Mernstack – CSS

CSS Selectors & Styling:-

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

Answer:

A CSS selector is the first part of a CSS Rule. It is a pattern of elements and other terms that tell the browser which HTML elements should be selected to have the CSS property values inside the rule applied to them.

- Element selector: Selects HTML elements based on their name
- Class selector: To make all buttons have a blue background, you can use the following CSS code:

  .button {background-color: blue; color: white;}

- ID selector: To change the background color of a header, you can use the following CSS code:
  #header {background-color: gray;}

Question 2: Explain the concept of CSS specificity. How do conflicts between multiple

Styles get resolved?

Answer:

- CSS Specificity is a fundamental concept in CSS that determines the order of style application. It is calculated based on the types of selectors used, including inline styles, IDs, classes, attributes, and element types.
- CSS conflicts between selectors occur when two or more selectors have conflicting styles applied to the same element, leading to unexpected results. In such cases, the browser has to decide which style to apply to the element. In this article, we'll understand what causes conflicting styles in CSS, and how to avoid and resolve them.

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

Answer:

Inline CSS

Inline CSS is a way of defining the styling of an HTML element by adding CSS rules directly to the element's tag using the "style" attribute. It is used for quick and simple styling changes to specific elements, without creating a separate CSS file.

Internal CSS

Internal CSS, also known as embedded CSS, involves adding CSS rules directly within the <style> element in the <head> section of an HTML document. It allows styling specific to that document.

External CSS

External CSS is used to place CSS code in a separate file and link to the HTML document. To use external CSS, create a separate file with the .css file extension that contains your CSS rules.

- Keeps styles separate from HTML, improving code readability and maintainability.
- Ideal for large projects to keep CSS centralized and reusable across multiple pages.
- Allows consistent styling across the website by referencing a single CSS file.

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?

Answer:

The CSS box model is a way to design and lay out web pages using a rectangular box around each HTML element. The box model has four main parts: content, padding, border, and margin. Each part affects the size of the element in a different way:

Content

The content area is where the text, images, and other media are displayed. The content area's dimensions are controlled by the width and height properties.

Padding

The padding area is the space between the content and the border. Padding is controlled by the padding property. Padding affects the dimensions and appearance of the element by adding space inside the element.

Border

The border area surrounds the padding and content. The border's thickness is controlled by the border property. The border area adds to the height and width of the element.

Margin

The margin area is the space between the border and other elements. The margin is controlled by the margin property. The margin controls the spacing between different elements and between the element and the edge of the screen.

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

Which is the default?

Answer:

In CSS, the default box-sizing is content-box, while border-box is another option. The main difference between the two is how they treat padding and borders when calculating the width and height of an element.

Content-box

- The default box-sizing behavior
- The width and height values apply only to the element's content
- The padding and border are added to the outside of the box

Border-box

- The browser accounts for any border and padding in the values you specify for an element's width and height
- The width and height values apply to the content, padding, and border

CSS Flexbox:-

Question 1: What is CSS Flexbox, and how is it useful for layout design? Explain the terms

flex-container and flex-item.

Answer:

CSS Flexbox, or Flexible Box Layout, is a tool for organizing elements on a web page. It's a key part of modern web design because it allows you to create responsive layouts that look good on any device.

- Flexbox is useful for building responsive designs that look great on any device.
- It's easier to design and build responsive web pages without using many float and position properties in the CSS code.
- It's a powerful layout model that provides an efficient way to arrange and align elements within a container.

Flex-container

- An HTML element that uses the flexbox layout to arrange its children
- To create a flex-container, set the element's display property to flex or inline-flex
- The flex-container can be displayed inline or in block formatting context

Flex-item

- A direct child of a flex-container
- Flex-items can grow or shrink to optimize the container space
- The align-self property specifies the alignment for a selected flex-item

Question 2: Describe the properties justify-content, align-items, and flex-direction

used in Flexbox.

Justify-content

The justify-content property in CSS Flexbox specifies how to position items along the main axis of a flex container.

- flex-end: Aligns items with the main-end side of the flexbox
- space-between: Aligns the first item with the main-start edge and the last item with the main-end edge
- space-around: Assigns equal spacing to each side of items
- space-evenly: Assigns even spacing to both ends of the flexbox and between its items

Align-items

The align-items property in CSS Flexbox specifies how items are aligned on the cross axis of a flexbox container. The cross axis is vertical by default.

- align-items: flex-start Aligns items at the start of the cross axis
- align-items: flex-end Aligns items at the end of the cross axis
- align-items: center Aligns items at the center of the cross axis
- align-items: baseline Aligns items so that the baseline of their text aligns along a horizontal line

Flex-direction

The flex-direction property in CSS is used to set the direction of the flex container's main axis. It can have one of four values: row, row-reverse, column, or column-reverse. row is the default value and arranges flex items horizontally.

- row: The flex container's main-axis is defined to be the same as the text direction. The main-start and main-end points are the same as the content direction.
- row-reverse: Behaves the same as row but the main-start and main-end points are opposite to the content direction.
- column: The flex container's main-axis is the same as the block-axis. The main-start and main-end points are the same as the before and after points of the writing-mode.
- column-reverse: Behaves the same as column but the main-start and main-end are opposite to the content direction.

CSS Flexbox:-

Question 1: Explain CSS Grid and how it differs from Flexbox. When would you use Grid

Over Flexbox?

Answer:

CSS Grid Layout

- CSS Grid Layout, is a two-dimensional grid-based layout system with rows and columns. It makes easier to design web pages without having to use floats and positioning. Like tables, grid layout allow us to align elements into columns and rows.
- To get started, you have to define a container element as a grid with display: grid, set the column and row sizes with grid-template-columns and grid-template-rows, and then place its child elements into the grid with grid-column and grid-row.

CSS Flexbox

- The CSS Flexbox offers one-dimensional layout. It is helpful in allocating and aligning the space among items in a container (made of grids). It works with all kinds of display devices and screen sizes. To get started you have to define a container element as a grid with display: flex;

You would use CSS Grid over Flexbox when you need to control the layout in both rows and columns, such as in complex designs. Flexbox is better for one-dimensional layouts, such as navigation bars and footers.

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

Answer:

Grid-template-columns

The grid-template-columns property in Cascading Style Sheets (CSS) specifies the number and width of columns in a grid layout. It's part of the CSS Grid.

```
<!DOCTYPE html>
<html>
<head>
  <title>
    CSS grid-template-columns Property
  </title>
  <style>
    .geeks {
      background-color: green;
      padding: 30px;
      display: grid;
      grid-template-columns: auto auto 200px 150px;
      grid-gap: 10px;
    }
    .GFG {
      background-color: white;
      border: 1px solid white;
      font-size: 30px;
      text-align: center;
    }
  </style>
</head>
<body>
  <div class="geeks">
    <div class="GFG">A</div>
    <div class="GFG">B</div>
    <div class="GFG">C</div>
```

```
        <div class="GFG">D</div>
        <div class="GFG">E</div>
        <div class="GFG">F</div>
        <div class="GFG">G</div>
        <div class="GFG">H</div>
    </div>
</body>
</html>
```

Grid-template-rows

The grid-template-rows property in CSS specifies the number and height of rows in a grid layout. It's used in CSS Grid Layout to define the rows of a grid.

```
<!DOCTYPE html>
<html>
<head>
    <title>
        CSS grid-template-rows Property
    </title>
    <style>
        .geeks {
            background-color: green;
            padding: 30px;
            display: grid;
            grid-template-columns: auto auto auto auto;
            grid-template-rows: auto auto;
            grid-gap: 10px;
        }
        .GFG {
            background-color: white;
```

```
            border: 1px solid white;
            font-size: 30px;
            text-align: center;
        }
    </style>
</head>
<body>
    <div class="geeks">
        <div class="GFG">A</div>
        <div class="GFG">B</div>
        <div class="GFG">C</div>
        <div class="GFG">D</div>
        <div class="GFG">E</div>
        <div class="GFG">F</div>
        <div class="GFG">G</div>
        <div class="GFG">H</div>
    </div>
</body>
</html>
```

Grid-gap

The grid-gap is the space between grid rows and columns. It accepts any length value, such as, px , % , em , and others. The default is 0, meaning no gaps between rows and columns.

```
<!DOCTYPE html>
<html>
<head>
    <title>
        CSS grid-gap Property
    </title>
```

```
    <style>
        body {
            text-align: center;
        }
        .grid-container {
            display: grid;
            grid-template-columns: auto auto auto;
            grid-column-gap: 50px;
            grid-row-gap: 10px;
            background-color: blue;
            padding: 10px;
        }
        .grid-container>div {
            background-color: white;
            text-align: center;
            padding: 20px 0;
            font-size: 30px;
        }
    </style>
</head>

<body>
    <div class="grid-container">
        <div class="item1">G</div>
        <div class="item2">E</div>
        <div class="item3">E</div>
        <div class="item4">K</div>
        <div class="item5">S</div>
```

```
  </div>
</body>
</html>
```

Responsive Web Design with Media Queries:-

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

Answer:

CSS media queries are a way to apply CSS styles based on the device and browser environment. They are a key part of responsive web design because they allow websites to adapt to different devices and screen sizes.

- Media queries improve the user experience by allowing websites to adapt to different devices.
- This can lead to higher user satisfaction and engagement.
- Media queries can also help websites rank higher in search engine results.

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

Answer:

```
@media screen and (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?

Answer:

- A "web-safe font" is a standard font that comes pre-installed on most computers and operating systems, guaranteeing that it will display correctly on almost any device, while a "custom web font" is a unique font that is not pre-installed and needs to be loaded

from an external source like a font library (like Google Fonts) to be used on a website, offering more design flexibility but potentially causing issues if a user's device doesn't support it properly.

- You would use a web-safe font over a custom font because it guarantees that your text will display correctly on virtually any user's device, regardless of their operating system or installed fonts, ensuring consistent readability across all platforms, while a custom font might not be available on every user's system and could potentially render differently depending on their device, potentially impacting the visual appearance of your website or document.

Question 2: 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 for an element. The font-family property can hold several font names as a "fallback" system. If the browser does not support the first font, it tries the next font.

- Select a font from Google Fonts
- Copy the link to embed the font
- Paste the link into the <head> of your HTML
- Upload the font files to your website
- Update and upload your CSS file
- Use the custom font in your CSS declarations