



CSS

(Cascading Style Sheets)

1. **Style:** Sets the look and feel.
2. **Colors & Fonts:** Customizes text and background.
3. **Layout:** Controls position and size.
4. **Selectors:** Targets specific **HTML** elements.





Importance of CSS



Premium Website



Premium Brand

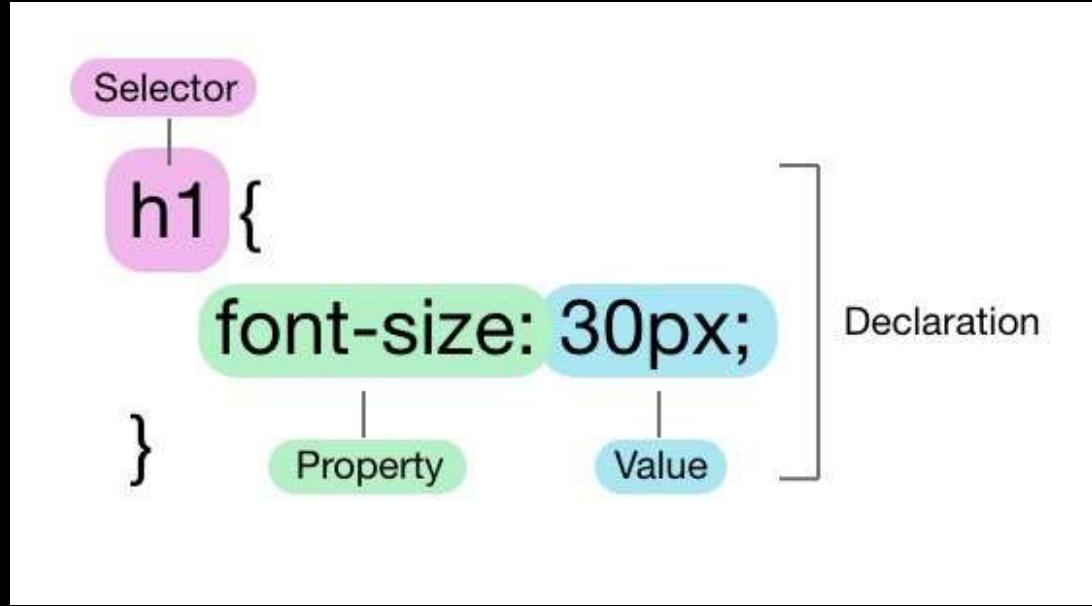


Premium Customer



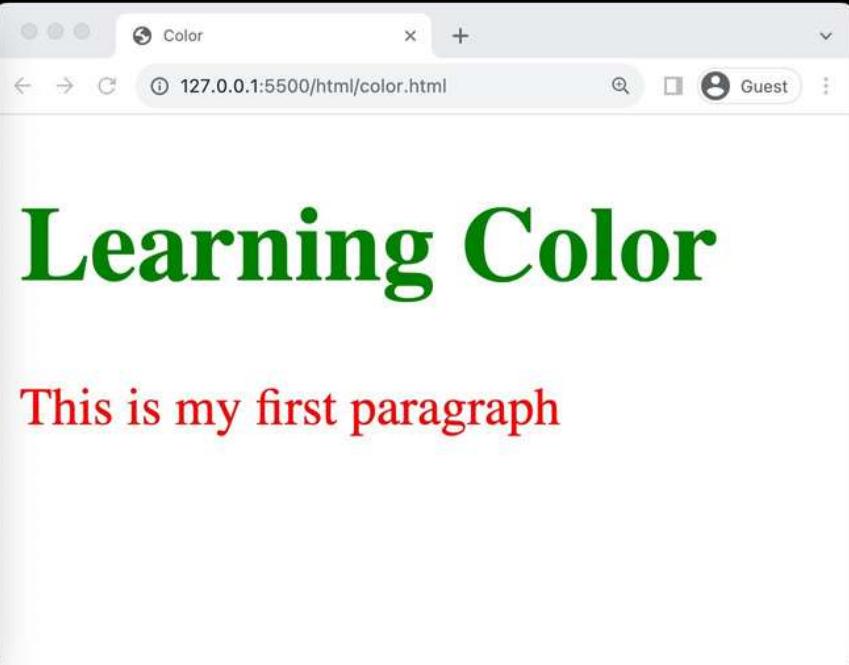
High Salary Developer

1. Basic Syntax



- **Selector:** The HTML element that you want to style. **Property:**
- The attribute you want to change (like font, color, etc.).
- **Value:** The specific style you want to apply to the property (like red, bold, etc.).

2. Color Property



The image shows a split-screen view. On the left is a code editor window titled "color.html" showing the following CSS and HTML code:

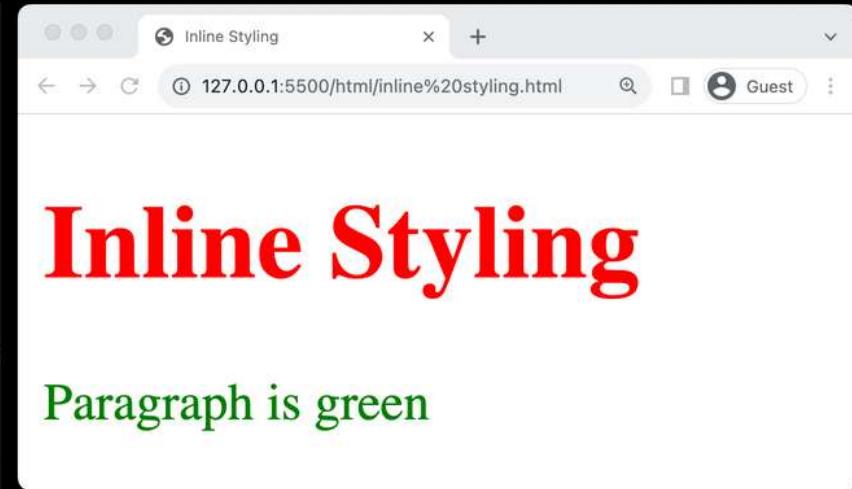
```
html > color.html >
  html > head > style > p
  7   <style>
  8     h1 {
  9       color: green;
 10    }
 11    p {
 12      color: red;
 13    }
 14  </style>
 15 </head>
 16 <body>
 17   <h1>Learning Color</h1>
 18   <p>This is my first paragraph</p>
 19 </body>
```

On the right is a browser window titled "Color" showing the rendered HTML. The title "Learning Color" is displayed in green, and the paragraph "This is my first paragraph" is displayed in red.

- **Definition:** The *CSS* color property defines the text color or foreground color in an HTML element.
- **Enhancement:** Use it to emphasize sections and elevate webpage aesthetics.

3. Including Styles (Inline Styling)

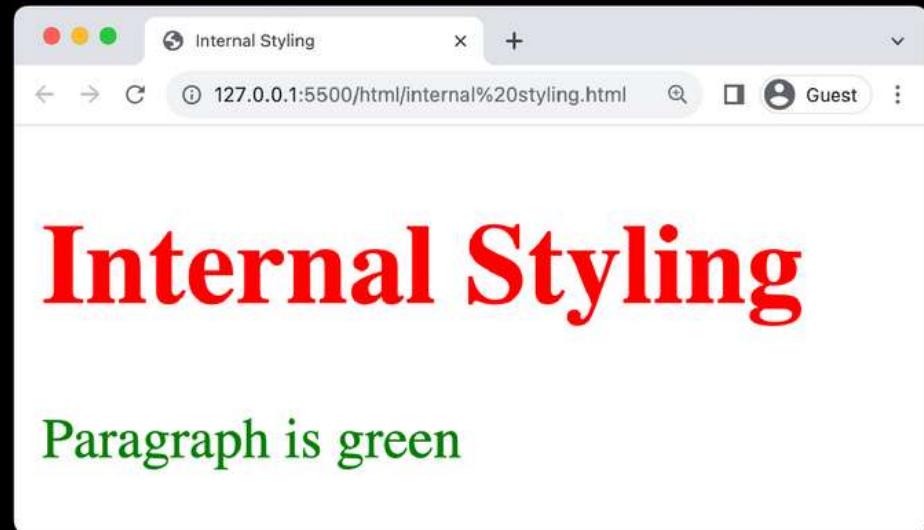
```
html > inline styling.html > html
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4      <title>Inline Styling</title>
5  </head>
6  <body>
7      <h1 style="color: red;">Inline Styling</h1>
8      <p style="color: green;">Paragraph is green<
9          p>
9  </body>
10 </html>
```



- **Direct Application:** Apply styles directly to **HTML** elements using the **style** attribute. **One-off Changes:** Ideal for **single**, unique style alterations.
- **Can Be Cluttered:** May lead to **cluttered** HTML if used extensively.
- **Limited Reusability:** Reduces the **reusability** of CSS rules in larger projects.

3. Including Styles (Internal Styling)

```
3 <head>
4     <title>Internal Styling</title>
5     <style>
6         h1 {color: red;}
7         p {color: green;}
8     </style>
9 </head>
10 <body>
11     <h1>Internal Styling</h1>
12     <p>Paragraph is green</p>
13 </body>
14 </html>
```

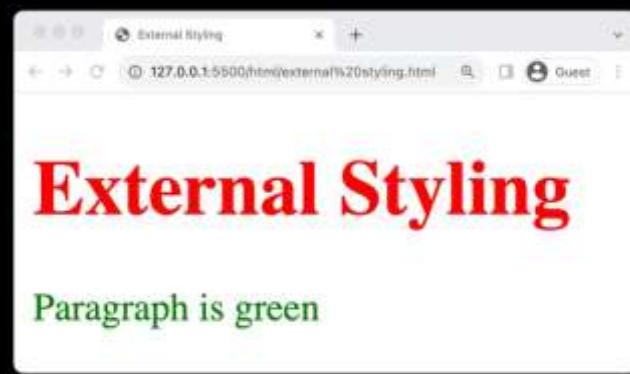


- **Embedded CSS:** Styles are placed within `<style>` tags in the HTML head section. **Cleaner than Inline:** More organized compared to inline styles. **Reusable Styles:** Allows for some reuse of styles across the page.

3. Including Styles (External Styling)

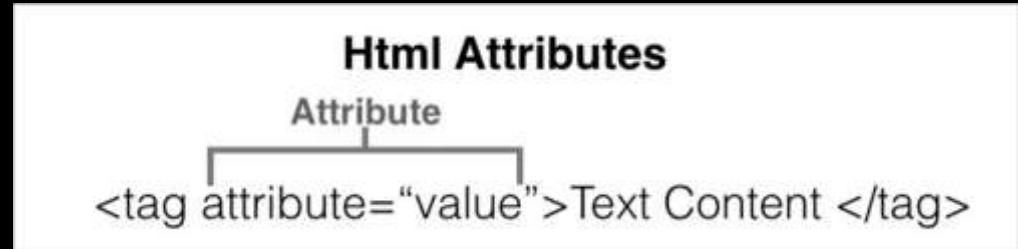
```
3 <head>
4     <title>External Styling</title>
5     <link rel="stylesheet" href="../css/
6         external styling.css">
7 </head>
8 <body>
9     <h1>External Styling</h1>
10    <p>Paragraph is green</p>
11 </body>
```

```
css > # external styling.css > ↵ p
1   <h1>
2       color: red;
3   </h1>
4
5   <p>
6       color: green;
7   </p>
```



- **Separate CSS File:** Stores styles in a **separate .cssfile**, linked to HTML.
- **Reusable:** Enables style **reuse** across multiple webpages. **Link in HTML:** Use
- the **<link>** tag within the **<head>** section to link the CSS. **Relative or**
- **Absolute Path:** The hrefattribute can contain a relative or absolute path to the CSS file.

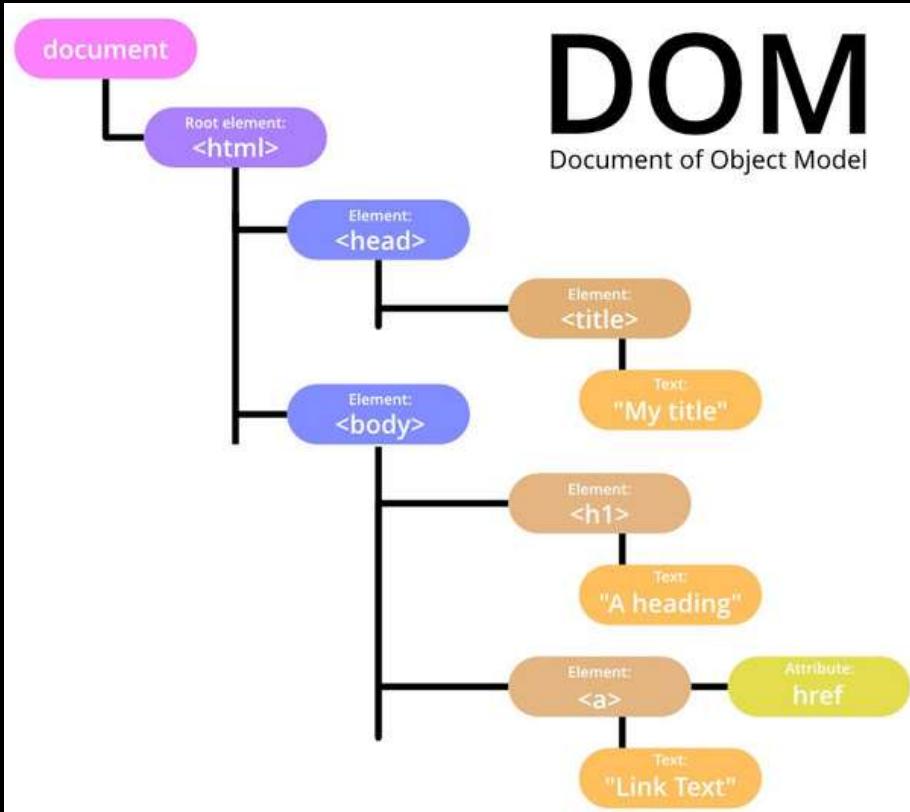
4. HTML Refresher (Tags & Attributes)



1. Elements that are used to create a website are called HTML Tags.
2. Tags can contain content or other HTML tags.
3. Define elements like text, images, links

1. Provides additional information about elements
2. Use name=value format

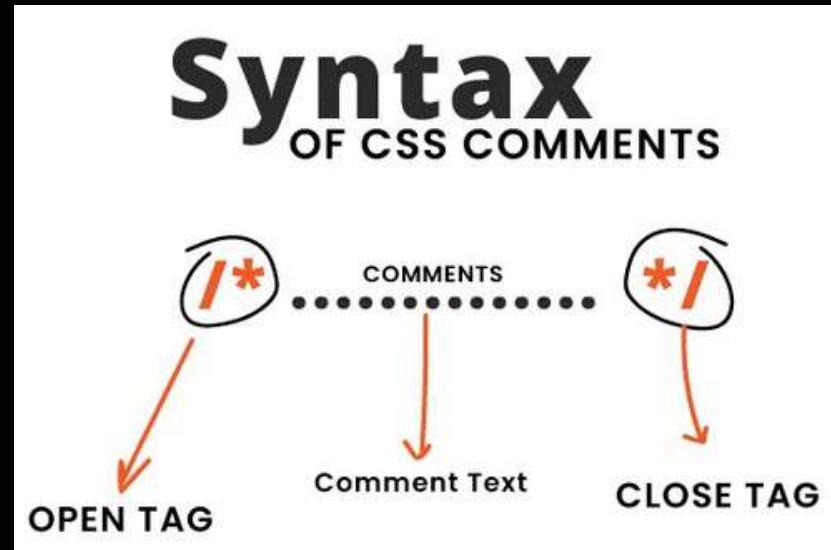
4. HTML Refresher (DOM)



- 1. Structure Understanding:** Helps in understanding the **hierarchical structure** of a webpage, crucial for applying targeted CSS styles.
- 2. Dynamic Styling:** Enables learning about dynamic styling, allowing for **real-time changes** and interactivity through CSS.

5 Comments

1. Used to add notes in HTML or CSS code
2. Not displayed on the web page
3. Syntax: <!--Comment here-->
4. Helpful for code organization
5. Can be multi-line or single-line



8. Selectors (Element selector)

```
3 <head>
4     <title>Element Selector</title>
5     <style>
6         h1 {
7             color: red
8         }
9     </style>
10    </head>
11    <body>
12        <h1>Universal Selector</h1>
13        <p>Lorem ipsum dolor sit amet consectetur
14            adipisicing elit. Nulla, nisi.</p>
```



- **Targets Elements:** Selects HTML elements based on their **tag name**.
- **Syntax:** Simply use the **element's name** **Uniform Styling:** Helps in
- applying **consistent styles** to all instances. **Ease of Use:**
- Straightforward and **easy** to implement for basic styling.

8. Selectors (Universal selector)

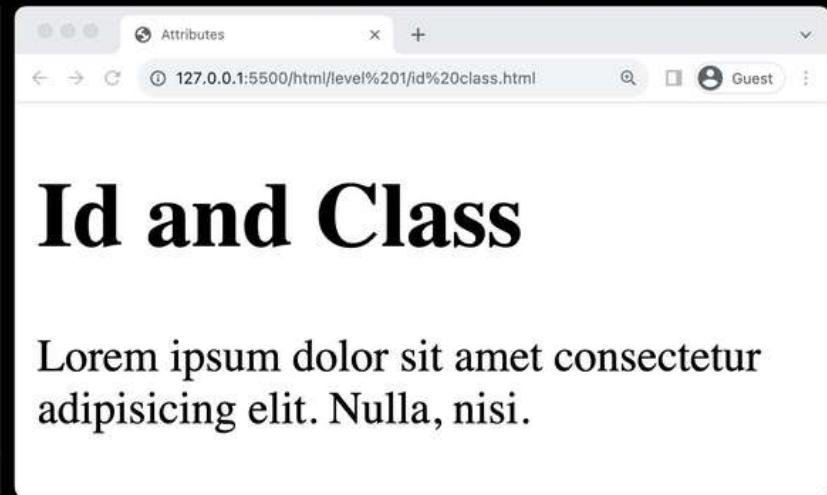
```
3 <head>
4     <title>Universal Selector</title>
5     <style>
6         * {
7             color: red
8         }
9     </style>
10    </head>
11    <body>
12        <h1>Universal Selector</h1>
13        <p>Lorem ipsum dolor sit amet consectetur
14            adipisicing elit. Nulla, nisi.</p>
```



- **Matches All:** Targets and styles **all elements** on a webpage. **Syntax:** Utilized as an asterisk (*). **Resets Styles:** Commonly used to **reset margins** and **paddings** globally.
- **Broad Styling:** Useful for setting universal attributes like **font** or **color**.
- **Usage Caution:** Can cause style **conflicts** due to its wide-reaching effects.

8. Selectors (id& classproperty)

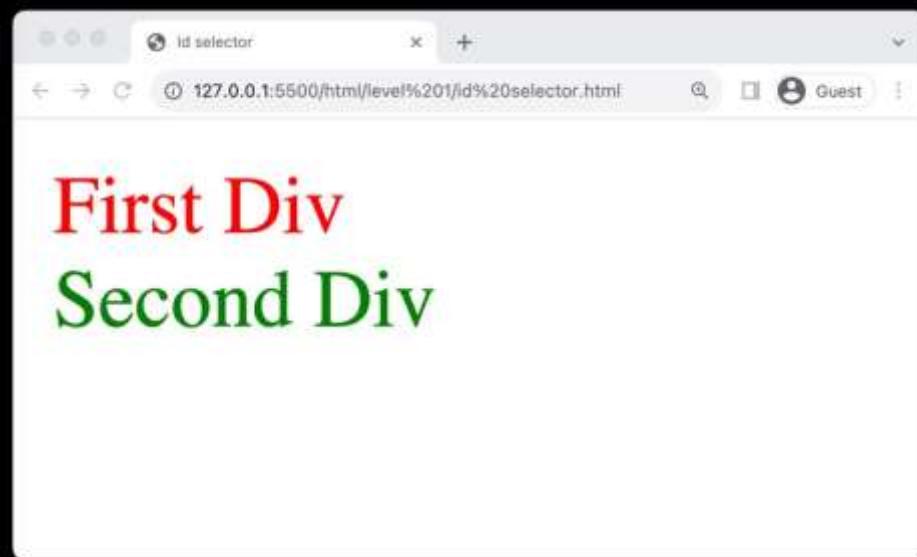
```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <title>Attributes</title>
5 </head>
6 <body>
7   <h1 id="top_heading">Id and Class</h1>
8   <p class="article">Lorem ipsum dolor sit amet
      consectetur adipisicing elit. Nulla, nisi.</p>
9 </body>
10 </html>
```



- **ID Property:** Assigns a unique identifier to a single HTML element.
- **Class Property:** Allows grouping of multiple HTML elements to style them collectively. **Reusable Classes:** Class properties can be reused across different elements for consistent styling.
- **Specificity and Targeting:** Both properties assist in targeting specific elements or groups of elements for precise styling.

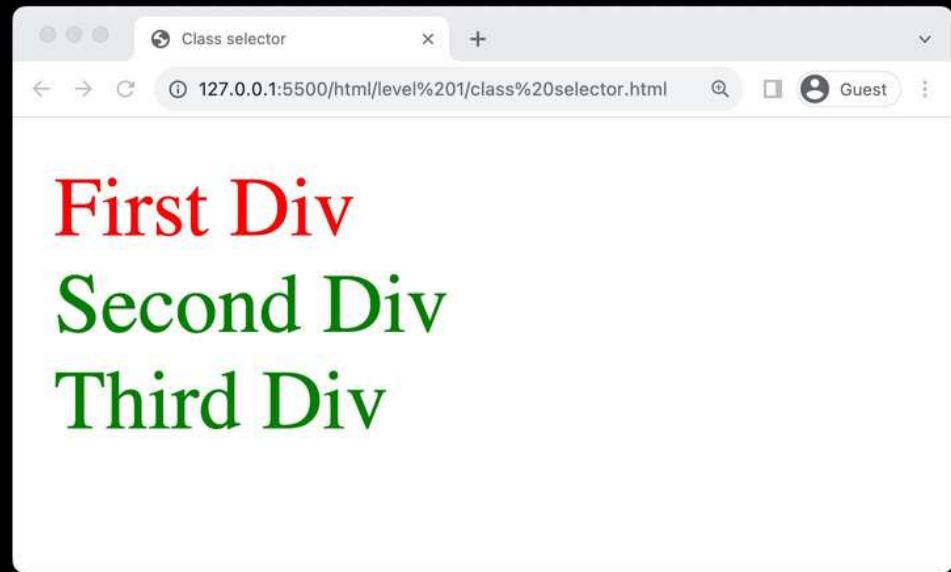
8. Selectors (Id selector)

```
3 <head>
4     <title>Id selector</title>
5     <style>
6         #first { color: red; }
7         #second { color: green; }
8     </style>
9 </head>
10 <body>
11     <div id="first">First Div</div>
12     <div id="second">Second Div</div>
13 </body>
```



8. Selectors (Class selector)

```
<head>
  <title>Class selector</title>
  <style>
    #first { color: red; }
    .second { color: green; }
  </style>
</head>
<body>
  <div id="first">First Div</div>
  <div class="second">Second Div</div>
  <div class="second">Third Div</div>
</body>
```



- **Group Styling:** Allows styling of **multiple elements** grouped under a class. **Syntax:** Utilizes the **dot (.)** symbol. **Reusable:** Can be used on
- **multiple elements** for **consistent styling**. **Versatility:** Ideal for
- **applying styles** to a category of elements.
-

8. Selectors (Group selector)

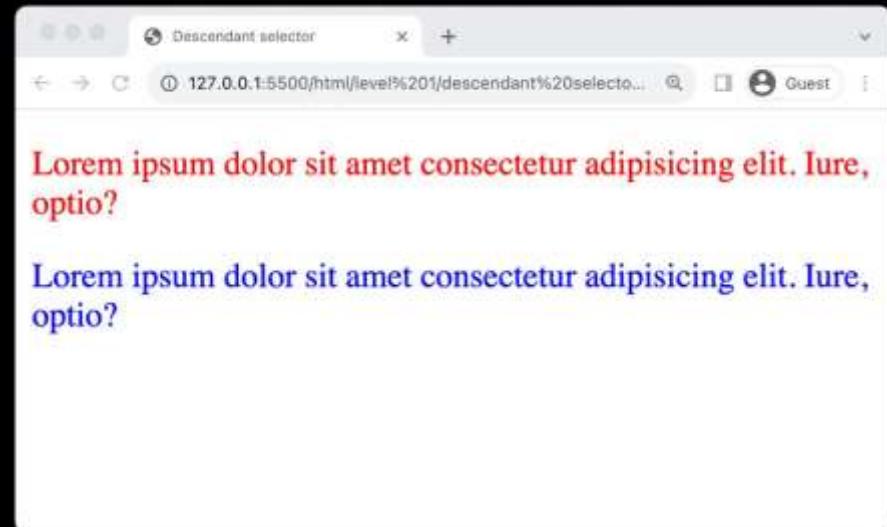
```
<head>
  <title>Group selector</title>
  <style>
    h1, h2, h3 {
      color: red
    }
  </style>
</head>
<body>
  <h1>Heading 1</h1>
  <h2>Heading 2</h2>
  <h3>Heading 3</h3>
  <h4>Heading 4</h4>
</body>
```



- **Multiple Elements:** Styles multiple elements simultaneously.
- **Syntax:** Separates selectors with commas. **Efficiency:** Reduces code redundancy and saves time.

8. Selectors (Descendant selector)

```
<head>
  <title>Descendant selector</title>
  <style>
    div p { color: red }
    p { color: blue }
  </style>
</head>
<body>
  <div>
    <p>
      Lorem ipsum dolor sit amet consectetur adipisicing elit. Iure, optio?
    </p>
  </div>
  <p>
    Lorem ipsum dolor sit amet consectetur adipisicing elit. Iure, optio?
  </p>
</body>
```



- **Nested Targeting:** Styles elements **nested** within a specified element. **Syntax:**
- Separate selectors with **spaces**. **Hierarchy-Based:** Allows styling based on the **hierarchical** structure of HTML.
- **Specific Styling:** Facilitates more **targeted** and specific styling of elements.

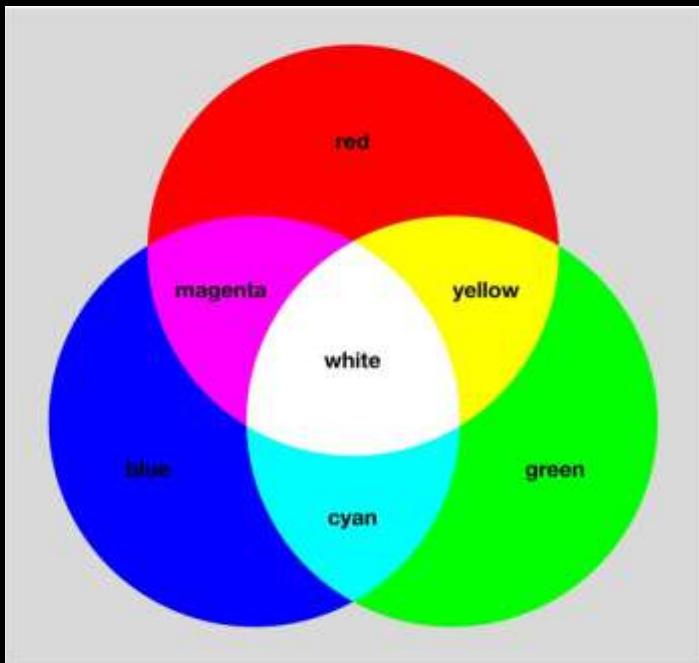
9. Background Color

```
<head>
    <title>Background Color</title>
    <style>
        #first { color: black; background-color: red; }
        #second { color: black; background-color: green; }
        button { color: black; background-color: blue; }
    </style>
</head>
<body>
    <div id="first">First</div>
    <div id="second">Second</div>
    <button>Click Me</button>
</body>
```



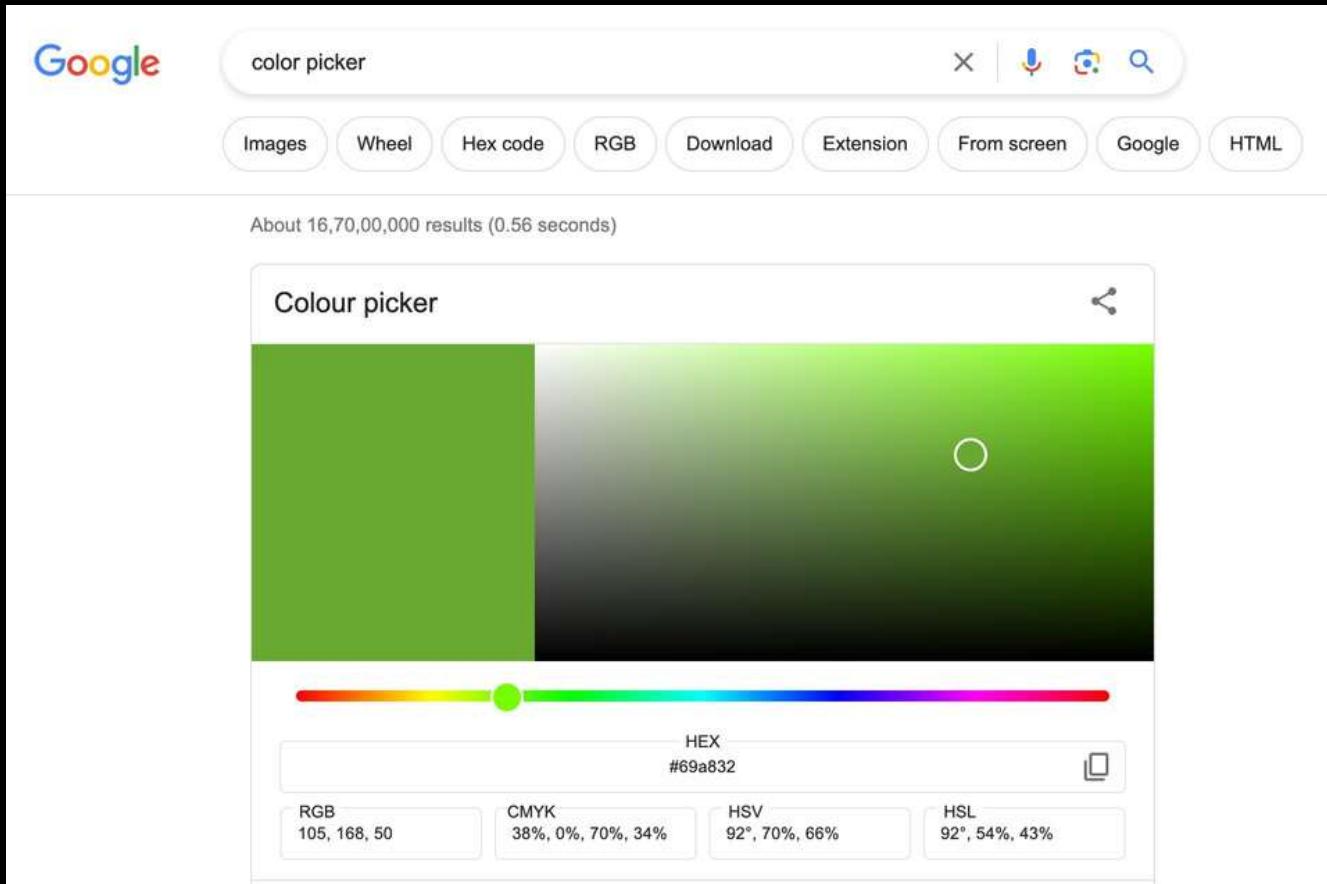
- **Definition:** Sets the background color of an element.
- **Syntax:** Utilized as `background-color: color;` **Visual**
- **Appeal:** Enhances the visual appeal and contrast of webpage elements.

10. Color System (Color Theory)

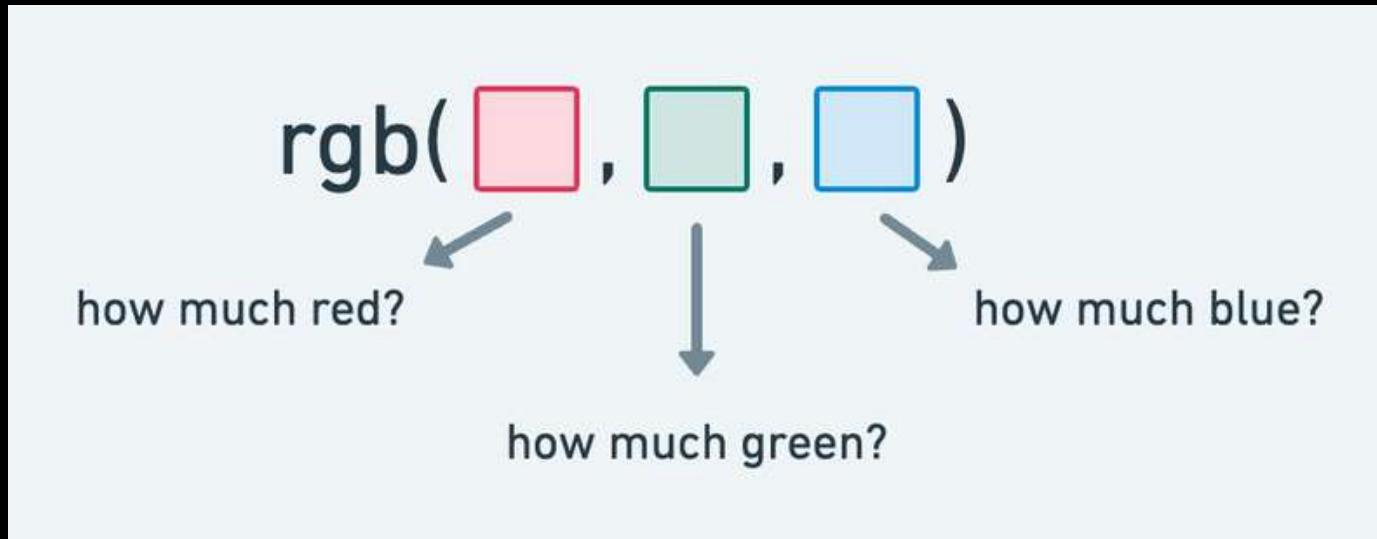


- **RGB Model:** Creates colors by mixing Red (R), Green (G), and Blue (B) light sources. **Additive Model:** More light means increased brightness.
- **Primary Colors:** R, G, and B are the foundational colors.
- **White & Black:** All combined yield white; absence equals black.
- **Color Depth:** Allows for millions of color variations.
-

10. Color System (color picker)



10. Color System (RGB Color Model)



- **Three Channels:** Consists of Red (R), Green (G), and Blue (B) channels to create a variety of colors. **Syntax:** Utilized as `rgb(r, g, b)` where r, g, and b are values between 0 and 255.

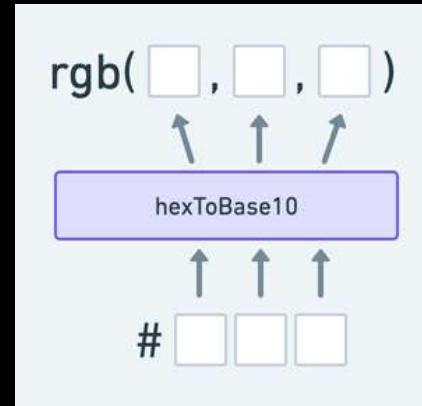
10. Color System (RGB Color Model)

```
<!DOCTYPE html>
<html lang="en">
<head>
    <title>RGB Color</title>
</head>
<body>
    <div style="background-color: red; color: white; padding: 10px; margin-bottom: 5px;">First</div>
    <div style="background-color: green; color: white; padding: 10px; margin-bottom: 5px;">Second</div>
    <div style="background-color: blue; color: white; padding: 10px; margin-bottom: 5px;">Third</div>
    <div style="background-color: #1a8060; color: white; padding: 10px; margin-bottom: 5px;">Fourth</div>
</body>
</html>
```



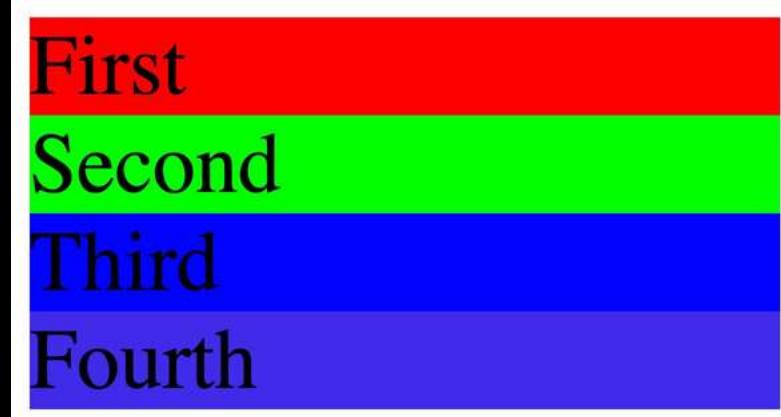
10. Color System (HEX Color Model)

- **Hexadecimal Codes:** Represents colors using hexadecimal values, consisting of **6 digits** combined from numbers and letters (A-F).
- **Syntax:** Written as **#RRGGBB Easy Color**
- **Matching:** Facilitates easy color matching with graphic design tools and branding colors.
- **Web Standards:** Widely supported and a common standard for defining colors in web design



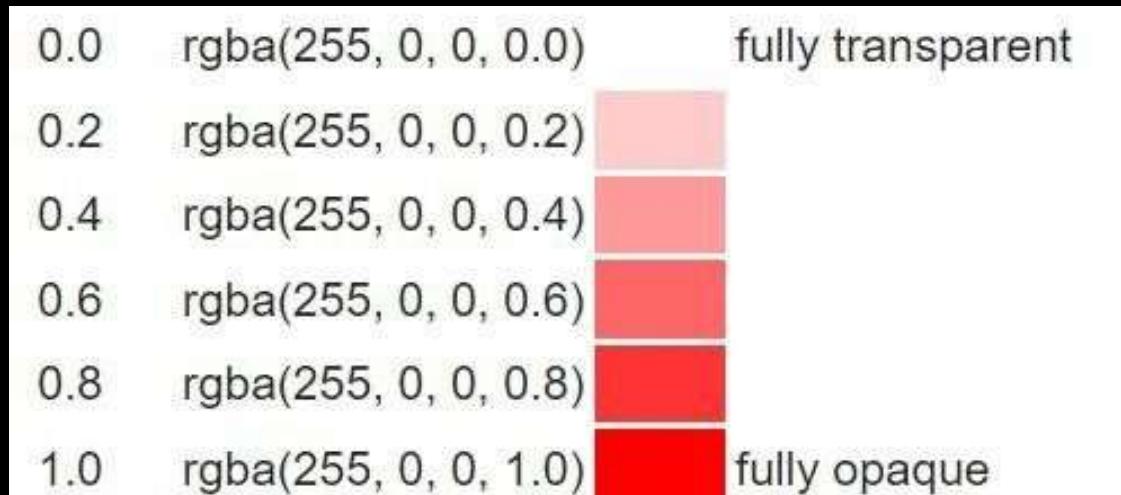
10. Color System (HEX Color Model)

```
<!DOCTYPE html>
<html lang="en">
<head>
    <title>Hex Color</title>
</head>
<body>
    <div style="background-color: #ff0000">First</div>
    <div style="background-color: #00ff00">Second</div>
    <div style="background-color: #0000ff">Third</div>
    <div style="background-color: #402ae9">Fourth</div>
</body>
</html>
```



10. Color System (Alpha Channel)

- **RGBA:** RGB's extension, includes alpha for opacity control (0-1 range).
- **Transparency Control:** Facilitates the adjustment of transparency levels in colors.
- **Visual Effects:** Enables the creation of visual effects like shadows and overlays.
- **Layering:** Assists in layering elements with varying degrees of visibility.



10. Color System (Alpha Channel)

```
<h1 style="color: □rgb(255,0,0,0.1);">First</div>
<h1 style="color: □rgb(255,0,0,0.25);">First</div>
<h1 style="color: □rgb(255,0,0,0.5);">First</div>
<h1 style="color: □rgb(255,0,0,0.75);">First</div>
<h1 style="color: □rgb(255,0,0,1.0);">First</div>
```

First

First

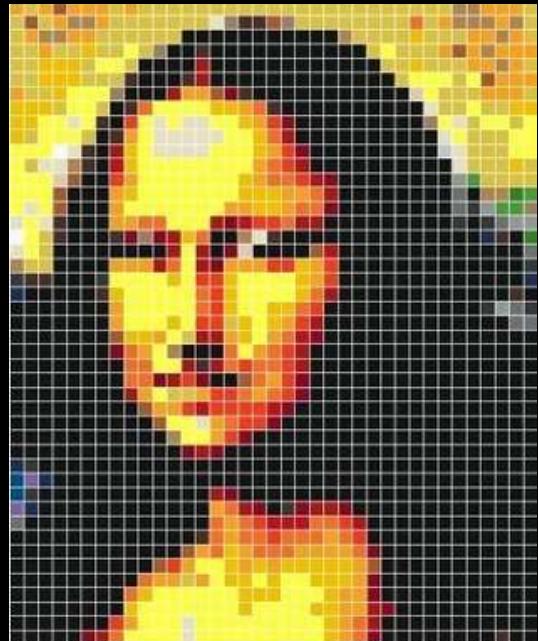
First

First

First

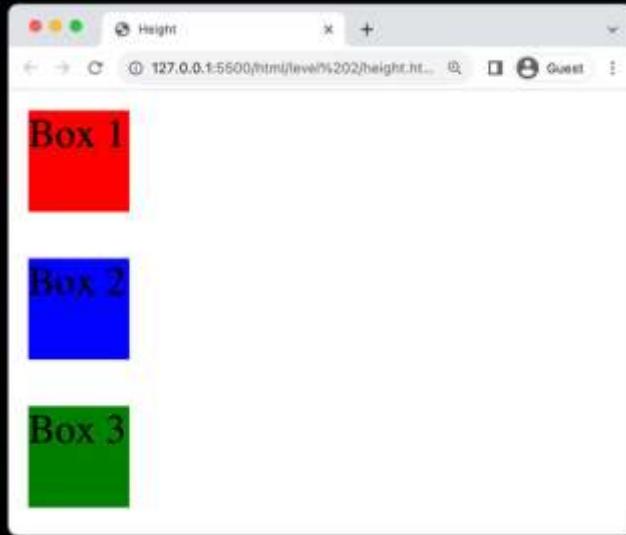
11. Absolute Units

- **Definition:** Pixels (px) are fixed-size units, representing a dot on a computer screen.
- **Precision:** Allows for precise control over element dimensions. **Graphics & Web Design:**
- Commonly used in graphics and web design for setting font sizes, margins, and more. **Cross-Browser Consistency:** Provides consistency across different browsers. **High-DPI Displays:** Can vary in appearance on high-DPI (dots per inch) displays.



12. Height & Width Property

```
<head>
    <title>Height</title>
    <style>
        .box { height: 40px; width: 40px; }
        #box1 {background-color: red;}
        #box2 {background-color: blue;}
        #box3 {background-color: green;}
    </style>
</head>
<body>
    <div id="box1" class="box">Box 1</div> <br>
    <div id="box2" class="box">Box 2</div> <br>
    <div id="box3" class="box">Box 3</div>
</body>
```



- **Dimensions Control:** Used to specify the `height` and `width` of elements.
- **Unit Variability:** Can use units like pixels (px) **Box Model Component:**
- Influences padding, border, and margin. **Min and Max Values:** Can
- utilize `min-height`, `max-height`, `min-width`, and `max-width` to set restrictions on dimensions.

13. Background image Property

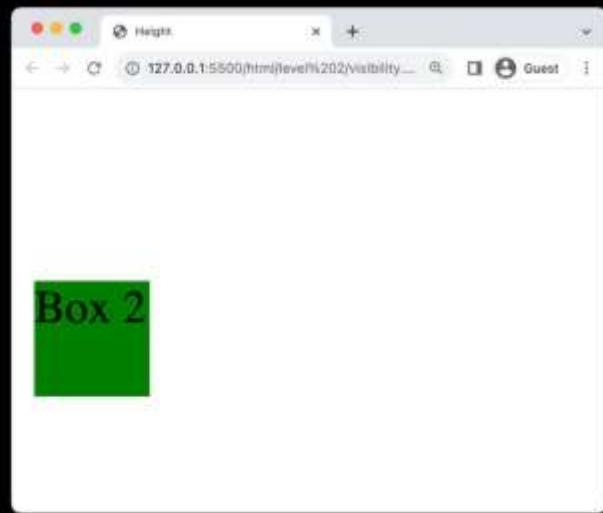
- **Usage:** Adds an **image** as a background to elements.
- **Syntax:** Defined using **background-image:**
`url('path/to/image');`
- **Repetition:** Control image repetition using
background-repeat.
- **Positioning:** Adjust image position using **background-position**.
- **Size Control:** Manipulate image size using
background-size. **Background-Attachment:** Sets
- whether the background image **scrolls** with the element or remains fixed. **Shorthand**(color, image, repeat, attachment, position)

```
<head>
  <title>Background Image</title>
  <style>
    #box1 {
      background-image: url(../../../../images/css.png);
      height: 500px;
      width: 500px;
    }
  </style>
</head>
<body>
  <div id="box1"></div>
</body>
```



14. Visibility Property

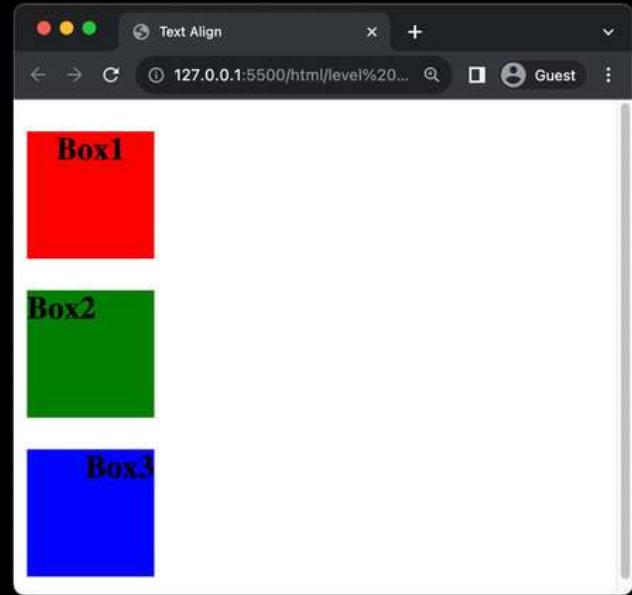
```
<head>
  <title>Visibility</title>
  <style>
    .box { height: 40px; width: 40px; }
    #box1 {background-color: red; visibility: hidden;}
    #box2 {background-color: green; visibility: visible;}
  </style>
</head>
<body>
  <div id="box1" class="box">Box 1</div> <br>
  <div id="box2" class="box">Box 2</div> <br>
</body>
```



- **Usage:** Controls the **visibility** of elements without changing the layout.
- **Values:** Can take **visible**, **hidden**, or **collapse** as values. **Space**
- **Occupancy:** Even when hidden, the element **occupies** space.
- **Interactivity:** Hidden elements are not accessible to **user interactions**.

15. Text-Align Property

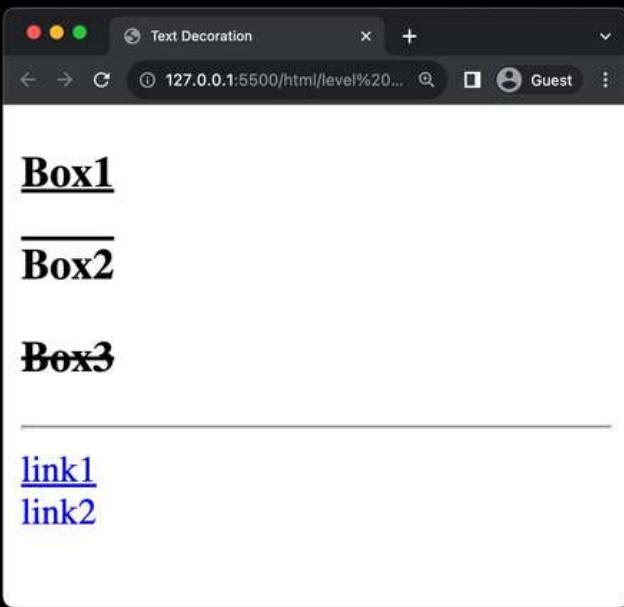
```
<head>
    <title>Text Align</title>
    <style>
        .box {height: 100px; width: 100px;}
        #box1 {background-color: red; text-align: center;}
        #box2 {background-color: green; text-align: left;}
        #box3 {background-color: blue; text-align: right;}
    </style>
</head>
<body>
    <h3 id="box1" class="box">Box1</h3>
    <h3 id="box2" class="box">Box2</h3>
    <h3 id="box3" class="box">Box3</h3>
</body>
```



- **Usage:** Controls the horizontal alignment of text within an element. **Values:** Can take values like left, right, center, and justify.
- **Visual Appeal:** Enhances readability and visual appeal by organizing text neatly.

16. Text-Decoration Property

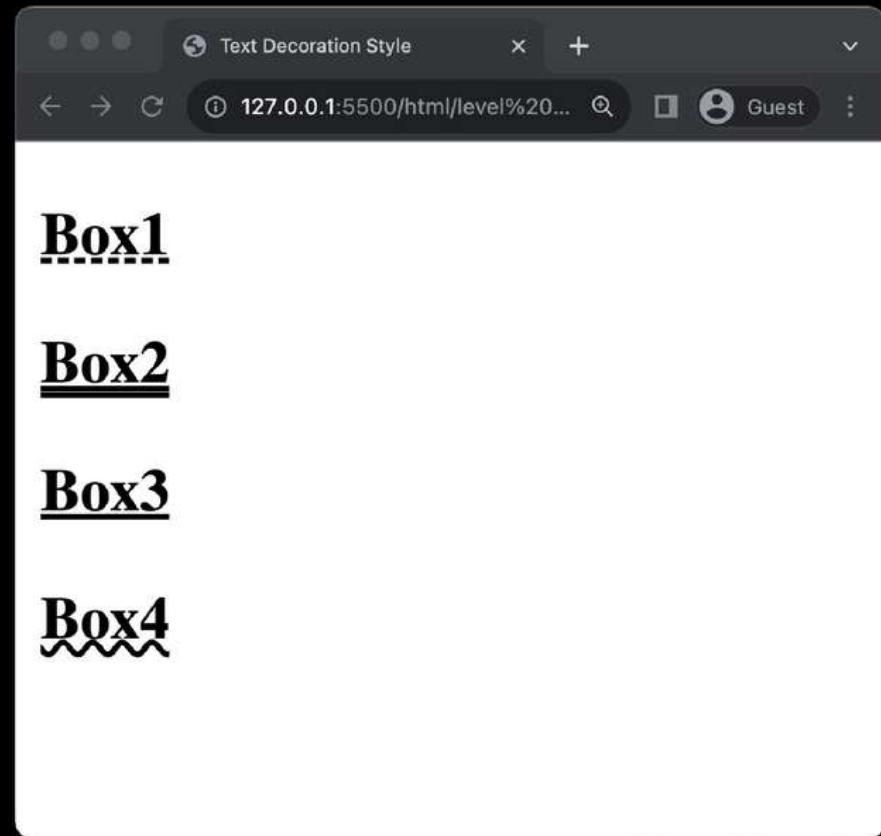
```
<head>
    <title>Text Decoration</title>
    <style>
        #box1 {text-decoration: underline;}
        #box2 {text-decoration: overline;}
        #box3 {text-decoration: line-through;}
    </style>
</head>
<body>
    <h3 id="box1" class="box">Box1</h3>
    <h3 id="box2" class="box">Box2</h3>
    <h3 id="box3" class="box">Box3</h3> <hr>
    <a href="#">link1</a> <br>
    <a href="#" style="text-decoration: none;">link2</a>
</body>
```



- **Usage:** Modifies the appearance of inline text. **Values:** Options include **none**, **underline**, **overline**, and **line-through**.
- **Hyperlinks:** Commonly used to remove underlines from hyperlinks for aesthetic purposes.

16. Text-Decoration Property (style)

```
<!DOCTYPE html>
<html lang="en">
<head>
    <title>Text Decoration Style</title>
    <style>
        .box {text-decoration: underline;}
        #box1 {text-decoration-style: dashed;}
        #box2 {text-decoration-style: double;}
        #box3 {text-decoration-style: solid;}
        #box4 {text-decoration-style: wavy;}
    </style>
</head>
<body>
    <h3 id="box1" class="box">Box1</h3>
    <h3 id="box2" class="box">Box2</h3>
    <h3 id="box3" class="box">Box3</h3>
    <h3 id="box4" class="box">Box4</h3>
</body>
</html>
```



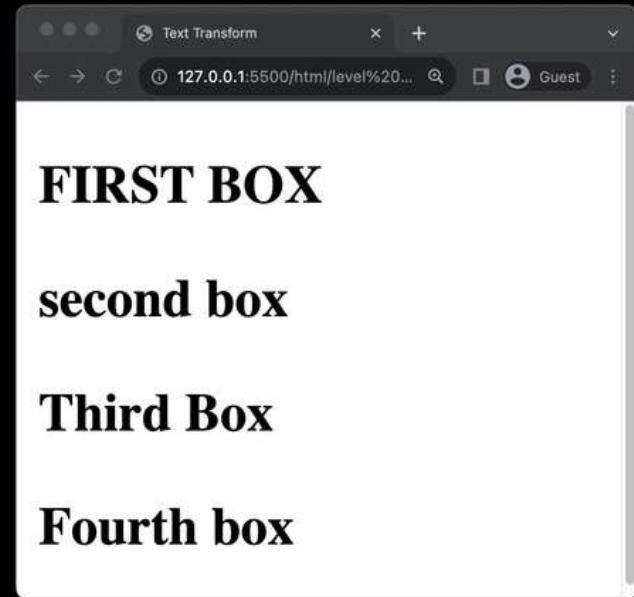
16. Text-Decoration Property (color)

```
<head>
  <title>Text Decoration Style</title>
  <style>
    .box {text-decoration: underline;}
    #box1 {text-decoration-color: red;}
    #box2 {text-decoration-color: blue;}
    #box3 {text-decoration-color: green;}
    #box4 {text-decoration-color: rgb(86, 20, 86);}
  </style>
</head>
<body>
  <h3 id="box1" class="box">Box1</h3>
  <h3 id="box2" class="box">Box2</h3>
  <h3 id="box3" class="box">Box3</h3>
  <h3 id="box4" class="box">Box4</h3>
</body>
```



17. Text-Transform Property

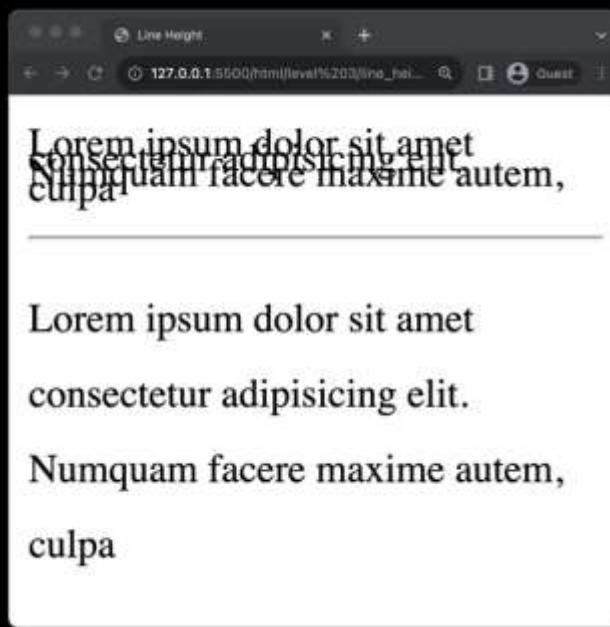
```
<head>
    <title>Text Transform</title>
    <style>
        #box1 {text-transform: uppercase;}
        #box2 {text-transform: lowercase;}
        #box3 {text-transform: capitalize;}
        #box4 {text-transform: none;}
    </style>
</head>
<body>
    <h3 id="box1" class="box">First box</h3>
    <h3 id="box2" class="box">Second box</h3>
    <h3 id="box3" class="box">Third box</h3>
    <h3 id="bo43" class="box">Fourth box</h3>
</body>
```



- **Usage:** Controls the **capitalization** of text. **Common Values:** Can be **uppercase**, **lowercase**, or **capitalize**. **None Value:** **none** value **disables** text transformations.
- **Typography:** Useful for setting text style and improving **typography** in web design.
-

18. Line Height

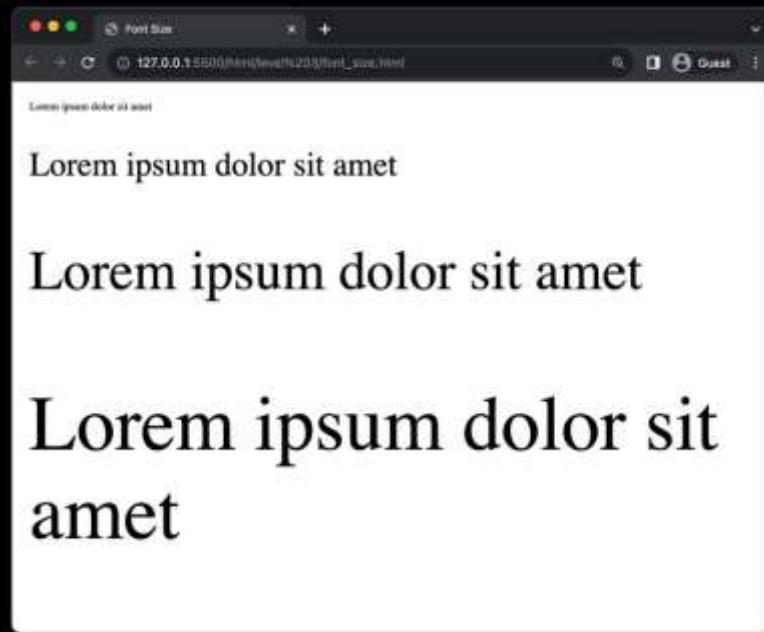
```
<head>
    <title>Line Height</title>
    <style>
        #first { line-height: 6px; }
        #second { line-height: 30px; }
    </style>
</head>
<body>
    <p id="first">Lorem ipsum dolor sit amet
    consectetur adipisicing elit. Numquam facere
    maxime autem, culpa</p> <hr>
    <p id="second">Lorem ipsum dolor sit amet
    consectetur adipisicing elit. Numquam facere
    maxime autem, culpa</p>
</body>
```



- **Usage:** Adjusts the amount of space above and below inline elements.
- **Readability:** Enhances text readability by preventing overcrowding.
- **Vertical Spacing:** Useful for controlling vertical spacing between lines of text.

19. Font Property (font-size)

```
<head>
  <title>Font Size</title>
  <style>
    #first {font-size: 5px;}
    #second {font-size: 15px;}
    #third {font-size: 25px;}
    #fourth {font-size: 35px;}
  </style>
</head>
<body>
  <p id="first">Lorem ipsum dolor sit amet</p>
  <p id="second">Lorem ipsum dolor sit amet</p>
  <p id="third">Lorem ipsum dolor sit amet</p>
  <p id="fourth">Lorem ipsum dolor sit amet</p>
</body>
```



- **Usage:** Sets the **size of the font** in web content. **Responsiveness:** Helps in creating responsive designs adaptable to various screen sizes. **Readability:** Crucial for ensuring the **readability** of text on websites.
-

19. Font Property (font-weight)

```
<head>
  <title>Font Weight</title>
  <style>
    #first {font-weight: 100;}
    #second {font-weight: 400;}
    #third {font-weight: 600;}
    #fourth {font-weight: 900;}
  </style>
</head>
<body>
  <p id="first">Lorem ipsum dolor sit amet</p>
  <p id="second">Lorem ipsum dolor sit amet</p>
  <p id="third">Lorem ipsum dolor sit amet</p>
  <p id="fourth">Lorem ipsum dolor sit amet</p>
</body>
```



- **Usage:** Defines the **thickness** of characters in a font. **Values:** Can take values like **normal**, **bold**, **bolder**, or numeric values (**100** to **900**). **Text Emphasis:** Utilized to emphasize text or **create contrast** between text elements.

19. Font Property (font-style)

```
<head>
    <title>Font Style</title>
    <style>
        #first {font-style: italic;}
        #second {font-style: oblique;}
        #third {font-style: normal;}
    </style>
</head>
<body>
    <p id="first">Lorem ipsum dolor sit amet</p>
    <p id="second">Lorem ipsum dolor sit amet</p>
    <p id="third">Lorem ipsum dolor sit amet</p>
</body>
```



- **Usage:** Controls the **style** of the font, mainly affecting its inclination.
- **Values:** Common values are **normal**, **italic**, and **oblique**. **Text**
- **Formatting:** Useful for **highlighting** or distinguishing certain text segments.

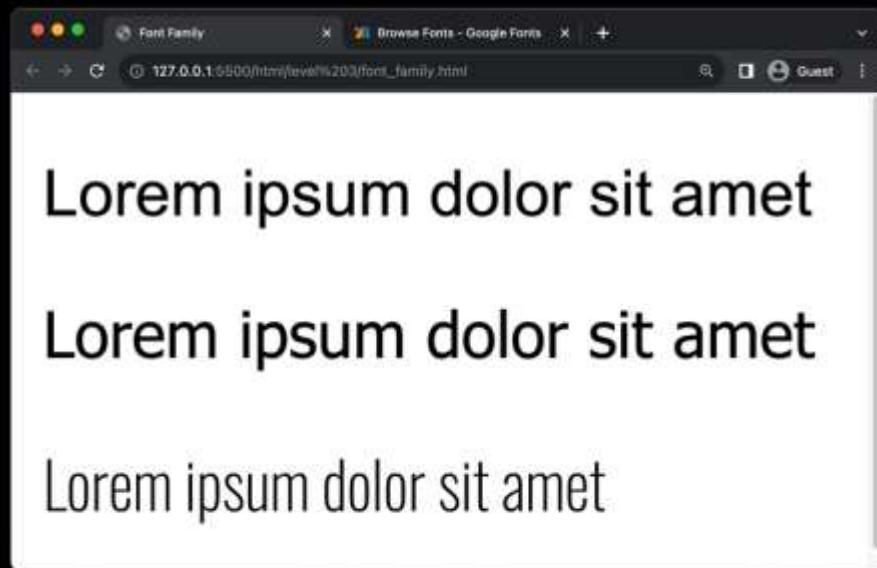
20. Font Family

- **Usage:** Defines which font should be used for text within an element. **Specific Fonts:** Common choices include Arial, Segoe UI, Times New Roman, and others. **Fallback Mechanism:**
- Incorporate a fallback font family in case the primary font is unavailable; helps in maintaining the site aesthetics.
- **Web Safe Fonts:** Employ web-safe fonts to ensure consistency across different browsers and operating systems. **Generic Family:** Always end the font family list with a generic family like serif or sans-serif as a last resort option.

Arial Narrow
Book Antiqua
Cambria
Century Gothic
Consolas
COPPERPLATE
Georgia
Impact
Lucida Sans Unicode
Papyrus
Script MJ Bold
Tahoma
Times New Roman
Verdana

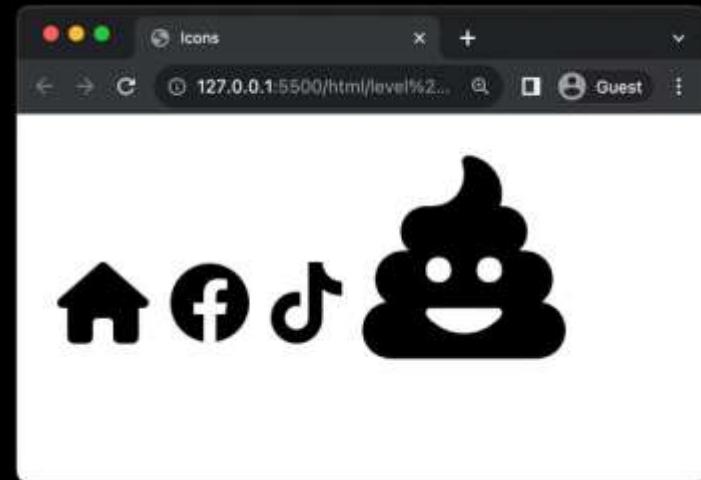
20. Font Family

```
<head>
  <title>Font Family</title>
  <link rel="preconnect" href="https://fonts.googleapis.com">
  <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>
  <link href="https://fonts.googleapis.com/css2?family=Oswald:wght@200&display=swap" rel="stylesheet">
  <style>
    #first {font-family: Arial, Helvetica, sans-serif;}
    #second {font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;}
    #third {font-family: 'Oswald', sans-serif;}
  </style>
</head>
<body>
  <p id="first">Lorem ipsum dolor sit amet</p>
  <p id="second">Lorem ipsum dolor sit amet</p>
  <p id="third">Lorem ipsum dolor sit amet</p>
</body>
```



21. Icons using Fonts

```
<head>
  <title>Icons</title>
  <script src="https://kit.fontawesome.com/43290fa92d.js" crossorigin="anonymous"></script>
</head>
<body>
  <i class="fa-solid fa-house"></i>
  <i class="fa-brands fa-facebook"></i>
  <i class="fa-brands fa-tiktok"></i>
  <i class="fa-solid fa-poo" style="font-size: 40px;"></i>
</body>
```



Using <https://fontawesome.com>

22. What is Box Model

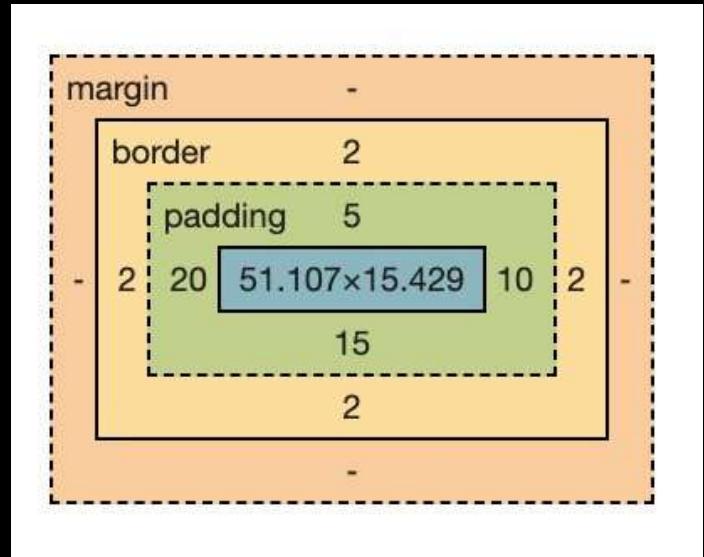
ਮਠਾਈ ਮਾਡਾਰ ਪੇ



- **Core Concept:** Central concept in CSS that outlines the design and layout of elements on the web page. **Components:** Consists of four main components -**margin**, **border**, **padding**, and **content**. **Margin:** The space outside the border, separating the element from others.
- **Border:** The outline that encapsulates the padding and content. **Padding:** The space between the border and the actual content, providing a buffer. **Content:** The innermost layer where text, images, or other media are housed.

23. Padding Property

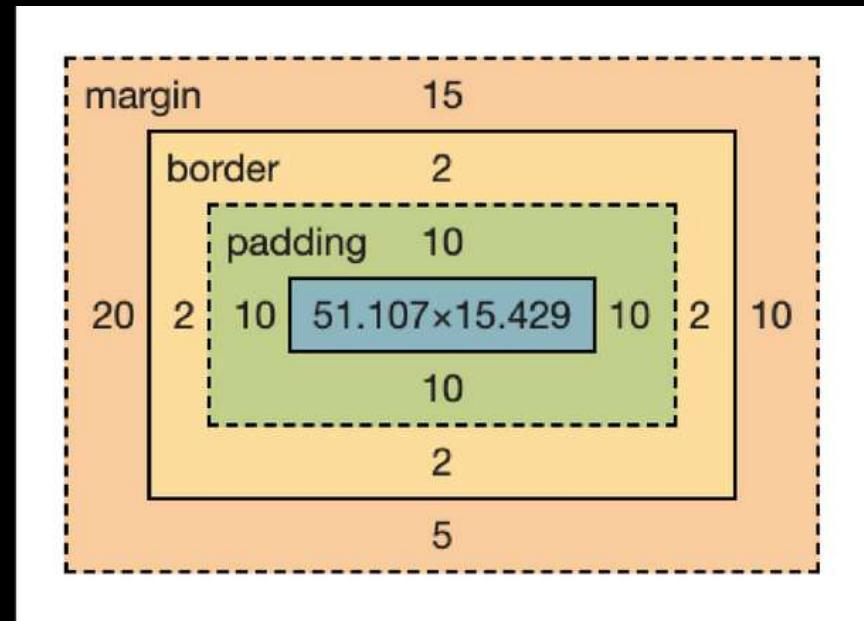
```
<head>
  <title>Padding</title>
  <style>
    * { margin: 0; padding: 0; }
    #button1 {
      padding: 5px 10px 15px 20px;
      background-color: aquamarine;
    }
  </style>
</head>
<body>
  <button id="button1">Click Me</button>
</body>
```



- **Usage:** Defines the space between the content of an element and its border. **Individual Sides:** Allows setting padding for individual sides
- using **padding-top**, **padding-right**, **padding-bottom**, and **padding-left**.
- **Shorthand:** Can use shorthand property padding to set all sides at once, e.g., **padding: 10px 20px 10px 20px**.

24. Margin Property

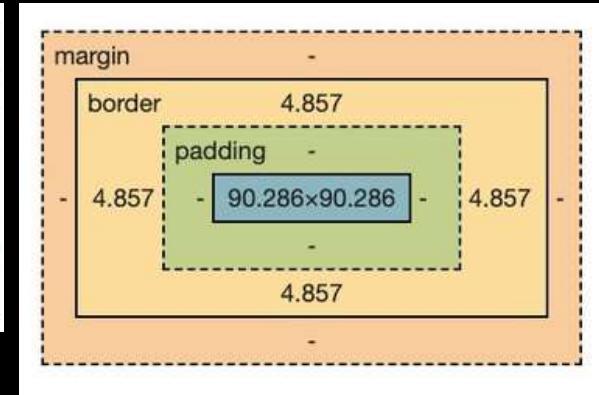
```
<head>
    <title>margin</title>
    <style>
        * { margin: 0; padding: 0; }
        #button1 {
            margin: 15px 10px 5px 20px;
            padding: 10px;
            background-color: aquamarine;
        }
    </style>
</head>
<body>
    <button id="button1">Click Me</button>
</body>
```



- **Functionality:** Sets the space around elements, separating them from others.
- **Individual Sides:** Customizable for **top**, **right**, **bottom**, and **left** sides.
- **Shorthand:** Allows quick setup, e.g., `margin: 10px 20px`. (clockwise) **Auto**
- **Value:** Can be used for central alignment with **auto** value.

24. Border Property

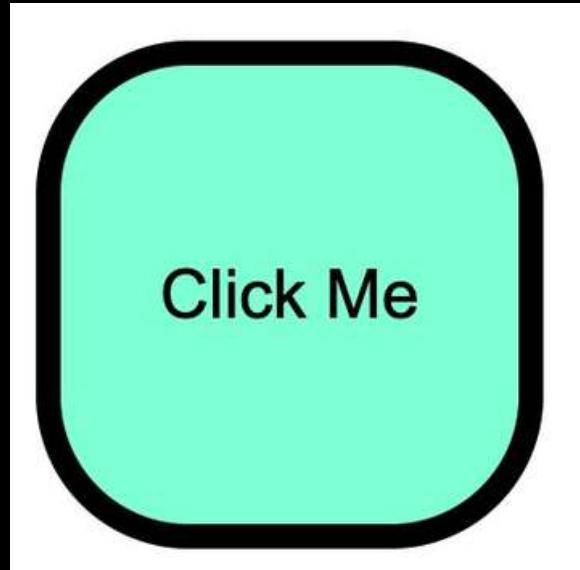
```
<head>
    <title>Border</title>
    <style>
        * { margin: 0; padding: 0; }
        #button1 {
            height: 100px; width: 100px;
            border: 5px dashed black;
            background-color: aquamarine;
        }
    </style>
</head>
<body>
    <button id="button1">Click Me</button>
</body>
```



- **Usage:** Creates an outline around HTML elements. **Components:**
- Defined by **width**, **style**, and **color** attributes. **Styles:** Includes options like **solid**, **dashed**, and **dotted**. **Shorthand:** Can set attributes at once, e.g., **border: 2px solid black**.

24. Border Property (border radius)

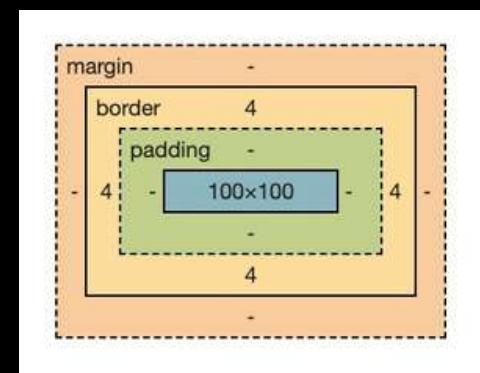
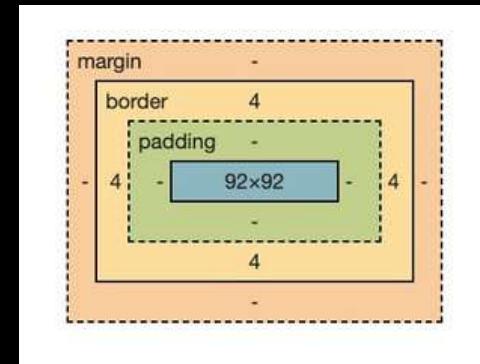
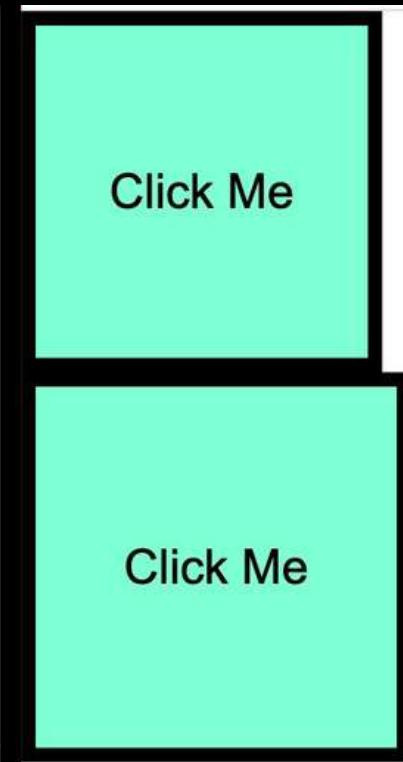
```
<head>
  <title>Border Radius</title>
  <style>
    * { margin: 0; padding: 0; }
    #button1 {
      height: 100px; width: 100px;
      border: 5px solid black;
      border-radius: 30px;
      background-color: aquamarine;
    }
  </style>
</head>
<body>
  <button id="button1">Click Me</button>
</body>
```



- **Usage:** Used to **create rounded corners** for elements. **Individual**
- **Corners:** Allows setting different radii for each corner.
- **Shorthand:** e.g., **border-radius: 10px 20px**.

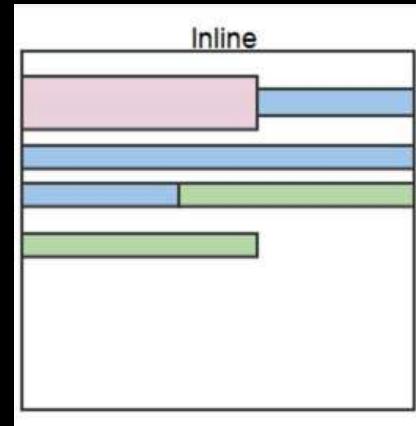
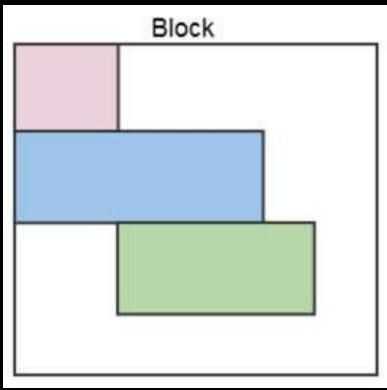
24. Border Property (box sizing)

```
<head>
  <title>Box Sizing</title>
  <style>
    * { margin: 0; padding: 0; }
    button {
      height: 100px; width: 100px;
      border: 4px solid black;
      background-color: aquamarine;
    }
    #button1 { box-sizing: border-box; }
    #button2 { box-sizing: content-box; }
  </style>
</head>
<body>
  <button id="button1">Click Me</button>
  <button id="button2">Click Me</button>
</body>
```



26. Display Property

(Block / Inline Elements)



Block Elements

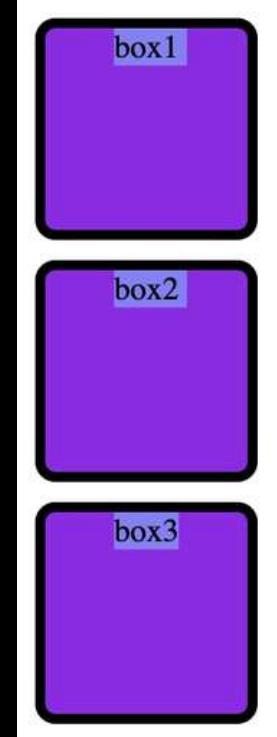
- **New Line:** Start on a new line. **Full Width:**
- Take up all horizontal space. **Styling:** Can have margins and padding. **Size:** Width and height can be set. **Examples:** <div>, <p>, <h1>, , .

Inline Elements

- **Flow:** Stay in line with text. **Width:** Just as wide as the content. **No Break:** No new line between elements. **Limited Styling:** Can't set size easily. **Examples:** , <a>, , .

26. Display Property (Block)

```
.box {  
    height: 100px;  
    width: 100px;  
    background-color: blueviolet;  
    margin: 10px;  
    text-align: center;  
    border: 5px solid black;  
    border-radius: 10px;  
  
    display: block;  
}
```



```
<head>  
    <title>Display Block</title>  
    <link rel="stylesheet" href="../../css/  
    level 5/display.css">  
</head>  
<body>  
    <div id="parent">  
        <div id="div1" class="box">box1</div>  
        <div id="div2" class="box">box2</div>  
        <div id="div3" class="box">box3</div>  
    </div>  
</body>
```

26. Display Property (Inline)

```
.box {  
    height: 100px;  
    width: 100px;  
    background-color: blueviolet;  
    margin: 10px;  
    text-align: center;  
    border: 5px solid black;  
    border-radius: 10px;  
  
    display: inline;  
}
```

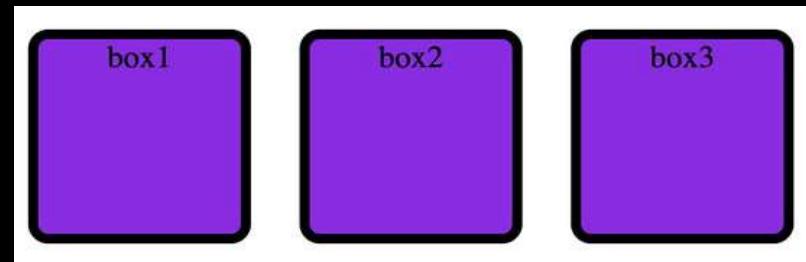
```
<!DOCTYPE html>  
<html lang="en">  
<head>  
    <title>Display Inline</title>  
    <link rel="stylesheet" href="../../css/  
        level 5/display_inline.css">  
</head>  
<body>  
    <div id="parent">  
        <div id="div1" class="box">box1</div>  
        <div id="div2" class="box">box2</div>  
        <div id="div3" class="box">box3</div>  
    </div>  
</body>  
</html>
```



26. Display Property (Inline-Block)

```
.box {  
    height: 100px;  
    width: 100px;  
    background-color: blueviolet;  
    margin: 10px;  
    text-align: center;  
    border: 5px solid black;  
    border-radius: 10px;  
  
    display: inline-block;  
}
```

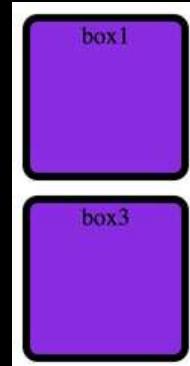
```
<head>  
    <title>Display Inline Block</title>  
    <link rel="stylesheet" href="../../css/  
        level 5/display_inline_block.css">  
</head>  
<body>  
    <div id="parent">  
        <div id="div1" class="box">box1</div>  
        <div id="div2" class="box">box2</div>  
        <div id="div3" class="box">box3</div>  
    </div>  
</body>
```



26. Display Property (None)

```
.box {  
    height: 100px;  
    width: 100px;  
    background-color: blueviolet;  
    margin: 10px;  
    text-align: center;  
    border: 5px solid black;  
    border-radius: 10px;  
}  
  
#div2 {  
    display: none;  
}
```

```
<head>  
    <title>Display None</title>  
    <link rel="stylesheet" href="../../css/  
        level 5/display_none.css">  
</head>  
<body>  
    <div id="parent">  
        <div id="div1" class="box">box1</div>  
        <div id="div2" class="box">box2</div>  
        <div id="div3" class="box">box3</div>  
    </div>  
</body>
```



27. Responsive Website



27. Responsive Website



1. Adapts layout for different screen sizes
2. Flexible layouts
3. Optimizes images and assets
4. Enhances user experience on mobile and desktop

28. Relative Units



CSS Units Cheat Sheet

px

Absolute pixel value

%

A percentage of the parent element.
100% is the width of the parent element

em

Relative to the font size of the element

vh

Relative to 1% of the viewport's height

rem

Relative to the font size of
the root element

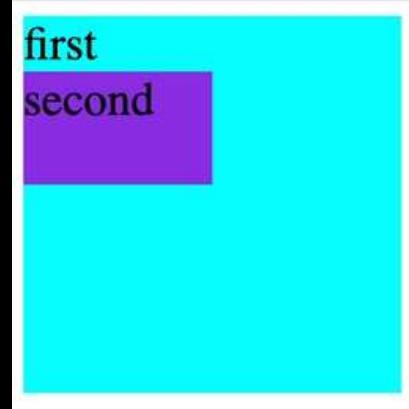
vw

Relative to 1% of the viewport's width

28.Relative Units (Percentage)

```
#first {  
    height: 200px;  
    width: 200px;  
    background-color: #aqua;  
    font-size: 25px;  
}  
  
#second {  
    background-color: #blueviolet;  
    width: 50%;  
    height: 30%;  
}
```

```
<body>  
    <div id="first">  
        first  
        <div id="second">  
            second  
        </div>  
    </div>  
</body>
```

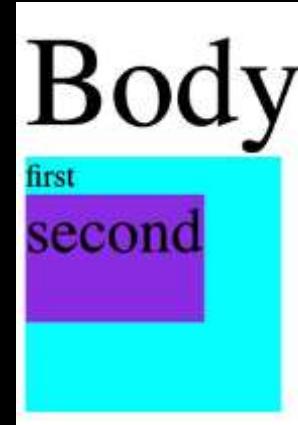


- **Relative Sizing:** Facilitates dynamic sizing **relative to parents**.
- **Adaptability:** Ensures **responsiveness** across various screens.
- **Dimensions:** Quickly set width and height as a percentage.

28.Relative Units (EM)

```
body {  
    font-size: 100px;  
}  
  
#first {  
    height: 200px;  
    width: 200px;  
    background-color: #aqua;  
    font-size: 25px;  
}  
  
#second {  
    background-color: #blueviolet;  
    width: 70%;  
    height: 50%;  
    font-size: 2em;  
}
```

```
<body>  
    Body  
    <div id="first">  
        first  
        <div id="second">  
            second  
        </div>  
    </div>  
</body>
```



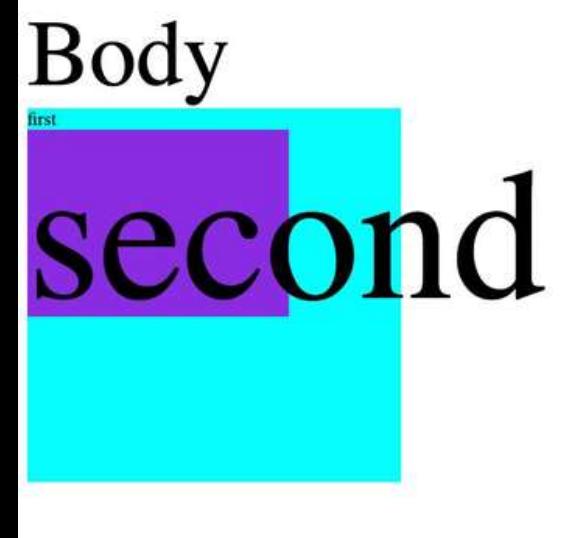
- **Relative Unit:** Sized relative to the parent element's font size.
- **Scalability:** Facilitates easy scaling of elements for responsive design.
- **Font Sizing:** Commonly used for setting font sizes adaptively.

28.Relative Units

(REM)

```
* {  
    font-size: 50px;  
}  
  
#first {  
    height: 200px;  
    width: 200px;  
    background-color: #aqua;  
    font-size: 10px;  
}  
  
#second {  
    background-color: #blueviolet;  
    width: 70%;  
    height: 50%;  
    font-size: 2rem;  
}
```

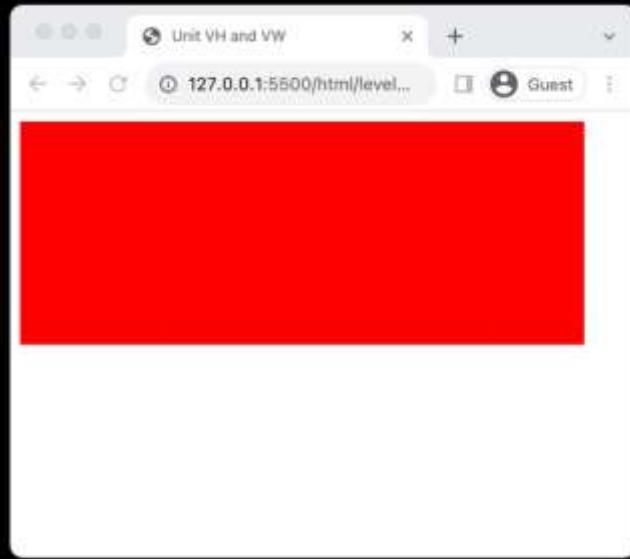
```
<body>  
    Body  
        <div id="first">  
            first  
                <div id="second">  
                    second  
                </div>  
            </div>  
        </body>
```



- **Relative Sizing:** Facilitates dynamic sizing relative to root element.
- **Adaptability:** Ensures responsiveness across various screens.
- **Dimensions:** Quickly set width and height as a percentage.

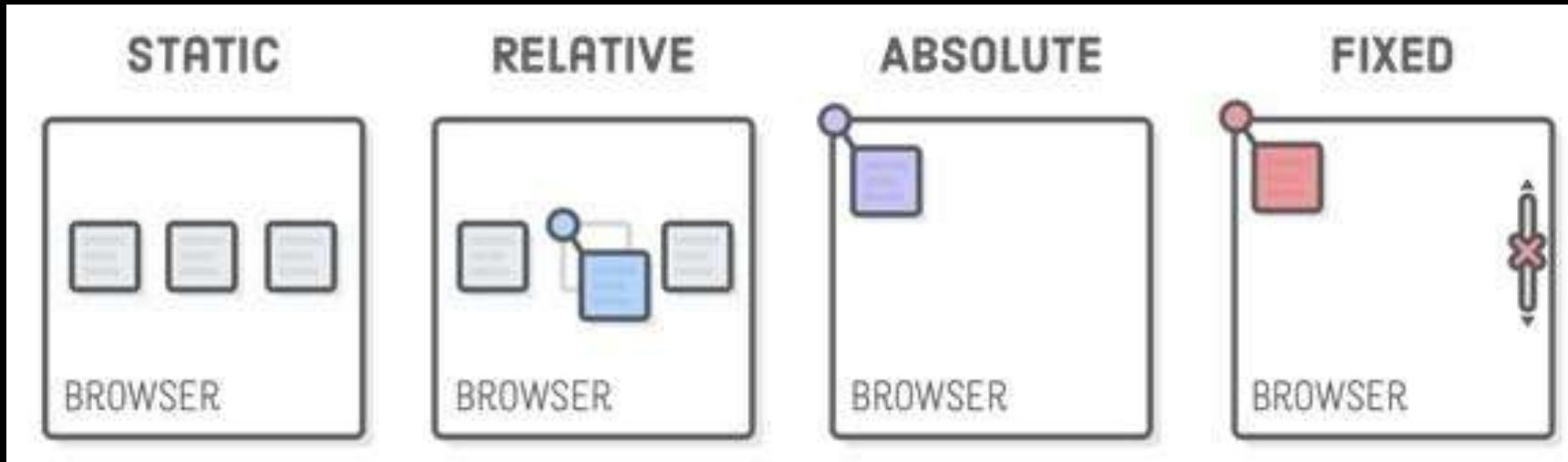
28.Relative Units (vw/vH)

```
<head>
  <title>Unit VH and VW</title>
  <style>
    #first {
      height: 50vh;
      width: 90vw;
      background-color: red;
    }
  </style>
</head>
<body>
  <div id="first"></div>
</body>
```



- **Viewport Relative Units:** Units based on viewport's width (vw) or height (vh) for responsive design. **Responsive Layouts:** Essential for creating adaptive layouts; e.g.,
- **height: 100vh** for full-screen sections. **Element Sizing:** Useful for defining heights and widths that scale with the
- **viewport.**

29. Position Property

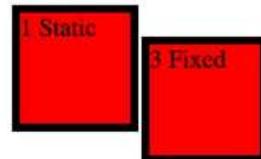


- **Static(default)** : Elements follow the normal document flow. (`top`, `right`, `bottom`, `left`, `z-index` would not work)
- **Relative**: Element's position adjusted from its normal position.
- **Absolute**: Positions element relative to the nearest positioned ancestor.
- **Fixed**: Element positioned relative to the `viewport`, does not move on scroll.

29. Position Property

```
div {  
    height: 70px;  
    width: 70px;  
    background-color: red;  
    border: 5px solid black;  
    margin: 20px;  
}  
  
#div1 {  
    position: static;  
}  
  
#div2 {  
    position: relative;  
    top: 20px;  
    left: 90px;  
}  
  
#div3 {  
    position: fixed;  
    top: 20px;  
    left: 90px;  
}  
  
#div4 {  
    position: absolute;  
    top: 200px;  
    left: 200px;  
}
```

```
<head>  
    <title>Position</title>  
    <link rel="stylesheet" href="../../css/level 5/positions.css">  
</head>  
<body>  
    <div id="div1">1 Static</div>  
    <div id="div2">2 Relative</div>  
    <div id="div3">3 Fixed</div>  
    <div id="div4">4 Absolute</div>  
</body>
```

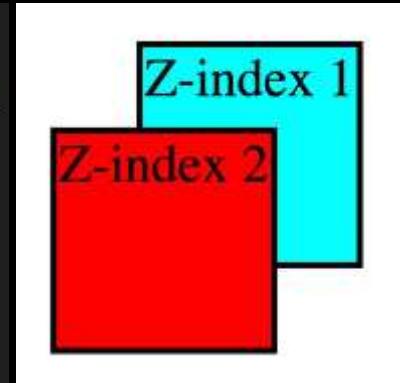


29. Position Property (z-index)

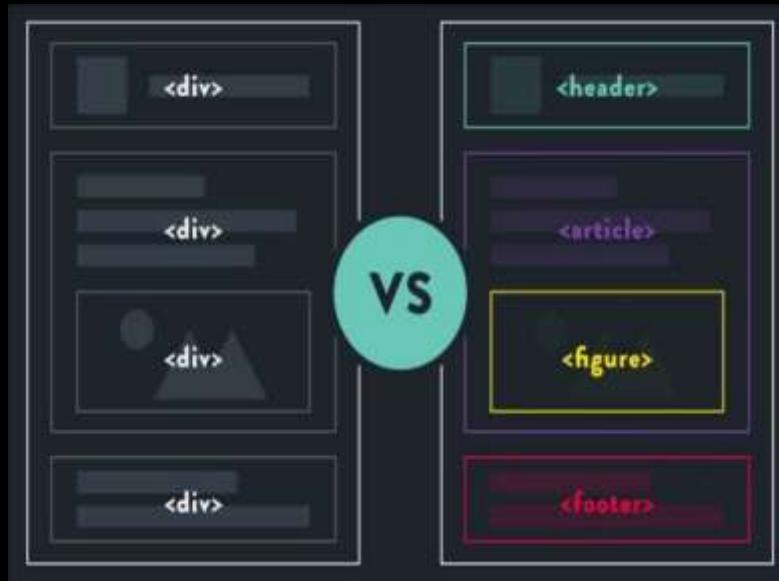
```
.container {  
    position: relative;  
}  
  
.box1, .box2 {  
    position: absolute;  
    border: 3px solid black;  
    width: 100px;  
    height: 100px;  
    text-align: center;  
    font-size: 25px;  
}  
  
.box1 {  
    background-color: red;  
    left: 20px;  
    top: 60px;  
    z-index: 2;  
}  
  
.box2 {  
    background-color: aqua;  
    left: 60px;  
    top: 20px;  
    z-index: 1;  
}
```

- **Stacking Order:** Determines the **stacking order** of elements along the Z-axis. **Position Context:** Only applies to elements with position set to relative, absolute, fixed, or sticky.
- **Integer Values:** Accepts **integer values**, including negative numbers.
- **Higher Values:** An element with a **higher z-index value appears above others.**

```
<head>  
    <title>Z-Index</title>  
    <link rel="stylesheet" href="../../css/  
        level 5/z-index.css">  
</head>  
<body>  
    <div class="container">  
        <div class="box1">Z-index 2</div>  
        <div class="box2">Z-index 1</div>  
    </div>  
</body>
```



30. Semantic Tags



30. Semantic Tags

Semantic Tags

- **Meaningful:** Describe content.
- **SEO:** Good for search engines.
- **Accessibility:** Useful for screen readers. Examples: `<header>`, `<footer>`, `<article>`, `<section>`, `<nav>`.

Non-Semantic Tags

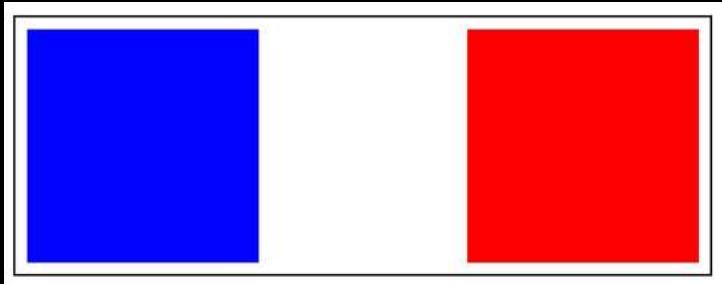
- **Generic:** No specific meaning.
- **For Styling:** Used for layout.
- **No SEO:** Not SEO-friendly.
- Examples: `<div>`, ``, `<i>`, ``.

31. Float Property

```
.container {  
    height: 110px;  
    width: 300px;  
    border: 1px solid #000;  
}  
  
.box {  
    width: 100px;  
    height: 100px;  
    margin: 5px;  
}  
  
.box1 {  
    background-color: red;  
    float: right;  
}  
  
.box2 {  
    background-color: blue;  
    float: left;  
}
```

- **Element Alignment:** Allows elements to be aligned to the left or right within their containing element. **Values:** Can take values like "left", "right", or "none" to determine the floating direction.
- **Old Layout Technique:** Less commonly used with the advent of Flexbox.
-

```
<head>  
    <title>Float Property</title>  
    <link rel="stylesheet" href="../../css/level-6/float.css">  
</head>  
<body>  
    <div class="container">  
        <div class="box box1"></div>  
        <div class="box box2"></div>  
    </div>  
</body>
```



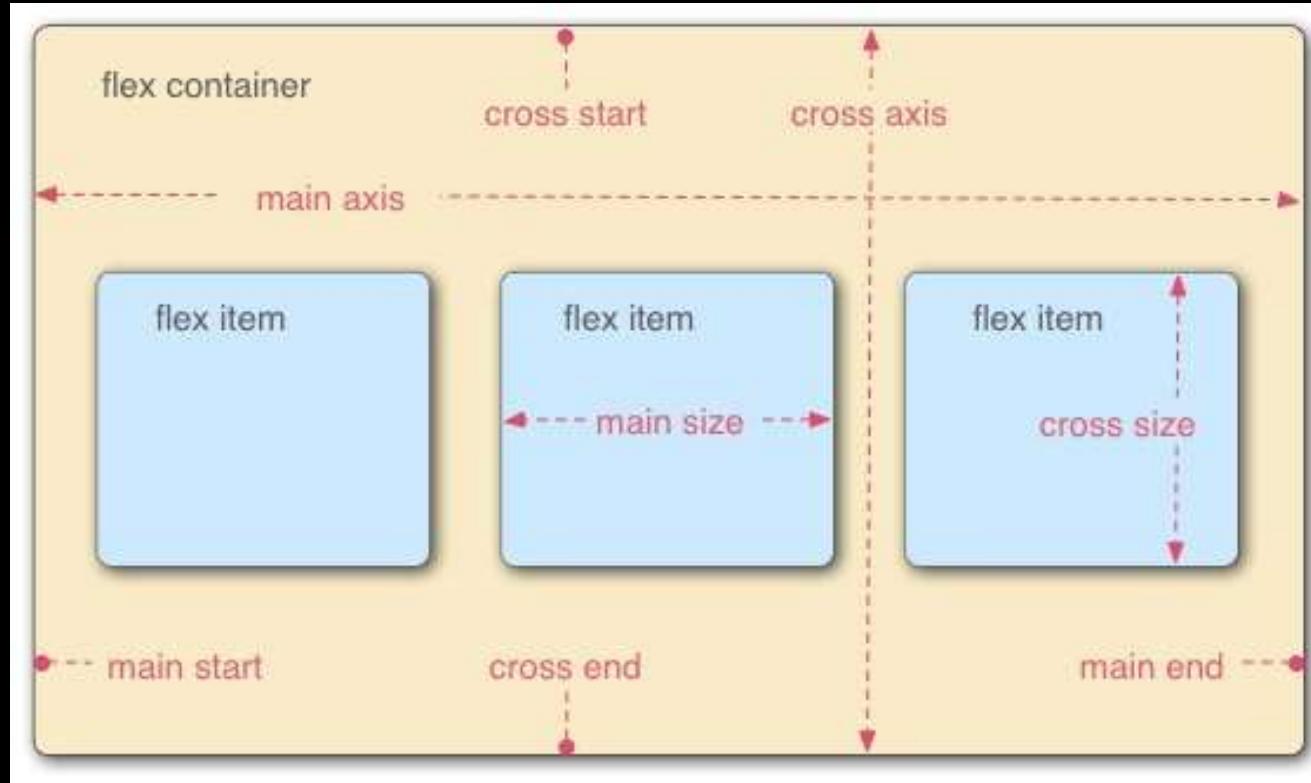
32.What is Flexbox?

Flexbox is a one-dimensional layout method for arranging items in rows or columns. Items *flex*(expand) to fill additional space or shrink to fit into smaller spaces.



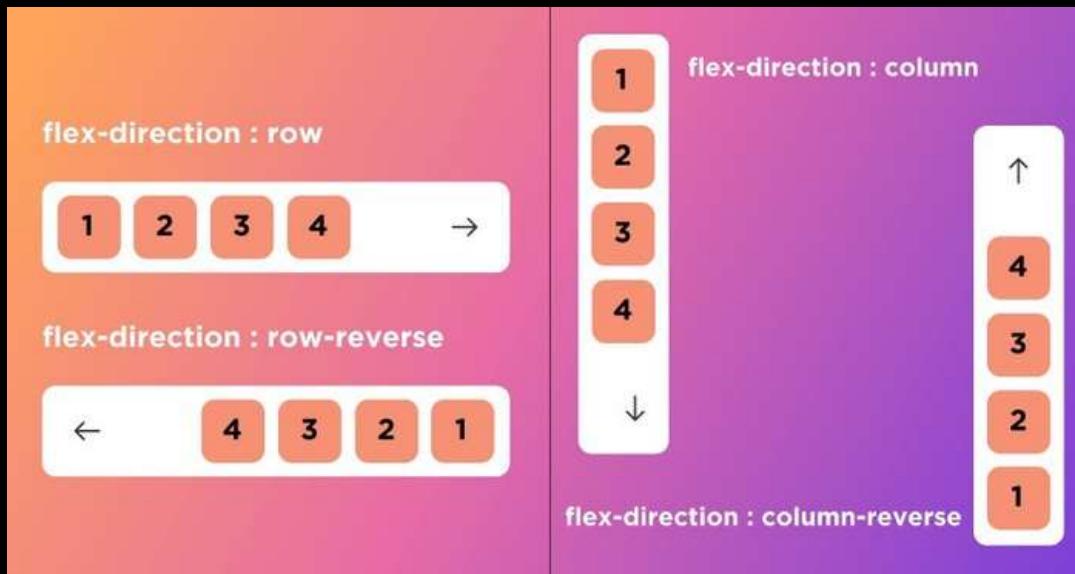
33. Flex Model

display: flex



34. Flexbox Direction

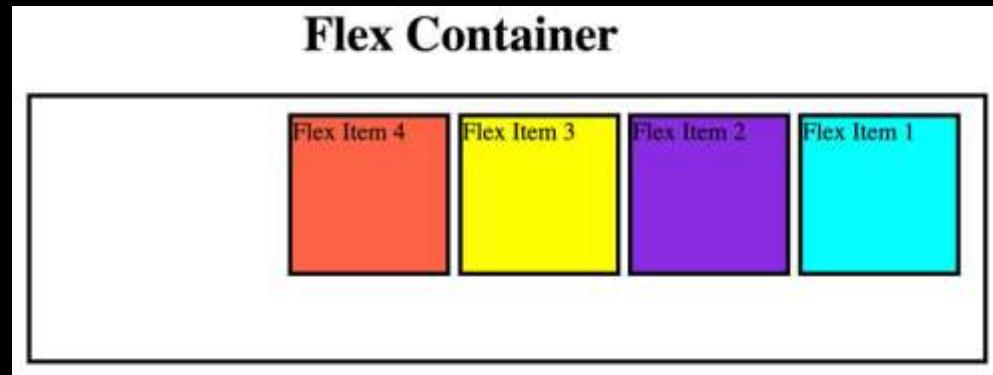
- **Property Name:** flex-direction is the property used to define the direction in a flex container. **Row**
- **Layout:** **row**value aligns the flex items **horizontally**, in a left-to-right fashion. **Column Layout:**
- **column**value stacks the flex items **vertically**, from top to bottom. **Reverse Direction:** Adding **-reverse** to
- **row or column** (as in **row-reverse** or **column-reverse**) reverses the order of the items.



34. Flexbox Direction

```
* {  
    margin: 0;  
    padding: 0;  
}  
  
.box {  
    height: 100px;  
    width: 100px;  
    border: 3px solid black;  
    margin-right: 5px;  
}  
  
#heading {margin-left: 200px;}  
  
#container {  
    height: 150px;  
    width: 600px;  
    padding: 10px;  
    margin: 20px;  
    border: 3px solid black;  
  
    display: flex;  
    flex-direction: row-reverse;  
}  
  
#box1 { background-color: aqua; }  
#box2 { background-color: blueviolet; }  
#box3 { background-color: yellow; }  
#box4 { background-color: tomato; }
```

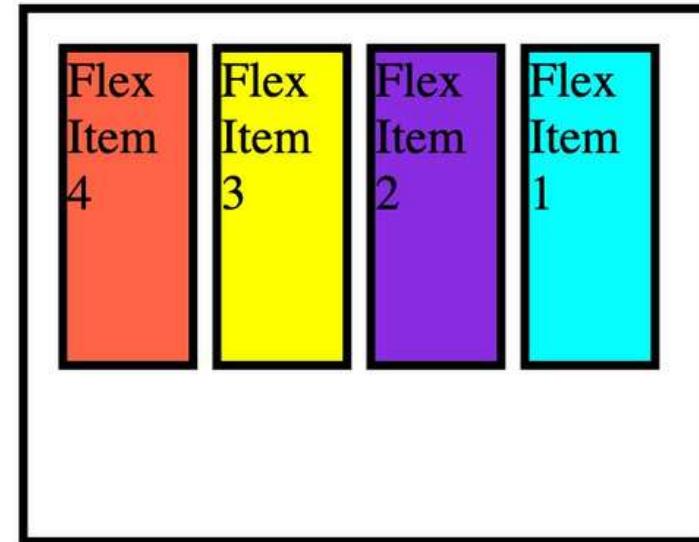
```
<body>  
    <h1 id="heading">Flex Container</h1>  
    <div id="container">  
        <div id="box1" class="box">Flex Item 1</div>  
        <div id="box2" class="box">Flex Item 2</div>  
        <div id="box3" class="box">Flex Item 3</div>  
        <div id="box4" class="box">Flex Item 4</div>  
    </div>  
</body>
```



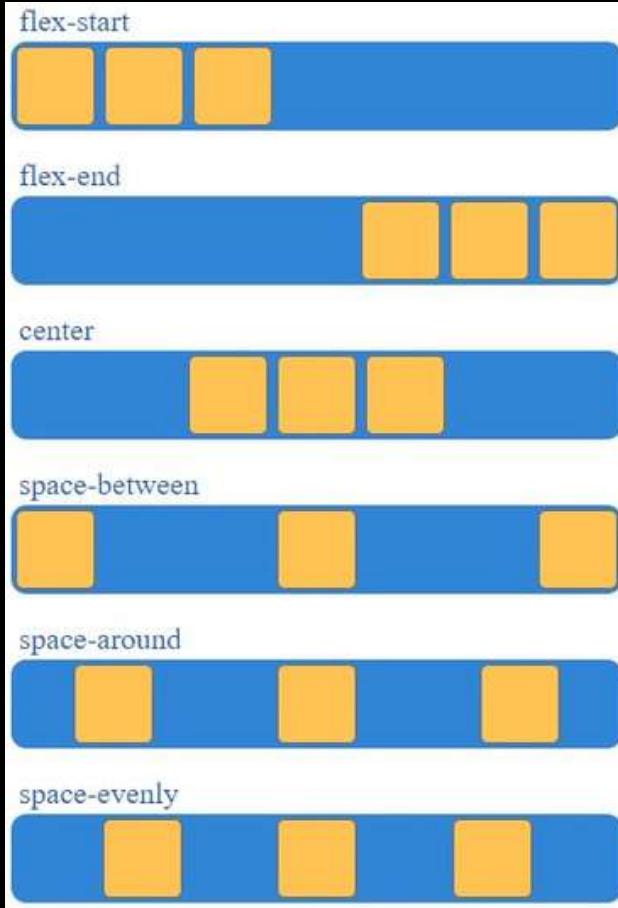
34. Flexbox Direction

```
.box {  
    height: 100px;  
    width: 100px;  
    border: 3px solid black;  
    margin-right: 5px;  
}  
  
#heading {margin-left: 50px;}  
  
#container {  
    height: 150px;  
    width: 200px; //  
    padding: 10px;  
    margin: 20px;  
    border: 3px solid black;  
  
    display: flex;  
    flex-direction: row-reverse;  
}
```

Flex Container



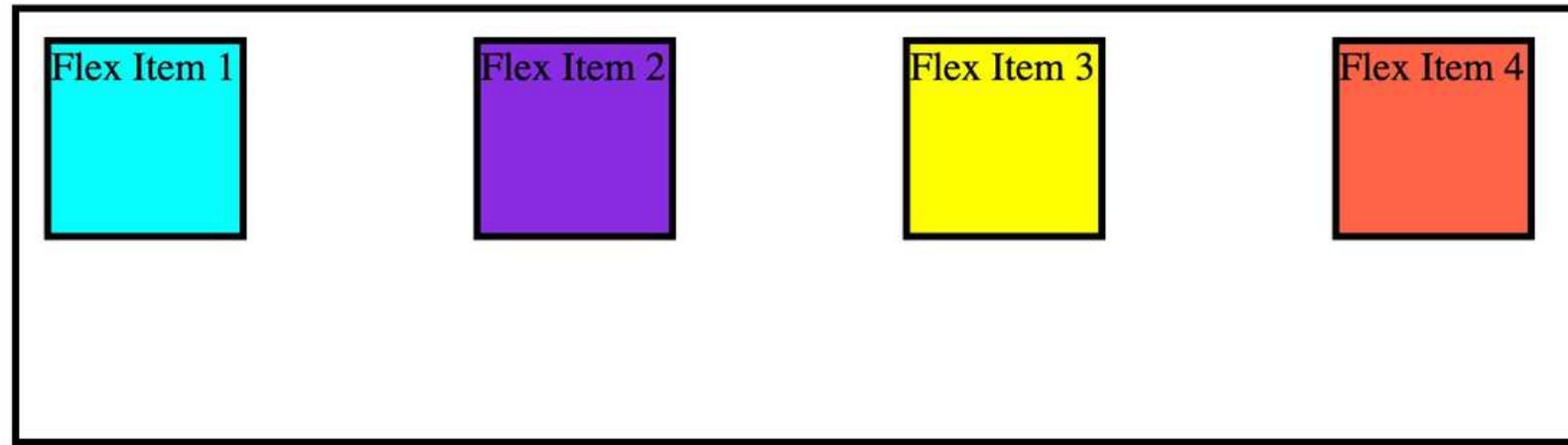
35. Properties: Flexbox container (Justify Content)



- **Alignment:** Aligns flex items along the main axis. **flex-**
- **start:** Items align to the start of the flex container.
- **flex-end:** Items align to the end of the flex container.
- **Center:** Items are centered within the flex container.
- **space-between/space-around/space-evenly:**
Distributes space between items evenly.

35. Properties: Flexbox container (Justify Content)

Flexbox Container

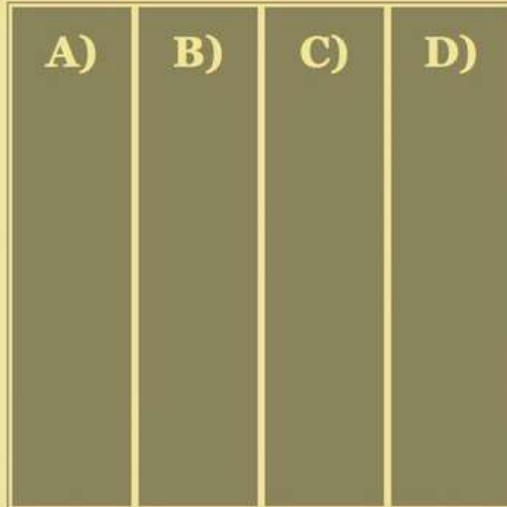


```
display: flex;  
justify-content: space-between;
```

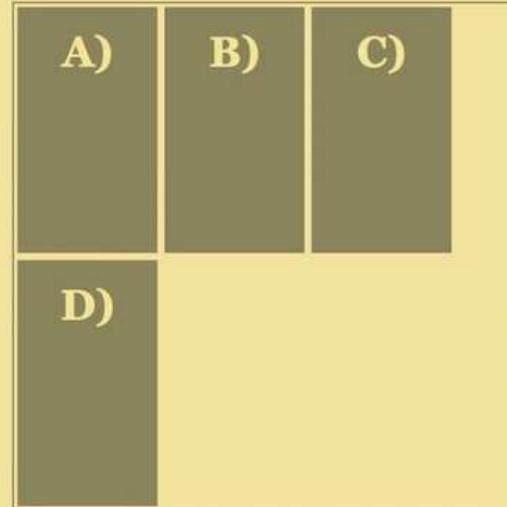
35. Properties: Flexbox container (Flex Wrap)

`flex-wrap:`

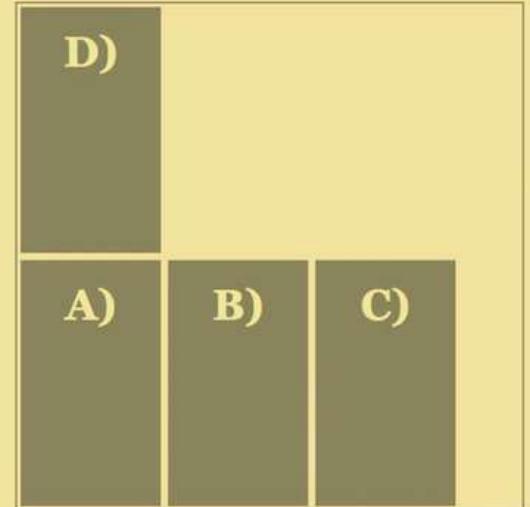
`nowrap` (*default*)



`wrap`



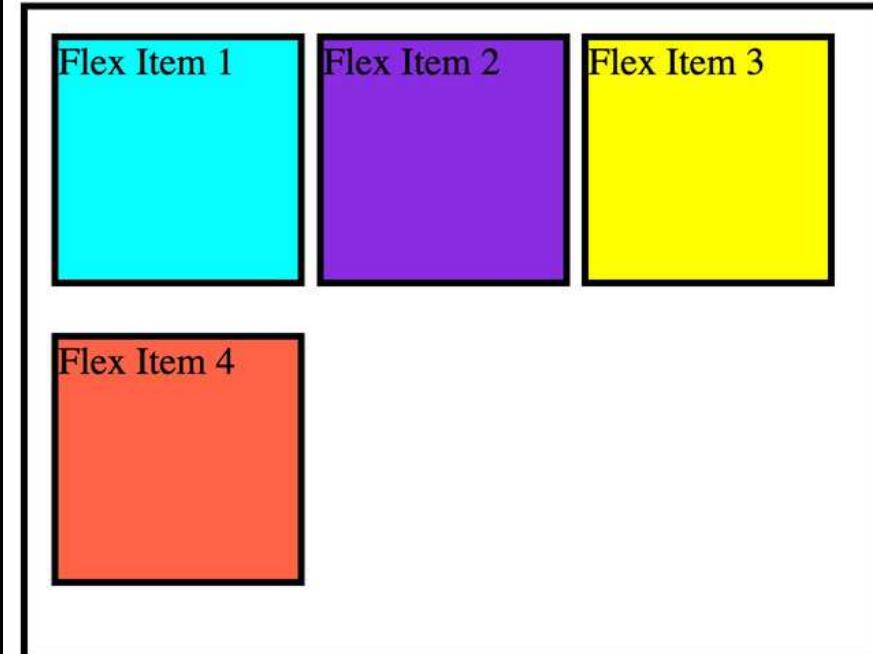
`wrap-reverse`



35. Properties: Flexbox container (Flex Wrap)

```
#container {  
    height: 250px;  
    width: 335px;  
    padding: 10px;  
    margin: 20px;  
    border: 3px solid black;  
  
    display: flex;  
    flex-wrap: wrap;  
}
```

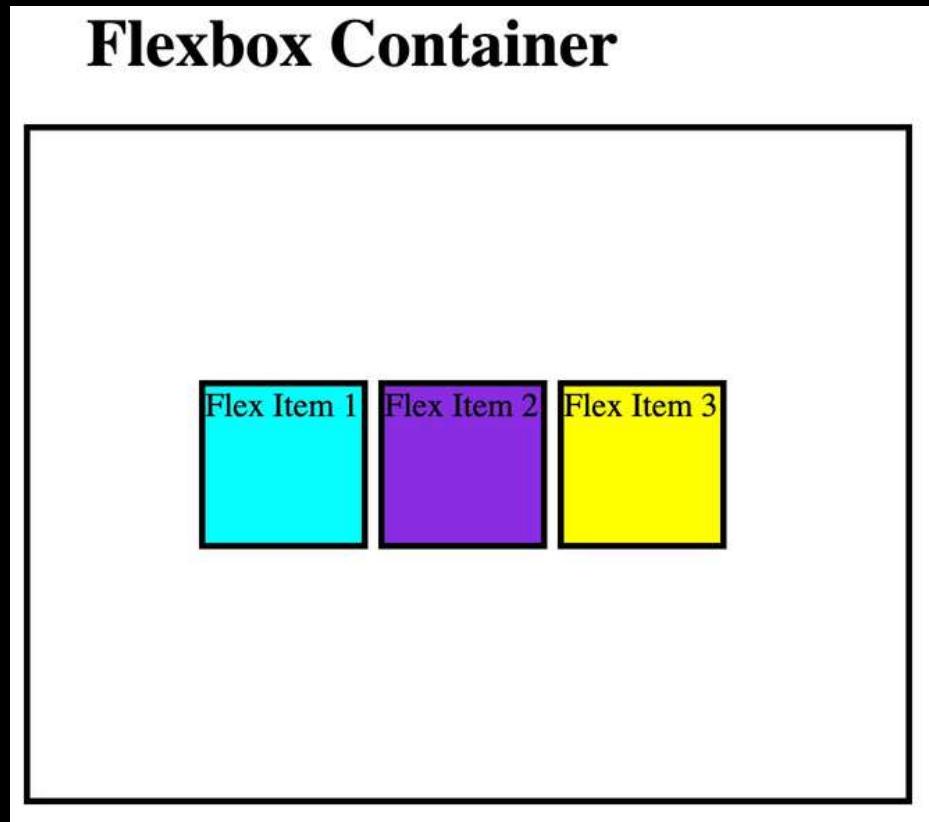
Flex Container



35. Properties: Flexbox container (Align Items)

This property is used to align the flex container's items along the cross-axis, which is perpendicular to the main axis.

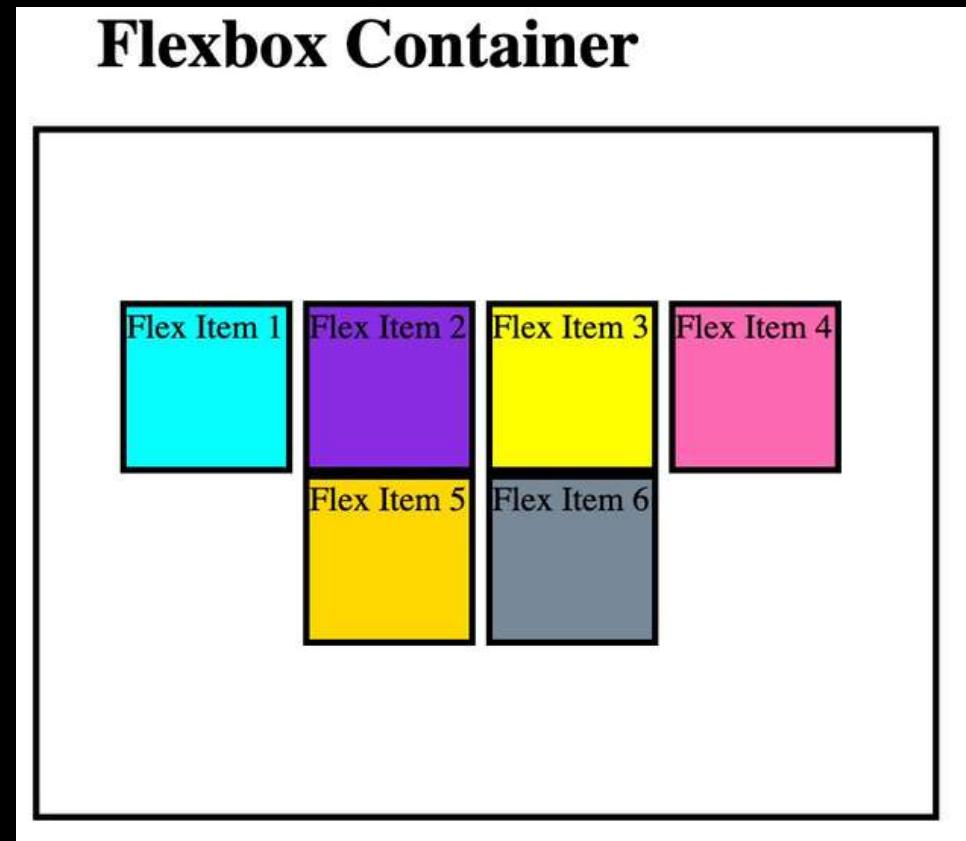
```
display: flex;  
flex-direction: row;  
justify-content: center;  
align-items: center;
```



35. Properties: Flexbox container (Align Content)

It is utilized to adjust the spacing between flex lines within a flex container, particularly when there is extra space along the cross-axis.

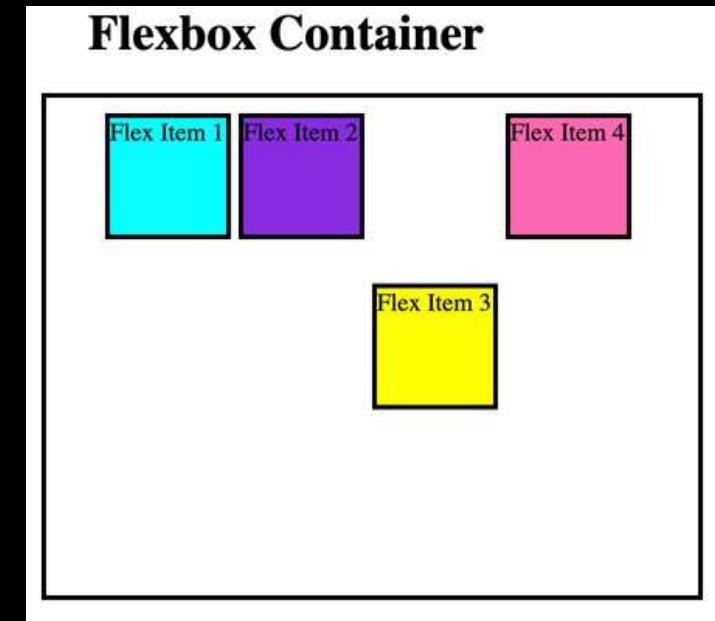
```
display: flex;  
flex-direction: row;  
justify-content: center;  
flex-wrap: wrap;  
align-content: center;
```



36. Properties: Flex Items (Align Self)

```
#container {  
    height: 300px;  
    width: 400px;  
    padding: 10px;  
    margin: 20px;  
    border: 3px solid black;  
  
    display: flex;  
    flex-direction: row;  
    justify-content: center;  
    flex-wrap: wrap;  
    align-items: start;  
}  
  
#box1 { background-color: aqua; }  
#box2 { background-color: blueviolet; }  
#box4 { background-color: hotpink; }  
#box3 {  
    background-color: yellow;  
    align-self: center;  
}
```

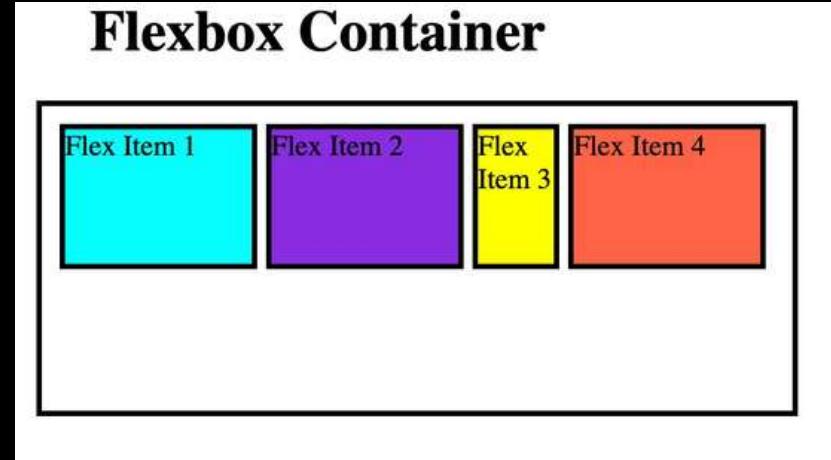
Allows individual flex items to **override the container's align-items property**, aligning them differently along the cross-axis.



36. Properties: Flex Items (Flex Shrink)

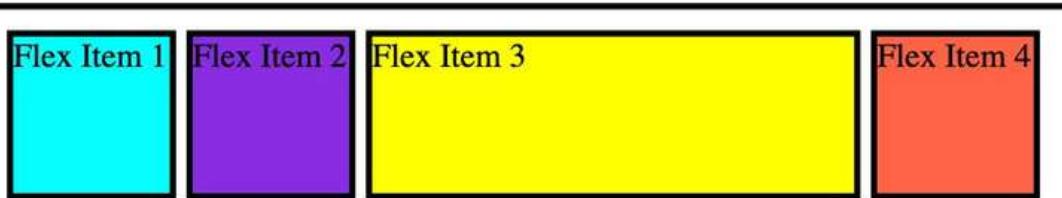
```
#box3 {  
    background-color: yellow;  
    flex-shrink: 4;  
}
```

The "flex-shrink" property in CSS determines how much a **flex item** will shrink relative to other items in the flex container if there is insufficient space.



36. Properties: Flex Items (Flex Grow)

Flexbox Container



```
#box3 {  
    background-color: yellow;  
    flex-grow: 1;  
}
```

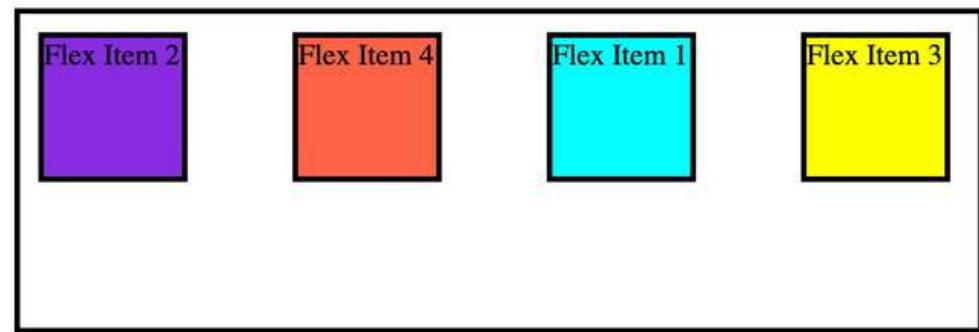
The "flex-grow" property in CSS specifies how much a **flex item** will grow **relative to other items** in the flex container when additional space is available.

36. Properties: Flex Items (Order)

```
#box1 {  
    background-color: #aqua;  
    order: 3  
}  
  
#box2 {  
    background-color: #blueviolet;  
    order: 1;  
}  
  
#box3 {  
    background-color: #yellow;  
    order: 4;  
}  
  
#box4 {  
    background-color: #tomato;  
    order: 2;  
}
```

The "order" property in CSS allows you to define the sequence in which flex items appear within the flex container, **overriding their original order** in the HTML.

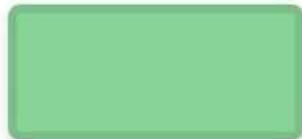
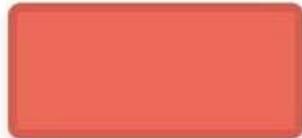
Flexbox Container



37. Grid

Flexbox

One-dimensional layout



Grid

Multi-dimensional layout



37. Grid

```
.container {  
    display: grid;  
    grid-template-columns: 50px 50px;  
    grid-template-rows: 50px 50px;  
}  
  
.item1 {  
    grid-column: 1 / 2;  
    grid-row: 1 / 2;  
    background-color: #lightblue;  
}  
  
.item2 {  
    grid-column: 2 / 2;  
    grid-row: 1 / 2;  
    background-color: #lightgreen;  
}  
  
.item3 {  
    grid-column: 1 / 2;  
    grid-row: 2 / 2;  
    background-color: #lightpink;  
}  
  
.item4 {  
    grid-column: 2 / 2;  
    grid-row: 2 / 2;  
    background-color: #lightyellow;  
}
```

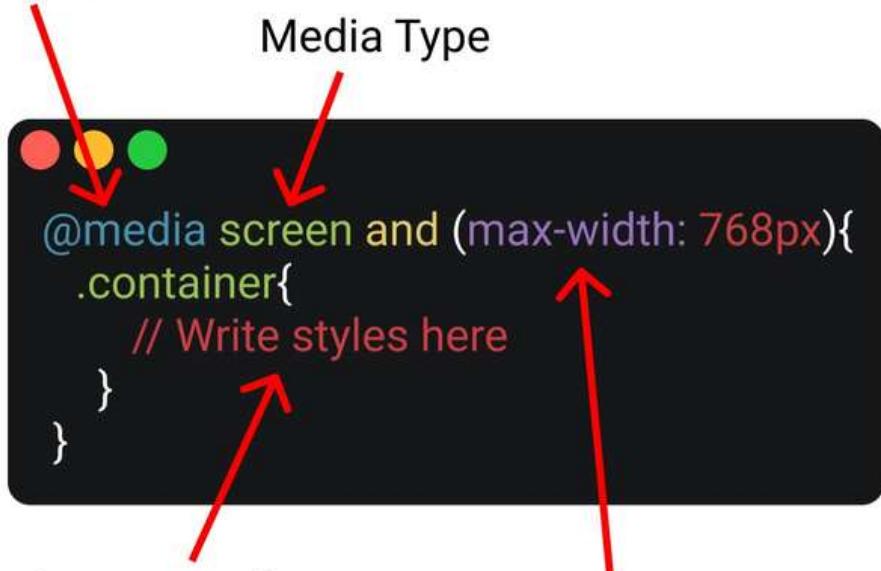
```
<div class="container">  
    <div class="item1">Item 1</div>  
    <div class="item2">Item 2</div>  
    <div class="item3">Item 3</div>  
    <div class="item4">Item 4</div>  
</div>
```



- 2D layout system for rows & columns. Activate with `display: grid;`. Children become grid items.
- Define structure with `grid-template` properties.
- Individual units called grid cells.
-

38. Media Queries

declaration



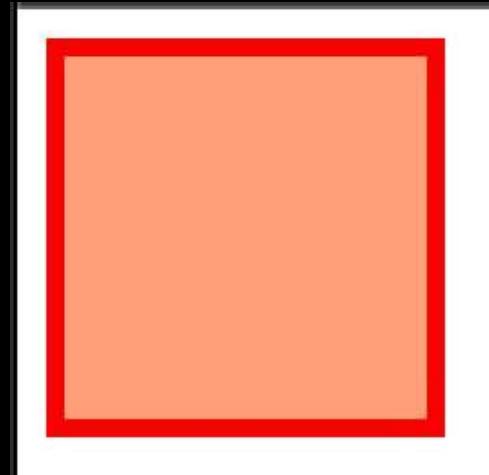
styles to apply
when all conditions
are met

Specifying amount
of screen to cover

- Tailor styles for specific device characteristics. Use to create responsive web designs. Apply styles based on conditions like screen size.
- Syntax: `@media (condition) { CSS rules }`.
- Can combine multiple conditions using and, or.

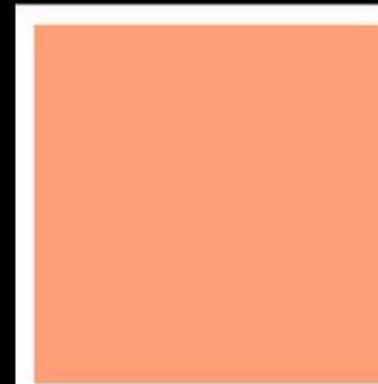
38. MediaQueries (width)

```
.box {  
    height: 100px;  
    width: 100px;  
    background-color: #lightsalmon;  
}  
  
@media screen and (width: 250px) {  
    .box {  
        border: 5px solid #red;  
    }  
}
```



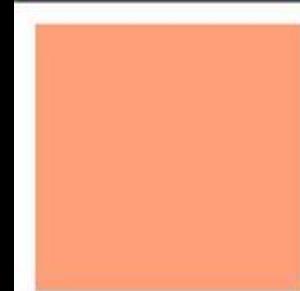
38. MediaQueries (min-width)

```
.box {  
    height: 100px;  
    width: 100px;  
    background-color: lightsalmon;  
}  
  
@media screen and (min-width: 300px) {  
    .box {  
        height: 150px;  
        width: 150px;  
    }  
}
```



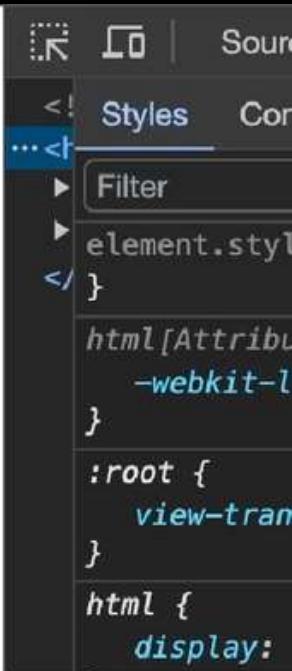
38. MediaQueries (max-width)

```
.box {  
    height: 100px;  
    width: 100px;  
    background-color: lightsalmon;  
}  
  
@media screen and (max-width: 250px) {  
    .box {  
        height: 50px;  
        width: 50px;  
    }  
}
```



38. MediaQueries (combination)

```
.box {  
    height: 100px;  
    width: 100px;  
    background-color: lightsalmon;  
}  
  
@media screen and (min-width: 250px)  
and (max-width: 300px) {  
    .box {  
        border-radius: 50%;  
    }  
}
```



The screenshot shows a browser's developer tools with the "Styles" tab selected in the sidebar. The main area displays a single rule under "element.style":

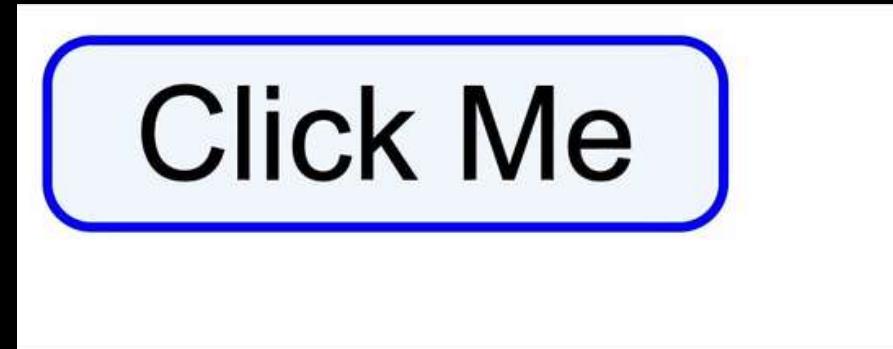
```
<! Styles Com...>  
...<!-->  
  > Filter  
  > element.style  
</-->
```

Below this, there are other collapsed sections for "html [Attributes]" and "css [Properties]". The right side of the interface shows the standard tree view of the DOM structure.

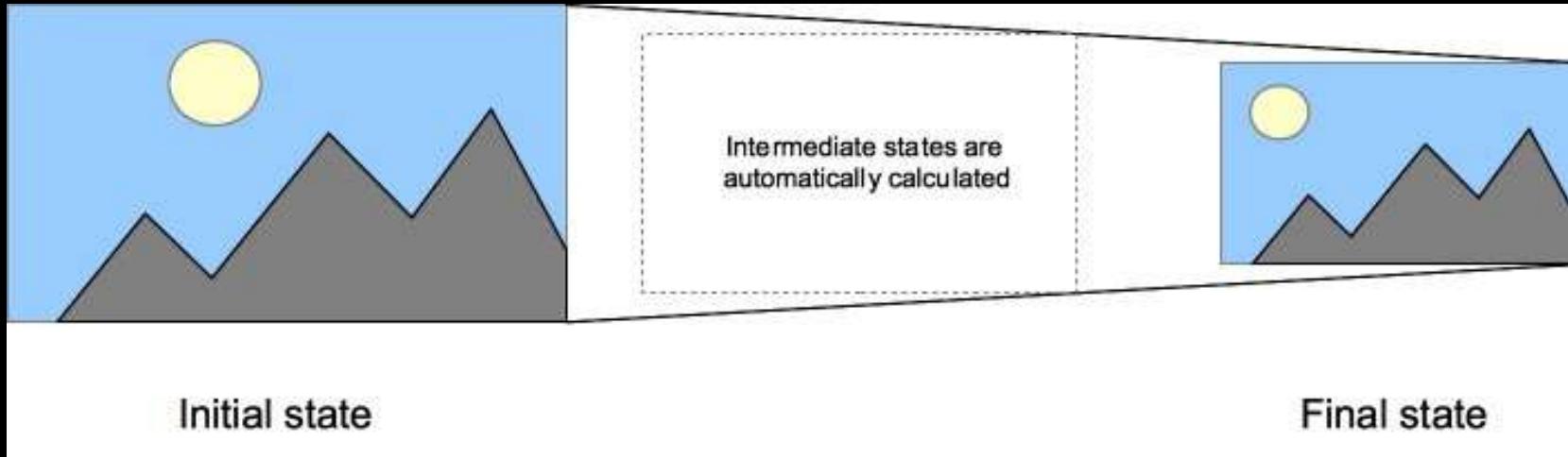
39. Pseudo Classes

```
.btn {  
    height: 20px;  
    width: 70px;  
    border: 1px solid blue;  
    border-radius: 5px;  
    background-color: aliceblue;  
}  
  
.btn:hover {  
    height: 25px;  
    width: 80px;  
    border: 1px solid red;  
}  
  
.btn:active {  
    height: 25px;  
    width: 80px;  
    border: 1px solid red;  
    background-color: indianred;  
}
```

- Used to define **special states** of HTML elements.
- Syntax: `selector:pseudo-class{ styles }`. Common
- examples: `:hover`, `:active`, `:first-child`. Target
- elements based on their position or user action.



40. Transitions

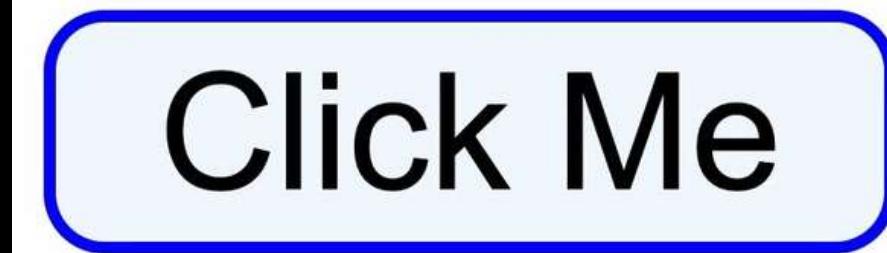


CSS transition is a property that enables smooth animation between changes in CSS property values

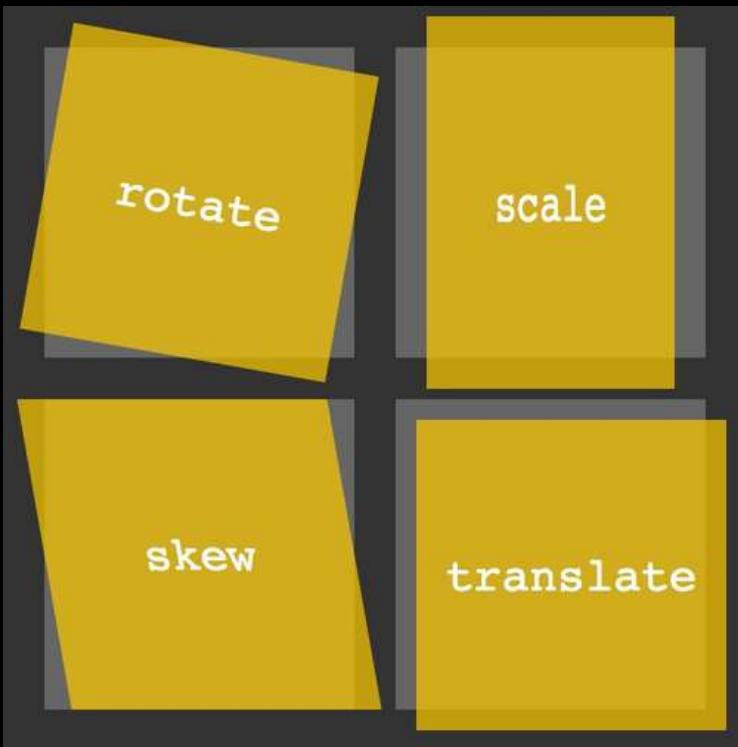
- **transition-property:** Defines which CSS properties will transition.
- **transition-duration:** Sets how long the transition lasts. **transition-**
- **timing-function:** Controls the speed curve of the transition.
- **transition-delay:** Specifies a delay before the transition starts.

40. Transitions

```
.btn {  
    height: 20px;  
    width: 70px;  
    border: 1px solid blue;  
    border-radius: 5px;  
    background-color: aliceblue;  
    transition-property: all;  
    transition-duration: 1s;  
    transition-timing-function: ease-in-out;  
    transition-delay: 1s;  
    /*transition: all 1s ease-in-out 1s;*/  
}  
  
.btn:hover {  
    height: 25px;  
    width: 80px;  
    border: 1px solid red;  
}  
  
.btn:active {  
    height: 25px;  
    width: 80px;  
    border: 1px solid red;  
    background-color: indianred;  
}
```



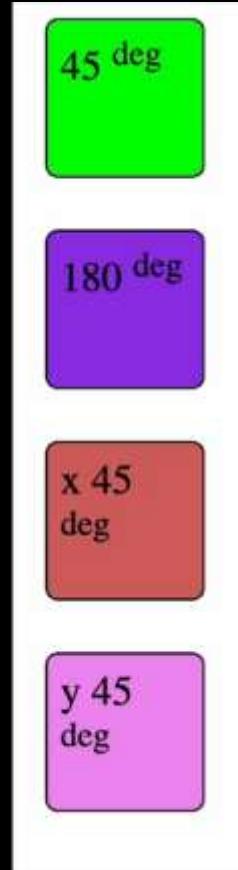
41. CSS Transform



- Allows modification of an element's shape and position. Can perform operations like
- **rotate, scale, and translate**. Does not affect the layout of surrounding elements.
- Used to create visual effects like **3Dspace** transformations.
- Implemented with functions like **rotate()**, **scale()**, and **translate()**.
-

41. CSS Transform (Rotate)

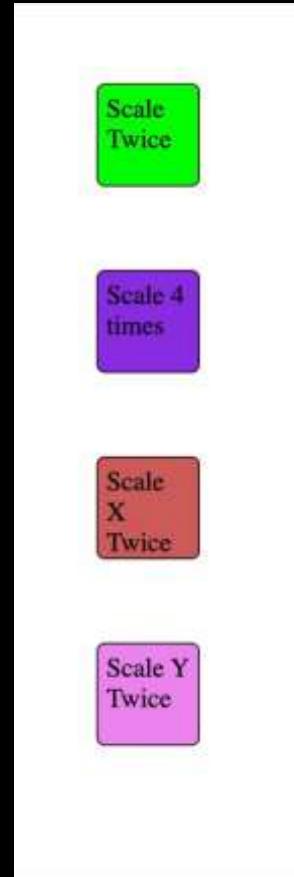
```
.box {  
    height: 50px;  
    width: 50px;  
    padding: 5px;  
    margin: 20px;  
    border: 1px solid black;  
    border-radius: 5px;  
    transition-property: all;  
    transition-duration: 1s;  
    transition-timing-function: ease-in-out;  
}  
  
#box1 { background-color: lime; }  
#box1:hover { transform: rotate(45deg); }  
  
#box2 { background-color: blueviolet; }  
#box2:hover { transform: rotate(180deg); }  
  
#box3 { background-color: indianred; }  
#box3:hover { transform: rotatex(45deg); }  
  
#box4 { background-color: violet; }  
#box4:hover { transform: rotatey(45deg); }
```



- Rotates an element around a fixed point. Defined using the
- `rotate()` function within the `transform` property. Default rotation point is the element's center.

41. CSS Transform (Scale)

```
.box {  
    height: 50px;  
    width: 50px;  
    padding: 5px;  
    margin: 50px;  
    border: 1px solid black;  
    border-radius: 5px;  
    transition-property: all;  
    transition-duration: 1s;  
    transition-timing-function: ease-in-out;  
}  
  
#box1 { background-color: lime; }  
#box1:hover { transform: scale(2); }  
  
#box2 { background-color: blueviolet; }  
#box2:hover { transform: scale(4); }  
  
#box3 { background-color: indianred; }  
#box3:hover { transform: scalex(2); }  
  
#box4 { background-color: violet; }  
#box4:hover { transform: scaley(3); }
```



41. CSS Transform (Translate)

```
.box {  
    height: 75px;  
    width: 75px;  
    padding: 5px;  
    margin: 50px;  
    border: 1px solid black;  
    border-radius: 5px;  
    transition-property: all;  
    transition-duration: 1s;  
    transition-timing-function: ease-in-out;  
}  
  
#box1 { background-color: lime; }  
#box1:hover { transform: translate(50px); }  
  
#box2 { background-color: blueviolet; }  
#box2:hover { transform: translate(50px, 50px); }  
  
#box3 { background-color: indianred; }  
#box3:hover { transform: translatex(50px); }  
  
#box4 { background-color: violet; }  
#box4:hover { transform: translatey(50px); }
```



41. CSS Transform (Skew)

```
.box {  
    display: inline-block;  
    height: 75px;  
    width: 75px;  
    padding: 5px;  
    margin: 50px;  
    border: 1px solid black;  
    border-radius: 5px;  
    transition-property: all;  
    transition-duration: 1s;  
    transition-timing-function: ease-in-out;  
}  
  
#box1 { background-color: lime; }  
#box1:hover { transform: skew(45deg); }  
  
#box2 { background-color: blueviolet; }  
#box2:hover { transform: skew(90deg); }
```



42. Animation

The keyframes at rule rule
start with "@keyframes"
keyword

animation name which is
specified in animation-name
property



```
@keyframes animation-name {  
    from { /* CSS code */ }  
    to { /* CSS code */ }  
}
```

specify where
animation
should end.
You can write
100% as well
instead of "to"

Specify when the
style change will
happen.
You can write 0%
as well, which is
same as "from"

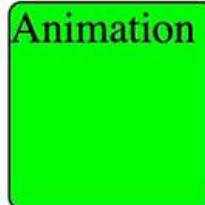
CSS properties.

42. Animation Properties

- **animation-name:** Specifies the name of the @keyframes defined animation. **animation-duration:**
- Defines the total time the animation takes to complete one cycle. **animation-timing-function:** Controls the pacing of the animation (e.g., linear, ease-in).
- **animation-delay:** Sets a delay before the animation starts, allowing for a pause before initiation. **animation-iteration-count:** Indicates the number of times the animation should repeat. **animation-direction:** Specifies the direction of the animation, allowing for reverse or alternate cycles.

42. Animation

```
.box {  
    height: 75px;  
    width: 75px;  
    border: 1px solid black;  
    border-radius: 5px;  
    position: absolute;  
    left: 10;  
    background-color: lime;  
  
    animation-name: ghumakkad;  
    animation-duration: 4s;  
    animation-timing-function: ease-in-out;  
    animation-delay: 0s;  
    animation-iteration-count: 4;  
  
    animation-direction: alternate;  
  
    /* animation: ghumakkad 4s ease-in-out 0s 4  
    alternate; */  
}  
  
@keyframes ghumakkad {  
    from {left: 10px}  
    to {left: 300px}  
}
```



42. Animation

Animation

```
@keyframes ghumakkad {  
    0% {left: 10px; top: 0px}  
    50% { left: 150px; top: 100px }  
    100% {left: 300px; top: 0px}  
}
```

Project MyntraClone