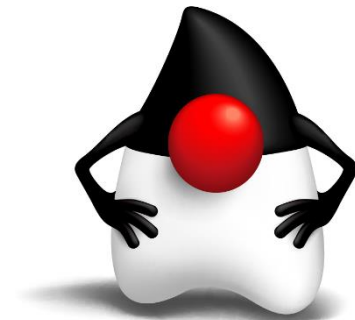




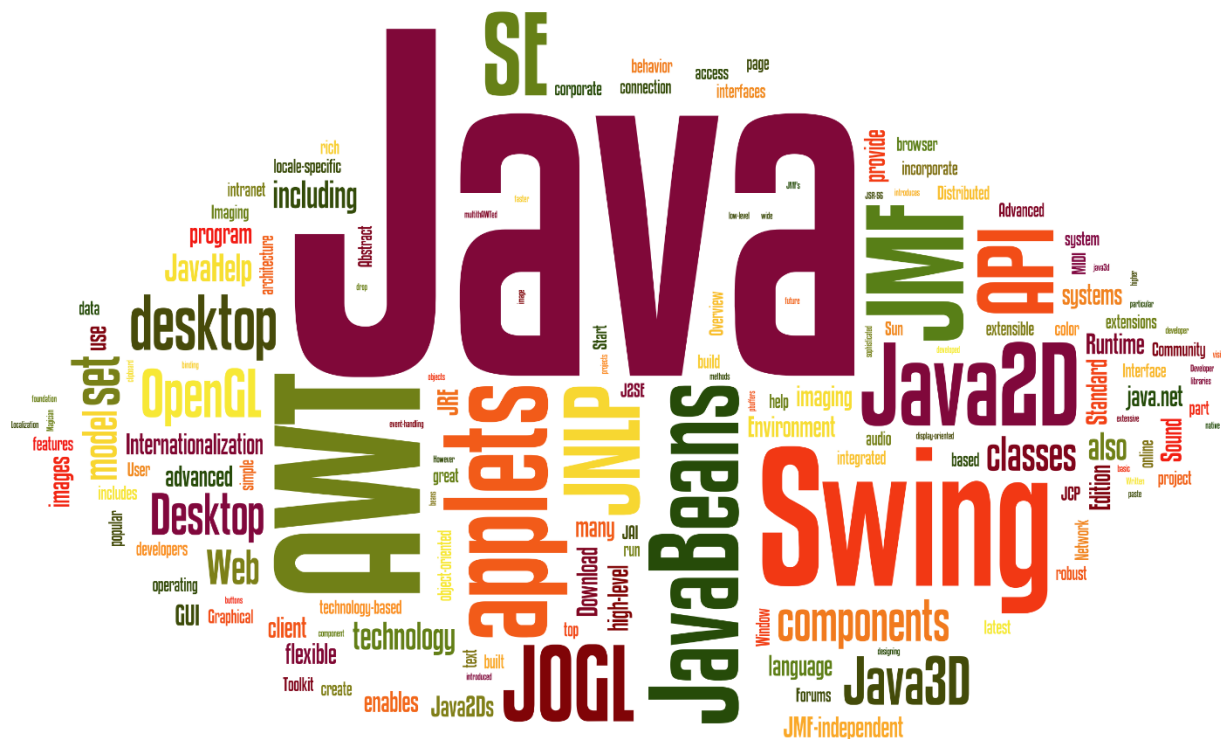
MOVING JAVA FORWARD

ORACLE®

JavaFX: Java's new Rich Client Platform



But developers had to learn multiple technologies



Tutorial and API Docs

<http://docs.oracle.com/javase/8/javafx/get-started-tutorial/>

<http://docs.oracle.com/javase/8/javase-clienttechnologies.htm>

Ensemble – Collection of Examples

<http://download.oracle.com/otndocs/products/javafx/2/samples/Ensemble/index.html>

Videos on JavaFX

[https://www.youtube.com/user/OracleLearning/search?quer](https://www.youtube.com/user/OracleLearning/search?query=javaafx)

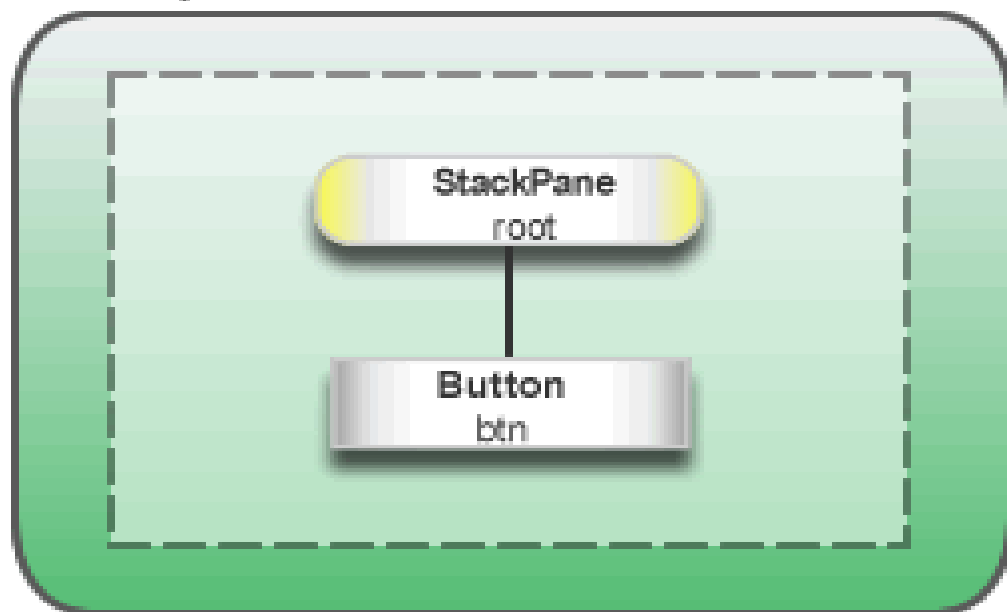
JavaFX Simplifies Application Development

Developers Focus on Capabilities Instead of Technologies



Stage `javafx.stage (window)`

Scene `javafx.scene`



```

13
14 public class BasicAppMain extends Application {
15
16     @Override
17     public void start(Stage primaryStage) {
18         Group root = new Group();
19         Scene scene = new Scene(root, 800, 600, Color.BLACK);
20         primaryStage.setScene(scene);
21         //modify the root or scene here
22         primaryStage.show();
23     }
24

```

Manual

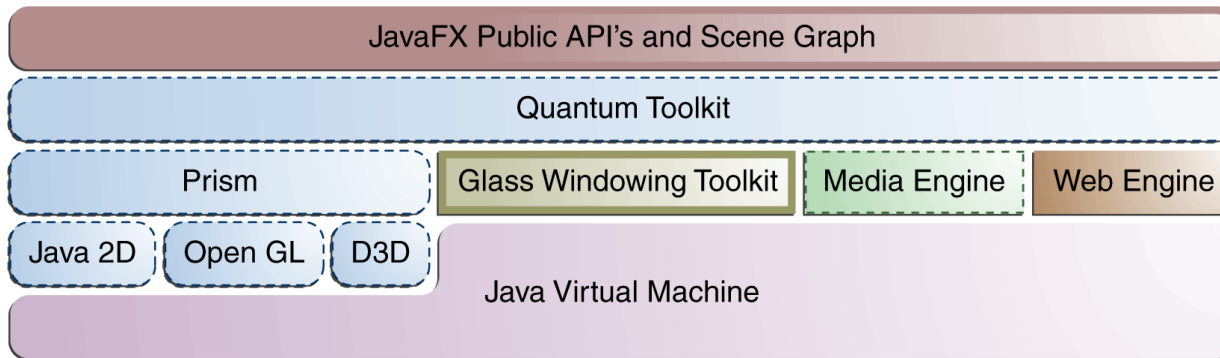
```

11 public class MainApp extends Application {
12
13     @Override
14     public void start(Stage stage) throws Exception {
15
16         //Stage
17         //Scene
18         //Root
19
20         Parent root = FXMLLoader.load(getClass().getResource("/fxml/Scene.fxml"));
21         Scene scene = new Scene(root);
22         scene.getStylesheets().add("/styles/Styles.css");
23         stage.setTitle("Rsvp example");
24         stage.setScene(scene);
25         stage.show();
26     }
27

```

FXML

JavaFX Runtime High Level Architecture



JavaFX Glossary

- **Glass Windowing Toolkit:** Provides native operating services, such as managing the windows, timers, and surfaces
- **Prism:** Graphics pipeline that can run on hardware and software renderers
- **Quantum Toolkit:** Ties Prism and Glass together and makes them available to the JavaFX APIs
- This is completely seamless in Java8



ORACLE

Threads in JavaFX

- **JavaFX application thread:** This is the primary thread used by JavaFX application developers. Any “live” scene, which is a scene that is part of a window, must be accessed from this thread. A scene graph can be created and manipulated in a background thread, but when its root node is attached to any live object in the scene, that scene graph must be accessed from the JavaFX application thread. This enables developers to create complex scene graphs on a background thread while keeping animations on 'live' scenes smooth and fast. The JavaFX application thread is a different thread from the Swing and AWT Event Dispatch Thread (EDT), so care must be taken when embedding JavaFX code into Swing applications.
- **Prism render thread:** This thread handles the rendering separately from the event dispatcher. It allows frame N to be rendered while frame N +1 is being processed. This ability to perform concurrent processing is a big advantage, especially on modern systems that have multiple processors. The Prism render thread may also have multiple rasterization threads that help off-load work that needs to be done in rendering.
- **Media thread:** This thread runs in the background and synchronizes the latest frames through the scene graph by using the JavaFX application thread.



ORACLE

JavaFX Pulse

- A pulse is an event that indicates to the JavaFX scene graph that it is time to synchronize the state of the elements on the scene graph with Prism.
- A pulse is throttled at 60 frames per second (fps) maximum and is fired whenever animations are running on the scene graph.
- Even when animation is not running, a pulse is scheduled when something in the scene graph is changed.
- When a pulse is fired, the state of the elements on the scene graph is synchronized down to the rendering layer.
- Numerous changes in the scene graph could lead to multiple layout or CSS updates, which could seriously degrade performance. The system automatically performs a CSS and layout pass once per pulse to avoid performance degradation.
- The Glass Windowing Toolkit is responsible for executing the pulse events.



Java APIs and FXML

Java APIs for JavaFX

- End-to-end Java development
- Java language features - generics, annotations, multi-threading
- Fluent API for UI construction
- Alternative JVM supported languages (e.g. Groovy, Scala) with JavaFX
- Leverage sophisticated Java IDEs, debuggers and profilers
- Java APIs preserve convenient JavaFX Script features (e.g., bind)

FXML

- Scriptable, XML-based markup language for defining UI
- Convenient alternative to developing UI programmatically in Java
- Easy to learn and intuitive for developers familiar with web technologies or other markup based UI technologies
- Powerful scripting feature allows embedding scripts within FXML. Any JVM scripting language can be used, including JavaScript, Groovy, and Scala



Graphics and Media

New Graphics Pipeline

- New hardware accelerated graphics pipeline (Prism)
- New windowing toolkit (Glass) for Prism
- Java2D software pipeline under Prism
- High-level support for making rich graphics simple
 - Shadows, Blurs, Reflections, Effects, 2D transforms
 - 3D Transforms; Full 3D objects

Media

- Stable media framework based on GStreamer
- VP6, MP3 playback of Web multimedia content
- Low latency audio
- Alpha channel support
- Performance improvements
- Full screen video

WebView and Swing Interoperability

WebView Component

- Embed Web content in JavaFX applications
- HTML rendering based on Webkit
- Hardware accelerated rendering using PRISM
- DOM access and manipulation

Swing and SWT Interop

- Embed JavaFX content into existing Swing applications
- Extend existing Swing applications with new JavaFX features such as WebView and high-performance graphics
- Applies to SWT applications as well

Browser Plugin

- Faster loading of JavaFX Web applications based on Prism
- Pre-loader for improved user experience with JavaFX Web applications

UI Controls



Charts



Area Chart



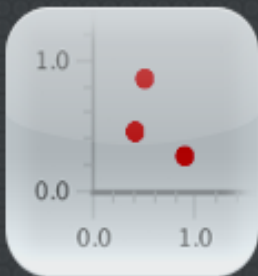
Bar Chart



Line Chart

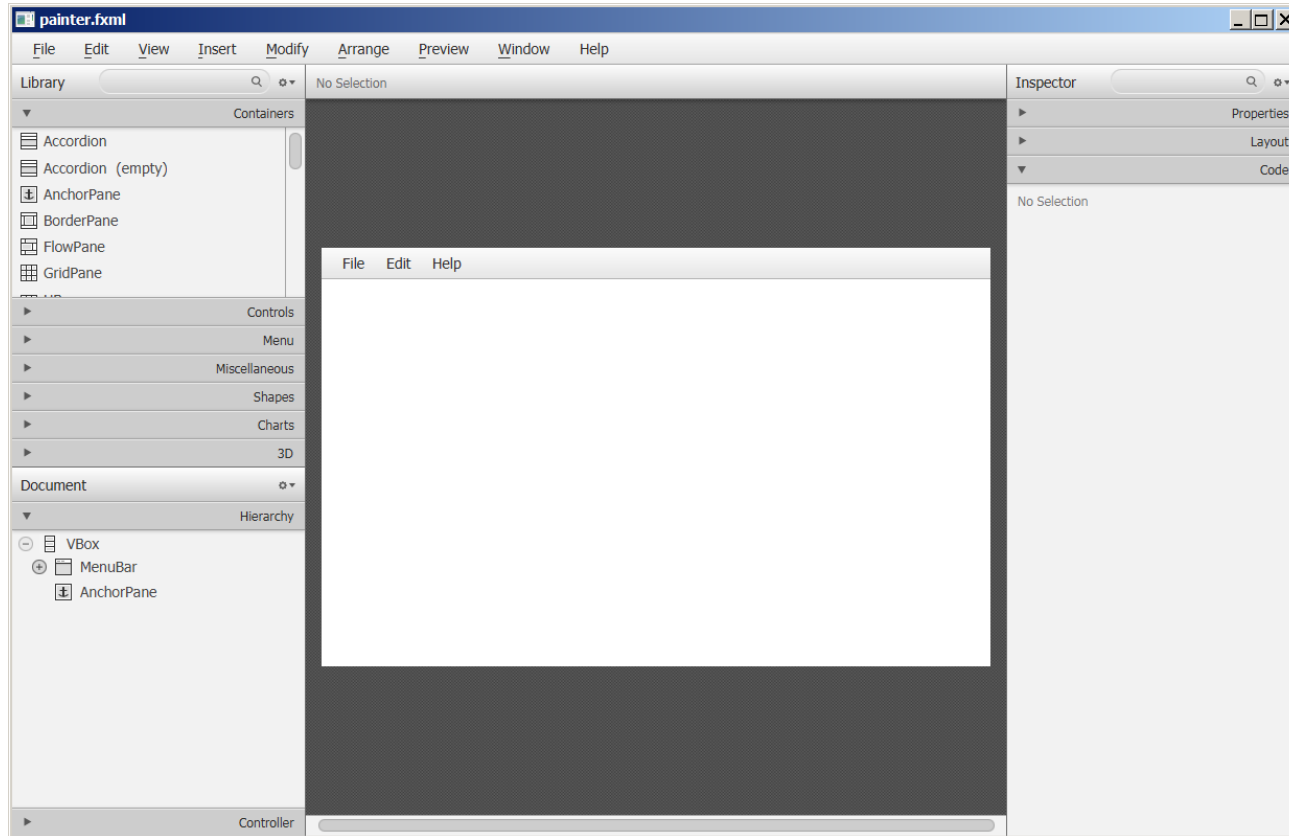


Pie Chart



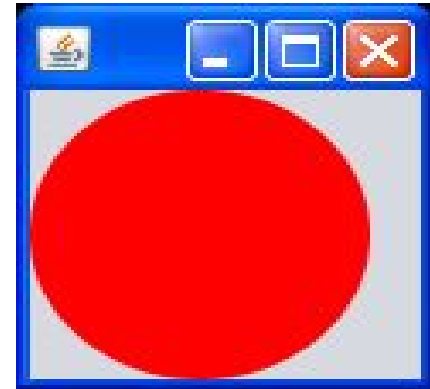
Scatter Chart

SceneBuilder



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

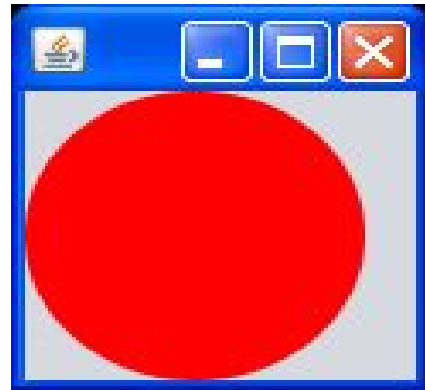


ORACLE

Let's Compare: FXML

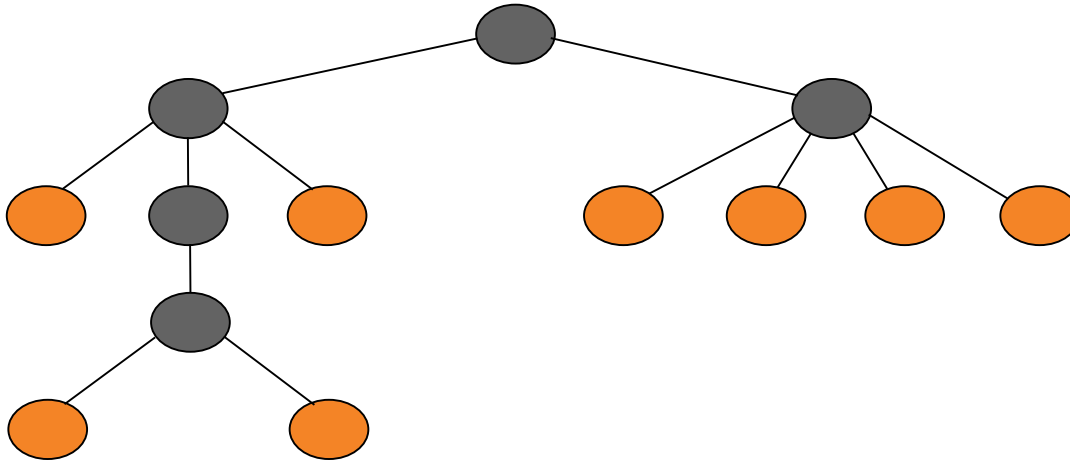
```
<BorderPane>
  <center>
    <Circle radius="50" centerX="50" centerY="50"/>
  </center>
</BorderPane>
```

```
public class JavaFXTest extends Application {
    @Override public void start(Stage stage) {
        stage.setTitle("FXML Example");
        Parent root = FXMLLoader.load(getClass().getResource("example.fxml"),
            ResourceBundle.getBundle("r.fxml_example"));
        stage.setScene(new Scene(root));
        stage.show();
    }
}
```

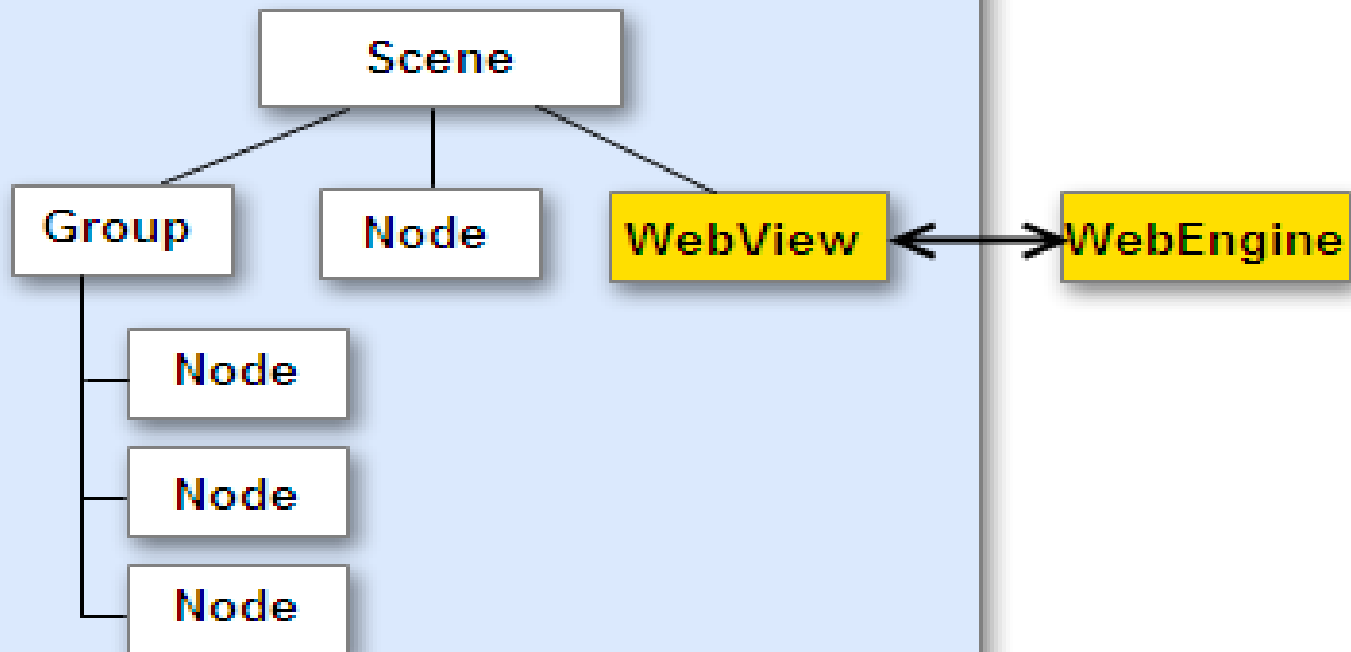


Scene Graph

- Directed Acyclic Graph
- Parents and children
- Representation of the GUI components



Scene Graph



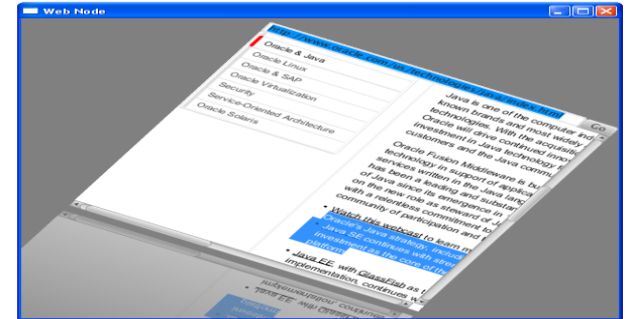
Media

- JavaFX supports both visual and audio media
- Cross-platform JavaFX media file format (fxm, mp3)
 - Platform specific formats supported via native players
- Media class represents a media file
- MediaPlayer provides control of the media rendering
- MediaView uses MediaPlayer to render media as Node
 - Many MediaViews can use the same MediaPlayer (cheaply)

Adding HTML Content

The Embedded Browser

- WebEngine
 - Provides basic web page browsing functionality
 - Supports user interaction: navigating links, submitting forms
- WebView
 - Web page as a Node in scenegraph
 - Effects can be applied
 - Encapsulates WebEngine object
 - No plugin support



Effects...

GaussianBlur



InnerShadow

Shadow

Reflection



SepiaTone



Transforms

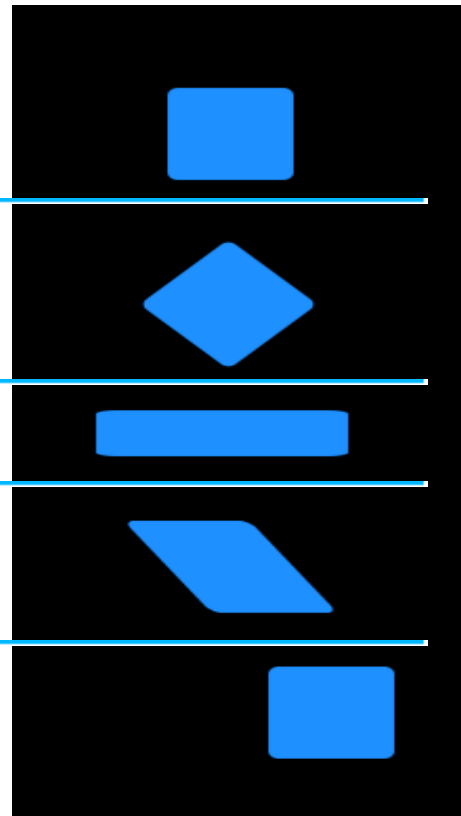
```
Rectangle rect=new Rectangle(0,0,60,60);  
rect.setFill(Color.DODGERBLUE);  
rect.setArcWidth(10);  
rect.setArcHeight(10);
```

```
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);
```



Binding

- Creates a dependency between a property and a changeable value
- High level API
 - Easy 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
- Concrete types for all primitives, String and Object
 - `DoubleProperty`, `StringProperty`, etc
- Simple API
 - `bind` / `unbind`
 - `bindBidirectional` / `unbindBidirectional`
 - `isBound`

Simple Binding Example

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

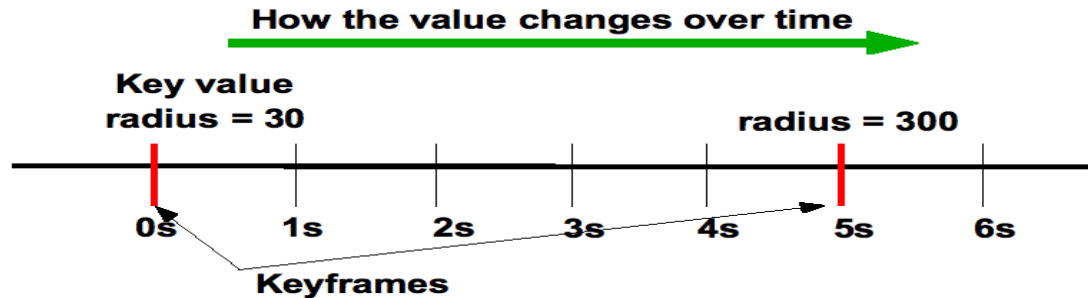
```
Line foldLine = new Line();  
foldLine.setEndX(200);  
foldLine.setEndY(200);  
foldLine.startXProperty().bind(topXProperty);  
foldLine.startYProperty().bind(topYProperty);
```

...

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

Timeline Based Animations

- **Timeline**
 - Modifies values of variables specified in KeyFrames
- **KeyFrame**: specifies that a variable should have
 - A particular value at a particular time
- **KeyValue**: Value to be interpolated for an interval



Animated Transitions

- Pre-defined, 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 share ancestry
 - A Timeline can be added to a Parallel / Sequential transition

Standard Java Tools for Easy Development



- Source editor with improved syntactic highlighting, code completion, refactoring etc.
- Full debugger and profiler support
- Project wizard for easy creation of JavaFX applications

Other Java IDEs

- Source editor with syntactic highlighting, code completion, refactoring etc.
- Full debugger and Profiler support



JavaFX is ...

- Cross platform: Windows GA, Mac & Linux Dev. Preview
- Familiar: 100% Java APIs
- Powerful: leverages underlying Java platform
- Modern: CSS skinning, HW acceleration, Webkit
- Backwards 'compatible': Swing & SWT interoperability
- Flexible: applicable to embedded, tablets and mobile
- Open Source: <http://openjdk.java.net/projects/openjfx>