* CSS plays an important role, by using CSS you simply got to specify a repeated style for element once & use it multiple times as because CSS will automatically apply the required styles.
* The main advantage of CSS is that style is applied consistently across variety of sites. One instruction can control several areas which is advantageous.
* Web designers needs to use few lines of programming for every page improving site speed.
* Cascading sheet not only simplifies website development, but also simplifies the maintenance as a change of one line of code affects the whole web site and maintenance time.
* It is less complex therefore the effort are significantly reduced.
* It helps to form spontaneous and consistent changes.
* CSS changes are device friendly. With people employing a batch of various range of smart devices to access websites over the web, there’s a requirement for responsive web design.
* It has the power for re-positioning. It helps us to determine the changes within the position of web elements who are there on the page.
* These bandwidth savings are substantial figures of insignificant tags that are indistinct from a mess of pages.
* Easy for the user to customize the online page
* It reduces the file transfer size.
* CSS, CSS 1 up to CSS3, result in creating of confusion among  web browsers.
* With CSS, what works with one browser might not always work with another. The web developers need to test for compatibility, running the program across multiple browsers.
* There exists a scarcity of security.
* After making the changes we need to confirm the compatibility if they appear. The similar change affects on all the browsers.
* The programing language world is complicated for non-developers and beginners. Different levels of CSS i.e. CSS, CSS 2, CSS 3 are often quite confusing.
* Browser compatibility (some styles sheet are supported and some are not).
* CSS works differently on different browsers. IE and Opera supports CSS as different logic.
* There might be cross-browser issues while using CSS.
* There are multiple levels which creates confusion for non-developers and beginners.
* CSS3 is split into many various documents known as Modules. each module adds new capability or extends options outlined in CSS2 over conserving backward compatibility. Work on CSS3 started around the time of publication of the initial CSS2 recommendation.
* The CSS3 version supports more browsers than CSS2.
* CSS3 introduces several new selectors. Those new selectors square measure largely in an exceeding type of pseudo-elements and pseudo-categories.
* The new addition of General relation Combinator will be wont to match relation parts of a given part through diacritic (~) combinator.
* CSS3 introduces several properties attended with new values and units. It facilitates styling of backgrounds, borders, boxes, etc…, that permits the USA to stay most of the styling at intervals the computer network and HTML standards and our document, while not a necessity for all those proprietary third-party package packages.
* New values and new units square measure introduced to support all those new properties. for example, Angle units deg, grad, rad, and switch or Time units s and ms.
* **Automatic critical CSS**: styled-components keeps track of which components are rendered on a page and injects their styles and nothing else, fully automatically. Combined with code splitting, this means your users load the least amount of code necessary.
* **No class name bugs**: styled-components generates unique class names for your styles. You never have to worry about duplication, overlap or misspellings.
* **Easier deletion of CSS**: it can be hard to know whether a class name is used somewhere in your codebase. styled-components makes it obvious, as every bit of styling is tied to a specific component. If the component is unused (which tooling can detect) and gets deleted, all its styles get deleted with it.
* **Simple dynamic styling**: adapting the styling of a component based on its props or a global theme is simple and intuitive without having to manually manage dozens of classes.
* **Painless maintenance**: you never have to hunt across different files to find the styling affecting your component, so maintenance is a piece of cake no matter how big your codebase is.
* **Automatic vendor prefixing**: write your CSS to the current standard and let styled-components handle the rest

**CSS Opacity**

The CSS opacity property is used to specify the transparency of an element. In simple word, you can say that it specifies the clarity of the image.

In technical terms, Opacity is defined as degree in which light is allowed to travel through an object.

**How to apply CSS opacity setting**

Opacity setting is applied uniformly across the entire object and the opacity value is defined in term of digital value less than 1. The lesser opacity value displays the greater opacity. Opacity is not inherited.

Example :

<!DOCTYPE html>

<html>

<head>

<style>

img.trans {

    opacity: 0.4;

    filter: alpha(opacity=40); /\* For IE8 and earlier \*/

}

</style>

</head>

<body>

<p>Normal Image</p>

<img src="rose.jpg" alt="normal rose">

<p>Transparent Image</p>

<img class="trans" src="rose.jpg" alt="transparent rose">

</body>

</html>

The *value* of the *background-color* property can be specified in three ways:

1. Using *RGB values* e.g. rgb(250,0,0), rgb(0,250,0), etc.
2. Using a *Hexadecimal(HEX) value* e.g. #FFFFFF, #000000, etc.
3. Using a valid *color name* e.g. “white”, “red”, “black”, “blue”, etc.

The code snippet below sets the background color for the entire body, <div>, <h2> and <p> elements of the web page.

body {

background-color: #FCD6CB;

}

div {

background-color: blue;

}

h2 {

background-color: #74C3E1;

}

P {

background-color: rgb(250,0,0);

}

In this article, we will see how an image repetition of the backup is controlled in CSS. This task can be achieved by using the *background-repeat property* that will help us to control the repetition of the image.

The **background-repeat property** in CSS is used to repeat the background image both horizontally and vertically. It also decides whether the background image will be repeated or not.

**Syntax:**

background-repeat: repeat|repeat-x|repeat-y|no-repeat|initial|inherit;

**Example 1:** In the example, we will make use of the repeat-x to repeat the image in the horizontal direction.

<!DOCTYPE html>

<html>

<head>

<title>background-repeat property</title>

<style>

body {

margin-top: 40px;

background-image: url(

"https://media.geeksforgeeks.org/wp-content/uploads/geeks-25.png");

background-repeat: repeat-x;

background-size: 150px 100px;

}

h1 {

text-align: center

}

</style>

</head>

<body>

<h1>GeeksforGeeks</h1>

</body>

</html>

The background-position property **sets the starting position of a background image**. Tip: By default, a background-image is placed at the top-left corner of an element, and repeated both vertically and horizontally.

background-attachment property

The **background-attachment** property sets whether a background image scrolls with the rest of the page, or is fixed.

There are two reasons behind this: **It enhances the legibility of style sheets**. The background property is a complex property in CSS, and if it is combined with color, the complexity will further increase.

There are two ways of centering block level elements:  
  
1. By setting the properties margin-left and margin-right to auto and width to some explicit value:  
  
BODY {width: 30em; background: cyan;}  
P {width: 22em; margin-left: auto; margin-right: auto}  
  
In this case, the left and right margins will each be four ems wide, since they equally split up the eight ems left over from (30em - 22em). Note that it was not necessary to set an explicit width for the BODY element; it was done here to keep the math clean.  
  
Another example:  
  
TABLE {margin-left: auto; margin-right: auto; width: 400px;}   
In most legacy browsers, a table's width is by default determined by its content. In CSS-conformant browsers, the complete width of any element (including tables) defaults to the full width of its parent element's content area. As browser becaome more conformant, authors will need to be aware of the potential impact on their designs.

The Specification defines how CSS properties should be implemented by browser vendors along with detailed algorithms, code samples and tabular information.

The Specification also include:

* The syntax and data types of the language
* Detailed explanation on CSS Selectors
* How you can assign values to properties
* The Cascade (the "C" in CSS)
* How inheritance works
* The Box Model e.t.c

Explanation on some of these topic are short and easy to understand while others are explained in great detail.

The Specification also specify how stylesheets can be included in your web document and how to target specific media e.g print or screen.

The CSS Specification **prior to CSS3 was a single Specification**, CSS3 on the other hand **is divided into Modules** which are **Independent Specifications** that can be worked on by different author(s) at different paces, that's why we have Selector Level 3 Specification, CSS Color 4, CSS Backgrounds and so on. Some of these modules are revisions of CSS2.1, and some are newly created, but all fall under the banner of CSS3.

The Specification should be your guide if you need to understand how a specific property or feature works behind the scene and how it works with other CSS properties. And if you are comfortable reading algorithms you won't get bored reading the CSS Specification.

There are three ways to integrate CSS into a Web page

1. Inline: HTML elements may have CSS applied to them via the STYLE attribute.

2. Embedded: By placing the code in a STYLE element within the HEAD element.

3. Linked/ Imported: Place the CSS in an external file and link it via a link element.

Embedded style sheets refer to when you embed style sheet information into an HTML document using the [<style>](https://www.quackit.com/html/tags/html_style_tag.cfm) element. You do this by embedding the style sheet information within [<style></style>](https://www.quackit.com/html/tags/html_style_tag.cfm) tags in the head of your document.

## Syntax

The CSS syntax for embedded style sheets is exactly the same as other CSS code.

For example, to use the following code, simply place it between the [<head></head>](https://www.quackit.com/html/tags/html_head_tag.cfm) tags of your HTML document:

<style>

p {

font-family: georgia, serif;

font-size: x-small;

}

hr {

color: #ff9900;

height: 1px;

}

a:hover {

color: #ff0000;

text-decoration: none;

}

</style>

Now, whenever any of those elements are used within the body of the document, they will be formatted as instructed in the above style sheet.

<!DOCTYPE html>

<html>

<head>

<style type="text/css">

p {

font-family: georgia, serif;

font-size: x-large;

color:#ff9900;

}

a:hover {

color: LimeGreen;

text-decoration: none;

}

</style>

</head>

<body>

<p>This paragraph tag has been preset using embedded style sheets. Same for the horizontal rule below and the hyperlink under that. </p>

<p>Also see <a href="/css/external\_style\_sheets.cfm" target="\_blank">external style sheets</a></p>

</body>

</html>

An external style sheet is a separate file where you can declare all the styles that you want to use on your website. You then link to the external style sheet from all your HTML pages.

This means you only need to set the styles for each element once. If you want to update the style of your website, you only need to do it in one place.

## Example

### Create the Style Sheet

Type CSS code into a plain text file, and save with a .css extension (for example, styles.css).

Here's an example of some CSS code.

body {

background-color: darkslategrey;

color: azure;

font-size: 1.1em;

}

h1 {

color: coral;

}

#intro {

font-size: 1.3em;

}

.colorful {

color: orange;

}

### Link to the Style Sheet from the HTML Documents

Add the following code between the [<head></head>](https://www.quackit.com/html/tags/html_head_tag.cfm) tags of all HTML documents that you want to reference the external style sheet. This code uses the HTML [<link>](https://www.quackit.com/html/tags/html_link_tag.cfm) element to link to the external style sheet.

<link rel="stylesheet" href="styles.css">

So, a basic example of a web page using this style sheet might look something like this:

<!DOCTYPE html>

<html>

<head>

<title>My Example</title>

<link rel="stylesheet" href="styles.css">

</head>

<body>

<h1>External Styles</h1>

<p id="intro">Allow you to define styles for the whole website.</p>

<p class="colorful">This has a style applied via a class.</p>

</body>

</html>

**The advantages of External Style Sheets are as follows :**

* With the help of External Style Sheets, the styles of numerous documents can be organized from one single file.
* In External Style Sheets, Classes can be made for use on numerous HTML element types in many forms of the site.
* In complex contexts, Methods like selector and grouping can be implemented to apply styles.

**The disadvantages of External Style Sheets are as follows :**

* An extra download is essential to import style information for each file.
* The execution of the file may be deferred till the external style sheet is loaded.
* While implementing style sheets, we need to test Web pages with multiple browsers in order to check compatibility issues.

**CSS selectors** are used *to select the content you want to style*. Selectors are the part of CSS rule set. CSS selectors select HTML elements according to its id, class, type, attribute etc.

There are several different types of selectors in CSS.

1. CSS Element Selector
2. CSS Id Selector
3. CSS Class Selector
4. CSS Universal Selector
5. CSS Group Selector

|  |  |
| --- | --- |
| **Sr.No.** | **Value & Description** |
| 1 | **all**  Suitable for all devices. |
| 2 | **aural**  Intended for speech synthesizers. |
| 3 | **braille**  Intended for braille tactile feedback devices. |
| 4 | **embossed**  Intended for paged braille printers. |
| 5 | **handheld**  Intended for handheld devices (typically small screen, monochrome, limited bandwidth). |
| 6 | **print**  Intended for paged, opaque material and for documents viewed on screen in print preview mode. Please consult the section on paged media. |
| 7 | **projection**  Intended for projected presentations, for example projectors or print to transparencies. Please consult the section on paged media. |
| 8 | **screen**  Intended primarily for color computer screens. |
| 9 | **tty**  Intended for media using a fixed-pitch character grid, such as teletypes, terminals, or portable devices with limited display capabilities. |
| 10 | **tv**  Intended for television-type devices. |

**A collection of rules or signatures that network traffic or system activity is compared against to determine an action to take**—such as forwarding or rejecting a packet, creating an alert, or allowing a system event.

 n the Project window, click the module in which you want to add a layout.

 In the main menu, select File > New > XML > Layout XML File.

 In the dialog that appears, provide the file name, the root layout tag, and the source set in which the layout belongs.

 Click Finish to create the layout.