

**ORACLE®**



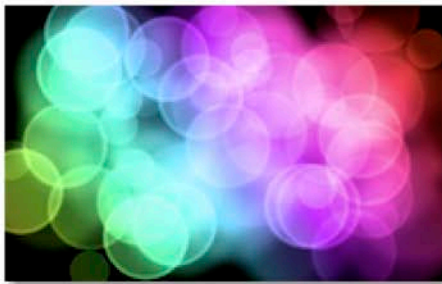
# JavaFX 2.0: Rich Internet Applications for the Java Platform

Simon Ritter

Technology Evangelist, Oracle



JavaFX is the evolution of the Java rich client platform, designed to provide a lightweight, hardware accelerated UI platform that meets tomorrow's needs.



Graphics



UI Controls



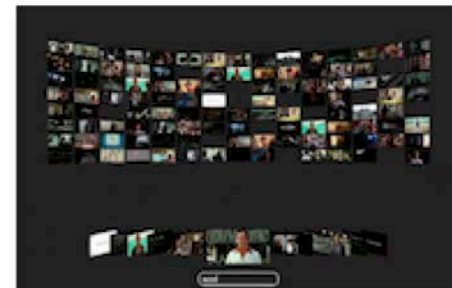
Visualization



Interaction



Animation



Media

# JavaFX Design Goals

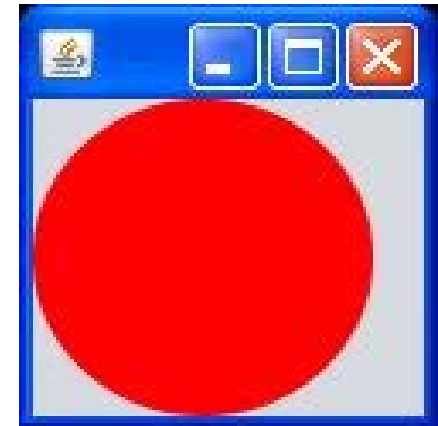
- Simple Programming model: the power of Java, the ease of JavaFX
- Interoperability between Java, JavaScript & HTML5
- High performance 2D and 3D Java graphics engine designed to exploit hardware advances in desktop & mobile
- Complete & integrated development lifecycle experience



# Let's Compare: JavaFX 1.x

```
import javafx.application.*;
import javafx.scene.shape.*;
import javafx.scene.paint.*;
```

```
Stage {
  scene: Scene {
    Content: [
      Circle {
        centerX: 50
        centerY: 50
        radius: 50
        fill: Color.RED
      }
    ]
  }
}
```



# Let's Compare: JavaFX 2.0

```
public class JavaFXTest extends Application {
    @Override public void start(Stage stage) {
        Group root = new Group();
        Scene scene = new Scene(root,100,100);
        stage.setScene(scene);

        Circle c1 =
            new Circle(50.0f, 50.0f, 50.0f, Color.RED);

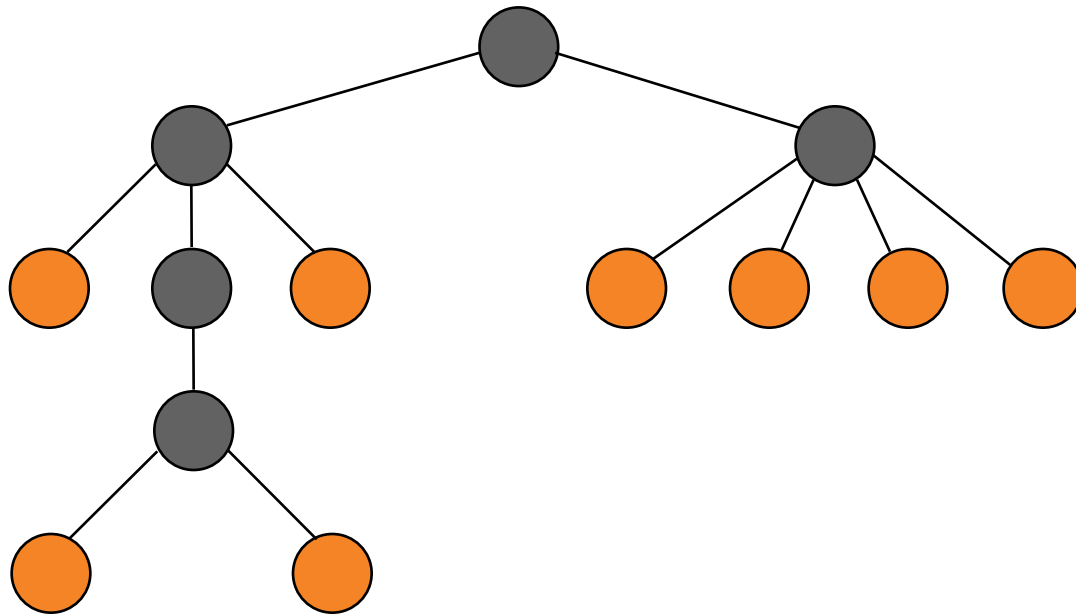
        root.getChildren().add(c1);
        stage.setVisible(true);
    }

    public static void main(String a[]) {
        Launcher.launch(JavaFXTest.class, null);
    }
}
```



# Scene Graph

- Directed Acyclic Graph
- Parents & children
- Representation of a GUI
- Drawing primitives and controls



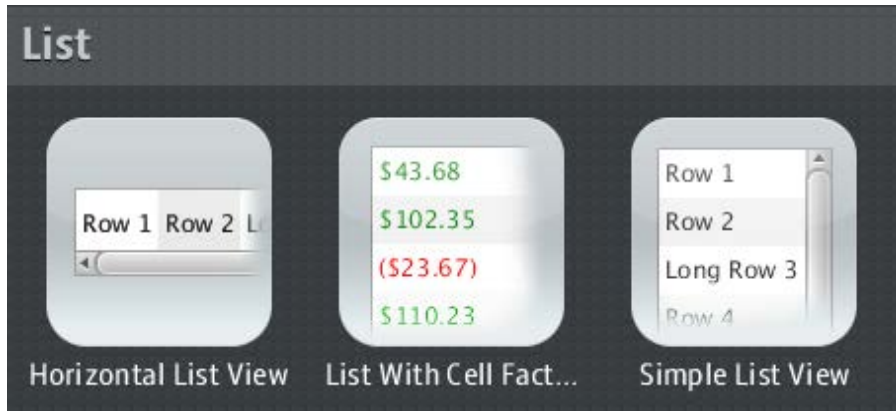
# Types of Nodes

- Shapes
- Images
- Media
- Web browser
- Text
- Controls
- Charts
- Group
- Container

# Media

- JavaFX supports both visual and audio media
- Cross platform JavaFX Media file (fxm, mp3)
- Media class represents a media file
- MediaPlayer plays a Media file
- MediaView is a Node which displays the Media
  - Many MediaViews can have the same MediaPlayer
    - And it is cheap to do so
  - MediaViews can scale the media proportionally or disproportionately
  - MediaView does not come with pre-built playback controls (you need a MediaControl)

# Controls



Many more...

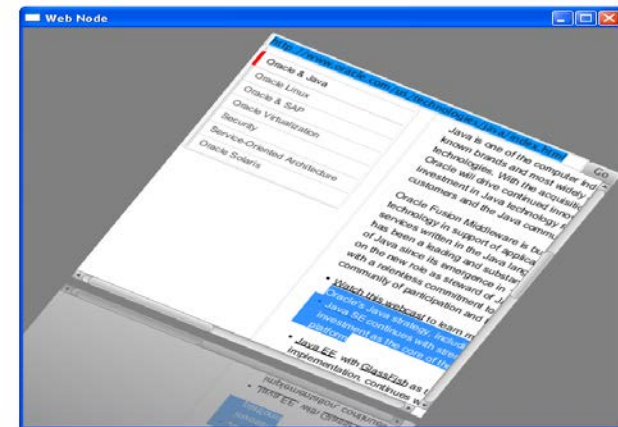
# Table

- Full featured table component
  - Resizeable columns
  - Columns can be moved
  - Groups of columns can be moved
- Uses standard MVC pattern
  - Create model for data
  - Attach to Table 'view' for display
- Efficient
  - Lazy loading of data – only displayed data is loaded

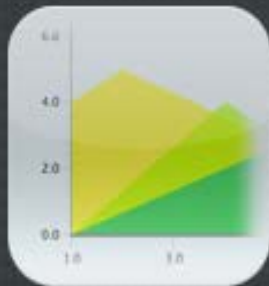
First	Last	Email
Jacob	Smith	jacob.smith@example.com
Isabella	Johnson	isabella.johnson@example.com
Ethan	Williams	ethan.williams@example.com
Emma	Jones	emma.jones@example.com
Michael	Brown	michael.brown@example.com

# Adding HTML Content: The Embedded Browser

- WebEngine
  - Provides basic web page browsing functionality. Renders web pages
  - Supports user interaction: navigating links, submitting HTML forms.
- WebView
  - Extension of a Node class
  - Encapsulates a WebEngine object
  - Incorporates HTML into the scene
    - To apply effects and transformations



# Charts



Area Chart



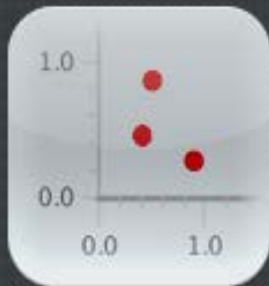
Bar Chart



Line Chart



Pie Chart



Scatter Chart

# Effects...

GaussianBlur



InnerShadow

# Shadow

Reflection



SepiaTone



# Transforms

```
Rectangle rect=new Rectangle(0,0,60,60);  
rect.setFill(Color.DODGERBLUE);  
rect.setArcWidth(10);  
rect.setArcHeight(10);
```

---

```
rect.setRotate(45);
```

---

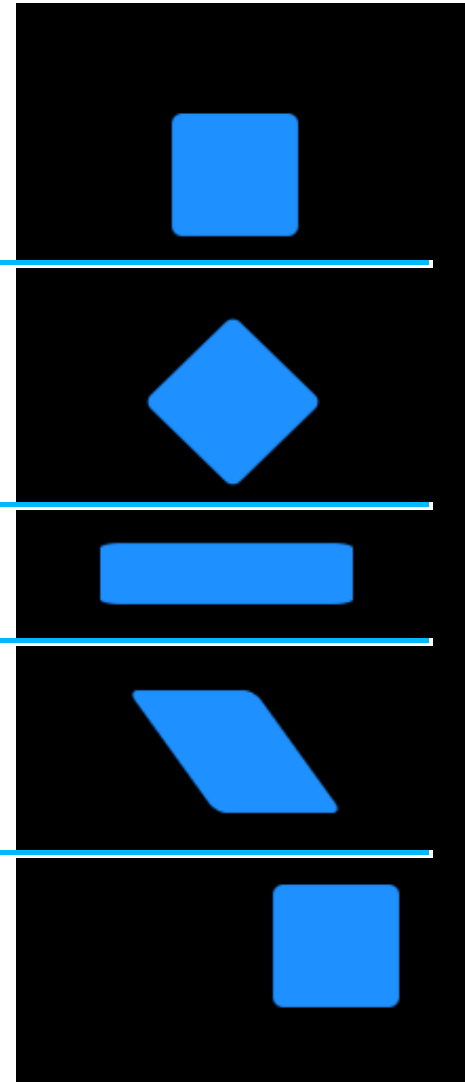
```
rect.setScaleX(2);  
rect.setScaleY(0.5);
```

---

```
Shear shear = new Shear(0.7, 0);  
rect.getTransforms().add(shear);
```

---

```
rect.setTranslateX(40);  
rect.setTranslateY(10);
```



# Layout

- A surprisingly hard problem!
- We've made fairly substantial changes in each release so far and we're going to do so again!
- Design Goals:
  - Easy to program with by hand
  - Works with animated transitions
  - Can be manipulated from CSS
  - Easy to use with RAD tools

# Layouts

- Pane
- AnchorPane
- BorderPane
- FlowPane
- GridPane
- HBox
- StackPane
- TilePane
- VBox

# Binding

- Creates a dependancy between a property and a changeable value
- High level API
  - Simple to use
  - Covers most common situations
- Low level API
  - Allows for more complex interactions
  - Optimised for fast execution and small footprint

# Properties

- Basis for high-level binding API
- Types for all primitives, String and Object
  - `DoubleProperty`, `StringProperty`, etc
- Subclasses `Observable`, `ReadOnlyProperty`, `WritableValue` interfaces
- Provides simple API
  - `bind`
  - `unbind`
  - `bindBidirectional/unbindBidirectional`
  - `isBound`
- Simple concrete classes

# Simple Binding Example

```
private SimpleDoubleProperty topXProperty =  
    new SimpleDoubleProperty();  
private SimpleDoubleProperty topYProperty =  
    new SimpleDoubleProperty();
```

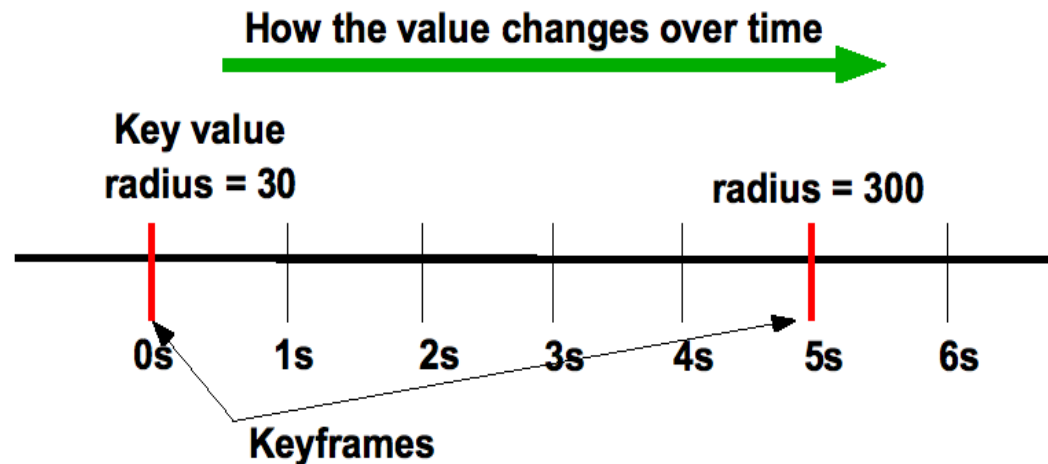
```
Line foldLine = new Line();  
foldLine.setEndX(200);  
foldLine.setEndY(200);  
foldLine.startXProperty().bind(topXProperty);  
foldLine.startYProperty().bind(topYProperty);
```

...

```
topXProperty.set(tx);  
topYProperty.set(ty);
```

# Timeline-Based Animation

- **Timeline**
  - Modifies values of variables specified by KeyFrames
- **KeyFrame**: specifies that a variable should have...
  - A particular value, at a particular time
- **KeyValue**: value to be interpolated for an interval
- Have **KeyFrame** modify a Node property
  - Use binding



# Animated Transitions

- Predefined, single-purpose animations
  - Fade, Path, Pause, Rotate, Scale, Translate
  - Can specify to, from, and by values
- Container transitions
  - Parallel, Sequential
  - Can be nested arbitrarily
- Transitions and Timelines have a similar ancestry
  - A timeline can be added to a Parallel / Sequential transition
- Transitions are being optimized for speed in 2.0

# Tasks

- The preferred way to work with threading
- A Task is a one-shot worker
  - Somewhat like a Callable with a lot more API
  - Can report:
    - Total amount of work to do
    - Amount of work complete
    - Percentage complete
    - Errors
    - Notification on completion
  - Implementations should also yield one or more “products” when they complete operation

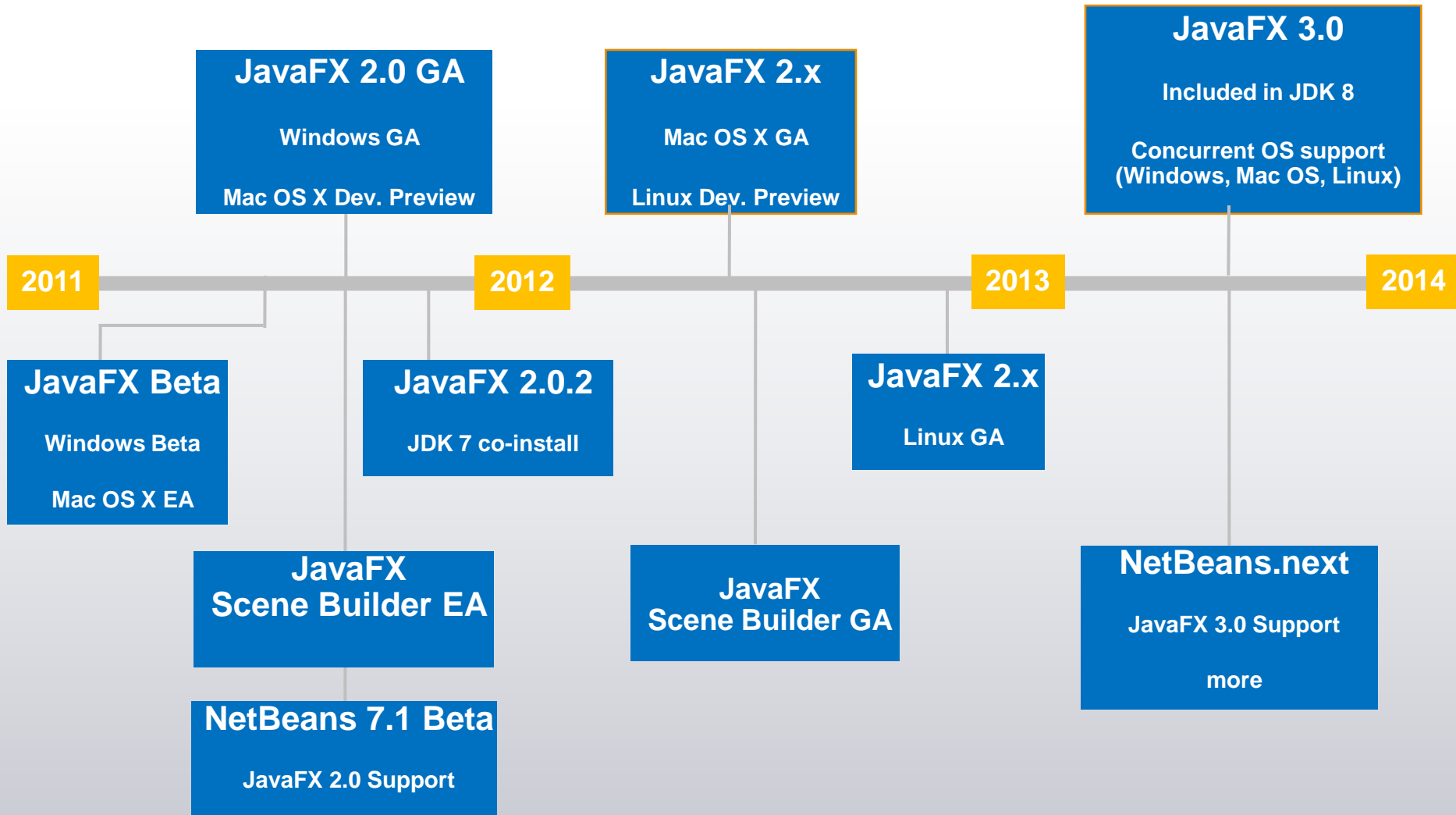
# Swing Integration

- We are **FINALLY** supporting embedding of JavaFX into existing Swing applications!
- Accomplishing this requires 3 objectives:
  - Java APIs for JavaFX
  - Ability to embed hardware accelerated 2D/3D scenes
  - Swing API for embedding the scene
- However (shed a tear), we are not going to support embedding Swing components in JavaFX scene graphs

# Experiments & Blueprints

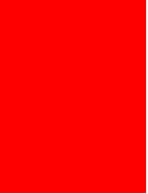
- At the same time we are working on the platform, we are building experiments and blueprints
- Experiments:
  - Small applications meant for outside developers to see and play with, but who's code is not necessarily ideal
- Blueprints:
  - Larger applications meant to simulate (or actually be) real world, and who's code is representative of a best practice and intended to be copied by developers

# JavaFX Roadmap



# Conclusions

- JavaFX provides a new way to write rich, visual applications
  - Java language based
  - Easy to extend existing applications
  - Integration with Swing
- Powerful features
  - Binding
  - Animations
  - Extensive component library
- Try it now and give feedback
  - <http://www.javafx.com>



The preceding is intended to outline our general product direction. It is intended for information purposes only, and may not be incorporated into any contract. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, and timing of any features or functionality described for Oracle's products remains at the sole discretion of Oracle.



**ORACLE<sup>®</sup>**

Thank You