Truman Forey

CSD 402

3/2/2025

This paper has an associated .java file, called “JavaFXDemonstrations.java.” Please look to that file when the paper asks to refer certain Demos.

***JavaFX – Hbox and VBox***

JavaFX is a powerful suite of tools and a platform that works inside the Java programming language. It’s primary usage is to create graphic user interfaces (or GUIs) for any program that is compatible with Java, and it has become an industry standard for GUIs ever since it was created in December 2008. The way JavaFX functions is that it creates interactive elements in the form Classes, which can be manipulated by Java code. Some of the most fundamental and practical Classes in the JavaFX library are the HBox and VBox Classes. These create simple, baseline elements that other JavaFX Classes and Objects can be built on top of. With how fundamentally important they are to the rest of JavaFX, I would like to discuss and dissect these two Classes today.

Before we start talking about these specific Classes, let’s establish how the basics of JavaFX work. In JavaFX, the various Objects that users see and/or interact with need to be added onto a singular Object with the ‘.getChildren().add()’ method. The order that the Objects are added determines the order that the Objects are displayed. This Object is then placed onto another comprehensive Object called a scene, which also usually declares the size of the screen that appears to the user. Finally, a scene is then put onto a stage, which is the viewing area that the end user experiences.

Now that we understand how to start using these Classes, let’s break down what they are. In essence, the HBox and VBox Classes are fairly similar in their form and their function. (Which might be obvious due to their very similar names.) What the HBox and VBox Classes do is format whatever ‘containers’ are inside of them into horizontal and vertical boxes, respectively. Note that by “boxes,” I’m referring to grid cells that can contain a variety of different content types – similar to what you might see in a Microsoft Excel spreadsheet. Notably, HBox and VBox Classes and be nested inside of each other to achieve specific results. [Refer to Demo 1 for a demonstration of a basic use of HBoxes and VBoxes, as well as nested HBox and VBox Classes.]

Another thing to consider when using HBox or VBox are the sizes of the Classes. As you may have seen in Demo 1, all items placed inside an HBox or VBox will naturally be placed right next to each other if the program isn’t given any instructions about where to locate the elements. The way to declare distance between items is simple – the number placed inside the VBox(#) object during declaration determines the distance between the items placed in the Classes. Once the Object is created, the .setAlignment method can be called which tells the program where to display the HBox/VBox. [Refer to Demo 2 for a demonstration of how a VBox can be formatted more properly.]

One of the greatest advantages of the HBox and VBox Classes is how they will automatically readjust the size of the child elements to the size of the window. Anyone who’s spent an extended amount of time working with visual tech design can testify to how particular element alignment can be, and how important it is to get right across multiple internal circumstances and external platforms. While there are methods associated with the HBox and VBox Classes that can provide greater control such as .fillHeight(), HBox and VBox auto-resizing by default can be incredibly helpful.

As a final example of what HBox and VBox are capable of, I would like you to look at Demo 3 of the attached code. In this demo, you can see a very simple curved line made of plus signs. While this output is admittedly unimpressive, the amount of work it took to create such a visual can be seen inside the code. Each horizontal line is an individual Object stored within the ‘vboxContainer’ Object, and each horizontal line itself is composed of multiple Label objects. As such, a total of 64 Objects are being used to create this very basic effect. And while the code is admittedly very rough, it’s also very helpful for demonstrating exactly how HBox and VBox work.

As I’ve demonstrated, the JavaFX suite of tools is heavily reliant on the HBox and VBox Classes. While they’re nowhere near the only options available to JavaFX developers, the simplicity and straightforward purpose of them makes them a reliable tool to accomplish many tasks. Their simplicity also means that they need to rely on other JavaFX Objects to create something tangible. While this is true for most Classes in JavaFX, it’s even more true with these two – a program that only uses HBoxes or VBoxes would only create a blank screen. As such, proper usage requires a deep understanding of JavaFX, many of the Classes within it, and how they work together.

References:

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

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