

WELCOME  
TO  
**JAVAPOLIS**



# The Scene Graph Project

---

Chet Haase  
Java Client Architect  
Sun Microsystems



- Overview
- Architecture
- API
- Code

- Overview
- Architecture
- API
- Code

- What?
- Why?
- When?

# What is a Scene Graph?

- Retained-mode view of a graphical scene
- Swing is a scene graph
  - Although component positioning is limited to position and size
- ... Java 2D is not
  - 2D is an “immediate mode” API
- Scene graphs can make creating a graphical application easier
  - Declare where you want things
  - Let the system figure out how to draw them

# DEMO

---

## Scene Graph



# What is in *this* Scene Graph?

- Nodes
  - Graphics, text, components, images
- State
  - Transforms, effects, visibility, ...
- Animation
  - Varying properties over time

- JavaFX Script uses a scene graph model

```
Frame {  
    title: "Circle Demo"  
    content: Canvas {  
        content: [  
            Circle {  
                cx: 100  
                cy: 100  
                radius: 100  
                fill: Color { blue: 0.8 }  
            }  
        ]  
    }  
    visible: true  
}
```

- It relies on a scene graph underneath
  - Jazz library in the prototype
  - “Jazz is dead”
    - This scene graph replaces Jazz
- But this scene graph is not just for Java FX
  - Also good for Desktop Java developers

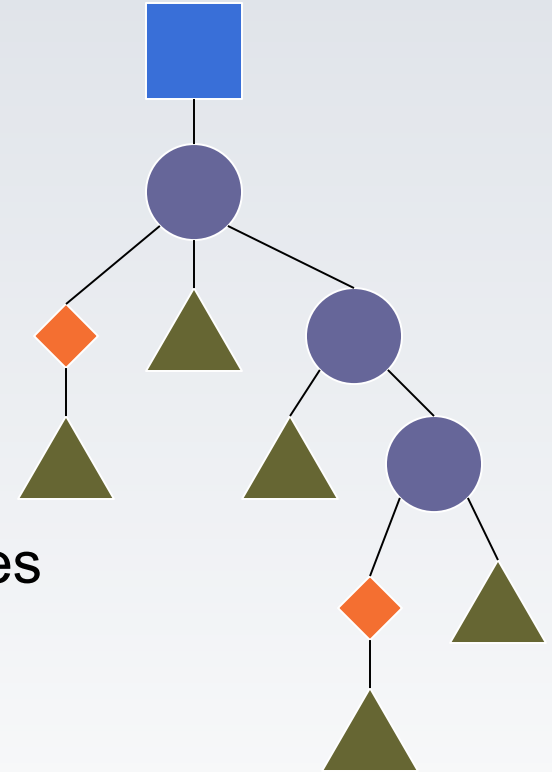
## When Will It Be Available?

- Announcing today!
- Open source project available now
  - <http://scenegraph.dev.java.net>
- Early early (early) access bits
  - Functional
  - Not final APIs
- Beta planned for early 2008

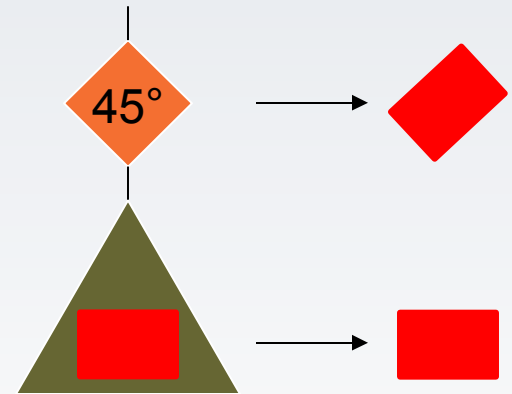
- Overview
- **Architecture**
- API
- Code

- Nodes
- State
- Animation

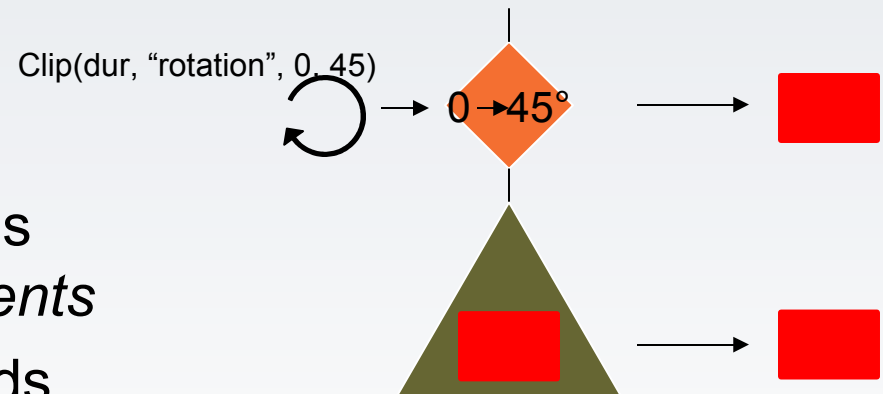
- SGNode base class
- Leaves and Parents
  - SGLeaf, SGParent
- ▲ SGLeaf subclasses
  - SGImage: BufferedImages
  - SGShape: Geometry display
  - SGText: Stroked/filled text
  - SGComponent: Swing component trees
- SGParent subclasses
  - SGGroup: parent of multiple children
  - ◆ SGFilter: state filtering of single child
- JSJPanel: Swing container of scene



- Filter nodes affect child that they parent
  - Plus any children of that [group] node
- Example filters
  - SGTransform
    - scale, translate, rotate, or AffineTransform
  - SGComposite
    - translucency
  - SGEffect
    - glow, blur, shadow,...



- Varying properties over time
- With interesting behaviors
  - Non-linear interpolation
  - Repeating/reversing animations
  - Chained animations
- TimingFramework++
  - TimingFramework library, as described in *Filthy Rich Clients*
  - Scene Graph animation adds timelines, single heartbeat, some refactoring



# DEMO

---

Animation

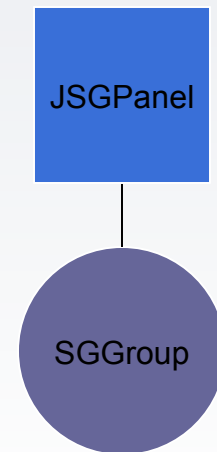


- Overview
- Architecture
- **API**
- Code

## Creating the Scene

- Create JSGLPanel (Swing component)
- Create a root node for the scene
- Set the root node as the scene
- Add the panel to the JFrame

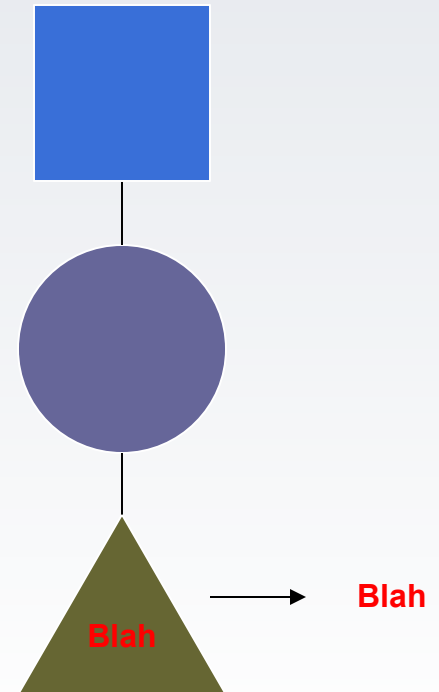
```
JFrame f = new JFrame("Demo");  
panel = new JSGLPanel();  
SGGroup rootNode = new SGGroup();  
panel.setScene(rootNode);  
f.add(panel);
```



## Creating a Node

- Create the node
- Set its content and state
- Add it into the scene

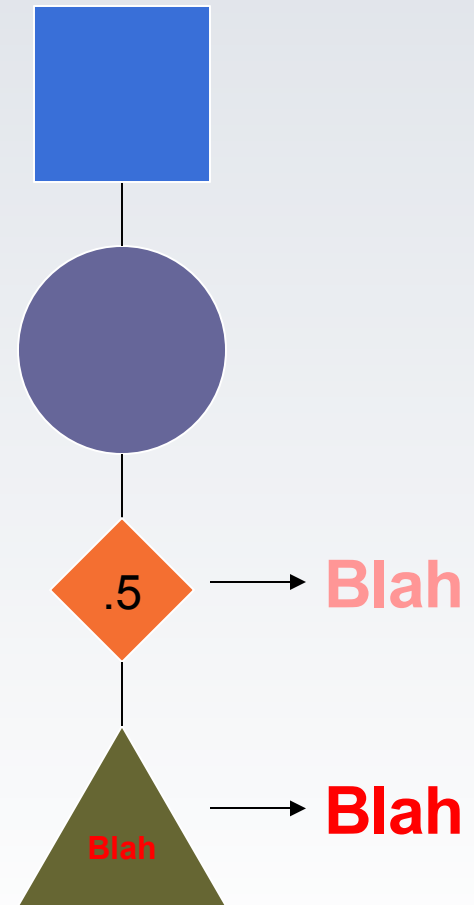
```
SGText textNode = new SGText();  
textNode.setText("Blah");  
textNode.setMode(SGText.FILL);  
textNode.setPaint(Color.RED);  
rootNode.add(textNode);
```



# Set State Filters

- Create filter node
- Set its state
- Parent the child that it affects
- Add filter to the scene

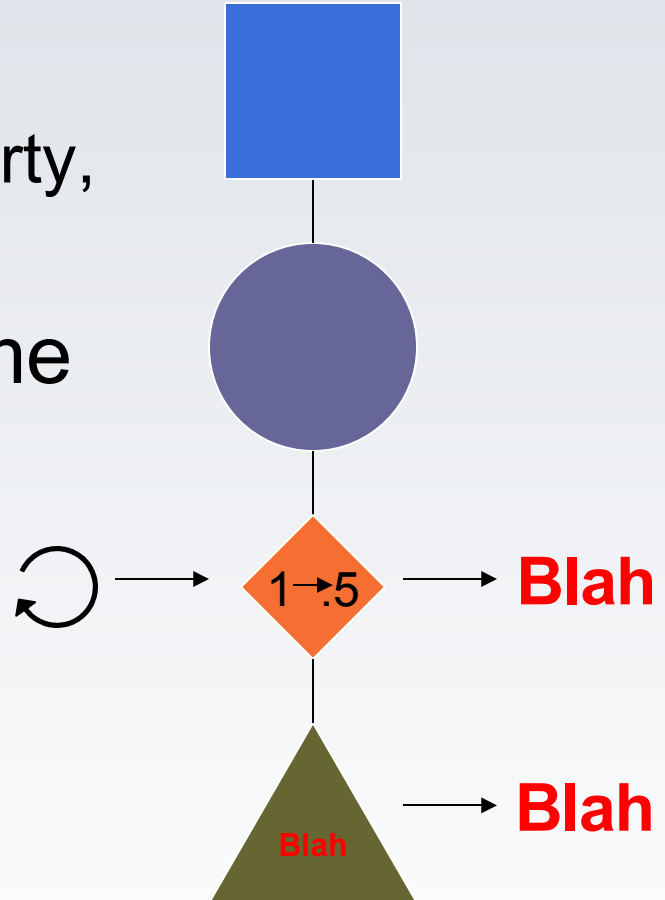
```
// Make textNode translucent
SGComposite comp =
    new SGComposite();
comp.setOpacity(.5f);
comp.setChild(textNode);
rootNode.add(comp);
```



# Animate a Node's Property

- Create an animation Clip
  - Set the duration, object/property, and values to be animated
- Schedule the Clip's start time

```
// Fade comp filter over 1 sec
Clip fader = Clip.create(
    1000,          // duration
    comp,         // animating obj
    "opacity",    // animating prop
    1f,           // "from" value
    .5f);         // "to" value
fader.start();
```



- Nothing to do!
- A visible scene knows how to draw itself

- Overview
- Architecture
- API
- Code

# DEMO

---

Scene Graph Demos  
and  
Code



- Overview
- Architecture
- API
- Code
- In Progress Items

- Easy creation of canonical, chainable effects

```
Effect effect = new DropShadow(textNode) ;  
rootNode.add(new SGEffect(effect) ) ;
```

- GPU acceleration via pixel shaders

- Sample effects:

- drop shadow

- blur

- spotlight

- diffuse lighting

- specular lighting

- bloom

- blend modes

- dodge

- burn

- multiply

- add

- more...

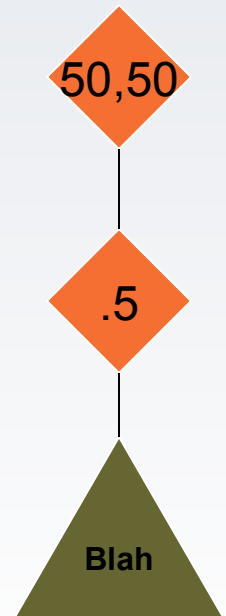
- HSV adjustment

- sepia tone

- ... and more!

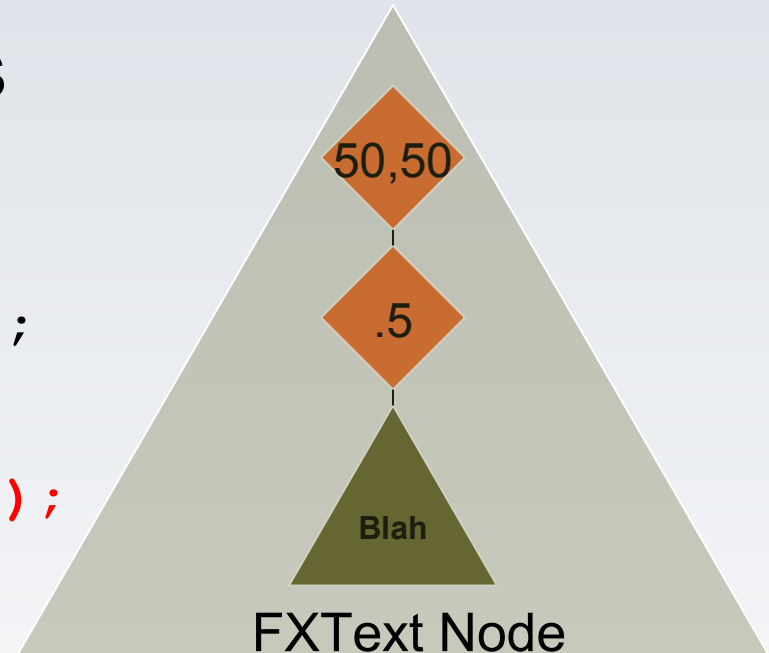
- Filter nodes add up
  - Fading: add an SGComposite
  - Translating: add an SGTransform.Translate
  - Effects: add an SGEffect
- Example: translated/faded text node

```
SGText textNode = new SGText();  
textNode.setText("Blah");  
SGComposite comp = new SGComposite();  
comp.setOpacity(.5f);  
comp.setChild(textNode);  
SGTransform.Translate = SGTransform.  
    createTranslation(50, 50, comp);
```



- Solution: Wrapper nodes that manage state

```
FXText textNode = new FXText();  
textNode.setText("Blah");  
textNode.setOpacity(.5f);  
textNode.setTranslation(50, 50);
```



- These wrapper nodes correspond to the nodes in Java FX Script

- Media
  - Video and audio
- HTML
- 3D

- Check out the open source project
  - <http://scenegraph.dev.java.net>
- And the Java FX project which uses it
  - <http://openjfx.dev.java.net>
- Expect the scene graph API to change in the near future
  - More features
  - More content types
  - Faster
  - Easier to use

**Go Write Cool Stuff!**

# Q&A

---

View JavaPolis talks @ [www.parleys.com](http://www.parleys.com)



Thank you for your  
attention

---

