CoSc3081
Web Programming

Instructor: Zinabu H.

zinabuscholar@gmail.com
zinabu@aku.edu.et

2024

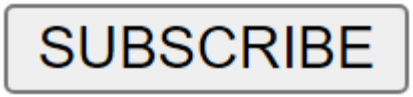
Chapter 3

Cascading Style Sheet

- CSS Basics
- Style Sheet Rules
- Style Sheet Properties
- CSS Measuring Unit

Introduction to CSS

- CSS (Cascading Style Sheets) is a styling language used to add style to a webpage
- HTML provides structure and adds content to a webpage, while CSS enhances the visual presentation of that content through various styles
- Example Page



HTML

CSS

CSS Syntax

- The syntax to style an element on a webpage

```
selector {
    property1: value;
    property2: value;
}
```

- The basic syntax of CSS includes 3 main parts:
  - **selector** - specifies the HTML element that we want to apply the styles
  - **property1** / property2- specifies the attribute of HTML elements that we want to change (color, background, and so on)
  - **value** - specifies the new value you want to assign to the property (color of the text to red, background to gray, and so on)

Attaching CSS with HTML

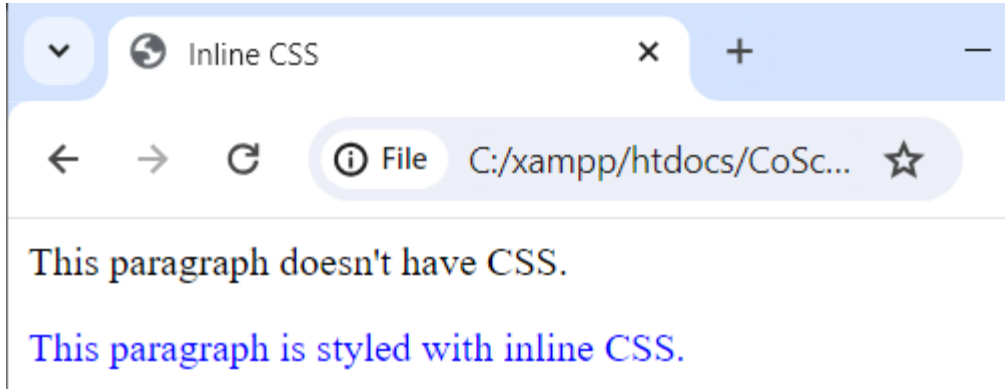
- CSS can be added to HTML documents in 3 ways:
  - **Inline** - by using the style attribute inside HTML elements
  - **Internal** - Styles defined at the head section of the document.
  - **External** - Styles defined in a separate file.
- The most common way to add CSS, is to keep the styles in external CSS files.

Attaching CSS with HTML: inline

- Inline style is the approach of adding CSS rules directly to the HTML element using the style attribute. For example,

```
<p>This paragraph doesn't have CSS.</p>
<p style="color:blue">This paragraph is styled with inline CSS.</p>
```

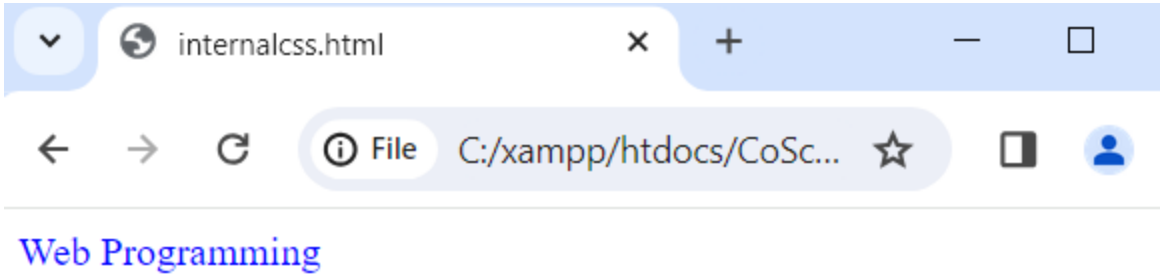
  - style - defines the CSS for the element <p>
  - color: blue- changes the text of the <p> element to the color blue



Attaching CSS with HTML: internal

- Internal CSS applies CSS styles to a specific HTML document. Internal CSS is defined inside an HTML document using <style> attribute within the head tag of an HTML

```
<head>
  <style>
    p {
        color: blue;
    }
  </style>
</head>
```



Attaching CSS with HTML: external

- External CSS is an approach to applying CSS styles to HTML pages by defining the CSS in a separate file

```
p {
        color: blue;
  }
```

- Here, we have CSS in a separate file named **style.css**. The external CSS file should have a .css extension.
- The external CSS file can be linked to the HTML document by using the **link** element in the HTML.

Attaching CSS with HTML: external…

- The external CSS file can be linked to the HTML document by using the **link** element in the HTML.

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

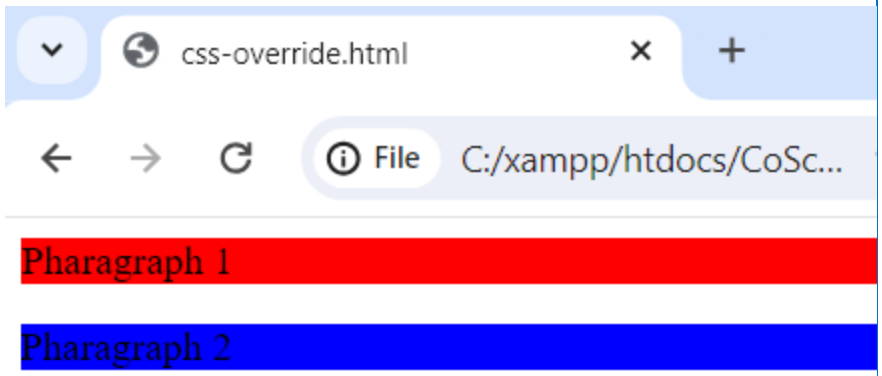
- We use the <link> tag to link the **style.css** file to an HTML document. In the above code,
  - **href="style.css"** - URL or file path to the external CSS file.
  - **rel="stylesheet"** - indicates the linked document is a CSS file

Attaching CSS with HTML

Note

- Inline Style Override Internal Style
- If an internal CSS and inline CSS are both applied to a tag, the styling from the inline tag is applied. Let's see an example.

```
<head>
    <style>
        p {
            background: red;
        }
    </style>
</head>
<body>
<p>Paragraph 1</P>
 <p style="background: blue">paragraph 2</p>
</body>
```



Style Sheet Rules :

CSS Essential Concepts

- CSS is the language for styling web pages, and it has a few essential concepts that are important to understand.
- These concepts includes
  - Inheritance
  - Rule Order
  - Style Rule Hierarchy
  - Specificity
  - Box Model

Style Sheet Rules : Inheritance

- In CSS, inheritance passes the styles directly from the parent element to its child elements.
- The child elements normally take the same styles that have been assigned to the parent, unless they are provided their own styles.
- This mechanism ensures a consistent design across a webpage.

Example on Board

Style Sheet Rules : Rule Order

- Rule order refers to the sequence in which CSS rules are applied to the HTML elements.
- The order of CSS rules determine the priority for the styles. The later rules overrides the earlier ones and ensures that the most recent styles are applied.

```
p {
    color: red;
}
/* overrides color previous color value */
p {
    color: blue;
}
```

- Here, the paragraph color is blue because it is declared last in the stylesheet, overriding all previous values.

Style Sheet Rules : Style Rule Hierarchy

- The style rule hierarchy determines the priority of CSS rules when multiple rules target the same element.
- In CSS, the following hierarchy of style priorities applies:
  - Inline styling: Styles applied directly within HTML element.
  - ID selectors: Styles elements with specific ID.
  - Class and attribute selectors: Styles elements with certain class or attribute.
  - Element selectors: Styles elements with specific tag name.

Style Sheet Rules : Specificity

- Specificity in CSS determines which style rules take precedence when multiple rules target the same element
- This helps the browser prioritize and apply the most relevant styles.

```
div p {
    color: blue;
}

p {
    color: orange;
}
```

```
<div>
    <p>paragraph</p>
    </div>
```

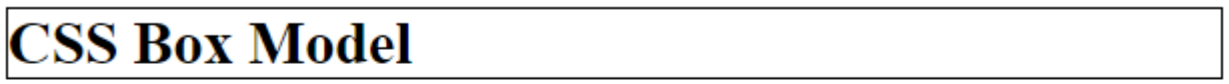
- In the above example, the element selector p sets the color of all p elements to orange.
- However, the selector div p is more specific for selecting paragraph so the color of paragraph is colored blue.

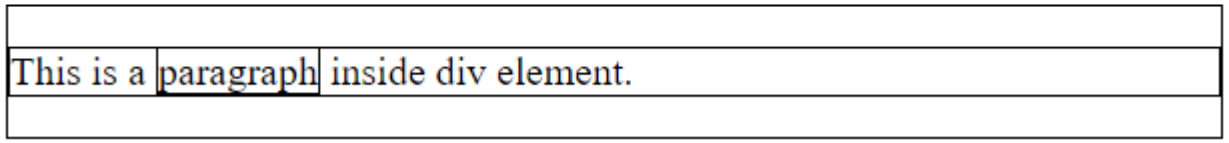
Style Sheet Rules : Box Model

- The box model specifies that every element in HTML is represented as a rectangular box. It helps to understand how elements are structured and interact with each other on a webpage.

```
h2, div, p, span {
    border: 1px
solid;
}
```

```
<body>
      <h2>CSS Box Model</h2>
      <div>
         <p>This is a
          <span>paragraph</span>
            inside div element.
        </p>
      </div>
   </body>
```





Style Properties: Font and Text

- CSS font properties are used to adjust the appearance of the text in an HTML document. Using the CSS fonts properties, we can customize the font **family**, **size**, **weight**, **style**, and **color** of text.

- font-family: defines the font applied to the text
- font-size: sets the size of the font

```
body {
    font-family: Helvetica;
    font-size: 16px;
}
```

- font-weight: sets the thickness i.e increase the boldness or lightness of the font
- font-style: sets the font to italic or oblique
- font-variant: changes the font to small-caps
- font-stretch: expands or narrows the text
- line-height: sets the distance between lines of the text

Foreground and Background Properties

- **CSS background** property is used to add a color or an image to the background of an element
- There are various CSS background properties that can be specified into a single background property
  - **background-image**: allows to add an image as a background of an element
  - **background-position**: specifies the position of the background image within the element
  - **background-size**: specifies the size of the background image
  - **background-repeat**: controls the repeating behavior of a background image

Foreground and Background Properties

- There are various CSS background properties that can be specified into a single background property
  - **background-attachment**: controls whether the background image scrolls or remains fixed with the page's content
  - **background-origin**: specifies the starting position of the background area within the element
  - **background-clip**: defines the background area for an element to clip
  - **background-color**: sets the background color in an element

Table Styling Properties

- **border** :specifies borders in the table
- **border-spacing** specifies the space between the borders of the adjacent cells
- Hover : Use the :**hover** selector on tr to highlight table rows on mouse over
- Table - Zebra Stripes
- Making a table responsive:
  - tables are not responsive in nature. However, to support mobile devices you can add responsiveness to your tables by enabling horizontal scrolling on small screens.
  - To do this simply wrap your table with a element and apply the style **overflow-x: auto;**

Styling List

Demo in Lab Session

- Creating Navigation bars
- Approach
  - First, create a **<nav>** element with **<ul>** and **<li>** for navigation links.
  - Use CSS flex for a horizontal layout, sticky positioning, and background styling.
  - Apply styling for text color, spacing, and alignment.
  - Add hover effects to enhance interactivity, like changing link colors.
  - Include optional elements (e.g., search bar) and style for consistency.
  - Consider media queries for responsiveness, and adjusting styles for different screen sizes

Layout and Positioning Properties

- In CSS, layout and positioning properties are used to control the arrangement and positioning of elements within a web page.
- Here are some of the key layout and positioning properties:
  - **display**: Determines the type of box used for an element. Common values include:
    - **block**: The element generates a block-level box.
    - **inline**: The element generates an inline-level box.
    - **inline-block**: The element generates an inline-level block container.
    - **flex**: The element is a flex container.
    - **grid**: The element is a grid container.

Layout and Positioning Properties…

- **float**: Specifies whether an element should be placed to the left or right of its container and allows content to flow around it
- **position**: Specifies the positioning method used for an element. Common values include:
  - **static**: The element is positioned according to the normal flow of the document.
  - **relative**: The element is positioned relative to its normal position.
  - **absolute**: The element is positioned relative to its nearest positioned ancestor.
  - **fixed**: The element is positioned relative to the browser window.
  - **sticky**: The element is positioned based on the user's scroll position.

Layout and Positioning Properties…

Box Model Properties:

- **width** and **height**: Sets the width and height of an element's content area.
- **margin**: Sets the margin area around an element.
- **padding**: Sets the padding area around an element's content.
- **border**: Sets the border properties of an element.

Layout and Positioning Properties…

Flexbox Properties:

- **flex-direction**: Defines the direction of the flex container's main axis.
- **justify-content**: Aligns flex items along the main axis of the flex container.
- **align-items**: Aligns flex items along the cross axis of the flex container.
- **flex-wrap**: Specifies whether flex items are forced onto a single line or can wrap onto multiple lines.

Layout and Positioning Properties…

Grid Properties:

- **grid-template-columns** and **grid-template-rows**: Defines the columns and rows of the grid.
- **grid-column** and **grid-row**: Specifies the grid lines that an item will span.
- **justify-items** and **align-items**: Aligns grid items inside their grid areas

Layout and Positioning Properties…

Positioning Properties:

- **top, right, bottom, left:** Defines the offset from the edges of the containing element for absolutely positioned elements.
- **z-index:** Specifies the stack order of positioned elements.

CSS Measuring Units

- CSS provides various units of measurement that allow you to specify lengths and sizes for various properties like width, height, margin, padding, font size, etc.
- Here are some of the most commonly used CSS units:
  - **Absolute Length Units:**
    - **px** (pixels): A single dot on a computer screen. It's a fixed-size unit and is not relative to any other property.
  - **Relative Length Units**
    - **em**: Equal to the computed value of the font-size property of the element. For example, if the font-size of an element is 16px, 1em is equal to 16px.

CSS Measuring Units…

- **Relative Length Units**
  - **rem**: Similar to em, but it's relative to the font-size of the root element (html). This makes it more predictable in cases where nested elements have different font-size values.
  - % (percentage): A percentage of the parent element's value. For example, width: 50% means the width will be half of its parent's width.
  - **vw** (viewport width): 1/100th of the viewport's width. 100vw is equal to the width of the viewport.
  - **vh** (viewport height): 1/100th of the viewport's height. 100vh is equal to the height of the viewport.

End of Chapter 3

Aksum University- AIT

2024