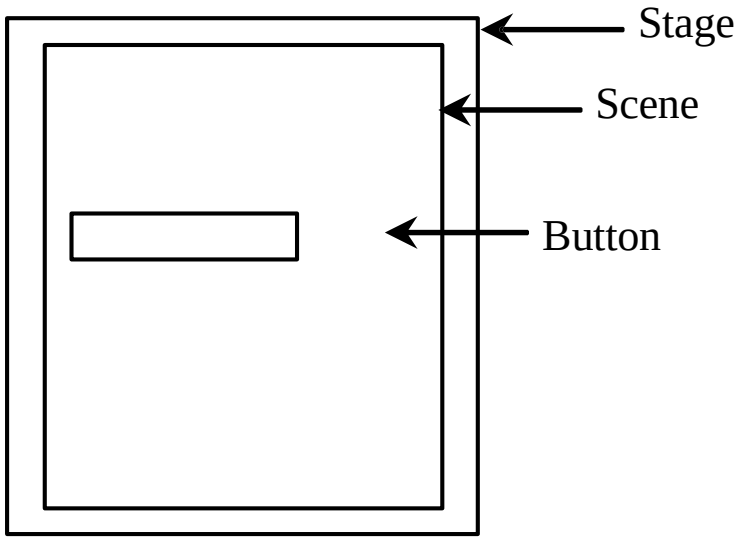
Object Oriented Programming

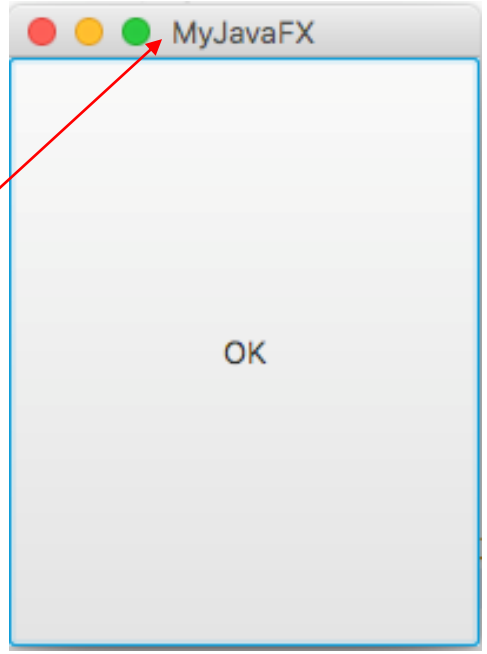
JAVA FX

Basic Structure of JavaFX Application

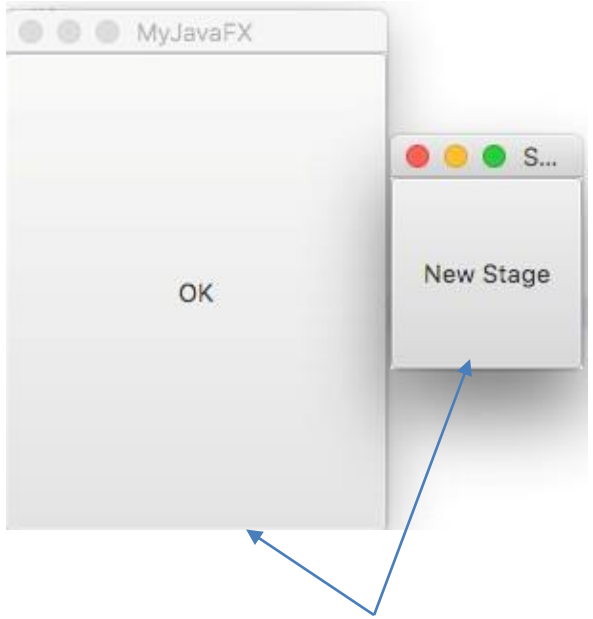
- Extend Application Class
- Override the start(Stage) method
- Stage, Scene, and Nodes



```
import javafx.application.Application;
import javafx.scene.Scene;
import javafx.scene.control.Button;
import javafx.stage.Stage;

public class MyJavaFX extends Application {

@override
public void start(Stage primaryStage) {
    // Create a button and place it in the scene

    Button btOK = new Button("OK");

    Scene scene = new Scene(btOK, 200, 250);
    primaryStage.setTitle("MyJavaFX");  // Set the stage title
    primaryStage.setScene(scene);  // Place the scene in the stage
    primaryStage.show();  // Display the stage
  }

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



```
import javafx.application.Application; import
javafx.scene.Scene;
import javafx.scene.control.Button; import
javafx.stage.Stage;

public class MultipleStageDemo extends Application {
public void start(Stage primaryStage) {

  // Create a scene and place a button in the scene
  Scene scene = new Scene(new Button("OK"), 200, 250);
  primaryStage.setTitle("MyJavaFX"); // Set the stage title
  primaryStage.setScene(scene); // Place the scene in the stage
  primaryStage.show(); // Display the stage

  Stage stage = new Stage(); // Create a new stage
  stage.setTitle("Second Stage"); // Set the stage title
  // Set a scene with a button in the stage
  stage.setScene(new Scene(new Button("New Stage"), 100, 100));
  stage.show(); // Display the stage
}
}
```



Scene can only have one child

How to put multiple elements in a scene?

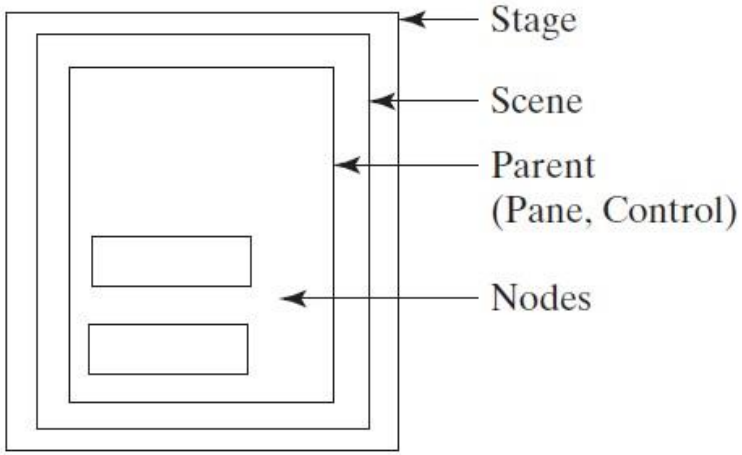
Panes, UI Controls, and Shapes

- When you run MyJavaFX, The button is always centered in the scene and occupies the entire window no matter how you resize it.
  - A better approach is to use **container classes**, called **panes**, for automatically laying out the nodes in a desired location and size.
  - You place nodes inside a pane and then place the pane into a scene.

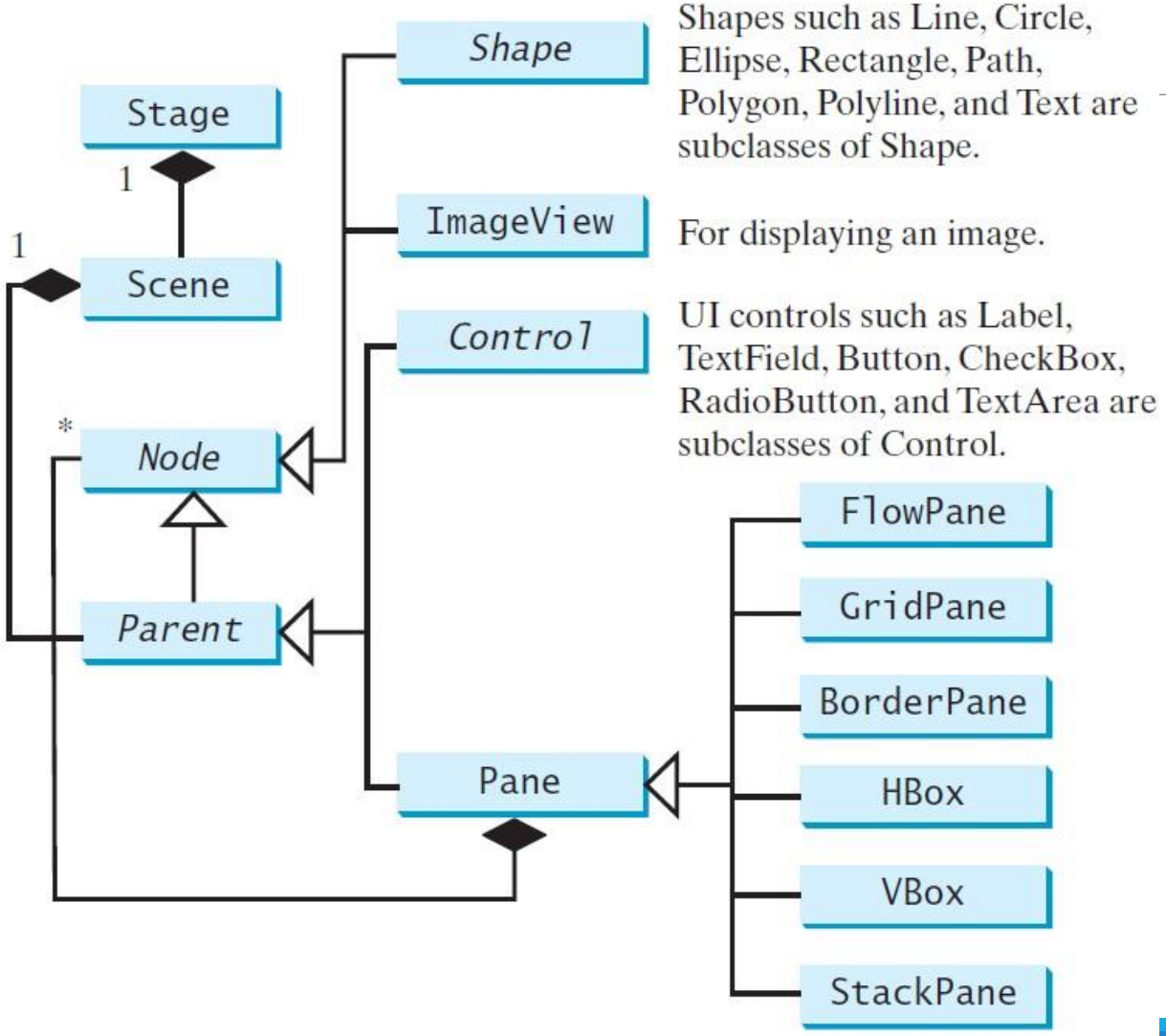
- A *Node* is a visual component such as a **shape**, an **image view**, a **UI control**, or a **pane**.

- **A node can be placed only in one pane**. *Otherwise an exception is thrown*

Panes, UI Controls, and Shapes



(a)



Shapes such as Line, Circle, Ellipse, Rectangle, Path, Polygon, Polyline, and Text are subclasses of Shape.

For displaying an image.

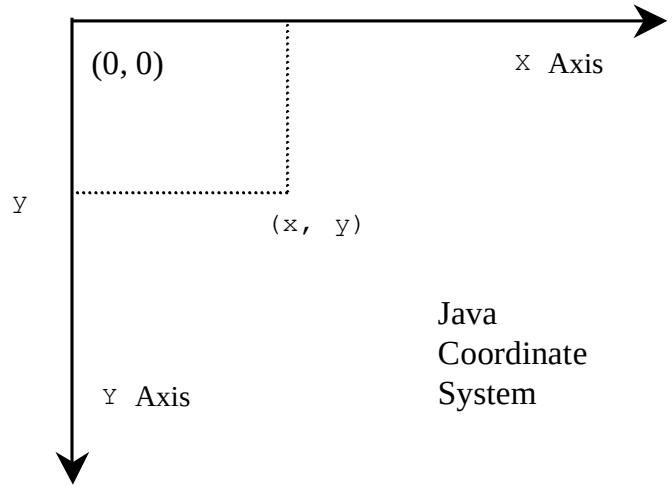
UI controls such as Label, TextField, Button, CheckBox, RadioButton, and TextArea are subclasses of Control.

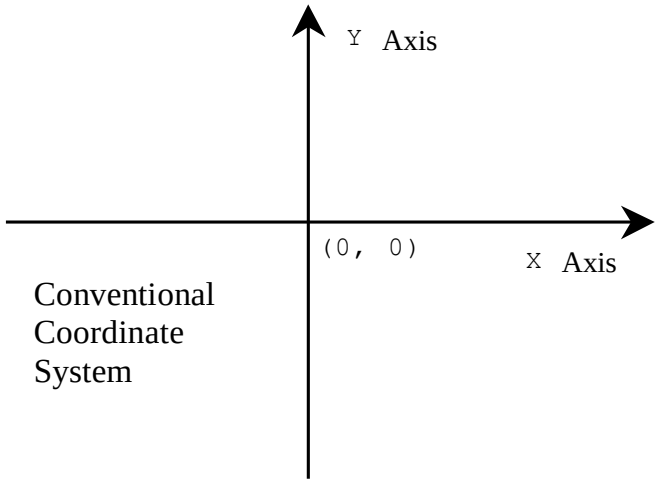
(b)

```
public class ButtonInPane extends Application {

  @Override
  public void start(Stage primaryStage) {

    // Create a scene and place a button in the scene
    StackPane pane = new StackPane();

    pane.getChildren().add(new Button("OK"));

    Scene scene = new Scene(pane, 200, 50);

    primaryStage.setTitle("Button in a pane"); // Set the stage title
    primaryStage.setScene(scene); // Place the scene in the stage
    primaryStage.show(); // Display the stage
  }

}
```

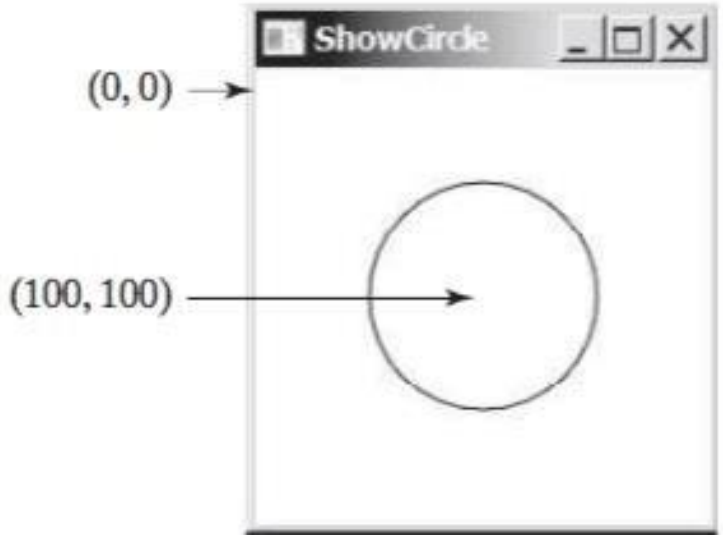
Display a Shape

The following example displays a circle in the center of the pane.

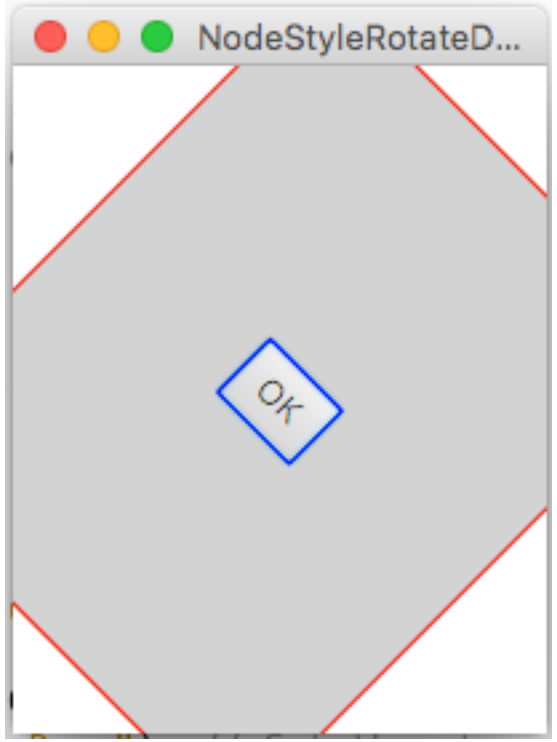




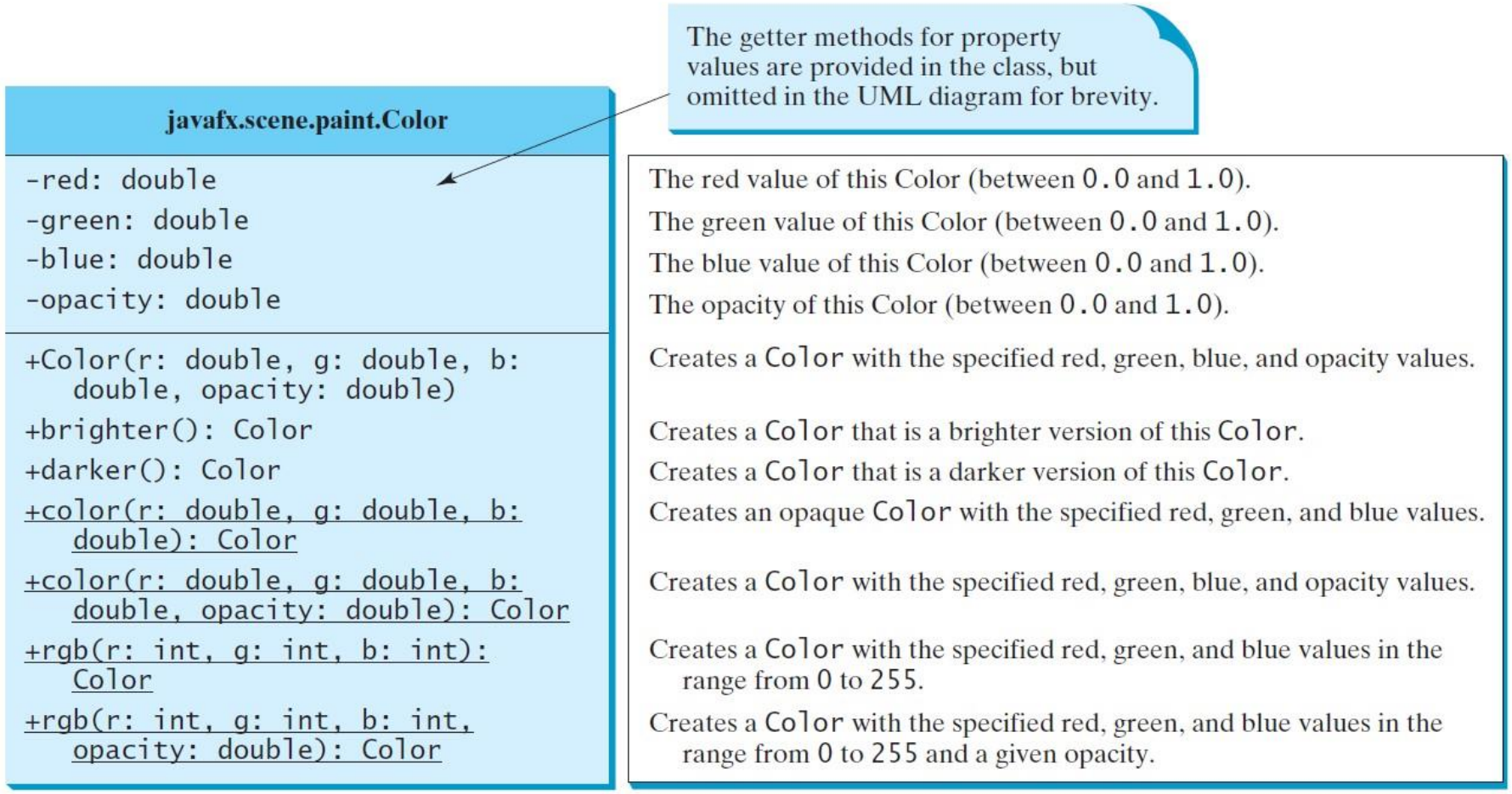
```
public class ShowCircle extends Application {

public void start(Stage primaryStage) {

Circle circle = new Circle();
    circle.setCenterX(100);
    circle.setCenterY(100);
    circle.setRadius(50);
    circle.setStroke(Color.BLACK);
    circle.setFill(null);

    Pane pane = new Pane();
    pane.getChildren().add(circle);

    Scene scene = new Scene(pane, 200, 200);
    primaryStage.setTitle("ShowCircle"); // Set the stage title
    primaryStage.setScene(scene); // Place the scene in the stage
    primaryStage.show(); // Display the stage
}
}
```



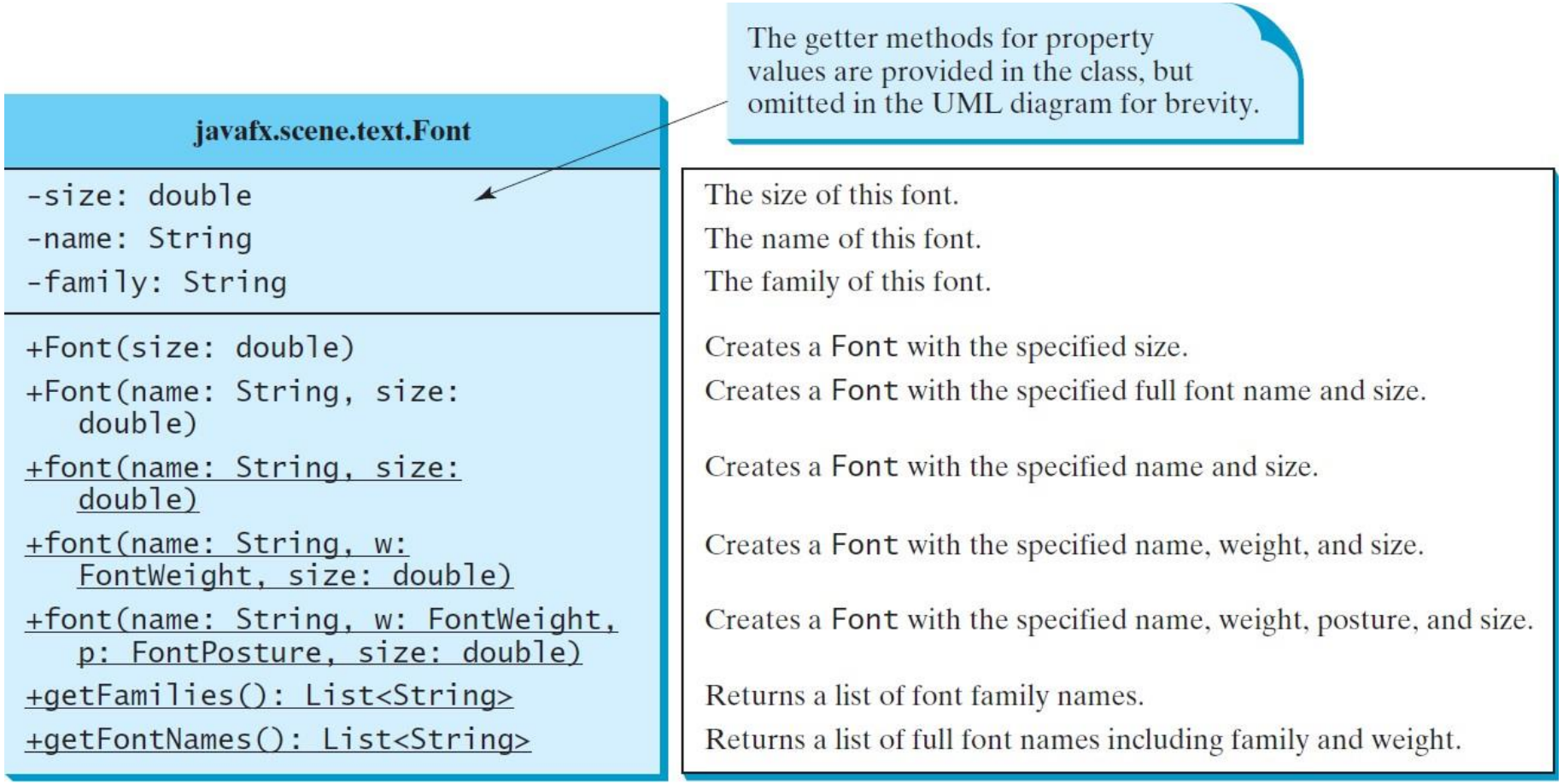
```
public class NodeStyleRotateDemo extends Application {

 @Override public void start(Stage primaryStage) {

   // Create a scene and place a button in the scene
   StackPane pane = new StackPane();
   Button btOK = new Button("OK");
   btOK.setStyle("-fx-border-color: blue;");
   pane.getChildren().add(btOK);

   pane.setRotate(45);
   pane.setStyle( "-fx-border-color: red; -fx-background-color: lightgray;");

   Scene scene = new Scene(pane, 200, 250);
   primaryStage.setTitle("NodeStyleRotateDemo"); // Set the stage title
   primaryStage.setScene(scene); // Place the scene in the stage


   primaryStage.show(); // Display the stage
 }
 }
```



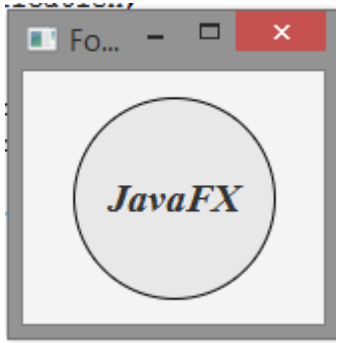
The Color Class



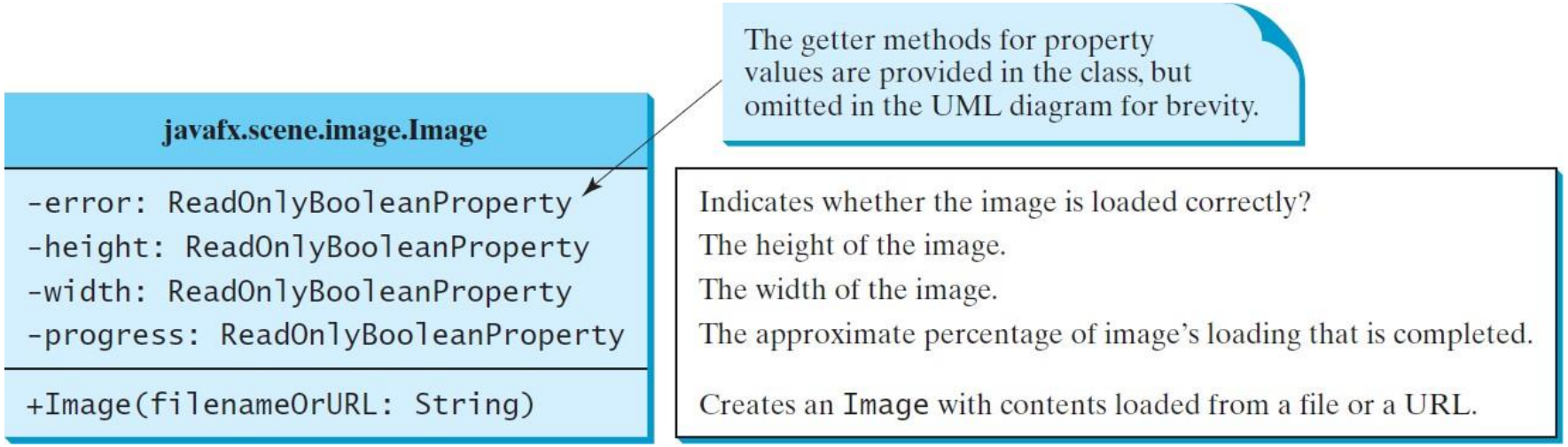
The Font Class



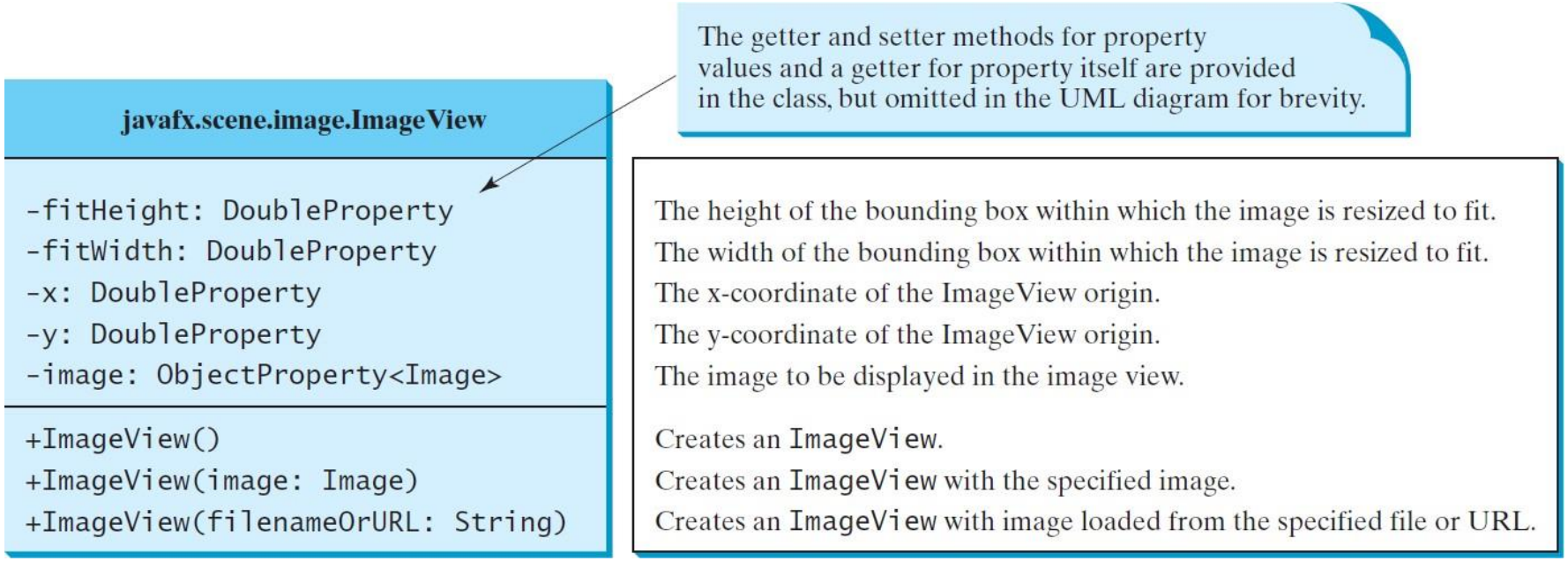
```
public class FontDemo extends Application {

 public void start(Stage primaryStage) {
   Pane pane = new StackPane();
    Circle circle = new Circle();
    circle.setRadius(50);
    circle.setStroke(Color.BLACK);
    circle.setFill(new Color(0.5, 0.5, 0.5, 0.1));
    pane.getChildren().add(circle); // Add circle to the pane

     Label label = new Label("JavaFX");

     label.setFont(Font.font("Times New Roman",
     FontWeight.BOLD, FontPosture.ITALIC, 20));

     pane.getChildren().add(label);
    Scene scene = new Scene(pane);
    primaryStage.setTitle("FontDemo"); // Set the stage title
    primaryStage.setScene(scene); // Place the scene in the stage
    primaryStage.show(); // Display the stage

  }
}
```



The Image Class



The ImageView Class



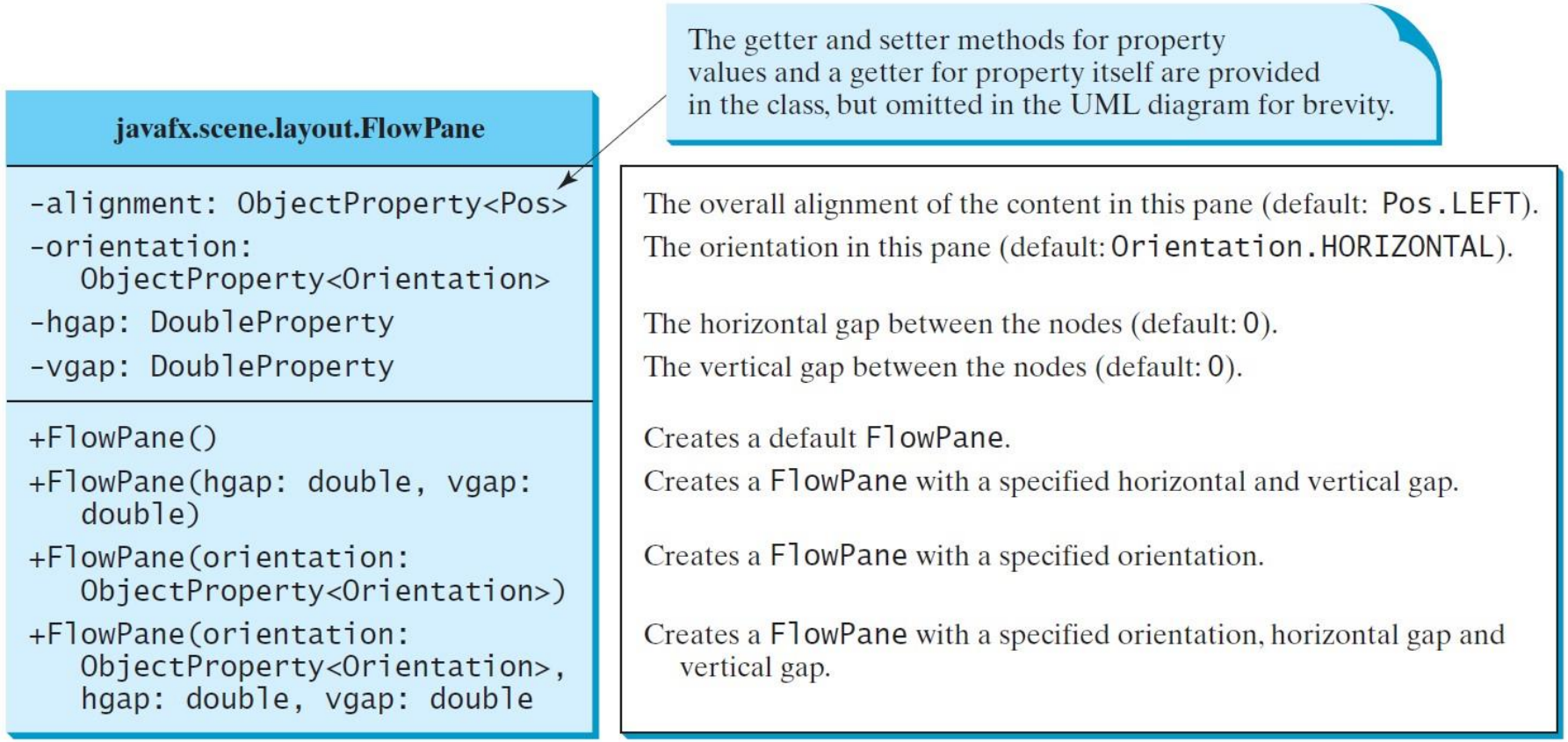
```
public class ShowImage extends Application {

public void start(Stage primaryStage) {

    Pane pane = new HBox(10);

    pane.setPadding(new Insets(5, 5, 5, 5));

    Image image = new Image("image/us.gif");
    pane.getChildren().add(new ImageView(image));

    ImageView imageView2 = new ImageView(image);
    imageView2.setFitHeight(100);
    imageView2.setFitWidth(100);
    pane.getChildren().add(imageView2);

    ImageView imageView3 = new ImageView(image);
    imageView3.setRotate(90);
    pane.getChildren().add(imageView3);

    Scene scene = new Scene(pane);
    primaryStage.setTitle("ShowImage"); // Set the stage title
    primaryStage.setScene(scene); // Place the scene in the stage
    primaryStage.show(); // Display the stage
  }
}
```

Layout Panes

JavaFX provides many types of panes for organizing nodes in a container.

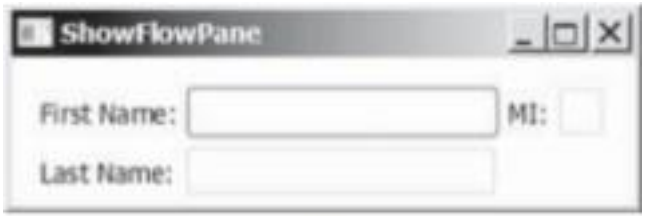
Class	*Description*
`Pane`	Base class for layout panes. It contains the `getChildren()` method for returning a list of nodes in the pane.
`StackPane`	Places the nodes on top of each other in the center of the pane.
`FlowPane`	Places the nodes row-by-row horizontally or column-by-column vertically.
`GridPane`	Places the nodes in the cells in a two-dimensional grid.
`BorderPane`	Places the nodes in the top, right, bottom, left, and center regions.
`HBox`	Places the nodes in a single row.
`VBox`	Places the nodes in a single column.

FlowPane

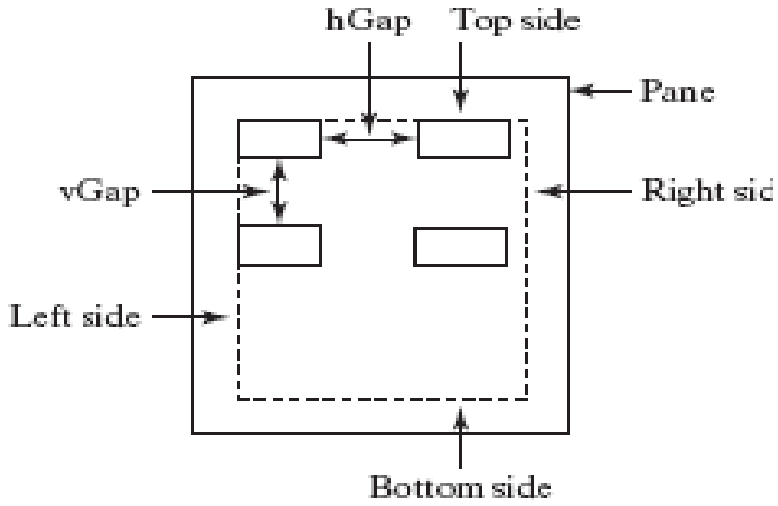

The getter and setter methods for property values and a getter for property itself are provided in the class, but omitted in the UML diagram for brevity.
javafx.scene.layout.FlowPane
-alignment: ObjectProperty<Pos>
-orientation: ObjectProperty<Orientation>
-hgap: DoubleProperty
-vgap: DoubleProperty
+FlowPane()
+FlowPane(hgap: double, vgap: double)
+FlowPane(orientation: ObjectProperty<Orientation>)
+FlowPane(orientation: ObjectProperty<Orientation>, hgap: double, vgap: double
The overall alignment of the content in this pane (default: Pos.LEFT).
The orientation in this pane (default: Orientation.HORIZONTAL).
The horizontal gap between the nodes (default: 0).
The vertical gap between the nodes (default: 0).
Creates a default FlowPane.
Creates a FlowPane with a specified horizontal and vertical gap.
Creates a FlowPane with a specified orientation.
Creates a FlowPane with a specified orientation, horizontal gap and vertical gap.

```
public class ShowFlowPane extends Application {

  @Override
  public void start(Stage primaryStage) {

    // Create a pane and set its properties
    FlowPane pane = new FlowPane();
    pane.setPadding(new Insets(11, 12, 13, 14));
    pane.setHgap(5);
    pane.setVgap(5);

    // Place nodes in the pane
    pane.getChildren().addAll(new Label("First Name:"),
    new TextField(), new Label("MI:"));

    TextField tfMi = new TextField();
    tfMi.setPrefColumnCount(1);
    pane.getChildren().addAll(tfMi, new Label("Last Name:"),
    new TextField());

    // Create a scene and place it in the stage
    Scene scene = new Scene(pane, 200, 250);
    primaryStage.setTitle("ShowFlowPane"); // Set the stage title
    primaryStage.setScene(scene); // Place the scene in the stage
    primaryStage.show(); // Display the stage
  }
  }
```

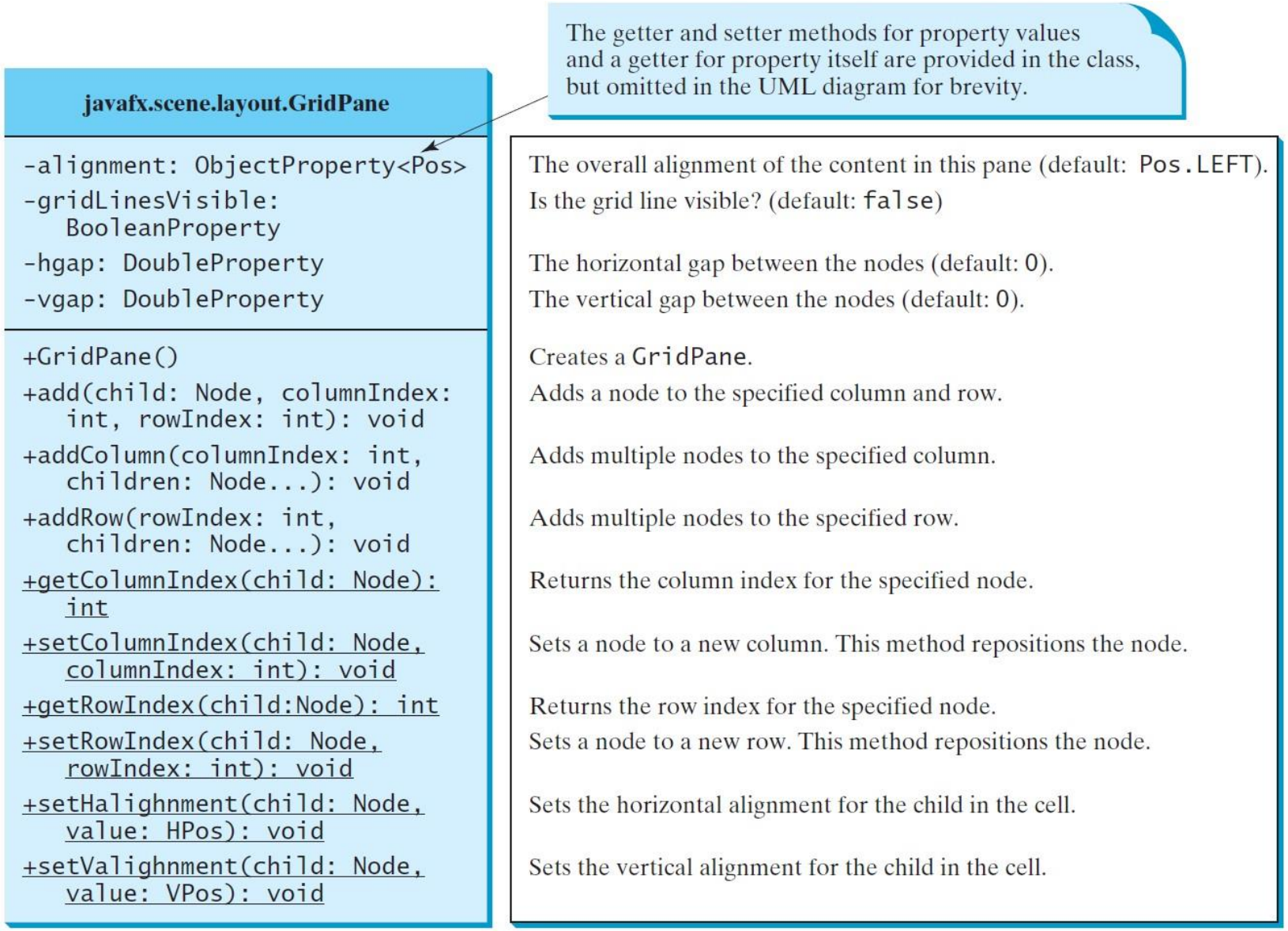
(a)



(b)

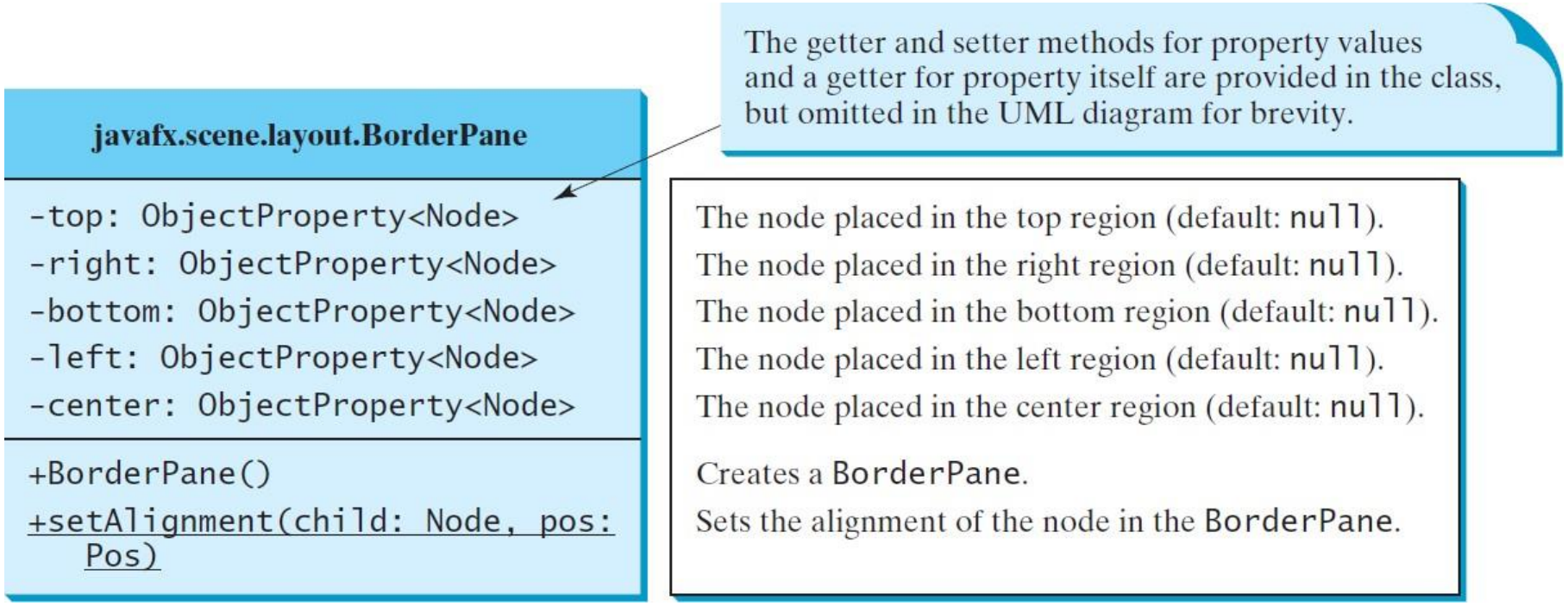


GridPane

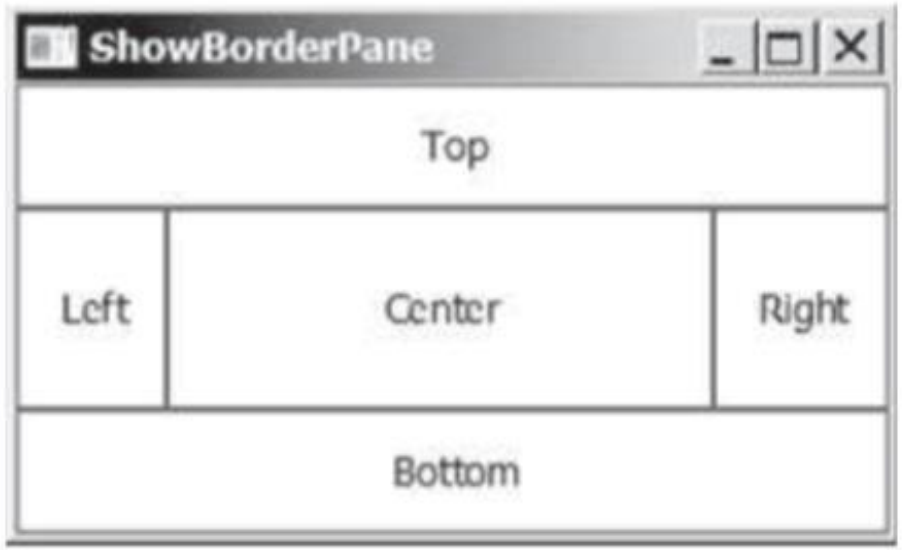

javafx.scene.layout.GridPane
-alignment: ObjectProperty<Pos>
-gridLinesVisible: BooleanProperty
-hgap: DoubleProperty
-vgap: DoubleProperty
+GridPane()
+add(child: Node, columnIndex: int, rowIndex: int): void
+addColumn(columnIndex: int, children: Node...): void
+addRow(rowIndex: int, children: Node...): void
+getColumnIndex(child: Node): int
+setColumnIndex(child: Node, columnIndex: int): void
+getRowIndex(child:Node): int
+setRowIndex(child: Node, rowIndex: int): void
+setHalighnment(child: Node, value: HPos): void
+setValighnment(child: Node, value: VPos): void
The getter and setter methods for property values and a getter for property itself are provided in the class, but omitted in the UML diagram for brevity.
The overall alignment of the content in this pane (default: Pos.LEFT).
Is the grid line visible? (default: false)
The horizontal gap between the nodes (default: 0).
The vertical gap between the nodes (default: 0).
Creates a GridPane.
Adds a node to the specified column and row.
Adds multiple nodes to the specified column.
Adds multiple nodes to the specified row.
Returns the column index for the specified node.
Sets a node to a new column. This method repositions the node.
Returns the row index for the specified node.
Sets a node to a new row. This method repositions the node.
Sets the horizontal alignment for the child in the cell.
Sets the vertical alignment for the child in the cell.

```
public class ShowGridPane extends Application {

  @Override
  public void start(Stage primaryStage) {

    // Create a pane and set its properties
    GridPane pane = new GridPane();

    pane.setAlignment(Pos.CENTER);
    pane.setPadding(new Insets(11.5, 12.5, 13.5, 14.5));
    pane.setHgap(5.5);
    pane.setVgap(5.5);

    // Place nodes in the pane
    pane.add(new Label("First Name:"), 0, 0);
    pane.add(new TextField(), 1, 0);
    pane.add(new Label("MI:"), 0, 1);
    pane.add(new TextField(), 1, 1);
    pane.add(new Label("Last Name:"), 0, 2);
    pane.add(new TextField(), 1, 2);
    Button btAdd = new Button("Add Name");
    pane.add(btAdd, 1, 3);

    GridPane.setHalignment(btAdd, HPos.RIGHT);

    // Create a scene and place it in the stage
    Scene scene = new Scene(pane);
    primaryStage.setTitle("ShowGridPane"); // Set the stage title
    primaryStage.setScene(scene); // Place the scene in the stage
    primaryStage.show(); // Display the stage
  }
  }
```

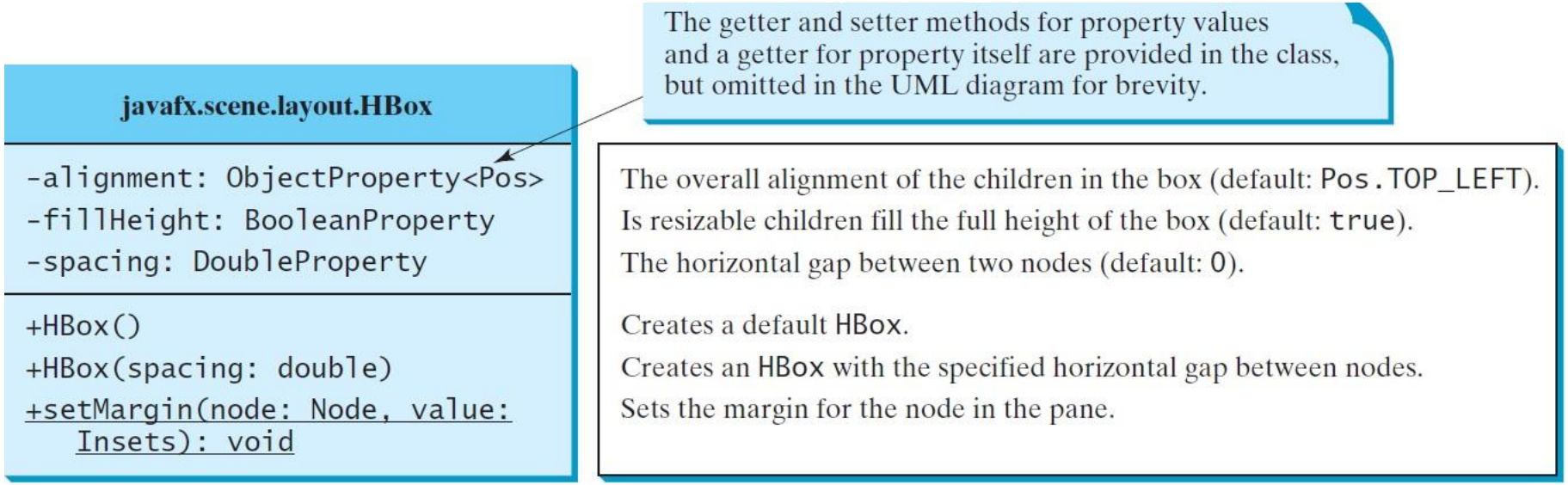
BorderPane


javafx.scene.layout.BorderPane
-top: ObjectProperty<Node>
-right: ObjectProperty<Node>
-bottom: ObjectProperty<Node>
-left: ObjectProperty<Node>
-center: ObjectProperty<Node>
+BorderPane()
+setAlignment(child: Node, pos: Pos)
The getter and setter methods for property values and a getter for property itself are provided in the class, but omitted in the UML diagram for brevity.
The node placed in the top region (default: null).
The node placed in the right region (default: null).
The node placed in the bottom region (default: null).
The node placed in the left region (default: null).
The node placed in the center region (default: null).
Creates a BorderPane.
Sets the alignment of the node in the BorderPane.

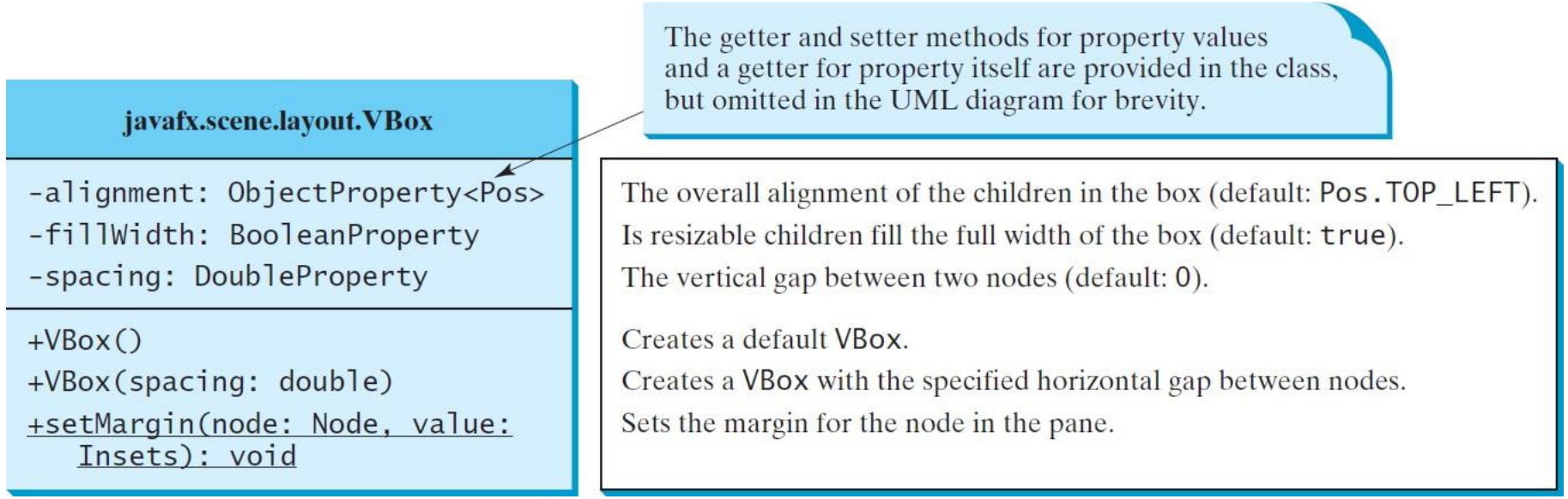
```
public class ShowBorderPane extends Application {

  @Override
  public void start(Stage primaryStage) {
    // Create a border pane
    BorderPane pane = new BorderPane();

    // Place nodes in the pane
    pane.setTop(new CustomPane("Top"));
    pane.setRight(new CustomPane("Right"));
    pane.setBottom(new CustomPane("Bottom"));
    pane.setLeft(new CustomPane("Left"));
    pane.setCenter(new CustomPane("Center"));

    // Create a scene and place it in the stage
    Scene scene = new Scene(pane);
    primaryStage.setTitle("ShowBorderPane"); // Set the stage title
    primaryStage.setScene(scene); // Place the scene in the stage
    primaryStage.show(); // Display the stage
  }
}

// Define a custom pane to hold a label in the center of the pane
class CustomPane extends StackPane {

  public CustomPane(String title) {

    getChildren().add(new Label(title));
   setStyle("-fx-border-color: red");
    setPadding(new Insets(11.5, 12.5, 13.5, 14.5));

    }
    }
```



HBox

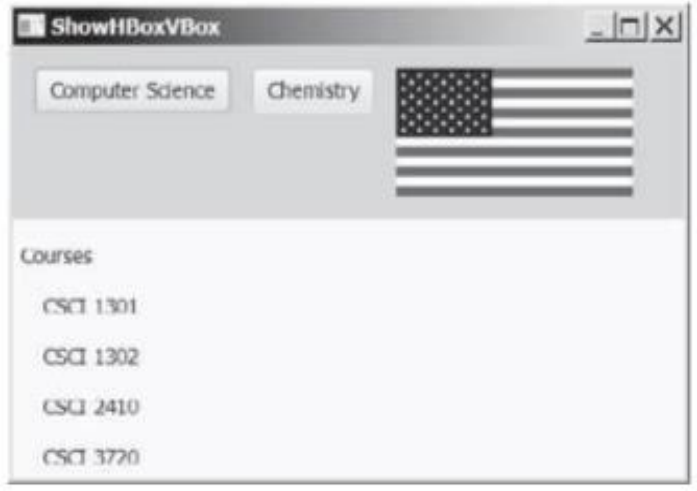

javafx.scene.layout.HBox
-alignment: ObjectProperty<Pos>
-fillHeight: BooleanProperty
-spacing: DoubleProperty
+HBox()
+HBox(spacing: double)
+setMargin(node: Node, value: Insets): void
The getter and setter methods for property values and a getter for property itself are provided in the class, but omitted in the UML diagram for brevity.
The overall alignment of the children in the box (default: Pos.TOP_LEFT).
Is resizable children fill the full height of the box (default: true).
The horizontal gap between two nodes (default: 0).
Creates a default HBox.
Creates an HBox with the specified horizontal gap between nodes.
Sets the margin for the node in the pane.

VBox


javafx.scene.layout.VBox
-alignment: ObjectProperty<Pos>
-fillWidth: BooleanProperty
-spacing: DoubleProperty
+VBox()
+VBox(spacing: double)
+setMargin(node: Node, value: Insets): void
The getter and setter methods for property values and a getter for property itself are provided in the class, but omitted in the UML diagram for brevity.
The overall alignment of the children in the box (default: Pos.TOP_LEFT).
Is resizable children fill the full width of the box (default: true).
The vertical gap between two nodes (default: 0).
Creates a default VBox.
Creates a VBox with the specified horizontal gap between nodes.
Sets the margin for the node in the pane.

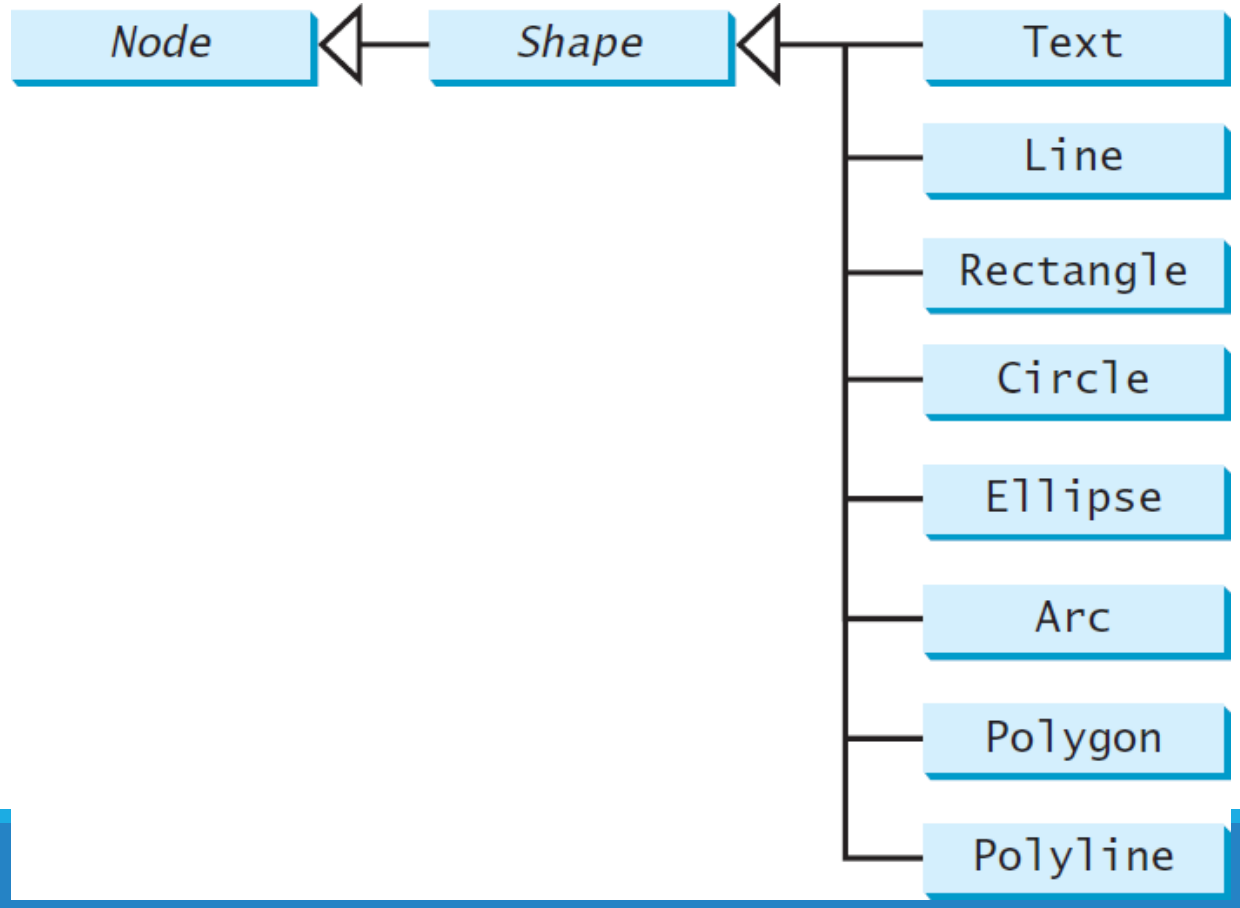
```
13  public class ShowHBoxVBox extends Application {
14    @Override
15    public void start(Stage primaryStage) {
16      // Create a border pane
17      BorderPane pane = new BorderPane();
18
19      // Place nodes in the pane
20      pane.setTop(getHBox());
21      pane.setLeft(getVBox());
22
23      // Create a scene and place it in the stage
24      Scene scene = new Scene(pane);
25      primaryStage.setTitle("ShowHBoxVBox");
26      primaryStage.setScene(scene)
27      primaryStage.show(); // Display the stage
28    }
```

```
30    private HBox getHBox() {
31      HBox hBox = new HBox(15);
32      hBox.setPadding(new Insets(15, 15, 15, 15));
33      hBox.setStyle("-fx-background-color: gold");
34      hBox.getChildren().add(new Button("Computer Science"));
35      hBox.getChildren().add(new Button("Chemistry"));
36      ImageView imageView = new ImageView(new Image("image/us.gif"));
37      hBox.getChildren().add(imageView);
38      return hBox;
39    }
40
41    private VBox getVBox() {
42      VBox vBox = new VBox(15);
43      vBox.setPadding(new Insets(15, 5, 5, 5));
44      vBox.getChildren().add(new Label("Courses"));
45
46      Label[] courses = {new Label("CSCI 1301"), new Label("CSCI 1302"),
47        new Label("CSCI 2410"), new Label("CSCI 3720")};

49      for (Label course: courses) {
50        VBox.setMargin(course, new Insets(0, 0, 0, 15));
51        vBox.getChildren().add(course);
52      }
53
54      return vBox;
55    }
56  }
```

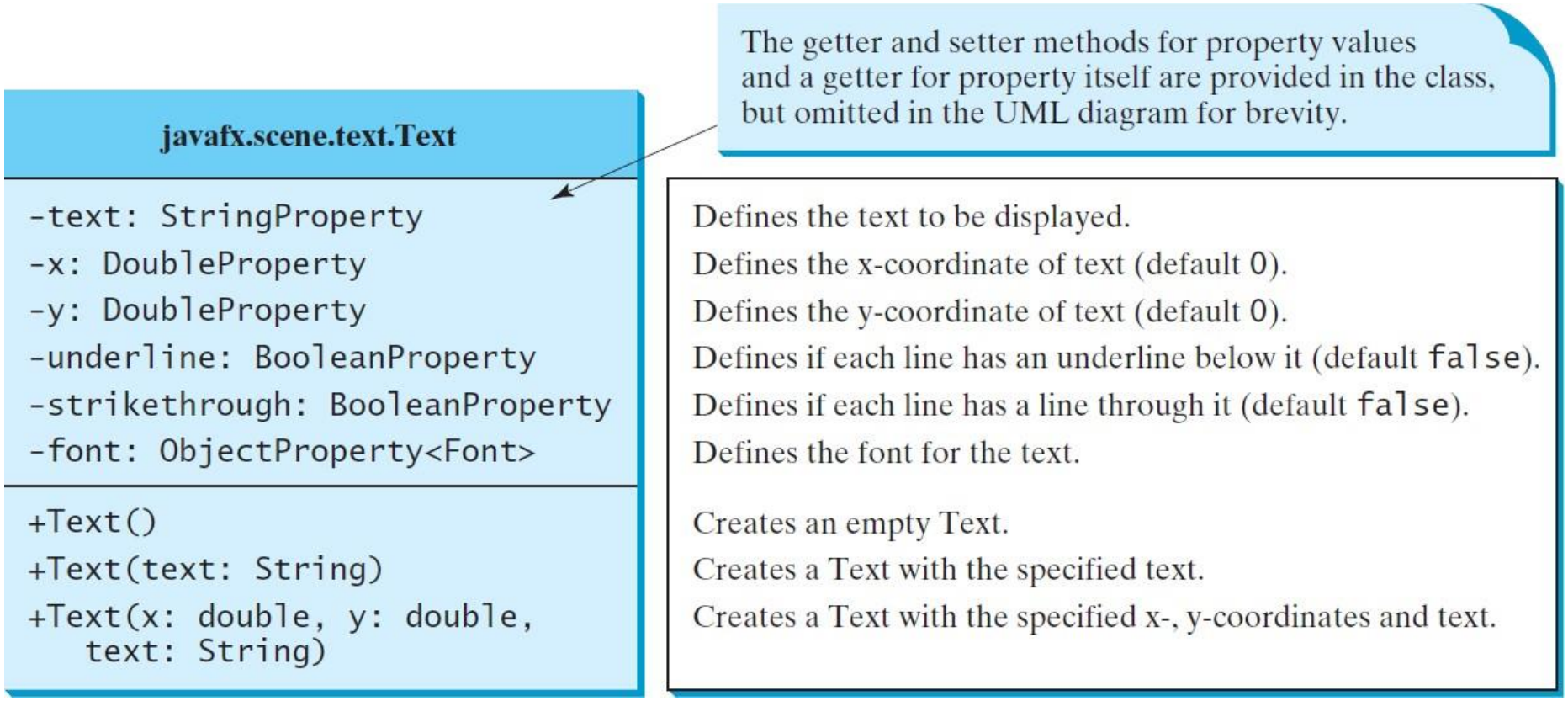


Shapes

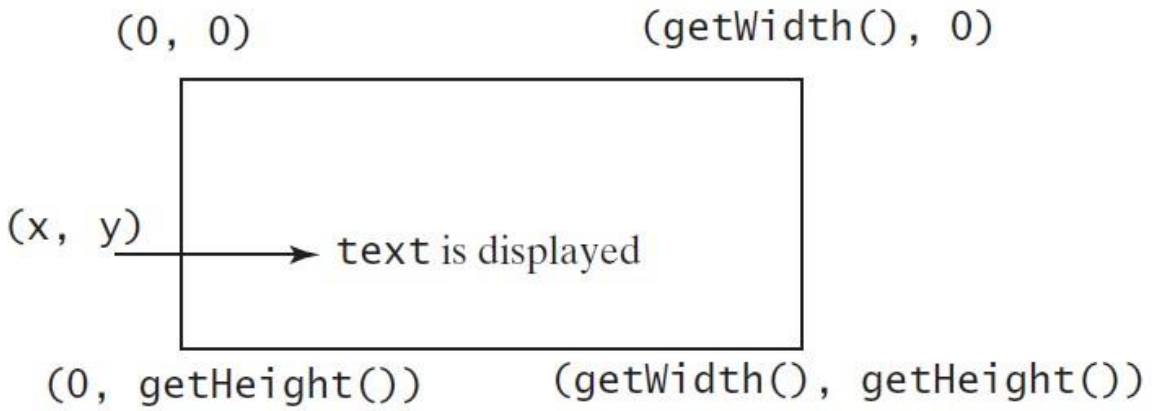
JavaFX provides many shape classes for drawing texts, lines, circles, rectangles, ellipses, arcs, polygons, and polylines.



Text


javafx.scene.text.Text
-text: StringProperty
-x: DoubleProperty
-y: DoubleProperty
-underline: BooleanProperty
-strikethrough: BooleanProperty
-font: ObjectProperty<Font>
+Text()
+Text(text: String)
+Text(x: double, y: double, text: String)
The getter and setter methods for property values and a getter for property itself are provided in the class, but omitted in the UML diagram for brevity.
Defines the text to be displayed.
Defines the x-coordinate of text (default 0).
Defines the y-coordinate of text (default 0).
Defines if each line has an underline below it (default false).
Defines if each line has a line through it (default false).
Defines the font for the text.
Creates an empty Text.
Creates a Text with the specified text.
Creates a Text with the specified x-, y-coordinates and text.

Text Example



(a) Text(x, y, text)

(b) *Three* Text *objects are displayed*

```
public class ShowText extends Application {
  @Override
  public void start(Stage primaryStage) {

    // Create a pane to hold the texts
    Pane pane = new Pane();
    pane.setPadding(new Insets(5, 5, 5, 5));
    Text text1 = new Text(20, 20, "Programming is fun");
    text1.setFont(Font.font("Courier", FontWeight.BOLD, FontPosture.ITALIC, 15));
    pane.getChildren().add(text1);

    Text text2 = new Text(60, 60, "Programming is fun\nDisplay text");
    pane.getChildren().add(text2);

    Text text3 = new Text(10, 100, "Programming is fun\nDisplay text");
    text3.setFill(Color.RED);
    text3.setUnderline(true);
    text3.setStrikethrough(true);
    pane.getChildren().add(text3);

    // Create a scene and place it in the stage
    Scene scene = new Scene(pane);
    primaryStage.setTitle("ShowText"); // Set the stage title
    primaryStage.setScene(scene); // Place the scene in the stage
    primaryStage.show(); // Display the stage
  }
  }
```

(b) *Three* Text *objects are displayed*

Line

The getter and setter methods for property values and a getter for property itself are provided in the class, but omitted in the UML diagram for brevity.

javafx.scene.shape.Line

```
-startX: DoubleProperty
-startY: DoubleProperty
-endX: DoubleProperty
-endY: DoubleProperty
```

```
+Line()
+Line(startX: double, startY:
   double, endX: double, endY:
   double)
```

The x-coordinate of the start point.

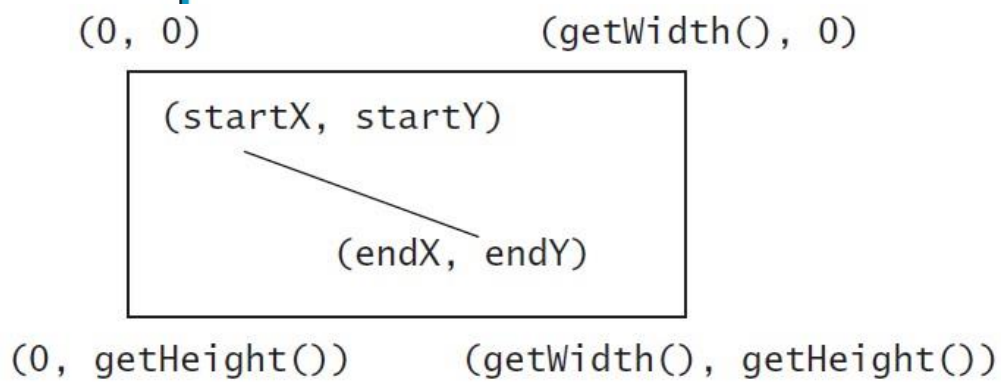
The y-coordinate of the start point.

The x-coordinate of the end point.

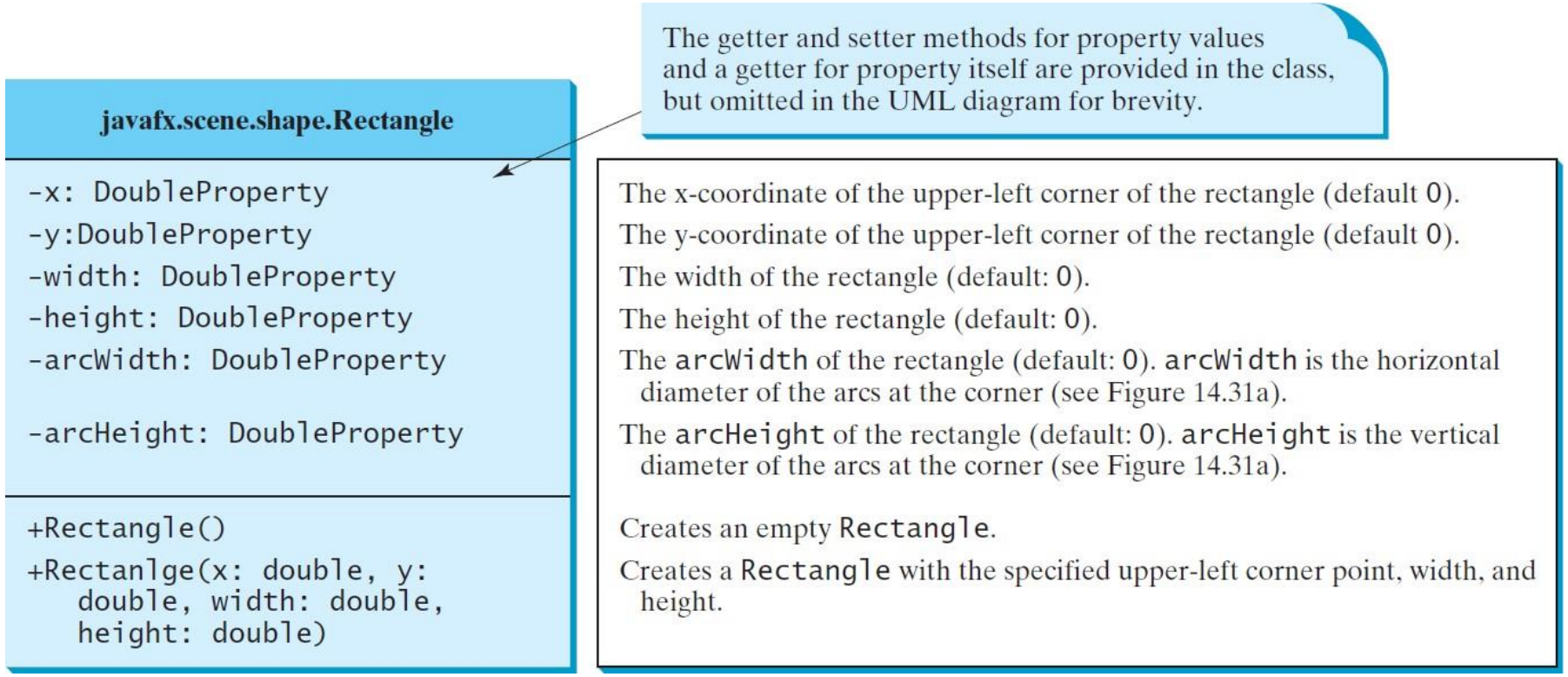
The y-coordinate of the end point.

Creates an empty `Line`.

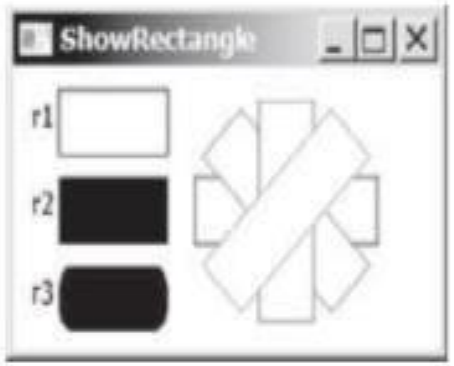
Creates a `Line` with the specified starting and ending points.

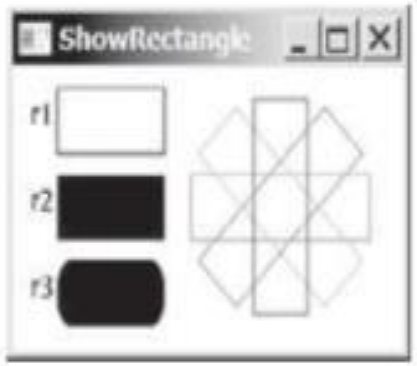


Rectangle


The getter and setter methods for property values and a getter for property itself are provided in the class, but omitted in the UML diagram for brevity.
javafx.scene.shape.Rectangle
-x: DoubleProperty
-y:DoubleProperty
-width: DoubleProperty
-height: DoubleProperty
-arcWidth: DoubleProperty
-arcHeight: DoubleProperty
+Rectangle()
+Rectanlge(x: double, y: double, width: double, height: double)
The x-coordinate of the upper-left corner of the rectangle (default 0).
The y-coordinate of the upper-left corner of the rectangle (default 0).
The width of the rectangle (default: 0).
The height of the rectangle (default: 0).
The arcWidth of the rectangle (default: 0). arcWidth is the horizontal diameter of the arcs at the corner (see Figure 14.31a).
The arcHeight of the rectangle (default: 0). arcHeight is the vertical diameter of the arcs at the corner (see Figure 14.31a).
Creates an empty Rectangle.
Creates a Rectangle with the specified upper-left corner point, width, and height.

```
public class ShowRectangle extends Application {
  @Override
  public void start(Stage primaryStage) {

    Pane pane = new Pane();
    Rectangle r1 = new Rectangle(25, 10, 60, 30);
    r1.setStroke(Color.BLACK);
    r1.setFill(Color.WHITE);
    Rectangle r2 = new Rectangle(25, 50, 60, 30);
    Rectangle r3 = new Rectangle(25, 90, 60, 30);
    r3.setArcWidth(15);
    r3.setArcHeight(25);
    // Create a group and add nodes to the group
    Group group = new Group();
    group.getChildren().addAll(new Text(10, 27, "r1"), r1,  new Text(10, 67, "r2"), r2, new Text(10, 107, "r3"), r3);

    for (int i = 0; i < 4; i++) {
      Rectangle r = new Rectangle(100, 50, 100, 30);
      r.setRotate(i * 360 / 8);
      r.setStroke(Color.color(Math.random(), Math.random(), Math.random()));
      r.setFill(Color.WHITE);
      group.getChildren().add(r);
    }

    // Create a scene and place it in the stage
    Scene scene = new Scene(new Pane(group), 250, 150);
    primaryStage.setTitle("ShowRectangle"); // Set the stage title
    primaryStage.setScene(scene); // Place the scene in the stage
    primaryStage.show(); // Display the stage

  }
  }
```





Thank You