Graphical User Interfaces

CSC 1051 – Data Structures and Algorithms I
Dr. Mary-Angela Papalaskari
Department of Computing Sciences
Villanova University

Course website:
http://www.csc.villanova.edu/~map/1051/

Many slides in this presentation are adapted from Prof. Barbara Zimmerman’s CSC 1051 slides and the slides accompanying Java Software Solutions, 9th edition by Lewis & Loftus
Outline

• Pixels & bits & colors
• JavaFX Introduction
• Shapes
Pixels and Graphics

Programs represent pictures as grids of picture elements or **pixels**

Picture **resolution**: Relates to how many pixels are used
Display size (300, 120) vs. (500, 320)

Scene scene = new Scene(root, 300, 120, Color.LIGHTGREEN);
Coordinate System

- The origin of the Java coordinate system is in the upper left corner
- All visible points have **positive, int** coordinates
Representing Images

**Bitmap**
- 1 bit

**Grayscale**
- 8 bits

**RGB Color**
- 3 colors: red, green, blue
- 24 bits

**sRGB Color**
- 3 colors: red, green, blue + alpha
- 32 bits
Example: Representing Pixels in RGB

\[ \text{rgb}(01110100, 01010110, 10001110) \]

\[ \text{red}=116 \quad \text{green}=86 \quad \text{blue}=142 \]

x = 11

y = 8

fff
Additive/Subtractive Color

We choose 3 primary colors that can be combined to produce almost all visible colors.

**Additive primaries**
- combining light
  - Red
  - Green
  - Blue

**Subtractive primaries**
- combining ink, thus subtracting light
  - Cyan
  - Yellow
  - Magenta

CSC 1051 M.A. Papalaskari, Villanova University
RGB Color in JavaFX

- Color: Each pixel has a color associated with it
  - We can specify a color using the Color.rgb method:
  - Each component has a value from 0-255

```java
Scene scene = new Scene(root, 300, 120, Color.rgb(0, 0, 255));
```

... Or we can use a (predefined) Color constant

```java
Scene scene = new Scene(root, 300, 120, Color.BLUE);
```

See Color class in the JavaFX API:
[https://docs.oracle.com/javase/8/javafx/api/javafx/scene/paint/Color.html](https://docs.oracle.com/javase/8/javafx/api/javafx/scene/paint/Color.html)
The Color class: methods

• The static `rgb` method in the `Color` class returns a `Color` object with a specific RGB value:

```
Color purple = Color.rgb(183, 44, 150);
Scene scene = new Scene(root, 300, 120, Color.rgb(0, 255, 0));
```

• The `color` method uses percentages:

```
Color maroon = Color.color(0.6, 0.1, 0.0);
```

• Both methods allow an extra parameter to specify alpha value:

```
Color purple = Color.rgb(183, 44, 150, 0.3);
Color maroon = Color.color(0.6, 0.1, 0.0, 0.8);
```
Example: HelloJavaFX

- HelloJavaFX program makes use of Inheritance

```java
public class HelloJavaFX extends Application
```

- New class **HelloJavaFX**
- Based on an existing parent class **Application**
- Reuses items from the parent class
  - attributes (variables)
  - methods (code)

- JavaFX programs inherit core graphical functionality from the Application class
Example: HelloJavaFX – Two methods

• A JavaFX program has a **start** method

```java
public void start(Stage primaryStage)
{
    ... // this method does all the work
}
```

• The **main** method is only needed to launch the JavaFX application (uses **start** method)

```java
public static void main(String[] args)
{
    launch(args); // set up and invoke start()
}
```

• **See HelloJavaFX.java**
import javafx.application.Application;
import javafx.scene.Group;
import javafx.scene.Scene;
import javafx.scene.paint.Color;
import javafx.scene.text.Text;
import javafx.stage.Stage;

public class HelloJavaFX extends Application {

    public void start(Stage primaryStage) {
        Text hello = new Text(50, 50, "Hello, JavaFX!");
        Text question = new Text(120, 80, "How's it going?");

        Group root = new Group(hello, question);
        Scene scene = new Scene(root, 300, 120, Color.LIGHTGREEN);
    }

    continued
continued

```java
primaryStage.setTitle("A JavaFX Program");
primaryStage.setScene(scene);
primaryStage.show();
```

// Launched the JavaFX application. This method is not required // in IDEs that launch JavaFX applications automatically.

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

**A JavaFX Program**

Hello, JavaFX!

How's it going?
Adding **Text** objects to a **Group**

- **NOTE:** position of each **Text** object is specified explicitly:

  ```java
  Text hello = new Text(50, 50, "Hello, JavaFX!");
  ```

- In the example, two **Text** objects are added to a **Group**:

  ```java
  Group root = new Group(hello, question);
  ```

- Order in the group ➔ order of adding to the scene
Scene displayed on primaryStage

- A Group serves as the root node of a Scene, which gets displayed on the primaryStage

```java
Scene scene =
    new Scene(root, 300, 120, Color.LIGHTGREEN);

primaryStage.setTitle("A JavaFX Program");
primaryStage.setScene(scene);
primaryStage.show();
```

- the primaryStage corresponds to the window being displayed
  - primaryStage is the parameter of the start method
Basic Shapes

- JavaFX shapes are represented by classes in one of the packages we import

```java
import javafx.scene.shape.*;
```

- A line segment is defined by the `Line` class, whose constructor accepts the coordinates of the two endpoints:

```java
// Line(startX, startY, endX, endY)
```

- For example:

```java
Line lineC = new Line(10, 20, 300, 80);
```
Basic Shapes

- A rectangle is specified by its upper left corner and its width and height:

```java
// Rectangle(x, y, width, height)
Rectangle r1 = new Rectangle(30, 50, 200, 70);
```

- A circle is specified by its center point and radius:

```java
// Circle(centerX, centerY, radius)
Circle c1 = new Circle(100, 150, 40);
```
Basic Shapes

• An ellipse is specified by its center point and its radius along the x and y axis:

\[
\text{Ellipse}(\text{centerX}, \text{centerY}, \text{radiusX}, \text{radiusY})
\]

\[
\text{Ellipse } e = \text{new Ellipse}(100, 50, 80, 30);
\]

• Shapes are drawn in the order in which they are added to the group

• The stroke and fill of each shape can be set

• See \textit{Einstein.java}
import javafx.application.Application;
import javafx.scene.Group;
import javafx.scene.Scene;
import javafx.scene.paint.Color;
import javafx.scene.shape.*;
import javafx.scene.text.Text;
import javafx.stage.Stage;

public class Einstein extends Application
{
    //-------------------------------------------------------------------------------
    // Creates and displays several shapes.
    //-------------------------------------------------------------------------------
    public void start(Stage primaryStage)
    {
        Line line = new Line(35, 60, 150, 170);
        Circle circle = new Circle(100, 65, 20);
        circle.setFill(Color.BLUE);
        continued
    }
}
Rectangle rect = new Rectangle(60, 70, 250, 60);
rect.setStroke(Color.RED);
rect.setStrokeWidth(2);
rect.setFill(null);

Ellipse ellipse = new Ellipse(200, 100, 150, 50);
ellipse.setFill(Color.PALEGREEN);

Text quote = new Text(120, 100, "Out of clutter, find " +
   "simplicity.\n-- Albert Einstein");

Group root = new Group(ellipse, rect, circle, line, quote);
Scene scene = new Scene(root, 400, 200);

primaryStage.setTitle("Einstein");
primaryStage.setScene(scene);
primaryStage.show();

// We will typically exclude the main method. Use it to launch
// the application if needed.
}
Rectangle rect = new Rectangle(60, 70, 250, 60);
rect.setStroke(Color.RED);
rect.setStrokeWidth(2);
rect.setFill(null);

Ellipse ellipse = new Ellipse(200, 100, 150, 50);
ellipse.setFill(Color.PALEGREEN);

Text quote = new Text(120, 100, "Out of clutter, find simplicity.
-- Albert Einstein" );

Group root = new Group(ellipse, rect, circle, line, quote);

Scene scene = new Scene(root, 400, 200);

primaryStage.setTitle("Einstein");
primaryStage.setScene(scene);
primaryStage.show();

// We will typically exclude the main method. Use it to launch
// the application if needed.
HelloJavaFX Sketch – Find reference points

Practice:
1. Add the center of the circle and other endpoint of line
2. Verify width and height of all objects

Out of clutter, find simplicity. -- Albert Einstein
Groups – order matters

Group root = new Group(ellipse, rect, circle, line, quote);

• First thing drawn is the ellipse then the rectangle
  – What would happen if I had the ellipse last?

• Groups can be nested within groups

• Translating a shape or group shifts its position along the x or y axis

• A shape or group can be rotated using the setRotate method

• See Snowman.java

CSC 1051 M.A. Papalaskari, Villanova University
import javafx.application.Application;
import javafx.stage.Stage;
import javafx.scene.Group;
import javafx.scene.Scene;
import javafx.scene.paint.Color;
import javafx.scene.shape.*;

public class Snowman extends Application
{
    public void start(Stage primaryStage)
    {
        Ellipse base = new Ellipse(80, 210, 80, 60);
        base.setFill(Color.WHITE);

        Ellipse middle = new Ellipse(80, 130, 50, 40);
        middle.setFill(Color.WHITE);
    }
}

continued
Circle head = new Circle(80, 70, 30);
  head.setFill(Color.WHITE);

Circle rightEye = new Circle(70, 60, 5);
Circle leftEye = new Circle(90, 60, 5);
Line mouth = new Line(70, 80, 90, 80);

Circle topButton = new Circle(80, 120, 6);
topButton.setFill(Color.RED);
Circle bottomButton = new Circle(80, 140, 6);
bottomButton.setFill(Color.RED);

Line leftArm = new Line(110, 130, 160, 130);
  leftArm.setStrokeWidth(3);
Line rightArm = new Line(50, 130, 0, 100);
  rightArm.setStrokeWidth(3);

Rectangle stovepipe = new Rectangle(60, 0, 40, 50);
Rectangle brim = new Rectangle(50, 45, 60, 5);
Group hat = new Group(stovepipe, brim);
hat.setTranslateX(10);
hat.setRotate(15);

continued

Move and rotate hat

(snowman in original position)
```
Group snowman = new Group(base, middle, head, leftEye, rightEye,
mouth, topButton, bottomButton, leftArm, rightArm, hat);
snowman.setTranslateX(170);
snowman.setTranslateY(50);

Circle sun = new Circle(50, 50, 30);
sun.setFill(Color.GOLD);

Rectangle ground = new Rectangle(0, 250, 500, 100);
ground.setFill(Color.STEELBLUE);

Group root = new Group(ground, sun, snowman);
Scene scene = new Scene(root, 500, 350, Color.LIGHTBLUE);

primaryStage.setTitle("Snowman");
primaryStage.setScene(scene);
primaryStage.show();
```