Introduction to JavaFX

A rich client platform for all screens

Richard Bair
Sun Microsystems, Inc.

Introduction to JavaFX

- JavaFX is the next generation client stack for the Java Platform
  - Consumer & Enterprise
  - Consistent cross-device API and development model
- Designed for high performance graphics on desktop, mobile, tv
  - Leverages OpenGL, Direct3D, SSE3 when possible on any target device
  - Ground-up rewrite of fonts, image handling, rasterization
- Designed for multi-language support
  - Java
  - JavaFX Script
  - more...

Outline

- JavaFX Script – a new programming language
- JavaFX scene graph
- User interface controls
- Styling
- Charts
- Layout
- Developer tools

What is JavaFX Script?

“DSL for the care and feeding of the Scenegraph”

Null Pointer-less Language

Expression Language

Object Oriented

JVM Language

Strongly Typed

Type Inference

Javascript / Scala like syntax

Why A New Language?

- A programming language is not about telling the computer what to do, but instead is about expressing the programmer's intent.
- A programming language needs to embody new, higher-level concepts and to abstract away irrelevant detail. (Brooks 1993, HOPL-II keynote)
- JavaFX Script is tightly integrated with, and works extremely effectively with the JavaFX runtime and scenegraph
- *It's fun!*

```
println("Hello, world!")
```

```
def PI = 3.14159265;
```

```
var name = “Richard”;
```

```
var name:String;
name = “Richard”;
```

Data Types

- Primitive types from Java:
  - Boolean, Integer, Long, String, ...
  - New: string interpolation expressions
    - println("The value of x is {x}");
- Object references (similar to Java)
- Number
- Duration
- Sequences

Sequences

- An ordered collection of objects
- Sequences are flat — they do not nest
- A sequence cannot be null (but it can be empty)

```
var numbers = [3, 1, 4, 1, 5];
insert [9, 2, 6] into numbers;
delete numbers[2];
```

Expressions, For-Loops, and Sequences

- Every “statement” is actually an expression that has a value

```
var b = if (a >= 0) a else -a;
```

- The value of a for-loop is a sequence of values from its body

```
for (x in [1..5]) {
    x * x
}

[1, 4, 9, 16, 25]
```

Classes, Mixins, and APIs

- Classes are normal classes similar to Java classes
- Mixin classes like Java interfaces
  - Can include function implementations
  - Can include variable declarations and initial values
- Extending classes
  - At most one normal superclass
  - Arbitrary number of mixin classes

Object Literals

- Concise “declarative” syntax for object creation
- A series of variable:initial-value pairs
- Can be used on public and public-init variables
- Can be nested arbitrarily
  - Useful for creating scene graph structures

```
var rect = Rectangle {
    x: 10
    y: 20
    width: 30
    height: 40
}
```

Object Literals and Binding

```
var leftMargin = 472;

var r1 = Rectangle {
    x: bind leftMargin
    ...
};

var r2 = Rectangle {
    x: bind leftMargin
    ...
}
```

when leftMargin changes the x-value of both Rectangles changes

JavaFX Library API Style

- The x, y, width, height variables on Rectangle are public!
  - What about encapsulation? Enforcing invariants?
- No getters
  - Variables can be public-read
- No setters
  - Variables are public and have a trigger
- No constructors
  - Variables are public-init allowing use in object literals
- Few listeners and callbacks
  - State variables exposed (public, public-init, or public-read)
  - This allows binding on them

Binds and Triggers

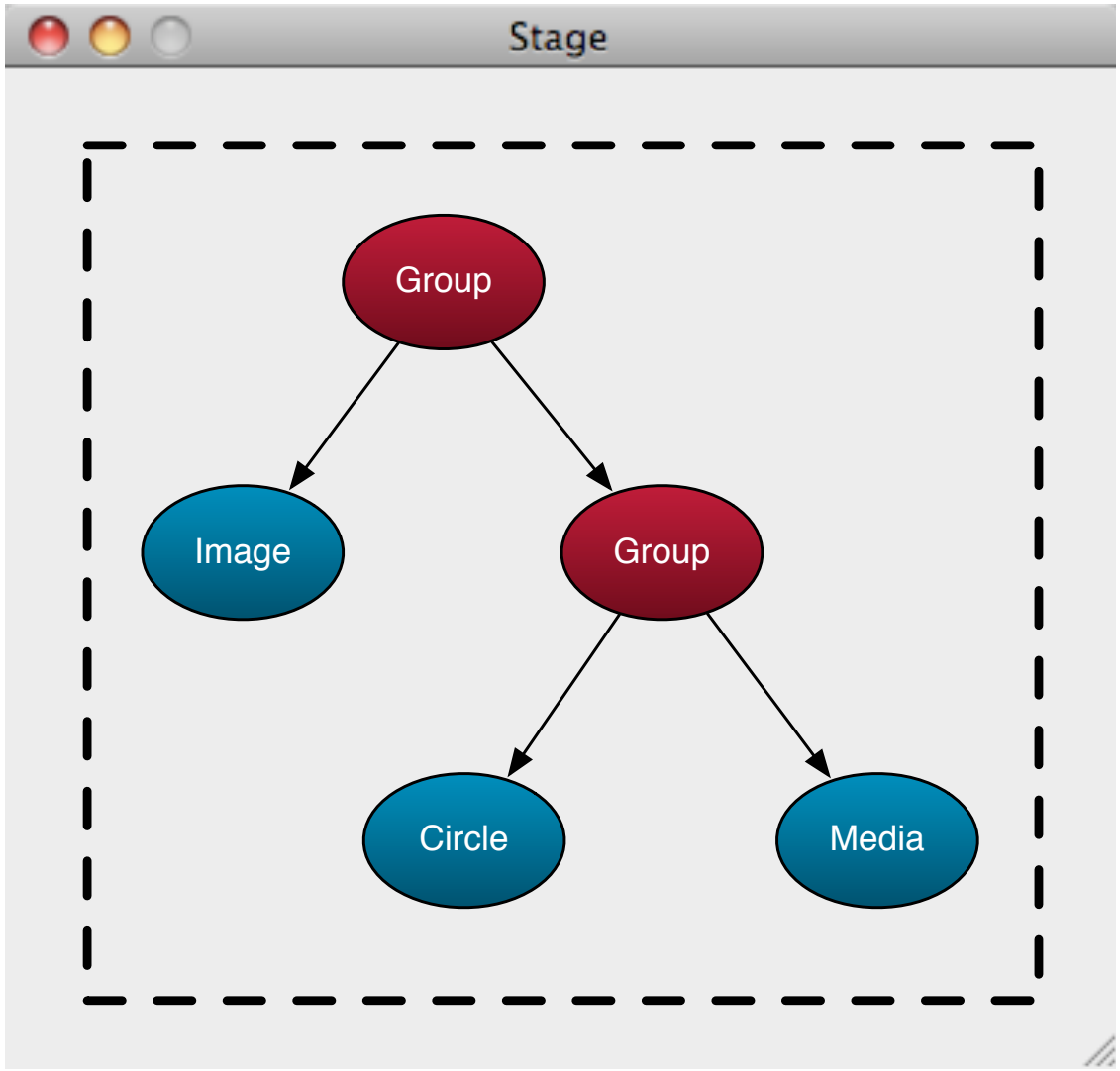
```
public var x1;
public var x2;
public-read var width = bind x2 - x1;


public var radius = 10 on replace {
    diameter = 2 * radius
}
```

Outline

- JavaFX Script – a new programming language
- JavaFX scene graph
- User interface controls
- Styling
- Charts
- Layout
- Developer tools

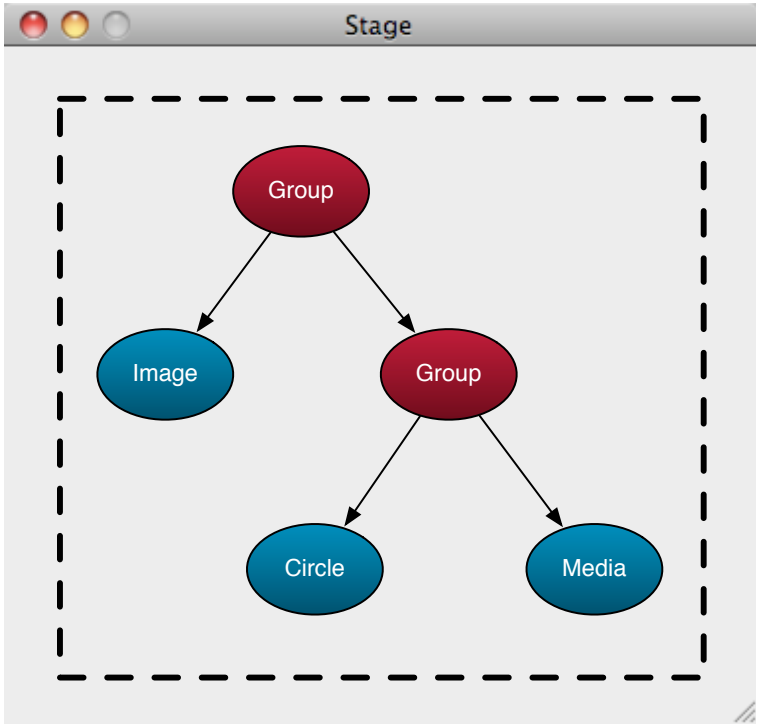
Scenegraph

- Data structure which represents all visual elements
- Easily reference any visual element in the app and manipulate it
- Comprised of Nodes
  - Leaf Nodes (shapes, images, text, etc)
  - Parent Nodes (Groups, Containers, etc)



Scenegraph Sample

```
Group {
  content: [
    ImageView { }
    Group {
      content: [
        Circle { },
        MediaView { }
      ]
    }
  ]
}
```



Nodes

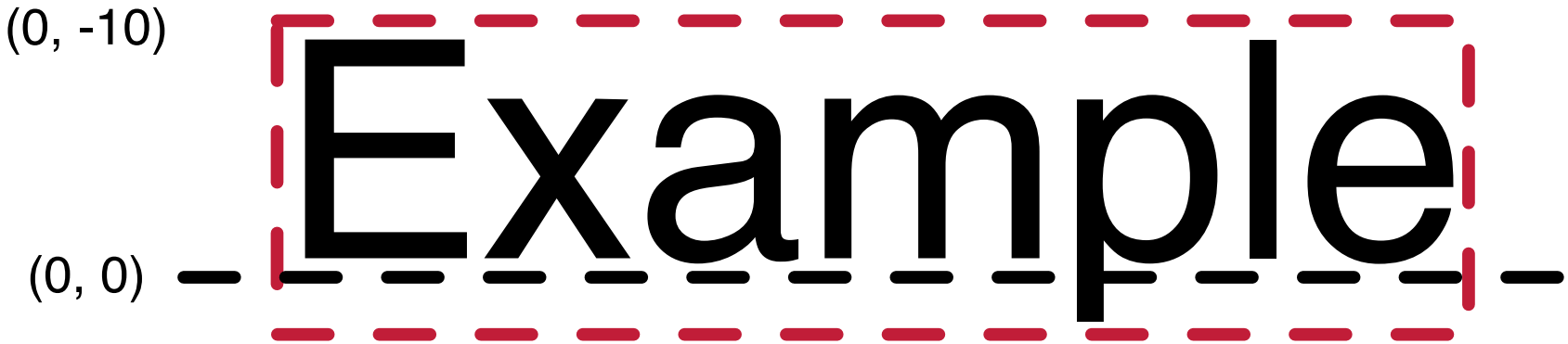
- Group
- CustomNode
- Container
- Control
- Line
- Path
- Rectangle
- ImageView
- MediaView
- Text
- more...

ImageView

- Image represents an in-memory bitmap
  - loaded via URL, from jar
- ImageView Node contains an Image
- Both ImageView and Image can scale
  - Preserve ratio
  - Fit within a specific width/height

Text Node

- x, y, TextOrigin
- Fonts can be specified on Text
- Supports multiline text
- By default, text positioned such that (x, y) is left baseline



Effects

- Any Node can have an Effect
- Many standard built in
  - Blend modes
  - Bloom
  - DisplacementMap
  - DropShadow
  - ColorAdjust
  - BoxBlur
  - Glow
  - Reflection
  - InnerShadow
  - more...

Media

- JavaFX supports both visual and audio media
- Cross platform JavaFX Media file (fxm, mp3)
- Also plays native formats (mov, wmv)
- Media class represents a media file
- MediaPlayer plays a Media file
- MediaView is the Node which displays the Media

Animation

- Animation is a key feature of every rich graphics application platform
- There are two supported animation types in JavaFX
  - Keyframe animations
  - Transitions

KeyFrame Animation

- KeyFrame: specifies that a variable should have...
  - a particular value
  - at a particular time
- Timeline
  - Modifies values of variables specified by KeyFrames
  - Doesn't necessarily do any animation itself
- How is animation actually done?
  - Arrange for a KeyFrame to modify an interesting Node variable
    - x, rotate, opacity, fill, ...

KeyFrame Animation Setup

```
var text = Text {
    x: 50
    y: 80
    content: "Hello, world!"
    rotate: 30
}

Timeline {
    keyFrames: at (4s) { text.rotate => 210.0 }
}.play();
```

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

DEMO – Simple Scene Graph

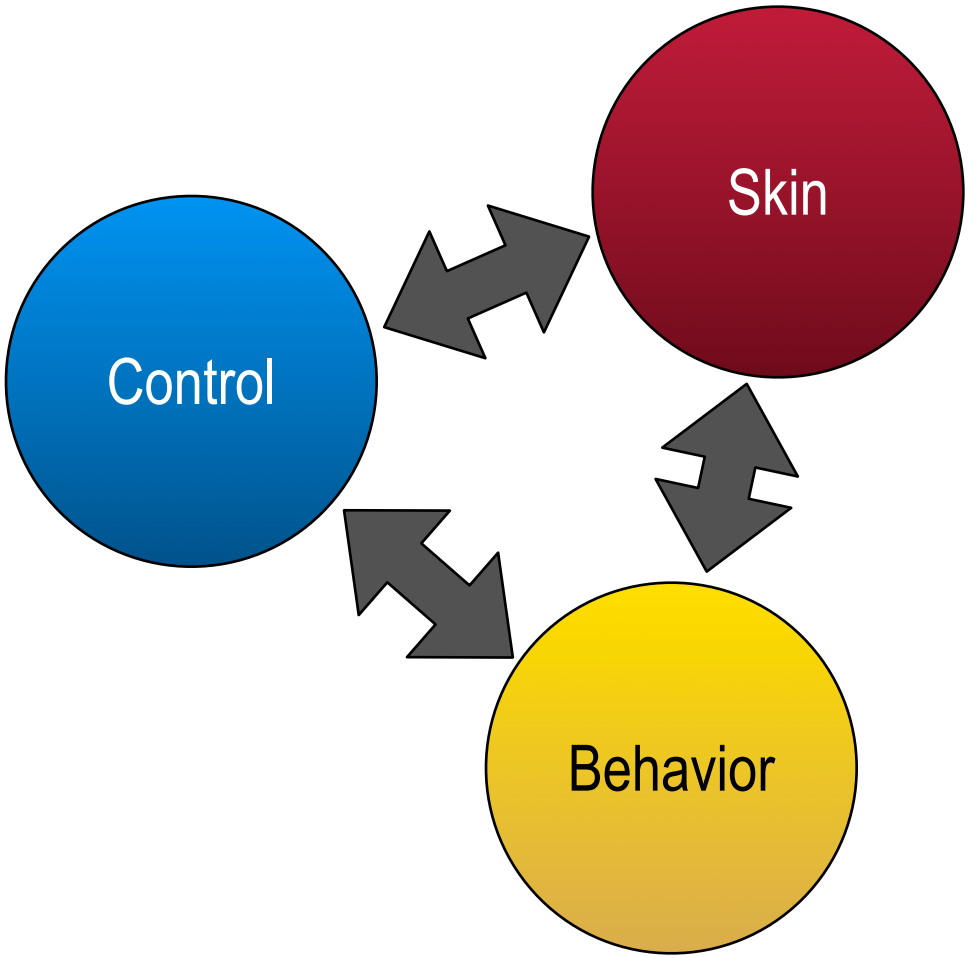
Outline

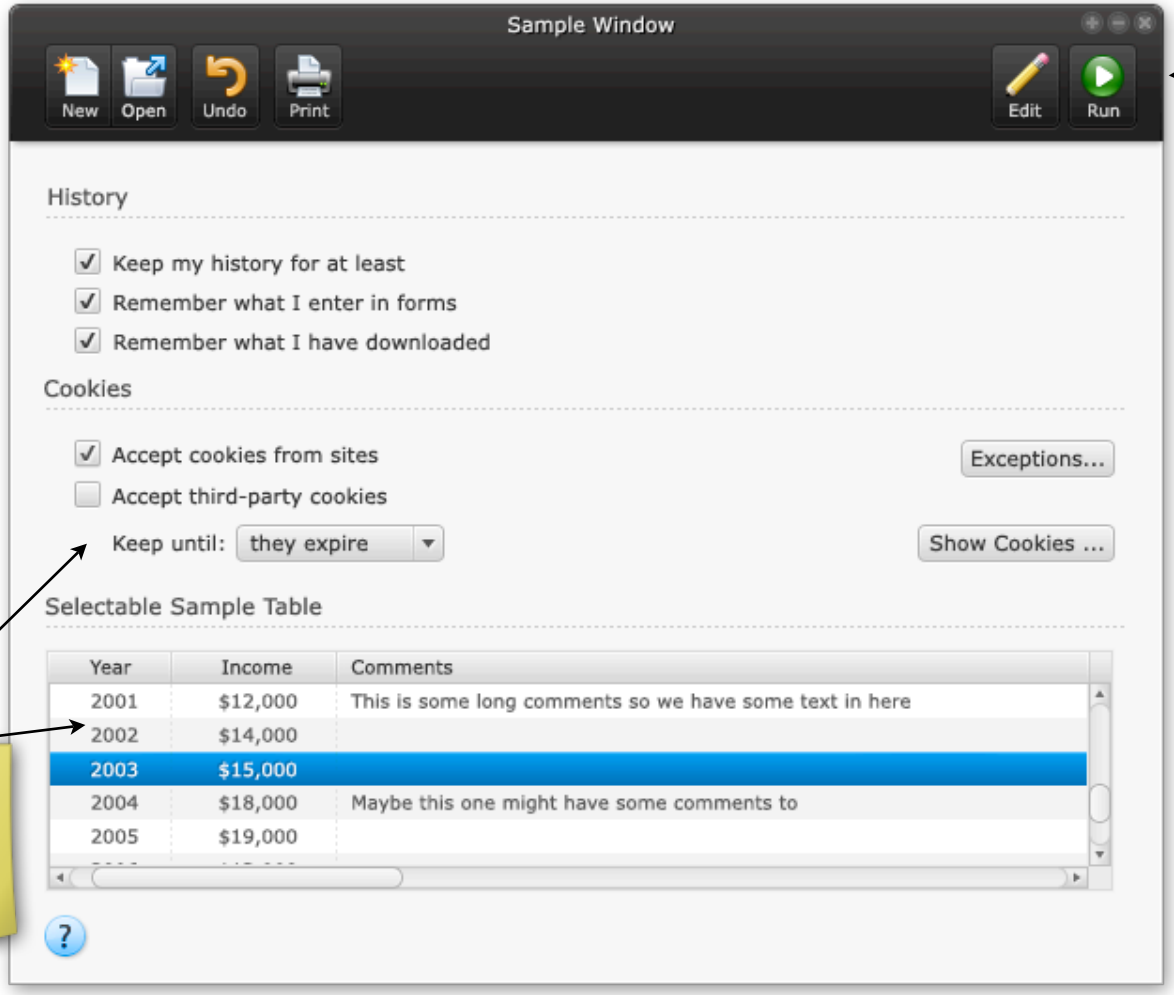
- JavaFX Script – a new programming language
- JavaFX scene graph
- User interface controls
- Styling
- Charts
- Layout
- Developer tools

JavaFX UI Controls

- Simple
- Useful
- Rich

Architecture





Dark color buttons on toolbar

Light color controls on almost white background

Controls in JavaFX

- Button
- ToggleButton
- RadioButton
- CheckBox
- Slider
- Label
- ScrollBar
- Hyperlink
- ProgressIndicator
- ProgressBar
- TextBox
- ListView
- TreeView
- PasswordBox
- ChoiceButton
- MenuButton
- SplitMenuButton
- Menus
- ToolBar
- ScrollView
- Multiline TextBox
- Horizontal ListView
- Popup
- Tooltip

Button

- action:function()
- Example:

```
Button {
    text: “Cancel”
    action: function() {
        println(“I’ve been clicked!”);
    }
}
```

Progress Indicator

- progress:Number (0..1)
- progress bar is-a progress indicator
- Example:

```
var task = HttpRequest { ... }
ProgressIndicator { progress: bind task.percentDone }
```

TextBox

- text:String
- promptText:String
- font:Font
- action:function()
- Example:

```
var t:TextBox = TextBox {
    promptText: “Search”
    action: function() {
        startSearch(t.text);
        t.text = “”;
    }
}
```

Multiline TextBox

- columns:Integer
- lines:Integer
- multiline:Boolean
- Example:

```
var t:TextBox = TextBox {
    columns: 30
    lines: 10
    multiline: true
}
```

List View

- Horizontal or Vertical
- Massively Scalable
- Custom Cells
- Dynamically variable row height
- Example:

```
ListView {
    items: [“Apples”, “Oranges”, “Bananas”]
    cellMaker: function() {
        ListCell { ... }
    }
}
```

DEMO – UI Controls

Outline

- JavaFX Script – a new programming language
- JavaFX scene graph
- User interface controls
- Styling
- Charts
- Layout
- Developer tools

Styling

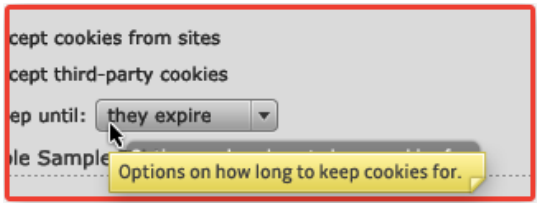
- Easy and Powerful (CSS)
- Highly Customized (fxz)
- Complete Control (code)

Styling

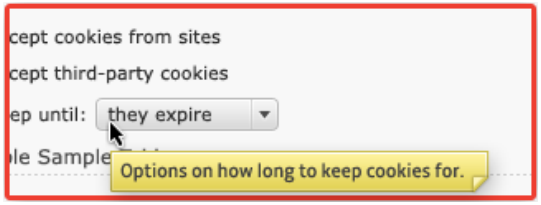
- Easy and Powerful (CSS)
- Highly Customized (fxz)
- Complete Control (code)

Styling

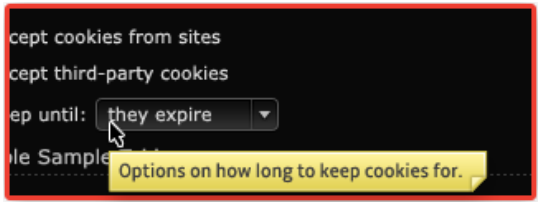
```
Scene {
    base-color: #646464
}

Tooltip {
    background-color: yellow;
    cursor: hand
}
```




```
Scene {
    base-color: #CBCBCB
}

Tooltip {
    background-color: yellow;
    cursor: hand
}
```




```
Scene {
    base-color: #111111
}

Tooltip {
    background-color: yellow;
    cursor: hand
}
```



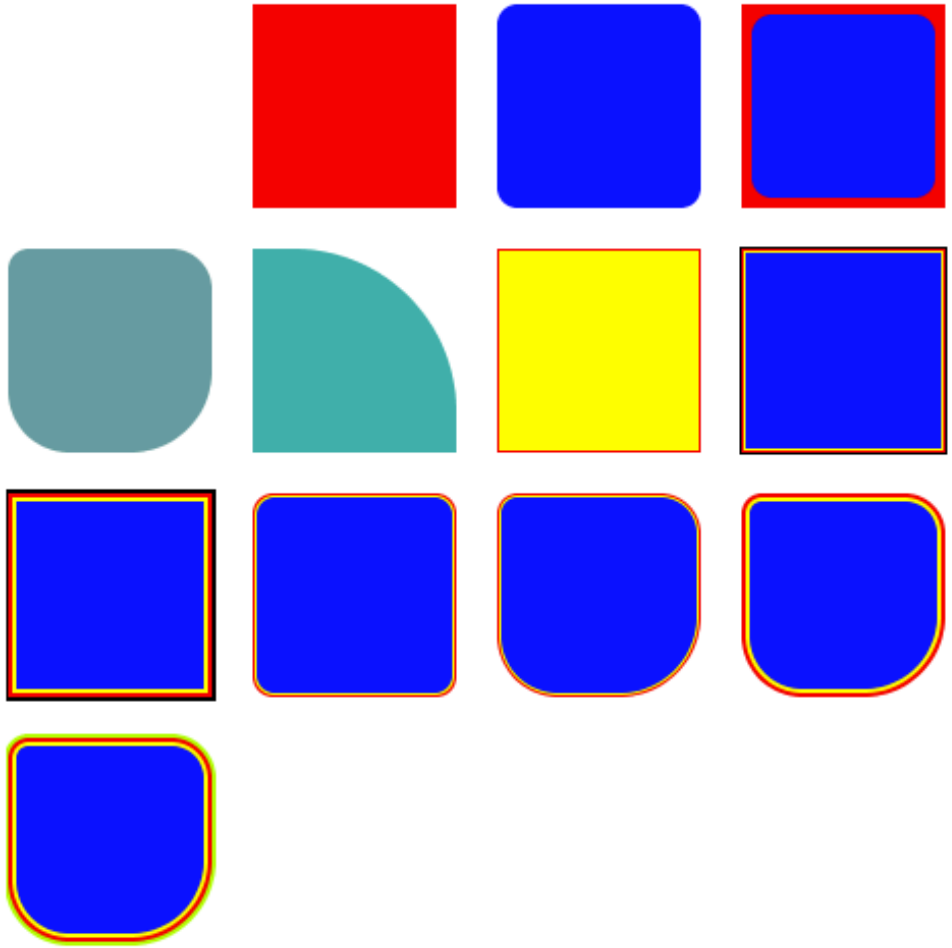
Styling

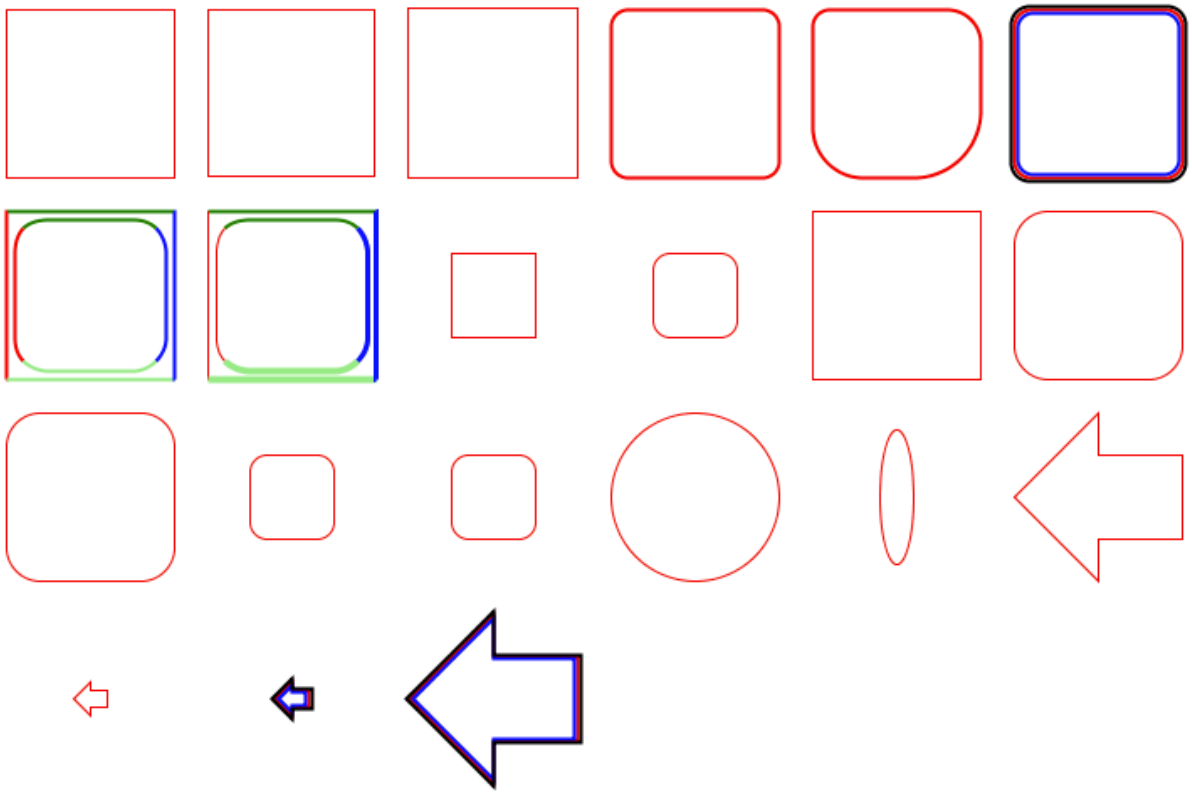
- CSS is our strategy for styling
- Caspian is our default CSS stylesheet
- CSS is fast, and works on mobile, desktop, and tv
- Stick to the spirit of HTML CSS
  - but do not be bound by it

Styling

- Break control skins in styleable parts
- In some ways similar to HTML CSS's Box
- Rectangle with independently rounded corners
  - or any arbitrary path
- Can have multiple
  - background fills
  - background images
  - border strokes
  - border images

Styling





Start with just the content, a Label

Cancel

Padding Top

Add padding

padding: 5px 14px 5px 14px;

Padding Left

Cancel

Padding Right

Padding Bottom

Add shadow

padding: 5px 14px 5px 14px;
background-radius: 7px;
background-fill: lookup(shadow-highlight-color)

Cancel

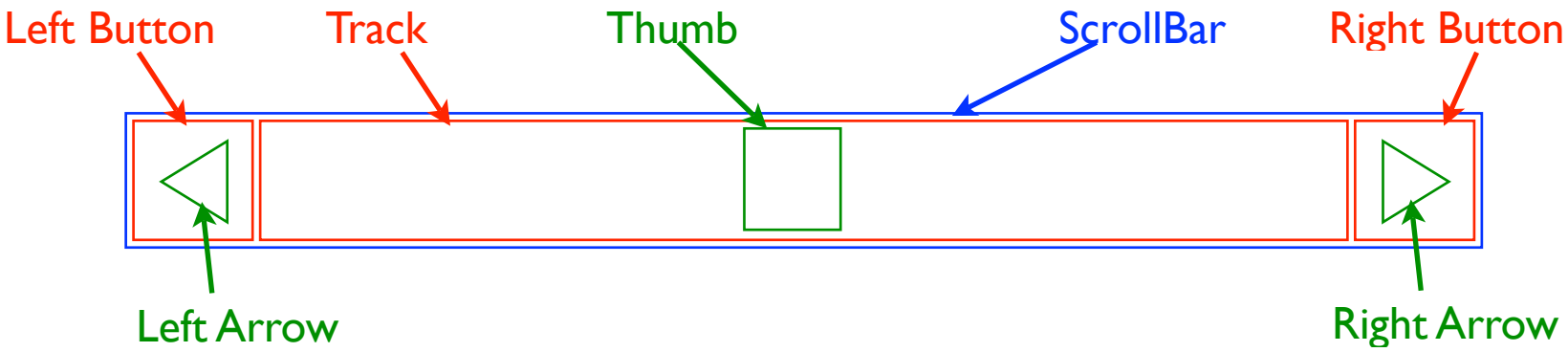
Cancel

1 1 1 1

Add fill

padding: 5px 14px 5px 14px
background-radius: 7px, 7px, 6px, 5px;
background-fill: lookup(shadow-highlight-color) ,
lookup(border-color) ,
lookup(inner-border-color) ,
lookup(body-color)
background-insets: 0 0 -1 0 , 0 0 0 0 ,
1 1 1 1 , 2 2 2 2

Cancel



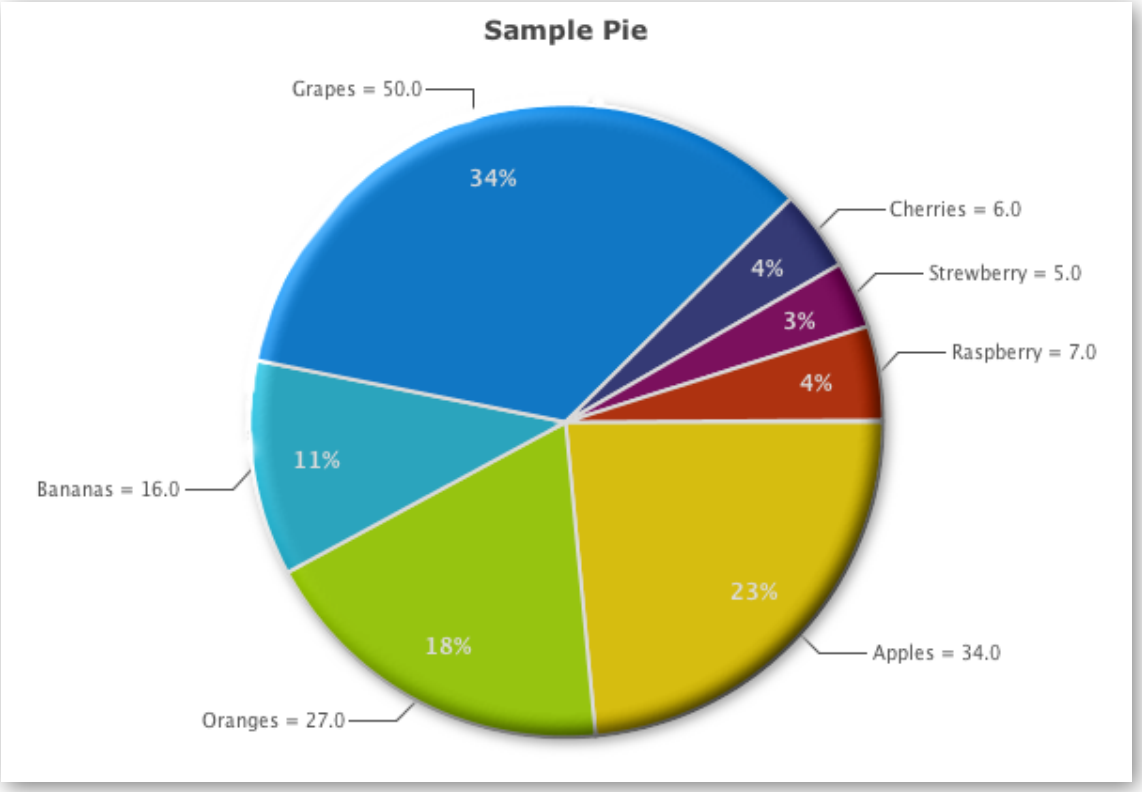
Outline

- JavaFX Script – a new programming language
- JavaFX scene graph
- User interface controls
- Styling
- Charts
- Layout
- Developer tools

Charts

- A basic set of charts for everyday use
  - Simple
  - Customizable
- To provide tools to help you build your own charts

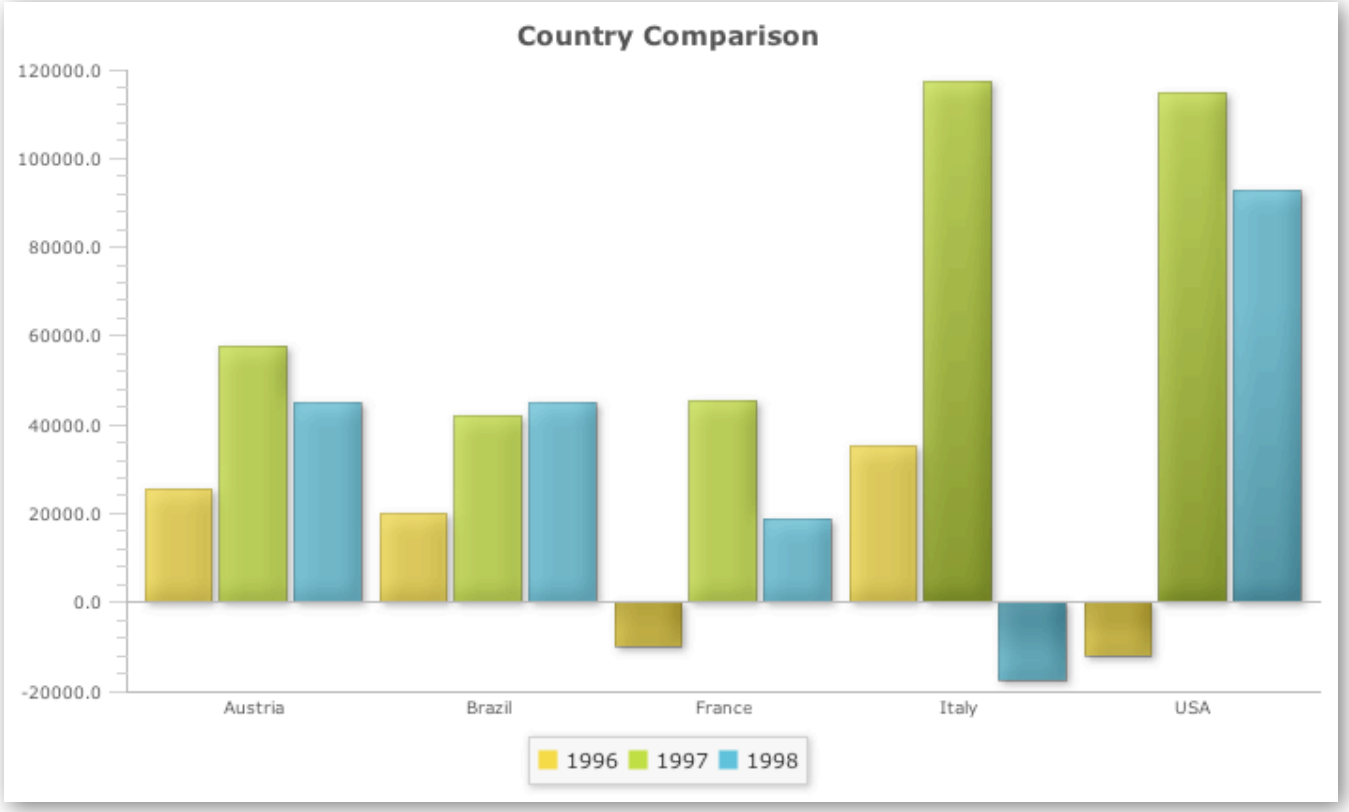
Pie Chart



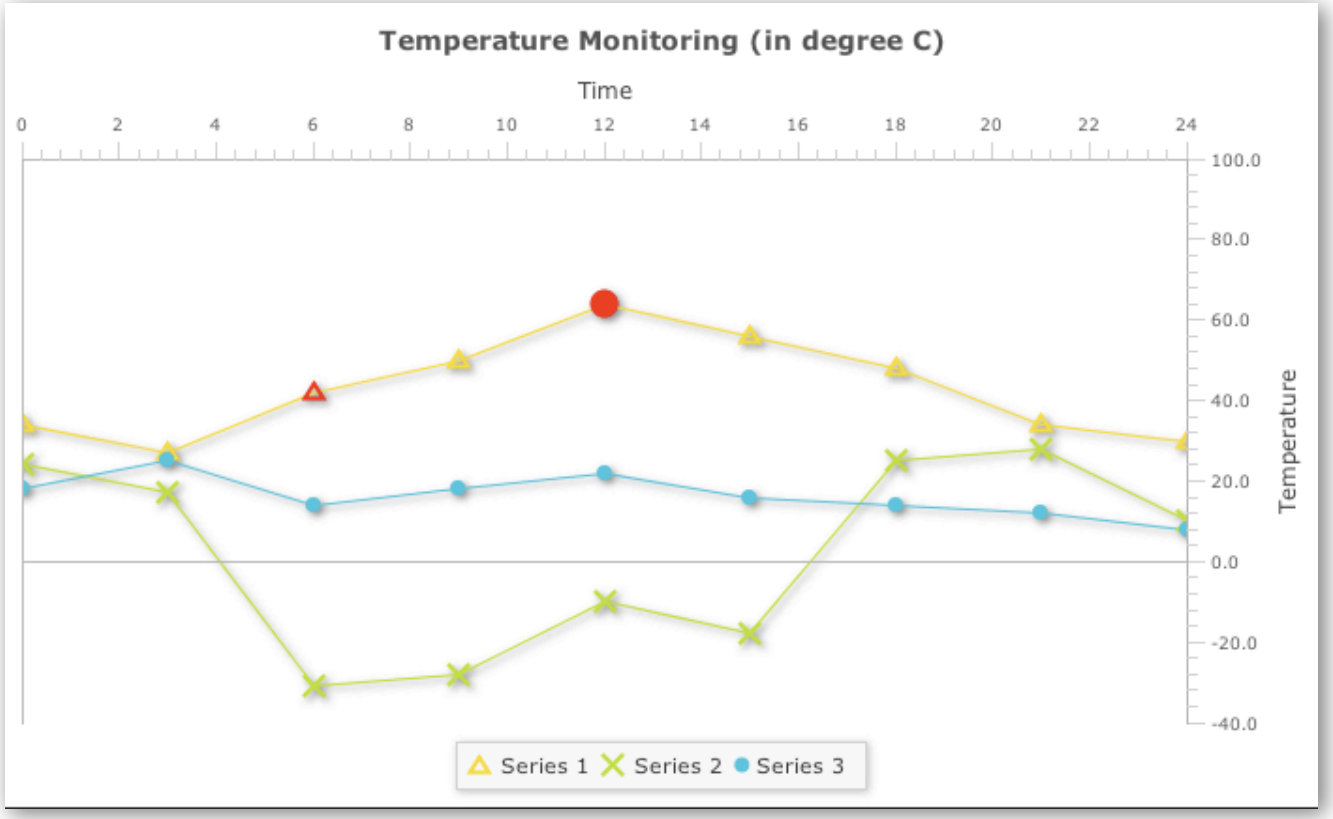
Sample Pie

```
PieChart {
    title: "Sample Pie"
    data: [
        PieChart.Data {
            label: "Apples" value: 34
            action: function(){ Alert.inform("Clicked") }
        },
        PieChart.Data { label: "Oranges" value: 27 },
        PieChart.Data { label: "Bananas" value: 16 },
        PieChart.Data { label: "Grapes" value: 50 },
        PieChart.Data { label: "Cherries" value: 6 },
        PieChart.Data { label: "Strawberry" value: 5 },
        PieChart.Data { label: "Raspberry" value: 7 }
    ]
}
```

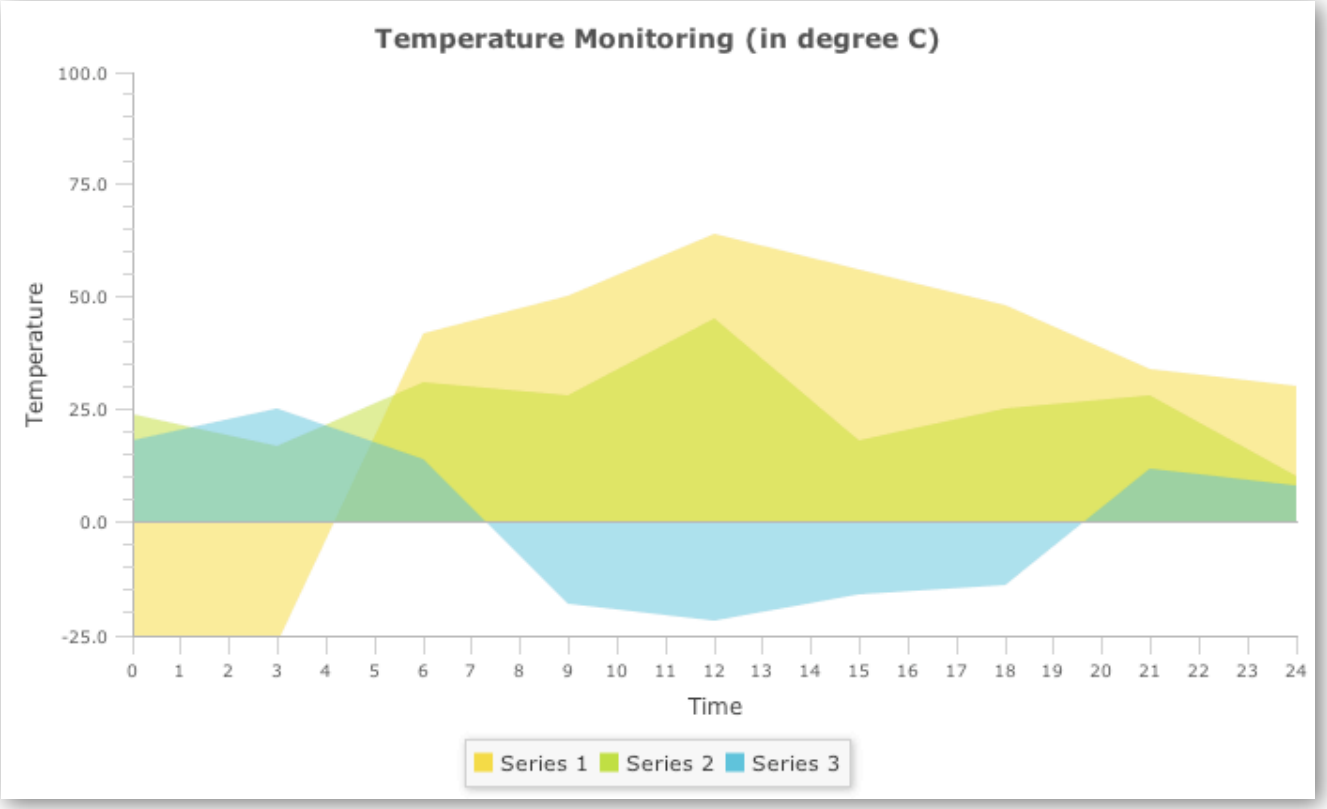
Bar Chart



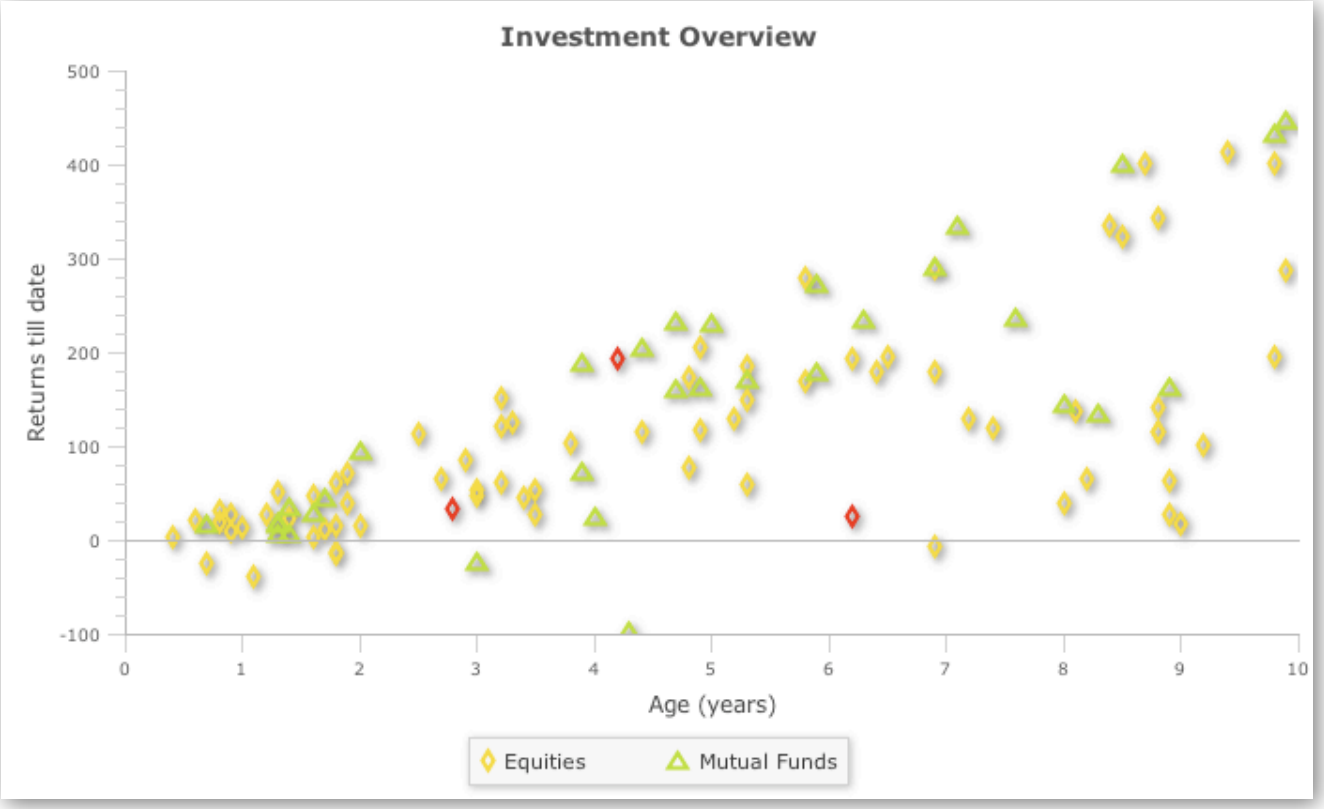
Line Chart



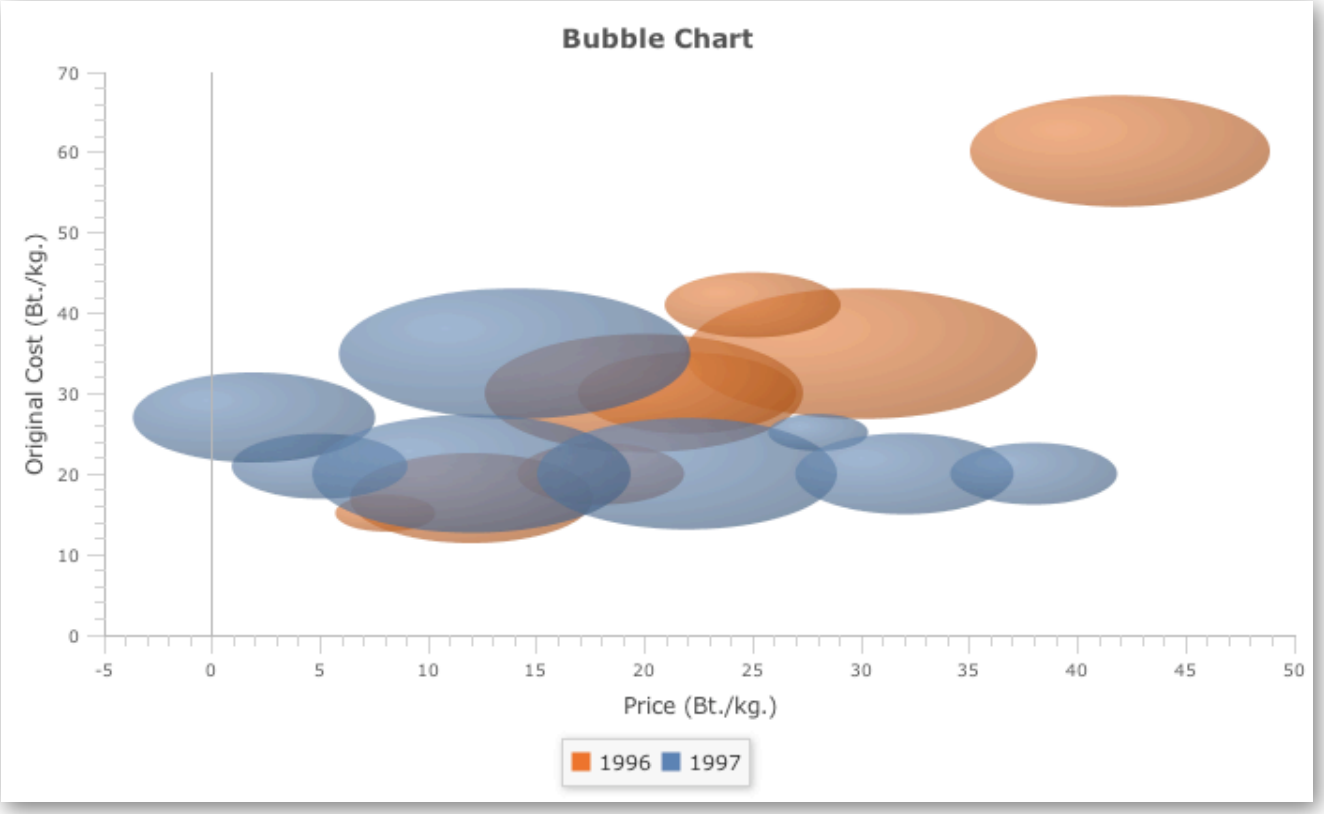
Area Chart



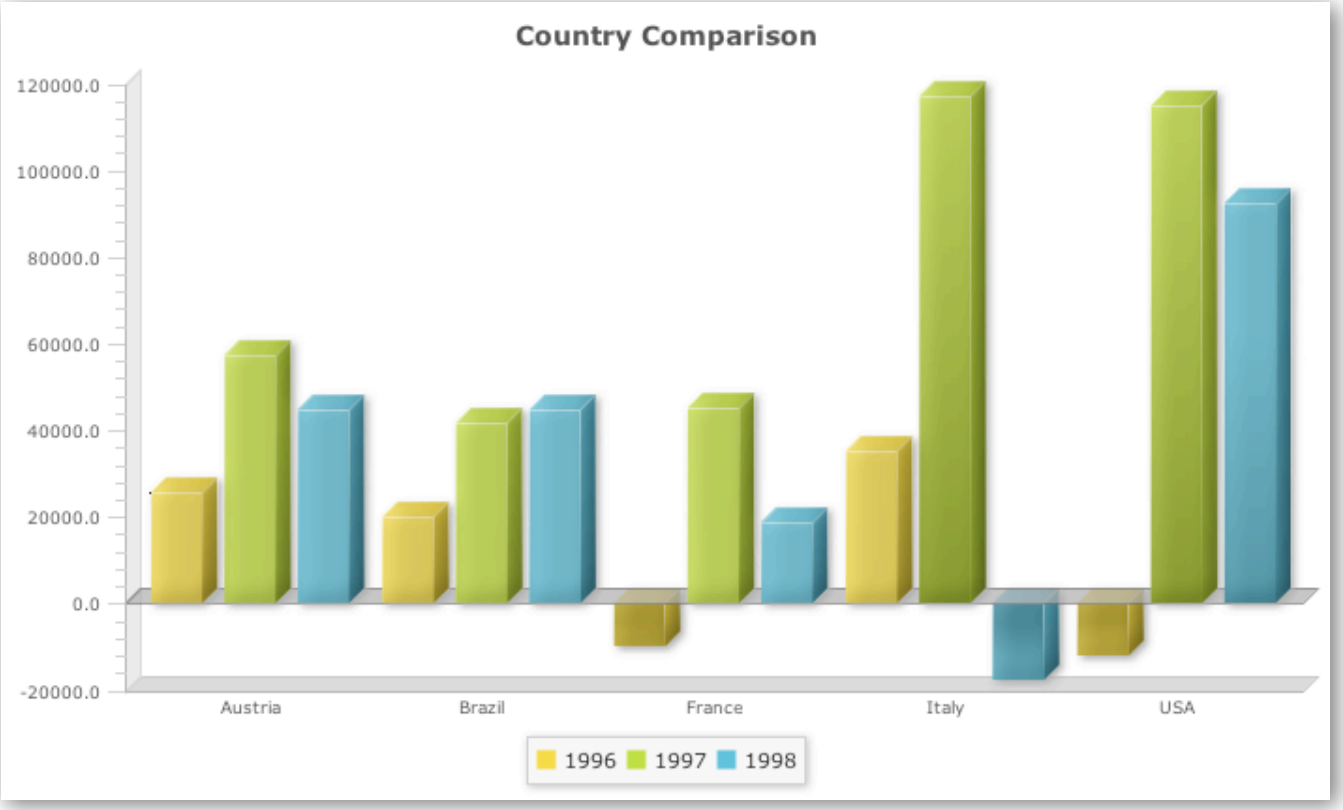
Scatter Chart



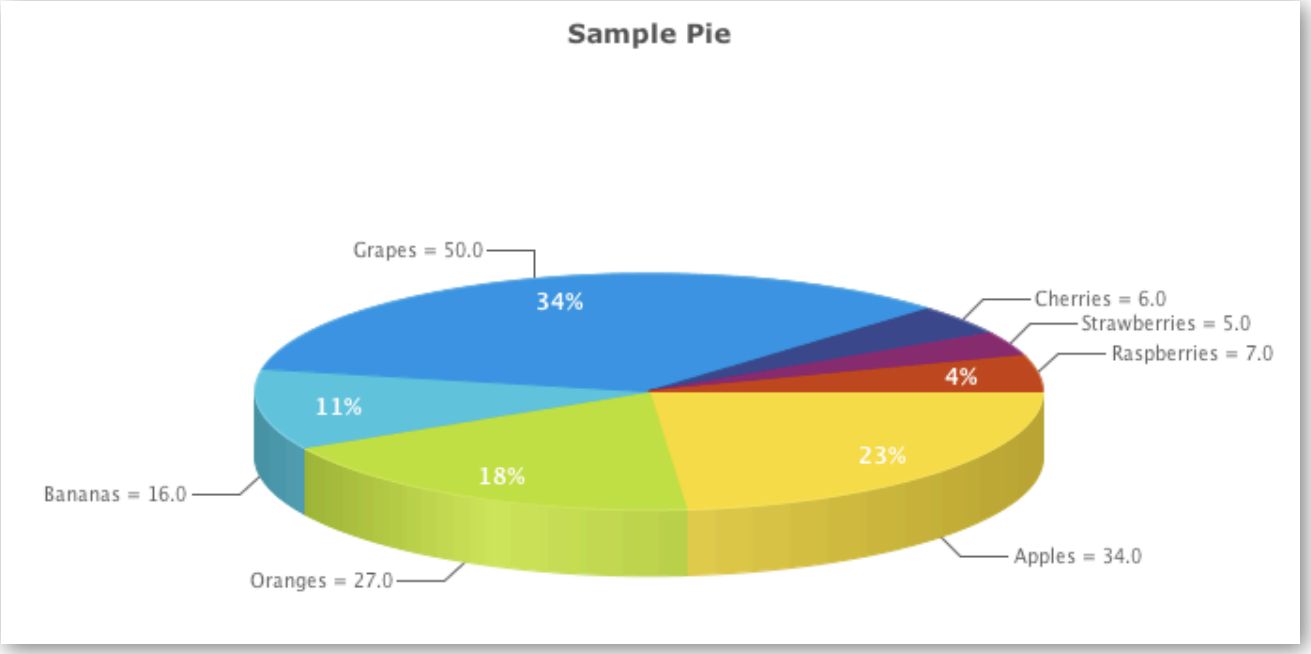
Bubble Chart



3D Bar Chart



3D Pie Chart



Outline

- JavaFX Script – a new programming language
- JavaFX scene graph
- User interface controls
- Styling
- Charts
- Layout
- Developer tools

Layout Containers

- Container-based layout
- Container is-a Node
- Built-in Containers in 1.2
  - Stack: stack all content nodes on top of each other
  - HBox: lay out content horizontally
  - VBox: lay out content vertically
  - Flow: layout out content either horizontally or vertically and line wrap
  - Panel: Customizable layout container

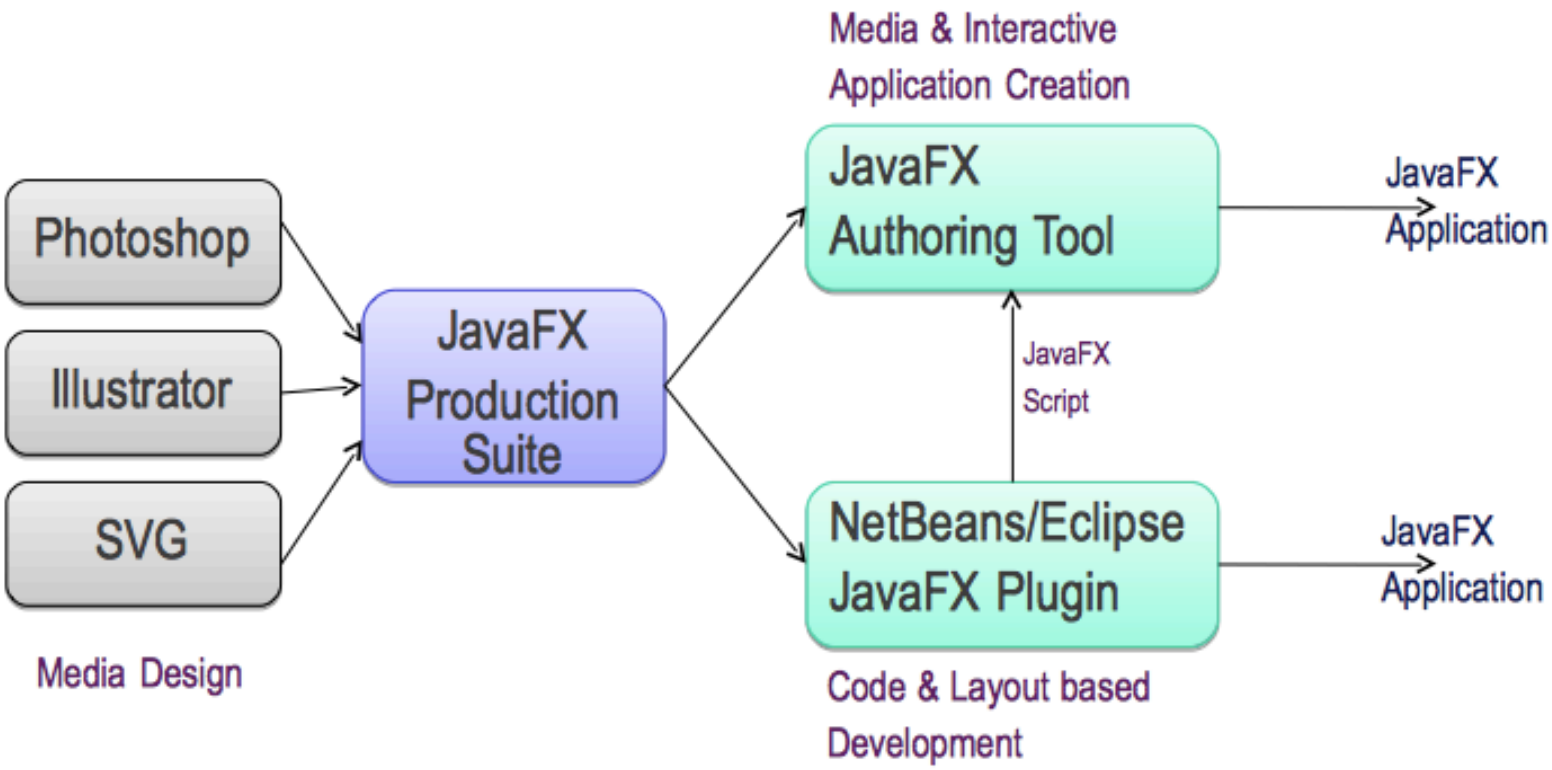
Flow Sample

```
Flow {
  width: 800
  height: 600
  content: for (img in images) {
    ImageView { image: img }
  }
}
```

Developer Tools

- NetBeans with JavaFX plug-in
  - Syntax highlighting
  - Code completion
  - SDK integration
- Eclipse, IntelliJ
- JavaFX Production Suite
  - Imports artwork from content creation tools
  - ... into the scenegraph as a Node
- JavaFX Authoring Tool
  - Creating JavaFX Content
  - Built completely on top of JavaFX and UI Controls

Developer-Designer Workflow



DEMO – JavaFX Production Suite

Call To Action

- fxexperience.com
- Visit javafx.com
  - Download JavaFX SDK + NetBeans
  - See demos
  - Download example code
  - Read tutorials, FAQs, white papers, documentation
  - Browse API Documentation

Thank You!