Course: Object Oriented Programming

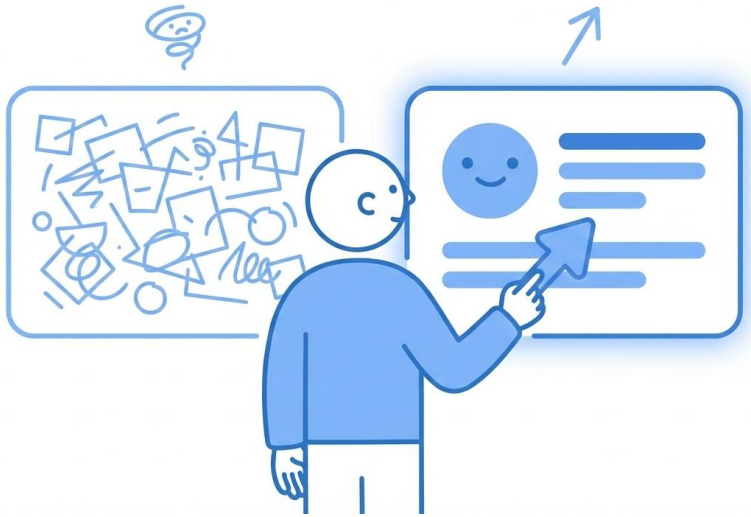
GUI Programming with JavaFX

Architecture, Components & Layouts

Evis Plaku

Good interfaces help people use software easily

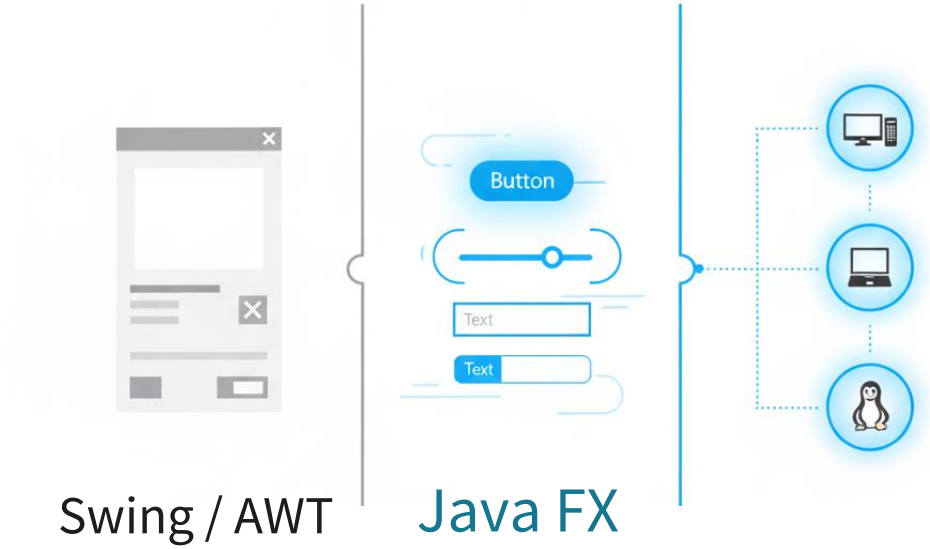
- People choose software that's easy and pleasant to use every day
- Business still rely on desktop software fo important work
- Building interfaces is a valuable skill that employers want and reward



Real Users Need Good Design

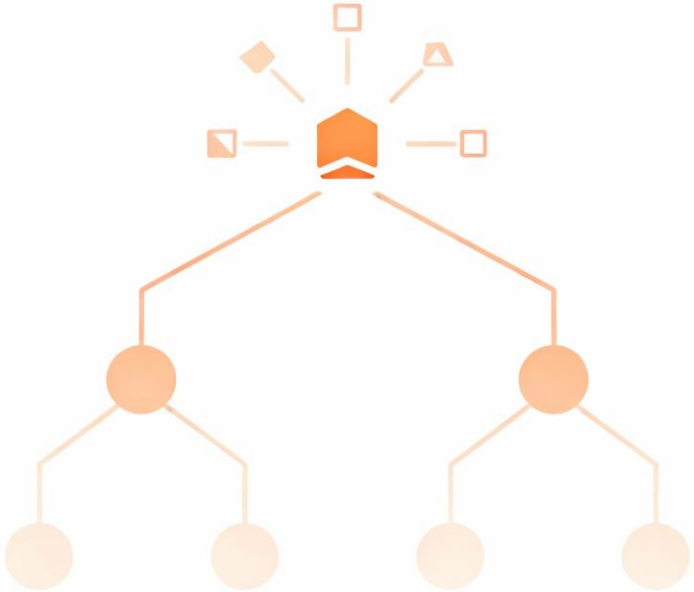
JavaFX is the modern choice for building Java desktop applications today

- Swing and AWT are old. JavaFX is built for today's graphics and speed
- Fast graphics, smooth animations, and lots of ready-to-use interface parts
- Same code runs on Windows, Mac, and Linux without changes needed



- Foundations: we'll start with basic concepts, then add components, then organize in layouts

- Build: create a working login form throughout the lecture to see everything in action
- You'll have skills to style apps, use advanced parts, and manage state

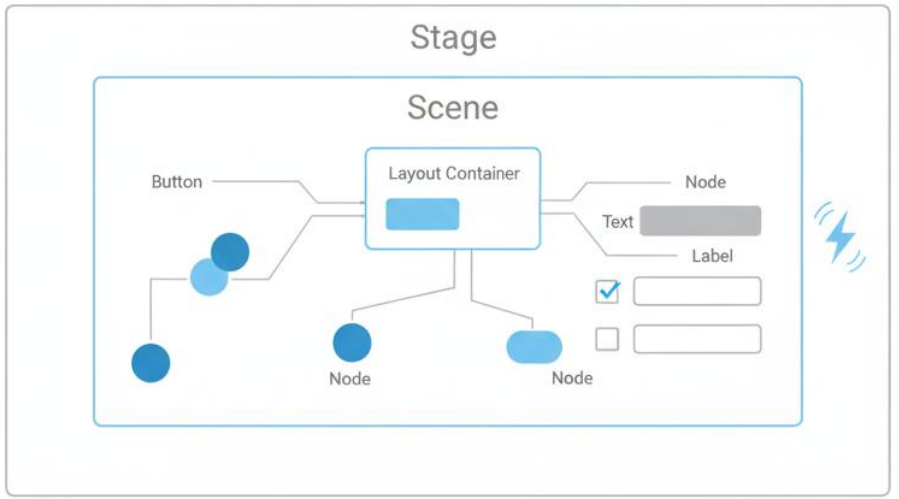


Java FX

Essentials & Architecture

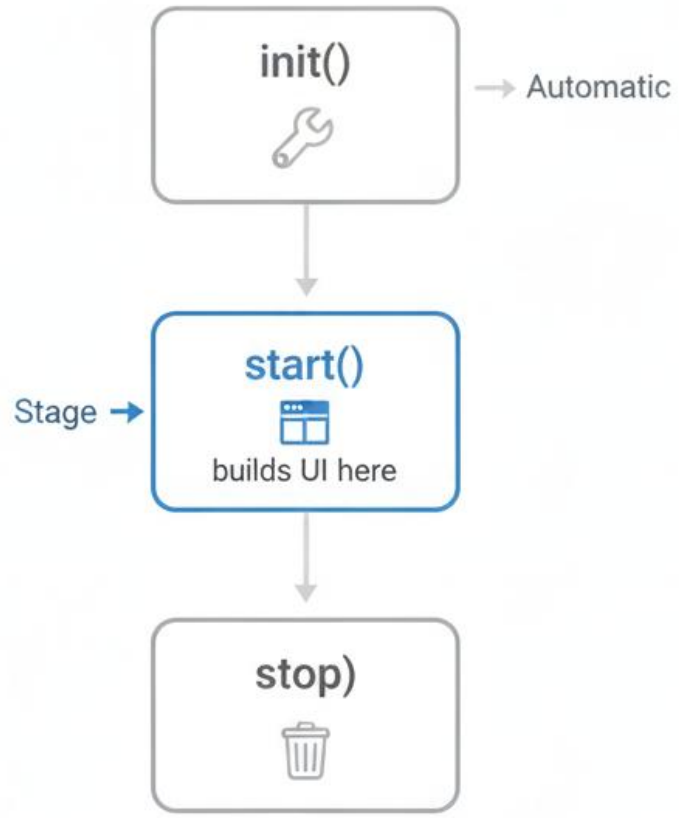
Think of JavaFX like a theater: Stage is the building, Scene is the stage, Nodes are actors.

- Stage is the window. Scene holds everything inside. Nodes are the buttons and text boxes
- Objects nest inside each other like folders in folders, creating a hierarchy
- Pieces can be reused, changed easily, and drawn to screen very quickly



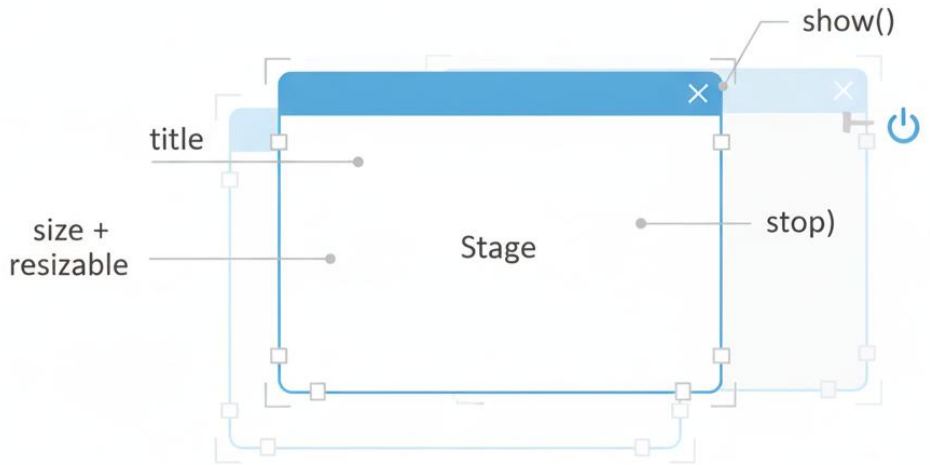
Every JavaFX app starts by extending Application and implementing three key methods

- init() prepares things, start() builds your interface, stop() cleans up when done
- The start method receives the Stage and is where you create your entire interface
- You call launch() which handles setup, then runs your start() method automatically



Stage is your application window. It's where everything the user sees appears

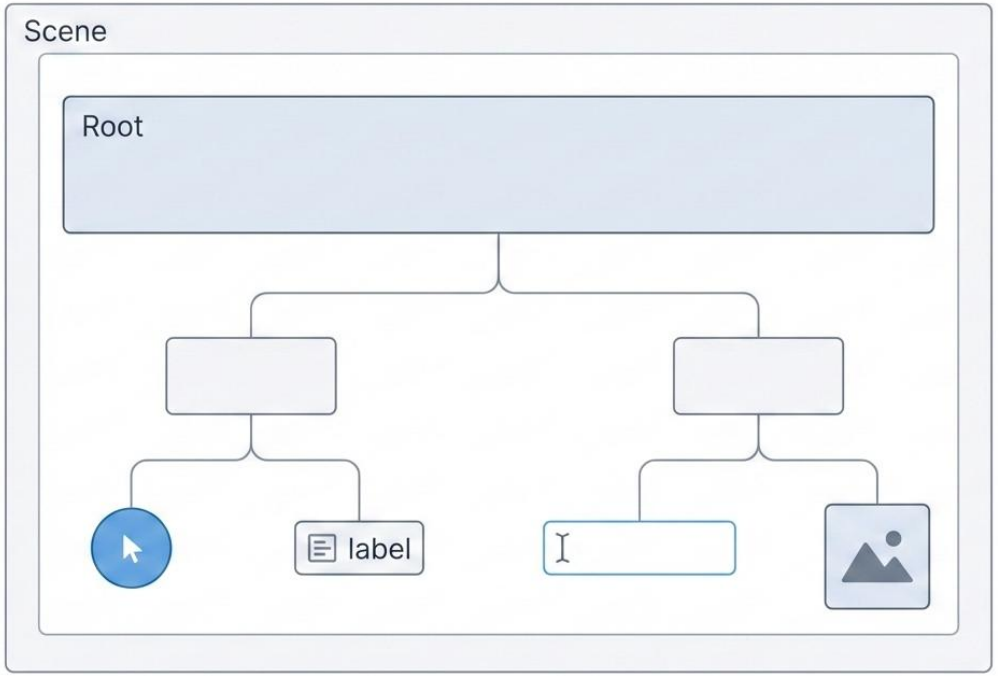
- Stage is the actual box on your computer screen that holds your application



- Set the **title**, **size**, and whether people can **resize** it however they want
- Call show() to display the window, and stop() to close it when done

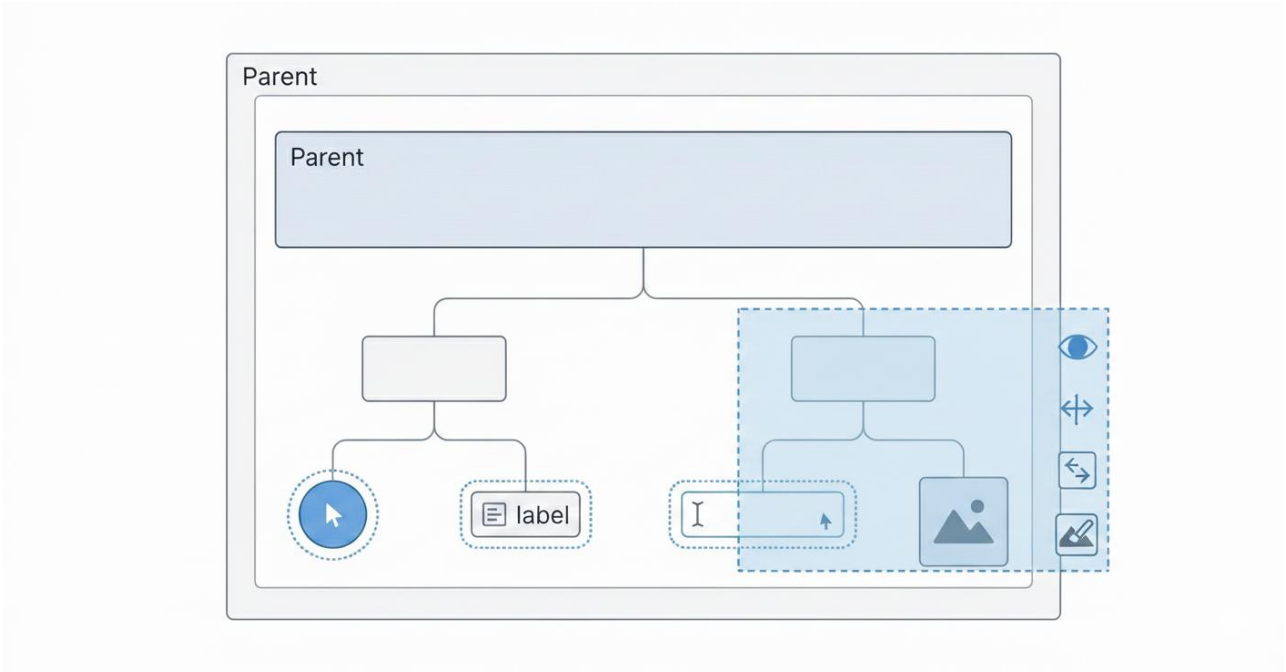
Scene holds all the buttons, text boxes, and other parts inside your Stage window

- Buttons, labels, text boxes, images: everything the user sees lives in the Scene
- One main container holds all other pieces, creating the structure like a family tree
- Stage is the window frame, Scene is what goes inside it



Nodes are the building blocks. They connect together like a tree to form your interface

- Every button, label, and text box is a Node. They're basic pieces of work
- The tree structure lets you move, hide, and draw groups of pieces together easily

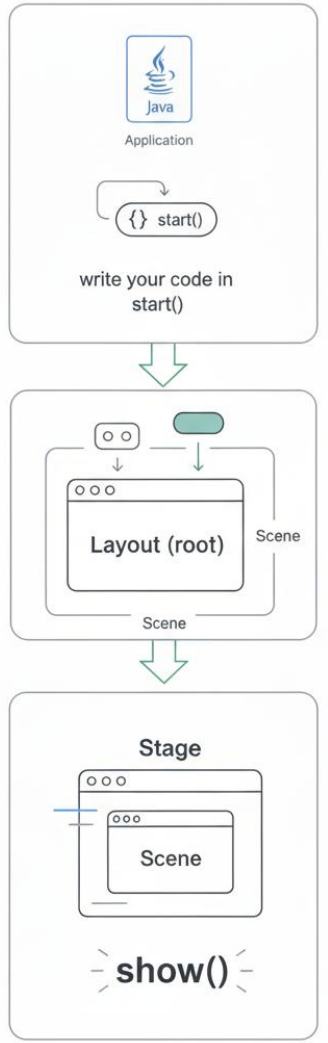


Write a few lines of code to create a window with buttons and labels inside

1. Create a class that extends Application, then write the start() method with your code

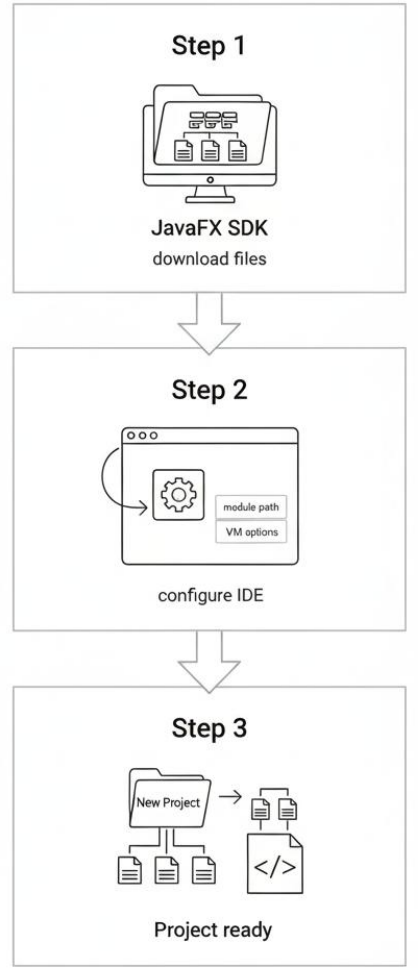
2. Create a layout container, add your buttons and labels to it, make a Scene

3. Set the Scene on the Stage, then call show() to display it to the user



Set up Eclipse to work with JavaFX so you can write and run your applications

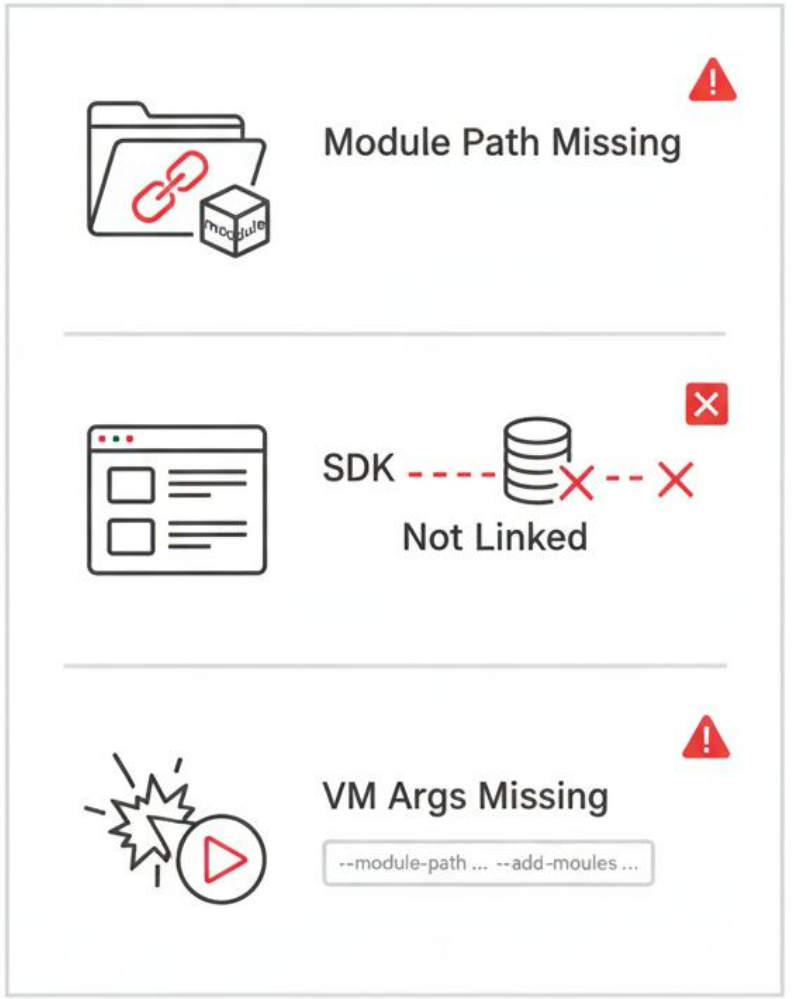
- Get the JavaFX library files and save them in a folder on your computer
- Configure Eclipse to find the JavaFX files using VM options and module paths
- Start a new Java project in Eclipse, add JavaFX libraries, and you're ready to code

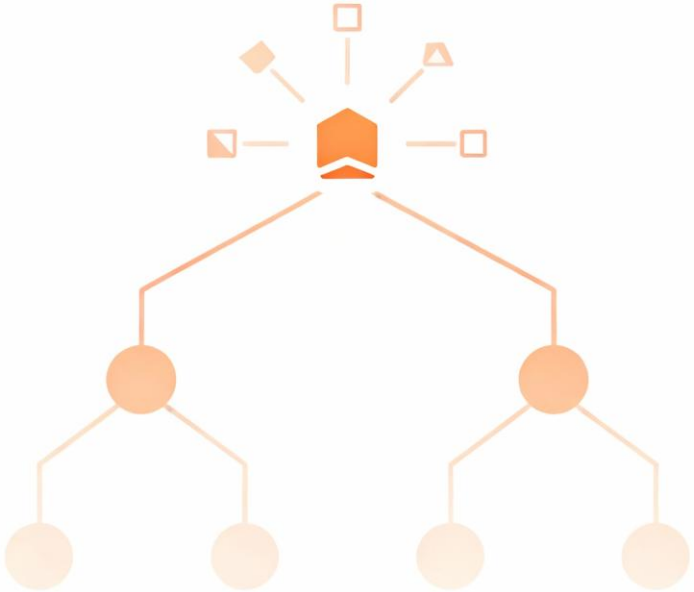


- Forgot to Configure Module Path: Error saying it can't find JavaFX classes. Check Eclipse settings

- JavaFX SDK Not Linked: Code won't compile because Eclipse doesn't know where the library files are located

- VM Arguments Missing: Program starts but crashes immediately. Make sure your run configuration includes the right flags





Core UI Components

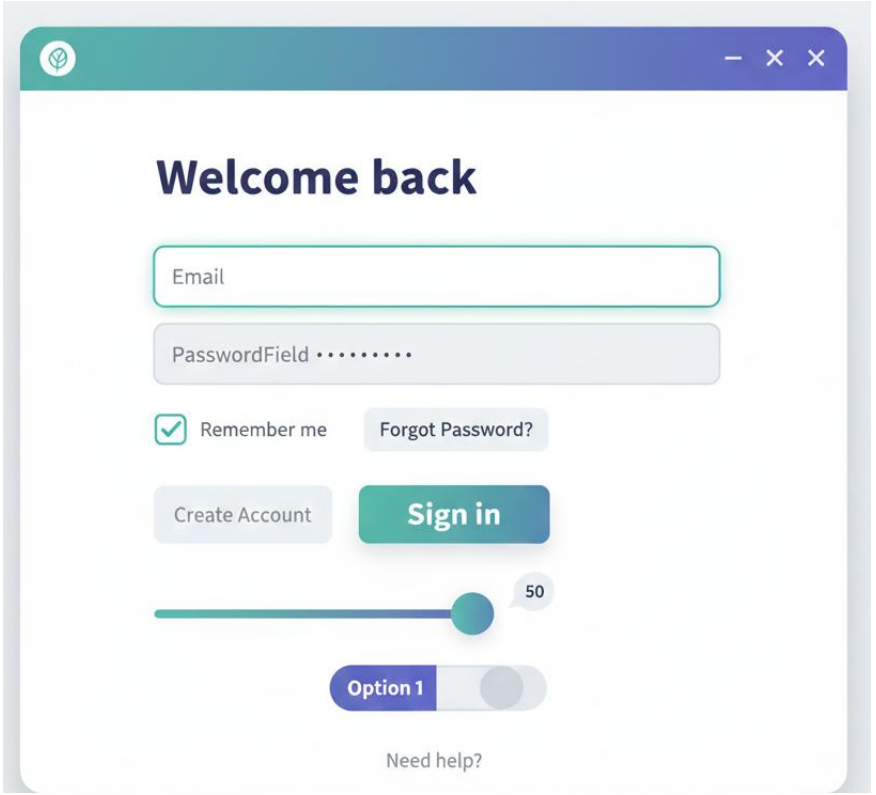
Event Handling

Controls are ready-made pieces like buttons and text boxes. JavaFX gives you many to choose from

- Text controls show or accept words.
  Selection controls let users pick options.
  Input controls take numbers

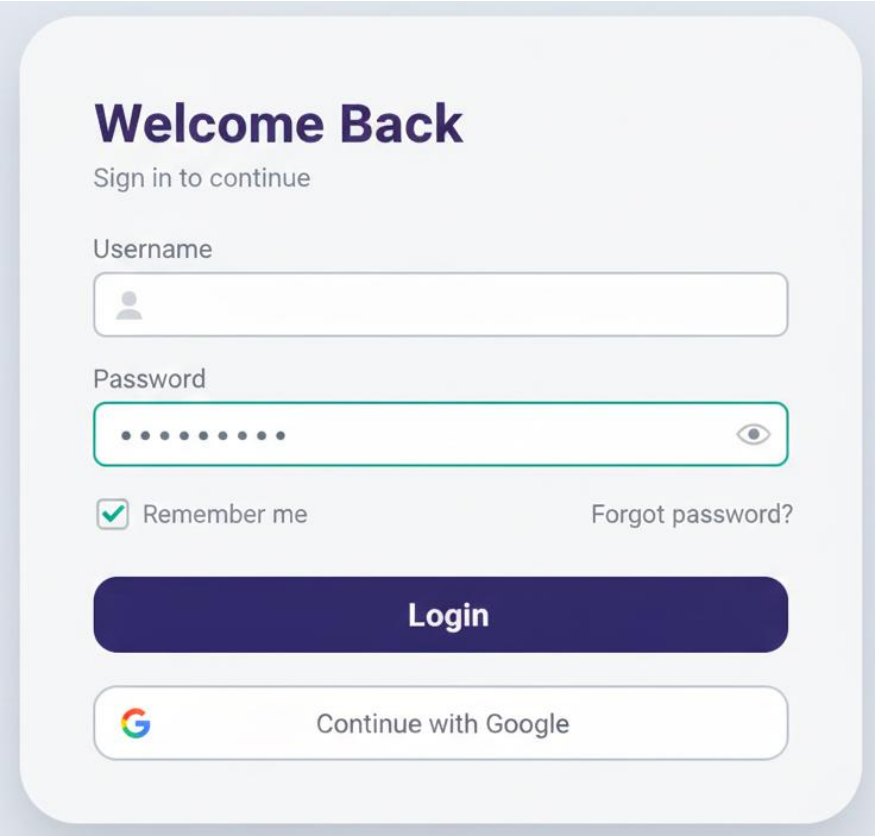
- Every control has settings you can change: text, color, size, whether it's enabled or not

- Label, TextField, PasswordField, Button, CheckBox. Learn these!



For illustration purposes only

- Labels: Use labels to explain what each text box is for. They don't do anything interactive
- TextFields: Users can type one line of text. Perfect for usernames, email addresses, or short answers
- PasswordField: Shows dots or asterisks instead of real letters. Keeps passwords and secrets safe and private



Programs wait for user actions. When something happens, your code responds and does work

- Clicking, typing, moving the mouse: **each action creates an event** your code can listen for

- ActionEvent happens on button clicks. KeyEvent happens when typing. MouseEvent happens when moving around

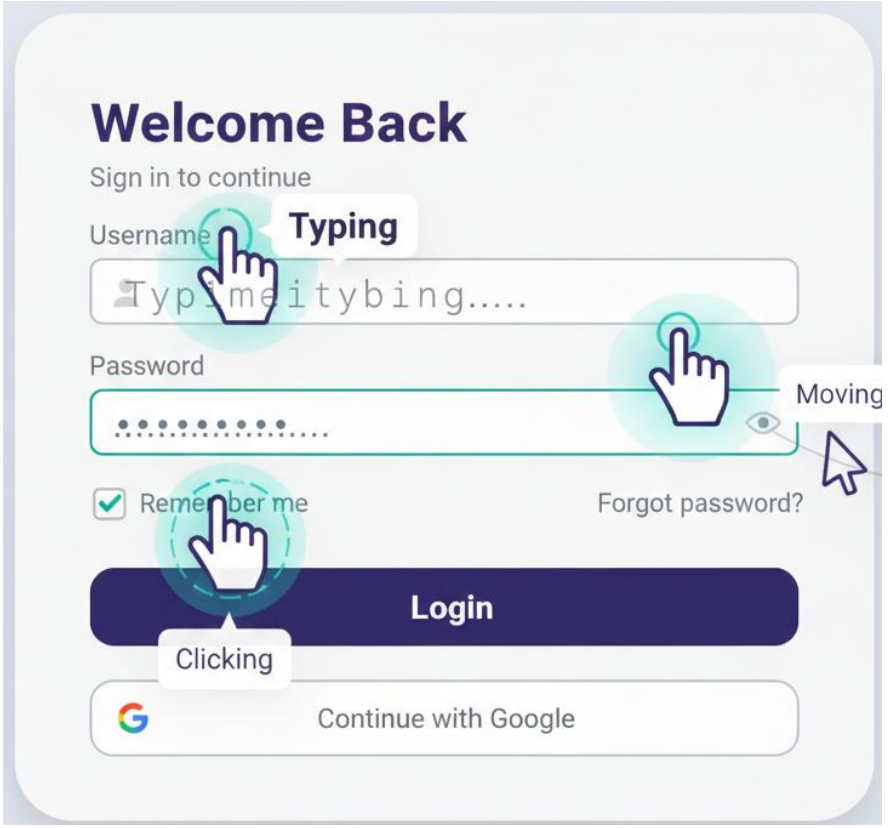
- When an event occurs, the handler runs your code right away to process it

Listen for key presses. Users expect to submit login forms by pressing Enter, not just clicking

- KEY_PRESSED fires when finger goes down. KEY_TYPED when a character appears. KEY_RELEASED when finger lifts up
- You can check which key was pressed. Enter, Escape, arrows: you control what happens for each
- Let users press Enter to submit instead of clicking the button. Makes the form feel natural

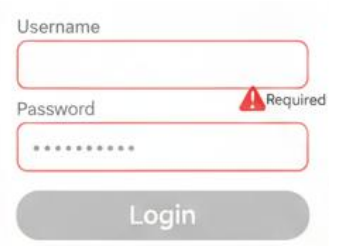
Track mouse movement, clicks, and when it enters or leaves a control on screen

- Detect single clicks, double clicks, right clicks. Know where the mouse is and what it's doing
- Trigger code when mouse enters a button for hover effects, or leaves to clean up
- Most of the time, use ActionEvent for buttons. **Mouse events are for special cases**



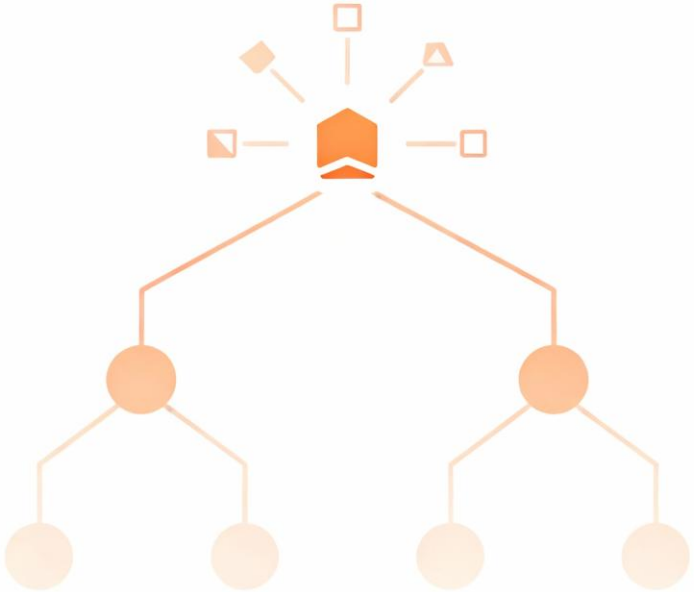
Check that users filled in required fields. Show users if something went wrong or worked perfectly

- Check that username and password are not empty before trying to log the user in



- Show a message if login failed. Show success if it worked. Make it clear and simple

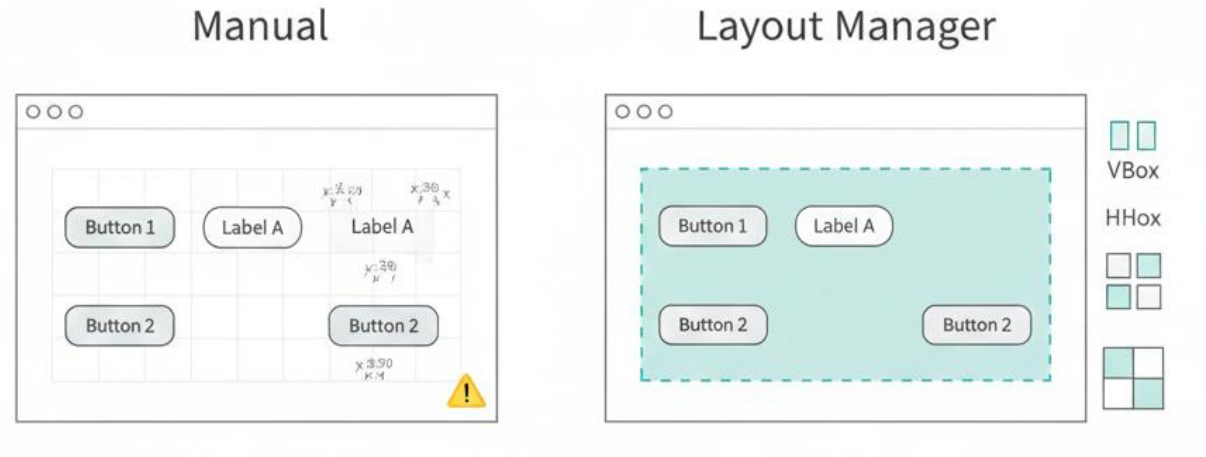
- Turn off the button while processing. Disable fields if validation fails. Guide users clearly



Layouts
Completing the Login Form

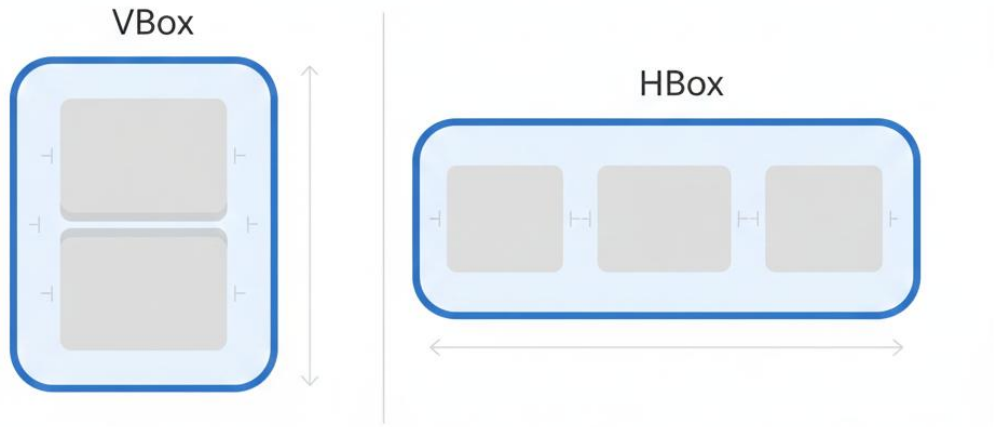
Don't position things manually. Let layout managers arrange everything automatically and correctly

- Hardcoding x and y positions fails when windows resize or run on different computers
- They resize and reposition everything when the window changes size or the screen is different



VBox stacks things vertically. HBox arranges things horizontally. Simple and powerful for most layouts

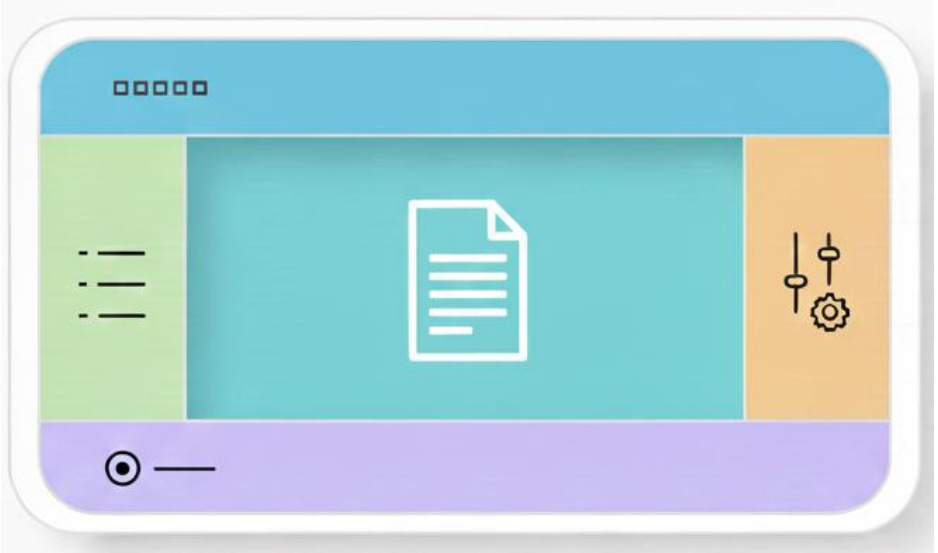
- Add space between items, align them left or center, and adjust padding around the edges
- Let buttons and fields grow when the window gets bigger. Keep everything looking balanced and good



UNIVERSITETI METROPOLITAN TIRANA

BorderPane divides the window into five regions. Perfect for building professional application layouts

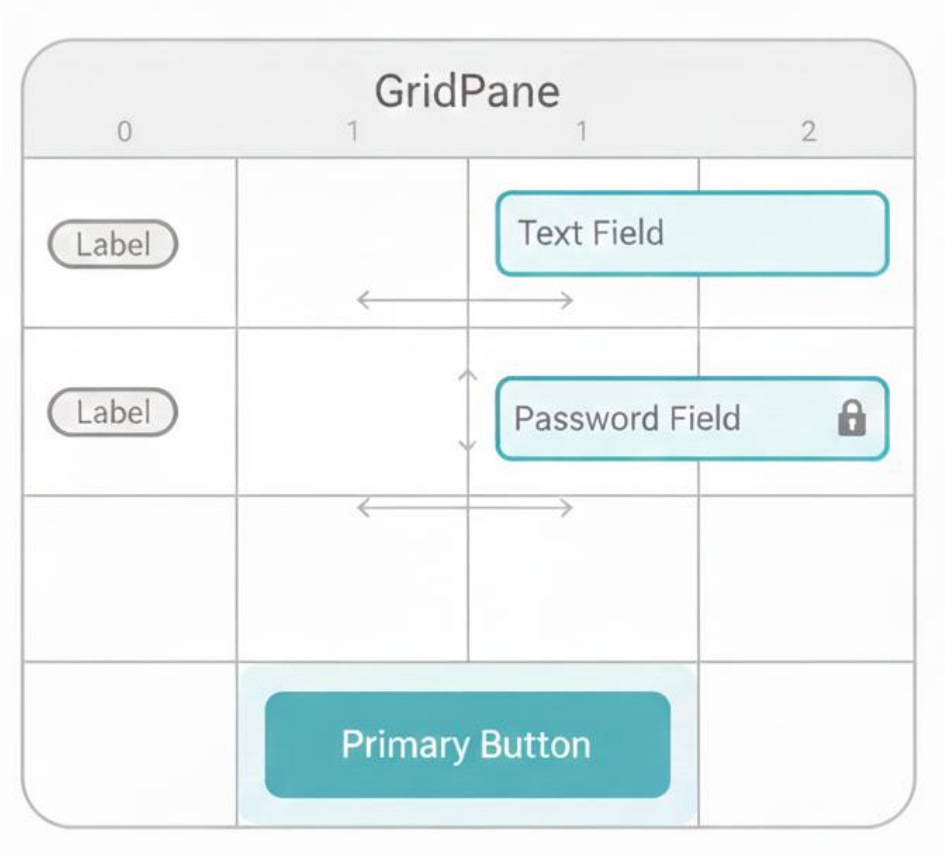
- Top for menus, bottom for status, left and right for sidebars, center for main content
- Use only the ones you need. Leave others empty. Center region gets all extra space
- This is how professional desktop apps organize themselves.



Arrange controls in rows and columns like a spreadsheet. Perfect for forms and data displays

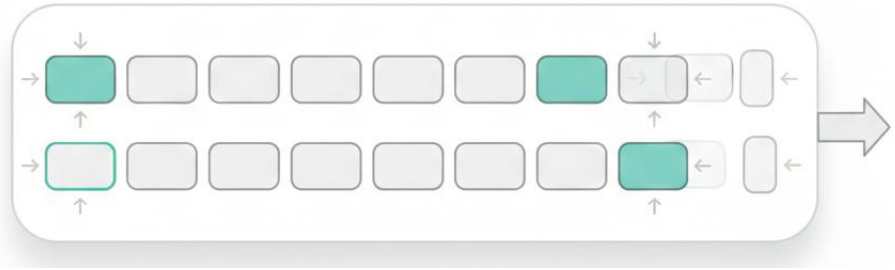
- Place labels in one column, text fields in another. Rows and columns keep everything aligned neatly

- Tell GridPane which columns should expand when the window gets bigger. Keep proportions balanced

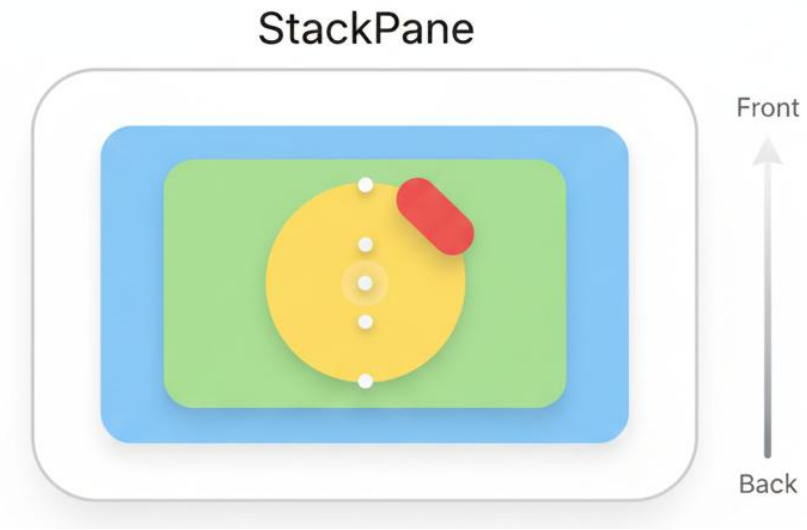


FlowPane wraps content like text. StackPane layers items on top of each other

- FlowPane: Items flow left to right, then wrap to the next line when space runs out. Good for buttons



- StackPane: Put one item on top of another. Perfect for backgrounds, overlays, and cards stacked together



Bring it all together. A fully working login form with proper structure and user interaction

- Stage holds Scene holds Layout holds Labels, TextFields, PasswordField, and Button all organized perfectly

- Click the button or press Enter to submit. Validation checks fields. Messages show success or error

- Form resizes gracefully. All pieces stay aligned.
  It looks like a real application users expect

UNIVERSITETI® METROPOLITAN TIRANA

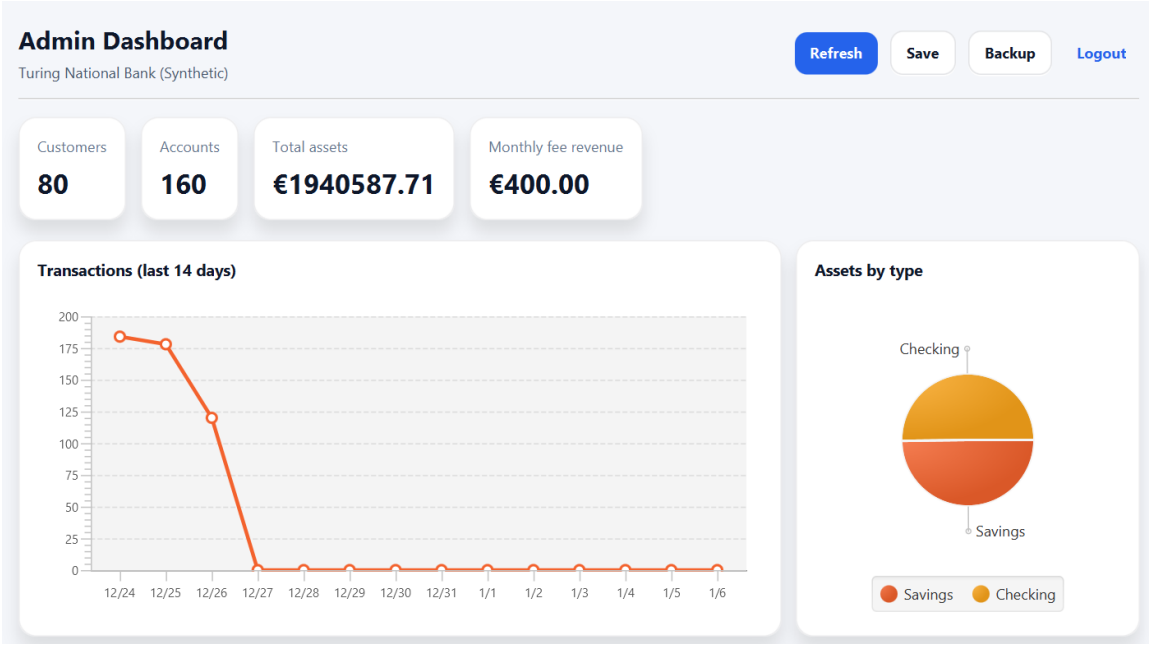
You've learned foundations, components, and layouts. You can now build functional desktop applications

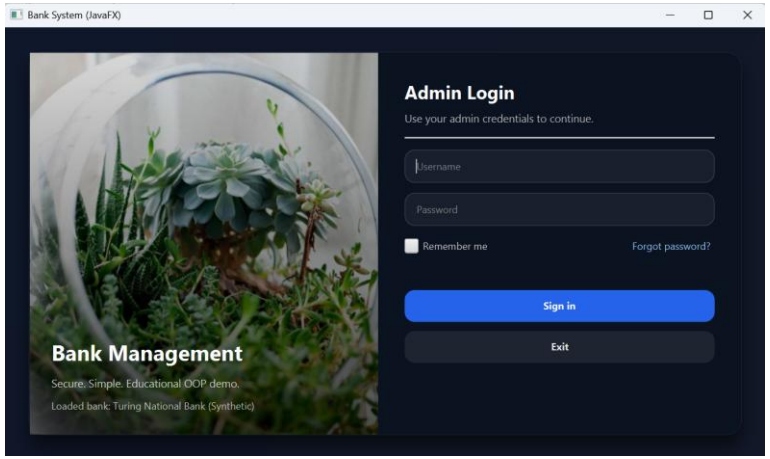
- JavaFX architecture, how to create windows and scenes, build interactive forms, and arrange them with layouts
- Learn to style with CSS, use advanced controls, manage application state, and build larger projects
- Create more forms. Experiment with different layouts. Build small projects to strengthen your skills today

UNIVERSITETI® METROPOLITAN TIRANA

Coming next

SUMMARY & NEXT STEPS





Life isn’t always pretty.

At least your app can be.