Module 3 – Frontend – CSS and CSS3

CSS Styler and selector

1. What is a CSS selector? Provide examples of element, class, and ID selectors.

- A CSS selector is a pattern used to select and style HTML elements. It tells the browser which elements on a web page the CSS rules should apply to.
- 1.Element Selector:-
  - ➢ Selects all HTML elements of a specific type.
  - ➢ Example:-
    ```
    p {
        color: blue;
    }
    ```
- 2.class selector:-
  - ➢ Selects all elements with a specific class attribute.
  - ➢ Denoted with a dot . followed by the class name.
  - ➢ HTML:-
    ```
    <div class="highlight">Text here</div>
    <p class="highlight">Another highlighted text</p>
    ```
  - ➢ CSS:-
    ```
    .highlight {
      background-color: yellow;
    }
    ```
- 3.ID selector:-
  - ➢ Selects a single element with a specific id attribute.

- Denoted with a hash # followed by the ID name.
- IDs should be unique on a page.
- HTML

  ```
  <h1 id="main-title">Welcome!</h1>
  ```

- CSS

  ```
  #main-title {
    font-size: 36px;
  }
  ```

2. Explain the concept of CSS specificity. How do conflicts between multiple styles get resolved?

- CSS specificity is a set of rules that browsers use to determine which CSS rule applies when multiple rules target the same HTML element. When conflicts arise, the rule with the highest specificity takes precedence.
- **Conflict resolution:-**
  - Inline styles win (specificity: 1000).
  - Higher specificity selectors override lower ones.
  - If specificity is equal, the last defined rule wins (source order).
  - !important can override everything except another !important with higher specificity.

3. What is the difference between internal, external, and inline CSS? Discuss the advantages and disadvantages of each approach.

- **<u>Inline css:-</u>**

  Advantages:-

  - ➢ Quick and easy for small changes or testing.
  - ➢ Overrides external and internal styles due to higher specificity.

  Disadvantages:-

  - ➢ Poor maintainability: styles are mixed with content.
  - ➢ Repetition: styles must be repeated for similar elements.
  - ➢ Difficult to scale for large websites.
  - ➢ Overrides other CSS unless specificity is carefully managed.

- **<u>Internal css:-</u>**

  Advantages:-

  - ➢ Keeps styles in one place within the HTML file.
  - ➢ Useful for single-page applications or documents.
  - ➢ No need to load external files (faster in small projects).

Disadvantages:-

- Not reusable across multiple pages.
- Increases page load time if styles are large.
- Still mixes content with presentation, though less so than inline CSS.

- **External css:-**

  Advantages:-

  - Best for maintaining large websites.
  - CSS is completely separated from HTML (better structure).
  - Styles can be reused across multiple HTML pages.

  Disadvantages:-

  - Requires an additional HTTP request to load the CSS file (though often cached).
  - Slightly more complex setup.
  - Not suitable for very small or quick one-off HTML pages.

CSS BOX MODEL

1. Explain the CSS box model and its components (content, padding, border,margin). How does each affect the size of an element?

- 1.content:-
  - ➢ The actual text, image, or other media inside the element.
  - ➢ The starting point of the element's size. You define the width and height of this area using width and height properties.

- 2.padding:-
  - ➢ Space between the content and the border.
  - ➢ Increases the total size of the element by adding space inside the border, but outside the content.

- 3.border:-
  - ➢ The line that wraps around the padding and content.
  - ➢ Adds to the overall size of the element. You can set width, style, and color of the border.

- 4.margin:-
    - Space outside the element’s border, used to create distance between this element and others.
    - Does not increase the element’s box size directly, but affects layout spacing between elements.

2. What is the difference between border-box and content-box box-sizing in CSS?Which is the default?

- **Border-box:-**
    - The width and height include the content, padding, and border.
    - The browser subtracts the padding and border from the total width/height to calculate the content size.
    - The total size of the element remains equal to the defined width and height.
- **Content box:-**
    - The width and height properties apply only to the content of the element.
    - Any padding and border are added outside the content dimensions.
    - Therefore, the total size of the element is larger than the defined width and height.
- What is default?

  Content box is default

CSS FLEXBOX

1. What is CSS Flexbox, and how is it useful for layout design? Explain the terms flex-container and flex-item.

- CSS Flexbox (Flexible Box Layout) is a layout model in CSS designed to arrange elements efficiently in a single dimension — either row (horizontal) or column (vertical).
- Flex box Automatically resize to fill available space
- Align, space, and order themselves without using floats or complicated positioning

- Why is flexbox useful for layout design?
    - Makes layouts more responsive and adaptive
    - Great for both horizontal and vertical alignment
    - Simplifies complex layouts (e.g., centering content)
    - Reduces the need for external hacks like floats or inline-block
    - Allows dynamic resizing of elements

- 1.flex container:-
    - The parent element that holds and controls the layout of the child elements.
- 2.flex items:-

- The child elements of the flex container.
- These items are automatically arranged in a row (default) or column based on container settings.
- Can grow, shrink, and be aligned independently.

2. **Describe the properties justify-content, align-items, and flex-direction used in Flexbox.**

- **1.justify-contnent:-**
  - flex-start: Items align at the start of the main axis
  - flex-end: Items align at the end of the main axis
  - center: Items are centered
  - space-between: Equal space between items
  - space-around: Equal space around items
  - space-evenly: Equal space between and around items
- **2.align-items:-**
  - stretch (default): Items stretch to fill the container
  - flex-start: Items align to the start of the cross axis
  - flex-end: Items align to the end of the cross axis
  - center: Items are centered on the cross axis
  - baseline: Items align according to their text baseline
- **3.flex-direction:-**
  - row (default): Left to right (horizontal)
  - row-reverse: Right to left (horizontal)

- column: Top to bottom (vertical)
- column-reverse: Bottom to top (vertical)

CSS GRID

1. Explain CSS Grid and how it differs from Flexbox. When would you use Grid over Flexbox?

- CSS Grid is a two-dimensional layout system in CSS. It allows you to design web layouts using rows and columns simultaneously, making it ideal for complex page structures.

feature	**CSS flexbox**	**CSS grid**
Layout dimension	One-dimensional (row **or** column)	Two-dimensional (rows **and** columns)
Best for	Aligning items in a line	Complex grid-based layouts
Item flow	Items placed in order automatically	Items can be precisely placed
Alignment	Along main and cross axis	Along rows, columns, and grid areas
Control	Content-driven sizing	Container-driven layout

When to Use Grid:

• You want to design a full page with rows and columns

• You want to place items in specific areas

• Example: Header, Sidebar, Main Content, Footer

When to Use Flexbox:

• You need items in a line (row or column)

- You want simple alignment and spacing
- Example: Menu bar, card list, button group

2. Describe the grid-template-columns, grid-template-rows, and grid- gap properties. Provide examples of how to use them.

- **<u>1.grid-template-columns:-</u>**
    - Defines the number and width of columns in a grid container.
    - **Ex. grid-template-columns: 200px 1fr 2fr;**
    - First column: fixed width of 200px
    - Second column: takes 1 fraction of remaining space
    - Third column: takes 2 fractions of remaining space

- **<u>2.grid-template-rows:-</u>**
    - Defines the number and height of rows in a grid container.
    - **Ex.grid-template-rows: 100px auto 50px;**
    - First row: 100px high
    - Second row: height based on content (auto)
    - Third row: fixed 50px

- **<u>3.grid-gap:-</u>**

- grid-gap has been replaced with the modern shorthand gap (works for both Grid and Flexbox).
- html:-

```
<div class="grid-container">
  <div>1</div><div>2</div><div>3</div>
  <div>4</div><div>5</div><div>6</div>
</div>
```

- css:-

```
.grid-container {
  display: grid;
  grid-template-columns: 100px 1fr 2fr;
  grid-template-rows: 50px auto;
  gap: 10px 20px; /* 10px row gap, 20px column gap
*/
}
```

Responsive Web Design with Media Queries

1. What are media queries in CSS, and why are they important for responsive design?

- What Are Media Queries in CSS?
  - ➢ Media queries are like rules in CSS that help your website look good on all devices — like mobile phones,tablets, and computers.
  - ➢ They check the screen size and change the style if needed.
  - ➢ Ex.

    ```
    @media (max-width: 600px) {
    body {
    background-color: lightblue;
    }
    }
    ```

- Media queries are important because they:
  - ➢ Make your website fit nicely on small or big screens
  - ➢ Help people read and use your website easily on Mobile
  - ➢ Change layout or style when screen size changes
  - ➢ You can use one website for all devices (no need to create separate mobile sites)

2. Write a basic media query that adjusts the font size of a webpage for screens smaller than 600px

- Here is a basic media query that adjusts the font size of the webpage when the screen is smaller than 600px:
- Ex.
  ```
  @media (max-width: 600px) {
  body {
  font-size: 14px;
  }
  }
  ```
- Explanation in Easy Words:
  - @media (max-width: 600px) → This means "if the screen is 600 pixels wide or smaller"
  - body { font-size: 14px; } → This changes the text size to 14px on small screens like mobile phones

Typography and Web Fonts

1. Explain the difference between web-safe fonts and custom web fonts. Why might you use a web-safe font over a custom font?

- **1. Web-Safe Fonts:**
  - ➢ These are common fonts that are already installed on most computers and devices.
  - ➢ Examples:

    Arial

    Times New Roman

    Verdana

    Georgia

    Courier New
  - ➢ They are considered "safe" because they will look the same across different devices and browsers.

- **2. Custom Web Fonts:**
  - ➢ These are special fonts that are not installed on all devices.
  - ➢ They are loaded using services like Google Fonts or @font-face in CSS.
  - ➢ Examples:

    Roboto ,Open Sans ,Lato ,Poppins

- **Why Use Web-Safe Fonts Over Custom Fonts?**
  You might choose web-safe fonts because:
  1.faster loading:-
  - ➢ No extra font files need to be downloaded — improves page speed.

  2.better compatibility:-
  - ➢ Works on all browsers and devices without any issue.

  3. No Internet Needed for Fonts:
  - ➢ Works even if there's no internet or the font CDN fails.

  4. Simple Projects:
  - ➢ Good for emails, small websites, or when speed and reliability are more important than style.

2. What is the font-family property in CSS? How do you apply a custom Google Font to a webpage?

- The font-family property in CSS is used to specify the typeface(font) for text content on a webpage. It allows you to set one or more font names, and the browser will use the first one available on the user's device.
- syntax-

```
selector {
font-family: "Font Name", fallback-font, generic-family;
}
```

How to Apply a Custom Google Font

To use a Google Font on your website, follow these steps:

1. Go to Google Fonts

- Visit https://fonts.google.com and choose a font.

2. Copy the Embed Link

- After selecting a font, Google will give you a <link> tag

Like this:

•<linkhref="https://fonts.googleapis.com/css2?family=Roboto&display=swap" rel="stylesheet">

- Place it inside the <head> section of your HTML document.

3. Use the Font in CSS

- Apply the font using the font-family property:

```
body {
font-family: 'Roboto', sans-serif;
}
```