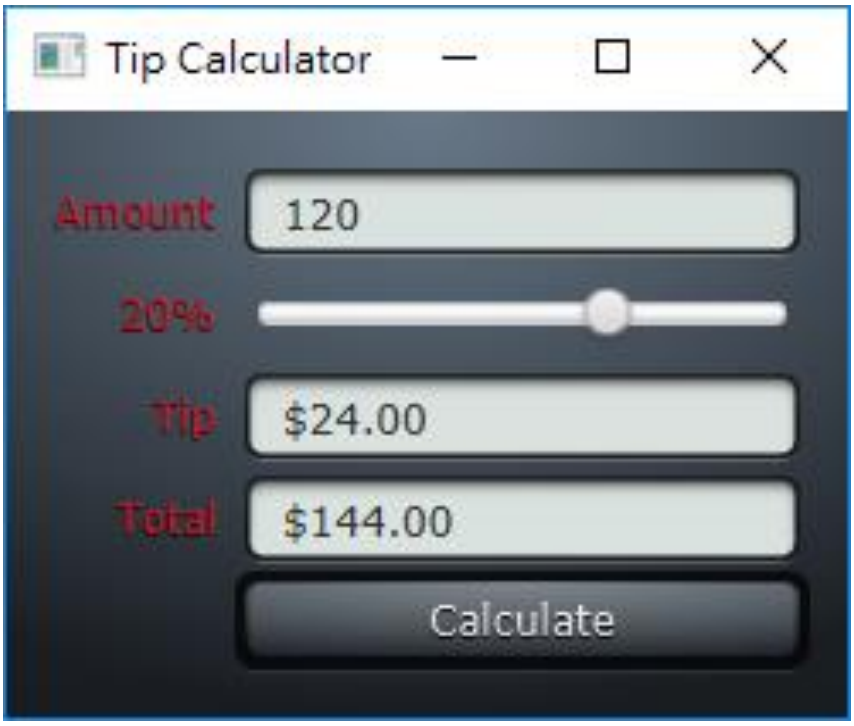
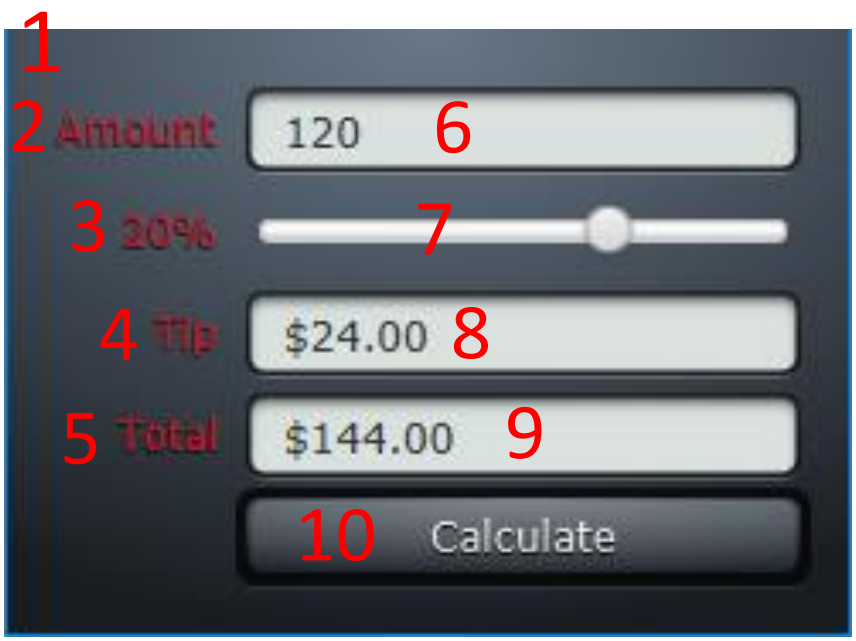
Course 03

Please implement following GUI by Scene Builder and complete the application with given codes. Study the codes carefully and make sure you get a best understanding of what/how/when the programs do.

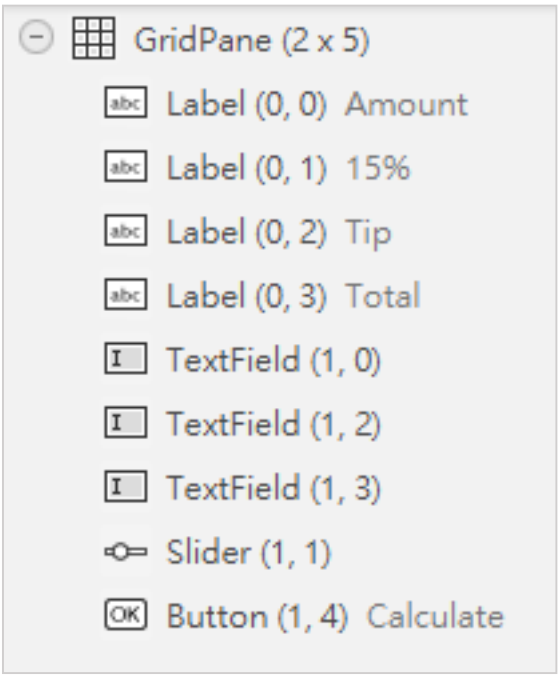
1. Tip Calculator



GUI Description:



Hierarchy:



0) File Name: TipCalculator.fxml
Controller Class: TipCalculatorController

1) GridPane

a) Stylesheets: TipCalculator.css
b) styleClass: background
c) 2 columns and 5 rows

d) Pref Width: Reset to default
e) Pref Height: Reset to default
f) Column 0
   Halignment: RIGHT
   Pref Width: Reset to default
g) Column 1
   Halignment: RIGHT
   Pref Width: Reset to default
h) Padding: 14 14 14 14 (TOP, RIGHT, BOTTOM, LEFT)
i) Hgap: 8

2) Label
a) styleClass: lab
b) Text: “Amount”

3) Label
a) styleClass: lab
b) fx:id: tipPercentageLabel
c) Text: “15%”

4) Label
a) styleClass: lab
b) Text: “Tip”

5) Label
a) styleClass: lab
b) Text: “Total”

6) TextField
a) styleClass: tex-field
b) fx:id: amountTextField

7) Slider
a) fx:id: tipPercentageSlider
b) Max: 30
c) Value: 15
d) Block Increment: 5

8) TextField
a) styleClass: tex-field

b) fx:id: tipTextField
c) Editable: uncheck
d) Focus Traversable: uncheck

9) TextField

a) styleClass: tex-field
b) fx:id: totalTextField
c) Editable: uncheck
d) Focus Traversable: uncheck

10) Button

a) id: button
b) Text: “Calculate”
c) Max Width: MAX_VALUE
d) On Action: calculateButtonPressed

TipCalculator.css

```
.background {
    -fx-background-repeat: repeat;
    -fx-background-color:
             linear-gradient(#38424b 0%, #1f2429 20%, #191d22 100%),
             linear-gradient(#20262b, #191d22),
             radial-gradient(center 50% 0%, radius 100%,
             rgba(114,131,148,0.9),
             rgba(255,255,255,0));
}

.lab {
    -fx-font-family: "Verdana";
    -fx-font-size: 12;
    -fx-text-fill: rgb(162,21,35,1);
    -fx-effect: dropshadow(one-pass-box, rgb(0,0,0,0.6), 0,0,0,1);
}

#button .text {
    -fx-effect: dropshadow(one-pass-box, rgb(0,0,0,0.8), 0,0,0,1);
}

#button {
```

```
    -fx-background-color:
                rgb(255,255,255,0.08), rgb(0,0,0,0.8), #090a0c,
                linear-gradient(#4a5661 0%, #1f2429 20%, #1f242a 100%),
                linear-gradient(#242a2e, #23282e),
                radial-gradient(center 50% 0%, radius 100%,
                                rgba(135,142,148,0.9),
                                rgba(255,255,255,0));
    -fx-background-radius: 7,6,5,4,3,5;
    -fx-background-insets: -3 -3 -4 -3, -3, 0, 1, 2, 0;
    -fx-font-family: "Verdana";
    -fx-text-fill: blue;
    -fx-text-fill: linear-gradient(white, #d0d0d0);
}

#button:focused, #button:hover {
    -fx-background-color:
                rgb(255,255,255,0.08), rgb(0,0,0,0.8), #090a0c,
                linear-gradient(#4a5661 0%, #1f2429 20%, #1f242a 100%),
                linear-gradient(#3f4950, #23282e),
                radial-gradient(center 50% 0%, radius 100%,
                rgba(135,142,148,0.9), rgba(255,255,255,0));
}

.tex-field {
    -fx-background-color:
                rgb(255,255,255,0.3), linear-gradient(rgb(0,0,0,0.5),
                rgb(0,0,0,0.8) 50%), rgb(218,226,224);
    -fx-background-radius: 6,5,4;
    -fx-background-insets: 0 0 -1 0, 0, 1.5;
    -fx-padding: 6 10 4 10;
    -fx-effect: innershadow(gaussian, rgb(0,0,0,0.8), 5, 0, 0, 2);
    -fx-font-family: "Verdana";
}

.tex-field:focused {
    -fx-background-color:
                rgb(235,235,235,0.5), rgb(0,0,0,0.4), rgb(255,255,255);
    -fx-test-fill: rgb(128,128,128);
}
```

TipCalculator.java

```
// Main application class that loads and displays the Tip Calculator's GUI.
import javafx.application.Application;
import javafx.fxml.FXMLLoader;
import javafx.scene.Parent;
import javafx.scene.Scene;
import javafx.stage.Stage;

public class TipCalculator extends Application {
   @Override
   public void start(Stage stage) throws Exception {
      Parent root =
         FXMLLoader.load(getClass().getResource("TipCalculator.fxml"));

      Scene scene = new Scene(root); // attach scene graph to scene
      stage.setTitle("Tip Calculator"); // displayed in window's title bar
      stage.setScene(scene); // attach scene to stage
      stage.show(); // display the stage
   }

   public static void main(String[] args) {
      // create a TipCalculator object and call its start method
      launch(args);
   }
}
```

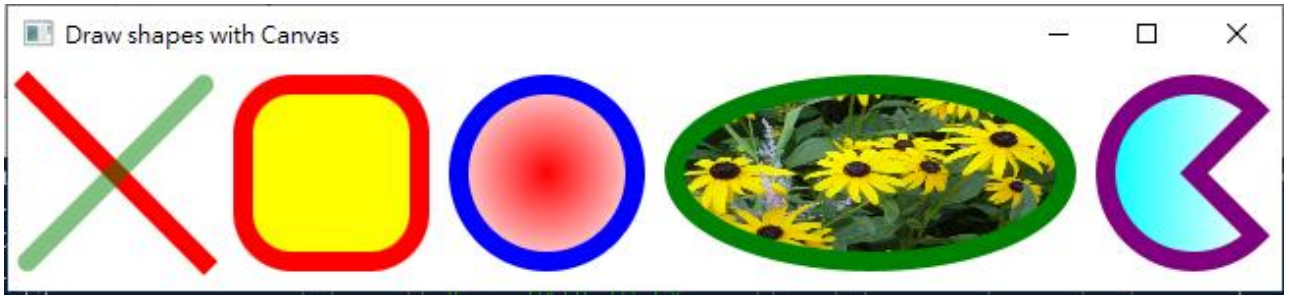
TipCalculatorController.java

```
// TipCalculatorController.java
// Controller that handles calculateButton and tipPercentageSlider events
import java.math.BigDecimal;
import java.math.RoundingMode;
import java.text.NumberFormat;
import javafx.beans.value.ChangeListener;
import javafx.beans.value.ObservableValue;
import javafx.event.ActionEvent;
import javafx.fxml.FXML;
import javafx.scene.control.Label;
import javafx.scene.control.Slider;
```

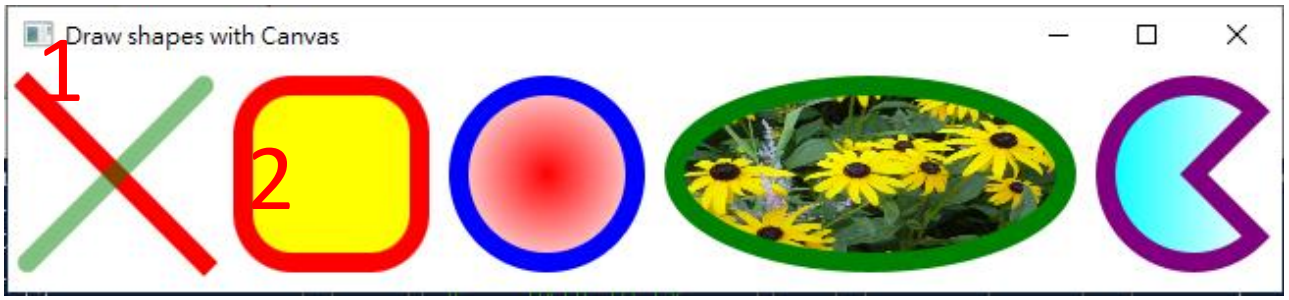
```
import javafx.scene.control.TextField;

public class TipCalculatorController {
   // formatters for currency and percentages
   private static final NumberFormat currency =
      NumberFormat.getCurrencyInstance();
   private static final NumberFormat percent =
      NumberFormat.getPercentInstance();

   private BigDecimal tipPercentage = new BigDecimal(0.15); // 15% default

   // GUI controls defined in FXML and used by the controller's code
   @FXML
   private TextField amountTextField;

   @FXML
   private Label tipPercentageLabel;

   @FXML
   private Slider tipPercentageSlider;

   @FXML
   private TextField tipTextField;

   @FXML
   private TextField totalTextField;

   // calculates and displays the tip and total amounts
   @FXML
   private void calculateButtonPressed(ActionEvent event) {
      try {
         BigDecimal amount = new BigDecimal(amountTextField.getText());
         BigDecimal tip = amount.multiply(tipPercentage);
         BigDecimal total = amount.add(tip);

         tipTextField.setText(currency.format(tip));
         totalTextField.setText(currency.format(total));
      }
      catch (NumberFormatException ex) {
         amountTextField.setText("Enter amount");
```

```
        amountTextField.selectAll();
        amountTextField.requestFocus();
      }
   }

   // called by FXMLLoader to initialize the controller
   public void initialize() {
      // 0-4 rounds down, 5-9 rounds up
      currency.setRoundingMode(RoundingMode.HALF_UP);

      // listener for changes to tipPercentageSlider's value
      tipPercentageSlider.valueProperty().addListener(
         new ChangeListener<Number>() {
            @Override
            public void changed(ObservableValue<? extends Number> ov,
               Number oldValue, Number newValue) {
               tipPercentage =
                  BigDecimal.valueOf(newValue.intValue() / 100.0);
               tipPercentageLabel.setText(percent.format(tipPercentage));
            }
         }
      );
   }
}
```

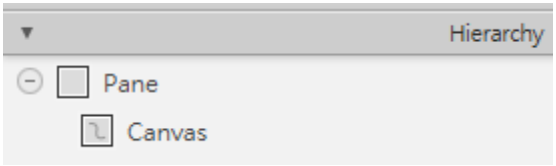
2. Canvas Shape



GUI Description:



Hierarchy:



0) File Name: CanvasShapes.fxml
Controller Class: CanvasShapesController

1) Pane
   a) Pref Width: Reset to default
   b) Pref Height: Reset to default
   c)

2) Canvas
   a) fx:id :drawingCanvas
   b) Width: 650

c) Height:115

CanvasShapes.java

```
// CanvasShapes.java
import javafx.application.Application;
import javafx.fxml.FXMLLoader;
import javafx.scene.Parent;
import javafx.scene.Scene;
import javafx.stage.Stage;

public class CanvasShapes extends Application {
   @Override
   public void start(Stage stage) throws Exception {
      Parent root =
         FXMLLoader.load(getClass().getResource("CanvasShapes.fxml"));

      Scene scene = new Scene(root);
      stage.setTitle("Draw shapes with Canvas");
      stage.setScene(scene);
      stage.show();
   }

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

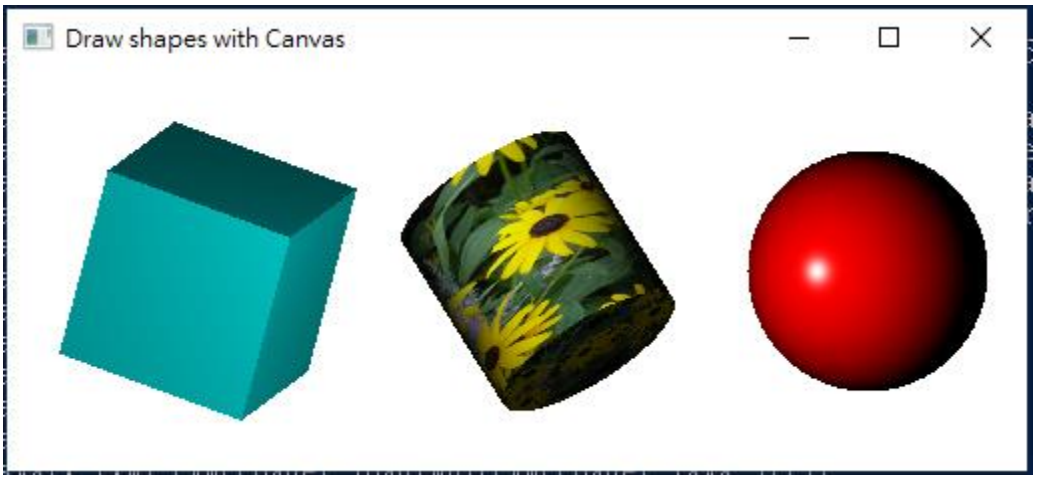
CanvasShapesController.java

```
// Fig. 22.14: CanvasShapesController.java
// Drawing on a Canvas.
import javafx.fxml.FXML;
import javafx.scene.canvas.Canvas;
import javafx.scene.canvas.GraphicsContext;
import javafx.scene.image.Image;
import javafx.scene.paint.Color;
import javafx.scene.paint.CycleMethod;
import javafx.scene.paint.ImagePattern;
import javafx.scene.paint.LinearGradient;
```

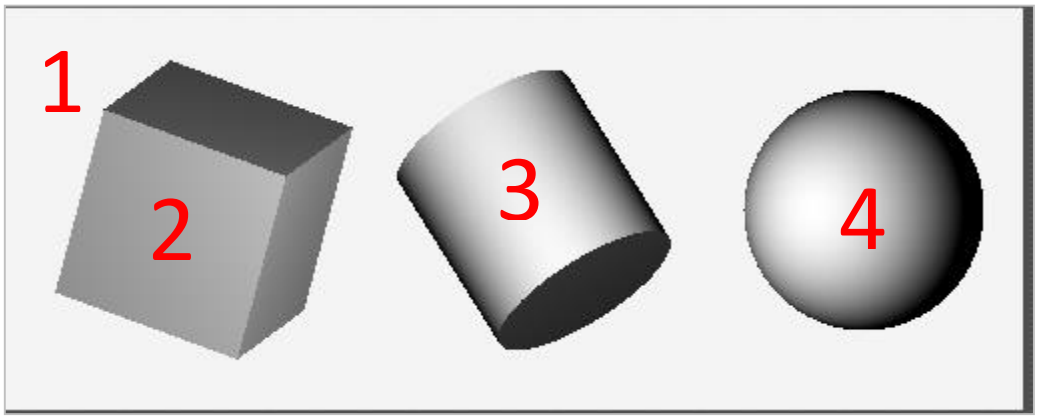
```
import javafx.scene.paint.RadialGradient;
import javafx.scene.paint.Stop;
import javafx.scene.shape.ArcType;
import javafx.scene.shape.StrokeLineCap;

public class CanvasShapesController {
   // instance variables that refer to GUI components
   @FXML private Canvas drawingCanvas;

   // draw on the Canvas
   public void initialize() {
      GraphicsContext gc = drawingCanvas.getGraphicsContext2D();
      gc.setLineWidth(10); // set all stroke widths

      // draw red line
      gc.setStroke(Color.RED);
      gc.strokeLine(10, 10, 100, 100);

      // draw green line
      gc.setGlobalAlpha(0.5); // half transparent
      gc.setLineCap(StrokeLineCap.ROUND);
      gc.setStroke(Color.GREEN);
      gc.strokeLine(100, 10, 10, 100);

      gc.setGlobalAlpha(1.0); // reset alpha transparency

      // draw rounded rect with red border and yellow fill
      gc.setStroke(Color.RED);
      gc.setFill(Color.YELLOW);
      gc.fillRoundRect(120, 10, 90, 90, 50, 50);
      gc.strokeRoundRect(120, 10, 90, 90, 50, 50);

      // draw circle with blue border and red/white radial gradient fill
      gc.setStroke(Color.BLUE);
      Stop[] stopsRadial =
         {new Stop(0, Color.RED), new Stop(1, Color.WHITE)};
      RadialGradient radialGradient = new RadialGradient(0, 0, 0.5, 0.5,
         0.6, true, CycleMethod.NO_CYCLE, stopsRadial);
      gc.setFill(radialGradient);
      gc.fillOval(230, 10, 90, 90);
```

```
        gc.strokeOval(230, 10, 90, 90);

        // draw ellipse with green border and image fill
        gc.setStroke(Color.GREEN);
        gc.setFill(new ImagePattern(new Image("yellowflowers.png")));
        gc.fillOval(340, 10, 200, 90);
        gc.strokeOval(340, 10, 200, 90);

        // draw arc with purple border and cyan/white linear gradient fill
        gc.setStroke(Color.PURPLE);
        Stop[] stopsLinear =
           {new Stop(0, Color.CYAN), new Stop(1, Color.WHITE)};
        LinearGradient linearGradient = new LinearGradient(0, 0, 1, 0,
           true, CycleMethod.NO_CYCLE, stopsLinear);
        gc.setFill(linearGradient);
        gc.fillArc(560, 10, 90, 90, 45, 270, ArcType.ROUND);
        gc.strokeArc(560, 10, 90, 90, 45, 270, ArcType.ROUND);
    }
}
```

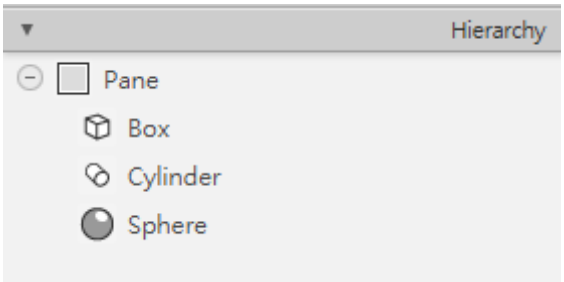
3. ThreeDimensionalShapes



GUI Description:



Hierarchy:



0) File Name: ThreeDimensionalShapes.fxml

Controller Class: ThreeDimensionalShapesController

1) **Pane**
   a) Pref Width:510
   b) Pref Height:200

2) **Box**
   a) fx:id :box
   b) Width: 100
   c) Height: 100
   d) Depth: 100
   e) Rotate: 30
   f) Rotation axis:1 1 1
   g) LayoutX :100
   h) LayoutY:100

3) **Cylinder**
   a) fx:id : cylinder
   b) Height: 100
   c) Radius: 50
   d) Rotate: -45
   e) Rotation axis: 1 1 1
   f) LayoutX : 265
   g) LayoutY: 100

4) **Sphere**
   a) fx:id : sphere
   b) Radius: 60
   c) Rotate:0
   d) Rotation axis : 0 0 1
   e) LayoutX : 430
   f) LayoutY: 100

ThreeDimensionalShapes.java

```
// ThreeDimensionalShapes.java
import javafx.application.Application;
import javafx.fxml.FXMLLoader;
import javafx.scene.Parent;
```

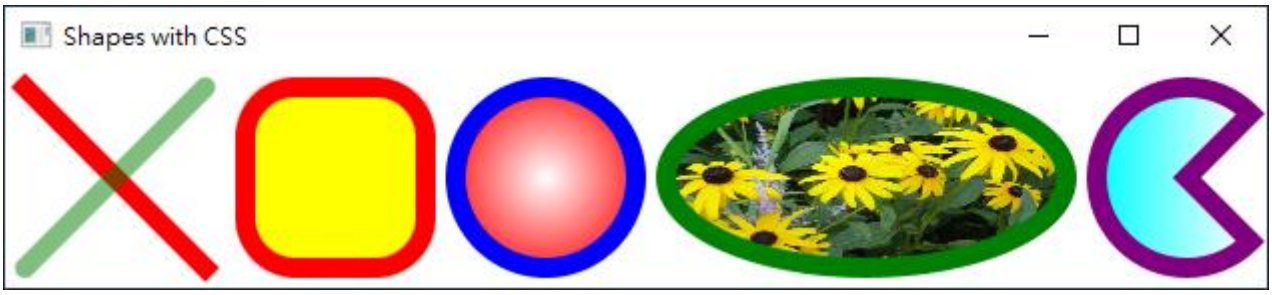
```
import javafx.scene.Scene;
import javafx.stage.Stage;

public class ThreeDimensionalShapes extends Application {
   @Override
   public void start(Stage stage) throws Exception {
      Parent root =

FXMLLoader.load(getClass().getResource("ThreeDimensionalShapes.fxml"));

      Scene scene = new Scene(root);
      stage.setTitle("Draw shapes with Canvas");
      stage.setScene(scene);
      stage.show();
   }

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

ThreeDimensionalShapesController.java

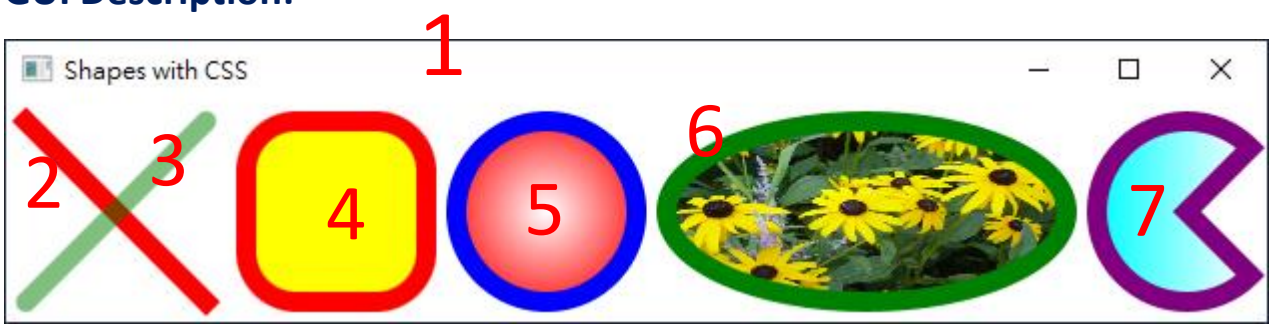
```
// Fig. 22.15: ThreeDimensionalShapesController.java
// Setting the material displayed on 3D shapes.
import javafx.fxml.FXML;
import javafx.scene.paint.Color;
import javafx.scene.paint.PhongMaterial;
import javafx.scene.image.Image;
import javafx.scene.shape.Box;
import javafx.scene.shape.Cylinder;
import javafx.scene.shape.Sphere;

public class ThreeDimensionalShapesController {
   // instance variables that refer to 3D shapes
   @FXML private Box box;
   @FXML private Cylinder cylinder;
   @FXML private Sphere sphere;
```

```
   // set the material for each 3D shape
   public void initialize() {
      // define material for the Box object
      PhongMaterial boxMaterial = new PhongMaterial();
      boxMaterial.setDiffuseColor(Color.CYAN);
      box.setMaterial(boxMaterial);

      // define material for the Cylinder object
      PhongMaterial cylinderMaterial = new PhongMaterial();
      cylinderMaterial.setDiffuseMap(new Image("yellowflowers.png"));
      cylinder.setMaterial(cylinderMaterial);

      // define material for the Sphere object
      PhongMaterial sphereMaterial = new PhongMaterial();
      sphereMaterial.setDiffuseColor(Color.RED);
      sphereMaterial.setSpecularColor(Color.WHITE);
      sphereMaterial.setSpecularPower(32);
      sphere.setMaterial(sphereMaterial);
   }
}
```

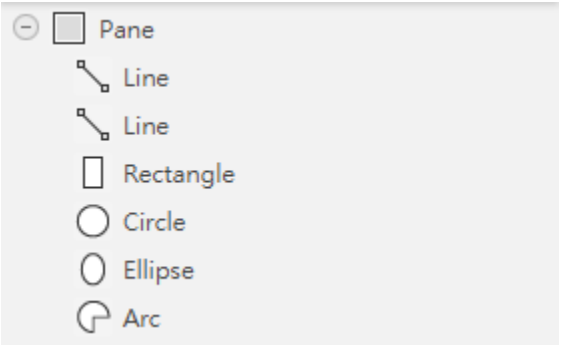
4. Basic Shapes



GUI Description:



Hierarchy:



0) File Name: BasicShapes.fxml
Controller Class: none

1) Pane

a) Stylesheets: BasicShapes.css
b) id: Pane
c) Min Width: USE_COMPUTED_SIZE
d) Min Height: USE_COMPUTED_SIZE

e) Pref Width: 630
f) Pref Height: 110
g) Max Width: USE_COMPUTED_SIZE
h) Max Height: USE_COMPUTED_SIZE

2) Line

a) id: line1
b) Layout X: 0
c) Layout Y: 0
d) Start X: 10
e) Start Y: 10
f) End X: 100
g) End Y: 100
h) fx:id: line2

3) Line

a) Layout X: 0
b) Layout Y: 0
c) Start X: 100
d) Start Y: 10
e) End X: 10
f) End Y: 100
g) fx:id: line2

4) Rectangle

a) Arc Width: 0
b) Fill: Black
c) Arc Height: 0
d) Stroke: Reset to Default
e) Strike Type: CENTERED
f) Width: 90
g) Height: 90
h) Layout X: 120
i) Layout Y: 10
j) fx:id: rectangle

5) Circle

a) Fill: Black
b) Stroke: Reset to Default
c) Strike Type: CENTERED

d) Radius: 45
e) Center X: 270
f) Center Y: 55
g) fx:id: circle

6) Ellipse

a) Fill: Black
b) Stroke: Reset to Default
c) Strike Type: CENTERED
d) Radius X: 100
e) Radius Y: 45
f) Center X: 430
g) Center Y: 55
h) fx:id: ellipse

7) Arc

a) Fill: Black
b) Stroke: Reset to Default
c) Strike Type: CENTERED
d) Radius X: 45
e) Radius Y: 45
f) Start Angle: 45
g) Length: 270
h) Center X: 590
i) Center Y: 55
j) fx:id: arc

BasicShapes.css

```
/* BasicShapes.css */
/* CSS that styles various two-dimensional shapes */

Line, Rectangle, Circle, Ellipse, Arc {
    -fx-stroke-width: 10;
}

#line1 {
    -fx-stroke: red;
}
```

```
#line2 {
    -fx-stroke: rgba(0%, 50%, 0%, 0.5);
    -fx-stroke-line-cap: round;
}

Rectangle {
   -fx-stroke: red;
   -fx-arc-width: 50;
   -fx-arc-height: 50;
   -fx-fill: yellow;
}

Circle {
   -fx-stroke: blue;
   -fx-fill: radial-gradient(center 50% 50%, radius 60%, white, red);
}

Ellipse {
   -fx-stroke: green;
   -fx-fill: image-pattern("yellowflowers.png");
}

Arc {
   -fx-stroke: purple;
   -fx-fill: linear-gradient(to right, cyan, white);
}
```

BasicShapes.java

```
// BasicShapes.java
import javafx.application.Application;
import javafx.fxml.FXMLLoader;
import javafx.scene.Parent;
import javafx.scene.Scene;
import javafx.stage.Stage;

public class BasicShapes extends Application {
   @Override
```

```
   public void start(Stage stage) throws Exception {
      Parent root =
         FXMLLoader.load(getClass().getResource("BasicShapes.fxml"));

      Scene scene = new Scene(root);
      stage.setTitle("Shapes with CSS");
      stage.setScene(scene);
      stage.show();
   }

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