Introduction to Java for C++ Programmers

JavaFx

Segment 1- Basics

Professor: Mahboob Ali

JavaFX vs Swing and AWT

- Swing and AWT are replaced by the JavaFX platform for developing rich Internet applications.
- When Java was introduced, the GUI classes were bundled in a library known as the *Abstract Windows Toolkit (AWT)*.
- AWT is fine for developing simple graphical user interfaces, but not for developing comprehensive GUI projects.
- In addition, AWT is prone to platform-specific bugs.

JavaFX vs Swing and AWT

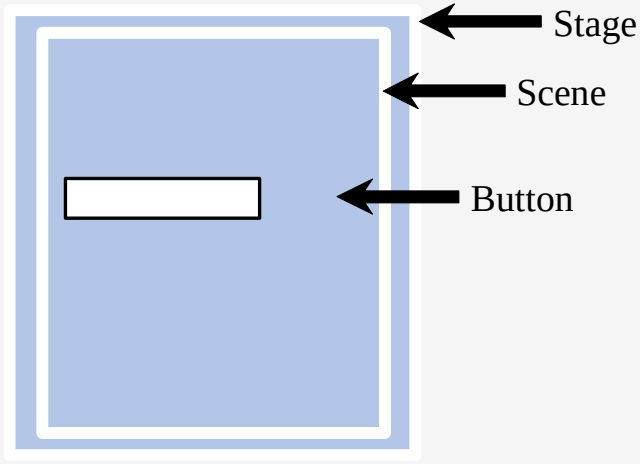
- The AWT user-interface components were replaced by a more robust, versatile, and flexible library known as *Swing components*.
-
- Swing components are painted directly on canvases using Java code.

- Swing components depend less on the target platform and use less of the native GUI resource.

- With the release of Java 8, Swing is replaced by a completely new GUI platform known as *JavaFX*.

Basic Structure of JavaFX

- Application
- Override the start(Stage) method
- Stage, Scene, and Nodes



Simple Button Example

Multiple Stage Example

Our First JavaFX Program

- JavaFX programs are based on the analogy of a stage (think "theater stage" for the moment).
- On the stage are scenes, and each scene is also made up of other components.
- On a theater stage, the stage may be divided into portions, where individual scenes take place.
- Each scene's set will have actors, props, backdrops, lighting, etc.
- In JavaFX, we create the components, add them to scenes, and then add scenes to the stage

Our First JavaFX Program

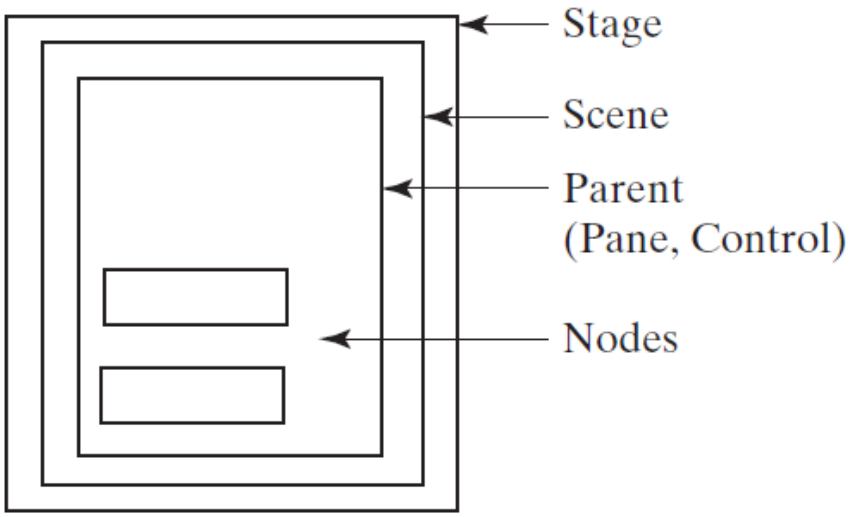
- By default, stages (windows) are resizable.
- Note that we have minimize and maximize buttons
- If we want our stage to be of fixed size (i.e., not resizable), we can set that property with `stage.setResizeable(false)`

PANES, UI CONTROLS, AND SHAPES

Panes, UI Controls, and Shapes

- Previous program we put the button directly on the scene, which centered the button and made it occupy the entire window.
- Rarely is this what we really want to do
- One approach is to specify the size and location of each UI element (like the buttons)
- A better solution is to put the UI elements (known as *nodes*) into *containers* called *panes*, and then add the panes to the scene.

Button in Pane Example



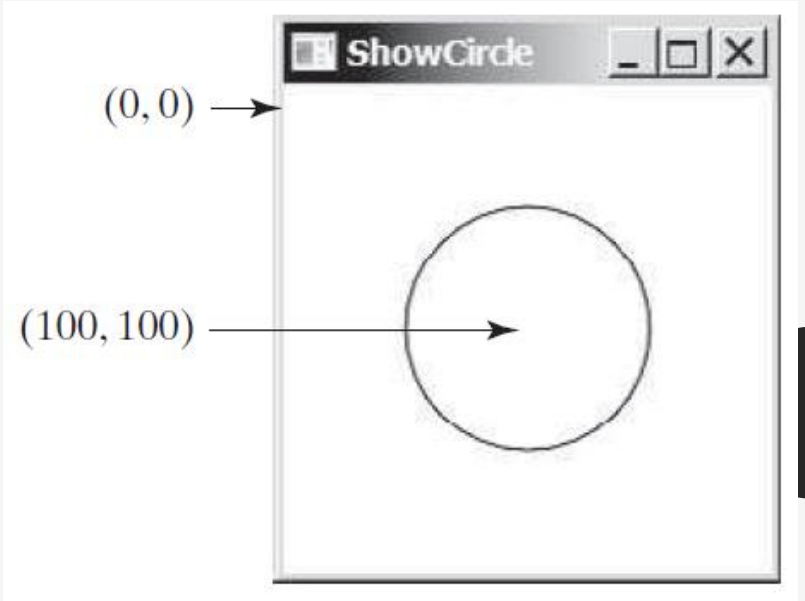
Panes, UI Controls, and Shapes

- Beyond the obvious, typical "active" UI elements (things we can interact with, like buttons, etc.), are static shapes – lines, circles, etc.
- Lets talk about coordinates within a pane.
- The top-left corner of a scene is *always* (o , o), and the (positive) X-axis goes to the right, and the (positive)

  Y-axis goes down.

  Visually, we're in Cartesian quadrant IV, but Y stays positive.
- All coordinates are in pixels

Show Circle Example



PROPERTY BINDING

Binding Properties

- JavaFX introduces a new concept called *binding property* that enables a *target object* to be bound to a *source object*.
- If the value in the source object changes, the target property is also changed automatically.
- The target object is simply called a *binding object* or a *binding property*.

Property Binding

- In the previous example, the (x, y) location of the center of the circle was static – it will always be located at (100, 100).

- What if we want it to be centered in the pane, such that if we re-size the window, the circle will move to stay centered?

- In order to do so, the circle's center has to be *bound* to the pane's height and width, such that a change to the height or width will force a change to the x or y value of the circle's center.

- This is what *property binding* is all about.

Show Circle Centered Example

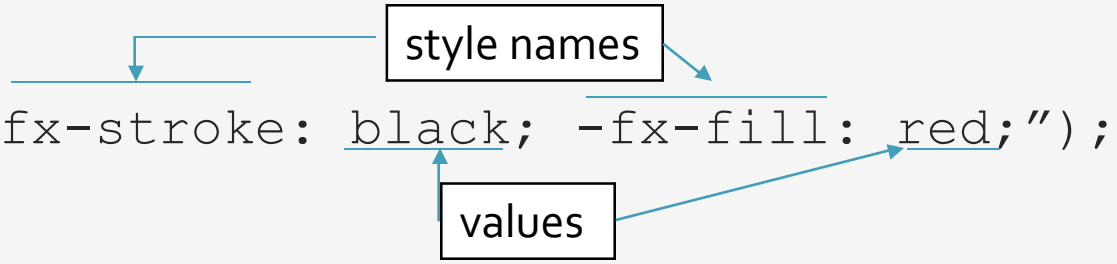
COMMON PROPERTIES AND METHODS FOR NODES

Common Node Properties & Methods

- Nodes share many common properties
- I am introducing only two properties in the slides – style and rotate
- JavaFX style properties are a lot like CSS (Cascading Style Sheets) use to specify styles in HTML (Web) pages.
- Thus, it's known as JavaFX CSS
- The official Java documentation on these properties can be found here.

Common Node Properties & Methods

- In JavaFX CSS, we can set more than one property at a time.
- Each property begins with the prefix `-fx-`, and is of the form styleName: value, with multiple settings separated by semicolons.
- For example:



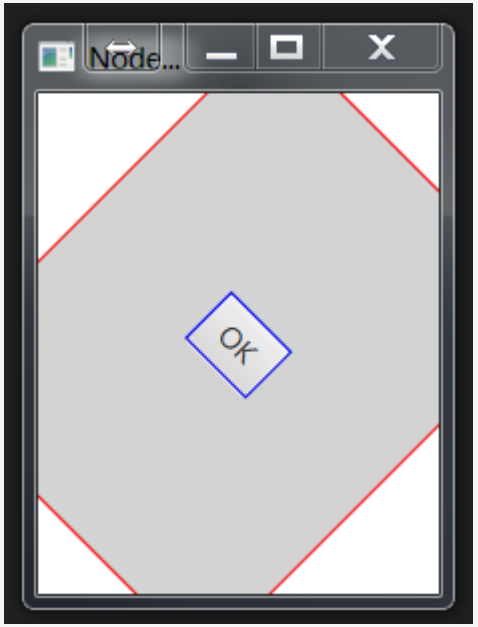

```
circle.setStyle("-fx-stroke: black; -fx-fill: red;");
```

- Is equivalent to:

```
circle.setStroke(Color.BLACK);
circle.setFill(Color.RED);
```

Common Node Properties & Methods

- Below example shows a program that creates a button with a blue border, uses JavaFX CSS to set its pane's style to a red border and a light gray fill, and then rotates the button 45 degrees
  - Note: positive rotation is clockwise; negative is CCW



THE IMAGE AND IMAGEVIEW CLASSES

The Image & ImageView Classes

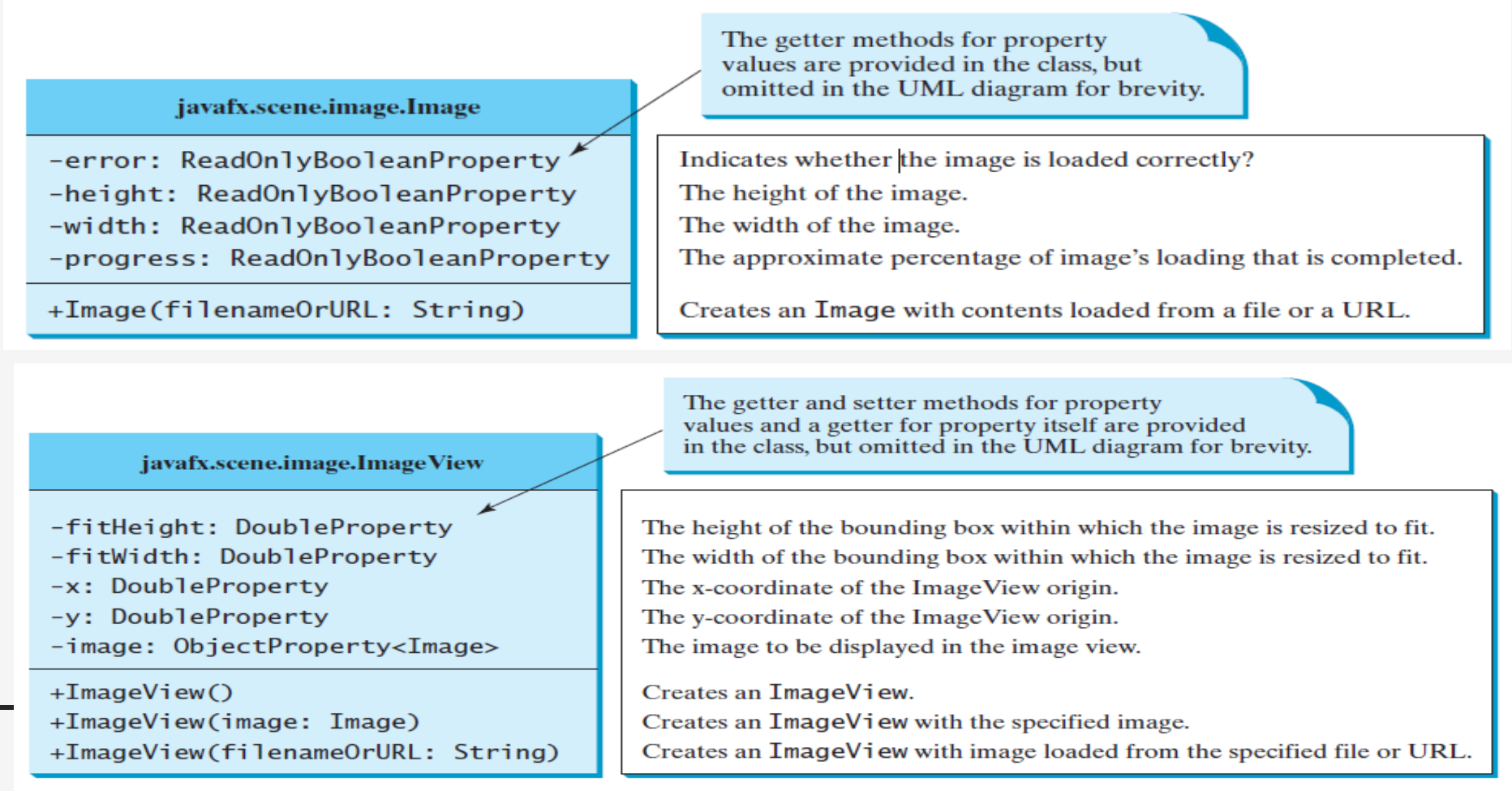
- Just like the File class to hold information about a file, but not to actually read / write to it.
  - For reading and writing , we used a Scanner / PrintWriter, connected to the File object

- Similarly, the Image class is a container for an image, but can't be used to actually display an image.
  - For that, we use the ImageView node (and attach it to a scene)

Show Image Example

The Image & ImageView Classes

- We construct an Image from a filename (or a URL), and then we can give the Image to an ImageView object to actually display it.
- The Image and ImageView classes.



LAYOUT PANES

Layout Panes

- So far we add our Nodes to a Pane, and then add the Pane to a Scene, and then the Scene to a Stage (probably the primary stage).
- How do we arrange (i.e., layout) the Nodes on the pane?
- Java has several different kinds of Panes that do a lot of the layout work for us.
- Previous example the Pane , HBox, and StackPane are used; we'll start with the FlowPane…

The FlowPane

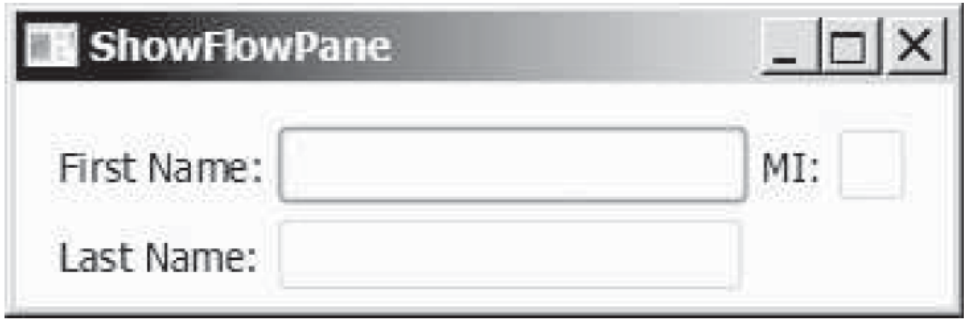
- The FlowPane can be set up to work in "reading order", by using Orientation.HORIZONTAL or in sequential top-to-bottom columns (Orientation.VERTICAL)

- One can also specify the gap between nodes (in pixels)

- So far, we have seen putting Button and ImageView nodes on a pane.

- Example below uses the FlowPane and Label & TextField nodes.

Show Flow Pane Example



The Grid Pane

- A few notes:
  - The "Add" button is right-aligned within its cell
  - The whole frame is centered
  - The labels get the default horizontal alignment of "left"
  - We specify the column first (backwards from arrays)
  - Not every cell needs to be filled
  - Elements can be moved from one cell to another

Grid Pane Example

The BorderPane

- The BorderPane divides the pane into five "regions"

Border Pane Example