**Assignment 7: JavaFX HBox and FlowPane**

The JavaFX HBox is a layout component which will place its child nodes in a horizontal row. The class that it uses is the javafx.scene.layout.HBox. One of the most common ways to utilize this class is with its constructor that takes in a list of the components for the layout. A very common component for this could be multiple buttons that one would want to place side by side in a horizontal manner. In order to be able to place the buttons side by side, one would have to create two new button objects and then create a new Hbox object and pass in the buttons as arguments to the HBox object.

As a stand-alone component the HBox class is not a visible object. This is because it requires being added to a scene graph. This in essence means that a new Scene object must be created in order for the Hbox to be “laid” on top to be visible.

When it comes to placing components in the HBox it is possible to customize some spacing in order to get a better overall design. One of the ways to do this is with spacing using the constructor or the setSpacing method. Both of these will give the programmer the ability to add some space to the nested nodes.

Another way of customizing components is controlling the margin. Like spacing, margin is used to give additional spacing to certain child nodes around the border of the component. The most common way of controlling the margin of a component is using the setMargin method.

Alignment is a huge advantage of the HBox object. Due to the fact that it is a container component it is possible to align its contents. The way this is done is with the setAlignment method which takes in a variety of parameters that allow for greater control of placement.

At certain times we want the ability for our child nodes in an HBox to fill all available space horizontally. The way we do this is with the setHgrow method. This method takes in a child node as a parameter to the method. In addition to the child node, we must pass in the horizontal expansion policy as well which gives us a few options that include: Policy.ALWAYS, Policy.SOMETIMES and Policy.NEVER. This feature can only take place if there is additional space left when the HBox is wider than the child nodes.

JavaFX contains another beneficial layout component called FlowPane. This feature allows the child components to wrap onto the next row or column if there is not enough space and depending on its vertical and horizontal requirements.

The way that you create a FlowPane is with its constructor. It is important that the child collection be added to the FlowPane in order for them to work. Just like the HBox, our FlowPane is not naturally visible so it must be added to our scene graph to make it a visible object.

It is possible to control the spacing also known as horizontal gap or vertical gap with the FlowPane. This provides a gap that can be set between the child nodes. In order to achieve this, we must use the setHgap and setVgap methods.

There is a default orientation to the FlowPane as well. It is set to a horizontal orientation which means that the child nodes will wrap horizontally onto the next line to show the additional nodes. It is possible to change this in order to customize the orientation a bit more. Instead of the default we can set the orientation to vertical with our setOrientation method. When using this method, we have the ability to change the direction to a wrapping that is vertical. So child nodes will wrap in a column format instead of our horizontal row format. When we use the setOrientation method we pass in the direction argument to control that flow. The FlowPane is great way to add customization to child nodes and to help them stay visible no matter what the size of the underlying scene graph may be. Without the FlowPane it becomes difficult to keep child components visible and it can make for a poor user experience.

Resources:

<http://tutorials.jenkov.com/javafx/hbox.html>

<http://tutorials.jenkov.com/javafx/flowpane.html>