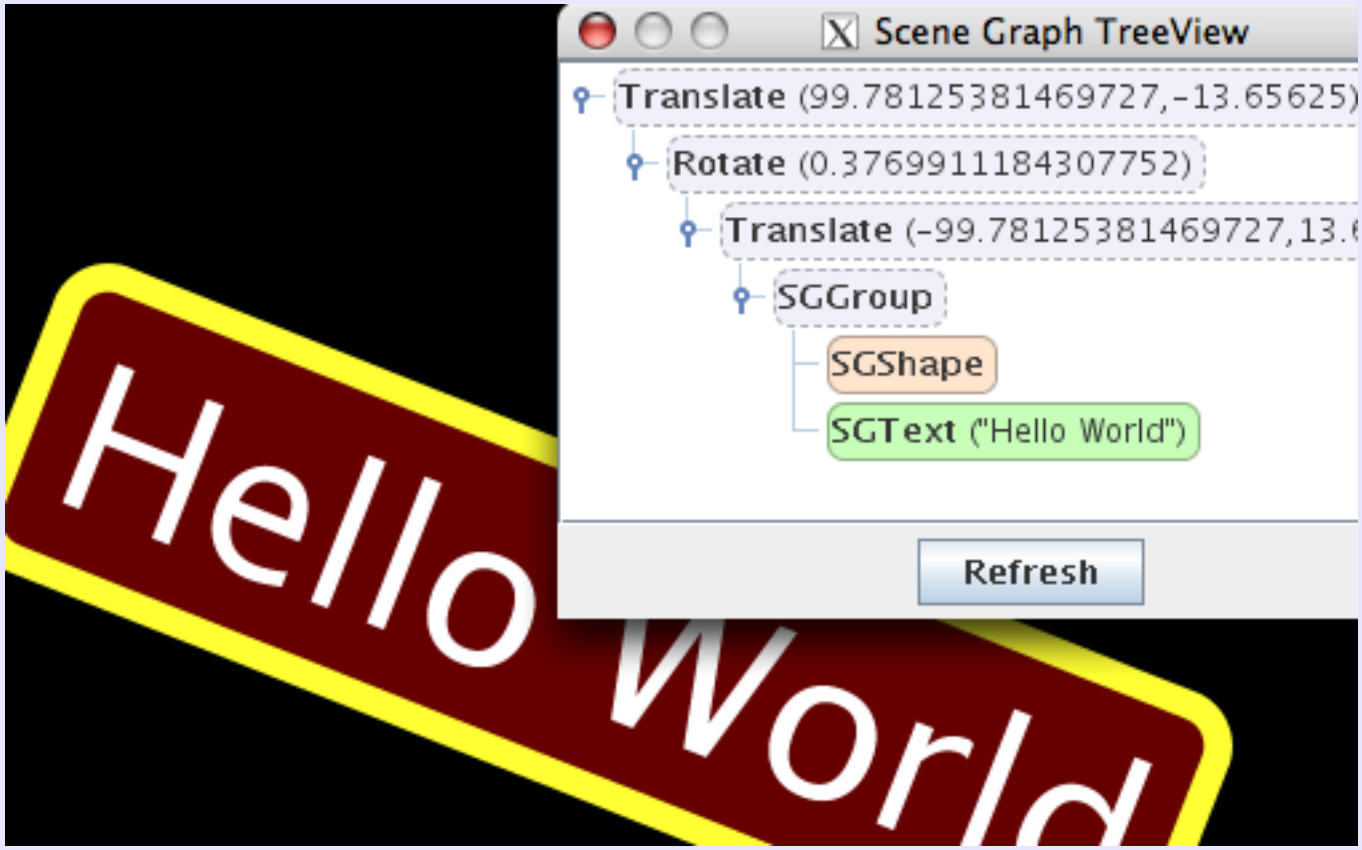
Java Scene Graph



Steven Reynolds
Senior Product Manager
INT

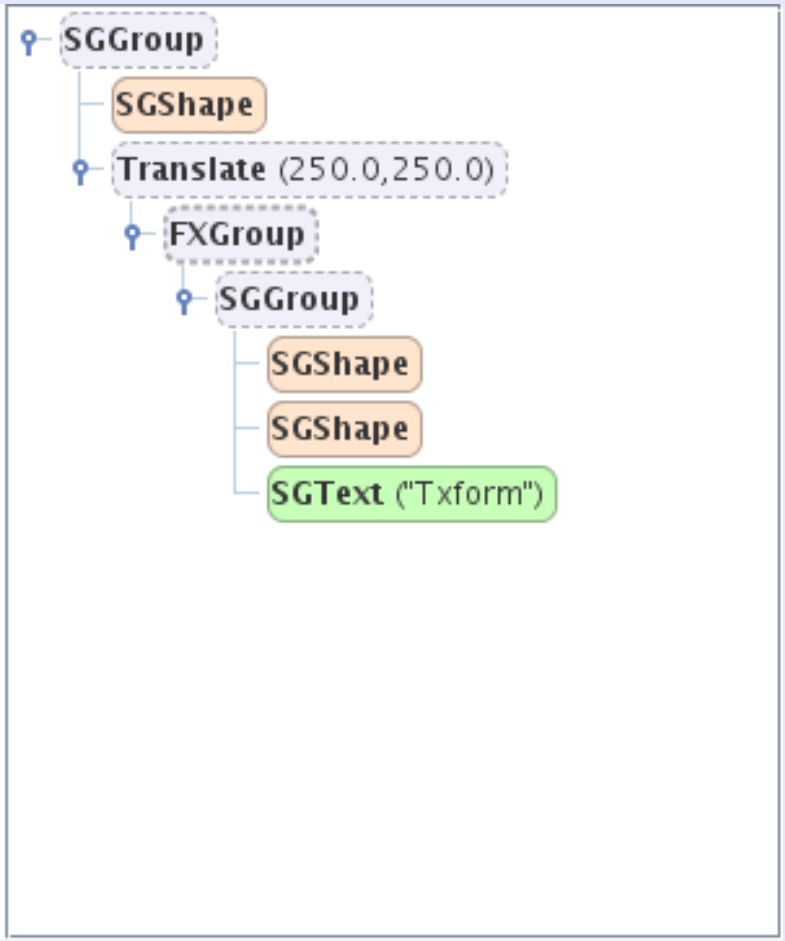
These slides are available at

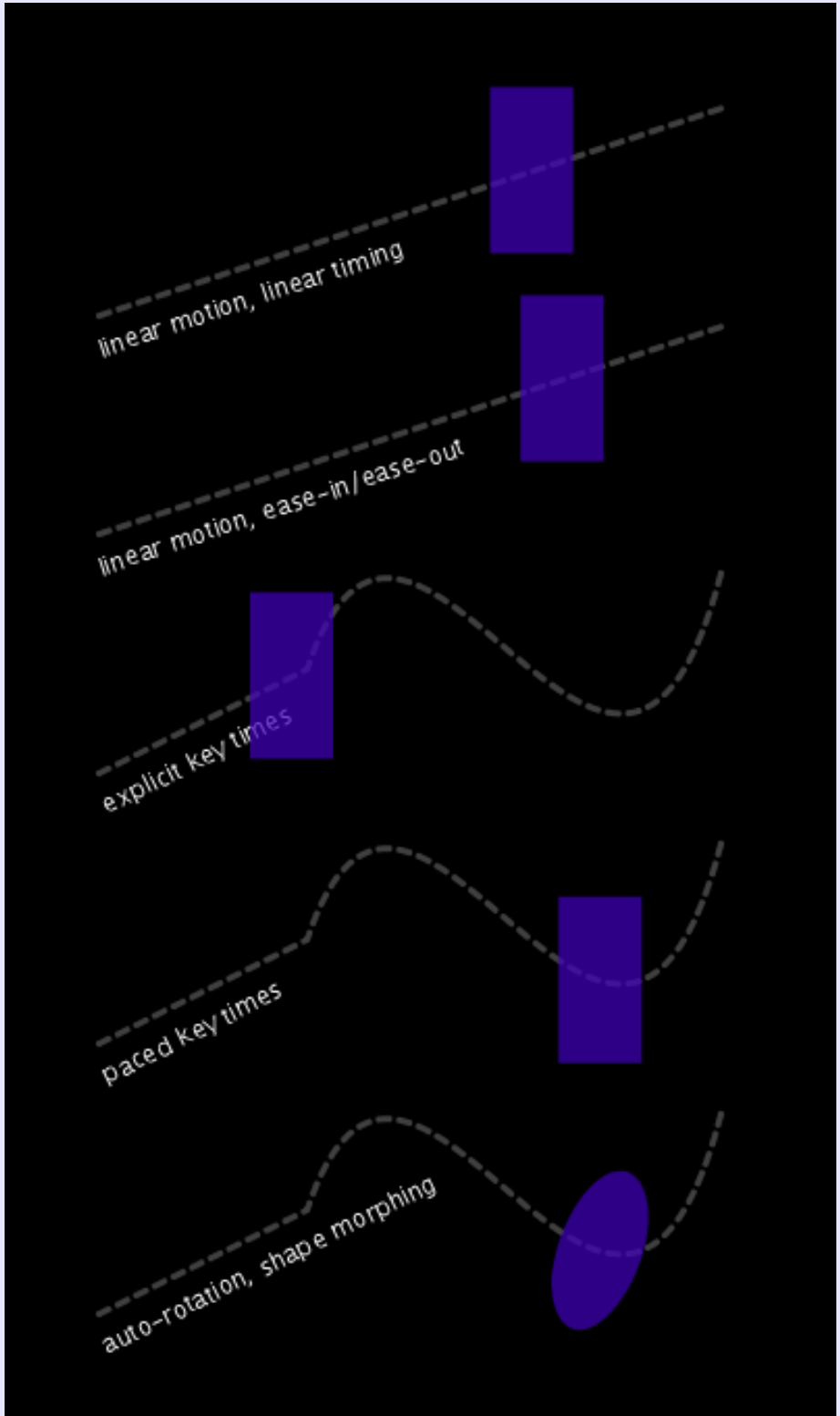
www.int.com/presentations/scenegraph

Demos

- demo.intro.Intro
- demo.fx.TranslateTest
- demo.anim.KeyFramesTest
- demo.alarm.EggTimer

demo.fx.TransformTest





KeyFramesTest

BoxBlurTest

What is a Scene Graph?

- Way to structure API for 2D or 3D graphics
- Objects are retained (kept) for frame after frame
- The scene is built up from a hierarchy of nodes (a graph)
- The system takes care of when to render nodes
- Compare to Immediate mode graphics

Immediate mode Graphics (think Java2D)

- Each API call is a draw command.
- Each stroke is done only once and then forgotten
- System as no memory other than lit pixels on screen

Scene graph or Retained mode (think Swing)

- Each API call creates an object attached
- to a graph
- Each object is kept until explicitly removed
- System draws visible objects whenever necessary

Why use a Scene Graph?

- Simpler to code an application
- Higher level than immediate mode drawLine etc
- Can be faster
  - Can cull a lot of objects that are not visible
- 3D is hard, so a Scene Graph is a huge win for 3D

Resources

Home:
http://scenegraph.dev.java.net/
Build server:
http://openjfx.java.sun.com/hudson/
Effectuation: The Intro:
http://weblogs.java.net/blog/campbell
Introducing The Scene Graph Project:
http://weblogs.java.net/blog/hansmuller

Status of Scene Graph

- Desktop profile is released with JavaFX
- Mobile Environment is coming soon
  - Phone emulator available as an alpha
- License for Scene Graph source is GPL *without* classpath exception
- Scene Graph binary jars are available through JavaFX and its license*

*I am not a lawyer

Future for Scene Graph

Unofficial comments from Chris Campbell

- Scenegraph is
  - an implementation detail for JavaFX
  - is too low level
- Better to have a Java implementation of the JavaFX API

Focus at Sun is on delivering JavaFX

API for Scene Graph is changing

Nuts & Bolts

- Scene Graph depends on effects
- Effects depends on antlr, JOGL, JDK 6u10 and DirectX9

The jar files are
Scene Graph – Scenario.jar
Effects -- Decora-HW.jar,
Decora-SSE.jar,
Decora-OGL.jar
Decora-D3Djar

Visualize the graph?

- TransformTest
- Intro (use control-T)

Both of these are available in the scengraph-demos from scenegraph.dev.java.net

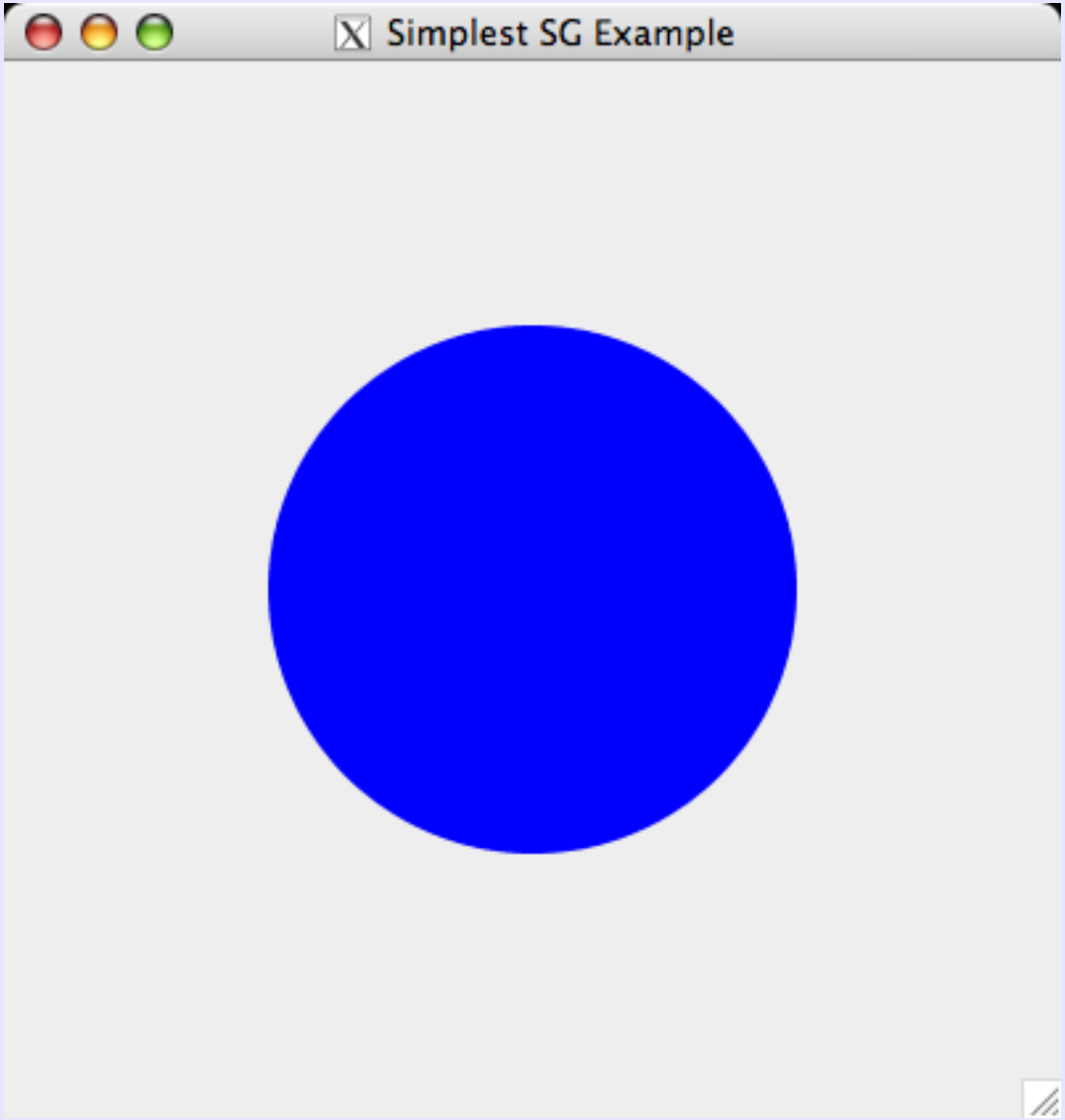
s

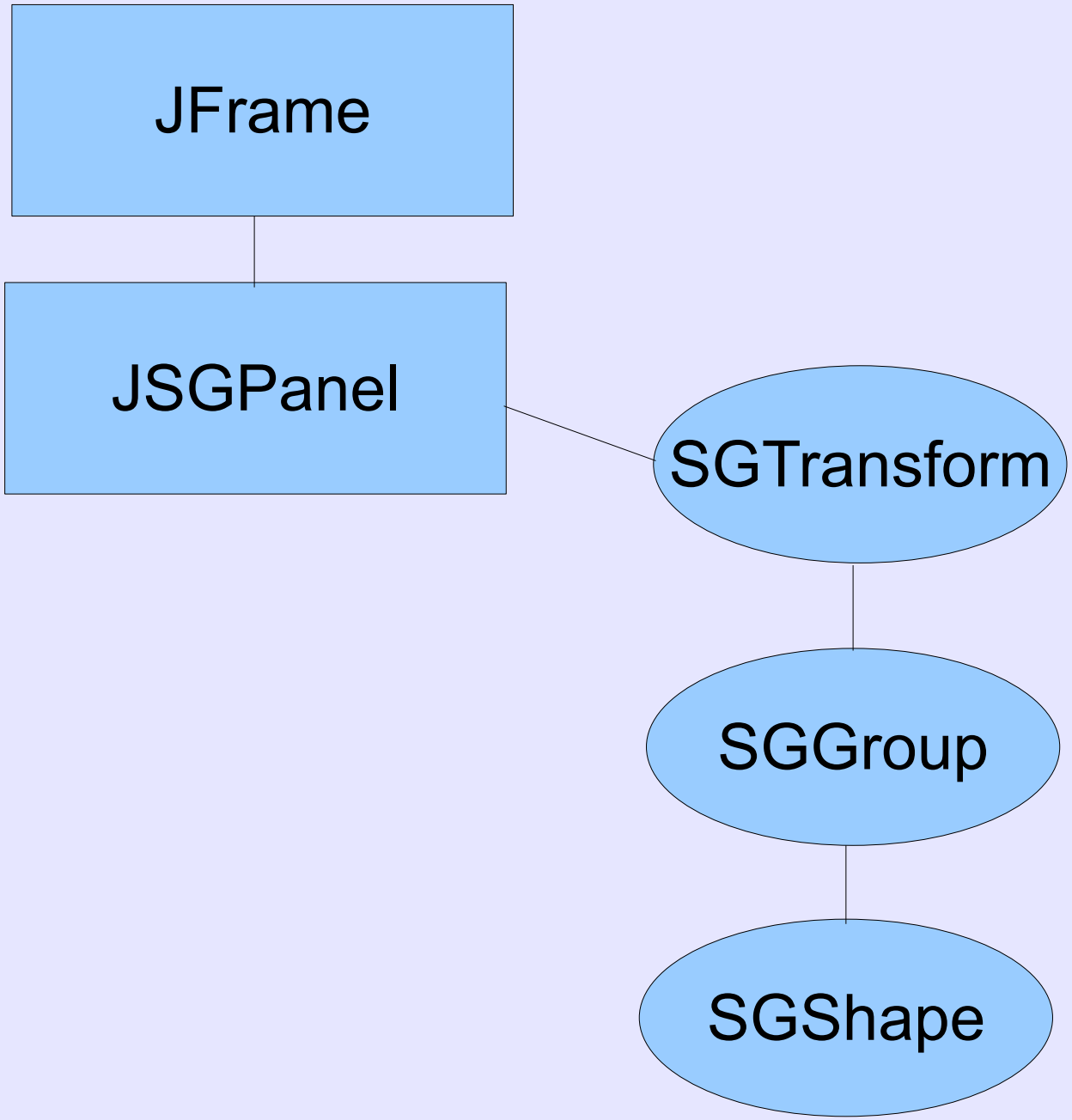
Before Scene Graph

JavaFX started with Jazz.
University of Maryland
http://www.cs.umd.edu/hcil/jazz/
Jazz separates each feature into a separate Java Class/Scene Graph node
Piccolo combines many features into each Java Class/Scene Graph node
Piccolo was built from experiences with Jazz

Jazz and Piccolo

Paper describing Jazz and Piccolo
"Toolkit Design for Interactive Structured Graphics," Bederson, Grosjean, and Meyer
IEEE Transactions on Software Engineering, August 2004.


Simplest SG Example


JFrame
JSGPanel
SGTransform
SGGroup
SGShape

```
public class Simplest {

    static final float layoutUnit = 200f;
    static final float ZOOM = 1.0f;
…

    public static void main(String[] args) {
        SwingUtilities.invokeLater(new Runnable() {

            public void run() {
                createAndShowGUI();
            }
        });
    }
}
```

```
private static void createAndShowGUI() {
  // set UI look and feel
  Simplest app = new Simplest();
  JFrame f = new JFrame("Simplest SG Ex");
  f.setDefaultCloseOperation(
                    JFrame.EXIT_ON_CLOSE);
  f.add(app.createMainPanel(),
                    BorderLayout.CENTER);
  f.pack();
  f.setLocationRelativeTo(null);
  f.setVisible(true);
}
```

```
private JComponent createMainPanel() {
  final JSGPanel panel = new JSGPanel() {
    public Dimension getPreferredSize() {
      int panelSize = (int) (2f * layoutUnit * ZOOM);
      return new Dimension(panelSize, panelSize);
    }
  };
  panel.setScene(createScene());
  return panel;
}
```

```
private SGNode createScene() {
  float x = 0.5f * layoutUnit;
  float y = 0.5f * layoutUnit;
  float size = layoutUnit;
  SGShape circle = new SGShape();
  circle.setShape(new Ellipse2D.Float(x, y, size,
size));
  circle.setFillPaint(Color.BLUE);
  SGGroup scene = new SGGroup();
  scene.add(circle);
  SGTransform.Scale zoomNode;
  zoomNode = SGTransform.createScale(1.0, 1.0,
null);
  zoomNode.setChild(scene);
  return zoomNode;
}
```

Scene Graph

- Packages:

com.sun.scenario.animation
com.sun.scenario.scenegraph
com.sun.scenario.scenegraph.event
com.sun.scenario.scenegraph.fx

...and effects...

Key Classes

Jazz

- SGNode
  - Lean and mean. Each node is very simple
  - Position is a separate node in the scene graph
  - Easier to extend
- FXNode
  - Full featured
  - Position etc are attributes
  - Subclass of SGNode
  - Used by JavaFX
  - Easier to program

Piccolo

Why choose when you can have both?

SGNode

Methods
- render
- globalToLocal(point)
- pick(point)

Attributes
- Object parent
- Map<String, Object> attributeMap
- List<SGNodeListener> nodeListeners
- List<SGMouseListener> mouseListeners

Hierarchy of SGNode

SGNode
- SGParent
  - SGFilter
    - SGClip, SGComposite, SGEffect, SGImageOp, SGRenderCache, SGTransform
  - SGGroup, SGWrapper
  - SGLeaf (has paint method like a swing comp.)
  - SGComponent (wrapper for a swing component)
  - SGImage
  - SGAbstractShape
    - SGText
    - SGAbstractGeometry
      - SGArc, SGCircle, SGCubicCurve, SGEllipse, SGLine, SGQuadCurve, SGRectangle, SGShape (takes a java.awt.Shape)

FXNode

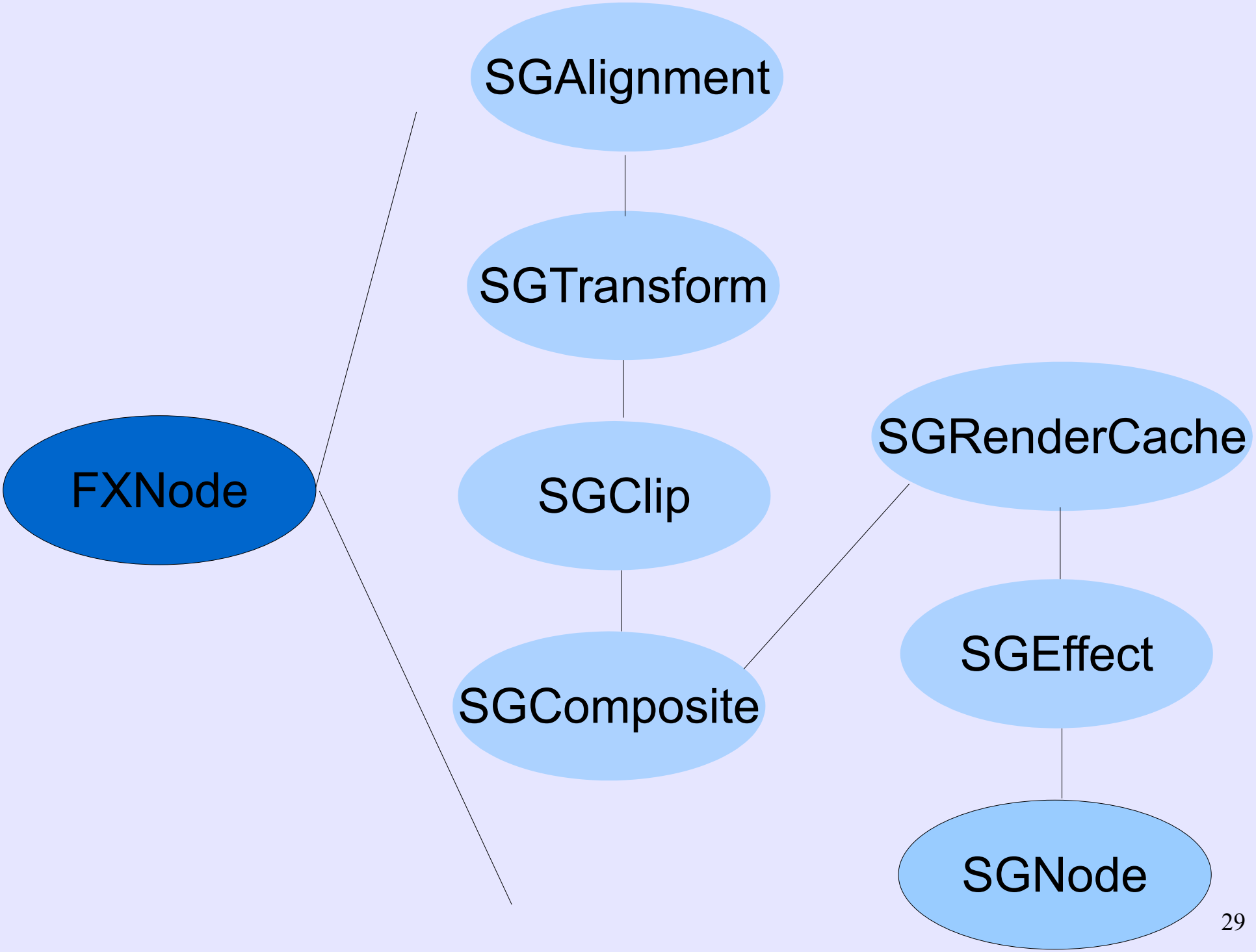
SGWrapper - Base class for nodes that maintain an internal graph of nodes.

- SGWrapper
  - FXNode
    - FXAbstractShape
      - FXShape
      - FXText
    - FXComponent
    - FXGroup
    - FXImage

FXNode

Attributes
SGAlignment, SGTransform.Affine, SGTransform.Translate, SGClip, SGComposite, SGRenderCache, SGEffect, and leafNode:SGNode

- Each is optional except leafNode
- All are descendents of SGFilter except SGNode


SGAlignment
SGTransform
FXNode
SGClip
SGRenderCache
SGComposite
SGEffect
SGNode

FXText

FXText is

SGText plus attributes from FXAbstractShape

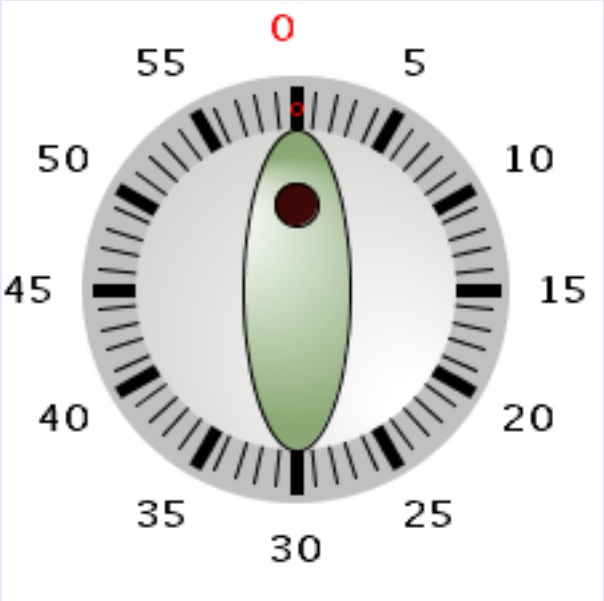
FXAbstractShape

FXAbstractShape is

SGAbstractShape plus attributes from FXNode
(the shape is the leaf for FXNode)

EggTimer

- The scene graph
- A Clip
- Some mouse handlers



EggTimer Scene Graph

EggTimer.makeScene

root: SGGroup
  g: SGGroup
    ocircle: FXShape (white)
    mcircle: FXShape (lightGray)
    icircle: FXShape (0xC7C9C6) diskFillColor
    major ticks (method createTicks)
    minor ticks
    knob (method createKnob)
    referenceDot (method createReferenceDot)
    labels (method createLabels)

EggTimer Clip


```
private Clip time;
...
time = Clip.create(Clip.INDEFINITE,
                   Clip.INDEFINITE,
                   new TimingTargetAdapter() { ... });
time.pause();
```

```
new TimingTargetAdapter() {
public void timingEvent(float t, long totalElapsed) {
...
  // update twice per second.
  long now = System.nanoTime();
  if (TimeUnit.NANOSECONDS.toMillis(now -
lastUpdate) > 500) {
     // moves 6 degree per minute
    double dt =
TimeUnit.NANOSECONDS.toMillis(now - start);
    dt = (6 / (double) 60000) * dt;
    rotateBy(-1 * Math.toRadians(dt));
    start = System.nanoTime();
    lastUpdate = start;
    // check for alarm end. Play sound, change led etc
```

```
kg.addMouseListener(new SGMouseAdapter() {

  int sx = -1;
  int sy = -1;

  public void
  mouseDragged(java.awt.event.MouseEvent e,
  SGNode node) {
    int cx = e.getX();
    int cy = e.getY();
    double theta = Math.atan2(r - cy, r - cx) -
  Math.atan2(r - sy, r - sx);
    rotateBy(theta);
     sx = cx;
     sy = cy;
  }
```

Top level node for the Knob

Mouse handler for Knob

Mouse handler for Knob

```
 ...
public void
  mouseReleased(java.awt.event.MouseEvent e,
  SGNode node) {
    led.setFillPaint(ledColors[1]);
    sx = -1;
    sy = -1;
    time.start();
  }
});
```

EggTimer.rotateBy

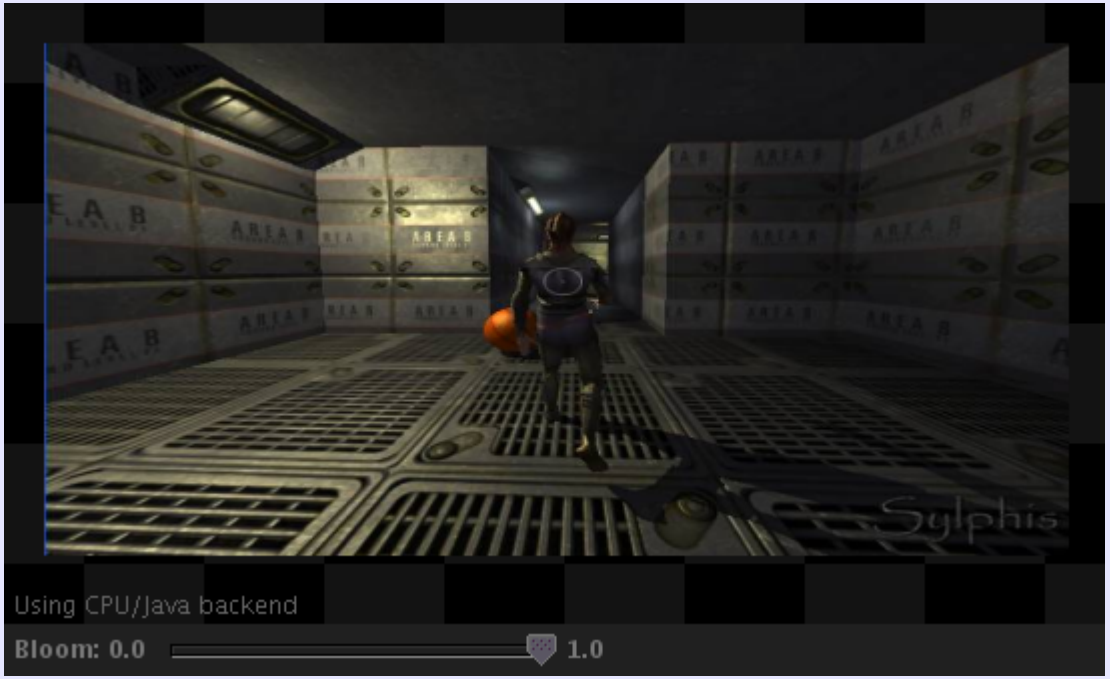
```
private void rotateBy(double theta) {
    AffineTransform at = new AffineTransform();
    at.rotate(theta, r, r);
    knobAffine.transformBy(at);
    refAffine.transformBy(at);
}
```

Effects

- What's possible? Essentially a filter on raster images
- How do they work?
  - Use shader program on the video card
  - SSE instruction on CPU

Effects

- packages
com.sun.scenario.effect
com.sun.scenario.effect.light


Using CPU/Java backend
Bloom: 0.0
1.0

BloomTest


Using CPU/Java backend
Bloom: 0.0
1.0

DropShadowTest

InnerShadowTest

PhongLightingTest

Acceleration Types

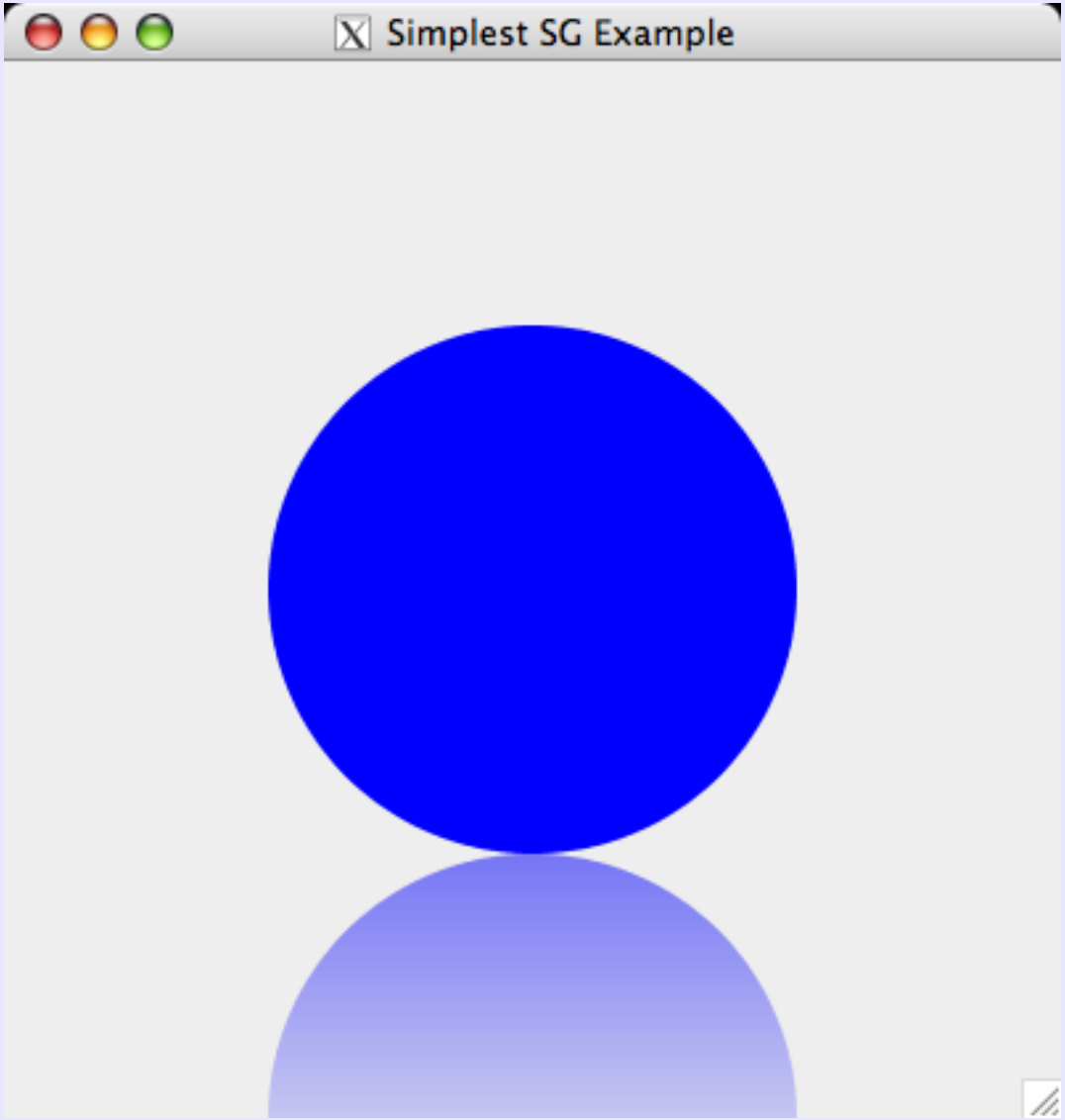
- Effect.AccelType enum constants:
  - On CPU: NONE, FIXED, SIMD
  - On GPU: DIRECT3D, OPENGL
- Blend modes:ADD,DARKEN, DIFFERENCE, EXCLUSION,... and many more...

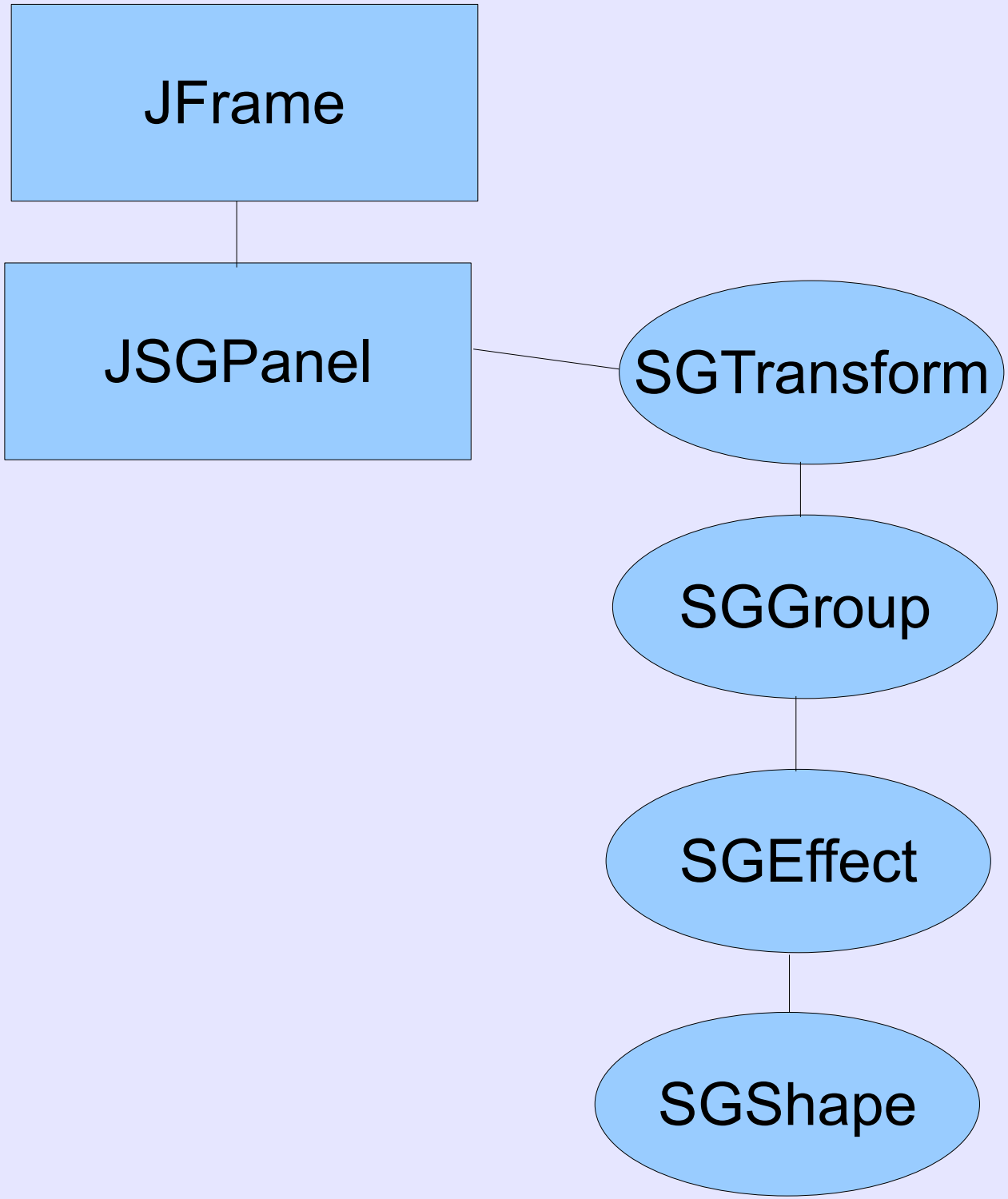
Effects

Blend Bloom BoxBlur Brightpass ColorAdjust Crop Defocus DisplacementMap DropShadow Flood GaussianBlur Glow InnerShadow InvertMask Merge MotionBlur Offset PerspectiveTransform PhongLighting Reflection SepiaTone Shadow ZoomRadialBlur

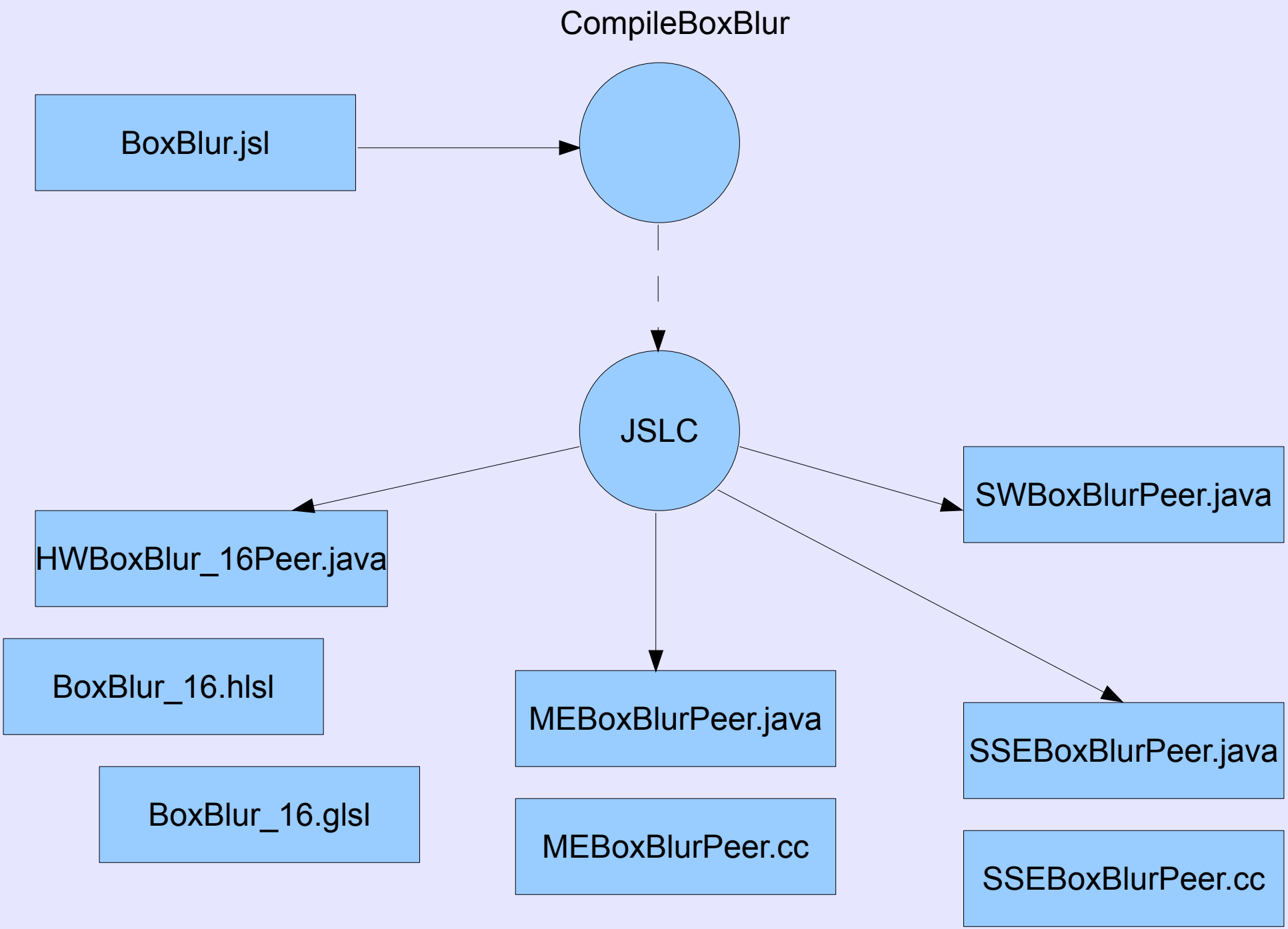
Adding an Effect

- Insert an SGEffect node into the graph, or
- FXNode.setEffect(Effect)

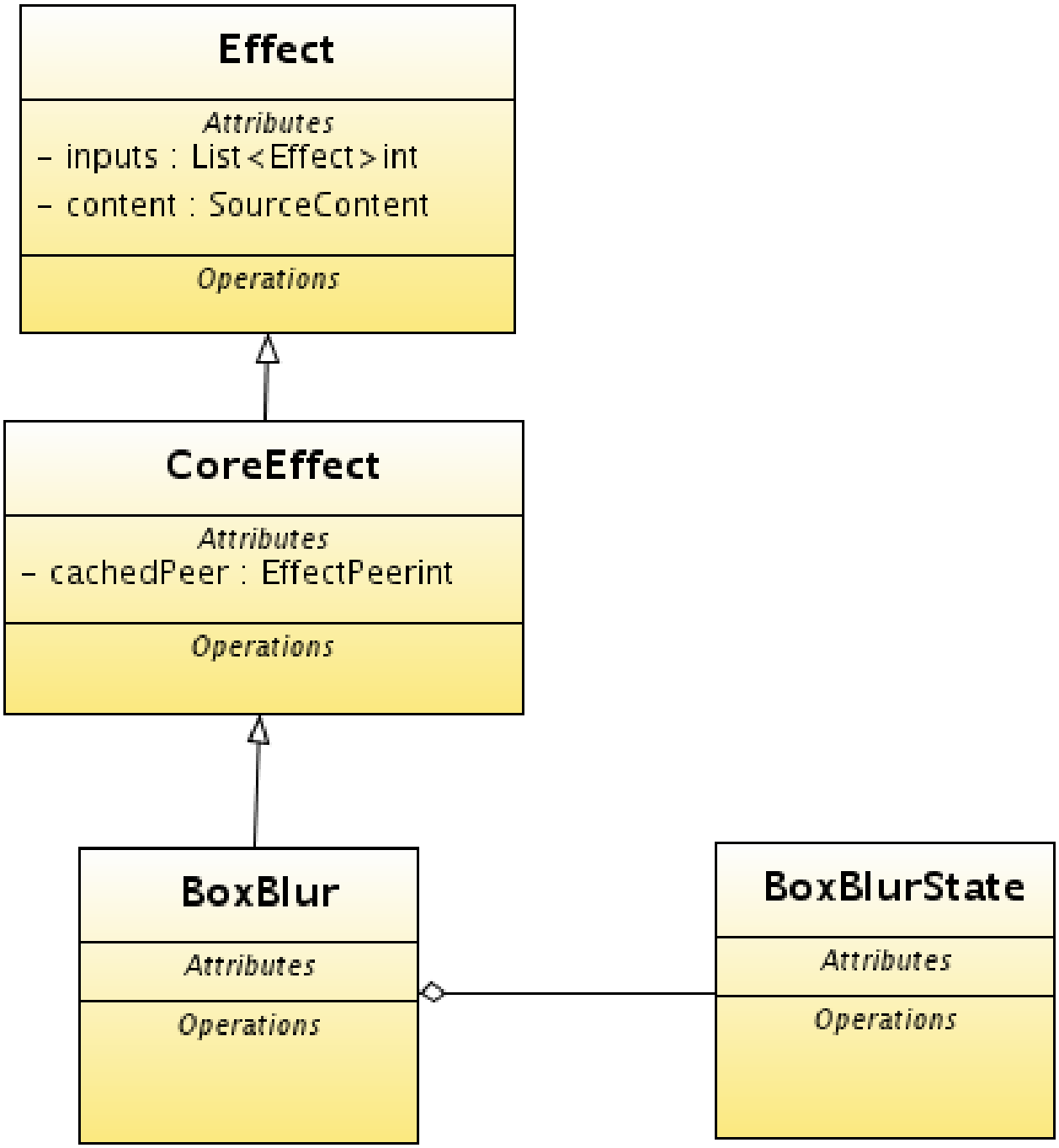

Simplest SG Example

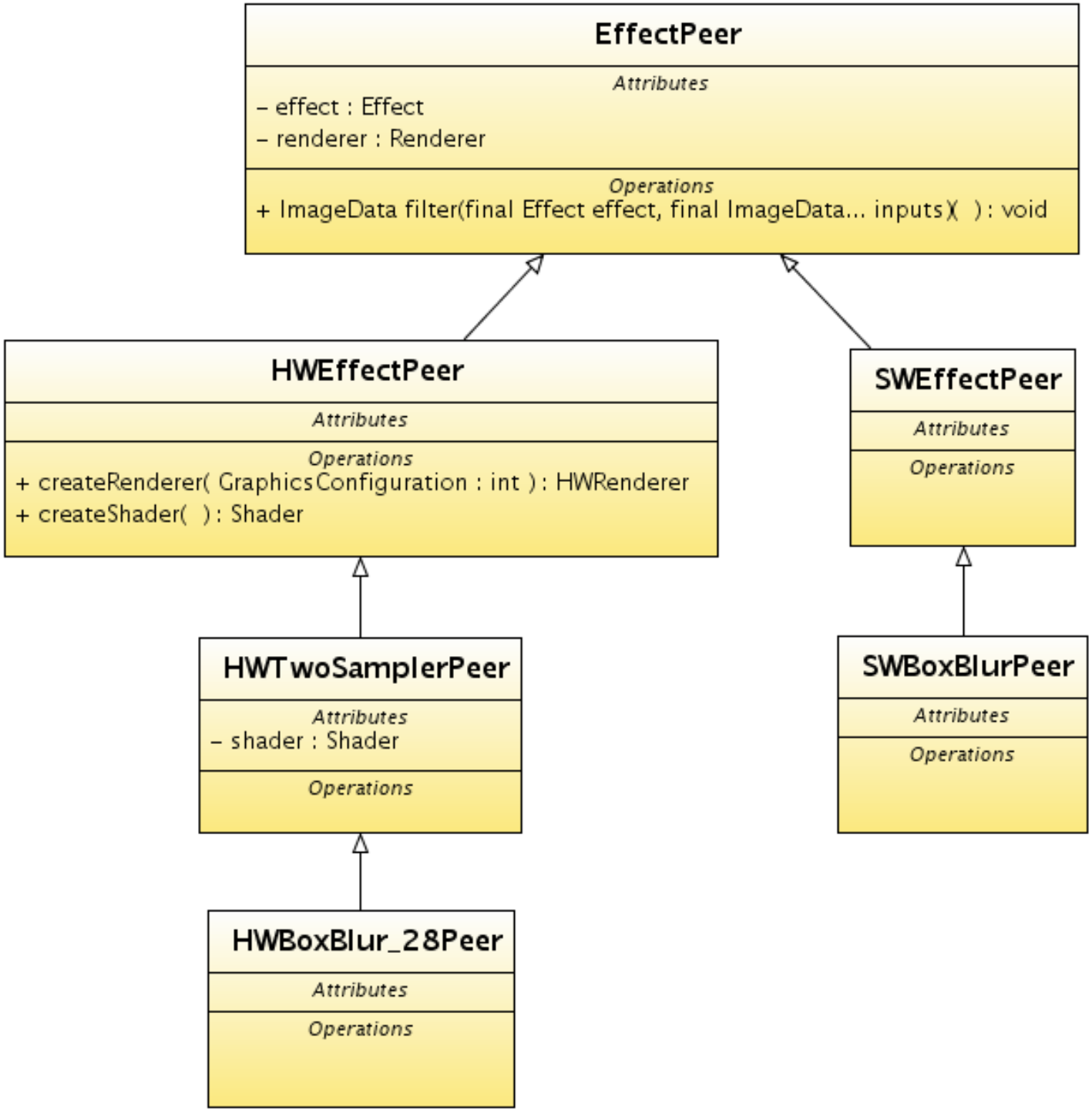

JFrame
JSGPanel
SGTransform
SGGroup
SGEffect
SGShape

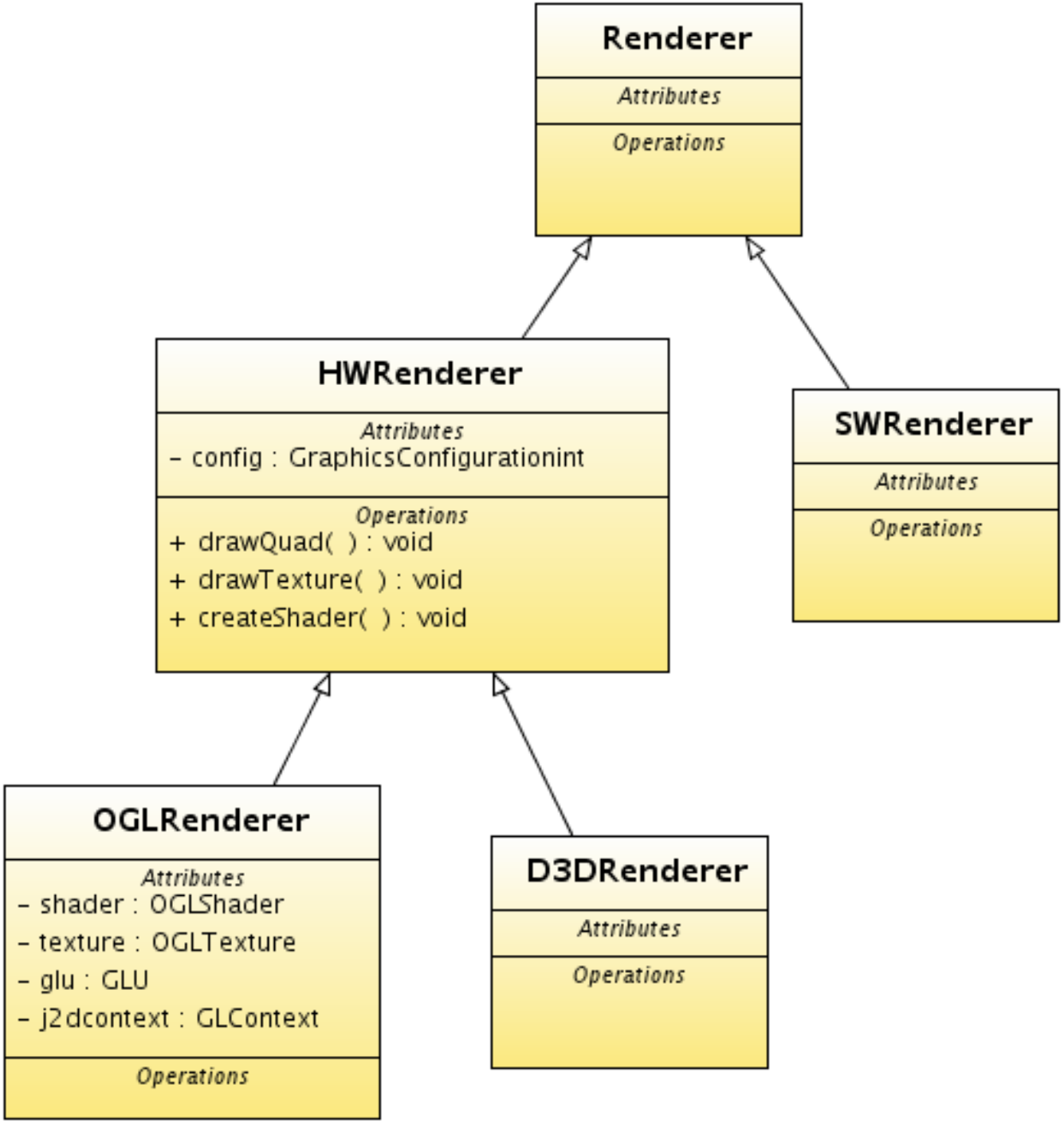
```
        circle.setShape(new Ellipse2D.Float(x, y,
size, size));
        circle.setFillPaint(Color.BLUE);
+        SGEffect effect = new SGEffect();
+        Effect myEffect = new Reflection();
+        effect.setEffect(myEffect);
+        effect.setChild(circle);
        SGGroup scene = new SGGroup();
-       scene.add(circle);
+        scene.add(effect);
        SGTransform.Scale zoomNode;
        zoomNode =
SGTransform.createScale(1.0, 1.0, null);
        zoomNode.setChild(scene);
```



Generates 56 files


Effect
Attributes
- inputs : List<Effect>int
- content : SourceContent
Operations
CoreEffect
Attributes
- cachedPeer : EffectPeerint
Operations
BoxBlur
Attributes
Operations
BoxBlurState
Attributes
Operations


EffectPeer
Attributes
– effect : Effect
– renderer : Renderer
Operations
+ ImageData filter(final Effect effect, final ImageData... inputs)( ) : void
HWEffectPeer
Attributes
Operations
+ createRenderer(GraphicsConfiguration : int) : HWRenderer
+ createShader( ) : Shader
SWEffectPeer
Attributes
Operations
HWTwoSamplerPeer
Attributes
– shader : Shader
Operations
SWBoxBlurPeer
Attributes
Operations
HWBoxBlur_28Peer
Attributes
Operations


Renderer
Attributes
Operations
HWRenderer
Attributes
- config : GraphicsConfigurationint
Operations
+ drawQuad( ) : void
+ drawTexture( ) : void
+ createShader( ) : void
SWRenderer
Attributes
Operations
OGLRenderer
Attributes
- shader : OGLShader
- texture : OGLTexture
- glu : GLU
- j2dcontext : GLContext
Operations
D3DRenderer
Attributes
Operations

Platform support

- Windows XP or Vista with Java SE 1.6 Update 10
- Mac OS X with Java SE 1.6

OpenGL Pipeline

- Enable opengl pipeline with (since Java 5)

  -Dsun.java2d.opengl=True

  -Djava.library.path=<pathToJOGL>
- First option turns on the opengl pipeline
- Second option lets JVM find JOGL libraries (needed by Effect's)

D3D Pipeline

- JDK 6u10 has a new Direct3D 9 hardware-accelerated pipeline.
  - Enabled by default on all hardware with Pixel Shaders 2.0 support.
  - Verify by `c:\> set J2D_TRACE_LEVEL=4`
    `c:\> java YourApp`

Shaders

- Video cards before shaders were
  - Fixed processing pipeline
  - Some configuration parameters
  - Controlled by a delimited API
- Shaders
  - Can send arbitrary code into the video card
  - Vertex or pixel shaders

Real-Time Rendering 3rd ed. Chapter 3

Why Shaders?

- Without shaders, all objects looked like plastic
- With shaders, damn near any type of surface is possible
- And as a bonus, the video card becomes a programmable resource that can be used for...
  - Acceleration of 2D effects
  - 2D visualization or plotting
  - Whatever else you can think of

GLSL Example

```
varying vec3 normal;
varying vec3 vertex_to_light_vector;
void main() {
    // Defining The Material Colors
    const vec4 AmbientColor = vec4(0.1, 0.0, 0.0, 1.0);
    const vec4 DiffuseColor = vec4(1.0, 0.0, 0.0, 1.0);

    // Scaling The Input Vector To Length 1
    vec3 normalized_normal = normalize(normal);
    vec3 normalized_vertex_to_light_vector =
normalize(vertex_to_light_vector);

    // Calculating The Diffuse Term And Clamping It To 0-1
    float DiffuseTerm = clamp(dot(normal, vertex_to_light_vector),
0.0, 1.0);

    // Calculating The Final Color
    gl_FragColor = AmbientColor + DiffuseColor * DiffuseTerm;
}
```

Summary

Scenegraph is

- Nodes FXNode (full featured) and SGNode (lean and mean)
- Clip for animation
- Events (mouse handlers)
- Effects

Acceleration?

- Ordinary drawing ops by the graphics pipeline
- Effects by shader, SSE, or...

Thank You

Thanks to the team at Sun that created the Scenegraph & Effects

- Chris Campbell
- Hans Muller
- Chet Haase
- Igor Kushnirskiy
- bchristi
- flar
- everyone else...

Thanks to the SoyLatte project (open source java on the mac)