Contents

[CSS rules and cascading 3](#_Toc508476712)

[CSS selectors 3](#_Toc508476713)

[Element and class selectors 3](#_Toc508476714)

[Id selector 3](#_Toc508476715)

[Universal selector \* 3](#_Toc508476716)

[Three types of attribute selectors 3](#_Toc508476717)

[Substring value attribute selector 4](#_Toc508476718)

[Pseudo classes 4](#_Toc508476719)

[Pseudo elements 5](#_Toc508476720)

[CSS cascade rules 6](#_Toc508476721)

[Box model 6](#_Toc508476722)

[Important notes 6](#_Toc508476723)

[Tips 6](#_Toc508476724)

[Click event in css 7](#_Toc508476725)

[SCSS 8](#_Toc508476726)

[Advanced 8](#_Toc508476727)

[Positioning 8](#_Toc508476728)

[Block and inline elements 10](#_Toc508476729)

[Layers 10](#_Toc508476730)

[Rules 11](#_Toc508476731)

[Visibility: 11](#_Toc508476732)

[Border 11](#_Toc508476733)

[Outline 11](#_Toc508476734)

[Scrollbar 11](#_Toc508476735)

[Dimension 12](#_Toc508476736)

[Lists 12](#_Toc508476737)

[Cursor 12](#_Toc508476738)

[Table 13](#_Toc508476739)

[Image 14](#_Toc508476740)

[Link 14](#_Toc508476741)

[Font 14](#_Toc508476742)

[Text 14](#_Toc508476743)

[Background 15](#_Toc508476744)

[Margin and Padding 15](#_Toc508476745)

[Measurement units 16](#_Toc508476746)

[References 17](#_Toc508476747)

# 02-06-2019

Inline: maintains same line. Height, width, margin-top, margin-bottom have no effect.

Inline-block: remains in same line. Height, width, margin-top, margin-bottom work.

Block: Just change line and Height, width, margin-top, margin-bottom etc. work

Box-sizing: border-box property of css ensures that border and padding are included in width and height of an element. This is good practice to include this property as whole.

\* {  
  box-sizing: border-box;  
}

Responsive design using flex

<https://internetingishard.com/html-and-css/responsive-design/>

Responsive images

<https://internetingishard.com/html-and-css/responsive-images/>

## Responsive design

It is good to first design for mobile and then move up for desktop. This is called mobile-first approach.

### Block and inline elements

Block elements take the entire possible. Inline elements take minimum space. Block starts at new line. Height and width of element works.

Inline: No new Line. Height and width of element has no effect.

Display: inline-block

No new line. But height and width of element is effective.

# Right align without float

Container: text-align:right; display:inline-block

# CSS rules and cascading

Say an element has multiple style classes defined as class=”classa classB classC”. The color attribute is defined as red, green and blue respectively in the above three classes, the css selectors are different. Then the color of element is not necessarily blue which is defined in classC, the last class. The actual color will depend on selection rules or specificity.

# CSS selectors

### Element and class selectors

h1 {

color: red;}

.first {

font-weight: bold;

}

### Id selector

#polite {

font-family: cursive;

}

<p id="polite"> — "Good morning."</p>

### Universal selector \*

### Three types of attribute selectors

[**attr**] select elements with attribute **attr** present

[**attr=val**] select elements with attribute **attr** having value as **val**

[**attr~=val**] select elements with attribute **attr** having one of values as **val**

Ingredients for my recipe: <i lang="fr-FR">Poulet basquaise</i>

<ul>

<li data-quantity="1kg" data-vegetable>Tomatoes</li>

<li data-quantity="3" data-vegetable>Onions</li>

<li data-quantity="3" data-vegetable>Garlic</li>

<li data-quantity="700g" data-vegetable="not spicy like chili">Red pepper</li>

<li data-quantity="2kg" data-meat>Chicken</li>

<li data-quantity="optional 150g" data-meat>Bacon bits</li>

<li data-quantity="optional 10ml" data-vegetable="liquid">Olive oil</li>

<li data-quantity="25cl" data-vegetable="liquid">White wine</li>

</ul>

And a simple style sheet:

/\* All elements with the attribute "data-vegetable"

are given green text \*/

[data-vegetable] {

color: green;

}

/\* All elements with the attribute "data-vegetable"

with the exact value "liquid" are given a golden

background color \*/

[data-vegetable="liquid"] {

background-color: goldenrod;

}

/\* All elements with the attribute "data-vegetable",

containing the value "spicy", even among others,

are given a red text color \*/

[data-vegetable~="spicy"] {

color: red;

}

The result is as follows:

Ingredients for my recipe: *Poulet basquaise*

* Tomatoes
* Onions
* Garlic
* Red pepper
* Chicken
* Bacon bits
* Olive oil
* White wine

### Substring value attribute selector

[attr|=val]: attribute’s value starts with val-

[attr^=val]: attribute’s value starts with val

[attr$=val]: attribute’s value ends with val

[attr\*=val]: attribute’s values has val embedded into it

### Pseudo classes

:active :any :checked :default :dir() :disabled :empty :enabled :first :first-child :first-of-type :fullscreen :focus :focus-within :hover :indeterminate :in-range :invalid :lang() :last-child :last-of-type :left :link :not() :nth-child() :nth-last-child() :nth-last-of-type() :nth-of-type() :only-child :only-of-type :optional :out-of-range :read-only :read-write :required :right :root :scope :target :valid :visited

These are not selectors but change the behavior of selectors in certain context.

/\* These styles will style our link

in all states \*/

a {

color: blue;

font-weight: bold;

}

/\* We want visited links to be the same color

as non visited links \*/

a:visited {

color: blue;

}

/\* We highlight the link when it is

hovered (mouse), activated

or focused (keyboard) \*/

a:hover,

a:active,

a:focus {

color: darkred;

text-decoration: none;

}

### Pseudo elements

::after ::before ::first-letter ::first-line ::selection ::backdrop

These are also appended after the selectors to change behavior. ::after is applied after all elements which satisfy the selector

<ul>

<li><a href="https://developer.mozilla.org/en-US/docs/Glossary/CSS">CSS</a> defined in the MDN glossary.</li>

<li><a href="https://developer.mozilla.org/en-US/docs/Glossary/HTML">HTML</a> defined in the MDN glossary.</li>

</ul>

Let's add this CSS rule:

/\* All elements with an attribute "href", which values

start with "http", will be added an arrow after its

content (to indicate it's an external link) \*/

[href^=http]::after {

content: '⤴';

}

We get this result:

* [CSS](https://developer.mozilla.org/en-US/docs/Glossary/CSS) defined in the MDN glossary.
* [HTML](https://developer.mozilla.org/en-US/docs/Glossary/HTML) defined in the MDN glossary.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Group of selectors |  |  | A, B |  | | Any element matching A and/or B |
| Descendant selector |  |  | A B |  | Any element matching B that is a descendant  of an element matching A  (that is: a child, or a child of a child, etc.) | |
| Child selector |  |  | A > B |  | Any element matching B that is a direct child  of an element matching A. | |
| Adjacent sibling selector |  |  | A + B |  | Any element matching B that is the  next sibling of an element matching A  (that is: the next child of the same parent.) | |
| General sibling selector |  |  | A ~ B |  | Any element matching B that is one of  the next siblings of an element matching A (that is: one of the next children of the same parent.) | |

### CSS cascade rules

There are situations when multiple rules apply for an element because of many selectors. Which rule wins is guided by a mechanism called cascade. **Specificity** is measure as to how specific a selector is. Element selector < class selector < id selector < !important. Thus !important has highest specificity and it will win against others. Measurement of specificity is done in thousands, hundreds, tens and ones.

Thousands: Score 1 if declaration is in style attribute otherwise 0

Hundreds: Score 1 if declaration is in id selector

Tens: Score 1 if declaration is in class or attribute or pseudo class selector

Ones: Score 1 if declaration is in element or pseudo element selector

Universal(\*), combination(+, ~, >,’’) have no effect on specificity.

If specificity is same for two declarations then the last one wins.

<https://developer.mozilla.org/en-US/docs/Learn/CSS/Introduction_to_CSS/Cascade_and_inheritance>

### Box model

<https://developer.mozilla.org/en-US/docs/Learn/CSS/Introduction_to_CSS/Box_model>

# Important notes

line-height means the space between two lines. It is meaningful for <p> tag. The line-height:125% will increase the space between two lines by 25%.

CSS order for margin, padding etc starts at top and then moves in cyclic manner as top, right, bottom, left.

# Tips

* Body element has a default margin of 8 px on all four sides. Div element has no default margin. Overflow:auto makes all elements inside a div not to leak outside. Scroll bars will appear.
* Class selectors

.class1.class2 means <div class="class1 class2"></div>

.class1 .class2 means

<div class="class1">

<div class="class2"></div>

</div>

.class1, .class2, .class3 {} means that all the classes are having same definitions.

* Display: inline-block introduces some margin between the elements. To get rid of that use float:left
* The vertical-align:middle does not bring current element to middle. Make line-height of parent div element to the height of parent div element. Then this will work.
* hover css selector only works on elements. It does not work with class selector alone. If you want to uniquely select an element and apply hover to it then include a class within the element as below:

button.button-transition:hover {

width:500px;

}

* Drop down using css

button { 

background-color: inherit;

color: inherit;

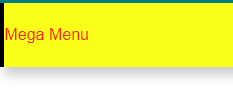
border: none;

outline: none;

}

This appears like pure text but the button is there. Inherit brings the parent background color which is black. Border gives plain look and outline removes the box look when it is clicked.

Shadow look



box-shadow: 0px 8px 16px 0px **rgba**(0, 0, 0, 0.2);

How the menu is displayed

The entire mega menu stuff and button are in a div. The drop down portion of the div is display:none. The hover event of button makes the drop down portion as display:block. The position is kept absolute so that other elements are not affected and the menu appears at top of other elements.

div:hover .content {

display: block;

}

### Click event in css

There is no click event in css. You can simulate it with checkbox hack.

* Provide a checkbox with position off the screen such as position: absolute; left:-9999px; top:-9999px. This will place it beyond the screen but since absolute, will not affect the placement of other elements.
* Provide one label element with **for** attribute. This will enable the label element click as if the checkbox is clicked. Decorate this label element like a button using css.
* Use checkbox:checked pseudo class and some selector along with this, so that when the selector is selected the css would be changed for that selector:

input[type=checkbox]:checked ~ .to-be-changed {

color: red;

}

Hereinabove the ~ operator is sibling selector. Now on click of checkbox all siblings with .to-be-changed class will be red. If you again click the label then checkbox is unchecked. Hence the selector will go back to its original state.

# SCSS

Angular 4/5

Whatever classes you define in styles.scss can be used globally. But variables defined in styles.scss cannot be used globally. You can define variables in \_variables.scss and import it in styles.scss that way can be used. Another way:

* Define a styles folder in src folder
* Keep all your styles files like \_variables.scss, \_mixins.scss, \_styles1.scss etc. in this folder
* In .angular-cli.json file provide this stub:

"stylePreprocessorOptions": {

"includePaths": [

"styles"

]

},

* Restart by ng serve.

Now all the style files which are stored in src/styles folder can be used as @import ‘variables’ in any .scss file of application.

# Advanced

### Positioning

Positioning means the placement of element with respect to its top, right, bottom and left. Means how top, right, bottom and left are interpreted. In normal situations the browser puts the elements on screen in a flow. The position values of elements affect this flow. The normal position of an element is the position when it is placed by browser in the flow. Position is meaningful in context of top, right, bottom and left.

**Relative**: Means that element is **placed in respect of its normal position**. So if top:10px, left:10px then the element would appear 10px displaced by its normal position. The position of other elements would not be affected. Other elements are not disturbed. This element moves relative to its original position. But this element retains its original position

.box {

display: inline-block;

width: 100px;

height: 100px;

background: red;

color: white;

}

#two {

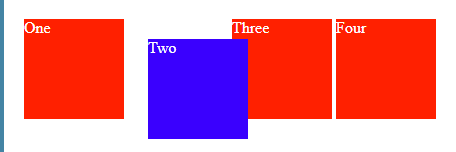
position: relative;

top: 20px;

left: 20px;

background: blue;

}



**Absolute**: The element is completely taken out of flow. For other elements it means as if it does not exist. The space of the element is occupied by other elements. Other elements will fill in as if this element does not exist. The element can be seen overlapping other elements because other elements do not believe that this element exists. The element is positioned relative to its nearest ancestor otherwise it will be positioned with relative to document body. This is relative to container. Other elements behave as if this element does not exist.

<div class="box" id="one">One</div>

<div class="box" id="two">Two</div>

<div class="box" id="three">Three</div>

<div class="box" id="four">Four</div>

#### CSS

.box {

display: inline-block;

width: 100px;

height: 100px;

background: red;

color: white;

}

#two {

position: absolute;

top: 20px;

left: 20px;

background: blue;

}



**Fixed**: This is same as absolute positioning except that the **containing element will be viewport**. So the positioning is done with respect of viewport. This can be used to create a floating element which is irrespective to scrolling. This is relative to viewport. Other elements behave as if this element does not exist.

Sticky: This positioning is hybrid of absolute and fixed positioning. Top, right, bottom and left are meaningful in context of position property. These are different to the margin attribute.

### Layers

This is same as z-index.

A z-index property can help you to create more complex webpage layouts. Following is the example which shows how to create layers in CSS.

<html>

<head>

</head>

<body>

<div style="background-color:red; width:300px; height:100px; position:relative; top:10px; left:80px; z-index:2">

</div>

<div style="background-color:yellow; width:300px; height:100px; position:relative; top:-60px; left:35px; z-index:1;">

</div>

<div style="background-color:green; width:300px; height:100px; position:relative; top:-220px; left:120px; z-index:3;">

</div>

</body>

</html>

### Rules

@import to import another css. Used for modular design.

<style tyle="text/css">

<!--

@import "mystyle.css";

or

@import url("mystyle.css");

.......other CSS rules .....

-->

</style>

!important. The stylesheets. That means the styles are applied in the same order the browser reads it.

### Visibility:

Content is still there but not visible. Possible values are visible, hidden, collapse.

### Border

Three properties

Border-color: top right bottom left

border-width

border-style: none, solid, dotted, dashed, double, groove, ridge, inset, outset, hidden

All properties can be used together

<p style="border:4px solid red;">

### Outline

<p style="outline:thin solid red;">

This text is having thin solid red outline.

</p>

Outline is like border but does not take space.

### Scrollbar

overflow: visible, hidden, scroll, auto

|  |  |
| --- | --- |
| **Value** | **Description** |
| visible | Allows the content to overflow the borders of its containing element. |
| hidden | The content of the nested element is simply cut off at the border of the containing element and no scrollbars is visible. |
| scroll | The size of the containing element does not change, but the scrollbars are added to allow the user to scroll to see the content. |
| auto | The purpose is the same as scroll, but the scrollbar will be shown only if the content does overflow. |

### Dimension

The height, width, line-height, max-height, min-height, max-width, min-width

### Lists

Ordered and unordered lists (ol and ul)

<ul style="list-style-type:square;">

<li>Maths</li>

<li>Social Science</li>

<li>Physics</li>

</ul>

<ol style="list-style-type:decimal;">

<li>Maths</li>

<li>Social Science</li>

<li>Physics</li>

</ol>

### Cursor

Defines the type of cursor

<html>

<head>

</head>

<body>

<p>Move the mouse over the words to see the cursor change:</p>

<div style="cursor:auto">Auto</div>

<div style="cursor:crosshair">Crosshair</div>

<div style="cursor:default">Default</div>

<div style="cursor:pointer">Pointer</div>

<div style="cursor:move">Move</div>

<div style="cursor:e-resize">e-resize</div>

<div style="cursor:ne-resize">ne-resize</div>

<div style="cursor:nw-resize">nw-resize</div>

<div style="cursor:n-resize">n-resize</div>

<div style="cursor:se-resize">se-resize</div>

<div style="cursor:sw-resize">sw-resize</div>

<div style="cursor:s-resize">s-resize</div>

<div style="cursor:w-resize">w-resize</div>

<div style="cursor:text">text</div>

<div style="cursor:wait">wait</div>

<div style="cursor:help">help</div>

</body>

</html>

### Table

Various attributes are

border-collapse: collapse or separate, border of every cell is merged or separate

|  |
| --- |
| Collapse Border Example |
| Cell A Collapse Example |
| Cell B Collapse Example |

|  |
| --- |
| Separate Border Example |
| Cell A Separate Example |
| Cell B Separate Example |

border-spacing: space between two cells

empty-cells: show,hide, inherit. Will show or hide empty cells of table

### Image

<img style="border:1px solid red; width:100%;" src="/css/images/logo.png" />

### Link

An anchor tag has various states such as link, visited, hover and active. These are meaningful in the same order. Active is at last, hover is before that and so on. You can define different color for them in the header of HTML so that it is valid for all the pages. A sample css for header is as below:

<style type="text/css">

a:link {color: #000000}

a:visited {color: #006600}

a:hover {color: #FFCC00}

a:active {color: #FF00CC}

</style>

### Font

**style="font-family:georgia,garamond,serif;"**

**style="font-style:italic;"**

**style="font-variant:small-caps;"**

**style="font-weight:bold;"**

font-weight possible values are: *normal, bold, bolder, lighter, 100, 200, 300, 400, 500, 600, 700, 800, 900. The 900 is most bold.*

## Text

* The **color**: color of a text.
* The **direction**: text direction.
* The **letter-spacing:** space between the letters that make up a word.
* The **word-spacing:** space between the words of a sentence.
* The **text-indent: 1cm,** indent the text of a paragraph. The first word of paragraph will be value apart.
* The **text-align:right**, align the text of a document.
* The **text-decoration: underline, overline, linethrough, blink.** For underline, overline, strikethrough and blink text.
* The **text-transform: capitalize, uppercase, lowercase**
* The **white-space** property is used to control the flow and formatting of text.
* The **text-shadow: 4px 4px 8px blue** property is used to set the text shadow around a text.

### Background

* The **background-color** is used to set the background color of an element.
* The **background-image** is used to set the background image of an element.
* The **background-repeat** is used to control the repetition of an image. Values are repeat, repeat-x, repeat-y
* The **background-position** property is used to control the position of an image.
* The **background-attachment** property is used to control the scrolling of an image in the background. Whether or not the background image is fixed or moves with scroll.
* The **background** property is used as a shorthand to specify a number of other background properties.

### Margin and Padding

Margin is outside of an html element. Padding is inside of an element. Margin will overlap but padding will not. If <div> container has padding: 20px then distance between two div containers will be 40px. But if margin:20px then the distance will be 20px only since margin is overlapping.

Html elements have default margin. Typically it is 1em.

|  |  |
| --- | --- |
|  | enter image description here |

### Measurement units

|  |  |  |
| --- | --- | --- |
| **Unit** | **Description** | **Example** |
| % | Defines a measurement as a percentage relative to another value, typically an enclosing element. | p {font-size: 16pt; line-height: 125%;} |
| cm | Defines a measurement in centimeters. | div {margin-bottom: 2cm;} |
| em | A relative measurement for the height of a font in em spaces. Because an em unit is equivalent to the size of a given font, if you assign a font to 12pt, each "em" unit would be 12pt; thus, 2em would be 24pt. | p {letter-spacing: 7em;} |
| ex | This value defines a measurement relative to a font's x-height. The x-height is determined by the height of the font's lowercase letter x. | p {font-size: 24pt; line-height: 3ex;} |
| in | Defines a measurement in inches. | p {word-spacing: .15in;} |
| mm | Defines a measurement in millimeters. | p {word-spacing: 15mm;} |
| pc | Defines a measurement in picas. A pica is equivalent to 12 points; thus, there are 6 picas per inch. | p {font-size: 20pc;} |
| pt | Defines a measurement in points. A point is defined as 1/72nd of an inch. | body {font-size: 18pt;} |
| px | Defines a measurement in screen pixels. | p {padding: 25px;} |
| vh | 1% of viewport height. | h2 { font-size: 3.0vh; } |
| vw | 1% of viewport width | h1 { font-size: 5.9vw; } |
| vmin | 1vw or 1vh, whichever is smaller | p { font-size: 2vmin;} |

One idea is set font-size in body and use em elsewhere in the html document. The em is relative to effective font size.

# References

css good tutorial

<https://www.tutorialspoint.com/css/css_measurement_units.htm>

https://robohash.org/abcd.png?size=250x250&set=set1