**What is css ?**

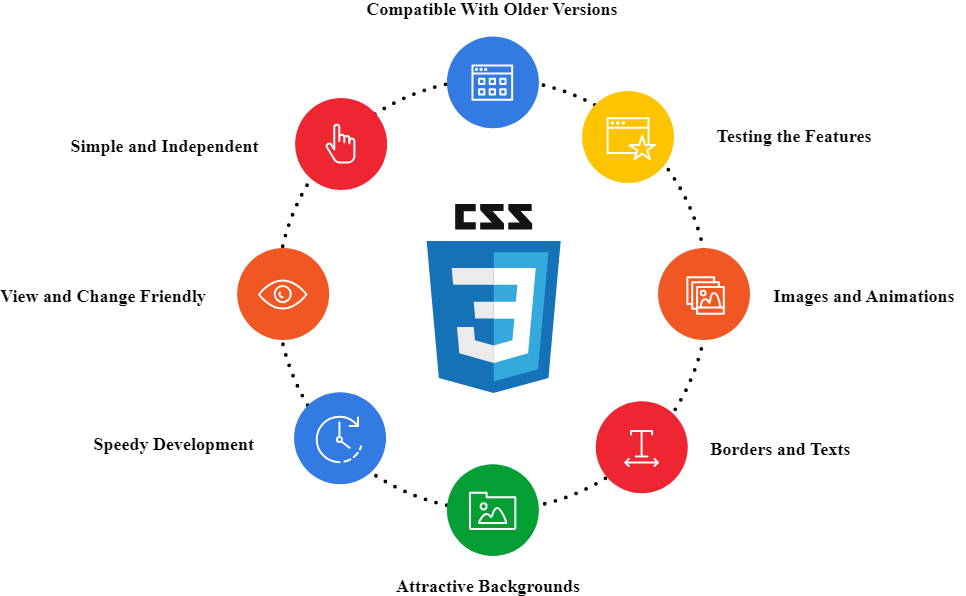
CSS stands for ***Cascading Style Sheets***. It is the language for describing added properties to the presentation of Web pages, including colors, layout, and fonts, thus making web pages presentable to the users.

CSS is designed to make style sheets for the web. It is independent of HTML and can be used with any XML-based markup language.

* **C**ascading: Falling of Styles
* **S**tyle: Adding designs/Styling our HTML tags
* **S**heets: Writing our style in different documents

Features of CSS

* CSS3 new features are used in designing better layouts and attractive web pages easily. It makes the html elements look more appealing to the eyes.
* CSS stands for Cascading Style Sheets. It is a style sheet language which is used to describe the look and formatting of a document written in markup language.
* It provides an additional feature to HTML.
* It is generally used with HTML to change the style of web pages and user interfaces. It can also be used with any kind of XML documents including plain XML, SVG and XUL.
* CSS is used along with HTML and JavaScript in most websites to create user interfaces for web applications and user interfaces for many mobile applications.
* One can add new looks to your old HTML documents.
* One can completely change the look of your website with only a few changes in CSS code.
* CSS style definitions are saved in external CSS files so it is possible to change the entire website by changing just one file.
* CSS provides more detailed attributes than plain HTML to define the look and feel of the website.
* CSS enables the separation of the content from the presentation. This separation provides a lot of flexibility and control over how the website has to look like. This is the main advantage of using CSS.



**Why is CSS important :**

HTML was NEVER intended to contain tags for formatting a web page!

HTML was created to describe the content of a web page, like:

<h1>This is a heading</h1>

<p>This is a paragraph.</p>

When tags like <font>, and color attributes were added to the HTML 3.2 specification, it started a nightmare for web developers. Development of large websites, where fonts and color information were added to every single page, became a long and expensive process.

To solve this problem, the World Wide Web Consortium (W3C) created CSS.

**Pros and cons of CSS ?**

What are the advantages of CSS?

* **Help to Make Spontaneous and Consistent Changes**
* **Saves Time**
* **Improve page loading speed**
* **Make the search engine better crawl your web page**
* **Device compatibility**
* **Ability to Re-position**

**The main advantages of CSS are given below:**

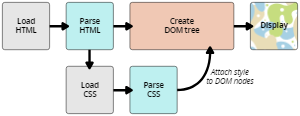
* **Separation of content from presentation - CSS provides a way to present the same content in multiple presentation formats in mobile or desktop or laptop.**
* **Easy to maintain - CSS, built effectively can be used to change the look and feel complete by making small changes. To make a global change, simply change the style, and all elements in all the web pages will be updated automatically.**
* **Bandwidth - Used effectively, the style sheets will be stored in the browser cache and they can be used on multiple pages, without having to download again.**

**What are the disadvantages of CSS?**

* **Cross browser issues** *(CSS works differently on different browsers)*
* **Vulnerability** *(It would only require a read/write access to the intended website to override the changes.)*
* **Confusion due to its many levels**
* **There is no parent selector: Currently, Using CSS, you can’t select a parent tag.**

CSS empowers a web designer to make extensive changes to the web layout of all pages of a website through a single file. It lets you design a light yet responsive website which is fast to load and great at impressing the audience with its display. Hence, every website needs to have its fair share of CSS regardless of the many advantages and disadvantages of CSS.

[How does CSS actually work?](https://developer.mozilla.org/en-US/docs/Learn/CSS/First_steps/How_CSS_works#how_does_css_actually_work)

When a browser displays a document, it must combine the document's content with its style information. It processes the document in a number of stages, which we've listed below. Bear in mind that this is a very simplified version of what happens when a browser loads a webpage, and that different browsers will handle the process in different ways. But this is roughly what happens. 

The diagram also offers a simple view of the process.

1. The browser loads the HTML (e.g. receives it from the network).
2. It converts the [HTML](https://developer.mozilla.org/en-US/docs/Glossary/HTML) into a [DOM](https://developer.mozilla.org/en-US/docs/Glossary/DOM) (*Document Object Model*). The DOM represents the document in the computer's memory. The DOM is explained in a bit more detail in the next section.
3. The browser then fetches most of the resources that are linked to by the HTML document, such as embedded images, videos, and even linked CSS! JavaScript is handled a bit later on in the process, and we won't talk about it here to keep things simpler.
4. The browser parses the fetched CSS, and sorts the different rules by their selector types into different "buckets", e.g. element, class, ID, and so on. Based on the selectors it finds, it works out which rules should be applied to which nodes in the DOM, and attaches style to them as required (this intermediate step is called a render tree).
5. The render tree is laid out in the structure it should appear in after the rules have been applied to it.
6. The visual display of the page is shown on the screen (this stage is called painting).

**How does CSS work under the hood?**

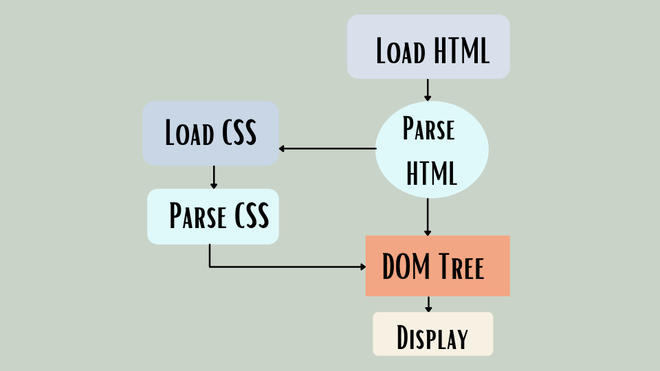
Are you confused whether the HTML will be interpreted before CSS’s interpretation or after? Or will it apply as soon as the browser constructs the [DOM](https://www.geeksforgeeks.org/dom-document-object-model/). Let’s discuss how it actually works. While displaying a document, the browser must combine the document’s content with style information. The document is processed in a number of stages.

**Working:**

* The browser loads the HTML and is converted into DOM (Document Object Model).
* Most of the resources are then fetched that is linked to the HTML document such as embedded videos, images, and linked CSS.
* The CSS fetched is then parsed by the browser. Based on the selector type used, it figures out which style is to be applied on which nodes in the DOM and attaches styles to them. These are called render trees.
* After the rules have been applied to the render tree, its layout structure should appear.
* At the Painting stage, a visual display of the page is shown on the screen.

Now if the CSS is parsed already, the elements in the parsed CSS get styled immediately as soon as they are laid on the page. (Preferred to load in <head>) On the other hand, if CSS is loaded late, elements are shown in their ” unstyled form” until their corresponding styles are parsed as “Flash Of Unstyled Content”.

The below diagram explains the *w*orking of CSS in a more simplified manner.

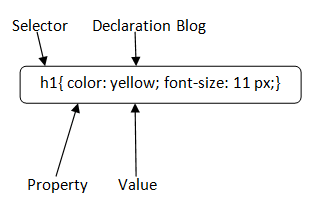


**Limitations of CSS:**

* Confusion among web browsers due to different levels like CSS, CSS 1 up to CSS3.
* Scarce existence of security because of its open text-based system.
* Cross-Browser issues, work differently on different browsers.

CSS Syntax

A CSS rule set contains a selector and a declaration block.



**Selector:** Selector indicates the HTML element you want to style. It could be any tag like <h1>, <title> etc.

**Declaration Block:** The declaration block can contain one or more declarations separated by a semicolon. For the above example, there are two declarations:

1. color: yellow;
2. font-size: 11 px;

Each declaration contains a property name and value, separated by a colon.

**COMMENTS IN CSS**

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

**CSS Selectors:**

A CSS selector selects the HTML element(s) you want to style.

Types of CSS selectors:

Refer :

<https://www.w3schools.com/css/css_selectors.asp>

1. Simple Selectors ( select by id, class, element (tag ) name ).
2. [Combinator selectors](https://www.w3schools.com/css/css_combinators.asp) (select elements based on a specific relationship between them)
3. [Pseudo-class selectors](https://www.w3schools.com/css/css_pseudo_classes.asp) (select elements based on a certain state)
4. [Pseudo-elements selectors](https://www.w3schools.com/css/css_pseudo_elements.asp) (select and style a part of an element)
5. [Attribute selectors](https://www.w3schools.com/css/css_attribute_selectors.asp) (select elements based on an attribute or attribute value)
6. **SIMPLE SELECTORS:**

**All CSS Simple Selectors**

| Selector | Example | Example description |
| --- | --- | --- |
| [#id](https://www.w3schools.com/cssref/sel_id.asp) | #firstname | Selects the element with id="firstname" |
| [.class](https://www.w3schools.com/cssref/sel_class.asp) | .intro  Any element with class name as ‘intro’ | Selects all elements with class="intro" |
| [Element.class](https://www.w3schools.com/cssref/sel_element_class.asp) | p.intro  Element p with class name as ‘intro’ | Selects only <p> elements with class="intro" |
| \*  Universal selector | \* | Selects all elements |
| [element](https://www.w3schools.com/cssref/sel_element.asp) | p | Selects all <p> elements |
| [element,element,..](https://www.w3schools.com/cssref/sel_element_comma.asp) | div, p | Selects all <div> elements and all <p> elements |

Example:

Following code will color all the text under the h1,h2 and p tag as red and align them to center .

h1, h2, p {

text-align: center;

color: red;

}

Example 2:

Following code will color all the text under the p tag with class name as center as red and align them to center .

p.center {

text-align: center;

color: red;

}

1. **CSS COMBINATORS**

A combinator is something that explains the relationship between the selectors.

A CSS selector can contain more than one simple selector. Between the simple selectors, we can include a combinator.

| **There are four different combinators in CSS:**  descendant selector (space)  child selector (>)  adjacent sibling selector (+)  general sibling selector (~) |
| --- |

**Descendant Selector**

The descendant selector matches all elements that are descendants of a specified element.

The following example selects all <p> elements inside <div> elements:

Example

div p {

background-color: yellow;

}

**Child Selector (>)**

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

The following example selects all <p> elements that are children of a <div> element:

Example

div > p {

background-color: yellow;

}

**Adjacent Sibling Selector (+)**

The adjacent sibling selector is used to select an element that is directly (not as child) after another specific element.

Sibling elements must have the same parent element, and "adjacent" means "**immediately following**".

The following example selects the first <p> element that are placed immediately after <div> elements:

Example

div + p {

background-color: yellow;

}

**General Sibling Selector (~)**

The general sibling selector selects all elements that are next siblings of a specified element.

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

Example

div ~ p {

background-color: yellow;

}

1. **PSEUDO-CLASS SELECTORS**

**What are pseudo-classes in CSS?**

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

For example, it can be used to:

Style an element when a user mouses over it

Style visited and unvisited links differently

Style an element when it gets focus

**Syntax**

The syntax of pseudo-classes:

selector:pseudo-class {

property: value;

}

List of Pseudo classes

All CSS Pseudo Classes

| Selector | Example | Example description |
| --- | --- | --- |
| [:active](https://www.w3schools.com/cssref/sel_active.asp) | a:active | Selects the active link |
| [:checked](https://www.w3schools.com/cssref/sel_checked.asp) | input:checked | Selects every checked <input> element |
| [:disabled](https://www.w3schools.com/cssref/sel_disabled.asp) | input:disabled | Selects every disabled <input> element |
| [:empty](https://www.w3schools.com/cssref/sel_empty.asp) | p:empty | Selects every <p> element that has no children |
| [:enabled](https://www.w3schools.com/cssref/sel_enabled.asp) | input:enabled | Selects every enabled <input> element |
| [:first-child](https://www.w3schools.com/cssref/sel_firstchild.asp) | p:first-child | Selects every <p> elements that is the first child of its parent |
| [:first-of-type](https://www.w3schools.com/cssref/sel_first-of-type.asp) | p:first-of-type | Selects every <p> element that is the first <p> element of its parent |
| [:focus](https://www.w3schools.com/cssref/sel_focus.asp) | input:focus | Selects the <input> element that has focus |
| [:hover](https://www.w3schools.com/cssref/sel_hover.asp) | a:hover | Selects links on mouse over |
| [:in-range](https://www.w3schools.com/cssref/sel_in-range.asp) | input:in-range | Selects <input> elements with a value within a specified range |
| [:invalid](https://www.w3schools.com/cssref/sel_invalid.asp) | input:invalid | Selects all <input> elements with an invalid value |
| [:lang(language)](https://www.w3schools.com/cssref/sel_lang.asp) | p:lang(it) | Selects every <p> element with a lang attribute value starting with "it" |
| [:last-child](https://www.w3schools.com/cssref/sel_last-child.asp) | p:last-child | Selects every <p> elements that is the last child of its parent |
| [:last-of-type](https://www.w3schools.com/cssref/sel_last-of-type.asp) | p:last-of-type | Selects every <p> element that is the last <p> element of its parent |
| [:link](https://www.w3schools.com/cssref/sel_link.asp) | a:link | Selects all unvisited links |
| [:not(selector)](https://www.w3schools.com/cssref/sel_not.asp) | :not(p) | Selects every element that is not a <p> element |
| [:nth-child(n)](https://www.w3schools.com/cssref/sel_nth-child.asp) | p:nth-child(2) | Selects every <p> element that is the second child of its parent |
| [:nth-last-child(n)](https://www.w3schools.com/cssref/sel_nth-last-child.asp) | p:nth-last-child(2) | Selects every <p> element that is the second child of its parent, counting from the last child |
| [:nth-last-of-type(n)](https://www.w3schools.com/cssref/sel_nth-last-of-type.asp) | p:nth-last-of-type(2) | Selects every <p> element that is the second <p> element of its parent, counting from the last child |
| [:nth-of-type(n)](https://www.w3schools.com/cssref/sel_nth-of-type.asp) | p:nth-of-type(2) | Selects every <p> element that is the second <p> element of its parent |
| [:only-of-type](https://www.w3schools.com/cssref/sel_only-of-type.asp) | p:only-of-type | Selects every <p> element that is the only <p> element of its parent |
| [:only-child](https://www.w3schools.com/cssref/sel_only-child.asp) | p:only-child | Selects every <p> element that is the only child of its parent |
| [:optional](https://www.w3schools.com/cssref/sel_optional.asp) | input:optional | Selects <input> elements with no "required" attribute |
| [:out-of-range](https://www.w3schools.com/cssref/sel_out-of-range.asp) | input:out-of-range | Selects <input> elements with a value outside a specified range |
| [:read-only](https://www.w3schools.com/cssref/sel_read-only.asp) | input:read-only | Selects <input> elements with a "readonly" attribute specified |
| [:read-write](https://www.w3schools.com/cssref/sel_read-write.asp) | input:read-write | Selects <input> elements with no "readonly" attribute |
| [:required](https://www.w3schools.com/cssref/sel_required.asp) | input:required | Selects <input> elements with a "required" attribute specified |
| [:root](https://www.w3schools.com/cssref/sel_root.asp) | root | Selects the document's root element |
| [:target](https://www.w3schools.com/cssref/sel_target.asp) | #news:target | Selects the current active #news element (clicked on a URL containing that anchor name) |
| [:valid](https://www.w3schools.com/cssref/sel_valid.asp) | input:valid | Selects all <input> elements with a valid value |
| [:visited](https://www.w3schools.com/cssref/sel_visited.asp) | a:visited | Selects all visited links |

Examples:

Refer: <https://www.w3schools.com/css/css_pseudo_classes.asp>

* Anchor pseudo-classes are: link, visited, hover, active
* Pseudo-classes can be combined with HTML classes:

When you hover over the link in the example, it will change color:

Example: here element a with classname highlight will change its color when mouse hovers on it.

a.highlight:hover {

color: #ff0000; }

* Example to show tooltip hover: this will make text inside p tag visible when hover on div element.

p {

display: none;

background-color: yellow;

padding: 20px;

}

div:hover p {

display: block;

}

1. **PSEUDO-ELEMENT SELECTORS**

**What are Pseudo-Elements?**

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  The syntax of pseudo-elements:  selector::pseudo-element {  property: value;  } |
| --- |

**All CSS Pseudo Elements**

| **Selector** | **Example** | **Example description** |
| --- | --- | --- |
| [::after](https://www.w3schools.com/cssref/sel_after.asp) | p::after | Insert something after the content of each <p> element |
| [::before](https://www.w3schools.com/cssref/sel_before.asp) | p::before | Insert something before the content of each <p> element |
| [::first-letter](https://www.w3schools.com/cssref/sel_firstletter.asp) | p::first-letter | Selects the first letter of each <p> element |
| [::first-line](https://www.w3schools.com/cssref/sel_firstline.asp) | p::first-line | Selects the first line of each <p> element |
| [::marker](https://www.w3schools.com/cssref/sel_marker.asp) | ::marker | Selects the markers of list items |
| [::selection](https://www.w3schools.com/cssref/sel_selection.asp) | p::selection | Selects the portion of an element that is selected by a user |

For examples refer: <https://www.w3schools.com/css/css_pseudo_elements.asp>

1. **ATTRIBUTE SELECTORS**

The [attribute] selector is used to select elements with a specified attribute.

The following example selects all <a> elements with a target attribute set in it :

Example

a[target] {

background-color: yellow; }

This attribute selector is useful in styling common fragments like elements inside Forms, all anchor elements , all buttons of your web site etc.,

| **The attribute selectors can be useful for styling forms without class or ID:**  Example  input[type="text"] {  width: 150px;  display: block;  margin-bottom: 10px;  background-color: yellow; }  input[type="button"] {  width: 120px;  margin-left: 35px;  display: block; } |
| --- |

**All CSS Attribute Selectors: attribute is all that is used in your html code while defining the particular element.**

| **Selector** | **Example** | **Example description** |
| --- | --- | --- |
| [[*attribute*]](https://www.w3schools.com/cssref/sel_attribute.php) | [target] | Selects all elements with a target attribute |
| [[*attribute*=*value*]](https://www.w3schools.com/cssref/sel_attribute_value.php) | [target="\_blank"] | Selects all elements with target="\_blank" |
| [[*attribute*~=*value*]](https://www.w3schools.com/cssref/sel_attribute_value_contains.php) | [title~="flower"] | Selects all elements with a title attribute that contains a space-separated list of words, one of which is "flower" |
| [[*attribute*|=*value*]](https://www.w3schools.com/cssref/sel_attribute_value_lang.php) | [lang|="en"] | Selects all elements with a lang attribute value starting with "en" |
| [[*attribute*^=*value*]](https://www.w3schools.com/cssref/sel_attr_begin.php) | a[href^="https"] | Selects all <a> elements with a href attribute value starting with "https" |
| [[*attribute*$=*value*]](https://www.w3schools.com/cssref/sel_attr_end.php) | a[href$=".pdf"] | Selects all <a> elements with a href attribute value ending with ".pdf" |
| [[*attribute*\*=*value*]](https://www.w3schools.com/cssref/sel_attr_contain.php) | a[href\*="w3schools"] | Selects all <a> elements with a href attribute value containing the substring "w3schools" |

**CSS FUNCTIONS**

**Refernce:** [**https://www.w3schools.com/cssref/css\_functions.php**](https://www.w3schools.com/cssref/css_functions.php)

DOM tree structure]

**A Document Object Model (DOM) tree is a hierarchical representation of an HTML or** [**XML**](https://en.wikipedia.org/wiki/XML) **document. It consists of a root node, which is the document itself, and a series of child nodes that represent the elements, attributes, and text content of the document. Each node in the tree has a parent node, except for the root node, and can have multiple child nodes.**

**Elements as Nodes**[[edit](https://en.wikipedia.org/w/index.php?title=Document_Object_Model&action=edit&section=7)]

**Elements in an HTML or XML document are represented as nodes in the DOM tree. Each element node has a tag name, attributes, and can contain other element nodes or text nodes as children. For example, an HTML document with the following structure:**

**<html>**

**<head>**

**<title>My Website</title>**

**</head>**

**<body>**

**<h1>Welcome</h1>**

**<p>This is my website.</p>**

**</body>**

**</html>**

**will be represented in the DOM tree as:**

**- Document (root)**

**- html**

**- head**

**- title**

**- "My Website"**

**- body**

**- h1**

**- "Welcome"**

**- p**

**- "This is my website."**

**Text Nodes**[[edit](https://en.wikipedia.org/w/index.php?title=Document_Object_Model&action=edit&section=8)]

**Text content within an element is represented as a text node in the DOM tree. Text nodes do not have attributes or child nodes, and are always leaf nodes in the tree. For example, the text content "My Website" in the title element and "Welcome" in the h1 element in the above example are both represented as text nodes.**

**Attributes as properties**[[edit](https://en.wikipedia.org/w/index.php?title=Document_Object_Model&action=edit&section=9)]

**Attributes of an element are represented as properties of the element node in the DOM tree. For example, an element with the following HTML:**

**<a href="https://example.com">Link</a>**

**will be represented in the DOM tree as:**

**- a**

**- href: "https://example.com"**

**- "Link"**