What are the benefits of using CSS?

CSS, Cascading Style Sheets define the presentation of the document that is written in the HTML. This document is delivered to the user, whether shown on a computer screen, on a cell phone, printed on paper or read aloud by a screen reader.

There are several benefits of using the style sheets.

- **Precise type and layout controls:** CSS provides print-like precision.
- **Less work:** Changing the appearance of an entire website is possible by editing one style sheet which ensures the consistency of formatting throughout the site.
- **More accessible sites:** When every aspect of presentation is handled by CSS, we can mark up the conent more meaningfully which makes it more accessible for non-visual or mobile devices.
- **Better or faster data-driven websites:** CSS improves the page loading speed. The code density on the website contributes to its speed. The larger the code, the slower the website's loading. Only a few code lines are required to make changes to a large number of pages on a website with CSS which helps in **Page Speed Optimization**

What are the disadvantages of CSS?

There really are not many disadvantages to using style sheets. There are some lingering hassles from browser inconsistencies, but they can either be avoided or worked around.

Here are few disadvantages that could be kept in mind while working with CSS;

- **Cross browser issues:** CSS works differently on different browser. Due to that, implementing the initial CSS changes on a website would be accessible on developer's end, however, it is very imperative to confirm the compatibility on all the browsers.
- **Confusion due to its many levels:** different versions of the CSS such as CSS, CSS2, CSS3 can be confusing for users.
- **Fixing HTML tags:** It is often necessary to fix not only one CSS file but also HTML tags associated with particular CSS selectors. It significantly increases the editing and testing time as well.
- **Different layout display:** If the browser is outdated then CSS data may be interpreted differently.
- **Vulnerability:** CSS is easily accessible because of its open text-based system. An accident or a mere act of mischief with files can end up disrupting the display and formatting of entire website.

What is the difference between CSS2 and CSS3?

CSS2	CSS3
1. CSS2 has browser extension issues.	1. CSS3 has complete support for almost all recent web browsers.
2. CSS2 has limited styling and some old design features.	2. CSS3 has all the latest styling properties as well as an addition of animation property as well.
3. CSS2 does not have compatibility with external font styles through google fonts and typecast.	3. CSS3 does have compatibility with external font styles through google fonts and typecast.
4. CSS2 had only few simple selectors.	4. CSS3 has increased amount of selectors.
5. CSS2 did not have provision to specifically design the web layout.	5. CSS3 has grid system and template layout module. It is helping in creating layout according to user components.
6. CSS2 was all in a single document with limited features.	6. CSS3 is split into various documents known as modules. The modularisation helped in working on a particular specification and faster evolution.
7. CSS2 was not supporting responsive web design.	7. CSS3 supports responsive design in such a manner that it looks good on all screen sizes.

Name a few CSS style components.

Below are the CSS style components which acts as the root to applying styling to HTML elements.

1. **Selector:**
   - This is the first part of a CSS rule which encapsulates the set of elements to which the rule applies.
   - Typically, this will be an ID selector- **#element-id** ,a Class selector- **.class-name** ,an Element selector- **div/section/p/img/....** ,or something more complex involving hierarchy or location- **:first-child/::before/::after** perhaps.

   Next, enclosed within a pair of curly braces, comes the **rule-set.** This is one or more rules which apply a particular style to the element(s) matched by the selector.
2. **Property:**
   - Propert is the style, such as margin, padding, or one of the many other stylable properties.
3. **Value:**
   - The **value** is the value to set in pair of the property such as margin, or padding, that may be unit-ed value like 15px, 10%, 1em or some other CSS- compatible value like a color name, a keyword, such as **right** for float.

For ex.:

```
p{
   color: red;
   position: relative;
   text-align: center;
}
```

Here in the example, **p** is selector, following style rules will be applied to all the paragraphs of the content.

color is property and its value is **red,**

position is another property and its value is **relative,**

text-align is another property with value of **center.**

What do you understand by CSS opacity?

The **Opacity** setting applies to the entire element-both the foreground and the background. The by-default value of opacity is set to 1.

The value for opacity is a number between 0(completely transparent) and 1(completely opaque). As opacity applies to the entire element but if there is a need of affecting just one, use an **RGBa or HSLa** color value instead.

For ex.:

1. When **Opacity=0.25**

```
<div style="width:25%; height:auto; background-color: #2b537a; margin-left: 5%;">
<h1 style="color: gold; background: white; opacity: 0.25;"> Playing with opacity </h1>
```

Playing with opacity

2. When **Opacity=0.5**

```
<div style="width:25%; height:auto; background-color: #2b537a; margin-left: 5%;">
<h1 style="color: gold; background: white; opacity: 0.5;"> Playing with opacity </h1>
```

Playing with opacity

3. When **Opacity=1**

```
<div style="width:25%; height:auto; background-color: #2b537a; margin-left: 5%;">
<h1 style="color: gold; background: white; opacity: 1;"> Playing with opacity </h1>
```

Playing with opacity

How can the background color of an element be changed?

The **background-color:** property is used to specify the background color of the element. The by-default value of background-color property is set to **transparent** . In addition to that, it accepts **color values(name or numeric).**

In order for coloring the background of whole page, we apply the background-color property to the body element. This is considered as one of the important exemption.

For ex.:

In the example below, I have used the **color and background-color** properties to highlight a word marked as a "glossary" class.

```
.glossary{
  color: #fff;
  background-color:#2b537a;
}
<p style="margin-left-5%;">Every variety of apple had their origin in Kashmir(<dfn class="glossary" style="color: #fff; background-color: #2b537a;"><i>Red Apple</i></dfn>)</p>
```

Every variety of apple had their origin in Kashmir(*Red Apple*)

How can image repetition of the background be controlled?

We can control the image repetition of the background by using the **background-repeat:** property. The by-default value of background-repeat property is set to **repeat** . In addition to that, it also accepts other values such as **no-repeat|repeat-x|repeat-y|space|round.**

For ex.:

1. If there is a need of repeating the background-image just once, we can use the **no-repeat** as a value:

```
div{
  width: 30%;
  height: 150px;
  background-color: #fff;
  background-image: url(https://images.pexels.com/photos/1445416/pexels-photo-1445416.jpeg?auto=compress&cs=tinysrgb&w=1260&h=750&dpr=1);
  background-size: 20%;
  background-repeat: no-repeat;
  margin-left: 5%;
}
```

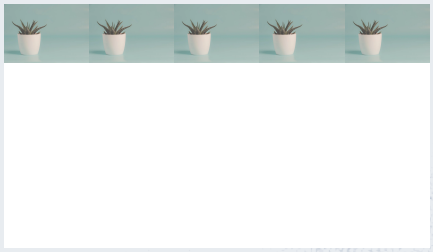
2. If there is a need of repeating the background-image, we can use the **repeat** as a value:

```
div{
  width: 30%;
```

```
    height: 150px;
    background-color: #fff;
    background-image: url(https://images.pexels.com/photos/1445416/pexels-photo-1445416.jpeg?
auto=compress&cs=tinysrgb&w=1260&h=750&dpr=1);
    background-size: 20%;
    background-repeat: repeat;
    margin-left: 5%;
}
```

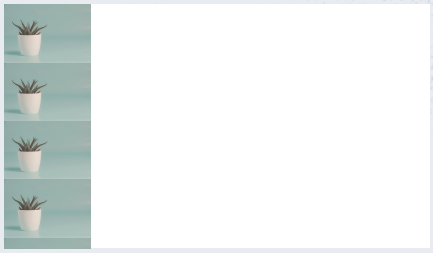
3. If there is a need of repeating the background-image only horizontally, we can use the **repeat-x** as a value:

```
div{
    width: 30%;
    height: 150px;
    background-color: #fff;
    background-image: url(https://images.pexels.com/photos/1445416/pexels-photo-1445416.jpeg?
auto=compress&cs=tinysrgb&w=1260&h=750&dpr=1);
    background-size: 20%;
    background-repeat: repeat-x;
    margin-left: 5%;
}
```



4. If there is a need of repeating the background-image only vertically, we can use the **repeat-y** as a value:

```
div{
    width: 30%;
    height: 150px;
    background-color: #fff;
    background-image: url(https://images.pexels.com/photos/1445416/pexels-photo-1445416.jpeg?
auto=compress&cs=tinysrgb&w=1260&h=750&dpr=1);
    background-size: 20%;
    background-repeat: repeat-y;
    margin-left: 5%;
}
```



5. When background-repeat is set to **space,** the browser calculates how many background images can fit across the width and height of the background area, then adds equal amounts of space between each image. The result is even rows and columns and no clipped images,then we can use the **space** as a value:

```
div{
   width: 30%;
   height: 150px;
   background-color: #fff;
   background-image: url(https://images.pexels.com/photos/1445416/pexels-photo-1445416.jpeg?
auto=compress&cs=tinysrgb&w=1260&h=750&dpr=1);
   background-size: 20%;
   background-repeat: space;
   margin-left: 5%;
}
```

6. When background-repeat is set to **round,** it makes the browser squish the background image horizontally and vertically (not necessarily proportionally) to fit in the background area an even number of times, then we can use the **round** as a value:

```
div{
   width: 30%;
   height: 150px;
   background-color: #fff;
   background-image: url(https://images.pexels.com/photos/1445416/pexels-photo-1445416.jpeg?
auto=compress&cs=tinysrgb&w=1260&h=750&dpr=1);
   background-size: 20%;
   background-repeat: round;
   margin-left: 5%;
}
```

What is the use of background-position property?

The **background-position** property specifies the position of the origin image in the background. Origin image is the image that is placed as the first image in the background. The by-default value of background-position property is set to **0% 0%(Same as left top)** . In addition to that, it accepts **length measurements(in px or em)|percentage(%)|left|center|right|top|bottom.**

To position the origin image provide the horizontal and vertical values that describe where to place it.

For ex.:

1. **Method 1:** Keyword Positioning

   The Keyword values **left, right, top, bottom and center** position the origin image relative to the element's outer edge. Here, I have used the values **left bottom** in first image and **right center** in the second image to explain further.

```
div{
   width: 30%;
   height: 150px;
   background-color: #fff;
   background-image: url(https://images.pexels.com/photos/1445416/pexels-photo-1445416.jpeg?
auto=compress&cs=tinysrgb&w=1260&h=750&dpr=1);
```

```
    background-size: 20%;
    background-repeat: no-repeat;
    background-position: left bottom;
    margin-left: 5%;
}
```

```
div{
    width: 30%;
    height: 150px;
    background-color: #fff;
    background-image: url(https://images.pexels.com/photos/1445416/pexels-photo-1445416.jpeg?
auto=compress&cs=tinysrgb&w=1260&h=750&dpr=1);
    background-size: 20%;
    background-repeat: no-repeat;
    background-position: right center;
    margin-left: 5%;
}
```

NOTE: If we provide only one keyword, then the missing keyword is assumed to be **center.**

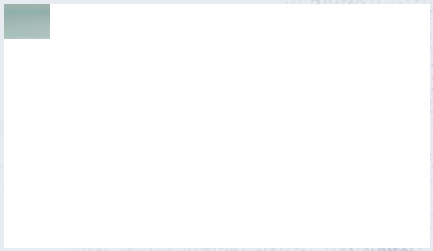
2. **Method 2:** Length Measurements(in px or em)

   Specifying position using length measurements in pixels or ems indicates an amount of offset from the top-left corner of the element to the top-left corner of the background origin image.

   When providing the length values, **the horizontal measurements always goes first.** Negative values are also allowed but it will make the image to hang outside the background-area. Here, I have used the values **100px 50px** and it will position the image 100px from the left edge and 50px down from the top edge of the element. The second image is using the negative values.

```
div{
    width: 30%;
    height: 150px;
    background-color: #fff;
    background-image: url(https://images.pexels.com/photos/1445416/pexels-photo-1445416.jpeg?
auto=compress&cs=tinysrgb&w=1260&h=750&dpr=1);
    background-size: 20%;
    background-repeat: no-repeat;
    background-position: 100px 50px;
    margin-left: 5%;
}
```

```
div{
   width: 30%;
   height: 150px;
   background-color: #fff;
   background-image: url(https://images.pexels.com/photos/1445416/pexels-photo-1445416.jpeg?
auto=compress&cs=tinysrgb&w=1260&h=750&dpr=1);
   background-size: 20%;
   background-repeat: no-repeat;
   background-position: -25px -15px;
   margin-left: 5%;
}
```



3. **Method 3:** Percentages

   Specifying percentage position using horizontal/vertical pairs, with **0% 0%** corresponding to the top-left corner and **100% 100%** corresponding to the bottom-right corner.

   It is important to note that **the percentage value applies to both the background area and image itself.** A horizontal value of 25% positions the point 25% from the left edge of the image at a point that is 25% from the left edge of the background positioning area. A vertical value of 100% positions the bottom edge of the image at the bottom edge of the positioning area.

```
div{
   width: 30%;
   height: 150px;
   background-color: #fff;
   background-image: url(https://images.pexels.com/photos/1445416/pexels-photo-1445416.jpeg?
auto=compress&cs=tinysrgb&w=1260&h=750&dpr=1);
   background-size: 20%;
   background-repeat: no-repeat;
   background-position: 25% 100%;
   margin-left: 5%;
}
```

NOTE: If we provide only one percentage, then the other is assumed to be **50% (centered)**

Which property controls the image scroll in the background?

The **background-attachment:** property control the image scroll in the background. It gives the choice of whether the background image scrolls with the content or stays in a fixed position.

When an image is **fixed,** it stays in the same position relative to the viewport of the browser. The by-default value of background-attachment property is set to **Scroll** . In addition to that, it accepts **fixed|local.**

For ex.:

In the example below, I have used the **background-attachment** property with a value of **fixed** and it would make the image stay fixed as in the background of the viewport and the written content will scroll up over it.

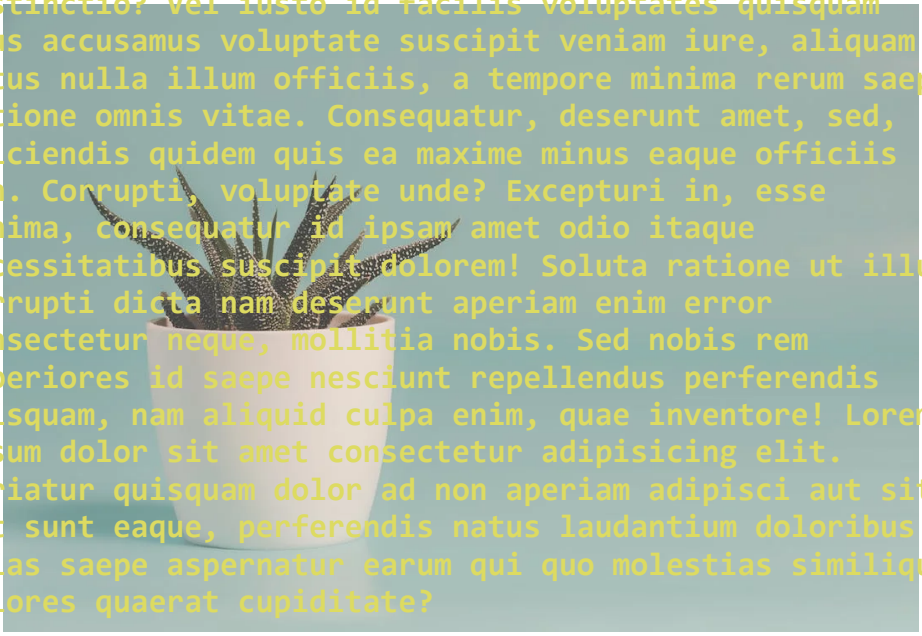
```
div{
    width: 70%;
    height: 500px;
    background-color: #fff;
    background-image: url(https://images.pexels.com/photos/1445416/pexels-photo-1445416.jpeg?
auto=compress&cs=tinysrgb&w=1260&h=750&dpr=1);
    color: rgb(221,219,97);
    font-weight: 900;
    font-family: monospace;
    font-size: x-large;
    background-size: 80%;
    background-repeat: no-repeat;
    background-position: center;
    background-attachment: fixed;
    margin-left: 5%;
}
```

Lorem ipsum dolor sit amet consectetur adipisicing elit. Voluptatem corporis amet, quam alias assumenda distinctio? Vel iusto id facilis voluptates quisquam eius accusamus voluptate suscipit veniam iure, aliquam natus nulla illum officiis, a tempore minima rerum saepe ratione omnis vitae. Consequatur, deserunt amet, sed, reiciendis quidem quis ea maxime minus eaque officiis non. Corrupti, voluptate unde? Excepturi in, esse minima, consequatur id ipsam amet odio itaque necessitatibus suscipit dolorem! Soluta ratione ut illum corrupti dicta nam deserunt aperiam enim error consectetur neque, mollitia nobis. Sed nobis rem asperiores id saepe nesciunt repellendus perferendis quisquam, nam aliquid culpa enim, quae inventore! Lorem ipsum dolor sit amet consectetur adipisicing elit. Pariatur quisquam dolor ad non aperiam adipisci aut sit est sunt eaque, perferendis natus laudantium doloribus alias saepe aspernatur earum qui quo molestias similique dolores quaerat cupiditate?

```
div{
    width: 70%;
    height: 500px;
    background-color: #fff;
    background-image: url(https://images.pexels.com/photos/1445416/pexels-photo-1445416.jpeg?
auto=compress&cs=tinysrgb&w=1260&h=750&dpr=1);
    color: rgb(221,219,97);
    font-weight: 900;
    font-family: monospace;
    font-size: x-large;
    background-size: 80%;
    background-repeat: no-repeat;
    background-position: center;
    background-attachment: scroll;
```

```
    margin-left: 5%;
}
```

Lorem ipsum dolor sit amet consectetur adipisicing elit. Voluptatem corporis amet, quam alias assumenda distinctio? Vel iusto id facilis voluptates quisquam eius accusamus voluptate suscipit veniam iure, aliquam natus nulla illum officiis, a tempore minima rerum saepe ratione omnis vitae. Consequatur, deserunt amet, sed, reiciendis quidem quis ea maxime minus eaque officiis non. Corrupti, voluptate unde? Excepturi in, esse minima, consequatur id ipsam amet odio itaque necessitatibus suscipit dolorem! Soluta ratione ut illum corrupti dicta nam deserunt aperiam enim error consectetur neque, mollitia nobis. Sed nobis rem asperiores id saepe nesciunt repellendus perferendis quisquam, nam aliquid culpa enim, quae inventore! Lorem ipsum dolor sit amet consectetur adipisicing elit. Pariatur quisquam dolor ad non aperiam adipisci aut sit est sunt eaque, perferendis natus laudantium doloribus alias saepe aspernatur earum qui quo molestias similique dolores quaerat cupiditate?



Why should background and color be used as separate properties?

Background and color are separate properties because they can be used for different purposes. The background property is used to set the background color or image of an element, while the color property is used to set the text color of that element. These properties can be used together to create a cohesive design, but they can also be used separately to create different effects. For example, you might want to set a background color for an element but leave the text color as the default.

The **color** property only specifies the foreground color which only applies to the text and border.

The **background** property is a Shorthand property to specify all of the background styles in one declaration.

For ex.:

```
div{
    width: 70%;
    height: 500px;
    background: white url(https://images.pexels.com/photos/1445416/pexels-photo-1445416.jpeg?auto=compress&cs=tinysrgb&w=1260&h=750&dpr=1) repeat-x center center fixed;
    ( background: background-color background-image background-repeat background-position background-attachment background-clip background-origin; ) Shorthand Property
    background-size: 20%;
    color: rgb(221, 219, 97);
}
```

Lorem ipsum dolor sit amet consectetur adipisicing elit. Aliquam hic exercitationem odit unde molestiae. Sapiente libero et excepturi. Neque illo odit ipsam. Sapiente enim eaque deleniti natus et laborum totam. Earum eligendi ipsa harum, mollitia praesentium quaerat laudantium consequuntur ea?Lorem ipsum dolor sit amet consectetur adipisicing elit. Fugiat qui a porro consequatur nobis laboriosam, earum blanditiis repellat nisi? Consectetur aperiam consequatur soluta autem corporis itaque nulla tempora doloribus, explicabo repellat repellendus rem a placeat culpa fugit magnam praesentium, iure neque incidunt? Quasi illo natus sunt corrupti officia soluta obcaecati deserunt asperiores atque. Ad vero quasi explicabo harum repellendus. Facere error eum rerum sed tempora, aliquam vero nesciunt rem id sunt nemo autem deserunt minus, similique, quas quo veritatis perspiciatis. Illo aliquid aspernatur voluptatibus suscipit, quae quos consequuntur. Ullam ipsa, tempora tempore illo dolor ratione porro voluptatum in nulla vel. Lorem ipsum, dolor sit amet

consectetur adipisicing elit. Voluptatem consequatur rem, voluptates voluptas tempora, id voluptatum illum corporis officiis nulla non earum hic. Vel est nostrum aspernatur illum nesciunt. Labore dolor veritatis nihil similique aut numquam maxime, ut recusandae a rerum iusto animi nisi repudiandae culpa corporis, harum inventore sunt illo amet provident hic repellendus excepturi dignissimos sequi? Molestiae neque officia nostrum quibusdam laborum explicabo soluta consectetur porro iure est voluptatum esse omnis sapiente culpa voluptate iste, dolores corrupti cupiditate consequatur tempora inventore perferendis illum! Consectetur, eveniet a libero blanditiis excepturi ab facilis consequuntur fugit at temporibus iure. Natus cum nihil nostrum aut quae impedit non saepe, neque laudantium omnis aliquam ab beatae fuga magni, cupiditate reiciendis minima, repellat nemo.

How to center block elements using CSS1?

We can center the block elements by setting the margin to **auto** on the left, right, top and bottom sides of a sized block element and it has the effect of centering the element in its container(body)

For ex.:

```
div{
    width: 25%;
    height: auto;
    background-color: white;
    marging: auto;
}
h1{
    color: gold;
    background: white;
    opacity: 1;
}
```

Playing with opacity

How to maintain the CSS specifications?

The CSS specifications are maintained by the World Wide Web Consortium (W3C). Even though every browser supports CSS, there are many inconsistencies in the supported specification version. Some browsers even have their own implementation of the specification and have proprietary (vendor) prefixes. Supporting all modern browsers is a daunting task, not to mention when developers need to support old and legacy browsers. All these problems cause a lot of trouble for developers, and it is hard for them to write CSS code that will render consistently across all browsers.

A CSS declaration is made up of a selector, or a group of selectors, and a declaration block. Inside of the declaration block, there are declarations with properties and values. The order of those declarations may seem unimportant at first glance, but defining an order does hold some advantages. Grouping is considered to be a good choice because it makes sense to write the positioning-related declarations first as some elements could be removed from the flow. The browser

digests it first, and the developer can read the block with ease knowing where the element will be positioned right away. Preferred order of styles is as below:

1. Positioning
2. Box model
3. Typographic
4. Visual

In addition to that, *Media Queries* is an enhancement of the @media rules of CSS and the "media" attribute in HTML. It adds parameters such as size of display, color depth and aspect ratio. This is because within a class of media there can still be important variations. for ex.: minimum or maximum screen size, resolution. This is also used in accordance with CSS in order to maintain its specifications.

What are the ways to integrate CSS as a web page?

There are mainly three ways to integrate CSS into a web page.

1. **Inline:**

HTML elements have CSS applied to them via **style** attribute.

For ex.:

```
<p style="color: cornflowerblue"> Hello Web Page </p>
```

Hello Web Page

2. **Embedded/Internal:**

By placing the CSS declarations in a **<style>** within the <head> element.

For ex.:

```
<style>
p{color: cornflowerblue;}
</style>
```

3. **External/Linked/Imported:**

Place the CSS in an external file and link it via a **<link>** attribute.

For ex.:

```
<link rel="stylesheet" href="style.css">
```

What is embedded style sheets?

An Embedded style sheet is a CSS style specification method used within HTML. We can embed the entire stylesheet in an HTML document by using the <style> element.

There are couple of advantages of Embedded style sheets as explained below:

- We can create classes for use on multiple tag types in the document.
- We can use the selector and grouping methods to apply styles in complex situations.
- No extra download is required to import the information.

What are the exxternal style sheets?

External style sheets are by far the most powerful way to use CSS because you can make style changes across an entire site simply by editing a single style sheet document. That is the advantage to having all the style information in one place, and not mixed in with the document source. Furthermore, because a single style document is downloaded and cached by the browser for the whole site, there is less code to download with every document, resulting in better performance.

An external style sheet is a plain-text document with at least one style sheet rule. It may not include any HTML tags. The style sheet should be named with the .css suffix.

There are two ways to apply an external style sheet: the link element and an @import rule.

What are the advantages and disadvantages of using external style sheets?

An external style sheet is a separate file that contains all of the style rules for a website, while an internal style sheet is embedded within the HTML file.

Advantages of using an external style sheet include:

- **Ease of maintenance:** With an external style sheet, you can make changes to the style rules in one file and have those changes apply to multiple pages on your website. This can save time and make it easier to keep your website consistent.
- **Faster page loading:** Because the style rules are stored in a separate file, the browser can cache that file and load it more quickly when a user visits multiple pages on your site.
- **Easier collaboration:** If multiple people are working on your website, an external style sheet can make it easier to collaborate and avoid conflicts.

Disadvantages of using an external style sheet include:

- **Additional HTTP request:** Using an external style sheet means that the browser has to make an additional HTTP request to retrieve that file, which can slow down page loading times slightly.
- **Dependency on the file:** If the external style sheet file is deleted or moved, your website's styling will be affected.
- **Less flexibility:** With an internal style sheet, you can target specific elements on a page using inline styles. This is not possible with an external style sheet.

What is the meaning of CSS Selector?

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 Selctor:** The element selector selects the HTML element by name.

   For ex.:

   ```
   p{
      text-align: center;
      color: cornflowerblue;
   }
   ```

2. **CSS ID Selector:** The id selector selects the id attribute of an HTML element to select a specific element. An id is always unique within the page so it is chosen to select a single, unique element. It is written with the hash character (#), followed by the id of the element.

   For ex.:

   ```
   #block{
      text-align: center;
      color: cornflowerblue;
   }
   ```

3. **CSS Class Selector:** The class selector selects HTML elements with a specific class attribute. It is used with a period character . (full stop symbol) followed by the class name.

   Note: A class name should not be started with a number.

   For ex.:

   ```
   .block{
      text-align: center;
      color: cornflowerblue;
   }
   ```

4. **CSS Universal Selector:** The universal selector is used as a wildcard character. It selects all the elements on the pages.

   For ex.:

   ```
   *{
      margin: 0;
      padding: 0;
      box-sizing: border-box;
   }
   ```

5. **CSS Group Selector:** The grouping selector is used to select all the elements with the same style definitions. Grouping selector is used to minimize the code. Commas are used to separate each selector in grouping.

For ex.:

```
h1,h2,p{
   text-align: center;
   color: cornflowerblue;
}
```

What are the media types allowed by CSS?

Media types, as included in the first part of a query, were introduced in CSS2 as a way to target styles to particular media. For example, this @media rule delivers a set of styles only when the document is printed

For ex.:

```
@media print {
   /* print-specific styles here */
}
```

The most current defined media types are **all, print, screen, and speech.** If you are designing for screen, the media type is optional.

What is the rule set?

A style sheet is made up of one or more style instructions (called style rules) that describe how an element or group of elements should be displayed. Each rule selects an element and declares how it should look. Basically, a ***rule set*** is made up of **Selector** and **Declaration-Block.**

The following example contains two rules. The first makes all the h1 elements in the document green; the second specifies that the paragraphs should be in a large, sans-serif font.

For ex:

```
h1{ color: green; }
p{ font-size: large; font-family: sans-serif; }
```