Flexboxes

Property	Values	Comments
Flex Container (parent)	The parent element in which flexible items are contained.	

`display:`	`flex`
	
	`inline-flex`
		The **flex** value generates a block level flex container, a.k.a. a box. This is the preferred and most commonly used value.
		The **inline-flex** value produces an inline flex container.

`flex-direction:`	`row`
	`column`
	`row-reverse`
	`column-reverse`
		The flex-direction property specifies the flow of the flexible items.
		The **row** value flows the flexible items horizontally across the flex container.
		The **column** value flows the flexible items vertically across the flex container.

`flex-wrap:`	`wrap`
	`nowrap`
	`wrap-reverse`
		The **nowrap** value specifies that the flexible items do not wrap. *This is the default value*.
		The **wrap** value allows the flexible items to wrap if necessary.
		The **wrap-reverse** value allows the flexible items to wrap, if necessary, in reverse order.

`flex-flow:`	`row nowrap`
	`row wrap`
	`row wrap-reverse`
	`column nowrap`
	`column wrap`
	`column wrap-reverse`
		The flex-flow property is a shortcut property that can stand in for both the flex-direction and the flex-wrap properties.
		The flex-flow property is assigned two space-separated values.
		The first value is the **flex direction**, and the second is the **flex wrap**.

`gap:`	Sets column and row spacing between the flex-items. One value applies to both rows and columns. With two space separated values, the 1st value applies to rows and the 2nd to columns.	
`row-gap:`	Sets row spacing between flex items.	
`column-gap:`	Sets column spacing between flex items.	

Property	Values	Comments
`justify-content:`	This property aligns or distributes space among items within the entire flex container along the main axis (as defined by the **flex-direction** property). For instance, if ROW is the assigned flex-direction value, then the main axis is the X or horizontal axis) **Positional alignment options** `center` positions flexible (child) items around the center `flex-start` positions flexible (child) items out from the starting point `flex-end` positions flexible (child) items back from the ending point **Distributed alignment** `space-between` Distributes items evenly across the main axis. The first item is flush with the start of the axis; the last is flush with the end. The remaining space is distributed evenly between the items. */ `space-around` As with the space-between value, this value distributes items evenly across the main axis, with the exception that items on either end have a half-size space on their outer edge. */ `space-evenly` Items are distributed so that the spacing between any two items (plus the spacing at the edges) is equal.	
`align-content:`	This property works similarly to the **justify-content** property above and has the same values. However, it differs in that it aligns or distributes space between items within the entire flex container along the cross axis. This property applies only to flex containers with either multiple rows or multiple columns. If the flexbox items are all contained within a single column or row, then this property has no effect.	
`place-content:`	Is a shorthand property for the **align-content** and **justify-content** properties. If the place-content property is assigned one value: the value applies to both align-content and justify-content properties. If the place-content property is assigned to two values: the first value applies to the align-content property, and the second value applies to the justify-content property.	
`align-items:`	This property aligns each item along its cross axis. If ROW is the assigned flex-direction value, then the cross axis is the Y or vertical axis, and this property will vertically align the flexible items within their row(s). This is useful when the length of each item along the cross axis is different. `center` positions flexible (child) items around the center */ `flex-start` positions flexible (child) items out from the starting point */ `flex-end` positions flexible (child) items back from the ending point */ `stretch` stretches each item to fill its cross axis */	

Property	Values	Comments
Flexible Items (children)	Every child of a flex container becomes a flexible item. Any text that is a direct child of a flex container is wrapped in an anonymous flexible item.	

`flex-basis:`	`length value`
	`or`
	`percentages`
	`or`
	`auto`
		The flex-basis property specifies the initial length of each flexible item along the main axis.
		The default value of this property is **auto**. The auto value determines the main axis length for the flexible item as set by the item's width or height property. If the item has no specified main axis length, then its content will determine its length.

	examples
	`flex-basis: 100px`
	`or`
	`flex-basis: 20%`
		The examples show two ways (units or percentages) to set the initial length of each flexible element.
		Using percentages will provide the most flexible layout. If a flexbox is set to no-wrap, it is best not to set the total width of all items to exceed 100%.

`flex-grow:`	`number value`	Zero is the default value. The higher the number, the more a flexible item will grow relative to its sibling. This property only applies if there is room in the parent item for the child item(s) to grow after the flex-basis length (whether absolute length or percentage) is applied. If the flex-basis length when applied to all flexible items is less than that of the flex container, then flex-grow specifies how much of the remaining space in the flex container should be assigned to the item (the flex grow factor).

	examples
	`nav {flex-grow: 1}`
	`article {flex-grow: 2}`
	`aside {flex-grow: 1}`
		In the examples to the left each flexible item element is a child of a flex container. Since the flex-basis value has not been set, the article element—which is set to a flex-grow value of 2—will try to grow twice as much as the other flexible items (nav and aside).

Property	Values	Comments
`flex-shrink:`	`number value`	One is the default value. The higher the number, the more a flexible item will shrink relative to its siblings. The property will not apply if the parent flex container's flex-wrap property is set to wrap or wrap-reverse. This property only applies if the parent flex container is smaller than the combined child flexible items after the flex-basis value is applied to them. If the size of all flex items is larger than the flex container, items shrink to fit according to their assigned flex-shrink value. The purpose of the flex-shrink property is to shrink flex items so that they don't overflow their flexbox container, which explains why the default value is not zero. If all flex items were set to a flex-shrink value of zero, then the flex items could potentially overflow their flex container.

	examples
	`nav {flex-shrink: 1}`
	`article {flex-shrink: 1}`
	`aside {flex-shrink: 3}`
		In the example to the left each flexible item element is a child of a flex container. The aside element, which is set to a flex-shrink value of 3, will try to shrink three times more than the other items.

`flex:`	`flex-grow value`
	`flex-shrink value`
	`flex-basis value`
		The flex property is a shortcut that allows you to set the values for flex-grow, flex-shrink, and flex-basis via a single property. These values must be ordered as listed to the left.

<table>
<tr><td>order:</td><td>number value</td><td>Flex items are, by default, displayed and laid out in the same order as they appear in the source document. The order property can be used to change this ordering.

By default, all flex items have an order value of 0.

Flex items given higher order values will appear later in the display order than flex items given lower order values.

You can assign negative values to the order property. Since a negative value is less than zero, flex items assigned a negative order value will precede items retaining the default value of zero.

Keep in mind that the flow direction will determine how the ordering looks.</td></tr>
<tr><td></td><td>examples
nav {order: 2}
article {order: 3}
aside {order: 1}</td><td>In this example the aside element will display first, the nav element second, and the article element last.</td></tr>
<tr><td>align-self</td><td colspan="2">This property has the same values as the align-items property but with the addition of the auto value. This property allows each individual flexible item to override the align-items property setting of its flexbox container. The default value is auto, which accepts rather than overrides the align-items value.</td></tr>
<tr><td colspan="3">For more info on flexboxes go to http://www.w3.org/TR/css-flexbox-1/</td></tr>
</table>