2. CSS cr.Prasanna Acharya

CSS stands for Cascading Style Sheets. CSS comes in file extension .css like HTML which comes in .html file extension. HTML is the backbone of every website but it creates an ugly website. CSS lets us create a beautiful and modern looking website. CSS does not control the functionality of the site. For functionality, JavaScript is required and it should be learnt after CSS. CSS is a layer on the top of HTML. We need to write HTML to make a website, but for modern looking website, CSS is a must. For rendering content, browser mainly render HTML, CSS and JavaScript.

CSS can be written in 3 different ways:

1. **Inline**(with style HTML attribute)
2. **Internal**(with the <style> element)
3. **External(**as .css file)

Inline CSS

Style is used as an attribute alongside any element that needs to be stylized.

Example:

```
<h1 style="font-size: 20px; color: crimson">This is h1</h1>
```

Internal CSS

Syntax:

<style>

selector { property: value }

</style>

Example:

```
<style>
        body {
            background-color: cornflowerblue;
        }
    </style>
```

Note: In the syntax, selector is Who?(i.e. who is that you want to change)

property is What?(i.e. what is that you want to change)
value is How?(i.e. how are gonna change)

External CSS

A separate .css file is created where all the properties are listed using the same CSS syntax and this file can be linked to any HTML file codes by using link+emmet abbreviation.

Types of Selector

- Element Selector

  Example(using internal css):

```
<style>
        body {
            background-color: cornflowerblue;
        }
</style>
```

  Note: Here, body is the element and it is used as element selector in internal css for stylizing the body element.

- Class Selector

  Example(using internal css):

```
<style>
        .uppercase {
            text-transform: uppercase;
        }
</style>
```

  In the body, we have:

```
<p class="uppercase">
        This is p.
    </p>
```

It is based on selecting HTML attribute, i.e. class.

- Id Selector

  Example(using internal css):

```
<style>
        #title {
            text-decoration: underline;
        }
```

```
</style>
```

In the body, we have:

```
<p id="title">
        This is p.
</p>
```

It is based on selecting HTML attribute, i.e. id.

Note:
Id must be unique. There must be id of only one type or name. Use class when you want to stylize more than one element and use id when you want to stylize only a specific element.

- Grouped selector

Example(using internal css):

```
<style>
    h2, .a-paragraph, div {
                color: darkblue;
    }
</style>
```

In the body, we have:

```
<div>
        This is div.
</div>
<h2>
        This is h2.
</h2>
<p class="a-paragraph">
        Lorem ipsum dolor <strong>here <span>hey</span> it is</strong> sit amet
consectetur, adipisicing elit. Illo, sequi?
 </p>
```

- Nesting selector

Example(using internal css):

```
<style>
p strong span {
            text-decoration: underline;
        }
</style>
```

In the body, we have:

```
<p class="a-paragraph">
        Lorem ipsum dolor <strong>here <span>hey</span> it is</strong> sit amet
consectetur, adipisicing elit. Illo, sequi?
    </p>
```

Here, hey within the span tag gets underlined due to nesting selector.

Commenting

For commenting, /*.....*/ is used(same as in C programming language that you learnt earlier)

Note: Ctrl + / is the shortcut key for commenting a particular line in VS code.

Color Specification

Colors can be specified using the following:

- Name:

  Example: red, blue, black, powderblue, cyan, magenta, grey, etc.
- Hexadecimal code:

  Example: #FF0000 (for red), #000000 (for black), #FFFFFF (for white), etc.
- rgb/rgba:

  Here, syntax for using color in rgb is:

  rgb(value1, value2, value3)

  where the three values are the r(red), g(green) and b(blue) values of the color. Each value ranges from 0 to 255.

  rgba can also be used whose syntax is:

  rgba(value1, value2, value3, value4)

  where, the fourth value is for alpha. alpha is the fourth parameter of rgba which specifies the opacity(visibility) of the color.

Note: For using gradient colors in CSS, use online CSS gradient generator.

Or, you can use the general syntax:

background:linear-gradient(direction, color1 starting_point_in_percentage, color2 ending_point_in_percentage);

Note: For layering gradient on the image, we can use the following syntax:

background:linear-gradient(direction, color1 starting_point_in_percentage, color2 ending_point_in_percentage), url("link");

Pseudo-classes

Pseudo-class is a keyword to a selector that specifies special state of selected elements.

Syntax:

selector:pseudo-class {
property: value;
}

Some of them are :hover, :link, :target, :active, :visited, focus, etc.

Example:

```
img:hover {
    background-color: gold;
}
```

Pseudo-elements

Pseudo-element is a keyword to a selector that let you stylize a specific part of the selected elemnt(s).

Syntax:

selector::pseudo-element {
property: value;
}

Some of them are ::after, ::before, ::first-letter, ::first-line, ::selection, etc.

In HTML,

```
<span>Some quotes</span>
```

In CSS,

```
span::before {
  content: "«";
  color: blue;
}

span::after {
  content: "»";
  color: red;
}
```

Result:

<<Some quotes>>

Note: In contrast to pseudo-elements, pseudo-classes can be used to style an element based on its *state*.

Measurements of font-size

- Absolute
  - cm - centimeters
  - mm - millimeters
  - in - inches(1in = 96px = 2.54cm)
  - **px** - pixels(1px = 1/96 of 1 in)
  - pt - points(1pt = 1/72 of 1 in)
  - pc - picas(1 pc = 12 pt)
- Relative
  - **rem** - relative to the font-size of the root element
  - **em** - relative to the font-size of the element
  - ex - relative to the x-height of the current font
  - ch - relative to the width of the "0" (zero)
  - **vw** - relative to 1% of the width of the viewport
  - **vh** - relative to 1% of the height of the viewport
  - vmin - relative to 1% of the viewport's smaller dimension
  - vmax - relative to 1% of the viewport's larger dimension
  - **%** - relative to the parent element

The bolded and underlined ones are important and are generally used.
--Regarding vw and vh, imagine the viewport of browser divided into 100*100 grid, then 1vw would be width of one cell of grid and 1vh would be height of one cell of grid.
--rem is based on root or html font size. As the default font size of html is 16px, so 1rem = 16px. If we style the font-size of html element as 10px in css, then 1rem would now be 10px. So, rem is based on html's font size or root.
--em is like rem but is relative to the font size of its parent.
--For resposiveness we generally use rem. User preference of small, medium or large font will not be effected. Sometimes, using em may not be responsive when it is used for the fonts of texts inside nested element as it is relative to the parent font.
--For making website responsive, we generally use realtive font sizes.

Some commonly used properties

For changing background color:

background-color: red;

Some properties for adding image:

background-image: url("img.jpg");

background-size: contain*(or cover)* (*or 150px) {or 150px(width) 100px(height)}*;

background-repeat: no-repeat;

background-position: center*(both x-axis and y a-xis) {or right(x-axis) bottom(y-axis)***};**

For aligning the text to center of block or inline-block element:

text-align: center;

Other values for text-align are right, left, justify.

For wrapping letter or image along with elements (around texts):

float:left (or right)**;**

For avoiding the overlapping of floating element on an element:

clear: right(or left or both)**;**

Here, right or left or both are the place of the elements where the overlapping elements are floating and need to be cleared or avoid overlapping.

For specifying border:

border-width: 2px;

border-style: solid (*or dashed or dotted or others*)**;**

border-color: blue;

This can also be alternately used as:

border: 2x solid blue;

For giving shadow to the box:

box-shadow: 5px 10px 15px red;

Here, 5px is for the position of shadow towards x-axis, 10px towards y-axis, 15px is for the level of bluriness of the shadow(more the pixel more the level) and red is for the color of the shadow.

For manipulating transparency of an object:

opacity: 0.5;

Here, the value in decimal is the level of transparency that ranges from 0-1. It is the alternative of alpha in rgba.

For adjusting the text inside a box:

- max-width
- min-width
- using both property

max-width is a property of inline block element for adjusting the texts within the maximum width of the box and not letting the text to overflow from the box.

But for min-width property, the width of the box will keep on expanding as there is more text inserted.

Using both property will only expand the box vertically keeping horizontal size of the box fixed as more text is inserted.

For avoiding overflow of large texts inside content box:

Property: **overflow**

Set value: **auto**(not hidden or visible)

For centering a block element on the centre of the page:

margin-left: auto;

margin-right: auto;

Alternatively,

margin: 50px auto;

or, **margin: 50px auto 50px auto;**

Also, **margin: auto;**

This will directly centers the element to the center from left and right.

For transitioning an element:

It is usually used for bringing transition to an element on hovering to an element(espicially hyperlink text in an anchor element).

transition: background-color 0.2s, color 0.5s, padding 1s;

Example:

In HTML,

```
<a href="nothing">Click Me!</a>
```

In CSS,

```
a {
            display: inline-block;
            font-size: 25px;
            background-color: white;
            border: black 2px solid;
            padding: 10px 25px;
            color: black;
            text-decoration: none;
            transition: background-color 0.2s, color 0.5s, padding 1s;
}
a:hover {
            background-color: black;
            color: white;
            padding: 15px 33px;
}
```

For transforming an element:

For translating,

transform: translate(x co-ordinate, y co-ordinate);

where, the origin is the original position of the element and x co-ordinate and y co-ordinate are in pixels or other units.

For rotating,

transform: rotate(degree_of_rotation);

where the degree of rotation is written as deg as unit like 45deg, 90deg, 30deg, etc.

For styling lists items:

list-style

The values may be **circle**, **decimal**, **none**, etc.

The display property

The HTML block elements are:

- paragraphs(<p>)
- headers(<h1> to<h6>)
- divisions( )
- lists and list items(<ol>, <ul>, <li>)
- forms(<form>)

The HTML inline elements are:

- spans()
- images(<img>)
- anchors(<a>)

The keyword 'display' is used as property for displaying block or inline element in a particular manner.

Example:

display: inline(if the element is block element) {or ***block***(if the element is inline element) or ***inline-block***}**;**

Here, with inline the element takes only the minimum possible space that it requires. With block the element takes maximum width of the screen as default. With inline-block the element acts as an inline but unlike inline display whose size can't be adjusted, the size of inline-block can be adjusted. Width and height property will not work with inline value.

Box model in CSS

Every single html elements on screen is treated as box by CSS. By styling the box, we can determine how the elements can be laid.

- **Padding** is the space between the elements and the border.
- **Margin** is the buffer zone between the current element and other element on the screen which creates space between the elements.
- **Border** also takes a size or value which lies between padding and margin.

- Moreover, we can not include in the box model but there can be an outline outside the border which does not take up the size or value in the box model, whose distance from the border is determined by the property: outline-offset

Some properties related to box model:

padding

border

margin

outline

outline-offset

box-sizing

(box-sizing has value of either **content-box** or **border-box**, in content-box the specified size of box is only the size of content inside the padding where border-box is the size of whole elements of box, i.e. content, padding, border and box)

CSS positioning

Positioning is how we move elements around the page.

Types:

1. Relative positioning
2. Absolute positioning
3. Fixed positioning
4. Sticky positioning

Rules of positioning the elements(assume there is no css)

- Content is everything

  Content inside the box determines how large things will be displayed and what the height and the width will be
- Order comes from code

  The order of the elements on the screen comes from html code
- Children sit on parents

  For example, if h1 tag is inside div tag, then h1 will be on the top of div.

Relative positioning

It is an idea of taking an element and moving it from one place where it naturally sits on one place of the page to another place slightly off.

Example:

```
.box{
            width: 300px;
            height: 300px;
```

```
            border: 10px solid black;
            position: relative;
            top: 30px;
            left: 40px;
}
```

Note: It will push the box 30px from the top and 40px from left.

In the body,we have:

```
<div class="box">

    </div>
```

Use:

It is used when we have to move just slightly from one place where it naturally sits.

Absolute positioning

It is moving of element based on the parent element.

Example:

```
.box{
            width: 300px;
            height: 300px;
            border: 2px solid black;
            position: relative;
            top: 30px;
            left: 40px;
}
.second-box{
            width: 30px;
            height: 30px;
            border: 2px solid black;
            background-color: blue;
            position: absolute;
            top: 10px;
            right: 20px;
}
```

Note: (i)This will move the element with class second-box to the position where there is 10px remaining from the top and 20px remaining from the right.

(ii)When the relative position of the parents element is taken away, the child element will now be placed in the position relative to the viewport.

In the body, we have:

```
<div class="box">
        <div class="second-box">
        </div>
```

```
</div>
```

Fixed positioning

It fixes element on a certain part of the viewport and is fixed even when we slide down the page.

Example:

```
.box{
            border: 2px black solid;
            padding: 5px;
            background-color: coral;
            position: fixed;
            top: 0;
            left: 0;
            width: 100%;
        }
```

In the body, we have:

```
<div class="box">
        This is header
    </div>
    <p>Lorem ipsum dolor sit, amet consectetur adipisicing elit. Quasi cum odio
molestiae provident.</p>
    <p>Lorem ipsum dolor sit, amet consectetur adipisicing elit. Quasi cum odio
molestiae provident.</p>
    <p>Lorem ipsum dolor sit, amet consectetur adipisicing elit. Quasi cum odio
molestiae provident.</p>
    <p>Lorem ipsum dolor sit, amet consectetur adipisicing elit. Quasi cum odio
molestiae provident.</p>
    <p>Lorem ipsum dolor sit, amet consectetur adipisicing elit. Quasi cum odio
molestiae provident.</p>
    <p>Lorem ipsum dolor sit, amet consectetur adipisicing elit. Quasi cum odio
molestiae provident.</p>
    <p>Lorem ipsum dolor sit, amet consectetur adipisicing elit. Quasi cum odio
molestiae provident.</p>
    <p>Lorem ipsum dolor sit, amet consectetur adipisicing elit. Quasi cum odio
molestiae provident.</p>
    <p>Lorem ipsum dolor sit, amet consectetur adipisicing elit. Quasi cum odio
molestiae provident.</p>
    <p>Lorem ipsum dolor sit, amet consectetur adipisicing elit. Quasi cum odio
molestiae provident.</p>
    <p>Lorem ipsum dolor sit, amet consectetur adipisicing elit. Quasi cum odio
molestiae provident.</p>
    <p>Lorem ipsum dolor sit, amet consectetur adipisicing elit. Quasi cum odio
molestiae provident.</p>
    <p>Lorem ipsum dolor sit, amet consectetur adipisicing elit. Quasi cum odio
molestiae provident.</p>
```

```
    <p>Lorem ipsum dolor sit, amet consectetur adipisicing elit. Quasi cum odio
molestiae provident.</p>
    <p>Lorem ipsum dolor sit, amet consectetur adipisicing elit. Quasi cum odio
molestiae provident.</p>
    <p>Lorem ipsum dolor sit, amet consectetur adipisicing elit. Quasi cum odio
molestiae provident.</p>
    <p>Lorem ipsum dolor sit, amet consectetur adipisicing elit. Quasi cum odio
molestiae provident.</p>
    <p>Lorem ipsum dolor sit, amet consectetur adipisicing elit. Quasi cum odio
molestiae provident.</p>
    <p>Lorem ipsum dolor sit, amet consectetur adipisicing elit. Quasi cum odio
molestiae provident.</p>
    <p>Lorem ipsum dolor sit, amet consectetur adipisicing elit. Quasi cum odio
molestiae provident.</p>
    <p>Lorem ipsum dolor sit, amet consectetur adipisicing elit. Quasi cum odio
molestiae provident.</p>
    <p>Lorem ipsum dolor sit, amet consectetur adipisicing elit. Quasi cum odio
molestiae provident.</p>
    <p>Lorem ipsum dolor sit, amet consectetur adipisicing elit. Quasi cum odio
molestiae provident.</p>
    <p>Lorem ipsum dolor sit, amet consectetur adipisicing elit. Quasi cum odio
molestiae provident.</p>
    <p>Lorem ipsum dolor sit, amet consectetur adipisicing elit. Quasi cum odio
molestiae provident.</p>
    <p>Lorem ipsum dolor sit, amet consectetur adipisicing elit. Quasi cum odio
molestiae provident.</p>
    <p>Lorem ipsum dolor sit, amet consectetur adipisicing elit. Quasi cum odio
molestiae provident.</p>
    <p>Lorem ipsum dolor sit, amet consectetur adipisicing elit. Quasi cum odio
molestiae provident.</p>
    <p>Lorem ipsum dolor sit, amet consectetur adipisicing elit. Quasi cum odio
molestiae provident.</p>
    <p>Lorem ipsum dolor sit, amet consectetur adipisicing elit. Quasi cum odio
molestiae provident.</p>
    <p>Lorem ipsum dolor sit, amet consectetur adipisicing elit. Quasi cum odio
molestiae provident.</p>
    <p>Lorem ipsum dolor sit, amet consectetur adipisicing elit. Quasi cum odio
molestiae provident.</p>
    <p>Lorem ipsum dolor sit, amet consectetur adipisicing elit. Quasi cum odio
molestiae provident.</p>
    <p>Lorem ipsum dolor sit, amet consectetur adipisicing elit. Quasi cum odio
molestiae provident.</p>
    <p>Lorem ipsum dolor sit, amet consectetur adipisicing elit. Quasi cum odio
molestiae provident.</p>
    <p>Lorem ipsum dolor sit, amet consectetur adipisicing elit. Quasi cum odio
molestiae provident.</p>
    <p>Lorem ipsum dolor sit, amet consectetur adipisicing elit. Quasi cum odio
molestiae provident.</p>
    <p>Lorem ipsum dolor sit, amet consectetur adipisicing elit. Quasi cum odio
molestiae provident.</p>
```

```
    <p>Lorem ipsum dolor sit, amet consectetur adipisicing elit. Quasi cum odio molestiae provident.</p>
    <p>Lorem ipsum dolor sit, amet consectetur adipisicing elit. Quasi cum odio molestiae provident.</p>
    <p>Lorem ipsum dolor sit, amet consectetur adipisicing elit. Quasi cum odio molestiae provident.</p>
    <p>Lorem ipsum dolor sit, amet consectetur adipisicing elit. Quasi cum odio molestiae provident.</p>
    <p>Lorem ipsum dolor sit, amet consectetur adipisicing elit. Quasi cum odio molestiae provident.</p>
    <p>Lorem ipsum dolor sit, amet consectetur adipisicing elit. Quasi cum odio molestiae provident.</p>
    <p>Lorem ipsum dolor sit, amet consectetur adipisicing elit. Quasi cum odio molestiae provident.</p>
    <p>Lorem ipsum dolor sit, amet consectetur adipisicing elit. Quasi cum odio molestiae provident.</p>
    <p>Lorem ipsum dolor sit, amet consectetur adipisicing elit. Quasi cum odio molestiae provident.</p>
    <p>Lorem ipsum dolor sit, amet consectetur adipisicing elit. Quasi cum odio molestiae provident.</p>
    <p>Lorem ipsum dolor sit, amet consectetur adipisicing elit. Quasi cum odio molestiae provident.</p>
    <p>Lorem ipsum dolor sit, amet consectetur adipisicing elit. Quasi cum odio molestiae provident.</p>
    <p>Lorem ipsum dolor sit, amet consectetur adipisicing elit. Quasi cum odio molestiae provident.</p>
    <p>Lorem ipsum dolor sit, amet consectetur adipisicing elit. Quasi cum odio molestiae provident.</p>
```

Use:

It is used when we want an element to take it with us on anywhere on the page or have a layer on the top of the page.

Sticky positioning

The fixed positioning will fix the element to the whole viewport. In similar way, sticky positioning works. But the difference between them is that fixed element will remain fixed to the viewport no matter what where sticky element will remain fixed only to the parent element.

Example:

```
.first-paragraph {
            background-color: blue;
            color: white;
}
.second-paragraph {
            background-color: red;
            color: white;
```

```
}
.box {
            background-color: bisque;
            color: black;
            border: 2px black solid;     /*Mentioning these properties of box class
inside children-*/
            padding: 10px;               /*div will override the property of the first-
paragraph and-*/
            position: sticky;            /*second-paragraph classes inside the parent
div*/
            top: 0;
            left: 0;
}
```

In the body, we have:

```
<div class="first-paragraph">
        <div class="box">
            Section 1
        </div>
        <p>Lorem ipsum dolor sit amet consectetur adipisicing elit. Beatae
exercitationem veritatis incidunt?</p>
        <p>Lorem ipsum dolor sit amet consectetur adipisicing elit. Beatae
exercitationem veritatis incidunt?</p>
        <p>Lorem ipsum dolor sit amet consectetur adipisicing elit. Beatae
exercitationem veritatis incidunt?</p>
        <p>Lorem ipsum dolor sit amet consectetur adipisicing elit. Beatae
exercitationem veritatis incidunt?</p>
        <p>Lorem ipsum dolor sit amet consectetur adipisicing elit. Beatae
exercitationem veritatis incidunt?</p>
        <p>Lorem ipsum dolor sit amet consectetur adipisicing elit. Beatae
exercitationem veritatis incidunt?</p>
        <p>Lorem ipsum dolor sit amet consectetur adipisicing elit. Beatae
exercitationem veritatis incidunt?</p>
        <p>Lorem ipsum dolor sit amet consectetur adipisicing elit. Beatae
exercitationem veritatis incidunt?</p>
        <p>Lorem ipsum dolor sit amet consectetur adipisicing elit. Beatae
exercitationem veritatis incidunt?</p>
        <p>Lorem ipsum dolor sit amet consectetur adipisicing elit. Beatae
exercitationem veritatis incidunt?</p>
        <p>Lorem ipsum dolor sit amet consectetur adipisicing elit. Beatae
exercitationem veritatis incidunt?</p>
        <p>Lorem ipsum dolor sit amet consectetur adipisicing elit. Beatae
exercitationem veritatis incidunt?</p>
        <p>Lorem ipsum dolor sit amet consectetur adipisicing elit. Beatae
exercitationem veritatis incidunt?</p>
        <p>Lorem ipsum dolor sit amet consectetur adipisicing elit. Beatae
exercitationem veritatis incidunt?</p>
        <p>Lorem ipsum dolor sit amet consectetur adipisicing elit. Beatae
exercitationem veritatis incidunt?</p>
```

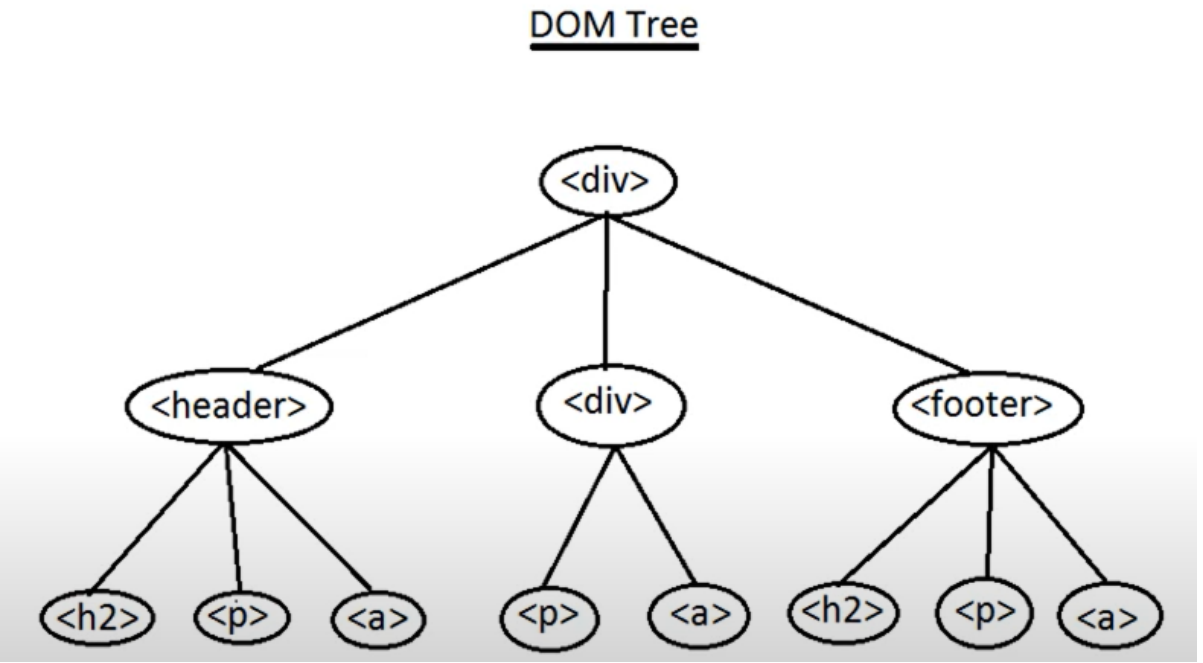
```
        <p>Lorem ipsum dolor sit amet consectetur adipisicing elit. Beatae
exercitationem veritatis incidunt?</p>
        <p>Lorem ipsum dolor sit amet consectetur adipisicing elit. Beatae
exercitationem veritatis incidunt?</p>
        <p>Lorem ipsum dolor sit amet consectetur adipisicing elit. Beatae
exercitationem veritatis incidunt?</p>
        <p>Lorem ipsum dolor sit amet consectetur adipisicing elit. Beatae
exercitationem veritatis incidunt?</p>
        <p>Lorem ipsum dolor sit amet consectetur adipisicing elit. Beatae
exercitationem veritatis incidunt?</p>
        <p>Lorem ipsum dolor sit amet consectetur adipisicing elit. Beatae
exercitationem veritatis incidunt?</p>
        <p>Lorem ipsum dolor sit amet consectetur adipisicing elit. Beatae
exercitationem veritatis incidunt?</p>
        <p>Lorem ipsum dolor sit amet consectetur adipisicing elit. Beatae
exercitationem veritatis incidunt?</p>
        <p>Lorem ipsum dolor sit amet consectetur adipisicing elit. Beatae
exercitationem veritatis incidunt?</p>
        <p>Lorem ipsum dolor sit amet consectetur adipisicing elit. Beatae
exercitationem veritatis incidunt?</p>
        <p>Lorem ipsum dolor sit amet consectetur adipisicing elit. Beatae
exercitationem veritatis incidunt?</p>
        <p>Lorem ipsum dolor sit amet consectetur adipisicing elit. Beatae
exercitationem veritatis incidunt?</p>
        <p>Lorem ipsum dolor sit amet consectetur adipisicing elit. Beatae
exercitationem veritatis incidunt?</p>
        <p>Lorem ipsum dolor sit amet consectetur adipisicing elit. Beatae
exercitationem veritatis incidunt?</p>
        <p>Lorem ipsum dolor sit amet consectetur adipisicing elit. Beatae
exercitationem veritatis incidunt?</p>
        <p>Lorem ipsum dolor sit amet consectetur adipisicing elit. Beatae
exercitationem veritatis incidunt?</p>
        <p>Lorem ipsum dolor sit amet consectetur adipisicing elit. Beatae
exercitationem veritatis incidunt?</p>
        <p>Lorem ipsum dolor sit amet consectetur adipisicing elit. Beatae
exercitationem veritatis incidunt?</p>
        <p>Lorem ipsum dolor sit amet consectetur adipisicing elit. Beatae
exercitationem veritatis incidunt?</p>
        <p>Lorem ipsum dolor sit amet consectetur adipisicing elit. Beatae
exercitationem veritatis incidunt?</p>
</div>
<div class="second-paragraph">
        <div class="box">
            Section 2
        </div>
        <p>Lorem ipsum dolor sit amet consectetur adipisicing elit. Beatae
exercitationem veritatis incidunt?</p>
        <p>Lorem ipsum dolor sit amet consectetur adipisicing elit. Beatae
exercitationem veritatis incidunt?</p>
        <p>Lorem ipsum dolor sit amet consectetur adipisicing elit. Beatae
exercitationem veritatis incidunt?</p>
```

```
        <p>Lorem ipsum dolor sit amet consectetur adipisicing elit. Beatae
exercitationem veritatis incidunt?</p>
        <p>Lorem ipsum dolor sit amet consectetur adipisicing elit. Beatae
exercitationem veritatis incidunt?</p>
        <p>Lorem ipsum dolor sit amet consectetur adipisicing elit. Beatae
exercitationem veritatis incidunt?</p>
        <p>Lorem ipsum dolor sit amet consectetur adipisicing elit. Beatae
exercitationem veritatis incidunt?</p>
        <p>Lorem ipsum dolor sit amet consectetur adipisicing elit. Beatae
exercitationem veritatis incidunt?</p>
        <p>Lorem ipsum dolor sit amet consectetur adipisicing elit. Beatae
exercitationem veritatis incidunt?</p>
        <p>Lorem ipsum dolor sit amet consectetur adipisicing elit. Beatae
exercitationem veritatis incidunt?</p>
        <p>Lorem ipsum dolor sit amet consectetur adipisicing elit. Beatae
exercitationem veritatis incidunt?</p>
        <p>Lorem ipsum dolor sit amet consectetur adipisicing elit. Beatae
exercitationem veritatis incidunt?</p>
        <p>Lorem ipsum dolor sit amet consectetur adipisicing elit. Beatae
exercitationem veritatis incidunt?</p>
        <p>Lorem ipsum dolor sit amet consectetur adipisicing elit. Beatae
exercitationem veritatis incidunt?</p>
        <p>Lorem ipsum dolor sit amet consectetur adipisicing elit. Beatae
exercitationem veritatis incidunt?</p>
        <p>Lorem ipsum dolor sit amet consectetur adipisicing elit. Beatae
exercitationem veritatis incidunt?</p>
        <p>Lorem ipsum dolor sit amet consectetur adipisicing elit. Beatae
exercitationem veritatis incidunt?</p>
        <p>Lorem ipsum dolor sit amet consectetur adipisicing elit. Beatae
exercitationem veritatis incidunt?</p>
        <p>Lorem ipsum dolor sit amet consectetur adipisicing elit. Beatae
exercitationem veritatis incidunt?</p>
        <p>Lorem ipsum dolor sit amet consectetur adipisicing elit. Beatae
exercitationem veritatis incidunt?</p>
        <p>Lorem ipsum dolor sit amet consectetur adipisicing elit. Beatae
exercitationem veritatis incidunt?</p>
        <p>Lorem ipsum dolor sit amet consectetur adipisicing elit. Beatae
exercitationem veritatis incidunt?</p>
        <p>Lorem ipsum dolor sit amet consectetur adipisicing elit. Beatae
exercitationem veritatis incidunt?</p>
        <p>Lorem ipsum dolor sit amet consectetur adipisicing elit. Beatae
exercitationem veritatis incidunt?</p>
        <p>Lorem ipsum dolor sit amet consectetur adipisicing elit. Beatae
exercitationem veritatis incidunt?</p>
        <p>Lorem ipsum dolor sit amet consectetur adipisicing elit. Beatae
exercitationem veritatis incidunt?</p>
        <p>Lorem ipsum dolor sit amet consectetur adipisicing elit. Beatae
exercitationem veritatis incidunt?</p>
        <p>Lorem ipsum dolor sit amet consectetur adipisicing elit. Beatae
exercitationem veritatis incidunt?</p>
```

```
        <p>Lorem ipsum dolor sit amet consectetur adipisicing elit. Beatae
exercitationem veritatis incidunt?</p>
        <p>Lorem ipsum dolor sit amet consectetur adipisicing elit. Beatae
exercitationem veritatis incidunt?</p>
        <p>Lorem ipsum dolor sit amet consectetur adipisicing elit. Beatae
exercitationem veritatis incidunt?</p>
        <p>Lorem ipsum dolor sit amet consectetur adipisicing elit. Beatae
exercitationem veritatis incidunt?</p>
        <p>Lorem ipsum dolor sit amet consectetur adipisicing elit. Beatae
exercitationem veritatis incidunt?</p>
        <p>Lorem ipsum dolor sit amet consectetur adipisicing elit. Beatae
exercitationem veritatis incidunt?</p>
        <p>Lorem ipsum dolor sit amet consectetur adipisicing elit. Beatae
exercitationem veritatis incidunt?</p>
        <p>Lorem ipsum dolor sit amet consectetur adipisicing elit. Beatae
exercitationem veritatis incidunt?</p>
        <p>Lorem ipsum dolor sit amet consectetur adipisicing elit. Beatae
exercitationem veritatis incidunt?</p>
        <p>Lorem ipsum dolor sit amet consectetur adipisicing elit. Beatae
exercitationem veritatis incidunt?</p>
        <p>Lorem ipsum dolor sit amet consectetur adipisicing elit. Beatae
exercitationem veritatis incidunt?</p>
        <p>Lorem ipsum dolor sit amet consectetur adipisicing elit. Beatae
exercitationem veritatis incidunt?</p>
        <p>Lorem ipsum dolor sit amet consectetur adipisicing elit. Beatae
exercitationem veritatis incidunt?</p>
</div>
```

Advanced CSS selectors

Firstly, let us understand the concept of **descendent**, **parent and child**, **adjacent elements** and **siblings**. For this, consider the figure below:



From the above figure,

For descendants:

- <h2>, <p> and <a> are the descendants of <header>
- <p> and <a> are the descendants of
- <h2>, <p> and <a> are the descendants of <footer>
- <header>, <h2>, <p>, <a>, , <p>, <a>, <footer>, <h2>, <p>, <a> are the descandents of

For parent and children:

- <h2>, <p> and <a> are the children of <header>
- <p> and <a> are the children of
- <h2>, <p> and <a> are the children of <footer>
- <header>, and <footer> are the children of

For siblings:

- <h2>, <p> and <a> are siblings whose parent is <header>
- <p> and <a> are siblings whose parent is
- <h2>, <p> and <a> are siblings whose parent is <footer>
- <header>, and <footer> are siblings whose parent is

For adjacent elements:

- <h2> and <p> are adjacent elements
- <p> and <a> are adjacent elements
- <header> and are adjacent elements
- and <footer> are adjacent elements

Some of the advanced CSS selectors are:

- Descendent selector
- Child selector or Direct child selector
- Adjacent selector or Adjacent sibling selector
- General sibling selector

Descendent selector

Selector syntax:

Selector1 Selector2

(spacebar is used)

It selects any html element targeted by selector2, which is descendant(s) of any html element targeted by selector1

Example:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
    <style>
        div header{
            border: 2px black solid;
            background-color: red;
        }
    </style>
</head>
<body>
    <div>
        <header>
            <h1>This is h1</h1>
            <span>This is span</span>
            <p>This is p</p>
        </header>
        <div>
            <h2>This is h2</h2>
            <p>This is another p</p>
        </div>
        <footer>
            <h3>This is h3</h3>
            <span>This is another span</span>
            <div>This is div</div>
        </footer>
    </div>
</body>
</html>
```

Here, the header element along with its children elements h1, span and p which are the descendants of div elements are selected.

Child selector/ Direct child selector

Selector syntax:

Selector1 > Selector2

(greater than sign(>) is used)

It selects any html element targeted by selector2, which is direct child of any html element targeted by selector1

Example:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
    <style>
        div > footer > h2{
            border: 2px black solid;
            background-color: red;
        }
    </style>
</head>
<body>
    <div>
        <header>
            <h2>This is h2</h1>
            <span>This is span</span>
            <p>This is p</p>
        </header>
        <div>
            <h2>This is another h2</h2>
            <p>This is another p</p>
        </div>
        <footer>
            <h2>This is one more h2</h3>
            <span>This is another span</span>
            <div>This is div</div>
        </footer>
    </div>
</body>
</html>
```

Here, h2 element of footer element is selected directly which is the child of footer element and which is in turn the child of div element.

Adjacent selector/ Adjacent sibling selector

Selector syntax:

Selector1 + Selector2

(plus sign(+) is used)

It selects any html element targeted by selector2, which is sibling and adjacent element and should be preceded by any html element targeted by selector1

Example:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
    <style>
        #selector h2 + span{
            border: 2px black solid;
            background-color: red;
        }
    </style>
</head>
<body>
    <div id="selector">
        <header>
            <h2>This is h2</h1>
            <span>This is span</span>
            <p>This is p</p>
        </header>
        <div>
            <h2>This is another h2</h2>
            <p>This is another p</p>
        </div>
        <footer>
            <h2>This is one more h2</h3>
            <span>This is another span</span>
            <div>This is div</div>
        </footer>
    </div>
</body>
</html>
```

Here, span elements of both header and footer are selected which are both the adjacent elements of corresponding h2 element that are in turn the descendants of div with id selector.

General sibling selector

Selector syntax:

Selector1 ~ Selector2

(tilde sign(~) is used)

It selects any html element targeted by selector2, which is sibling and need not be adjacent element and should be preceded by any html element targeted by selector1

Example:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
    <style>
        #selector h2 ~ p{
            border: 2px black solid;
            background-color: red;
        }
    </style>
</head>
<body>
    <div id="selector">
        <header>
            <h2>This is h2</h1>
            <span>This is span</span>
            <p>This is p</p>
        </header>
        <div>
            <h2>This is another h2</h2>
            <p>This is another p</p>
        </div>
        <footer>
            <h2>This is one more h2</h3>
            <span>This is another span</span>
            <div>This is div</div>
        </footer>
    </div>
</body>
</html>
```

Here, p elements of both header and div(child of div) are selected which are both the siblings of corresponding h2 element that are in turn the descendants of div with id selector.

Other selectors

There are some other advanced selectors like:

- Not selector
- nth-child selector

Not selector

Selector syntax:

Group of selectors with common class or element name:not(selector that needs to be excluded from selection)

Here, semi-colon and parentheses with keyword not are used.

Example:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
    <style>
        .box{
            height: 20px;
            border: black 2px solid;
        }
        .box:not(.dontselectme){              /*This will select every elements with
box class except */
            background-color: royalblue;    /*box class with dontselectme sub-class*/
        }
    </style>
</head>
<body>
    <div id="parent">
        <div class="box"></div>
        <div class="box"></div>
        <div class="box"></div>
        <div class="box"></div>
        <div class="box dontselectme"></div> <!--dontselectme is the sub-class of the
box class-->
        <div class="box"></div>
        <div class="box"></div>
        <div class="box"></div>
        <div class="box"></div>
    </div>
</body>
</html>
```

Here, every elements with box class except box class with dontselectme sub-class.

n-th child selector

Selector syntax:

Group of selectors with common class or element name:nth-child(positional number of a particular child among the group)

Example:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
    <style>
        .box{
            height: 20px;
            border: black 2px solid;
        }
        .box:nth-child(8){    /*This will select the nth child whose number is
mentioned in the box(say 8)*/
            background-color: yellow;
        }
    </style>
</head>
<body>
    <div id="parent">
        <div class="box"></div>
        <div class="box"></div>
        <div class="box"></div>
        <div class="box"></div>
        <div class="box"></div>
        <div class="box"></div>
        <div class="box"></div>
        <div class="box"></div>   <!--This one is the 8th child-->
        <div class="box"></div>
        <div class="box"></div>
    </div>
</body>
</html>
```

Here, 8th child among the given group of children with common box class is selected.

Here, we can also use "**even**" or "**odd**" inside the parentheses instead of number for selecting the group of either of the even childen or odd childen.

All selector

It selects everything inside the body of html.

Syntax:

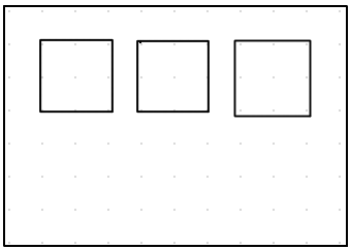
***{property: value;}**

Example:

```
*{
            box-sizing: border-box;
}
```

Flexbox

Flexbox is a CSS display type designed to help us craft CSS layouts in much easier way. It helps to control the position, size and spacing of elements relative to their parent elements and each other. It works with great responsiveness. So, flexbox is a layout module that makes it easier to design flexible and responsive layout structure without the use of elements like float, position or table. We need to apply a display type of flex to the parent container for using flex properties to make the boxes flexible. The children elements inside the parent element are called flex items which are flexible i.e. we can control how they shrink, grow, etc.



Its uses are:

- Navigation bars and menus
- Grid layouts
- Bar charts
- Equal height columns

Properties related to flex box:

- flex-grow
- flex-shrink
- flex-wrap
- flex-basis

flex-grow and flex-shrink

They specify the rate at which flex-items grow or shrink. This is when the size of viewport increases or decreases. flex-growth and flex-shrink are for relative measurement. Note that these properties are given to container which contains flex items.
flex-growth and flex-shrink take value in number without any units which specifies the rate of growth or shrink.
Example:
flex-grow: 1;
flex-shrink: 0;

flex-wrap

When we don't want to scroll to right and don't want elements to grow or shrink arbitrarily and just want to wrap the elements or flex items to the next line when the size of viewport is smaller than the size of flex items, then flex-wrap property is used. Note that this property is given to the container which contains flex items.
Values for flex-wrap property: **wrap** or **wrap-inverse**

flex-basis

flex-basis is pretty similar to min-width or max-width. But, the difference is that, in min-width and max-width if the page viewport width is smaller than the size of elements, then we get a scroll bar as the size of the elements remain constant but in flex-basis the size changes and adjusts with the change in size of viewport but we do not get the scroll bar.
Actually, flex-basis is the starting width which is often used with other properties like flex-wrap, flex-grow and flex-shrink.
It is similar to max-width if we use it merely and is similar to min-width if we use it with flex-grow.
If we use flex-basis and give it some value(say 100px), then when the width of viewport is lesser than the flex items with the given width, the flex-box shrinks in any case.
Note that this property is given to the flex items which are inside the flex container.
It takes values in px, % or any other related values.
Note:

- There are some other properties related to flex box like:
    - **flex-direction**: It directs the flex items horizontally or vertically with the values **row** or **column** respectively where the default value is row. **column-reverse** and **row-reverse** can also be used as value.
    - **align-items**: It aligns the flex items inside the flex container in vertical direction in either center, starting line or ending line with values **center**, **flex-start** or **flex-end** respectively where the default value is flex-start.

- **justify-content:** It aligns the flex items inside the flex container in horizontal direction with values like **center**, **left** and **right** or adjusts the space between flex items in different ways with different values like **space-around**, **space-between** and **space-evenly** which is also in the horizontal direction.
  Note that justify-content and align-item properties are given to the flex container.
- **flex-flow:** It is the short hand property for flex-direction and flex-wrap properties. The first value is same as that meant for flex-direction and seocnd for flex-wrap.
- **order:** It is for rearranging or reordering the flex-items from the default positions. The value for this property is a number. The order is manipulated according to the different number with different values. Like for four flex items, the flex item with value 5 will be at last and that with value -10 will be at first in order. Any number can be given as value but ordering is in accordance to the value of the number given to the individual items. Note that the default number as value for any flex item for order property is 0(zero).
- **align-self:** It aligns only an individual item without disturbing the position of other items. It takes values same as align items and is given to the individual flex item inside the flex container.

- The demonstration of flex properties is shown in the codes below:
  In HTML,

```
<div class="fcontainer">
    <div class="fitem item1">1</div>
    <div class="fitem item2">2</div>
    <div class="fitem item3">3</div>
    <div class="fitem item4">4</div>
</div>
```

In CSS,

```
.fcontainer{
border: 2px solid grey;
height: 45vh;
display: flex;
/* flex-direction: column-reverse; */
/* justify-content: flex-end; */
/* justify-content: flex-start; */
justify-content: space-between;    /* This is horizontal alignment */
align-items: center;        /* This is vertical alignment */
/* align-items: flex-end; */
/* align-items: flex-start; */
```

```
    /* flex-wrap: wrap; */
    /* flex-wrap: wrap-reverse; */
    /* flex-flow: column wrap; */
    }

    .fitem{
        background-color: red;
        border: 2px solid black;
        margin:12px;
        width: 122px;
        height: 122px;
        flex-basis: 10%;
        flex-shrink: 2;
    }
    .item1{
        order: 3;
        /* flex-grow: 3; */
        /* flex-shrink: 2; */
        /* align-self: center; */
    }
    /* .item4{ */
        /* order: 4; */
        /* flex-grow: 2; */
    }
```

- We can use flex box for tables:
  In HTML,

```
<div id="container">
        <div>First</div>
        <div>Second</div>
        <div>Third</div>
        <div>Fourth</div>
        <div>Fifth</div>
        <div>Sixth</div>
        <div>Seventh</div>
        <div>Eighth</div>
    </div>
```

In CSS,

```
        *{
            box-sizing: border-box;
        }
        #container{
            border: red 2px solid;
            display: flex;
            flex-wrap: wrap;
        }
        #container div{
```

```
        border: 1px solid black;
        flex-basis: 20%;
        flex-grow: 1;
        flex-shrink: 0;
    }
```

Note that while making table and if you want to fit the flex items in an exact way(like 5 boxes with 20% flex-basis or 4 boxes with 25% flex-basis in one line), then we need to give the related elements the property of box-sizing with value of border-box.

- We can use flex-box for page layout:

  In HTML,

```
<div id="container">
    <div class="header">Header</div>
    <div class="navigation">Navigation</div>
    <div class="container2">
        <div class="menu">Menu</div>
        <div class="content">
            <h1>Hello world</h1>
            <div>Lorem ipsum dolor sit amet consectetur adipisicing elit.
            Ipsam ut iusto, harum deleniti dolor esse numquam ipsum
            nostrum labore earum!
        </div>
    <h2>Hello world again</h2>
    <div>Lorem ipsum, dolor sit amet consectetur adipisicing elit.
         Iusto ea repudiandae, consectetur atque optio voluptatum.
    </div>
    </div>
</div>
```

In CSS,

```
    #container{
        display: flex;
        flex-wrap: wrap;
    }
    .header{
        flex-basis: 100%;
        border-bottom: 3px solid grey;
        background-color: lavenderblush;
        padding: 30px;
        box-sizing: border-box;
        flex-grow: 1;
        flex-shrink: 0;
        margin: 0 0 5px 0;
        text-align: center;
    }
    .navigation{
        flex-basis: 100%;
```

```
            flex-grow: 0;
            flex-shrink: 1;
            border: 1px solid black;
        }
        .container2{
            display: flex;
        }
        .container2 >.menu{
            border: 1px solid red;
            flex-basis: 20%;
            flex-grow: 1;
            flex-shrink: 0;
        }
        .container2 >.content{
            flex-basis: 80%;
            flex-grow: 1;
            flex-shrink: 0;
        }
```

- We can also add the three properties of flex item which are flex-grow, flex-shrink and flex-basis with their value in the same property i.e. **flex** in the following way:
  flex: 1 0 200px;

  Here, 1 is flex-growth, 0 is flex-shrink and 200px is flex-basis.

CSS Grids

It is a grid-based layout system with rows and columns. It also offers an easy and convenient layout.
Grid container is the entire container of the grid.
Grid lines are the lines in the grid.
Grid cell is one small box of the grid.

Basically, the difference between flex-box and CSS grid is that flex-box is one-dimensional layout in columns or rows where grid is the two-dimensional layout in columns and rows. Also, in flex-box, there are multiple flex items with all kinds of alignments but in CSS grid, there is a single main grid inside which cells and grid items are customized.

First, use:
display:grid;
(to the grid container)
Also in container,

Property for separating number of columns or rows and scale them:

grid-template-rows and **grid-template-columns**

Values can be given in different ways:

70% 30% (This value will create either rows or columns, two in number such that each occupy height(for row template) or width(for column template) of 70% and 30% in the container)

or,**40% 30% 30%** (This value will create either rows or columns, three in number such that each occupy height or width of 40% 30% and 30% in the container)

or, **1fr 1fr 1fr** (This value will create either rows or columns, three in number such that each occupy height or width of 1 fraction each of the total fraction of the container in the container)

or, **1fr 2fr 1fr**(This value will create either rows or columns, three in number such that each occupy height or width of 1 fraction, 2 fraction and 1 fraction of the total fraction of the container in the container)

or,**repeat(3, 1fr)** (This value will create either rows or columns, three in number by repeating 1fr three times and thus works same as 1fr 1fr 1fr)

or, **repeat(4, 1fr 2fr)** (This value will repeat 1fr and 2fr 4 times such that it will create 8 rows or columns)

or, **100px 1fr 1fr**

or, **100px 1fr 50px** (These values will create the rows or columns of the specified size and also of the specified fraction from the total fraction of the container which is in accordance to the size of container and the number of rows or columns)

Using these values in fraction is more relevant than using in percentage for this property.

Property for maintaining gap between columns:

grid-column-gap

Property for maintaining gap between rows:

grid-row-gap

Property for maintaining gap between all columns and rows:

grid-gap

These three properties take value in regular web measurements like px, em, rem, vh, vw,etc.

Now in grid items,

Property for justifying(horizontal) grid items:

justify-items

Property for aligning(vertical) grid items:

align-items

Values for these two properties:

start, **end**, **center**

Also, in individual grid item,

Property for justifying or aligning individual grid item:

justify-self

align-self

Values:

start, **end**, **center**

Note: **place-self** property can be used in place of both align-items and justify-items.

Property for spanning individual grid items from one line to another:

Along column through vertical lines:

grid-column

Along row through horizontal lines:

grid-row

These properties can be used in the following way:

grid-column: 1/3;

grid-row: 2/4;

1 and 3 in grid-column represent the vertical line numbers and 2 and 4 in grid-row represent the horizontal line numbers.

There are properties which can be used alternately for these properties. They are:

grid-column-start

grid-column-end

grid-row-start

grid-row-end

Here, grid-column-start and grid-column-end take the value in a number for the starting vertical line and the ending vertical line respectively of the span of the column. And, similarly grid-row-start and grid-row-end take the value in a number for the starting horizontal line and the ending horizontal line respectively of the span of the column.

Example:

grid-column-start: 1;

grid-column-end: 3;

We can also span the grid items through the grid cells instead of grid lines in the following ways:
grid-column: 1/ span 3;
grid-row:1/ span 2;
Here, for the property grid-column, the grid item will be spanned through grid cells in horizontal way such that the given grid item will be spanned from 1st grid cell in row to the 3rd cell.
And, for the property grid-row, the grid item will be spanned through grid cells in vertical way such that the given grid item will be spanned from 1st grid cell in column to the 3rd cell.

The use of these properties for creating a sample of layout is shown below:
In body of HTML,

```
<div class="container">
    <div class="one">One</div>
    <div class="two">Two</div>
    <div class="three">Three</div>
    <div class="four">Four</div>
    <div class="five">Five</div>
</div>
```

In CSS,

```
.container{
    display: grid;
    height: 500px;

    /* grid-template-columns: 20% 50% 30%; */
    /* grid-template-columns: 50% 50%; */
    /* grid-template-columns: 1fr 1fr 1fr; */
    /* grid-template-columns: 1fr 2fr 1fr; */
    /* grid-template-columns: repeat(3, 1fr); */
    /* grid-template-columns: repeat(4, 1fr 2fr); */
    grid-gap: 10px;
    /* grid-row-gap: 30px; */
    /* grid-column-gap: 30px; */
    grid-template-columns: 100px 1fr;
    grid-template-rows: 150px 1fr 1fr 50px;
}
.container div{
    background: #eee;
}
.container div:nth-child(odd){
    background: #ddd;
}
.container > .one{
    grid-column: 1/4;
```

```
        }
        .container > .two{
            grid-row-start: 2;
            grid-row-end: 4;
        }
        .container > .three{
            grid-column: 2/4;
        }
        .container > .four{
            grid-column: 2/4;
        }
        .container > .five{
            grid-column: 1/4;
        }
```

The use of these properties for manipulating items is shown below:
In body of HTML,

```
    <div class="container">
        <div class="one">One</div>
        <div class="two">Two</div>
        <div class="three">Three</div>
        <div class="four">Four</div>
        <div class="five">Five</div>
    </div>
```

In CSS,

```
        .container{
            display: grid;
            height: 600px;
            width: 600px;
            background-color: blue;
            /* grid-row-gap: 10px; */
            grid-gap: 10px;
            grid-template-columns: repeat(3, 1fr);
            /* justify-items: start; */
            /* justify-items: end; */
            justify-items: center;
            /* align-items: start; */
            align-items: center;
            /* align-items: end; */
        }
        .container div{
            background-color: grey;
            width: 80px;
            height: 80px;
        }
        .container div:nth-child(odd){
            background-color: brown;
        }
```

```
        .container > .one{
            grid-column: 1/ span 3;
            align-self: start;
        }
        .container > .five{
            grid-column: span 2;
            justify-self: end;
        }
```

Fonts manipulation

Types of fonts on the basis of availability and access:

- **web-safe fonts**: fonts which come pre-installed in the OS
- **web-fonts**: fonts to be taken from web

Font stacking: Fonts can be stacked which specifies a prioritized list of font family names or generic names. From the stacked fonts, first font from the left will be taken and is mostly prioritized and if it is not available, i.e. not installed in the OS, then other stacks of fonts will be taken which is also done according to availability and priority. In this stacking, take into mind that the last font(as in the right) should be generally available in most of the operating systems.

Hence, web safe fonts are fonts that are the fonts that are pre-installed by many operating systems. While not all the systems have the same fonts installed, you can use the web safe font stack to choose several fonts that look similar and are installed on the various systems that you want to support. If you want to use fonts othere than ones pre-installed, as of CSS3, you can use web fonts.

We can import web fonts and place its code in our html code inside head tag which is also called as fonts embedding. The best website to import fonts or for font embedding is googlefonts.com

Font stacking is done as value within the the property, font-family for any header, paragraph, span, etc. Here, for stacking the fonts as value for font-family property, we stack the name of the fonts.

Example:

```
p {
        font-family: 'Roboto', sans-serif;
}
```

Some properties for customizing the fonts:

font-size(for size of fonts)

font-weight(for boldness of the texts)

font-style(for italicising the texts)

text-decoration(for underlining the texts)

line-height(for manipulating the space between the lines if the text is broken into more than one lines)
letter-spacing(for manipulating the spaces between all the letters)
Here, the value for font-size and line-height is in the web measurements like px, em,etc. , for font-weight is **bold**, for font-style is **italic** and for text-decoration is **underline**.

Example use of fonts:

In HTML,

```
<h1>This is header</h1>
    <p>Lorem ipsum dolor sit amet <strong>consectetur</strong> adipisicing elit.
<span>Repellendus</span>, quaerat!</p>
```

In CSS,

```
h1 {
            font-family: 'Roboto', sans-serif;
            font-size: 50px;
        }
        p {
            font-family: 'IBM Plex Sans Thai Looped', sans-serif;
            line-height: 100px;
        }
        p{
            font-style: italic;
        }
        span{
            font-weight: bold;
            font-style: normal;
            text-decoration: underline;
        }
```