CSS Selectors & Styling

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

CSS selectors are patterns used to select and style HTML elements. Examples include:

- **Element Selector:** p { color: blue; } (Targets all <p> elements)
- **Class Selector:** .button { background-color: red; } (Targets elements with class="button")
- **ID Selector:** #header { font-size: 20px; } (Targets element with id="header")

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

CSS specificity determines which styles are applied when multiple rules target the same element. It follows this order of priority:

1. Inline styles (style="color: red;") - highest specificity
2. ID selectors (#id)
3. Class, pseudo-class, and attribute selectors (.class, :hover, [type=text])
4. Element and pseudo-element selectors (div, h1, ::before) When conflicts arise, the style with the highest specificity is applied. If specificity is equal, the last declared rule takes precedence.

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

- **Inline CSS:** style="color: red;"
    - **Advantages:** Quick, overrides other styles
    - **Disadvantages:** Hard to maintain, not reusable
- **Internal CSS:** <style> p { color: red; } </style> inside <head>
    - **Advantages:** Works for single-page styling
    - **Disadvantages:** Not reusable across multiple pages
- **External CSS:** <link rel="stylesheet" href="styles.css">
    - **Advantages:** Reusable across multiple pages, better organization
    - **Disadvantages:** Requires an external file, may cause additional HTTP requests

CSS Box Model

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

The CSS box model defines how elements are structured in terms of space:

- **Content:** The actual text or image inside an element
- **Padding:** Space between content and border
- **Border:** A line surrounding the padding and content
- **Margin:** Space outside the border separating elements Each of these affects the total element size by adding extra space around content.

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

- **content-box (default):** Width/height applies only to content; padding and border increase the total size.
- **border-box:** Width/height includes content, padding, and border, making layouts easier to manage.

CSS Flexbox

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

CSS Flexbox is a layout model that provides efficient alignment and distribution of elements.

- **Flex-container:** The parent element that enables flex properties (display: flex;)
- **Flex-item:** The child elements inside the flex-container that adjust dynamically

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

- **justify-content:** Aligns items along the main axis (e.g., center, space-between)
- **align-items:** Aligns items along the cross-axis (e.g., flex-start, center)

- **flex-direction:** Defines the main axis (row, column)

CSS Grid

Question 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, unlike Flexbox, which is one-dimensional. Use Grid for full-page layouts and Flexbox for smaller, flexible sections.

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

- **grid-template-columns:** Defines column structure (grid-template-columns: 1fr 2fr;)
- **grid-template-rows:** Defines row structure (grid-template-rows: 100px auto;)
- **grid-gap:** Defines spacing (grid-gap: 10px;) Example:

```
.container {
 display: grid;
 grid-template-columns: 1fr 1fr;
 grid-template-rows: auto;
 grid-gap: 10px;
}
```

Responsive Web Design with Media Queries

Question 1: What are media queries in CSS, and why are they important for responsive design? Media queries allow styles to adapt based on screen size, making web designs responsive.

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

```
@media (max-width: 600px) {
 body {
  font-size: 14px;
 }
}
```

Typography and Web Fonts

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

- **Web-safe fonts:** Pre-installed on most devices (e.g., Arial, Times New Roman)
- **Custom web fonts:** Require external loading (e.g., Google Fonts) Web-safe fonts ensure consistency across devices, while custom fonts improve branding but may slow down page load speed.

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

- The font-family property specifies the text font.
- Example using Google Fonts:

```
<link href="https://fonts.googleapis.com/css2?family=Roboto&display=swap" rel="stylesheet">
<style>
  body {
    font-family: 'Roboto', sans-serif;
  }
</style>
```