**CSS TUTORIAL**

(Cascading style sheets)

It is used to style and format web pages created with HTML.

**TYPES:**

**INLINE CSS**

<h1 style="color:#ff9d00;background-color:black">hello</h1>

**Internal CSS**

<style>

    h1{

        color:red;

        background-color: black;

    }

    p{

        color:yellow;

    }

</style>

<h1>hello</h1>

<p>paragraph</p>

* Using style tag means it means we should use the internal CSS is the meaning of that.
* Div tag means division tag and then used in the CSS for styling purposes.
* Div tag doesn’t have any height by default if something is written inside the div section only it is shown.

<style>

    div{

        background-color: red;

    }

</style>

<div>

</div>

* Only written something means only the background color is visible or else nothing is seen.
* The div height is taken as the height of the content written inside that.

<style>

    div{

        background-color:red;

        height:100px;

    }

</style>

<div>

</div>

* But height can also be taken to the div content.

<style>

    div{

        background-color:red;

        height:100px;

        width: 300px;

    }

</style>

<center>

<div>

    <h1 style="color:white">Error makes clever</h1>

    <h2 style="color:blue">coimbatore</h2>

</div>

</center>

**External CSS**

* Create a separate CSS style file and give it as a link to the HTML page

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

<center>

<div>

    <h1 >Error makes clever</h1>

    <h2 >coimbatore</h2>

</div>

</center>

h1{

    color:red;

    background-color: black;

}

h2{

    color: aqua;

    background-color: blueviolet;

}

**CSS SELECTORS:**

Element selector

Class selector

ID selector

Universal selector

**Element selector:**

* Element selector means that the CSS code is accessible only for that particular tag.
* If you write the style for h1 then all the h1 contents get the same style that you have written

**Class selector:**

* If there are many h1 tags but you need different color and styles for each one then we use a class selector.
* The attribute that is used for this is known as class and value is written inside them
* To access use the .class name

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

<center>

<div>

    <h1 class="one">Error makes clever</h1>

    <h1 >coimbatore</h1>

</div>

</center>

.one{

    color:red;

    background-color: black;

}

h1

{

    color: aqua;

    background-color: blueviolet;

}

**ID selector:**

* Id is the attribute that is used for this selector
* And before this, we need to use #
* The ID is unique so no other tag can have the same id on them
* But class be in the same name for two different tags

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

<center>

<div>

    <h1 class="one">Error makes clever</h1>

    <h1 id="two">coimbatore</h1>

</div>

</center>

.one{

    color:red;

    background-color: black;

}

#two

{

    color: orange;

    background-color: blueviolet;

}

**Universal selector:**

* Is the symbol that is used it is for all the tags and elements inside them

**CSS BOX MODEL**

* It is a box model that is needed for wrapping around every HTML element.

**Margins**

**Borders**

**Padding**

**Content**

**CONTENT:**

* The content is padded then bordered then it is put with a margin in it.
* All the elements are written and created inside a box and the box can be controlled by us.

**Padding:**

* Padding is done up down left and right.
* It is the space between the content and the background color box.

**Margin:**

* It is also the same 4 types
* It is the space between the background box and the webpage

**Border:**

* It is the border that is done to give a line to the box of the element.
* To set a border we need 3 things:

**border-color**

**border width**

**border-style**

* text-align is the replacement of the center tag from the HTML with options like center top right and left.

**span tag:**

* takes only the space of the code that is written not the other than that.

**display property:**

**block:**

* the takes the width of the whole screen.
* by default, it is block only.
* only that element comes in a single-line
* list elements are also in block only

**inline:**

* it takes the width of the content till it is present only
* height and width cannot be set
* all the contents come in the same line only but it takes all the elements in the same line when the size becomes small

**inline-block:**

* height and width can be set in additional
* by default when width and height are not set means it is also considered as an inline element only
* even when the width is set means it does not go in a new line only the width is increased here not like inline even after the content also the space can be determined.

**None:**

* This will hide from the HTML and completely remove the element

**Nav tag:**

* It is also a Container
* Navigation is present just to make the reader understand that it is the navigation page
* To mention the same style for many elements means u can use commas and mention them
* Hover is a tag in CSS that is been used to change the color when the mouse is been touched( it is written inside different parentheses only).
* A cursor tag is used to make it a pointer
* Text-align means it takes the entire box inside that and not only the text inside it.

**CSS positioning:**

* It properly controls the placement of elements on a webpage.

**Static:**

* It is the default position is static and it cannot be changed until we use margin and padding for them

**Relative:**

* It is done so that the image goes down but the next boxes don’t go down
* Top, left, right, down
* Only the text is moved not the other box moved
* z-index- cannot be applied when it is static and which has more index has the priority to be on up of another object

**Absolute:**

* When an absolute is like a random cow in the ground
* When left or right or top or down are given then it moves from the screen alone.
* The whole screen is the place of absolute move
* But for me, it should move only inside the box only

**Fixed:**

* The word gets fixed at that position itself and the whole screen is behind that word only so whenever you move your screen also that word travels with you.

**Sticky**:

* It is relative by default and when a pixel is given it is relative until that pixel comes then becomes fixed and travels with us completely to the entire screen.

**CSS FLEXBOX**

* A bunch of unordered elements are easily ordered and organized properly by using the flex cupboard.

Display block display flex

* The div gets changed to flex container and the items get changed to flex items and the direction is default row
* Column and row are present
* The items inside the flex are wrapped inside the div element only using the wrap tag by default it is no wrap

<style>

        div

        {

            background-color: greenyellow;

            width: 300px;

            padding: 10px;

            display:flex;

            flex-direction: row;

            flex-wrap: wrap;

        }

        h1

        {

            background-color: black;

            color: white;

            padding:5px;

            margin: 2px;

        }

</style>

<div>

        <h1>One</h1>

        <h1>Two</h1>

        <h1>Three</h1>

        <h1>Four</h1>

        <h1>Five</h1>

        <h1>Six</h1>

</div>

**JUSTIFY CONTENT**

* Mostly used
* Space-around is used
* Space-between is also used

**FLEX-GROW:**

* It is given inside the child and it makes it grow that.

**FLEX-BASIS:**

* Flex basis is used to set the initial width of the flex-items.
* Can be used when the screen is small or large.

**MAIN AXIS VS CROSS AXIS:**

* The main axis is the row if its flex-direction is also the row
* If the main axis is a row means its cross-axis is a column
* The main axis is a column if its flex-direction is a column
* If the main axis is a column, then its cross-axis is a row

* **Height:**

fit content means it takes the height of the content.

* **Justify-content: center;**

works with the main axis

* **Align-items: center;**

works with cross-axis

**MEDIA QUERY**

**RESPONSIVE WEBSITE:**

* It means that even when the size of the webpage is decreased for several times it is not in the webpage. which gets adjusted with the width of the screen.
* Media query is used to apply styles based on the characteristics of the screen size, device type, and other features.

<style>

h1{

color:red;

}

@media print{

h1{

color: blue;

}

}

</style>

<h1>HELLO</h1>

* orientation means the current size of the screen.
* landscape: laptops and computers.
* portrait: mobile phones.

<style>

h1{

color:red;

}

@media print

{

h1{

color: blue; }

}

@media print and (orientation:landscape){

h1{

color:yellow;

} }

</style>

<h1>HELLO</h1>

**needed to use this with a meta tag:**

<meta name="viewport" content="width=device-width, initial-scale=1.0">

width=device-width: the content is adjusted with size of the size of the screen

initial-scale=1.0: the initial zoom-in size is set up as 1

**screen-width there are two types:**

**max-width:**

max width of the screen so till that size that particular color is visible.

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Media Query Test</title>

<style>

h1 {

color: red;

}

@media screen and (max-width: 600px) {

h1 {

color: pink;

}

}

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

h1 {

color: yellow;

}

}

</style>

</head>

<body>

<h1>HELLO</h1>

</body>

</html>

**Min-width:**

* It is the same as that and we need to give the minimum width till which this color should be visible
* It is opposite to max-width because it comes from a minimum 0.

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Media Query Test</title>

    <style>

        div{

            background-color: red;

            height:100px;

        }

        @media screen and (min-width:600px)

        {

            div{

                background-color: blue;

            }

        }

        @media screen and (min-width:900px)

        {

            div{

                background-color: yellow;

            }

        }

    </style>

</head>

<body>

    <div>

    </div>

</body>

</html>

* The order is very important while writing the media query
* The order should be from high to low so that the middle one doesn’t cover the small one

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Media Query Test</title>

    <style>

        div{

            background-color: black;

            height:100px;

        }

        @media screen and (max-width:1000px)

        {

            div{

                background-color: yellow;

            }

        }

        @media screen and (max-width:900px)

        {

            div{

                background-color:blue;

            }

        }

        @media screen and (max-width:600px)

        {

            div{

                background-color: red;

            }

        }

    </style>

</head>

<body>

    <div>

    </div>

</body>

</html>

**Example**

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Media Query Test</title>

    <style>

        .o{

            background-color: black;

            height:100px;

            margin: 5px;

            width:100px;

            float:left;

        }

        @media screen and (max-width:600px)

        {

            .o{

                float: none;

            }

        }

    </style>

</head>

<body>

    <div>

    <div class="o"></div>

        <p>Lorem ipsum dolor, sit amet consectetur adipisicing elit. Saepe in molestias quod ex, porro odio quia ipsam numquam hic praesentium pariatur eveniet error voluptas distinctio, doloribus non magni dolor corrupti.</p>

    </div>

</body>

</html>

**Write a media query for displaying the content on a laptop and mobile:**

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<style>

    .container{

        display: flex;

        justify-content: space-around;

        flex-wrap: wrap;

    }

    .one{

        background-color: greenyellow;

        margin: 5px;

        height:100px;

        flex-basis: 20%;

    }

    @media screen and (max-width:600px)

    {

        .one{

            flex-basis:40%;

        }

    }

</style>

<div class="container">

    <div class="one"></div>

    <div class="one"></div>

    <div class="one"></div>

    <div class="one"></div>

</div>