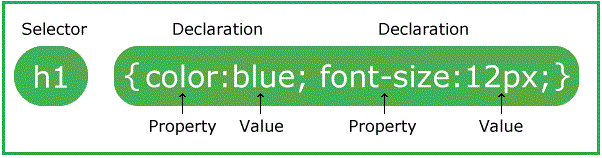
**Cascading Style Sheets (CSS)**

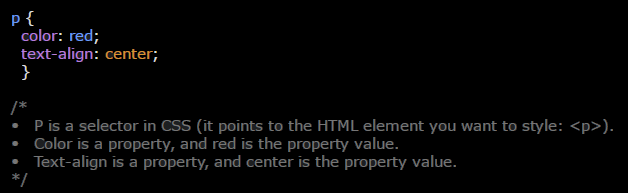
* CSS is a styling language that describes the style of an HTML document.
* Css was created by World Wide Web Consortium (W3C).
* Unlike HTML, Css is a case sensitive language.

**CSS Syntax:**

****

1. **Selector:** points to the HTML element you want to style.
2. The **declaration block:** contains one or more declarations separated by semicolons (;)
3. Each **declaration:** includes a CSS property name and a value, separated by a colon (:)
4. Multiple CSS declarations are separated with semicolons (;) and declaration blocks are surrounded by curly braces {}
5. All structure [selector + declaration block] called **“ruleset = rule”.**

**Example:**



**CSS Selectors (More html selectors in css mean more specification)**

CSS selectors are used to "find" (or select) the HTML elements you want to style. We can divide CSS selectors into five categories:

* Simple selectors (Used to select elements based on **name**, **id**, **class**, **Grouping**, and **Global selector**).
* [Combinator selectors](https://www.w3schools.com/css/css_combinators.asp) (select elements based on a specific relationship between them).
* [Pseudo-class selectors](https://www.w3schools.com/css/css_pseudo_classes.asp) (select elements based on a certain state).
* [Pseudo-elements selectors](https://www.w3schools.com/css/css_pseudo_elements.asp) (select and style a part of an element).
* [Attribute selectors](https://www.w3schools.com/css/css_attribute_selectors.asp) (select elements based on an attribute or attribute value).

**We will explain each type separately in details.**

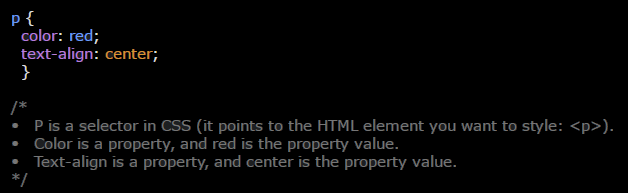
**Firstly; Simple selectors:**

Simple selectors are selectors that used to select element based on its **name**, **id**, **class**, **Grouping**, and **Global selector**). These selectors are:

1. **(Element Selector=tag selector=type selector)**

In this selector we mentioned the Html element name, to style it with css code.

**Example:**



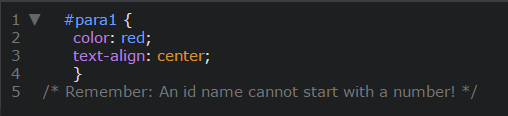
1. **Id selector:**

To use this selector we firstly add an **id name** to the html element. (Ex; <p id= “Para”>this is a paragraph</p>)

Then we write a hash (#) character, followed by the id of the element. (Ex; #para{Declaration block}) in css page.

The id name of an html element is a **unique** id name within a page, so the id selector is used to select one unique element! (For only one time)

**Example:**

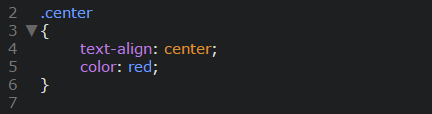
****

1. **Class selector:**

To use this selector we firstly add a **class name** to the html element. (Ex; <p class= “Para”>this is a paragraph</p>)

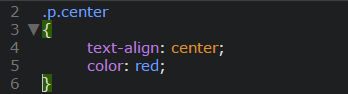
Then we write a period (.) character, followed by the class name in css page. (Ex; .para{Declaration block}) in css page.

The Class of an element **isn’t unique** within a page like id name, so the class selector can be used to select a multiply html elements!

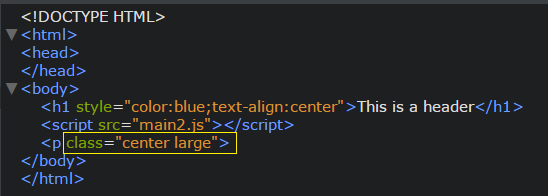


You can also specify that only specific HTML elements should be affected by a class.

HTML elements can also refer to more than one class.



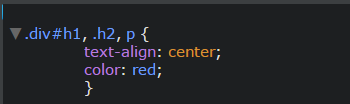
Html element can contain more than one class style.



**Note:** A class name cannot start with a number!

1. **Grouping Selector**

The grouping selector selects all the HTML elements with the same style definitions.



**Note:** It will be better to group the selectors, to minimize the code.

**Secondly; Combinator (Contextual) selector**

## Descendent selector (space).

The descendant selector Select all elements that are descendants of a specified element (parent node).

The following example selects all <p> elements inside <div> elements:  div p {background-color: yellow ;}

# Child selector (>).

The child selector selects all elements that are the direct children of a specified element.

The following example selects all <p> elements that are children of a <div> element: div > p {background-color: yellow ;}

# Adjacent sibling selector (+).

# The adjacent sibling selector selects only the first element placed immediately after a specified element.

# The following example selects only the first <p> element that is placed immediately after <div> elements: div + p {background-color: yellow ;}

# General sibling selector (~).

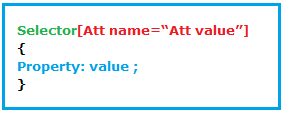
The general sibling selector selects all elements that placed after a specified element in the same parent.

The following example selects all <p> elements that are siblings of <div> elements:

**Attribute selector []**

Attribute Selector is used to select html elements based on the attribute name and attribute value.

Its syntax is:



**Note:**

1. Selector may be (id, class, Groping …). Any type
2. Selector and Att value are optional (if you want to give **less specification properties** to any html element have [Att name= “Att Value”] then you should write this syntax without selector, like that → [Att name= “Att value”]{ property: value ;}).
3. You can write the Att value with ‘single’ or “double” quotation marks.
4. If “Att value” hasn’t any specific characters like (Comma, colon, semicolon) then you can write “Att value” without quotation marks.
5. There is no space between the Selector and the square brackets [].
6. There is another options related with this selector:
   1. You can write only the **first** part of specific Att value (the first one letter or more); through this syntax. Selector[Att name^= “first part of the value”] the above note can be applied also in this syntax. Ex; [type^="t"]{ background-color: red; }. for html code→ <input type= “textarea”/>
   2. You can write only the **Last** part of specific Att value (the last one letter or more); through this syntax. Selector[Att name$= “Last part of the value”] the above note can be applied also in this syntax. Ex; [type$="ea"] {background-color: red ;}. for html code→ <input type= “textarea”/>
   3. You can write any part of a specific Att value (The first one letter or more, The last one letter or more, and any part inside this value); through this syntax.

Selector[Att name\*= “Any part in the value”] the above note can be applied also in this syntax.

Ex; [type\*="ea"]{ background-color: red;} for html code→ <input type= “textarea”/>

[type\*="x"]{ background-color: red ;} for html code→ <input type= “textarea”/>

[type\*="xta"]{ background-color: red ;} for html code→ <input type= “textarea”/>

* 1. To select any value that has a space list of values, and contains a word specified as the value in the attribute expression. Write this syntax.

Selector[Att name~= “Any part in the value”] the above note can be applied also in this syntax.

Ex; [title~="my"]{ background-color: red;} for html code→ <a href= “url” title= “Visit my page”>Click to visit my page</a>

* 1. You can write the next syntax when you found the Att value has the hyphen character (-) after it **directly :**

Selector[Att name|= “complete value before the hyphen ”] the above note cannot be applied in this syntax.

Ex; [lang|="en"]{ background-color: red;padding:30px;}→ for html code→ <a href= “url” lang= “en-us”>Press here</a>

[lang|="part"]{background-color: red;padding:30px;}→ for html code→ <a href= “url” lang= “part-us”>Press here </a>

**Note:** you should write the complete word before the hyphen character like in the two examples above; you shouldn’t write it like this because it doesn’t work:

[lang|="art"]{background-color: red;padding:30px;}→ for html code→ <a href= “url” lang= “part-us”>Press here </a>

**Pseudo-classes selector (User Interface)**

A pseudo-class is used to define a special state of an html element.

**selector:pseudo-class{property: value;}**

**these states are :**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Link state (Unvisited)** | | Used to decoration un-visited links (only links). | | **a:link{background-color:black;}** |
| **Visited state** | | Used to decoration visited links (only links). | | **a:link{background-color:black;}** |
| **Hover state** | | Used to decoration links (or any html element) when hovered on (on mouse over) it. | | **a:link{background-color:black;}** |
| **Active** | | Used to decoration links or any html element (when press on it to make it active). | | **a:link{background-color:black;}** |
| **Target** | | Used to decorate any html element when it targeted (Only when element id or name  Appears as a **targeted fragment identifier** in the end of the page URL).  We can notice it when the targeted element became in the top of the page. | | **p:target{background-color: #ccc; border-left: 15px solid red; padding: 20px 0px 20px 20px; }** |
| **Checked** | | It represents any radio (<input type="radio">), checkbox (<input type="checkbox">) or option (<option> in a <select>) element that is checked to an on state.  **Note:** we should first add checked value in html tag to make the element checked then add it to css tag to style it.  Note: we add css style in this case to the tag which description the checkbox, radio, or option tag | | **External css:**  **input:checked + span{background-color: #ccc;}**  **Html page :**  **<input id="ff" type="radio" name="gender" checked/>**  **<span for="ff">male</span>** |
| **Default** | |  | |  |
| **Dir(direction)** | |  | |  |
| **Enabled**  **(any editable tag by default)** | | It represents any enabled element. An element is enabled if it can be activated (selected, clicked on, typed into, etc.) or accepts focus. The element also has a disabled state, in which it can't be activated or accept focus.  Ex: Submit, bottom, text, and password fields……etc.  **Note:** radio and checkbox were not enabled elements. | | **input:enabled{ background-color: #ccc;}** |
| **Disabled**  **(Opposite to enabled)** | | It represents any disabled element. An element is disabled if it can’t be activated (selected, clicked on, typed into, etc.) or accepts focus. The element also has an enabled state, in which it can be activated or accept focus.  **Note:** we should first add disabled value in html tag to make the element disabled then add it to css tag to style it. | | **input:disabled{ background-color: #ccc;}** |
| **First-child** | | It represents the first element (Child) among a group of sibling elements (Childs) | | **P:first-child{background-color:red;}** |
| **First-of-type** | | It represents **the first element of its type** among a group of sibling elements.  **Note:** the parent has different siblings with different types and there is more than element (Ex; 5 elements) have the same type. Thin this pseudo will choose the first one (child) among their siblings (brothers) in this type; it isn’t necessary for this child if it’s not the first child in all different siblings of this parent.  Can be in any location within his parent | | **span:first-of-type{background-color:red;}**  **Html page:**  **<div>**  **<p>This is the only paragraph</p>**  **<span>This is a span</span> </div>** |
| **Last-child** | | It represents the last element (Child) among a group of sibling elements (Childs) | | **P:last-child{background-color:red;}** |
| **last-of-type** | | It represents **the last element of its type** among a group of sibling elements.  **Note:** the parent has different siblings with different types and there is more than element (Ex; 5 elements) have the same type. Thin this pseudo will choose the last one (child) among their siblings (brothers) in this type; it isn’t necessary for this child if it’s not the last child in all different siblings of this parent.  Can be in any location within his parent | | **p:last-of-type{background-color:red;}**  **Html page:**  **<div>**  **<p>This is the only paragraph</p>**  **<span>This is a span</span> </div>** |
| **only-child** | | It represents the only element (Child) of a specific parent. Or represents an element without any siblings. This is the same as :first-child :last-child or :nth-child(1) :nth-last-child(1), but with a lower specificity.  Note: the parent has only one child.  Example means that any paragraph found as a unique child in any parent give a red color to it. | | **P:only-child{ color: red;}**  **Html page:**  **<div>**  **<p>This is the only paragraph</p>**  **</div>** |
| **Only-of-type** | | It represents an element that has no siblings of the same type in a specific parent.  **Note:** the parent has **different Childs**. There is one child among them has a distinctive type and this child must be **the first child** among his siblings.  Example men any element has a unique type in a parent give it red color.  Note: the parent may contain another Childs but there is only one child with type paragraph. | | **P:only-of-type{ color: red;}**  **Html page:**  **<div>**  **<p>This is the only paragraph</p>**  **<span>This is a span</span> </div>** |
| **Focus** | | It represents an element (such as a form input) that has received autofocus. It is generally appears when the user clicks or taps on an element or selects it with the keyboard's "tab" key.  **Note:** This pseudo-class applies only to the **focused element itself**. | | **input:disabled{ background-color: #ccc;}**  **Html page :**  **<input id="ff" type="text" autofocus/>**  **<input type="textarea"/>** |
| **Focus-within** | | It represents an element (such as a form input) that has not received autofocus. It is generally appears when the user clicks or taps on an element or selects it with the keyboard's "tab" key. | | **input: focuswithin{ background-color: #ccc; width: 250px ;}** |
| **Fullscreen** | |  | |  |
| **valid** | | It represents any <input> or other <form> element likes: email or Url. Whose contents validate successfully. This allows to easily making valid fields adopt an appearance that helps the user confirm that their data is formatted properly.  **Note:** This pseudo-class is useful for highlighting correct (valid) fields for the user. | | **input[type="email"]:valid{border: 2px solid green;}** |
| **Invalid** | | It represents any <input> or other <form> element likes: email or Url. Whose contents validate. This allows to easily making invalid fields adopt an appearance that helps the user confirm that their data is formatted properly.  **Note:** this pseudo-class is useful for highlighting field errors for the user. | | **input[type="email"]:invalid{border: 2px solid red;}** |
| **In-range** | | It represents an <input> element whose current value is within the range limits specified by the min and max attributes. <https://developer.mozilla.org/en-US/docs/Web/CSS/:in-range> | | **input:in-range { background-color: rgba(0, 255, 0, 0.25);}** |
| **Out-of-range** | | It represents an <input> element whose current value isn’t within the range limits specified by the min and max attributes. | | **input:out-of-range { background-color: rgba(255, 0, 0, 0.25); border: 2px solid red; }.** |
| **Indeterminate** | |  | |  |
| **Matches(selectors)** | |  | |  |
| **Required** | | It represents any <input>, <select>, or <textarea> element that have the required attribute set on it. All html elements are optional by default. | | **input: required{ background-color: #ccc;}**  **Html page :**  **<input id="ff" type="text" required/>** |
| **Optional** | | It represents any <input>, <select>, or <textarea> element that does not have the required attribute set on it. All html elements are optional by default. | | **input: optional{ background-color: #ccc;}** |
| **Placholder-shown** | | It represents any <input> or <textarea> element that is currently displaying placeholder text.  **Note:** You should add placeholder to html element. | | **:placeholder-shown { border: 2px solid silver;}** |
| **Read-only** | | It represents an html element (such as input or textarea) that is not editable by the user. (if html element]en isn’t readonly by default such: input , you must add a readonly attribute to this element).  **Note:**  readonly elements by default like; paragraph  read-write elements (contenteditable) by default like; most inputs. | | **input:read-only { background-color: red;}**  **Html code:**  **<input type= “text” value= “readonly text” readonly/>** |
| **Read-write** | | It represents an element (such as input or textarea) that is editable by the user.  (If html element isn’t editable by default such: p ,you must add a contenteditable attribute to this element).  **Note:**  read-write (contenteditable) elements by default like; most inputs. | |  |
| **These tags name structural pseudo class elements** | | | | |
| **Root** | | It used to style the root element in a dom tree.In HTML, :root represents the <html> element only.  We can write it with 3 syntaxes: 1) html:root 2) :root 3) \*:root. | | **html:root{background-color: red;}**  **:root{background-color: red;}**  **\*:root{background-color: red;}** |
| **Empty** | | It represents any html element that has no children. Children can be either element nodes or text (including whitespace). Comments, processing instructions, and CSS content do not affect whether an element is considered empty. | | **Css code → P:empty{boerder: 2px solid red;}**  **Html code → <p></p>** |
| **nth pseudo classes** | | | | |
| Selector:nth-child(n) CSS pseudo-class matches elements based on their position in a group of siblings.Where; n can be [number, keyword, or a mathematical formula].  **When:**  n is a number it equal (0,1,2,3,4,5,6,… ,∞).  n is a Keyword it equal (even or odd).  n is a mathematical formula it equal (an+b)th.  **Where:**  n: (0,1,2,3,4,5,6,… ,∞).  a: integer(default is 1)  b: integer(default is 0)  Ex:  #div section p:nth-child(3n+2)  On putting n=0 then the product is 2.  On putting n=1 then the product is 5.  On putting n=2 then the product is 8.  On putting n=3 then the product is 11.  Then paragraphs No. 2,5,8,11,…∞ will be chosen to styled.  Ex:  P,span:nth-child(n)  Means that all spans in this parent will be chosen to be styled as in this case n=0,1,2,3,4,5,6,… ∞.  Note:  (2n+0)= (2n) for even numbers.  (2n+1)= for odd numbers. | | | | |
| **Nth-child(n)** | | It styles element based on its position in a group of siblings (Same or different types) in one parent. **THE COUNT BEGIN FROM TOP TO DOWN FOR ALL SIBLINGS IN THIS PARENT** | | **p:nth-child(4) { Background-color: red; }**  **Html page:**  **<div><p>P1</p><p>P2</p><p>P3</p><p>P4</p></div>** |
| **Nth-last-child(n)** | | It style element based on its position in a group of siblings (Same or different types) in one parent.  **THE COUNT BEGIN FROM Down TO TOP FOR ALL SIBLINGS IN THIS PARENT** | | **p:nth-last-child(1) { Background-color: red; }**  **Html page:**  **<div><p>P4</p><p>P3</p><p>P2</p><p>P1</p></div>** |
| **Nth-of-type(n)** | | It style element based on its position in a group of siblings (only Same type) in one parent. **THE COUNT BEGIN FROM TOP TO DOWN FOR SIBLINGS with one type IN THIS PARENT** | | **p:nth-last-child(1) { Background-color: red; }**  **Html page:**  **<p>P1</p><span>SP</span><</p><p>P2</p><p>P3</p></div>** |
| **Nth-last-of-type(n)** | | It style element based on its position in a group of siblings (only Same types) in one parent. **THE COUNT BEGIN FROM DOWN TO TOP FOR SIBLINGS WITH ONE TYPE IN THIS PARENT** | | **p:nth-last-child(1) { Background-color: red; }**  **Html page:**  **<div>**  **<p>P3</p><span>SP</span><</p><p>P2</p><p>P1</p></div>** |
| **Css not(Negation) pseudo class** | | | | |
| **Not(selector)** | It used to select many selectors except that located between the brackets | | #Div p:not(.hh) {Background-color: red;} | |
| **Css lang pseudo class** | | | | |
| **Lang(ISO language code)** | It selects element based on its ISO language code that it determined to be in to styled it with Css. | | P:lang(fr){color:red;}  Html page: <p lang="fr">Top page</p> | |

**Pseudo-elements selectors**

A CSS pseudo-element is used to style specified parts of an element.

For example, it can be used to:

* Style the first letter, or line, of an element.
* Insert content before, or after, the content of an element.

**Syntax:**

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

1. Selector may be (id, class, Groping …). Any type.
2. There is no space between the Selector and ::pseudo-element.

**The Pseudo-elements are:**

|  |  |  |
| --- | --- | --- |
| ::First-letter | It applies styles to the first letter of the first line of a [block-level element](https://developer.mozilla.org/en-US/docs/Web/CSS/Visual_formatting_model#Block-level_elements_and_block_boxes), but only when not proceeded by other content (such as images or inline tables). | p:lang(fr)::first-letter{ font-size:16pt;}  Html page: <p lang= “fr”>Top page</p> |
| ::first-line | It applies styles to the first line of a block-level element. Note that the length of the first line depends on many factors, including the width of the element, the width of the document, and the font size of the text.  **Note:** This property is useful for responsive the window viewport . | p:lang(fr)::first-letter{ font-size:16pt;}  Html page: <p lang= “fr”>Top page<br/>Top page  Top page <br> Top page </p> |
| ::selection | This property is used to give css properties to the selection event (selection like that you do with mouse to copy something)  **Note:** Only certain CSS properties can be applied to ::selection selector:  1) Color 2) Background-color 3) Cursor 4) Caret-color 5) Outline 6) Text-decoration 7) Text-shadow 8) Text-emphasis-color | #div p::selection{background-color:yellow} |
| ::after | In CSS, ::after creates a pseudo-element that is the last child of the selected element. It is often used to add cosmetic content to an element with the content property. It is inline by default. | #div p::after{content:close-quote;} |
| ::before | In CSS, ::before creates a pseudo-element that is the first child of the selected element. It is often used to add cosmetic content to an element with the content property. It is inline by default. | #div p::after{content:close-quote;} |

**Universal Selector**

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

**Example:**

{  
Text-align: center;  
  color: blue;  
}

You can add more selectors to the asterisk selector to make it more specification.

**Ex;**

**.red**

**{Border: 3px solid red}**

**It means style any element with class= “red” by border: 3px solid red;**

**How to Add CSS Codes to the Html page**

There are three ways of inserting a style sheet:

1. External CSS.
2. Internal CSS.
3. Inline CSS.

We will illustrate them in details:

**Firstly; External CSS file.**

**This is an external css file was written separately, and then it can be linked with Html page by this code:**

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

* Where: rel is an abbreviation to relationship.
* Recently it isn’t necessary to write text/css (the mime type) as the browsers Get to know it automatically.
* External css file must be saved with a (.css) extension.
* The external .css file should not contain any HTML tags.

**Secondly;** **Internal CSS.**

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

**The internal style is defined inside the <style> element, within the <head> tag.**

**Example:**

<head>  
<style>  
 body {  
   background-color: linen;  
 }  
 h1 {  
  color: maroon;  
  margin-left: 40px;  
 }  
</style>  
</head>

**Thirdly; Inline CSS.**

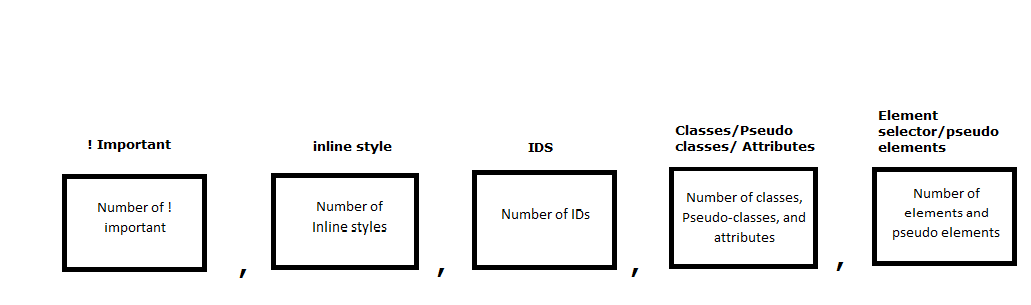
An inline style may be used to apply a unique style for a single element.

Example:

<h1 style="color:blue;text-align:center;">This is a heading</h1>

**Note:** An inline style loses many of the advantages of a style sheet (by mixing content with presentation). Use this method sparingly.

**Specificity Hierarchy of selectors**

****

**! Important (1,0,0,0,0)**

**Inline style (0,1,0,0,0)**

**ID selector (0,0,1,0,0)**

**Class/Pseudo class/Attributes (0,0,0,1,0)**

**Element selector/Pseudo element (0,0,0,0,1)**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **! important** | **Inline style** | **ID selector** | **Class selector** | **Element selector** |
| **P{background-color: red ! important ;}** | **1** | **0** | **0** | **0** | **0** |
| **Style=”background-color: red;”** | **0** | **1** | **0** | **0** | **0** |
| **P #id{background-color: red;}** | **0** | **0** | **1** | **0** | **1** |
| **Section, Div ,P,{background-color: red;}** | **0** | **0** | **0** | **0** | **3\*1=3** |
| **#section Div .class #p span{background-color: red;}** | **0** | **0** | **2\*1=2** | **1** | **2\*1=2** |

**Note: If there are two selectors or more with the same number then the closest one to the interested html element will be applied.**

**Note: Universal selector and inherited values have a specificity of zero. If there is 5 inherited value then the total number (5\*0=0).**

**Units of length**

**Units in css divided into 2 types:**

* 1. **Relative (percentage) units.**
  2. **Absolute units.**

|  |  |  |  |
| --- | --- | --- | --- |
| **Relative (Percentage) units** | | **Absolute units** | |
| **Relative units** | | **Fixed units** | |
| **vw** | Relative to 1% of the width of the viewport\* | **Centimeters (cm)** |  |
| **vh** | Relative to 1% of the height of the viewport\* | **Millimeters (mm)** |  |
| **em** | Relative to the font-size of the element (2em means 2 times the size of the current font) | **Inches (in)** | (1in = 96px = 2.54cm) |
| **rem** | Relative to font-size of the root element | **Pixels (px)** | (1px = 1/96th of 1in) |
| **ex** | Relative to the x-height of the current font (rarely used) | **Points (pt)** | (1pt = 1/72 of 1in) |
| **vmin** | Relative to 1% of viewport's\* smaller dimension | **Picas (pc)** | (1pc = 12 pt) |
| **vmax** | Relative to 1% of viewport's\* larger dimension |  |  |
| **ch** | Relative to width of the "0" (zero) |  |  |
| **%** | Relative to the parent element |  |  |

**Note:**

1. Absolute length units are not recommended for use on screen, because screen sizes vary so much. However, they can be used if the output medium is known, such as for print layout.
2. Pixels (px) are relative to the viewing device. For low-dpi devices, 1px is one device pixel (dot) of the display. For printers and high resolution screens 1px implies multiple device pixels.
3. The em and rem units are practical in creating perfectly scalable layout! \* Viewport = the browser window size. If the viewport is 50cm wide, 1vw = 0.5cm.
4. 1em=16px. So 2em=32px. 2px=0.125em.
5. em and ex units on the font-size property are relative to the parent element's font size (unlike all other properties, where they're relative to the font size on the element). This means em units and percentages do the same thing for font-size.

**Shorthand property.**

It’s a way to specify all the properties in one single property.

It’s a useful way to short the css codes that related to a specific css property.

* 1. **CSS Background Shorthand.**

The shorten background property syntax is:

Selector {background: background-color background-image background-repeat background-attachment background-position;}

**Note:** It does not matter if one of the property values is missing, as long as the other ones are in this order. Note that we do not use the background- attachment property, as it does not have a value.

* 1. **Css Border Shorthand**

The shorten border property syntax is:

Selector { border-width border-style (required) border-color}

* 1. **Css Margin - Shorthand**

The shorten border property syntax is:

Selector {margin: Top-margin Right-margin Bottom-margin Left-margin;}

* 1. **List – Shorthand.**

The shorten border property syntax is:

Selector {list-style: list-style-type list-style-position list-style-image;}

* 1. **Font-shorthand**

The shorten border property syntax is:

Selector {font: font-style font-variant font-weight font-size (required)/line-height font-family(required);}

**Note:** The font-size and font-family values are required. If one of the other values is missing, their default values are used.

* 1. **Padding - Shorthand**

The shorten border property syntax is:

* + - * 1. Selector {Padding: padding-top padding-right padding-bottom padding-left;}

**Note:** if you don’t write the 4th values together then [1st value = 3rd value] and [4th value = 2nd value].

* + - * 1. Selector {Padding: padding-top padding-right padding-bottom ;} then the 4th value(padding-left) =the 2nd value (Padding-top).
        2. Selector {Padding: padding-top padding-right ;} then the 3rd value=1st value & 4th value=2nd value.
        3. Selector {Padding: One value ;} then the 1st,2nd,3rd, and 4th values is the same.

**Note:**

1. Negative values are not allowed.
2. If an element has a specified width, the padding added to that element will be added to the total width of the element. This is often an undesirable result.

Example:

Here, the <div> element is given a width of 300px. However, the actual width of the <div> element will be 350px (300px + 25px of left padding + 25px of right padding):

div {  
  width: 300px;  
  padding: 25px;  
 }

To keep the width at 300px, no matter the amount of padding, you can use the box-sizing property. This causes the element to maintain its width; if you increase the padding, the available content space will decrease.

div {  
  width: 300px;  
  padding: 25px;  
  box-sizing: border-box;  
 }

* 1. **Outline-shorthand:**

The shorten outline property syntax is:

Selector {outline: outline-width outline-style (required) outline-color ;}

**Note:** The outline property is specified as one, two, or three values from the list above. The order of the values does not matter.

* 1. **Border-radius shorthand**

The shorten border-radius property syntax is:

Selector{border-top-left-radius border-top-right-radius border-bottom-right-radius  border-bottom-left-radius;}

**Specific side Property**

It’s an easy way to specify a deferent property for each side. (Top, right, bottom, and left).

1. **CSS Border - Individual Sides.**

In CSS, there are also properties for specifying each of the borders (top, right, bottom, and left):

p {  
  border-top-style: dotted;  
  border-right-style: solid;  
  border-bottom-style: dotted;  
  border-left-style: solid;  
}

* We can also write the 4 values of the 4 border-sides with this syntax:

border-style: dotted solid double dashed; for (Top right bottom left) sides respectably

* And write a 3 values for the 3 border-sides with this syntax:

 border-style: dotted solid double; for (Top right bottom) sides

* And write a 2 values for the 2 border-sides with this syntax:

 border-style: dotted solid; for (Top and bottom right and left) sides

* And write 1 value for all (4) border-sides with this syntax:

 border-style: dotted; for (all sides) sides

1. **CSS Margin - Individual Sides.**

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

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

* We can also write the 4 values of the 4 margin-sides with this syntax:

margin: 10px 20px 10px 5px; for (Top right bottom left) sides respectably

* And write a 3 values for the 3 border-sides with this syntax:

 margin: 10px 20px 10px; for (Top right bottom) sides

* And write a 2 values for the 2 border-sides with this syntax:

 margin: 10px 20px; for (Top and bottom right and left) sides

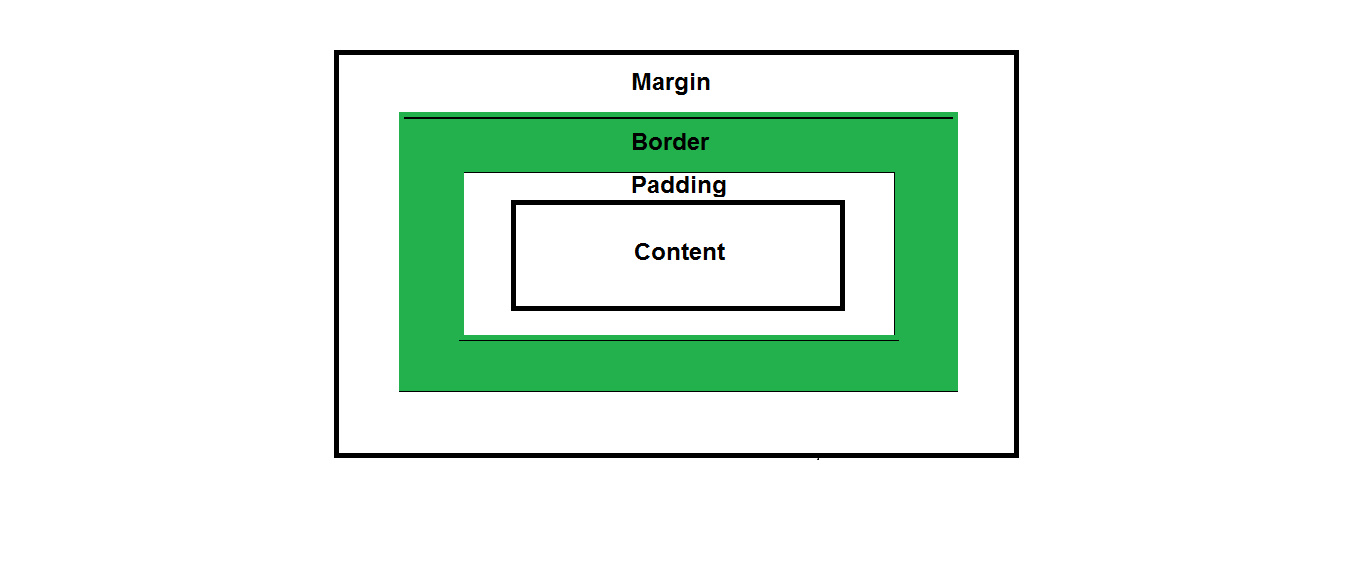
* And write 1 value for all (4) border-sides with this syntax:

 margin: 10px; for (all sides) sides

**Css Box Module**

All HTML elements can be considered as boxes. 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.

**Important:** When you set the width and height properties of an element with CSS, you just set the width and height of the **content area**. To calculate the full size of an element, you must also add padding, borders and margins.

Example

This <div> element will have a total width of 350px:

div {

width: 320px;

padding: 10px;

border: 5px solid gray;

margin: 0;

}

Here is the calculation:

320px (width)

+ 20px (left + right padding)

+ 10px (left + right border)

+ 0px (left + right margin)

= 350px

* + The total width of an element should be calculated like this:
  + Total element width = width + left padding + right padding + left border + right border + left margin + right margin
  + The total height of an element should be calculated like this:
  + Total element height = height + top padding + bottom padding + top border + bottom border + top margin + bottom margin

**Css Transform property**

Css Transform property allows you to transform an html element according to its origin point (50% 50% 0px).

Transform means: translate, rotate, Scale, and Skew.

**Translate:**

Means **changing position** of an html element according to its pivot point (50% 50% 0px)..

**Rotate:**

Means **changing orientation** of an html element according to its pivot point (50% 50% 0px)..

**Scale:**

Means **changing Size** of an html element according to its pivot point (50% 50% 0px)..

**Skew:**

Means **changing slant** of an html element according to its pivot point (50% 50% 0px)..

**Syntax:**

Transform: perspective function + one or more transform functions separated by space;

**Where:**

1. Perspective function is used to define depth perspective [z in pixels].
2. Transform function

|  |  |  |
| --- | --- | --- |
| **1D transform function** | **2D transform function** | **3D transform function** |
| translateX(x)  Used to move an html element horizontally along x-axis.  Where: (+x) move to right; (-x) move to left. | translate(x,y)  Used to move an html element in two dimensions. | Translate3D(x,y,z)  Used to move an html element in 3 dimensions. |
| translateY(y)  Used to move an html element vertically along y-axis.  Where: (+y) move to top; (-y) move to down. |
| translateZ(z)  Used to move an html element in depth along z-axis.  Where: (+z) move toward your eyes; (-z) move away your eyes. |
| **Note**:(±x) in px or % , (±y) in px or % , (±z) in px | | |
| RotateX(x)  Used to rotate an html element in along x-axis.  Where: (+x) rotate with clockwise; (-x) rotate anticlockwise. | Rotate(x,y)  Used to rotate an html element in two dimensions. | Rotate3D(x,y,z)  Used to rotate an html element in two dimensions. |
| RotateY(y)  Used to rotate an html element along y-axis.  Where: (+y) rotate around y-axis; (-y) rotate around y-axis |
| RotateZ(z)  Used to rotate an html element along z-axis.  Where: (+x) rotate around z-axis; (-x) rotate around z-axis. |
| **Note:** ±x,y,z in deg | | |
| ScaleX(x)  Used to stretch (Scaled) an html element along x-axis. | Scale(x,y)  Used to scale an html element in two dimensions. | Scale3D(x,y,z)  Used to scale an html element in two dimensions. |
| ScaleY(y)  Used to stretch (Scaled) an html element along y-axis. |
| ScaleZ(z)  Used to stretch (Scaled) an html element along z-axis. |
| **Note:** +x,y,z in number | | |
| skewX(x)  Used to oblique (skew) an html element along x-axis. | Skew(x,y)  Used to oblique (skew) an html element in two dimensions. | - |
| skewY(y)  Used to oblique (skew) an html element along y-axis. |
| **Note: ±**x,y in deg |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

**EX;**

P{transform: translate3d(33px,40px,50px);}

P{transform: translate3d(33px,40px,50px) scale(2) rotate(59deg);}

**Css Transform-origin point**

* This property is used to change the position of the pivot point of an html element.
* Pivot point is an imagrnary point; used transform an html element in different ways.
* Pivot point by ***default placed at the center*** of an html element which equal to (transform-origin: center center 0px;**OR**transform-origin: 50% 50% opx;)

Syntax:

Transform-origin: x-offset y-offset z-offset;

Where:

* x-offset can be in pixels, %, left, center, right.
* y-offset can be in pixels, %, top, center, bottom.
* z-offset can be in pixels only.

**Note:**

Left= 0% , Right=100% , Top=0% , Bottom=100% , Center= 50%

**Tips**

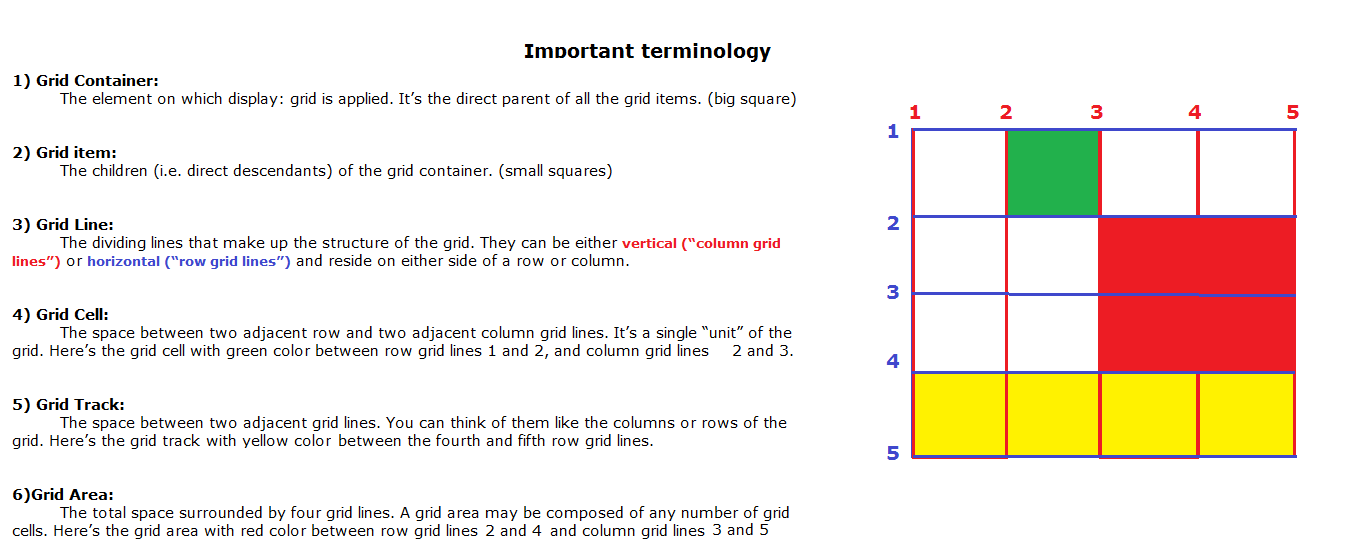
* **Note: For W3C compliant CSS: If you define the color property, you must also define the background-color.**
* **The inherit (parent value) and initial (default value) keywords can be used for any CSS property, and on any HTML element.**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Properties | Keyword Value | | | | | | | | | | | | Description | | | | | | | | | | | | | | | Example | | | | |
| - | - | | | | | | | | | | | | To make a comments only inside css codes (style element).  You can add comments wherever you want in the code: | | | | | | | | | | | | | | | p {   color: red;  /\* Set text color to red \*/} | | | | |
| Comments in one line | | | | | | | | | | | | | | | /\*p {color: red;}\*/ | | | | |
| Comments in multiply lines | | | | | | | | | | | | | | | /\* This is a multi-line comment \*/ p { color: red; } | | | | |
| Css3 properties are divided into 3 types (supported, partial support, not supported).  Vendor prefixes are some words put before specific css3 properties (partial support, not supported) to make it supported in all browsers.  **Note:**   * You write css3 property 5 times (4 times with the 4 prefixes and 1 time without the prefixes). * You can use this site (can I use) to know which css3 properties need vendor prefixes. * Not all css3 properties can be supported with these prefixes as there is some css3 properties don’t support at all (deprecated properties). | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| -webkit- | Specific css3 properties (i.e. box-shadow) | | | | | | | | | | | | -webkit- is a vendor prefix used to support specific css3 properties in the next browsers: (Chrome, Safari, newer versions of Opera, almost all iOS browsers including Firefox for iOS; basically, any WebKit based browser) | | | | | | | | | | | | | | | -webkit-transition: all 4s ease;  -moz-transition: all 4s ease;  -ms-transition: all 4s ease;  -o-transition: all 4s ease;  transition: all 4s ease; | | | | |
| -moz- | Specific css3 properties (i.e. box-shadow) | | | | | | | | | | | | -moz- is a vendor prefix used to support specific css3 properties in (Firefox) | | | | | | | | | | | | | | |
| -o- | Specific css3 properties (i.e. border-radius) | | | | | | | | | | | | -o- is a vendor prefix used to support specific css3 properties in (old pre-WebKit versions of Opera) | | | | | | | | | | | | | | |
| -ms- | Specific css3 properties (i.e. text-shadow) | | | | | | | | | | | | -ms- is a vendor prefix used to support specific css3 properties in the next browsers: (Internet Explorer and Microsoft Edge) | | | | | | | | | | | | | | |
| background-color | The background of an element is the total size of the element, including padding and border (but not the margin).  Tip: Use a background color and a text color that makes the text easy to read. | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Color (Color name, HEX, HSL, RGB, RGBA, HSLA) | | | | | | | | | | | | To define a color for background | | | | | | | | | | | | | | | body { background-color: red; } | | | | |
| background-image  or  background | Used to add background-image. Note: When using a background image, use an image that does not disturb the text.  Tip: determine a background-color with the background-image to replace it if background-color doesn’t appear for any reason. | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| url() | | | | | | | | | | | | To define background image by its url address  You can add more than one photo separated by comma. | | | | | | | | | | | | | | | body { background-image: url("img.png"), url(“img2.jpg”) , url(“img3.jpg”);} | | | | |
| None (default) | | | | | | | | | | | | No background-image. | | | | | | | | | | | | | | | body { background-image: none;} | | | | |
| Linear-gradient (direction or angle, color-stop1, color-stop2, color-stop3,...) | | | | | | | | | | | | * The linear-gradient () function sets a linear gradient as the background image. * To create a linear gradient you must define at least two color stops. * Color stops are the colors you want to render smooth transitions among. * Direction equal [ to left, to right, to top, to bottom, to left top, to left bottom, to right top, to right bottom ]. * Angle equal ± [0:360deg]. * Direction and angle aren’t necessary. * Default direction is   [ to bottom=180deg=-180deg ]   * Color-stop value consists of a color value, followed by an optional stop position (a percentage between 0% and 100% or a length (px, cm) along the gradient axis).   - To make a straight color follow this code→   * For repeating follow this code. → | | | | | | | | | | | | | | |  | | | | |
| background: linear-gradient(130deg,red 0%,red 25%, blue 25%,blue 50%, yellow 50%,yellow 100%); | | | | |
| repeating-linear-gradient() | | | | | | | | | | | | background: repeating-linear-gradient(130deg,red 40%, blue 50%, yellow 20pt); | | | | |
| background-image: radial-gradient(shape size at position, start-color-stop1,stop-color2,stop-color3, ..., last-stop-color);  repeating-radial-gradient() → | | | | | | | | | | | | * The radial-gradient () function sets a radial gradient as the background image. * A radial gradient is defined by its center. * To create a radial gradient you must define at least two color stops. * Shape of the gradient equal:  1. ellipse (default) 2. Circle.  * size of the gradient:  1. Farthest-corner (default). 2. Closest-side. 3. Closest-corner. 4. Farthest-side.  * position:  1. 30% 70% [for X & Y Respectively] , 2. Top left. 3. Top right. 4. Center left. 5. Center right. 6. Center center. 7. Bottom left. 8. Bottom center. 9. Bottom right. 10. 30px 70px[for X & Y Respectively]  * Color-stop value consists of a color value, followed by an optional stop position (a percentage between 0% and 100% or a length (px, cm) along the gradient axis). * You can cancel shape or size .→   - To make a straight color follow this code→  - For repeating follow this code. → | | | | | | | | | | | | | | | background-image: radial-gradient(  circle closest-side at 60% 55%, blue, green, yellow, black); | | | | |
| background-image: radial-gradient(  closest-side at 60% 55%, blue, green, yellow, black);  or | | | | |
| background-image: radial-gradient(  circle at 60% 55%, blue, green, yellow, black); | | | | |
| background-image: radial-gradient(circle at center center, red 0%, red 25%, green 25%, green 50%, yellow 50%, yellow 100%); | | | | |
| background-image: repeating-radial-gradient( circle at center center, red 0%, red 25%, green 25%, green 50%, yellow 50%, yellow 100%); | | | | |
|  | | | | | | | | | | | |  | | | | |
| background-repeat (For images) | Repeat | | | | | | | | | | | | The background image is repeated both vertically and horizontally. The last image will be clipped if it does not fit. This is default | | | | | | | | | | | | | | | background-repeat: repeat; | | | | |
| Repeat-y | | | | | | | | | | | | For repetition only along Y-axis | | | | | | | | | | | | | | | background-repeat: repeat-y; | | | | |
| Repeat-x | | | | | | | | | | | | For repetition only along X-axis | | | | | | | | | | | | | | | background-repeat: repeat-x; | | | | |
| No-repeat | | | | | | | | | | | | For Showing the background image only once | | | | | | | | | | | | | | | background-repeat: no-repeat; | | | | |
| Round | | | | | | | | | | | | The background-image is repeated and stretched to fill the space between repeated images (no gaps) and no clipping at the ends. | | | | | | | | | | | | | | | background-repeat: round; | | | | |
| Space  (opposite to round) | | | | | | | | | | | | The background-image is repeated as much as possible without clipping. The first and last images are pinned to either side of the element, and whitespace is distributed evenly between the images | | | | | | | | | | | | | | | background-repeat: space; | | | | |
| Opacity / Transparency | The opacity property specifies the opacity/transparency of an element. It can take a value from 0.0 - 1.0. The lower value, the more transparent. (default value is 1)  Note: it can also be written with as %.  Note: When using the opacity property to add transparency to the background of an element, all of its child elements inherit the same transparency. This can make the text inside a fully transparent element hard to read.  Note: This property isn’t work in (IE7 and earlier ); so if you wnt to ctivte it in (IE7 and earlier ) use the next syntx:  Filter: alpha (opacity=value)  Where: this vlue [0:100] without ny units.  Ex: Filter: alpha (opacity=50)  Trick: opacity: 0 = visibility: hidden | | | | | | | | | | | | For making opacity / transparency | | | | | | | | | | | | | | | P{color: rgb(244,33,22,0.5)  p {opacity: 50%} = p {opacity: 0.5} | | | | |
| background-position | The background-position property is used to specify the position of the background image. | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Top | | | | | | | | | | | | To move the background to the center top of the page | | | | | | | | | | | | | | | background-position: top; | | | | |
| Center right = Right | | | | | | | | | | | | To move the background to the center right of the page | | | | | | | | | | | | | | | background-position: right; | | | | |
| Center left = left | | | | | | | | | | | | To move the background to the center left of the page | | | | | | | | | | | | | | | background-position: left; | | | | |
| Center Bottom = bottom | | | | | | | | | | | | To move the background to the center bottom of the page | | | | | | | | | | | | | | | background-position: bottom; | | | | |
| Center Center = Center | | | | | | | | | | | | To move the background to the center of the page | | | | | | | | | | | | | | | background-position: center; | | | | |
| Top left | | | | | | | | | | | | To move the background to the top left of the page | | | | | | | | | | | | | | | background-position: top left; | | | | |
| Top right | | | | | | | | | | | | To move the background to the top right of the page | | | | | | | | | | | | | | | background-position: top right; | | | | |
| Bottom left | | | | | | | | | | | | To move the background to the bottom left of the page | | | | | | | | | | | | | | | background-position: bottom left; | | | | |
| Bottom right | | | | | | | | | | | | To move the background to the bottom right of the page | | | | | | | | | | | | | | | background-position: bottom right; | | | | |
| background-attachment | The background-attachment property specifies whether the background image should scroll or be fixed (will not scroll with the rest of the page): | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Fixed | | | | | | | | | | | |  | | | | | | | | | | | | | | | background-attachment: fixed | | | | |
| Local | | | | | | | | | | | |  | | | | | | | | | | | | | | | background-attachment: local; | | | | |
| Scroll | | | | | | | | | | | |  | | | | | | | | | | | | | | | background-attachment: scroll; | | | | |
| Background-size | This property is used to specify the size of the background (width and height) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| auto | | | | | | | | | | | | Default value. The background image is displayed in its original size | | | | | | | | | | | | | | |  | | | | |
| Cover | | | | | | | | | | | | Resize the background image to cover the entire container, even if it has to stretch the image or cut a little bit off one of the edges | | | | | | | | | | | | | | | background-size: cover; | | | | |
| Contain | | | | | | | | | | | | Resize the background image to make sure the image is fully visible | | | | | | | | | | | | | | | background-size: contain; | | | | |
| x- length y-length | | | | | | | | | | | |  | | | | | | | | | | | | | | | background-size: 50px, 100px; | | | | |
| %-width %-height | | | | | | | | | | | |  | | | | | | | | | | | | | | | {background-size: 100% 100%;} | | | | |
| clip | This property is used to clipping (cutting) image which must has an absolute position. **Note:** position of the image must be absolute. | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| auto | | | | | | | | | | | | No clipping will be applied (default value) | | | | | | | | | | | | | img{position: absolute; clip: auto; | | | | | | |
| Rect(top, right, bottom, left) | | | | | | | | | | | | Specify the position of clipping from all sides. | | | | | | | | | | | | | img{position: absolute;  clip: rect(0px,60px,20px,20px)} | | | | | | |
| Clip-path | The clip-path property lets you clip an element to a basic shape or to an SVG source.  **Note**: The clip-path property will replace the deprecated clip property. | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| none | | | | | | | | | | | | This is default. No clipping is done | | | | | | | | | | | | | clip-Path: none; | | | | | | |
| Clip-source | | | | | | | | | | | | Defines a URL to an SVG <clipPath> element | | | | | | | | | | | | | clip-Path: clip-source; | | | | | | |
| Clip-shape | | | | | | | | | | | | Clips an element to a basic shape: circle, ellipse, polygon or inset | | | | | | | | | | | | | clip-Path: clip-shape; | | | | | | |
| Box-margin | | | | | | | | | | | | Uses the margin box as the reference box | | | | | | | | | | | | | clip-Path: box-margin; | | | | | | |
| border-box | | | | | | | | | | | | Uses the border box as the reference box | | | | | | | | | | | | | clip-Path: border-box; | | | | | | |
| padding-box | | | | | | | | | | | | Uses the padding box as the reference box | | | | | | | | | | | | | clip-Path: padding-box; | | | | | | |
| content-box | | | | | | | | | | | | Uses the content box as the reference box | | | | | | | | | | | | | clip-Path: content-box; | | | | | | |
| fill-box | | | | | | | | | | | | Uses the object bounding box as reference box | | | | | | | | | | | | | clip-Path: fill-box; | | | | | | |
| stroke-box | | | | | | | | | | | | Uses the stroke bounding box as reference box | | | | | | | | | | | | | clip-Path: stroke-box; | | | | | | |
| view-box | | | | | | | | | | | | Uses the SVG viewport as reference box | | | | | | | | | | | | | clip-Path: view-box; | | | | | | |
| Object-fit | This property is used to specify how an <img> or <video> should resize to fit its containers | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| None | | | | | | | | | | | | The replaced content is not resized | | | | | | | | | | | | | #koala{width: 500px;height: 200px;object-fit: none; background-color: red;} | | | | | | |
| Fill | | | | | | | | | | | | The replaced content is sized to fill the element’s content box. If necessary the object will stretch or squished to fit.  **Note:**  This is default. | | | | | | | | | | | | | #koala{width: 500px;height: 200px;object-fit: fill ;background-color: red;} | | | | | | |
| Contain | | | | | | | | | | | | The replaced content is scaled to maintain its aspect ratio while fitting within the element's content box | | | | | | | | | | | | | #koala{width: 500px;height: 200px;object-fit: contain; background-color: red;} | | | | | | |
| cover | | | | | | | | | | | | The replaced content is sized to maintain its aspect ratio while filling the element's entire content box. The object will be clipped to fit | | | | | | | | | | | | | #koala{width: 500px;height: 200px;object-fit: cover; background-color: red;} | | | | | | |
| Scale-down | | | | | | | | | | | | The content is sized as if none or contain were specified (would result in a smaller concrete object size) | | | | | | | | | | | | | #koala{width: 500px;height: 200px;object-fit: scale-down; background-color: red;} | | | | | | |
| mix-Blend-mode  Background-blend-mode | These properties belong to blend-mode css data type.  **Where:**   * Mix-blende-mode is css property sets how an element’s content should blend with the it’s background and with the content of its parent. * Background-blend-mode is used to define the blending (mixing) mode of two or more background layer (color or/and image). | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| normal | | | | | | | | | | | | The final color is the top color, regardless of what the bottom color is. (default value) | | | | | | | | | | | body{background-image: url('Koala.jpg'), url(desert.jpg);background-blend-mode: normal;} | | | | | | | | |
| Multiply | | | | | | | | | | | | The final color is the result of multiplying the top nd bottom colors.  Note:   * A black layer (color) leads to a black final layer * A White layer leads to no change | | | | | | | | | | | body{background-image: url('Koala.jpg'); background-color: #333;  background-blend-mode: multiply;} | | | | | | | | |
| Screen | | | | | | | | | | | |  | | | | | | | | | | | body{background-image: url('Koala.jpg'), url(desert.jpg);background-blend-mode: screen;} | | | | | | | | |
| overly | | | | | | | | | | | | The final color is the result of multiply if the bottom color is darker, or screen if the bottom color is lighter.  This blend mode is equivalent to hard-light but with the layers swapped. | | | | | | | | | | | body{background-image: url('Koala.jpg'), url(desert.jpg);background-blend-mode: overly;} | | | | | | | | |
| Darken | | | | | | | | | | | | The final color is composed of the darkest values of each color channel. | | | | | | | | | | | body{background-image: url('Koala.jpg'), url(desert.jpg);background-blend-mode: darken;} | | | | | | | | |
| Lighten | | | | | | | | | | | | The final color is composed of the lightest values of each color channel. | | | | | | | | | | | body{background-image: url('Koala.jpg'), url(desert.jpg);background-blend-mode: lighten;} | | | | | | | | |
| Color-dodge | | | | | | | | | | | | The final color is the result of dividing the bottom color by the inverse of the top color.  A black foreground leads to no change. A foreground with the inverse color of the backdrop leads to a fully lit color.  This blend mode is similar to screen, but the foreground need only be as light as the inverse of the backdrop to create a fully lit color. | | | | | | | | | | | body{background-image: url('Koala.jpg'), url(desert.jpg);background-blend-mode: color-dodge;} | | | | | | | | |
| Burn-color | | | | | | | | | | | | The final color is the result of inverting the bottom color, dividing the value by the top color, and inverting that value.  A white foreground leads to no change. A foreground with the inverse color of the backdrop leads to a black final image.  This blend mode is similar to multiply, but the foreground need only be as dark as the inverse of the backdrop to make the final image black. | | | | | | | | | | | body{background-image: url('Koala.jpg'), url(desert.jpg);background-blend-mode: burn-color-;} | | | | | | | | |
| difference | | | | | | | | | | | | The final color is the result of subtracting the darker of the two colors from the lighter one.  A black layer has no effect, while a white layer inverts the other layer's color. | | | | | | | | | | | body{background-image: url('Koala.jpg'), url(desert.jpg);background-blend-mode: difference-;} | | | | | | | | |
| Saturation | | | | | | | | | | | | The final color has the saturation of the top color, while using the hue and luminosity of the bottom color.  A pure gray backdrop, having no saturation, will have no effect. | | | | | | | | | | | body{background-image: url('Koala.jpg'), url(desert.jpg);background-blend-mode: saturation;} | | | | | | | | |
| Color | | | | | | | | | | | | The final color has the hue and saturation of the top color, while using the luminosity of the bottom color.  The effect preserves gray levels and can be used to colorize the foreground. | | | | | | | | | | | body{background-image: url('Koala.jpg'), url(desert.jpg);background-blend-mode: color;} | | | | | | | | |
| luminosity | | | | | | | | | | | | The final color has the luminosity of the top color, while using the hue and saturation of the bottom color.  This blend mode is equivalent to color, but with the layers swapped. | | | | | | | | | | | body{background-image: url('Koala.jpg'), url(desert.jpg);background-blend-mode: luminosity;} | | | | | | | | |
| Soft-light | | | | | | | | | | | | The final color is similar to hard-light, but softer.  This blend mode behaves similar to hard-light.  The effect is similar to shining a diffused spotlight on the backdrop. | | | | | | | | | | | body{background-image: url('Koala.jpg'), url(desert.jpg);background-blend-mode: soft-color;} | | | | | | | | |
| Hard-color | | | | | | | | | | | | The final color is the result of multiply if the top color is darker, or screen if the top color is lighter.  This blend mode is equivalent to overlay but with the layers swapped.  The effect is similar to shining a harsh spotlight on the backdrop. | | | | | | | | | | | body{background-image: url('Koala.jpg'), url(desert.jpg);background-blend-mode: hard-color;} | | | | | | | | |
| exclusion | | | | | | | | | | | | The final color is similar to difference, but with less contrast.  As with difference, a black layer has no effect, while a white layer inverts the other layer's color. | | | | | | | | | | | body{background-image: url('Koala.jpg'), url(desert.jpg);background-blend-mode: exclusion;} | | | | | | | | |
| hue | | | | | | | | | | | | The final color has the hue of the top color, while using the saturation and luminosity of the bottom color. | | | | | | | | | | | body{background-image: url('Koala.jpg'), url(desert.jpg);background-blend-mode: hue;} | | | | | | | | |
| Border Style | Dashed | | | | | | | | | | | | Defines a dashed border | | | | | | | | | | | border-style: dashed | | | | | | | | |
| Solid | | | | | | | | | | | | Defines a solid border | | | | | | | | | | | border-style: solid; | | | | | | | | |
| Dotted | | | | | | | | | | | | Defines a dotted border | | | | | | | | | | | border-style: dotted; | | | | | | | | |
| Double | | | | | | | | | | | | Defines a double border | | | | | | | | | | | border-style: double; | | | | | | | | |
| Groove | | | | | | | | | | | | Defines a 3D grooved border. The effect depends on the border-color value | | | | | | | | | | | border-style: groove; | | | | | | | | |
| Hidden | | | | | | | | | | | | Defines a hidden border | | | | | | | | | | | border-style: hidden; | | | | | | | | |
| Inset | | | | | | | | | | | | Defines a 3D inset border. The effect depends on the border-color value | | | | | | | | | | | border-style: inset; | | | | | | | | |
| Outset | | | | | | | | | | | | Defines a 3D outset border. The effect depends on the border-color value | | | | | | | | | | | border-style: outset; | | | | | | | | |
| Ridge | | | | | | | | | | | | Defines a 3D ridged border. The effect depends on the border-color value | | | | | | | | | | | border-style: ridge; | | | | | | | | |
| None | | | | | | | | | | | | Defines no border | | | | | | | | | | | border-style: none; | | | | | | | | |
| Border-width | The border-width property specifies the width of the four borders. | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| px, pt, cm, em, etc. | | | | | | | | | | | |  | | | | | | | | | | | | | | | border-width: 5px; | | | | |
| Medium | | | | | | | | | | | |  | | | | | | | | | | | | | | | border-width: medium; | | | | |
| Thick | | | | | | | | | | | |  | | | | | | | | | | | | | | | border-width: thick; | | | | |
| Thin | | | | | | | | | | | |  | | | | | | | | | | | | | | | border-width: thin; | | | | |
| Border Color | The border-color property is used to set the color of the four borders.  Note: If border-color is not set, it inherits the color of the element. | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Color (Color name, HEX, HSL, RGB, RGBA, HSLA) | | | | | | | | | | | |  | | | | | | | Border-color:#ccc; | | | | | | | | | | | | |
| border-radius | This property used To give rounded corners to any html element.  This property is the shorthand for border-top-left-radius border-top-right-radius border-bottom-right-radius border-bottom-left-radius  Respectively.  **We can wrote four values as this**:  { border-top-left-radius border-top-right-radius border-bottom-right-radius border-bottom-left-radius;}  **Or write three values s this :**  { border-top-left-radius border-top-right-radius border-bottom-right-radius ;} note: 4th value=2nd value  **Or write 2 values as this :**  { border-top-left-radius border-top-right-radius ;} note: 1st value=3rd value and 4th value=2nd value  **Or one value as this :**  {Border-radius: Lengths ;}  **Note:**  this value will apply on 1st 2nd 3rd 4th values (four corners). | | | | | | | | | | | | | | | | | | | P{border-radius: 5px 6px 3px 8px;}  P{border-radius: 5px 6px;}  P{border-radius: 5px 6px;}  P{border-radius: 5px;} | | | | | | | | | | | | |
| Length or percentage | | | | | | | | | |  | | | | | | | | | Div{border-radius: 20px;} | | | | | | | | | | | | |
| Horizontal radius Length/ vertical radius Length  OR  Horizontal radius percentage/ vertical radius percentage | | | | | | | | | |  | | | | | | | | | Div{border-radius: 20%/50%;}  Div{border-radius: 20px/30px;} | | | | | | | | | | | | |
| border-top-left-radius length or % border-top-right-radius length or % border-bottom-right-radius length or % border-bottom-left-radius length or %  **/**  border-top-left-radius length or % border-top-right-radius length or % border-bottom-right-radius length or % border-bottom-left-radius length or % | | | | | | | | | | | | | | | | | | | Div{border-radius: 20% 30% 67% 37% / 94% 32% 83% 36%;} | | | | | | | | | | | | |
| border-top-left-radius  border-top-right-radius  border-bottom-right-radius  border-bottom-left-radius | Length value  OR  Length / length | | | | | | | | | | | | To give any html element "rounded corners". | | | | | | | Border-radius:5px; | | | | | | | | | | | | |
| Border-image  (shorthand) | This property allows you to specify an image to be used as the border around an element. (Instead of the normal border around an element).  Border-image is a shorthand property where:  {border-image: border-image-source border-image-slice border-image-width border-image-outset border-image-repeat ;}  **Note:** default value is border-image: none 100% 1 0 strech | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Border-image-source | This property specifies the path to the image to be used as a border (instead of the normal border around an element).  **Note:**   * If the value is "none", or if the image cannot be displayed, the border styles will be used. * none is the default value. | | | | | | | | | | | | | | | | | | | Border-image-source: url(border-png) | | | | | | | | | | | | |
| Border-image-slice | This property specifies how to slice the image specified by border-image-source. The image is always sliced into nine sections: four corners, four edges and the middle.  Note: The "middle" part is treated as fully transparent, unless the fill keyword is set. | | | | | | | | | | | | | | | | | | |  | | | | | | | | | | | | |
| Border-image-width | This property is used to specify the width of the border-image. | | | | | | | | | | | | | | | | | | |  | | | | | | | | | | | | |
| Number | % | | | | | | | | | | | | | The default value of number is 1. | | | | | |  | | | | | | | | | | | | |
| top right bottom left | | | | | | | | | | | | | Specify the width in 4 number values.  Note: 1st value=3rd value and 2nd value=4th value. | | | | | |  | | | | | | | | | | | | |
| auto | | | | | | | | | | | | |  | | | | | |  | | | | | | | | | | | | |
| Border-image-outset | This property is used to control the amount of area between border image and the border of the element itself | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Length or % | | | | | | | | | | | | | Specify the area amount in length | | | | | | border-image-width: 10px; | | | | | | | | | | | | |
| Number | | | | | | | | | | | | | Specify the area amount in one number value | | | | | | border-image-width: 1; | | | | | | | | | | | | |
| top right bottom left | | | | | | | | | | | | | Specify the area amount in 4 number values.  Note: 1st value=3rd value and 2nd value=4th value. | | | | | | border-image-width: 1 0 1 1; | | | | | | | | | | | | |
| Border-image-repeat | This property specify whether the border image should be repeated, rounded or stretched | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| stretch | | | | | | | | | | | | | | The border image should stretch to fill the area | | | | |  | | | | | | | | | | | | |
| Repeat | | | | | | | | | | | | | | The border image is tilled (repeated) to fill the area | | | | |  | | | | | | | | | | | | |
| round | | | | | | | | | | | | | |  | | | | |  | | | | | | | | | | | | |
| space | | | | | | | | | | | | | |  | | | | |  | | | | | | | | | | | | |
| filter | The filter property defines visual effects (like blur and saturation) to an element (often <img>).  **Note:** To use multiple filters, separate each filter with a space (See "More Examples" below). | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| none | | | | | | | | | | | | | |  | | | | |  | | | | | | | | | | | | |
| blur(px) | | | | | | | | | | | | | | blur(px) Applies a blur (clearness) effect to the image. A larger value will create more blur.  **Note:** If no value is specified, 0 is used. | | | | | Img{width: 600px; height: 400px; filter:blur(40px)} | | | | | | | | | | | | |
| Brightness(% or 0:1) | | | | | | | | | | | | | | brightness(%) Adjusts (edit) the brightness of the image.  Note:   1. 0% will make the image completely black. 2. 100% (1) is default and represents the original image. 3. Values over 100% will provide brighter results. | | | | | Img{width: 600px; height: 400px; filter: brightness(.4)} | | | | | | | | | | | | |
| Contrast(% or 0:1) | | | | | | | | | | | | | | contrast(%) Adjusts the contrast of the image.  Note:   1. 0% will make the image completely black. 2. 100% (1) is default, and represents the original image. 3. Values over 100% will provide results with more contrast. | | | | | Img{width: 600px; height: 400px; filter: contrast(.4)} | | | | | | | | | | | | |
| Sepia(% or 0:1) | | | | | | | | | | | | | | sepia(%) Converts the image to sepia.   1. 0% (0) is default and represents the original image. 2. 100% will make the image completely sepia.   **Note:** Negative values are not allowed. | | | | | Img{width: 600px; height: 400px; filter: sepia(.4)} | | | | | | | | | | | | |
| Grayscale(% or 0:1) | | | | | | | | | | | | | | This property specify the percentage of grey color in <img>  Note:   1. 0% (0) is default and represents the original image. 2. 100% will make the image completely gray (used for black and white images).   **Note:** Negative values are not allowed. Play it » | | | | | Img{width: 600px; height: 400px; filter: grayscale (.4)} | | | | | | | | | | | | |
| Custom() | | | | | | | | | | | | | |  | | | | |  | | | | | | | | | | | | |
| drop-shadow(h-shadow v-shadow blur spread color) | | | | | | | | | | | | | | This property Applies a drop shadow effect to the image.  **Where:**   * h-shadow - Required. Specifies a pixel value for the horizontal shadow. Negative values place the shadow to the left of the image. * v-shadow - Required. Specifies a pixel value for the vertical shadow. Negative values place the shadow above the image. * blur - Optional. This is the third value, and must be in pixels. Adds a blur effect to the shadow. A larger value will create more blur (the shadow becomes bigger and lighter). Negative values are not allowed. If no value is specified, 0 is used (the shadow's edge is sharp). * spread - Optional. This is the fourth value, and must be in pixels. Positive values will cause the shadow to expand and grow bigger, and negative values will cause the shadow to shrink. If not specified, it will be 0 (the shadow will be the same size as the element).   Note: Chrome, Safari and Opera, and maybe other browsers, do not support this 4th length; it will not render if added.   * color - Optional. Adds a color to the shadow. If not specified, the color depends on the browser (often black).   **Note:** This filter is similar to the box-shadow property | | | | | filter: drop-shadow(8px 8px 10px red); | | | | | | | | | | | | |
| Hue-rotate(deg) | | | | | | | | | | | | | | hue-rotate(deg) Applies a hue rotation on the image. The value defines the number of degrees around the color circle the image samples will be adjusted. 0deg is default, and represents the original image.  Note: Maximum value is 360deg | | | | | Img{width: 600px; height: 400px; filter: hue-rotate(40deg)} | | | | | | | | | | | | |
| Invert() | | | | | | | | | | | | | | invert(%) Inverts the samples in the image.   1. 0% (0) is default and represents the original image. 2. 100% will make the image completely inverted.   **Note:** Negative values are not allowed. | | | | | Img{width: 600px; height: 400px; filter: invert(.4)} | | | | | | | | | | | | |
| Opacity(% or 0:1) | | | | | | | | | | | | | | opacity(%) Sets the opacity level for the image. The opacity-level describes the transparency-level, where:   1. 0% is completely transparent. 2. 100% (1) is default and represents the original image (no transparency).   **Note:** Negative values are not allowed.  Tip: This filter is similar to the opacity property. | | | | | Img{width: 600px; height: 400px; filter: opacity(.4)} | | | | | | | | | | | | |
| Saturate(% or 0:1) | | | | | | | | | | | | | | saturate(%) Saturates the image.   1. 0% (0) will make the image completely un-saturated. 2. 100% is default and represents the original image.   Values over 100% provide super-saturated results.  **Note:** Negative values are not allowed. | | | | | Img{width: 600px; height: 400px; filter: Saturate(.4)} | | | | | | | | | | | | |
| url() | | | | | | | | | | | | | | url() The url() function takes the location of an XML file that specifies an SVG filter, and may include an anchor to a specific filter element. Example: | | | | | filter: url(svg-url#element-id) | | | | | | | | | | | | |
| Margins | The CSS margin properties are used to create space around elements, outside of any defined borders (Outer filling). | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| px, pt, cm, em, etc. | | | | | | | | | | | | Used as an outer | | | | | | | | | | | | | | | Margin:5px; | | | | |
| Auto | | | | | | | | | | | | Used to horizontally center only block elements that have a definite width and height.  **Note:**  1- Center aligning has no effect if the width property is not set (or set to 100%).  2- This property is useless with texts, so use text-align with texts.  3- You can centering <img> (Html tag) by adding to it;  {Margin-left: auto; margin-right: auto;  Display: block ;}. If it is a background-image (Css property) then used {background-position}.  4- auto property specify only the right and left margins. Then you can say that;  {Margin: 50px auto 10px auto ;} and the element still centered, with specific top and down margins.  5- Margins can be positive or negative value.  6- margin collapse | | | | | | | | | | | | | | | Div {Margin: auto;}  img { display: block; margin-left: auto;  margin-right: auto; width: 40%;} | | | | |
| Padding  padding-top  padding-right  padding-bottom  padding-left | This property sets the padding area on all four sides of an element. (inner filling). | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| px, pt, cm, em, etc. | | | | | | | | | | | | Used to make inner filling.  Has only a positive values. | | | | | | | | | | | | | | | Padding:5px; | | | | |
| Height and width  Max-width  Max-height  Min-width  Min-height | Height and width properties set the height/width of the area inside the padding, border, and margin of the element (Content area).  Max-width properties set the maximum width of an element. It prevents the used value of the width property from becoming larger than the value specified by max-width.  **Note**:   * - The height and width properties do not include padding, borders, or margins. * - The width CSS property sets an element's width. By default, it sets the width of the content area, but if box-sizing is set to border-box, it sets the width of the border area. * - max-width and min-width **overrides** width, but min-width overrides max-width | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| px, pt, cm, em, %, etc. | | | | | | | | | | | |  | | | | | | | | | | | | | | | div { height: 200px; width:220px;} | | | | |
| Auto | | | | | | | | | | | | This is default. The browser calculates the height and width | | | | | | | | | | | | | | | div { height: auto; width: auto;} | | | | |
| max-content | | | | | | | | | | | | The intrinsic preferred width. | | | | | | | | | | | | | | | div { height inherit; width: max-content;} | | | | |
| min-content | | | | | | | | | | | | The intrinsic minimum width. | | | | | | | | | | | | | | | div { height inherit; width: min-content;} | | | | |
| fit-content | | | | | | | | | | | |  | | | | | | | | | | | | | | | div { height inherit; width: fit-content;} | | | | |
| Outline | An outline is a line that is drawn around elements, OUTSIDE the borders, not outside the margin to make the element "stand out". This property is the shorthand to set various outline properties in a single declaration: outline-style, outline-width, and outline-color.  **Note:**  Borders and outlines are very similar. However, outlines differ from borders in the following ways:   1. Outline differs from borders! Unlike border, the outline never takes up space, as it is drawn outside the element's border, and may overlap other content. 2. Also, the outline is NOT a part of the element's dimensions; but border is. 3. the element's total width and height is not affected by the width of the outline, but affected by border (total size of box-module) 4. According to the spec, outlines don't have to be rectangular, although they usually are. 5. Border can treated as a 4 separately sides (top, right, bottom, left); but outline is not (it always treated as a box). | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Outline-width | The outline-width property specifies the width of the outline. | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| px, pt, cm, em, etc. | | | | | | | | | | | |  | | | | | | | | | | | | | | border-width: 5px; | | | | | |
| Medium | | | | | | | | | | | |  | | | | | | | | | | | | | | border-width: medium; | | | | | |
| Thick | | | | | | | | | | | |  | | | | | | | | | | | | | | border-width: thick; | | | | | |
| Thin | | | | | | | | | | | |  | | | | | | | | | | | | | | border-width: thin; | | | | | |
| px, pt, cm, em, etc. | | | | | | | | | | | |  | | | | | | | | | | | | | | border-width: 5px; | | | | | |
| Outline-style | The outline-style property specifies the style of the outline. | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Dashed | | | | | | | | | | | | Defines a dashed border | | | | | | | | | | | | | | outline-style: dashed | | | | | |
| Solid | | | | | | | | | | | | Defines a solid border | | | | | | | | | | | | | | outline-style: solid; | | | | | |
| Dotted | | | | | | | | | | | | Defines a dotted border | | | | | | | | | | | | | | outline-style: dotted; | | | | | |
| Double | | | | | | | | | | | | Defines a double border | | | | | | | | | | | | | | outline-style: double; | | | | | |
| Groove | | | | | | | | | | | | Defines a 3D grooved border. The effect depends on the border-color value | | | | | | | | | | | | | | outline-style: groove; | | | | | |
| Hidden | | | | | | | | | | | | Defines a hidden border | | | | | | | | | | | | | | outline-style: hidden; | | | | | |
| Inset | | | | | | | | | | | | Defines a 3D inset border. The effect depends on the border-color value | | | | | | | | | | | | | | outline-style: inset; | | | | | |
| Outset | | | | | | | | | | | | Defines a 3D outset border. The effect depends on the border-color value | | | | | | | | | | | | | | outline-style: outset; | | | | | |
| Ridge | | | | | | | | | | | | Defines a 3D ridged border. The effect depends on the border-color value | | | | | | | | | | | | | | outline-style: ridge; | | | | | |
| None | | | | | | | | | | | | Defines no border | | | | | | | | | | | | | | outline-style: none; | | | | | |
| Dashed | | | | | | | | | | | | Defines a dashed border | | | | | | | | | | | | | | outline-style: dashed; | | | | | |
| Outline-color | The outline-color property is used to set the color of the outline.  **Note:** invert - performs a color inversion (which ensures that the outline is visible, regardless of color background) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| color name, hex, hsl, hsla, rgb, rgba, invert | | | | | | | | | | | |  | | | | | | | | | | | | | | Outlinr-color:red; | | | | | |
| Outline-offset | The outline-offset property adds space between an outline and the edge/border of an element.  **Note:**   * The space between an element and its outline is transparent. * Outline-offset and resize property are both called user interface properties | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Length | | | | | | | | | | | | - | | | | | | | | | | | | | | P{outline: 22px;} | | | | | |
| None = 0px | | | | | | | | | | | | For no-outline. (Used to remove the default outline of many html elements like: inputs.) | | | | | | | | | | | | | | P{outline: none;} | | | | | |
| text-align | This property sets the horizontal alignment of a block element or table-cell box. This means it works like vertical-align but in the horizontal direction. | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Justify | | | | | | | | | | | | The inline contents are justified. Text should be spaced to line up its left and right edges to the left and right edges of the line box, except for the last line. | | | | | | | | | | | | | | | text-align: justify; | | | | |
| Justify-all | | | | | | | | | | | | Same as justify, but also forces the last line to be justified. | | | | | | | | | | | | | | |  | | | | |
| Match-parent = Inherit | | | | | | | | | | | | Property should inherit its value from its parent element. | | | | | | | | | | | | | | | text-align: inherit; | | | | |
| Center | | | | | | | | | | | | Text in the center | | | | | | | | | | | | | | | text-align: center; | | | | |
| Right | | | | | | | | | | | | Text in the right | | | | | | | | | | | | | | | text-align: right; | | | | |
| left | | | | | | | | | | | | Text in the left | | | | | | | | | | | | | | | text-align: left; | | | | |
| start | | | | | | | | | | | | The same as left if direction is left-to-right and right if direction is right-to-left. | | | | | | | | | | | | | | |  | | | | |
| End  <string> لسة ماخدتوش | | | | | | | | | | | | The same as right if direction is left-to-right and left if direction is right-to-left. | | | | | | | | | | | | | | |  | | | | |
| Text-align-last | This property sets how the last line of a block is aligned. | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| auto | | | | | | | | | | | | | | | Default value. The last line is justified and aligned left | | | | | | | | | | | | | | text-align-last: auto; | | |
| Justify | | | | | | | | | | | | | | | The last line is stretched to make it width equal to the width of the above lines in the same block elements. | | | | | | | | | | | | | | text-align-last: justify; | | |
| Center | | | | | | | | | | | | | | | Last line of the Text should be in the center | | | | | | | | | | | | | | text-align-last: center; | | |
| Right | | | | | | | | | | | | | | | Last line of the Text should be in the right | | | | | | | | | | | | | | text-align-last: right; | | |
| left | | | | | | | | | | | | | | | Last line of the Text should be in the left | | | | | | | | | | | | | | text-align-last: left; | | |
| start | | | | | | | | | | | | | | | The same as left if direction is left-to-right and right if direction is right-to-left. | | | | | | | | | | | | | | text-align-last: start; | | |
| end | | | | | | | | | | | | | | | The same as right if direction is left-to-right and left if direction is right-to-left. | | | | | | | | | | | | | | text-align-last: end; | | |
| Direction | ltr | | | | | | | | | | | | | | | Writing direction from left to right (English) | | | | | | | | | | | | | | | text | |
| rtl | | | | | | | | | | | | | | | Writing direction from right to left (Arabic) | | | | | | | | | | | | | | | P {Direction:rtl;} | |
| unicode-bidi | The unicode-bidi property is used together with the direction property to set or return whether the text should be overridden to support multiple languages in the same document. | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| normal | | | | | | | | | | | |  | | | | | | | | | | | | | | | p { direction: rtl; unicode-bidi: normal;} | | | | |
| embed | | | | | | | | | | | |  | | | | | | | | | | | | | | | p { direction: rtl; unicode-bidi: embed;} | | | | |
| bidi-override | | | | | | | | | | | |  | | | | | | | | | | | | | | | p {direction: rtl; unicode-bidi: bidi-override;} | | | | |
| isolate | | | | | | | | | | | |  | | | | | | | | | | | | | | | p {direction: rtl; unicode-bidi: isolate;} | | | | |
| Isolate-override | | | | | | | | | | | |  | | | | | | | | | | | | | | | p {direction: rtl; unicode-bidi: isolate-override;} | | | | |
| plaintext | | | | | | | | | | | |  | | | | | | | | | | | | | | | p {direction: rtl; unicode-bidi: plaintext;} | | | | |
| Vertical-align | This property sets the vertical alignment of an element.  Note: that vertical-align only applies to inline, inline-block and table-cell elements: you can't use it to vertically align block-level elements.  Note: Negative values are allowed. | | | | | | | | | | | | | | | | | | | | | | | | | | |  | | | | |
| Values for table cells | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| baseline | | | | | | | | | | | | The element is aligned with the baseline of the parent. This is default | | | | | | | | | | | | | | |  | | | | |
| Top | | | | | | | | | | | | The element is aligned with the top of the tallest element on the line | | | | | | | | | | | | | | | img.top { vertical-align: top;} | | | | |
| middle | | | | | | | | | | | | The element is placed in the middle of the parent element | | | | | | | | | | | | | | | img.top { vertical-align: middle;} | | | | |
| bottom | | | | | | | | | | | | The element is aligned with the lowest element on the line | | | | | | | | | | | | | | | img.top { vertical-align: text-top;} | | | | |
| Text-bottom | | | | | | | | | | | | The element is aligned with the bottom of the parent element's font | | | | | | | | | | | | | | | img.top { vertical-align: text-bottom;} | | | | |
| Super | | | | | | | | | | | | The element is aligned with the superscript baseline of the parent | | | | | | | | | | | | | | | img.top { vertical-align: super;} | | | | |
| Sub | | | | | | | | | | | | The element is aligned with the subscript baseline of the parent | | | | | | | | | | | | | | | img.top { vertical-align: middle;} | | | | |
| Text-top | | | | | | | | | | | | The element is aligned with the top of the parent element's font | | | | | | | | | | | | | | | img.top { vertical-align: bottom;} | | | | |
| % | | | | | | | | | | | | Raises or lower an element in a percent of the "line-height" property. Negative values are allowed | | | | | | | | | | | | | | | img.top { vertical-align: baseline;} | | | | |
| length | | | | | | | | | | | | Raises or lower an element by the specified length. Negative values are allowed. | | | | | | | | | | | | | | |  | | | | |
| color | Color (Color name, HEX, HSL, RGB, RGBA, HSLA) | | | | | | | | | | | | To identify the text-color | | | | | | | | | | | | | | | P {color:red;} | | | | |
| Text-decoration | The shorten text-decoration property syntax is:  Selector {text-decoration: text-decoration-line (required) text-decoration-color text-decoration-style;} | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Text-decoration-line | This property sets the kind of text decoration to use (like underline, overline, and line-through).  Note: You can also combine more than one value, like underline and overline to display lines both under and over the text. | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| none | | | | | | | | | | | | Without the default underline | | | | | | | | | | | | | | | a { text-decoration: none;} | | | | |
| Line-through | | | | | | | | | | | | ~~To make a line passing through the link~~ | | | | | | | | | | | | | | | a { text-decoration: line-through;} | | | | |
| Overline | | | | | | | | | | | | To make a line over the link | | | | | | | | | | | | | | | a { text-decoration: overline;} | | | | |
| underline | | | | | | | | | | | | To make a line under the link | | | | | | | | | | | | | | | a { text-decoration: underline;} | | | | |
| Text-decoration-color | Color (Color name, HEX, HSL, RGB, RGBA, HSLA) | | | | | | | | | | | | This property specifies the color of the text-decoration. | | | | | | | | | | | | | | | P{ Text-decoration-color: red;} | | | | |
| Text-decoration-style | This property sets the style of the lines specified by text-decoration-line. The style applies to all lines that are set with text-decoration-line. | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| dashed | | | | | | | | | | | | Draws a dashed line. | | | | | | | | | | | | | | | a {text-decoration-style: dashed;} | | | | |
| dotted | | | | | | | | | | | | Draws a dotted line. | | | | | | | | | | | | | | | a {text-decoration-style: dotted;} | | | | |
| wavy | | | | | | | | | | | | Draws a wavy line. | | | | | | | | | | | | | | | a {text-decoration-style: wavy;} | | | | |
| double | | | | | | | | | | | | Draws a double line. | | | | | | | | | | | | | | | a {text-decoration-style: double;} | | | | |
| solid | | | | | | | | | | | | Draws a single solid line. | | | | | | | | | | | | | | | a {text-decoration-style: solid;} | | | | |
| Word-Wrap | This property allows you to break long words and wrap it onto the next line. (example of long word: url links) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| normal | | | | | | Break words only at allowed break points (breaking words from the spacing located between these words) this is the default value. | | | | | | | | | | | | | | | | | | | | | P{word-wrap: normal} | | | | |
| Break-word | | | | | | Allows unbreakable words to be broken(one long word without any spacing) | | | | | | | | | | | | | | | | | | | | | P{word-wrap: break-word} | | | | |
| Word-break | The word-break property specifies how words should break when reaching the end of a line. Like <wbd> in html. | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| normal | | | | Default value. Uses default line break rules | | | | | | | | | | | | | | | | | | | |  | | | | | | | |
| Break-all | | | | To prevent overflow, word may be broken at any character | | | | | | | | | | | | | | | | | | | |  | | | | | | | |
| Keep-all | | | | Word breaks should not be used for Chinese/Japanese/Korean (CJK) text. Non-CJK text behavior is the same as value "normal" | | | | | | | | | | | | | | | | | | | |  | | | | | | | |
| Break-word | | | | To prevent overflow, word may be broken at arbitrary points | | | | | | | | | | | | | | | | | | | |  | | | | | | | |
|  |  | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|  | | | | | | | | | | | |  | | | | | | | | | | | | | | |  | | | | |
|  | | | | | | | | | | | |  | | | | | | | | | | | | | | |  | | | | |
|  | | | | | | | | | | | |  | | | | | | | | | | | | | | |  | | | | |
|  | | | | | | | | | | | |  | | | | | | | | | | | | | | |  | | | | |
|  | | | | | | | | | | | |  | | | | | | | | | | | | | | |  | | | | |
| Text-transform | It used to specify the text case to be applied. | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Capitalize | | | | | | | | | | | | The first letter in each word in the text is capital | | | | | | | | | | | | | | | p.uppercase { text-transform: capitalize; } | | | | |
| Full-width | | | | | | | | | | | |  | | | | | | | | | | | | | | | p.uppercase { text-transform: full-width; } | | | | |
|  | | | | | | | | | | | |  | | | | | | | | | | | | | | |  | | | | |
| Lowercase | | | | | | | | | | | | All letters in the word will be smalls | | | | | | | | | | | | | | | p.uppercase { text-transform: lowercase; } | | | | |
| None | | | | | | | | | | | | Is a keyword that prevents the case of all characters from being changed. | | | | | | | | | | | | | | | p.uppercase { text-transform: none; } | | | | |
| uppercase | | | | | | | | | | | | All letters in the word will be capitals | | | | | | | | | | | | | | | p.uppercase { text-transform: uppercase; } | | | | |
| Text-indent | The text-indent property specifies the indentation (Initial distance) of the first line in a text-block.  Note: Negative values are allowed. The first line will be indented to the left if the value is negative. | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Length | | | | | | | | | | | | Defines a fixed indentation in px, pt, cm, em, etc. Default value is 0. | | | | | | | | | | | | | | | | | p { text-indent: 50px;} | | |
| % | | | | | | | | | | | | Defines the indentation in % of the width of the parent element | | | | | | | | | | | | | | | | | P{text-indent: 20%} | | |
| Text-overflow | The text-overflow property specifies how overflowed content that is not displayed should be signaled (view) to the user. It can be clipped, display an ellipsis (...), or display a custom string.  **Note:** to use this property you should specify   * white-space: nowrap; - overflow: hidden; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| clip | | | | | | | | | | | | Default value. The text is clipped and not accessible | | | | | | | | | | | | | | | P{text-overflow: clip;} | | | | |
| ellipsis | | | | | | | | | | | | Render an ellipsis ("...") to represent the clipped text | | | | | | | | | | | | | | | P{text-overflow: ellpsis;} | | | | |
| string | | | | | | | | | | | | Render the given string to represent the clipped text | | | | | | | | | | | | | | | P{text-overflow: string;} | | | | |
| Text-justify | The text-justify property specifies the justification method of text when text-align is set to "justify". | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| inter-word | | | | | | | | | | | | The text is justified by adding space between words (effectively varying word-spacing), which is most appropriate for languages that separate words using spaces, like English or Korean. | | | | | | | | | | | | | | |  | | | | |
| inter-character | | | | | | | | | | | | The text is justified by adding space between characters (effectively varying letter-spacing), which is most appropriate for languages like Japanese. | | | | | | | | | | | | | | |  | | | | |
| auto | | | | | | | | | | | | The browser chooses the best type of justification for the current situation based on a balance between performance and quality, but also on what is most appropriate for the language of the text (e.g., English, CJK languages, etc.). This is the default justification used if text-justify is not set at all. | | | | | | | | | | | | | | |  | | | | |
| none | | | | | | | | | | | | The text justification is turned off. This has the same effect as not setting text-align at all, although it is useful if you need to turn justification on and off for some reason. | | | | | | | | | | | | | | |  | | | | |
| Letter-spacing | It used to specify the space between letters and characters. Positive values of letter-spacing cause characters to spread farther apart, while negative values of letter-spacing bring characters closer together. | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| px, pt, cm, em, etc | | | | | | | | | | | |  | | | | | | | | | | | | | | | h1 { letter-spacing: 3px; } | | | | |
| Normal | | | | | | | | | | | |  | | | | | | | | | | | | | | | h1 { letter-spacing: normal; } | | | | |
| Line-height | It used to specify the space between vertical lines . | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| px, pt, cm, em, etc | | | | | | | | | | | |  | | | | | | | | | | | | | | | p.small { line-height: 0.8 ; } | | | | |
| Normal | | | | | | | | | | | |  | | | | | | | | | | | | | | | p.small { line-height: normal ; } | | | | |
| Word-spacing | This property specifies the space between the words (increases or decreases the white space between words).  Note: Negative values are allowed | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| length | | | | | | | | | | | | Defines an additional space between words (in px, pt, cm, em, etc). | | | | | | | | | | | | | | | h1 { word-spacing: 10px; } | | | | |
| normal | | | | | | | | | | | | Defines normal space between words (0.25em) . This is default | | | | | | | | | | | | | | | h1 { word-spacing:normal } | | | | |
| White-space | The white-space property specifies how white-space inside an element (between the words) is handled. | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| normal | | | | | | | | | | | | Sequences of whitespace will collapse into a single whitespace. Text will wrap when necessary. This is default | | | | | | | | | | | | | | | p {white-space: normal;} | | | | |
| Nowrap | | | | | | | | | | | | Sequences of whitespace will collapse into a single whitespace. Text will never wrap to the next line. The text continues on the same line until a <br> tag is encountered | | | | | | | | | | | | | | | p {white-space: nowrap;} | | | | |
| Pre | | | | | | | | | | | | Whitespace is preserved by the browser. Text will only wrap on line breaks. Acts like the <pre> tag in HTML | | | | | | | | | | | | | | | p {white-space: pre;} | | | | |
| Pre-line | | | | | | | | | | | | Sequences of whitespace will collapse into a single whitespace. Text will wrap when necessary, and on line breaks | | | | | | | | | | | | | | | p {white-space: pre-line;} | | | | |
| Pre-wrap | | | | | | | | | | | | Whitespace is preserved by the browser. Text will wrap when necessary, and on line breaks | | | | | | | | | | | | | | | p {white-space: pre-wrap;} | | | | |
| break-spaces | | | | | | | | | | | | The behavior is identical to that of pre-wrap, except that:  Any sequence of preserved white space always takes up space, including at the end of the line.  A line breaking opportunity exists after every preserved white space character, including between white space characters.  Such preserved spaces take up space and do not hang, and thus affect the box’s intrinsic sizes (min-content size and max-content size). | | | | | | | | | | | | | | | p {white-space: break-spaces;} | | | | |
| Writing-mode | Specify whether lines of text are laid out horizontally or vertically. ( specify the direction of the text) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| horizontal-tb | | | | | | | | | | | | Let the content flow horizontally from left to right, vertically from top to bottom in multi lines. | | | | | | | | | | | | | | | P{writing-mode: horizontal-tb} | | | | |
| vertical-rl | | | | | | | | | | | | Let the content flow vertically from top to bottom, horizontally from right to left in multi lines. | | | | | | | | | | | | | | | P{writing-mode: verticwraal-rl} | | | | |
| vertical-lr | | | | | | | | | | | | Let the content flow vertically from top to bottom, horizontally from left to right in multi lines. | | | | | | | | | | | | | | | P{writing-mode: vertical-lr} | | | | |
| Text-shadow | This property adds shadow effect to the text. Each shadow is described by some combination of X and Y offsets (required) from the element, blur radius, and color.  Where:   1. Offset-x (h-shadow): specifies the horizontal distance; a (+ve) value places the shadow to the right of the text , and (-ve) value places it to the left (required) 2. Offset-y (v-shadow): specifies the vertical distance; a (+ve) value places the shadow under the text, and the (-ve) value places it above the text (required). 3. Blur-radius: This is a <length> value. The higher the value, the bigger the blur; the shadow becomes wider and lighter. It defaults to 0 (Optional). 4. Color: The color of the shadow. It can be specified either before or after the offset values (Optional).   **Notes**:   1. You can apply more than one shadow on one text; and this shadow lists are separated by a comma. 2. When more than one shadow is given, shadows are applied front-to-back, with the first-specified shadow on top. 3. This property applies to both ::first-line and ::first-letter pseudo-elements. | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | | | | | | | | | | | | | h1 { text-shadow: 1px 2px 2px red; } | | | | |
| none | | | | | Default value. No shadow | | | | | | | | | | | | | | | | | | | | | | h1 { text-shadow: none; } | | | | |
| Box-shadow | This property adds shadow effects around an element's frame. A box shadow is described by inset keyword, X and Y offsets relative to the element, blur and spread radius, and color.  Where:   1. Inset: The presence of the inset keyword changes the shadow to one inside the frame (as if the content was depressed inside the box). Inset shadows are drawn inside the border (even transparent ones), above the background, but below content. If inset keyword is not specified (default), the shadow is assumed to be a drop shadow (as if the box were raised above the content). ***(optional)*** 2. Offset-x: specifies the horizontal distance; a (+ve) value places the shadow to the right of the text, and (-ve) value places it to the left ***(required).*** 3. Offset-y: specifies the vertical distance; a (+ve) value places the shadow under the text, and the (-ve) value places it above the text ***(required).*** 4. Blur-radius: This is a third <length> value. The larger this value, the bigger the blur, so the shadow becomes bigger and lighter. Negative values are not allowed. If not specified, it will be 0 (the shadow's edge is sharp). ***(optional)*** 5. Spread-radius: This is a fourth <length> value. Positive values will cause the shadow to expand and grow bigger; negative values will cause the shadow to shrink. If not specified, it will be 0 (the shadow will be the same size as the element). ***(optional)*** 6. Color: The color of the shadow. It can be specified either before or after the offset values ***(Optional).*** If not specified, it defaults to current color.   **Notes:**   1. You can apply more than one shadow on one text; and this shadow lists are separated by a comma. 2. Box-shadow generator is an interactive tool allowing you to generate a box-shadow. 3. If a border-radius is specified on the element with a box shadow, the box shadow takes on the same rounded corners. 4. The z-ordering of multiple box shadows is the same as multiple text shadows (the first specified shadow is on top). | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | | | | | | | | | | | | h1 { box-shadow: inset 1px 2px 2px 3px red; } | | | | | |
| none | | | | | Default value. No shadow | | | | | | | | | | | | | | | | | | | | | | h1 { text-shadow: none; } | | | | |
| font |  | | | | |  | | | | | | | | | | | | | | | | | | | | | |  | | | | |
| Font Family | "Times New Roman", Times, serif; | | | | | The differ between font-family and generic-family is that:   * **generic family** - a group of font families with a similar look (like "Serif" or "Monospace") * **font family** - a specific font family (like "Times New Roman" or "Arial") | | | | | | | | | | | | | | | | | | | | | | P{"Trebuchet MS", Helvetica, sans-serif;} | | | | |
| To identify font-family.   * If font-name consist of 2 or more section, then use double or single quotes and comma to separate between them. * Always use fullback system (several font names). * Start with the font you want, and end with a generic family, to let the browser pick a similar font in the generic family, if no other fonts are available. | | | | | | | | | | | | | | | | | | | | | |
| font-style | It used to specify the style of a text to be display. | | | | | | | | | | | | | | | | | | | | | | | | | | |  | | | | |
| Normal | | | | | | | | | | | | The text is shown normally | | | | | | | | | | | | | | | P {font-style: Normal} | | | | |
| Italic | | | | | | | | | | | | For leaning the text; The font-style property is mostly used to specify italic text. | | | | | | | | | | | | | | | P {font-style: italic} | | | | |
| Oblique | | | | | | | | | | | | it’s very similar to italic, but less supported | | | | | | | | | | | | | | | P {font-style: oblique} | | | | |
| Font Weight | It used to specify the boldness of a font | | | | | | | | | | | | | | | | | | | | | | | | | | |  | | | | |
| Normal | | | | | | | | | | | | For normal weight (default) | | | | | | | | | | | | | | |  | | | | |
| Bold | | | | | | | | | | | | For thick text | | | | | | | | | | | | | | |  | | | | |
| Bolder | | | | | | | | | | | | For thicker text | | | | | | | | | | | | | | |  | | | | |
| Lighter | | | | | | | | | | | | For thinner text | | | | | | | | | | | | | | |  | | | | |
| 100-200-300-400-500-600-700-800-900 | | | | | | | | | | | | 400 (default) , 700 (bold) | | | | | | | | | | | | | | |  | | | | |
| Font-variant | Small-caps | | | | | | | | | | | | For making all letters in word capital. | | | | | | | | | | | | | | | p{font-variant: small-caps;} | | | | |
| Normal | | | | | | | | | | | | The default display. | | | | | | | | | | | | | | | P{font-variant: normal;} | | | | |
| Font-size | Used to specify the size of the text | | | | | | | | | | | | | | | | | | | | | | | | | | |  | | | | |
| Pt, px, pt, cm, em, vmin,vmax,rem,vh  *100% = 1em = 1rem = 16px = 12pt* (default) | | | | | | | | | | | | Font-size in | | | | | | | | | | | | | | | P {font-size:200%; }  = p {font-size:32px;} | | | | |
| Large, x-large, xx-large, xxx-large,  -webkit-xxx-large | | | | | | | | | | | | To make font-size big and big | | | | | | | | | | | | | | |  | | | | |
| Small, x-small, xx-small. | | | | | | | | | | | | To make font-size small and small | | | | | | | | | | | | | | |  | | | | |
| medium | | | | | | | | | | | | The default size (equal 16px). | | | | | | | | | | | | | | |  | | | | |
| Larger and smaller | | | | | | | | | | | | Size is larger or smaller than the font-size of the parent element. | | | | | | | | | | | | | | |  | | | | |
| unset | | | | | | | | | | | |  | | | | | | | | | | | | | | |  | | | | |
| Font-google | 'Dancing Script', cursive; or any other type | | | | | | | | | | | | To embedded an external font-family | | | | | | | | | | | | | | | <head><link rel=”font-family url”>  <style> p {'Dancing Script', cursive;} </style></head> | | | | |
| Any element | :link | | This order  Should be  respected | | | | | | | | | | a normal, unvisited link | | | | | | | | | | | | | | | a:link { color: red;} | | | | |
| :visited | | a link the user has visited | | | | | | | | | | | | | | | a:visited { color: green;} | | | | |
| :hover | | a link when the user mouses over it | | | | | | | | | | | | | | | a:hover {color: hotpink;} | | | | |
| :active | | a link the moment it is clicked | | | | | | | | | | | | | | | a:active { color: blue;} | | | | |
| To make a tooltip hover you should   1. Type your tooltip text in a inline child element like : span 2. Add to it by Css {visibility: hidden ;}. 3. In the parent of this child add a hover selector and use {visibility: visible ;}. 4. **Note:** 5. Hover selector be in the parent element (not in another child in this parent). 6. **Example:**   **.tooltip-container { height: 50px; width: 50%; background-color: black; color: white;}**  **span { background-color: red; visibility: hidden;}**  **.tooltip-container:hover span**  **{visibility: visible;}** | | | | | | | | | | | | | | | | | | | | | | | | | | | p {  display: none;  background-color: yellow;  padding: 20px; }  div:hover p { display: block; } | | | | |
| Pointer-events | The pointer-events property defines whether or not an element reacts to pointer events. | | | | | | | | | | | | | | | | | | | | | | | | | | |  | | | | |
| Auto | | | | | | | | | | | | | | | | The element reacts to pointer events, like :hover and click. This is default | | | | | | | | | | | Div{pointer-events : auto;} | | | | |
| none | | | | | | | | | | | | | | | | The element does not react to pointer events | | | | | | | | | | | Div{pointer-events : none;} | | | | |
| all | | | | | | | | | | | | | | | |  | | | | | | | | | | | Div{pointer-events : all;} | | | | |
| Fill | | | | | | | | | | | | | | | |  | | | | | | | | | | | Div{pointer-events : fill;} | | | | |
| painted | | | | | | | | | | | | | | | |  | | | | | | | | | | | Div{pointer-events : painted;} | | | | |
| Stroke | | | | | | | | | | | | | | | |  | | | | | | | | | | | Div{pointer-events : stroke;} | | | | |
| visible | | | | | | | | | | | | | | | |  | | | | | | | | | | | Div{pointer-events : visible;} | | | | |
| visiblefill | | | | | | | | | | | | | | | |  | | | | | | | | | | | Div{pointer-events : visiblefill;} | | | | |
| visiblepainted | | | | | | | | | | | | | | | |  | | | | | | | | | | | Div{pointer-events : visiblepinted;} | | | | |
| visiblestroke | | | | | | | | | | | | | | | |  | | | | | | | | | | | Div{pointer-events : visiblestroke;} | | | | |
| CSS Multiple Columns allows you to divide your text into multi columns like in newspapers. | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| column-count | | numbers | | | | | | | | | | | | | | | | The column-count property specifies the number of columns an element should be divided into. | | | | div { column-count: 3;} | | | | | | | | | | |
| column-width | | Length or percentage | | | | | | | | | | | | | | | | The column-width property specifies a suggested, optimal width for the columns. | | | | div { column-width: 30px;} | | | | | | | | | | |
| column-gap | | Length or percentage | | | | | | | | | | | | | | | | This property specifies the gap between the columns. | | | | div { column-gap: 3;} | | | | | | | | | | |
| column-rule-style | | Dotted, solid. Dashed, groove, inset, outset.. | | | | | | | | | | | | | | | | The column-rule-style property specifies the style of the rule between columns. | | | | div { column-rule-style: solid;} | | | | | | | | | | |
| column-rule-width | | Length or percentage | | | | | | | | | | | | | | | | The column-rule-width property specifies the width of the rule between columns: | | | | div { column-rule-width: 3px;} | | | | | | | | | | |
| column-rule-color | | color | | | | | | | | | | | | | | | | The column-rule-color property specifies the color of the rule between columns: | | | | div { column-rule-color: #aaa;} | | | | | | | | | | |
| column-rule | | The column-rule property is a shorthand property for setting all the column-rule-\* properties above. Where:  {Column-rule : column-rule-width column-rule-style column-rule-color;} EX: div { column-rule: 1px solid red;} | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| column-span | | The column-span property specifies how many columns an element should span across. | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| All | | | | | | | | | | | | | | | |  | | | | h2 { column-span: all;} | | | | | | | | | | |
| none | | | | | | | | | | | | | | | |  | | | | h2 { column-span:none;} | | | | | | | | | | |
| columns | | The columns property is a shorthand property for:   * column-width * column-count   The column-width part will define the minimum width for each column, while the column-count part will define the maximum number of columns. By using this property, the multi-column layout will automatically break down into a single column at narrow browser widths, without the need of media queries or other rules. | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Auto | | | | | | | | | | | | | | | | Default value. Sets both the column-width and column-count to "auto" | | | | div { columns: auto auto;} | | | | | | | | | | |
| Length or percentage number | | | | | | | | | | | | | | | | Defines the minimum width for each column and the maximum number of columns | | | | div { columns: 100px 3;} | | | | | | | | | | |
| quotes | This property is used to change the style of quotations mark that appears when using <q>element.  To use this property you should:   * + - * 1. Put some text inside <q> element.         2. In css write the next syntax: selector{quotes: “right quote value” “left quote value”}.   Note:   * Each <q> element takes 2 values one for the right quote, the other for left quote. * If there is more than <q> element in the same parent, then you should separate between the 2 values of each <q> element by comma.   p{quotes: “right quote value 1” “left quote value 1” , “right quote value 2” “left quote value 2” , “right quote value 3” “left quote value 3”;} | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| \0022 | " | | | | | | | | Double quote | | | | | | | | | | | | | | | | | | | #a { quotes: "”" “”";} | | | | |
| \0027 | ' | | | | | | | | single quote | | | | | | | | | | | | | | | | | | | #a { quotes: "’" “’";} | | | | |
| \2039 | < | | | | | | | | single, left angle quote | | | | | | | | | | | | | | | | | | | #a { quotes: "<" “>";} | | | | |
| 203A | > | | | | | | | | single, right angle quote | | | | | | | | | | | | | | | | | | | #a { quotes: "\203A";} | | | | |
| \00AB | « | | | | | | | | double, left angle quote | | | | | | | | | | | | | | | | | | | #a { quotes: "\00AB" “00BB";} | | | | |
| \00BB | » | | | | | | | | double, right angle quote | | | | | | | | | | | | | | | | | | | #a { quotes: "’" “’";} | | | | |
| \2018 | ‘ | | | | | | | | left quote (single high-6) | | | | | | | | | | | | | | | | | | | #a { quotes: "201C" “\201E";} | | | | |
| \2019 | ’ | | | | | | | | right quote (single high-9) | | | | | | | | | | | | | | | | | | | #a { quotes: "\00AB" “00BB";} | | | | |
| \201C | “ | | | | | | | | left quote (double high-6) | | | | | | | | | | | | | | | | | | | #a { quotes: "\2018" “\203A";} | | | | |
| \201D | ” | | | | | | | | right quote (double high-9) | | | | | | | | | | | | | | | | | | | #a { quotes: "\2018" “\203A";} | | | | |
| \201E | „ | | | | | | | | double quote (double low-9) | | | | | | | | | | | | | | | | | | | #a { quotes: "201C" “\201E";} | | | | |
| Cursor | none | | | | | | | |  | | | | | | | | | | | | | | | | | | |  | | | | |
| alias | | | | | | | |  | | | | | | | | | | | | | | | | | | |  | | | | |
| all-scroll | | | | | | | |  | | | | | | | | | | | | | | | | | | |  | | | | |
| auto | | | | | | | |  | | | | | | | | | | | | | | | | | | |  | | | | |
| Cell | | | | | | | |  | | | | | | | | | | | | | | | | | | |  | | | | |
| copy | | | | | | | |  | | | | | | | | | | | | | | | | | | |  | | | | |
| Col-resize | | | | | | | |  | | | | | | | | | | | | | | | | | | |  | | | | |
| Context-menu | | | | | | | |  | | | | | | | | | | | | | | | | | | |  | | | | |
| Default | | | | | | | |  | | | | | | | | | | | | | | | | | | |  | | | | |
| Crosshair | | | | | | | |  | | | | | | | | | | | | | | | | | | |  | | | | |
| e-resize | | | | | | | |  | | | | | | | | | | | | | | | | | | |  | | | | |
| ew-resize | | | | | | | |  | | | | | | | | | | | | | | | | | | |  | | | | |
| n-resize | | | | | | | |  | | | | | | | | | | | | | | | | | | |  | | | | |
| ne-resize | | | | | | | |  | | | | | | | | | | | | | | | | | | |  | | | | |
| nesw-resize | | | | | | | |  | | | | | | | | | | | | | | | | | | |  | | | | |
| No-drop | | | | | | | |  | | | | | | | | | | | | | | | | | | |  | | | | |
| Grab | | | | | | | |  | | | | | | | | | | | | | | | | | | |  | | | | |
| Grabbing | | | | | | | |  | | | | | | | | | | | | | | | | | | |  | | | | |
| help | | | | | | | |  | | | | | | | | | | | | | | | | | | |  | | | | |
| move | | | | | | | |  | | | | | | | | | | | | | | | | | | |  | | | | |
| Not-allowed | | | | | | | |  | | | | | | | | | | | | | | | | | | |  | | | | |
| Ns-resize | | | | | | | |  | | | | | | | | | | | | | | | | | | |  | | | | |
| Nw-resize | | | | | | | |  | | | | | | | | | | | | | | | | | | |  | | | | |
| Nwse-resize | | | | | | | |  | | | | | | | | | | | | | | | | | | |  | | | | |
| Pointer | | | | | | | |  | | | | | | | | | | | | | | | | | | |  | | | | |
| text | | | | | | | |  | | | | | | | | | | | | | | | | | | |  | | | | |
| Progress | | | | | | | |  | | | | | | | | | | | | | | | | | | |  | | | | |
| Row-resize | | | | | | | |  | | | | | | | | | | | | | | | | | | |  | | | | |
| s-resize | | | | | | | |  | | | | | | | | | | | | | | | | | | |  | | | | |
| Se-resize | | | | | | | |  | | | | | | | | | | | | | | | | | | |  | | | | |
| Sw-resize | | | | | | | |  | | | | | | | | | | | | | | | | | | |  | | | | |
| w-resize | | | | | | | |  | | | | | | | | | | | | | | | | | | |  | | | | |
| Vertical-text | | | | | | | |  | | | | | | | | | | | | | | | | | | |  | | | | |
| Zoom-in | | | | | | | |  | | | | | | | | | | | | | | | | | | |  | | | | |
| Zoom-out | | | | | | | |  | | | | | | | | | | | | | | | | | | |  | | | | |
| wait | | | | | | | |  | | | | | | | | | | | | | | | | | | |  | | | | |
| Carset-color | This property is used to specify the color of the cursor (carset) for only **editable** elements like inputs or elements with contenteditable value. | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| auto | | | | | | | | Default value (the browses uses current color ) | | | | | | | | | | | | | | | | | | | input{caret-color: auto;} | | | | |
| color | | | | | | | | Color name or code | | | | | | | | | | | | | | | | | | | input{caret-color: red;} | | | | |
| list-style-type | This property specifies the type of list item marker. | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| none | | | | | | | | No marker is shown | | | | | | | | | | | | | | | | | | | ul.a {list-style-type: none;} | | | | |
| disc | | | | | | | | Default value. The marker is a filled circle | | | | | | | | | | | | | | | | | | | ul.a {list-style-type: disc;} | | | | |
| armenian | | | | | | | | The marker is traditional Armenian numbering | | | | | | | | | | | | | | | | | | | ul.a {list-style-type: armenian;} | | | | |
| Circle | | | | | | | | The marker is a circle | | | | | | | | | | | | | | | | | | | ul.a {list-style-type:circle;} | | | | |
| cjk-ideographic | | | | | | | | The marker is plain ideographic numbers | | | | | | | | | | | | | | | | | | | ul.a {list-style-type: cjk-ideographic;} | | | | |
| Decimal | | | | | | | | The marker is a number | | | | | | | | | | | | | | | | | | | ul.a {list-style-type: Decimal;} | | | | |
| Decimal-leading-zero | | | | | | | | The marker is a number with leading zeros (01, 02, 03, etc.) | | | | | | | | | | | | | | | | | | | ul.a {list-style-type: Decimal-leading-zero;} | | | | |
| georgain | | | | | | | | The marker is traditional Georgian numbering | | | | | | | | | | | | | | | | | | | ul.a {list-style-type: georgain;} | | | | |
| hebrew | | | | | | | | The marker is traditional Hebrew numbering | | | | | | | | | | | | | | | | | | | ul.a {list-style-type: hebrew;} | | | | |
| hiragana | | | | | | | | The marker is traditional Hiragana numbering | | | | | | | | | | | | | | | | | | | ul.a {list-style-type: hiragana;} | | | | |
| hiragana-iroha | | | | | | | | The marker is traditional Hiragana iroha numbering | | | | | | | | | | | | | | | | | | | ul.a {list-style-type: hiragana-iroha;} | | | | |
| katakana | | | | | | | | The marker is traditional Katakana numbering | | | | | | | | | | | | | | | | | | | ul.a {list-style-type: circle;} | | | | |
| katakana-iroha | | | | | | | | The marker is traditional Katakana iroha numbering | | | | | | | | | | | | | | | | | | | ul.a {list-style-type: katakana-iroha;} | | | | |
| Lower-alpha | | | | | | | | The marker is lower-alpha (a, b, c, d, e, etc.) | | | | | | | | | | | | | | | | | | | ul.a {list-style-type: Lower-alpha;} | | | | |
| Upper-alpha | | | | | | | | The marker is upper-alpha (A, B, C, D, E, etc.) | | | | | | | | | | | | | | | | | | | ul.a {list-style-type: Lower-greek;} | | | | |
| Lower-greek | | | | | | | | The marker is lower-greek | | | | | | | | | | | | | | | | | | | ul.a {list-style-type: Lower-greek;} | | | | |
| Upper-greek | | | | | | | | The marker is upper-greek | | | | | | | | | | | | | | | | | | | ul.a {list-style-type: Upper-greek;} | | | | |
| Lower-latin | | | | | | | | The marker is lower-latin (a, b, c, d, e, etc.) | | | | | | | | | | | | | | | | | | | ul.a {list-style-type: Lower-latin;} | | | | |
| Upper-latin | | | | | | | | The marker is upper-latin (A, B, C, D, E, etc.) | | | | | | | | | | | | | | | | | | | ul.a {list-style-type: Upper-latin;} | | | | |
| Lower-roman | | | | | | | | The marker is lower-roman (i, ii, iii, iv, v, etc.) | | | | | | | | | | | | | | | | | | | ul.a {list-style-type: Lower-roman;} | | | | |
| Upper-roman | | | | | | | | The marker is upper-roman (I, II, III, IV, V, etc.) | | | | | | | | | | | | | | | | | | | ul.a {list-style-type: Upper-roman;} | | | | |
| square | | | | | | | | The marker is a square | | | | | | | | | | | | | | | | | | | ul.a {list-style-type: square;} | | | | |
| list-style-image | This property is used to replace the list-item marker with an image.  Note: Always specify the list-style-type property in addition. This property is used if the image for some reason is unavailable. | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| none | | | | | | | | No image list (Default value) | | | | | | | | | | | | | | | | | | |  | | | | |
| url() | | | | | | | | The path to the image to be used as a list-item marker | | | | | | | | | | | | | | | | | | |  | | | | |
| list-style-position | This property specifies the position of the list-item markers (bullet points). | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| inside | | | | | | | | The bullet points will be inside the list item block. As it is part of the list item, it will be part of the text and push the text at the start: | | | | | | | | | | | | | | | | | | | ul.a {  list-style-position: intside;  } | | | | |
| outside | | | | | | | | The bullet points will be outside the list item block. The start of each line of a list item will be aligned vertically: | | | | | | | | | | | | | | | | | | | ul.a {  list-style-position: outside;  } | | | | |
| border-collapse | Collapse | | | | | | | | Notice that any table has a double border. This is because both the table and the <th> and <td> elements have separate borders.  The border-collapse property sets whether the table borders should be collapsed into a single border.  When cells are collapsed, the border-style value of outset behaves like groove, and inset behaves like ridge. | | | | | | | | | | | | | | | | | | | #table2  {  border-collapse: separate;  } | | | | |
| initial=Separate | | | | | | | | (double border)Default value.  When cells are separated, the distance between cells is defined by the border-spacing property.  (this property is equivalent to a margin around each cell) | | | | | | | | | | | | | | | | | | | #table2  {  border-collapse: separate;  border-spacing:6px;  } | | | | |
| Overflow & Overflow-X & overflow-Y | This property Specifies what happens if content overflows an element's box.  This property specifies whether to clip content or to add scrollbars when an element's content is too big to fit in a specified area.  Note: The overflow property only works for block elements with a specified height. | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| visible | | | The overflow is not clipped. It renders outside the element's box. This is default | | | | | | | | | | | | | | | | | | | Div {overflow: visible;}  Div {overflow-x: visible; } or div {overflow-y: visible;} | | | | | | | | | |
| auto | | | If overflow is clipped, a scroll-bar should be added both vertically and horizontally to see the rest of the content | | | | | | | | | | | | | | | | | | | Div {overflow: auto;}  Div {overflow-x: auto; } or div {overflow-y: auto;} | | | | | | | | | |
| Hidden | | | The overflow is clipped, and the rest of the content will be invisible | | | | | | | | | | | | | | | | | | | Div {overflow: hidden;}  Div {overflow-x: hidden; } or div {overflow-y: hidden;} | | | | | | | | | |
| scroll | | | The overflow is clipped, but a scroll-bar is added to see the rest of the content the scroll bar may be vertically or horizontally according to the direction of content overflow.  If the content overflow along width (x-axis) then horizontal scroll bar will added, If the content overflow along height (y-axis) then vertical scroll bar will added | | | | | | | | | | | | | | | | | | | Div {overflow: scroll;}  Div {overflow-x: scroll; } or div {overflow-y: scroll;} | | | | | | | | | |
| Caption-side | This property specifies the placement of a table **caption**. Note: this property applied table element not in the caption element. | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| top | | | Caption locate at table top (default value) | | | | | | | | | | | | | | | | | | | table{caption-side: bottom;} | | | | | | | | | |
| bottom | | | Caption locate at table bottom | | | | | | | | | | | | | | | | | | | table{caption-side: bottom;} | | | | | | | | | |
| display | none | | | It used to hide elements, and by using java script the user can display them on clicking on a specific bottom | | | | | | | | | | | | | | | | | | |  | | | | | | | | | |
| block | | | To convert inline elements to block (the element will take the full width ),but you can’t add another block elements inside it | | | | | | | | | | | | | | | | | | |  | | | | | | | | | |
| inline | | | To convert block elements to inline (located in the same level) | | | | | | | | | | | | | | | | | | |  | | | | | | | | | |
| Inline-block | | | To make elements inline (located in the same level);and block elements(treated as a box) | | | | | | | | | | | | | | | | | | |  | | | | | | | | | |
| Contents | | |  | | | | | | | | | | | | | | | | | | |  | | | | | | | | | |
| Flex | | |  | | | | | | | | | | | | | | | | | | |  | | | | | | | | | |
| Inline-flex | | |  | | | | | | | | | | | | | | | | | | |  | | | | | | | | | |
| Flow-root | | |  | | | | | | | | | | | | | | | | | | |  | | | | | | | | | |
| Grid | | |  | | | | | | | | | | | | | | | | | | |  | | | | | | | | | |
| Inline-grid | | |  | | | | | | | | | | | | | | | | | | |  | | | | | | | | | |
| List item | | |  | | | | | | | | | | | | | | | | | | |  | | | | | | | | | |
| Run-in | | |  | | | | | | | | | | | | | | | | | | |  | | | | | | | | | |
| subgrid | | |  | | | | | | | | | | | | | | | | | | |  | | | | | | | | | |
| table | | |  | | | | | | | | | | | | | | | | | | |  | | | | | | | | | |
| table-caption | | |  | | | | | | | | | | | | | | | | | | |  | | | | | | | | | |
| table-cell | | |  | | | | | | | | | | | | | | | | | | |  | | | | | | | | | |
| table-column | | |  | | | | | | | | | | | | | | | | | | |  | | | | | | | | | |
| table-column-group | | |  | | | | | | | | | | | | | | | | | | |  | | | | | | | | | |
| table-footer-group | | |  | | | | | | | | | | | | | | | | | | |  | | | | | | | | | |
| table-header-group | | |  | | | | | | | | | | | | | | | | | | |  | | | | | | | | | |
| table-row | | |  | | | | | | | | | | | | | | | | | | |  | | | | | | | | | |
| table-row-group | | |  | | | | | | | | | | | | | | | | | | |  | | | | | | | | | |
| Visibility | The visibility property specifies whether or not an element is visible.  Tip: Hidden elements take up space on the page. If you want to hide or remove this space Use the display property. | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Collapse | | | | | | | | Only for table rows (<tr>), row groups (<tbody>), columns (<col>), column groups (<colgroup>). This value removes a row or column, but it does not affect the table layout. The space taken up by the row or column will be available for other content.  If collapse is used on other elements, it renders as "hidden" | | | | | | | | | | | | | | | | | | | Div{visibility: collapse;} | | | | |
| visible | | | | | | | | Default value. The element is visible | | | | | | | | | | | | | | | | | | | Div{visibility: visible;} | | | | |
| Hidden | | | | | | | | Used to hidden element (this property look like: display: none ;)  But the difference that the element (with visibility: hidden ;) will still takes up the same space as before. The element will be hidden, but still affect the layout: | | | | | | | | | | | | | | | | | | | Div{visibility: hidden;} | | | | |
| Max-width |  | | | | | | | | It is useful for making the element responsive when resizing the browser’s window.  It’s useful with textarea tag as it giving a max-width to it prevent the it from stretching when the user add a huge information on it; then the designer shape will be damage.  Note: user can change the width and height of the textarea field. So we add max-width and max-height to prevent that. | | | | | | | | | | | | | | | | | | |  | | | | |
| position | This property is used to move specific html element using top, left, right, bottom properties. | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| static | | | | | | | | The default value, it can’t be moved because it ignores the values of top, right, left, and bottom. | | | | | | | | | | | | | | | | | | | Div {Position:static;} | | | | |
| Relative | | | | | | | | Used to move an html element from its position  **Note:**   * When you move a relative element, its initial position within the parent element will be empty (preserved position). * Relative element doesn’t leave its layer. * It use surrounding elements as an edge to move from it.   Meaning when you move a relative element with top property (i.e. 20px) it will move 20px from the **edge of the upper element** (not from the upper edge of the page browser). | | | | | | | | | | | | | | | | | | | P{  position : relative;  top: 20px;  } | | | | |
| Absolute | | | | | | | | Used to move the html element relative to the edges of its first ancestor that has a {position: relative}, but if there isn’t any ancestor in html page with {position: relative} then it move relative to the edges of the **browser viewport**.(meant the view part of the browser page that appear to users eye )  Ex: body{width: 3000px; height: 3000px;}  .relative{ position: absolute; top: 0px; right: 0px;}  **Note** in this example that the absolute element take the top right position of the viewport not the top right position of the full browsers page.  **Note:**   * Absolute element leave its current layer (parent layer) * Only in this case of absolute position you can add more than position to the same element. | | | | | | | | | | | | | | | | | | | #class { position: relative;}  #class section  { position: relative; position: absolute; top: 90px;}  #class section P{position: absolute;}  Where #class is the parent of section, and section is the parent of paragraph. | | | | |
| fixed | | | | | | | | Used to pin and fix the html element relative to the edges of the browser’s viewport.  **Note:**   * fixed element leave its current layer (parent layer)   uses in (fixed navigation bar, fixed social media……).  Idea:  Html page:  <div>  <p class="fixed">social media icon</p>  </div>  Css style  .fixed {position: fixed; top: 0px; left: -60px; background-color: #f00; height: 30px; width: 100px;}  .relative :hover{ position: fixed; left:10px; transition: left 2s ease-in-out;} | | | | | | | | | | | | | | | | | | | P { position: fixed;} | | | | |
| sticky | | | | | | | | Used to position and stick an html element relative to the top edge of the browser’s viewport.  **Note:**   * In this case you can’t add more than position to the same element. * To activate this position you should write the top property | | | | | | | | | | | | | | | | | | | P {position: sticky;top:0px;} | | | | |
| z-index | ( -∞ : +∞ ) | | | | | | | | For overlapping elements | | | | | | | | | | | | | | | | | | |  | | | | |
| all | This property is used to resets all css properties of a specific element.  Note: this property isn’t effect on direction and Unicode-bidi properties of the text. | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| initial | | | | | | | This property is used to resets all css properties of a specific element.  To default value | | | | | | | | | | | | | | | | | | | | | Div {color: red; position: relative; all: initial;} | | | |
| Inherit | | | | | | |  | | | | | | | | | | | | | | | | | | | | | Div {color: red; position: relative; all: inherit;} | | | |
| unset | | | | | | |  | | | | | | | | | | | | | | | | | | | | | Div {color: red; position: relative; all: all;} | | | |
| User-select | The user-select property specifies whether the text of an element can be selected to be copied.  **Note:** In web browsers, if you double-click on some text it will be selected / highlighted. This property can be used to prevent this. | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Auto | | | | | | | | Default. Text can be selected if the browser allows it | | | | | | | | | | | | | | | | | | p{user-select: auto;} | | | | | |
| None | | | | | | | | Prevent text selection (prevent copy, cut or deleted this text) | | | | | | | | | | | | | | | | | | p{user-select: none;} | | | | | |
| Text | | | | | | | | The text can be selected by the user (used with browsers that doesn’t allow text-selection by default) | | | | | | | | | | | | | | | | | | p{user-select: text;} | | | | | |
| all | | | | | | | | Text selection is made with one click instead of a double-click | | | | | | | | | | | | | | | | | | p{user-select: all;} | | | | | |
| flow | none | | | | | | | |  | | | | | | | | | | | | | | | | | | |  | | | | |
| Left | | | | | | | |  | | | | | | | | | | | | | | | | | | |  | | | | |
| right | | | | | | | |  | | | | | | | | | | | | | | | | | | |  | | | | |
| clear | none | | | | | | | |  | | | | | | | | | | | | | | | | | | |  | | | | |
| Left | | | | | | | |  | | | | | | | | | | | | | | | | | | |  | | | | |
| Right | | | | | | | |  | | | | | | | | | | | | | | | | | | |  | | | | |
| both | | | | | | | |  | | | | | | | | | | | | | | | | | | |  | | | | |
| content | This property is used to add a generated content (content was added without using Html elements).  The content property is used with the ::before and ::after pseudo-elements, to insert generated content. | | | | | | | | | | | | | | | | | | | | | | | | | | |  | | | | |
| “text” Or ‘text ‘ | | | | | | | | | | | | You used this quotes to write a text. | | | | | | | | | | | | | | | input:out-of-range + label::after { content: 'out of range!';} | | | | |
| None | | | | | | | | | | | |  | | | | | | | | | | | | | | |  | | | | |
| Attr(attribute) | | | | | | | | | | | |  | | | | | | | | | | | | | | |  | | | | |
| normal | | | | | | | | | | | | Used to set the content to it’s default value (default value) | | | | | | | | | | | | | | |  | | | | |
| Close-quote | | | | | | | | | | | |  | | | | | | | | | | | | | | |  | | | | |
| No-close-quote | | | | | | | | | | | |  | | | | | | | | | | | | | | |  | | | | |
| Open-quote | | | | | | | | | | | |  | | | | | | | | | | | | | | |  | | | | |
| No-open-quote | | | | | | | | | | | |  | | | | | | | | | | | | | | |  | | | | |
| url(link, image, audio, video….) | | | | | | | | | | | |  | | | | | | | | | | | | | | |  | | | | |
| string | | | | | | | | | | | |  | | | | | | | | | | | | | | |  | | | | |
| counter | | | | | | | | | | | |  | | | | | | | | | | | | | | |  | | | | |
| justify-content | center | | | | | | | | | | | |  | | | | | | | | | | | | | | |  | | | | |
| Flex-end | | | | | | | | | | | |  | | | | | | | | | | | | | | |  | | | | |
| Flex-start | | | | | | | | | | | |  | | | | | | | | | | | | | | |  | | | | |
| Space-around | | | | | | | | | | | |  | | | | | | | | | | | | | | |  | | | | |
| Space-between | | | | | | | | | | | |  | | | | | | | | | | | | | | |  | | | | |
| resize | This property used to enable the user to control the width or height or both of a specific element.(give it the power to resize an element ).  Note: to activate this property you must specify the overflow property equal one of this values(hidden, auto, scroll).  Tip: In many browsers, <textarea> is resizable by default. Here, we have used the resize property to disable the resizability:  **Note:** Outline-offset and resize property are both called user interface properties | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Horizontal | | | | | | | | | To control the width of the specific element. | | | | | | | | | | | | | | | | | Div {resize: horizontal; overflow: hidden;} | | | | | |
| Vertical | | | | | | | | | To control the height of the specific element. | | | | | | | | | | | | | | | | | Div {resize: vertical; overflow: auto;} | | | | | |
| both | | | | | | | | | To control both the height and width on the specific element. | | | | | | | | | | | | | | | | | Div {resize: both; overflow: scroll;} | | | | | |
| None | | | | | | | | | no resize | | | | | | | | | | | | | | | | | Div {resize: none;} | | | | | |
| z-index | The z-index property specifies the stack order of an element.  An element with greater stack order is always in front of an element with a lower stack order.  **Note:** z-index only works on positioned elements (position: absolute, position: relative, position: fixed, or position: sticky). | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Auto | | | | | Sets the stack order equal to its parents. This is default | | | | | | | | | | | | | | | P{z-index: auto;} | | | | | | | | | | | |
| Number | | | | | Sets the stack order of the element. Negative numbers are allowed | | | | | | | | | | | | | | | P{z-index: -2;} | | | | | | | | | | | |
| transform | Css Transform is a property allows you to transform an html element according to its pivot point.  Transform means (translate(move), rotate, scale, skew, matrix…..)  **Note:** the default pivot of any html element is (50% 50% 0px). | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| None | | | | | | | | | | | | Defines that there should be no transformation | | | | | | | | | | | | | | | | | | |  |
| matrix(n,n,n,n,n,n) | | | | | | | | | | | | This is a 2D Transforms method used to combine all the 2D transform methods into one property.  The matrix() method take six parameters, containing mathematic functions, which allows you to rotate, scale, move (translate), and skew elements.  The parameters are as follow: matrix(scaleX(),skewY(),skewX(),scaleY(),translateX(),translateY())  Note: These 6 values were written without any units.  Where the default value of matrix is → Matrix(1,0,0,1,0,0)  As :  scale(x) , scale(y) =1  scew(x), skew(y), translate(x), translate(y) = 0. | | | | | | | | | | | | | | | | | | | p{transform: matrix(0.4,1,2,0.5,2,3);} |
| matrix3d(n,n,n,n,n,n,n,n,n,n,n,n,n,n,n,n) | | | | | | | | | | | | This is 3D Transforms method | | | | | | | | | | | | | | | | | | |  |
| translateX(x) | | | | | | | | | | | | This is a 2D Transforms method Used to move an html element horizontally right and left from itscurrent position along x-axis.  Where: (+x) move the element to right; (-x) move it to left.  **Note**: (±x) in px or % | | | | | | | | | | | | | | | | | | |  |
| translateY(y) | | | | | | | | | | | | This is a 2D Transform method Used to move an html element vertically up and down from its current position along y-axis.  Where: (+y) move the element to top; (-y) move it to down.  **Note**: (±y) in px or % | | | | | | | | | | | | | | | | | | |  |
| translateZ(z) | | | | | | | | | | | | This is 3D Transforms method Used to move an html element along z-axis (toward and away from the user’s eye).  Where: (+z) move the element toward the user’s eye.  ; (-z) move the element away from the user’s eye.  **Note:**   * (±z) in px only. * This method is shorthand to {translate3d(0,0,z);}. * This method required **perspective property** to be activate (we must put a perspective property in the parent of the current element). | | | | | | | | | | | | | | | | | | | Div{perspective: 400px;}  Div P{transform: translateZ(20px);}  Div P{transform: translate3d(0,0,20px);} |
| translate(x,y) | | | | | | | | | | | | This is 2D Transforms method Used to move an html element in two dimensions from its current position along (x and y-axis together).  Where: x value means x-axis (width) & y value means y-axis (height)  **Note**:(±x) in px or % and (±y) in px or % | | | | | | | | | | | | | | | | | | |  |
| translate3d(x,y,z) | | | | | | | | | | | | This is 3D Transforms method Used to move an html element in three dimensions from its current position along (x,yand z-axis).  **Note**:   * (±x) in px or % and (±y) in px or % (±z) in px only * This method required **perspective property** to be activate (we must put a perspective property in the parent of the current element). | | | | | | | | | | | | | | | | | | |  |
| scaleX(x) | | | | | | | | | | | | This is 2D Transforms method Used to scale (stretch) an html element along x-axis (width only).  Where: value of (x) is a number [0.1:∞ [.  it may be (0.2, 0.5, 1, 1.5, 2, 2.5, 3,…….)  **Note**: if you write values of (x) with negative values the browser will consider it positive values as this property (scale) doesn’t support negative values. | | | | | | | | | | | | | | | | | | |  |
| scaleY(y) | | | | | | | | | | | | This is 2D Transforms method Used to scale (stretch) an html element in along y-axis (height only).  Where: value of (y) is a number [0.1:∞ [.  it may be (0.2, 0.5, 1, 1.5, 2, 2.5, 3,…….)  **Note**: if you write values of (y) with negative values the browser will consider it positive values as this property (scale) doesn’t support negative values. | | | | | | | | | | | | | | | | | | |  |
| scaleZ(z) | | | | | | | | | | | | This is 3D Transforms method Used to scale (stretch) an html element from its position along z-axis only (toward and away the user’s eye).  Where: (+z) move the element toward the user’s eye.  ; (-z) move the element away from the user’s eye.  Where: value of (z) is a number [0.1:∞ [.  it may be (0.2, 0.5, 1, 1.5, 2, 2.5, 3,…….)  **Note**   * if you write values of (z) with negative values the browser will consider it positive values as this property (scale) doesn’t support negative values. * This property is a shorthand to scale3d(0,0,z). * This method required **perspective property** to be activate (we must put a perspective property in the parent of the current element). | | | | | | | | | | | | | | | | | | | P{transform: scaleZ(3);}  =  P{transform: scale3d(0,0,3);} |
| scale(x,y) | | | | | | | | | | | | This is 2D Transforms method used to increase or decrease the size of an html element (according to the parameters given for the width (x) and height (y)).  Where: values of (x) and (y) are numbers [0.1:∞ [it may be (0.2, 0.5, 1, 1.5, 2, 2.5, 3,…….)  **Note**: if you write values of x or y or both with negative values the browser will consider it positive values as this property (scale) doesn’t support negative values. | | | | | | | | | | | | | | | | | | | P{transform: scale (3,2);} |
| scale3d(x,y,z) | | | | | | | | | | | | This is 3D Transforms method Used to scale (zoom) an html element in three dimensions along x,y and z-axis.  Where: x,y,z are a number (0.5, 1, 1.5, 2, 2.5, 3,….)  It doesn’t read a negative value. Like  scale (-4,-9,6) and it considered it a positive value | | | | | | | | | | | | | | | | | | | P{transform: rotate3d(0,0,20deg);} |
| rotateX(±angle) | | | | | | | | | | | | This is 3D Transforms method Used to rotate an html element from its current position along x-axis. | | | | | | | | | | | | | | | | | | | P{transform: rotateX(20deg);} |
| rotateY(±angle) | | | | | | | | | | | | This is 3D Transforms method Used to rotate an html element from its current position along y-axis. | | | | | | | | | | | | | | | | | | | P{transform: rotateY(20deg);} |
| rotateZ(±angle) | | | | | | | | | | | | This is 3D Transforms method Used to rotate an html element with clockwise and anticlockwise (toward and away the user’s eye).  Where: (+z) move toward the user’s eye.  ; (-z) move away the user’s eye.  Note:   * This property is a shorthand to rotate3d(0,0,z). * This method required **perspective property** to be activate (we must put a perspective property in the parent of the current element). | | | | | | | | | | | | | | | | | | | P{transform: rotate(20deg);}  =  P{transform: rotate3d(0,0,20deg);} |
| rotate(±angle) | | | | | | | | | | | | This is 2D Transforms method Used to rotate an html element from its current position with clockwise or anticlockwise.  Where: positive angle used to rotate the element with clockwise while the negative one used to rotate it counter clockwise.  Note: angle may be in rad, deg, turn… | | | | | | | | | | | | | | | | | | | P{transform: rotate(20deg);}  P{transform: rotate(-20deg);} |
| rotate3d(x,y,z,angle) | | | | | | | | | | | | This is 3D Transforms method; The amount of rotation created by rotate3d() is specified by three <number>s and one <angle>. The <number>s represents the x-, y-, and z-coordinates of the vector denoting the axis of rotation. The <angle> represents the angle of rotation; if positive, the movement will be clockwise; if negative, it will be counter-clockwise. | | | | | | | | | | | | | | | | | | | P{transform: rotate3d(1,1,1,20deg);} |
| skewX(±angle) | | | | | | | | | | | | This is 2D Transforms method used to skew (cause something to be not straight or exact; to twist or distort) an html element along the X-axis only by the given angle.  **Note:** x-angle may be positive or negative. | | | | | | | | | | | | | | | | | | | P{transform: skewX(20deg);}  P{transform: skewX(-20deg);} |
| skewY(angle) | | | | | | | | | | | | This is 2D Transforms method used to skew (cause something to be not straight or exact; to twist or distort) an html element along the y-axis only by the given angles.  **Note:** y-angle may be positive or negative | | | | | | | | | | | | | | | | | | | P{transform: skewY(20deg);}  P{transform: skewY(-20deg);} |
| skew(x,y) | | | | | | | | | | | | This is 2D Transforms method used to skew (cause something to be not straight or exact; to twist or distort) an html element along the X and Y-axis (both) by the given angles.  Where: x and y are positive or negative angles.  Note:   * if you don’t write y-angle then the browser will consider the value of y-angle equal zero so the element will skew along x-axis only.   Note : in this case → skew(x) = skewX(x) | | | | | | | | | | | | | | | | | | | P{transform: skew(30deg,40deg);}  P{transform: skew(30deg);} |
| Backface-visibility | This property is used to hide and show the back face (side) of a container element | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| visible | | | | | | | | | | | | Used to show the back face of a block element (default value). | | | | | | | | | | | | | | | | | | | Div {backface-visibility: visible;} |
| hidden | | | | | | | | | | | | Used to hide the back face of a block element. | | | | | | | | | | | | | | | | | | | Div {backface-visibility: hidden;} |
| Transform-origin | This property is used to change the position of the **pivot point** of an html element.  **Note:** pivot: is an imaginary **fixed** point used to transform an html element in different ways.  **Fixed:** means that when you look at this point during the transform of the element you can notice that this point doesn’t change its coordinate.  Syntax:  Transform-origin: x-offset y-offset z-offset;  Where:  x-offset can be in pixels,%, left, center, right  y-offset can be in pixels,%, top, center, bottom  z-offset can be in pixels.  remember that:  Left, top= 0%, right, bottom =100%, center=50%  **Note:**   * By default pivot point is placed at **the center** of an html element {transform-origin= center center 0px} or {transform-origin= 50% 50% 0px;}. * Left value means that pivot fixed point locate at the middle of left rib (side) of the element.     **Note:**  the above is standard values you can use any other values you need (specific values) by using percentage. Ex1; {transform-origin:30% 80% 20px;} Ex2:{transform-origin:30% 30% 40px;} | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Transform-style | This property sets whether children of an element are positioned in the 3D space or are flattened in the plane of the element.  Note:   * This property is applied only on the Childs of a specific parent. * This property must be used together with the transform property. | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Flat | | | | | | | | | | | | Indicates that the children of the element are lying in the plane of the element itself. This is default | | | | | | | | | | | | | | | | | | |  |
| preserve-3d | | | | | | | | | | | | Indicates that the children of the element should be positioned in the 3D-space. | | | | | | | | | | | | | | | | | | |  |
| perspective(n) | This property defines how far the element is away from the user eye.  is used to give 3D space to the parent element with 3d transform properties.  Note:   1. This property only used with the elements that Childs have a 3d-trnsform properties (i.e. rotate(x): | rotate (z): ….) 2. The lower value means more 3d effect as the element will be closer to user’s eyes. 3. The higher value means less 3d effect as the element will be farther to user’s eyes. 4. When defining the perspective property for a specific parent, its child elements that get the perspective view, not the parent itself. | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| None | | | | | | | | | | | | This value means that there is no perspective (default value) | | | | | | | | | | | | | | Parent {perspective: none;} | | | | | |
| Length | percentage | | | | | | | | | | | |  | | | | | | | | | | | | | | Parent {perspective: 300px;} | | | | | |
| perspective-origin | The perspective-origin property defines at from which position the user is looking at the 3D-positioned element.  Note:   * When defining the perspective-origin property for an element, it is the CHILD elements that will get the effect, NOT the element itself. (like perspective) * This property must be used in conjunction with the perspective property! As it do nothing by itself | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| x-axis Defining where the view is placed at the x-axis  Possible values:  left  center  right  length  %  Default value: 50% | | | | | | | | | | | |  | | | | | | | | | | | | | | Div{perspective-origin: left;} | | | | | |
| Defining where the view is placed at the y-axis  Possible values:  top  center  bottom  length  %  Default value: 50% | | | | | | | | | | | |  | | | | | | | | | | | | | | Div{perspective-origin: top;} | | | | | |
| transition | This property is used to change property values of a specific html element **smoothly**, over a given duration.  If you have two different values for the same element one of these values (first value) was written in the style code of a specific element but the other one(second value) on a :hover property for the same element; what you notice that by default the transition between the first value and the second value will be immediately as soon as mouses over the element to prevent that (sudden transition you use transition property).  **Note:**   * To add this property to a specific element you should:   + - * 1. Write the first and second values normally.         2. Put the transition property with the first value * Transition property is shorthand for the next properties.   selector{Transition :transition-property(required) transition-duration(required) transition-timing-function(optional) transition-delay(optional);}  **Note:**   * If the duration part is not specified, the transition will have no effect, because the default value is 0. * When the cursor mouses out of the element, it will gradually change back to its original style. * If you want to add transition for several css property then you must use comma to separate between them. EX: div{transition: width 2s, height 4s;} | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Transition-property | This property specifies the property that you want to add transition to it.  **Note:**   * This property is required. * It’s very necessary to write transition-duration value with transition-property to activate it. * If you write a shorthand property for a specific element then all sub-properties for this element will affected by this transition property. * You can write several css properties separated by comma. | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| All | | | | | | | | | | | | Means that transition property will apply on **all** properties of a specific html element. | | | | | | | | | | | | | | P {Transition-property: all; transition-duration: 2s} | | | | | |
| None | | | | | | | | | | | | Means that transition property has no effect in a specific element | | | | | | | | | | | | | | P {Transition-property: none; transition-duration : 0s ;} | | | | | |
| Property name | | | | | | | | | | | | Means that transition property will apply on ***specific*** properties of a specific html element. | | | | | | | | | | | | | | P {Transition-property: opacity, height, color; transition-duration :3s,1.3s, 2s ;} | | | | | |
| transition-duration | This property is used to specify the time that the element will take to transfer from the first value to the second one.  Note:   * This property is required. * It’s very necessary to write this property with transition-property to activate the later. * You can write several transition-duration for different transition-properties separated by comma. Where each transition-duration related to a transition-property respectively. | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| X s or x ms | | | | | | | | | | | | Time can be in seconds or in millisecond  where 1s=1000ms | | | | | | | | | | | | | | P {Transition-property: width, opacity, color; transition-duration : 2000ms, 3s, 4s ;} | | | | | |
| Transition-timing-function | This property used to control the speed curve of the effect of transition property from the start point to the end point. (This property allows a transition effect to change speed over its duration) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Ease | | | | | | | | | | | | Default value. Specifies a transition effect with a slow start, then fast, then end slowly (equivalent to cubic-bezier(0.25,0.1,0.25,1)) | | | | | | | | | | | | | | P{transition-timing-function: ease;} | | | | | |
| Linear | | | | | | | | | | | | Specifies a transition effect with the same speed from start to end (equivalent to cubic-bezier(0,0,1,1)) | | | | | | | | | | | | | | P{transition-timing-function:linear;} | | | | | |
| Ease-in | | | | | | | | | | | | Specifies a transition effect with a slow start (equivalent to cubic-bezier(0.42,0,1,1)) | | | | | | | | | | | | | |  | | | | | |
| Ease-out | | | | | | | | | | | | Specifies a transition effect with a slow end (equivalent to cubic-bezier(0,0,0.58,1)) | | | | | | | | | | | | | |  | | | | | |
| ease-in-out | | | | | | | | | | | | Specifies a transition effect with a slow start and end (equivalent to cubic-bezier(0.42,0,0.58,1)) | | | | | | | | | | | | | |  | | | | | |
| step-start | | | | | | | | | | | | Equivalent to steps(1, start) | | | | | | | | | | | | | |  | | | | | |
| step-end | | | | | | | | | | | | Equivalent to steps(1, end) | | | | | | | | | | | | | |  | | | | | |
| steps(Number,start|end) | | | | | | | | | | | | Specifies a stepping function, with two parameters. The first parameter specifies the number of intervals(steps) in the function. It must be a positive integer (greater than 0). The second parameter, which is optional, is either the value "start" or "end", and specifies the point at which the change of values occur within the interval. If the second parameter is omitted, it is given the value "end"  **Note**: if the second parameter is:  - Start : that means the ***changing*** in the element from one value to another occurs ***Coinciding*** with the starting of steps.  - End: means that the ***changing*** in the element from one value to another occurs ***after*** the starting of steps. | | | | | | | | | | | | | | P{transition-timing-function:steps(5,start);} | | | | | |
| cubic-bezier(n,n,n,n) | | | | | | | | | | | | Define your own values in the cubic-bezier function. Possible values are numeric values from 0 to 1.  Use css3 Bezier Curve Tester.com website to help you. | | | | | | | | | | | | | |  | | | | | |
| transition-delay | This property specifies when the transition effect will start. | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| x s or x ms | | | | | | | | | | | | The transition-delay value is defined in seconds (s) or milliseconds (ms). | | | | | | | | | | | | | | | P{transition-delay: 3s} | | | | |
| Animation | It’s a CSS property allows you to animate(give life for) HTML elements without using JavaScript or Flash!. Where An animation lets an element gradually change from one style to another. And You can change as many CSS properties you want, as many times you want.  To give animate to any html element you should follow these3 steps:   * 1. Write this word **@keyframes.**   2. Write animation name beside @keyframes. Tip   3. Open curly brackets.   4. Write the steps of animation.   These steps may be only **two selectors (steps)** like; from{ style properties } to{style properties} or **several selectors (steps)** like; 0%{ style properties } 10%{ style properties } 38%{ style properties }. **Note:**   * + - * 1. From = 0% & to=100%         2. In case of percentage if you have two selectors (steps) with the same style properties you can combine it together separated by comma.   EX; 0%{color: red;} 50{color: black;} 100%{color: red;} then you can combine it like this: 0%,100%{color: red;} 50%{color: black;}  **Note**: all steps above wasn’t written inside the style of a specific html element . like this; ~~div{@keyframes coloring}~~ but like this : @keyframes coloring.   * 1. Write css decelerations you want inside each step. (required)   Then write the next steps **inside** the element you want to animate.   * 1. Animation-name (required)   2. Animation-duration: xs|ms (required)   3. Animation-delay: xs|ms **Note:** negative values are allowed   4. Animation-iteration-count: Number.   5. Animation-direction   6. Animation-timing-function.   7. Animation-fill-mode.   Note: this property Is a shorthand property:  Animation: {animation-name animation-duration animation-timing-function animation-delay animation-iteration-count animation-direction animation-play-state;}  Note: default value of animation shorthand is {animation: none 0 ease 0 1 none none running}  Example: **div { animation: example 5s linear 2s infinite alternate running; }**  We will illustrate each property in details. | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| \  Animation name | This is a name from your mind (any name). | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| None (default value) | | | | | | | | | | | | To cancel animation of a specific element. | | | | | | | | | | | | | | Animation-name: none; | | | | | |
| Any name | | | | | | | | | | | | it’s better to use name point to the function of this animation. | | | | | | | | | | | | | | Animation-name: coloring; | | | | | |
| Animation-duration | This property defines how long time an animation should take to complete. If the animation-duration property is not specified, no animation will occur, because the default value is 0s (0 seconds). | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| xs | xms | | | | | | | | | | | | - | | | | | | | | | | | | | | Animation-duration: 3s; | | | | | |
| animation-iteration-count | This property specifies the number of times an animation should run. | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| infinite | | | | | | | | | | | | Used to make the animation continue for ever | | | | | | | | | | | | | | |  | | | | |
| Numbers | | | | | | | | | | | | Default value is 1 | | | | | | | | | | | | | | | Animation-iteration-count: 3; | | | | |
| Animation-direction | This property specifies whether an animation should be played forwards, backwards or in alternate cycles (direction of the animation) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| normal | | | | | | | | | | | | The animation is played as normal (forwards). This is default | | | | | | | | | | | | | | | Animation-direction: normal; | | | | |
| reverse | | | | | | | | | | | | The animation is played in reverse direction (backwards) | | | | | | | | | | | | | | | Animation-direction: reverse; | | | | |
| Alternate | | | | | | | | | | | | The animation is played forwards first, then backwards | | | | | | | | | | | | | | | Animation-direction: alternate; | | | | |
| alternate-reverse | | | | | | | | | | | | The animation is played backwards first, then forwards | | | | | | | | | | | | | | | Animation-direction: alternaye-reverse; | | | | |
| animation-timing-function |  | | | | | | | | | | | |  | | | | | | | | | | | | | | |  | | | | |
| ease | | | | | | | | | | | | Specifies an animation with a slow start, then fast, then end slowly (this is default) | | | | | | | | | | | | | | | Animation-timing-function: ease; | | | | |
| linear | | | | | | | | | | | | Specifies an animation with the same speed from start to end | | | | | | | | | | | | | | | Animation-timing-function: linear; | | | | |
| Ease-in | | | | | | | | | | | | Specifies an animation with a slow start | | | | | | | | | | | | | | | Animation-timing-function: ease-in; | | | | |
| Ease-out | | | | | | | | | | | | Specifies an animation with a slow end | | | | | | | | | | | | | | | Animation-timing-function: ease-out; | | | | |
| ease-in-out | | | | | | | | | | | | Specifies an animation with a slow start and end | | | | | | | | | | | | | | | Animation-timing-function: ease-in-out; | | | | |
| Cubic-bezier(n,n,n,n,) | | | | | | | | | | | | Lets you define your own values in a cubic-bezier function | | | | | | | | | | | | | | | Animation-timing-function: cubic-bezier(1,1,1,.3); | | | | |
|  | | | | | | | | | | | |  | | | | | | | | | | | | | | |  | | | | |
| Animation-delay | This property specifies a delay for the start of an animation.  **Note**: Negative values are also allowed. If using negative values, the animation will start as if it had already been playing for N seconds. | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| xs |xms | | | | | | | | | | | | The default value is 0s= 0ms | | | | | | | | | | | | | | Animation-delay: 3s; | | | | | |
| animation-fill-mode | The animation-fill-mode property specifies a style for the target element when the animation is not playing (before it starts, after it ends, or both).  **Note:** CSS animations do not affect an element before the first keyframe is played or after the last keyframe is played. The animation-fill-mode property can override this behavior. | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| None | | | | | | | | | | | | Default value. Animation will not apply any styles to the element before or after it is executing | | | | | | | | | | | | | | animation-fill-mode: none; | | | | | |
| Forwards | | | | | | | | | | | | The element will retain to the style values of the last keyframe (100%{} or to{}) | | | | | | | | | | | | | | animation-fill-mode: forwards; | | | | | |
| Backwards | | | | | | | | | | | | The element will get the style values that is set by the first keyframe (0%{} or from{}), and retain this during the animation-delay period. | | | | | | | | | | | | | | animation-fill-mode: backwards; | | | | | |
| both | | | | | | | | | | | | The animation will follow the rules for both forwards and backwards, extending the animation properties in both directions | | | | | | | | | | | | | | animation-fill-mode: both; | | | | | |
| Animation-play-state | This property used to control the state of the animation | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Running | | | | | | | | | | | | Used to run the animation (default value) | | | | | | | | | | | | | | {Animation-play-state: running;} | | | | | |
| paused | | | | | | | | | | | | Used to stop the animation (its more useful with js or j query ) | | | | | | | | | | | | | | {Animation-play-state: paused;} | | | | | |
| Scroll-behavior | This property is used to add a smooth scrolling effect to the document: | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| smooth | | | | | | | | | | | | Allows a smooth animated "scroll effect" between elements within the scrolling box | | | | | | | | | | | | | | html { scroll-behavior: smooth;} | | | | | |
| auto | | | | | | | | | | | | Allows a straight jump "scroll effect" between elements within the scrolling box. This is default | | | | | | | | | | | | | | Body{scroll-behavior: auto;} | | | | | |
| Tab-size | This property used to set the width of the tab character .  Note: In HTML, the tab character is usually displayed as a **single space-character**, except for some elements, like <textarea> and <pre>, and the result of the tab-size property will only be visible for those elements. So to specify the width of the tab button use this property.  Note: no browser is currently supporting the length unit values. | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Number without any values | | | | | | | | | | | | 8 is the default value. | | | | | | | | | | | | | | Pre {tab-size:16;} | p{tab-size: 8;} | | | | | |
| Table-layout | The table-layout CSS property sets the algorithm used to lay out <table> cells, rows, and columns. | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| auto | | | | | | | | | | | | By default, most browsers use an automatic table layout algorithm. The widths of the table and its cells are adjusted to fit the content. | | | | | | | | | | | | | | Table,th,td{table-layout: auto; width:100px;} | | | | | |
| fixed | | | | | | | | | | | | Table and column widths are set by the widths of table and col elements or by the width of the first row of cells. Cells in subsequent rows do not affect column widths.  Under the "fixed" layout method, the entire table can be rendered once the first table row has been downloaded and analyzed. This can speed up rendering time over the "automatic" layout method, but subsequent cell content might not fit in the column widths provided. Cells use the overflow property to determine whether to clip any overflowing content, but only if the table has a known width; otherwise, they won't overflow the cells. | | | | | | | | | | | | | | Table,th,td{table-layout: fixed; width:100px;} | | | | | |
| Empty-cells | The empty-cells property sets whether or not to display borders on empty cells in a table.  Note: This property has no effect if border-collapse is "collapse". | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Hide | | | | | | | | | | | | Hide borders on empty cells | | | | | | | | | | | | | | table { empty-cells: hide;} | | | | | |
| show | | | | | | | | | | | | Display borders on empty cells. This is default | | | | | | | | | | | | | | table { empty-cells: show;} | | | | | |
| Box-sizing | As we know before in box-model that the total width of an element calculated as :  Width of content + left and right padding + left and right border = actual (total) width of an element.  And the height of an element calculated as:  height of content + top and bottom padding + top and bottom border = actual (total) height of an element  That means the width property that you write in css property of this element for only the content. And you added border or padding or both to this element this element look bigger.  ( And you added border or padding or both to this element this element look bigger.).  To solve this problem we need to contain the width/height of the border and padding to content width/height of the element. This actually what does box-sizing do!  Tip: apply box-sizing: border-box; for all your pages by using Global selector. → \*{box-sizing: border-box;} | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| content-box | | | | | | | | | | | | Means that the width properties you write contain only the content-width of the element. And when you added border or padding or both to this element this element look bigger. (And you added border or padding or both to this element this element look bigger.) | | | | | | | | | | | | | | div{ box-sizing: border-box; } | | | | | |
| border-box | | | | | | | | | | | | By applying this property the width/height that you specify for element contain the width/height of (border+ padding).  EX; div{width: 300px;height: 200px; border: 3px solid red; padding: 30px; box-sizing: border-box;}.  Means the total width of the div equal to 300px  6px (left + right borders) + 60px (left + right padding) + content-width = 300px.  Then content-width= 300-66=234.  And the total height equal:  6px (top + bottom borders) + 60px (top + bottom padding) + content-height = 200px.  Then the content-height= 200-66=134. | | | | | | | | | | | | | | \*{box-sizing: content-box;} | | | | | |
| Box-decoration-break | The box-decoration-break property specifies how the background, padding, border, border-image, box-shadow, margin, and clip-path of an element is applied when the box for the element is fragmented (broken) in multiply lines.  **Note:** it applied in inline elements because these elements which can be fragmented (block elements don’t broken) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Slice | | | | | | | | | | | | Box decorations are applied to the element as a whole and break at the edges of the element fragments. (default value) | | | | | | | | | | | | | | Span { border: 3px solid red; Box-decoration-break: slice;} | | | | | |
| clone | | | | | | | | | | | | Box decorations apply to each fragment of the element as if the fragments were individual elements. Borders wrap the four edges of each fragment of the element, and backgrounds are redrawn in full for each fragment. | | | | | | | | | | | | | | Span { border: 3px solid red;Box-decoration-break: clone;} | | | | | |
| Var() | The var() function used to insert specific value to any number of properties you need.  It consists of (custom property, fallback value).  Where:   * Custom property any suitable name start with 2 dashed. Like: (-- custom property). * Fallback value is an alternative value works only if –custom property is omitted (doesn’t work for any reason).   **Note:** var() function may be local scope or global scope.  **Where:**   * Local scope means that function will apply in the Childs of a specific parent.(div selector) * Global scope means that function will apply in all html elements. (:root or body selector)   To use this property follow these steps:   * 1. In global or local selector write the properties that you want to apply in a specific html element.   2. Write --custom property inside the global or local selector. :root{custom property : #ff0;}   3. In the child element you want to apply this function write:   Css property: var(-- custom property, fallback value ).  **Note:**  You can write custom property inside <html> tag in html page like this:  <html lang=”en” style= “—custom property: value;”> | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| (custom-property, fallback value) | | | | | | | | | | | **-** | | | | | | | :root {--akram: yellow;}  Div{ color: var(--akram, ornge);}  Another example:  Body{--background-color: black;}  P{background-color: var(--background-color, alt value);} | | | | | | | | | | | | | |
| Counter | This css property allow you to numerate any html elements dynamically (automatic numbering) in addition to styling this numerte.   * To add a counter to a specific elements follow these steps:   **In the parent element**   * 1. add counter-reset property. (note : you can use several counter reset properties separated by space)   **In the elements you want to numerate**   * 1. Use ::before pseudo-class.   2. Write counter-increment property.   3. Write content property. | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Counter-reset | Suitable name counter-reset number | | | | | | | | | | | | | | | | | In this property we write any suitable name and then add a number.  **Note**:   1. This property was written in the parent element of the elements you want to numerate. 2. This name just an id for the elements that you want to add numbers to it [this name doesn’t appear in the browse]. 3. This number tells the browser from which number should the numbering start. If you don’t write any number then it will be 0 (default ). (negative values are available) | | | | div  {  counter-reset: akram 2;  }  body div p::before  {  counter-increment: akram ;  content: “-” counter(akram upper-roman);  } | | | | | | | | | | |
| Counter-increment | Counter-reset name number | | | | | | | | | | | | | | | | | In this property we wrote the counter-reset name *without its number*.  **Note:**   * This property is used to specify the mount of step jump in numbering (default is 1). | | | |
| content | “additional text” Counter(counter-reset name , style) | | | | | | | | | | | | | | | | | In this property we add counter-reset name without its number as well as the numbering style (disc, decimal, upper alpha, lower alpha, upper roman, lower roman….)  **Note:**   * Numbering style is [optional] as the default style is ***numbers***. * Note: additional text also optional. | | | |

**CSS Grid Layout (شبكة تخطيط الصفحة)**

* CSS Grid Layout is a method used to divide an html page into major regions based on columns and rows terms. Like: table.
* You can apply Css grid layout on any block element like: html, div, section, aside……
* As we know in css3 the elements divided into parents and Childs; in css grid the parent element called container and the child element called item.

**Note:** parent = grid container and child= grid item. There is no different.

* 
* There are grid properties for the container elements and there are different grid properties for the item elements as shown in the below table.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Grid container properties** | | | | | | | |
| Display | When you give display property to the container element with a value of grid or inline-grid. All direct children of grid containers become **grid items automatically.** | | | | | | |
| Grid | Means that all Grid items of this container are   * 1. Placed in rows by default.   2. Span the full width of the grid container. | | | Html{display: grid;} | | |
| inline-grid | Means that all Grid items of this container are   * 1. Placed in rows by default.   2. Span the full width of the content of the item. | | | Div{display: inline-grid} | | |
| Grid-template-column | This property is used to divide your container element to number of columns (through determining the width of these columns).  **Note:** each width in this property expresses one column. [Number of width = Number of columns]  div{Grid-template-column: width of the 1st column width of the 2nd column width of the 3rd column width of the 4th column width of the 5th column width of the 6th column }  Means the div is divided into 6th columns. | | | | | | |
| Auto Auto Auto Auto…… | | | Means that all rows should have the same height.(default) | | { grid-template-rows: Auto auto auto auto auto;} | |
| Lenght1, Lenght2, Lenght3, Lenght4,…. | | | To specify width for each row. | | {grid-template-rows: 80px 200px auto 40px;} | |
| Percentage | | | Is a non-negative <percentage> value relative to the inline size of the grid container. If the size of the grid container depends on the size of its tracks, then the percentage must be treated as auto.  The intrinsic size contributions of the track may be adjusted to the size of the grid container and increase the final size of the track by the minimum amount that would result in honoring the percentage. | |  | |
| Flex | | | Is a non-negative dimension with the unit fr specifying the track’s flex factor. Each <flex>-sized track takes a share of the remaining space in proportion to its flex factor.  When appearing outside a minmax() notation, it implies an automatic minimum (i.e. minmax(auto, <flex>)). | | {grid-template-rows: 1fr 1fr 3fr 1fr 2fr ;} | |
| max-content | | | Is a keyword representing the largest maximal content contribution of the grid items occupying the grid track. | |  | |
| min-content | | | Is a keyword representing the largest minimal content contribution of the grid items occupying the grid track. | |  | |
| fit-content( [ <length> | <percentage> ] ) | | | Represents the formula min(max-content, max(auto, argument)), which is calculated similar to auto (i.e. minmax(auto, max-content)), except that the track size is clamped at argument if it is greater than the auto minimum. | |  | |
| repeat( [ <positive-integer> | auto-fill | auto-fit ] , <track-list> ) | | | Represents a repeated fragment of the track list, allowing a large number of columns that exhibit a recurring pattern to be written in a more compact form. | |  | |
| Subgrid | | | The subgrid value indicates that the grid will adopt the spanned portion of its parent grid in that axis. Rather than being specified explicitly, the sizes of the grid rows/columns will be taken from the parent grid’s definition.  **Note:** The subgrid value is from Level 2 of the Grid specification and currently only has implementation in Firefox 71 and onwards. | | {grid-template-rows: repeat(6,2fr);} | |
| Grid-template-rows | This property is used to add row track through determined the height of each track row.  Ex: div {Grid-template-rows: height of the 1st row height of the 2nd row height of the 3rd row height of the 4th row height of the 5th row;}  This example Means that div consist of five track rows.  **Note:**   * Height of each row is separated than the other one by a space. * Height of each grid item was determined by its content (default value) so if you have 6 grid items and specify the height of only 3 items then the 2 other grid items height will take the default value based on its content. | | | | | | |
| None | | | Specify no grid | | | div{grid-template-rows: none} |
| Auto | | | Height of row depend on the content of the grid item (default)  **Note:** auto value = 1.8rem=1.8em=28px | | | { grid-template-rows: Auto auto auto auto auto;} |
| Length | | | To specify a fixed (absolute) height for each track row.  **Note:** Negative values aren’t allowed. | | | {grid-template-rows: 80px 200px auto auto;} |
| Percentage | | | To specify a relative height for each track row.  **Note:** Negative values aren’t allowed. | | | {grid-template-rows: 10% 30% auto 100px;} |
| Flex | | | The <flex> is data type consist of a <number> followed by the unit fr. Like: 3fr.  **Note:**   * This number may be   + - * 1. float number(has decimal point) → 2.5 or 3.1         2. integer number (hasn’t decimal point)→ 2 or 3 * 1fr = auto (default value)=28px=1.8rem=1.8em   So you can use 2fr, 3fr, 4fr,…. To determine the The default value (auto) and its multiples.  EX; 3fr= (3\*28px=84px) =(3\*1.8=5.4rem=5.4em) | | | {grid-template-rows: 1fr 2fr 3fr 1fr 2fr ;} |
| max-content | | | Is a keyword representing the largest maximal content contribution of the grid items occupying the grid track. | | |  |
| min-content | | | Is a keyword representing the largest minimal content contribution of the grid items occupying the grid track. | | |  |
| Minmax(min value, max value) | | | Used to specify the minimum and maximum height of the row track. | | | {grid-template-rows: minmax(30%, auto);} |
| fit-content( [ <length> | <percentage> ] ) | | | Represents the formula min(max-content, max(auto, argument)), which is calculated similar to auto (i.e. minmax(auto, max-content)), except that the track size is clamped at argument if it is greater than the auto minimum. | | |  |
| repeat( positive-integer | auto-fill | auto-fit , track-height ) | | | Used to apply a specific height to multiply track rows.  **Note:** repeat() can also be used within track listings. | | | div{grid-template-rows: repeat(,2fr);}  div{grid-template-rows: 30px repeat(3, 40%) 40px;} |
| subgrid | | | The subgrid value indicates that the grid will adopt the spanned portion of its parent grid in that axis. Rather than being specified explicitly, the sizes of the grid rows/columns will be taken from the parent grid’s definition.  **Note:** The subgrid value is from Level 2 of the Grid specification and currently only has implementation in Firefox 71 and onwards. | | | {grid-template-rows: repeat(6,2fr);} |
| grid-template-area |  | | |  | | |  |
| None | | |  | | |  |
| Specific name | | |  | | |  |
| grid-template | The grid-template CSS property is a shorthand property for defining grid columns, rows, and areas. | | | | | | |
| None | | |  | |  | |
| <'grid-template-rows'> / <'grid-template-columns'> | | |  | |  | |
| [ <line-names>? <string> <track-size>? <line-names>? ]+ [ / <explicit-track-list> ]? | | |  | |  | |
| Grid-auto-rows | minmax(min, max) | | | Is a functional notation that defines a size range greater than or equal to min and less than or equal to max. If max is smaller than min, then max is ignored and the function is treated as min. As a maximum, a <flex> value sets the track’s flex factor. It is invalid as a minimum. | | {grid-auto-rows: minmax(200px, auto);} | |
| grid-auto-columns |  | | |  | |  | |
| grid-auto-flow |  | | |  | |  | |
| grid-template-areas |  | | |  | |  | |
| Grid |  | | |  | |  | |
| Gird-gap | Grid Gaps also called (Gutters)  It’s the space (gaps) between columns and rows.  **Note:**   * Grid gaps are only created in between columns and rows, and not along the edge of the grid container. * Grid gap is a shorthand property for grid-row-gap and grid-column-gap respectively. * One value sets equal row and column gaps. * Grid gap was replaced by gap. * Default value of gap is 0px. | | | | | | |
| Length, % | | | To specify the (gap) space between the rows and columns respectively. | | {grid- gap: 20px 30px;}  {grid- gap: 20px;} | |
| Calc(value1±vlaue2) | | |  | | div{gap: calc(10% + 20px);}  body{gap: calc(20px + 10%) calc(10% - 5px);} | |
| Grid-row-gap | Length, % | | | To specify the (gap) space between the rows. | | {grid-column-gap: 20px;} | |
| Grid-column-gap | Length, % | | | To specify the (gap) space between the columns. | | {grid-row-gap: 20px;} | |
| Justify-items | Used to control the position of an items horizontally | | | | | | |
| Left Or Start | | |  | | .container{ Justify-items : start;} | |
| Center | | |  | | .container{ Justify-items : center;} | |
| Right or end | | |  | | .container{ Justify-items : end;} | |
| Align-items | Used to control the position of items vertically. | | | | | | |
| Start or top | | |  | |  | |
| center | | |  | |  | |
| End or bottom | | |  | |  | |
| Baseline | | |  | |  | |
| Flex-end | | |  | |  | |
| Flex-start | | |  | |  | |
| strech | | |  | |  | |
| **Grid item layout** | | | | | | | |
| grid-column-start | Number of the start column | | Define the start column of a specific grid item | | .item1 { grid-column-start: 3;} | | |
| grid-column-end | Number of the End column | | Define the end column of a specific grid item | | .item1 { grid-column-end:5;} | | |
| grid-column | Number of the start column / Number of the End column | | Defines numbers of vertical columns should item take. | | .item1 { grid-column: 3/5;} | | |
| grid-row-start | Start row-line number | | Define the start row of a specific item | | .item1 { grid-row-start: 3;} | | |
| grid-row-end | End row-line numbers | | Define the end row of a specific item | | .item1 { grid-row-wnd:5;} | | |
| grid-row | Start row-line number/end row-line number | | Defines numbers of rows should item take. | | .item1 { grid-row: 3/ 5;} | | |
| grid-area | Grid-area is shorthand for the next properties;  body{Grid-area: grid-row-start grid-column-start grid-row-end grid-column-end;}  ex: .item{grid-area: 2/3/5/6;} | | | | | | |
| name | | To use grid area name instead of numbers you should follow these steps:   * Inside a specific child write a suitable name for its grid area → .child {grid-area: myarea}. * Inside the container (parent) of this child write → .container {grid-template-areas: ‘myarea myarea myarea myarea’ ;}.   Notice:   * The number of myarea = the number of columns. * Using of single quotes with grid-template-areas. * A period sign(.) represents a grid item with no name | | .item1 { grid-area: header; }  .item2 { grid-area: menu; }  .item3 { grid-area: main; }  .item4 { grid-area: right; }  .item5 { grid-area: footer; }  .grid-container {  grid-template-areas:  'header header header header header header'  'menu main main main right right'  'menu footer footer footer footer footer'; | | |
| span | | This keyword is used to express the number of rows or columns should item expand through.  We can use span instead of the value of :   1. grid-row-start 2. grid-column-start 3. grid-row-end 4. grid-column-end | | div{grid-column: span 2;}  div{grid-column: 2/span 2;}  div{grid-column: 3/span 5;} | | |

**Flex box layout (Flexible Box Module)**

Flex: means flexible.

Box: means block element.

So those flex box means a flexible html block element.

Definition: flex box is a css3 property designed to solve specific problems in <float> and <position> properties.

What are these problems?

Problems of <float> and <position> properties are:

* 1. Vertically centering Childs of specific parent (parent with <float> property) inside it.
  2. Making all the children of a specific parent (container) take up an equal amount of the available width/height, regardless of how much width/height is available.
  3. Making all columns in a multiple column layout adopt the same height even if they contain a different amount of content.
  4. changing the layout order of flex items, without affecting the source order. It’s possible by order property

Flex box solving all of these problems as we illustrate next.

To use flex box property we have to deal with a container element and its item elements as each one has a specific flex properties.

Let’s start with the container:

There are some important terms related with container:

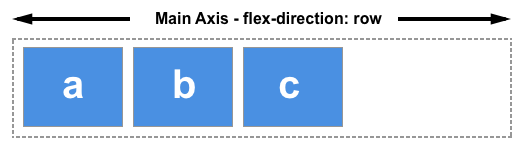
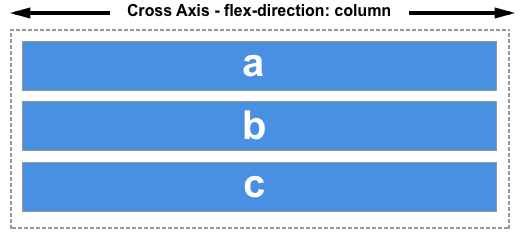
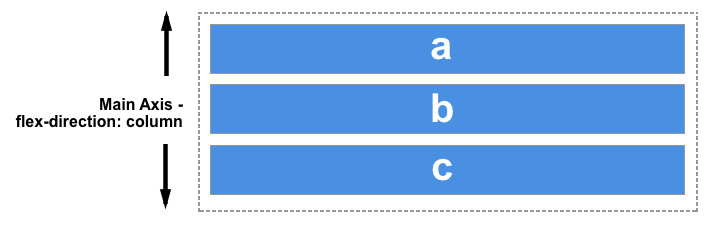
1. Main axis
2. Cross axis
3. Start line (start edge)
4. End line (end edge)

The main and the cross axis are perpendicular to each other so if you know the direction of one axis you will know the other direction.

We can determine the direction of the main axis by another property called flex-direction. This property has 4 values (row [default value], row-reverse, and column, column-reverse).

So if you determine flex-direction: row; or flex-direction: row-reverse; the direction of main axis will be horizontally and the direction of cross axis will be vertically. but if you determine flex-direction: column; or flex-direction: column-reverse;; the direction of main axis will be vertically and the direction of cross axis will be horizontally.

These are some picture to illustrate that:

(fig1) (fig2) (fig3) (fig4)

(fig1) shows that when the flex-direction property of the container is row or row-reverse, the main axis will run horizontally along the row in the inline direction. So the cross axis will be vertically.

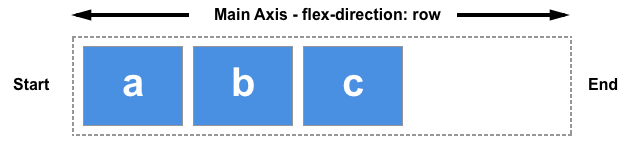
(fig2) shows the direction of cross axis of (fig1)

(fig3) shows that when the flex-direction property of the container is column or column-reverse, the main axis will run from the top of the page to the bottom — in the block direction. So the cross axis will be horizontally.

(fig4) shows the direction of cross axis of (fig3).

We will know the importance of these axes in aligning and justifying flex properties later.

Each axis has a start (edge) and end (edge) ; that can be illustrate with these pictures:

(fig1) (fig2)

* (fig1) shows if the flex-direction is row and I am working in English, then the start edge of the main axis will be on the left, the end edge on the right.
* (fig2) shows if the flex-direction is row and I am working in Arabic, then the start edge of the main axis will be on the right and the end edge on the left.

Note:

In both cases the start edge of the cross axis is at the top of the flex container and the end edge at the bottom, as both languages have a horizontal writing mode.

**The flex container**

Flex container is a normal container with display: flex or display: inline-flex property. As soon as we do this the container become **flex** **container** direct children of that container become **flex items**. As soon as applying the property of display: flex or display: inline-flex some initial (default) values are defined to the flex container and flex items, and they will behave in the following way:

1. Items display in a row (the flex-direction property's default is row).
2. The items start from the start edge of the main axis. (Depending on the type of language En or AR).
3. The items do not stretch on the main dimension, but can shrink.
4. The items will stretch to fill the size of the cross axis (all items have the same height).
5. The [flex-wrap](https://developer.mozilla.org/en-US/docs/Web/CSS/flex-wrap) property is set to nowrap. As all items in the container will locate in one row line (default), using the size of the container as their size in the main axis. If there are more items than can fit in the container, they will not wrap but will instead overflow (decrease the width and increase the height). If some items became taller than others, all items will stretch along the cross axis to fill its full size. So all items became equals in height and width.
6. The [flex-basis](https://developer.mozilla.org/en-US/docs/Web/CSS/flex-basis) property is set to auto.

To change these default values of **flex-container** we use the next properties:

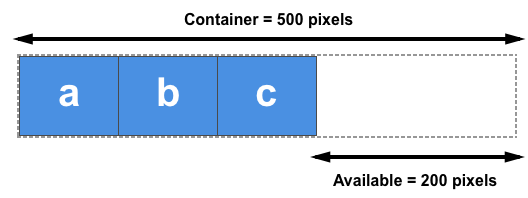
1. Flex-direction
2. Flex-wrap
3. Flex-flow
4. justify-content
5. align-items
6. align-content

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Flex-direction | The flex-direction property defines in which direction container should position its flex items. (horizontally or vertically) | | | |
| row | | The row value position the flex items horizontally ( where the start and end lines are switched depending on the language; En or AR)  if language is AR (direction is rtl) then the Start edge will be in left hand and the end edge in the right hand.  If the language is En (direction is ltr) then the Start edge will be in right hand and the end edge in the left hand. | .flex-container{display: flex; flex-direction: row; } |
| Row-reverse | | The row-reverse value positions the flex items horizontally (where the start and end lines are switched depending on the language; En or AR)  if language is AR (direction is rtl) then the Start edge will be in left hand and the end edge in the right hand.  If the language is En (direction is rtl) then the Start edge will be in right hand and the end edge in the left hand. | .flex-container{display: flex; flex-direction: row-reverse; } |
| Column | | The column value position the flex items vertically from top to bottom (where the main axis switches and our items now display in a column).  Note: this property doesn’t depend on the language s in both languages the flow will vertically from top to down. | .flex-container{display: flex; flex-direction: column; } |
| Column-reverse | | The column-reverse value positions the flex items vertically (but from bottom to top).  Note: this property doesn’t depend on the language s in both languages the flow will vertically from top to down. | .flex-container{display: flex; flex-direction: column-reverse; } |
| Flex-wrap | The flex-wrap property specifies whether the flex items should wrap in to line or not. (this property is affected when the flex-direction property is row or row-reverse)  **Note:** wrap happening when the total width of the items being too wide for the flex container then it begin to wrap in the next line. | | | |
| Nowrap | | The nowrap value specifies that the flex items will not wrap but it will shrink smaller and smaller to fit the width of container (this is default):  Note: Using nowrap would cause an overflow if the items were not able to shrink, or could not shrink small enough to fit. | .flex-container { display: flex; flex-wrap: wrap;} |
| Wrap | | The wrap value specifies that the flex items will wrap if necessary.  **Note:** wrap happen when the maximum number of Childs located at the same horizontal line and there is no enough current width in the container to embed another Childs. | .flex-container {  display: flex;  flex-wrap: wrap;  } |
| wrap-reverse | | The wrap-reverse value specifies that the flexible items will wrap if necessary, in reverse order: | .flex-container {  display: flex; flex-wrap: wrap-reverse;} |
| Flex-flow | The flex-flow property is a shorthand property for setting both the flex-direction and flex-wrap properties. | | | |
| Flex-direction flex-wrap | | - | .flex-container {  display: flex;  flex-flow: row wrap;  } |
| justify-content | The justify-content property is used to align the flex items (Childs) inside the container. (this property is affected when the flex-direction property is row or row-reverse) | | | |
| center | | The center value aligns the flex items at the center of the container: | .flex-container {display: flex; justify-content: center;} |
| Flex-start | | The flex-start value aligns the flex items at the beginning of the container (this is default). | .flex-container {display: flex; justify-content: flex-start;} |
| Flex-end | | The flex-end value aligns the flex items at the end of the container. | .flex-container {display: flex; justify-content: flex-end;} |
| Space-round | | Make space around all items sides (top, bottom, left, and right) like margin property. | .flex-container{display: flex; justify-content: space-round;} |
| Space-between | | Make space between adjacent items from (left, right) like margin-left and margin-right property. | .flex-container {display: flex; justify-content: space between;} |
| align-items | The align-items property will align (position) the items on the cross axis. | | | |
| Stretch | It stretches all flex items to fill the parent in the direction of the cross axis. (default value)  **Note:** If the parent hasn't got a fixed width in the cross axis direction, then all flex items will become as long as the longest flex items. | |  |
| baseline |  | |  |
| Flex-start | It align (position) all items at the start edge of the cross axis. | |  |
| Center | Used to center items along the cross axis. | |  |
| Flex-end | It align all items at the end edge of the cross axis. | |  |
| Align-self | This property is used to make a special align (position) for individual item as this property override the align-items | | | |
| auto |  | | button:first-child {align-self: auto;} |
| Stretch | It stretches all flex items to fill the parent in the direction of the cross axis. (default value)  **Note:** If the parent hasn't got a fixed width in the cross axis direction, then all flex items will become as long as the longest flex items. | | button:first-child {align-self: strech;} |
| baseline |  | | button:first-child {align-self: baseline;} |
| Flex-start | It align (position) all items at the start edge of the cross axis. | | button:first-child {align-self: flex-start;} |
| Center | Used to center items along the cross axis. | | button:first-child {align-self: center;} |
| Flex-end | It align all items at the end edge of the cross axis. | | button:first-child {align-self: flex-end;} |
| align-content |  |  | |  |
| Justify-content | The justify-content property will align (position) the items on the main axis. | | | |
| Flex-start | it makes all the items sit at the start of the main axis. (default value) | |  |
| center | it makes all the items sit at the center of the main axis. | |  |
| Flex-end | it makes all the items sit at the end of the main axis. | |  |
| Space-around |  | |  |
| Space-between |  | |  |
| space-evenly |  | |  |

**Flex items**

Important term:

Available space: If we have three 100 pixel-wide items in a container which is 500 pixels wide, then the space we need to lay out our items is 300 pixels. This leaves 200 pixels of available space. If we don’t change the initial values then flex box will put that space after the last item. This picture illustrates the “Available space” definition:



* **properties of flex items**

1. order
2. Flex-basis
3. Flex-grow
4. Flex-shrink
5. Flex

|  |  |  |  |
| --- | --- | --- | --- |
| order | numbers | This property allows you to change flex items order without needing to change their orders in html codes.  Default value is 0 | .tic1 {order: 3;} |
| Flex-basis | This property specify the initial main axis size of a flexible item before adding any available space to any flex items in this container. | | |
| Auto = content | Default value. The initial length is equal to the **fixed length** you determined for this flex-item. If you didn’t determine any **fixed length** for this item then the initial width will be according to the content.  **Note:**   * Content value isn’t support in some old browsers so auto one is more safe. * Default value is auto |  |
| Length / % |  |  |
| Fill |  |  |
| Fit-content |  |  |
| max-content |  |  |
| Min-content |  |  |
| Flex-grow | This property is used to make a specific item to grow in width [bigger than **flex-basis**]; it make this grow from the available space of the container, so if the container has no available space then the item has no flow.  Illustrate:  If you have a container with 600px width and it contain 4 items inside it; by default each item will take 150px width (600/4) so there is no available space then if you write flex-grow property to any items there is no change happen.  If you specify the width of the container to 700 and it contain 4 items inside it; by default each item will take 233.3px width (700/4) and it also has no change when adding flex-grow to any item.  So if you want to activate this property you should first override the default width value of flex items **[flex-basis]**.  Assume that we have a container with 600px width contain 4 items each one has a **fixed width** 100px. Then the available space will equal 200px (600-400) in this case adding flex-grow property to one item, will cause growing in its width from 100px to 300px (100 fixed width + 200 available space).  Flex grow property take a number values it may be 1, 2, 3, 4, …. This numbers refer to the percentage (amount) of available space should this item take.  So if we have only one item with flex-grow property 1 means that item will take all available space In the container. Values 2, 3, 4, … have the same result as there is only one item that has flex-grow property.  If we have more than one item has a flex-grow:1 means that the available space (200px) will divided equally between these items . if you add flex-grow property to two items the available space will divided equally by percentage (1:1) ; 100px to one item and 100 to another so the total width of each item equal 200px. And the items without flex-grow property still the same width (100px). If we specify the flex-grow:2 for one item and flex-grow: 3 for another item, that mean the available space (200px) will divided with a percentage (2:3) for the first and second item respectively. So the total width of the first item equal 180px (100 fixed width + 80 available space) and the total width for the second item will equal 220px (100 fixed width + 120 available space). To make sure (100px (flex-item) + 100px (flex-item) + 180px (flex-item with flex-grow=2) +220px(flex-item with flex-grow=3) = 600px (container width))  Note: default value is 0 | | |
| Positive numbers | Default value is 0 | Container {width: 600px; display: flex;}  Div {width: 100px;}  #Div1 {width: 100px; flex-grow: 2;}  #Div2 { width: 100px; flex-grow: 3;} |
| Flex-shrink | This property is used to make a specific item to shrink in width [smaller than **flex-basis**]; it make this shrinking by giving some of its width to available space of its container.  Note: default value is 1 | | |
| flex | The flex property is a shorthand property allows you to set the three values in this order — flex-grow flex-shrink flex-basis.  Default (initial value) value is : item {flex: 0 1 auto;} | | |
| none | Means flex: 0 0 auto. Using flex: none will create fully inflexible flex items. The items cannot grow or shrink but will be laid out using flexbox with a **flex-basis of auto** |  |
| auto | Means flex: 1 1 auto. Everything is as with flex: initial; but in this case the items can grow and fill the container as well as shrink if required. |  |
| Numbers | This is as if you used flex: 1 1 0. The items can grow and shrink from a flex-basis of 0. | Item {flex: 1;} or item{flex: 2;} or item {flex: 3;} |