UNIT 3: CASCADING STYLE SHEET & CSS 3

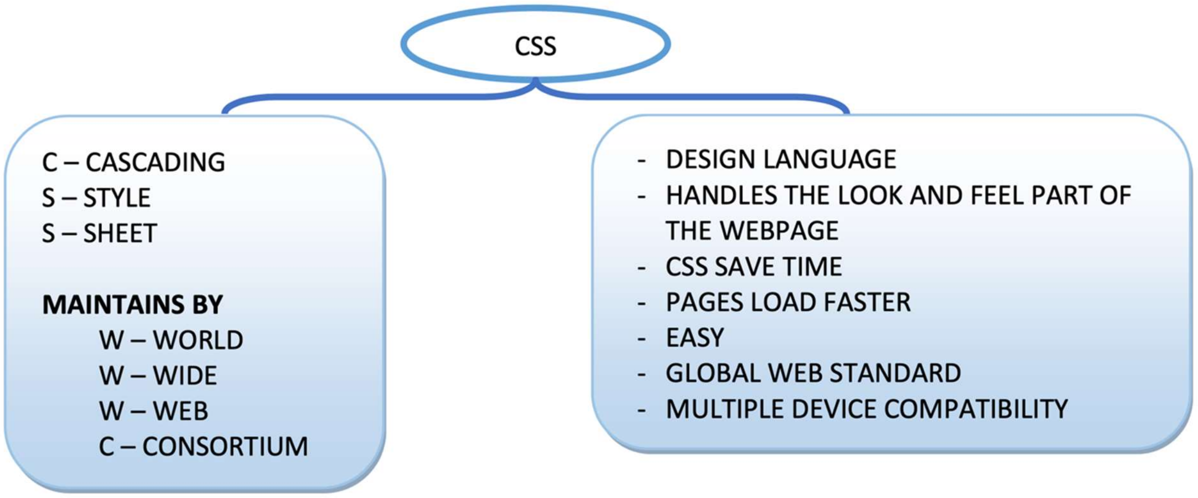
CS-04 Networking & Internet Environment

- **Introduction to css & types of css**
- **Class & id selector**
- **Css font property**
- **Css text property**
- **Css background property**
- **Css margin property**
- **Css list property**
- **Css 3 border property**
- **Css 3 gradient property**
- **Css 3 drop shadow property**
- **Css 3 2d & 3d transform property**
- **Css 3 transition property**
- **Css 3 box sizing property**
- **Css 3 position property**
- **Css 3 media query**
- **Css flexbox properties (display, flex-direction, flex-wrap, flex-flow, justify- content, align-items, align-content, gap row-gap, column-gap)**
- **Css advance properties (display, flex-direction, flex-wrap, flex-flow, justify- content, align-items, align-content, gap row-gap, column-gap)**
- **How to use google fonts & custom fonts (@font-face)**
- **Bem naming convention**

-: Assignment 3: -

1. **Css stands for?**
2. **Explain css syntax.**
3. **Explain comments in css**
4. **Inside which tag css code is written.**
5. **Explain types of css**
6. **Explain css font and text property wth example**
7. **Explain background property with example**
8. **Explain border and transation property of css3 with example.**
9. **Explain transformation property.**
10. **Explain class and id selector with example.**
11. **Explain margin & padding property with exmaple.**
12. **Explain box and position property.**
13. **Explain media query**
14. **Explain flexbox**
15. **Explain following property: display, flex-direction, flex-wrap, flex-flow, justify- content, align-items, align-content, gap row-gap, column-gap**
16. **Explain @ font-face**
17. **Explain bem naming convention with example.**

INTRODUCTION TO CSS



- CSS stands for Cascading Style Sheets
- Styles define how to display HTML elements
- Styles are normally stored in Style Sheets
- CSS describes how HTML elements are to be displayed on screen, paper, or in other media
- CSS saves a lot of work. It can control the layout of multiple web pages all at once
- External style sheets are stored in CSS files
- CSS Solved a Big Problem
- HTML was NEVER intended to contain tags for formatting a web page!
- HTML was created to describe the content of a web page, like:
  <h1>This is a heading</h1> <p>This isa paragraph.</p>
- When tags like <font>, and color attributes were added to the HTML 3.2 specification, it started a nightmare for web developers.
- To solve this problem, the World Wide Web Consortium (W3C) created CSS.
- CSS removed the style formatting from the HTML page!

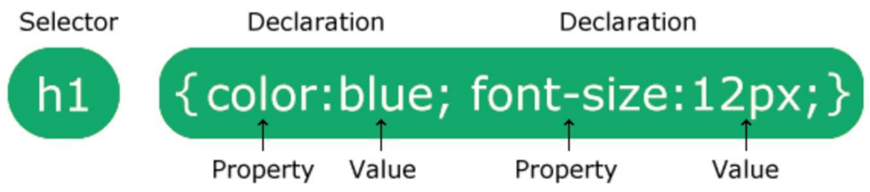
Advantages of CSS

- **CSS saves time** − You can write CSS once and then reuse same sheet in multiple HTML pages.
- **Pages load faster** − If you are using CSS, you do not need to write HTML tag attributes every time. Just write one CSS rule of a tag and apply it to all the occurrences of that tag. So less code means faster download times.
- **Easy maintenance** − To make a global change, simply change the style, and all elements in all the web pages will be updated automatically.
- **Superior styles to HTML** − CSS has a much wider array of attributes than HTML, so you can give a far better look to your HTML page in comparison to HTML attributes.
- **Multiple Device Compatibility** − Style sheets allow content to be optimized for more than one type of device.
- **Global web standards** − Now HTML attributes are being deprecated and it is being recommended to use CSS.

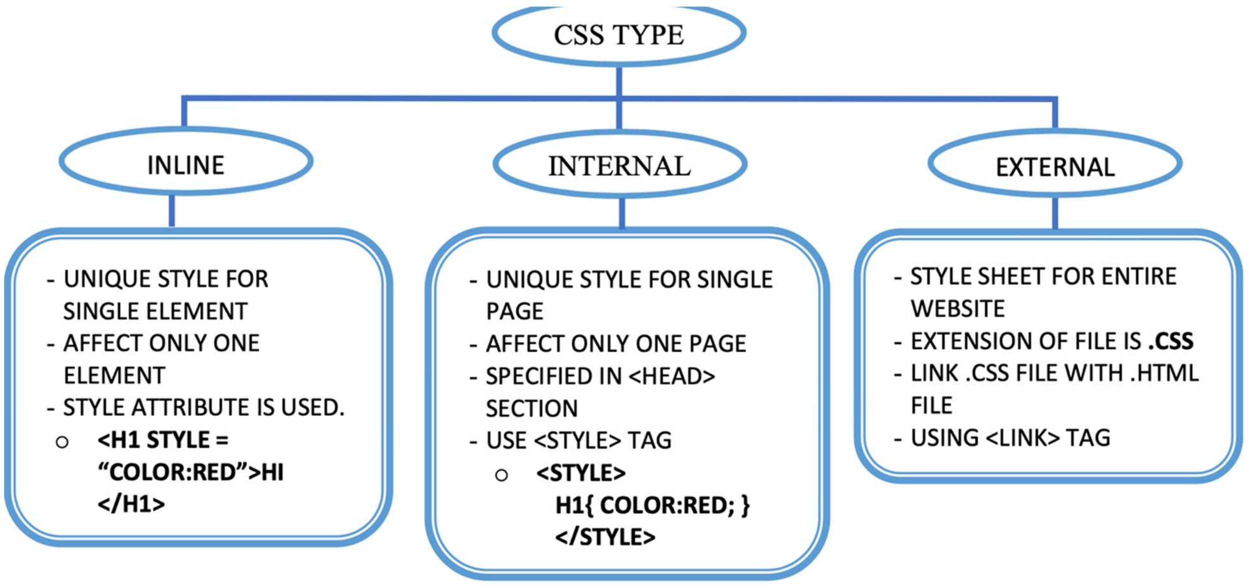
Syntax:

- The basic goal of the Cascading Stylesheet (CSS) language is to allow a browser engine to paint elements of the page with specific features, like colors, positioning, or decorations. The CSS syntax reflects this goal and its basic building blocks are:

- The property which is an identifier, that is a human-readable name, that defines which feature is considered.
- The value which describe how the feature must be handled by the engine. Each property has a set of valid values, defined by a formal grammar, as well as a semantic meaning, implemented by the browser engine.



TYPES OF CSS


CSS TYPE
INLINE
INTERNAL
EXTERNAL
- UNIQUE STYLE FOR SINGLE ELEMENT
- AFFECT ONLY ONE ELEMENT
- STYLE ATTRIBUTE IS USED.
○ <H1 STYLE = "COLOR:RED">HI </H1>
- UNIQUE STYLE FOR SINGLE PAGE
- AFFECT ONLY ONE PAGE
- SPECIFIED IN <HEAD> SECTION
- USE <STYLE> TAG
○ <STYLE> H1{ COLOR:RED; } </STYLE>
- STYLE SHEET FOR ENTIRE WEBSITE
- EXTENSION OF FILE IS .CSS
- LINK .CSS FILE WITH .HTML FILE
- USING <LINK> TAG

When a browser reads a style sheet, it will format the document according to it. There are three ways of inserting a style sheet:

- External Style Sheet
- Internal Style Sheet
- Inline Style

External Style Sheet

- An external style sheet is ideal when the style is applied to many pages.
- With an external style sheet, you can change the look of an entire Web site by
- changing one file.
- Each page must link to the style sheet using the <link> tag. o The <link> tag goes inside the head section:

```
<head>
   <link rel="stylesheet" type="text/css" href="mystyle.css" />
</head>
```

- The browser will read the style definitions from the file mystyle.css, and format the
- document according to it.

- An external style sheet can be written in any text editor. The file should not contain
- any html tags. Your style sheet should be saved with a .css extension.
- An example of a style sheet file is shown below:
    - Hr {color: sienna}
    - p {margin-left: 20px}
    - body {background-image: url("images/back40.gif")}

Internal Style Sheet

- An internal style sheet should be used when a single document has a unique style. You define internal styles in the head section by using the <style> tag, like this:

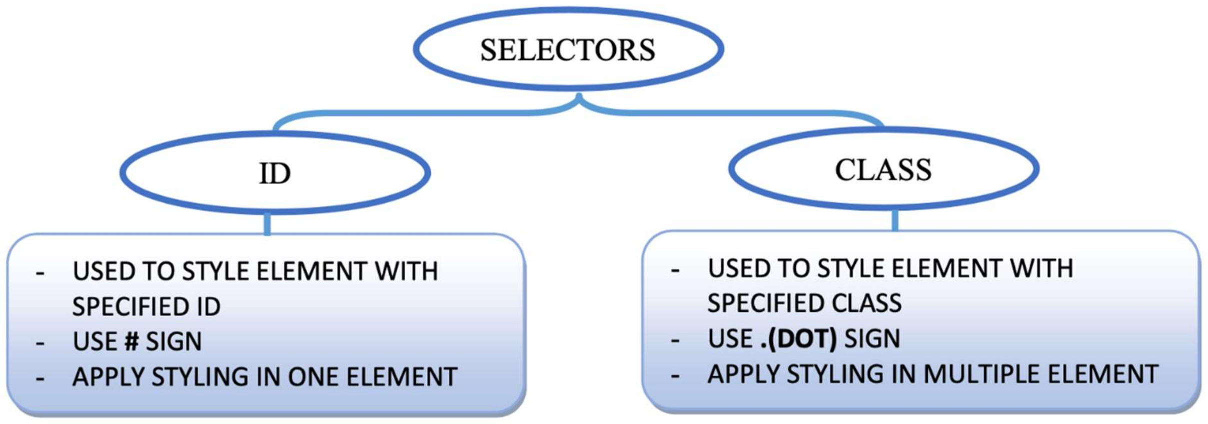
```
<head>
<style type="text/css">
hr {color: sienna}
p {margin-left: 20px}
body {background-image: url("images/back40.gif")}
</style>
</head>
```

- The browser will now read the style definitions, and format the document according to it.

Inline Style Sheet

- An inline style loses many of the advantages of style sheets by mixing content with presentation. Use this method sparingly, such as when a style is to be applied to a single occurrence of an element.
- To use inline styles you use the style attribute in the relevant tag. The style attribute can contain any CSS property.
- The example shows how to change the color and the left margin of a paragraph:
- <p style="color: sienna; margin-left: 20px"> This is a paragraph</p>

CSS SELECTORS


SELECTORS
ID
CLASS
- USED TO STYLE ELEMENT WITH SPECIFIED ID
- USE # SIGN
- APPLY STYLING IN ONE ELEMENT
- USED TO STYLE ELEMENT WITH SPECIFIED CLASS
- USE .(DOT) SIGN
- APPLY STYLING IN MULTIPLE ELEMENT

Class Selector

- The .class selector selects elements with a specific class attribute.
- To select elements with a specific class, write a period (.) character, followed by the name of the class.
- You can also specify that only specific HTML elements should be affected by a class. To do this, start with the element name, then write the period (.) character, followed by the name of the class.

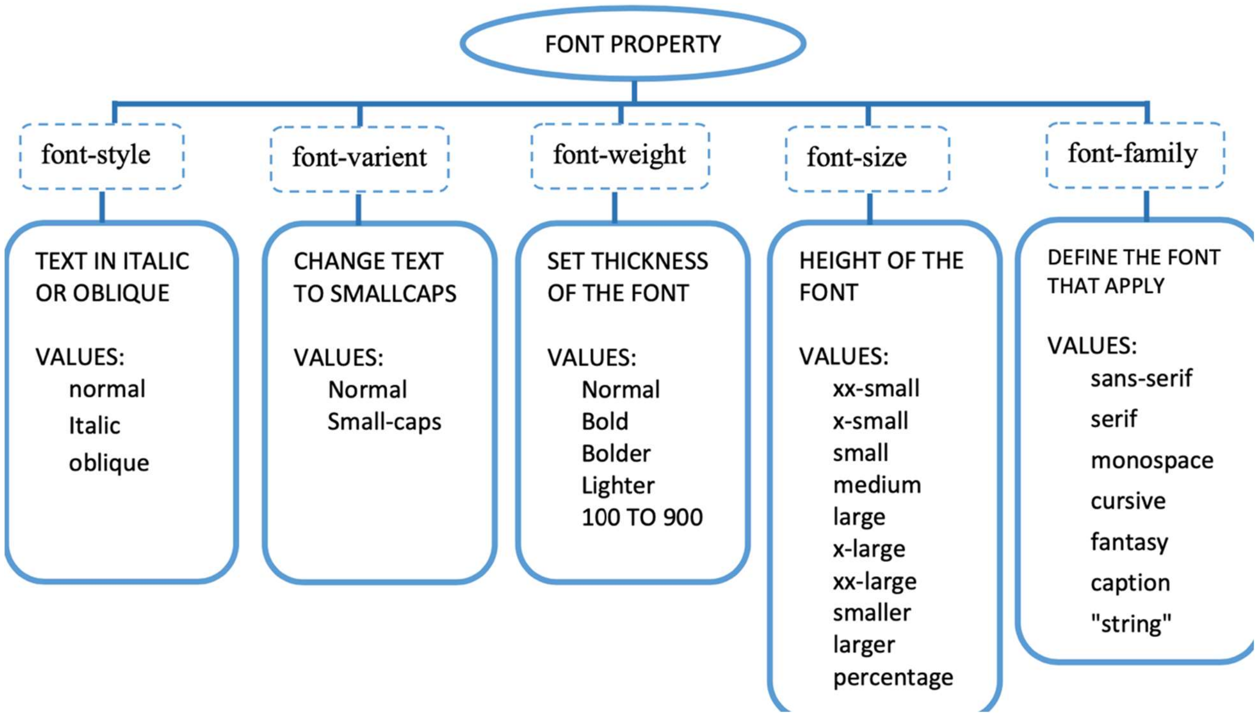
Example:

```
p.hometown {
background-color: yellow;
}
```

ID Selector

- The id selector uses the id attribute of an HTML element to select a specific element.
- The id of an element is unique within a page, so the id selector is used to select one unique element!
- You can also define styles for HTML elements with the id selector. The id selector is defined as a #.
- The style rule below will match the element that has an id attribute with a value of "green": #green {color: green}

CSS FONT PROPERTIES



FONT PROPERTY
font-style
font-varient
font-weight
font-size
font-family
TEXT IN ITALIC OR OBLIQUE
VALUES: normal Italic oblique
CHANGE TEXT TO SMALLCAPS
VALUES: Normal Small-caps
SET THICKNESS OF THE FONT
VALUES: Normal Bold Bolder Lighter 100 TO 900
HEIGHT OF THE FONT
VALUES: xx-small x-small small medium large x-large xx-large smaller larger percentage
DEFINE THE FONT THAT APPLY
VALUES: sans-serif serif monospace cursive fantasy caption "string"

- CSS Font property is used to control the look of texts. By the use of CSS font property you can change the text size, color, style and more. You have already studied how to make text bold or underlined. Here, you will also know how to resize your font using percentage.
- These are some important font attributes:
- **CSS Font color:** This property is used to change the color of the text. (standalone attribute)
- **CSS Font family:** This property is used to change the face of the font.
- **CSS Font size:** This property is used to increase or decrease the size of the font.
- **CSS Font style:** This property is used to make the font bold, italic or oblique.
- **CSS Font variant:** This property creates a small-caps effect.
- **CSS Font weight:** This property is used to increase or decrease the boldness and lightness of the font.

1. CSS Font Color

- **CSS font color** is a standalone attribute in CSS although it seems that it is a part of CSS fonts. It is used to change the color of the text.
- There are three different formats to define a color:
  - By a color name

  - By hexadecimal value
  - By RGB
- h1 { color: red; }
- h2 { color: #9000A1; }
- p { color:rgb(0, 220, 98); }

2. CSS Font Family

- CSS font family can be divided in two types:
  - **Generic family**: It includes Serif, Sans-serif, and Monospace.
  - **Font family:** It specifies the font family name like Arial, New Times Roman etc.
  - **Serif:** Serif fonts include small lines at the end of characters. Example of serif: Times new roman, Georgia etc.
  - **Sans-serif:** A sans-serif font doesn't include the small lines at the end of characters. Example of Sans-serif: Arial, Verdana etc.
- h1 { font-family: sans-serif; }
- h2 { font-family: serif; }
- p { font-family: monospace; }

3. CSS Font Size

- CSS font size property is used to change the size of the font.
- These are the possible values that can be used to set the font size

Font Size Value	Description
xx-small	used to display the extremely small text size.
x-small	used to display the extra small text size.
small	used to display small text size.
medium	used to display medium text size.
large	used to display large text size.
x-large	used to display extra large text size.
xx-large	used to display extremely large text size.
smaller	used to display comparatively smaller text size.
larger	used to display comparatively larger text size.
size in pixels or %	used to set value in percentage or in pixels.

- <p style="font-size:xx-small;"> This font size is extremely small.</p>
- <p style="font-size:x-small;"> This font size is extra small</p>
- <p style="font-size:small;"> This font size is small</p>
- <p style="font-size:medium;"> This font size is medium. </p>
- <p style="font-size:large;"> This font size is large. </p>
- <p style="font-size:x-large;"> This font size is extra large. </p>
- <p style="font-size:xx-large;"> This font size is extremely large. </p>
- <p style="font-size:smaller;"> This font size is smaller. </p>
- <p style="font-size:larger;"> This font size is larger. </p>
- <p style="font-size:200%;"> This font size is set on 200%. </p>

- <p style="font-size:20px;"> This font size is 20 pixels. </p>

4. CSS Font Style

- CSS Font style property defines what type of font you want to display. It may be italic, oblique, or normal.
- h2 { font-style: italic; }
- h3 { font-style: oblique; }
- h4 { font-style: normal; }

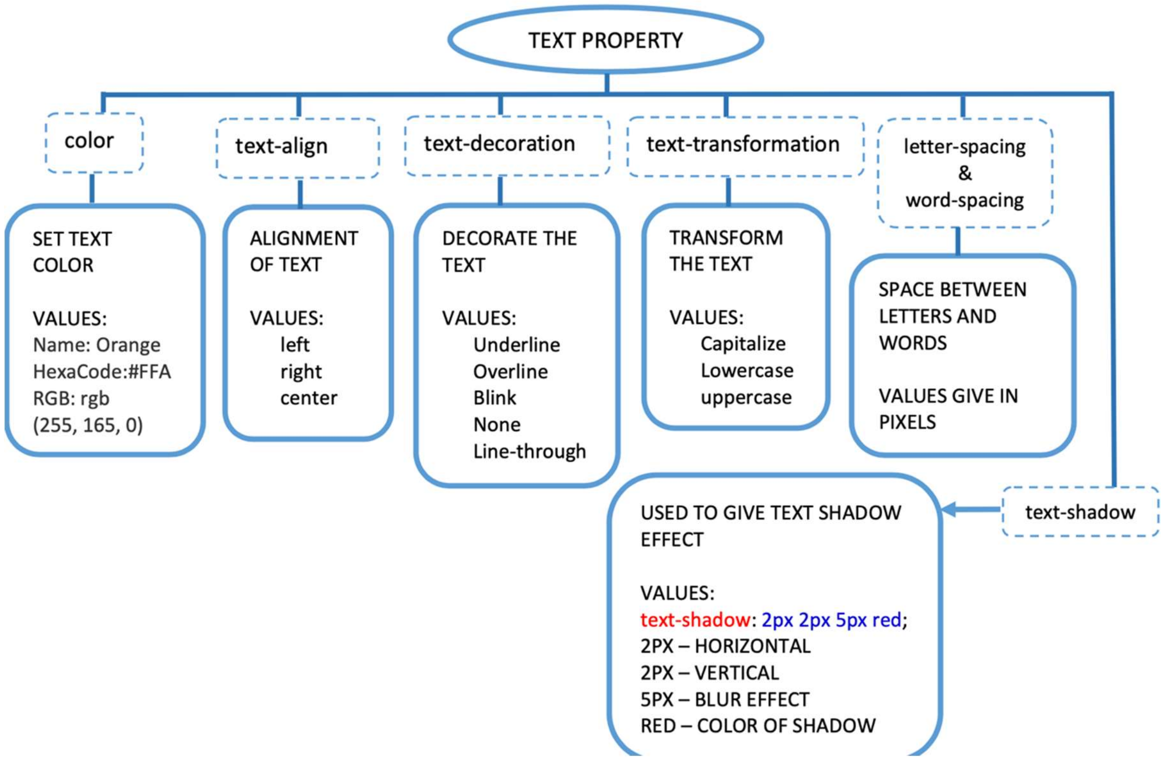
5. CSS Font Variant

- CSS font variant property specifies how to set font variant of an element. It may be normal and small-caps.
- p { font-variant: small-caps; }
- h3 { font-variant: normal; }

6. CSS Font Weight

- CSS font weight property defines the weight of the font and specify that how bold a font is. The possible values of font weight may be normal, bold, bolder, lighter or number (100, 200..... upto 900).
- <p style="font-weight:bold;">This font is bold.</p>
- <p style="font-weight:bolder;">This font is bolder.</p>
- <p style="font-weight:lighter;">This font is lighter.</p>
- <p style="font-weight:100;">This font is 100 weight.</p>
- <p style="font-weight:200;">This font is 200 weight.</p>
- <p style="font-weight:300;">This font is 300 weight.</p>
- <p style="font-weight:600;">This font is 600 weight.</p>
- <p style="font-weight:700;">This font is 700 weight.</p>
- <p style="font-weight:800;">This font is 800 weight.</p>
- <p style="font-weight:900;">This font is 900 weight.</p>

CSS TEXT PROPERTY



- The **color** property is used to set the color of a text.
- The **direction** property is used to set the text direction.
- The **letter-spacing** property is used to add or subtract space between the letters that make up a word.
- The **text-align** property is used to align the text of a document.
- The **text-decoration** property is used to underline, overline, and strikethrough text.
- The **text-transform** property is used to capitalize text or convert text to uppercase or lowercase letters.
- The **text-indent** property is used to indent the text of a paragraph.
- The **text-shadow** property is used to set the text shadow around a text.
- The **word-spacing** property is used to add or subtract space between the words of a sentence.
- **Word spacing** is used to specify the space between the words of the line.

1.TEXT DIRECTION

- The following example demonstrates how to set the direction of a text. Possible values are ltr or rtl.
- <p style = "direction:rtl;">

2. LETTER SPACING

- This property is used to specify the space between the characters of the text. The size can be given in px.
- <p style = "letter-spacing:size";>

2.TEXT ALIGNMENT

- Text alignment property is used to set the horizontal alignment of the text.
- The text can be set to left, right, centered and justified alignment.
- In justified alignment, line is stretched such that left and right margins are straight.
- <p style = "text-align:alignment type;">

3.TEXT DECORATION

- Text decoration is used to add or remove decorations from the text.
- Text decoration can be underline, overline, line-through or none.
- <p style = "text-decoration:decoration type;”>

4.TEXT TRANSFORMATION

- Text transformation property is used to change the case of text, uppercase or lowercase.
- Text transformation can be uppercase, lowercase or captitalise .
- Capitalise is used to change the first letter of each word to uppercase.
- <p style = "<p style = "text-decoration:decoration type;">

5.TEXT INDENTATION

- Text indentation property is used to indent the first line of the paragraph.
- The size can be in px, cm, pt.
- <p style = "text-indent:80px;”>

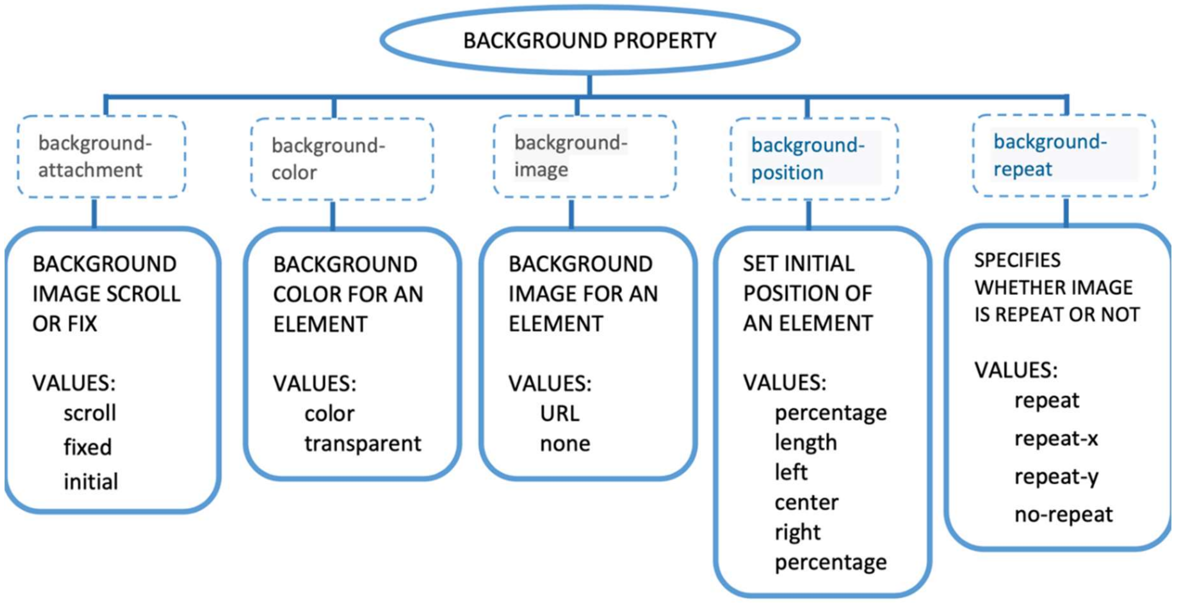
6.TEXT SHADOW

- Text shadow property is used to add shadow to the text.
- You can specify the horizontal size, vertical size and shadow color for the text.
- <p style = "text-shadow:horizontal size vertical size color name;”>

7.WORD SPACING

- Word spacing is used to specify the space between the words of the line.
- The size can be given in px.
- <p style = "word-spacing:size;”>

CSS BACKGROUND PROPERTY


BACKGROUND PROPERTY
background-attachment
background-color
background-image
background-position
background-repeat
BACKGROUND IMAGE SCROLL OR FIX
VALUES: scroll fixed initial
BACKGROUND COLOR FOR AN ELEMENT
VALUES: color transparent
BACKGROUND IMAGE FOR AN ELEMENT
VALUES: URL none
SET INITIAL POSITION OF AN ELEMENT
VALUES: percentage length left center right percentage
SPECIFIES WHETHER IMAGE IS REPEAT OR NOT
VALUES: repeat repeat-x repeat-y no-repeat

- CSS background property is used to define the background effects on element. There are 5 CSS background properties that affects the HTML elements:
- background-color
- background-image
- background-repeat
- background-attachment
- background-position

1.CSS background-color

- The background-color property is used to specify the background color of the element.
- p{   background-color: #b0d4de; }

2. CSS background-image

- The background-image property is used to set an image as a background of an element. By default the image covers the entire element.
- p{ background-image: url("paper1.gif"); }

3. CSS background-repeat

- By default, the background-image property repeats the background image horizontally and vertically. Some images are repeated only horizontally or vertically.
- body { background-repeat: repeat-x; }

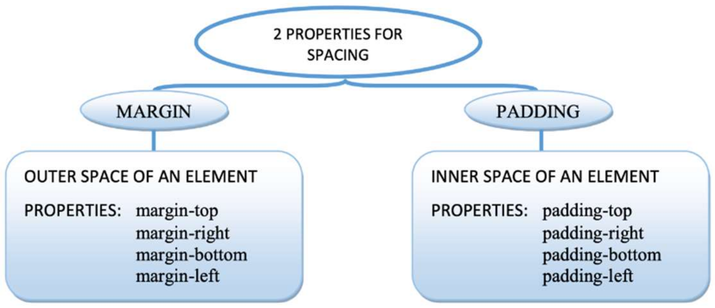
4. CSS background-attachment

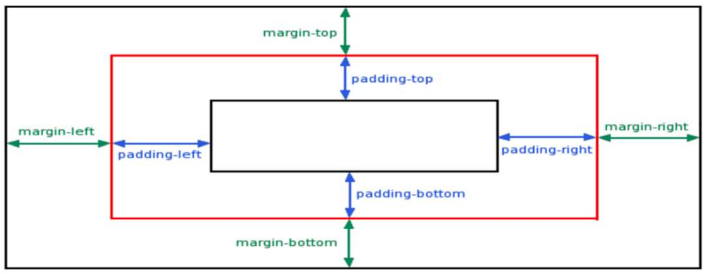
- The background-attachment property is used to specify if the background image is fixed or scroll with the rest of the page in browser window. If you set fixed the background image then the image will not move during scrolling in the browser.
- p{ background-attachment: fixed; }

5. CSS background-position

- The background-position property is used to define the initial position of the background image. By default, the background image is placed on the top-left of the webpage.
- You can set the following positions:
    - center
    - top
    - bottom
    - left
    - right
- p{ background-attachment: fixed; }

MARGIN AND PADDING PROPERTIES OF CSS


2 PROPERTIES FOR SPACING
MARGIN
PADDING
OUTER SPACE OF AN ELEMENT
PROPERTIES: margin-top margin-right margin-bottom margin-left
INNER SPACE OF AN ELEMENT
PROPERTIES: padding-top padding-right padding-bottom padding-left



MARGIN PROPERTY

- CSS Margin property is used to define the space around elements. It is completely transparent and doesn't have any background color. It clears an area around the element.
- Top, bottom, left and right margin can be changed independently using separate properties. You can also change all properties at once by using shorthand margin property.
- **Syntax**
  margin: [length | percentage | auto]| initial | inherit
- **Example**
  h1{margin-left:25px;}
  p{margin: 50px 100px; }
- This shorthand notation can take one, two, three, or four whitespace separated values.
- If one value is set, this margin applies to all 4 sides. If two values are set, the first value applies to top and bottom, the second value applies to the right and left side. Three values apply to the top, horizontal (i.e. right and left) and bottom side.
- Four values apply to the top, right, bottom, left side in that order.

Property	Description
margin	This property is used to set all the properties in one declaration.
margin-left	it is used to set left margin of an element.
margin-right	It is used to set right margin of an element.
margin-top	It is used to set top margin of an element.
margin-bottom	It is used to set bottom margin of an element.

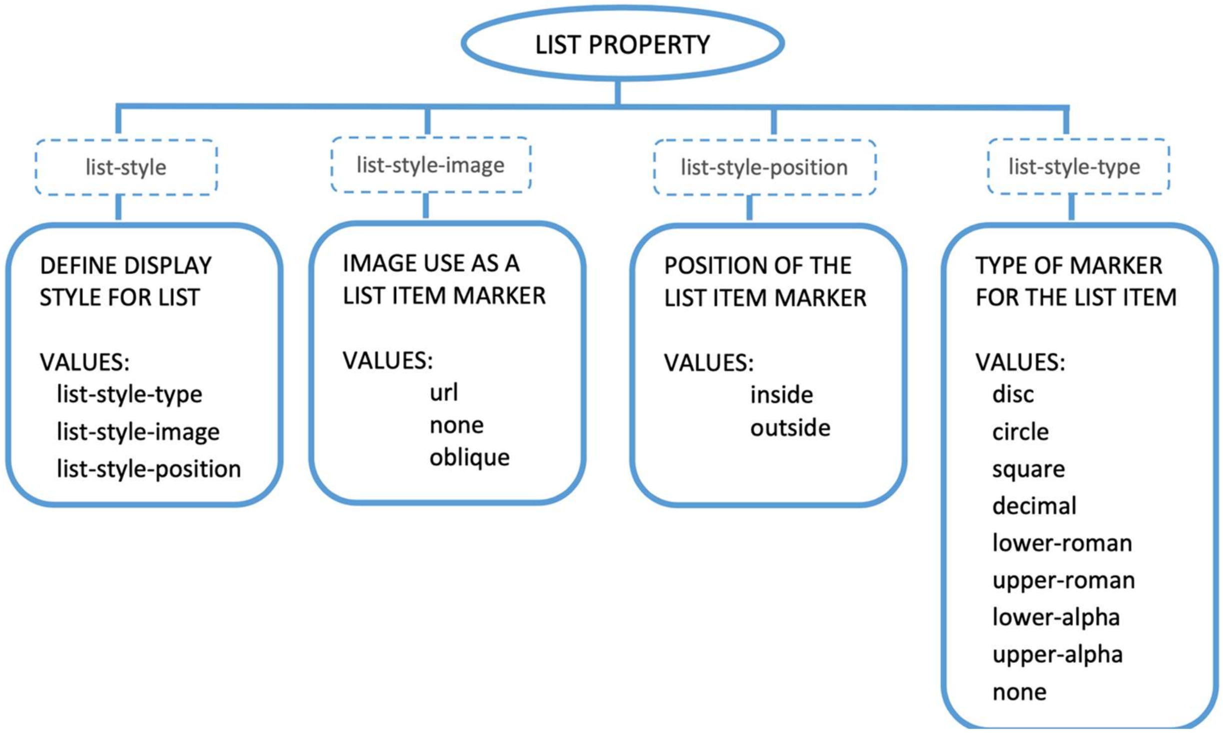
CSS Padding

- CSS Padding property is used to define the space between the element content and the element border.
- It is different from CSS margin in the way that CSS margin defines the space around elements. CSS padding is affected by the background colors. It clears an area around the content.
- Top, bottom, left and right padding can be changed independently using separate properties. You can also change all properties at once by using shorthand padding property.
- **Syntax**
  padding: [length | percentage] 1 to 4 values | initial | inherit
- **Example**
  p.one {
  padding-right: 20px; }
  p.two { padding: 35px 15px;}
- This shorthand notation can take one, two, three, or four whitespace separated values.
- If one value is set, this padding applies to all 4 sides.
- If two values are set, the first value applies to top and bottom, the second value
- applies to the right and left side.
- Three values apply to the top, horizontal (i.e. right and left) and bottom side. o Four values apply to the top, right, bottom, left side in that order.

Property	Description
padding	It is used to set all the padding properties in one declaration.
padding-left	It is used to set left padding of an element.
padding-right	It is used to set right padding of an element.
padding-top	It is used to set top padding of an element.

padding-bottom	It is used to set bottom padding of an element.

CSS LIST PROPERTIES


LIST PROPERTY
list-style
list-style-image
list-style-position
list-style-type
DEFINE DISPLAY STYLE FOR LIST
VALUES: list-style-type list-style-image list-style-position
IMAGE USE AS A LIST ITEM MARKER
VALUES: url none oblique
POSITION OF THE LIST ITEM MARKER
VALUES: inside outside
TYPE OF MARKER FOR THE LIST ITEM
VALUES: disc circle square decimal lower-roman upper-roman lower-alpha upper-alpha none

TYPES OF HTML LISTS

There are three different types of list in HTML:

- Unordered lists — A list of items, where every list items are marked with bullets.
- Ordered lists — A list of items, where each list items are marked with numbers.
- Definition list — A list of items, with a description of each item.
- Lists can be classified as ordered lists and unordered lists. In ordered lists, marking of the list items is with alphabet and numbers, whereas in unordered lists, the list items are marked using bullets.

We can style the lists using CSS. CSS list properties allow us to:

- Set the distance between the text and the marker in the list.
- Specify an image for the marker instead of using the number or bullet point.
- Control the marker appearance and shape.
- Place the marker outside or inside the box that contains the list items.
- Set the background colors to list items and lists.

The CSS properties to style the lists are given as follows:

- **list-style-type**: This property is responsible for controlling the appearance and shape of the marker.
- **list-style-image**: It sets an image for the marker instead of the number or a bullet point.
- **list-style-position**: It specifies the position of the marker.
- **list-style**: It is the shorthand property of the above properties.
- **marker-offset**: It is used to specify the distance between the text and the marker. It is unsupported in IE6 or Netscape 7.

list-style-type:

- It allows us to change the default list type of marker to any other type such as square, circle, roman numerals, Latin letters, and many more. By default, the ordered list items are numbered

with Arabic numerals (1, 2, 3, etc.), and the items in an unordered list are marked with round bullets (•).
- **Example**: list-style-type:decimal;

list-style-position:
- It represents whether the appearing of the marker is inside or outside of the box containing the bullet points. It includes two values.
- **inside**: It means that the bullet points will be in the list item. In this, if the text goes on the second line, then the text will be wrap under the marker.
- **outside**: It represents that the bullet points will be outside the list item. It is the default value.
- **Example**: list-style-position:inside;

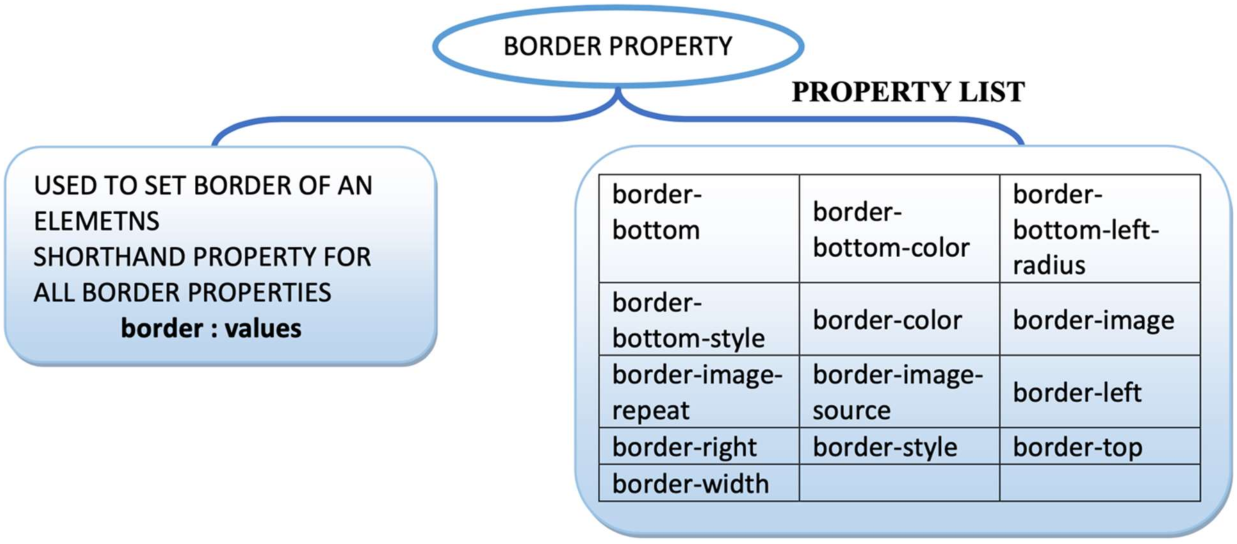
list-style-image:
- It specifies an image as the marker. Using this property, we can set the image bullets. Its syntax is similar to the background-image property. If it does not find the corresponding image, the default bullets will be used.
- **Example:** list-style-image: url(img.png);

list-style:
- It is the shorthand property that is used to set all list properties in one expression. The order of the values of this property is type, position, and image. But if any property value is missing, then the default value will be inserted.
- **Example:** list-style: lower-alpha inside url(img.png);

CSS 3

- Cascading Style Sheets (CSS) is a style sheet language used for describing the look and formatting of a document written in a markup language.CSS3 is a latest standard of css earlier versions(CSS2).The main difference between css2 and css3 is follows
  - Media Queries
  - Namespaces
  - Selectors Level 3
  - Color
- CSS3 is collaboration of CSS2 specifications and new specifications, we can called this collaboration is module. Some of the modules are shown below
  - Selectors
  - Box Model
  - Backgrounds
  - Image Values and Replaced Content
  - Text Effects
  - 2D Transformations
  - 3D Transformations
  - Animations
  - Multiple Column Layout
  - User Interface

CSS 3 BORDER PROPERTY


BORDER PROPERTY
PROPERTY LIST
USED TO SET BORDER OF AN ELEMETNS
SHORTHAND PROPERTY FOR ALL BORDER PROPERTIES
border : values

border-bottom	border-bottom-color	border-bottom-left-radius
border-bottom-style	border-color	border-image
border-image-repeat	border-image-source	border-left
border-right	border-style	border-top
border-width		

- The CSS border properties are use to specify the style, color and size of the border of an element. The CSS border properties are given below.
- border-style
- border-color
- border-width
- border-radius

1. CSS border-style

- The Border style property is used to specify the border type which you want to display on the web page.
- There are some border style values which are used with border-style property to define a border.

Value	Description
none	It doesn't define any border.
dotted	It is used to define a dotted border.
dashed	It is used to define a dashed border.
solid	It is used to define a solid border.
double	It defines two borders wIth the same border-width value.
groove	It defines a 3d grooved border. effect is generated according to border-color value.
ridge	It defines a 3d ridged border. effect is generated according to border-color value.
inset	It defines a 3d inset border. effect is generated according to border-color value.
outset	It defines a 3d outset border. effect is generated according to border-color value.

2. CSS border-width

- The border-width property is used to set the border's width. It is set in pixels. You can also use the one of the three pre-defined values, thin, medium or thick to set the width of the border.
- **Note**: The border-width property is not used alone. It is always used with other border properties like "border-style" property to set the border first otherwise it will not work.
- **Syntax**: border-width: 5px;

3. CSS border-color

- There are three methods to set the color of the border.
- Name: It specifies the color name. For example: "red".

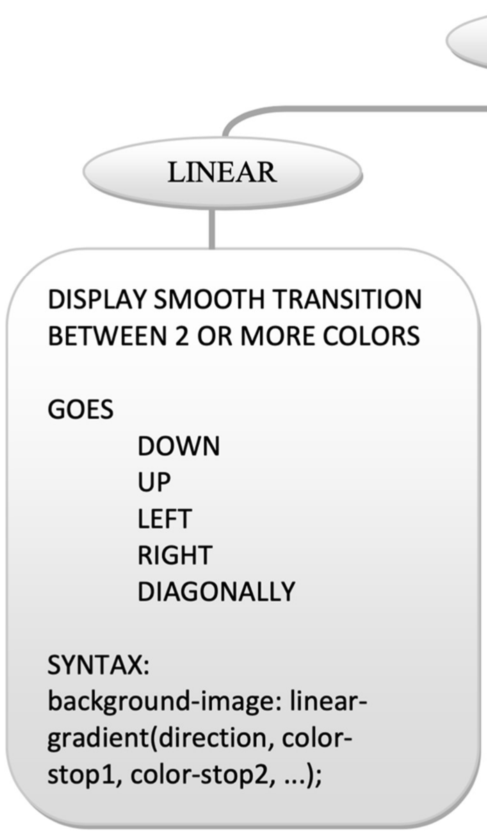
- RGB: It specifies the RGB value of the color. For example: "rgb(255,0,0)".
- Hex: It specifies the hex value of the color. For example: "#ff0000".
- There is also a border color named "transparent". If the border color is not set it is inherited from the color property of the element.
- **Note**: The border-color property is not used alone. It is always used with other border properties like "border-style" property to set the border first otherwise it will not work.

4. CSS border-radius

- This CSS property sets the rounded borders and provides the rounded corners around an element, tags, or div. It defines the radius of the corners of an element.
- It is shorthand for border top-left-radius, border-top-right-radius, border-bottom-right-radius and border-bottom-left-radius. It gives the rounded shape to the corners of the border of an element.
- We can specify the border for all four corners of the box in a single declaration using the border-radius. The values of this property can be defined in percentage or length units.
- If we provide a single value (such as border-radius: 30px;) to this property, it will set all corners to the same value.
- When we specify two values (such as border-radius: 20% 10% ;), then the first value will be used for the top-left and bottom-right corners, and the second value will be used for the top-right and bottom-left corners.
- When we use three values (such as border-radius: 10% 30% 20%;) then the first value will be used for the top-left corner, the second value will be applied on top-right, and bottom-left corners and the third value will be applied to the bottom-right corner.
- Similarly, when this property has four values (border-radius: 10% 30% 20% 40%;) then the first value will be the radius of top-left, the second value will be used for the top-right, the third value will be applied on bottom-right, and the fourth value is used for bottom-left.
- **Syntax:** border-radius: 1-4 length | % / 1-4 length | % | inherit | initial;

Property	Description
border	Sets all the border properties in one declaration
border-bottom	Sets all the bottom border properties in one declaration
border-bottom-color	Sets the color of the bottom border
border-bottom-style	Sets the style of the bottom border
border-bottom-width	Sets the width of the bottom border
border-color	Sets the color of the four borders
border-left	Sets all the left border properties in one declaration
border-left-color	Sets the color of the left border
border-left-style	Sets the style of the left border
border-left-width	Sets the width of the left border
border-right	Sets all the right border properties in one declaration
border-right-color	Sets the color of the right border
border-right-style	Sets the style of the right border
border-right-width	Sets the width of the right border
border-style	Sets the style of the four borders
border-top	Sets all the top border properties in one declaration
border-top-color	Sets the color of the top border
border-top-style	Sets the style of the top border
border-top-width	Sets the width of the top border
border-width	Sets the width of the four borders

CSS 3 GRADIENT


GRADIENT
LINEAR
RADIAL
DISPLAY SMOOTH TRANSITION BETWEEN 2 OR MORE COLORS
GOES
DOWN
UP
LEFT
RIGHT
DIAGONALLY
SYNTAX:
background-image: linear-gradient(direction, color-stop1, color-stop2, ...);
DISPLAY SMOOTH TRANSITION BETWEEN 2 OR MORE COLORS
DEFINE BY THEIR CENTER
SYNTAX:
background-image: radial-gradient(shape size at position, start-color, ..., last-color);

- The CSS3 gradient feature provides a flexible solution to generate smooth transitions between two or more colors.
- Earlier, to achieve such effect we had to use the images.
- Using CSS3 gradients you can reduce the download time and saves the bandwidth usages.
- The elements with gradients can be scaled up or down to any extent without losing the quality,
- also the output will render much faster because it is generated by the browser.
- Gradients are available in two styles: linear and radial.

Creating CSS3 Linear Gradients

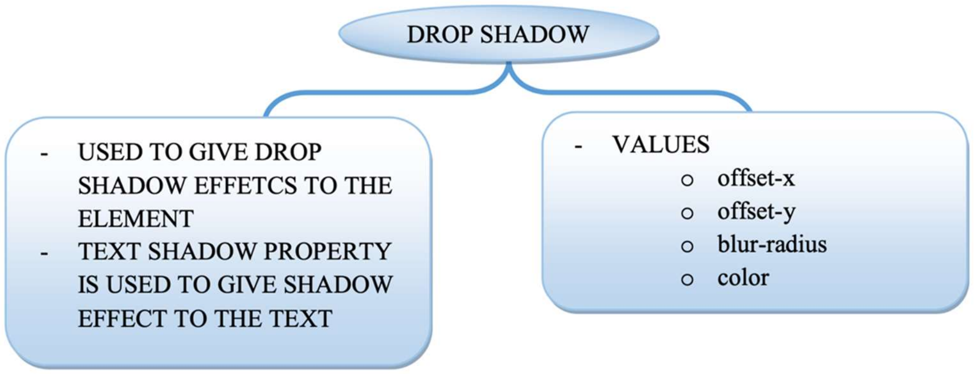
- To create a linear gradient you must define at least two color stops.
- However to create more complex gradient effects you can define more color stops. Color stops are the colors you want to render smooth transitions among.
- You can also set a starting point and a direction (or an angle) along which the gradient effect is applied.
- The basic syntax of creating the linear gradients using the keywords can be given with:
- linear-gradient(direction, color-stop1, color-stop2, ...)

Creating CSS3 Radial Gradients

- In a radial gradient color emerge from a single point and smoothly spread outward in a circular or elliptical shape rather than fading from one color to another in a single
- direction as with linear gradients.
- The basic syntax of creating a radial gradient can be given with: radial-gradient(shape size at position, color-stop1, color-stop2, ...);

- The arguments of the radial-gradient() function has the following meaning:
- **position**: Specifies the starting point of the gradient, which can be specified in units (px, em, or percentages) or keyword (left, bottom, etc).
- **shape**: Specifies the shape of the gradient's ending shape. It can be circle or ellipse.
- **size**: Specifies the size of the gradient's ending shape. The default is farthest-side.

DROP SHADOW

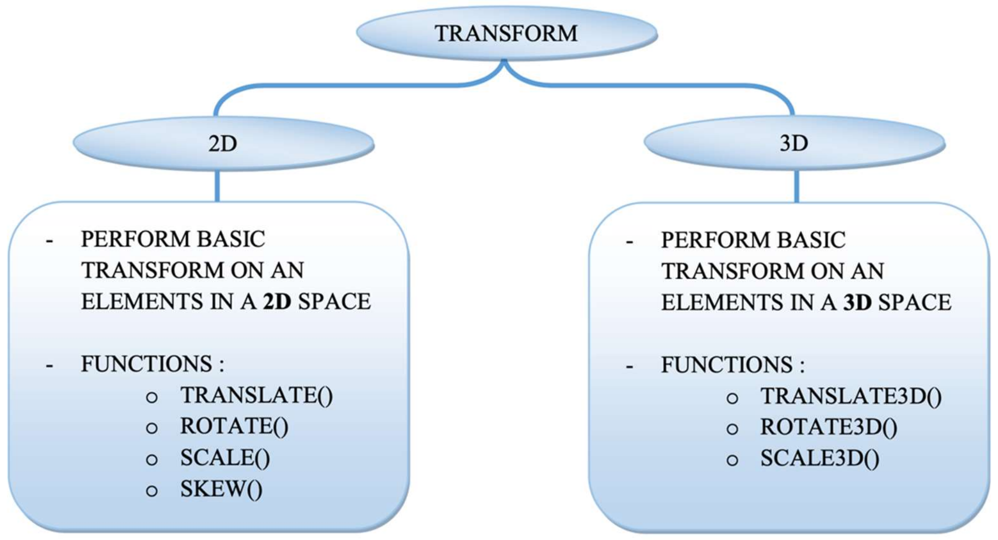

DROP SHADOW
- USED TO GIVE DROP SHADOW EFFETCS TO THE ELEMENT
- TEXT SHADOW PROPERTY IS USED TO GIVE SHADOW EFFECT TO THE TEXT
- VALUES
o offset-x
o offset-y
o blur-radius
o color

- The CSS3 gives you ability to add drop shadow effects to the elements like you do in Photoshop without using any images.
- Prior to CSS3, sliced images are used for creating the shadows around the elements that was quite annoying.

CSS3 box-shadow Property

- The box-shadow property can be used to add shadow to the element's boxes.
- You can even apply more than one shadow effects using a comma-separated list of shadows.
- The basic syntax of creating a box shadow can be given with:
  - box-shadow: offset-x offset-y blur-radius color;
- The components of the box-shadow property have the following meaning:
  - **offset-x** : Sets the horizontal offset of the shadow.
  - **offset-y** : Sets the vertical offset of the shadow.
  - **blur-radius** : Sets the blur radius. The larger the value, the bigger the blur and more the shadow's edge will be blurred. Negative values are not allowed.
  - **color** : Sets the color of the shadow. If the color value is omitted or not specified, it takes the value of the color property.

2D & 3D TRANSFORM PROPERTY


TRANSFORM
2D
3D
- PERFORM BASIC TRANSFORM ON AN ELEMENTS IN A 2D SPACE
- FUNCTIONS :
TRANSLATE()
ROTATE()
SCALE()
SKEW()
- PERFORM BASIC TRANSFORM ON AN ELEMENTS IN A 3D SPACE
- FUNCTIONS :
TRANSLATE3D()
ROTATE3D()
SCALE3D()

2D Transformation of Elements

- With CSS3 2D transform feature you can perform basic transform manipulations such as move, rotate, scale and skew on elements in a two-dimensional space.
- A transformed element doesn't affect the surrounding elements, but can overlap them, just like the absolutely positioned elements.
- Using CSS Transform and Transform Functions

The translate() Function

- Moves the element from its current position to a new position along the X and Y axes. This can be written as translate(tx, ty). If ty isn't specified, its value is assumed to be zero.
- **Example**: transform: translate(200px, 50px);

The rotate() Function

- The rotate() function rotates the element around its origin (as specified by the transform-origin property) by the specified angle. This can be written as rotate(a).
- **Example:** transform: rotate(30deg);

The scale() Function

- The scale() function increases or decreases the size of the element. It can be written as scale(sx, sy). If sy isn't specified, it is assumed to be equal to sx.
- **Example**: transform: scale(1.5);
- The functionscale(1.5)proportionally scale the width and height of the image 1.5 times to its original size. The function scale(1) or scale(1, 1) has no effect on the element.

The skew()Function

- The skew() function skews the element along the X and Y axes by the specified angles. It can be written as skew(ax, ay). If ay isn't specified, its value is assumed to be zero.
  Example: transform: skew(-40deg, 15deg);

translate(*x*,*y*)	Defines a 2D translation, moving the element along the X- and the Y-axis
translateX(*n*)	Defines a 2D translation, moving the element along the X-axis
translateY(*n*)	Defines a 2D translation, moving the element along the Y-axis
scale(*x*,*y*)	Defines a 2D scale transformation, changing the elements width and height

scaleX(*n*)	Defines a 2D scale transformation, changing the element's width
scaleY(*n*)	Defines a 2D scale transformation, changing the element's height
rotate(*angle*)	Defines a 2D rotation, the angle is specified in the parameter
skew(*x-angle,y-angle*)	Defines a 2D skew transformation along the X- and the Y-axis
skewX(*angle*)	Defines a 2D skew transformation along the X-axis
skewY(*angle*)	Defines a 2D skew transformation along the Y-axis

3D Transformation of Elements

- With CSS3 3D transform feature you can perform basic transform manipulations such as move, rotate, scale and skew on elements in a three-dimensional space.
- A transformed element doesn't affect the surrounding elements, but can overlap them, just like the absolutely positioned elements. However, the transformed element still takes space in the layout at its default (un-transformed) location.
- Using CSS Transform and Transform Functions
- The CSS3 transform property uses the transform functions to manipulate the coordinate system used by an element in order to apply the transformation effect.

The translate3d() Function

- Moves the element from its current position to a new position along the X, Y and Z-axis. This can be written as translate(tx, ty, tz). Percentage values are not allowed for third translation-value parameter (i.e. tz).
- **Example**: transform: translate3d(25px, 25px, 50px);
- The function translate3d(25px, 25px, 50px) moves the image 25 pixels along the positive X and Y-axis, and the 50 pixels along the positive Z-axis.

The rotate3d() Function

- The rotate3d() function rotates the element in 3D space by the specified angle around the [x,y,z] direction vector. This can be written as rotate(vx, vy, vz, angle).
- **Example**: transform: rotate3d(0, 1, 0, 60deg);
- The function rotate3d(0, 1, 0, 60deg) rotates the image along the Y-axis by the angle 60 degrees. You can use negative values to rotate the element in opposite direction.

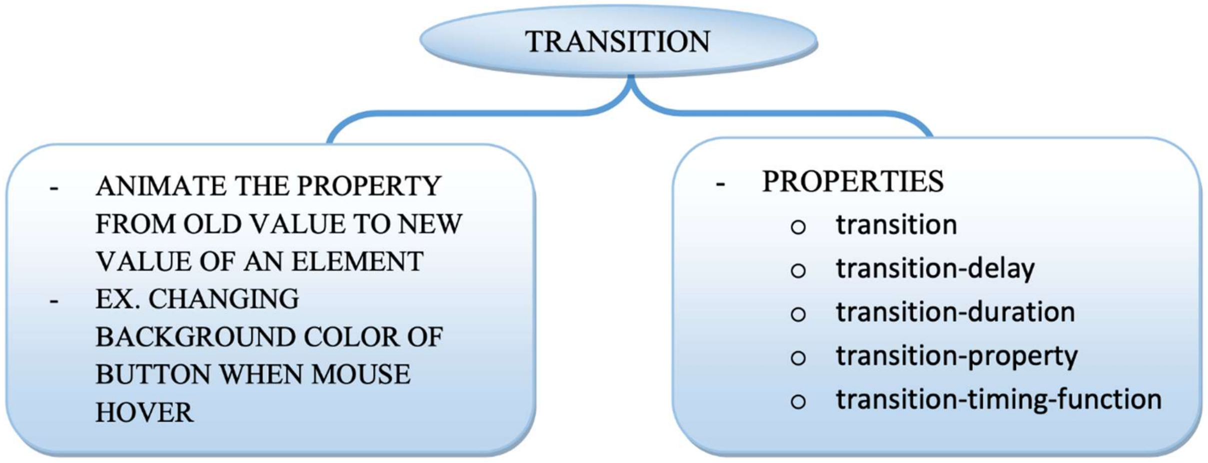
The scale3d() Function

- The scale3d() function changes the size of an element. It can be written as scale(sx, sy, sz). The effect of this function is not evident unless you use it in combination with other transform functions such as rotate and the perspective, as shown in the example below.
- **Example**: transform: scale3d(1, 1, 2); rotate3d(1, 0, 0, 60deg);
- The function scale3d(1, 1, 2) scales the elements along the Z-axis and the function rotate3d(1, 0, 0, 60deg) rotates the image along the X-axis by the angle 60 degrees.

translate3d(*x,y,z*)	Defines a 3D translation
translateX(*x*)	Defines a 3D translation, using only the value for the X-axis
translateY(*y*)	Defines a 3D translation, using only the value for the Y-axis
translateZ(*z*)	Defines a 3D translation, using only the value for the Z-axis
scale3d(*x,y,z*)	Defines a 3D scale transformation
scaleX(*x*)	Defines a 3D scale transformation by giving a value for the X-axis
scaleY(*y*)	Defines a 3D scale transformation by giving a value for the Y-axis
scaleZ(*z*)	Defines a 3D scale transformation by giving a value for the Z-axis
rotate3d(*x,y,z,angle*)	Defines a 3D rotation
rotateX(*angle*)	Defines a 3D rotation along the X-axis
rotateY(*angle*)	Defines a 3D rotation along the Y-axis
rotateZ(*angle*)	Defines a 3D rotation along the Z-axis

perspective(*n*)	Defines a perspective view for a 3D transformed element

CSS TRANSITION PROPERTY


TRANSITION
- ANIMATE THE PROPERTY FROM OLD VALUE TO NEW VALUE OF AN ELEMENT
- EX. CHANGING BACKGROUND COLOR OF BUTTON WHEN MOUSE HOVER
- PROPERTIES
o transition
o transition-delay
o transition-duration
o transition-property
o transition-timing-function

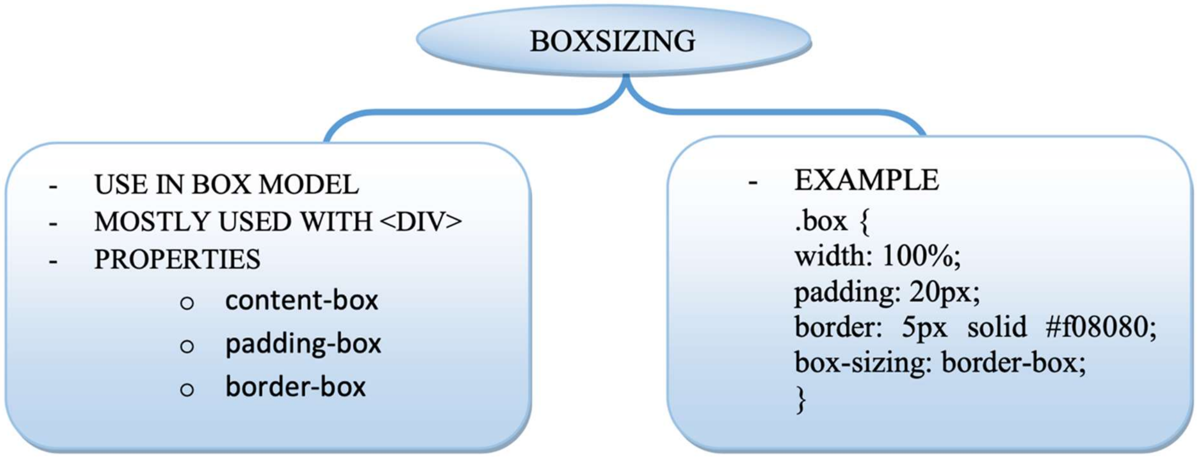
- Normally when the value of a CSS property changes, the rendered result is instantly updated. A common example is changing the background color of a button on mouse hover.
- In a normal scenario the background color of the button is changes immediately from the old property value to the new property value when you place the cursor over the button.
- CSS3 introduces a new transition feature that allows you to animate a property from the old value to the new value smoothly over time.
- The following example will show you how to animate the background-color of an HTML button on mouse hover.

Performing Multiple Transitions

- Each of the transition properties can take more than one value, separated by commas, which provides an easy way to define multiple transitions at once with different settings.

Property	**Description**
transition	A shorthand property for setting all the four individual transition properties in a single declaration.
transition-delay	Specifies when the transition will start.
transition-duration	Specifies the number of seconds or milliseconds a transition animation should take to complete.
transition-property	Specifies the names of the CSS properties to which a transition effect should be applied.
transition-timing-function	Specifies how the intermediate values of the CSS properties being affected by a transition will be calculated.

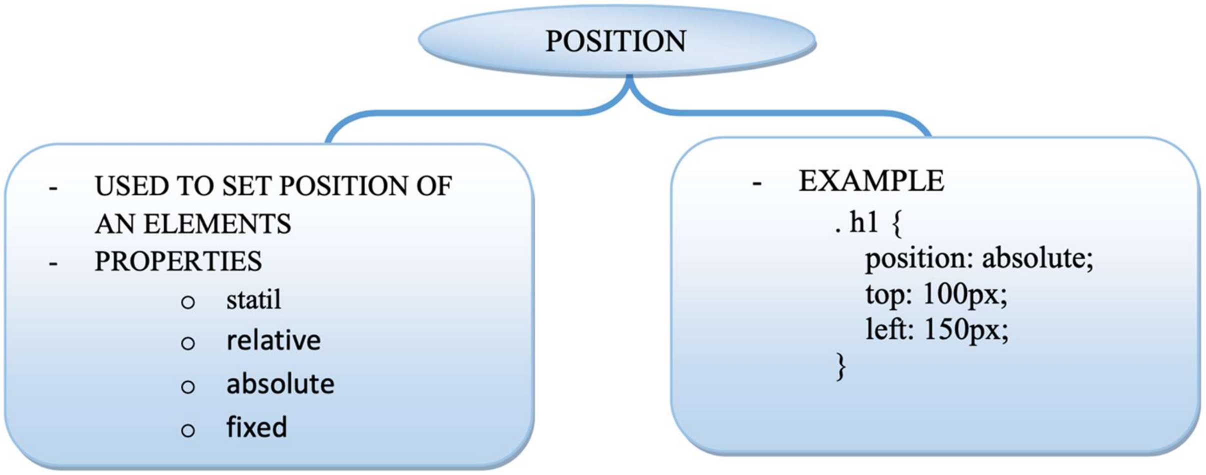
CSS 3 BOXSIZING



- The box-sizing CSS property is used to alter the default CSS box model, which is normally used by the browser to calculate the widths and heights of the elements.
- Syntax: box-sizing: content-box | padding-box | border-box | initial | inherit

Value	**Description**
content-box	The specified width and height properties include only the content of the element. It does not include the padding, border or margin.
padding-box	The specified width and height properties include the padding size, and do not include the border or margin.
border-box	The specified width and height properties include the padding and border, but not the margin.

CSS 3 POSITION PROPERTY



- The position CSS property specifies how an element is positioned.
- Tip: Elements with a position other than static are said to be positioned. Their vertical placement in the stacking context is determined by the z-index property.
- **Syntax**: position: static | relative | absolute | fixed | initial | inherit

Value	Description
static	The element's box is a normal box, laid out according to the normal flow. The top, right, bottom, left, and z-index properties are ignored for static boxes. This is default value.

relative	The element is positioned relative to its normal position (this is called the position in normal flow).
absolute	The element is positioned relative to its first ancestor element whose position value is other than static.
fixed	The element is fixed with respect to the screen's viewport and doesn't move when scrolled. When printing, the element is printed on every page.

CSS 3 MEDIA QUERIES

- Media queries allow you to customize the presentation of your web pages for a specific range of devices like mobile phones, tablets, desktops, etc. without any change in markups.
- Media queries can be used to check many things, such as:
  - width and height of the viewport
  - width and height of the device
  - orientation (is the tablet/phone in landscape or portrait mode?)
  - resolution
- Using media queries are a popular technique for delivering a tailored style sheet to tablets, iPhone, and Androids.
- A media query consists of a media type and zero or more expressions that match the type and conditions of a particular media features such as device width or screen resolution.
- Since media query is a logical expression it can be resolve to either true or false.
- The result of the query will be true if the media type specified in the media query matches the type of device the document is being displayed on, as well as all expressions in the media query are satisfied.
- When a media query is true, the related style sheet or style rules are applied to the target device. Here's a simple example of the media query for standard devices.
- **Syntax**
  - A media query consists of a media type and can contain one or more expressions, which resolve to either true or false.
  - @media not|only mediatype and (expressions) { CSS-Code;
  - }
- **Tip**: Media queries are an excellent way to create responsive layouts. Using media queries you can customize your website differently for users browsing on devices like smart phones or tablets without changing the actual content of the page.

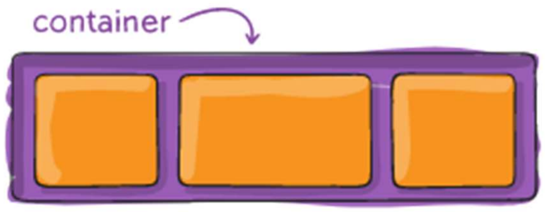
CSS Flexbox Properties

The Flexbox Layout (Flexible Box) module (a W3C Candidate Recommendation as of October 2017) aims at providing a more efficient way to lay out, align and distribute space among items in a container, even when their size is unknown and/or dynamic (thus the word "flex").

The main idea behind the flex layout is to give the container the ability to alter its items' width/height (and order) to best fill the available space (mostly to accommodate to all kind of display devices and screen sizes). A flex container expands items to fill available free space or shrinks them to prevent overflow.

Most importantly, the flexbox layout is direction-agnostic as opposed to the regular layouts (block which is vertically-based and inline which is horizontally-based). While those work well for pages, they lack flexibility (no pun intended) to support large or complex applications (especially when it comes to orientation changing, resizing, stretching, shrinking, etc.).

Properties for the Parent (flex container)



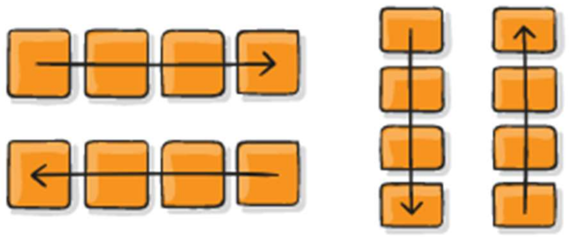
display

This defines a flex container; inline or block depending on the given value. It enables a flex context for all its direct children.

```
.container {
  display: flex; /* or inline-flex */
}
```

Note that CSS columns have no effect on a flex container.

flex-direction

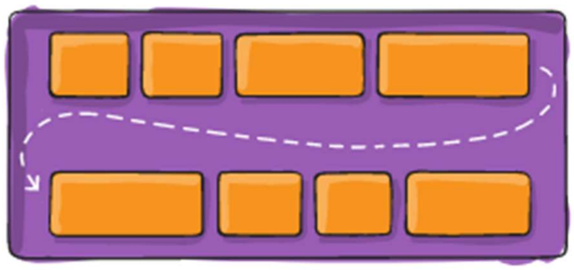


This establishes the main-axis, thus defining the direction flex items are placed in the flex container. Flexbox is (aside from optional wrapping) a single-direction layout concept. Think of flex items as primarily laying out either in horizontal rows or vertical columns.

```
.container {
  flex-direction: row | row-reverse | column | column-reverse;
}
```

row (default): left to right in ltr; right to left in rtl
row-reverse: right to left in ltr; left to right in rtl
column: same as row but top to bottom
column-reverse: same as row-reverse but bottom to top

flex-wrap



By default, flex items will all try to fit onto one line. You can change that and allow the items to wrap as needed with this property.

```
.container {
  flex-wrap: nowrap | wrap | wrap-reverse;
}
```

nowrap (default): all flex items will be on one line
wrap: flex items will wrap onto multiple lines, from top to bottom.
wrap-reverse: flex items will wrap onto multiple lines from bottom to top.

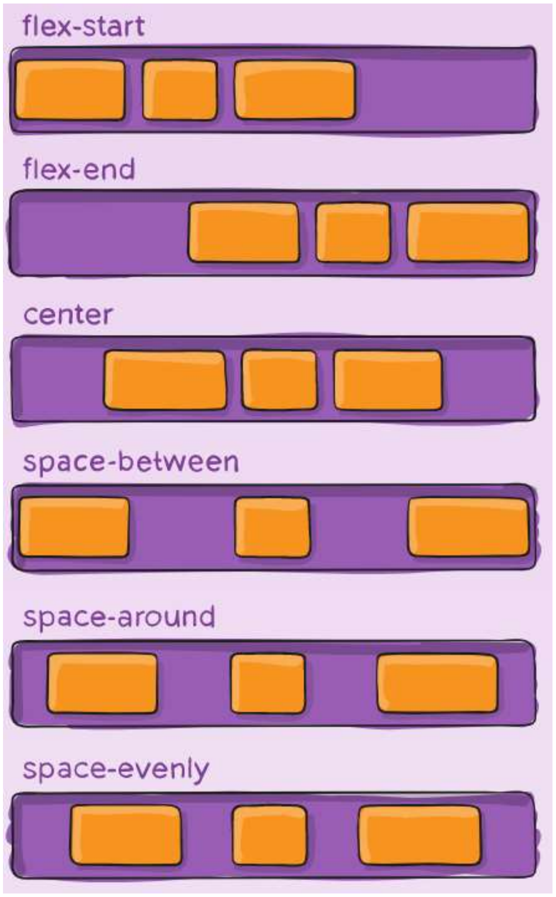
flex-flow

This is a shorthand for **the flex-direction and flex-wrap** properties, which together define the flex container's main and cross axes. The default value is row nowrap.

```
.container {
  flex-flow: column wrap;
}
```

justify-content

This defines the alignment along the main axis. It helps distribute extra free space leftover when either all the flex items on a line are inflexible, or are flexible but have reached their maximum size. It also exerts some control over the alignment of items when they overflow the line

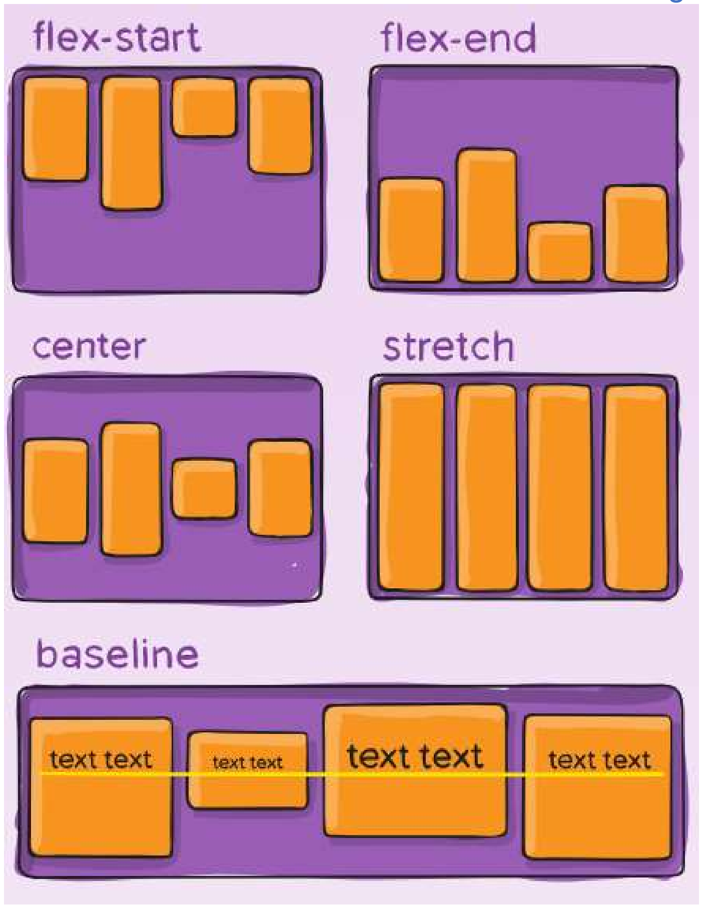


```
.container {
  justify-content: flex-start | flex-end | center | space-between | space-around | space-evenly | start | end | left | right ... + safe | unsafe;
}
```

flex-start (default): items are packed toward the start of the flex-direction.
flex-end: items are packed toward the end of the flex-direction.
start: items are packed toward the start of the writing-mode direction.
end: items are packed toward the end of the writing-mode direction.
left: items are packed toward left edge of the container, unless that doesn't make sense with the flex-direction, then it behaves like start.
right: items are packed toward right edge of the container, unless that doesn't make sense with the flex-direction, then it behaves like start.
center: items are centered along the line
space-between: items are evenly distributed in the line; first item is on the start line, last item on the end line
space-around: items are evenly distributed in the line with equal space around them. Note that visually the spaces aren't equal, since all the items have equal space on both sides. The first item will have one unit of space against the container edge, but two units of space between the next item because that next item has its own spacing that applies.
space-evenly: items are distributed so that the spacing between any two items (and the space to the edges) is equal.

align-items

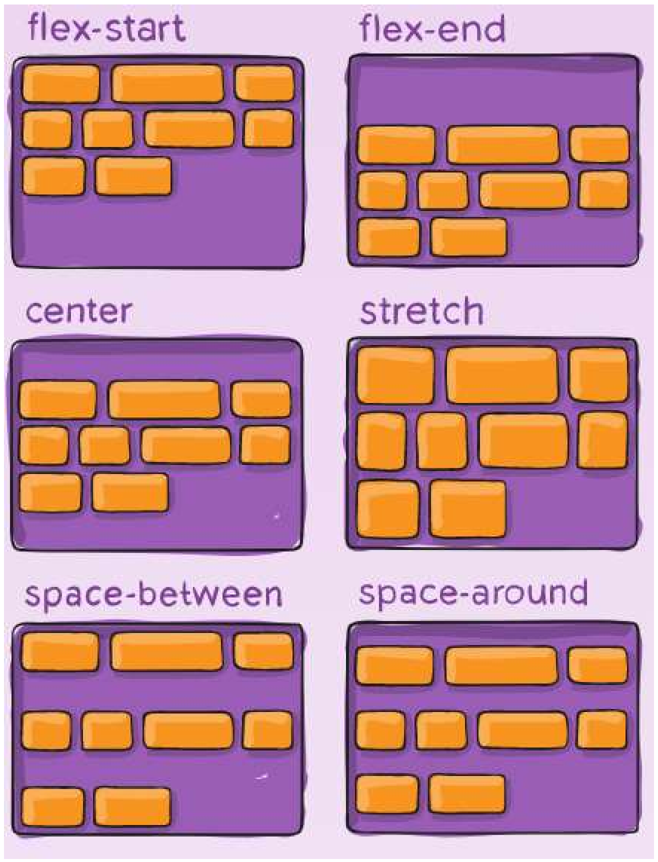
This defines the default behavior for how flex items are laid out along the cross axis on the current line. Think of it as the justify-content version for the cross-axis (perpendicular to the main-axis).



```
.container {
  align-items: stretch | flex-start | flex-end | center | baseline | first baseline | last baseline | start | end | self-start | self-end;
}
```

stretch (default): stretch to fill the container (still respect min-width/max-width)
flex-start / start / self-start: items are placed at the start of the cross axis. The difference between these is subtle, and is about respecting the flex-direction rules or the writing-mode rules.
flex-end / end / self-end: items are placed at the end of the cross axis. The difference again is subtle and is about respecting flex-direction rules vs. writing-mode rules.
center: items are centered in the cross-axis
baseline: items are aligned such as their baselines align

align-content
This aligns a flex container's lines within when there is extra space in the cross-axis, similar to how justify-content aligns individual items within the main-axis.



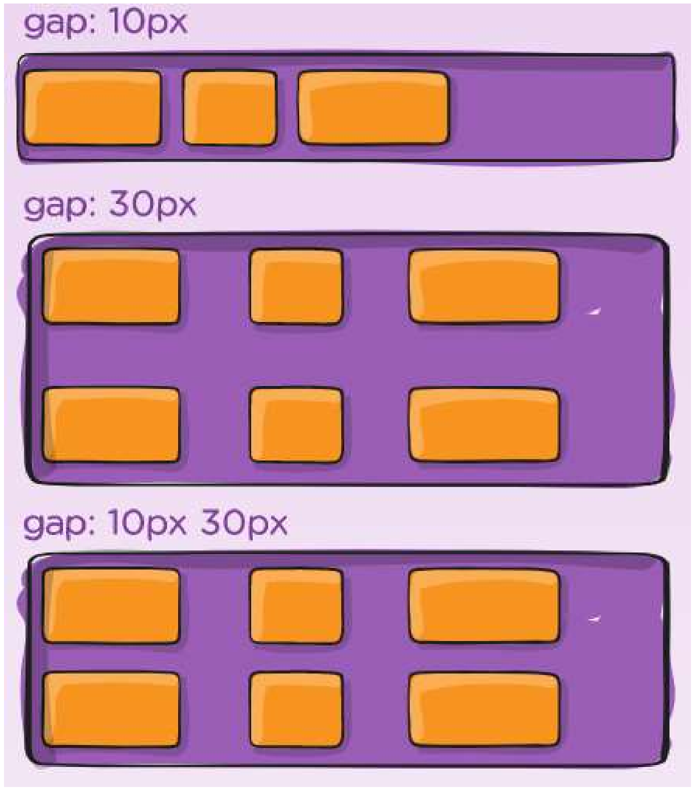
Note: This property only takes effect on multi-line flexible containers, where flex-wrap is set to either wrap or wrap-reverse). A single-line flexible container (i.e. where flex-wrap is set to its default value, no-wrap) will not reflect align-content.

```
.container {
  align-content: flex-start | flex-end | center | space-between | space-around | space-evenly | stretch | start | end | baseline | first baseline | last baseline;
}
```

normal (default): items are packed in their default position as if no value was set.
flex-start / start: items packed to the start of the container. The (more supported) flex-start honors the flex-direction while start honors the writing-mode direction.
flex-end / end: items packed to the end of the container. The (more support) flex-end honors the flex-direction while end honors the writing-mode direction.
center: items centered in the container
space-between: items evenly distributed; the first line is at the start of the container while the last one is at the end
space-around: items evenly distributed with equal space around each line
space-evenly: items are evenly distributed with equal space around them
stretch: lines stretch to take up the remaining space

gap, row-gap, column-gap

The gap property explicitly controls the space between flex items. It applies that spacing only between items not on the outer edges.



```
.container {
  display: flex;
  ...
  gap: 10px;
  gap: 10px 20px; /* row-gap column gap */
  row-gap: 10px;
  column-gap: 20px;
}
```

The behavior could be thought of as a minimum gutter, as if the gutter is bigger somehow (because of something like justify-content: space-between;) then the gap will only take effect if that space would end up smaller.

It is not exclusively for flexbox, gap works in grid and multi-column layout as well.

Overflow

The overflow property specifies what should happen if content overflows an element's box.This property specifies whether to clip content or to add scrollbars when an element's content is too big to fit in a specified area.
Note: The overflow property only works for block elements with a specified height.

```
div.ex1 {
  overflow: visible|hidden|clip|scroll|auto|initial|inherit;
}
```

Property Values
Visible:The overflow is not clipped. It renders outside the element's box. This is default
Hidden:The overflow is clipped, and the rest of the content will be invisible. Content can be scrolled programmatically (e.g. by setting scrollLeft or scrollTo())
Clip:The overflow is clipped, and the rest of the content will be invisible. Forbids scrolling, including programmatic scrolling.
Scroll:The overflow is clipped, but a scroll-bar is added to see the rest of the content
Auto:If overflow is clipped, a scroll-bar should be added to see the rest of the content
Initial:Sets this property to its default value. Read about initial

text-overflow

The text-overflow property specifies how overflowed content that is not displayed should be signaled to the user. It can be clipped, display an ellipsis (...), or display a custom string.
Both of the following properties are required for text-overflow:
white-space: nowrap;
overflow: hidden;

Cursor

The cursor property in CSS controls what the mouse cursor will look like when it is located over the element in which this property is set. Obviously, it's only relevant in browsers/operating systems in which there is a mouse and cursor. They are used essentially for UX to convey the idea of certain functionality. So try not to break that affordance.

The cursor property of CSS allows you to specify the type of cursor that should be displayed to the user.
One good usage of this property is in using images for submit buttons on forms. By default, when a cursor hovers over a link, the cursor changes from a pointer to a hand. However, it does not change form for a submit button on a form. Therefore, whenever someone hovers over an image that is a submit button, it provides a visual clue that the image is clickable.

Values	Usage
alias	It is used to display the indication of the cursor of something that is to be created.
auto	It is the default property in which the browser sets the cursor.
all-scroll	It indicates the scrolling.
col-resize	Using it, the cursor will represent that the column can be horizontally resized.
cell	The cursor will represent that a cell or the collection of cells is selected.
context-menu	It indicates the availability of the context menu.
default	It indicates an arrow, which is the default cursor.
copy	It is used to indicate that something is copied.
crosshair	In it, the cursor changes to the crosshair or the plus sign.
e-resize	It represents the east direction and indicates that the edge of the box is to be shifted towards right.
ew-resize	It represents the east/west direction and indicates a bidirectional resize cursor.
n-resize	It represents the north direction that indicates that the edge of the box is to be shifted to up.
ne-resize	It represents the north/east direction and indicates that the edge of the box is to be shifted towards up and right.
move	It indicates that something is to be shifted.
help	It is in the form of a question mark or ballon, which represents that help is available.
None	It is used to indicate that no cursor is rendered for the element.
No-drop	It is used to represent that the dragged item cannot be dropped here.
s-resize	It indicates an edge box is to be moved down. It indicates the south direction.
Row-resize	It is used to indicate that the row can be vertically resized.
Se-resize	It represents the south/east direction, which indicates that an edge box is to be moved down and right.

Sw-resize	It represents south/west direction and indicates that an edge of the box is to be shifted towards down and left.
Wait	It represents an hourglass.
<url>	It indicates the source of the cursor image file.
w-resize	It indicates the west direction and represents that the edge of the box is to be shifted left.
Zoom-in	It is used to indicate that something can be zoomed in.
Zoom-out	It is used to indicate that something can be zoomed out.

Syntax:
cursor:<url>|auto|default|none|context-menu| help| pointer| progress| wait|cell |crosshair| text|vertical-text|alias|copy|move|no-drop|not-allowed|grab|grabbing|e-resize|n-resize|ne-resize|nw-resize|s-resize|se-resize|sw-resize|w-resize|ew-resize|ns-resize|nesw-resize|nwse-resize|col-resize|row-resize|all-scroll|zoom-in|zoom-out

Visiblity

The CSS visibility property is used to specify whether an element is visible or not.
Note: An invisible element also take up the space on the page. By using display property you can create invisible elements that don't take up space.

Value	**Description**
`visible`	Default value. The box and its contents are visible.
`hidden`	The box and its content are invisible, but still affect the layout of the page.
`collapse`	This value causes the entire row or column to be removed from the display. This value is used for row, row group, column, and column group elements.
`inherit`	Specifies that the value of the visibility property should be inherited from the parent element i.e. takes the same visibility value as specified for its parent.

filter

CSS Filters are a powerful tool that authors can use to achieve varying visual effects (sort of like Photoshop filters for the browser). The CSS filter property provides access to effects like blur or color shifting on an element's rendering before the element is displayed. Filters are commonly used to adjust the rendering of an image, a background, or a border.

Syntax: filter: none | invert() | drop-shadow() | brightness() | saturate() | blur() | hue-rotate() | contrast() | opacity() | grayscale() | sepia() | url();

Example:
```
/* <filter-function> values */
filter: blur(5px);
filter: brightness(0.4);
filter: contrast(200%);
filter: drop-shadow(16px 16px 20px blue);
filter: grayscale(50%);
filter: hue-rotate(90deg);
filter: invert(75%);
filter: opacity(25%);
filter: saturate(30%);
```

```
filter: sepia(60%);
/* Multiple filters */
filter: contrast(175%) brightness(3%);
/* Use no filter */
filter: none;
```

Backdrop-filter:

The backdrop-filter property in CSS is used to apply filter effects (grayscale, contrast, blur, etc) to the background/backdrop of an element. The backdrop-filter has the same effect as the filter property, except the filter effects are applied only to the background and instead of to the element's content.

Syntax: backdrop-filter: blur() | brightness() | contrast() | drop-shadow() | grayscale() | hue-rotate() | invert() | opacity() | saturate() | sepia() | none | initial | inherit

Example:

```
<style>
                .container {
                        background-image:abc-25.png;
                        background-size: cover;
                        display: flex;
                        align-items: center;
                        justify-content: center;
                        height: 100px;
                        width: 360px;
                }
                .foreground {
                        backdrop-filter: brightness(25%);
                        padding: 2px;
                }
        </style>

<body>
        <h1 style="color: green">
                KSC
        </h1>

        <b>CSS | backdrop-filter</b>

        <div class="container">
                <div class="foreground">
                        This text is not affected
                        by backdrop-filter.
                </div>
        </div>
</body>
</html>
```

object-fit

The object-fit property defines how an element responds to the height and width of its content box. It's intended for images, videos and other embeddable media formats in conjunction with the object-position property. Used by itself, object-fit lets us crop an inline image by giving us fine-grained control over how it squishes and stretches inside its box.

Syntax:
object-fit: fill|contain|cover|scale-down|none|initial|inherit;

fill: this is the default value which stretches the image to fit the content box, regardless of its aspect-ratio.
contain: increases or decreases the size of the image to fill the box whilst preserving its aspect-ratio.
cover: the image will fill the height and width of its box, once again maintaining its aspect ratio but often cropping the image in the process.
none: image will ignore the height and width of the parent and retain its original size.
scale-down: the image will compare the difference between none and contain in order to find the smallest concrete object size.

@font-face

Web fonts allow Web designers to use fonts that are not installed on the user's computer.
When you have found/bought the font you wish to use, just include the font file on your web server, and it will be automatically downloaded to the user when needed.

Different Font Formats

TrueType Fonts (TTF)
TrueType is a font standard developed in the late 1980s, by Apple and Microsoft. TrueType is the most common font format for both the Mac OS and Microsoft Windows operating systems.

OpenType Fonts (OTF)
OpenType is a format for scalable computer fonts. It was built on TrueType, and is a registered trademark of Microsoft. OpenType fonts are used commonly today on the major computer platforms.

The Web Open Font Format (WOFF)
WOFF is a font format for use in web pages. It was developed in 2009, and is now a W3C Recommendation. WOFF is essentially OpenType or TrueType with compression and additional metadata. The goal is to support font distribution from a server to a client over a network with bandwidth constraints.

The Web Open Font Format (WOFF 2.0)
TrueType/OpenType font that provides better compression than WOFF 1.0.

SVG Fonts/Shapes
SVG fonts allow SVG to be used as glyphs when displaying text. The SVG 1.1 specification define a font module that allows the creation of fonts within an SVG document. You can also apply CSS to SVG documents, and the @font-face rule can be applied to text in SVG documents.

Embedded OpenType Fonts (EOT)
EOT fonts are a compact form of OpenType fonts designed by Microsoft for use as embedded fonts on web pages.

Example:
```
@font-face {
  font-family: myFirstFont;
  src: url(sansation_light.woff);
}
div {
  font-family: myFirstFont;
}
```

BEM nameing convention

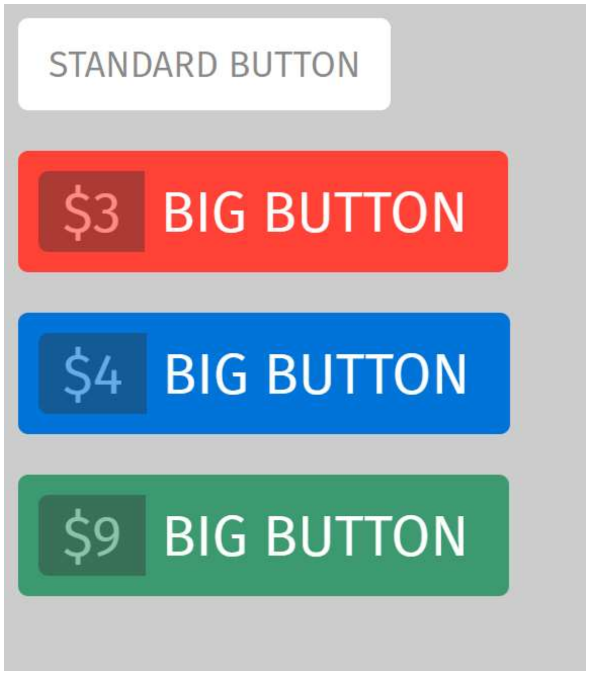
The Block, Element, Modifier methodology (commonly referred to as BEM) is a popular naming convention for classes in HTML and CSS. Developed by the team at Yandex, its goal is to help developers better understand the relationship between the HTML and CSS in a given project.

Example:
```
/* Block component */
.btn {}
/* Element that depends upon the block */
.btn__price {}
/* Modifier that changes the style of the block */
.btn--orange {}
.btn--big {}

<a class="btn btn--big btn--orange" href="https://css-tricks.com">
  <span class="btn__price">$9.99</span>
  <span class="btn__text">Subscribe</span>
</a>
```

Explanation:
In this CSS methodology a block is a top-level abstraction of a new component, for example a button: .btn { }. This block should be thought of as a parent. Child items, or elements, can be placed inside and these are denoted by two underscores following the name of the block like .btn__price { }. Finally, modifiers can manipulate the block so that we can theme or style that particular component without inflicting changes on a completely unrelated module. This is done by appending two hyphens to the name of the block just like btn--orange.

Complete CSS:
```
/* Block */
.btn {
  text-decoration: none;
  background-color: white;
  color: #888;
  border-radius: 5px;
  display: inline-block;
  margin: 10px;
  font-size: 18px;
  text-transform: uppercase;
  font-weight: 600;
  padding: 10px 5px;
}

/* Element */
.btn__price {
  background-color: white;
  color: #fff;
  padding-right: 12px;
  padding-left: 12px;
  margin-right: -10px; /* realign button text padding */
  font-weight: 600;
  background-color: #333;
  opacity: .4;
  border-radius: 5px 0 0 5px;
}

/* Element */
.btn__text {
  padding: 0 10px;
  border-radius: 0 5px 5px 0;
}

/* Modifier */
.btn--big {
  font-size: 28px;
  padding: 10px;
  font-weight: 400;
}

/* Modifier */
.btn--blue {
  border-color: #0074D9;
  color: white;
  background-color: #0074D9;
}

/* Modifier */
.btn--orange {
  border-color: #FF4136;
  color: white;
  background-color: #FF4136;
}

/* Modifier */
.btn--green {
  border-color: #3D9970;
  color: white;
  background-color: #3D9970;
}
```

```
body {
  font-family: "fira-sans-2", sans-serif;
  background-color: #ccc;
}
```



Why should we consider BEM?

1. If we want to make a new style of a component, we can easily see which modifiers and children already exist. We might even realize we don't need to write any CSS in the first place because there is a pre-existing modifier that does what we need.
2. If we are reading the markup instead of CSS, we should be able to quickly get an idea of which element depends on another (in the previous example we can see that .btn__price depends on .btn, even if we don't know what that does just yet.)
3. Designers and developers can consistently name components for easier communication between team members. In other words, BEM gives everyone on a project a declarative syntax that they can share so that they're on the same page.

Naming rules:

- Names are written in lowercase letters.
- Words are separated by a hyphen (-).
- The **block** name defines the namespace for its elements and modifiers.
- The **element** name is separated from the block name by a double underscore (__).
- The **modifier** name is separated from the block or element name by a single underscore (_).
- The modifier value is separated from the modifier name by a single underscore (_).