Brian Preston

CSD 402

March 2, 2025

**Module 11, Programming Assignment**

JavaFX is a powerful framework used for developing graphical user interfaces or (GUIs) in Java. I have learned that it also helps provide a detailed set of APIs that allow developers to create interactive applications easily. As I have learned, JavaFX follows a structured approach to GUI development using the Model-View-Controller (MVC) pattern, which helps in separating the application's logic from its user interface components.

According to Natasha Ferguson, "JavaFX is a set of packages and APIs for developing programs with graphical user interfaces, 3D graphics, etc. A graphical user interface, or GUI, enables the user to interface with a program using graphical components, such as windows, buttons, text boxes, etc." (Ferguson, N.2022).  This quote really helped me understand why JavaFX is important for GUI development—it provides the necessary tools to build modern, responsive applications with visual elements that users can interact with.

A key aspect of JavaFX that I have learned is the Stage and Scene model. Every JavaFX application must have at least one Stage (which serves as the main window) and one Scene (which contains all the UI elements). This structure makes it easy to manage different screens within an application, as we can simply switch scenes without having to create a new stage. This is an efficient way to handle UI transitions, similar to how a play might change scenes on a single stage rather than switching to a completely new theater.

**The Importance of Layout Management in JavaFX**

One of the most important aspects of JavaFX development is layout management, which controls how UI elements are arranged on the screen. Proper layout management ensures that components are positioned correctly, scale properly on different screen sizes, and maintain a clean, organized appearance. In JavaFX, we have several types of layout containers such as HBox, VBox, GridPane, and BorderPane. Each layout has its own purpose and is used depending on how we want to arrange components in our UI. From my experience, choosing the right layout can make or break an application's design and usability. If the layout is not structured correctly, elements may overlap or appear disorganized. For this paper, I am focusing on two of the simplest and most commonly used layouts: HBox and VBox.

**JavaFX HBox**

**What is HBox?**

A JavaFX HBox is a layout container that arranges UI components horizontally in a single row. This means that when we add elements like buttons or labels to an HBox, they will be placed side by side rather than stacked on top of each other. Personally I have practiced this and feel that this layout is beneficial when designing toolbars, navigation menus, or grouping buttons.

**Code Example:**

A computer screen shot of a program code

AI-generated content may be incorrect.

I feel that **HBox** is a great choice when you want a simple way to align items side by side. For example, in a settings window, you might have labels on the left and input fields on the right, all arranged in an HBox. I also believe that using HBox makes applications more intuitive because it follows the natural left-to-right reading pattern in many languages.

**JavaFX VBox**

**What is VBox?**

A JavaFX VBox is a layout container that arranges UI components vertically, stacking them in a single column. This layout is useful for forms, lists, and any UI where elements should be arranged from top to bottom.

Code Example:

A screenshot of a computer code

AI-generated content may be incorrect.

**Why Use VBox?**

I believe VBox is an excellent choice when designing forms, login pages, or any UI that requires a clear vertical structure. It helps maintain an organized layout and ensures elements are properly spaced.

According to Harsh, "Vbox is used as the layout in our application and the children of VBox are arranged in a vertical column." This is exactly why I find VBox useful—it simplifies UI design and makes it easy to structure elements in a logical order.

**When to Use HBox vs. VBox?**

* Use HBox when you need to arrange components in a single row. Example: A horizontal menu bar.
* Use VBox when you need to stack elements vertically. Example: A login form with labels, text fields, and a submit button.

I have learned that choosing the right layout depends on the structure of your UI. If I ever need elements placed side by side, I know HBox is the best choice. If I need elements stacked, then VBox is my go-to option.

JavaFX is an essential framework for building interactive applications with rich GUIs. Layout management plays a crucial role in structuring UI components effectively. Both HBox and VBox are fundamental layout managers in JavaFX, helping developers organize UI elements efficiently. I now understand that choosing the right layout is important for improving user experience, readability, and overall design.

References:

GeeksforGeeks. (2018, September 7). *JavaFX: HBox class*. <https://www.geeksforgeeks.org/javafx-hbox-class/>

GeeksforGeeks. (2018a, September 6). *JavaFX: Vbox class*. <https://www.geeksforgeeks.org/javafx-vbox-class/>

Harsh. (2024, March 26). *JavaFX VBox*. Code 360 by coding ninjas. <https://www.naukri.com/code360/library/javafx-vbox>

Ferguson, N. (2022, November 9). *JavaFX Basics*. Medium. <https://medium.com/@teamtechsis/javafx-basics-8fea407f060e>

*W3schools.com*. Introduction to Java. <https://www.w3schools.com/java/java_intro.asp#gsc.tab=0>