13.2 Laying Out Nodes in a Scene Graph

A layout determines the size and positioning of nodes in the scene graph.

Node Size

In general, a node's size should *not* be defined *explicitly*. Doing so often creates a design that looks pleasing when it first loads, but deteriorates when the app is resized or the content updates. In addition to the `width` and `height` properties, most JavaFX nodes have the properties `prefWidth`, `prefHeight`, `minWidth`, `minHeight`, `maxWidth` and `maxHeight` that specify a node's *range* of acceptable sizes as it's laid out within its parent node:

- The minimum size properties specify a node's smallest allowed size in points.
- The maximum size properties specify a node's largest allowed size in points.
- The preferred size properties specify a node's preferred width and height that should be used by the layout in most cases.

Node Position and Layout

Panes

A node's position should be defined *relative* to its parent node and the other nodes in its parent. JavaFX **layout panes** are container nodes that arrange their child nodes in a scene graph relative to one another, based on their sizes and positions. Child nodes are controls, other layout panes, shapes and more.

Most JavaFX layout panes use *relative positioning*—if a layout-pane node is resized, it adjusts its children's sizes and positions accordingly, based on their preferred, minimum and maximum sizes. Figure 13.1 describes each of the JavaFX layout panes, including those presented in Chapter 12. In this chapter, we'll use `Pane`, `BorderPane`, `GridPane` and `VBox` from the `javafx.scene.layout` package.

Layout	Description
`AnchorPane`	Enables you to set the position of child nodes relative to the pane's edges. Resizing the pane does not alter the layout of the nodes.
`BorderPane`	Includes five areas—top, bottom, left, center and right—where you can place nodes. The top and bottom regions fill the `BorderPane`'s width and are vertically sized to their children's preferred heights. The left and right regions fill the `BorderPane`'s height and are horizontally sized to their children's preferred widths. The center area occupies all of the `BorderPane`'s remaining space. You might use the different areas for tool bars, navigation, a main content area, etc.
`FlowPane`	Lays out nodes consecutively—either horizontally or vertically. When the boundary for the pane is reached, the nodes wrap to a new line in a horizontal `FlowPane` or a new column in a vertical `FlowPane`.
`GridPane`	Creates a flexible grid for laying out nodes in rows and

	columns.
`Pane`	The base class for layout panes. This can be used to position nodes at fixed locations—known as absolute positioning.
`StackPane`	Places nodes in a stack. Each new node is stacked atop the previous node. You might use this to place text on top of images, for example.
`TilePane`	A horizontal or vertical grid of equally sized tiles. Nodes that are tiled horizontally wrap at the `TilePane`'s width. Nodes that are tiled vertically wrap at the `TilePane`'s height.
`HBox`	Arranges nodes horizontally in one row.
`VBox`	Arranges nodes vertically in one column.

Fig. 13.1

JavaFX layout panes.