**CSS & CSS3 ASSIGNMENT**

**• What are the benefits of using CSS?**

**ANS:**

CSS is a programming language that has many benefits as we can create a web page by styling and formatting the website according to our needs. Here let us see what are the various advantages we can get by using CSS for designing web pages which are as follows:

* **Helps in Making Continuous Changes:** CSS helps in making changes that are repetitive in the page and easily we can make changes without going back to the coding again and again. By this the coding does not become confusing and consistent changes can be easily made.
* **Vast Variety of Designs:** CSS is a styling sheet programming language that offers many varieties of designs according to the output display which was not possible in HTML or other web designing programming.
* **Easy for users:** Working with CSS is easy for the users if they want to make changes according to them.
* **Device Friendly:** CSS is device friendly because it helps to design webpages in such a way that it becomes compatible to run on all the devices like computer or mobile.
* **Time saving**: CSS can easily style any webpage in very less time by making changes on a single page which can be used on the whole page.

**• What are the disadvantages of CSS?**

**ANS:**

Web pages can be styled very efficiently by using CSS but there are some drawbacks to CSS. Below are the disadvantages of CSS;

* **Browser Issue:** After designing a webpage it is not sure that the page is going to work similarly in every browser thus it can show different results in different browsers.
* **Can be Confusing:** The different levels of CSS like CSS1, CSS2, and CSS3 are confusing for many beginners but they can be understood easily after.
* **Security:** There will be always security concerns while working in CSS because it is an open program that can easily affect the display by anyone if they try to disturb it.
* **Fixing Problems For Beginners:** If there is an issue in the programming then for beginners it can be a little difficult as they have to make changes in the HTML coding also including the CSS part.

• **What is the difference between CSS2 and CSS3?**

**ANS:**

CSS3 is divided into many different documents called Modules. Every module adds new capability or extends features defined in CSS2 over preserving backward compatibility. Work on CSS3 started around the time of publication of the original CSS2 recommendation.

The CSS3 version supports many more browsers than CSS2.

CSS3 introduces many new selectors. Those new selectors are mostly in a form of pseudo-elements and pseudo classes.

New addition of General Sibling Combinator that may be used to match sibling elements of a given element through tilde (~) combinator.

CSS3 introduces many properties accompanied with new values and units. It facilitates styling of backgrounds, borders, boxes, etc..., which allows us to keep most of the styling within the WWW and HTML standards and our document, without a need for all those proprietary third party software packages.

New values and new units are introduced to support all those new properties. For instance Angle units deg, grad, rad and turn or Time units s and ms.

**• Name a few CSS style components:**

**ANS:**

1. **Typography:** Defines the font family, size, weight, line height, and color of text on a webpage.
2. **Box Model:** Comprises properties like padding, border, margin, and content, allowing for the layout and spacing of elements on the page.
3. **Flexbox:** A layout model that provides a more efficient way to layout, align, and distribute space among items in a container, even when their size is unknown or dynamic.
4. **Grid:** Another layout model that divides a page into rows and columns, enabling precise placement and alignment of elements.
5. **Backgrounds and Borders:** Properties for styling the background of elements, including colors, images, gradients, and borders.
6. **Transitions and Animations:** Allows for the creation of smooth transitions between different states of an element or even animations for more interactive and engaging user experiences.
7. **Responsive Design:** Techniques such as media queries and viewport units allow designs to adapt to different screen sizes and devices.
8. **Pseudo-classes and Pseudo-elements:** Enables styling based on specific conditions like the hover state of an element or inserting content before or after an element.
9. **Transforms and Transforms:** Properties like rotate, scale, skew, and translate that can alter the appearance and position of elements in 2D and 3D space.
10. **Selectors:** Various ways to select and style specific elements or groups of elements on a webpage, including element selectors, class selectors, ID selectors, attribute selectors, etc.

**• What do you understand by CSS opacity?**

**ANS:**

The CSS opacity property determines the transparency of an element on a scale of 0.0 to 1.0. The greater the opacity value, the more clear the element is. The element is totally transparent or clear when the value is 1.0 which is also the default value, at 0.5 the element displayed quite blur, and at 0.0 the element is totally opaque.

Syntax of opacity property in CSS

HTML element{

opacity: value;

}

Example: Applying opacity to the HTML element

In the given example, we have created the three headings using the <h1>, <h2>, and <h3> element. In heading 1 we have specified the opacity value to 0.0, in heading 2 we have specified the opacity value to 0.5, and in heading 3 we have specified the value to 1.0.

<head>

<title>CSS Opacity</title>

<style>

h1 {

background-color: red;

opacity: 0.0;

}

h2{

opacity: 0.5;

background-color: red;

}

h3{

opacity: 1.0;

background-color: red;

}

</style>

</head>

<body>

<h1>Heading 1</h1>

<h2>Heading 2</h2>

<h3>Heading 3</h3>

</body>

</html>

Output:

So after implementation what we can see is, the heading 1 with the value 0.0 is not visible at all. In heading 2 whose opacity value is 0.5, the content is quite blurred but visible while in heading 3 whose opacity value is 1.0 is clearly visible.

• How can the background color of an element be changed?

**ANS:**

The background color of an HTML element can be changed using CSS (Cascading Style Sheets). You can do this by targeting the element you want to change and then applying the background-color property to it. Here's a simple example:

html

Copy code

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Change Background Color</title>

<style>

/\* CSS code to change background color \*/

.myElement {

background-color: blue; /\* Change 'blue' to any color you want \*/

}

</style>

</head>

<body>

<div class="myElement">

This is a div with a blue background.

</div>

</body>

</html>

In this example, the background-color property is set to blue for the element with the class myElement. You can replace blue with any color value you like, such as color names (e.g., red, green, yellow) or hexadecimal color codes (e.g., #ff0000 for red, #00ff00 for green).

• How can image repetition of the backup be controlled?

**ANS:**

The background-repeat CSS property sets how background images are repeated. A background image can be repeated along the horizontal and vertical axes, or not repeated at all.

Syntax:

CSS

/\* Keyword values \*/

background-repeat: repeat-x;

background-repeat: repeat-y;

background-repeat: repeat;

background-repeat: space;

background-repeat: round;

background-repeat: no-repeat;

/\* Two-value syntax: horizontal | vertical \*/

background-repeat: repeat space;

background-repeat: repeat repeat;

background-repeat: round space;

background-repeat: no-repeat round;

/\* Global values \*/

background-repeat: inherit;

background-repeat: initial;

background-repeat: revert;

background-repeat: revert-layer;

background-repeat: unset;

Values

<repeat-style>

The one-value syntax is a shorthand for the full two-value syntax:

Single value Two-value equivalent

repeat-x repeat no-repeat

repeat-y no-repeat repeat

repeat repeat repeat

space space space

round round round

no-repeat no-repeat no-repeat

In the two-value syntax, the first value represents the horizontal repetition behavior and the second value represents the vertical behavior. Here is an explanation of how each option works for either direction:

repeat: The image is repeated as much as needed to cover the whole background image painting area. The last image will be clipped if it doesn't fit.

space: The image is repeated as much as possible without clipping. The first and last images are pinned to either side of the element, and whitespace is distributed evenly between the images. The background-position property is ignored unless only one image can be displayed without clipping. The only case where clipping happens using space is when there isn't enough room to display one image.

round: As the allowed space increases in size, the repeated images will stretch (leaving no gaps) until there is room (space left >= half of the image width) for another one to be added. When the next image is added, all of the current ones compress to allow room. Example: An image with an original width of 260px, repeated three times, might stretch until each repetition is 300px wide, and then another image will be added. They will then compress to 225px.

no-repeat: The image is not repeated (and hence the background image painting area will not necessarily be entirely covered). The position of the non-repeated background image is defined by the background-position CSS property.

• What is the use of the background-position property?

**ANS:**

The background-position Property

The background-position CSS properties are used to set for<position> each background images. and this property is specified as one or more position values separated by commas.

background-position properties play important role in CSS to decorate your web pages design. you can set background images according to your need on your web HTML pages with the help of CSS properties. the background images set in the viewpoint of web pages on which portion you want to show your background images.

background-position properties values:

background-position: center;

background-position: center;

background-position: left;

background-position: left;

background-position: top;

background-position: top;

background-position: 30% 70%;

background-position: 30% 70%;

background-position: right 25% bottom 60%;

background-position: right 25% bottom 60%;

background-position: bottom 40px right 200px;

background-position: bottom 40px right 200px;

set background-image to be center

body {

background-image: url('background-image.png');

background-attachment: fixed;

background-repeat: no-repeat;

background-position: center;

}

body {

background-image: url('background-image.png');

background-attachment: fixed;

background-repeat: no-repeat;

background-position: center;

}

set background-image to be left

body {

background-image: url('background-image.png');

background-attachment: fixed;

background-repeat: no-repeat;

background-position: left;

}

body {

background-image: url('background-image.png');

background-attachment: fixed;

background-repeat: no-repeat;

background-position: left;

}

set background-image to be top

body {

background-image: url('background-image.png');

background-attachment: fixed;

background-repeat: no-repeat;

background-position: top;

}

body {

background-image: url('background-image.png');

background-attachment: fixed;

background-repeat: no-repeat;

background-position: top;

}

• Which property controls the image scroll in the background?

**ANS:**

The background-attachment property is used to specify that the background image is fixed or scroll with the rest of the page in the browser window.

This property has three values scroll, fixed, and local. Its default value is scroll, which causes the element to not scroll with its content. The local value of this property causes the element to scroll with the content. If we set the value to fixed, the background image will not move during scrolling in the browser.

This CSS property can support multiple background images. We can specify a different value of the background-attachment property for each background-image, separated by commas. Every image will match with the corresponding value of this property.

Syntax:

background-attachment: scroll | fixed | local | initial | inherit;

Property Values:

scroll: It is the default value that prevents the element from scrolling with the contents, but scrolls with the page.

fixed: Using this value, the background image doesn't move with the element, even the element has a scrolling mechanism. It causes the image to be locked in one place, even the rest of the document scrolls.

local: Using this value, if the element has a scrolling mechanism, the background image scrolls with the content of the element.

initial: It sets the property to its default value.

inherit: It inherits the property from its parent element.Let's understand this property by using some illustrations.

• Why should background and color be used as separate properties?

**ANS:**

Readability and Accessibility: Using contrasting colors for text and background improves readability, especially for people with visual impairments. By keeping these properties separate, you can easily adjust the text color to ensure it's legible against the background color.

Design Flexibility: Separating background and text color properties allows for greater design flexibility. You might want to change the background color without affecting the text color, or vice versa. For example, you may have a light background with dark text for one section of your website and a dark background with light text for another section. Keeping these properties separate allows you to make these changes easily.

Maintenance and Scalability: Separating concerns makes your code easier to maintain and scale. If you ever need to update the background color or text color, you can do so without affecting the other property. This modularity makes your CSS code more organized and easier to manage as your project grows.

Semantic Markup: Separating the background and text color properties follows the principle of separating content (HTML) from presentation (CSS). It promotes semantic markup by allowing you to focus on the content structure in HTML and the visual styling in CSS.

• How to center block elements using CSS1?

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.

• How to maintain the CSS specifications?

ANS:

Modularize CSS: Break your CSS into smaller, reusable modules. This helps in better organization and makes it easier to update specific parts without affecting the entire codebase.

Use a Preprocessor: Utilize CSS preprocessors like Sass or Less to write more maintainable CSS with features like variables, mixins, and nesting. Preprocessors help in keeping your stylesheets organized and easier to maintain.

Follow a Naming Convention: Adopt a naming convention like BEM (Block Element Modifier) or OOCSS (Object-Oriented CSS) to keep your CSS classes consistent and predictable. This makes it easier to understand the purpose of each style rule and avoids conflicts.

Document Your Code: Write clear and concise comments to explain the purpose of different CSS rules and components. This documentation helps developers understand the codebase and makes future updates or modifications more manageable.

Version Control: Use a version control system like Git to track changes in your CSS files. This allows you to revert to previous versions if needed and collaborate with other developers more efficiently.

Perform Regular Audits: Periodically review your CSS codebase to identify unused styles, duplicate rules, or outdated conventions. Removing unnecessary code improves performance and makes the codebase easier to maintain.

Stay Updated: Keep abreast of the latest CSS specifications, browser updates, and best practices. New features and improvements are continually introduced, so staying updated ensures that your stylesheets remain compatible and efficient.

Testing Across Browsers: Test your CSS across different browsers and devices to ensure consistent rendering and functionality. Use tools like BrowserStack or CrossBrowserTesting to automate and streamline this process.

Refactor as Needed: As your project grows or requirements change, refactor your CSS code to maintain scalability and performance. Refactoring involves restructuring code without altering its external behavior, making it easier to maintain and extend in the long run.

Collaborate and Communicate: Encourage collaboration among team members and establish clear communication channels for discussing CSS-related issues and updates. This ensures that everyone is aligned and can contribute effectively to maintaining the CSS specifications.

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

**ANS:**

1. Inline CSS
2. Internal CSS
3. External CSS
4. Inline CSS

For inline styles, we use the style attribute of the HTML tags. CSS is passed as a string to the style attribute which adds the styles to the tags.

For example: If we want to make our web page background blue with inline CSS, we can write something like this:

<body style="background-color: blue"></body>

But it is a best practice to separate styles from the HTML tags that is the reason internal and external CSS comes into the picture.

1. Internal CSS

For Internal CSS, a style tag is used in which all the styles related to the webpage are added. This style tag is added in the head tag of the page so that the styles are added even before the HTML document is rendered.

For example: If we want to make our web page background as blue with internal css, we can right something like this:

<head>

<style>

body {

background-color: blue;

}

</style>

</head>

Internal CSS does its job but imagine a scenario where we have to use the same styles for more than one page. In this case, if we use internal CSS it will lead to writing the same code twice, to overcome this external CSS is being used.

1. External CSS

In external CSS we use a separate file with a .css extension where we write all our styles. This CSS file can be used by multiple webpages by using a link tag which is added under the head tag.

For example: If we want to make our web page background as blue with external CSS, we have to make changes in two files.

First, we will add all our styles to our CSS file, for our example, it will be styles.css

body{

background-color: blue;

}

Then we will link this css file to our html file as follows:

<head>

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

</head>

• What is embedded style sheets?

**ANS:**

Embedded style is the style attached to one specific document. The style information is specified as a content of the STYLE element inside the HEAD element and will apply to the entire document.

<HEAD>

<STYLE TYPE="text/css">

<!--

P {text-indent: 10pt}

-->

</STYLE>

</HEAD>

Note: The styling rules are written as a HTML comment, that is, between <!-- and --> to hide the content in browsers without CSS support which would otherwise be displayed.

• What are the external style sheets?

**ANS:**

External Style Sheet is a template/document/file containing style information which can be linked with any number of HTML documents. This is a very convenient way of formatting the entire site as well as restyling it by editing just one file. The file is linked with HTML documents via the LINK element inside the HEAD element. Files containing style information must have extension .

css, e.g. style.css. <HEAD> <LINK REL=STYLESHEET HREF="style.css" TYPE="text/css"> </HEAD>

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

**ANS:**

Advantages:

Separation of Concerns: External style sheets separate the presentation (CSS) from the structure (HTML) of a web page. This promotes cleaner, more maintainable code by clearly delineating the roles of HTML and CSS.

Code Reusability: You can link the same external style sheet to multiple HTML documents, ensuring consistent styling across your website. This saves time and effort and reduces the likelihood of inconsistencies.

Ease of Maintenance: Making changes to the styling of a website becomes easier with external style sheets. Instead of modifying CSS rules in multiple HTML files, you can edit a single external CSS file, which updates the styling across all linked HTML documents.

Improved Performance: External style sheets can be cached by the browser, meaning that once a user visits a page, the CSS file is stored locally. Subsequent visits to other pages using the same CSS file will result in faster load times since the browser doesn't need to download the CSS file again.

Facilitates Collaboration: External style sheets promote collaboration among developers by providing a centralized location for all styling rules. This makes it easier for multiple team members to work on different aspects of the design simultaneously.

Disadvantages:

Additional HTTP Request: Each external style sheet requires an additional HTTP request, which can marginally increase page load time, particularly if there are multiple external CSS files. However, this disadvantage is often outweighed by the benefits of code organization and caching.

Potential Dependency: If the external style sheet fails to load for any reason (e.g., network issues), it can affect the appearance of the web page. However, this risk can be mitigated by ensuring reliable hosting and implementing fallback styles or progressive enhancement techniques.

Complexity in Managing Many Files: In larger projects with numerous CSS files, managing external style sheets can become challenging. Ensuring consistency, avoiding redundancy, and organizing CSS files effectively require careful planning and maintenance.

• What is the meaning of the CSS selector?

**ANS:**

Class Selectors:

CSS class selectors let web developers target HTML elements using a class attribute denoted by a period (.), then the class name within a CSS stylesheet.

You associate HTML elements with a class selector by adding the “class” attribute to the element’s tag and specifying its name:

.new-class {

/\* Add CSS styling \*/

}

You can then apply the class above to your HTML:

<p class=" new-class">We styled this paragraph using the class we created. </p>

Class selectors give you fine-grained control over styling, which is important for making certain elements look different from those of the same type within the same webpage. You can use classes to style buttons differently:

.main-btn {

background-color: #0078d4;

color: white;

padding: 15px 25px;

}

.second-btn {

background-color: #eee;

color: #333;

padding: 10px 20px;

}

ID Selectors:

ID selectors are CSS mechanisms that let developers target single HTML elements with a unique identifier. They contain a hash symbol (#) followed by the ID name. You associate HTML elements with CSS ID selectors by adding the “id” attribute to the tag and specifying the ID name:

#single-element {

/\* CSS styles go here \*/

}

You can then apply the above ID selector to your HTML:

<div id="single-element">We styled this individual element with a unique ID.</div>

Attribute Selectors:

Developers use CSS attribute selectors to target HTML elements based on their attributes’ presence or specific value. They give you a way to apply styles that meet specific attribute criteria. You set up attributes using square brackets ([…]). They consist of an attribute name and an optional value or value pattern enclosed in quotation marks:

a[target="\_blank"] {

/\* CSS styles go here \*/

}

You would use the CSS attribute in your HTML to let you click a link and visit a new site:

<a href="https://example.com" target="\_blank">Visit Example</a>

Using the above type of CSS attribute selector ensures the style only applies to links with a target attribute set to “\_blank.”

Pseudo-Classes:

Pseudo-classes target elements based on their state or position within the document object model (DOM). You designate pseudo-classes using a colon ( : ) followed by the CSS pseudo-class’s name. A pseudo selector applies styling to elements in a particular state or context, like links being hovered over, checked checkboxes, or elements that are the first child of a parent.

When it comes to a CSS ID vs. class, the biggest difference is that IDs target single elements, while classes cover multiple elements.

a:hover {

/\* CSS styles go here \*/

}

You would use the above pseudo-class in HTML like this to apply a specific style when someone hovers over a link:

<a href="#">Hover over this link!</a>

Pseudo-classes are typically used in the following instances:

Styling form elements

Creating navigation menus

Targeting elements in a specific context

Combinators:

CSS ombinatory selectors define relationships or connections between HTML elements, letting developers target elements based on where they fall in the DOM. You can create complex selectors to pinpoint elements for styling. There are four main types of CSS selector combinators.

Whitespace (Descendant Combinator):

A whitespace character represents the descendant combinator. It selects all elements descended from a specific element, regardless of how deeply they’re nested in an HTML structure.

.parent\_element p {

/\* CSS styles go here \*/

}

The above example selects all <p> elements that descend from an element using the class “parent\_element.”

Child Combinator:

The child combinator is represented using the ( >) symbol. It selects elements that are direct children of a specific component. Below is an example of how to use the CSS operator.

.parent\_element > p {

/\* CSS styles go here \*/

}

The above selects only the <p> elements that descend from the children of an element with a class “parent\_element.”

Adjacent Sibling Combinator :

Represented by the ( + ) symbol, the adjacent sibling combinator selects elements immediately preceded by a specified element. You use the CSS + selector as shown below.

H3 + p {

/\* CSS styles go here \*/

}

The above example selects all <p> elements directly following <h3> elements. It won’t apply styling to <p> elements not preceded by an <h3>.

General Sibling Combinator:

The general sibling combinator uses the ( ~ ) symbol to select elements that are siblings of specific elements and share the same parent.

H3 ~ p {

/\* CSS styles go here \*/

}

The above selects all <p> elements with the same parent as an <h3> element.

Developers typically use CSS combinators to:

Apply fine-grained styling to elements with precision

Set up complex layouts and designs

Target interactive elements on web pages

Enhance accessibility by targeting specific elements within forms or tables

• What are the media types allowed by CSS?

**ANS:**

all: Applies to all devices.

css

Copy code

@media all {

/\* CSS rules \*/

}

print: Applies to printers and print preview.

css

Copy code

@media print {

/\* CSS rules \*/

}

screen: Applies to computer screens, tablets, smartphones, etc. This is the default media type if none is specified.

css

Copy code

@media screen {

/\* CSS rules \*/

}

speech: Applies to screen readers that "read out" the document.

css

Copy code

@media speech {

/\* CSS rules \*/

}

tty: Applies to teletypes and similar media devices.

css

Copy code

@media tty {

/\* CSS rules \*/

}

tv: Applies to television-type devices.

css

Copy code

@media tv {

/\* CSS rules \*/

}

You can also combine media types using the and keyword:

css

Copy code

@media screen and (min-width: 768px) {

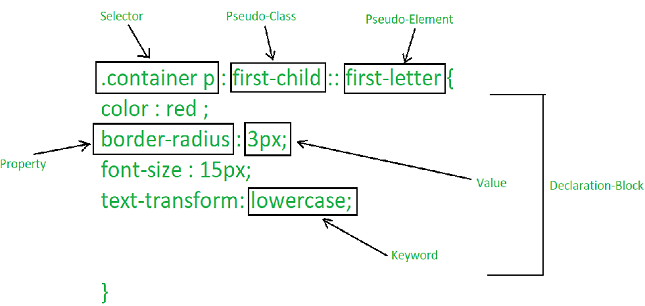
/\* CSS rules for screens with a minimum width of 768px \*/

}

• What is the rule set?

A CSS ruleset is various affirmations to various pieces or elements of the document. The objective is to apply a bunch of properties for certain distinct qualities to a solitary, or a particular arrangement of components in the connected HTML page.

**Visualization of CSS Ruleset :**



*CSS ruleset*

The “.” in the beginning indicates that the rule created will be a class, also “container” indicates the name of the[selector](https://www.geeksforgeeks.org/css-syntax-and-selectors/). Similarly “first-child” indicates the [pseudo-class](https://www.geeksforgeeks.org/css-pseudo-classes/), and elements inside curly brackets are elements of a declaration block, which contains some CSS properties and their corresponding values. The CSS ruleset will be applied when the selector name is called on the main HTML page.

CSS Ruleset is the main building block of the CSS stylesheet.