JAVA FX

Muhammad Imran Sarwar

OSSW - ICOSST 2009
Al-khwarizmi Institute of Computer Science, UET, Lahore
The overall goal of this presentation is to learn
- What is JavaFX?
- And what can it do for you and your customers?
WHAT IS JAVA FX

- JavaFX is the new client stack for graphics enriched Java applications across all devices.

... Mobile Computer Television ...
FOUR THINGS TO TAKE AWAY

- Simple, Elegant and Leverages the power of Java
- Example Applications of JavaFX
- Introduction of JavaFX script language and APIs
- How can you get start developing JavaFX applications

Note: This presentation is an Example JavaFX script application
OUTLINE

- Introduction to JavaFX
- JavaFX script
- Scene Graph
- Animations and Transitions
- Sample Applications
- Get started developing JavaFX
- Q & A
INTRODUCTION TO JAVAFX

- Provides common API across multiple devices
- Scales from small devices to powerful desktops
- Brings rich media and graphical APIs to Java
- Simplifies building graphical consumer and enterprise applications
JavaFX SCRIPT

- Expression language
- Declarative and procedural
- Integrates with Java
- Loosely based on JavaScript
- The source files of the JavaFX script are called "Scripts"
- Everything in JavaFX is an expression
  - All blocks are expressions
  - The last line is the result
PRIMITIVE DATA TYPES

- Boolean
- Integer
- Number
- String
- Duration
- Primitives cannot be null
Both single and double quotes can be used

Variable values can inserted in Strings
- `var i=10`
- “Value of i = {i}”

- `var even = i mod 2 == 0;`
- `var s4 = “The number is {if (even) then ‘even’ else ‘odd’}”`
- `var s5 = “The hex code for 1024 is {%x 1024}”`

Multi Line strings can be declared
DURATION

- var dur1 = 23s; // seconds
- var dur2 = 2.3ms; // milliseconds
- var dur3 = 25.5 * 1.13s; // seconds
- var dur4 = 5m; // minutes
- var dur5 = 10h; // hours
JavaFX has "Sequences", not arrays
- A Sequence is an immutable ordered list of non-null elements
- JavaFX supports sequence comprehensions
- Sequences can be "sliced"
- Sequences can optimize memory usage
- Use [] for sequences
DECLARING MORE SEQUENCES

// A sequence of one can omit the braces
var names: String[] = "Imran";
// empty sequences are not null
var names: String[];
println(sizeof names); // prints 0
// Elements are accessed by index number
var names = ["Imran", "Wasif", "Umar"];
Var wasif = names[2];
// And you can populate a sequence using a for loop
var hellos = for(i in [1..3]){"Hello #{i}"};
MODIFYING SEQUENCES

//inserting items into a sequence
Var names: String[] = {“Imran”};
insert “Wasif” into names;
//inserting before a certain index
insert “Umar” before names[1];
//inserting after a certain index
Insert “Hammad” after names[1];
//deleting from a sequence
delete “Umar” from names;
OPERATORS

+  -  /  *

++  --

* =  +=  /=  -=

and or not

=  ==  !=

mod
FLOW CONTROL

- if(booleanExpression) then a else b
- if(booleanExpression) a else b
- if(booleanExpression) { a } else { b }
- while(booleanExpression){ ... }
- for(i in sequence){ ... }
- Can get index of item “i” by “indexOf i”
- break
- continue
VARIABLES

- Declaring Variables is simple
- name:type style syntax
- var for variables and def for definitions
- Type inference
- Examples
  - var dur1 = 23s; // seconds
  - def PI = 3.141592; // constant values
  - var number: Number = 3
Main Constructs of JavaFX

- Two main constructs of JavaFX
- bind
- object literals
- These constructs form the heart of JavaFX
bind is the way to **tie** the value of one variable to the value of an expression

- Binding must be defined when the variable is initialized
- Bindings are statically compiled
- Bound variables cannot be set manually
public class Distance{
    public var x1: Number;
    public var x2: Number;
    // Whenever x1 or x2 changes, distance will be updated
    // Binding can be used for invariants
    public-read var distance: Number = bind x2-x1;
}
OBJECT LITERALS

- Concise declarative syntax for object creation
- Similar to JavaScript
- Combine with bind for maximum effect
- variable: initial-value
- initial-value is an expression
OBJECT LITERAL EXAMPLE

// creates a Rectangle
// x: 10 is not assignment, it's an initialization!

var rect: Rectangle{
    x: 10
    y: 10
    width: 50
    height: 50
}
NESTING OBJECT LITERALS

// creates a Rectangle with the color for its fill
var rect: Rectangle{
    x: 10
    y: 10
    width: 50
    height: 50
    fill: Color {
        red: 1
        green: 0
        blue: 0
    }
}
// a variation that allows me to reference color latter

```javascript
var color: Color;
var rect = Rectangle{
    x: 10
    y: 10
    width: 50
    height: 50
    fill: color = Color {
        red: 1
        green: 0
        blue: 0
    }
}
```

```javascript
var rect = Rectangle{
    var color = Color {
        red: 1
        green: 0
        blue: 0
    }
    x: 10
    y: 10
    width: 50
    height: 50
    fill: color
}
```
SCENE GRAPH

- Describes the graphics and controls in a scene
- Each node in the graph has a single parent
- Special “group” node has zero or more children
- “leaf” node has no child
- Graph is set on a Scene
- Scene is set on a Stage
CONTAINER VIEW

Stage

Scene

Graph
STAGE

- Top level container for the scene
- Contains only one Scene
- Can set Stage width and height
- Potentially represented by
  - JFrame on desktop
  - JApplet on web page
  - SVG player on a mobile
**NODES**

- Node is the base class of the entire scene graph
- Every Node has bounds
- Every Node has transforms
  - translate
  - scale
  - rotate
- Every Node has a parent
CUSTOM NODE

- Primary method of Scene Graph encapsulation
- All other nodes are not extendable
- Simply override the create() method
# SHAPES

Shapes are the building blocks in JavaFX

<table>
<thead>
<tr>
<th>Basic Shapes</th>
<th>Common Variables</th>
</tr>
</thead>
<tbody>
<tr>
<td>Arc</td>
<td>stroke</td>
</tr>
<tr>
<td>Circle</td>
<td>strokewidth</td>
</tr>
<tr>
<td>Ellipse</td>
<td>fill</td>
</tr>
<tr>
<td>Line</td>
<td>smooth</td>
</tr>
<tr>
<td>Path</td>
<td></td>
</tr>
<tr>
<td>Polygon</td>
<td></td>
</tr>
<tr>
<td>Rectangle</td>
<td></td>
</tr>
</tbody>
</table>
COLOR

- Color
- 150+ built in colors
- All the html and CSS built in values
- Example usage
  - `Color.web("#aabbcc")`
  - `Color.web("blue")`
  - `Color.rgb(128,222,21)`
  - `Color.RED`
LINEAR GRADIENTS

- \( \text{startX, startY, endX, endY} \)
  - Define the direction of the gradient
  - On the unit square
- Stops define each step in the gradient, Each stop
  - Has an offset between 0 and 1
  - Has a color
Images

- ImageView node shows the images
- Image represents an in-memory Image
- Image can load images in FG thread or BG thread
- Both Image and ImageView can scale
  - Preserve ratio
  - Fit with in a specific width and height
  - When fit on image level, keeps smaller image in memory
- x, y and TextOrigin
- By default, text is positioned such that (x,y) is left baseline
- Fonts can be specified on Text
- Favor fill over stroke
- Supports multiline text
- Use alignment to align multiline text
- To center text, compute the center via layout bounds
**TEXTBOX**

- Used for text input
- Use CSS to style the TextBox
- “text” is changed to reflect the actual text in the TextBox
- “value” is changed when the text is “committed” through via ENTER, TAB, etc.
- “action” function is invoked when ENTER is pressed
- “columns” specifies the preferred width based on the font size and number of characters to display
Demo 3
EFFECTS

- Effects are placed in nodes
- Many standard built in
  - DropShadow
  - ColorAdjust
  - GaussianBlur
  - Glow
  - Reflection
  - more...
DEMO 4
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
- MediaView is the Node that displays the multimedia content
- No built in movie playing control
Animation

- Animation is a key feature of any rich graphics application platform
- Two supported animation types in JavaFX
  - Keyframe Animations
  - Transitions
KEYFRAME ANIMATIONS

- A general purpose animation mechanism for JavaFX
- Uses the concept of timeline and keyframes specifying values at given times
- Built in the language syntax - can animate any variable
- “tween” key word and custom interpolators
- Long syntax vs short syntax
- “autorepeat” and “autoreverse” behavior
- Actions
- Nested timelines
TRANSITIONS

“Precanned”, single purpose animations
- Transition/Scale/Rotation/Fade
- By values
- Animation along a path
- Containers
- Parallel
- Sequential
- Pause transition
SAMPLE APPLICATIONS
SUMMARY

- JavaFX provides the APIs necessary for graphical applications
- Boost your imagination
- Attract your consumers
- Simple and leveraging the power of Java
CONCLUSION

- Check your creativity and skills
- Try JavaFX

- References
  - http://javafx.com
  - http://parleys.com
THANK YOU FOR BEING ATTENTIVE