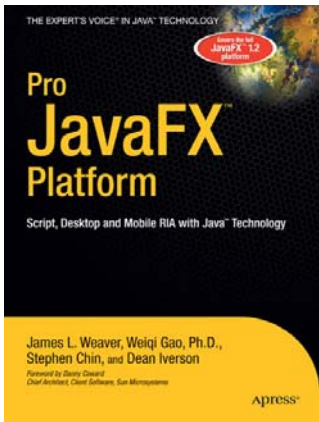


JFXtras

Utilities and Add-ons for the JavaFX™ Platform

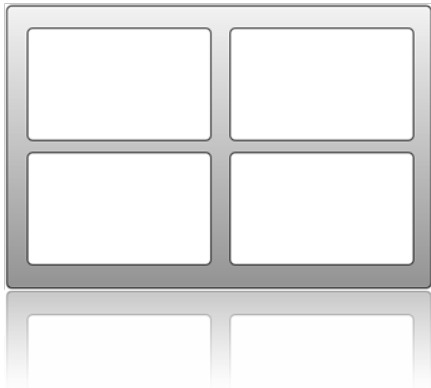


Supercharging Your JavaFX Programs with WidgetFX and JFXtras

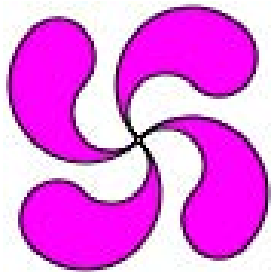
Stephen Chin
Inovis, Inc.

Keith Combs
Inovis, Inc.

JFXtras – Utilities and Add-ons for JavaFX



Layouts



Shapes



Borders



Controls

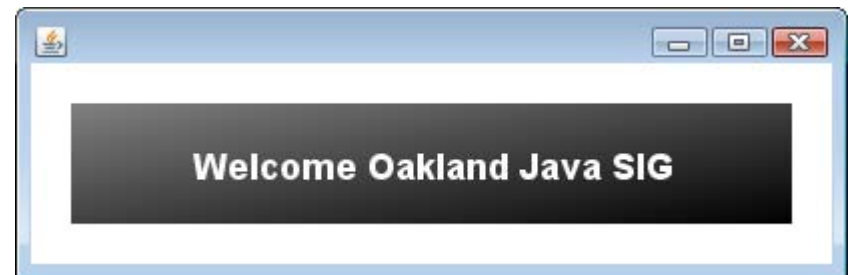
JFXtras Layouts – Before and After

```

var scene:Scene;
scene = Scene {
  width: 400
  height: 100
  var text:Text;
  content: [
    Rectangle {
      x: 20
      y: 20
      width: bind scene.width - 40
      height: bind scene.height - 40
      fill: LinearGradient {stops: [
        Stop {color: Color.GRAY}
        Stop {offset: 1, color: Color.BLACK}
      ]}
    }
  ]
  text = Text {
    content: "Welcome Oakland Java SIG"
    textOrigin: TextOrigin.TOP
    translateX: bind (scene.width -
      text.layoutBounds.width) / 2
    translateY: bind (scene.height -
      text.layoutBounds.height) / 2
    font: Font.font(null, FontWeight.BOLD, 18)
    fill: Color.WHITE
  }
} ] }
  
```

```

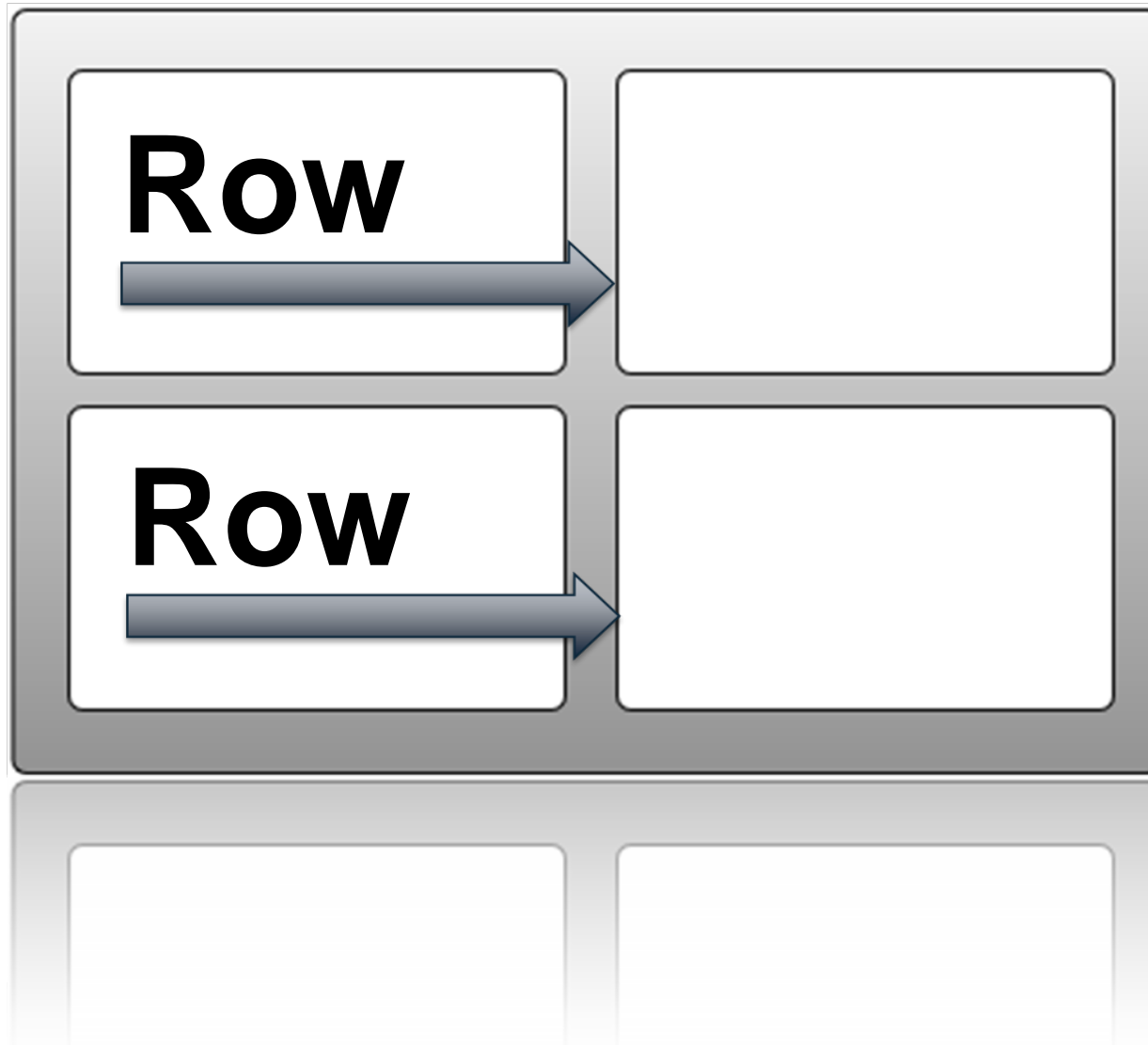
ResizableScene {
  width: 400
  height: 100
  content: EmptyBorder {
    borderWidth: 20
    node: Deck {
      content: [
        ResizableRectangle {
          fill: LinearGradient {stops: [
            Stop {color: Color.GRAY}
            Stop {offset: 1, color: Color.BLACK}
          ]}
        }
        Text {
          content: "Welcome Oakland Java SIG"
          font: Font.font(null, FontWeight.BOLD, 18)
          fill: Color.WHITE
          layoutInfo: LayoutInfo {hpos: CENTER}
        }
      ]
    }
  }
}
  
```



Media Explorer Example

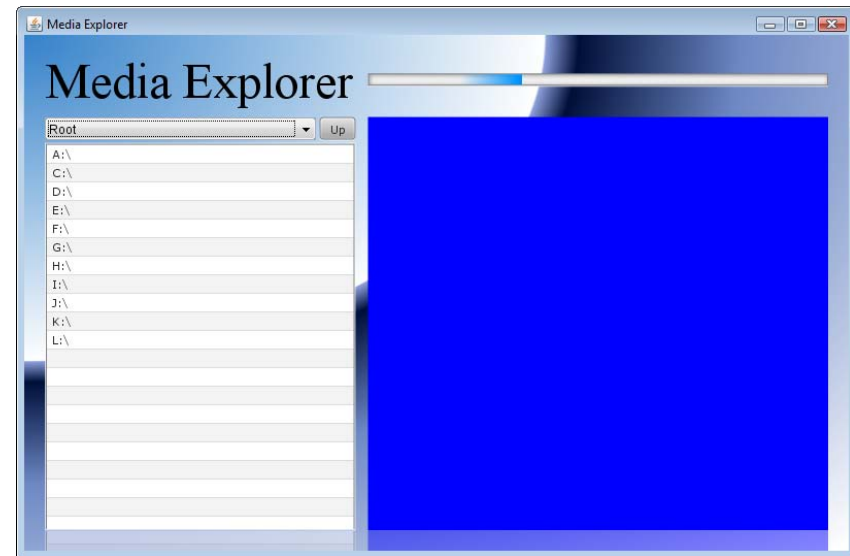


JFXtras Grid

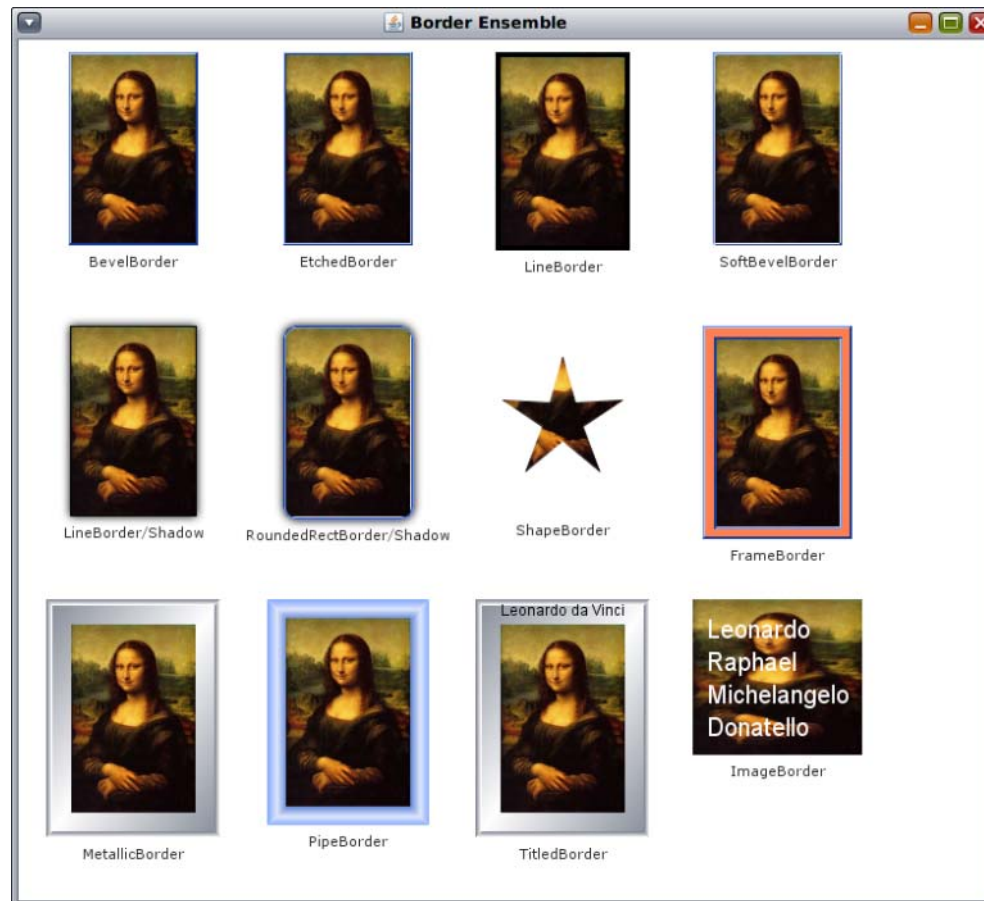


JFXtras Grid

```
Grid {  
    effect: Reflection {}  
    border: 20  
    vgap: 12  
    hgap: 12  
    rows: bind [  
        row([text, progressBar]),  
        row([navigator, mediaGrid])  
    ]  
}
```



JFXtras Borders



JFXtras Borders

Function:

```
override function create() {
  TitledBorder {
    id: "imageTitle"
    title: file.getName()
    content: FrameBorder {
      id: "imageFrame"
      node: ResizableImageView {
        preserveRatio: true
        smooth: true
        image: bind image
      }
    }
  }
}
```

CSS:

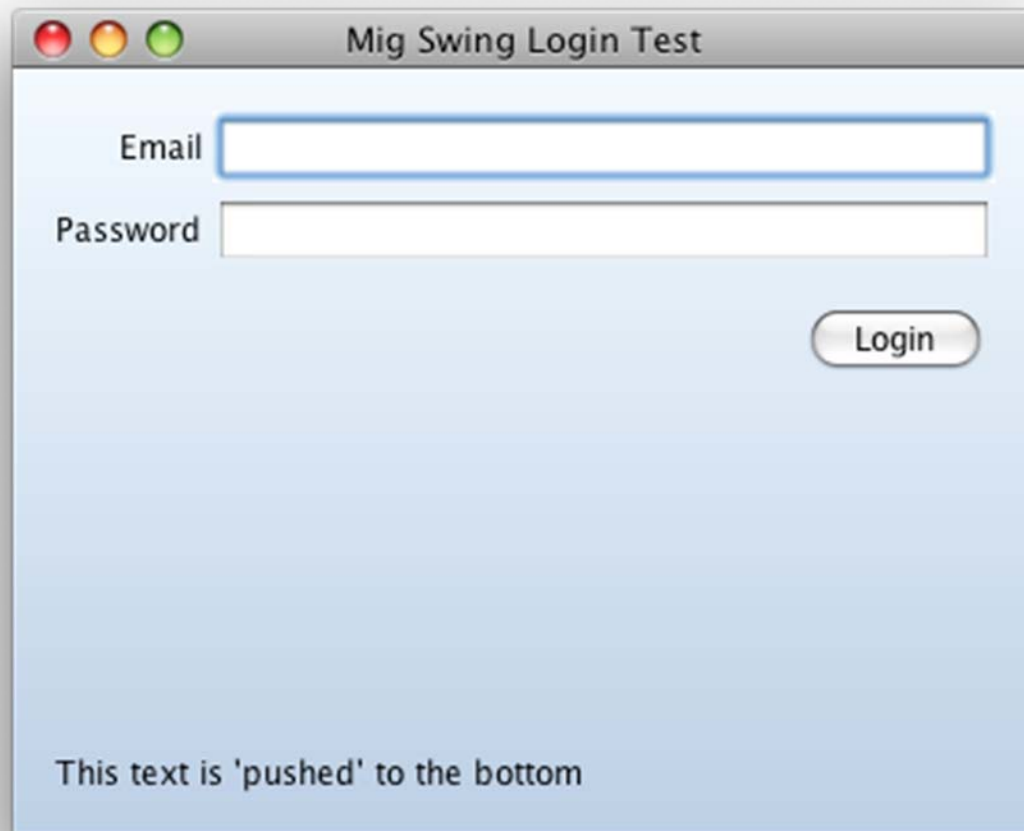
```
#imageTitle {
  position: "bottom";
  justification: "center";
  font: bold 12pt Serif;
  text-color: #000060;
}

#imageFrame {
  border-left-width: 12;
  border-top-width: 12;
  border-bottom-width: 20;
  border-right-width: 12;
  background-fill:
    #00007020;
}
```

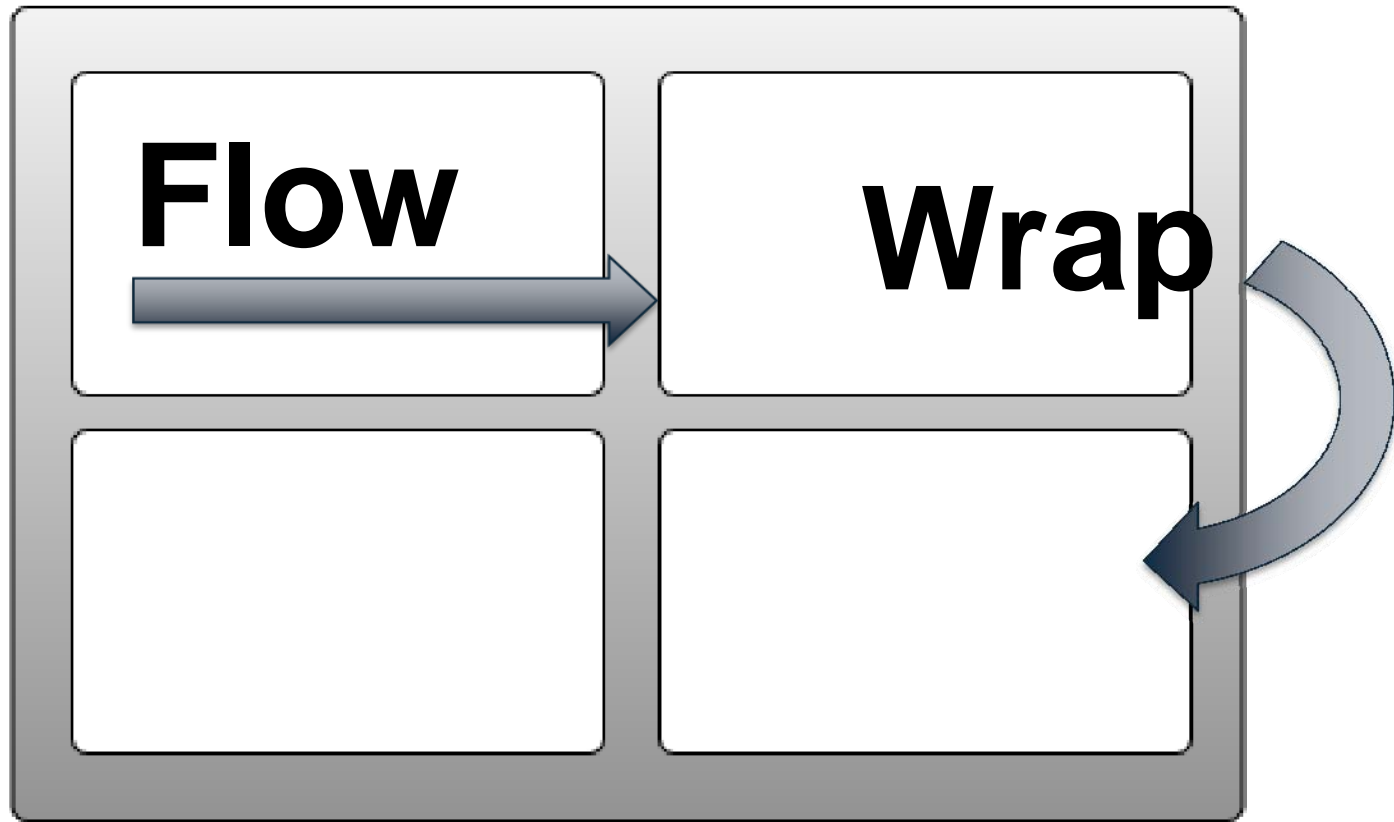
JFXtras Borders



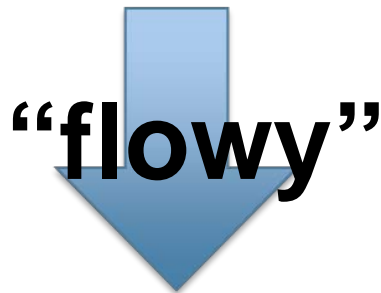
MigLayout for JavaFX



Flexible Grid-Based Layout

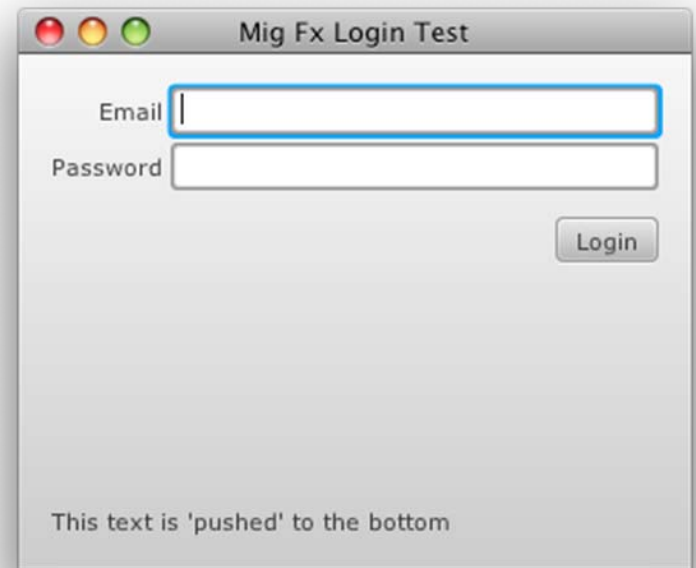


MigLayout Constraints

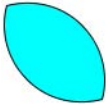
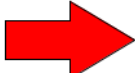








```
MigLayout {  
    constraints: “fill, wrap”  
  
    // to be continued  
}
```

```
MigLayout {
  constraints: "fill, wrap"
  columns: "[[]]"
  rows: "[[]]4mm[[]]push[[]]"
  content: [
    Label {
      text: "Email"
      layoutInfo: nodeConstraints( "ax right" )
    }
    TextBox {
      layoutInfo: nodeConstraints( "growx, pushx" )
    }
    Label {
      text: "Password"
      layoutInfo: nodeConstraints( "ax right" )
    }
    TextBox {
      layoutInfo: nodeConstraints( "growx, pushx" )
    }
    Button {
      text: "Login"
      layoutInfo: nodeConstraints( "skip, right" )
    }
    Label {
      text: "This text is 'pushed' to the bottom"
      layoutInfo: nodeConstraints( "span" )
    }
  ]
}
```



JFXtras Shapes

	Almond	Intersection of two circles (Vesica Piscis)	<code>centerX, centerY, width</code>
	Arrow	Arrow shape	<code>x, y, width, height, depth, rise</code>
	Asterisk	Asterisk with rounded corners	<code>centerX, centerY, width, radius, beams, roundness</code>
	Astroid	Hypocloid with four cusps	<code>centerX, centerY, radius</code>
	Balloon	Rectangular shape with a tab	<code>x, y, width, height, arc, anglePosition, tabWidth, tabHeight, tabLocation, tabDisplacement</code>
	Cross	Symmetrical cross shape	<code>centerX, centerY, width, radius, roundness</code>
	Donut	Regular polygon with a hole	<code>centerX, centerY, innerRadius, outerRadius, sides</code>
	Lauburu	Four comma-shaped heads	<code>centerX, centerY, radius</code>

Continued...

JFXtras Shapes (continued)



MultiRoundRectangle Rectangle with configurable corners `x, y, width, height, topLeftWidth/Height, topRightWidth/Height, bottomLeftWidth/Height, bottomRightWidth/Height`



Rays Multiple rays extend from center `centerX, centerY, rays, radius, extent, rounded`



RegularPolygon Polygon with equal sides `centerX, centerY, sides, radius`



ReuleauxTriangle Curved triangle shape `centerX, centerY, radius`



RoundPin Cone with rounded top `centerX, centerY, height, radius`



Star2 Multipoint star `centerX, centerY, innerRadius, outerRadius, count`



ETriangle Equilateral triangle `x, y, width`

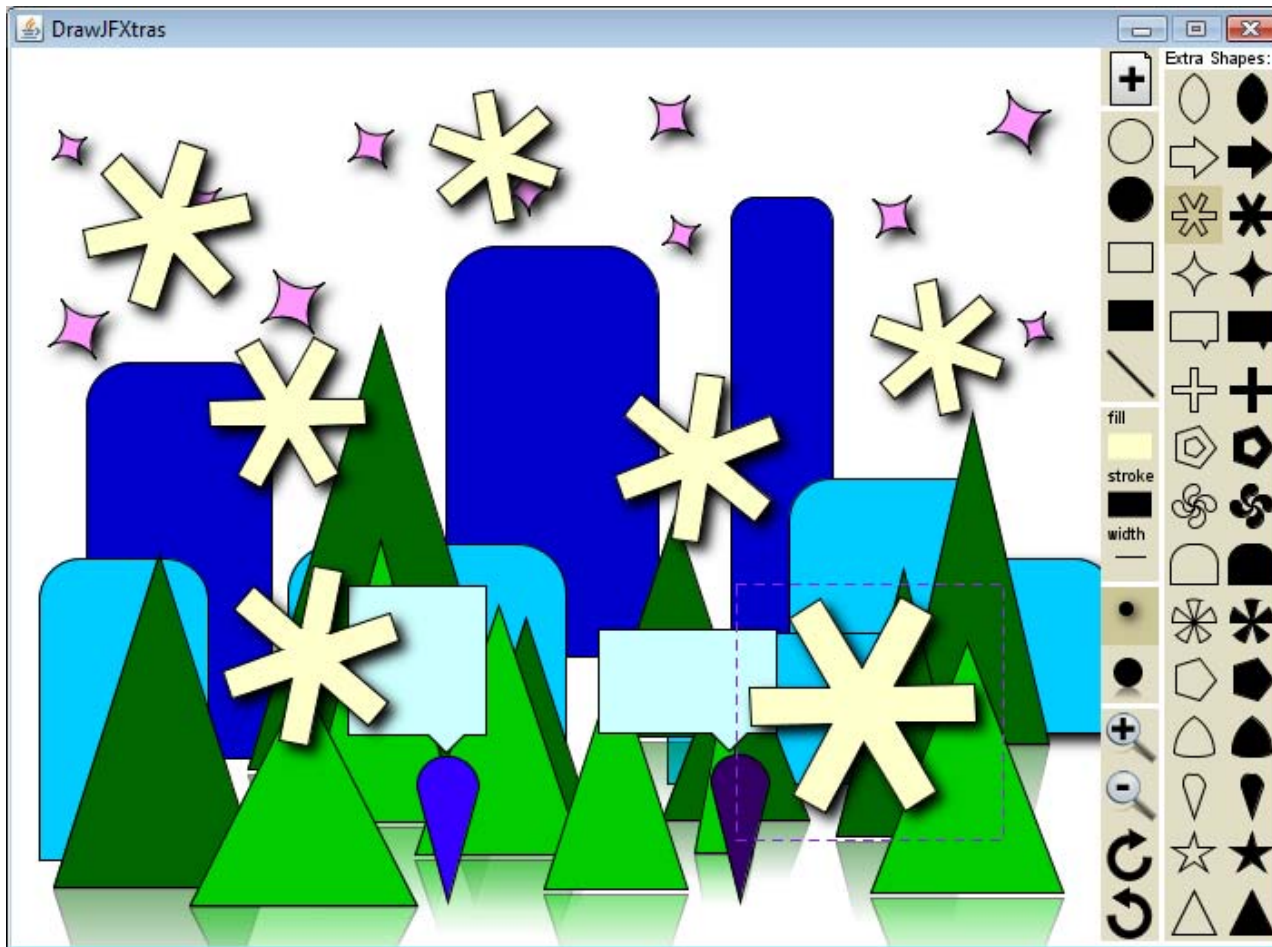


ITriangle Isosceles triangle `x, y, width, height`

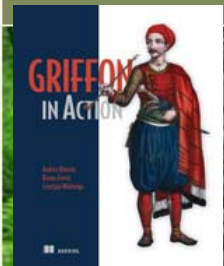


RTriangle Right triangle `x, y, width, height, anglePosition`

JFXtras Shapes



Sphere Challenge



Andres Almiray's Weblog

http://www.jroller.com/aalmiray/entry/griffon_gfxbuilder_fxbuilder_side_by

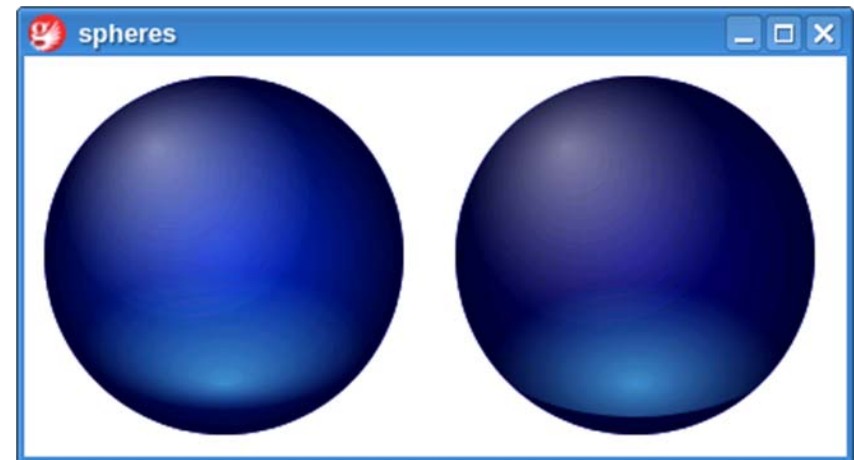
“The following snapshot shows a couple of spheres drawn with GfxBuilder and FxBuilder, can you guess which one is which?”

...

This is by no means a post to bash JavaFX rather to point out some of its deficiencies”

-- Andres Almiray

(taken completely out of context)

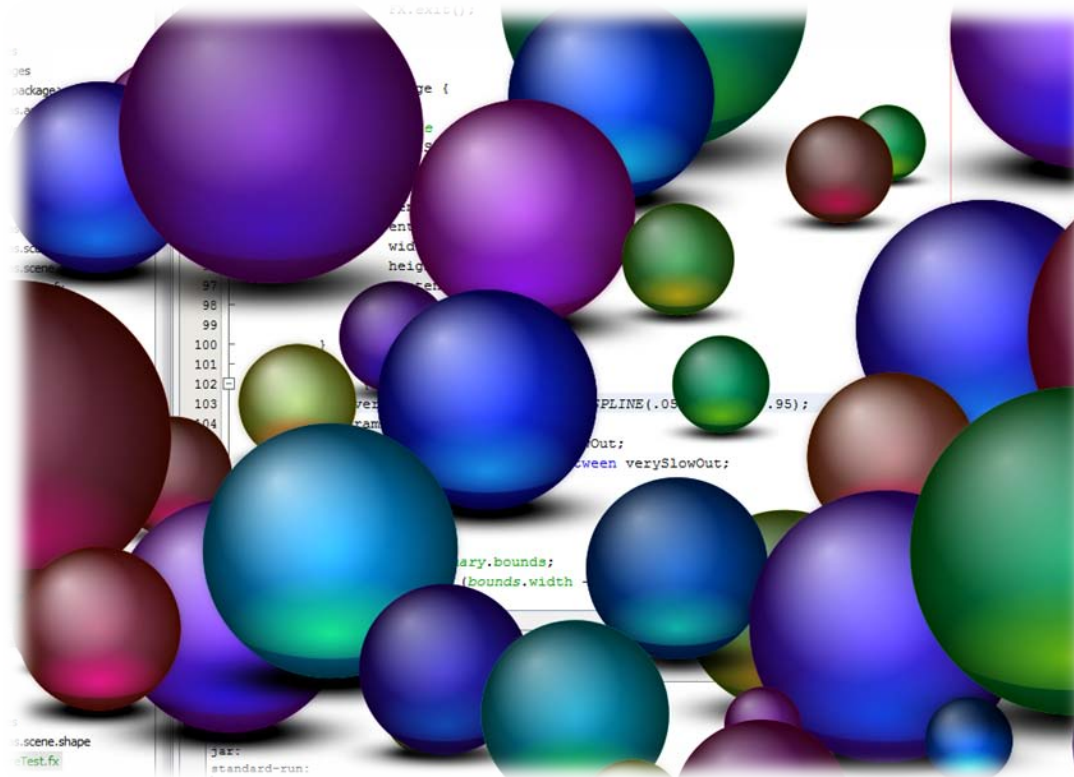


Sphere Challenge – JavaFX Response

- > Composition:
 - RadialGradient for the Sphere
 - Three additional RadialGradients for the light sources
 - A blurred shadow underneath
- > Features:
 - All Bound/Relative Coordinates
 - Configurable –
 - Base, Ambient, Specular, Shine Colors
 - Shadow Size and Height
 - Uses New JFXtras ColorUtil Library
 - JavaFX Caching for High Performance



JFXtras Spheres Demo



JFXtras Shelf





<http://jfxtras.org/>

JFXtras

on to WidgetFX...

Utilities and Add-ons for the JavaFX™ Platform

Backup Slides...

Row and Column Constraints

- > Size Types
 - “[min:pref:max][100:pref:500][3cm!]”
- > Gap Types
 - “[]5mm[]rel[]unrel[]push[]”

```
MigLayout {  
    constraints: "fill, wrap"  
    columns: "[[]]"  
    rows: "[[]]4mm[]push[]"  
  
    // to be continued  
}
```

Component Constraints

- > “cell 0 1”
- > “wrap 1cm”
- > “newline 1cm”
- > “grow, push”
- > “span 3”
- > “skip 2”
- > “width min:300:max, height 20mm!”
- > “alignx left”
- > “aligny right”
- > “sizegroup label”