UI Flexbox

Thanks for your purchasement! Feel free to request features or report bugs to info@jeffreylanters.nl

Making fully responsive Unity UI's has never been easier! UI Flexbox is bringing the most populair CSS layout tool to Unity.

The Flexible Box Module, usually referred to as flexbox, was designed as a one-dimensional layout model, and as a method that could offer space distribution between items in an interface and powerful alignment capabilities. This article gives an outline of the main features of flexbox, which we will be exploring in more detail in the rest of these guides.

Getting started

Create a new Rect Transform within your canvas and add the UIFlexbox component. Add a serie of Rect Transforms as children and make sure the number of Flex Items is matching. The Flex Items will be aligned automatically in the editor and on runtime when necessary.

Variables

1. Flex Container

1.1 Flex Direction

- `Row` (default) from left to right.
- `Column` from to to bottom.

This establishes the main-axis, thus defining the direction flex items are placed in the flex container. Flexbox is (aside from optional wrapping) a single-direction layout concept. Think of flex items as primarily laying out either in horizontal rows or vertical columns. The order will be defined by the order in the editor hierarchy.

1.2 Justify Content

- `Stretch` (default) items are stretch over the containers size.
- `Start` items are packed toward the start line.
- `Center` items are centered along the line.
- `End` items are packed toward to end line.

This defines the alignment along the main axis. It helps distribute extra free space left over when either all the flex items on a line are inflexible, or are flexible but have reached their maximum size. It also exerts some control over the alignment of items when they overflow the line. If not stretched, the size can be defined using the Item Size property.

1.3 Content spacing

- `None` items will have no spacing inbetween
- `Space Around` items are evenly distributed in the line with equal space around them. Note that visually the spaces aren't equal, since all the items have equal space on both sides. The first item will have one unit of space against the container edge, but two units of space between the next item because that next item has its own spacing that applies.
- `Space Between` items are distributed so that the spacing between any two items (and the space to the edges) is equal.

When using spacing you can define the value using the Spacing property.

2. Flex Items

2.1 Grow / Shrink

This defines the ability for a flex item to grow if necessary. It accepts a unitless value that serves as a proportion. It dictates what amount of the available space inside the flex container the item should take up.

If all items have flex-grow set to 1, the remaining space in the container will be distributed equally to all children. If one of the children has a value of 2, the remaining space would take up twice as much space as the others (or it will try to, at least). Negative numbers are invalid.

2.2 Align Self

- `Stretch` (default) item is stretch over the containers size.
- `Start` item is packed toward the start line.
- `Center` item is centered along the line.
- `End` item is packed toward to end line.

This allows the default alignment (or the one specified by align-items) to be overridden for individual flex items.