# Font

There are many properties related to the font, such as the face, weight, style, etc. These properties allow you to change the style or complete look of your text.

## Font-Family

font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;

## Font-Style

font-style: italic;

## Font-Variant

font-variant: small-caps;

## Font-Weight

font-weight: bold;

## Font-Size

font-size: larger;

## Font

font: style variant weight size family;

# Text

Text properties allow one to manipulate alignment, spacing, decoration, indentation, etc., in the document.

## Text-Align

text-align: justify;

## Letter-Spacing

letter-spacing: .15em;

## Text-Decoration

text-decoration: underline;

## Word-Spacing

word-spacing: 0.25em;

## Text-Transform

text-transform: uppercase;

## Text-Indent

text-indent: 0.5cm;

## Line-Height

line-height: normal;

# Background

As the name suggests, these properties are related to background, i.e., you can change the color, image, position, size, etc., of the document.

## Background-Image

background-image: url("Path");

## Background-Position

background-position: right top;

## Background-Size

background-size: cover;

## Background-Repeat

background-repeat: no-repeat;

## Background-Attachment

background-attachment: scroll;

## Background-Color

background-color: fuchsia;

## Background

background: color image repeat attachment position;

# Border

Border properties are used to change the style, radius, color, etc., of buttons or other items of the document.

## Border-Width

border-width: 5px;

## Border-Style

border-style: solid;

## Border-Color

border-color: aqua;

## Border-Radius

border-radius: 15px;

## Border

border: width style color;

# Box Model

In laymen's terms, the CSS box model is a container that wraps around every HTML element. It consists of margins, borders, padding, and the actual content. It is used to create the design and layout of web pages.

## Float

float: none;

## Clear

clear: both;

## Display

display: block;

## Height

height: fit-content;

## Width

width: auto;

## Margin

margin: top right bottom left;

## Padding

padding: top right bottom left;

## Overflow

overflow: hidden;

## Visibility

visibility: visible;

# Colors

With the help of the color property, one can give color to text, shape, or any other object.

## Color

color: cornsilk;

## Opacity

opacity: 4;

# Template Layout

Specifies the visual look of the content inside a template

## Box-Align

box-align : start;

## Box-Direction

box-direction : normal;

## Box-Flex

box-flex : normal;

## Box-Flex-Group

box-flex-group : 2;

## Box-Orient

box-orient : inline;

## Box-Pack

box-pack : justify;

## Box-Sizing

box-sizing : margin-box;

## max-width

max-width: 800px;

## min-width

min-width: 500px;

## max-height

max-height: 100px;

## min-height

min-height: 80px;

# Table

Table properties are used to give style to the tables in the document. You can change many things like border spacing, table layout, caption, etc.

## Border-Collapse

border-collapse: separate;

## Empty-Cells

empty-cells: show;

## Border-Spacing

border-spacing: 2px;

## Table-Layout

table-layout: auto;

## Caption-Side

caption-side: bottom;

# Columns

These properties are used explicitly with columns of the tables, and they are used to give the table an incredible look.

## Column-Count

column-count : 10;

## Column-Gap

column-gap : 5px;

## Column-rule-width

column-rule-width : medium;

## Column-rule-style

column-rule-style : dotted ;

## Column-rule-color

column-rule-color : black;

## Column-width

column-width : 10px;

## Column-span

column-span : all;

# List & Markers

List and marker properties are used to customize lists in the document.

## List-style-type

list-style-type: square;

## List-style-position

list-style-position : 20px;

## List-style-image

list-style-image : url(8image.gif8);

## Marker-offset

marker-offset : auto;

# Animations

CSS animations allow one to animate transitions or other media files on the web page.

## Animation-name

animation-name : myanimation;

## Animation-duration

animation-duration : 10s;

## Animation-timing-function

animation-timing-function : ease;

## Animation-delay

animation-delay : 5ms;

## Animation-iteration-count

animation-iteration-count : 3;

## Animation-direction

animation-direction : normal;

## Animation-play-state

animation-play-state : running;

## Animation-fill-mode

animation-fill-mode : both;

# Transitions

Transitions let you define the transition between two states of an element.

## Transition-property

transition-property: none;

## Transition-duration

transition-duration : 2s;

## Transition-timing-function

transition-timing-function: ease-in-out;

## Transition-delay

transition-delay : 20ms;

# CSS Flexbox

Flexbox is a layout of CSS that lets you format HTML easily. Flexbox makes it simple to align items vertically and horizontally using rows and columns. Items will "flex" to different sizes to fill the space. And overall, it makes the responsive design more manageable.

# Parent Properties (flex container)

## display

display: flex;

## flex-direction

flex-direction: row | row-reverse | column | column-reverse;

## flex-wrap

flex-wrap: nowrap | wrap | wrap-reverse;

## flex-flow

flex-flow: column wrap;

## justify-content

justify-content: flex-start | flex-end | center | space-between | space-arou

## align-items

align-items: stretch | flex-start | flex-end | center | baseline | first bas

## align-content

align-content: flex-start | flex-end | center | space-between | space-around

# Child Properties (flex items)

## order

order: 5; /\* default is 0 \*/

## flex-grow

flex-grow: 4; /\* default 0 \*/

## flex-shrink

flex-shrink: 3; /\* default 1 \*/

## flex-basis

flex-basis: | auto; /\* default auto \*/

## flex shorthand

flex: none | [ <'flex-grow'> <'flex-shrink'>? || <'flex-basis'> ]

## align-self

align-self: auto | flex-start | flex-end | center | baseline | stretch;

# CSS Grid

Grid layout is a 2-Dimensional grid system to CSS that creates complex responsive web design layouts more easily and consistently across browsers.

# Parent Properties (Grid container)

## display

display: grid | inline-grid;

## grid-template-columns

grid-template-columns: 12px 12px 12px;

## grid-template-rows

grid-template-rows: 8px auto 12px;

## grid-template

grid-template: none | <grid-template-rows> / <grid-template-columns>;

## column-gap

column-gap: <line-size>;

## row-gap

row-gap: <line-size>;

## grid-column-gap

grid-column-gap: <line-size>;

## grid-row-gap

grid-row-gap: <line-size>;

## gap shorthand

gap: <grid-row-gap> <grid-column-gap>;

## grid-gap shorthand

grid-gap: <grid-row-gap> <grid-column-gap>;

## justify-items

justify-items: start | end | center | stretch;

## align-items

align-items: start | end | center | stretch;

## place-items

place-items: center;

## justify-content

justify-content: start | end | center | stretch | space-around | space-betwe

## align-content

align-content: start | end | center | stretch | space-around | space-between

## place-content

place-content: <align-content> / <justify-content> ;

## grid-auto-columns

grid-auto-columns: <track-size> ...;

## grid-auto-rows

grid-auto-rows: <track-size> ...;

## grid-auto-flow

grid-auto-flow: row | column | row dense | column dense;

# Child Properties (Grid items)

## grid-column-start

grid-column-start: <number> | <name> | span <number> | span <name> | auto;

## grid-column-end

grid-column-end: <number> | <name> | span <number> | span <name> | auto;

## grid-row-start

grid-row-start: <number> | <name> | span <number> | span <name> | auto;

## grid-row-end

grid-row-end: <number> | <name> | span <number> | span <name> | auto;

## grid-column shorthand

grid-column: <start-line> / <end-line> | <start-line> / span <value>;

## grid-row shorthand

grid-row: <start-line> / <end-line> | <start-line> / span <value>;

## grid-area

grid-area: <name> | <row-start> / <column-start> / <row-end> / <column-end>;

## justify-self

justify-self: start | end | center | stretch;

## align-self

align-self: start | end | center | stretch;

## place-self

place-self: center;