

## Introduction

The VGroup container is a Group container that uses the VerticalLayout class. Use the properties of the VGroup class to modify the characteristics of the VerticalLayout class.

## Class declaration

Following is the declaration for **spark.components.VGroup** class:

```
public class VGroup
    extends Group
```

## Public properties

| S.N. | Property & Description                                                                                                                                                                                      |
|------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 1    | <b>firstIndexInView : int</b><br><br>[read-only] The index of the first layout element that is part of the layout and within the layout target's scroll rectangle, or -1 if nothing has been displayed yet. |
| 2    | <b>gap : int</b><br><br>The vertical space between layout elements, in pixels.                                                                                                                              |
| 3    | <b>horizontalAlign : String</b><br><br>The horizontal alignment of layout elements.                                                                                                                         |
| 4    | <b>lastIndexInView : int</b><br><br>[read-only] The index of the last row that's part of the layout and within the container's scroll rectangle, or -1 if nothing has been displayed yet.                   |
| 5    | <b>paddingBottom : Number</b><br><br>Number of pixels between the container's bottom edge and the bottom edge of the last layout element.                                                                   |
| 6    | <b>paddingLeft : Number</b><br><br>The minimum number of pixels between the container's left edge and the left edge of the layout element.                                                                  |
| 7    | <b>paddingRight : Number</b><br><br>The minimum number of pixels between the container's right edge and the right edge of the layout element.                                                               |

|    |                                    |                                                                                                              |
|----|------------------------------------|--------------------------------------------------------------------------------------------------------------|
| 8  | <b>paddingTop : Number</b>         | Number of pixels between the container's top edge and the top edge of the first layout element.              |
| 9  | <b>requestedMaxRowCount : int</b>  | The measured height of this layout is large enough to display at most requestedMaxRowCount layout elements.  |
| 10 | <b>requestedMinRowCount : int</b>  | The measured height of this layout is large enough to display at least requestedMinRowCount layout elements. |
| 11 | <b>requestedRowCount : int</b>     | The measured size of this layout is tall enough to display the first requestedRowCount layout elements.      |
| 12 | <b>rowCount : int</b>              | [read-only] The current number of visible elements.                                                          |
| 13 | <b>rowHeight : Number</b>          | If variableRowHeight is false, then this property specifies the actual height of each child, in pixels.      |
| 14 | <b>variableRowHeight : Boolean</b> | Specifies whether layout elements are allocated their preferred height.                                      |
| 15 | <b>verticalAlign : String</b>      | The vertical alignment of the content relative to the container's height.                                    |

## Public methods

| S.N. | Method & Description          |
|------|-------------------------------|
| 1    | <b>VGroup</b><br>Constructor. |

## Methods inherited

This class inherits methods from the following classes:

- spark.components.Group
- spark.components.supportClasses.GroupBase
- mx.core.UIComponent
- mx.core.FlexSprite
- flash.display.Sprite
- flash.display.DisplayObjectContainer
- flash.display.InteractiveObject
- flash.display.DisplayObject
- flash.events.EventDispatcher
- Object

## Flex VGroup Example

Let us follow the following steps to check usage of VGroup in a Flex application by creating a test application:

| Step | Description                                                                                                                                                   |
|------|---------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 1    | Create a project with a name <i>HelloWorld</i> under a package <i>com.tutorialspoint.client</i> as explained in the <i>Flex - Create Application</i> chapter. |
| 2    | Modify <i>HelloWorld.mxml</i> as explained below. Keep rest of the files unchanged.                                                                           |
| 3    | Compile and run the application to make sure business logic is working as per the requirements.                                                               |

Following is the content of the modified mxml file **src/com.tutorialspoint/HelloWorld.mxml**.

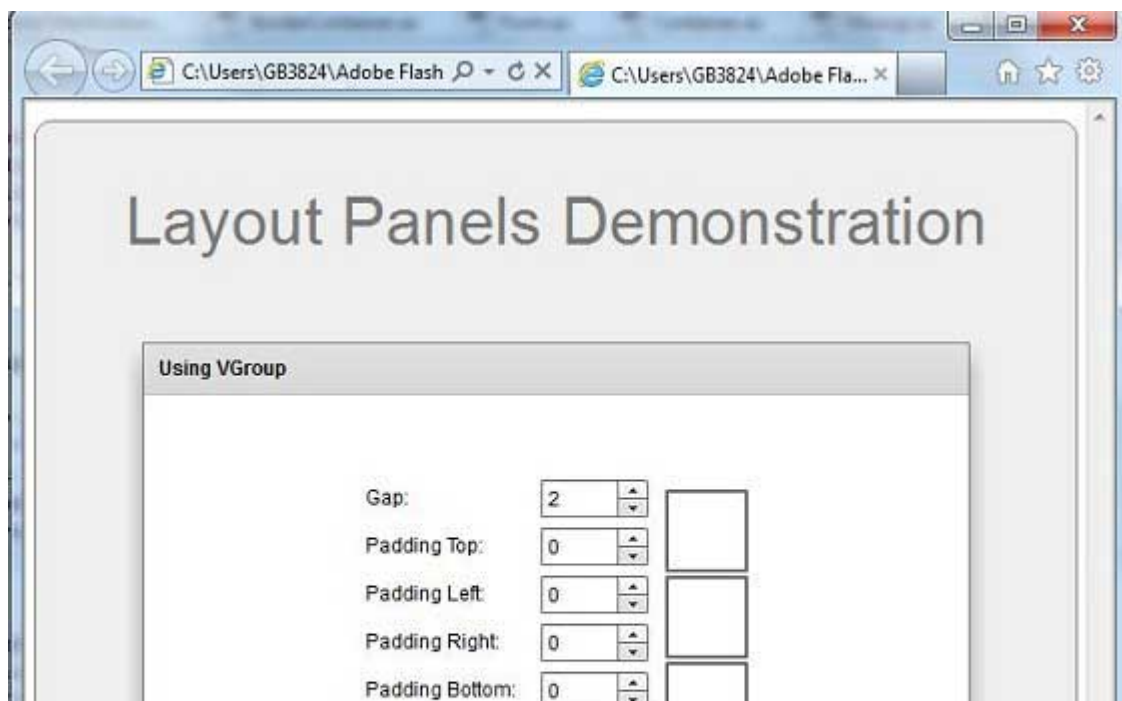
```
<?xml version="1.0" encoding="utf-8"?>
<s:Application xmlns:fx="http://ns.adobe.com/mxml/2009"
    xmlns:s="library://ns.adobe.com/flex/spark"
    xmlns:mx="library://ns.adobe.com/flex/mx"
    width="100%" height="100%" minWidth="500" minHeight="500"
    >
    <fx:Style source="/com/tutorialspoint/client/Style.css"/>
    <fx:Script>
        <![CDATA[
            private function applyVGroupProperties():void
            {
                mainVGroup.paddingTop = padTopV.value;
                mainVGroup.paddingLeft = padLeftV.value;
                mainVGroup.paddingRight = padRightV.value;
                mainVGroup.paddingBottom = padBottomV.value;
                mainVGroup.gap = gapV.value;
            }
        ]]>
    </fx:Script>
    <s:BorderContainer width="630" height="480"
        styleName="container">
        <s:VGroup width="100%" height="100%" gap="50"
            horizontalAlign="center" verticalAlign="middle">
            <s:Label
                fontSize="40" color="0x777777" styleName="heading"/>
            <s:Panel
```

```

width="500" height="300" includeInLayout="true" visible="true">
<s:layout>
  <s:HorizontalLayout gap="10" verticalAlign="middle"
    horizontalAlign="center"/>
</s:layout>
<s:VGroup top="10" left="15">
  <s:HGroup verticalAlign="middle">
    <s:Label text="Gap:" width="100"/>
    <s:NumericStepper />
  </s:HGroup>
  <s:HGroup verticalAlign="middle">
    <s:Label text="Padding Top:" width="100"/>
    <s:NumericStepper />
  </s:HGroup>
  <s:HGroup verticalAlign="middle">
    <s:Label text="Padding Left:" width="100"/>
    <s:NumericStepper />
  </s:HGroup>
  <s:HGroup verticalAlign="middle">
    <s:Label text="Padding Right:" width="100"/>
    <s:NumericStepper />
  </s:HGroup>
  <s:HGroup verticalAlign="middle">
    <s:Label text="Padding Bottom:" width="100"/>
    <s:NumericStepper />
  </s:HGroup>
  <s:Button label="Apply Properties"
    click="applyVGroupProperties()"/>
</s:VGroup>
<s:VGroup left="300" top="25" >
  <s:BorderContainer width="50" height="50"
    borderWidth="2" color="0x323232" />
  <s:BorderContainer width="50" height="50"
    borderWidth="2" color="0x323232" />
  <s:BorderContainer width="50" height="50"
    borderWidth="2" color="0x323232" />
</s:VGroup>
</s:Panel>
</s:VGroup>
</s:BorderContainer>
</s:Application>

```

Once you are ready with all the changes done, let us compile and run the application in normal mode as we did in [Flex - Create Application](#) chapter. If everything is fine with your application, this will produce following result: [ [Try it online](#) ]



Apply Properties

Loading [MathJax]/jax/output/HTML-CSS/fonts/TeX/fontdata.js