CSS

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

* A **CSS selector** is a pattern used to target and style specific HTML elements.
* Example = Element Selector =

p {

color: blue;

}

* Class Selector =

.centered{

text-align: center;

color: green;

}

* Id selector =

#main-title {

font-size: 24px;

color: red;

}

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

* CSS specificity is a hierarchy of selector strength. When multiple rules apply to the same element, the browser uses specificity to determine which rule takes precedence.

|  |  |  |
| --- | --- | --- |
| **Selector Type** | **Specificity Value** | **Example** |
| Inline styles | 1000 | <h1 style=”…”> |
| ID selectors | 100 | #header |
| Class, attribute, pseudo-class | 10 | .nav,[type=”text”],  :hover |
| Element and pseudo-elements | 1 | H1,::before |
| Universal selector \* | 0 | \* |

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

* Inline CSS = CSS written directly inside an HTML element using the style attribute.
* Advantages =
* Quick and easy for small tweaks.
* No need for external files—everything’s in one place.
* Useful for testing or overriding styles temporarily.
* Disadvantage =
* Poor maintainability for large projects.
* Repetitive and cluttered HTML.
* Hard to apply consistent styling across multiple elements.
* Lowest in reusability and scalability.
* Internal CSS = CSS written inside a <style> tag within the <head> of an HTML document.
* Advantages =
* Keeps styles centralized for a single page.
* Easier to manage than inline CSS.
* No need for external file loading
* Disadvantages =
* Styles apply only to that page.
* Not ideal for multi-page websites.
* Can increase page load time if overused.
* External CSS = CSS stored in a separate .css file and linked via <link> in the HTML <head> .
* Advantages =
* Best for large scale projects.
* Promotes clean separation of content and style.
* Enables consistent styling across multiple pages.
* Easier to maintain and update.
* Disadvantages =
* Requires an extra HTTP requestr to load the CSS file.
* Doesn’t work offline unless cached.
* Slightly slower initial load compared to inline styles.

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

* Content
* This is where your actual text, images, or other content lives.
* Controlled by width and height properties.
* Example: width: 200px; height: 100px;
* Padding
* Space between the content and the border.
* Transparent area that pushes the border outward.
* Example: padding: 20px; adds 20px on all sides.
* Border
* The visible edge around the padding and content.
* Can be styled with thickness, color, and type.
* Example: border: 5px solid black;
* Margin
* Space outside the border, separating the element from others.
* Also transparent, and doesn’t affect the element’s internal size.
* Example: margin: 10px; adds spacing around the element.

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

* Content Box =
* Default behaviorin CSS
* width and height apply only to the content
* Padding and border are added outside the specified dimensions
* Border Box =
* width and height include content + padding + border
* Makes layout more predictable and easier to manage

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

* Flexbox is a one-dimensional layout model that makes it easy to align and distribute space among items in a container—even when their size is unknown or dynamic. It’s ideal for laying out items in a row or column, and it adapts beautifully to different screen sizes.
* Flex Container =
* The parent element that holds flex items. You turn it into a flex container by setting: display: flex
* Flex Item =
* Any direct child of a flex container becomes a flexitem. These items can grow, shrink, and be aligned using Flexbox properties.

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

* Justify Content = Aligns items alongthemainaxis (horizontal if row, vertical if column).
* flex-start = Items align at the start
* flex-end = Items align at the end
* center = Items are centered
* space-between = Equal space between items
* space-around = Equal space around items
* space-evenly = Equal space betweenandaround items
* Align-items = Aligns items alongthecrossaxis (perpendicular to the main axis).
* stretch = Items stretch to fill container (default)
* flex-start = Align at the top (or left if column)
* flex-end = Align at the bottom (or right if column)
* center = Vertically centered
* baseline = Align based on text baseline
* flex-direction =
* row = items go left to right
* row-reverse = items go right to left
* column = items go top to bottom
* column-reverse = items go bottom to top

**Que:-8 Explain CSS Grid and how it differs from Flexbox. When would you use Grid over Flexbox?**

* CSS Grid is designed for layout control in both rows and columns-perfect for full-page or section-based designs.
* Key Features:
* Two-dimensional: You can align items both horizontally and vertically.
* Explicit layout: You define the grid structure first, then place items.
* Preciseplacement: Use grid-row, grid-column, or named grid areas.
* Flexbox is ideal for aligning items in a single direction-either row or column.
* Key Features:
* One-dimensional: Aligns items along one axis at a time.
* Content-driven: Items adapt to content size.
* Great for components: Navigation bars, button groups, cards.
* Grid Over Flexbox:
* Use Grid when:
* You need a structured layout with rows and columns.
* You want precisecontrol over placement and spacing.
* You're building complexdesigns like dashboards or galleries.
* Use Flexbox when:
* You're aligning items **in** onedirection.
* You want dynamicsizing based on content.
* You're designing components, not full layouts.

**Que:-9 Describe the grid-template-columns, grid-template-rows, and grid-gap properties. Provide examples of how to use them.**

* Grid-template-columns:
* Defines the number and widthof columns in a grid container.
* Example:

.container {

display: grid;

grid-template-columns: 100px 1fr 2fr;

}

* Grid-template-rows:
* Defines the height of rows in a grid container
* Example:
* .container {

display: grid;

grid-template-rows: 50px auto 100px;

}

* Grid-gap:
* Adds spacing between grid items. Though grid-gap is shorthand, modern CSS prefers gap.
* Example:
* .container {

display: grid;

grid-template-columns: repeat (3, 1fr);

grid-template-rows: 100px 100px;

grid-gap: 20px 10px;

}

**Que:-10 What are media queries in CSS, and why are they important for responsive design?**

* Media queries are a CSS3 feature that apply styles conditionally based on the characteristics of the user’s device or viewport.
* Important because:
* Adapts to any screen size
* Improves usability on mobile and desktop
* Avoids horizontal scrolling and awkward layouts
* Delivers a consistent experience across devices

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

* /\* Default font size for larger screens \*/

body {

font-size: 16px;

}

/\* Media query for smaller screens \*/

@media (max-width: 600px) {

body {

font-size: 14px;

}

}

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

|  |  |  |
| --- | --- | --- |
| **Reason** | **Web-Safe Fonts** | **Custom Fonts** |
| Fast loading | Yes | Can slow down |
| Offline compatibility | Yes | Requires download |
| Universal fallback | Yes | Needs backup fonts |
| Branding flexibility | Limited | High |
| Accessibility | Often better | Depends on font |

* Choose web-safe fonts when:
* You want maximum compatibility and performance
* You're building a lightweight site or email template
* You need a fallback font stack for custom fonts

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

* The font-family property defines the font stack-a prioritized list of fonts to use. If the first font isn’t available, the browser tries the next one, and so on.
* Applying a custom Google Font to a webpage:
* Step 1: Link the Font in <head>
* Add this inside your HTML <head> section:
* <link href="https://fonts.googleapis.com/css2?family=Roboto&display=swap" rel="stylesheet">
* Step 2: Use font-family in CSS
* Now apply it in your stylesheet:
* body {

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

}