**Flexbox: -**

In Today’s world we see web design is very essential thing.

It’s very important to design responsive layout. But using basic CSS properties take lot of efforts to do, and also very tedious thing to do.

But Flexbox makes it easier to design flexible responsive layout structure. As its name suggest, it provides flexibility to the element so that element is adjusted automatically in available space of viewport.

Now we can see how to this.

Those elements we want our flex-items wrap them in parent element and provide display property to the parent as display: flex then child elements become flexible items.

There are various properties available for us to apply on that flex items for making responsive layout.

Now we see some important properties of flex items

|  |  |  |
| --- | --- | --- |
| **Property** | **Value** | **Description** |
| **flex-grow** | Grow factor eg.1 | Grow factor is used for how much element is grow relative to rest elements in available free space |
| **flex-shrink** | Shrink factor eg.1 | Shrink is used for how much element is shrink relative to the rest elements |
| **flex-basis** | Length unit | Initial length of the item |
| **flex-direction** | row | Display flexible items horizontally (default value) |
| **flex-direction** | row-reverse | Display items horizontally as row but in reverse order |
| **flex-direction** | column | Display flexible items vertically as column |
| **flex-direction** | Column-reverse | Display flexible items vertically as column but in reverse order |
| **flex-wrap** | nowrap | Flexible items will not wrap in container if size reduced (default value) |
| **flex-wrap** | wrap | Flexible items will wrap if necessary, in container. |
| **flex-wrap** | wrap-reverse | Flexible items will wrap if necessary, in container. And wrap items in reverse order |
| **align-items** | stretch | flex-items are stretched to fit the container (default value) |
| **align-items** | center | Items are positioned at the center of container |
| **align-items** | Flex-start | Items are positioned at the beginning of the container |
| **align-items** | Flex-end | Items are positioned at the end of the container |
| **align-items** | baseline | Items are positioned at the end of the container (note that when font size is differed then you get difference in flex-start and baseline) |
| **Align-self** | Same as align-items | This property is same as align-items but it is applied on individual flex-items |
| **Justify-content** | Flex-start | Items are positioned at the beginning of the container (default value) |
| **Justify-content** | Flex-end | Items are positioned at the end of the container |
| **Justify-content** | center | Items are positioned at the center of the container |
| **Justify-content** | Space-between | Items have space between them |
| **Justify-content** | Space-around | Items have space before, between and after |
| **Justify-content** | Space-evenly | Items have equal space around them |
| **align-content** | Same as justify-content | Work same as justify-content. only difference is justify-content work along main-axis of container while align-content is on cross-axis |

**Grid: -**

Before grid there is no any functionality for design two-dimensional layout. Grid is latest layout optimized for two-dimensional layouts. It is the best and ideal layout for precise positioning of elements.

Apply display: grid to the parent element so that this element becomes grid-container and child elements becomes grid items.

Now we see some properties of grid layout:

|  |  |  |
| --- | --- | --- |
| **Property** | **Value** | **Description** |
| **grid-template-rows** | 1fr 1fr…1fr | Explicitly sets the number of rows with their mentioned size. (1fr means take all available space) |
| **grid-template-rows** | repeat(n,1fr) | Same as above but instead of writing long code it reduces the repetitive code |
| **grid-template-columns** | 1fr 1fr…1fr | Same like above but explicitly sets columns |
| **grid-gap** | Length value | Sets the gutter (gutter=gap in between grid items) |
| **grid-row-start** | Row-number | On which row grid item is start to displaying |
| **grid-row-end** | span n | Number of rows grid item will span. (Default value is n=1) |
| **grid-row-end** | Row-line number | On which row to end the display of the item |
| **Grid-column-start/**  **Grid-column-end** | Same as grid-row-start/grid-row-end | Same as grid-row-start/ grid-row-end |
| **grid-area** | grid-row-start/ grid-column-start/ grid-row-end/ grid-column-end | Shorthand property for grid-row-start grid-column-start grid-row-end and grid-column-end |
| **grid-area** | item Name | grid-area is also used to assign name to grid item. Which is used as reference in grid-template-area |
| **grid-template-area** | Use grid-area reference here with layout structure | Specifies areas within the grid layout |
| **grid-auto-flow** | row | If we explicitly define rows and columns but you left with some items then for that element there is implicit grid concept. This value defines row for that extra item. (Default value) |
| **grid-auto-flow** | column | Implicit grid instead of rows here column is defined for extra items. |
| **grid-auto-flow** | dense | Optional value that can help keep your grid compact and prevent lots of gaps due to inconsistent sized grid items. |
| **grid-auto-rows** | Length value | It defines the row size for implicit grid. |
| **grid-auto-columns** | Length value | It defines the column size for implicit grid. |
| **align-items** | stretch | Aligns grid items along the block(column) axis. Stretch fill the whole cell. (Default value) |
| **align-items** | start | Aligns grid items at the start of column axis. |
| **align-items** | end | Aligns grid items at the end of column axis. |
| **align-items** | center | Aligns grid items at the center of the cell. |
| **align-items** | baseline | Aligns grid items along text baseline. |
| **justify-items** | stretch | Aligns grid items along the inline(row) axis. Stretch fill the whole cell. (Default value) |
| **justify-items** | start | Aligns grid items at the start of row axis. |
| **justify-items** | end | Aligns grid items at the start of row axis. |
| **justify-items** | center | Aligns grid items at the center of the cell. |
| **justify-items** | baseline | Aligns grid items along text baseline. |
| **justify-content** | start | Sometimes the total size of your grid might be less than the size of its grid container. This could happen if all of your grid items are sized with non-flexible units like px. In this case you can set the alignment of the grid within the grid container. This property aligns the grid along the inline (row) axis.  Align the grid at starting of the container. |
| **justify-content** | end | Align the grid end of the container. |
| **justify-content** | stretch | Resizes the grid items to allow the grid to fill the full width of the grid container. |
| **justify-content** | center | Aligns the grid in the center of the grid container |
| **justify-content** | space-around | places an even amount of space between each grid item, with half-sized spaces on the far ends |
| **justify-content** | space-between | places an even amount of space between each grid item, with no space on the far ends. |
| **justify-content** | space-evenly | places an even amount of space between each grid item, with including the far ends |
| **align-content** | start | Sometimes the total size of your grid might be less than the size of its grid container. This could happen if all of your grid items are sized with non-flexible units like px. In this case you can set the alignment of the grid within the grid container. This property aligns the grid along the block (column) axis.  Align the grid at starting of the container. |
| **align-content** | end | Align the grid end of the container. |
| **align-content** | stretch | Resizes the grid items to allow the grid to fill the full height of the grid container. |
| **align-content** | center | Aligns the grid in the center of the grid container |
| **align-content** | space-around | places an even amount of space between each grid item, with half-sized spaces on the far ends |
| **align-content** | space-between | places an even amount of space between each grid item, with no space on the far ends. |
| **align-content** | space-evenly | places an even amount of space between each grid item, with including the far ends |