# Ch - 7
# JAVAFX basics and Event-driven programming and animations

Prepared By,

Hruta Desai

Button:

Cancel

Disabled

Boxy

Cancel

Toggle Button:

First

Second

Third

Disable

Long ..... really Long

Hyperlink:

Hello I am a hyperlink

I can have an icon too

CheckBox:
Normal/Undefined/Selected

Hit me dude

Are you sure

There you go

Radio Button:

Bye

Hello

Menu Buttons:

Menu Button

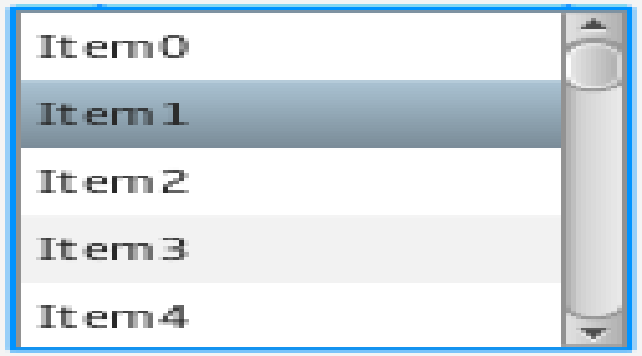
Menu Button

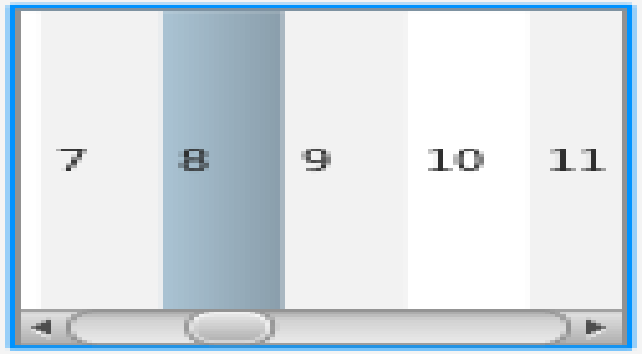
ScrollView:

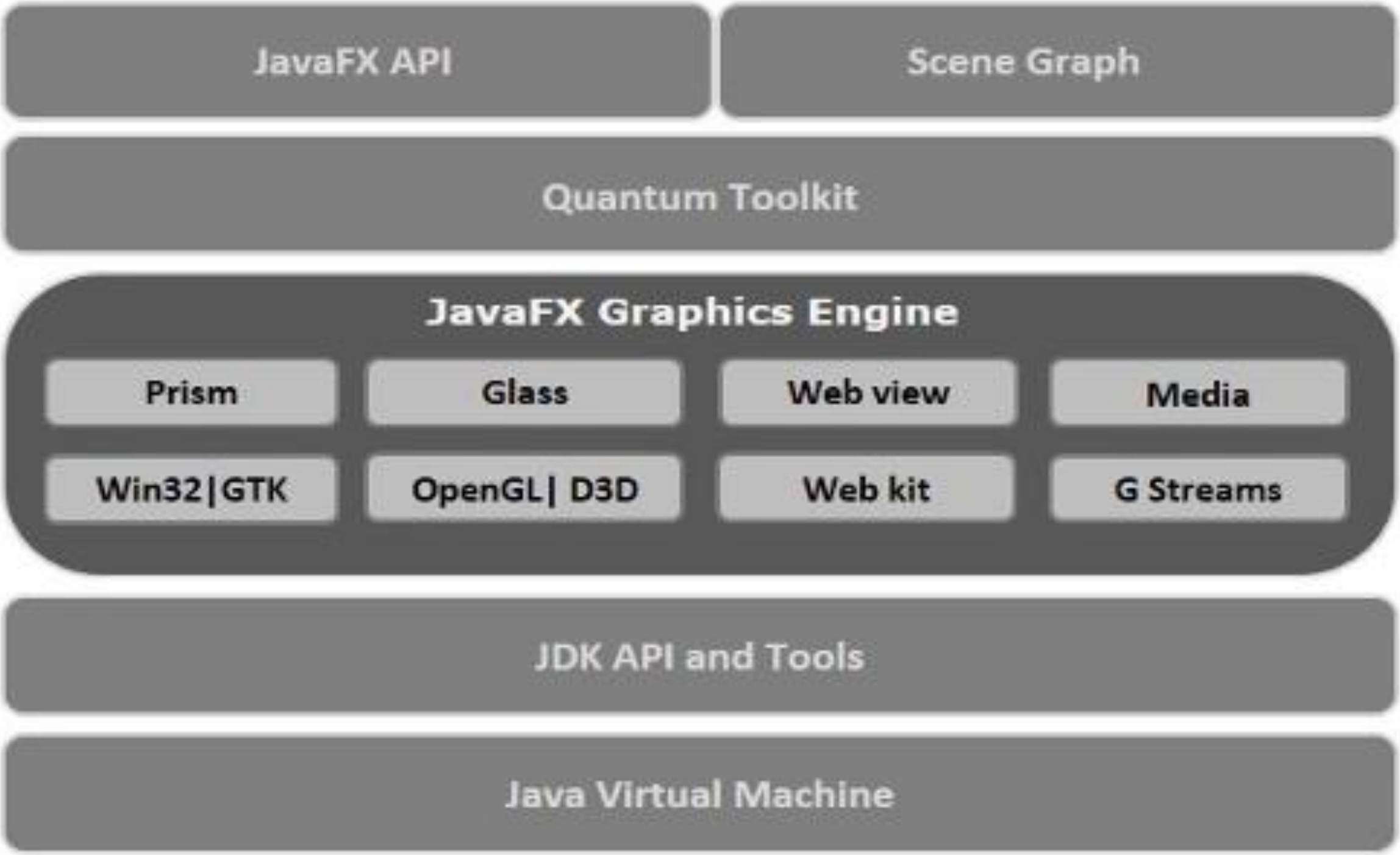

ProgressBar:

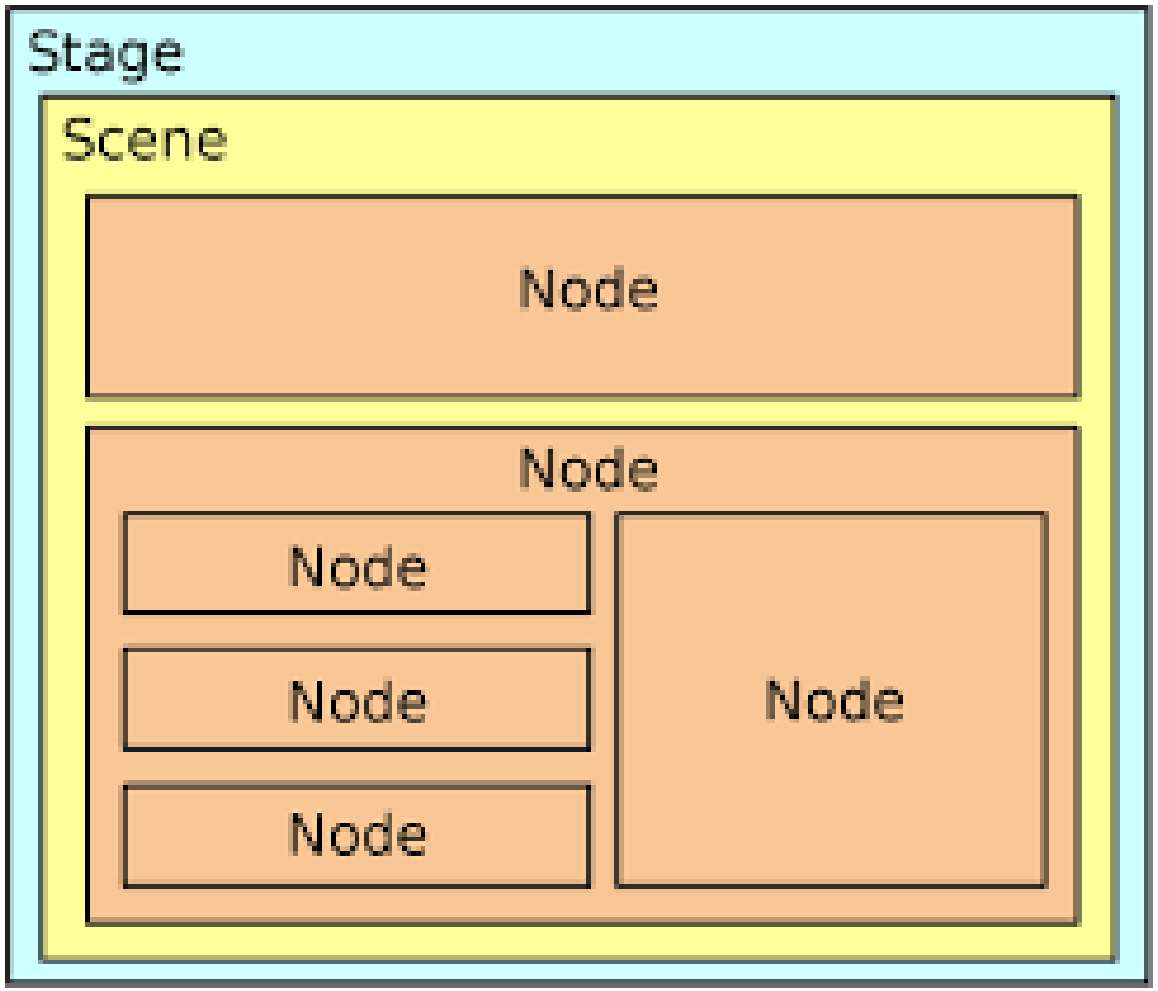
ProgressIndicator
Indeterminate:

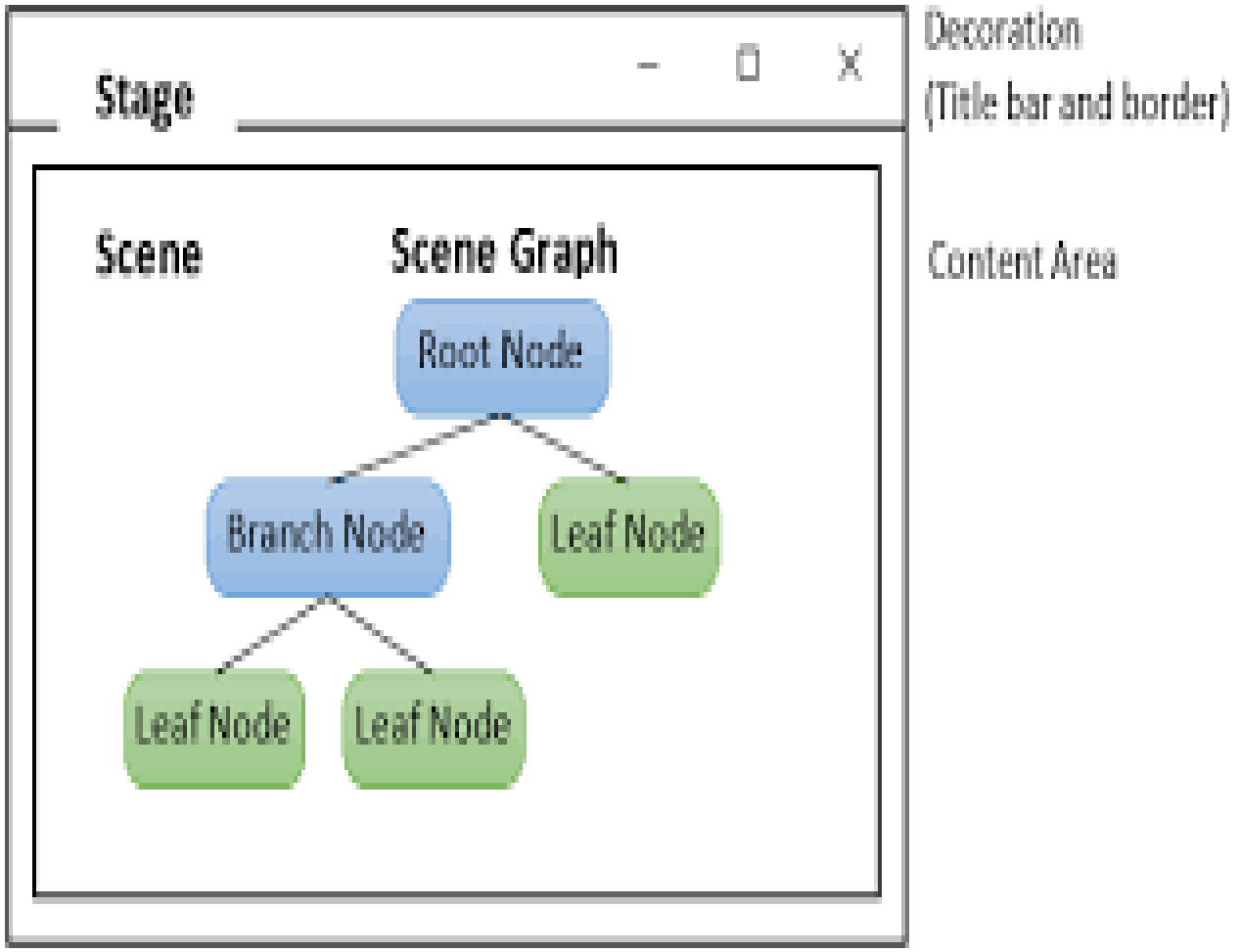
ListView:

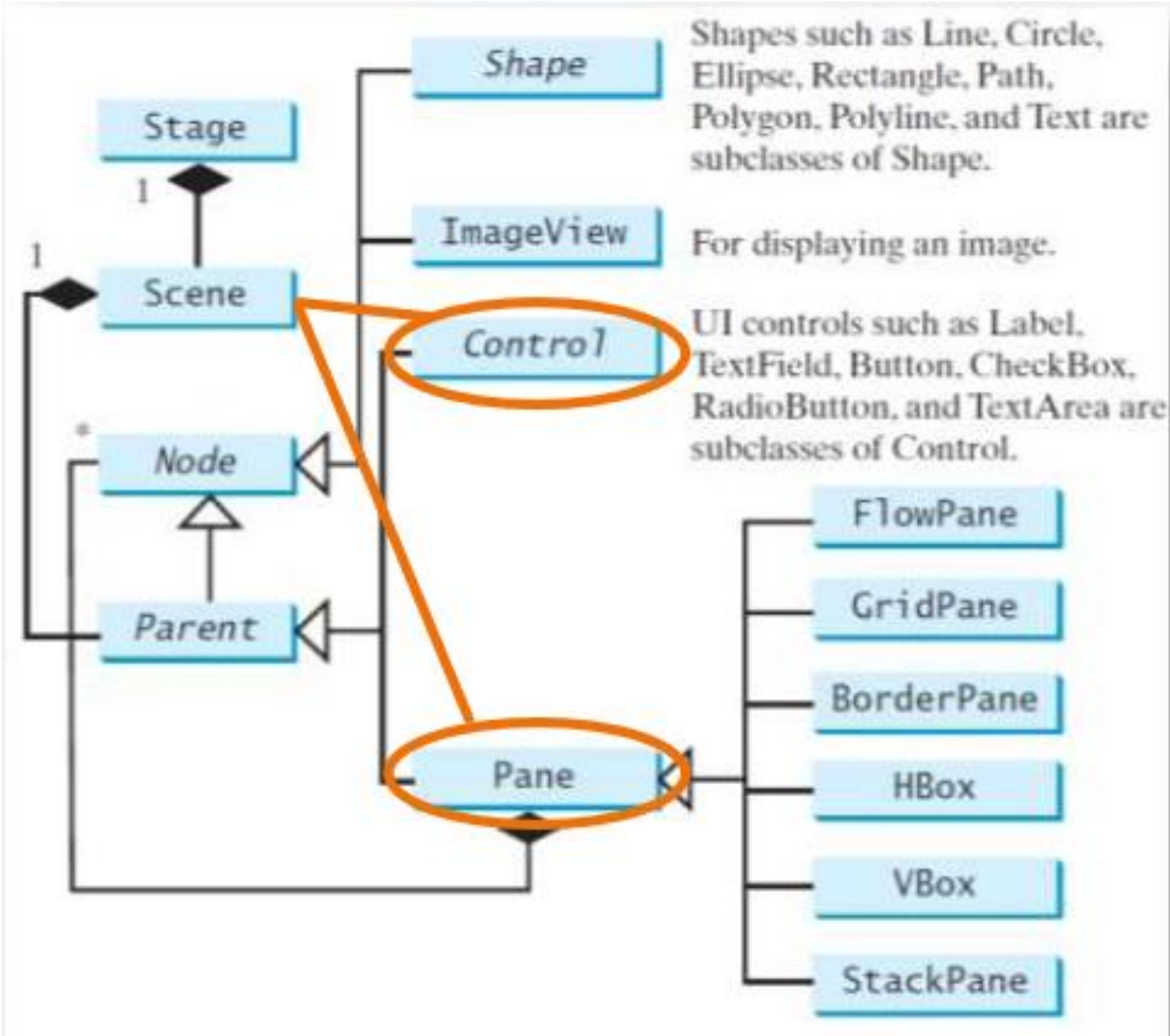


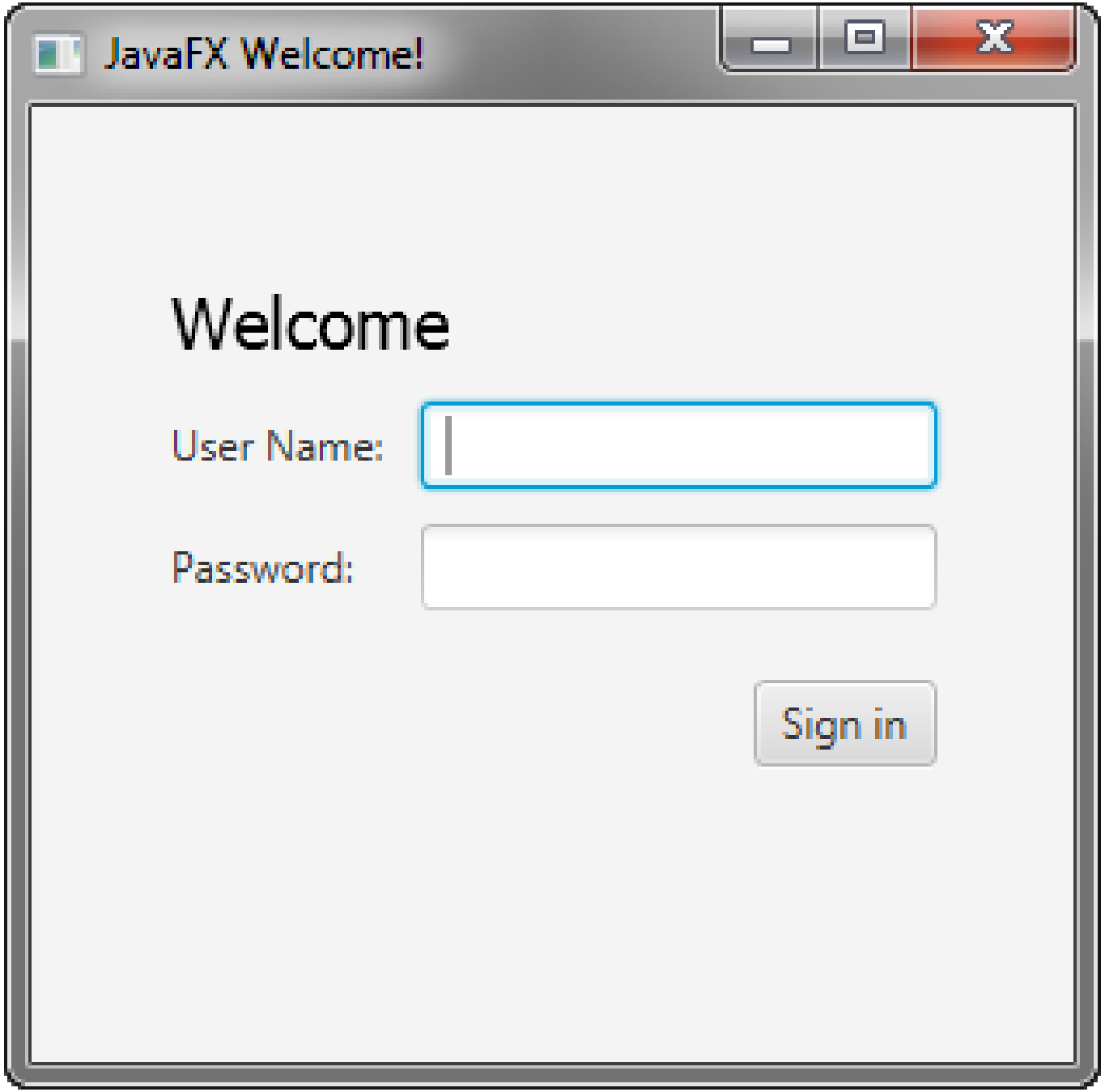




JavaFX API
Scene Graph
Quantum Toolkit
JavaFX Graphics Engine
Prism
Glass
Web view
Media
Win32|GTK
OpenGL| D3D
Web kit
G Streams
JDK API and Tools
Java Virtual Machine

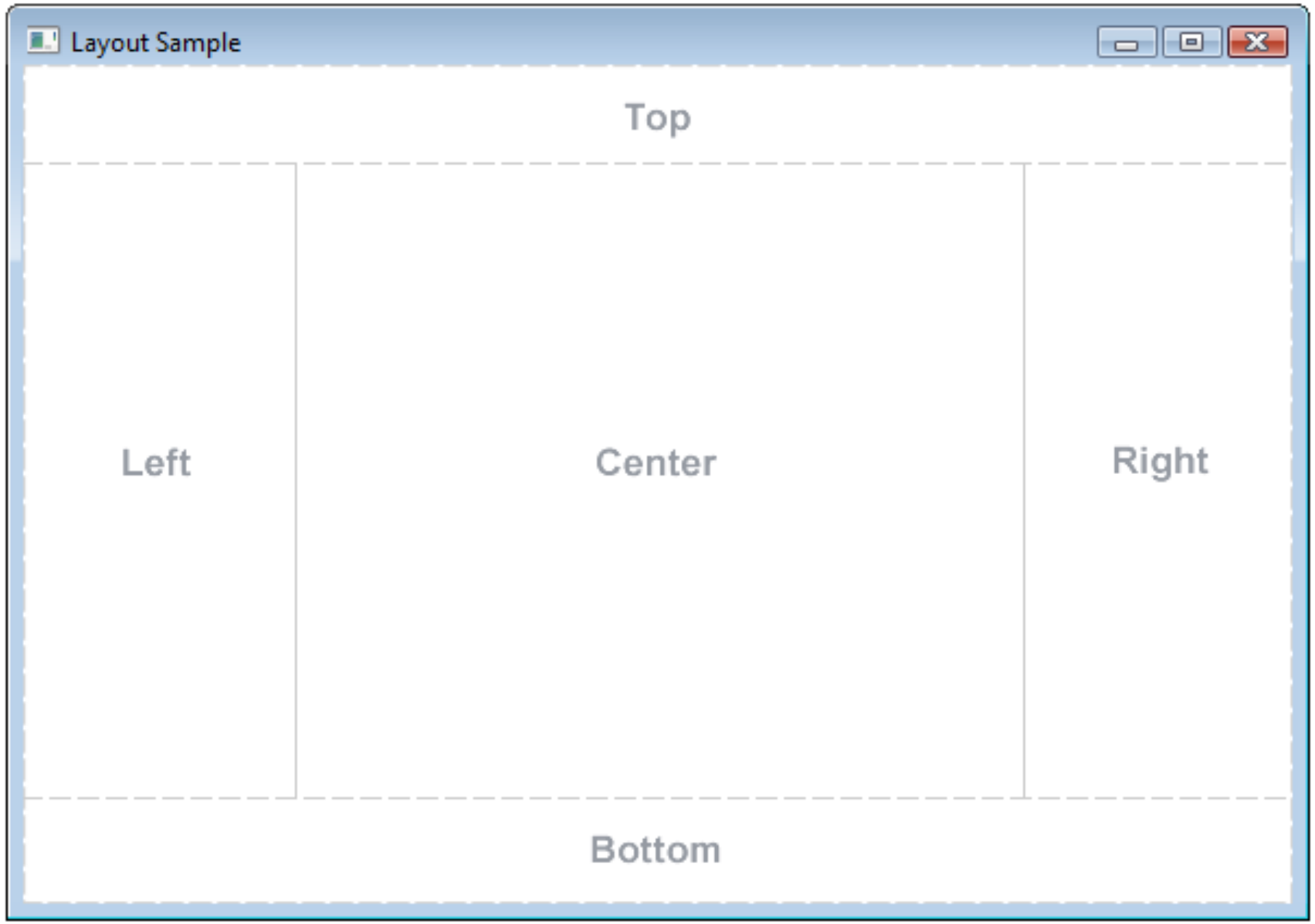

Stage
Scene
Node
Node
Node
Node
Node


Stage
Scene
Scene Graph
Root Node
Branch Node
Leaf Node
Leaf Node
Leaf Node
Decoration
(Title bar and border)
Content Area


Shape
Shapes such as Line, Circle, Ellipse, Rectangle, Path, Polygon, Polyline, and Text are subclasses of Shape.
Stage
1
ImageView
For displaying an image.
1
Scene
Control
UI controls such as Label, TextField, Button, CheckBox, RadioButton, and TextArea are subclasses of Control.
*
Node
FlowPane
GridPane
Parent
BorderPane
Pane
HBox
VBox
StackPane

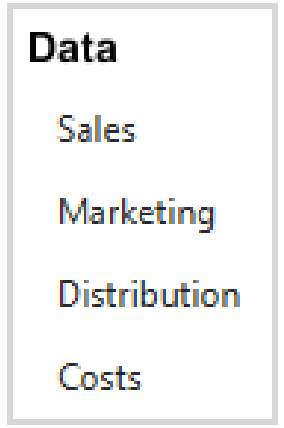

JavaFX Welcome!
Welcome
User Name:
Password:
Sign in

# Border Pane

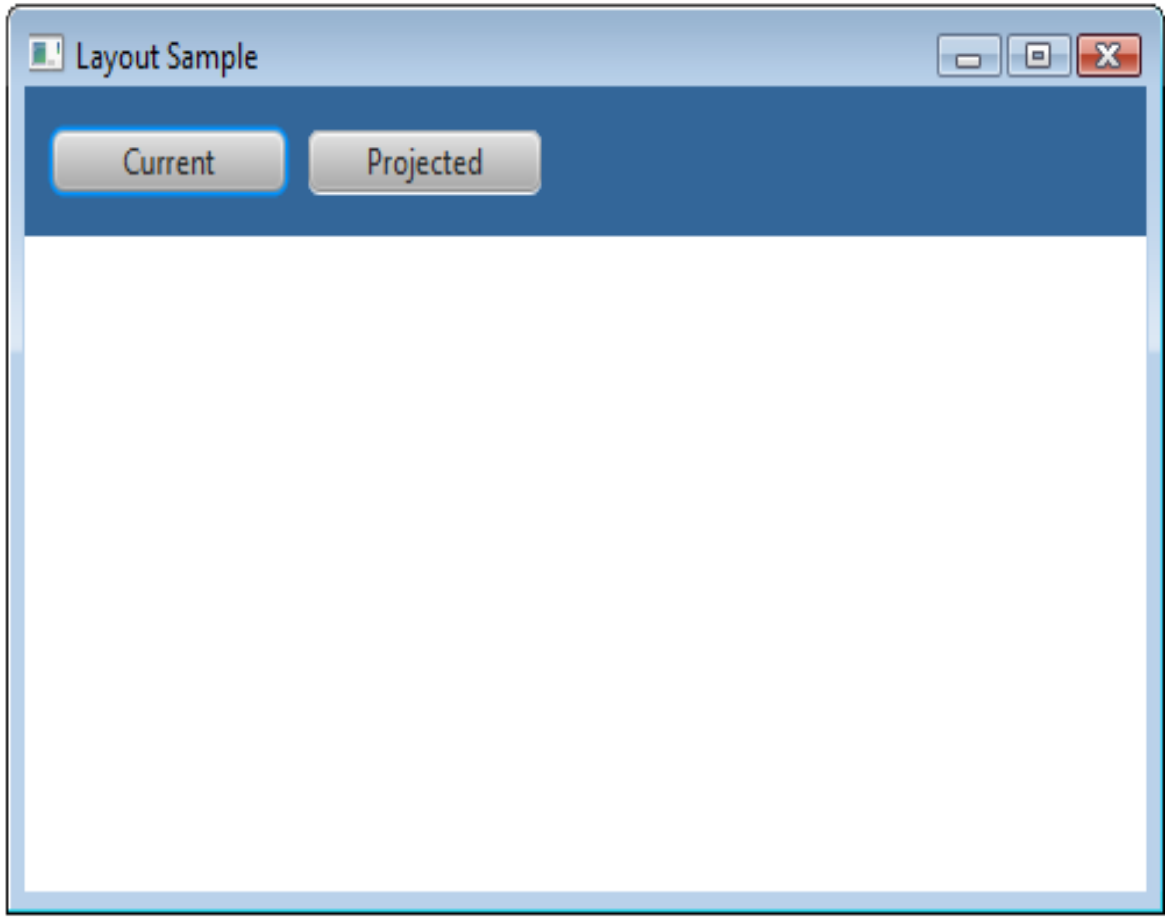


## HBox Pane

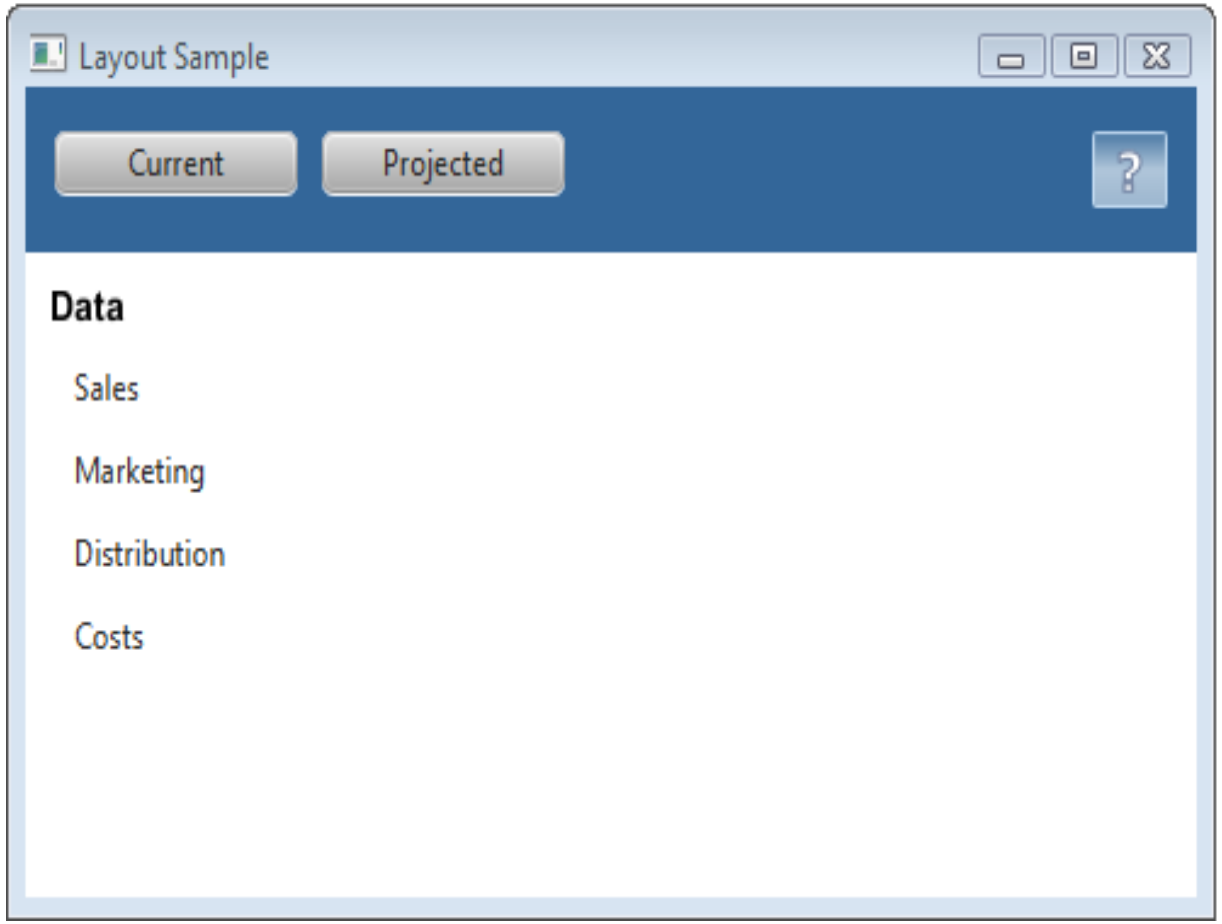

## VBox Pane




## HBox Pane in Border Pane




## VBox Pane in Border Pane

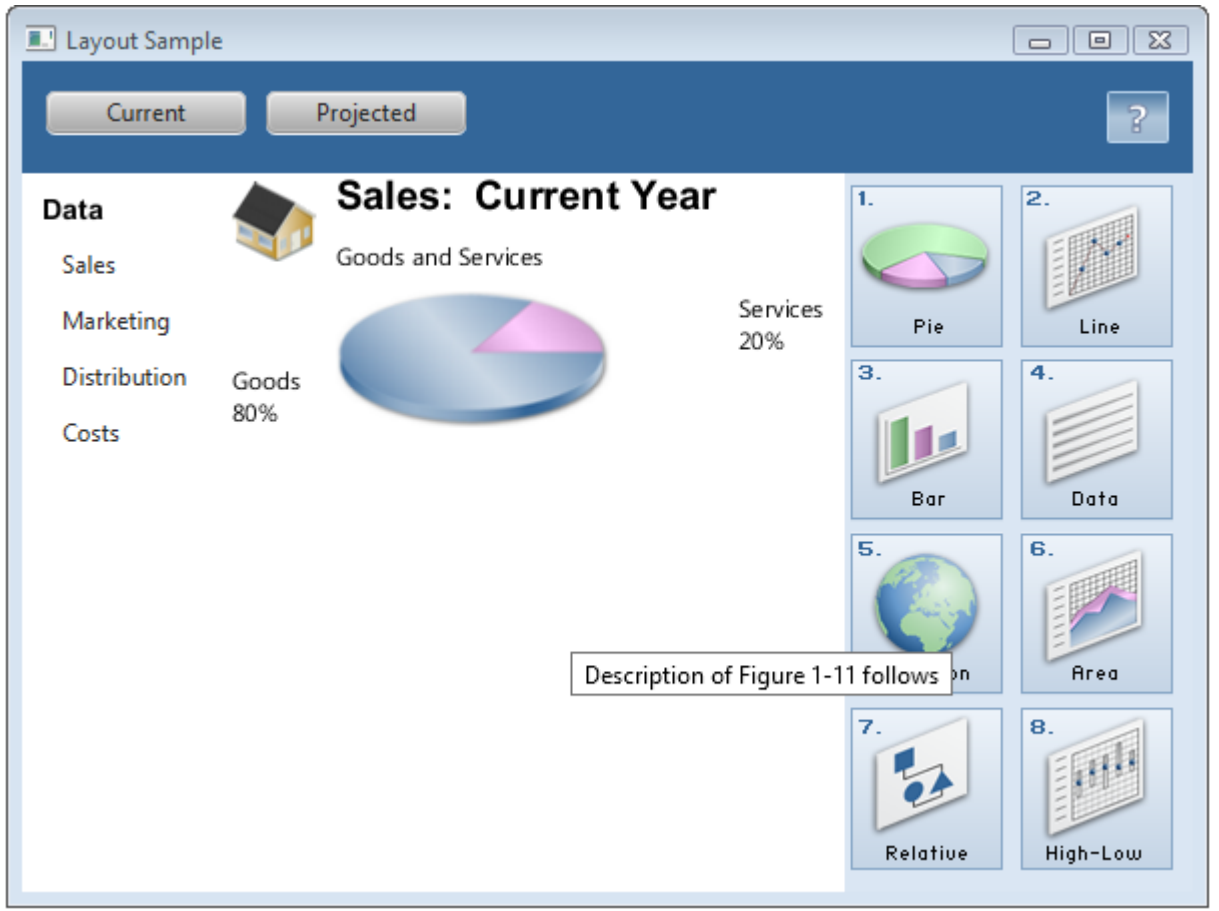


# Stack Pane

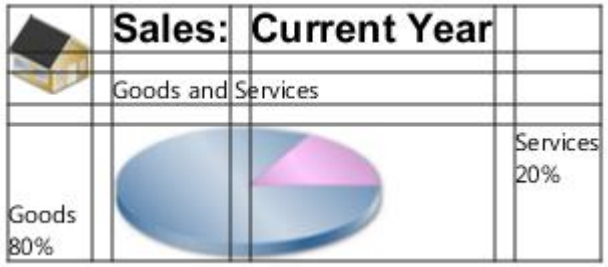
# Stack Pane in HBox Pane


# Horizontal Flow Pane


# Flow Pane in Border Pane

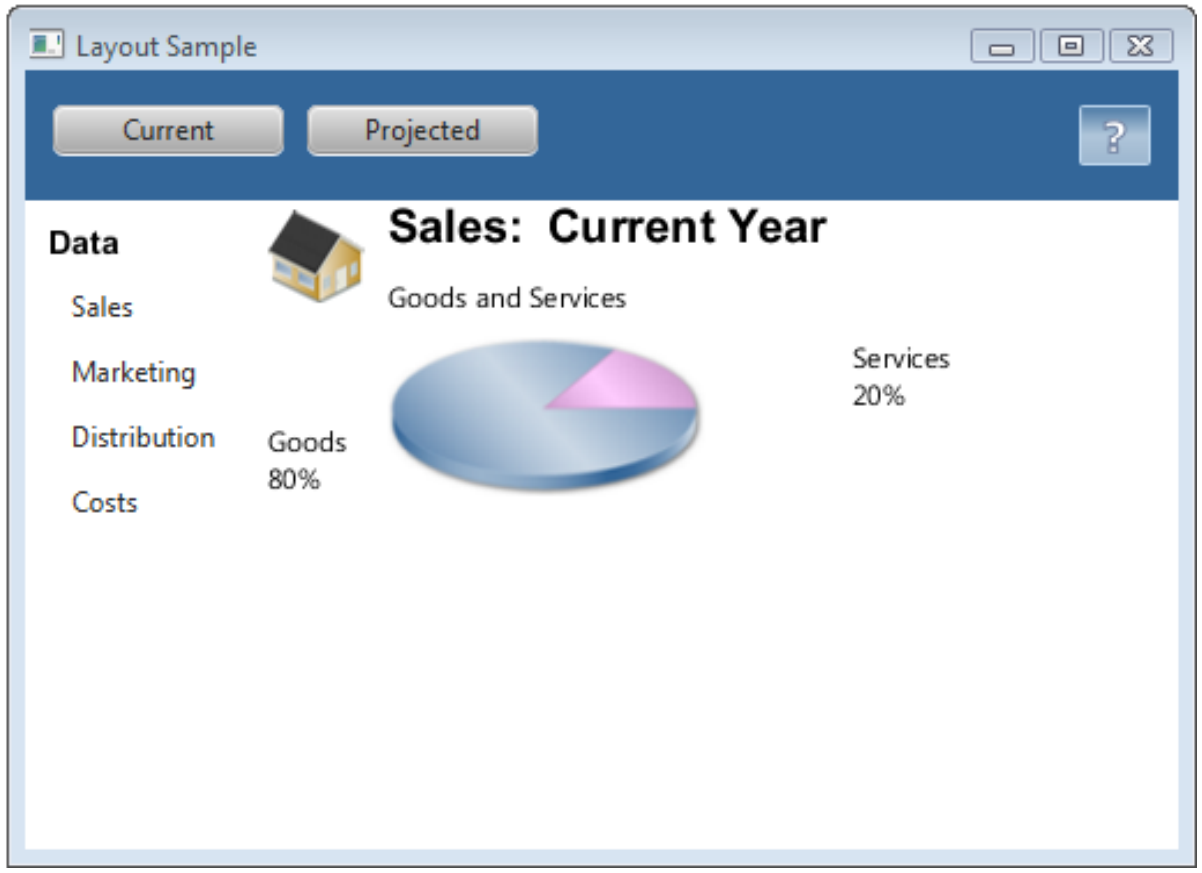


# Grid Pane

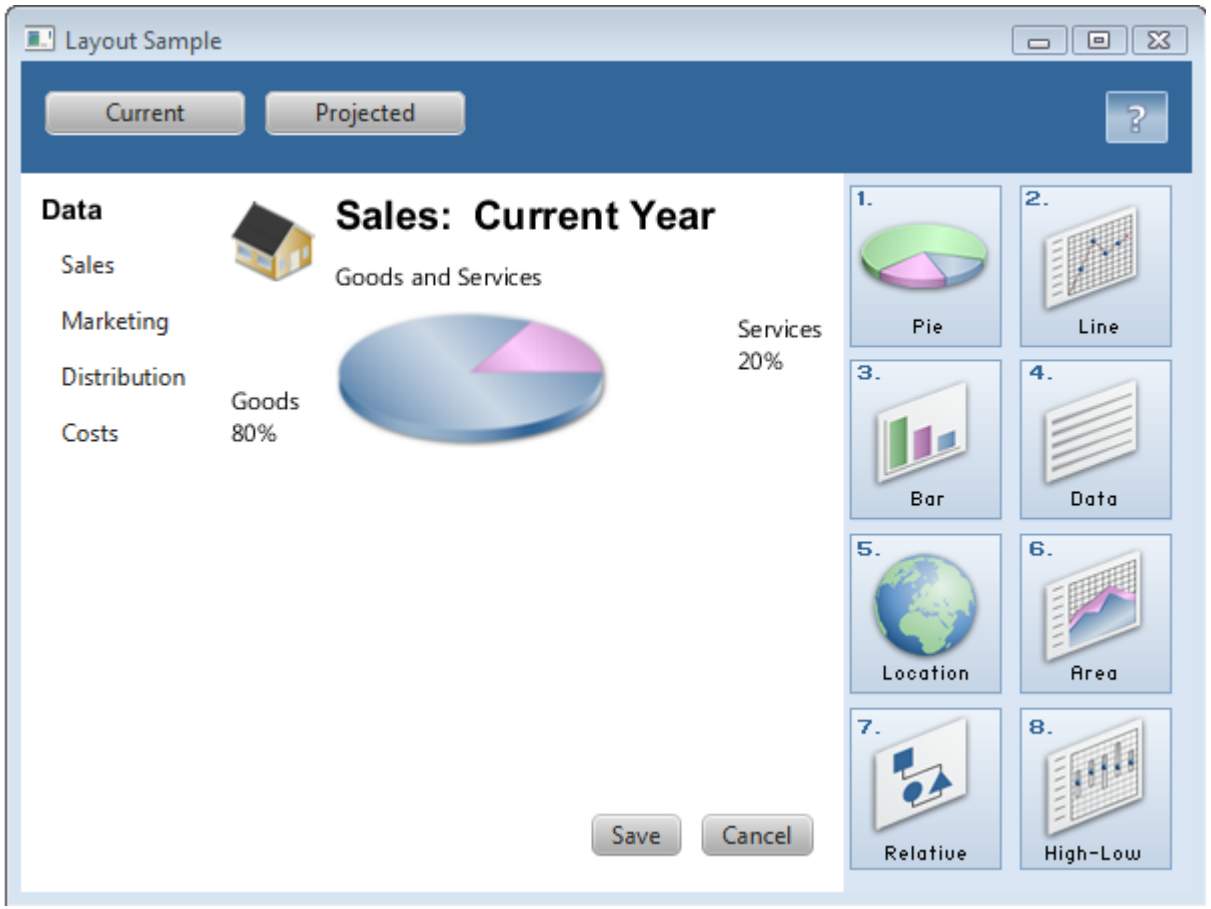



# Anchor Pane

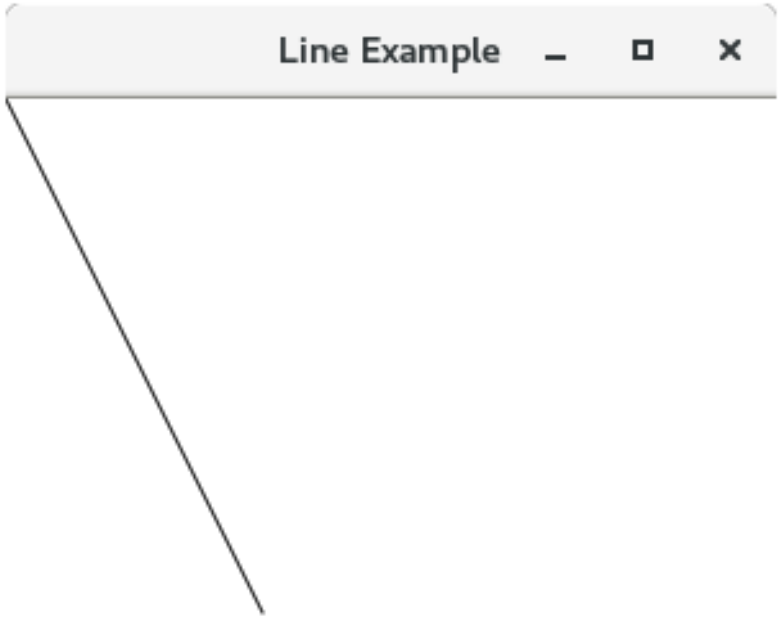

# Grid Pane in Border Pane



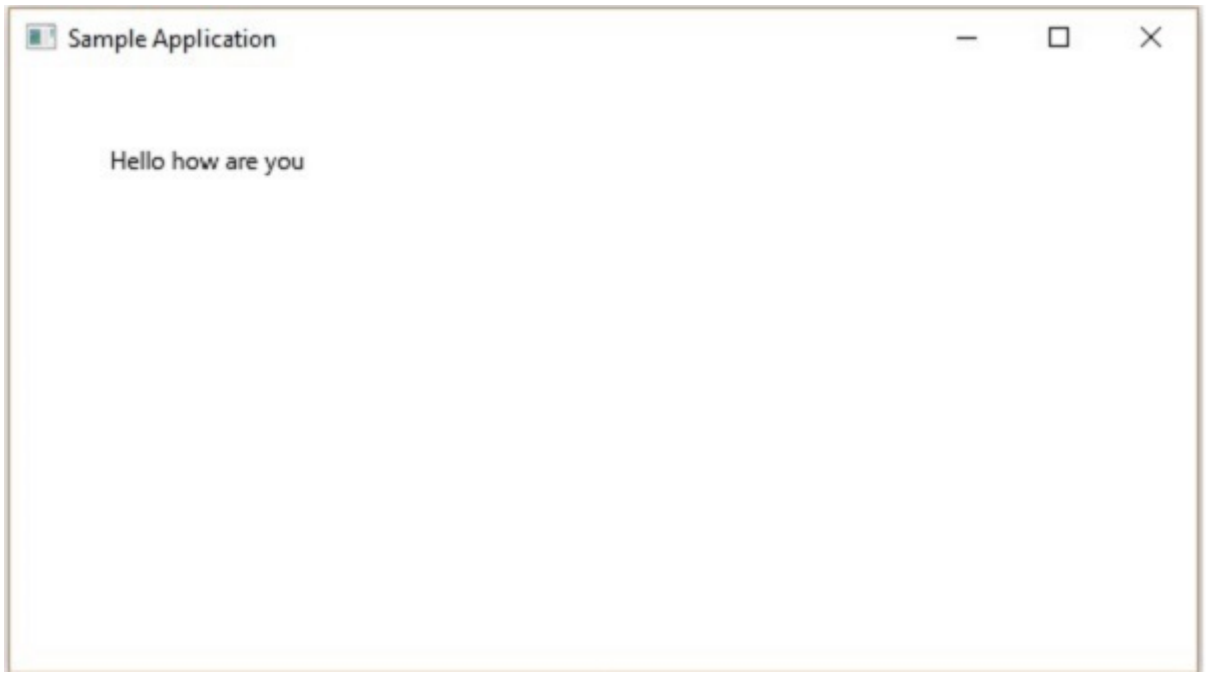

# Anchor Pane Border Pane



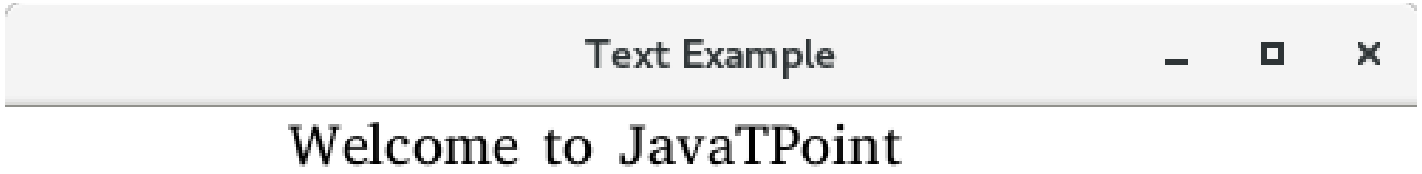
```
1.
2.public class LineDrawingExamples extends Application{
3.
4.    @Override
5.    public void start(Stage primaryStage) throws Exception {
6.        Line line = new Line(); //instantiating Line class
7.        line.setStartX(0); //setting starting X point of Line
8.        line.setStartY(0); //setting starting Y point of Line
9.        line.setEndX(100); //setting ending X point of Line
10.       line.setEndY(200); //setting ending Y point of Line
11.       Group root = new Group(); //Creating a Group
12.       root.getChildren().add(line); //adding the class object //to the group
13.       Scene scene = new Scene(root,300,300);
14.       primaryStage.setScene(scene);
15.       primaryStage.setTitle("Line Example");
16.       primaryStage.show();
17.   }
18.   public static void main(String[] args) {
19.       launch(args);
20.   }
21.}
```



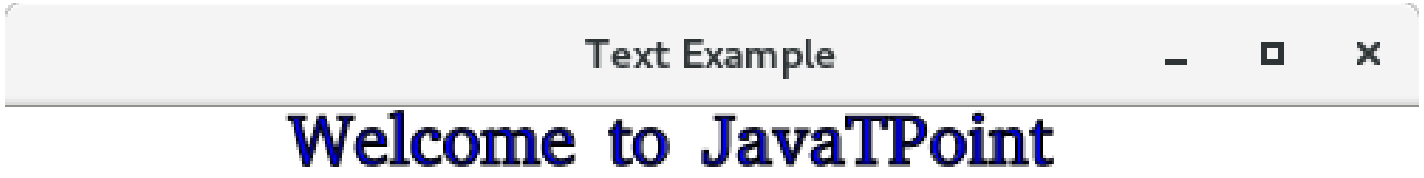
```java
public class TextExample extends Application {
   @Override
   public void start(Stage stage) {
      //Creating a Text object
      Text text = new Text();

      //Setting the text to be added.
      text.setText("Hello how are you");

      //setting the position of the text
      text.setX(50);
      text.setY(50);

      //Creating a Group object
      Group root = new Group(text);

      //Creating a scene object
      Scene scene = new Scene(root, 600, 300);

      //Setting title to the Stage
      stage.setTitle("Sample Application");

      //Adding scene to the stage
      stage.setScene(scene);

      //Displaying the contents of the stage
      stage.show();
   }
   public static void main(String args[]){
      launch(args);
   }
}
```



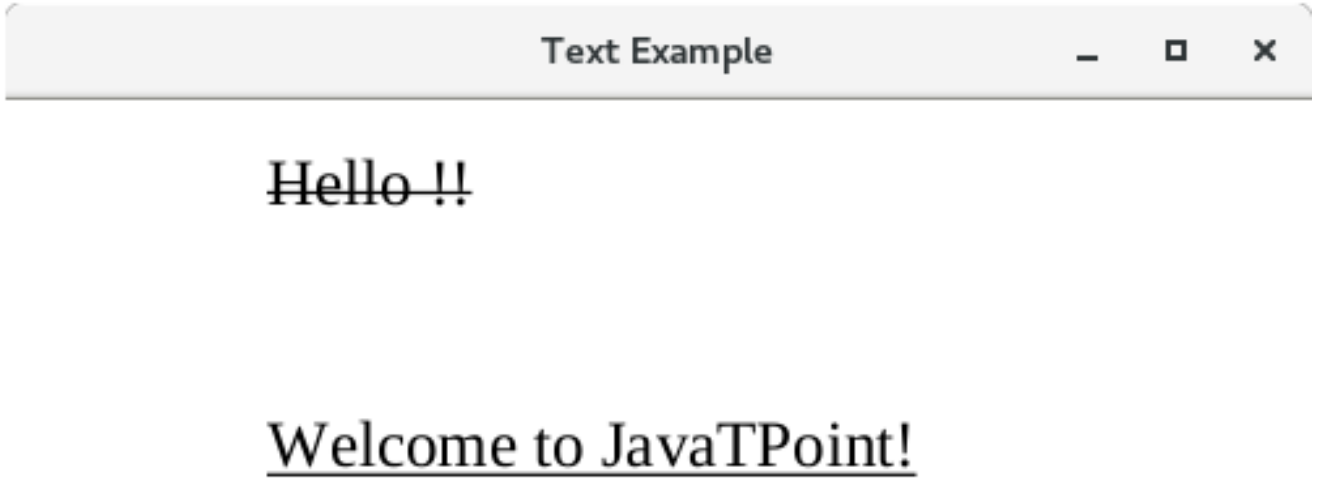
```
public void start(Stage primaryStage) throws Exception {
    // TODO Auto-generated method stub
    Text text = new Text();

    text.setX(100);
    text.setY(20);
    text.setFont(Font.font("Abyssinica SIL",FontWeight.BOLD,FontPosture.REGULAR,20));
    text.setText("Welcome to JavaTPoint");
    Group root = new Group();
    Scene scene = new Scene(root,500,200);
    root.getChildren().add(text);
    primaryStage.setScene(scene);
    primaryStage.setTitle("Text Example");
    primaryStage.show();
}
publicstaticvoid main(String[] args) {
    launch(args);

}
}
```



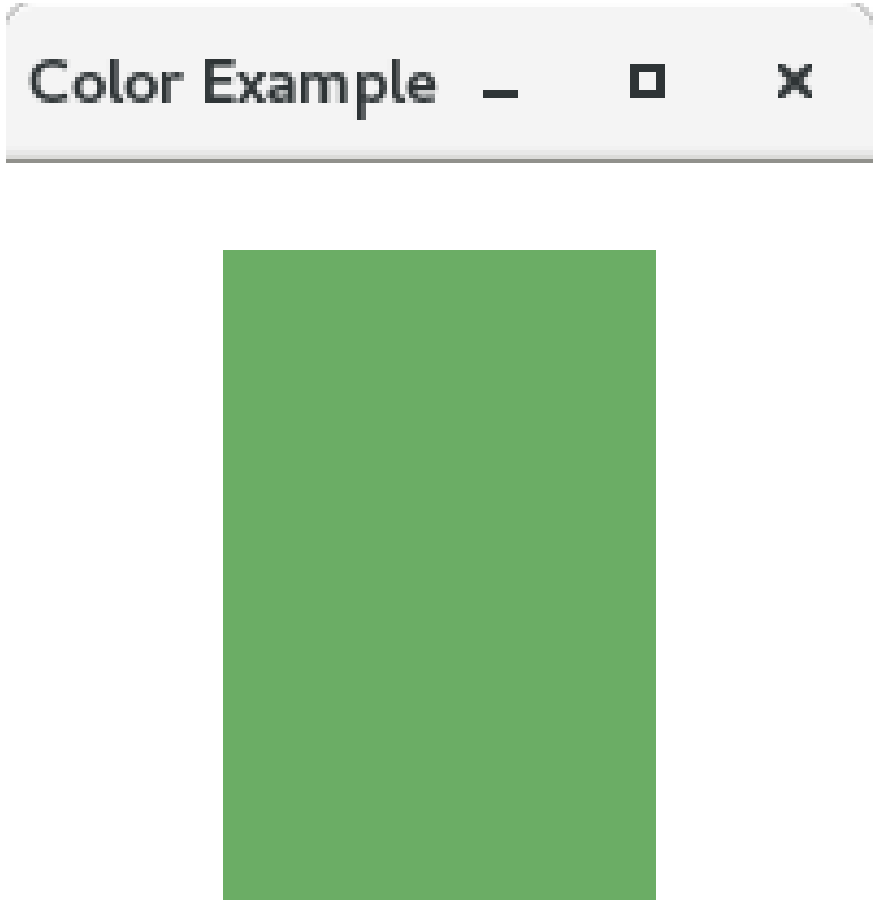
```
public class TextExample extends Application{
@Override
publicvoid start(Stage primaryStage) throws Exception {
   // TODO Auto-generated method stub
   Text text = new Text();

   text.setX(100);
   text.setY(20);
   text.setFont(Font.font("Abyssinica SIL",FontWeight.BOLD,FontPosture.REGULAR,25));
   text.setFill(Color.BLUE);// setting colour of the text to blue
    text.setStroke(Color.BLACK); // setting the stroke for the text
    text.setStrokeWidth(1); // setting stroke width to 2
   text.setText("Welcome to JavaTPoint");
   Group root = new Group();
   Scene scene = new Scene(root,500,200);
   root.getChildren().add(text);
   primaryStage.setScene(scene);
   primaryStage.setTitle("Text Example");
   primaryStage.show();
}
```



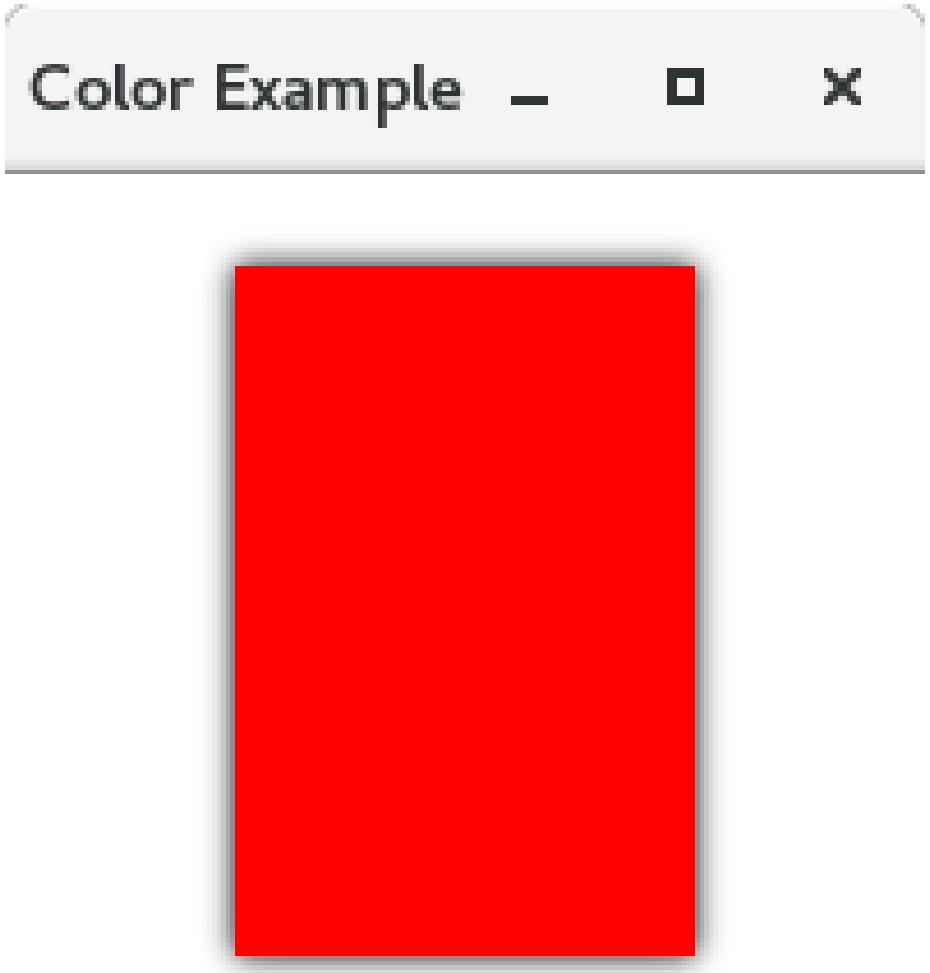
```
public void start(Stage primaryStage) throws Exception {
   // TODO Auto-generated method stub
   Text text = new Text();

   text.setX(100);
   text.setY(40);
   text.setFont(Font.font("Liberation Serif",25));
   text.setText("Hello !!");
   text.setStrikethrough(true);
   Text text1=new Text();
   text1.setX(100);
   text1.setY(140);
   text1.setText("Welcome to JavaTPoint!");
   text1.setFont(Font.font("Liberation Serif",25));
   text1.setUnderline(true);
   Group root = new Group();
   Scene scene = new Scene(root,500,200);
   root.getChildren().addAll(text,text1);
   primaryStage.setScene(scene);
   primaryStage.setTitle("Text Example");
   primaryStage.show();
}
```



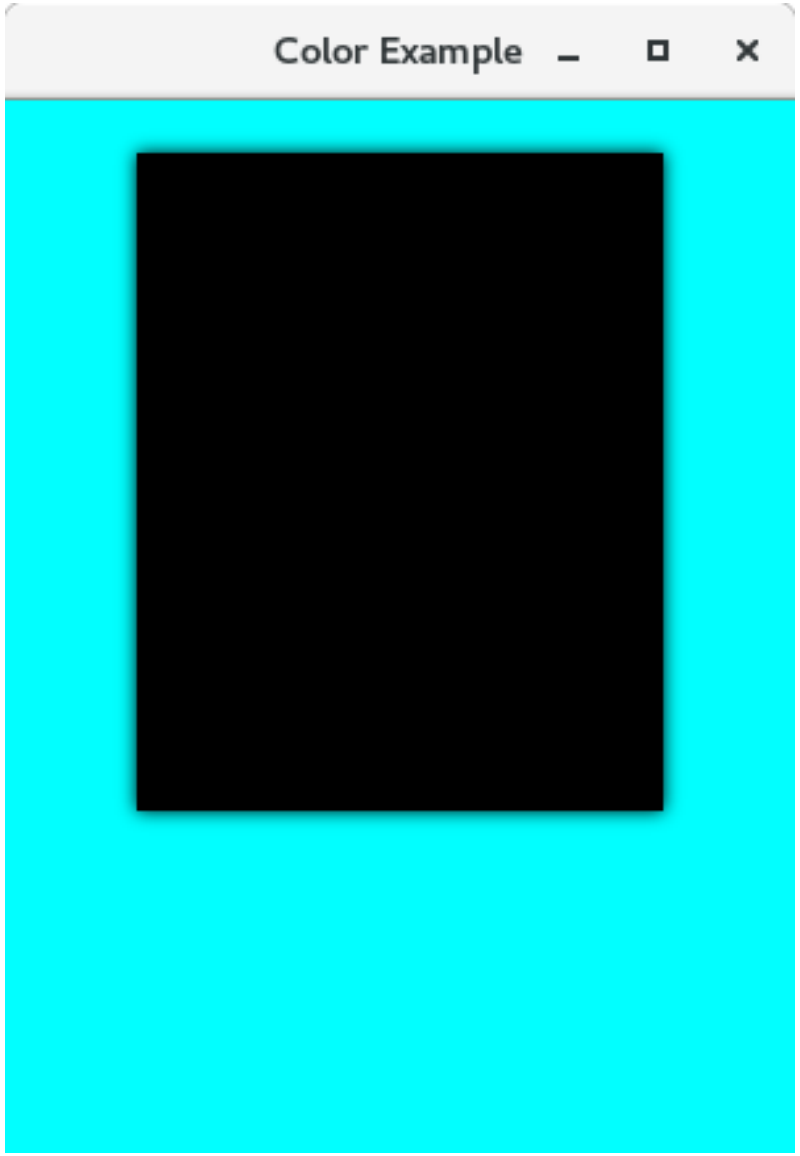
```
public void start(Stage primarystage) {
   Group root = new Group();
   primarystage.setTitle("Color Example");
   Rectangle rect = new Rectangle();
   rect.setX(50);
   rect.setY(20);
   rect.setWidth(100);
   rect.setHeight(150);
   int red=20;
   int green=125;
   int blue=10;
   rect.setFill(Color.rgb(red, green, blue,0.63));
   root.getChildren().add(rect);
   Scene scene = new Scene(root,200,200);
   primarystage.setScene(scene);
   primarystage.show();
 }
```



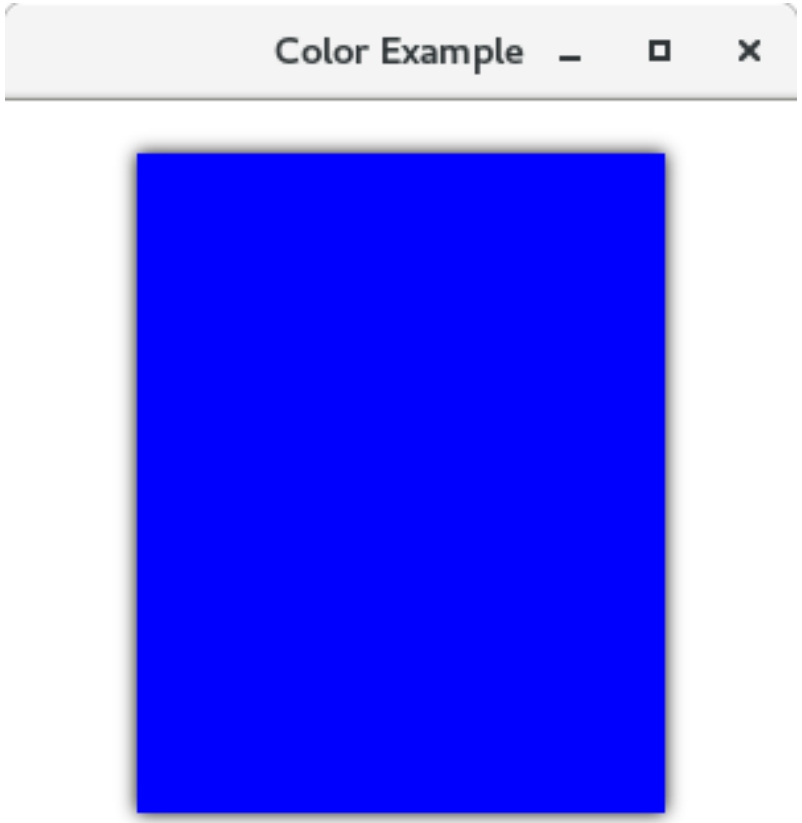
```
public class Shape_Example extends Application {

 @Override
 public void start(Stage primarystage) {
  Group root = new Group();
  primarystage.setTitle("Color Example");
  Rectangle rect = new Rectangle();
  rect.setX(50);
  rect.setY(20);
  rect.setWidth(100);
  rect.setHeight(150);
  rect.setFill(Color.RED); //passing color name
  rect.setEffect(new DropShadow());
  root.getChildren().add(rect);
  Scene scene = new Scene(root,200,200);
  primarystage.setScene(scene);
  primarystage.show();
 }
```



```
public class Shape_Example extends Application {

  @Override
  public void start(Stage primarystage) {
    Group root = new Group();
    primarystage.setTitle("Color Example");
    Rectangle rect = new Rectangle();
    rect.setX(50);
    rect.setY(20);
    rect.setWidth(200);
    rect.setHeight(250);
    rect.setEffect(new DropShadow());
    root.getChildren().add(rect);
    Scene scene = new Scene(root,300,400,Color.hsb(180, 1, 1));
    primarystage.setScene(scene);
    primarystage.show();
  }
```



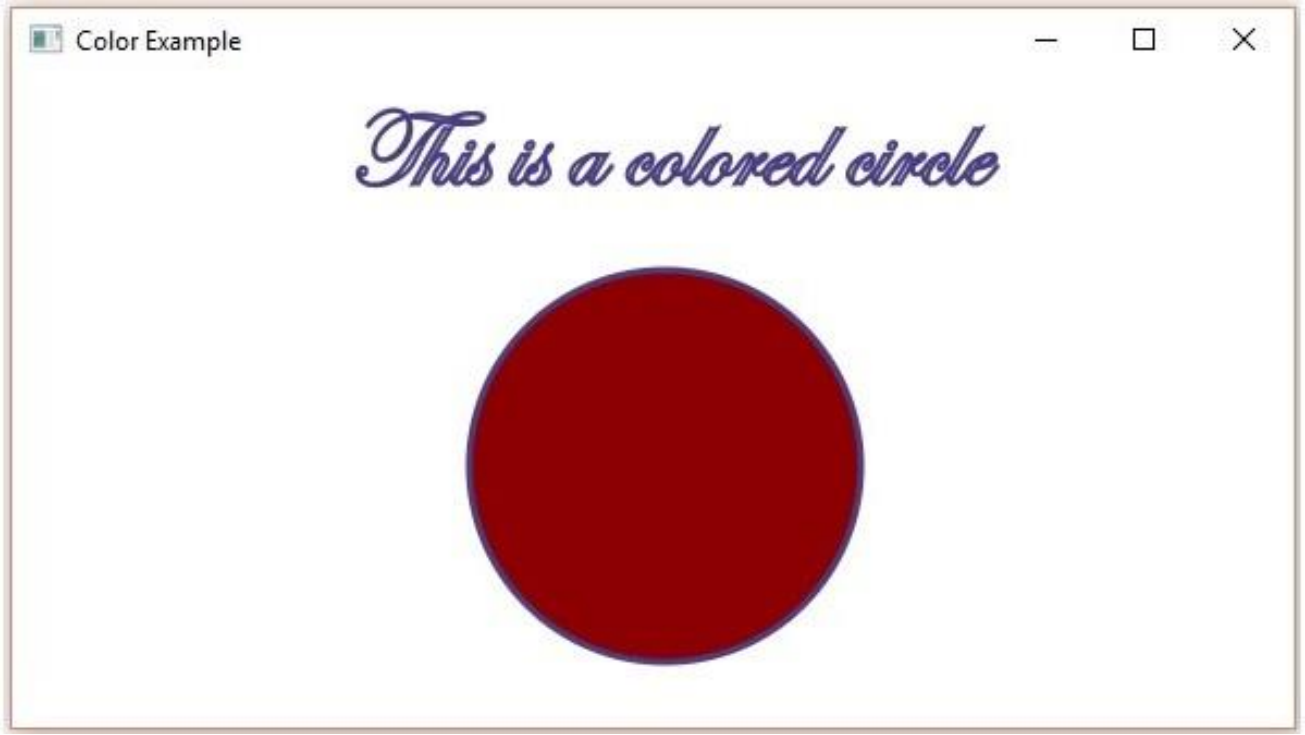
```
public class Shape_Example extends Application {

  @Override
  public void start(Stage primarystage) {
    Group root = new Group();
    primarystage.setTitle("Color Example");
    Rectangle rect = new Rectangle();
    rect.setX(50);
    rect.setY(20);
    rect.setWidth(200);
    rect.setHeight(250);
    rect.setEffect(new DropShadow());
    rect.setFill(Color.web("#0000FF",1));
    root.getChildren().add(rect);
    Scene scene = new Scene(root,300,400);
    primarystage.setScene(scene);
    primarystage.show();
  }
```



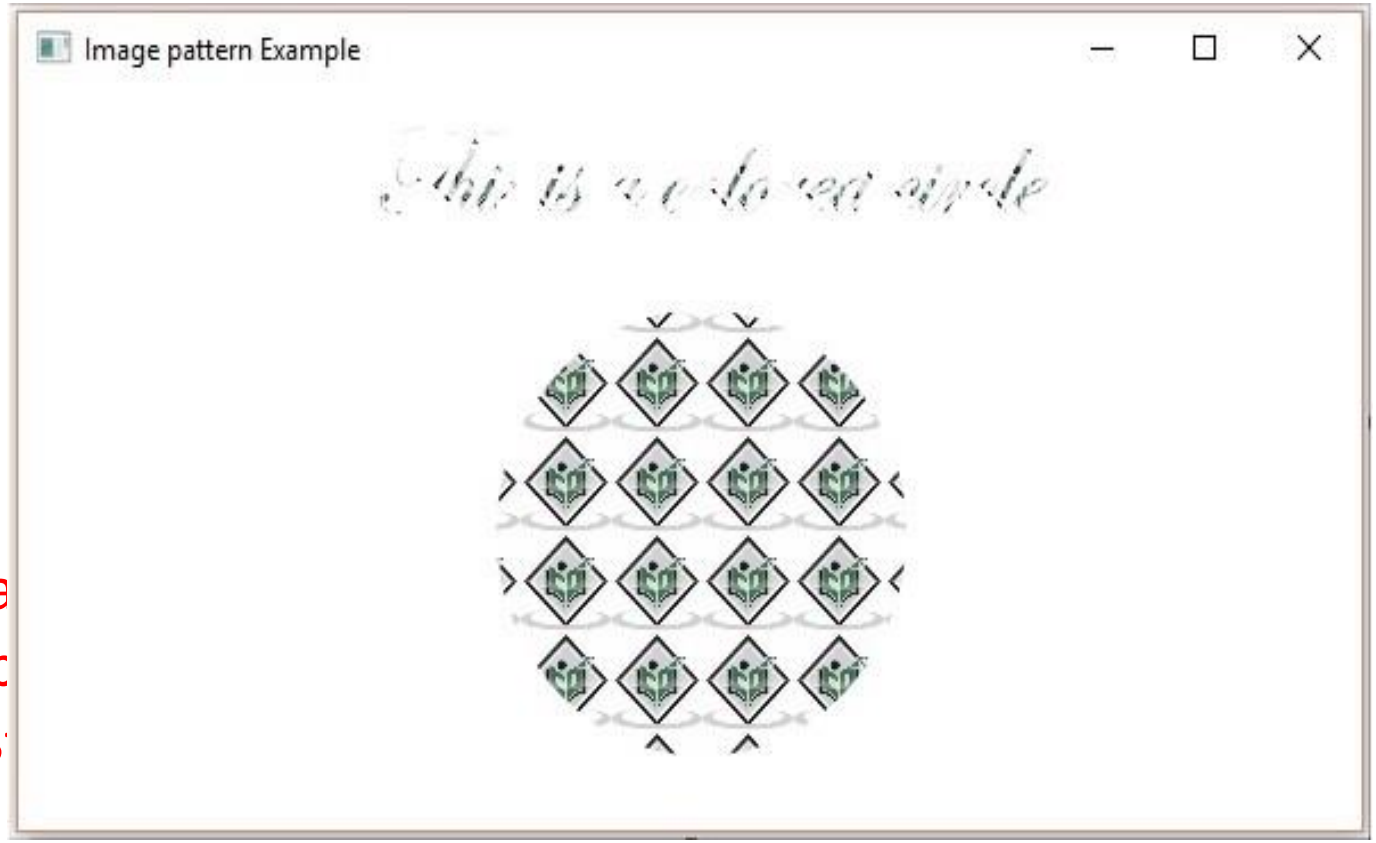
```
//Setting color to the text
Color color = new Color.BEIGE
text.setFill(color);

//Setting color to the stroke
Color color = new Color.DARKSLATEBLUE
circle.setStroke(color);
```

```
//creating color object by passing RGB values
Color c = Color.rgb(0,0,255);

//creating color object by passing HSB values
Color c = Color.hsb(270,1.0,1.0);

//creating color object by passing the hash
code for web
Color c = Color.web("0x0000FF",1.0);
```

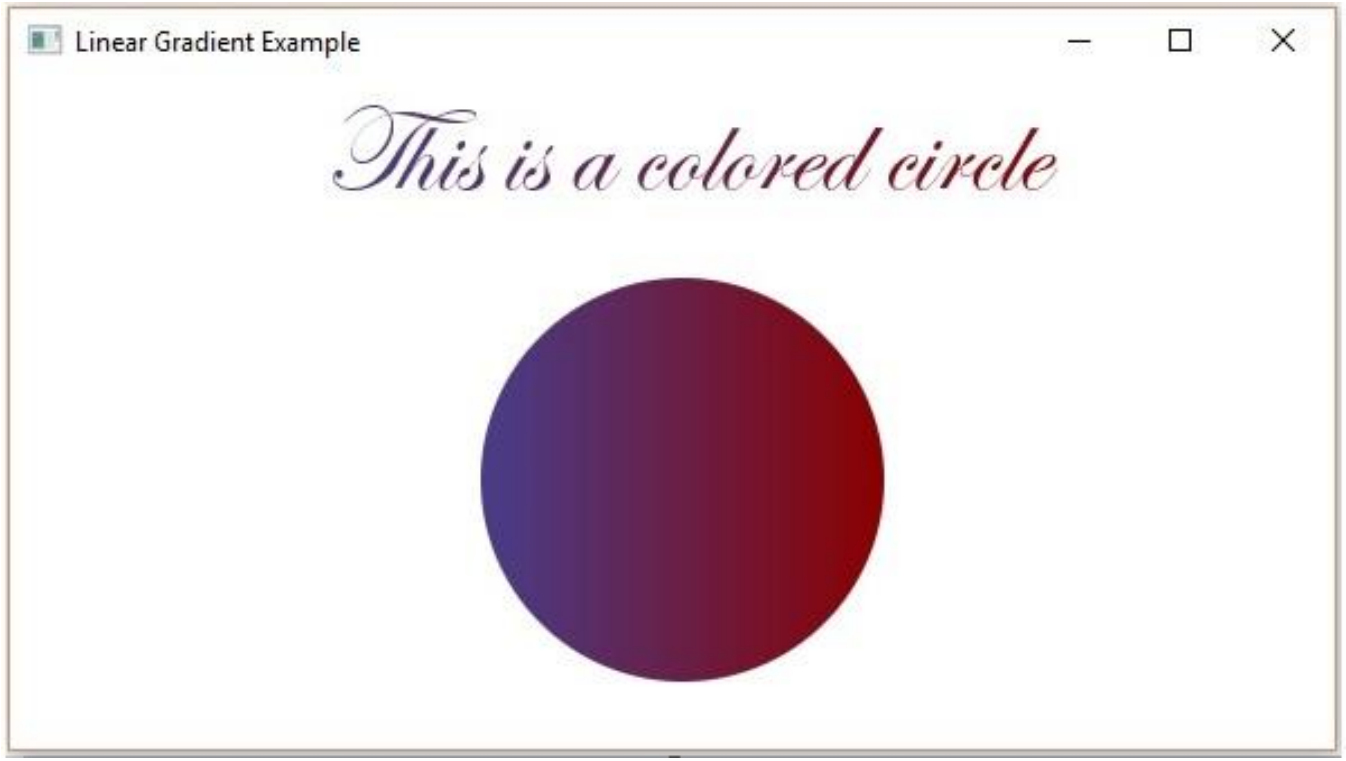
```
Circle circle = new Circle();
circle.setCenterX(300.0f);
    circle.setCenterY(180.0f);
    circle.setRadius(90.0f);
    circle.setFill(Color.DARKRED);
     circle.setStrokeWidth(3);
     circle.setStroke(Color.DARKSLATEBLUE);
    Text text = new Text("This is a colored circle");
     text.setFont(Font.font("Edwardian Script ITC", 50));
     text.setX(155);
    text.setY(50);
    text.setFill(Color.BEIGE);
     text.setStrokeWidth(2);
     text.setStroke(Color.DARKSLATEBLUE);
    Group root = new Group(circle, text);
    Scene scene = new Scene(root, 600, 300);
    stage.setTitle("Color Example");
    stage.setScene(scene);
    stage.show();
```



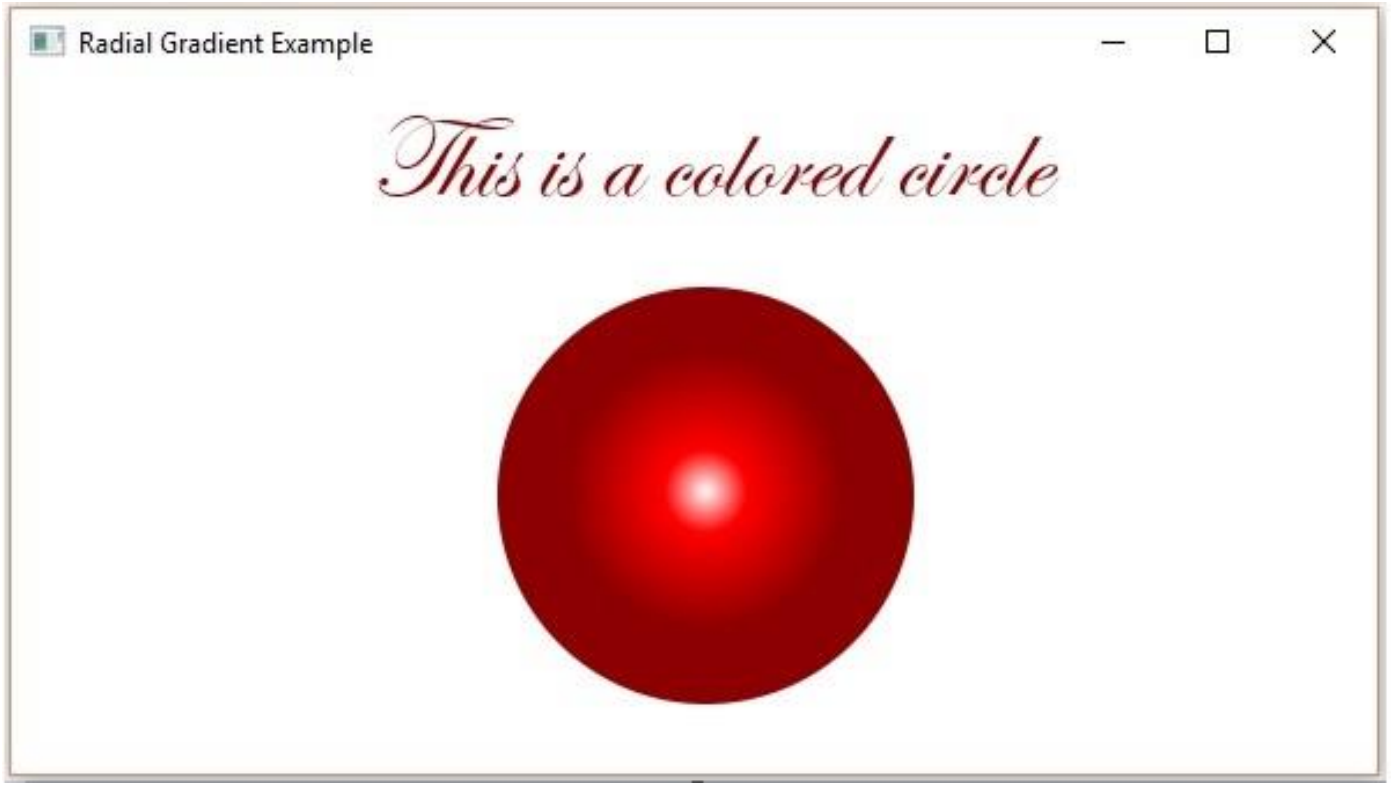
```
Circle circle = new Circle();
circle.setCenterX(300.0f);
      circle.setCenterY(180.0f);
      circle.setRadius(90.0f);
      Text text = new Text("This is a colored circle");
      text.setFont(Font.font("Edwardian Script ITC", 50));
      text.setX(155);
      text.setY(50);
      String link = "https://encrypted-tbn1.gsta
                  + "/images?q=tbn:ANd9GcRQub
                  + "rOxSzQuQ9zl5ysnjRn87VOC8
       Image image = new Image(link);
    ImagePattern radialGradient = new ImagePattern(image, 20, 20, 40, 40, false);
      circle.setFill(radialGradient);
      text.setFill(radialGradient);
      Group root = new Group(circle, text);
       Scene scene = new Scene(root, 600, 300);
       stage.setTitle("Image pattern Example");
      stage.setScene(scene);
       stage.show();
```



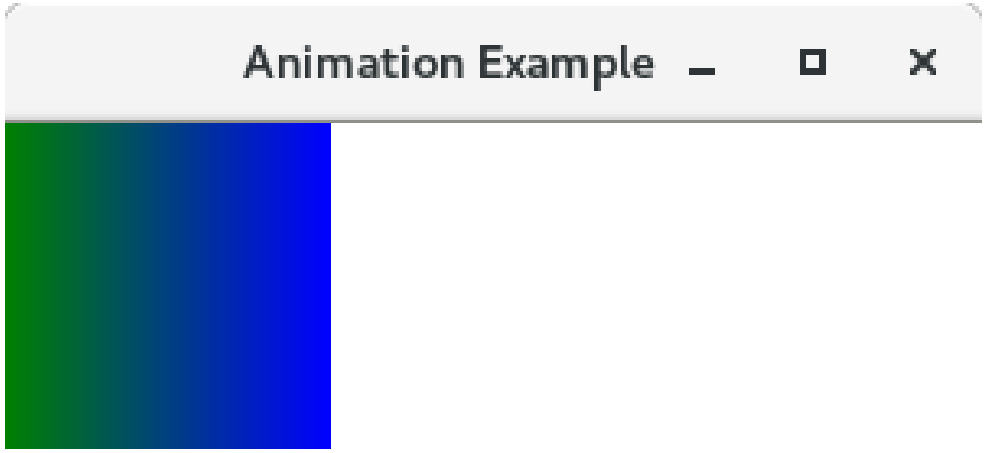
```
Circle circle = new Circle();
circle.setCenterX(300.0f);
    circle.setCenterY(180.0f);
    circle.setRadius(90.0f);
    Text text = new Text("This is a colored circle");
    text.setFont(Font.font("Edwardian Script ITC", 55));
    text.setX(140);
    text.setY(50);
    Stop[] stops = new Stop[] {
        new Stop(0, Color.DARKSLATEBLUE),
        new Stop(1, Color.DARKRED)
     };
    LinearGradient linearGradient =
        new LinearGradient(0, 0, 1, 0, true,
            CycleMethod.NO_CYCLE, stops);
     circle.setFill(linearGradient);
     text.setFill(linearGradient);
     Group root = new Group(circle, text);
    Scene scene = new Scene(root, 600, 300);
    stage.setTitle("Linear Gradient Example");
    stage.setScene(scene);
     stage.show();
```



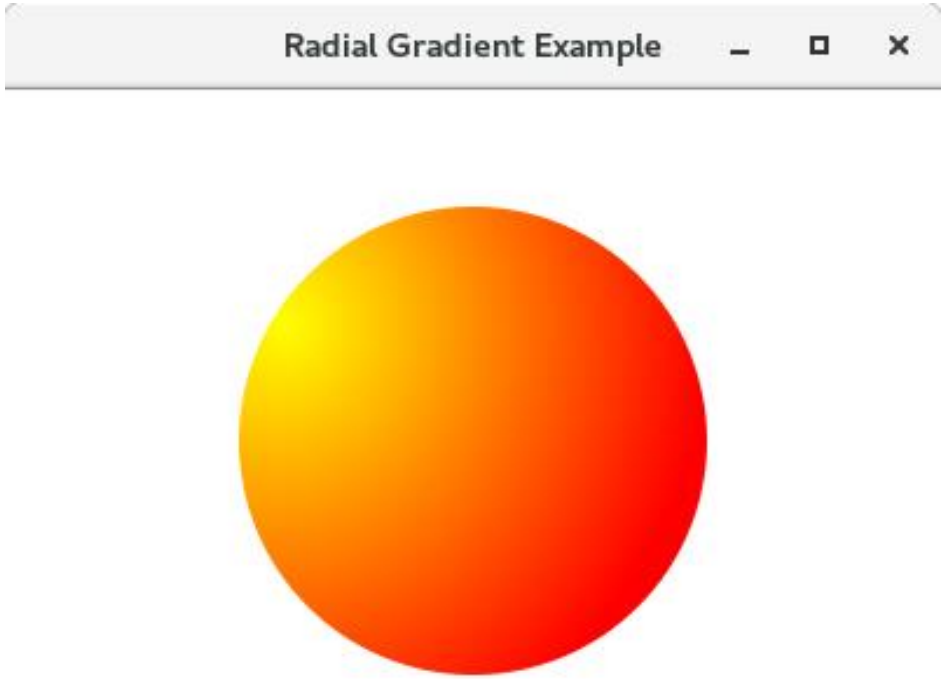
```
Circle circle = new Circle();
     circle.setCenterX(300.0f);
    circle.setCenterY(180.0f);
    circle.setRadius(90.0f);
    Text text = new Text("This is a colored circle");
    text.setFont(Font.font("Edwardian Script ITC", 50));
    text.setX(155);
    text.setY(50);
     Stop[] stops = new Stop[] {
        new Stop(0.0, Color.WHITE),
        new Stop(0.3, Color.RED),
        new Stop(1.0, Color.DARKRED)
      };
    RadialGradient radialGradient =
        new RadialGradient(0, 0, 300, 178, 60,  false, CycleMethod.NO_CYCLE, stops);
     circle.setFill(radialGradient);
     text.setFill(radialGradient);
    Group root = new Group(circle, text);
    Scene scene = new Scene(root, 600, 300);
    stage.setTitle("Radial Gradient Example");
stage.setScene(scene);
stage.show();
```



```
public void start(Stage primaryStage) {
        VBox root = new VBox();
        final Scene scene = new Scene(root,300, 250);
         Stop[] stops = new Stop[] { new Stop(0,
Color.GREEN), new Stop(1, Color.BLUE)};
         LinearGradient linear = new
LinearGradient(0, 0, 1, 0, true,
CycleMethod.NO_CYCLE, stops);
        Rectangle rect = new Rectangle(0, 0, 100, 100);
        rect.setFill(linear);
        root.getChildren().add(rect);
        primaryStage.setScene(scene);
        primaryStage.setTitle("Animation Example");
        primaryStage.show();
    }
    public static void main(String[] args) {
        launch(args);
    }
    }
```



```
public static void main(String[] args) {
      Application.launch(args);
   }
   @Override
   public void start(final Stage primaryStage) {
      primaryStage.setTitle("Radial Gradient Example");
      Group root = new Group();
      Scene scene = new Scene(root, 400, 300, Color.WHITE);
      primaryStage.setScene(scene);
      addRectangle(scene);
      primaryStage.show();
   }
   private void addRectangle(final Scene scene) {
      Circle C = new Circle(200,150,100);
       RadialGradient gradient1 = new RadialGradient(0,  .1,  100,100,  200,
false,CycleMethod.NO_CYCLE,
            new Stop(0, Color.YELLOW),  new Stop(1, Color.RED));
       C.setFill(gradient1);
      final Group root = (Group) scene.getRoot();
      root.getChildren().add(C);

   }
}
```

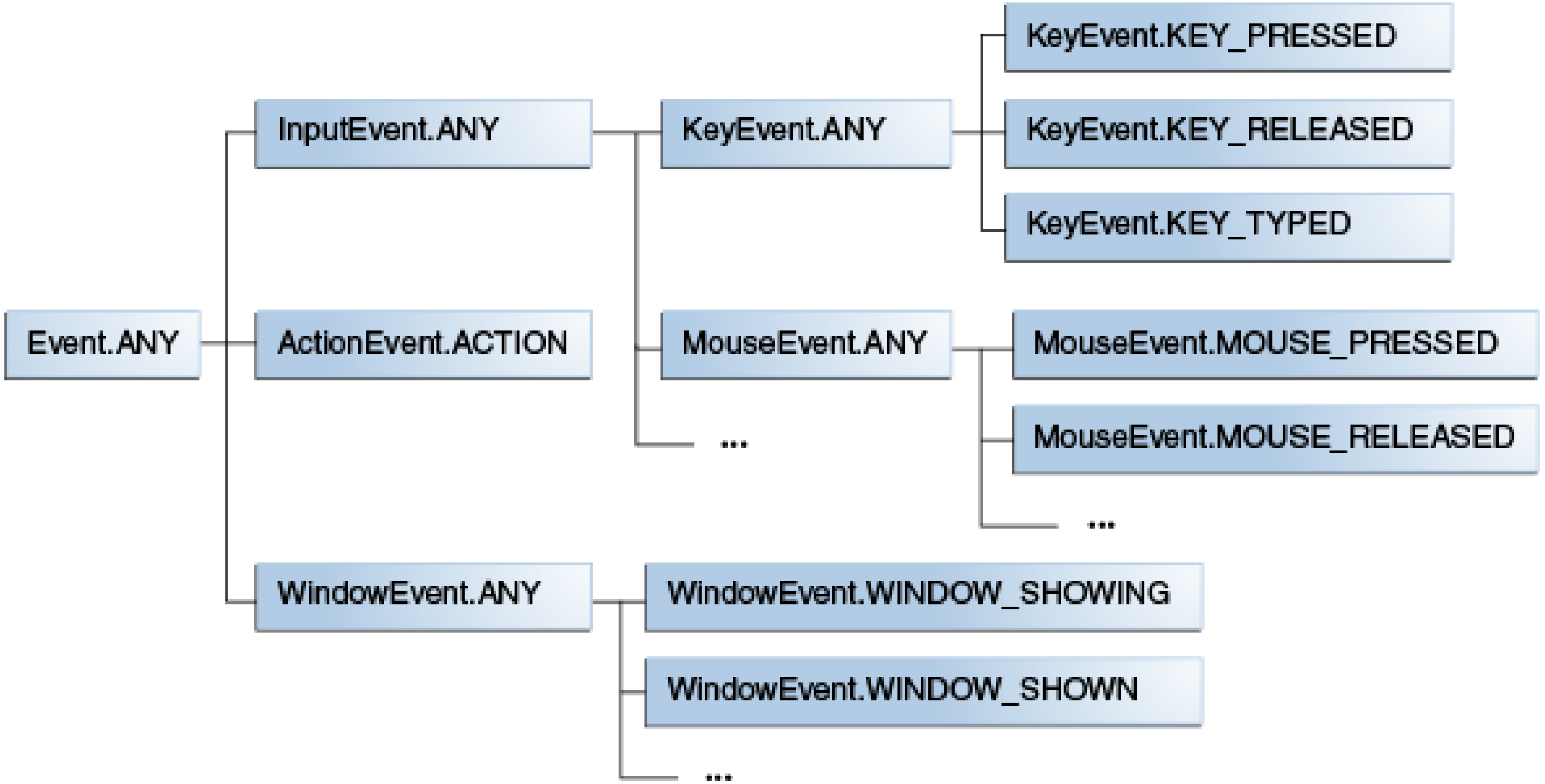


```
//Passing FileInputStream object as a parameter
FileInputStream inputstream = new
FileInputStream("C:\\images\\image.jpg");
Image image = new Image(inputstream);

//Loading image from URL
//Image image = new Image(new FileInputStream("url for the
image));
```

```
Image image = new Image(new FileInputStream
      ("path of the image"));
 ImageView imageView = new ImageView(image);
 imageView.setX(50);
imageView.setY(25);
imageView.setFitHeight(455);
imageView.setFitWidth(500);
imageView.setPreserveRatio(true);
Group root = new Group(imageView);
 Scene scene = new Scene(root, 600, 500);
stage.setTitle("Loading an image");
stage.setScene(scene);
stage.show();
```

```
Image image = new Image(new FileInputStream("file path"));
    ImageView imageView1 = new ImageView(image);
    imageView1.setX(50);
    imageView1.setY(25);
    imageView1.setFitHeight(300);
    imageView1.setFitWidth(250);
    imageView1.setPreserveRatio(true);
      ImageView imageView2 = new ImageView(image);
    imageView2.setX(350);
    imageView2.setY(25);
    imageView2.setFitHeight(150);
    imageView2.setFitWidth(250);
    imageView2.setPreserveRatio(true);
      ImageView imageView3 = new ImageView(image);
     imageView3.setX(350);
    imageView3.setY(200);
    imageView3.setFitHeight(100);
    imageView3.setFitWidth(100);
imageView3.setPreserveRatio(true);
Group root = new Group(imageView1, imageView2, imageView3);
    Scene scene = new Scene(root, 600, 400);
    stage.setTitle("Multiple views of an image");
    stage.setScene(scene);
    stage.show();
```

```
Image image = new Image(new FileInputStream("C:\\images\\logo.jpg"));
    int width = (int)image.getWidth();
    int height = (int)image.getHeight();
      WritableImage wImage = new WritableImage
      (width, height);
      PixelReader pixelReader = image.getPixelReader();
    PixelWriter writer = wImage.getPixelWriter();
    for(int y = 0; y < height; y++) {
      for(int x = 0; x < width; x++) {
        Color color = pixelReader.getColor(x, y);
        writer.setColor(x, y, color.darker());


     ImageView imageView = new ImageView(wImage);
      Group root = new Group(imageView);
    Scene scene = new Scene(root, 600, 500);
    stage.setTitle("Writing pixels ");
    stage.setScene(scene);
    stage.show();
```


Event.ANY
InputEvent.ANY
ActionEvent.ACTION
WindowEvent.ANY
KeyEvent.ANY
MouseEvent.ANY
...
KeyEvent.KEY_PRESSED
KeyEvent.KEY_RELEASED
KeyEvent.KEY_TYPED
MouseEvent.MOUSE_PRESSED
MouseEvent.MOUSE_RELEASED
...
WindowEvent.WINDOW_SHOWING
WindowEvent.WINDOW_SHOWN
...

```
public void start(Stage primaryStage) {
// Omitted
        btEnlarge.setOnAction( new EnlargeHandler());
}
class EnlargeHandler implements EventHandler<ActionEvent> {
        public void handle(ActionEvent e) {
        circlePane.enlarge();
        }
}
```

```
 public void start(Stage primaryStage) {
// Omitted
        btEnlarge.setOnAction( new class EnlargeHandlner
        implements EventHandler<ActionEvent>()
        {
                public void handle(ActionEvent e)
                {
                        circlePane.enlarge();
                }
        });
}
```

ANY:

MOUSE_PRESSED:

MOUSE_RELEASED:

MOUSE_CLICKED:

MOUSE_MOVED:

MOUSE_ENTERED:

MOUSE_ENTERED_TARGET:

MOUSE_EXITED:

MOUSE_EXITED_TARGET:

DRAG_DETECTED:

MOUSE_DRAGGED:

ANY

KEY_PRESSED

KEY_RELEASED

KEY_TYPED

| SN | Transition | Description |
|---|---|---|
| 1 | Rotate Transition | Rotate the Node along one of the axes over the specified duration. |
| 2 | Scale Transition | Animate the scaling of the node over the specified duration. |
| 3 | Translate Transition | Translate the node from one position to another over the specified duration. |
| 4 | Fade Transition | Animate the opacity of the node. It keeps updating the opacity of the node over a specified duration in order to reach a target opacity value |
| 5 | Fill Transition | Animate the node's fill color so that the fill color of the node fluctuates between the two color values over the specified duration. |
| 6 | Stroke Transition | Animate the node's stroke color so that the stroke color of the node fluctuates between the two color values over the specified duration. |
| 7 | Perform the list of transitions on a node in the sequential order. | |
| 8 | Parallel Transition | Perform the list of transitions on a node in parallel. |
| 9 | Path Transition | Move the node along the specified path over the specified duration. |

# JavaFX Rotate Transition

| Property | Description | Setter Methods |
|---|---|---|
| axis | This is a object type property of the class Point3D. This represents the axis of rotate transition. | setAxis(Point3D value) |
| byAngle | This is a double type property. This represents the angle by which the object will be rotated. | setByAngle(double value) |
| duration | This is the object type property of the class Duration. This represents the duration of the rotate transition. | setDuration(Duration value) |
| fromAngle | It is a double type property. It represents the start Angle of the rotate transition. | setFromAngle(double value) |
| node | It is an object type property of the class Node. It represents the node on which the rotate transition to be applied. | setNode(Node value) |
| toAngle | It is a double type property. It represents the stop angle value for the rotate transition. | setToAngle(double value) |

```
Rectangle rect = new Rectangle(200,100,200,200);
    rect.setFill(Color.LIMEGREEN);
    rect.setStroke(Color.HOTPINK);
    rect.setStrokeWidth(5);
    RotateTransition rotate = new RotateTransition();
      rotate.setAxis(Rotate.Z_AXIS);
       rotate.setByAngle(360);
      rotate.setCycleCount(500);
      rotate.setDuration(Duration.millis(1000));
      rotate.setAutoReverse(true);
      rotate.setNode(rect);
      rotate.play();
```

Image

# JavaFX Scale Transition

| Property | Description | Setter Methods |
|---|---|---|
| byX | This is a double type property. It represents the incremented stop X factor value. | setByX(double value) |
| byY | This is a double type property. It represents the incremented stop Y factor value. | setByY(double value) |
| byZ | This is a double type property. It represents the incremented stop Z factor value. | setByZ(double value) |
| duration | This is an object type property of the class Duration . It represents the duration of scale transition. | setDuration(Duration value) |
| fromX | This is a double type property. It represents the start X value of ScaleTransition. | setFromX(double value) |
| fromY | This is a double type property. It represents the start Y value of ScaleTransition. | setFromY(double value) |
| fromZ | This is a double type property. It represents the start Z value of ScaleTransition. | setFromZ(double value) |
| node | This is an object type property of the class Node. It represents onto which, the scale transition is applied. | setNode(Node node) |
| toX | This is a double type property. It represents the stop X scale value of the scale transition. | setToX(double value) |
| toY | This is a double type property. It represents the stop Y scale value of the scale transition. | setToY(double value) |
| toZ | This is a double type property. It represents the stop Z scale value of the scale transition. | setToZ(double value) |

```
Circle circle = new Circle();
    circle.setCenterX(300.0f);
    circle.setCenterY(135.0f);
    circle.setRadius(50.0f);
    circle.setFill(Color.BROWN);
    circle.setStrokeWidth(20);
```

```
ScaleTransition scaleTransition = new ScaleTransition();
scaleTransition.setDuration(Duration.millis(1000));
scaleTransition.setNode(circle);
scaleTransition.setByY(1.5);
scaleTransition.setByX(1.5);
scaleTransition.setCycleCount(50);
scaleTransition.setAutoReverse(false);
scaleTransition.play();
```

Image

# JavaFX Translate Transition

| Property | Description | Setter Method |
|---|---|---|
| byX | This is a double type property. It represents the incremented X-coordinate value at which, the translation stops. | setByX(double value) |
| byY | This is a double type property. It represents the incremented Y-coordinate value at which, the translation stops. | setByY(double value) |
| byZ | This is a double type property. It represents the incremented Z-coordinate value at which, the translation stops. | setByZ(double value) |
| duration | This is an object type property of the class Duration . It represents the duration of Translate transition. | setDuration(Duration value) |
| fromX | This is a double type property. It represents the X coordinate value from which the translation starts. | setFromX(double value) |
| fromY | This is a double type property. It represents the Y coordinate value from which the translation starts. This is a double type property. It represents the X coordinate value from which the translation starts. | setFromY(double value) |
| fromZ | This is a double type property. It represents the Z coordinate value from which the translation starts. | setFromZ(double value) |
| node | This is an object type property of the class Node. It represents the node onto which, the scale transition is applied. | setNode(Node node) |
| toX | This is a double type property. It represents the stop X coordinate value of the translate transition. | setToX(double value) |
| toY | This is a double type property. It represents the stop Y coordinate value of the translate transition. | setToY(double value) |
| toZ | This is a double type property. It represents the stop Z coordinate value of the scale transition. | setToZ(double value) |

```
Circle cir = new Circle(50,100,50);
        cir.setFill(Color.RED);
        cir.setStroke(Color.BLACK);
        TranslateTransition translate = new TranslateTransition();
            translate.setByX(400);
            translate.setDuration(Duration.millis(1000));
            translate.setCycleCount(500);
            translate.setAutoReverse(true);
            translate.setNode(cir);
            translate.play();
```

Image

# JavaFX Fade Transition

| Property | Description | Setter Methods |
|---|---|---|
| byValue | It is a double type property. It represents the incremented stop opacity value of the Fade transition. | setByValue(double property) |
| Duration | This is an object type property of the class Duration. It represent the duration of this fade transition. | setDuration(Duration duration) |
| fromValue | This is a double type property. It represents the start opacity for the fade transition. | setFromValue(double value) |
| node | This is an object type property of the class Node. This represents the node onto which, the transition is to be applied. | setNode(Node node) |
| toValue | This is a double type property. This represents the stop value of the opacity for the fade transition. | setToValue(double value) |

```
Circle cir = new Circle(250,120,80);
        cir.setFill(Color.RED);
        cir.setStroke(Color.BLACK);
```

```
FadeTransition fade = new FadeTransition();
    fade.setDuration(Duration.millis(5000));
    fade.setFromValue(10);
    fade.setToValue(0.1);
    fade.setCycleCount(1000);
    fade.setAutoReverse(true);
    fade.setNode(cir);
    fade.play();
```

Image

# JavaFX Fill Transition

| Property | Description | Setter Methods |
|---|---|---|
| duration | It is an object type property of the class Duration. It represents the duration of the Fill Transition. | setDuration(Duration duration) |
| fromValue | It is a double type property. It represents the start color value for the fill transition. | setFromValue(Color value) |
| shape | It is an object type property of the class Shape. It represents the shape on which the fill transition is applied. | setShape(Shape shape) |
| toValue | This is a double type property. It represents the stop color value for the fill transition. | setToValue(Color value) |

```
Polygon poly = new Polygon();
    poly.getPoints().addAll(new Double[]
{220.0,270.0,170.0,220.0,170.0,120.0,220.0,70.0,270.0,120.0,270.0,220.0});
    poly.setStroke(Color.BLACK);
```

```
FillTransition fill = new FillTransition();
fill.setAutoReverse(true);
fill.setCycleCount(50);
fill.setDuration(Duration.millis(1000));
fill.setFromValue(Color.BLACK);
fill.setToValue(Color.WHITE);
fill.setShape(poly);
fill.play();
```

Image

# JavaFX Stroke Transition

| Property | Description | Setter Methods |
|---|---|---|
| duration | This is an object type property of the class Duration. It represents the duration of the stroke Transition. | setDuration(Duration duration) |
| fromValue | This is a color type property. It represents the initial value of the color for the stroke transition. | setFromValue(Color value) |
| shape | This is an object type property of the class Shape. It represents the shape onto which the Stroke transition will be applied. | setShape(Shape shape) |
| toValue | This is color type property. It represents the target value of the color for the stroke transition. | setToValue(Color value) |

```
Circle cir = new Circle(200,150,100);     cir.setStroke(Color.BLUE);
      cir.setFill(Color.RED);
      cir.setStrokeWidth(10);

    StrokeTransition stroke = new StrokeTransition();
      stroke.setAutoReverse(true);
      stroke.setCycleCount(500);
      stroke.setDuration(Duration.millis(500));
      stroke.setFromValue(Color.BLACK);

      stroke.setToValue(Color.PURPLE);
      stroke.setShape(cir);
      stroke.play();
```

Image

# JavaFX Sequential Transition

| Property | Description | Setter Method |
|---|---|---|
| node | It is an object type property of the class Node. It represent the node onto which the transition is to be applied. | setNode(Node node) |

```
Polygon poly = new Polygon();

      //Setting points for the polyogn
      poly.getPoints().addAll(new Double[]
{320.0,270.0,270.0,220.0,270.0,270.0,320.0,120.0,370.0,270.
0,370.0,220.0});
       poly.setFill(Color.LIMEGREEN);
      poly.setStroke(Color.BLACK);
    Duration dur1 = Duration.millis(1000);
     Duration dur2 = Duration.millis(500);
    PauseTransition pause = new
            PauseTransition(Duration.millis(1000));
      FadeTransition fade = new FadeTransition(dur2);
     fade.setFromValue(1.0f);
     fade.setToValue(0.3f);
     fade.setCycleCount(2);
     fade.setAutoReverse(true);
TranslateTransition translate = new
TranslateTransition(dur1);
       translate.setToX(-150f);
       translate.setCycleCount(2);
       translate.setAutoReverse(true);
        RotateTransition rotate = new RotateTransition(dur2);
       rotate.setByAngle(180f);
       rotate.setCycleCount(4);
       rotate.setAutoReverse(true);
        ScaleTransition scale = new ScaleTransition(dur1);
       scale.setByX(1.5f);
       scale.setByY(1.2f);
       scale.setCycleCount(2);
       scale.setAutoReverse(true);
        SequentialTransition seqT = new SequentialTransition
(poly,rotate, pause, fade, translate,  scale);
       seqT.play();
```

Image

# JavaFX Parallel Transition

| Property | Description | Setter Methods |
|---|---|---|
| node | This is the property of the type object of class Node. It represents the node onto which the transition is to be applied. | setNode(Node node) |

```
Polygon poly = new Polygon();
      poly.getPoints().addAll(new Double[]
{320.0,270.0,270.0,220.0,270.0,270.0,320.0,120.0,370.0,270.0,
370.0,220.0});
         poly.setFill(Color.LIMEGREEN);
        poly.setStroke(Color.BLACK);
      Duration dur1 = Duration.millis(1000);
       Duration dur2 = Duration.millis(1000);
      PauseTransition pause = new
              PauseTransition(Duration.millis(1000));
        FadeTransition fade = new FadeTransition(dur2);
       fade.setFromValue(1.0f);
       fade.setToValue(0.3f);
       fade.setCycleCount(5);
       fade.setAutoReverse(true);
```

```
TranslateTransition translate = new TranslateTransition(dur1);
        translate.setToX(-150f);
        translate.setCycleCount(5);
        translate.setAutoReverse(true);
         RotateTransition rotate = new RotateTransition(dur2);
        rotate.setByAngle(360f);
        rotate.setCycleCount(5);
        rotate.setAutoReverse(true);
         ScaleTransition scale = new ScaleTransition(dur1);
        scale.setByX(1.5f);
        scale.setByY(1.2f);
        scale.setCycleCount(5);
        scale.setAutoReverse(true);
         ParallelTransition seqT = new ParallelTransition
(poly,rotate, pause, fade, translate,  scale);
        seqT.play();
```

Image Image2

# JavaFX Pause Transition

| Property | Description | Setter Methods |
|---|---|---|
| duration | It is a type of object of the class Duration. It represents the lifespan of the transition. | setDuration(Duration duration) |

```
Polygon poly = new Polygon();
      poly.getPoints().addAll(new Double[]
{320.0,270.0,270.0,220.0,270.0,270.0,320.0,120.0,370.0,270.0,
370.0,220.0});
       poly.setFill(Color.LIMEGREEN);
       poly.setStroke(Color.BLACK);
       RotateTransition rt = new
RotateTransition(Duration.millis(500),poly);
       rt.setByAngle(180);
       rt.setCycleCount(2);
       rt.setAutoReverse(true);
       TranslateTransition translate = new
TranslateTransition(Duration.millis(500),poly);
       translate.setToX(-150f);
       translate.setCycleCount(2);
       translate.setAutoReverse(true);
```

```
FadeTransition fade = new
FadeTransition(Duration.millis(500),poly);
       fade.setFromValue(1.0f);
       fade.setToValue(0.5f);
       fade.setCycleCount(2);
       fade.setAutoReverse(true);
        SequentialTransition seqTransition = new
SequentialTransition (fade,new
PauseTransition(Duration.millis(2000)),rt,
       new PauseTransition(Duration.millis(2000)),translate);
        seqTransition.play();
```

Image

# JavaFX Path Transition

| Property | Description | Setter Methods |
|---|---|---|
| duration | This property is an object type of the class Duration. This represents the lifespan of the transition. | setDuraton(Duration duration) |
| node | This is an object of the class Node. This represents the node onto which the transition will be applied. | setNode(Node node) |
| orientation | This is a object type property referenced by **PathTransition.OrientationType.** It represents the upright orientation of the node along the path. | SetOrientation(PathTransition.OrientationType orientation-type) |
| path | This is an object type property of the class Shape. It specified the shape through which the outline of the animated path undergoes. | setPath(Shape shape) |

```
Polygon poly = new Polygon();
   poly.getPoints().addAll(new Double[]
{320.0,270.0,270.0,220.0,270.0,270.0,320.0,120.0,370.0,270.0,370.0,220.0});
     poly.setFill(Color.LIMEGREEN);
   poly.setStroke(Color.BLACK);

 Path path = new Path();
   path.getElements().add (new MoveTo (150f, 70f));
   path.getElements().add (new CubicCurveTo (240f, 230f, 500f, 340f, 600, 50f));
     PathTransition pathTransition = new PathTransition();
    pathTransition.setDuration(Duration.millis(1000));
   pathTransition.setNode(poly);
     pathTransition.setPath(path);
 pathTransition.setOrientation(OrientationType.ORTHOGONAL_TO_TANGENT);
     pathTransition.setCycleCount(10);
   pathTransition.setAutoReverse(true);
 pathTransition.play();
```

Image

```
public class ObservablePropertyDemo {
 public static void main(String[] args) {
        DoubleProperty balance = new SimpleDoubleProperty();
        balance.addListener(new InvalidationListener() {
                public void invalidated(Observable ov) {
                        System.out.println("The new value is “balance.doubleValue());
                }
        });
        balance.set(4.5);
```

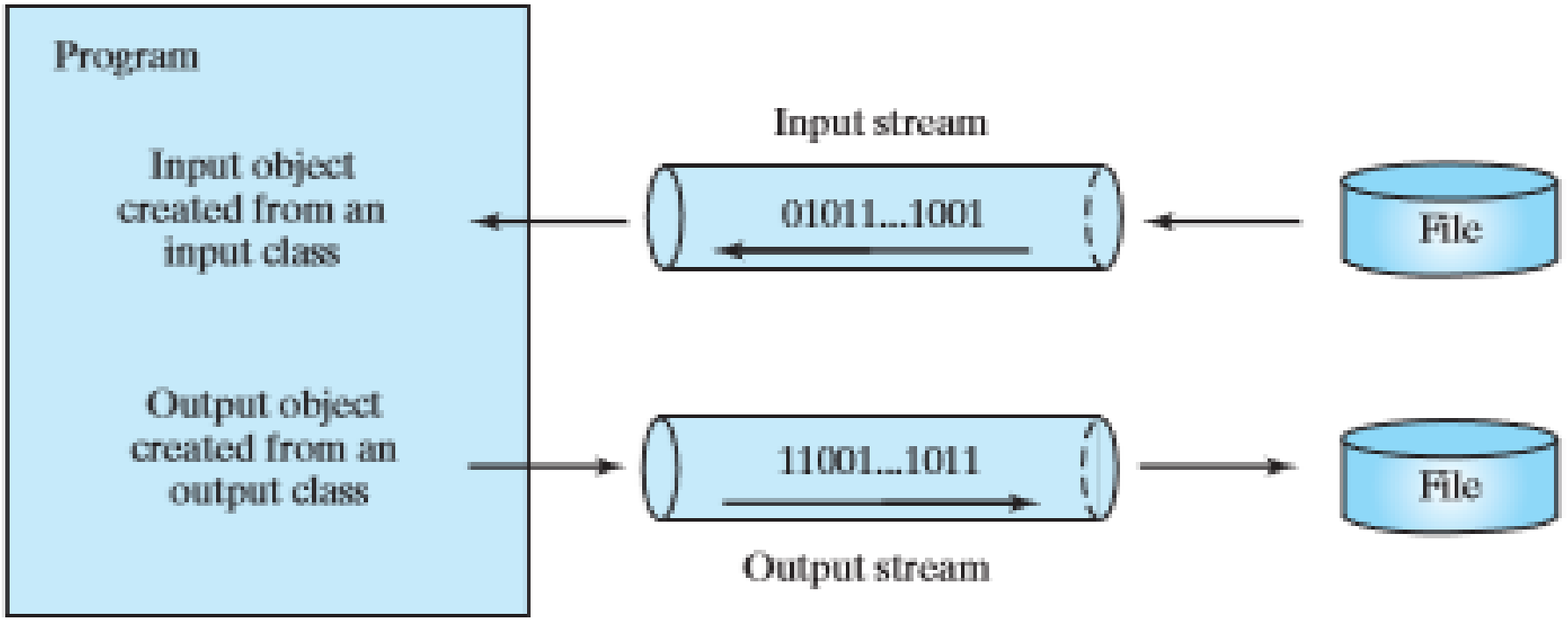
```
StackPane root = new StackPane();
    Button btn=new
Button("This is a button");

    Scene scene=new Scene(root,300,300);
    root.getChildren().add(btn);
    primaryStage.setScene(scene);
    primaryStage.setTitle("Button Class
Example");
    primaryStage.show();
```

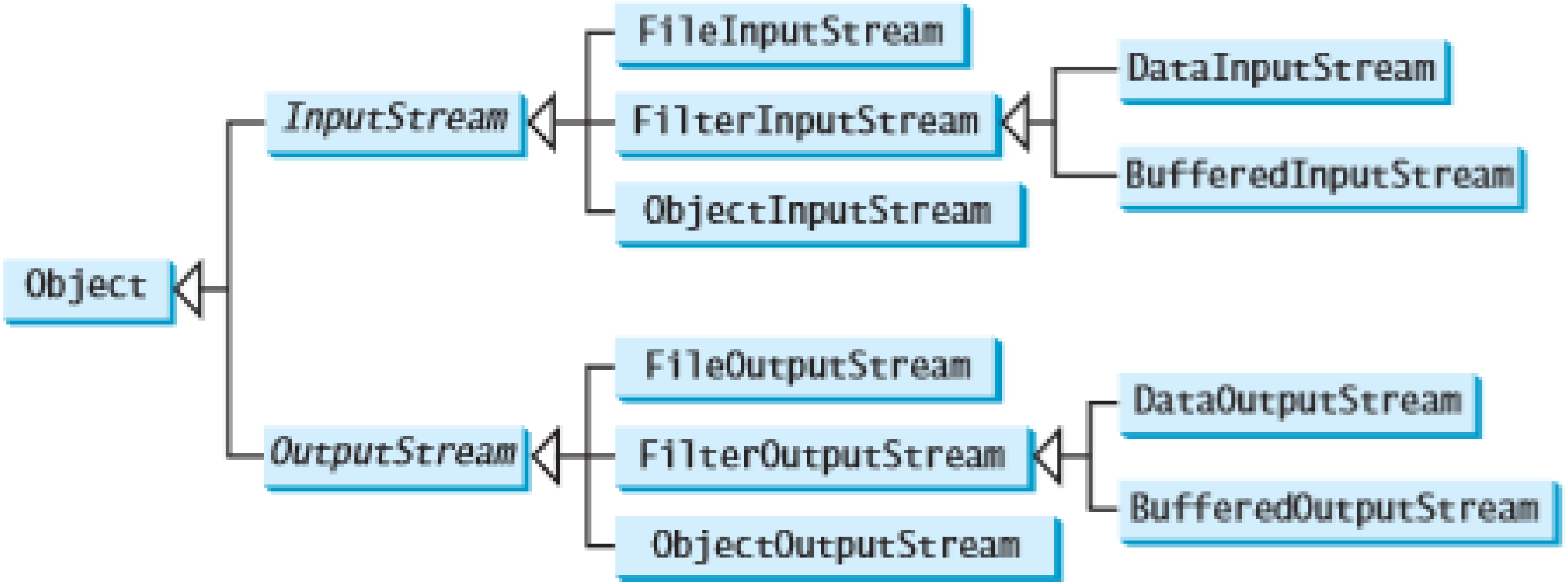
```
ToggleGroup group = new ToggleGroup();
  RadioButton button1 = new RadioButton("option 1");
  RadioButton button2 = new RadioButton("option 2");
  RadioButton button3 = new RadioButton("option 3");
  RadioButton button4 = new RadioButton("option 4");
  button1.setToggleGroup(group);
  button2.setToggleGroup(group);
  button3.setToggleGroup(group);
  button4.setToggleGroup(group);

  VBox root=new VBox();
  root.setSpacing(10);
  root.getChildren().addAll(button1,button2,button3,button4);
  Scene scene=new Scene(root,400,300);
  primaryStage.setScene(scene);
  primaryStage.setTitle("Radio Button Example");
  primaryStage.show();
```

```
String path = "/home/javatpoint/Downloads/test.mp3";
      Media media = new Media(new File(path).toURI().toString());
        MediaPlayer mediaPlayer = new MediaPlayer(media);
        mediaPlayer.setAutoPlay(true);
      primaryStage.setTitle("Playing Audio");
      primaryStage.show();
```

```
String path = "/home/javatpoint/Downloads/test.mp4";
        Media media = new Media(new File(path).toURI().toString());
        MediaPlayer mediaPlayer = new MediaPlayer(media);
        MediaView mediaView = new MediaView(mediaPlayer);
        mediaPlayer.setAutoPlay(true);
        Group root = new Group();
      root.getChildren().add(mediaView);
      Scene scene = new Scene(root,500,400);
      primaryStage.setScene(scene);
      primaryStage.setTitle("Playing video");
      primaryStage.show();
```

```
PrintWriter output = new PrintWriter("temp.txt");
output.print("Java 101");
output.close();
Scanner input = new Scanner(new File("temp.txt"));
System.out.println(input.nextLine());
```




Object
InputStream
FileInputStream
FilterInputStream
DataInputStream
BufferedInputStream
ObjectInputStream
OutputStream
FileOutputStream
FilterOutputStream
DataOutputStream
BufferedOutputStream
ObjectOutputStream

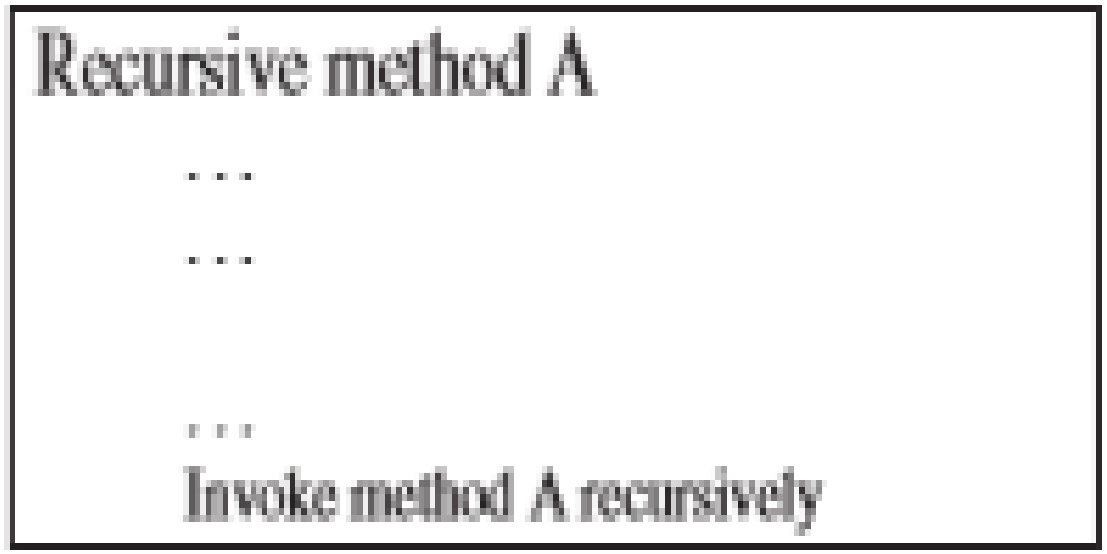
```
public class Student implements java.io.Serializable{
  private int stuRollNum;
  private int stuAge;
  private String stuName;
  private transient String stuAddress;
  private transient int stuHeight;

  public Student(int roll, int age, String name, String address, int height) {
    this.stuRollNum = roll;
    this.stuAge = age;
    this.stuName = name;
    this.stuAddress = address;
    this.stuHeight = height;
  }

  public int getStuRollNum() {
    return stuRollNum;
  }
  public void setStuRollNum(int stuRollNum) {
    this.stuRollNum = stuRollNum;
  }
```

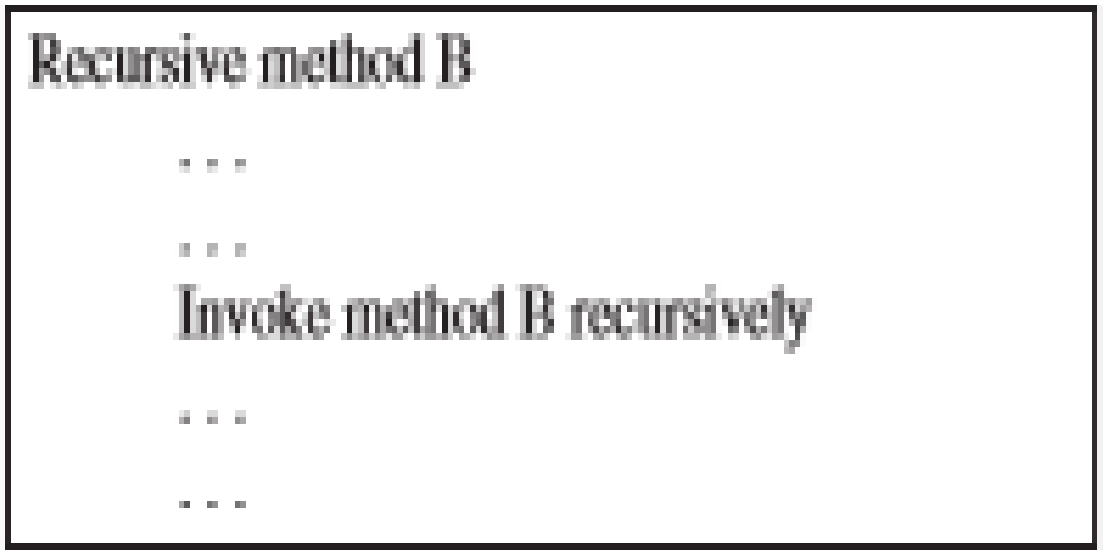
```
  public int getStuAge() {
    return stuAge;
  }
  public void setStuAge(int stuAge) {
    this.stuAge = stuAge;
  }
  public String getStuName() {
    return stuName;
  }
  public void setStuName(String stuName) {
    this.stuName = stuName;
  }
  public String getStuAddress() {
    return stuAddress;
  }
  public void setStuAddress(String stuAddress) {
    this.stuAddress = stuAddress;
  }
  public int getStuHeight() {
    return stuHeight;
  }
  public void setStuHeight(int stuHeight) {
    this.stuHeight = stuHeight;
  }
}
```

```
import java.io.FileOutputStream;
import java.io.ObjectOutputStream;
import java.io.IOException;
public class SendClass
{
  public static void main(String args[])
 {
   Student obj = new Student(101, 25, "Chaitanya", "Agra", 6);
    try{
            FileOutputStream fos = new FileOutputStream("Student.ser");
            ObjectOutputStream oos = new ObjectOutputStream(fos);
            oos.writeObject(obj);
            oos.close();
            fos.close();
            System.out.println("Serialzation Done!!");
   }
catch(IOException ioe){
            System.out.println(ioe);}}}
```

```
Student o=null;
  try{
    FileInputStream fis = new FileInputStream("Student.ser");
    ObjectInputStream ois = new ObjectInputStream(fis);
    o = (Student)ois.readObject();
    ois.close();
    fis.close();
   }
   catch(IOException ioe)
   {
     ioe.printStackTrace();
     return;
   }catch(ClassNotFoundException cnfe)
    {
     System.out.println("Student Class is not found.");
     cnfe.printStackTrace();
     return;
    }
    System.out.println("Student Name:"+o.getStuName());
       System.out.println("Student Age:"+o.getStuAge());
       System.out.println("Student Roll No:"+o.getStuRollNum());
       System.out.println("Student Address:"+o.getStuAddress());
       System.out.println("Student Height:"+o.getStuHeight());
    }
```

```java
public class TestObjectStreamForArray {
  public static void main(String[] args)
      throws ClassNotFoundException, IOException {
    int[] numbers = {1, 2, 3, 4, 5};
    String[] strings = {"John", "Susan", "Kim"};

    try ( // Create an output stream for file array.dat
      ObjectOutputStream output = new ObjectOutputStream(new
        FileOutputStream("array.dat", true));
    ) {
      // Write arrays to the object output stream
      output.writeObject(numbers);
      output.writeObject(strings);
    }

    try ( // Create an input stream for file array.dat
      ObjectInputStream input =
        new ObjectInputStream(new FileInputStream("array.dat"));
    ) {
      int[] newNumbers = (int[])(input.readObject());
      String[] newStrings = (String[])(input.readObject());

      // Display arrays
      for (int i = 0; i < newNumbers.length; i++)
        System.out.print(newNumbers[i] + " ");
      System.out.println();

      for (int i = 0; i < newStrings.length; i++)
        System.out.print(newStrings[i] + " ");
    }
  }
}
```


Recursive method A
. . .
. . .
. . .
Invoke method A recursively


Recursive method B
. . .
. . .
Invoke method B recursively
. . .
. . .

```
package java.lang;

public interface Comparable {
  public int compareTo(Object o)
}
```

```
package java.lang;

public interface Comparable<T> {
  public int compareTo(T o)
}
```

```
Comparable c = new Date();
System.out.println(c.compareTo("red"));
```

```
Comparable<Date> c = new Date();
System.out.println(c.compareTo("red"));
```

```
List list = new ArrayList();
list.add(10);
list.add("10");
```

With Generics, it is required to specify the type of object we need to store.

```
List<Integer> list = new ArrayList<Integer>();
list.add(10);
list.add("10");// compile-time error
```

**Type-safety**

```
List list = new ArrayList();
list.add("hello");
String s = (String) list.get(0);//typecasting
```

After Generics, we don't need to typecast the object.

```
List<String> list = new ArrayList<String>();
list.add("hello");
String s = list.get(0);
```

**Type casting is not required**

```
List<String> list = new ArrayList<String>();
list.add("hello");
list.add(32);//Compile Time Error
```

**Compile-Time Checking**

```
public class GenericMethodDemo {
public static void main(String[] args ) {
        Integer[] integers = {1, 2, 3, 4, 5};
        String[] strings = {"London", "Paris", "New York", "Austin“
        };
        GenericMethodDemo.<Integer>print(integers);
        GenericMethodDemo.<String>print(strings);
}
public static <E> void print(E[] list) {
        for (int i = 0; i < list.length; i++)
        System.out.print(list[i] + " ");
        System.out.println();
        }
}
```

**Generic Method**

```
public class Max {
        public static Comparable max(Comparable o1, Comparable o2) {
                if (o1.compareTo(o2) > 0)
                        return o1;
                else
                        return o2; } }
```

```
public class MaxUsingGenericType {
        public static <E extends Comparable<E>> E max(E o1, E o2) {
                if (o1.compareTo(o2) > 0)
                        return o1;
                else
                         return o2; } }
```

```
GenericStack stack;
stack = new GenericStack<String>();
stack.push("Welcome to Java");
stack.push(new Integer(2));
```

```
public class WildCardNeedDemo {
	public static void main(String[] args ) {
		GenericStack<Integer> intStack = new GenericStack<>();
		intStack.push(1); // 1 is autoboxed into new Integer(1)
		intStack.push(2);
		intStack.push(-2);
		System.out.print("The max number is " + max(intStack));
	}
	public static double max(GenericStack<Number> stack) {
		double max = stack.pop().doubleValue(); // Initialize max
		while (!stack.isEmpty()) {
		double value = stack.pop().doubleValue();
		if (value > max)
		{
			max = value;
		}
		return max; }}
```

**Why we need Wild card Generics?**

```
public class AnyWildCardDemo {
	public static void main(String[] args ) {
		GenericStack<Integer> intStack = new GenericStack<>();
		intStack.push(1); // 1 is autoboxed into new Integer(1)
		intStack.push(2);
		intStack.push(-2);
		print(intStack);
	}
	public static void print(GenericStack<?> stack) {
		while (!stack.isEmpty()) {
			System.out.print(stack.pop() + " ");
		}
	}
}
```

# Any Wild Card

```
public class SuperWildCardDemo {
	public static void main(String[] args) {
		GenericStack<String> stack1 = new GenericStack<>();
		GenericStack<Object> stack2 = new GenericStack<>();
		stack2.push("Java");
		stack2.push(2);
		stack1.push("Sun");
		add(stack1, stack2);
		AnyWildCardDemo.print(stack2);
	}
	public static <T> void add(GenericStack<T> stack1, GenericStack<?
	super T> stack2) {
		while (!stack1.isEmpty())
			stack2.push(stack1.pop());
	}
}
```

**Super Wild card**

```
ArrayList<String> list = new ArrayList<>();
list.add("Oklahoma");
String state = list.get(0);
```

```
ArrayList list = new ArrayList();
list.add("Oklahoma");
String state = (String)(list.get(0));
```

```
public static <E> void print(E[] list) {
        for (int i = 0; i < list.length; i++)
        System.out.print(list[i] + " ");
        System.out.println();
}
```

```
public static void print(Object[] list) {
        for (int i = 0; i < list.length; i++)
        System.out.print(list[i] + " ");
        System.out.println();
}
```

1. Cannot Use new E():

```
E object = new E();
```

2. Cannot Use new E[] :

```
E[] elements = new E[capacity];
E[] elements = (E[])new Object[capacity];

ArrayList<String>[] list = new ArrayList<String>[10];
```

You can use the following code to circumvent this restriction:

```
ArrayList<String>[] list = (ArrayList<String>[])new ArrayList[10];
```

3. A Generic Type Parameter of a Class Is Not Allowed in a Static Context :

```
public class Test<E> {
        public static void m(E o1) {  // Illegal
        }
        public static E o1; // Illegal
        static {
                E o2; // Illegal
```

3. A Generic Type Parameter of a Class Is Not Allowed in a Static Context :

```
public class Test<E> {
        public static void m(E o1) {  // Illegal
        }
        public static E o1; // Illegal
        static {
                E o2; // Illegal
        }
}
```