**CSS Layout Module Properties**

-Sakthivel Sengodan

**Position Property:**

Positions:- HTML elements are positioned static by default.

There are five ***position property*** ***values***. Which are:-

1. Static
2. Relative
3. Absolute
4. Fixed
5. Sticky (toggling of relative and fixed)

Static positioned elements are not affected by the top, bottom, left, and right properties. An element with position: static; is not positioned in any special way; it is always positioned according to the normal flow of the page.

An element with position: Relative; is positioned relative to its normal position. Setting the top, right, bottom, and left properties of a relatively-positioned element will cause it to be adjusted away from its normal position. Other content will not be adjusted to fit into any gap left by the element.

An element with position: Absolute; is positioned relative to the nearest positioned ancestor (instead of positioned relative to the viewport, like fixed). However; if an absolute positioned element has no positioned ancestors, it uses the document body, and moves along with page scrolling. Note: A "positioned" element is one whose position is anything except static.

An element with position: fixed; is positioned relative to the viewport, which means it always stays in the same place even if the page is scrolled. The top, right, bottom, and left properties are used to position the element. A fixed element does not leave a gap in the page where it would normally have been located.

An element with position: sticky; is positioned based on the user's scroll position. A sticky element toggles between relative and fixed, depending on the scroll position. It is positioned relative until a given offset position is met in the viewport - then it "sticks" in place (like position:fixed)

Example:- position: relative

**Display Flex:**

The Flexible Box Layout Module, makes it easier to design flexible responsive layout structure without using float or positioning. To start using the Flexbox model, you need to first define a ***flex container*** (parent element) and it should have display: flex.

The flex container (parent element) has six ***properties***. Which are:

1. Flex-direction
2. Flex-wrap
3. Flex-flow (Flex-direction Flex-wrap)
4. Justify-content
5. Align-items
6. Align-content

The flex-direction property defines in which direction the container wants to stack the flex items. It has four values. row, row-reverse, column and column-reverse

The row value stacks the flex items horizontally (from left to right)

The row-reverse value stacks the flex items horizontally (but from right to left)

The column value stacks the flex items vertically (from top to bottom)

The column-reverse value stacks the flex items vertically (but from bottom to top)

The flex-wrap property specifies whether the flex items should wrap or not. It has three values. wrap, nowrap and wrap-reverse

The wrap value specifies that the flex items will wrap if necessary

The nowrap value specifies that the flex items will not wrap (this is default)

The wrap-reverse value specifies that the flexible items will wrap if necessary, in reverse order

The flex-flow property is a shorthand property for setting both the flex-direction and flex-wrap properties. Example: flex-flow: row wrap;

The justify-content property is used to align the flex items. It has five values. center, flex-start, flex-end, space-around and space-between

The center value aligns the flex items at the center of the container

The flex-start value aligns the flex items at the beginning of the container (this is default)

The flex-end value aligns the flex items at the end of the container

The space-around value displays the flex items with space before, between, and after the lines

The space-between value displays the flex items with space between the lines

The align-items property is used to align the flex items vertically. It has five values. center, flex-start, flex-end, stretch and baseline

The center value aligns the flex items in the middle of the container

The flex-start value aligns the flex items at the top of the container

The flex-end value aligns the flex items at the bottom of the container

The stretch value stretches the flex items to fill the container (this is default)

The baseline value aligns the flex items such as their baselines aligns

The align-content property is used to align the flex lines. It has six values. center, flex-start, flex-end, stretch, space-between and space-around

The center value displays display the flex lines in the middle of the container

The flex-start value displays the flex lines at the start of the container

The flex-end value displays the flex lines at the end of the container

The stretch value stretches the flex lines to take up the remaining space (this is default)

The space-between value displays the flex lines with equal space between them

The space-around value displays the flex lines with space before, between, and after them

Perfect Centering, Set both the justify-content and align-items properties to center, and the flex item will be perfectly centered

The flex item (child element) has six ***properties***. The direct child elements of a flex container automatically becomes flexible (flex) items. Which are:

1. Order
2. Flex-grow
3. Flex-shrink
4. Flex-basis
5. Flex
6. Align-self

The order property specifies the order of the flex items. The order value must be a number, default value is 0. The order property can change the order of the flex items.

The flex-grow property specifies how much a flex item will grow relative to the rest of the flex items. The value must be a number, default value is 0. Make the flex item grow defined times faster than the other flex items.

The flex-shrink property specifies how much a flex item will shrink relative to the rest of the flex items. The value must be a number, default value is 1. Do not let the flex item shrink as much as (defined) the other flex items.

The flex-basis property specifies the initial length of a flex item. Set the initial length of the flex item to some values px/em/%.

The flex property is a shorthand property for the flex-grow, flex-shrink, and flex-basis properties. Make the flex item not growable (0), not shrinkable (0), and with an initial length of some pixels.

The align-self property specifies the alignment for the selected item inside the flexible container. The align-self property overrides the default alignment set by the container's align-items property. It has three values center, flex-start and flex-end.

**Display Grid**:

The CSS Grid Layout Module offers a grid-based layout system, with rows and columns, making it easier to design web pages without having to use floats and positioning. To start using the Grid layout model, you need to first define a ***grid container*** (parent element) and it should have display: grid.

An HTML element becomes a grid container when its display property is set to grid or inline-grid. All direct children of the grid container automatically become ***grid items***.

Grid container has eight ***properties***:

1. grid-column-gap
2. grid-row-gap
3. grid-gap: grid-column-gap grid-row-gap
4. grid-template-columns
5. grid-template-rows
6. justify-content
7. align-content
8. grid-template-areas

The vertical lines of grid items are called columns

The horizontal lines of grid items are called rows.

The spaces between each column/row are called grid column/row gaps. The values px/em/%

The grid-template-columns property defines the number of columns in your grid layout, and it can define the width of each column. The value is a space-separated-list, where each value defines the length of the respective column. The values auto/1fr/px/em/%

The grid-template-rows property defines the height of each row. The value is a space-separated-list, where each value defines the height of the respective row. The values auto/1fr/px/em/%

The justify-content property is used to align the whole grid inside the container. The values space-evenly, space-around, space-between, center, start and end

The align-content property is used to vertically align the whole grid inside the container. The values space-evenly, space-around, space-between, center, start and end

The grid-template-areas each row is defined by apostrophes (' ') The columns in each row is defined inside the apostrophes, separated by a space. It holds grid-area’s name which is defined in grid-item.

Grid item has seven ***properties***:

1. grid-column-start
2. grid-column-end
3. grid-column: grid-column-start/ grid-column-end
4. grid-row-start
5. grid-row-end
6. grid-row: grid-row-start/grid-row-end
7. grid-area

The lines between columns are called column lines

The lines between rows are called row lines

Lines needs to calculate from top to bottom or left to right.

Example: The grid has three column then it is having four column lines. It applies to row as well.

The grid-area property can also be used to assign names to grid items. The values Custom name