**CSS Interview Questions:-**

1. **What CSS stands for?**

**Ans:** CSS stands for Cascading Style Sheets

1. **What is meant by CSS?**

Ans: CSS is used to give layouts, style a website/application. CSS describes how HTML elements are to be displayed on screen, paper, or in other media. It saves a lot of work. It can control the layout of multiple web pages all at once. External stylesheets are stored in CSS files.

1. Why we use CSS?

Ans: CSS is used to define styles for your web pages, including the design, layout and variations in display for different devices and screen sizes. A CSS rule consists of a selector and a declaration block.

CSS Example  
body {  
background-color: lightblue;  
}

h1 {  
color: white;  
text-align: center;  
}

p {  
font-family: verdana;  
font-size: 20px;  
}

1. **What is meant by Cascading?**

Ans: It cascades (replaces) the last and latest value of the same property.

1. **What is the syntax of link tag for external CSS & its use?**

Ans: <link rel=’stylesheet’ type=”text/css” href=”path of CSS”>

Here **rel** stands for Relation & **href** stands for Hyperlink reference are compulsory attributes.

1. **What are the types of CSS?**

Ans: Inline CSS, Internal CSS & External CSS

1. **What is the difference among Inline, Internal & External CSS?**

Ans: Order of priority =Inline (highest) > Internal (medium) > External (lowest)

1. **Difference between Internal & External CSS?**

Ans: It’s priority depends on the sequence as browser is reading everything line by line.

If Internal CSS (style tag) placed before link tag of external css, then External CSS is given highest priority than internal css. Because the browser is reading the code one by one line. So it cascades. Therefore, it totally depends on how you write.

1. **What are the different ways to apply CSS to HTML documents?**

Ans: **Different ways to apply CSS to the HTML document-**

* **Inline CSS**- It is used to apply a unique style for a single element.
* **Internal CSS**- An internal style sheet may be used if one single HTML page has a unique style.
* **External CSS**- With an external style sheet, you can change the look of an entire website by changing just one file!  
  Each HTML page must include a reference to the external style sheet file inside the element, inside the head section.

1. **Which gets highest (first) priority?**

Ans: Always Inline among 3 of them.

1. **Which types of CSS we use in practical?**

Ans: External CSS due to Readability,

1. **In which scenario Inline CSS used?**

Ans: used if one single HTML page has a unique style.

1. What is CSS selectors?

Ans: Selectors  
CSS selectors are used to “find” (or select) the HTML elements you want to style.

The CSS element Selector  
The element selector selects HTML elements based on the element name.

Example

Here, all

elements on the page will be center-aligned, with a red text color:

p {  
text-align: center;  
color: red;  
}

The CSS 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!

To select an element with a specific id, write a hash (#) character, followed by the id of the element.

Example  
The CSS rule below will be applied to the HTML element with id=“para1”:

#para1 {  
text-align: center;  
color: red;  
}  
Note: An id name cannot start with a number!

The CSS class Selector  
The class selector selects HTML elements with a specific class attribute.

To select elements with a specific class, write a period (.) character, followed by the class name.

Example  
In this example all HTML elements with class=“center” will be red and center-aligned:

.center {  
text-align: center;  
color: red;  
}

You can also specify that only specific HTML elements should be affected by a class.

Example:

In this example only

elements with class=“center” will be red and center-aligned:

p.center {  
text-align: center;  
color: red;  
}  
HTML elements can also refer to more than one class.

Example:

In this example the

element will be styled according to class=“center” and to class=“large”:

This paragraph refers to two classes.

Note: A class name cannot start with a number!

The universal selector (\*) selects all HTML elements on the page.

Example  
The CSS rule below will affect every HTML element on the page:

* {  
  text-align: center;  
  color: blue;  
  }  
  The CSS Grouping Selector  
  The grouping selector selects all the HTML elements with the same style definitions.

Look at the following CSS code (the h1, h2, and p elements have the same style definitions):

h1 {  
text-align: center;  
color: red;  
}

h2 {  
text-align: center;  
color: red;  
}

p {  
text-align: center;  
color: red;  
}  
It will be better to group the selectors, to minimize the code.

To group selectors, separate each selector with a comma.

Example  
In this example we have grouped the selectors from the code above:

h1, h2, p {  
text-align: center;  
color: red;  
}

How To Add CSS:

When a browser reads a style sheet, it will format the HTML document according to the information in the style sheet.

Three Ways to Insert CSS  
There are three ways of inserting a style sheet:

External CSS  
Internal CSS  
Inline CSS  
External CSS

With an external style sheet, you can change the look of an entire website by changing just one file!

Each HTML page must include a reference to the external style sheet file inside the element, inside the head section.

Example

External styles are defined within the element, inside the section of an HTML page:

# This is a heading

This is a paragraph.

An external style sheet can be written in any text editor, and must be saved with a .css extension.

The external .css file should not contain any HTML tags.

Here is how the “mystyle.css” file looks:

“mystyle.css”  
body {  
background-color: lightblue;  
}

h1 {  
color: navy;  
margin-left: 20px; //Negative margin tells that you want negative spacing for your element. Meaning, Example :- The way positive margin-top pushes content down, Negative top margin pulls content up.

}  
Note: Do not add a space between the property value (20) and the unit (px):  
Incorrect (space): margin-left: 20 px;  
Correct (no space): margin-left: 20px;

Internal CSS

An internal style sheet may be used if one single HTML page has a unique style.

The internal style is defined inside the element, inside the head section.

Example

Internal styles are defined within the element, inside the section of an HTML page:

body { background-color: linen; }

h1 {  
color: maroon;  
margin-left: 40px;  
}

# This is a heading

This is a paragraph.

Inline CSS  
An inline style may be used to apply a unique style for a single element.

To use inline styles, add the style attribute to the relevant element. The style attribute can contain any CSS property.

Example  
Inline styles are defined within the “style” attribute of the relevant element:

# This is a heading

This is a paragraph.

Multiple Style Sheets

If some properties have been defined for the same selector (element) in different style sheets, the value from the last read style sheet will be used.

Assume that an external style sheet has the following style for the

# element:

h1 {  
color: navy;  
}  
Then, assume that an internal style sheet also has the following style for the

# element:

h1 {  
color: orange;  
}

Example  
If the internal style is defined after the link to the external style sheet, the

# elements will be “orange”:

h1 { color: orange; }

Example  
However, if the internal style is defined before the link to the external style sheet, the

# elements will be “navy”:

h1 { color: orange; }

Cascading Order

What style will be used when there is more than one style specified for an HTML element?

All the styles in a page will “cascade” into a new “virtual” style sheet by the following rules, where number one has the highest priority:

Inline style (inside an HTML element)  
External and internal style sheets (in the head section)  
Browser default  
So, an inline style has the highest priority, and will override external and internal styles and browser defaults.

CSS Comments  
CSS comments are not displayed in the browser, but they can help document your source code.

CSS Comments  
Comments are used to explain the code, and may help when you edit the source code at a later date.

Comments are ignored by browsers.

A CSS comment is placed inside the element, and starts with /\* and ends with \*/:

Example  
/\* This is a single-line comment \*/  
p {  
color: red;  
}  
You can add comments wherever you want in the code:

Example  
p {  
color: red; /\* Set text color to red \*/  
}  
Comments can also span multiple lines:

Example  
/\* This is  
a multi-line  
comment \*/

p {  
color: red;  
}

HTML and CSS Comments

In the following example, we use a combination of HTML and CSS comments:

Example

p { color: red; /\* Set text color to red \*/ }

## My Heading

Hello World!

This paragraph is styled with CSS.

CSS comments are not shown in the output.

CSS Colors  
Colors are specified using predefined color names, or RGB values.

CSS Background Color

You can set the background color for HTML elements:

Hello World

Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diam nonummy nibh euismod tincidunt ut laoreet dolore magna aliquam erat volutpat. Ut wisi enim ad minim veniam, quis nostrud exerci tation ullamcorper suscipit lobortis nisl ut aliquip ex ea commodo consequat.

Example

# Hello World

Lorem ipsum...

CSS Text Color  
You can set the color of text:

Hello World  
Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diam nonummy nibh euismod tincidunt ut laoreet dolore magna aliquam erat volutpat.

Ut wisi enim ad minim veniam, quis nostrud exerci tation ullamcorper suscipit lobortis nisl ut aliquip ex ea commodo consequat.

Example

# Hello World

Lorem ipsum...

Ut wisi enim...

CSS Border Color  
You can set the color of borders:

Hello World  
Hello World  
Hello World

Example

# Hello World

# Hello World

# Hello World

CSS Color Values

In CSS, colors can also be specified using RGB values, HEX values, HSL values, RGBA values, and HSLA values:

Same as color name “Tomato”:

rgb(255, 99, 71)  
#ff6347

Example

# ...

# ...

The CSS background properties are used to add background effects for elements.

CSS background-color  
The background-color property specifies the background color of an element.

Example  
The background color of a page is set like this:

body {  
background-color: lightblue;  
}  
With CSS, a color is most often specified by:

a valid color name - like “red”  
a HEX value - like “#ff0000”  
an RGB value - like “rgb(255,0,0)”  
Look at CSS Color Values for a complete list of possible color values.

Other Elements  
You can set the background color for any HTML elements:

Example  
Here, the, and elements will have different background colors:

h1 {  
background-color: green;  
}

div {  
background-color: lightblue;  
}

p {  
background-color: yellow;  
}

1. **What are the types of CSS Selectors?**

Ans: CSS Selectors: id(#) and class(.)

Order of priority: id(highest) > class (medium) > element selector(lowest)

Eg: While giving background-color for element: Red; class: Green; id: Magenta

Between element & class, the element’s bg-color overriddens by class’s bg-color.

But among three of them, both bg-color overrides by id.

But when id clashes with inline, then inline wins according to Specificity Hierarchy.

Through the above example, we can see that id selector has the highest priority, while element selector has the lowest. Priority is fine, again why ID should be unique? Because id has a specificity score of 100, the class has 10, and the element has 1. So, if two elements has the same id, it would be nearly impossible to make them have a different style. The only way to do make them have different styles is by adding inline styles which has a specificity score of 1000(highest), which is mostly not recommended.

**CSS Specificity**: Think of specificity as a score/rank that determines which style declaration is ultimately applied to an element.

**Specificity Rules:**

* The !important rule in CSS is used to add more importance to a property/value than normal.

In fact, if you use the !important rule, it will override ALL previous styling rules for that specific property on that element!

* **Equal specificity: the latest rule wins**- If the same rule is written twice into the external style sheet, then the latest rule wins:

#### Eg: #myid {   background-color: blue; } .myclass {   background-color: gray; } p {   background-color: red !important; }

#### In result, it will display p element selector’s bg-color.

#### (5) Types of CSS Selectors

#### Universal selector - 0 mark provided which is \* { } & inherited values are ignored.

* **Inline styles** - Example: <h1 style="color: pink;">
* Element Selector- The element selector (TAGs) selects HTML elements based on the element name.

**Elements and pseudo-elements** - Example: h1, ::before

* ID Selector- The id selector (attributes) 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!  
  Can be selected using the “#” symbol. **IDs** - Example: #navbar
* Class Selector- The class selector (attributes) selects HTML elements with a specific class attribute.  
  To select elements with a specific class, write a period (.) character, followed by the class name.

**Classes, pseudo-classes (refer pg 74 in ppt), attribute selectors** - Example: .test, :hover, [href]

:after, :before are the pseudo elements which we don’t need to write in html but they behave like one

1. **What is the Syntax of CSS?**

Ans: Syntax:  
h1 {  
color:black; font-size: 10px;  
}  
Here:  
h1 is Selector.  
color:black and font-size are declaration.  
color is property and black is value in the declaration.  
font is property and size is value in the delcaration.

The selector points to the HTML element you want to style.  
The declaration block contains one or more declarations separated by semicolons.  
Each declaration includes a CSS property name and a value, separated by a colon.  
Multiple CSS declarations are separated with semicolons, and declaration blocks are surrounded by curly braces.

Example  
In this example all

elements will be center-aligned, with a red text color:

p {  
color: red;  
text-align: center;  
}

Example Explained  
p is a selector in CSS (it points to the HTML element you want to style:

).  
color is a property, and red is the property value  
text-align is a property, and center is the property value.

In inline, style= “Property:Value/Text;” pairs.

In internal, <style>

tagname {

property\_name:value/text;

}

.className{

propertyName:value;

}

#idName{

propertyName:value;

}

</style>

1. **What is the use of class attributes?**

Ans: The HTML class attribute is used to specify a class for an HTML element.

The class attribute is often used to point to a class name in a style sheet. It can also be used by a JavaScript to access and manipulate elements with the specific class name.

1. **What is the use of ID?**

Ans: The HTML id attribute is used to specify a unique id for an HTML element.  
This specifies a unique id for an HTML element. The value of the id attribute must be unique within the HTML document.  
The id attribute is used to point to a specific style declaration in a style sheet. It is also used by JavaScript to access and manipulate the element with the specific id.  
The syntax for id is: write a hash character (#), followed by an id name. Then, define the CSS properties within curly braces {}.

**Note:** The id name is case sensitive!  
**Note:** The id name must contain at least one character, cannot start with a number, and must not contain whitespaces (spaces, tabs, etc.).

#### Why it is recommended to have a unique ID for each element while the class can be the same for many elements?

#### Ans:

1. **Difference between Class and Id?**

Ans: A class name can be used by multiple HTML elements, while an id name must only be used by only one HTML element within the page.

1. Difference between display:inline, display:block & display: inline-block?

Ans:

|  |  |  |
| --- | --- | --- |
| **Inline** | **Block-level elements** | **Inline-block** |
| Width & height will not work here. | They works. | They works. |
| The elements/content takes only the required amount of space as it takes only content’s width. | The element takes 100% width of the device. |  |
| It allows other inline elements to come in the same row. | It does not allows other block level elements to occupy it’s same row. |  |
| By default, no padding is added but margin is added. | By default, padding & margin are added. | Top & bottom padding & margin are respected |
| * You **cannot** set the height and width of inline-level elements in CSS | * You **can** set the height and width of block-level elements in CSS |  |
| * Inline elements flow **left to right**, meaning inline elements appear on the same line unless the line wraps to the next line or there’s an explicit line break (<br/>) to force a line break, | * Block-level elements flow **top to bottom**, meaning all block-level elements appear on their own line without an explicit line break (<br/>) |  |
| If there is no content and no border to your inline elements, nothing will be shown, even if you set background-color, margin & a height and width. That is because height and width are not settable CSS properties for inline elements. | p is block element, so height and width in same CSS are respected. |  |
| Tags:  <img>, <i>, <b>, <span>, <button>,  <a>, q, strong, select, script, small, big, sub, sup, textarea, var, time, tt, object, output, samp, map,  <input> & <label> are inline elements. | Tags:  Div, p, li, ul, ol, h1-h6, header, hr, footer, article, aside, nav, main, form, table, section, fieldset, tfoot, video, address, blockquote, canvas, dd, dl, dt, figcaption, figure, noscript, pre. |  |

## ****Note:**** Setting the display property of an element only changes ****how the element is displayed****, NOT what kind of element it is. So, an inline element with display: block; is not allowed to have other block elements inside it.

## Explanation of display: block, inline, inline-block & flex.

1. Learned more about display: inline; and display:block; as a quick revision for flex box. There we have seen a problem.  
   There we have seen a behaviour of CSS-  
   For the below code-

<**style**>

.child {

background-color: red;

height: 100px;

width: 100px;

display: block;

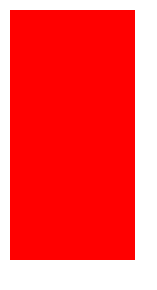
}</**style**><**div** id="parent">

<**div** class="child">

</**div**>

<**div** class="child">

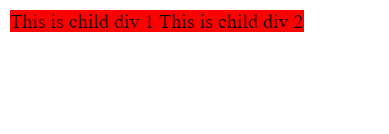
</**div**></**div**>

Output was-  
  
2 collided divs with each of height: 100px; width:100px;  
**Then we changed the display of child div from block to inline-**  
Output was-  
  
We didn’t got any output at all. Both red boxes are gone . Why this is happening, why our browser is not able to output anything after seting display to display: inline; ?

**Answer is-**  
CSS has a restriction. For any inline element, we can’t give it width and height because if we can give the height and width to the inline element, it will violate one important property of inline element. The property of inline element says it will take the height and width of the content.  
Reference- [Inline Properties](https://web.stanford.edu/class/archive/cs/cs193x/cs193x.1176/lectures/05/block-inline#height-and-width-of-inline-elements:~:text=If there is no content and no border to your inline elements, nothing will be shown, even if you set a height and width. That is because height and width are not settable CSS properties for inline elements.)

https://web.stanford.edu/class/archive/cs/cs193x/cs193x.1176/lectures/05/block-inline#height-and-width-of-inline-elements:~:text=If%20there%20is%20no%20content%20and%20no%20border%20to%20your%20inline%20elements%2C%20nothing%20will%20be%20shown%2C%20even%20if%20you%20set%20a%20height%20and%20width.%20That%20is%20because%20height%20and%20width%20are%20not%20settable%20CSS%20properties%20for%20inline%20elements.

Since both of our child divs don’t have any content, hence it is looking like nothing is in output. But, let’s say if we give some content to the child div. Let’s see the output-



Code used for above output-

<**style**>

.child {

background-color: red;

height: 100px;

width: 100px;

display: inline;

}</**style**><**div** id="parent">

<**div** class="child">

This is child div 1

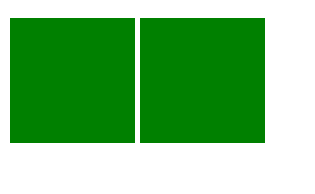
</**div**>

<**div** class="child">

This is child div 2

</**div**></**div**>

1. For the above case, we can also use display: inline-block to get the below results-

  
Above property is used to make inline element to set height and width.

3 . Then we have learned about flexbox and it’s properties-

* The flex-direction property defines in which direction the container wants to stack the flex items.
* The flex-wrap property specifies whether the flex items should wrap or not.
* The justify-content property is used to align the flex items.
* The align-items property is used to align the flex items.
* The align-content property is used to align the flex lines.

1. Then we have created the navbar by using flexbox and other CSS properties

Codepen link- [https://codepen.io/rhythm\_varshney/full/zYmwLoo 3](https://codepen.io/rhythm_varshney/full/zYmwLoo)

1. ‘+’ & :checked indicates what in selectors?

Ans: ‘+’ indicates next element/~~along with~~

for eg: input:checked+.toggle{

background-color:black;

}

:checked indicates background color changes only when checkbox checked

Note- <https://www.youtube.com/watch?v=OzpClSj9uQo&list=PLz2TLYnYStc1i5MiYwe0-naGiXCMVhcSm>

* To do animation like transitions smoothly moves in CSS, we need **same** property.

Eg: left: 5px; in child then left:calc(100%-35px); in input:checked+.toggle .knob

transition: 4s ease-in-out;

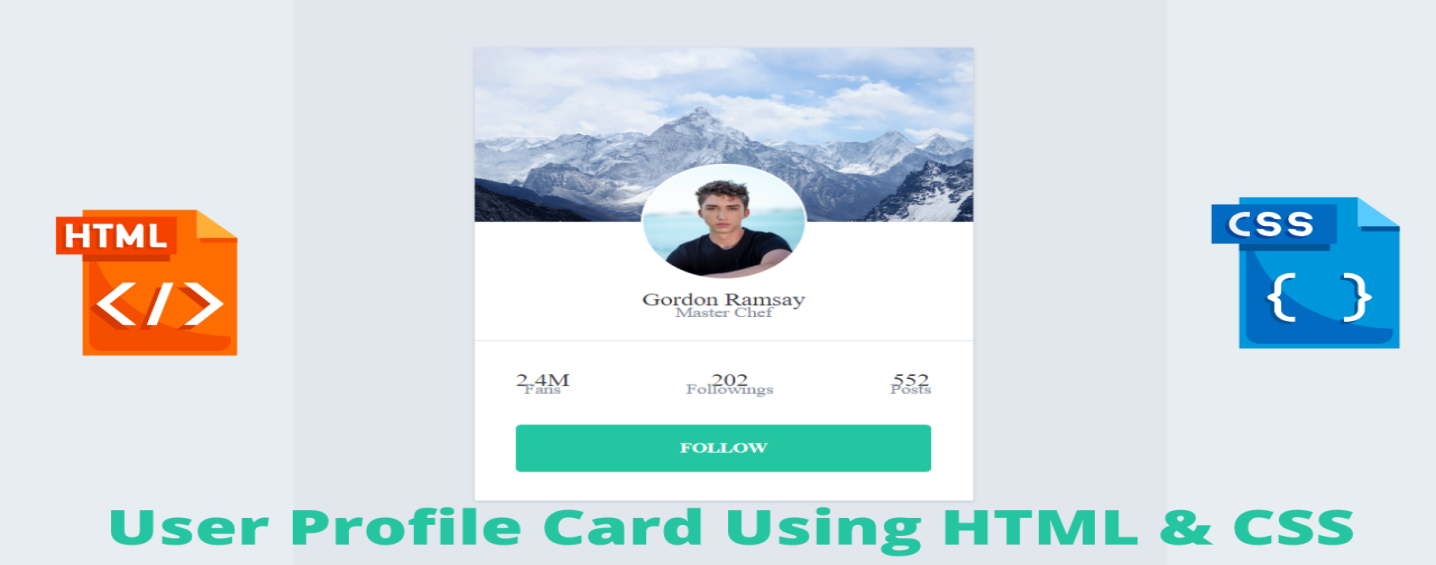
* Position: relative is for parent & position: absolute is for child
* To hide checkbox from user, we use below code for ref

input[type=”checkbox”]{

display:none;

}

* When height & width:30px; are of same height makes square that time we use border-radius:50%; in order to make circle for knob.
* Transition property works only when opacity:0; instead of display:none; because transition property will not work in display properties.
* Content property is used for text to be view by user.
* Display:none; is equal to opacity:0; & display:block; is equal to opacity:1;
* Transform: translateX(-50%); Here X denotes horizontal axis’s center position
* We can’t use transform property two times in same selector as it cascades.
* So use transform:translate(-50%, -50%); this brings as middle of center position like profile pic below img



* Images are inline elements.
* Margin: 0 15px; here 0 indicates top & bottom | 15px indicates right & left
* To make a block element like div in center, margin:0 auto; property can be used.

22)

**2] CSS Selectors**

CSS selectors are used to “find” (or select) the HTML elements you want to style. We can select elements based on name, id, class.  
**3] Border :** we learnt about border, How to add border in CSS and play with some examples, and playing with Hover effect. border ex. border: 1px solid #fff;  
4] **Margin :** margin property are use to create extra space around the content or element. We having properties to setting the margin for each side of an element ( top, right, bottom, and left).

margin-top

margin-right

margin-bottom

margin-left

**5]Padding :**

padding is used to create space around an element’s content inside of any defined borders. We having properties to setting the padding for each side of an element ( top, right, bottom, and left).

padding-top

padding-right

padding-bottom

padding-left

1. CSS Border:

Ans: CSS Borders  
The CSS border properties allow you to specify the style, width, and color of an element’s border.

I have borders on all sides.

I have a red bottom border.

I have rounded borders.

I have a blue left border.

CSS Border Style  
The border-style property specifies what kind of border to display.

The following values are allowed:

dotted - Defines a dotted border  
dashed - Defines a dashed border  
solid - Defines a solid border  
double - Defines a double border  
groove - Defines a 3D grooved border. The effect depends on the border-color value  
ridge - Defines a 3D ridged border. The effect depends on the border-color value  
inset - Defines a 3D inset border. The effect depends on the border-color value  
outset - Defines a 3D outset border. The effect depends on the border-color value  
none - Defines no border  
hidden - Defines a hidden border  
The border-style property can have from one to four values (for the top border, right border, bottom border, and the left border).

Example  
Demonstration of the different border styles:

p.dotted {border-style: dotted;}  
p.dashed {border-style: dashed;}  
p.solid {border-style: solid;}  
p.double {border-style: double;}  
p.groove {border-style: groove;}  
p.ridge {border-style: ridge;}  
p.inset {border-style: inset;}  
p.outset {border-style: outset;}  
p.none {border-style: none;}  
p.hidden {border-style: hidden;}  
p.mix {border-style: dotted dashed solid double;}

CODE:

p.dotted {border-style: dotted;} p.dashed {border-style: dashed;} p.solid {border-style: solid;} p.double {border-style: double;} p.groove {border-style: groove;} p.ridge {border-style: ridge;} p.inset {border-style: inset;} p.outset {border-style: outset;} p.none {border-style: none;} p.hidden {border-style: hidden;} p.mix {border-style: dotted dashed solid double;}

## The border-style Property

This property specifies what kind of border to display:

A dotted border.

A dashed border.

A solid border.

A double border.

A groove border.

A ridge border.

An inset border.

An outset border.

No border.

A hidden border.

A mixed border.

Note: None of the OTHER CSS border properties (which you will learn more about in the next chapters) will have ANY effect unless the border-style property is set!

CSS Border Width  
The border-width property specifies the width of the four borders.

The width can be set as a specific size (in px, pt, cm, em, etc) or by using one of the three pre-defined values: thin, medium, or thick:

Example  
Demonstration of the different border widths:

p.one {  
border-style: solid;  
border-width: 5px;  
}

p.two {  
border-style: solid;  
border-width: medium;  
}

p.three {  
border-style: dotted;  
border-width: 2px;  
}

p.four {  
border-style: dotted;  
border-width: thick;  
}

CODE:

p.one { border-style: solid; border-width: 5px; }

p.two {  
border-style: solid;  
border-width: medium;  
}

p.three {  
border-style: dotted;  
border-width: 2px;  
}

p.four {  
border-style: dotted;  
border-width: thick;  
}

p.five {  
border-style: double;  
border-width: 15px;  
}

p.six {  
border-style: double;  
border-width: thick;  
}

## The border-width Property

This property specifies the width of the four borders:(for the top border, right border, bottom border, and the left border):

Some text.

Some text.

Some text.

Some text.

Some text.

Some text.

**Note:** The "border-width" property does not work if it is used alone. Always specify the "border-style" property to set the borders first.

Specific Side Widths  
The border-width property can have from one to four values (for the top border, right border, bottom border, and the left border):

Example  
p.one {  
border-style: solid;  
border-width: 5px 20px; /\* 5px top and bottom, 20px on the sides \*/  
}

p.two {  
border-style: solid;  
border-width: 20px 5px; /\* 20px top and bottom, 5px on the sides \*/  
}

p.three {  
border-style: solid;  
border-width: 25px 10px 4px 35px; /\* 25px top, 10px right, 4px bottom and 35px left \*/  
}

CODE:

p.one { border-style: solid; border-width: 5px 20px; /\* 5px top and bottom, 20px on the sides \*/ }

p.two {  
border-style: solid;  
border-width: 20px 5px; /\* 20px top and bottom, 5px on the sides \*/  
}

p.three {  
border-style: solid;  
border-width: 25px 10px 4px 35px; /\* 25px top, 10px right, 4px bottom and 35px left \*/  
}

CSS Border Color  
The border-color property is used to set the color of the four borders.The color can be set by:

name - specify a color name, like “red”  
HEX - specify a HEX value, like “#ff0000”  
RGB - specify a RGB value, like “rgb(255,0,0)”  
HSL - specify a HSL value, like “hsl(0, 100%, 50%)”  
transparent  
Note: If border-color is not set, it inherits the color of the element.

Example: Demonstration of the different border colors:

p.one {  
border-style: solid;  
border-color: red;  
}

p.two {  
border-style: solid;  
border-color: green;  
}

p.three {  
border-style: dotted;  
border-color: blue;  
}

CODE:

p.one { border-style: solid; border-color: red; }

p.two {  
border-style: solid;  
border-color: green;  
}

p.three {  
border-style: dotted;  
border-color: blue;  
}

## The border-color Property

This property specifies the color of the four borders:

A solid red border

A solid green border

A dotted blue border

**Note:** The "border-color" property does not work if it is used alone. Use the "border-style" property to set the borders first.

Specific Side Colors

The border-color property can have from one to four values (for the top border, right border, bottom border, and the left border).

Example

p.one {  
border-style: solid;  
border-color: red green blue yellow; /\* red top, green right, blue bottom and yellow left \*/  
}

CSS Margins:  
The CSS margin properties are used to create space around elements, outside of any defined borders.

With CSS, you have full control over the margins. There are properties for setting the margin for each side of an element (top, right, bottom, and left).

Margin - Individual Sides

CSS has properties for specifying the margin for each side of an element:

margin-top  
margin-right  
margin-bottom  
margin-left  
All the margin properties can have the following values:

auto - the browser calculates the margin  
length - specifies a margin in px, pt, cm, etc.  
% - specifies a margin in % of the width of the containing element  
inherit - specifies that the margin should be inherited from the parent element  
Tip: Negative values are allowed.

Example

Set different margins for all four sides of a

element:

p {  
margin-top: 100px;  
margin-bottom: 100px;  
margin-right: 150px;  
margin-left: 80px;  
}

CODE:

div { border: 1px solid black; margin-top: 100px; margin-bottom: 100px; margin-right: 150px; margin-left: 80px; background-color: lightblue; }

## Using individual margin properties

This div element has a top margin of 100px, a right margin of 150px, a bottom margin of 100px, and a left margin of 80px.

Margin - Shorthand Property  
To shorten the code, it is possible to specify all the margin properties in one property.

The margin property is a shorthand property for the following individual margin properties:

margin-top  
margin-right  
margin-bottom  
margin-left  
So, here is how it works:

If the margin property has four values:

margin: 25px 50px 75px 100px;  
top margin is 25px  
right margin is 50px  
bottom margin is 75px  
left margin is 100px  
Example  
Use the margin shorthand property with four values:

p {  
margin: 25px 50px 75px 100px;  
}  
CODE:

div { border: 1px solid black; margin: 25px 50px 75px 100px; background-color: lightblue; }

## The margin shorthand property - 4 values

This div element has a top margin of 25px, a right margin of 50px, a bottom margin of 75px, and a left margin of 100px.

If the margin property has three values:  
margin: 25px 50px 75px;  
top margin is 25px  
right and left margins are 50px  
bottom margin is 75px

Example  
Use the margin shorthand property with three values:

p {  
margin: 25px 50px 75px;  
}  
CODE:

div { border: 1px solid black; margin: 25px 50px 75px; background-color: lightblue; }

## The margin shorthand property - 3 values

This div element has a top margin of 25px, a right and left margin of 50px, and a bottom margin of 75px.

If the margin property has two values:

margin: 25px 50px;  
top and bottom margins are 25px  
right and left margins are 50px

Example  
Use the margin shorthand property with two values:

p {  
margin: 25px 50px;  
}

CODE:

div { border: 1px solid black; margin: 25px 50px; background-color: lightblue; }

## The margin shorthand property - 2 values

This div element has a top and bottom margin of 25px, and a right and left margin of 50px.

If the margin property has one value:

margin: 25px;  
all four margins are 25px  
Example  
Use the margin shorthand property with one value:

p {  
margin: 25px;  
}

CODE:

div { border: 1px solid black; margin: 25px; background-color: lightblue; }

## The margin shorthand property - 1 value

This div element has a top, bottom, left, and right margin of 25px.

The auto Value  
You can set the margin property to auto to horizontally center the element within its container.

The element will then take up the specified width, and the remaining space will be split equally between the left and right margins.

Example  
Use margin: auto:

div {  
width: 300px;  
margin: auto;  
border: 1px solid red;  
}

CODE:

div { width: 300px; margin: auto; border: 1px solid red; }

## Use of margin: auto

You can set the margin property to auto to horizontally center the element within its container. The element will then take up the specified width, and the remaining space will be split equally between the left and right margins:

This div will be horizontally centered because it has margin: auto;

The inherit Value  
This example lets the left margin of the

element be inherited from the parent element (

):

Example

Use of the inherit value:

div {  
border: 1px solid red;  
margin-left: 100px;  
}

p.ex1 {  
margin-left: inherit;  
}

CODE:

div { border: 1px solid red; margin-left: 100px; }

p.ex1 {  
margin-left: inherit;  
}

## Use of the inherit value

Let the left margin be inherited from the parent element:

This paragraph has an inherited left margin (from the div element).

All CSS Margin Properties

margin :A shorthand property for setting all the margin properties in one declaration  
margin-bottom:Sets the bottom margin of an element  
margin-left:Sets the left margin of an element  
margin-right:Sets the right margin of an element  
margin-top:Sets the top margin of an element

CSS Padding  
Padding is used to create space around an element’s content, inside of any defined borders.  
The CSS padding properties are used to generate space around an element’s content, inside of any defined borders. With CSS, you have full control over the padding. There are properties for setting the padding for each side of an element (top, right, bottom, and left).

Padding - Individual Sides  
CSS has properties for specifying the padding for each side of an element:

padding-top  
padding-right  
padding-bottom  
padding-left  
All the padding properties can have the following values:

length - specifies a padding in px, pt, cm, etc.  
% - specifies a padding in % of the width of the containing element  
inherit - specifies that the padding should be inherited from the parent element  
Note: Negative values are not allowed.

Example  
Set different padding for all four sides of a

element:

div {  
padding-top: 50px;  
padding-right: 30px;  
padding-bottom: 50px;  
padding-left: 80px;  
}  
CODE:

div { border: 1px solid black; background-color: lightblue; padding-top: 50px; padding-right: 30px; padding-bottom: 50px; padding-left: 80px; }

## Using individual padding properties

This div element has a top padding of 50px, a right padding of 30px, a bottom padding of 50px, and a left padding of 80px.

CSS Box Model  
All HTML elements can be considered as boxes.

The CSS Box Model  
In CSS, the term “box model” is used when talking about design and layout.

The CSS box model is essentially a box that wraps around every HTML element. It consists of: margins, borders, padding, and the actual content. The image below illustrates the box model:

Explanation of the different parts:

Content - The content of the box, where text and images appear  
Padding - Clears an area around the content. The padding is transparent  
Border - A border that goes around the padding and content  
Margin - Clears an area outside the border. The margin is transparent  
The box model allows us to add a border around elements, and to define space between elements.

Example

Demonstration of the box model:

div {  
width: 300px;  
border: 15px solid green;  
padding: 50px;  
margin: 20px;  
}

CODE:

div { background-color: lightgrey; width: 300px; border: 15px solid green; padding: 50px; margin: 20px; }

## Demonstrating the Box Model: CPBM

The CSS box model is essentially a box that wraps around every HTML element. It consists of: borders, padding, margins, and the actual content.

This text is the content of the box. We have added a 50px padding, 20px margin and a 15px green border. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

1. Pseudo Classes:
2. Used to define a special state of an elements.
3. Actually : is pseudo class which provides us to give access to different states of the element.

Types of Pseudo Classes:

1. Link (4), Input (12) & General(15).
2. Link: hover | link | visited | active
3. Input: checked | optional | required | disabled | enabled | valid | invalid | in-range | out-of- range | read-only |read-write | focus |
4. General: first-child | last-child | nth-child(n) | nth-last-child(n) | only-child | first-of-type | last-of-type | nth-of-type | nth-last-of-type(n) | only-of-type |empty | target | root | not(selector) | lang(language)

Pseudo Elements:

It is used to style a specific part of an HTML element.

Types:

::first-line This is used to add styles to the first line of some text. This text could be paragraphs (<p>), headings (<h1> - <h6>) etc. This pseudo element only works with block level elements.

::first-letter This is used to add styles to the first letter of some text. This text could be paragraphs (<p>), headings (<h1> - <h6>) etc. This pseudo element also works only with block level elements.

::selection It is used to add styles to the selected text. In another words, it is used to style the text highlight when we select them.

::before & ::after In this, ::before is used to add some content before the content HTML element and ::after is used to add some content after the content HTML element. This content could be Text, Image or Blank within single/double quotes.

1. In CSS, the “auto” value can indeed be used for margin, but not for padding.  
   When the “auto” value is applied to a margin property, it allows the browser to automatically calculate and distribute the margin space for the specified element. This can be useful in centering elements or achieving specific layout requirements.

For example

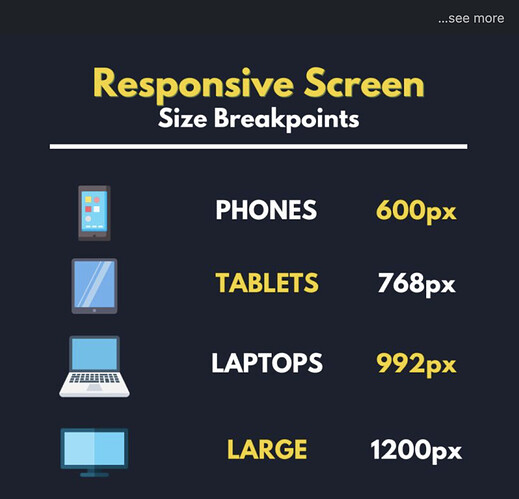
.element {

margin: auto;

}

Here, the “auto” value for margin will horizontally center the element within its containing parent element. The browser will calculate and distribute equal margins on the left and right sides to achieve center alignment.

1. **Responsive Design:** API stands for Application Programming Interface. In the context of APIs, the word Application refers to any software with a distinct function. Interface can be thought of as a contract of service between two applications. This contract defines how the two communicate with each other using requests and responses. Their API documentation contains information on how developers are to structure those requests and responses.



**Grid:** The CSS Grid Layout Module offers a grid-based layout system, with rows and columns, making it easier to design web pages without having to use floats and positioning.

An HTML element becomes a grid container when its display property is set to grid or inline-grid

You can adjust the gap size by using one of the following properties:

* column-gap
* row-gap
* gap  
  Click the link below for example:

<https://codepen.io/Athira231996/pen/abRaEea>

Pseudo Classes:

<https://codepen.io/Athira231996/pen/GRYGQBy>

Combinators:

<https://codepen.io/Athira231996/pen/zYmaRjx>

Parallax Effect:

<https://codepen.io/Athira231996/pen/BaqVrKv>

Overlay Effect:

<https://codepen.io/Athira231996/pen/dygKmyV>

CSS (Cascading Style Sheet)

Ways to add CSS: 3 Ways

1. Inline Style: By giving a style attribute to the HTML tag.

2. Internal Style: By creating a <style> </style> tag inside the head tag.

3. External Style: By adding a separate CSS file with the help of a link tag.

(The file extension must be .css)

Increasing Order of Priority : External (least) > Internal > Inline

Selectors: 3 types

1. Class Selector: (.) represented by a dot

2. ID Selector: (#)

3. Tag Selector: Tag name

Combinators => When you have to apply CSS in combination of selectors

1. Decendent Selector => space

2. Direct Child Selector => > (greater than symbol)

3. Adjacent Sibling Selector => + (plus symbol)

4. General Sibling Selector => ~ (tilde)

Comments in CSS: /\* YOUR COMMENTS HERE \*/ (Ctrl + /)

NOTE: CSS IS ALL ABOUT EXPERIENTS

Margins:

margin: value in px;

margin: top right bottom left; (SHORTHAND)

Borders:

border: width pattern color;

Border patterns: solid|dotted|dashed

Hover ? => When you put your mouse on an element.

Link : Created with the help of anchor tag

States of Link :

:link => Default link

:hover => When you keep your mouse on it

:active => When you are clicking it

:visited => When you have opened at least once

Display:

Basics:

none

block

inline

inline-block

Advanced:

flex

grid

Visibility:

visible

hidden

Q. What is the diff between display: none and visibility:hidden?

A. Visibility hidden occupies the space, however display none removes the space of the element

Position :

static: Default position of every element

relative: Move the item from TLBR from its current position.

Parent

absolute: Move it in the relation of browser TLBR. (Partial)

Child

fixed: Fix the item at any place in browser

sticky: Sets the position according to the parent element while scroll.

Shadow?

Types of shadow:

1. Text Shadow

/\* text-shadow: horizontal-shadow vertical-shadow blur color \*/

box-shadow: 5px 5px 5px grey;

2.Box Shadow

/\* box-shadow: horizontal-shadow vertical-shadow blur spread color \*/

box-shadow: 5px 5px 5px grey;

Combinators => When you have to apply CSS in combination of selectors

1. Decendent Selector => space

2. Direct Child Selector => > (greater than symbol)

3. Adjacent Sibling Selector => + (plus symbol)

4. General Sibling Selector => ~ (tilde)

Units of measuement in CSS ?

1. Pixel (px)

2. em

3. rem

4. vh, vw => viewport height, viewport width (Responsive)

1 em => 16px; default

1 rem => 16px; default

Visible area on a browser is called viewport

Gradient

1. Linear: Goes from one direction to another.

2. Radial: Goes from center to outside of the element.

/\* background-image: linear-gradient(to left, yellow, blue); \*/

background: radial-gradient(yellow, blue)

Flex Layout: Value in display property

justify-content: To align the items in the direction of flex with space distribution in various ways

flex-direction: row, column and their reverse orders

align-items: To change the alignment of items in the other direction of the flex. (Vertically)

align-self: TO CHANGE THE ALIGNMENT OF ONLY SPECIFIC CHILDREN IN A FLEX CONTAINER

flex-wrap: To push the items in the new line if they go out of the screen size

Background Image:

background: url(PATH OF THE IMAGE);

background-size:cover & contain

Overflow : When the child has more width and height than its parent, then it flows out of the parent.

Text Tags for creating cards:

<h1>....<h6>, <p>

/\* 1. Create the problem of overflow by giving white-space: nowrap

2. Solve the problem of overflow using overflow: hidden

3. Provide text-overflow: ellipsis to put a ... at the end of the line\*/

element{

white-space: nowrap;

overflow: hidden;

text-overflow: ellipsis;

}

Pseudo: Not real (Fake)

1. Pseudo Classes => (:), :link , :active, :hover, :visited,

2. Pseudo Elements => (::)

FLEXBOX LAYOUT WAS DESIGNED TO ONLY DEVELOP 1-DIMENSIONAL LAYOUTS.

RWD (Responsive Web Design): The style of designing web pages which are visible and have good look and feel on all different sized devices.

E.g Desktop, Laptop, Tablets, Mobiles etc

320px — 480px: Mobile devices.

481px — 768px: iPads, Tablets.

769px — 1024px: Small screens, laptops.

1025px — 1200px: Desktops, large screens.

1201px and more — Extra large screens, TV.

Break Points: Range in px for different screen sizes

Media Query: A way to write different CSS for different screen sizes on same web page.

Syntax: @media screen and (max-width: 480px) and (min-width: 320px) {

}

Mobile First Approach: When you give preference to mobile sizes first over desktop sizes.

Step 1: Use meta tag with viewport.

Step 2: Use % to set width in items rather than using px.

Step 3: Use media queries

**How many alphanumerical characters are in a hex color code?**

**Answer: 6 Eg:- #1s1s3d**