Extending JDORFX: A JavaFX Graphics Framework for ooRexx

Alireza Ismaili
Vienna University of Economics and Business (WU)
21 October 2026



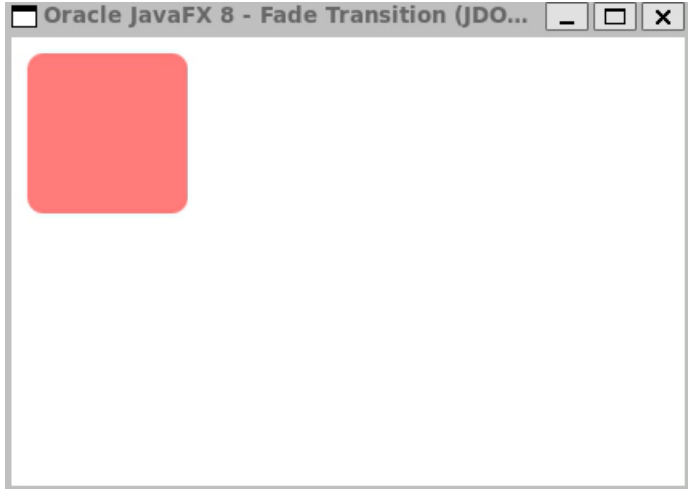
Making graphics accessible through simple commands

Direct JavaFX
Requires using Java classes and managing objects and application infrastructure.

JDORFX
Exposes selected JavaFX functionality through textual commands in ooRexx.

EXAMPLE — SHAPE ANIMATION

```
animationFade fadeTransition myRect 3000 1.0 0.1 "-1" .true
animationPlay fadeTransition
```



Goal: Create graphical applications without programming directly with JavaFX

How JDORFX works

ooRexx program	BSF4ooRexx	JavaFXDrawingHandler	JavaFX
Sends graphics commands	Connects ooRexx and Java	Parses and executes commands	Canvas, Scene Graph, 3D, Animations

JDOR:
Java2D-based, primarily raster rendering

JDORFX:
JavaFX-based, Canvas plus retained Scene Graph nodes

Starting Point and Objectives

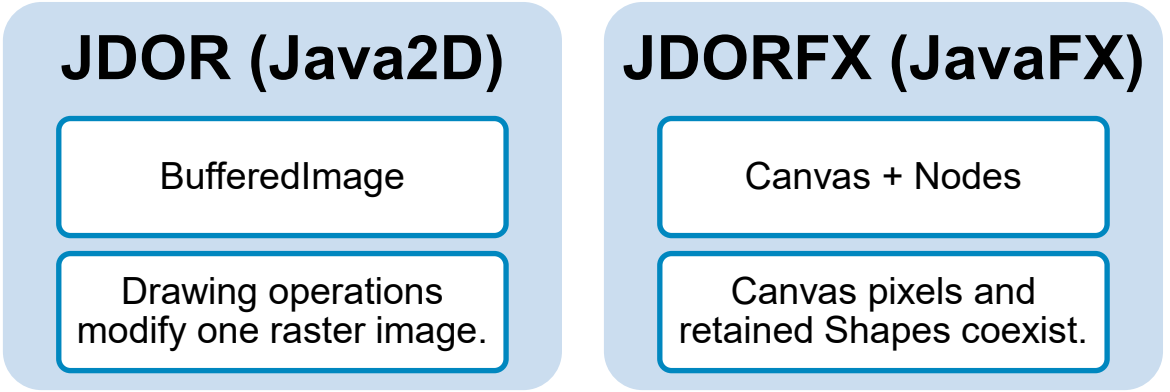
Inherited JDORFX implementation

- 2D Graphics
- Basic 3D
- Cameras and Lights
- Materials and Textures

Areas requiring further development

31 missing JDOR commands	Compatibility and architectural limitations	Limited 3D controls	No animation command interface

The Challenge: Two Different Rendering Models



Example problem
How should clearRect, copyArea, or saveImage behave when the current drawing contains both Canvas pixels and retained Shape nodes?

Challenge: Preserve JDOR behaviour without losing JavaFX-specific capabilities.

Overview of My Contributions

31
JDOR commands

Previously missing functionality implemented



Architecture

Rendering compatibility, threading and window handling

6
New 3D commands

Camera, lighting and face culling

18
Animation commands

Transitions, timelines and playback

Preserving the existing command-oriented interface

Completing JDOR Command Support

31 previously unimplemented commands, across multiple areas:

Images and I/O
Loading, saving, copying, printing and clipboard access

Graphics state
pushGC, popGC, clipping and image stacks

Rendering
Paint, gradients, compositing and XOR

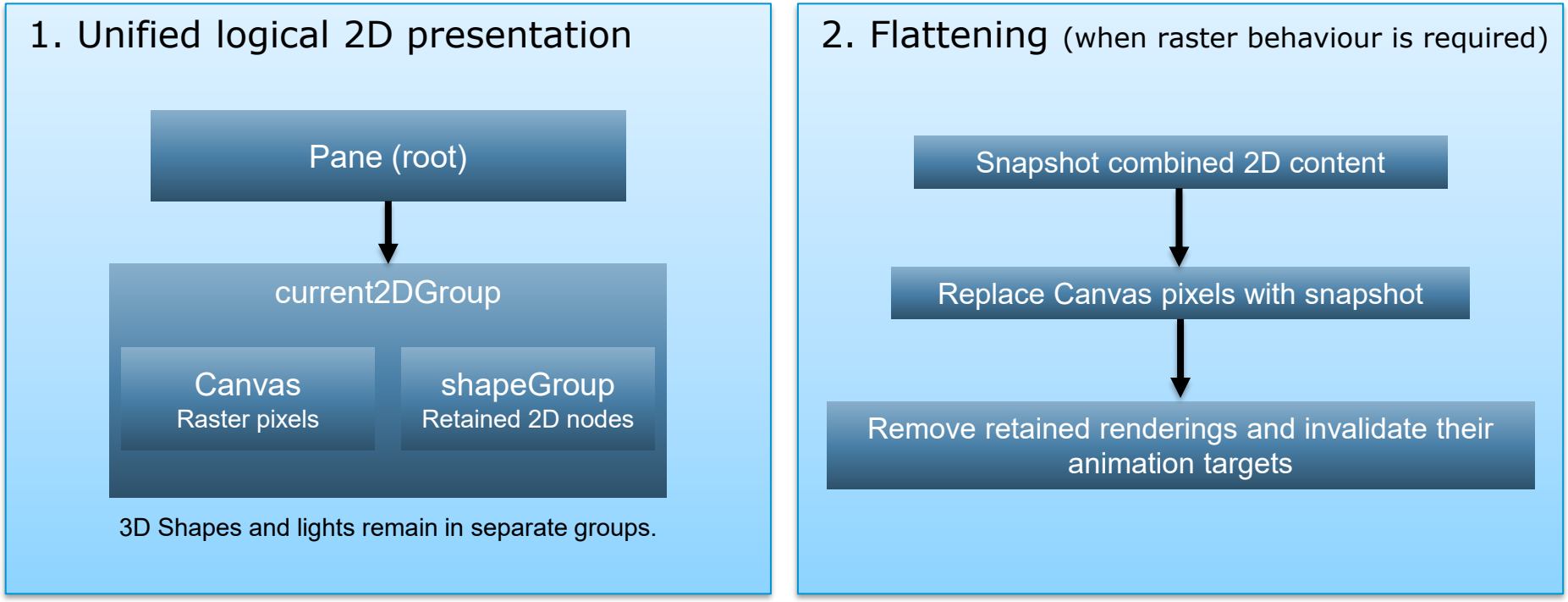
EXAMPLE – IMAGE OPERATIONS

loadImage *LOGO "assets/oorexx_256.png"*
moveTo *15 20*
drawImage *LOGO 220 220*
saveImage *"output.png"*



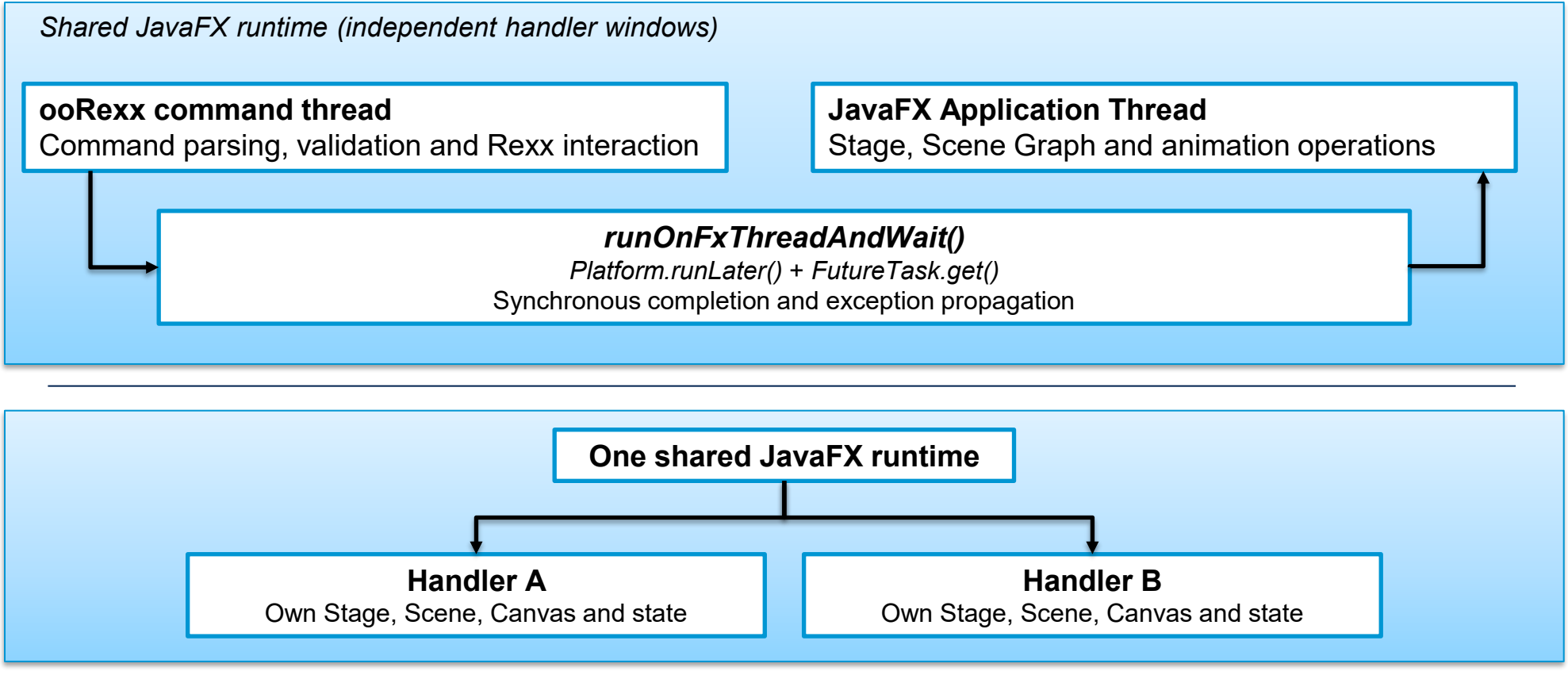
Key challenge: Java2D and JavaFX do not provide identical graphics APIs.

Reconciling Canvas and Scene Graph Rendering



Trade-off: Better JDOR raster compatibility, but flattened nodes are no longer independently animatable.

JavaFX Threading and Window Management



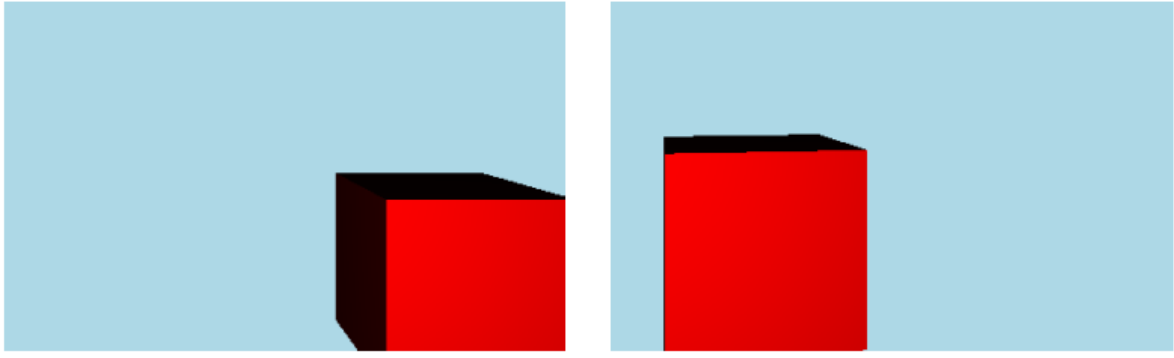
Extending 3D Functionality

Six additional commands for manipulating existing 3D scenes

Cameras	`translateCamera, rotateCamera, cameraNearClip, cameraFarClip`
Lights	`translatePointLight`
3D Shapes	`cullFace`

EXAMPLE - CAMERA TRANSFORMATION

-- translateCamera cameraName deltaX deltaY deltaZ
translateCamera *VIEW 200 50 0*
-- rotateCamera cameraName angle pivotX pivotY pivotZ axisX axisY axisZ
rotateCamera *VIEW 10 300 200 0 0 1 0*



(a) Initial camera view (b) After camera translation and rotation

From creating static 3D scenes to manipulating their existing objects.

Introducing JavaFX Animation Support

18 new animation commands — two complementary approaches

Predefined Transitions
Fade, Rotate, Scale, Translate, Path, Fill and Stroke

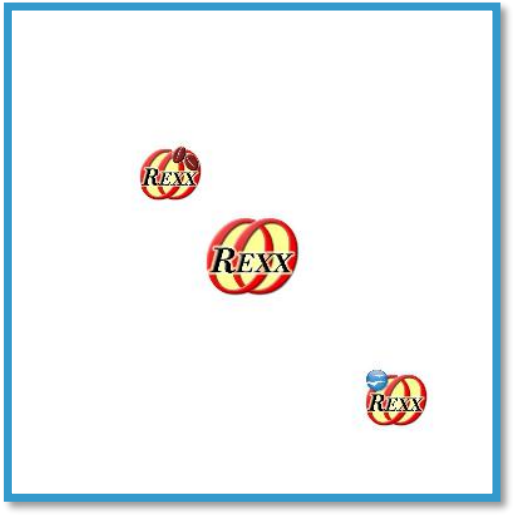
Timeline Animations
KeyValues and KeyFrames for animating selected properties

Composition and Playback

- Sequential
- Parallel
- Play / Pause / Stop

EXAMPLE - ANIMATION

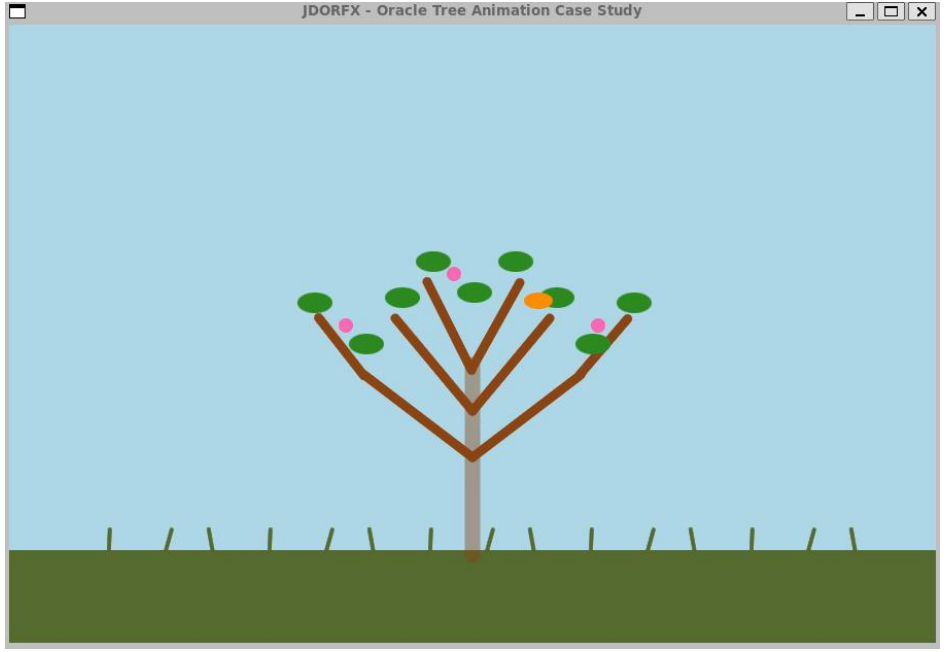
animationRotate *TURN PICTURE 3033 450 1 .false*
setInterpolator *TURN LINEAR*
animationPlay *TURN*



Supported targets: Retained 2D Shapes, 3D Shapes, Cameras and Lights

Demonstrating and Evaluating Animation

Oracle Tree Animation Case Study



Redirected handler commands during execution

Example	JDOR	JDORFX Transition	Reduction
Rotation animation	298	31	89.6%
Compositing animation	10,915	169	98.5%

Measures command traffic, not execution speed or rendering performance.

Results and Compatibility Status

92
No known compatibility limitations
Implemented and evaluated against JDOR

3
Platform-limited
newImage, render, stroke

5
Partially compatible
image, clearRect, copyArea, drawShape, fillShape

0
Missing commands
Previously missing JDOR commands implemented

140
Total canonical JDORFX commands

Complete command coverage does not imply identical behaviour.

Limitations and Future Work

Current Limitations

- Raster vs. retained rendering
  - Flattening removes independent 2D rendering nodes.
- Limited animation properties
  - Not all JavaFX properties, bindings or custom animations are exposed.
- Limited interactivity
  - No general JavaFX event-handler interface

Possible Extensions

- Systematic compatibility tests
  - Compare JDOR and JDORFX outputs and behavior.
- Improved rendering architecture
  - Preserve ordering and retained nodes more effectively.
- Broader JavaFX functionality
  - Event handling, MeshView, effects and additional properties

CONCLUSION

Extending JDORFX

Completed JDOR command coverage

- 31 previously missing commands implemented

Improved the internal architecture

- Rendering compatibility, threading and window management

Extended JavaFX capabilities

- 6 new 3D commands and 18 animation commands

Bringing more of JavaFX's graphics and animation capabilities to ooRexx, while preserving the JDOR programming model.

Thank you for your attention!