Keanu Foltz

3/2/25

CSD 402

Module 11.2

HBox and VBox

The HBox and VBox are JavaFX layout containers that have similar uses. Both are layout panes that are included in the javafx.scene.layout package, since they inherit from the Pane class they have many attributes and methods related to layout management. We use the HBox and VBox layout panes to organize and arrange User Interface elements in our GUI. I will explain more about both the HBox and VBox layout panes as well as how to apply them in JavaFX.

The HBox layout pane allows us to arrange nodes horizontally, in a single row. Any time it would be ideal to make a horizontal grouping of controls such as toolbars or menus, it is recommended that we use the HBox layout. The HBox layout is very customizable, you are able to adjust the alignment and spacing of the nodes arranged by the HBox.

The VBox layout pane is similar to the HBox layout pane, however, it allows us to arrange nodes vertically rather than horizontally. There are many ways that you can apply the VBox pane in JavaFX to organize nodes in a vertical column, you may see this commonly used for vertical groupings of controls such as vertical menus and forms. The VBox layout pane also allows for a good amount of customization when it comes to spacing and alignment of the nodes arranged by the VBox.

The HBox and VBox can also be combined to create UI layouts that have elements you would like to arrange both vertically and horizontally. HBox can be nested in VBox and VBox can be nested in HBox to create a mix of vertical and horizontal layouts. Depending on the layout that we want, whether we nest the HBox in the VBox or the VBox in the HBox will change. Whatever alignment we want will have the other layout pane nested inside of it, for example, if we want Horizontal alignment where the nodes are stacked vertically, we would nest the VBox in the HBox and vice versa.

There are many advantages to using VBox and HBox. The main advantage is that it makes managing the layouts a lot simpler while also giving us a good amount of customization. For example, with HBox and VBox, the spacing between nodes can be set in the constructor itself or by using the setSpacing() method. By setting the spacing between nodes or elements, you are able to get consistent alignment without having to worry about overlap or elements being too close to each other. You can also do things like set the padding so that there is space between elements and the edge of the layout for a cleaner look.

Another benefit to using VBox and HBox is that both support CSS styling. This allows for much more customization of the layout and the nodes. This includes anything from spacing to margins, to background color and more giving us a bigger range of customization and ability to make a polished UI. We can apply custom CSS styling with setStyle to any nodes in VBox and HBox.

One more big advantage of HBox and VBox is that both layout panes will resize and adjust based on the window size. This means that even if the window size changes, the layout panes will redistribute the space among the nodes. When creating something that could be used across different devices, this can be an important aspect to consider for making the UI user friendly.

To conclude, the layout panes VBox and HBox are very simple but useful tools from JavaFX. VBox and HBox allow us to arrange and organize UI elements/ nodes easily while still giving the layout a clean and polished look. There are many practical applications that we can use VBox and HBox for, it all just depends on what kind of layout we are looking to create to organize our UI elements. By using the VBox and HBox layout panes together we can make our UIs more organized and easy to navigate for the user.

References:

<https://docs.oracle.com/javafx/2/layout/builtin_layouts.htm>

<https://www.geeksforgeeks.org/javafx-vbox-class/>

<https://www.geeksforgeeks.org/javafx-hbox-class/>

<https://www.javacodegeeks.com/2012/07/javafx-20-layout-panes-hbox-and-vbox.html>