CSS Assignment

CSS Selectors & Styling

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

Answer: A CSS selector is a pattern used to select and apply styles to HTML elements.

Examples:

- **Element Selector:** Targets all <p> elements:

css

```
p {
  color: red;
}
```

- **Class Selector:** Targets elements with the class .box:

css

Copy

```
.box {
  background-color: yellow;
}
```

- **ID Selector:** Targets an element with the ID #header:

css

```
#header {
  font-size: 20px;
}
```

- **Universal Selector:** Applies styles to all elements on the page:

css

```
* {
  margin: 0;
}
```

- **Group Selector:** Targets multiple elements at once:

css

```
h1, h2, p {
  font-family: Arial, sans-serif;
}
```

---

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

Answer: CSS specificity determines which CSS rule applies when multiple rules target the same element.

Specificity Calculation Order:

1. **Inline styles (style="color: red;")** → 1000
2. **ID selectors (#header)** → 100
3. **Class, attribute, and pseudo-class selectors (.box, [type="text"], :hover)** → 10
4. **Element selectors (p, h1, div)** → 1
5. **Universal selector (*)** and **Inherited styles** → 0

How Conflicts Are Resolved:

1. Higher specificity wins.
2. If specificity is equal, the last rule in the CSS file wins.
3. The !important rule overrides other styles, unless another !important rule with higher specificity exists.

Example:

css

#header { color: blue; }

h1 { color: red !important; }

---

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

Answer: CSS can be applied in three ways: Inline, Internal, and External.

1. **Inline CSS**
   Defined directly within an HTML element using the style attribute.
   - **Advantages:**
     - High specificity.
     - Quick for small changes.
   - **Disadvantages:**
     - Difficult to maintain for large projects.
     - Increases HTML file size.
     - Reduces reusability. Example:

html

This is red text.

2. **Internal CSS**
   Defined inside a <style> tag within the <head> section of the HTML file.
   - **Advantages:**
     - Useful for single-page styling.
     - No extra HTTP request.
   - **Disadvantages:**
     - Not reusable across multiple pages.
     - Can increase page load time. Example:

html

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

3. **External CSS**
   Stored in a separate .css file and linked to HTML using a <link> tag.
   - **Advantages:**
     - Best for large projects (maintainability & reusability).
     - Reduces HTML file size.

        - Allows caching for faster load times.
    - **Disadvantages:**
        - Requires an extra HTTP request.
        - Styles won't apply if the CSS file is missing. Example:

html

<link rel="stylesheet" href="styles.css">

CSS Box Model

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

Answer: The CSS Box Model defines how elements are structured and spaced on a webpage. It consists of four parts:

1. **Content:** The actual text, image, or element inside the box.
2. **Padding:** Space inside the border, around the content.
3. **Border:** The boundary around the element.

4. **Margin:** Space outside the border, separating elements.

How it affects element size: The total width/height of an element is calculated by adding up the content, padding, border, and margin:

css

Total width = Content + Padding + Border + Margin

Example:

- Content = 200px, Padding = 10px, Border = 5px, Margin = 5px
- Final width = 200px + 10px + 10px + 5px + 5px = 230px

---

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

Answer:

1. **content-box (Default):**
    - Width/height only includes the content. Padding & border are added separately, increasing the total size. Example:

css

width = 200px, padding = 10px, border = 5px → total = 230px

2. **border-box:**
    - Width/height includes padding & border. The content shrinks to fit within the set size. Example:

css

width = 200px (including padding & border, content shrinks)

CSS Flexbox

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

Answer: Flexbox (Flexible Box) is a CSS layout model that helps in designing responsive, one-dimensional layouts. It allows items to adjust and distribute space within a container.

- **Flex Container:** The parent element that holds flex items. Defined using display: flex;.

- **Flex Item:** The child elements inside the flex container. They automatically adjust based on the container’s rules.

---

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

Answer:

1. **justify-content:** Aligns items horizontally along the main axis.
    - Possible values: flex-start, flex-end, center, space-between, space-around, space-evenly.
2. **align-items:** Aligns items vertically along the cross axis.
    - Possible values: stretch, flex-start, flex-end, center, baseline.
3. **flex-direction:** Defines the direction of the main axis.
    - Possible values: row, row-reverse, column, column-reverse.

CSS Grid

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

Answer: CSS Grid is a two-dimensional layout system for designing layouts that require both rows and columns, while Flexbox is one-dimensional, either row or column.

Key Differences:

- **CSS Grid:** Suitable for complex layouts (dashboards, card layouts, etc.).
- **Flexbox:** Best for simpler layouts like navigation bars or buttons.

When to use Grid:
When designing layouts with both rows and columns, such as complex forms or dashboards.

---

9) Describe the grid-template-columns, grid-template-rows, and grid-gap properties. Provide examples.

Answer:

1. **grid-template-columns:** Defines the size of columns in a grid. Example: grid-template-columns: 100px 200px auto;

2. **grid-template-rows:** Defines the size of rows in a grid. Example: grid-template-rows: 100px 150px auto;
3. **grid-gap (or gap):** Sets space between rows and columns. Example: grid-gap: 20px;

Responsive Web Design with Media Queries

10) What are media queries in CSS, and why are they important for responsive design?

Answer: Media queries allow CSS to apply different styles based on the device's screen size, resolution, or other characteristics, making websites responsive. They ensure an optimal user experience across various devices.

Importance:

- Adjusts layouts for mobile, tablet, and desktop.
- Improves user experience and accessibility.
- Supports a mobile-first approach.

---

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

Answer:

```
css


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

Typography and Web Fonts

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

Answer:

1. **Web-Safe Fonts:**
    - Pre-installed on most devices (e.g., Arial, Times New Roman).
    - Faster loading, no extra downloads.
    - Limited style options.
2. **Custom Web Fonts:**
    - Loaded from external sources using @font-face or services like Google Fonts.

- More design flexibility but slower loading if not optimized.

When to Use Web-Safe Fonts:
For speed and compatibility across all devices without relying on external resources.

---

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

Answer: The font-family property specifies the font to be used for text. Multiple fonts can be listed as fallbacks.

Applying a Google Font:

1. Import in <head>:

html

<link href="https://fonts.googleapis.com/css2?family=Roboto:wght@400;700&display=swap" rel="stylesheet">

2. Apply in CSS:

css

Copy

body {

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