CSS | Day 1 | Pre-work

CSS

Day 1

Table of contents:

CSS

Design characteristics play an important part and are a key component of every successful Web Site. Web Design also requires a good understanding of how your users experience sites and browse content. In practice, design for a website will require learning the layout language of the web called CSS.

CSS stands for **Cascading Style Sheet**. CSS is based on a **set of rules** that describe the **priority** of how styles are rendered on the page. The rules can be as simple as defining colors, sizes, fonts, alignment etc.

Anatomy of Cascading Style Sheets

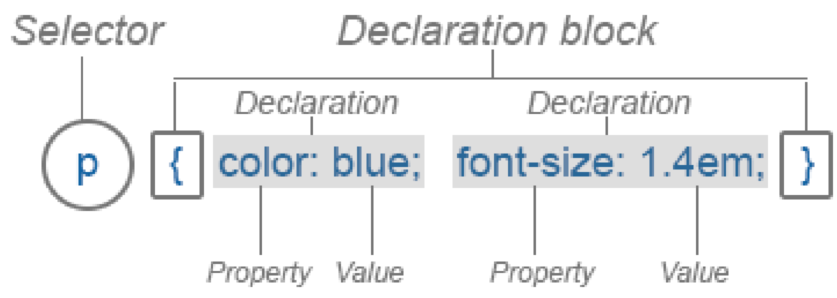


Image 1.0 - Anatomy of a CSS rule

Selector - tells the web browser which element or **elements on a page to style**.

For example: Paragraph (as you can see in the example above). If we define the CSS rule to the **<p>** element **it will be applied to all paragraph elements** in the whole document.

Declaration block - the declaration block begins with an opening brace **{** and ends with a closing brace **}**. This block includes **all the formatting options** you want to apply to the selector.

Declaration - every declaration has two parts: a **property** and a **value**. A colon ":" separates the property name and its value, and the whole declaration ends with a semicolon.

Property - CSS gives you many different formatting options, called properties. A property indicates a certain style effect. Most properties have pretty self-explaining names like **font-size**, **margin-top**, **color** and **text-align**. After the property name we must add a colon (:) to separate it from the value.

Value - after the property name we need to enter the value, e.g. to make the text color blue or to change the text size (see the example below). Different CSS properties require specific type of values - e.g. color (like **red**, or **#FF0000**), a length (like **18px**, **100%** or **5em**), a URL (like **img/img1.jpg**) or specific keyword (**top**, **center** or **bottom**).

Note: Bad practice - DON'T write the rule on a single line! Break them into **multiple lines**.

Bad practice:

```
p { color:blue; font-size: 1.5em;}
```

Good practice:

```
p {
color:blue;
font-size: 1.5em ;
}
```

There are 3 different ways to include those CSS rules or styling into our HTML:

Internal Style Sheet

External Style Sheet

Inline Style

Internal Style Sheet

You can style your page **within an HTML page**, by inserting the CSS rules with the **<style>** tag inside the **<head>** tag of the web page, as we do here:

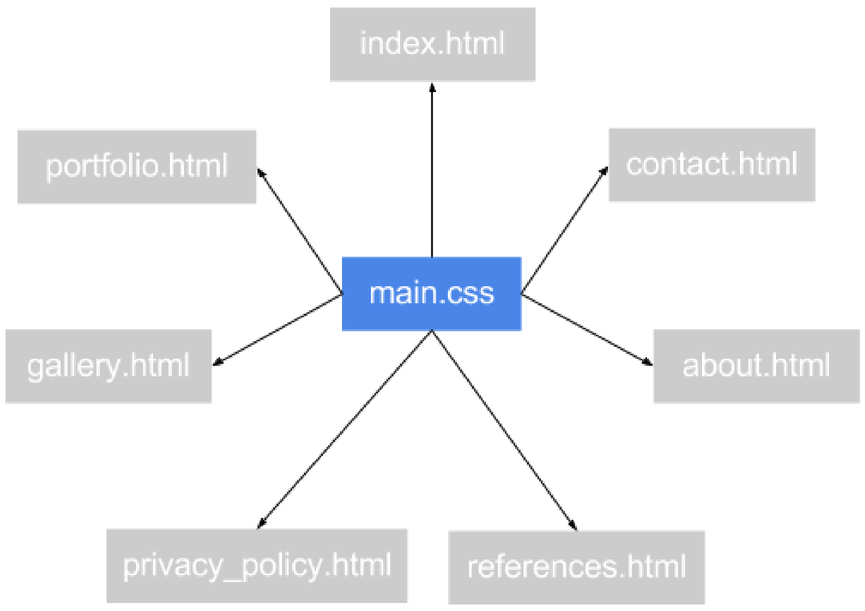
```
<!DOCTYPE html>
<html>
  <head>
     <title>FSWD CSS internal style sheet (1)</title>
     <style>
        h1 {
           color:red;
           font-size: 3em;
           font-family:Arial;
        }
        p {
           color:blue;
           font-size:1.5em;
           background-color: yellow;
        }
     </style>
  </head>
  <body>
     <h1> Hello there</h1>
     <p>today is such a lovely day!</p>
  </body>
</html>
```

Note: If the styles are only inside the **<head>** tag and not nested into the **<style>** tag they will not be recognized by the browser.

External Style Sheet

When you want to style a set of pages or just one single page, a better way to manage the styling is to create a single CSS document and link that document to all the html documents. This way you can have a centralized file from where you control the whole site. This is a cleaner way of organizing your code!

The following figure illustrates the principle:

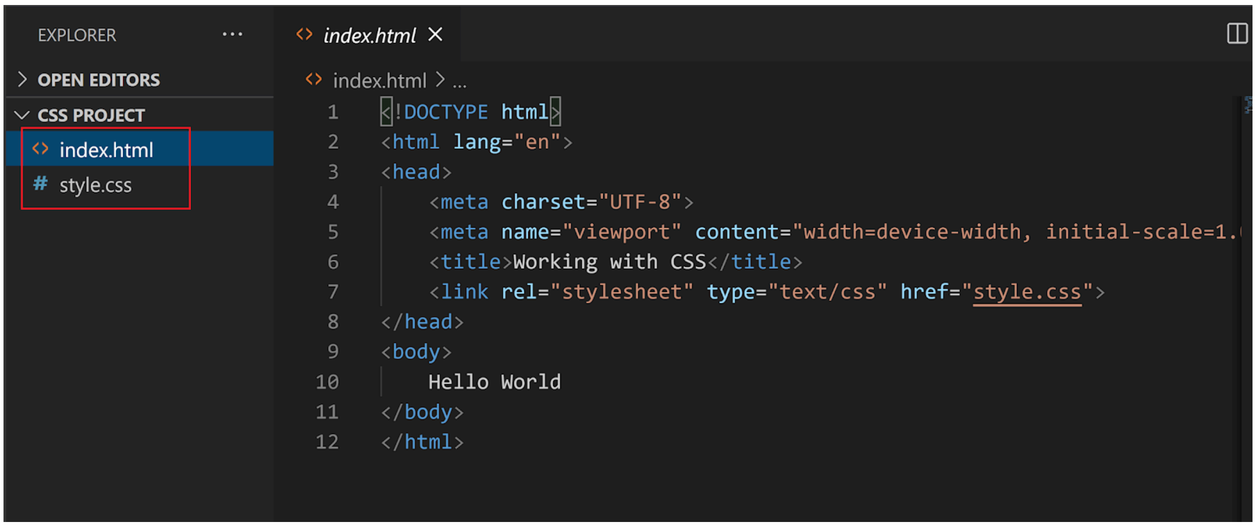


Changing the main CSS document affects all other html pages, so your web app looks more professional. You could though keep the main.css for more general styling and create specific ones with the same name as the page they have the styling for. It will be more professional and organized.
Put the following code inside the **<head>** tag:

```
<head>
    <link  rel='stylesheet' type='text/css'  href='styles.css'>
</head>
```

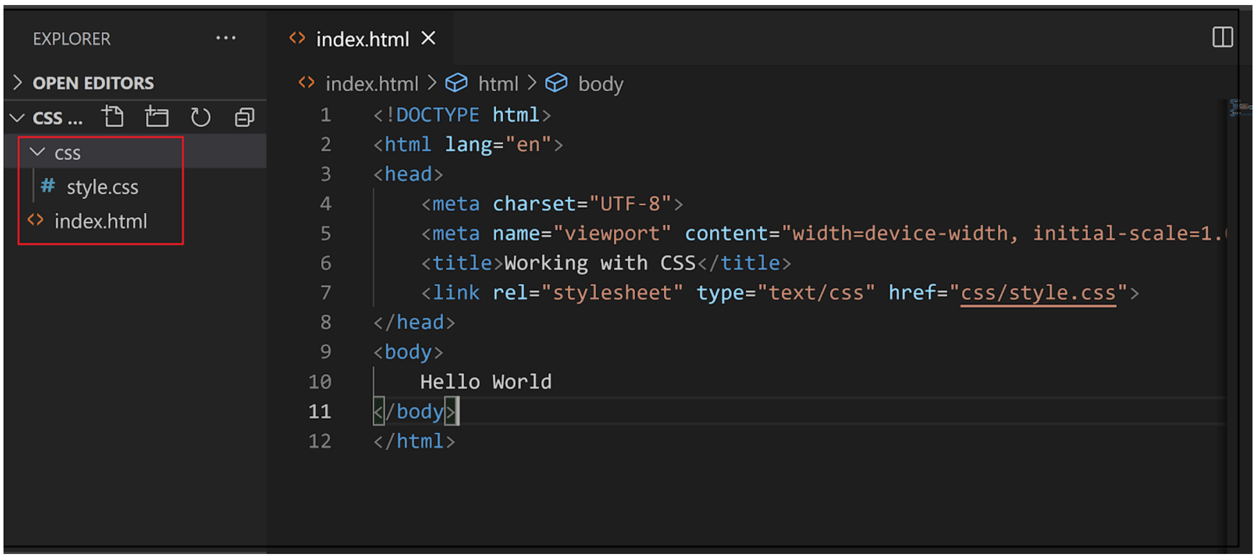
Note: While including external CSS documents, you don't have to use the additional **<style>** tag inside the **<head>**.

The value of the **href** parameter depends on the location of the **style.css** file in relation to our HTML document - in this case, **style.css** and your HTML document are in the same folder.



```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.
    <title>Working with CSS</title>
    <link rel="stylesheet" type="text/css" href="style.css">
</head>
<body>
    Hello World
</body>
</html>
```

But if the style.css file is located in another folder the **href** would change, like in the following example where the **href=" css/style.css"** is located inside a folder called 'css', where the html file is in the same location as the css folder:



```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.
    <title>Working with CSS</title>
    <link rel="stylesheet" type="text/css" href="css/style.css">
</head>
<body>
    Hello World
</body>
</html>
```

Inline Style

Instead of writing a lot of rules at once, you can define certain rules for specific element directly into it, like in the example below (which results in italic, blue text within the tags):

```
<div style='font-style:italic; color:blue;'>Hello there</div>
```

This technique is bad practice! It is important to keep your code well organized and this technique mixes both CSS and HTML, which makes it harder to read!

CSS Selectors

Let’s examine the following statement:

```
h1 {
  color: blue;
}
```

This rule will select all **<h1>** tags in the page and make them blue, but what if we want to make only one h1 element blue? This is where selectors are useful - they allow us to be more specific in our selection.

With ***CSS Selectors*** we can point to the elements that should change their appearances based on their name, id, class, attribute and many other characteristics.

Element Selector

It selects elements based on the element name such as “p” for paragraph, “div” for division etc.

The syntax is as follows:

```
div {
    background-color: gray;
}
```

This rule selects all div tags in the page and gives them a gray background.

ID Selector

The ID selector selects the element by its id. Id is an HTML element attribute that is used to **define a unique element** on the page.
Here is how we assign the “id” attribute to an HTML element:

```
<div id="comments">Unique Comments Section</div>
```

The following rule selects the element with property “id” of value “comments” and makes the background of the element yellow, which in this case is this specific **<div>** element:

```
#comments {
  background-color: yellow;
}
```

In CSS we access the “id” elements using the hash (#) symbol.
The “id” value of an element **must** be **unique** within a page, so the id selector is used to select **only one element**.

Class Selector

The class selector works similarly to the ID selector. The difference is that it is aimed to **identify multiple elements** with the same value as the class attribute, instead of a unique one.
The Class selector selects all HTML elements with a specific class attribute.

In CSS we access the “class” elements using the dot (.) symbol.

Let’s consider that through the following example:

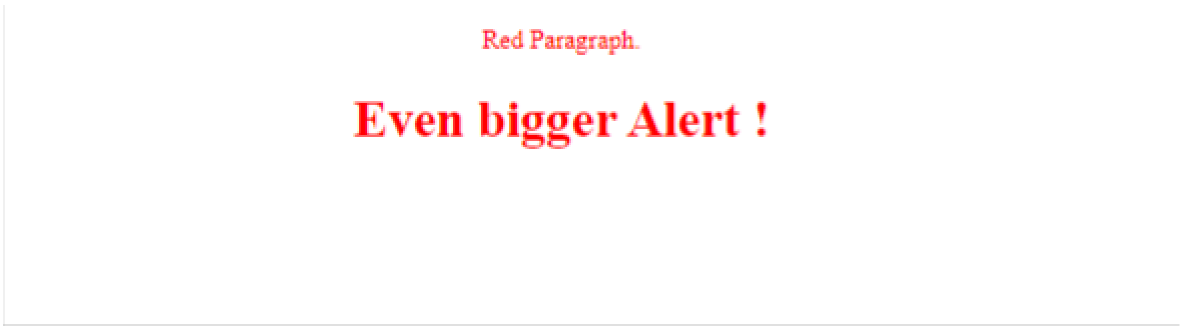
index.html

```
<p class="alert">Red Paragraph.</p>
<h1  class="alert">Even bigger Alert !</h1>
```

style.css

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

The result is one red paragraph and one red heading both centered.



Red Paragraph.

Even bigger Alert !

Descendant Selector

Descendant selectors allow us to select elements which are **nested** inside other elements, so we can deeply navigate through them.

For example the following rule sets all text within **<b>** tags that is nested as a descendant of a **<p>** tag to red:

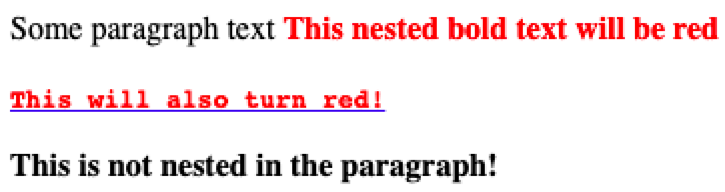
index.html

```
<p>Some paragraph text <b>This  nested bold text will be red</b></p>
<p><code><a  href="#"><b>This will also turn red!</b></a></code></p>
<b>This is not nested in the paragraph!</b>
```

style.css

```
p b {
   color: red;
}
```

The result will be:


Some paragraph text **This nested bold text will be red**

This will also turn red!

This is not nested in the paragraph!

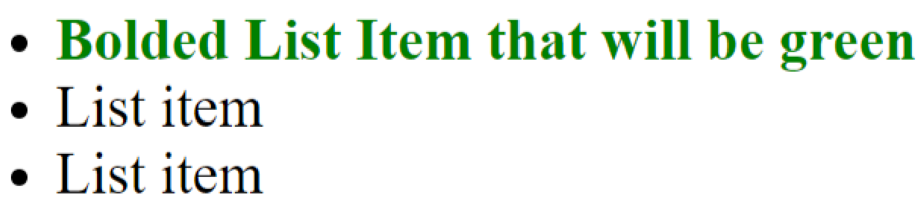
As you can see, only the text inside the <b> tags which are descendants (not necessarily direct) of the <p> tag turned red.
Let’s consider a more advanced example:

```
ul li b {
   color: green;
}
```

This rule makes it green, the text within the **<b>** tag that is inside a list element of an unordered list. HTML code that uses this CSS rule could look like this:

```
<ul>
 <li><b>Bolded List Item that will be green</b></li>
 <li>List item</li>
 <li>List item</li>
<ul>
```

So it will affect only the first list item.


- **Bolded List Item that will be green**
- List item
- List item

Child Selector

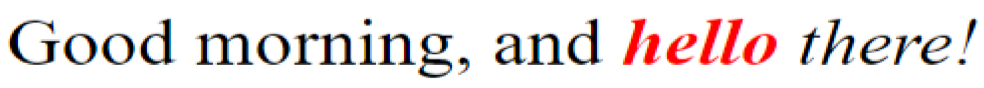
Child Selector is very similar to the descendant selector, but it is much **more restrictive** in terms of applying styles just to those elements that are a direct child of another element.

Let’s consider two similar scenarios:

```
p b {
   color: red;
}
```

This rule will affect all bolded paragraphs even if they are within additional **<i>** tags like this:

```
<p>Good morning, and <i><b>hello</b> there!</i></p>
```

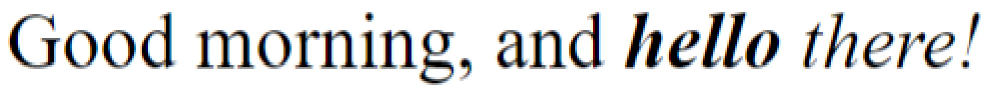


Good morning, and ***hello*** *there!*

In comparison, if we use the Child Selector it will style just the bolded element that is a direct descendant of a paragraph element, and is not itself contained within another element:

```
p > b {
   color: red;
}
```

Now the Hello string (in the snippet above) will not change the color because it is not a direct descendant of the paragraph.



Good morning, and ***hello*** *there!*

Display Property

In CSS we can display elements as we like. The display property is used to declare how an element is shown.
We can display elements as block elements:

```
a{
  display: block;
}
```

We can display elements as inline elements:

```
li{
  display: inline;
}
```

We can also hide elements:

```
li{
  display: none;
}
```

Note: The element will be hidden, and the page will be displayed as if the element does not exist!

or using:

```
li{
  visibility: hidden;
}
```

Note: The element will be hidden, but the element will still take up the same space as before, which will affect the layout!

CSS Comments

Comments are used to leave messages to other developers or ourselves to explain the code. This may for example help when you edit the source code at a later date or want to share information about its functionality. Comments are not displayed on the website because they are usually ignored by browsers.
A CSS comment starts with /* and ends with */

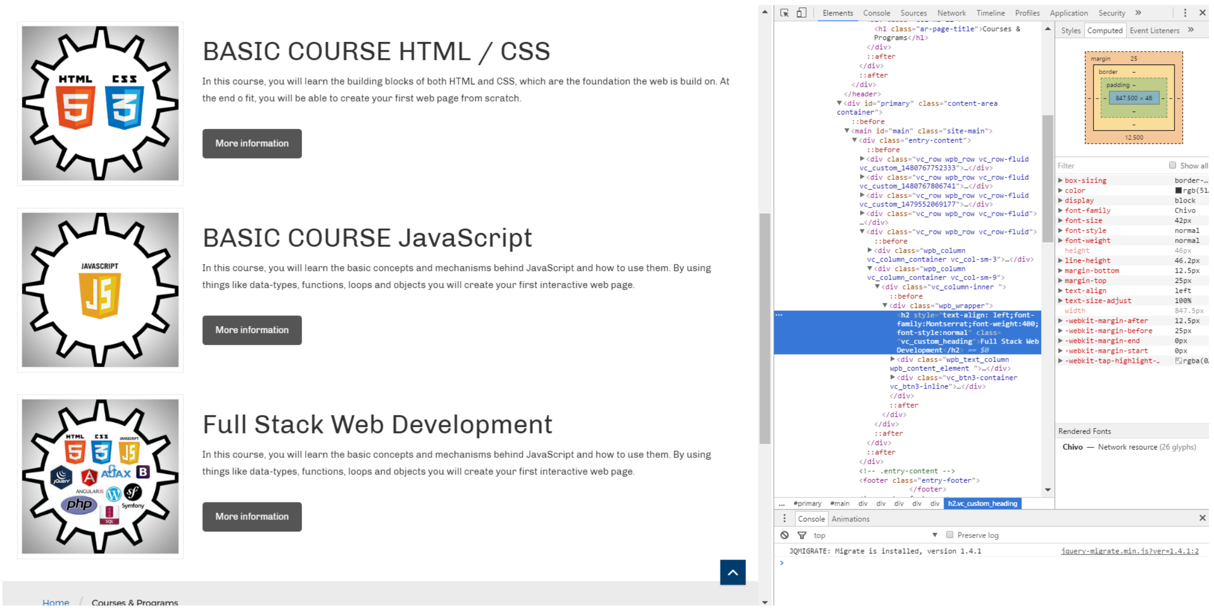
```
h2 {
      color: blue;
  /* This is how to create single-line comment */
}

/* This is
how to create
a multi-line
comment */
```

Box Model

Even if it does not look like it, **every element in web design is a rectangular box**. The CSS properties affecting the layout of a page are based around the box model. Virtually all elements have (or can have) these properties, including the document body.

To understand visually how each element within the web page is wrapped inside a rectangular box, open your Google Chrome browser, use the shortcut Ctrl + Shift + C and click on any element on any web page, you should get something like this:



BASIC COURSE HTML / CSS

In this course, you will learn the building blocks of both HTML and CSS, which are the foundation the web is build on. At the end o fit, you will be able to create your first web page from scratch.

More information

BASIC COURSE JavaScript

In this course, you will learn the basic concepts and mechanisms behind JavaScript and how to use them. By using things like data-types, functions, loops and objects you will create your first interactive web page.

More information

Full Stack Web Development

In this course, you will learn the basic concepts and mechanisms behind JavaScript and how to use them. By using things like data-types, functions, loops and objects you will create your first interactive web page.

More information

Home / Courses & Programs

```
<h1 class="ar-page-title">Courses &
Programs</h1>
      </div>
      ::after
    </div>
    ::after
  </div>
</header>
▼<div id="primary" class="content-area
container">
    ::before
  ▼<main id="main" class="site-main">
    ▼<div class="entry-content">
        ::before
      ▶<div class="vc_row wpb_row vc_row-fluid
      vc_custom_1480767752333">…</div>
      ▶<div class="vc_row wpb_row vc_row-fluid
      vc_custom_1480767806741">…</div>
      ▶<div class="vc_row wpb_row vc_row-fluid
      vc_custom_1479552069177">…</div>
      ▶<div class="vc_row wpb_row vc_row-fluid">
      …</div>
      ▼<div class="vc_row wpb_row vc_row-fluid">
          ::before
        ▶<div class="wpb_column
        vc_column_container vc_col-sm-3">…</div>
        ▼<div class="wpb_column
        vc_column_container vc_col-sm-9">
          ▼<div class="vc_column-inner ">
              ::before
            ▼<div class="wpb_wrapper">
...             <h2 style="text-align: left;font-
                family:Montserrat;font-weight:400;
                font-style:normal" class=
                "vc_custom_heading">Full Stack Web
                Development</h2> == $0
              ▶<div class="wpb_text_column
              wpb_content_element ">…</div>
              ▶<div class="vc_btn3-container
              vc_btn3-inline">…</div>
              </div>
              ::after
            </div>
          </div>
          ::after
        </div>
        ::after
      </div>
      <!-- .entry-content -->
      <footer class="entry-footer">
            </footer>
```

... #primary #main div div div div div h2.vc_custom_heading

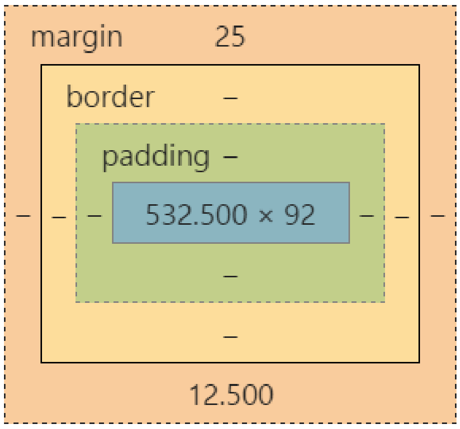
```
margin 25
border -
padding -
847.500 × 46
12.500
```

box-sizing	border-…
color	■rgb(51…
display	block
font-family	Chivo
font-size	42px
font-style	normal
font-weight	normal
height	46px
line-height	46.2px
margin-bottom	12.5px
margin-top	25px
text-align	left
text-size-adjust	100%
width	847.5px
-webkit-margin-after	12.5px
-webkit-margin-before	25px
-webkit-margin-end	0px
-webkit-margin-start	0px
-webkit-tap-highlight-…	rgba(0…

Rendered Fonts

Chivo — Network resource (26 glyphs)

```
JQMIGRATE: Migrate is installed, version 1.4.1        jquery-migrate.min.js?ver=1.4.1:2
```



In the right-top corner you can see the box model (the orange-green-blue box) which starts on the outside with the object's margin. Inside this is the border, then there is padding between the border and the inner contents, and finally there's the object's contents.

These are the properties that affect the box model:

Dimensions
Margin
Border
Padding
Content

Once you have the hang of the box model, you will be well on your way to creating professionally laid-out pages, since these properties alone will make up much of your page styling.
It is especially important to control the appearance of each box on the web page once you start creating the page layout.

Dimensions

A box is by default designed to just hold its contents. To set your own dimensions for a box/element you can use the **height** and **width** properties.

width: 500px;
height: 600px;

These are absolute values, and the element will always have the given size, no matter the size of its content. If the content is smaller it doesn't affect the container but if it is bigger, it will overflow. Though a limit could be given either for maximum or minimum size, just by adding the words **-min** or **-max** to the attributes width or height. When setting a -min value, the element will respect that size if their content is smaller otherwise, it will expand to accommodate the content. The -max attribute is the opposite: it will overflow immediately if the content is bigger than the container. Setting a min-height is recommendable for dynamic content.

```
max-width: 500px;
min-height: 600px;
```

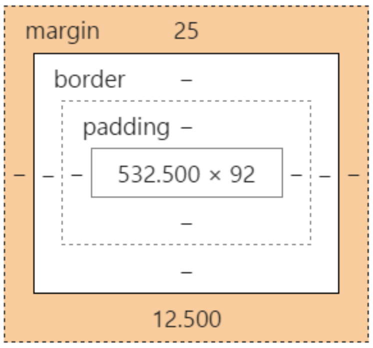
The most popular ways to specify the size of a box are to use pixels, percentages, or ems. Traditionally, pixels have been the most popular method because they allow designers to accurately control their size.

When you use percentages, the size of the box is relative to the size of the browser window or, if the box is encased within another box, it is a percentage of the size of the containing box.

When you use ems, the size of the box is based on the size of text within it.

Margin

The margin property controls the gap between boxes. Its value is commonly given in pixels, although you may also use percentages or ems.

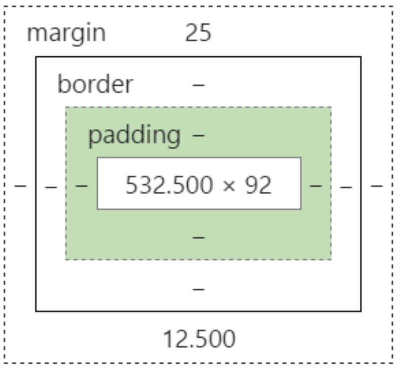


The margins of an element can be changed en masse with the margin property, or individually with margin-left, margin-top, margin-right, and margin-bottom. When setting the margin property, you can supply one, two, three, or four arguments, which have the effects commented in the following rules:

```
/* Set all margins to 1 pixel */
margin:1px;
/* Set top and bottom to 1 pixel, and left and right to 2 */
margin:1px 2px;
/* Set top to 1 pixel, left and right to 2, and bottom to 3 */
margin:1px 2px  3px;
/* Set top to 1 pixel, right to 2, bottom to 3, and left to 4 */
margin :1px 2px 3px 4px;
```

Padding

The **padding** property allows you to specify how much space should appear **between the content of an element and its border**.



The deepest of the box model levels (other than the contents of an element) is the padding, which is applied inside any borders and/or margins. The main properties used to modify padding are **padding, padding-left, padding-top, padding-right, and padding-bottom**.

The four ways of accessing individual property settings used for the margin and the border properties also apply with the padding property, so all the following are valid rules:

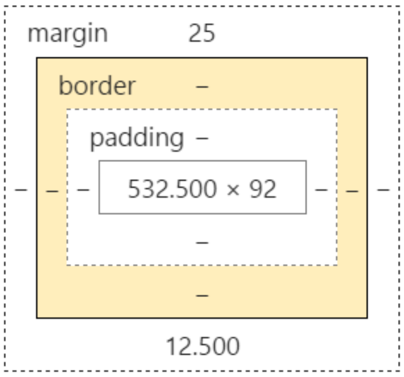
```
/* All padding */
padding: 1px;
/* Top/bottom and left/right */
padding: 1px 2px;
/* Top, left/right and bottom */
padding: 1px 2px 3px;
/* Top, right, bottom and left */
padding: 1px 2px  3px 4px;
```

Or you can use a shorthand (where the values are in **clockwise order: top, right, bottom, left**):

```
padding: 10px 5px 3px  1px;
```

Border

Every box has borders (even if it is not visible or is specified to be 0 pixels wide). The borders separate the edge of one box from another. Every element has 4 borders: **top, right, bottom** and **left.**



Each of those directions has the main properties: **width, style and color.**

The shorthand border property that includes the most popular attributes in only one line of css:

```
border: 1px dashed green;
```

You can also control each border of an element separately, targeting them by specifying the directions:

```
border-bottom: 3px dashed green;
border-top: 5px solid blue;
```

You can control the style of a border using the border-style property. This property can take the following values:

solid a single **solid line**

dashed a series of **short lines**

dotted a series of **square dots** (if the border is 4px wide, then the dots are 4px squared with a 4px gap between them)

double two **solid lines** (the value of the border-width property creates the total width of the two lines)

groove looks to be **carved** into the page

hidden means **no border** is displayed

inset looks **embedded** into the page

outset looks like it is **coming out** of the page

ridge looks to **stick out** from the page

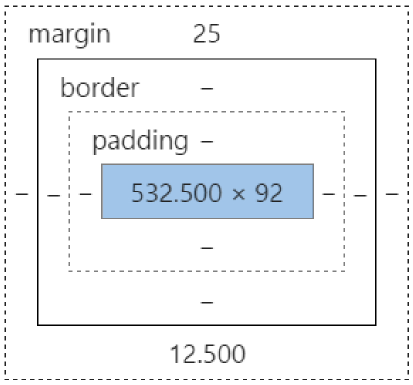
The three attributes **Margin**, **Padding** or **Border-width** can be specified giving 4 different values at once:

```
margin: 10px 5px 6px 2px;
padding: 8px 10px 5px 3px;
border-width: 2px 1px 4px 2px;
```

The sequence that these values are given has its importance, they correspond to a clockwise order: **top, right, bottom, left.**

Object Content

Deep within the box model levels, at its center, lies an element that can be styled in all the ways discussed in this chapter, and which can (and usually will) contain further subelements, which in turn may contain their own subelements, and so on, each with its own styling and box model settings.



For a better understanding in selectors and all things concerning CSS, we recommend taking a look at:
https://www.w3schools.com/css/default.asp

Last modified: Friday, 19 April 2024, 11:51 AM