**CSS style**

<h1 style="color: blue;">

Style = “color:red”

Css is a style sheet

Color, images center, box right left

Style and format the pages.

3 types of css:

Inline- when the code is done in a single line

Internal- when the html attributes are separate and the styles are added from a separate place

External CSS

**Inline**

<p style="color:chocolate; background-color: bisque;"> color is aqua and bg is bisque</p>

All code is done within a line so it is called as a inline CSS

<a style=”color: value;background clor :red”> text to be displayed

**Internal CSS**:

Added 2 attributes h1 and paragraph p

Now add a style tag

<style>

Here pass the attributes and then provide the styles

</style>

<style>

    h1{

*color*: darkgreen;

*background-color*: chartreuse;

    }

</style>

Pass the style attrbutes inside style in a curly braces

**Div tag**

**It is like a container or a division. For dividing some values.**

Div tag is used for seprating the section

It does not have any value to it, if nothing is defnied it wont do anything.

We have to specify a color or bg color attributes to it,

If not it will take the existing value formats.

In div value is passed separetly, in the style tag- those attributes of the div is passed.

div {

*background-color*: yellowgreen;

*height*: 100px;

*width*: 200px;

        }

    </style>

    <h1>headings</h1>

    <p>pradaaaaaaaaaaaaaaaaaaaaaaaa</p>

    <div>

        <h3 style="color: blue;">heading in divs</h3>

        <p style="color: black;"> paragraph in black</p>

    </div>

**External CSS: mostly** is used in many cases. Cause for huge projects are used.

Now before inline and internal all the styles are formatted within the html document,

Whereas externall css is done outside the html docuemtn.

Name the file as .css

Link tag for relation

<link rel="stylesheet" href="styles.css">

Link relation is the type of file and the href is for location of the css file for mapping.

**Selectors: element, class, id and universal.**

Element selector- direcly call the element in the css style

Class selector- class=”” in the tag

For calling the class tag in ths style using the . dot in front of the css commands

Id selector- id- id should be unique

For calling the calue id- use the # in front.

For calling al the values from the selector you can use the star or aestrix symbol in the css sheet.

.#\*{

color: red;

}

<h1 class ="classselector">

        heading styles

    </h1>

    <h1 id="two">

        second heading

    </h1>

        <h1>third heading</h1>

**CSS Box model**

All the element will be in a box shape.

Select the element and select the body.

Now create an attribute and add a style tag.

Inside that add padding.

**Four types of padding are there.**

**Padding top right, left bottom**

**Margin top right left botttom**

**If padding top- the text inside the box will be moved**

**If margin top- the entire box along with the text will be moved.**

**Border**

**For setting up a borer- we need to add 3 things**

**Color width style**

 border-color: aquamarine;

        border-width: 5px;

        border-style: double;

        margin: 100px;

        padding: 100px;

**Display property**

**Add display tag:**

**4 types-**

**Inline**

**Inline**

**Inline block**

**Block-Always the element property will block which means, it will occupy the entire width.**

**It will not allow other elemtns in the row. Nothing in front and back.**

**Inline- the element will be based on the text size alone. Width based on the content**

**Will allow the other elements in front and back.**

**If inline block- it will not move to a next element. Additionaly we can add the width to the element**

**And since it is not a block , without moving to the next elment- width can be adjusted.**

**If we need to add any width then inline block should be used**

**Navigation**

**Nav tag**

**Is also same like div tag.**

**To work on a UI**

**Basic things-**

**First seprate the things in the web page in the order.**

**And then add the div and class to the features**

**Heaer, search, product, contact.**

**Add it in the div and then under the nav tags.**

**Now some items will have efault margin and padding- remove all in the style sheet.**

**Using \*{margin: 0, padding: 0}**

**How to handle mouse hover in the CSS**

**On mouse hovering- hovering option is there.**

li:hover{

    color: wheat;

}

li:hover{

    color: wheat;

    cursor: pointer;

}

**Cusrsor- pointer for changing the mouse to pointer on mouse hovering the attribute.**

**Text align center will align the text as well as the inline elements too.**

**When mouse hovering the background color changes.**

.box:hover{

    background-color: black ;

    color: white;

}

**Box is the class of the div.**

**Use the hover tag{ color, bg color:}**

**For changing the text and font size in inline.**

**<p style = margin top: 30px; font size=12;> </P>**

**We can give it in a separate file and styles too.**

**CSS POSTIONING:**

**Important topic**

**5 types of positioning:**

**Static,**

**Relative**

**Absolute**

**Fixed**

**Sticky**

**By default all the position of the attributes are static.**

**First we have to change the position to relative to move.**

**We can move he move the object by margin and padding.**

**But on that, the front and after objects will be moved.**

**If its relative- it can be appended on top of others.**

**If its static- you can not move it using the keywords- top right left right**

**But in relative – it can be moved and it can be appened. Can use top right left right**

**In relative- it will be original position and can be moved through a rope like.**

**Z index**

**To define which value should be on top. High number- on top..**

top: 80px;

right: 10px;

margin-top: 10px;

position: relative;

z-index: 3;

**Absolute**

**If we define any adjustment- it is based on the border.**

**Relative**

**10pixels**

**Object-----------------------🡪current position**

**Absolute 10pixel**

**Window border-------------------------🡪Current position**

**If we make parent tag as**

**If the parent tag is static- the the child object if absolute will anywhere.**

**If the parent absolute and relative- then based on that the child object will move if absolute.**

**Parent tag as index--------------🡪 current position**

**If the parent tag is mdae/absoluteand relative then chile object is absolute. Then it will consider parent object as base before moving.**

**Fixed**

.fh{

        position: fixed;

        margin-top: 10px;

        margin-left: 100;

    }

**It will fixed on top of everything- it wont move. Stay on the same screen**

**Stikcy position**

**On a aparticular condition we can make it stick to the screen.**

**CSS Flexbox**

**It is like a layout.**

**When giving the conditions, flex pass those in the parent div.**

**It groups the values inside the div tag.**

 div {

        display: flex;

        flex-direction: row;

        flex-wrap: wrap;

        justify-content: space-between;

        background-color: darkolivegreen;

        padding: 10px;

    }

You will be passing the values to those tags when they are under the parent tag.

Flex direction will provide the direction in which the tags inside the parent tag goes.

Flex wrap- automatically adjusts the tags inside the div, equally.

Justifty content- also provides so many options for even spacing.

\

\

    div {

        display: flex;

        flex-direction: row;

        flex-wrap: wrap;

        justify-content: space-between;

        justify-content: center;

        background-color: darkolivegreen;

        padding: 10px;

    }

 flex-basis: 21%;

    flex-grow: 1;

**flex basis- will define the element percentage in the screen. Along with thte padding and the margin size.**

**And Flex grow gives an option to adjust the element to the screen width**

<style>

div{

    display: flex;

    background-color: darkolivegreen;

    color: black;

    flex-wrap: wrap;

    /\* justify-content: space-around; \*/

}

h1{

    background-color: goldenrod;

    margin: 2px;

    padding: 2px;

    flex-basis: 21%;

    flex-grow: 1;

}

</style>

**Flex- important**

**Main axis- cross axis**

**Flex axis- if row-- --- then main axis is row and cross axis is column**

**And if flex axis is column |||- then main axis is column and cross axis is row.**

**In main axis we will use- justify**

**In cross axis we will use align will be used.**

**CSS media query**

**To make the page responsive- we are going to use the media query**

**For this we will be using th tag @media**

**Media for passing some values in**

**{}**

**This is like an condition- if the width is is this then have values like this**

**We can make multiple media conditions based on the width size which will define the mobile or PC view**

<style>

    h1 {

        color: red;

    }

    @media print {

        h1{

            color: blue;

        }

    }

</style>

**Max width**

**Min width**

**If we define the max width- the system will set that properties from 0 to 600 pixels for that features.**

**When we are using responsive designs- then we have to use the meta code**

**Meta – responsive design**

<style>

    div{

        background-color: red;

     height: 100px;

     width: 50px;

     content: solid;

     border: 15px;

     border-color: black;

     border-style: solid;

    }

<style>

    h1 {

        color: red;

    }

    @media screen and (max-width:800px) {

        /\* when the screen size from 0 to 600 this code will be effective \*/

        h1 {

            color: blue;

            height: 100px;

            width: fit-content;

        }

        div {

            background-color: brown;

        }

    }

**Max and min width.**

**Max width defines 0 to that max value**

**Min width defines from that value to infinity..**

<style>

.num{

display: flex;

justify-content: space-around;

flex-wrap: wrap;

}

.one{

    background-color: green;

    margin: 5px;

    height: 100px;

    /\* width: 50px; \*/

    flex-basis: 20%;

}

@media screen and (max-width:700px) {

   .one{

    flex-basis: 40%;

   }

}

@media screen and (max-width:400px) {

   .one{

    flex-basis: 100%;

   }

}

**Instead of inline block- flex and float**

**What to use and when to use.**

**Flex basis- justify content- focus harder.**