CSS for Beginners – 1

1.CSS introduction – Use Case / Implementation

2. Selectors

3. Colors

4. Fonts

5. Borders

6. Background

7. Margins

8. Float

Introduction to CSS

CSS (Cascading Style Sheets) is a language used to define the style and appearance of HTML content. It controls how elements on a webpage should look, including colors, fonts, layouts, and spacing. Without CSS, web pages would only display basic HTML content without any design or layout customization.

What CSS is Used For:

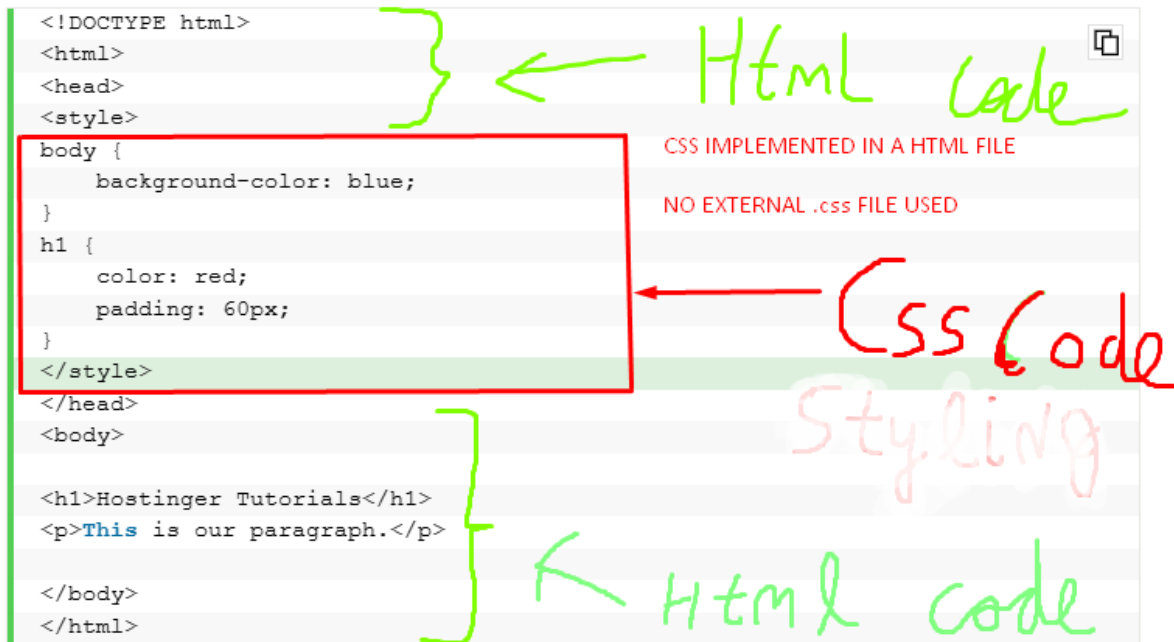- **Styling text**: Adjust fonts, sizes, and colors.
- **Layout management**: Define how elements are positioned, arranged, and sized on the page.
- **Visual enhancements**: Add borders, backgrounds, and shadows for a more engaging UI

Syntax : Selector {

property : value;

}

Ways to Implement CSS

There are three main ways to include CSS in an HTML document:

1. **Internal CSS**: CSS can be included in the <style> tag inside the <head> section of the HTML file.

```
<!DOCTYPE html>
<html>
<head>
<style>
body {
    background-color: blue;
}
h1 {
    color: red;
    padding: 60px;
}
</style>
</head>
<body>

<h1>Hostinger Tutorials</h1>
<p>This is our paragraph.</p>

</body>
</html>
```

Html Code

CSS IMPLEMENTED IN A HTML FILE

NO EXTERNAL .css FILE USED

Css Code
Styling

Html code

- **Use case**: Suitable for single HTML documents.
- **Limitations**: Affects only the current page, making it less flexible for multi-page sites.

2. **Inline CSS**: This method uses the style attribute directly within HTML elements to apply styles.

```
<!DOCTYPE html>
<html>
<body style="background-color:black;">

<h1 style="color:white;padding:30px;">Hostinger Tutorials</h1>
<p style="color:white;">Something usefull here.</p>

</body>
</html>
```

CSS added to <P> & <h1> tag

- **Use case**: Quick, one-off styles for individual elements.
- **Limitations**: Hard to maintain for large projects.

3. **External CSS**: The most common and efficient method. A separate .css file is linked to the HTML file.


```
.box{
    width: 100px;
    height: 100px;
    border: 1px solid;
    font-size: 40px;
    text-align: center;
    background: ■tomato;


}
```
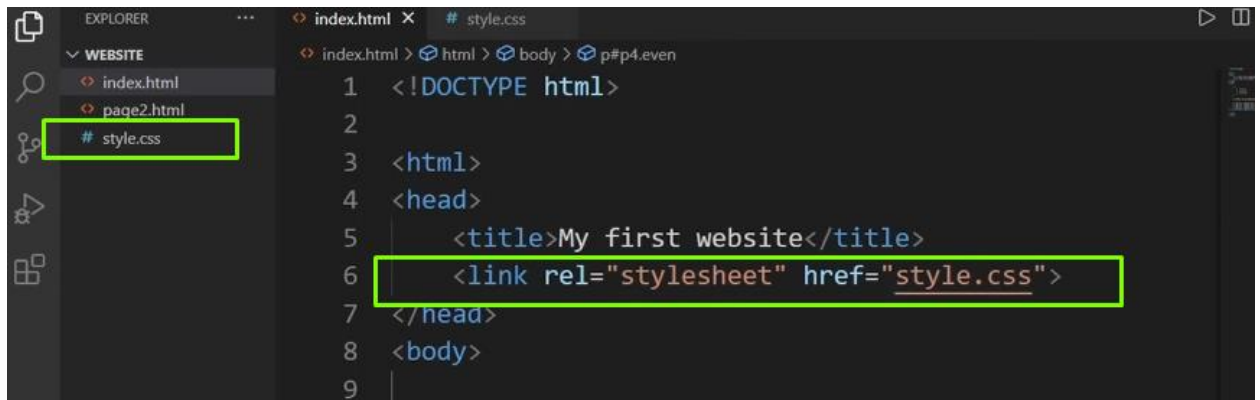
Seperate .CSS file

How to link a html file to a css file?

In the **<head>** section of your HTML sheet, add a reference to

your external **.css** file right after **<title>** tag:

- **Use case**: Great for styling large websites with multiple pages, as it allows styles to be reused across all pages.
- **Advantages**: Centralized styles, easier to maintain and update.

SELECTORS

CSS selectors are patterns used to select and style specific elements in an HTML document. They help apply styles efficiently based on element types, classes, IDs, attributes, and more.

Types of CSS Selectors C

1**. Universal Selector (*)** : Selects all elements in a document.

```css
* {
    color: red;
}
```
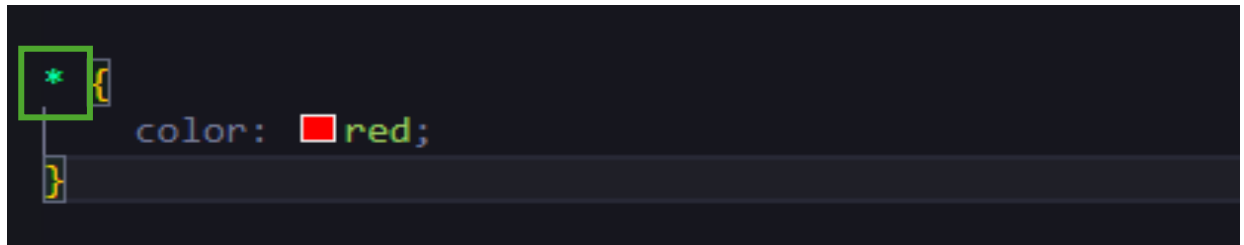
2. **Element (Type) Selector:** Selects all elements of a given type.

```css
p {
    color: rgb(14, 230, 111);
}
```
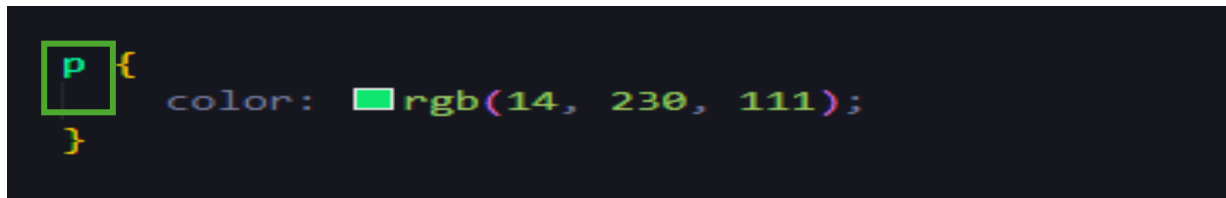
Applies styles to all <p> elements.

3. **Class Selector (.):** Selects all elements with a specific class.

```css
.para2{
    text-decoration: underline;
    font-size: 15px;
    font-weight: bolder;
    text-transform: lowercase;
}
```

Applies styles to elements having class="para".

4. ID Selector (#): Selects an element with a specific ID (must be unique).

```css
#para{
    text-decoration: line-through;
    font-size: 25px;
    text-transform: capitalize;
}
```

COLORS

Colors in CSS can be defined in different ways to style text, backgrounds, borders, and more. CSS provides multiple methods to specify colors, including names, HEX codes, RGB, and transparency variations.

- **Named Colors** : CSS has 140+ predefined color names like red, blue, green, aqua, etc.

```css
* {
    color: red;
}
```

- **HEX (Hexadecimal) Colors:** A six-digit code representing the intensity of Red, Green, and Blue (#RRGGBB). Each pair (00-FF) defines the intensity of the color.

```css
h6 {
    color: #fa09c3;
}
```

Examples: #000000 → Black

#ffffff → White

#ff0000 → Red

#00ff00 → Green

#0000ff → Blue

- **RGB (Red, Green, Blue):** Defines colors using values between 0-255.

```
p {
    color: rgb(14, 230, 111);
}
```
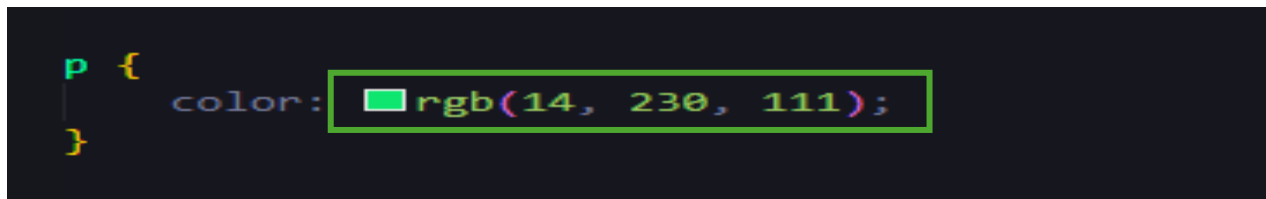
📌 Examples:

rgb(255, 0, 0) → Red

rgb(0, 255, 0) → Green

rgb(0, 0, 255) → Blue

rgb(255, 255, 255) → White

- **RGBA (RGB + Alpha Transparency):** Same as RGB but with an alpha (opacity) value (0-1).

```
#yasir {
    background: ☐rgba(33, 111, 121, 1);
}
```

📌 Alpha values:

0.0 → Fully transparent

0.5 → 50% transparent

1.0 → Fully opaque

TYPOGRAPHY

Text and Fonts are one of the most important aspects of design, affecting readability and visual appeal. CSS provides multiple ways to customize text & fonts, including text-alignment, text-decoration, text-transform, font-family, font-size, font-weight and more.

Note: *Text is the written characters, while font is the style used to change the appearance of those characters*

Here's a step-by-step guide to help you understand how to use fonts in CSS.

1. Specifying Font Families

You can define the font family using the font-family property. This property allows you to choose from various types of fonts.
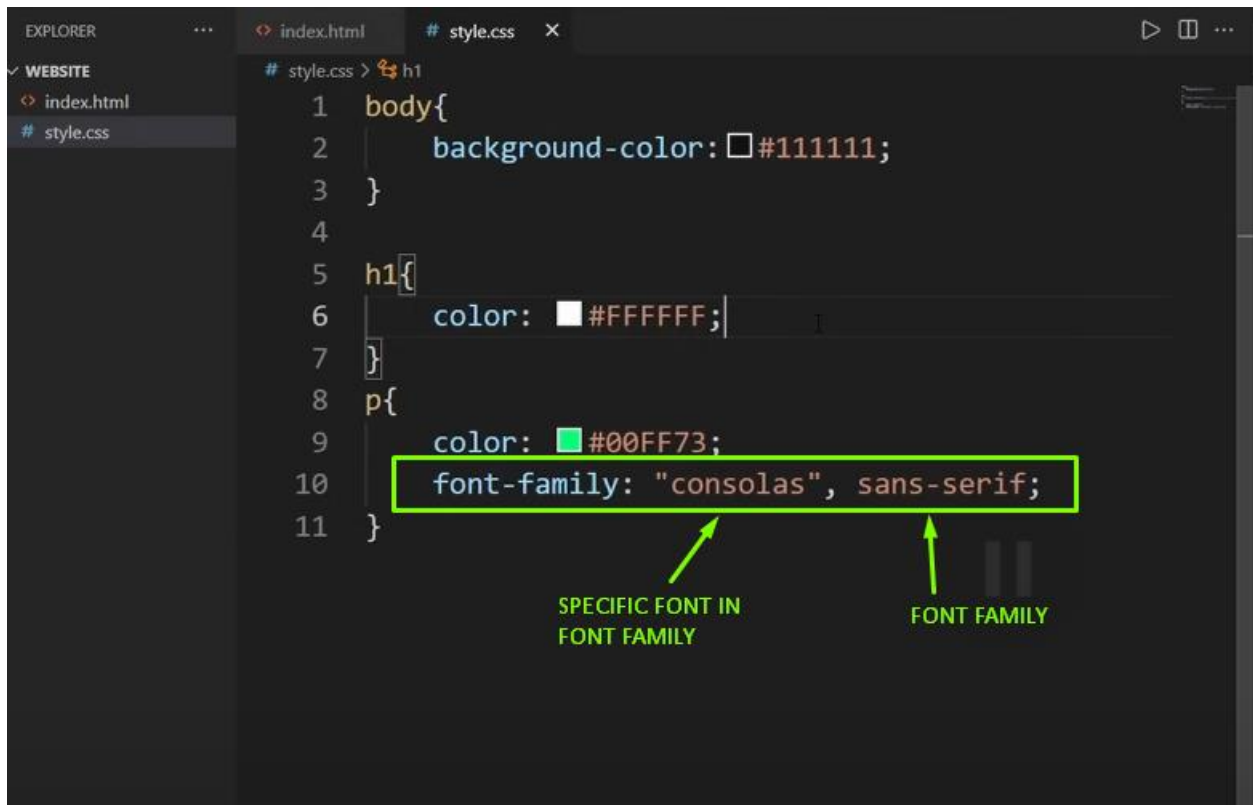
There are contigencies for font families.

INCASE the browser doesn't support the specified font , it will automatically move on to use the next best font available in the font family.

2. Using Web Fonts

To use web fonts, you can either import a font from a service like Google Fonts or host your own font files.
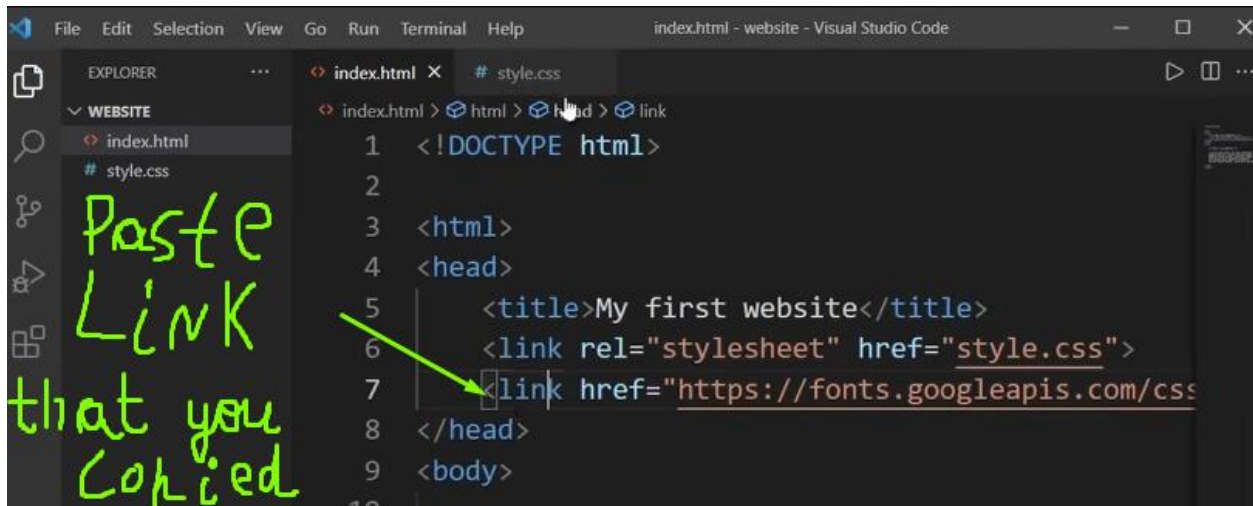
Google Fonts (importing):

1. Go to [Google Fonts](#).
2. Choose a font and click on the "+" icon.

3. In the selected fonts panel, copy the <link> tag
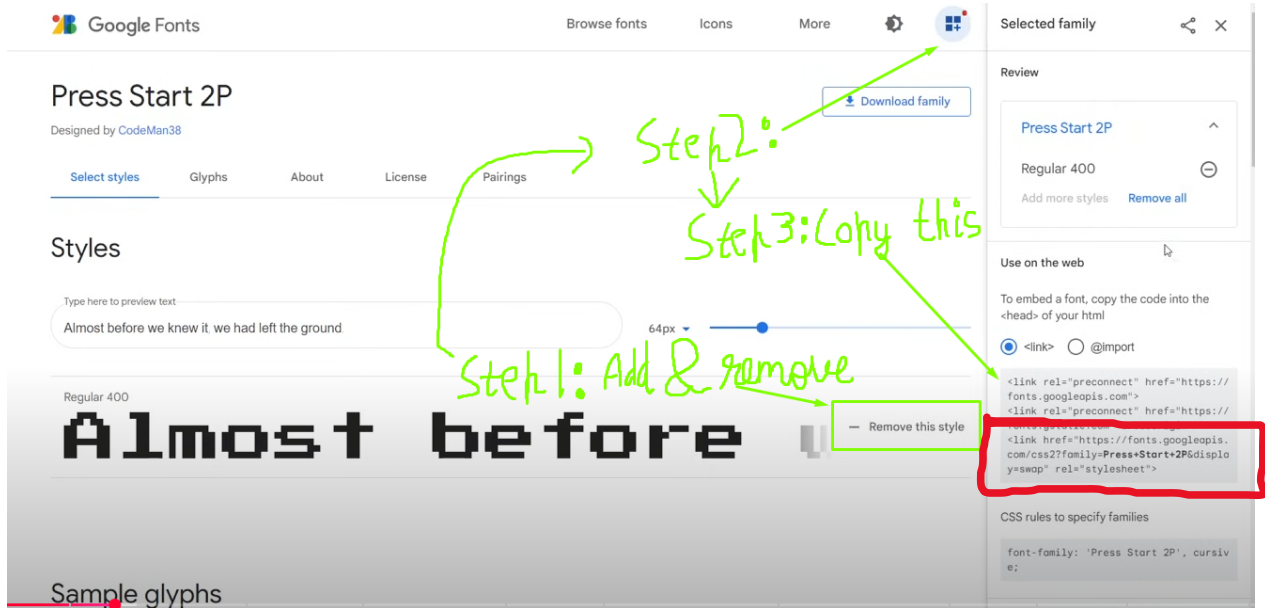
4. paste it inside the <head> section of your HTML file.

5. You can now use the font in your CSS file.

```css
body{
    background-color: #111111;
}

h1{
    color: #FFFFFF;
    font-family: "Press Start 2P", "consolas"
}
p{
    color: #00FF73;
    font-family: "consolas", sans-serif;
}
```
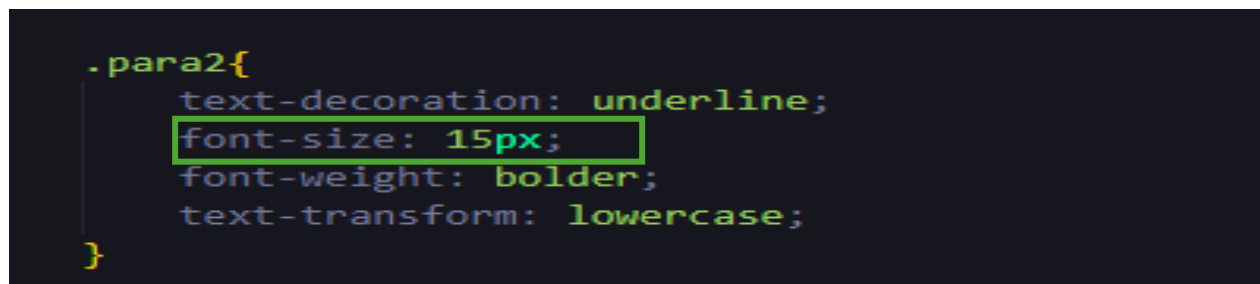
Main Font

Backup Font

INCASE THE BROWSER DOES NOT SUPPORT

6. You can change the font's styling too like its style , weight , size , decoration , uppercase/lowercase

```
File  Edit  Selection  View  Go  Run  Terminal  Help          style.css - website - Visual Studio Code          —    □

  EXPLORER        …      <> index.html       # style.css   ×                                              ▷ ◻
  ∨ WEBSITE              # style.css > ⛦ p
     <> index.html        1   body{
     # style.css          2       background-color:□#111111;
                          3   }
                          4
                          5   h1{
                          6       color: ■#FFFFFF;
                          7       font-family: "Press Start 2P", "consolas", s
                          8   }
                          9   p{
                         10       color: ■#00FF73;
                         11       font-family: "consolas", sans-serif;
                         12       font-style: italic;
                         13       font-weight: bold;
                         14       text-decoration: ■cyan dotted underline;
                         15       font-size: 18px;
                         16   }
```

You can Modify your font in different ways (handwritten note)

- **font-size - Text Size:** Controls the size of the text.


```
.para2{
    text-decoration: underline;
    font-size: 15px;
    font-weight: bolder;
    text-transform: lowercase;
}
```

- **font-weight - Text Boldness:** Defines how **bold** the text appears.

```
.para2{
    text-decoration: underline;
    font-size: 15px;
    font-weight: bolder;
    text-transform: lowercase;
}
```

Values:

- 100 (Thin)
- 400 (Normal)
- 700 (Bold)
- 900 (Extra Bold)

- **text-align - Text Alignment:** Aligns text horizontally.

```
h1 {
    color: red;
    background-color: green;
    text-align: center;
}
```

📌 Values:

left (default)

right

center

justify (spreads text evenly)

- **text-decoration** - Underline, Overline, Strikethrough

Adds decorative lines to text.

```css
.para2{
    text-decoration: underline;
    font-size: 15px;
    font-weight: bolder;
    text-transform: lowercase;
}
```

Values:

- none → No decoration
- underline → Adds an underline
- overline → Line above text
- line-through → Strikethrough text

- **text-transform -** Uppercase, Lowercase, Capitalize Changes text case.

```css
.para2{
    text-decoration: underline;
    font-size: 15px;
    font-weight: bolder;
    text-transform: lowercase;
}
```

BOX MODEL

The CSS Box Model is the fundamental concept that defines how elements are displayed on a webpage. It describes the rectangular boxes generated for HTML elements and how they interact with content, padding, border, and margin.

Each element in HTML is treated as a box, and understanding this model helps in proper layout design and spacing control.

margin 50

border 5

padding 50

30 5 20 794 × 160 20 5 30

50

5

50
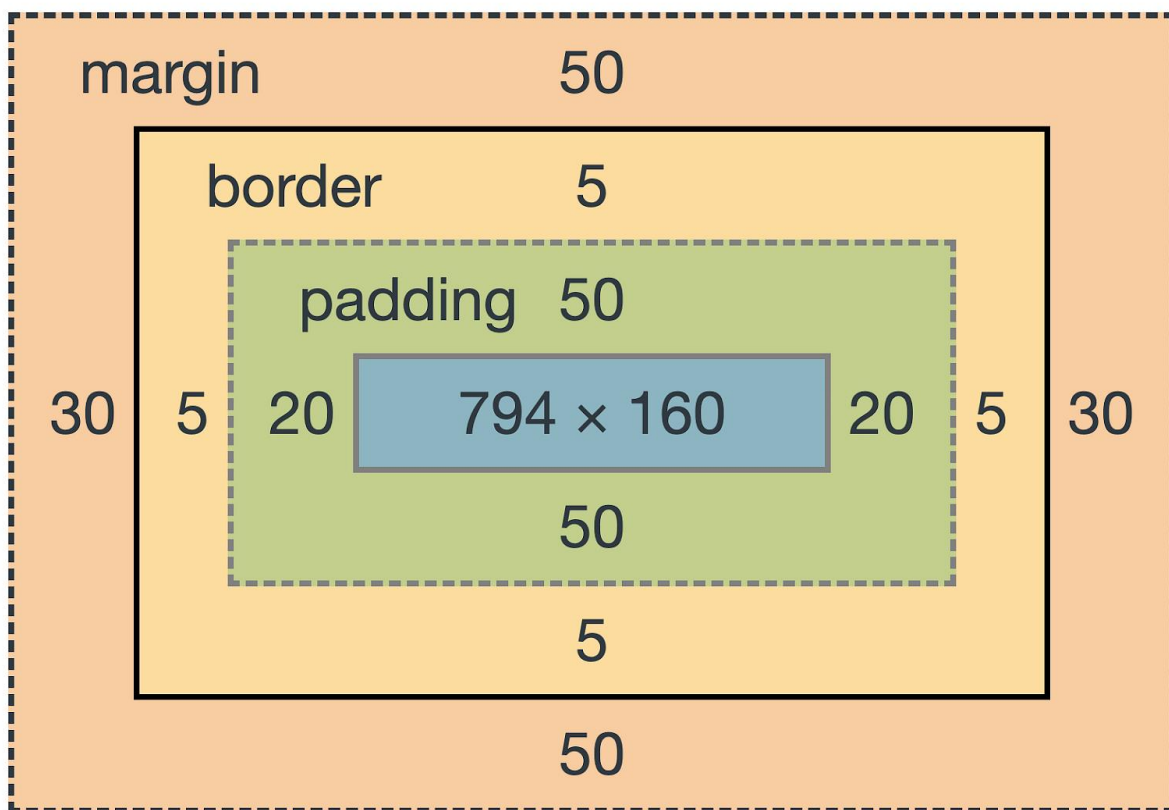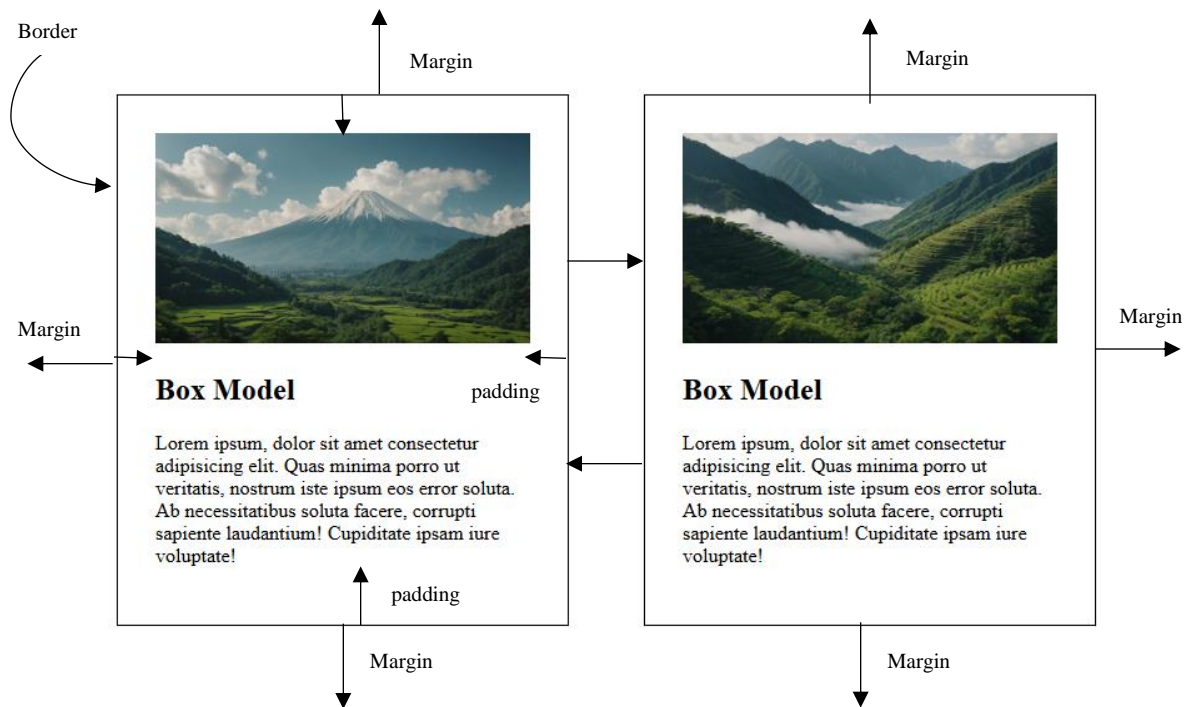
Every element in CSS follows the box model, which consists of four layers around the content:

- **Content** → The actual text, image, or any content inside the box.
- **Padding** → Space between the content and the border.
- **Border** → A visible (or invisible) line around the padding.
- **Margin** → Space between this element and other elements.

```css
.card{
    width: 300px;
    padding: 30px;
    border: 1px solid ☐black;
    margin: 30px;
}
```

Explanation of Each Box Model Component

1. Content (The Core): The content area holds **text, images, videos, or other HTML elements**. The size of this area depends on the width and height of the element.

📝 **Example:** If you have a paragraph <p> inside a div, the text inside is the **content** part of the box.

2. Padding (Space Inside the Border): Padding is the **space between the content and the border**. It creates an internal gap but does not affect other elements.
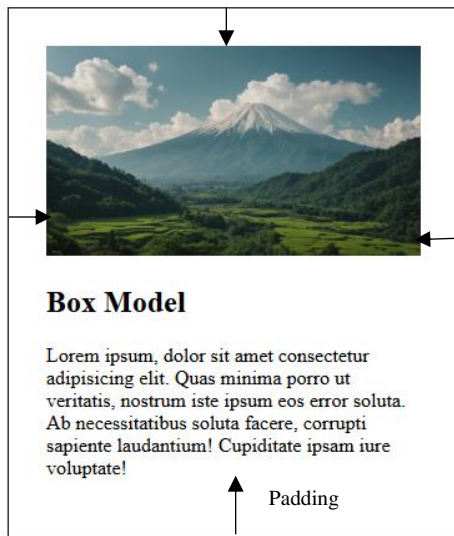
📌 Key Points:

- padding: 20px; → Same padding on all sides
- padding: 10px 20px; → 10px (top & bottom), 20px (left & right)
- padding: 10px 15px 20px 25px; → (top, right, bottom, left)

🔍 Example:

```css
.card{
    width: 300px;
    padding: 30px;
    border: 1px solid ☐black;
    margin: 30px;
}
```

Effect: The content inside the div will have a **30px gap from the border** on all sides.

3. **Border (The Edge of the Box):** The border is a **visible line surrounding the padding and content**. You can style it using border-width, border-style, and border-color.

📌 **Border Types:**

- solid → Continuous line
- dotted → Dots as border
- dashed → Dashes as border
- double → Double line

🔍 **Example:**

```
.card{
    width: 300px;
    padding: 30px;
    border: 1px solid ☐black;
    margin: 30px;
}
```

Effect: A red dashed border will surround the content.

4. **Margin (Space Outside the Border):** Margin is the **space between the element and surrounding**

elements. It creates gaps without affecting the element itself.

📌 **Margin Shorthand:**

✓ margin: 20px; → Same margin on all sides

✓ margin: 10px 20px; → 10px (top & bottom), 20px (left & right)

✓ margin: 10px 15px 20px 25px; → (top, right, bottom, left)

🔎 **Example:**

```
.card{
    width: 300px;
    padding: 30px;
    border: 1px solid ☐black;
    margin: 30px;
}
```

Effect: This div will have a **30px gap from other elements**.

◆ **Total Box Size Calculation**

The **actual size** of an element depends on all parts of the box model.

Formula for Total Box Size

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

Total Height = Content Height + Top & Bottom Padding + Top & Bottom Border + Top & Bottom Margin

🔎 **Example:**

```css
div {
  width: 200px;
  height: 100px;
  padding: 10px;
  border: 5px solid black;
  margin: 20px;
}
```

📝 **Actual Size Calculation:**

- Content: 200px (width)
- Padding: 10px + 10px = 20px
- Border: 5px + 5px = 10px
- Margin: 20px + 20px = 40px

📌 **Final width = 200 + 20 + 10 + 40 = 270px**
📌 **Final height = 100 + 20 + 10 + 40 = 170px**

4.BACKGROUNDS

CSS backgrounds are a crucial part of design, allowing you to visually enhance elements by setting colors, images, or even gradients behind content. Here's a breakdown of key concepts:

- **Background Color**: You can set a solid color to fill the background of an element. This can help establish a consistent look or highlight sections on the page.
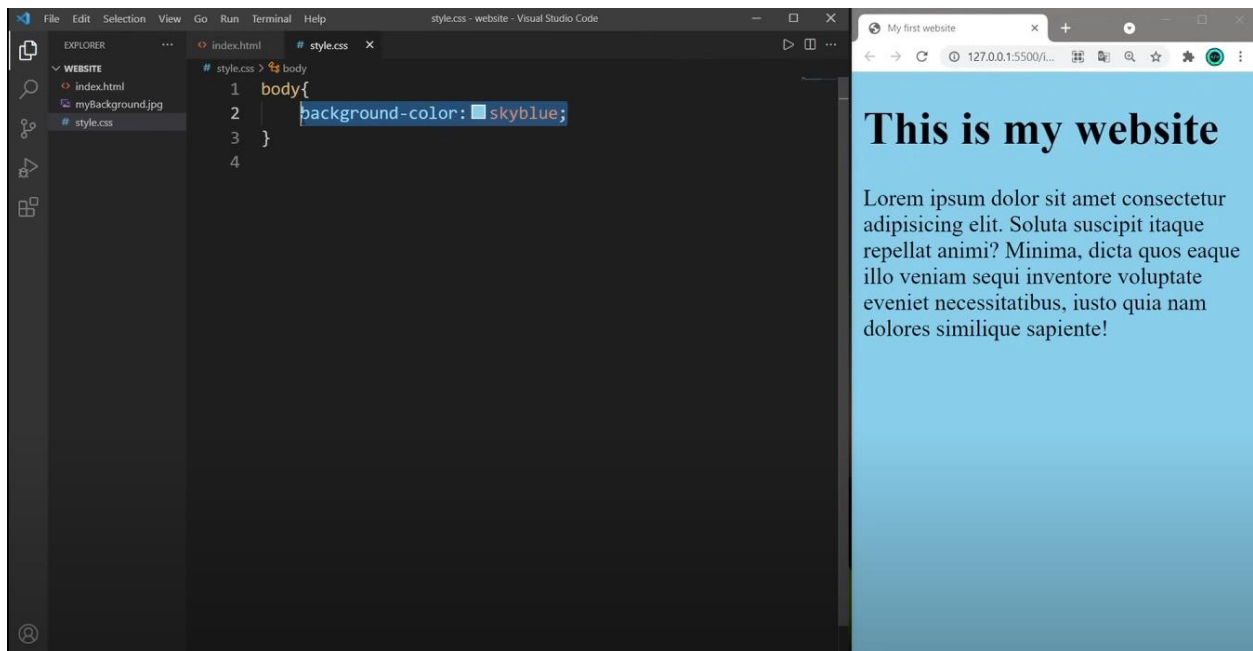
You can also apply a gradient background by specifying the colors that you want forming the gradient inside the paranthesis.

```
style.css > body
1  body{
2      background: linear-gradient(■skyblue, ■lightgreen
3  }
4
```

Color1, Color2

This is my website

Lorem ipsum dolor sit amet consectetur adipisicing elit. Soluta suscipit itaque repellat animi? Minima, dicta quos eaque illo veniam sequi inventore voluptate eveniet necessitatibus, iusto quia nam dolores similique sapiente!
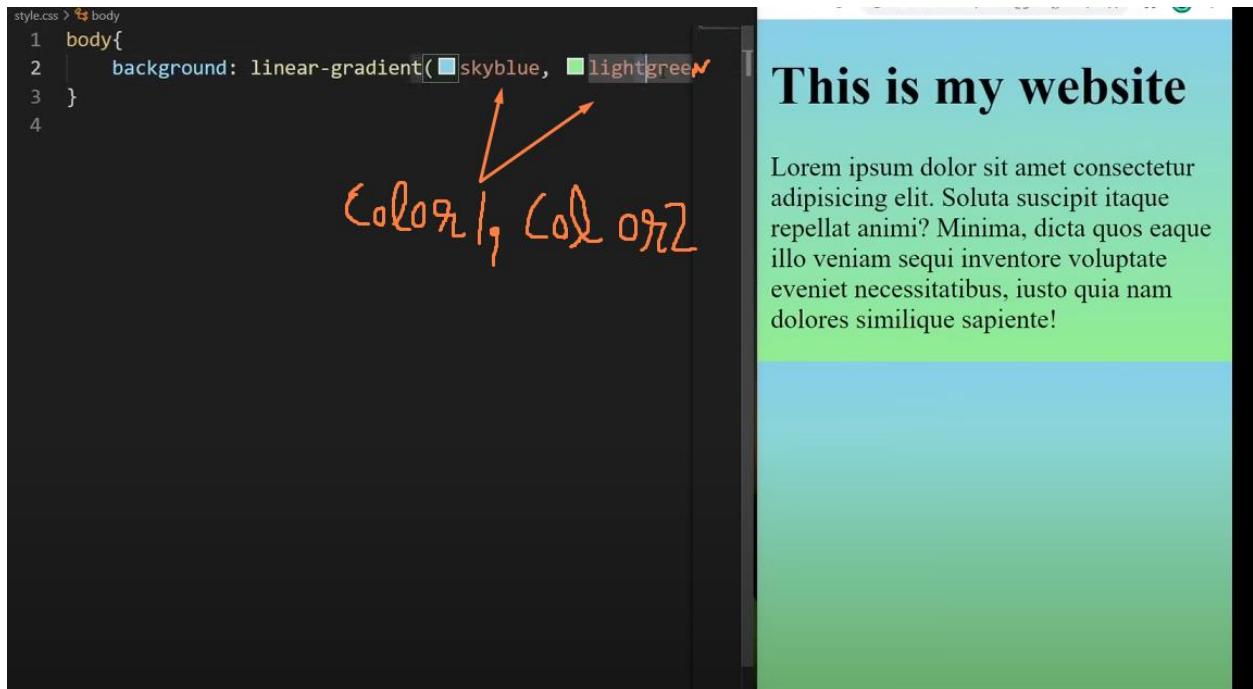
Background Repeat: By default, background images repeat both horizontally and vertically. You can prevent repetition, making the image appear only once, or you can control its repetition direction (horizontally or vertically).

Unfortunately our background too is repeating after the paragraph as you can see so to fix that issue we can do a simple fix and use :

background-repeat

and set it to   no-repeat

After this it won't repeat but there will be a white space below so to fix that make sure you add:

background-attachment: fixed;

So that there is no white space below the paragraph.
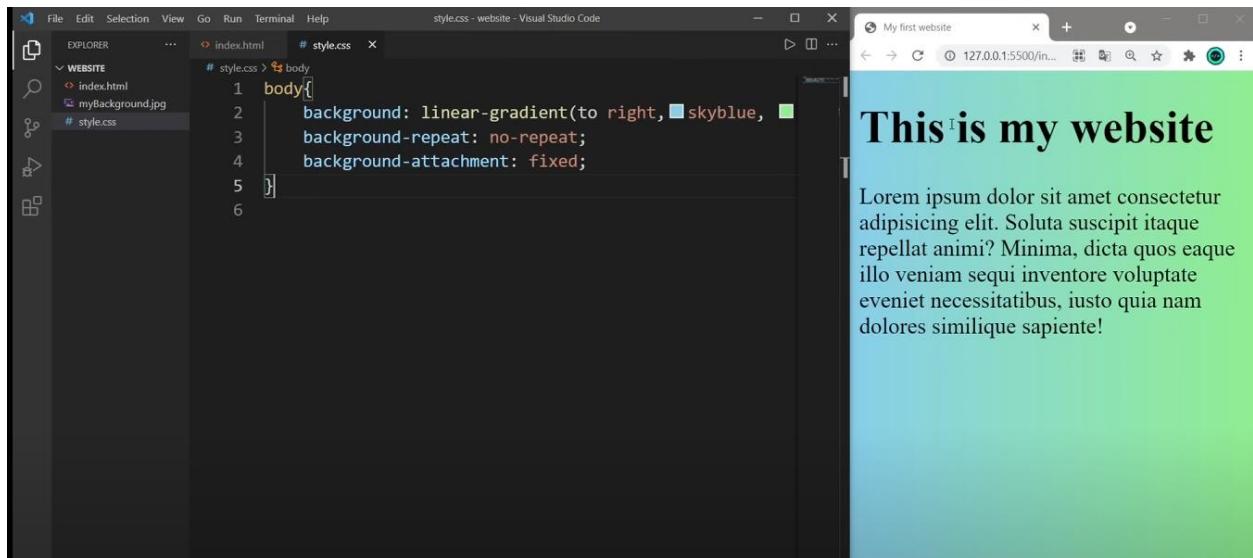
Background Positioning: This determines where an image is placed within the background. It can be controlled in terms of specific values or relative keywords (such as center, top, bottom). This is particularly useful when you want to create a certain focal point for the background image
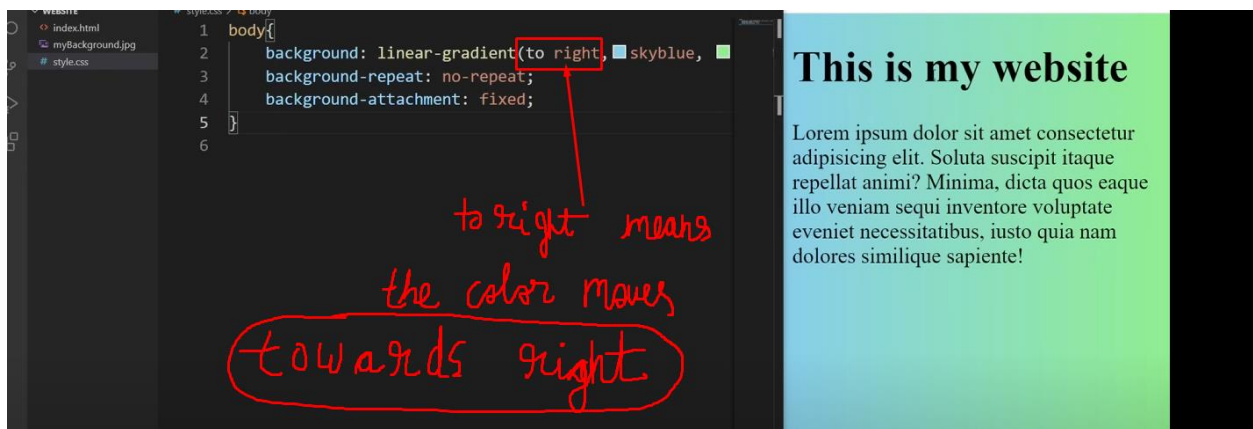
Background Attachment: This property controls whether the background moves when the user scrolls the page. A fixed background remains in place, creating a parallax effect, which is popular in modern web design.

```
1   body{
2
3   }
4   h1{
5       background: linear-gradient(to right, ■skyblue, ■
6       background-repeat: no-repeat;
7       background-attachment: fixed;
8   }
9   p{
10      background: linear-gradient(to right, ■skyblue, ■
11      background-repeat: no-repeat;
12      background-attachment: fixed;
13  }
14
```

BACKGROUND ONLY COVERS SPECIFIC ELEMENT ON SITE

This is my website

Lorem ipsum dolor sit amet consectetur adipisicing elit. Soluta suscipit itaque repellat animi? Minima, dicta quos eaque illo veniam sequi inventore voluptate eveniet necessitatibus, iusto quia nam dolores similique sapiente!

In the above code we have two separate backgrounds, one for [h1] and 1 for [p] which are completely independent from each other even though the properties might be the same.

Background Image: Backgrounds can include images, which can either be fixed or scrolling, depending on the design. Images can be applied to cover the full area of an element, creating visually striking effects.


```
1   body{
2       background-image: url("myBackground.jpg");
```

```
1  body{
2      background-image: url("myBackground.jpg");
3      background-repeat: no-repeat;
4      background-attachment: fixed;
5      background-position: center;         CENTERS THE IMAGE
6      background-size: cover;
7  }           RESIZES THE IMAGE ACCORDING
8  h1{            TO SIZE OF WINDOW
9      background: linear-gradient(to right, skyblue,
10     background-repeat: no-repeat;
11     background-attachment: fixed;
12 }
13 p{
14     background: linear-gradient(to left, skyblue, w
15     background-repeat: no-repeat;
16     background-attachment: fixed;
17 }
18
```

This is my website

Lorem ipsum dolor sit amet consectetur adipisicing elit. Soluta suscipit itaque repellat animi? Minima, dicta quos eaque illo veniam sequi inventore voluptate eveniet necessitatibus, iusto quia nam dolores similique sapiente!

5.BORDERS

Margins in CSS define the space outside an element, separating it from other elements. Unlike padding, which creates space inside an element between the content and its border, margins control the outer spacing.

Margins help structure layouts, ensuring elements don't overlap and maintain proper spacing.

Common misconception : Margins are easily confused with padding so always remember that **Margins are the space around an element** and **Padding is the space between the content(for example the text) and the border.**
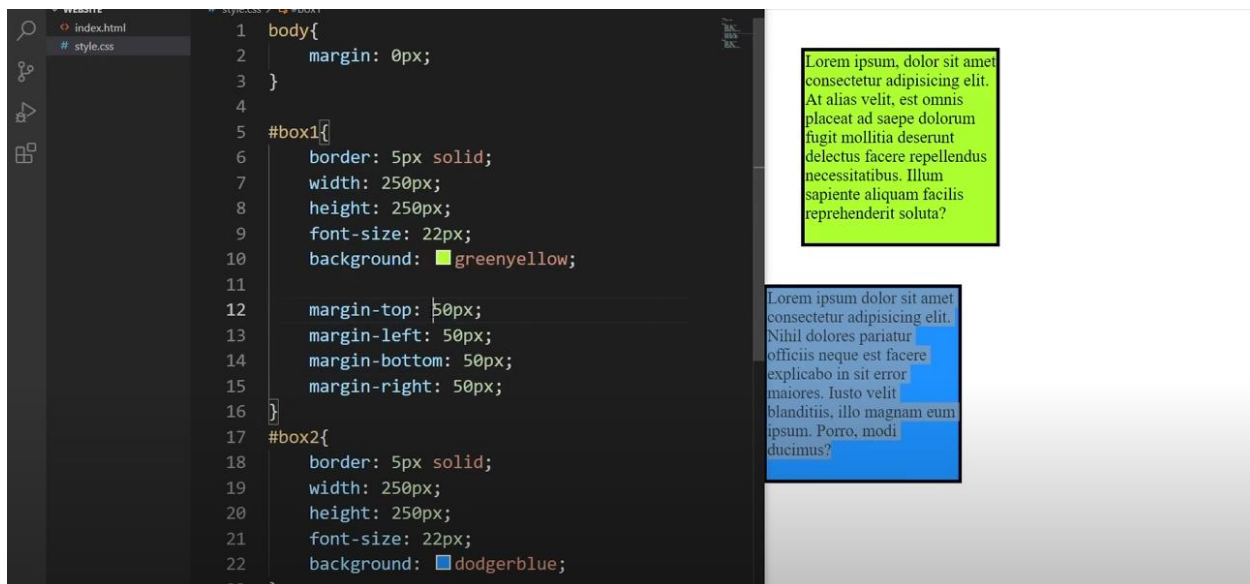
1. Margin Syntax and Different Forms

CSS provides different ways to define margins:

Uniform Margin

Sets the same margin on all four sides of an element.

- Example: margin: 20px; (applies 20px to top, right, bottom, and left)

Individual Margins

Allows setting different values for each side.

- margin-top: Controls the space above the element.
- margin-right: Controls the space to the right.
- margin-bottom: Controls the space below.
- margin-left: Controls the space to the left.

Each side can have independent values, allowing precise control over spacing.

Two-Value Margin

- When two values are given, the first applies to **top & bottom**, and the second applies to **left & right**.
- Example: margin: 10px 20px; (10px for top/bottom, 20px for left/right)

Three-Value Margin

- When three values are provided, they apply to **top, left/right, bottom**.
- Example: margin: 10px 20px 30px; (10px top, 20px left/right, 30px bottom)

Four-Value Margin

- A four-value margin allows setting all sides explicitly in the order: **top, right, bottom, left**.
- Example: margin: 10px 20px 30px 40px; (10px top, 20px right, 30px bottom, 40px left)

Auto Margins

Using margin: auto; can automatically center block-level elements horizontally if a width is set.

```
5   #box1{
6       border: 5px solid;
7       width: 250px;
8       height: 250px;
9       font-size: 22px;
10      background: ■greenyellow;
11      /*
12      margin-top: 50px;
13      margin-left: 50px;
14      margin-bottom: 50px;
15      margin-right: 50px;
16
17      margin: 50px;
18      */
19      margin-right: auto;
20      margin-left: auto;
21  }
22  #box2{
23      border: 5px solid;
24      width: 250px;
25      height: 250px;
26      font-size: 22px;
27      background: ■dodgerblue;
28  }
```
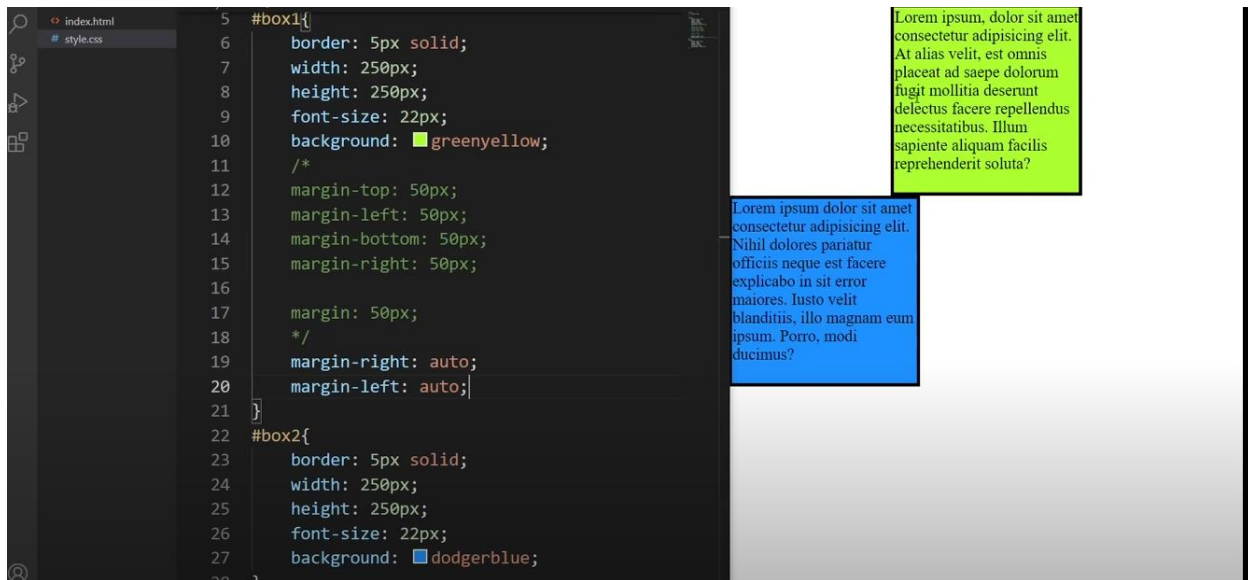
Percentage-Based Margins

Margins can also be defined as a percentage relative to the width of the
parent element, making them responsive.


```
5   #box1{
6       border: 5px solid;
7       width: 250px;
8       height: 250px;
9       font-size: 22px;
10      background: ■greenyellow;
11      /*
12      margin-top: 50px;
13      margin-left: 50px;
14      margin-bottom: 50px;
15      margin-right: 50px;
16
17      margin: 50px;
18      */
19      margin-left: 50%;
20  }
21  #box2{
22      border: 5px solid;
23      width: 250px;
24      height: 250px;
25      font-size: 22px;
26      background: ■dodgerblue;
27  }
```
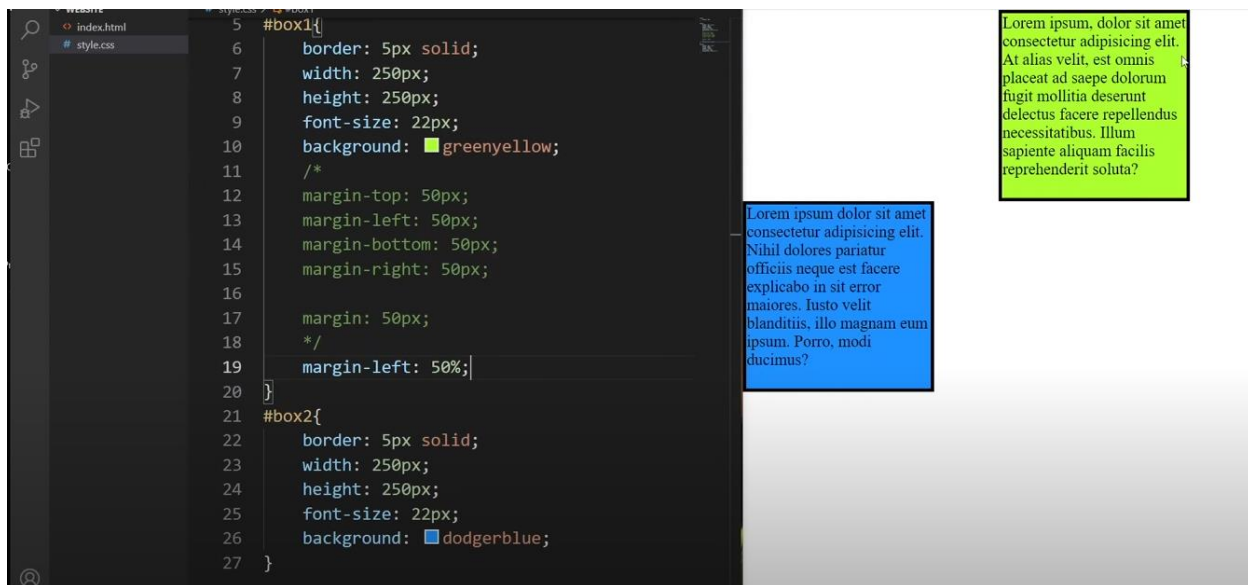
2. Special Cases and Behavior

- **Collapsing Margins**: When two vertical margins meet, the larger one
  takes effect instead of adding together.

- **Negative Margins**: Margins can take negative values, causing elements to move closer to each other.

6.FLOAT

The float property in CSS is used to position elements by making them "float" to the left or right of their container. It was originally designed for text wrapping around images but has been widely used for creating layouts before modern flexbox and grid systems.

1. Float Syntax and Values

The float property accepts the following values:

- **left**: The element is pushed to the left, allowing other content to flow beside it on the right.
- **right**: The element is pushed to the right, and other content flows on the left.
- **none**: Default value, meaning the element does not float.
- **inherit**: The element inherits the float value from its parent.

```
1   .box{
2       width: 100px;
3       height: 100px;
4       border: 1px solid;
5       font-size: 40px;
6       text-align: center;
7       background: ■tomato;
8
9       float: left;
10  }
```

Box1 Box2 Box3 Lorem ipsum dolor sit amet, consectetur adipisicing elit. Quae maiores beatae earum consequatur nostrum! Harum aliquid magnam, minima dolor, rerum mollitia fugiat labore temporibus consequatur eveniet aut sit reprehenderit in.

Lorem ipsum dolor sit amet, consectetur adipisicing elit. Quae maiores beatae earum consequatur nostrum! Harum aliquid magnam, minima dolor, rerum mollitia fugiat labore temporibus consequatur eveniet aut sit reprehenderit in.

Lorem ipsum dolor sit amet, consectetur adipisicing elit. Quae maiores beatae earum consequatur nostrum! Harum aliquid magnam, minima dolor, rerum mollitia fugiat labore temporibus consequatur eveniet aut sit reprehenderit in.

2. How Float Affects Layout

When an element is floated, it is taken out of the normal document flow, meaning surrounding elements behave as if the floated element isn't there. This can create issues with parent containers collapsing when they no longer recognize the floated element inside them.

3. Clearing Floats (Clearing the Flow Issue)

Since floating elements are removed from the normal document flow, their parent containers might not expand to wrap around them properly. The clear property is used to fix this issue.

- **clear: left;** - Prevents elements from appearing next to a floated element on the left.
- **clear: right;** - Prevents elements from appearing next to a floated element on the right.
- **clear: both;** - Prevents elements from appearing next to floated elements on either side.
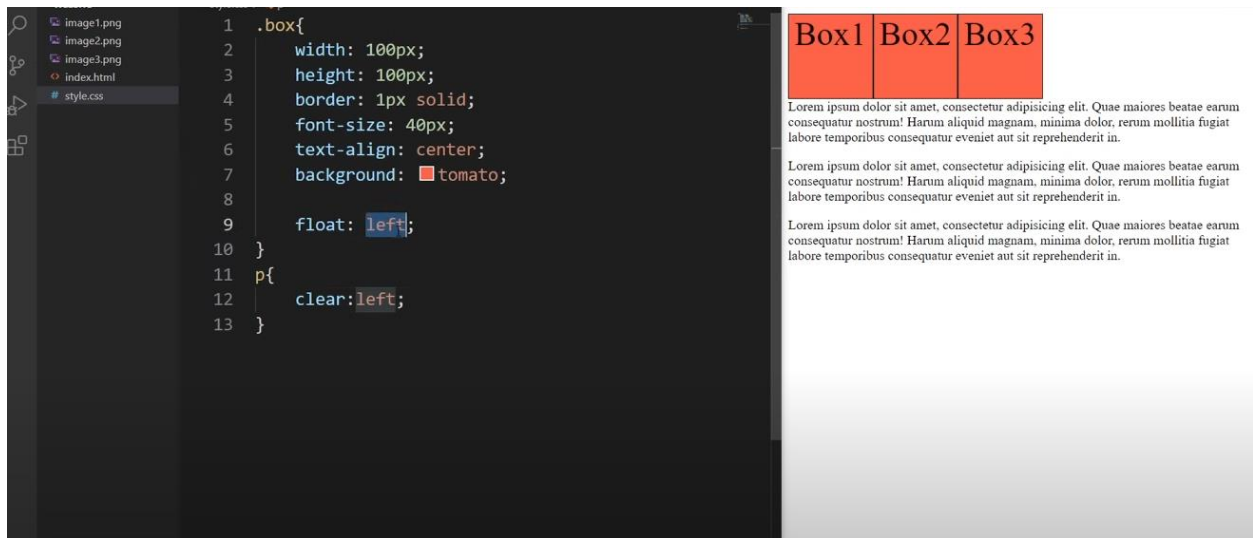
A common technique to clear floats is using the clearfix method, where an empty element or pseudo-element is added after the floated elements to clear them.

4. Common Uses of Float

- Wrapping text around images.
- Creating multi-column layouts (before flexbox/grid).
- Structuring navigation menus.

However, modern CSS techniques like flexbox and grid have largely replaced float-based layouts. Float is still useful for specific cases, like floating images within text.