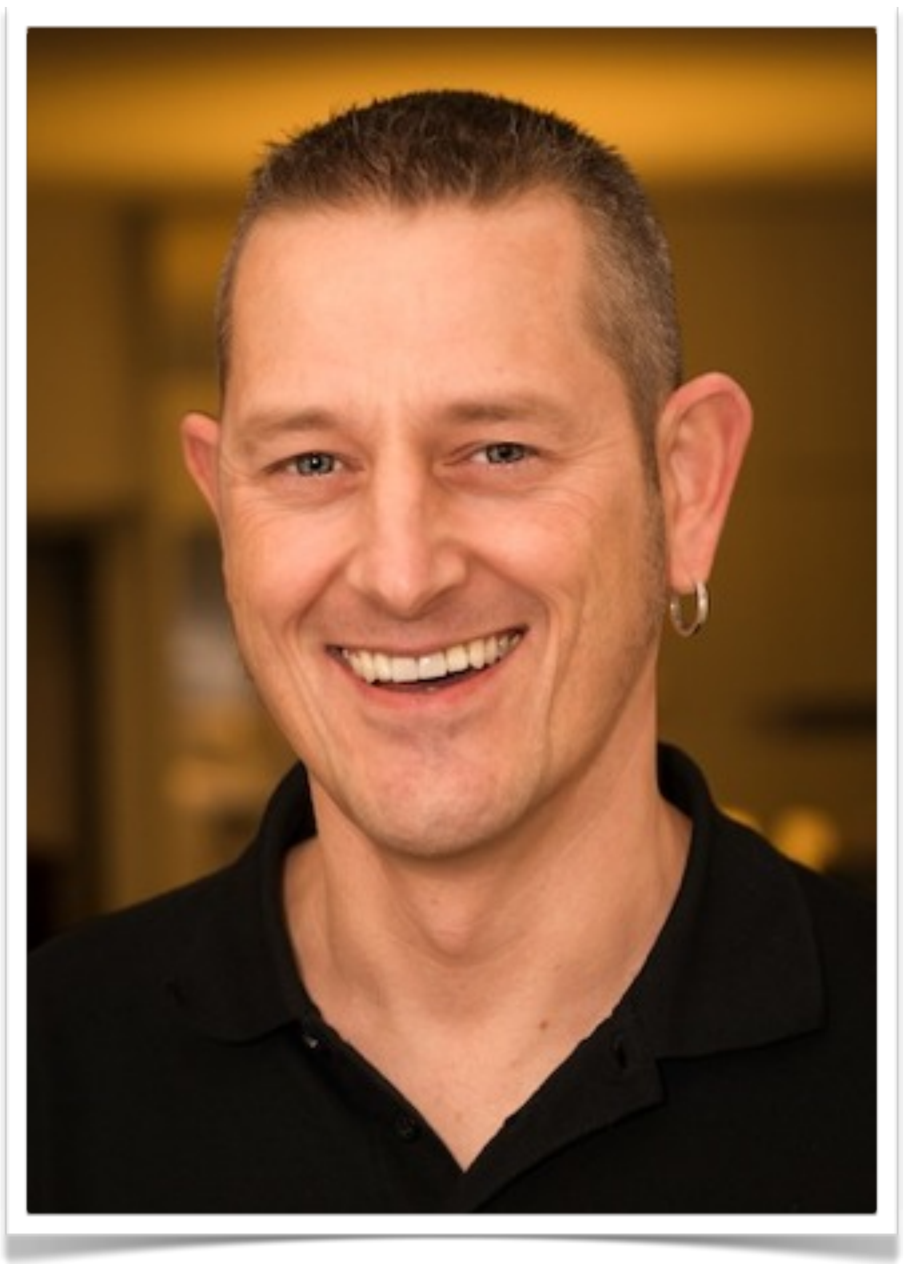
GENERAL

JavaFx

OVERVIEW



GERRIT GRUNWALD

canoo Engineering AG

TWITTER: @HANSOLO_

BLOG: HARMONIC-CODE.ORG

Agenda

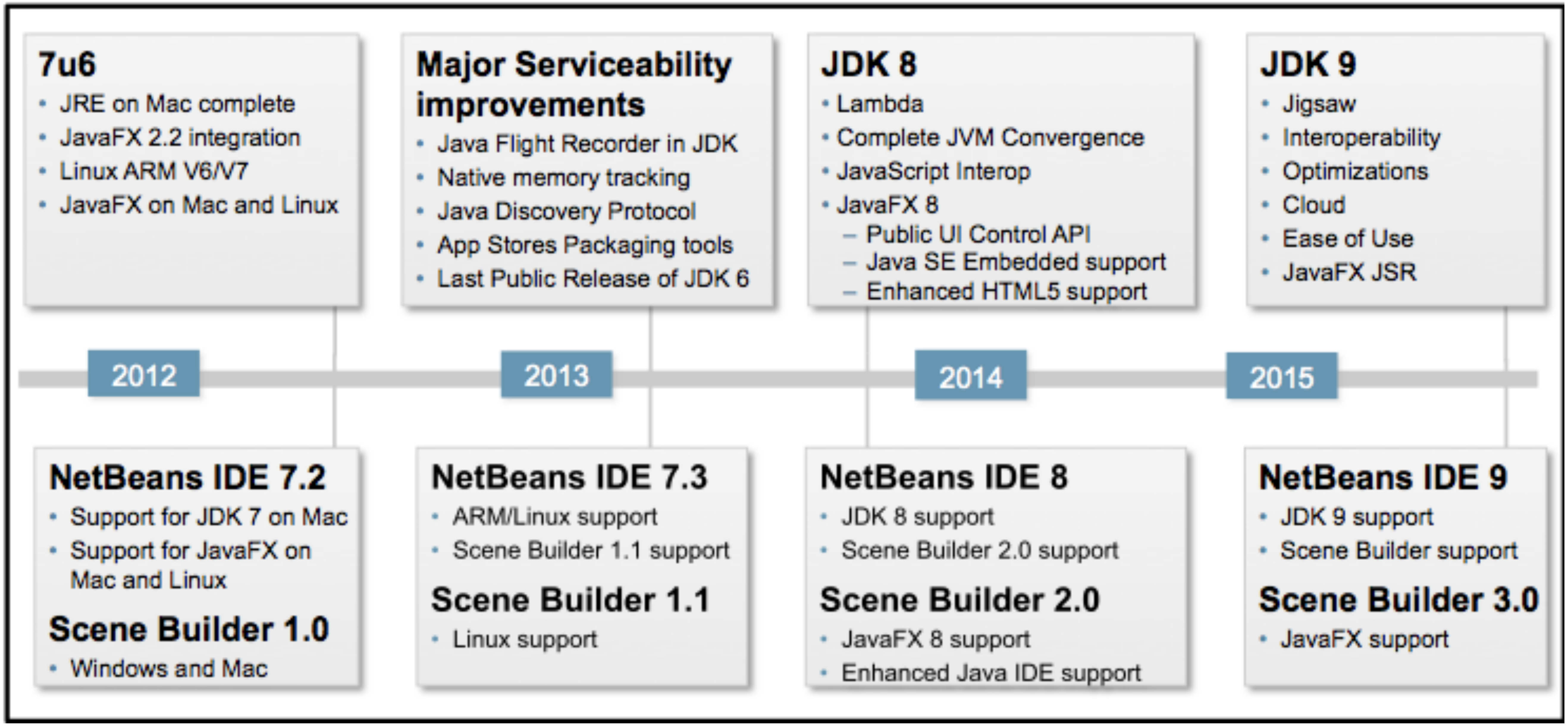
- HISTORY
- SCENE GRAPH
- JAVA API
- PROPERTIES
- BINDINGS
- CONTROLS
- CSS
- WEBVIEW
- JFXPANEL
- CHARTS

Some

HISTORY

11/2006	F3
05/2007	JAVA FX 1.0
02/2009	JAVA FX 1.1
06/2009	JAVA FX 1.2
04/2010	JAVA FX 1.3
08/2010	JAVA FX 1.3.1
10/2011	JAVA FX 2.0
04/2012	JAVA FX 2.1
08/2012	JAVA FX 2.2

Roadmap



7u6
- JRE on Mac complete
- JavaFX 2.2 integration
- Linux ARM V6/V7
- JavaFX on Mac and Linux

Major Serviceability improvements
- Java Flight Recorder in JDK
- Native memory tracking
- Java Discovery Protocol
- App Stores Packaging tools
- Last Public Release of JDK 6

JDK 8
- Lambda
- Complete JVM Convergence
- JavaScript Interop
- JavaFX 8
  - Public UI Control API
  - Java SE Embedded support
  - Enhanced HTML5 support

JDK 9
- Jigsaw
- Interoperability
- Optimizations
- Cloud
- Ease of Use
- JavaFX JSR

2012 2013 2014 2015

NetBeans IDE 7.2
- Support for JDK 7 on Mac
- Support for JavaFX on Mac and Linux

Scene Builder 1.0
- Windows and Mac

NetBeans IDE 7.3
- ARM/Linux support
- Scene Builder 1.1 support

Scene Builder 1.1
- Linux support

NetBeans IDE 8
- JDK 8 support
- Scene Builder 2.0 support

Scene Builder 2.0
- JavaFX 8 support
- Enhanced Java IDE support

NetBeans IDE 9
- JDK 9 support
- Scene Builder support

Scene Builder 3.0
- JavaFX support

What

JAVA FX

really is...

It is the successor to

JAVA SWING

and it's still not

FINISHED

Available for

- WINDOWS
- MAC OS X
- LINUX
- ARM *

* PREVIEW

Available for

- APPLE IOS*
- ANDROID*

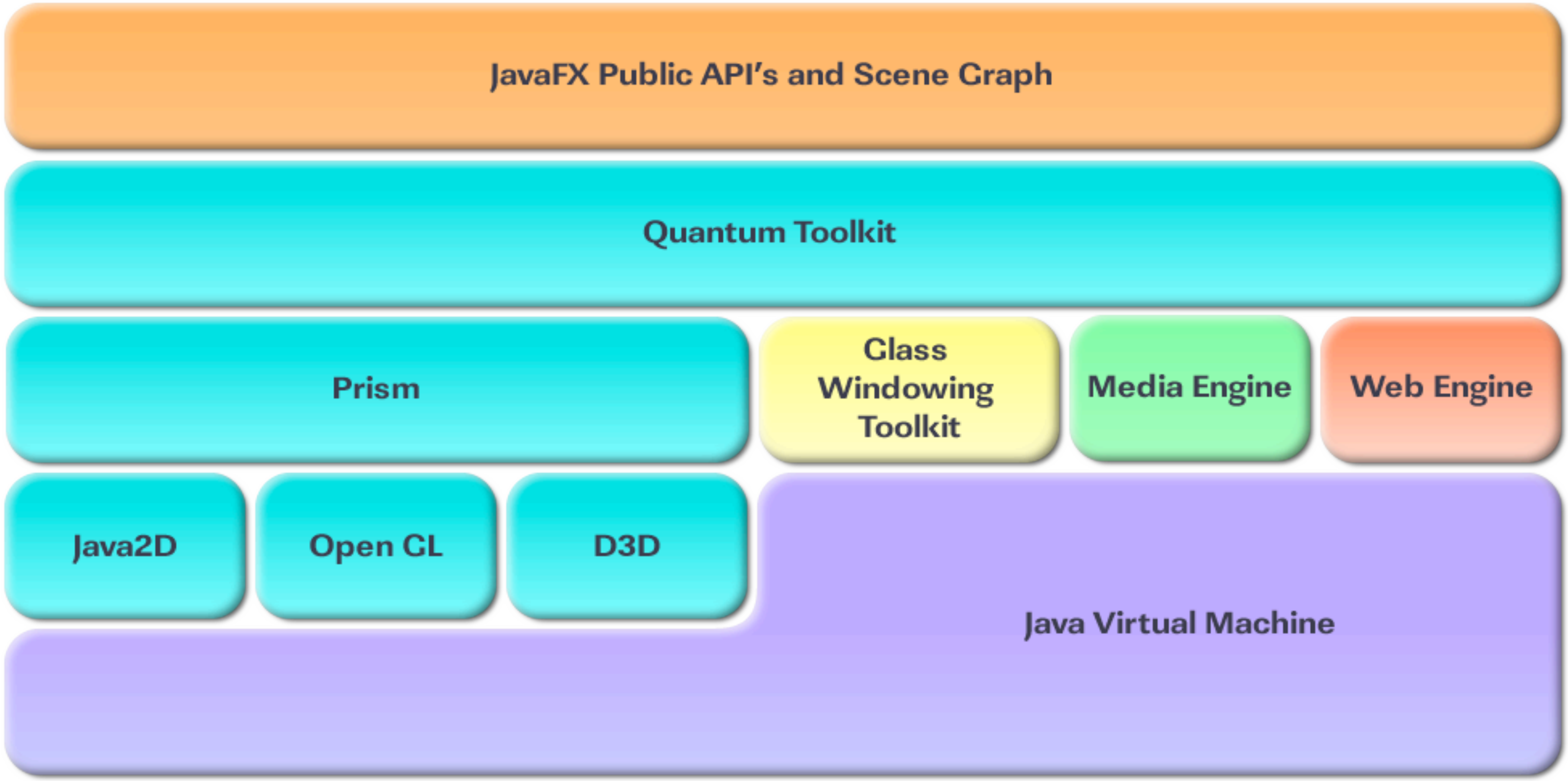
* PREVIEW

Versions

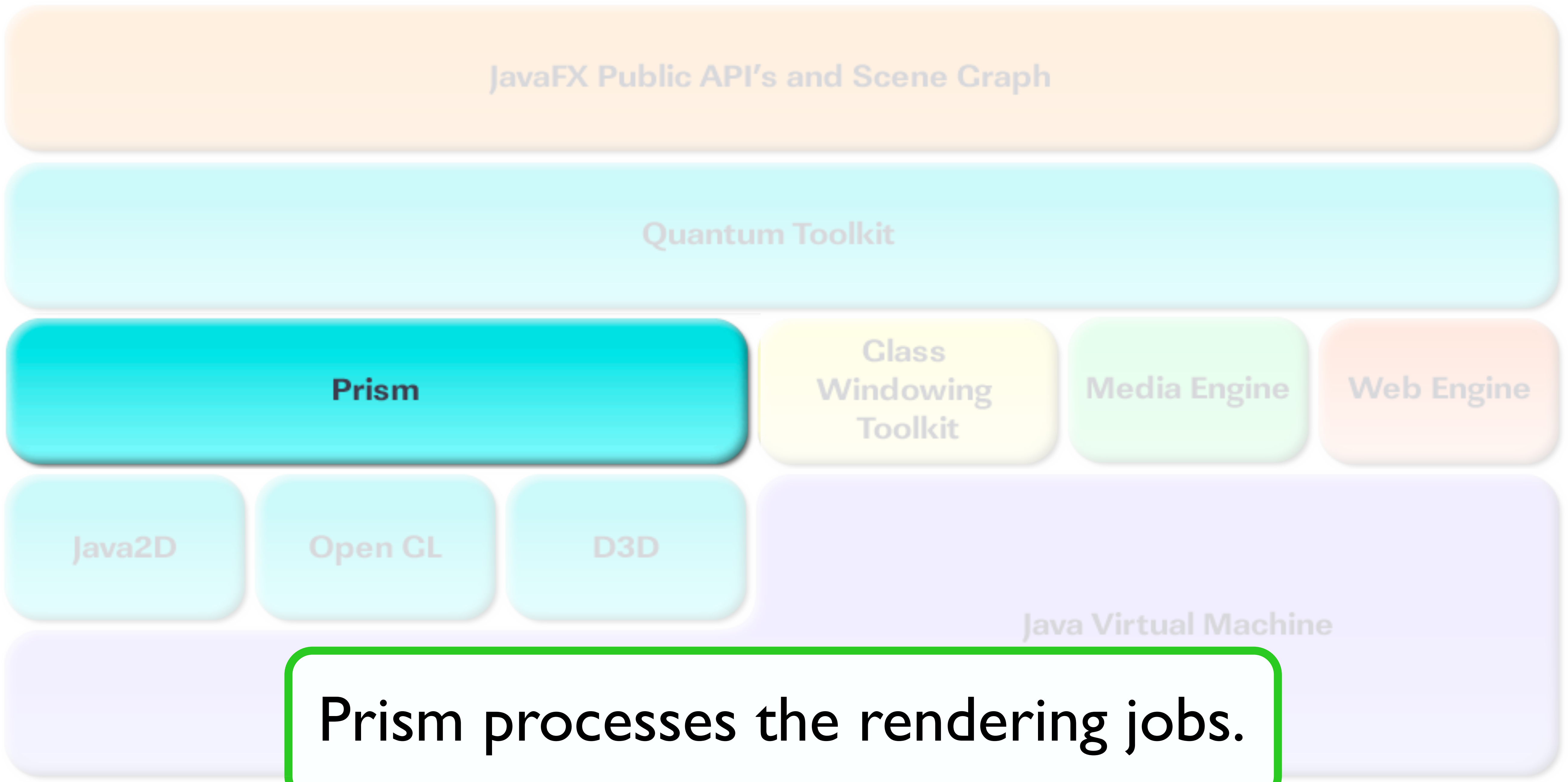
- JAVAFX 2.2 BUNDLED WITH JDK >JAVA 7U6
- STANDALONE FOR JAVA6*

* WINDOWS ONLY

The architecture

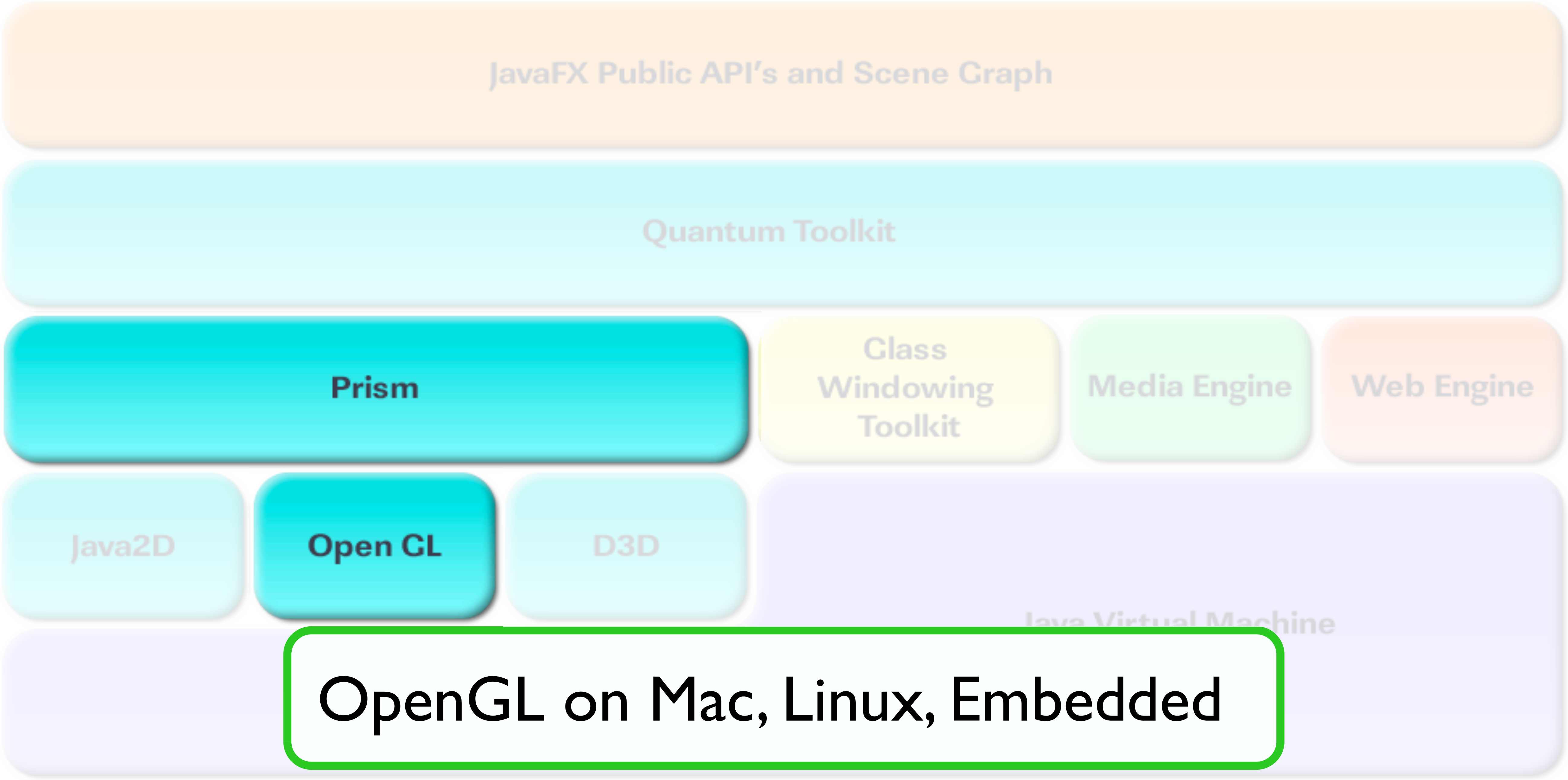


The architecture



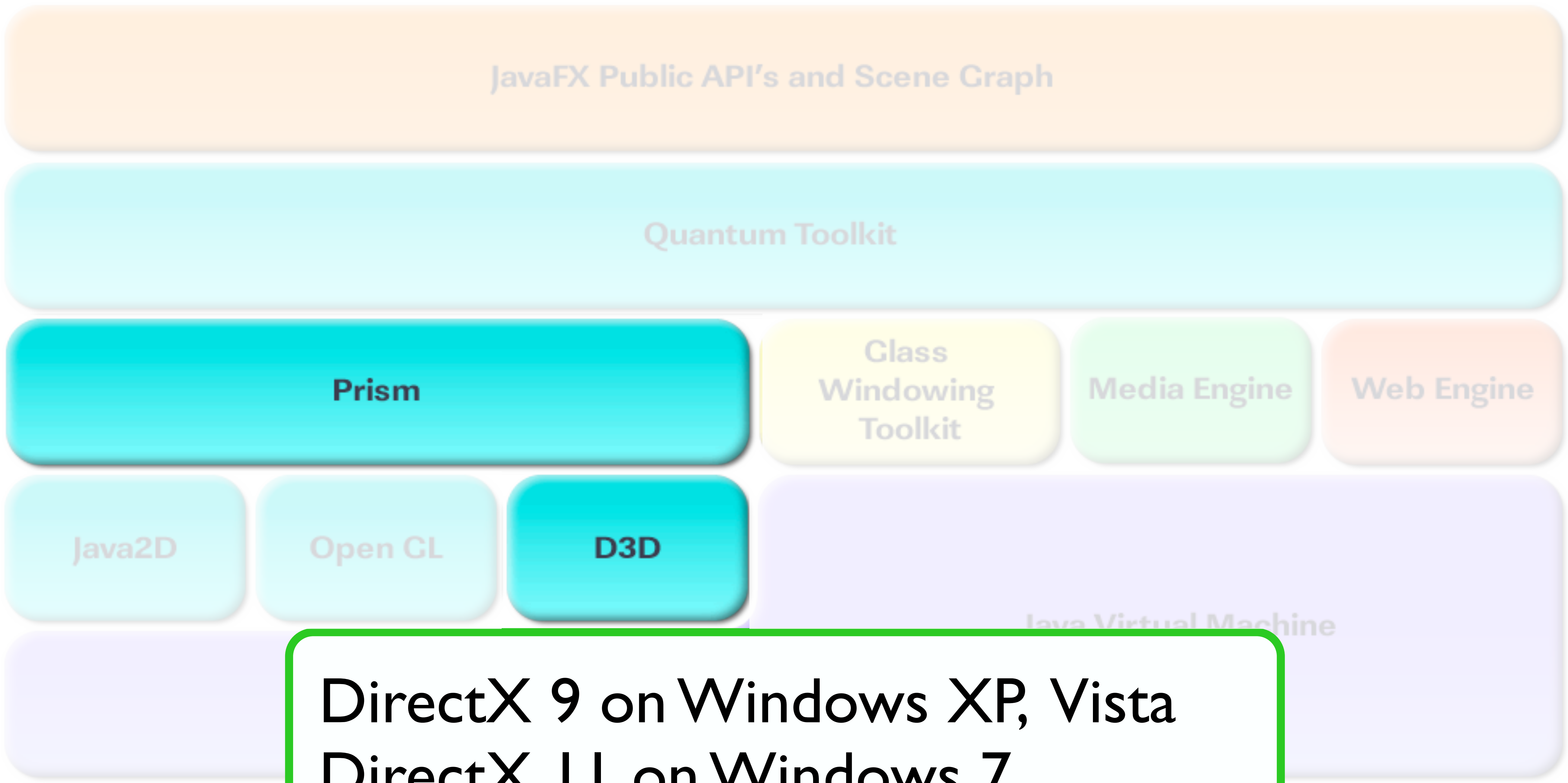
Prism processes the rendering jobs.

The architecture



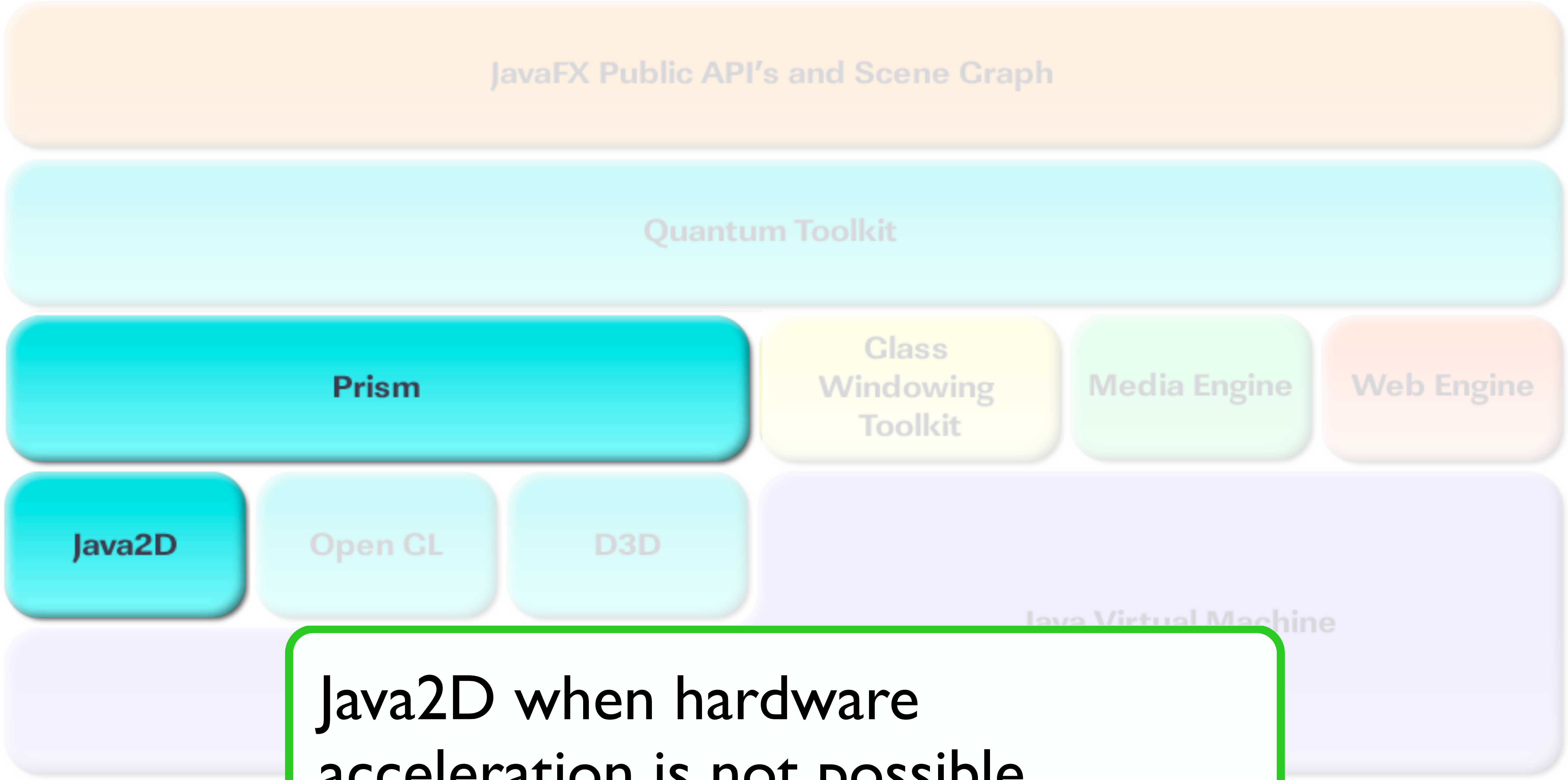
OpenGL on Mac, Linux, Embedded

The architecture



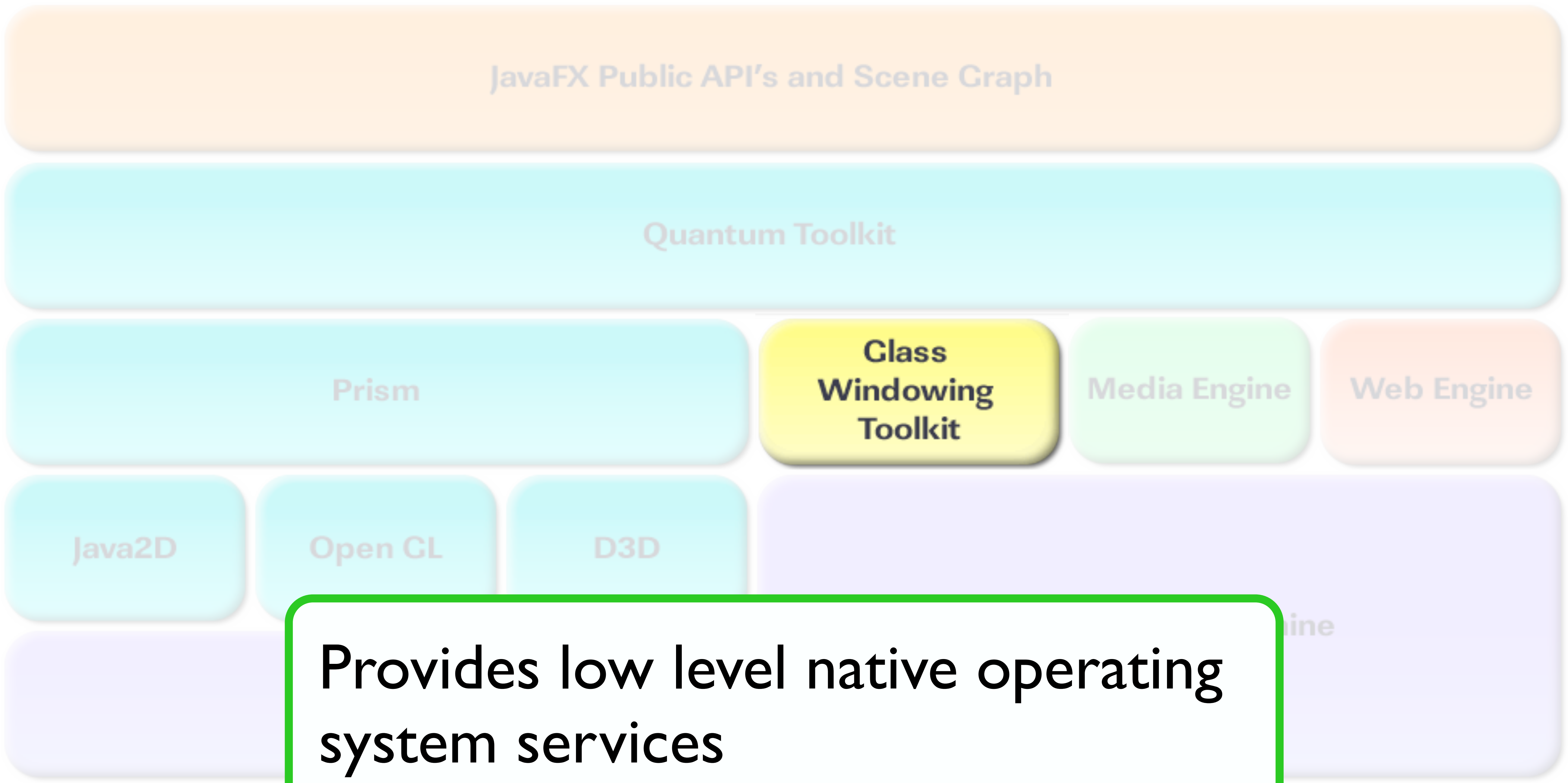
DirectX 9 on Windows XP, Vista
DirectX 11 on Windows 7

The architecture



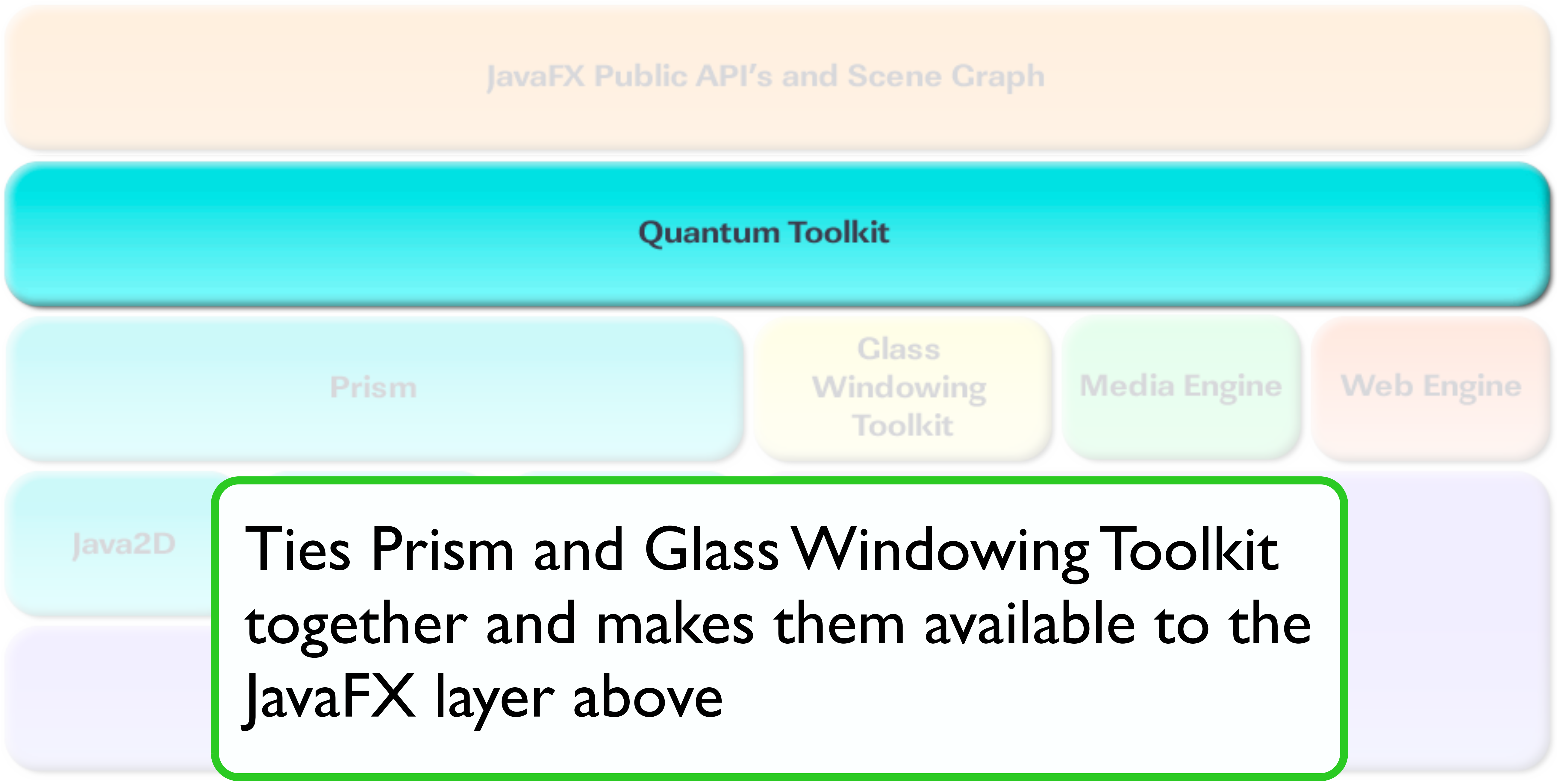
Java2D when hardware acceleration is not possible

The architecture



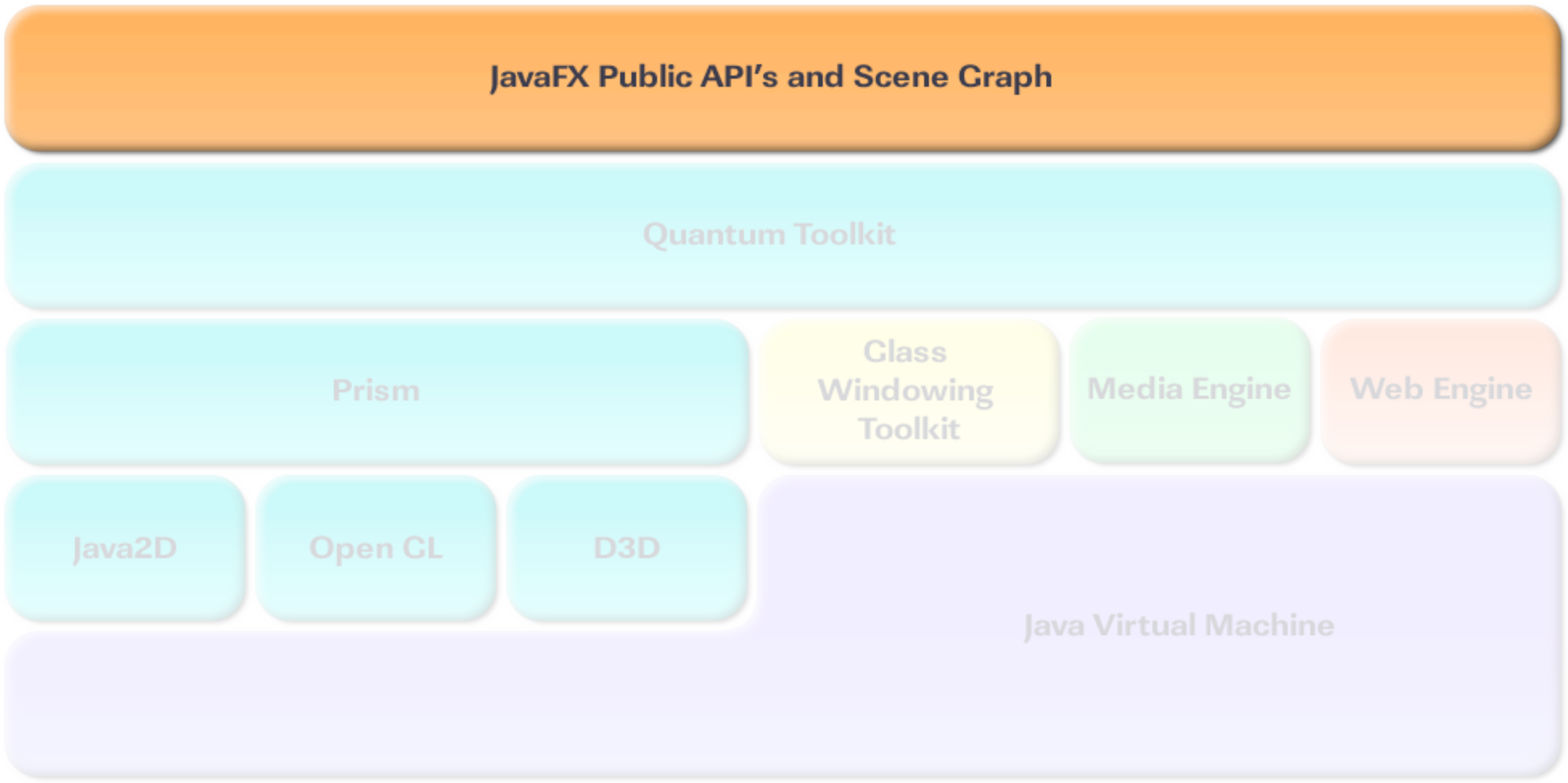
Provides low level native operating system services

The architecture



Ties Prism and Glass Windowing Toolkit together and makes them available to the JavaFX layer above

The architecture



Open Source

* JAVAFX SOURCE CODE IS PART OF THE OPEN JFX PROJECT [HTTP://OPENJDK.JAVA.NET/PROJECTS/OPENJFX/](http://openjdk.java.net/projects/openjfx/)

nearly complete open source around 02/2013

Again a new

PLUGIN

Browser Plugin

- FASTER LOADING OF JAVAFX WEB APPS BASED ON PRISM
- PRE-LOADER FOR IMPROVED USER EXPERIENCE

The

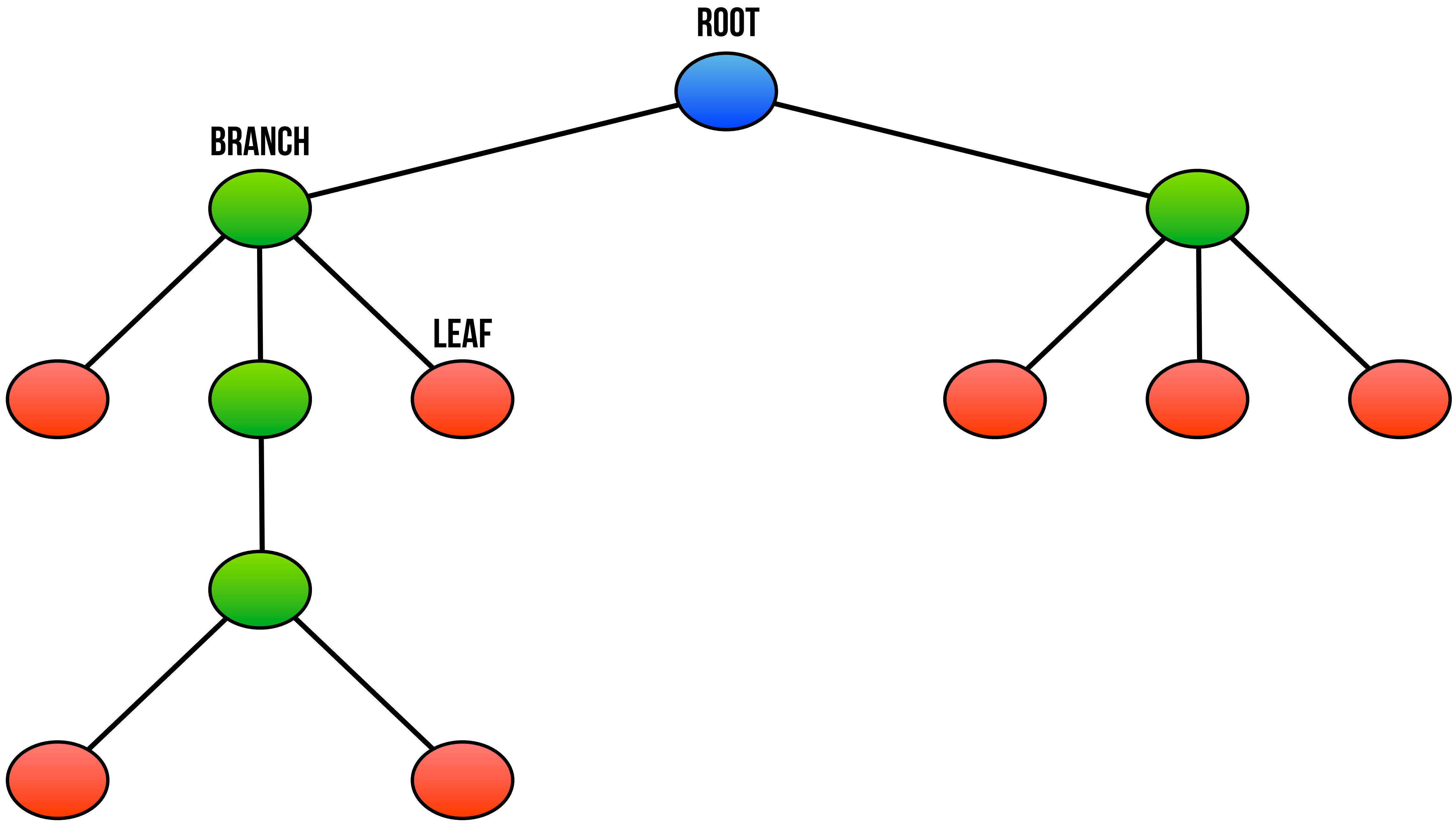
SCENE GRAPH

Collection of
NODES

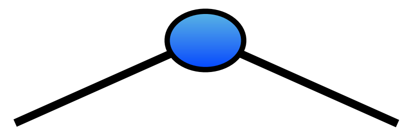
Scene Graph

- HANDLES THE UI
- TREE STRUCTURE
- HAS ONE ROOT NODE
- BRANCH + LEAF NODES

Scene Graph

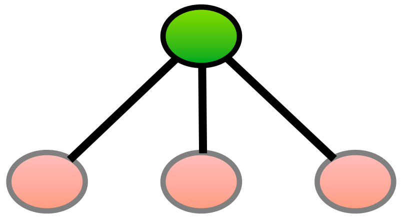


Root Node



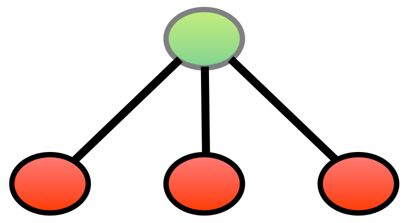
- THE ONLY NODE WITHOUT A PARENT NODE

Branch Nodes



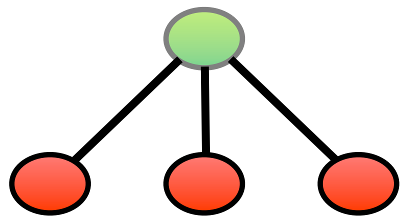
- Are derived from `javafx.scene.Parent`
- Can contain other nodes

Leaf Nodes



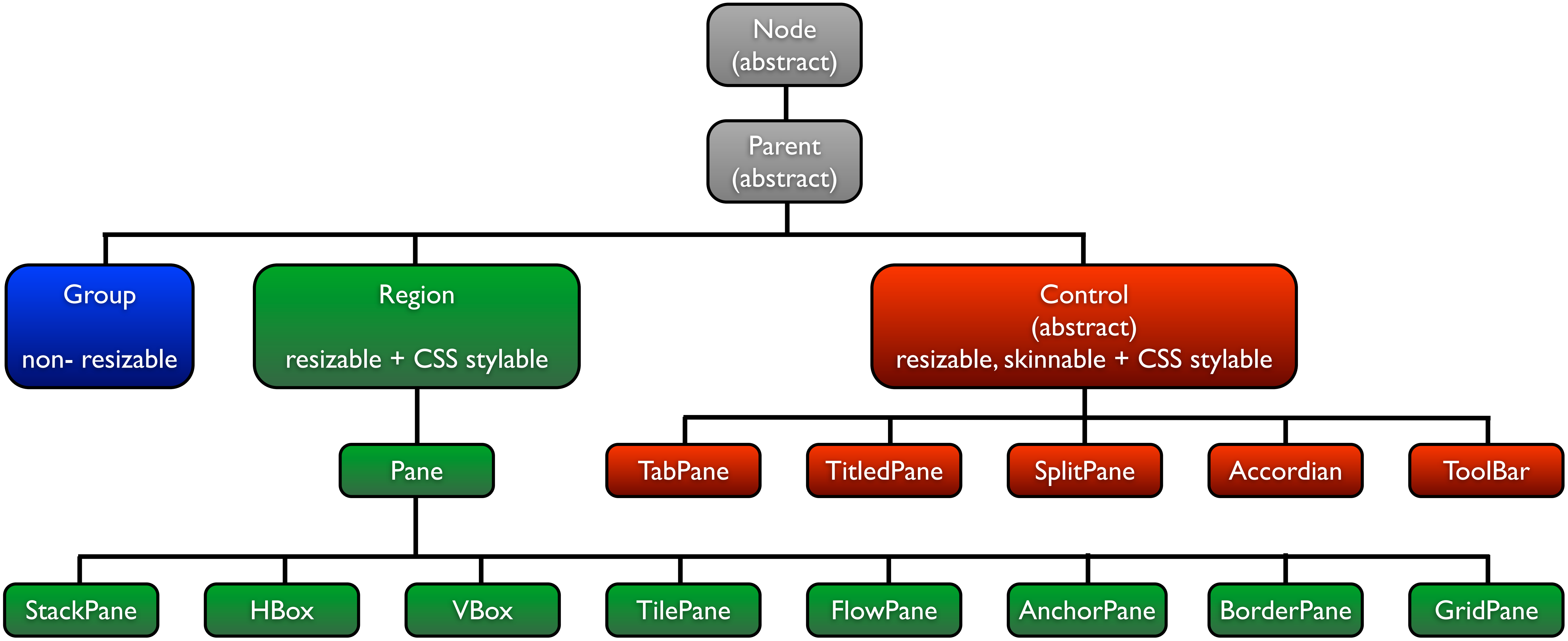
- SHAPES
- IMAGES
- TEXT
- WEBVIEW
- MEDIA
- CONTROLS
- CHARTS

Leaf Nodes

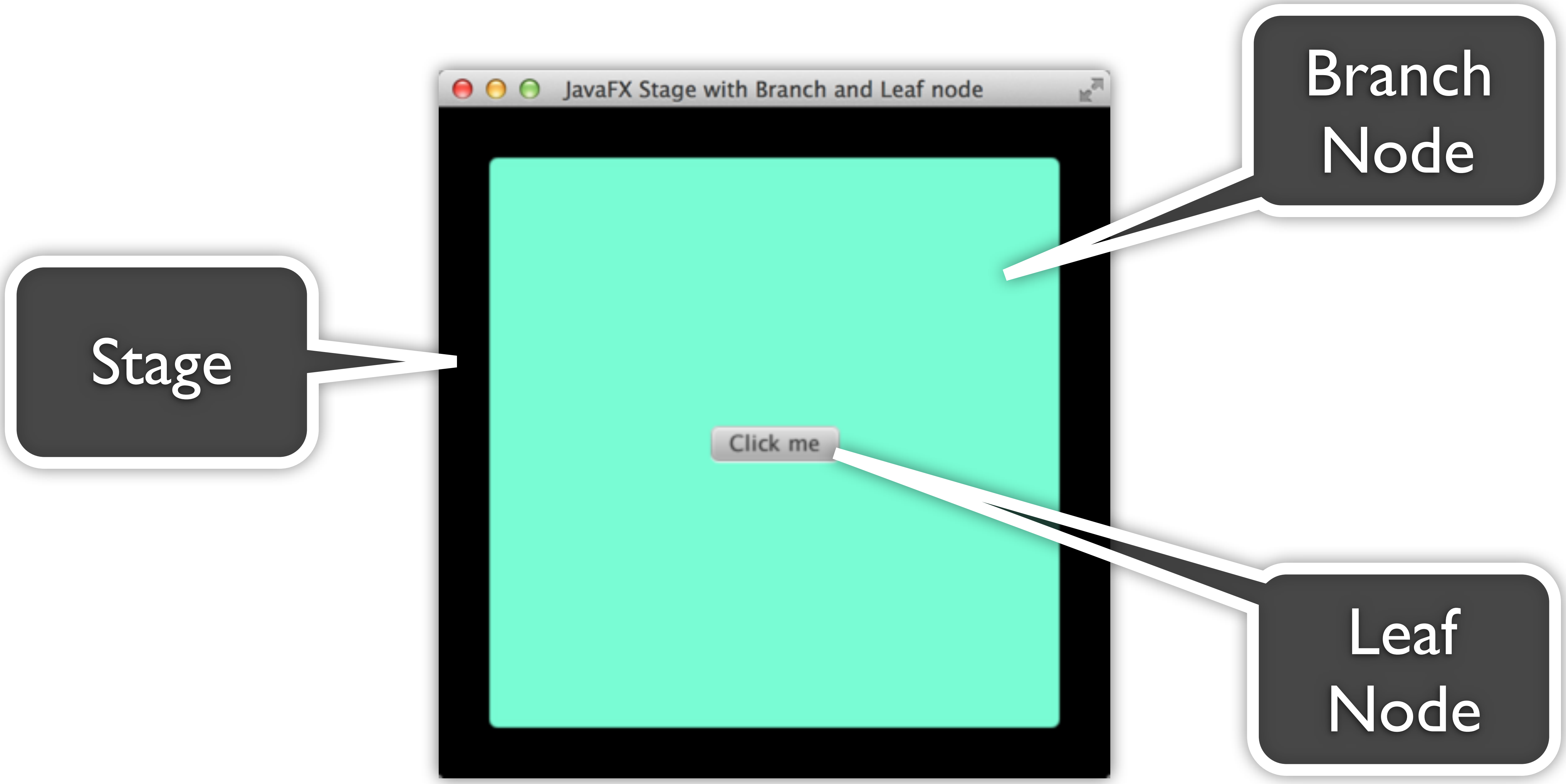


* HAVE NO `getChildren()`

The Nodes


Node
(abstract)
Parent
(abstract)
Group
non- resizable
Region
resizable + CSS stylable
Control
(abstract)
resizable, skinnable + CSS stylable
Pane
TabPane
TitledPane
SplitPane
Accordian
ToolBar
StackPane
HBox
VBox
TilePane
FlowPane
AnchorPane
BorderPane
GridPane

In JavaFx


JavaFX Stage with Branch and Leaf node
Click me
Stage
Branch Node
Leaf Node

Speed limit

60 FPS

Scene Graph

- ROOT NODE IS A PARENT
- STAGE IS CONTAINER FOR ROOT
- ALIVE...NO DEAD BITMAPS

A typical app

```
public class SceneGraphStructure extends Application {
 @Override public void start(Stage stage) {
   stage.setTitle("Hello World");
   Button button = new Button("Say 'Hello World'");
   button.setOnAction(new EventHandler<ActionEvent>() {
     @Override public void handle(ActionEvent evt) {
       System.out.println("Hello World");
     }
   });
   StackPane root = new StackPane();
   root.getChildren().add(button);
   stage.setScene(new Scene(root, 300, 250));
   stage.show();
 }
```

Scene Graph

```
  public static void main(String[] args) {
    launch(args);
  }
}
```

Start JavaFx application

The

JAVA

Api

JavaFx Script is

NOT DEAD

It lives on as

VISAGE

Now we have

PURE JAVA

Some EXAMPLES

Code examples

```
// Java FX 1.x
public def timer = Timeline {
  repeatCount: Timeline.INDEFINITE
    keyframes: KeyFrame {
         time: 1s
       action: function() {...}
}}
```

```
// Java FX 2.x
private Timeline timer =
TimelineBuilder.create()
  .cycleCount(Timeline.INDEFINITE)
  .keyFrames(
    new KeyFrame(Duration.seconds(1),
    new EventHandler() {...}
  ))
  .build();
```

Code examples

```
// Java FX 1.x
view = ImageView {
       image:image
  translateX:bind x + (view.scaleX - 1)
  translateY:bind y + (view.scaleY - 1)
};
```

```
// Java FX 2.x
view = new ImageView(image);
view.translateXProperty().bind(
  x.add(view.getScaleX() - 1));
view.translateYProperty().bind(
  y.add(view.getScaleY() - 1));
```

Properties and BINDINGS

Properties

Java Swing

```java
// Property string
private static final String VALUE_PROPERTY = "value";

// A double property
private double value = 0;

// The getter method
public double getValue() {
  return value;
}

// The setter method
public void setValue(double newValue) {
  double oldValue = value;
  value = newValue;
  firePropertyChange(VALUE_PROPERTY, oldValue, value);
}
```

Properties

```
// A double property
private DoubleProperty value = new SimpleDoubleProperty(0);

// The getter method
public double getValue() {
  return value.get();
}

// The setter method
public void setValue(double newValue) {
  value.set(newValue);
}

// The property method
public DoubleProperty valueProperty() {
  return value;
}
```

JavaFx

Properties

JavaFx

```java
// A double property
DoubleProperty value;

// The getter method
public double getValue() {
  return value.get();
}

// The setter method
public void setValue(double newValue) {
  value.set(newValue);
}

// The property method
public DoubleProperty valueProperty() {
  return value;
}
```

Bindings

- HIGH-LEVEL BINDING
  - FLUENT API
  - BINDINGS CLASS
- LOW-LEVEL BINDING

Bindings

- UNIDIRECTIONAL BINDING

```
bind();
```

- BIDIRECTIONAL BINDING

```
bindBidirectional();
```

High-Level

```java
IntegerProperty number1 = new SimpleIntegerProperty(1);
IntegerProperty number2 = new SimpleIntegerProperty(2);
DoubleProperty  number3 = new SimpleDoubleProperty(0.5);


// High-Level Binding (Fluent API)
NumberBinding sum1    = number1.add(number2);

NumberBinding result1 = number1.add(number2).multiply(number3);



// High-Level Binding (Binding class)
NumberBinding sum2    = Bindings.add(number1, number2);

NumberBinding result2 = Bindings.add(number1, multiply(number2, number3));
```

High-Level

- FLUENT API IS SELFEXPLAINING
- MORE READABLE CODE
- MIGHT BE A BIT SLOWER

Low-Level

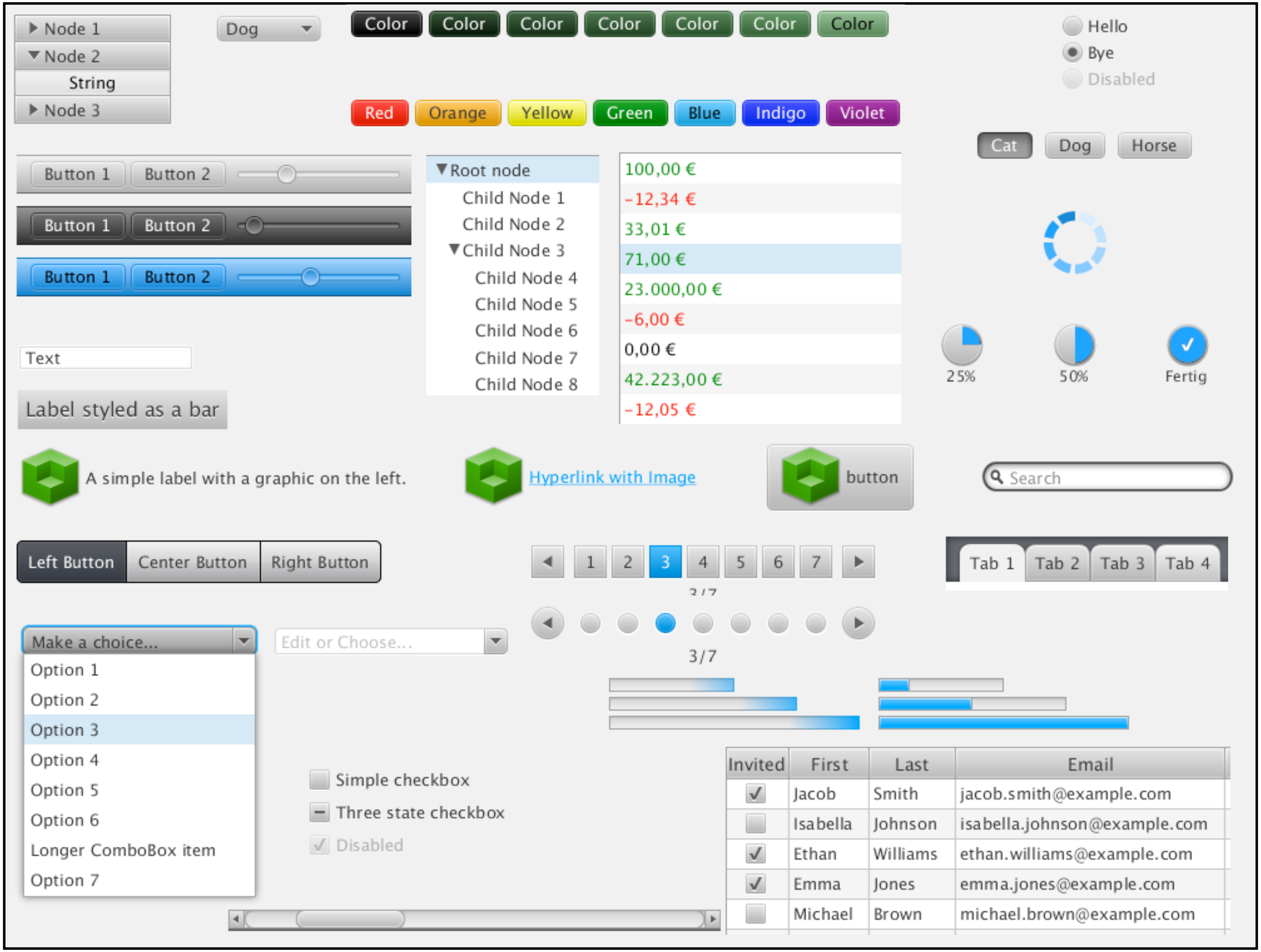
```
IntegerProperty number1 = new SimpleIntegerProperty(1);
IntegerProperty number2 = new SimpleIntegerProperty(2);
DoubleProperty  number3 = new SimpleDoubleProperty(0.5);

// Low-Level Binding
DoubleBinding db = new DoubleBinding() {
  {
    super.bind(number1, number2, number3);
  }

  @Override protected double computeValue() {
    return (number1.get() + number2.get() * number3.get());
  }
}
```

Low-Level

- OVERRIDES A BINDING CLASS
- IS MORE FLEXIBLE
- COULD BE FASTER

JavaFx CONTROLS

Some
EXAMPLES

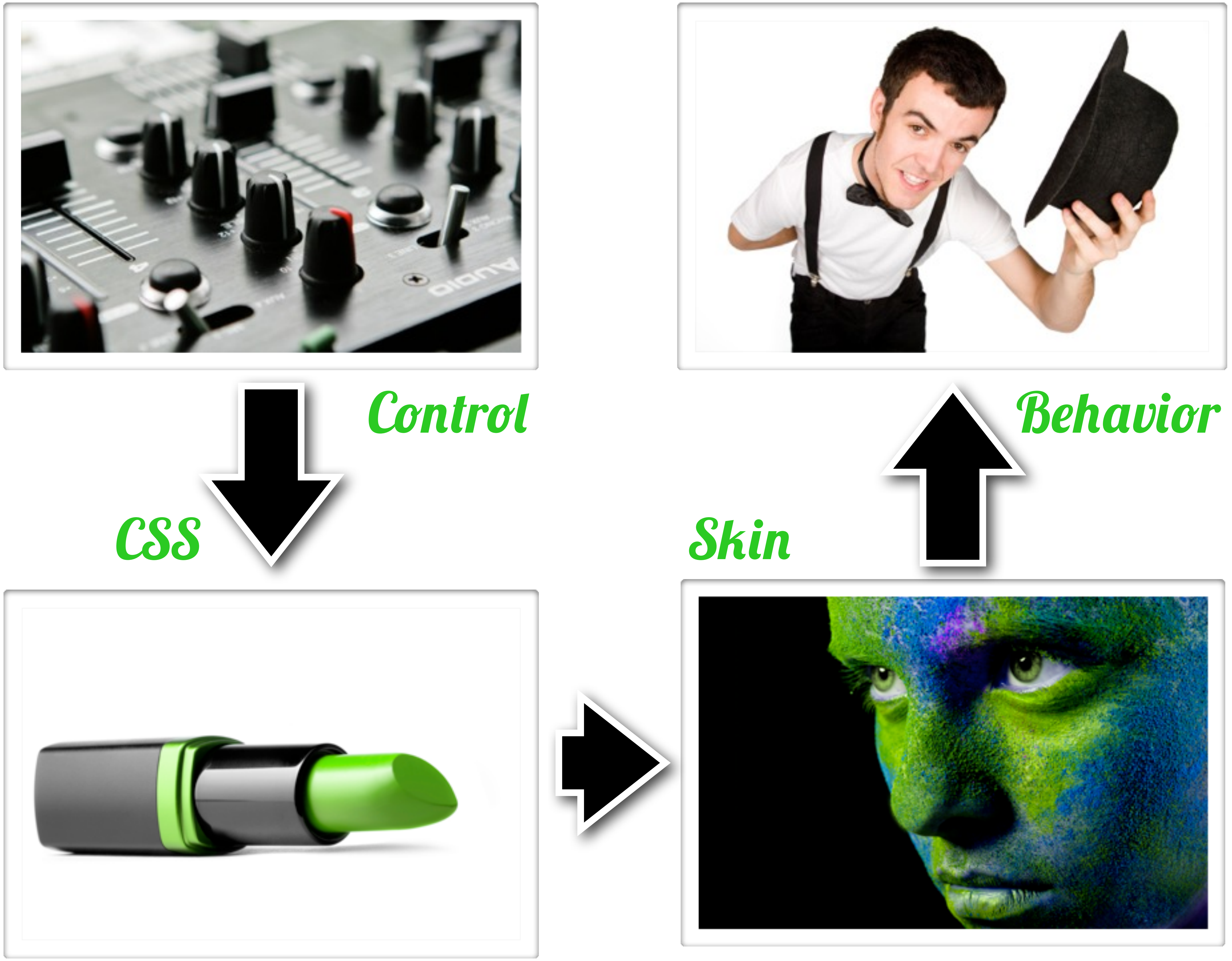


```
100,00 €
-12,34 €
33,01 €
71,00 €
23.000,00 €
-6,00 €
0,00 €
42.223,00 €
-12,05 €
```

Invited	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

Control structure

- CONTROL
- SKIN
- BEHAVIOR
- CSS


Control
CSS
Skin
Behavior

Styling with CSS

Remember

LOOK + FEELS

in Swing ?

Forget them...

Using CSS

- ONE DEFAULT CSS CASPIAN.CSS FOR ROOT AND CONTROLS
- JAVAFX CSS IS BASED ON W3C CSS 2.1 + SOME ADDITIONS

Using CSS

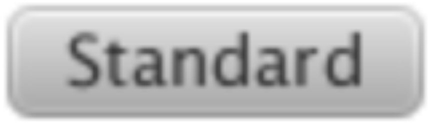
- EITHER OVERRIDE THE DEFAULTS TO STYLE YOUR APP
- OR APPLY YOUR OWN CSS FILE

The Caspian.css

```
.root {
    -fx-base          : #d0d0d0;
    -fx-background    : #f4f4f4;
    -fx-color         : -fx-base;
    -fx-hover-base    : ladder(-fx-base,
                               derive(-fx-base, 20%) 20%,
                               derive(-fx-base, 30%) 35%,
                               derive(-fx-base, 40%) 50%);
    -fx-pressed-base: derive(-fx-base, -20%);
    -fx-focused-base: -fx-base;
    -fx-body-color    : linear-gradient(to bottom,
                                        derive(-fx-color, 34%) 0%,
                                        derive(-fx-color, -18%) 100%);
    ...
```

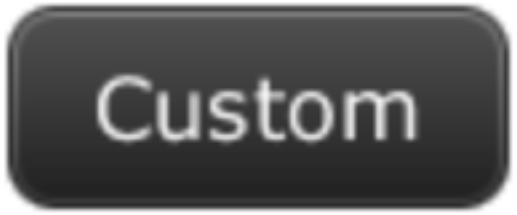
The default CSS

```
.button {
    -fx-skin              : "com.sun.javafx.scene.control.skin.ButtonSkin";
    -fx-background-color : -fx-shadow-highlight-color, -fx-outer-border,
                           -fx-inner-border, -fx-body-color;
    -fx-background-insets: 0 0 -1 0, 0, 1, 2;
    -fx-background-radius: 5, 5, 4, 3;
    -fx-padding           : 0.166667em 0.833333em 0.25em 0.833333em;
    -fx-text-fill         : -fx-text-base-color;
    -fx-alignment         : CENTER;
    -fx-content-display  : LEFT;
}
```

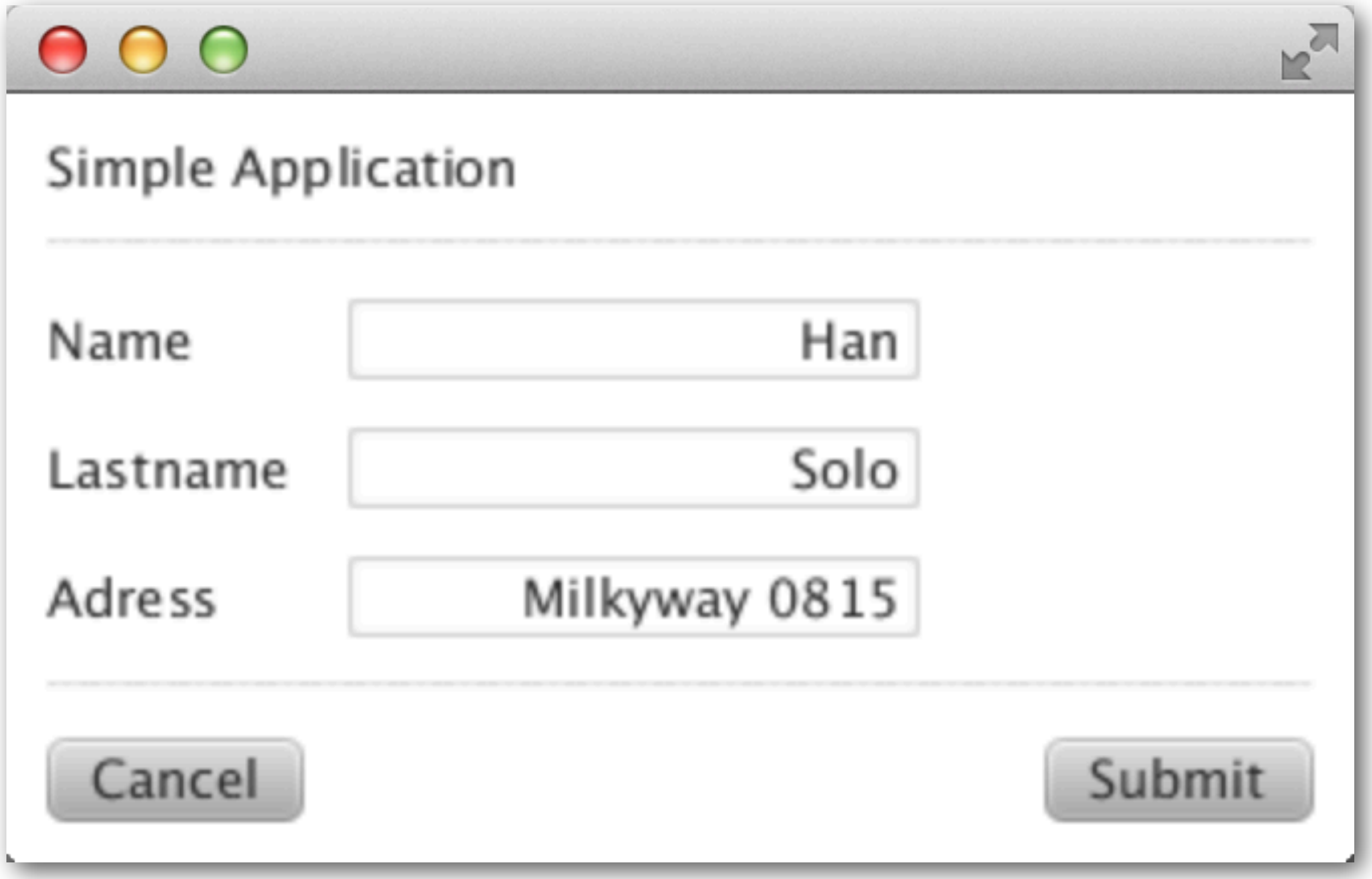


The custom CSS

```
.root {
    -fx-base: #252525; /* scene.getRoot().setStyle("-fx-base: #252525"); */
}

.button {
    -fx-font-family       : "Verdana";
    -fx-font-size         : 16px;
    -fx-background-radius: 9, 9, 8, 7;
    -fx-padding           : 9px 16px 9px 16px;
}
```

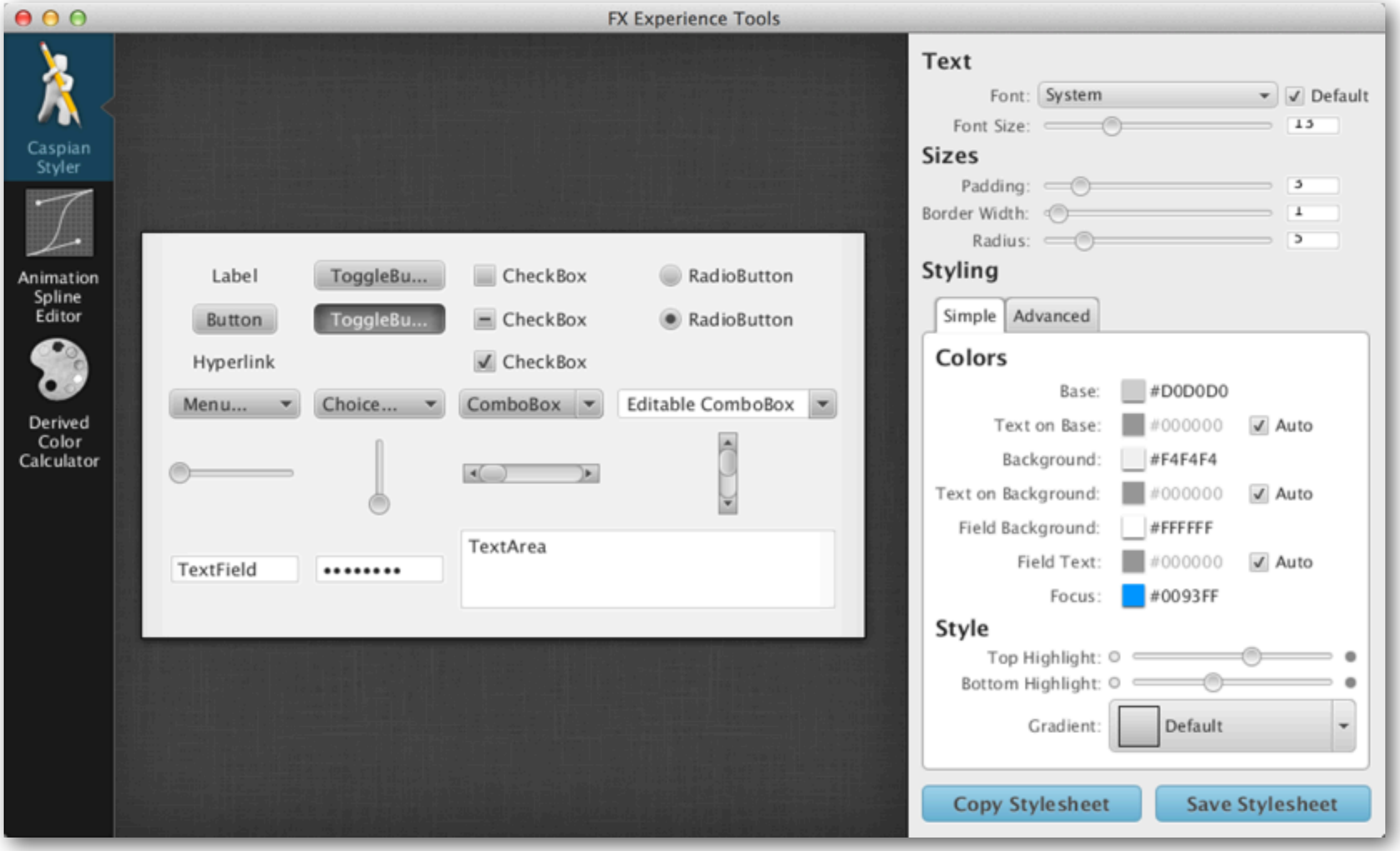


A simple app


Simple Application

Name	Han
Lastname	Solo
Adress	Milkyway 0815

Caspian Styler



Text

Font: System ✓ Default

Font Size: 13

Sizes

Padding: 3

Border Width: 1

Radius: 5

Styling

Colors

Base:	#D0D0D0	
Text on Base:	#000000	✓ Auto
Background:	#F4F4F4	
Text on Background:	#000000	✓ Auto
Field Background:	#FFFFFF	
Field Text:	#000000	✓ Auto
Focus:	#0093FF	

Style

Top Highlight:

Bottom Highlight:

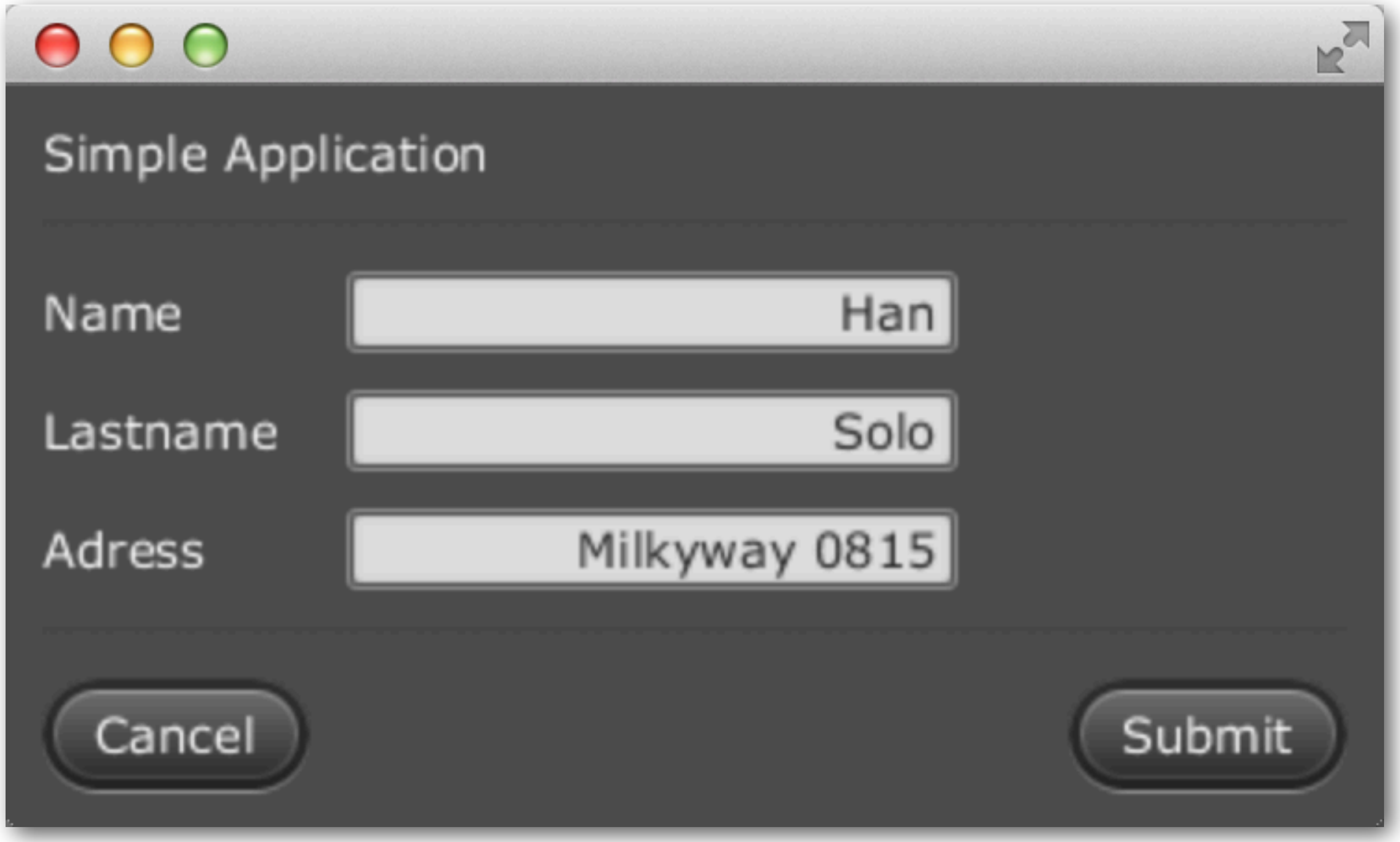
Gradient: Default

Apply some CSS

```
Scene scene = new Scene(pane, Color.rgb(75, 75, 75));
scene.getStylesheets().add("file:///customStylesheet.css");
```

```css
.root {
    -fx-font-family               : "Verdana";
    -fx-font-size                 : 13.0px;
    -fx-base                      : #363636;
    -fx-background                : #5C5C5C;
    -fx-focus-color               : #FF001B;
    -fx-control-inner-background: #DCDCDC;
    -fx-inner-border              : linear-gradient(to bottom, derive(-fx-color, 90.23825953613186%) 0%,
                                                                derive(-fx-color, 17.136566353587632%) 100%);
    -fx-body-color                : linear-gradient(to bottom, derive(-fx-color, 45.810810810810811%) 0%,
                                                                derive(-fx-color, -9.603603603603602%) 100%);
}
.button {
    -fx-background-radius         : 30, 30, 29, 28;
    -fx-padding                   : 7px 14px 7px 14px;
}
.label {
    -fx-padding                   : 7px 22px 7px 14px;
}
.label {
    -fx-padding                   : 7px 8px 7px 10px;
}
.text-field {
    -fx-padding                   : 7px 10px 7px 10px;
}
.label {
    -fx-text-fill                 : -fx-text-background-color;
}
.button {
    -fx-background-insets         : 0 0 -1 0, 0, 3, 4;
}
.button:focused {
    -fx-background-insets         : -1.4, 0, 3, 4;
}
.separator:horizontal .line {
    -fx-border-color              : derive(-fx-background, -80%) transparent transparent transparent;
}
```

A simple app


Simple Application

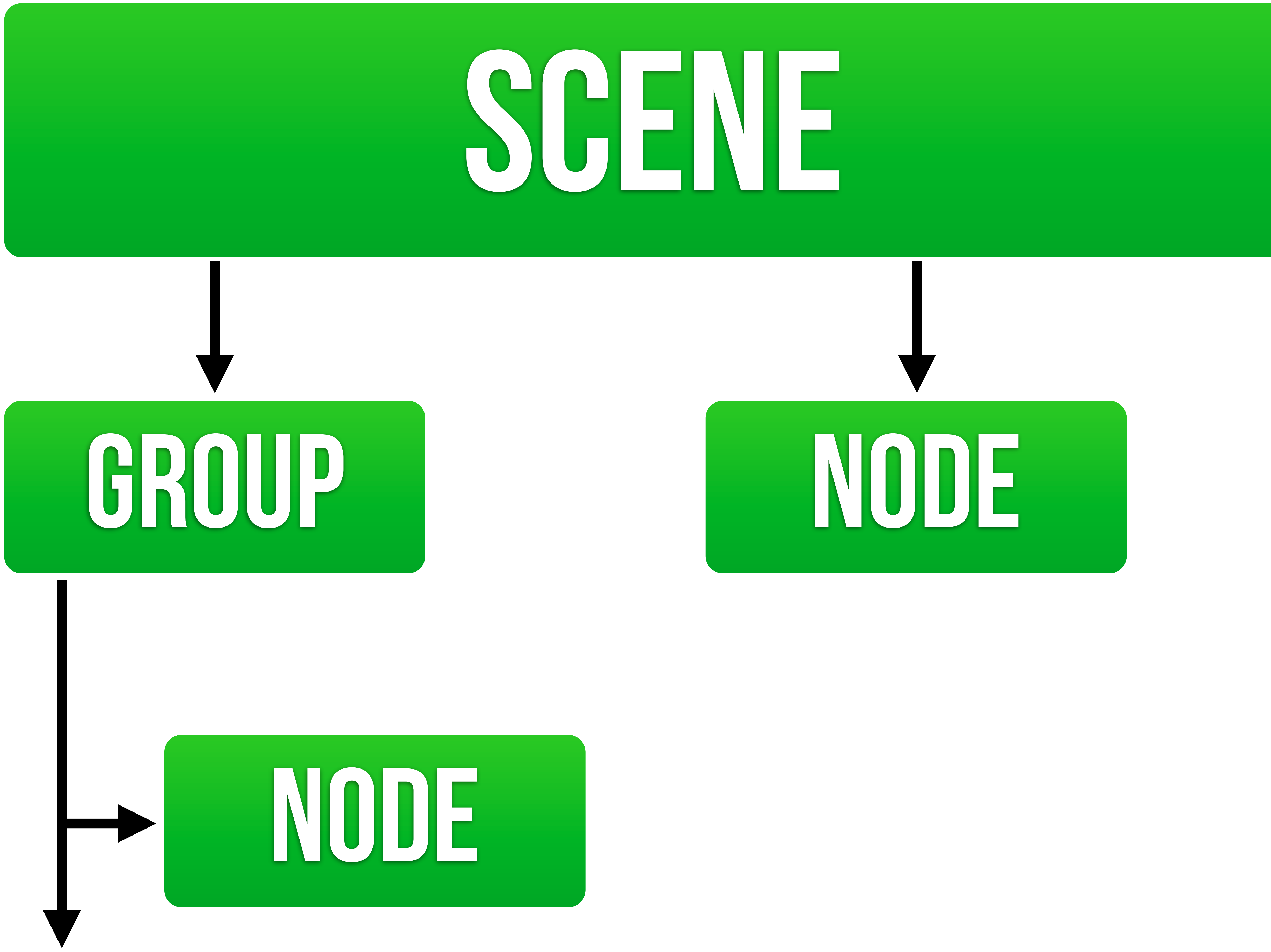
Name: Han

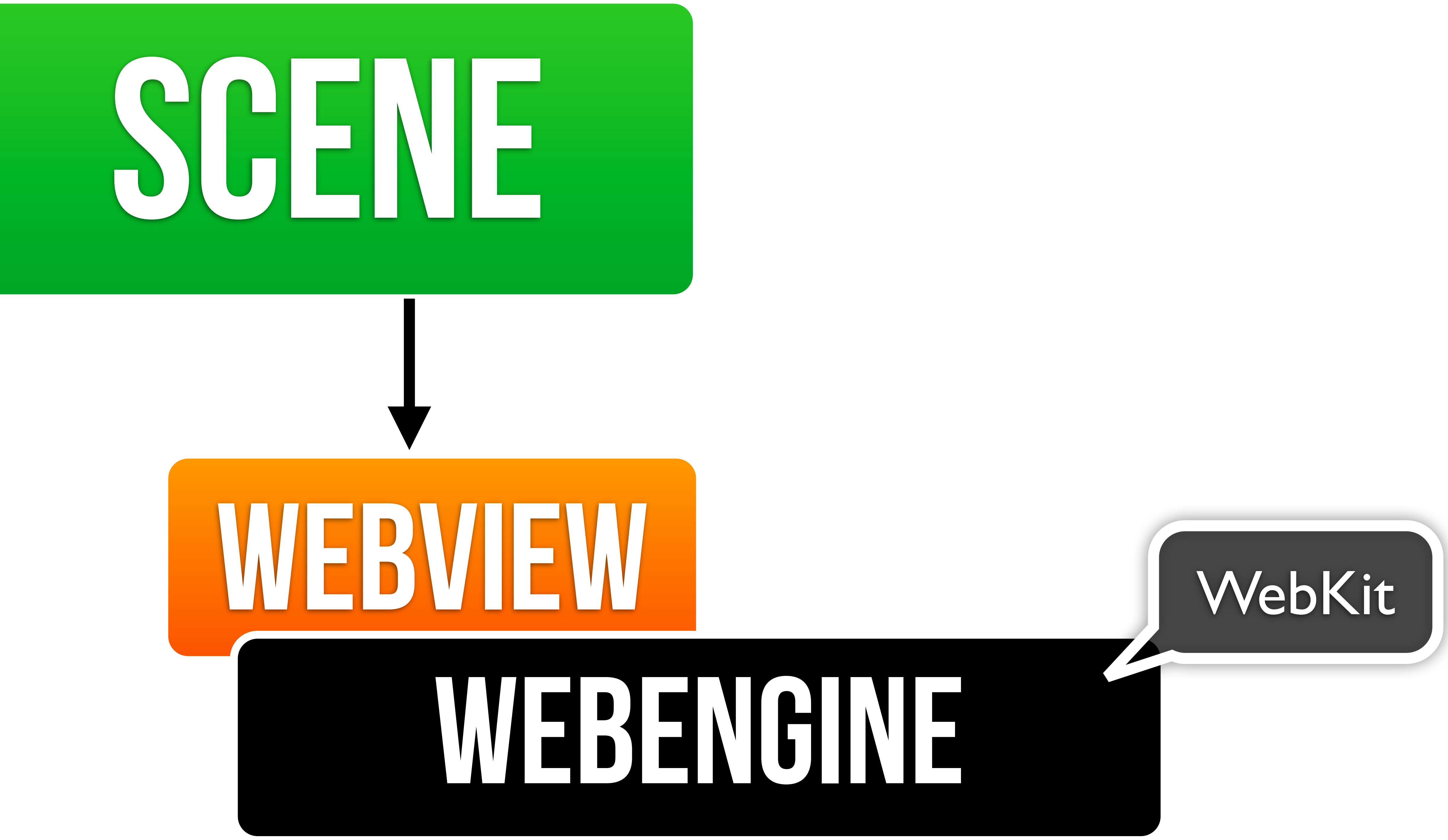
Lastname: Solo

Adress: Milkyway 0815

WebView and

WEBENGINE


SCENE
GROUP
NODE
NODE


SCENE
WEBVIEW
WEBENGINE
WebKit

WebView

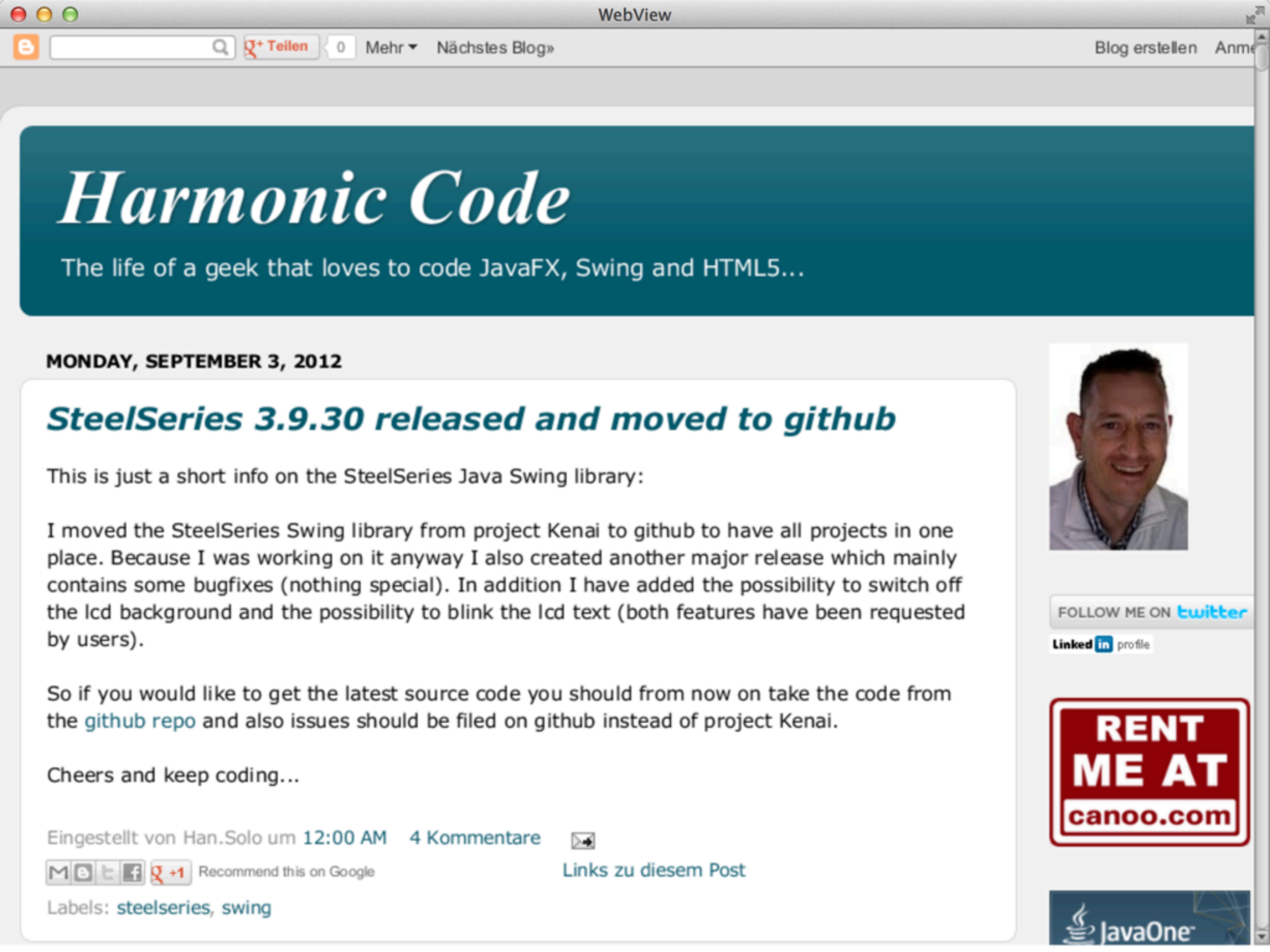
- EXTENSION OF NODE
- ENCAPSULATES WEBENGINE
- INCORPORATES HTML INTO THE SCENE

WebEngine

- PROVIDES WEBPAGE FUNCTION
- SUPPORTS USER INTERACTION
- ENABLES DOM ACCESS AND JS

WebView

```
stage.setTitle("WebView");

WebView   browser = new WebView();
WebEngine engine  = browser.getEngine();
engine.load("http://harmonic-code.org");

StackPane pane = new StackPane();
pane.getChildren().add(browser);
stage.setScene(new Scene(pane, 980, 720));
stage.show();
```



Harmonic Code

The life of a geek that loves to code JavaFX, Swing and HTML5...

MONDAY, SEPTEMBER 3, 2012

SteelSeries 3.9.30 released and moved to github

This is just a short info on the SteelSeries Java Swing library:

I moved the SteelSeries Swing library from project Kenai to github to have all projects in one place. Because I was working on it anyway I also created another major release which mainly contains some bugfixes (nothing special). In addition I have added the possibility to switch off the lcd background and the possibility to blink the lcd text (both features have been requested by users).

So if you would like to get the latest source code you should from now on take the code from the github repo and also issues should be filed on github instead of project Kenai.

Cheers and keep coding...

Eingestellt von Han.Solo um 12:00 AM 4 Kommentare

Recommend this on Google Links zu diesem Post

Labels: steelseries, swing

What about INTERACTION

How JavaFx

INTERACTS

with Html5

Interaction

```
<head>
  <title>MyPage</title>
  <script type="text/javascript">
        var gauge;
        ...
  </script>
...
```

Interaction

```
WebView browser = new WebView();
WebEngine engine = browser.getEngine();
engine.load("http://mypage.html");

// JavaFX interact with WebView
engine.executeScript("gauge.setValue(5)");
```

How Html5 INTERACTS with JavaFx

Listen to
DOM EVENTS

Interaction

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8">
   <title>MyPage</title>
</head>
   <body>
    <button id="buttonId">Click me</button>
  </body>
</html>
```

Interaction

```
// WebView interact with JavaFX (Part I: Document Events)
engine.getLoadWorker().stateProperty().addListener(new ChangeListener<State>() {
  @Override
  public void changed(ObservableValue<? extends State> ov, State old, State now) {
      if (newState == State.SUCCEEDED) {
        Document doc = (Document) engine.executeScript("document");
        EventTarget button = (EventTarget) document.getElementById("buttonId");
        button.addEventListener("click", new DocEventListener(), true);
      }
    }
  });

private static class DocEventListener implements EventListener {
  @Override
  public void handleEvent(Event event) {
    System.out.println("Event received: " + event.getType());
  }
}
```

Listen to

STATUS

Interaction

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8">
  <title>MyPage</title>
  <script type="text/javascript">
      function setStatus(id) {
          window.status = id;
      }
  </script>
</head>
   <body>
     <button id="buttonId">Click me</button>
  </body>
</html>
```

Interaction

```
// WebView interact with JavaFX (Part II: via window.status)
engine.getLoadWorker().stateProperty().addListener(new ChangeListener<State>() {
  @Override
  public void changed(ObservableValue<? extends State> ov, State old, State now) {
    if (newState == State.SUCCEEDED) {
      engine.setOnStatusChanged(new EventHandler<WebEvent<String>>() {
        @Override
        public void handle(WebEvent<String> event) {
          // Get the window.status value
          System.out.println(“Status value: “ + event.getData());
        }
      });
    }
  }
});
```

Inject a

JSOBJECT

JSObject

```
// WebView interact with JavaFX (Part II: via window.status)
class Bridge {
  public void exit() {
    Platform.exit();
  }
}

...

// Inject the JSObject with the name "java" into the html page
JSObject jsobj = (JSObject) webEngine.executeScript("window");
jsobj.setMember("java", new Bridge());

...

// Remove the JSObject again
JSObject.removeMember();
```

JSObject

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8">
  <title>Close JavaFX from HTML</title>
  <script type="text/javascript">
      function closeJavaFXProgram() {
          java.exit();
      }
  </script>
</head>
   <body>
    <button onclick="closeJavaFXProgram()">Close Java</button>
  </body>
</html>
```

Migrating with JFXPANEL

What

IS IT ?

- Behaves like JPanel
- Hosts a JavaFx Scene
- Forwards events
- Should be accessed from the Edt

How DOEZ IT work ?

Creation

```
// Add a JFXPanel to a Swing JFrame
JFrame   frame   = new JFrame("JFXPanel");
JFXPanel fxPanel = new JFXPanel();
frame.add(fxPanel);

Platform.runLater(new Runnable() {
  @Override public void run() {
    initFX(fxPanel);
  }
});
```

Initialization

```
// Initialize the JFXPanel
void initFX(JFXPanel fxPanel) {
    // Code to create a JavaFX scene
  ...

  fxPanel.setScene(scene);
}
```

So you could use

JAVA FX

in Swing...

...means also

HTML 5

in Swing

BAM

But

KEEP

in mind

2 *You have*

UI-THREADS

It's up to you to

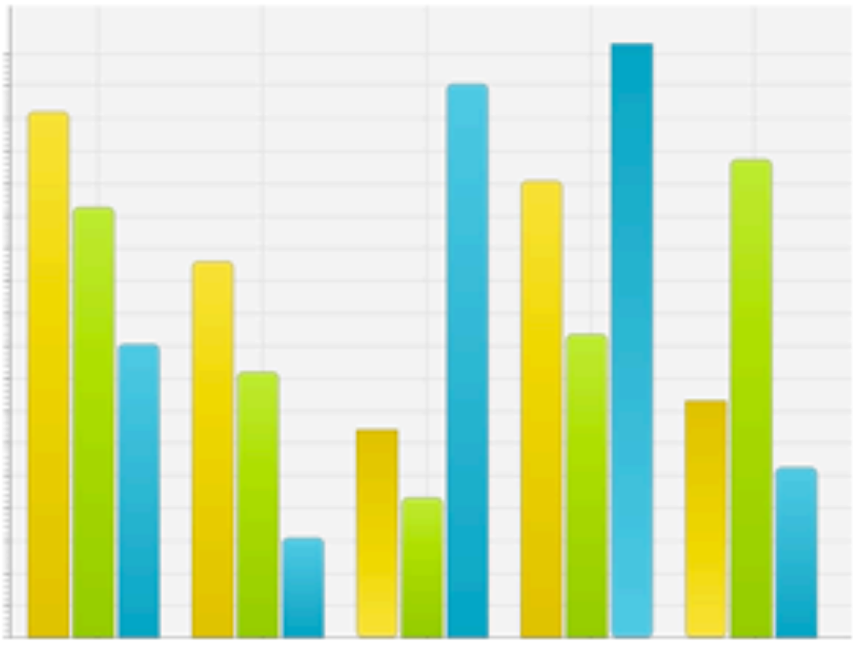
SYNCRONIZE

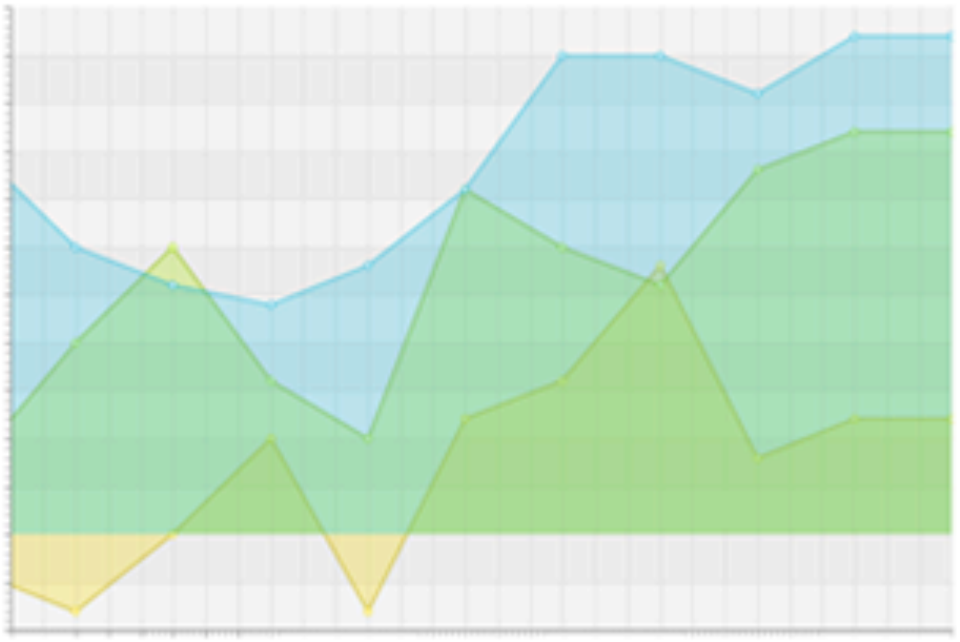
them manualy

JavaFx

CHARTS

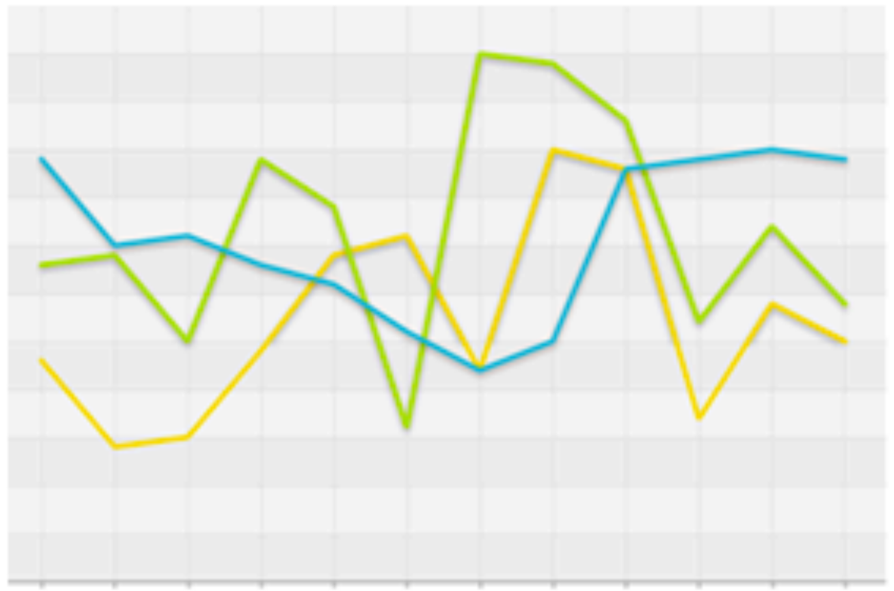
BAR

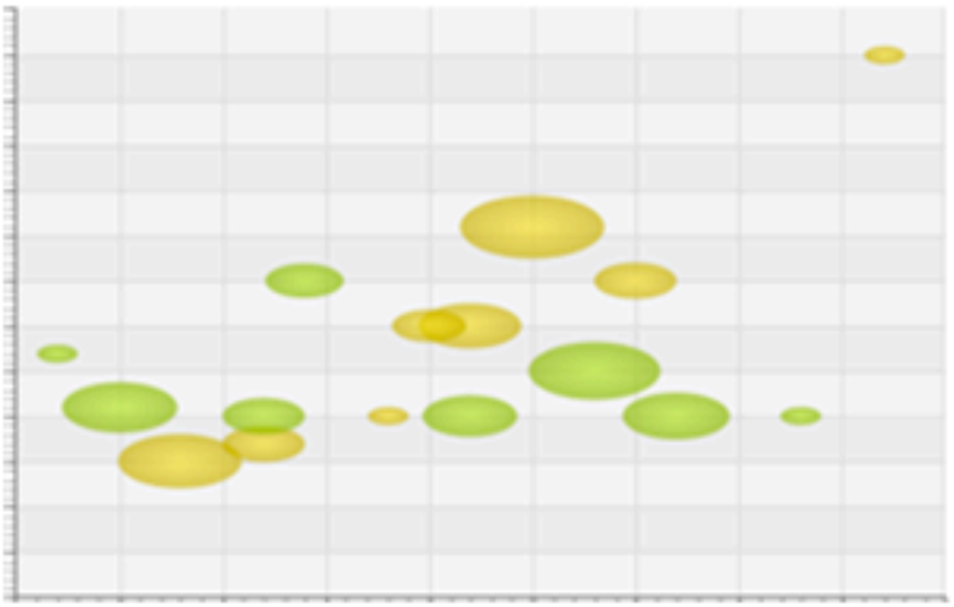




AREA

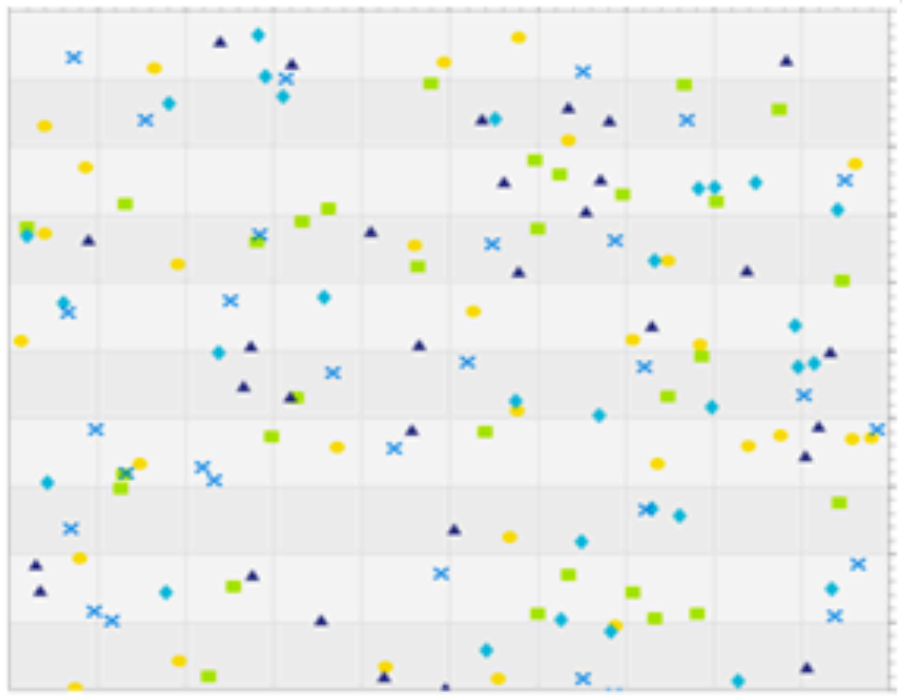
LINE

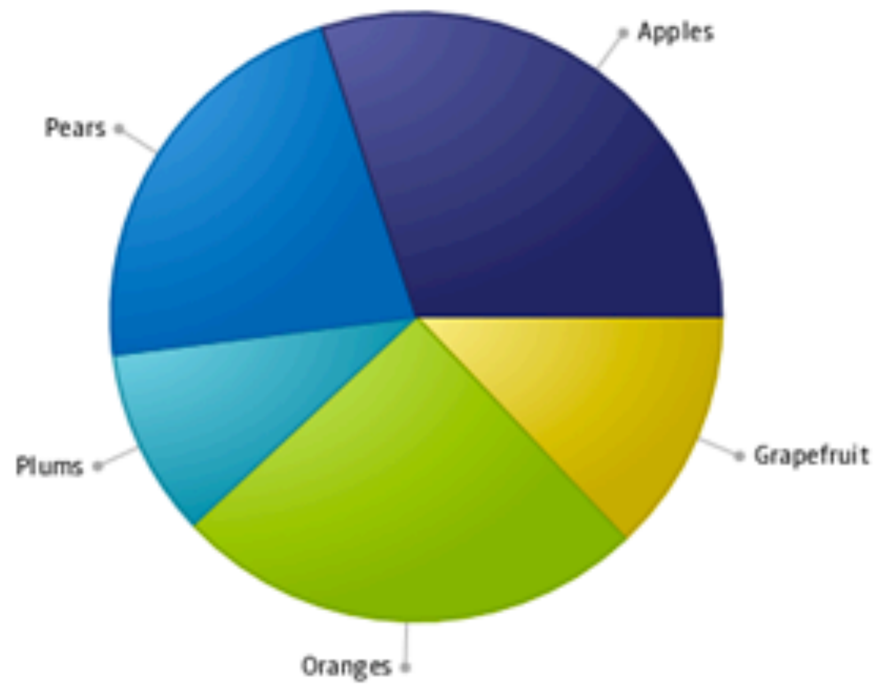




BUBBLE

SCATTER





PIE

JavaFx Charts

- CAN BE ANIMATED
- CAN BE STYLED USING CSS
- CAN BE CUSTOMIZED

Creating a Piechart

```
@Override public void start(Stage stage) {
  Scene scene = new Scene(new Group(), 500, 500);
  stage.setTitle("Imported fruits");

  ObservableList<PieChart.Data> pieChartData =
    FXCollections.observableArrayList(
      new PieChart.Data("Grapefruit", 13),
      new PieChart.Data("Oranges", 25),
      new PieChart.Data("Plums", 10),
      new PieChart.Data("Pears", 22),
      new PieChart.Data("Apples", 30));
  final PieChart chart = new PieChart(pieChartData);
  chart.setTitle("Imported Fruits");

  ((Group) scene.getRoot()).getChildren().add(chart);
  stage.setScene(scene);
  stage.show();
}
```

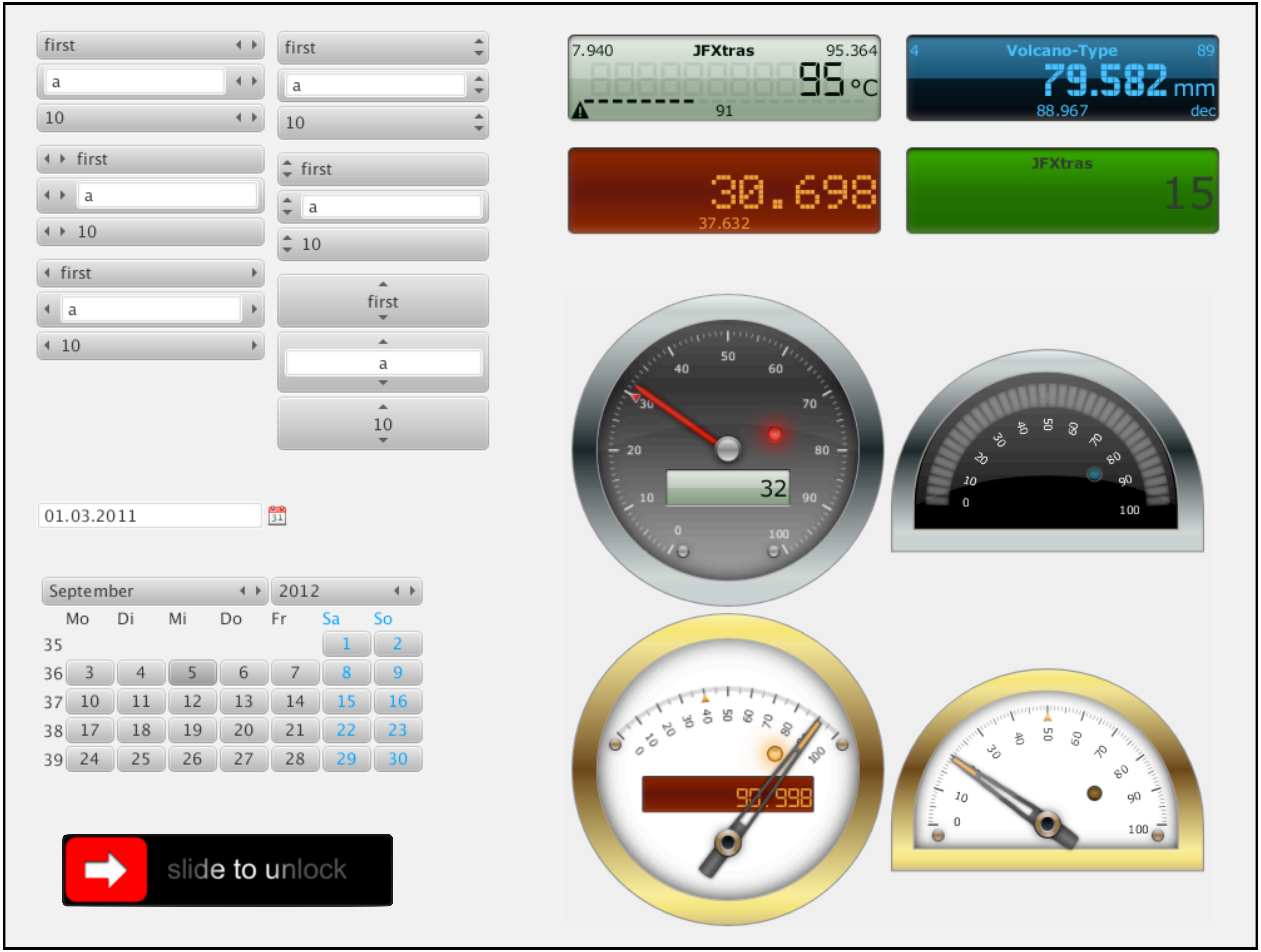
Need more

CONTROLS ?

here you go

JFXTRAS

Some
EXAMPLES


first
a
10
first
a
10
first
a
10
first
a
10
first
a
10
first
a
10
01.03.2011
September
2012
Mo Di Mi Do Fr Sa So
slide to unlock
7.940 JFXtras 95.364
95 °C
91
4 Volcano-Type 89
79.582 mm
88.967 dec
30.698
37.632
JFXtras
15
32
90.998

YOU WANNA BE PART OF

THE PARTY ?

WE WANT YOU AT JFXTRAS

What's new in

JAVA FX 8

JavaFx 8

- SUPPORT FOR EMBEDDED
- 3D SUPPORT
- SWING-NODE (HOPEFULLY)
- PUBLIC API FOR CONTROLS
- PERFORMANCE++
- NO FOCUS ON PLUGIN ANYMORE

Keep coding...