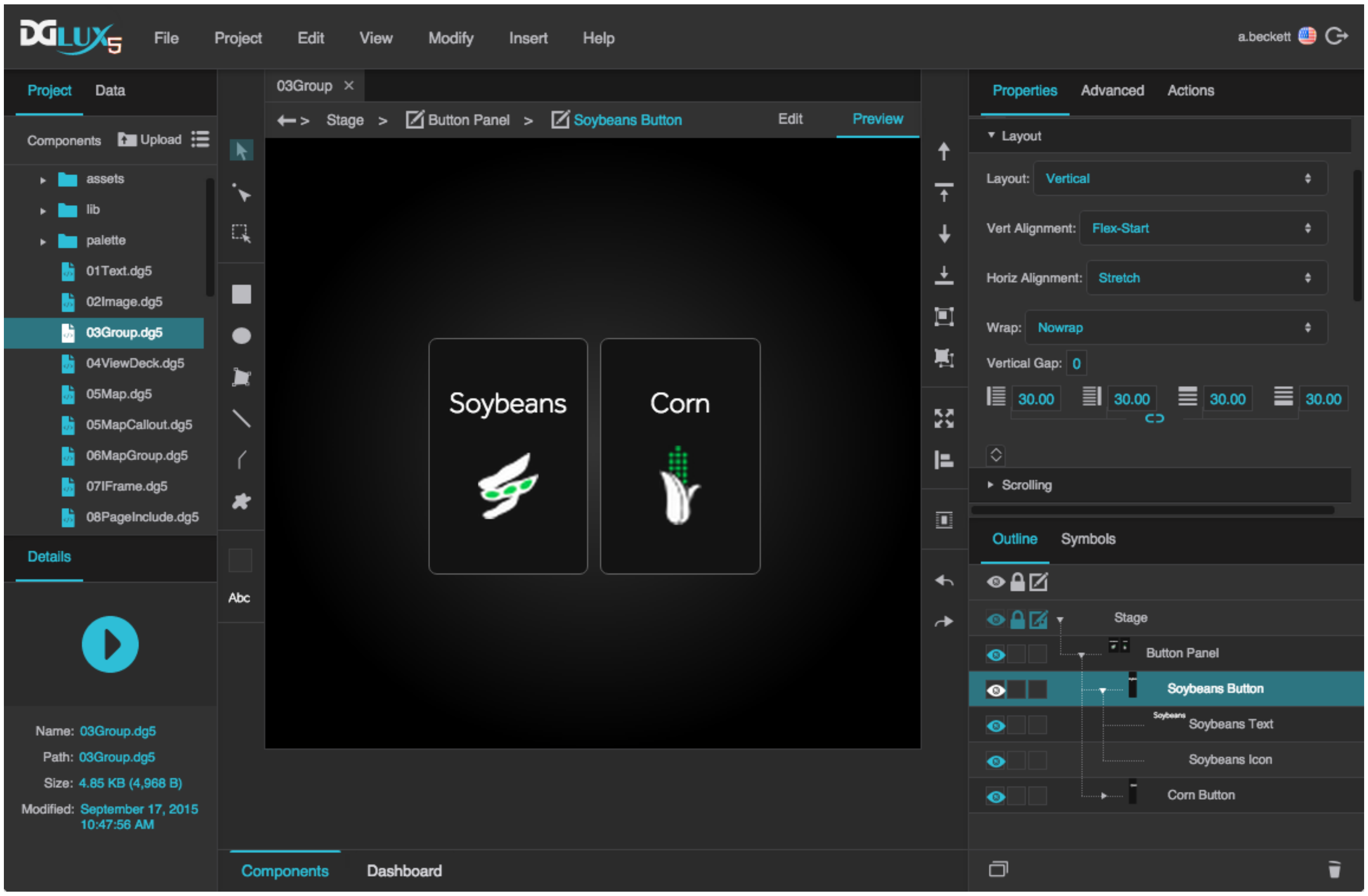
Group

A group is a container that holds other DGLux5 components and can control their layout.

Groups are a fundamental part of most DGLux5 projects. They are often essential when creating clean and responsive layouts. They are also important when creating any widget that comprises multiple components, such as a gauge, heat map, button, pop-up, or callout. Additionally, groups allow you to create a symbol comprising multiple components. Finally, groups can hold repeaters.

The Stage, or root component, of a page behaves like a group.

For a detailed reference of properties that affect groups, see Common Properties and Group and Stage Properties.



Name: 03Group.dg5
Path: 03Group.dg5
Size: 4.85 KB (4,968 B)
Modified: September 17, 2015 10:47:56 AM

Layout: Vertical
Vert Alignment: Flex-Start
Horiz Alignment: Stretch
Wrap: Nowrap
Vertical Gap: 0
30.00 30.00 30.00 30.00

Group components in DGLux5. The main group component, "Button Panel," contains two subgroups.

Create a Group

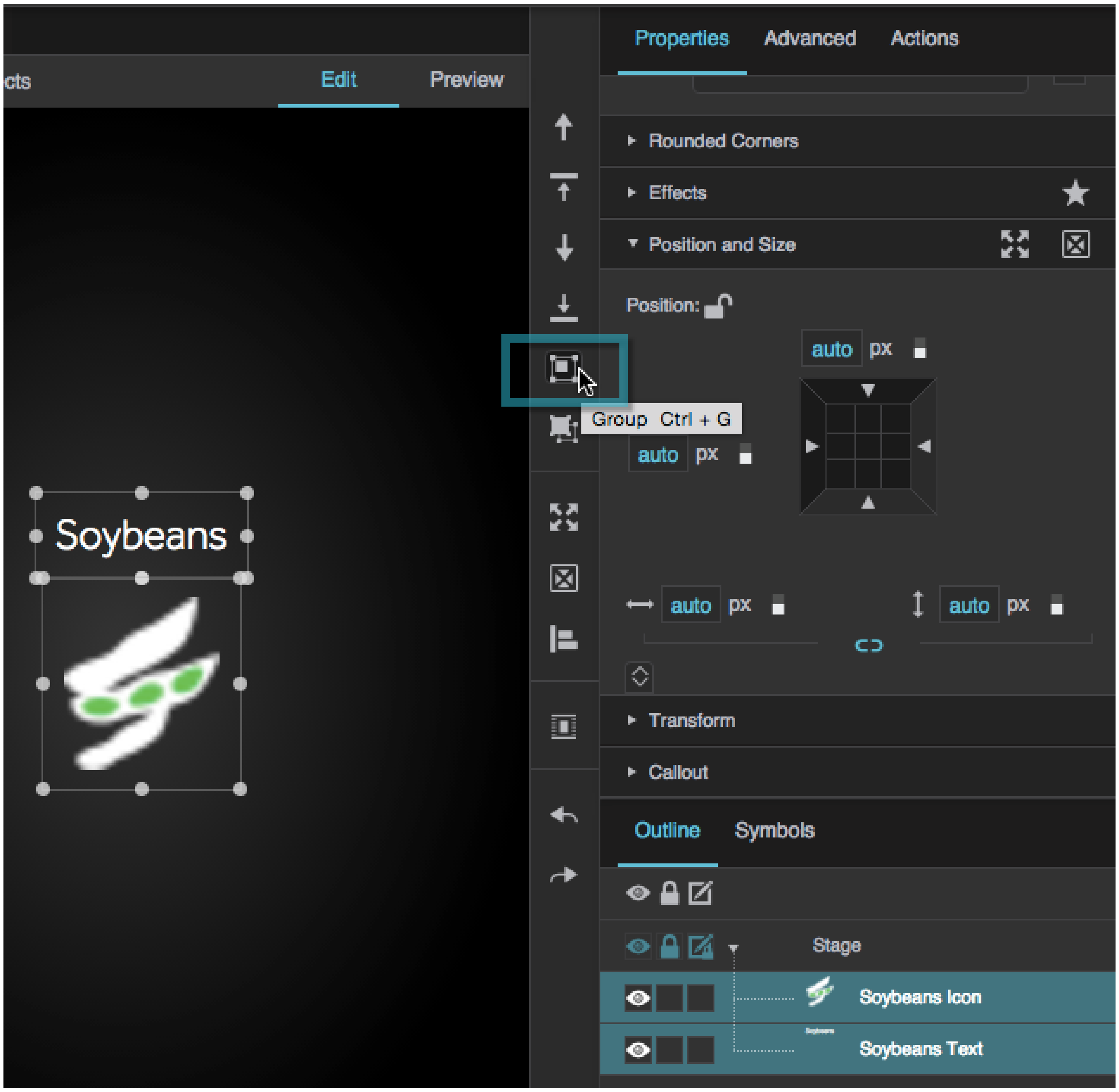
There are two main ways to create groups.

The first way is to group existing objects. This creates a group that has Absolute layout and Fit Ratio scaling set by default. Fit Ratio scaling means that elements in the group will resize with the group while maintaining their aspect ratio.

1. Select all of the objects you want to group.

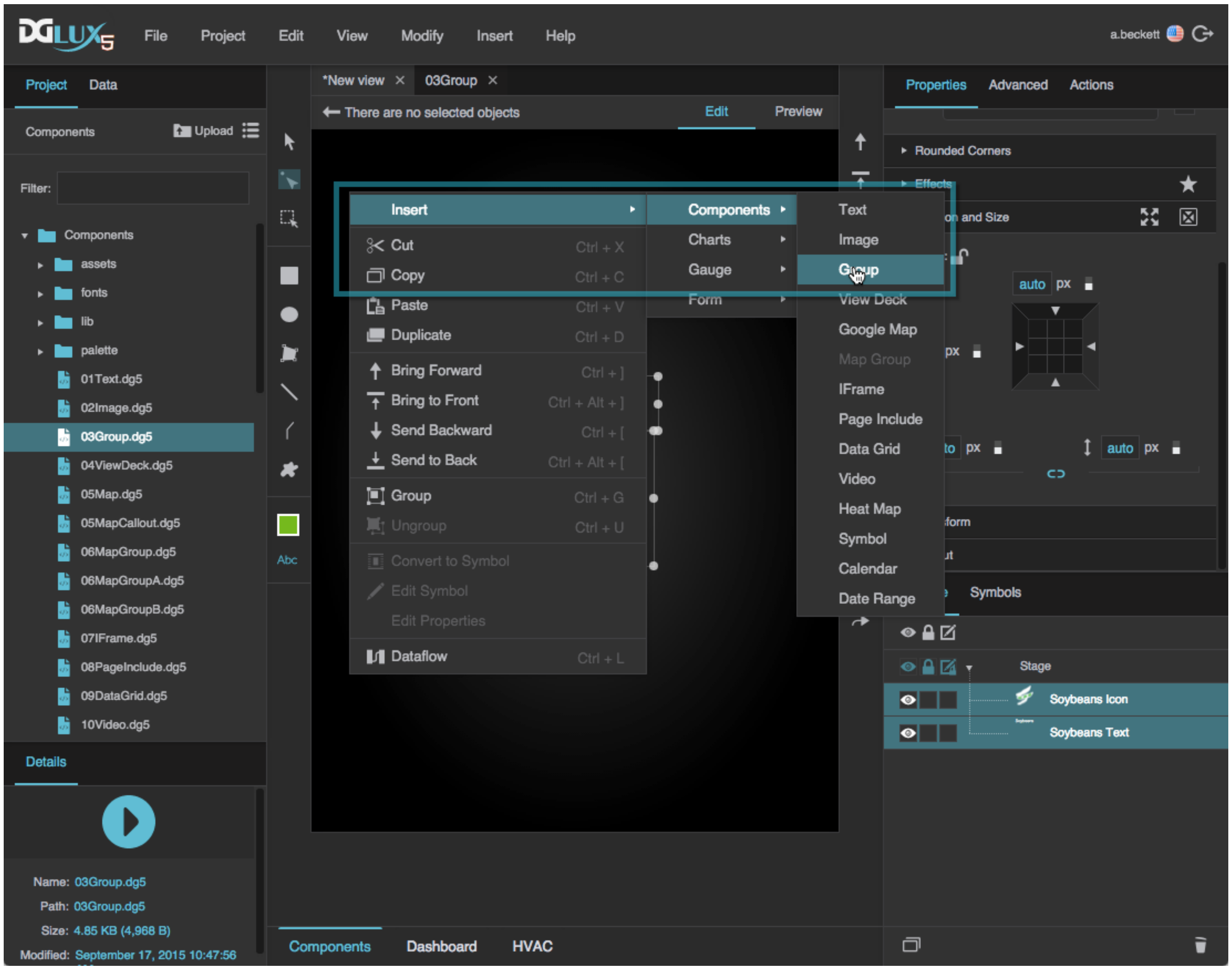
   You can do this by Shift-clicking, or by using the Select tool to drag a selection rectangle.

2. In the Quick Access panel to the right of the Document window, click **Group**.



The second way is to insert an empty group and give it children. This creates a group that has Absolute layout and no scaling set by default. No scaling means that elements in the group do not resize with the group.

1. Right-click in the Outline or Document window, and select **Insert** > **Components** > **Group**.



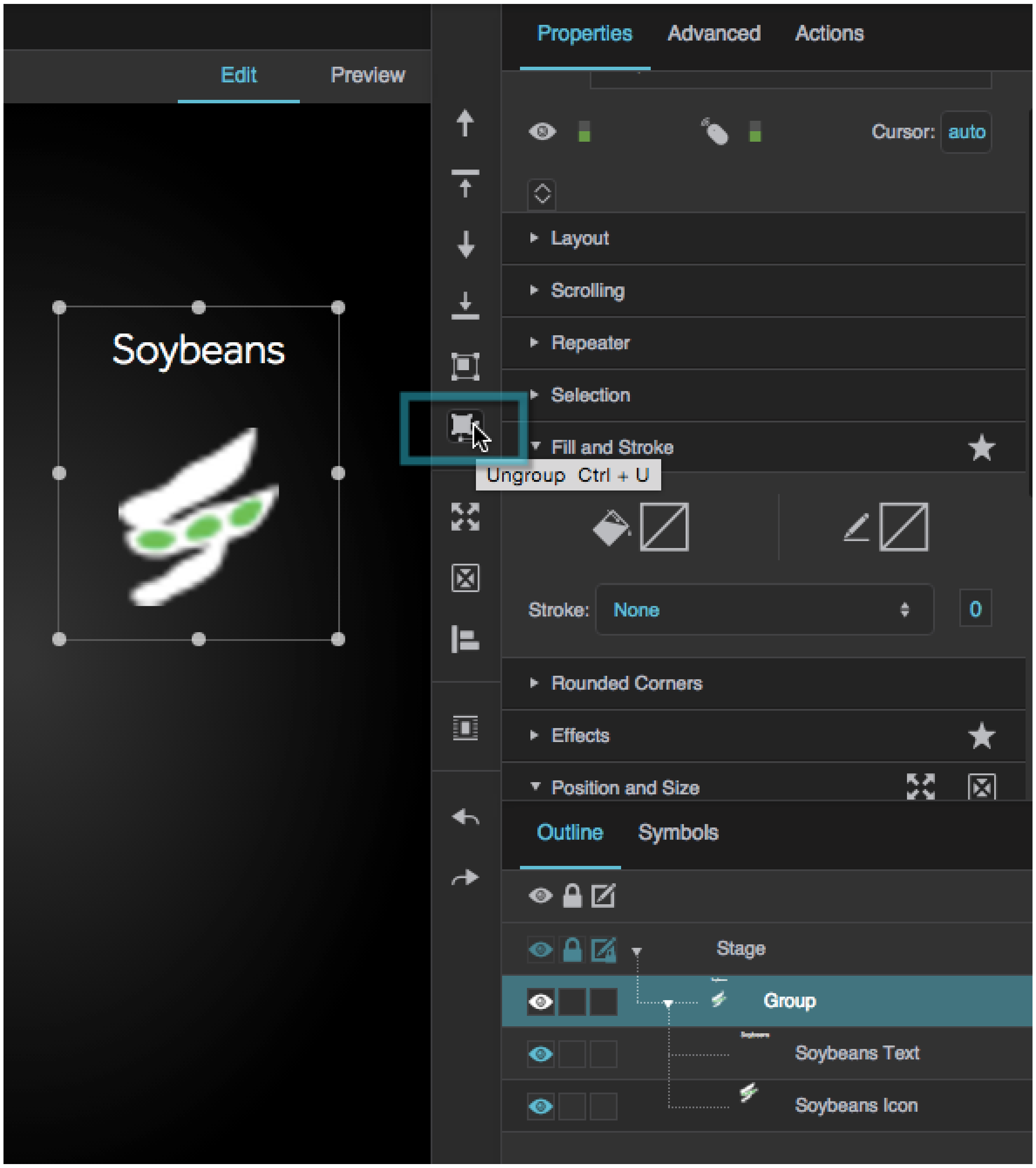
2. Insert more objects as children of the group.

   You can do this by dragging objects into the group or by right-clicking the group and inserting objects.

Ungroup Items

To delete a group component and transfer its children to its parent:

1. Select the group.
2. In the Quick Access panel to the right of the Document window, click **Ungroup**.



Groups FAQ

This section provides answers to some common questions about using groups in DGLux5.

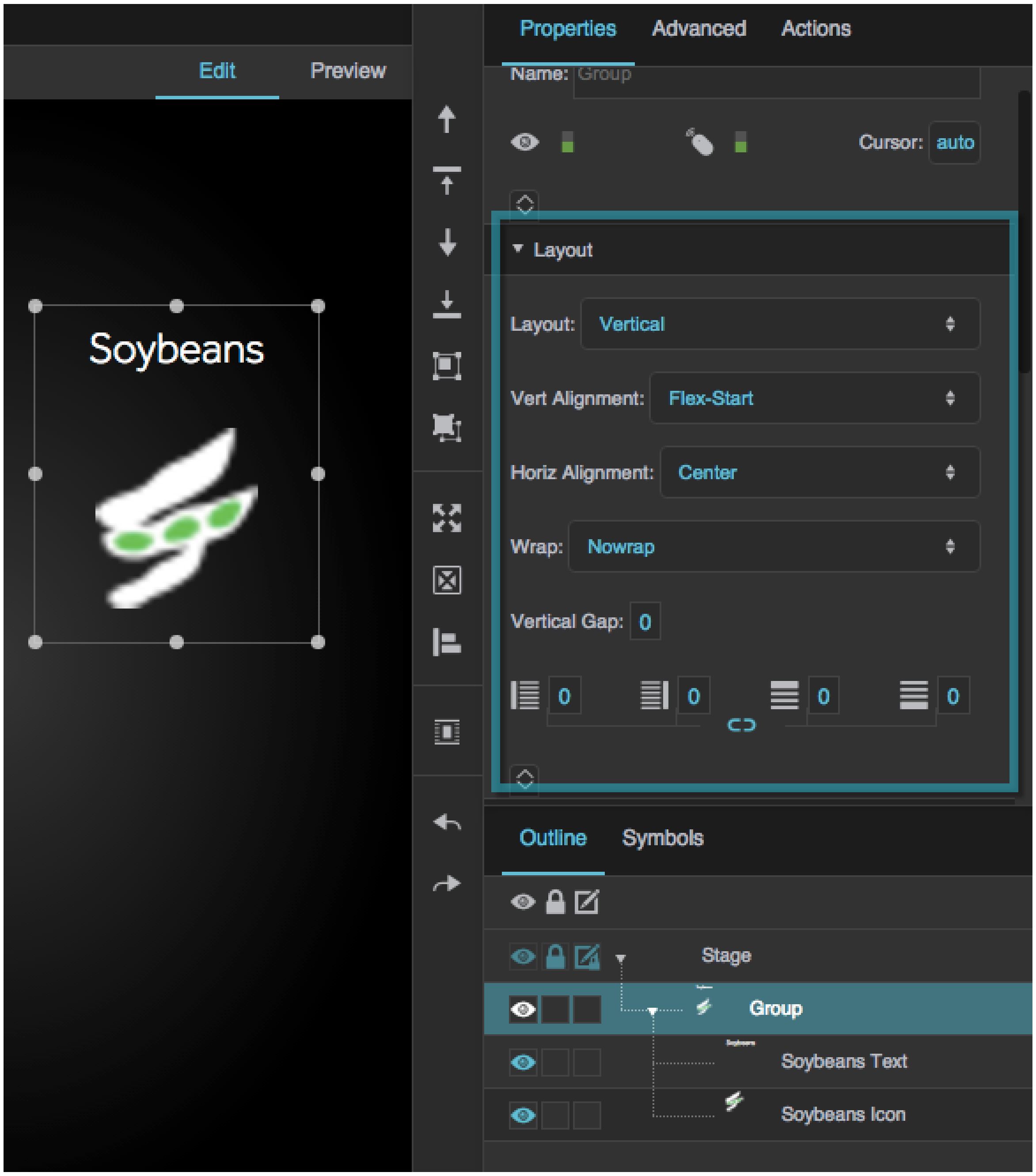
Layout properties follow the CSS3 `flexbox` concept. You can read more about `flexbox` in online resources like this one.

Click to display/hide all elements

How do I position items in rows or columns? How do I position them freely?

To change the Layout type of a group or the Stage:

1. In the Outline or Document window, select the group.
2. In the Property Inspector, use the **Layout** properties:
    - To position items in one or more columns, set **Layout** to Vertical.



Layout

Layout: Vertical

Vert Alignment: Flex-Start

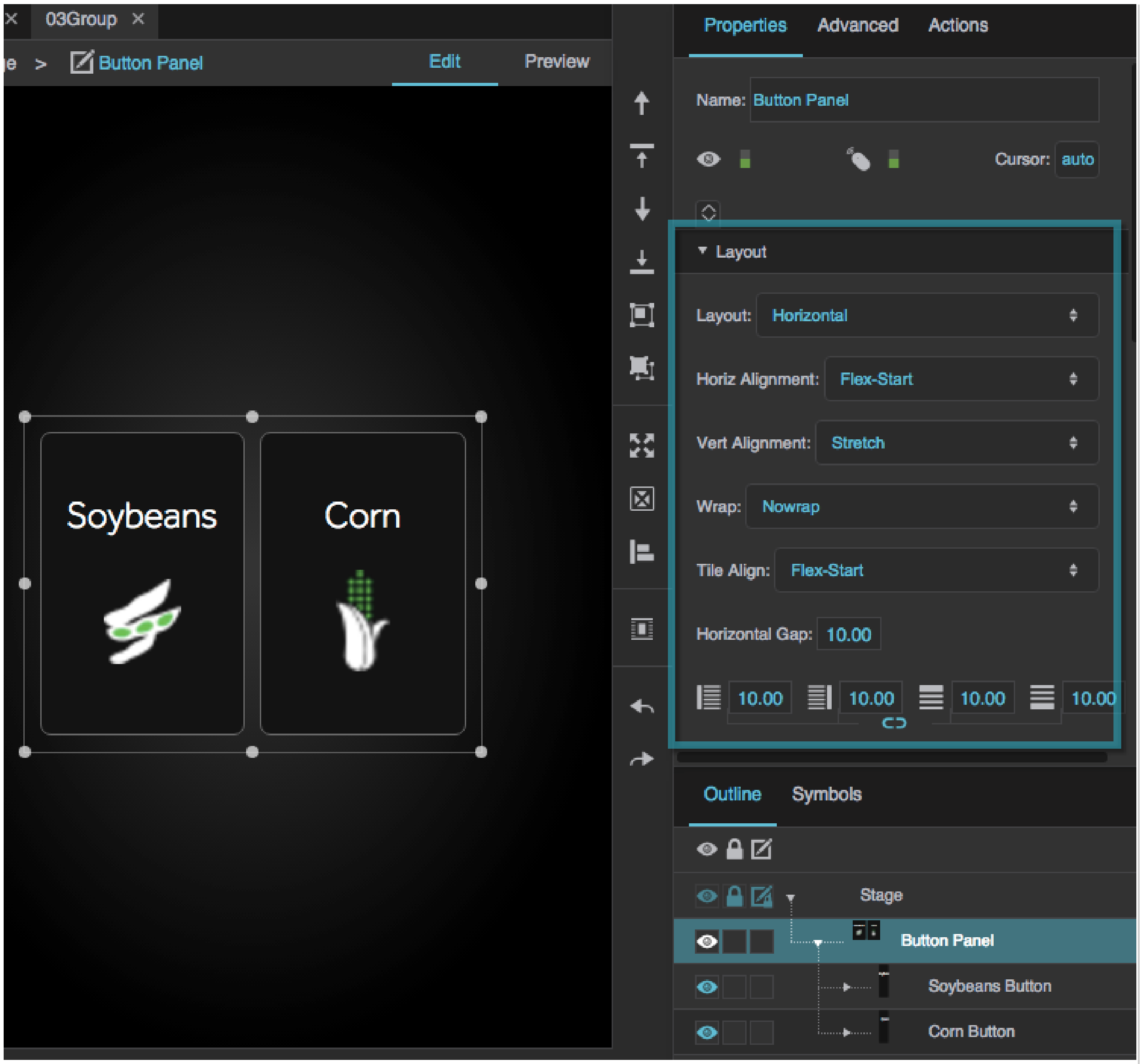
Horiz Alignment: Center

Wrap: Nowrap

Vertical Gap: 0

0 0 0 0

    - To position items in one or more rows, set **Layout** to Horizontal.



Name: Button Panel
Cursor: auto
Layout
Layout: Horizontal
Horiz Alignment: Flex-Start
Vert Alignment: Stretch
Wrap: Nowrap
Tile Align: Flex-Start
Horizontal Gap: 10.00
10.00 10.00 10.00 10.00

- To control whether there is one row or column, or as many as needed, use the **Wrap** property.
- To position items freely, set **Layout** to Absolute.

  When using Absolute layout, Scaling is typically recommended.

How can I adjust the alignment of items in horizontal or vertical layout?

To change the alignment of items in a group:

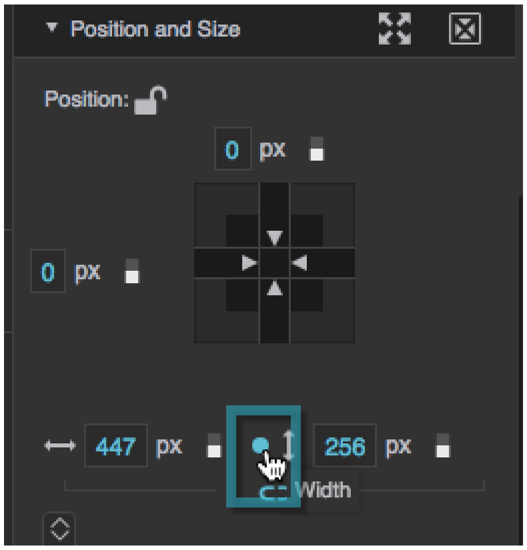
1. In the Outline or Document window, select the group.
2. In the Property Inspector, under **Layout**, ensure that **Layout** is set to Horizontal or Vertical.
3. In the Property Inspector, use the following properties:
   - To control whether there is one row or column, or as many as needed, use the **Wrap** property.
   - To control the alignment of items within rows or columns, use the **Horizontal Layout** and

Vertical Layout properties.
  - If **Wrap** is enabled, control the alignment of entire rows or columns using the **Tile Align** property.
  - To control the space between elements, use **Horizontal Gap** and **Vertical Gap**.
  - To control the space between the boundary of the group and its contents, use the **Padding** properties.

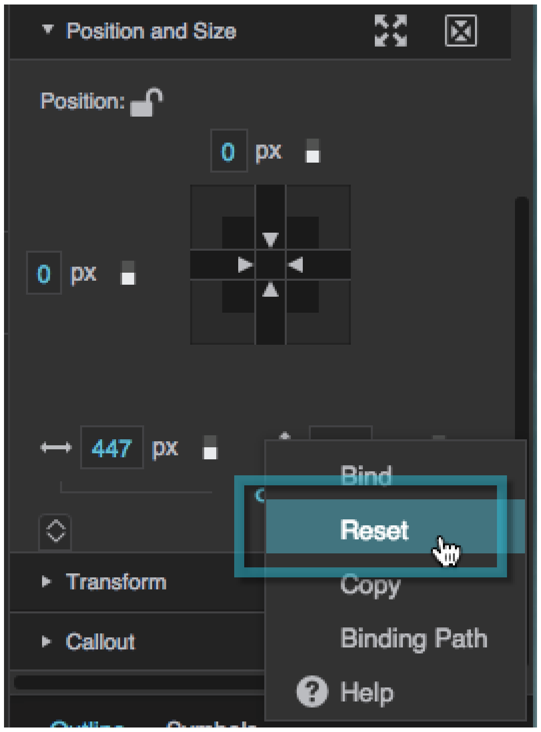
How do I make the size of the group follow the size of its content?
You can make the size of a group follow the size of its content by setting the group's width and height to Auto. These steps work only if the group uses Horizontal or Vertical layout, and the children of the group have a size that is not determined by their parent, for example a pixel size, or a size determined by their children.

1. In the Outline or Document window, select the group.
2. In the Property Inspector, under **Position and Size**, set **Width** to Auto:

    1. Hover over the **Width** property until a blue dot appears.



    2. Click the blue dot, and select **Reset**.



3. Repeat step 2 for the **Height** property.

How do I make the size of the content follow the size of the group?
If the group uses Absolute layout:
Use the Scaling properties for the group.

1. Position elements where you want them inside the group.

2. Select the group, and in the Property Inspector, set **Scaling** to Fit Ratio or Fit.

   Fit Ratio maintains the aspect ratio of children, and Fit does not.

   When the group resizes, the children scale to the size of the group.

If the group uses Horizontal or Vertical layout:
Use **Stretch** to make elements grow in the opposite direction from their layout to fill the container.
These steps are intended to be used only when **Wrap** is set to **Nowrap**. For a Vertical layout:

1. In the Outline or Document window, select the group.
2. In the Property Inspector, under **Layout**, ensure that **Layout** is set to Vertical and **Wrap** is set to **Nowrap**.
3. Set **Horizontal Alignment** to Stretch.
4. Select the children that you want to stretch, and set their **Width** values to Auto:
   1. Under **Position and Size**, hover over the **Width** property until a blue dot appears.

   2. Click the blue dot, and select **Reset**.

      The children's width follows the width of the group.

Use Flex-Grow to make elements grow in the same direction as their layout to fill the container. For a Vertical layout:

1. In the Outline or Document window, select the group.
2. In the Property Inspector, under **Layout**, ensure that **Layout** is set to Vertical.
3. Select the children that you want to grow to fill available vertical space.
4. Set the **Flex-Grow** property of the children to a positive number:
    1. Under **Position and Size**, for **Flex-Grow**, enter 1.
5. (Recommended) Set the height of the children to Auto:
    1. Under **Position and Size**, hover over the **Height** property until a blue dot appears.
    2. Click the blue dot, and select **Reset**.

More Resources

This post in the DGLogik Community Forum describes using a group to enforce a border surrounding all page content.

These wiki pages show some basic use cases:

- Creating a simple page with vertical and horizontal layouts: Create a Simple Page.
- Using Absolute layout and Scaling to make a group whose children scale: Absolute Layout.
- Creating a heat map: Heat Map.

Group and Stage Properties

These property groups apply to group components and the Stage.

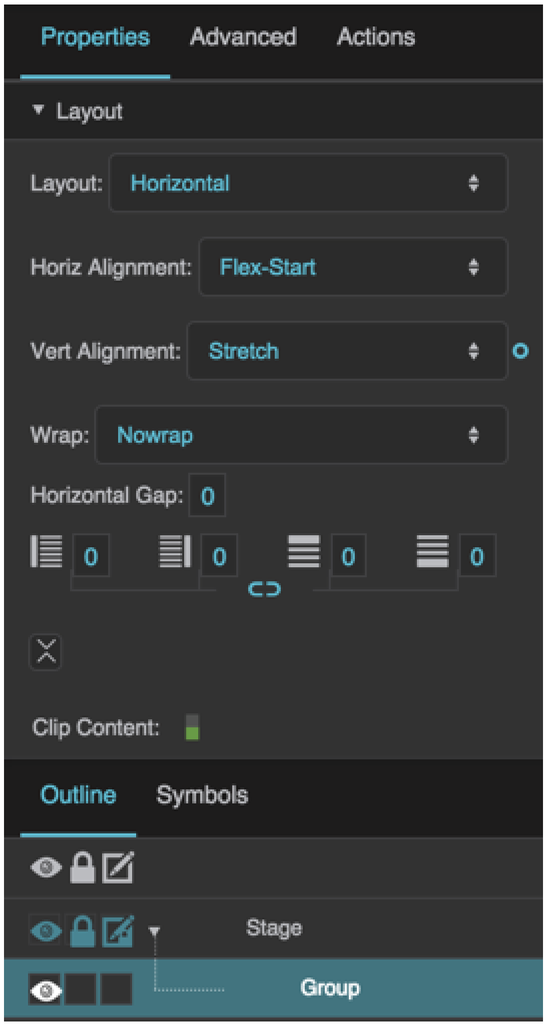
- Layout properties affect the layout of this group's children.
- Scaling properties affect the placement of this group's children when Absolute layout is used.

Groups and the Stage are also affected by common properties.

2019/07/17 19:17

Layout Properties

These properties affect the placement of this group's children.



Properties | Advanced | Actions

Layout

Layout: Horizontal

Horiz Alignment: Flex-Start

Vert Alignment: Stretch

Wrap: Nowrap

Horizontal Gap: 0

0 0 0 0

Clip Content:

Outline | Symbols

Stage

Group

The Layout properties in the Property Inspector

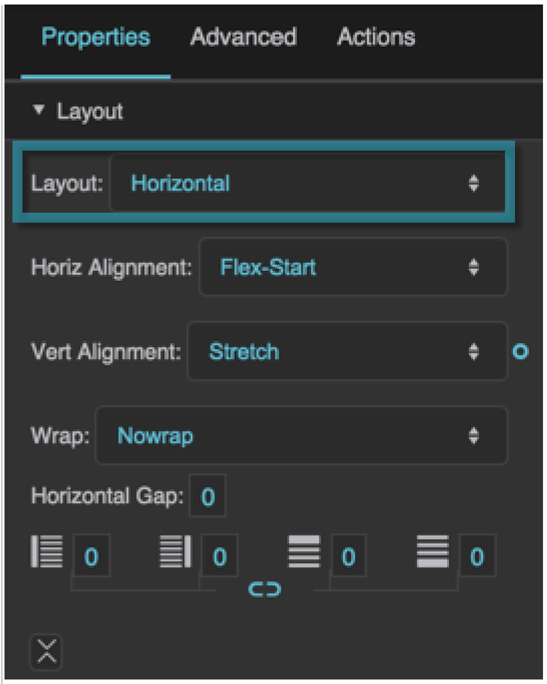
Click to display/hide all elements

Layout
Defines how elements are positioned within this container.

Absolute
Elements are positioned freely.

Horizontal
Elements are positioned in order horizontally left to right.

Vertical
Elements are positioned in order vertically top to bottom.



Layout: Horizontal
Horiz Alignment: Flex-Start
Vert Alignment: Stretch
Wrap: Nowrap
Horizontal Gap: 0
0 0 0 0

The Layout property

Horizontal Alignment
Defines the horizontal positioning of all elements within this container.

Flex-Start
Elements are packed against the left edge.

Flex-End
Elements are packed against the right edge.

Center
Elements are packed together around the horizontal center.

Space-Between
Elements are distributed with the left element flush against the left edge, the right element flush against the right edge, and equal space between the items.

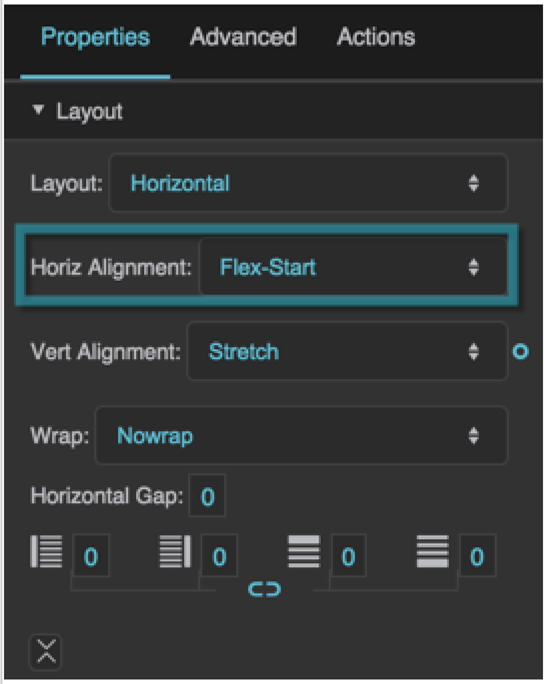
Space-Around*
Elements are distributed with equal space between them. The space at the left and right of the container equals half of the space between two adjacent items.

Baseline**
The baseline of all elements is aligned to the baseline of the parent container.

Stretch**
Elements stretch horizontally to fit the parent container. Only works when the elements have a width value of "auto."

* Only available when the Layout property is set to Horizontal

** Only available when the Layout property is set to Vertical



Layout: Horizontal

Horiz Alignment: Flex-Start

Vert Alignment: Stretch

Wrap: Nowrap

Horizontal Gap: 0

0 0 0 0

The Horizontal Alignment property

Vertical Alignment
Defines vertical positioning of all elements within this container.

Flex-Start
Elements are packed against the top edge.

Flex-End
Elements are packed against the bottom edge.

Center
Elements are packed together around the vertical center.

Space-Between
Elements are distributed with the top element flush against the top edge, the bottom element flush against the bottom edge, and equal space between the items.

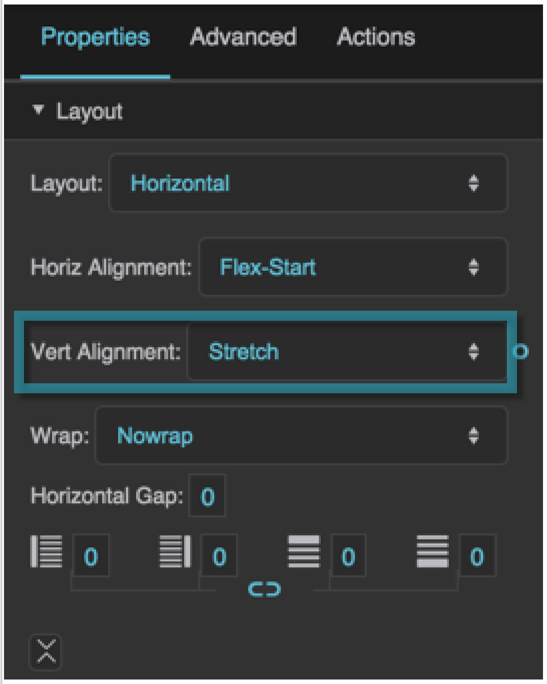
Space-Around*
Elements are distributed with equal space between them. The space at the top and bottom of the container equals half of the space between two adjacent items.

Baseline**
The baseline of all elements is aligned to the baseline of the parent container.

Stretch**
Elements stretch vertically to fit the parent container. Only works when the elements have a height value of "auto."

* Only available when the Layout property is set to Vertical

** Only available when the Layout property is set to Horizontal



Layout: Horizontal
Horiz Alignment: Flex-Start
Vert Alignment: Stretch
Wrap: Nowrap
Horizontal Gap: 0
0 0 0 0

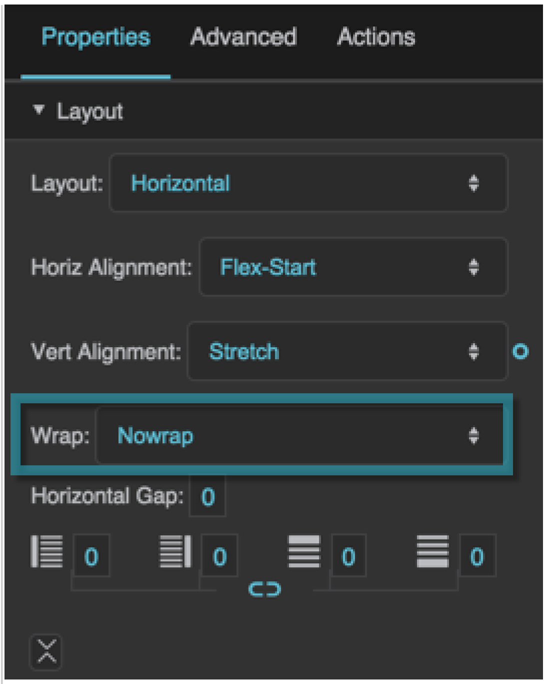
The Vertical Alignment property

Wrap
Defines whether elements within this container can wrap when there is not enough space to fit them in a single row or column. When the Layout property is set to Vertical, elements can wrap into columns. When the Layout property is set to Horizontal, elements can wrap into rows.

Nowrap
Elements do not wrap.

Wrap
The first element that does not fit in a row or column starts a new row or column. Elements in a row are positioned from left to right. Elements in a column are positioned from top to bottom. The first row is on the top, or the first column is on the left.

Wrap-Reverse
Elements behave as in Wrap, except that the first row is on the bottom, or the first column is on the right.



Layout: Horizontal

Horiz Alignment: Flex-Start

Vert Alignment: Stretch

Wrap: Nowrap

Horizontal Gap: 0

0 0 0 0

The Wrap property

Tile Align
Defines positioning of the columns and rows containing wrapped elements within this container.

Flex-Start
The rows are packed against the top edge, or the columns are packed against the left edge.

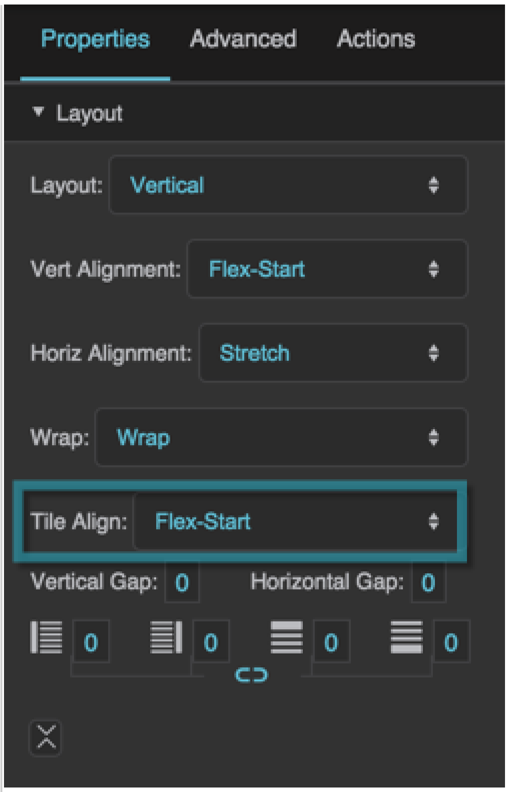
Flex-End
The rows are packed against the bottom edge, or the columns are packed against the right edge.

Center
The rows or columns are packed together around the center of the container.

Space-Between
The rows or columns are positioned against both edges, with equal space between them.

Space-Around
The rows or columns are positioned with equal space between them, and space along both edges equal to half of the space between two rows or columns.

Stretch
The rows or columns stretch to fit the parent container.



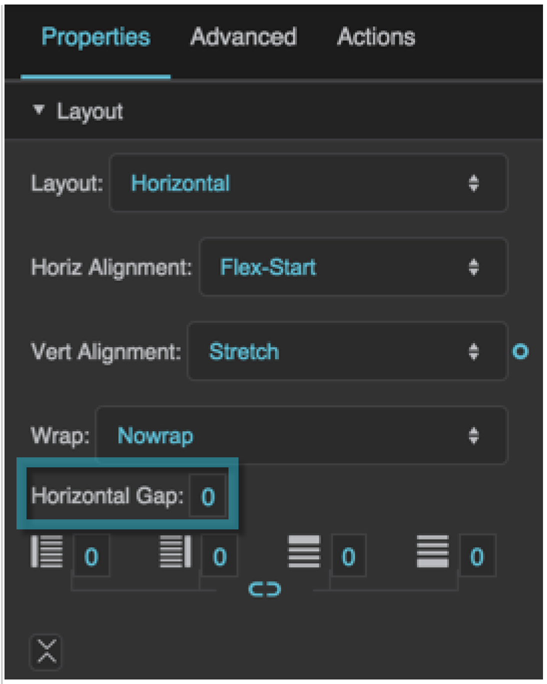
Layout: Vertical
Vert Alignment: Flex-Start
Horiz Alignment: Stretch
Wrap: Wrap
Tile Align: Flex-Start
Vertical Gap: 0
Horizontal Gap: 0
0 0 0 0

The Tile Align property

Horizontal Gap
Defines the minimum number of pixels between elements in a horizontal layout, or between columns in a wrapped vertical layout.

Notes about padding with a wrapped horizontal layout:

- When Horizontal Alignment is Flex-Start, Horizontal Gap defines a right padding, but you must define a left padding separately.
- For all other Horizontal Alignments, Horizontal Gap defines a left padding, but you must define a right padding separately.



Layout: Horizontal
Horiz Alignment: Flex-Start
Vert Alignment: Stretch
Wrap: Nowrap
Horizontal Gap: 0

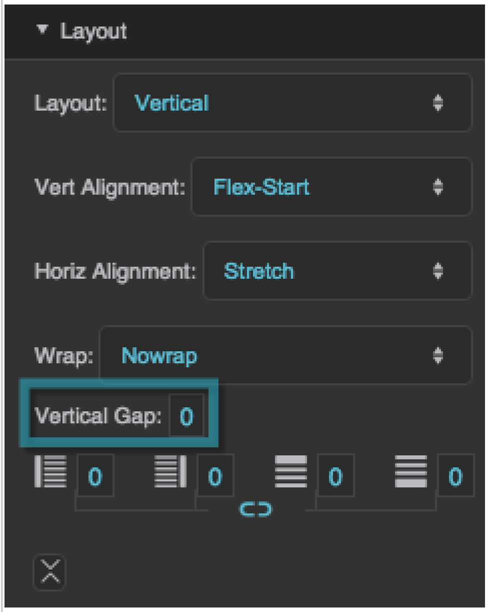
0 0 0 0

The Horizontal Gap property

Vertical Gap
Defines the minimum number of pixels between elements in a vertical layout, or between rows in a wrapped horizontal layout.

Notes about padding with a wrapped vertical layout:

- When Vertical Alignment is Flex-Start, Vertical Gap defines a bottom padding, but you must define a top padding separately.
- For all other Vertical Alignments, Vertical Gap defines a top padding, but you must define a bottom padding separately.



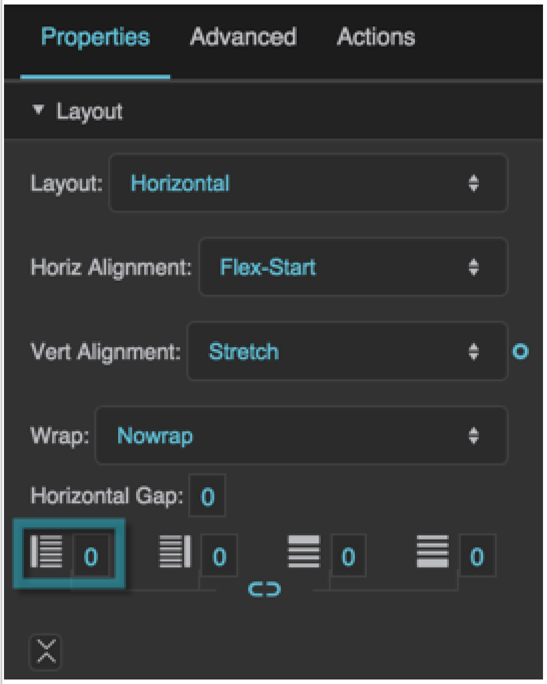
Layout: Vertical
Vert Alignment: Flex-Start
Horiz Alignment: Stretch
Wrap: Nowrap
Vertical Gap: 0

0 0 0 0

The Vertical Gap property

Left Padding

Defines a number of pixels of space between the left container boundary and the leftmost edge of the content. Negative numbers are not allowed.



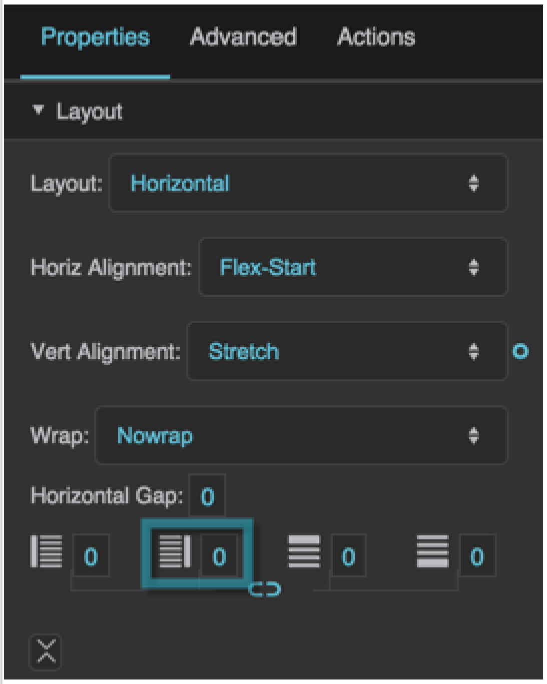
Layout: Horizontal
Horiz Alignment: Flex-Start
Vert Alignment: Stretch
Wrap: Nowrap
Horizontal Gap: 0

0 0 0 0

The Left Padding property

Right Padding

Defines a number of pixels of space between the right container boundary and the rightmost edge of the content. Negative numbers are not allowed.

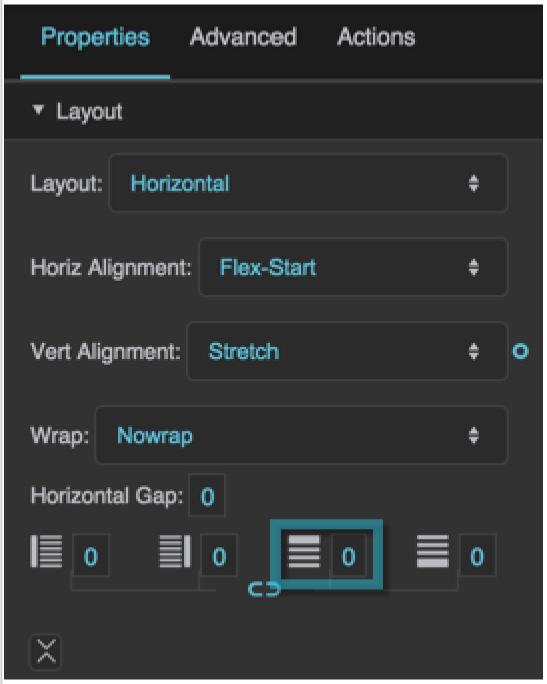


Layout: Horizontal
Horiz Alignment: Flex-Start
Vert Alignment: Stretch
Wrap: Nowrap
Horizontal Gap: 0

0 0 0 0

The Right Padding property

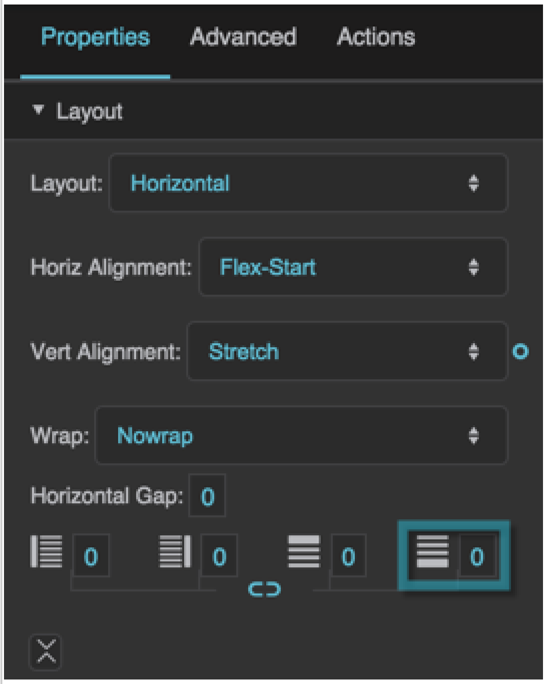
Top Padding
Defines a number of pixels of space between the top container boundary and the topmost edge of the content. Negative numbers are not allowed.



Layout: Horizontal
Horiz Alignment: Flex-Start
Vert Alignment: Stretch
Wrap: Nowrap
Horizontal Gap: 0
0 0 0 0

The Top Padding property

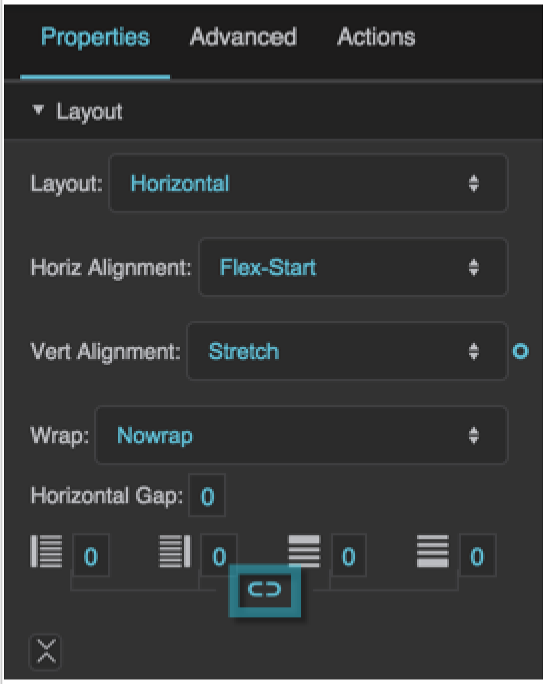
Bottom Padding
Defines a number of pixels of space between the bottom container boundary and the bottommost edge of the content. Negative numbers are not allowed.



Layout: Horizontal
Horiz Alignment: Flex-Start
Vert Alignment: Stretch
Wrap: Nowrap
Horizontal Gap: 0
0 0 0 0

The Bottom Padding property

Link Paddings
Links all four padding editors so that their values are equal.



Layout: Horizontal
Horiz Alignment: Flex-Start
Vert Alignment: Stretch
Wrap: Nowrap
Horizontal Gap: 0
0 0 0 0

The Link Paddings property

Clip Content
Defines the handling of content that overflows this container.

TRUE
All overflowed content is clipped and is not visible.

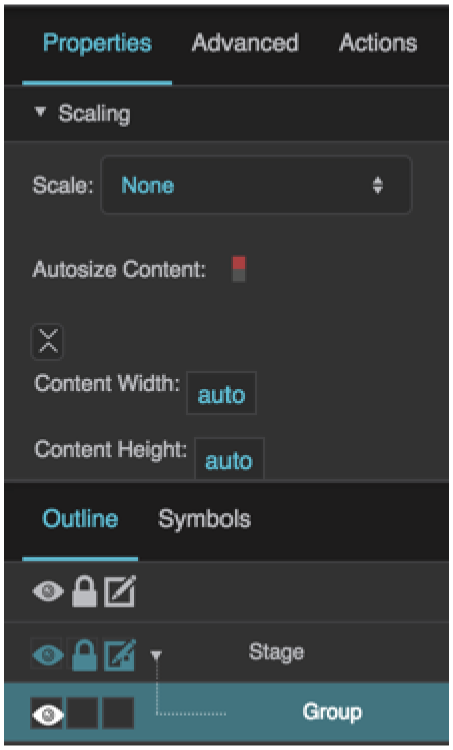
FALSE
All overflowed content is visible and responds to mouse events.

The Clip Content property

2019/07/17 19:17

Scaling Properties

These properties affect the placement and scaling of this group's children. Scaling can be used only with Absolute layout.



Scale: None

Autosize Content:

Content Width: auto

Content Height: auto

The Scaling properties in the Property Inspector

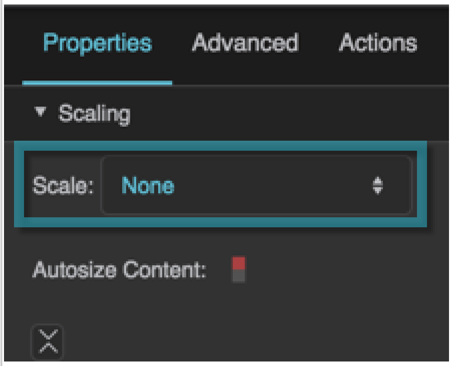
Click to display/hide all elements

Scale
Specifies the scaling behavior of children of this group when the group size changes. Only works when group layout is absolute.

None
Children do not scale when group size changes.

Fit ratio
Children scale while maintaining their aspect ratios.

Fit
Children scale without maintaining their aspect ratios.



Scale: None

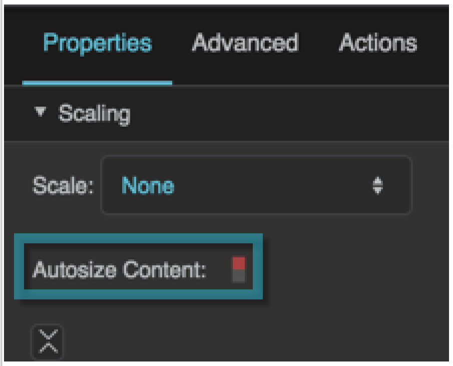
Autosize Content:

The Scale property

Autosize Content
Specifies whether this group's content inherits the group size.

TRUE
Content inherits the group size. This property can also be used to bring overflowing content into the bounds of the group, but only if the content overflows to the right or bottom of the group.

FALSE
Content does not inherit the group size.

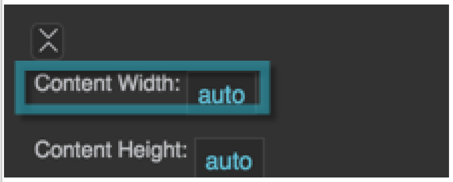


Scale: None

Autosize Content:

The Autosize Content property

Content Width
Returns the width of the group content before scaling happens, in pixels. Scaling is calculated based on this width. This is a read-only property, so changing it manually does nothing.

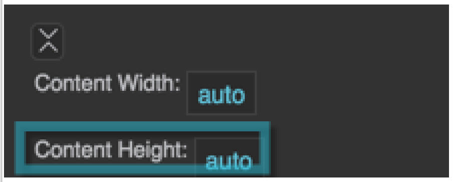


Content Width: auto

Content Height: auto

The Content Width property

Content Height
Returns the height of the group content before scaling happens, in pixels. Scaling is calculated based on this height. This is a read-only property, so changing it manually does nothing.



Content Width: auto

Content Height: auto

The Content Height property

2019/07/17 19:17

Previous: Image

Next: View Deck