the resulting image in the canvas.
• The canvas will transparently forward user interaction events to the scene.
JavaFX Design Considerations

- No exposed shader support, so you're limited to built in rendering capabilities/materials
- No direct access to matrix transforms, limited to basic Orthographic and Perspective camera implementations
- No support for custom vertex stream configurations (limited to vertex/normal and vertex/normal/tc)
JavaFX Performance Considerations

• JavaFX 3D is an early stage technology, needs a lot of performance work

• Current JavaFX 3D design is meant for casual development

• SWT FXCanvas implementation doesn't blit directly into the platform framebuffer, relies on expensive texture read backs and copies from GART and GPU memory
JavaFX Performance Considerations

• All mesh data access is through slow observable arrays, not fast off-heap ByteBuffers

• Implemented as a simple forward renderer, so scene complexity (number of lights, etc) is limited

• Limited to simple scenes, no built in spatial acceleration structures (no BVH, k-d trees, octrees, etc).
JavaFX Design Considerations

• 3D scenegraph built directly on top of the 2D orthographic UI rendering system
• JavaFX has a default Y-down spatial orientation, which is common for 2D but unusual for 3D
• SubScenes, which act as independent Scenes rendered within the real Scene and using their own camera, allow some level of control over this issue, but require extra work to set up.
Dealing with Implementation Quirks

• JavaFX is incompatible with GTK3, and must be run under GTK2 to avoid hard crashes.

• Under Linux or other platforms using SWT’s GTK+ backend you must disable GTK with an environment variable: SWT_GTK3=0

• GTK3 support is in the works for a future JavaFX release.
Before and After
Eclipse ICE
https://www.eclipse.org/ice/

Eclipse EAVP
https://projects.eclipse.org/projects/technology.eavp

JavaFX 3D Documentation
http://docs.oracle.com/javase/8/javafx/graphics-tutorial/javafx-3d-graphics.htm
Questions?
Please Evaluate This Talk

Sign-in: www.eclipsecon.org
Select session from schedule