User Interfaces

Designing a UI for your application

Randy J. Fortier
randy.fortier@uoit.ca
@randy_fortier

Outline

- JavaFX
- User interface components
- Event handling
- FXML
- Canvas graphics

User Interfaces

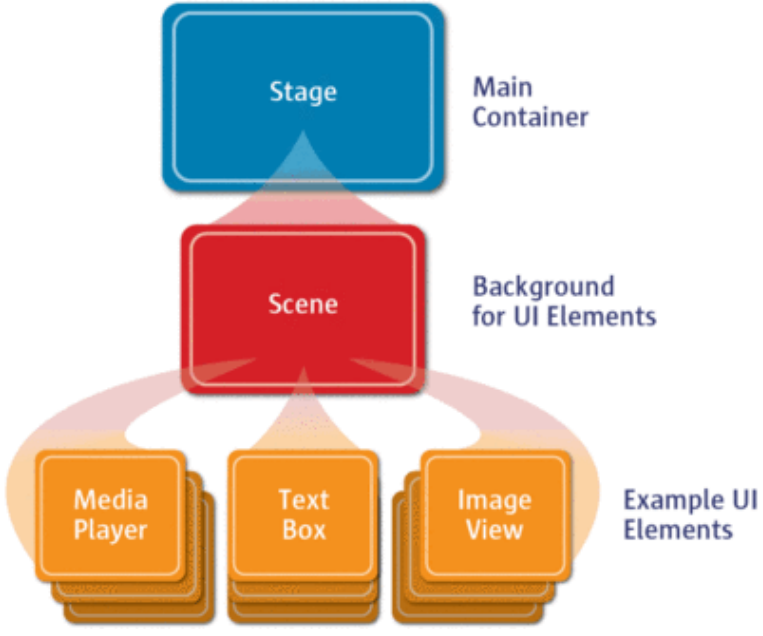
JavaFX

JavaFX

- A user interface library for Java
  - A replacement for Java Swing
  - v1.0 introduced in 2008
  - v8.0 integrated into Java SE
- Features:
  - Modern look and feel, compared to Swing
  - XML-based UI definitions
  - Stylable controls (via CSS)
  - 2D graphics (v2.1+)
  - 3D graphics (v9.0+)

JavaFX

- Core Objects:



User Interfaces

UI Components

JavaFX UI Components

- Button - button
- ButtonBar - a palette of buttons
- CheckBox - a boolean toggle button
- ChoiceBox - dropdown list
- ComboBox - editable, dropdown list
- Label - uneditable text
- Hyperlink - a clickable label
- PasswordField - a text field with hidden text

JavaFX UI Components (cont'd)

- RadioButton - a choice button (grouped)
- Separator - visual line separator for buttons, etc.
- Slider - slide to pick a number (discrete or continuous)
- Spinner - number selector
- TextArea - multi-line text entry
- TextField - single-line text entry
- ToggleButton - button with an on/off state
- Toolbar - a horizontal or vertical toolbar

JavaFX Layouts

- BorderPane - group with N,S,W,E, and Centre
- FlowPlane - a word-wrapped pane
- GridPane - a grid-oriented group
- HBox - horizontal linear layout
- VBox - vertical linear layout

Higher-level Controls

- ProgressBar - horizontal bar showing progress
- ProgressIndicator - circular control showing activity
- ScrollPane - adds a scroll bar to a UI
- WebView - shows a web page

Popups and Choosers

- Alert - popup window
- ColorPicker - colour chooser
- DatePicker - date chooser

Grouping Elements

- ListView - a vertical list of items
- SplitPane - a horizontal or vertical dual panel
- TabPane - a group of stacked UIs, with tabs to select between them
- Tab - individual tab/page in a tab panel

Menus

- MenuBar - the horizontal menu bar
- Menu - menus and submenus
- ContextMenu - popup menu
- MenuItem - normal menu item (icons,shortcuts,accel)
- CheckMenuItem - checkable menu item
- RadioMenuItem - choice menu item

Trees

- TreeView - view hierarchical data in a tree
- TreeTableView - table with collapsible sections
- TreeItem - normal tree item
- CheckBoxTreeItem - checkable tree item

Tables

- Table
- TableColumn
- CellFactory
- CellValueFactory
- PropertyValueFactory

Basic JavaFX Program

```
public class Main extends Application {
    @Override
    public void start(Stage primaryStage) throws Exception {
        primaryStage.setTitle("JavaFX Demo");

        Scene scene = new Scene(layout, 600, 600);
        primaryStage.setScene(scene);
        primaryStage.show();
    }

    public static void main(String[] args) {
        launch(args);
    }
}
```

User Interfaces

Event Handling

Handling Events

- You can handle events in code:

```
Button addButton = new Button("Insert");
addButton.setOnAction(new EventHandler<ActionEvent>() {
    @Override public void handle(ActionEvent e) {
        String firstName = firstNameField.getText();
        // handle the event
    }
});
```

User Interfaces

FXML

FXML

- FX Markup Language
- XML-based syntax for declaring user interfaces
  - Similar syntax to HTML and Android layouts
- A Controller class
  - Initialization
  - Event handlers

Sample FXML

- The following is a sample FXML file:

```
<BorderPane xmlns="http://javafx.com/javafx/8"
            xmlns:fx="http://javafx.com/fxml/1"
            fx:controller="sample.Controller">

    <top>
        <Label fx:id="firstNameLabel"
               text="First name:" />
    </top>
    <center>
        <TextField fx:id="firstNameField" />
    </center>
    <bottom>
        <Button text="Next"
                onAction="#next"/>
    </bottom>
</BorderPane>
```

Sample Controller

- The following is a sample controller class:

```
public class Controller {
    @FXML private Label firstNameLabel;
    @FXML private TextField firstNameField;

    public void next(ActionEvent e) {
        // handle the event
    }
}
```

User Interfaces

2D Graphics

Canvas

- Similar to Canvas in HTML5 and Android:
  - Drawing basic shapes
  - Drawing images
  - Animation

Basic Drawing

- The skeleton code:

```
@FXML private Canvas canvas;

@Override
public void start(Stage primaryStage) throws Exception{
    Group root = new Group();
    Scene scene = new Scene(root, 800, 600, Color.LIGHTGRAY);

    canvas = new Canvas();
    canvas.widthProperty().bind(primaryStage.widthProperty());
    canvas.heightProperty().bind(primaryStage.heightProperty());

    root.getChildren().add(canvas);
    primaryStage.setScene(scene);
    primaryStage.show();

    draw(root); // the good stuff happens here
}
```

Basic Drawing

- Now we can draw basic shapes:

```
private void draw(Group group) {
    GraphicsContext gc = canvas.getGraphicsContext2D();

    gc.setStroke(Color.BLUE);
    gc.fillRect(50, 50, 100, 75);

    gc.setFill(Color.BLUE);
    gc.fillRect(250, 50, 100, 75);
}
```

Basic Shapes

- Lines:

```
private void draw(Group group) {
    GraphicsContext gc = canvas.getGraphicsContext2D();
    gc.clearRect(0, 0, canvas.getWidth(), canvas.getHeight());

    gc.setStroke(Color.BLUE);

    gc.strokeLine(50, 50, 150, 250);
}
```

Basic Shapes

- Rectangular shapes:

```
private void draw(Group group) {
    GraphicsContext gc = canvas.getGraphicsContext2D();

    gc.setFill(Color.BLUE);
    gc.setStroke(Color.BLUE);

    gc.fillRect(250, 50, 100, 75);
    gc.strokeRect(250, 175, 100, 75);

    gc.fillRoundRect(450, 50, 100, 75, 10, 10);
    gc.strokeRoundRect(450, 175, 100, 75, 20, 20);
}
```

Basic Shapes

- Round shapes:

```
private void draw(Group group) {
    GraphicsContext gc = canvas.getGraphicsContext2D();

    gc.strokeOval(650, 50, 100, 75);
    gc.fillOval(650, 175, 100, 75);

    gc.strokeArc(50, 350, 100, 75, 115.0, 45.0, ArcType.ROUND);
    gc.fillArc(50, 500, 100, 75, 45.0, 115.0, ArcType.ROUND);
}
```

Basic Shapes

- Polygons:

```
private void draw(Group group) {
    GraphicsContext gc = canvas.getGraphicsContext2D();

    gc.strokePolygon(new double[] {250, 310, 300, 250},
                     new double[] {350, 360, 380, 400}, 4);
    gc.fillPolygon(new double[] {250, 310, 300, 250},
                     new double[] {500, 510, 530, 550}, 4);
}
```

Basic Shapes

- Text:

```
private void draw(Group group) {
    GraphicsContext gc = canvas.getGraphicsContext2D();

    Font font = new Font("Arial", 24);
    gc.setFont(font);
    gc.strokeText("CSCI2020u", 450, 400);
    gc.fillText("CSCI2020u", 450, 550);
}
```

Drawing Images

- Images:

```
private void draw(Group group) {
    GraphicsContext gc = canvas.getGraphicsContext2D();

    Image image = new Image("picture.png");
    gc.drawImage(image, 200, 300);
}
```

User Interfaces

Animation

Animation

- Animation involves a timed loop
- Choose a frames-per-second value
- Set the timer callback, accordingly
- Implement a callback function that draws a single frame

Animation

```
@Override
public void start(Stage primaryStage) throws Exception{
    ...
    Timeline timeline = new Timeline();
    timeline.setCycleCount(Timeline.INDEFINITE);
    EventHandler eventHandler = new EventHandler<ActionEvent>() {
        @Override
        public void handle(ActionEvent e) {
            gc.setFill(Color.LIGHTGRAY);
            gc.fillRect(x, y, 200, 300);
            // increment x offset and y offset
        }
    });
    KeyFrame keyFrame = new KeyFrame(Duration.millis(50), eventHandler);
    timeline.getKeyFrames().add(keyFrame);
    timeline.playFromStart();
    ...
```

Wrap-Up

- In this section we learned about:
  - JavaFX
    - Skeleton code
    - Basic controls
    - High-level controls
    - Layouts and groups
    - Event handling
    - FXML
    - 2D graphics
    - Animation