Overview on Flexbox and Grid: *by Subhash Jadhav*

1. **flexbox:**

- flexible box layout.

- Used to achieve responsive web designs.

- Provides web authors with control over how elements are positioned, aligned, and sized within their container.

- Children automatically become flex items

- ***display: flex***

# Flex container properties

|  |  |  |  |
| --- | --- | --- | --- |
| **Sr. No** | **Flex Properties** | **Values** | **Default Value** |
| 1 | ***flex-direction***  Defines in which direction the container wants to stack flex-items. | **1. column**: Stacks the flex items vertically (from top to bottom)  2. **column-reverse**: Stacks the flex items vertically (from bottom to top)  3**. row**: Stacks the flex items horizontally (from left to right)  4. **row-reverse**: Stacks the flex items horizontally (from right to left) | row |
| 2 | ***flex-wrap***  Specifies whether the flex items should wrap or not | 1. **wrap:** Specifies that the flex items will wrap if necessary  2. **nowrap**: Specifies that the flex items will not wrap  3. **wrap-reverse**: Specifies that the flex items will wrap if necessary, in reverse order | nowrap |
| 3 | ***flex-flow***  Shorthand property for setting both the ***flex-direction*** and ***flex-wrap*** |  | row nowrap |
| 4 | ***justify-content***  Used to align the flex items **horizontally** | 1. **center:** Aligns the flex items at the center of the container  2. **flex-start**: Aligns the flex items at the beginning of the container  3. **flex-end**: Aligns the flex items at the end of the container  4. **space-around**: Displays the flex-items with space before, between and after the lines.  5. **space-between**: Displays the flex items with space between the lines | flex-start |
| 5 | ***align-items***  Used to align the flex items **vertically** | 1. **center**: Aligns the flex itemsin the middle of the container  2. **flex-start**: Aligns the flex items at the top of the container  3. **flex-end**: Aligns the flex items at the bottom of the container  4. **stretch**: Stretches the flex items to fill the container  5. **baseline**: Aligns the flex items such as their baseline aligns (The baseline is a term used in European & West Asian typography meaning an imaginary line upon which the characters of a font rest) | stretch |
| 6 | ***align-content***  Used to align the flex lines (main axis) **vertically** | 1. **space-around**: Displays the flex lines with equal space between them  2. **space-around**: Displays the flex lines with space before, between and after them  3. **stretch**: Stretches the flex lines to take up the remaining space  4. **center**: Displays the flex lines in the middle of the container  5. **flex-start**: Displays the flex lines at the start of the container  6. **flex-end**: Displays the flex lines at the end of the container | stretch |

**# Difference between ‘align-items’ and ‘align-content’**

|  |  |  |
| --- | --- | --- |
| **Sr. No** | **align-items** | **align-content** |
| 1 | Determines how the items as whole are aligned within the container | Determines the spacing between the flex lines.  When there is only one flex line, align-content has no effect |

Note : `align-content` is similar to `align-items`, but instead of aligning flex items, it aligns flex lines.

# Flex items properties

|  |  |  |  |
| --- | --- | --- | --- |
| Sr. No | Properties | Values | Default Value |
| 1 | ***Order***  Specifies order of the flex items | Number  <div class=”flex-container”>  <div style=”order: 2”>1</div>  <div style=”order: 1”>2</div>  </div> | 0 |
| 2 | ***flex-grow***  Specifies how much a flex item will grow (horizontally) relative to the rest of the flex items | Number | 0 |
| 3 | ***flex-shrink***  Specifies how much a flex item will shrink (horizontally) relative to the rest of the flex items | Number | 1 |
| 4 | ***flex-basis***  Specifies initial length of the flex items | Number with units like px, vw etc. | auto |
| 5 | ***flex***  Shorthand property for the flex-grow, flex-shrink and flex-basis properties |  | 0 1 auto |
| 6 | ***align-self***  Specifies the alignment for the selected item inside the flexible container  Overrides the default alignment set by the container’s align-item property | 1. **center**: Aligns the flex itemsin the middle of the container  2. **flex-start**: Aligns the flex items at the top of the container  3. **flex-end**: Aligns the flex items at the bottom of the container  4. **stretch**: Stretches the flex items to fill the container  5. **baseline**: Aligns the flex items such as their baseline aligns (The baseline is a term used in European & West Asian typography meaning an imaginary line upon which the characters of a font rest) | auto |

2. **Grid:**

- Offers grid-based layout system (2D), with rows and columns, making it easier to design responsive web pages.

- A grid layout consists of a parent element, with one or more child elements.

- ***display: grid***

- Children of the grid container become grid items.

- **Grid columns:** The vertical lines of grid items are called columns.

- **Grid rows:** The horizontal lines of grid items are called rows.

- **Grid gaps:** The spaces between each column/row are called gaps.

- **Grid lines:** The lines between columns/rows are called column/row lines.

# Grid container Properties

|  |  |  |  |
| --- | --- | --- | --- |
| **Sr. No** | Properties | Values | Default Value |
| 1 | ***grid-template-columns***  Specifies the number (and the width) of columns in a grid layout  The values are a space separated list, where each value specifies the size of the respective column | 1. **none**: Columns are created if needed  2. **auto**: The size of the columns is determined by the size of the container and on the size of the content of items in the column  3. **max-content**: Sets the size of each column to depend on the largest item in the column  4. **min-content**: Sets the size of each column to depend on the smallest item in the column  5. **length**: Sets the size of the columns, by using legal length value. Ex. 30px | none |
| 2 | ***grid-template-rows***  Specifies the number (and the heights) of the rows in a grid layout  The values are a space separated list, where each value specifies the size of the respective row | 1. **none**: No size is set. Rows are created if needed  2. **auto**: The size of the rows is determined by the size of the container and on the size of the content of items in the row  3. **max-content**: Sets the size of each row to depend on the largest item in the row  4. **min-content**: Sets the size of each row to depend on the smallest item in the row  5. **length**: Sets the size of the rows, by using legal length value. Ex. 30px | none |
| 3 | ***Justify-content***  Used to align (**horizontally**) the whole grid inside the container | 1. **center:** Aligns the grid items at the center of the container  2. **start**: Aligns the grid items at the beginning of the container  3. **end**: Aligns the grid items at the end of the container  4. **space-around**: Displays the grid-items with space before, between and after the lines.  5. **space-between**: Displays the grid items with space between the lines | start |
| 4 | ***align-content***  Used to **vertically** align the whole grid inside the container | 1. **center:** Aligns the grid items at the center of the container  2. **start**: Aligns the grid items at the beginning of the container  3. **end**: Aligns the grid items at the end of the container  4. **space-around**: Displays the grid-items with space before, between and after the lines.  5. **space-between**: Displays the grid items with space between the lines | stretch |
| 5 | ***gap/grid-gap***  Defines the size of the gap between the rows and columns  Shorthand for row-gap and column-gap  Formerly known as grid-gap | gap/grid-gap : row-gap column-gap | normal normal |
| 6 | ***grid-auto-columns/rows***  Sets a default size for all columns/rows  The **grid-template-columns/rows** property overrides this property | grid-auto-columns/rows : auto | max-content | min-content | length | auto |
| 7 | ***grid-auto-flow***  Controls how auto placed items get inserted in the grid | 1. **row**: Places items by filling each row  2. **column**: Places items by filling each column  3. **dense**: Place items to fill any holes in the grid  4. **row dense**: Places items by filling each row, and fill any holes in the grid  5. **column dense**: Places items by filling each column, and fill any holes in the grid | row |
| 8 | ***grid-column-start/end***  Defines on which column-line the item will start/end | 1. **auto**: The item will be placed following the flow  2. **span n**: Specifies the number of columns the item will span  3. column-number | auto |
| 9 | ***grid-column***  Shorthand property for **grid-column-start** and **grid-column-end** | grid-column: grid-column-start / grid-column-end | auto / auto |
| 10 | ***grid-row-start/end***  Defines on which row-line the item will start/end | 1. **auto**: The item will be placed following the flow  2. **span n**: Specifies the number of rows the item will span  3. row-number | auto |
| 11 | ***grid-row***  Shorthand property for **grid-row-start** and **grid-row-end** | grid-row: grid-row-start / grid-row-end | auto / auto |
| 12 | ***grid-template-areas***  Specifies areas within the grid layout (row-wise)  You can name grid items by using the **grid-area** property, and then reference to the name in the **grid-template-areas** property | 1. **none**: No named grid areas  2. itemnames: A sequence that specifies how each columns and rows should display | none |
| 13 | ***grid-template***  Shorthand for **grid-template-row**, **grid-template-columns**, and **grid-template-areas** | 1. **none**: No specific sizing of columns or rows  2. **grid-template rows / grid-template-columns** : Specifies the size(s) of the columns and rows  3. **grid-template-areas**: Specifies the grid layout using named items | none none none |

# Grid item properties

|  |  |  |  |
| --- | --- | --- | --- |
| Sr. No | Properties | Values | Default Value |
| 1 | ***grid-area***  Shorthand property for **grid-row-start**, **grid-column-start**, **grid-row-end**, and **grid-column-end** | grid-area: grid-row-start / grid-column-start / grid-row-end / grid-column-end; | auto / auto / auto / auto |