WEB TECHNOLOGY
Unit – II

Contents

CSS

Creating a Style Sheet

A Style Sheet is a file or section in which CSS rules are written. CSS can be added to a web page in three ways:

a) **Inline CSS**

- CSS is written directly inside an HTML element using the style attribute.

- Used for quick, single-element styling.

Example:

```
<p style="color: blue; font-size: 16px;">This is styled using inline CSS.</p>
```

b) **Internal CSS**

- CSS is written inside a <style> tag in the <head> section of the HTML document.

- Used when styling is specific to one web page.

Example:

```
<!DOCTYPE html>
<html>
<head>
    <style>
        body {
            background-color: lightgray;
        }
        p {
            color: green;
            font-size: 18px;
        }
    </style>
</head>
<body>
    <p>This is styled using internal CSS.</p>
</body>
</html>
```

c) **External CSS**

- CSS is written in a separate file with a .css extension and linked to the HTML file.

- Used when the same styling needs to be applied to multiple web pages.

Example:

1. **CSS File (styles.css)**:

```
body {
    background-color: white;
}
h1 {
    color: blue;
    text-align: center;
}
p {
    font-size: 16px;
    line-height: 1.5;
}
```

2. **HTML File**:

```
<!DOCTYPE html>
<html>
<head>
    <link rel="stylesheet" href="styles.css">
</head>
<body>
    <h1>External CSS Example</h1>
```

```
    <p>This paragraph is styled using an external style sheet.</p>

</body>

</html>
```

CSS Properties

CSS properties are used to define how HTML elements should look. Each property has a name and a value.

Common CSS Properties

Property	Description	Example
color	Sets the text color.	`color: red;`
background-color	Sets the background color.	`background-color: yellow;`
font-size	Changes the size of the text.	`font-size: 20px;`
font-family	Changes the font style.	`font-family: Arial, sans-serif;`
text-align	Aligns text (left, center, right).	`text-align: center;`
margin	Sets the space outside an element.	`margin: 10px;`
padding	Sets the space inside an element.	`padding: 15px;`
border	Adds a border around an element.	`border: 2px solid black;`

Property	Description	Example
width	Sets the width of an element.	`width: 50%;`
height	Sets the height of an element.	`height: 200px;`

Example: Combining CSS Properties

```
<!DOCTYPE html>
<html>
<head>
    <style>
        div {
            background-color: lightblue;
            width: 300px;
            height: 150px;
            border: 2px solid black;
            padding: 20px;
            margin: 10px auto;
            text-align: center;
        }
        p {
            color: darkblue;
            font-size: 18px;
            font-family: "Arial", sans-serif;
```

```
    }
  </style>
</head>
<body>
  <div>
    <p>This is a styled box.</p>
  </div>
</body>
</html>
```

Benefits of CSS

1. **Separation of Content and Style**: HTML handles content, while CSS handles design.

2. **Consistency**: With external CSS, the same styles can be applied across multiple pages.

3. **Ease of Maintenance**: Changes in the CSS file automatically update the entire website.

4. **Faster Loading**: Smaller files result in faster loading times.

Conclusion

Creating a Style Sheet: CSS can be added inline, internally, or externally.

CSS Properties: Control text, layout, colors, and borders to enhance web pages.

CSS makes websites visually appealing and user-friendly while keeping the code organized.

CSS Styling: Background, Text Format, and Controlling Fonts

CSS is used to style web pages, including controlling the background, text formatting, and font properties. Below are the details for these concepts:

Background Styling

The background of an HTML element can be styled using the following CSS properties:

a) **background-color**

- Sets the background color of an element.

- You can use color names, HEX codes, RGB values, or HSL values.

Example:

```
<div style="background-color: lightblue; padding: 20px;">
    This div has a light blue background.
</div>
```

b) **background-image**

Sets an image as the background of an element.

Example:

```
<div style="background-image: url('example.jpg'); background-size: cover; height: 200px;">
    This div has an image background.
</div>
```

c) **background-repeat**

- Controls whether the background image repeats.

- Options: repeat, no-repeat, repeat-x, repeat-y.

Example:

```
<div style="background-image: url('example.jpg'); background-repeat: no-repeat; height: 200px;">

    Background image does not repeat.

</div>
```

d) **background-size**

- Defines the size of the background image.

- Options: cover, contain, or specific dimensions like 100px 100px.

Example:

```
<div style="background-image: url('example.jpg'); background-size: cover; height: 200px;">

    The background image covers the entire element.

</div>
```

Text Formatting

CSS provides several properties to format text.

a) **color**

Sets the text color.

Example:

```
<p style="color: green;">This text is green.</p>
```

b) **text-align**

- Aligns text horizontally.

- Options: left, right, center, justify.

Example:

```
<p style="text-align: center;">This text is centered.</p>
```

c) **text-decoration**

- Adds decoration to text, such as underline, overline, or strikethrough.

- Options: none, underline, line-through.

Example:

```
<p style="text-decoration: underline;">This text is underlined.</p>
```

d) **line-height**

Sets the spacing between lines of text.

Example:

```
<p style="line-height: 1.5;">This paragraph has increased line spacing.</p>
```

e) **letter-spacing**

Adjusts the space between letters.

Example:

```
<p style="letter-spacing: 2px;">This text has extra letter spacing.</p>
```

Controlling Fonts

CSS offers properties to control the appearance of fonts.

a) **font-family**

- Defines the font style of text.

- Use common font names (e.g., Arial, Times New Roman) or include fallbacks.

Example:

```
<p style="font-family: Arial, sans-serif;">This text uses Arial font.</p>
```

b) **font-size**

- Sets the size of the font.

- Units: px, em, rem, %.

Example:

```
<p style="font-size: 20px;">This text has a font size of 20px.</p>
```

c) **font-weight**

- Sets the thickness of the text.

- Options: normal, bold, lighter, or numeric values like 100, 400, 700.

Example:

```
<p style="font-weight: bold;">This text is bold.</p>
```

d) **font-style**

- Defines if the text should be italic or normal.

- Options: normal, italic.

Example:

```
<p style="font-style: italic;">This text is italicized.</p>
```

e) **text-transform**

- Controls the capitalization of text.

- Options: uppercase, lowercase, capitalize.

Example:

```
<p style="text-transform: uppercase;">This text is in uppercase.</p>
```

Example: Combining Properties

Here's an example that combines background, text formatting, and font styling:

```
<!DOCTYPE html>

<html>

<head>

    <title>CSS Styling Example</title>

</head>

<body style="font-family: Arial, sans-serif; background-color: lightgray; text-align: center;">



    <div style="background-color: lightblue; padding: 20px; border: 1px solid black; margin: 20px auto; width: 50%;">

        <h1 style="color: darkblue; font-size: 24px; text-transform: uppercase;">CSS Styling Example</h1>

        <p style="color: black; font-size: 18px; line-height: 1.5; font-style: italic;">

            This div demonstrates CSS styling, including background, text formatting, and font control.

        </p>

    </div>



</body>

</html>
```

Summary

1. **Background Styling**: Use background-color, background-image, etc., to style the background of elements.

2. **Text Formatting**: Use properties like color, text-align, line-height to control text appearance.

3. **Font Control**: Use font-family, font-size, and font-weight to customize fonts.

CSS: Working with Block Elements, Lists, and Tables

CSS is widely used to style block elements, lists, and tables in a web page to make them visually appealing and well-structured. Let's break down these topics in detail:

Working with Block Elements

Block elements are HTML elements that take up the entire width of their parent container, starting on a new line. Examples include , <p>, <h1>, <section>, etc.

CSS Properties for Block Elements:

a) **width and height**

- Define the size of the block element.

- Units: px, %, em, etc.

Example:

```
<div style="width: 50%; height: 100px; background-color: lightblue;">
    This is a block element with width and height.
</div>
```

b) **margin**

- Adds space outside the block element.

Example:

```
<div style="margin: 20px; background-color: lightgreen; padding: 10px;">
    This block element has a margin of 20px.
</div>
```

c) **padding**

Adds space inside the block element, between the content and the border.

Example:

```
<div style="padding: 15px; background-color: lightcoral;">
    This block element has padding of 15px.
</div>
```

d) **border**

Adds a border around the block element.

Example:

```
<div style="border: 2px solid black; background-color: lightyellow; padding: 10px;">
    This block element has a 2px solid black border.
</div>
```

e) **display**

- Defines how the block element is displayed.

- block: Default for block elements.

inline-block: Makes the element behave like an inline element while still respecting block element properties.

Example:

```
<div style="display: inline-block; width: 100px; height: 100px; background-color: lightblue;">
    Inline-block element
</div>
```

Working with Lists

Lists are used to display items in an organized way. HTML supports two types of lists:

1. **Ordered Lists (<ol>)**: Items are numbered.

2. **Unordered Lists (<ul>):** Items have bullets.

CSS Properties for Lists:

a) **list-style-type**

- Defines the style of the bullet or numbering.

- Options: disc, circle, square, none, decimal.

Example:

```
<ul style="list-style-type: square;">
    <li>Item 1</li>
    <li>Item 2</li>
    <li>Item 3</li>
</ul>
```

b) **list-style-image**

Replaces bullets with a custom image.

Example:

```
<ul style="list-style-image: url('bullet.png');">
    <li>Custom bullet 1</li>
    <li>Custom bullet 2</li>
</ul>
```

c) **margin and padding**

Adjust spacing around and inside the list items.

Example:

```
<ul style="margin: 20px; padding: 10px;">
    <li>Item with margins and padding</li>
</ul>
```

d) **Nested Lists**

Lists can be nested and styled differently.

Example:

```
<ul style="list-style-type: circle;">
    <li>Outer Item
        <ul style="list-style-type: square;">
            <li>Inner Item</li>
        </ul>
    </li>
</ul>
```

Working with Tables

Tables are used to display data in rows and columns.

CSS Properties for Tables:

a) **border**

Adds a border to the table, rows, or cells.

Example:

```
<table style="border: 1px solid black; border-collapse: collapse;">
    <tr>
        <td style="border: 1px solid black;">Cell 1</td>
        <td style="border: 1px solid black;">Cell 2</td>
```

```
    </tr>
</table>
```

b) **padding and text-align**

padding: Adds space inside the table cells.

text-align: Aligns text inside the cells.

Example:

```
<table style="border: 1px solid black; border-collapse: collapse;">
    <tr>
        <td style="padding: 10px; text-align: center;">Centered Cell</td>
    </tr>
</table>
```

c) **background-color**

Adds color to the table, rows, or cells.

Example:

```
<table style="border: 1px solid black; border-collapse: collapse;">
    <tr style="background-color: lightgray;">
        <td>Row with background</td>
    </tr>
</table>
```

d) **width and height**

Define the size of the table or cells.

Example:

```
<table style="border: 1px solid black; width: 100%;">
    <tr>
        <td style="height: 50px;">Cell with height</td>
    </tr>
</table>
```

e) **border-spacing**

Defines the space between table cells (works when border-collapse is set to separate).

Example:

```
<table style="border-spacing: 10px;">
    <tr>
        <td>Spaced Cell 1</td>
        <td>Spaced Cell 2</td>
    </tr>
</table>
```

Example: Styling a Table

Here's an example that combines multiple CSS properties for a table:

```
<!DOCTYPE html>
<html>
<head>
    <title>CSS Tables and Lists</title>
</head>
<body>
```

```
    <h2>Styled Table</h2>
    <table style="width: 100%; border: 1px solid black; border-collapse: collapse;">
        <thead style="background-color: lightblue;">
            <tr>
                <th style="border: 1px solid black; padding: 10px;">Header 1</th>
                <th style="border: 1px solid black; padding: 10px;">Header 2</th>
            </tr>
        </thead>
        <tbody>
            <tr>
                <td style="border: 1px solid black; padding: 10px;">Data 1</td>
                <td style="border: 1px solid black; padding: 10px;">Data 2</td>
            </tr>
            <tr style="background-color: lightgray;">
                <td style="border: 1px solid black; padding: 10px;">Data 3</td>
                <td style="border: 1px solid black; padding: 10px;">Data 4</td>
            </tr>
        </tbody>
    </table>


</body>
</html>
```

Summary

Block Elements: Styled using width, height, margin, padding, border, etc.

Lists: Use list-style-type, list-style-image, and spacing properties for customization.

Tables: Use border, padding, text-align, background-color, and more to style rows, columns, and cells.

CSS: ID and Class

In CSS, ID and Class are used to select HTML elements and apply styles to them. Both help you target specific elements on your webpage, but they work in slightly different ways. Let's break them down:

CSS ID

The ID selector is used to apply styles to a single unique element on the page. Each ID must be unique within the HTML document, meaning no two elements can have the same ID.

Syntax:

- In HTML, you assign an ID to an element using the id attribute.

- In CSS, you reference the ID by using a # symbol followed by the ID name.

Example:

HTML:

```
<div id="header">
    <h1>Welcome to My Website</h1>
</div>
```

CSS:

```
#header {
    background-color: lightblue;
    color: white;
    text-align: center;
    padding: 20px;
```

```
}
```

Key Points about ID:

- An ID is used to target one specific element.

- You can only have one element with a given ID in the entire document.

- IDs are commonly used when you need to target a single element for JavaScript or when you need to apply a unique style.

CSS Class

The Class selector is used to apply styles to multiple elements that share the same class name. You can use the same class for many elements, making it more flexible compared to an ID.

Syntax:

- In HTML, you assign a class to an element using the class attribute.

- In CSS, you reference the class by using a . (dot) symbol followed by the class name.

Example:

HTML:

```
<p class="highlight">This is a highlighted paragraph.</p>

<p class="highlight">This is another highlighted paragraph.</p>
```

CSS:

```
.highlight {
    background-color: yellow;
    font-weight: bold;
}
```

Key Points about Class:

- A class can be used on multiple elements.

- Classes allow you to style groups of elements that need the same style.

- Classes are often used for general styles that apply to more than one element on a page.

Differences between ID and Class

Feature	ID	Class
Uniqueness	An ID must be unique within a page.	A class can be used on multiple elements.
Usage	Targets a single element.	Targets multiple elements with the same class.
Selector Syntax	#idName (e.g., #header)	.className (e.g., .highlight)
Specificity	IDs have higher specificity in CSS.	Classes have lower specificity than IDs.

Example of Using Both ID and Class Together

You can use both ID and Class on the same element to apply different styles. Here's an example:

HTML:

```
<div id="main-content" class="highlight">
    This div has both an ID and a class.
</div>
```

CSS:

```
#main-content {
    background-color: lightgray;
```

```
    padding: 15px;
}

.highlight {
    color: red;
}
```

In this example:

- The #main-content ID applies a light gray background and padding.

- The .highlight class makes the text inside the div red.

Conclusion

- ID is used for unique elements and should only appear once per page.

- Class is used for multiple elements and is more flexible since it can be applied to many elements.

- IDs are more specific than classes in CSS, meaning they will override styles from classes if both are applied to the same element.

Box Model in CSS

- The Box Model is a fundamental concept in CSS (Cascading Style Sheets) that determines how the elements of a webpage are displayed and spaced. Every HTML element is considered as a box, and the box model defines the size and space around each element.

- The box model consists of content, padding, border, and margin, and understanding these parts will help you control the layout and positioning of elements on a webpage.

Let’s break down each part of the box model:

Box Model Overview

Every HTML element is represented as a box with the following parts:

Content: The actual content of the element, like text, images, or other elements.

Padding: Space between the content and the border. Padding makes the content inside the box more spacious.

Border: A line surrounding the padding (if any) and content. Borders can be styled in various ways (solid, dashed, etc.).

Margin: Space outside the border. It separates the element from other elements around it.

The overall width of an element is calculated as:

Total Width = Content Width + Left Padding + Right Padding + Left Border + Right Border + Left Margin + Right Margin

Border Properties

The border defines the outermost edge of an element's box. It wraps around the padding and content areas.

Key Border Properties:

border-width: Defines the thickness of the border.

border-style: Specifies the style of the border (e.g., solid, dashed, dotted).

border-color: Sets the color of the border.

Example:

```
<div style="border-width: 2px; border-style: solid; border-color: blue;">
    This div has a blue border.
</div>
```

In this example:

- 2px is the thickness of the border.

- solid is the style of the border.

- blue is the color of the border.

You can also set the border for each side (top, right, bottom, and left) separately:

border-top: 2px solid red;

border-right: 2px dashed green;

border-bottom: 3px dotted blue;

border-left: 1px solid black;

Padding Properties

Padding is the space between the content and the border. It creates space inside the box around the content. The padding helps ensure that the content doesn't touch the border directly.

Key Padding Properties:

padding-top: Sets the padding at the top of the element.

padding-right: Sets the padding on the right side of the element.

padding-bottom: Sets the padding at the bottom of the element.

padding-left: Sets the padding on the left side of the element.

You can set padding for all four sides in one line:

padding: 20px; /* Applies 20px padding to all sides (top, right, bottom, left) */

Or, set different padding values for each side:

padding: 10px 20px 30px 40px;

/* Top padding: 10px, Right padding: 20px, Bottom padding: 30px, Left padding: 40px */

Example:

```
<div style="padding: 20px; border: 2px solid black;">
    This div has padding inside the box.
</div>
```

In this example:

20px padding is added on all four sides, creating space inside the element.

Margin Properties

Margin is the space outside the border. It creates space between an element and other surrounding elements, preventing them from being too close together.

Key Margin Properties:

margin-top: Sets the margin at the top of the element.

margin-right: Sets the margin on the right side of the element.

margin-bottom: Sets the margin at the bottom of the element.

margin-left: Sets the margin on the left side of the element.

You can set the margin for all four sides in one line:

margin: 20px; /* Applies 20px margin to all sides (top, right, bottom, left) */

Or, set different margin values for each side:

margin: 10px 20px 30px 40px;

/* Top margin: 10px, Right margin: 20px, Bottom margin: 30px, Left margin: 40px */

Example:

```
<div style="margin: 20px; padding: 10px; border: 1px solid black;">
    This div has a margin of 20px around it.
</div>
```

In this example:

20px margin is added on all sides, creating space between the element and other content around it.

Conclusion:

The CSS Box Model helps you control the layout and spacing of HTML elements by defining the size of the content box, the padding (inside space), the border (outer edge), and the margin (outer space). Understanding how to use the border, padding, and margin properties allows you to create well-structured and visually appealing web pages.

Border: Space around the element, between the content and the margin.

Padding: Space between the content and the border.

Margin: Space between the border and other surrounding elements.

Grouping

CSS Grouping is a way to apply the same styles to multiple HTML elements at once, which makes your CSS code shorter and easier to read.

Example:

```
<h1>Hello</h1>

<p>Welcome to my site</p>

<h2>About Us</h2>


<style>
   h1, h2, p {
       color: blue;
       font-family: Arial, sans-serif;
   }
</style>
```

In this example, the styles (color and font) are applied to h1, h2, and p tags in a single rule.

Dimension

CSS Dimension properties are used to set the width and height of elements.

Common Dimension Properties:

width: Sets the width of the element.

height: Sets the height of the element.

max-width and max-height: Restricts the maximum width/height of an element.

min-width and min-height: Sets the minimum width/height of an element.

Example:

```
<div style="width: 300px; height: 150px; background-color: lightblue;">
    This div is 300px wide and 150px tall.
</div>
```

Display

The display property controls how an element is shown on the page.

Types of Display:

block: Element takes the full width of its container (e.g.,).

inline: Element takes only the necessary width (e.g.,).

none: Hides the element completely.

Example:

```
<p style="display: none;">This text is hidden.</p>
<div style="display: inline;">Inline element</div>
```

Positioning

CSS Positioning controls how elements are placed on the page.

Types of Positioning:

static: Default positioning (elements appear in normal flow).

relative: Positions the element relative to its normal position.

absolute: Positions the element relative to its nearest positioned ancestor.

fixed: Positions the element relative to the browser window.

sticky: Sticks the element to a specific position while scrolling.

Example:

```
<div style="position: absolute; top: 50px; left: 100px;">
    This is an absolutely positioned box.
</div>
```

Floating

CSS Floating is used to place elements side by side or to float them to a specific side.

Common Float Values:

left: Floats the element to the left.

right: Floats the element to the right.

none: No floating.

Example:

```
<img src="image.jpg" style="float: left; margin-right: 10px;">
<p>This text flows around the image.</p>
```

Align

CSS Align properties are used to align text or elements within their containers.

Text Alignment:

text-align: Aligns text (e.g., left, center, right).

Vertical Alignment:

vertical-align: Aligns text or elements vertically (e.g., top, middle, bottom).

Example:

```
<p style="text-align: center;">This text is centered.</p>
```

Pseudo-Class

A Pseudo-Class is a keyword added to selectors to define a special state of an element.

Common Pseudo-Classes:

:hover: When the mouse is over the element.

:focus: When the element is focused (e.g., in a form).

:nth-child(n): Selects the nth child element.

Example:

```
<a href="#" style="color: blue;">Normal link</a>

<a href="#" style="color: blue; hover { color: red; }">Link turns red when hovered</a>
```

Navigation Bar

A Navigation Bar is a list of links styled to look like a menu.

Example:

```
<ul style="list-style: none; display: flex; background-color: #333; padding: 0;">

  <li style="margin: 10px;"><a href="#" style="color: white; text-decoration: none;">Home</a></li>

  <li style="margin: 10px;"><a href="#" style="color: white; text-decoration: none;">About</a></li>

  <li style="margin: 10px;"><a href="#" style="color: white; text-decoration: none;">Contact</a></li>

</ul>
```

Image Sprites

An Image Sprite is a single image that contains multiple smaller images. You use CSS to display a specific part of the sprite.

Why Use Image Sprites?

To reduce the number of HTTP requests, which improves performance.

Example:

```
<style>
    .icon {
        background-image: url('sprites.png');
        background-repeat: no-repeat;
    }
    .home {
        width: 50px;
        height: 50px;
        background-position: 0 0;
    }
    .about {
        width: 50px;
        height: 50px;
        background-position: -50px 0;
    }
</style>


<div class="icon home"></div>
<div class="icon about"></div>
```

Attribute Selector

An Attribute Selector allows you to select elements based on their attributes and attribute values.

Types of Attribute Selectors:

[attr]: Selects elements with the attribute.

[attr="value"]: Selects elements with the attribute equal to a specific value.

[attr^="value"]: Selects elements with the attribute starting with a specific value.

[attr$="value"]: Selects elements with the attribute ending with a specific value.

Example:

```
<input type="text" placeholder="Name">

<input type="password" placeholder="Password">


<style>
    input[type="text"] {
        border: 1px solid blue;
    }
    input[type="password"] {
        border: 1px solid green;
    }
</style>
```

CSS Color

CSS (Cascading Style Sheets) allows you to set colors for text, backgrounds, borders, and other elements. Colors can make your webpage attractive and visually organized.

Ways to Specify Colors in CSS:

1. **Named Colors**: Use predefined color names like red, blue, green, etc.

color: red;

2. **Hexadecimal Values**: Specify colors using a # followed by 6 digits (0-9, A-F).

Example: #FF0000 is red, #00FF00 is green, and #0000FF is blue.

background-color: #FF5733;

3. **RGB Values**: Use rgb() to define a color by its red, green, and blue components (values between 0 and 255).

color: rgb(255, 0, 0); /* Red */

4. **RGBA Values**: Like rgb() but includes an alpha value for transparency (0 for fully transparent, 1 for opaque).

background-color: rgba(255, 0, 0, 0.5); /* Semi-transparent red */

5. **HSL Values**: Use hsl() to define a color by hue, saturation, and lightness.

color: hsl(0, 100%, 50%); /* Red */

6. **HSLA Values**: Like hsl() but with an alpha value for transparency.

background-color: hsla(0, 100%, 50%, 0.5);

Examples of Using CSS Colors:

<h1 style="color: blue;">This is blue text</h1>

* This paragraph has a yellow background. *

* This box has a red border. *

Creating Page Layout

CSS helps in creating the structure and layout of a webpage. A page layout organizes the content visually and makes it user-friendly.

Key CSS Properties for Page Layout:

1. **Width and Height**: Define the size of elements.

width: 80%; /* Element takes 80% of the container width */

height: 200px;

2. **Display**: Determines how elements are displayed.

block, inline, flex, grid.

display: flex; /* Align items in rows or columns */

3. **Position**: Place elements on the page (static, relative, absolute, fixed, sticky).

position: absolute; top: 10px; left: 20px;

4. **Margin and Padding**: Add spacing inside and outside elements.

margin: 20px; /* Space outside the element */

padding: 10px; /* Space inside the element */

5. **Flexbox**: Align elements in a container.

display: flex;

justify-content: center; /* Center horizontally */

align-items: center; /* Center vertically */

6. **Grid**: Create complex layouts with rows and columns.

display: grid;

grid-template-columns: 1fr 2fr; /* Two columns */

Example: Two-Column Layout

```
<div style="display: flex;">
  <div style="width: 50%; background-color: lightblue;">Left Column</div>
  <div style="width: 50%; background-color: lightgreen;">Right Column</div>
</div>
```

Site Designs

CSS allows you to style your entire website and make it visually consistent. A good site design includes navigation, typography, spacing, and responsive design.

Key Techniques for Site Design:

1. **Responsive Design**: Use media queries to adjust layouts for different screen sizes.

```
@media (max-width: 768px) {
    div {
        width: 100%; /* Make the div full width on small screens */
    }
}
```

2. **Navigation Bar**: Create a horizontal or vertical menu for navigating the site.

```
<ul style="list-style: none; display: flex; background-color: #333; padding: 10px;">
    <li style="margin-right: 10px;"><a href="#" style="color: white; text-decoration: none;">Home</a></li>
    <li style="margin-right: 10px;"><a href="#" style="color: white; text-decoration: none;">About</a></li>
    <li><a href="#" style="color: white; text-decoration: none;">Contact</a></li>
</ul>
```

3. **Typography**: Use CSS to control the font style, size, and weight for better readability.

```
font-family: Arial, sans-serif;
font-size: 16px;
font-weight: bold;
```

4. **Backgrounds and Colors**: Add appealing backgrounds with colors, gradients, or images.

```
background: linear-gradient(to right, red, yellow);
```

5. **Consistent Styling**: Use classes and IDs to apply the same style across multiple pages.

```
.container {
    width: 80%;
    margin: 0 auto; /* Center the content */
}
```

Example of a Simple Page Design

```
<!DOCTYPE html>
<html lang="en">
<head>
    <title>Simple Page</title>
    <style>
        body {
            font-family: Arial, sans-serif;
            margin: 0;
            padding: 0;
        }
        header {
            background-color: #333;
            color: white;
            text-align: center;
```

```
        padding: 10px;
    }
    nav {
        background-color: #444;
        padding: 10px;
    }
    nav a {
        color: white;
        margin-right: 15px;
        text-decoration: none;
    }
    main {
        display: flex;
        margin: 20px;
    }
    .sidebar {
        width: 25%;
        background-color: #f0f0f0;
        padding: 10px;
    }
    .content {
        width: 75%;
        padding: 10px;
    }
    footer {
```

```
            background-color: #333;
            color: white;
            text-align: center;
            padding: 10px;
            position: fixed;
            bottom: 0;
            width: 100%;
        }
    </style>
</head>
<body>
    <header>
        <h1>Welcome to My Website</h1>
    </header>
    <nav>
        <a href="#">Home</a>
        <a href="#">About</a>
        <a href="#">Contact</a>
    </nav>
    <main>
        <div class="sidebar">Sidebar Content</div>
        <div class="content">Main Content</div>
    </main>
    <footer>
        &copy; 2024 My Website
```

```
    </footer>

</body>

</html>
```

Summary

CSS Colors: Add colors using names, hex codes, RGB, RGBA, HSL, and HSLA.

Page Layout: Use width, height, flexbox, grid, and positioning to arrange content.

Site Design: Combine navigation bars, typography, spacing, and responsive techniques for a visually appealing and functional website.