Module 3 – Frontend – CSS and CSS3

CSS Selectors & Styling

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

- A CSS selector is a pattern that matches specific HTML elements, allowing you to apply CSS styles to those elements. CSS selectors are used to target HTML elements on your pages, allowing you to apply styles based on their ID, class, type attributes, and more.

<u>**Types of Selectors:**</u>

- **Element Selector:**Selects elements based on their tag name (e.g., p, div, h1). <u>Example</u>**:** p { color: blue; } - This will style all paragraph (<p>) elements with blue text.
- **Class Selector:**Selects elements with a specific class attribute. Class names are preceded by a dot (.). <u>Example</u>**:** .highlight { font-weight: bold; } - This will make all elements with the class "highlight" bold.
    -
- **ID Selector:**Selects a specific element with a unique ID attribute. ID names are preceded by a hash or pound sign (#). <u>Example</u>**:** #main-content { text-align: center; } - This will center the text within the element with the ID "main-content".

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

- **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.

- **Causes of Conflicting Styles in CSS:** There are several ways that conflicting styles can arise in CSS. One common cause is using multiple stylesheets, either external or internal, that have overlapping or conflicting rules. When styles from multiple stylesheets are applied to the same element, the browser will have to decide which style to use, leading to conflicts.
- Another cause of conflicting styles is the use of multiple selectors that apply to the same element. If these selectors have conflicting styles, the browser will have to decide which style to apply, based on the specificity and order of the selectors.
- There are three reasons for CSS style conflict:
    - Specificity
    - Inheritance
    - !important
- **Specificity:** The more specific selector will override the less specific one. Specificity is calculated based on the number of elements, classes, and IDs in the selector. For example, the selector with an ID has a higher specificity than the one with a class, and the one with a class has a higher specificity than the one with an element.
- **Inheritance:** Inheritance is the process by which styles applied to a parent element are passed down to its child elements. If two selectors have the same specificity, the one that comes later in the stylesheet will override the earlier one. If one of the conflicting styles is inherited, the browser will use the inherited style over the non-inherited style, even if the non-inherited style has a higher specificity or is defined later in the cascade order.
- **!important:** The styles marked with the **!important** keyword will always take precedence over any other styles

.

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

- Inline, internal, and external CSS represent three different ways to apply style rules to HTML.
- Inline CSS applies styles directly to individual HTML elements, internal CSS styles are defined within a single HTML document, and external CSS styles are stored in a separate file and linked to the HTML.

Inline CSS:

- **Definition:** Styles are written directly within the style attribute of an HTML element.
- **Advantages:** Quick and easy to implement for individual elements, useful for isolated styling changes.
- **Disadvantages:** Lacks reusability, can clutter HTML code, and has the highest specificity, potentially overriding other styles.

Internal CSS:

- **Definition:**Styles are placed within a <style> element in the <head> section of an HTML document.
- **Advantages:**Useful for single-page styling, reduces HTTP requests, and can override external styles.
- **Disadvantages:**Not reusable across multiple pages, can become cumbersome for larger projects.

External CSS:

- **Definition:**Styles are stored in a separate .css file and linked to the HTML document using a <link> tag in the <head> section.
- **Advantages:**Maintainable, reusable across multiple pages, promotes consistent styling, and can improve performance by reducing HTML size.

- **Disadvantages:**Requires managing a separate CSS file, and may increase initial load time if not optimized.

CSS Box Model

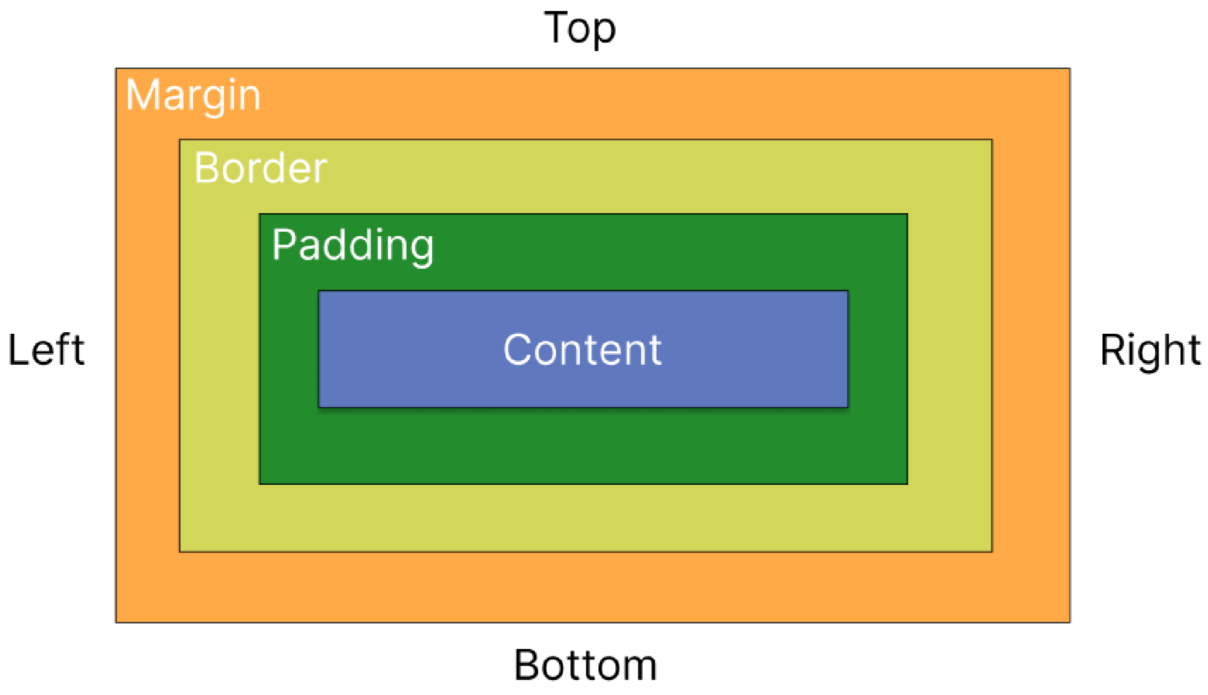
Question 1: Explain the CSS box model and its components (content, padding, border,margin). How does each affect the size of an element?

The CSS box model describes how HTML elements are displayed and sized on a webpage. It consists of four main components: content, padding, border, and margin. The content is the actual data of the element (like text or an image). Padding creates space around the content, inside the border. The border is a line surrounding the padding and content. Finally, the margin creates space around the border, separating the element from other elements. Each component directly affects the overall size of the element.

Here's how each component influences the element's size:

Content:The content's size is primarily determined by the width and height properties. If no width or height is specified, the content will expand to fit its content.
Padding:Padding increases the size of the element by adding space around the content. The total width and height of the element will be increased by the padding value on each side.
Border:The border adds a line around the padding and content. The border's width also adds to the overall size of the element, increasing both width and height.
Margin:Margins create space around the outside of the element's border. While they don't directly increase the element's size, they affect the spacing between the element and other elements,

impacting the overall layout.

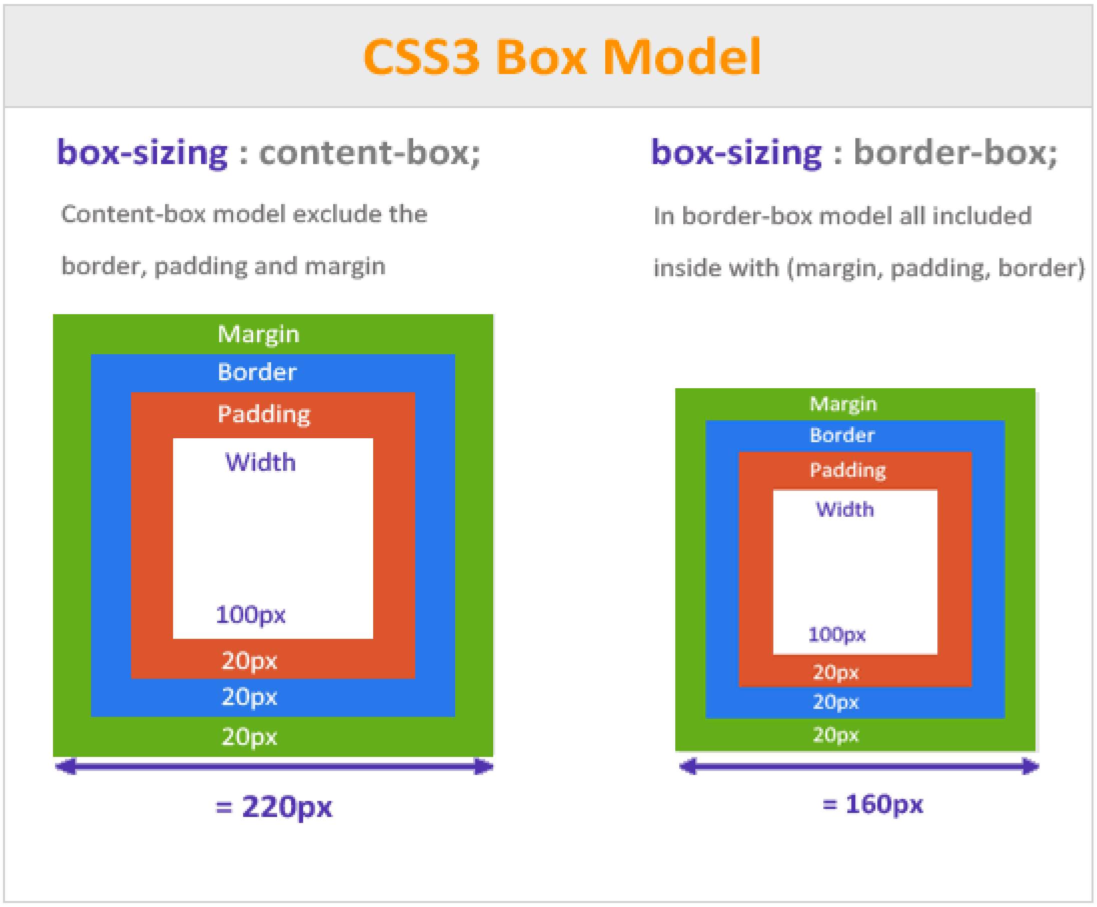


Question 2: What is the difference between border-box and content-box box-sizing in CSS?Which is the default?

content-box: This is the default value of box-sizing. The dimension of element only includes 'height' and 'width' and does not include 'border' and 'padding' given to element. Padding and Border take space outside the element.

border-box: In this value, not only width and height properties are included but you will find padding and border inside of the box for example .box {width: 200px; border: 10px solid black;} renders a box that is 200px wide.

when using box-sizing : content-box; the content size remain same while border-box size grows as padding and border grow. but when using *box-sizing: border-box; , the size of border-box remains same while size of content decreases as padding and border grow*


CSS3 Box Model
box-sizing : content-box;
Content-box model exclude the border, padding and margin
Margin
Border
Padding
Width
100px
20px
20px
20px
= 220px
box-sizing : border-box;
In border-box model all included inside with (margin, padding, border)
Margin
Border
Padding
Width
100px
20px
20px
20px
= 160px

CSS Flexbox

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

- Flexbox is short for the Flexible Box Layout module.
- Flexbox is a layout method for arranging items in rows or columns.
- Flexbox makes it easier to design a flexible responsive layout structure, without using float or positioning.

CSS Flexible Box Layout ModuleBefore the Flexible Box Layout module, there were four layout modes:

- Block, for sections in a webpage
- Inline, for text
- Table, for two-dimensional table data
- Positioned, for explicit position of an element

CSS flexbox is supported in all modern browsers.

CSS Flexbox ComponentsA flexbox always consists of:

- a Flex Container - the parent (container) <div> element
- Flex Items - the items inside the container <div>

Property	**Description**
align-self	Specifies the alignment for a flex item (overrides the flex container's align-items property)

flex	A shorthand property for the flex-grow, flex-shrink, and the flex-basis properties
flex-basis	Specifies the initial length of a flex item
flex-grow	Specifies how much a flex item will grow relative to the rest of the flex items inside the container
flex-shrink	Specifies how much a flex item will shrink relative to the rest of the flex items inside the container
order	Specifies the order of the flex items inside the container

Property	**Description**
align-content	Modifies the behavior of the flex-wrap property. It is similar to align-items, but instead of aligning flex items, it aligns flex lines
align-items	Vertically aligns the flex items when the items do not use all available space on the cross-axis

display	Specifies the display behavior (the type of rendering box) for an element
flex-direction	Specifies the direction of the flex items inside a flex container
flex-flow	A shorthand property for flex-direction and flex-wrap
flex-wrap	Specifies whether the flex items should wrap or not, if there is not enough room for them on one flex line
justify-content	Horizontally aligns the flex items when the items do not use all available space on the main-axis

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

- Flexbox provides several properties to control alignment and spacing, with align-items and justify-content being fundamental for centering elements.
- To center an element, we use the align-items property to align the item on the cross axis, which in this case is the block axis running vertically.
- We use justify-content to align the item on the main axis, which in this case is the inline axis running horizontally.

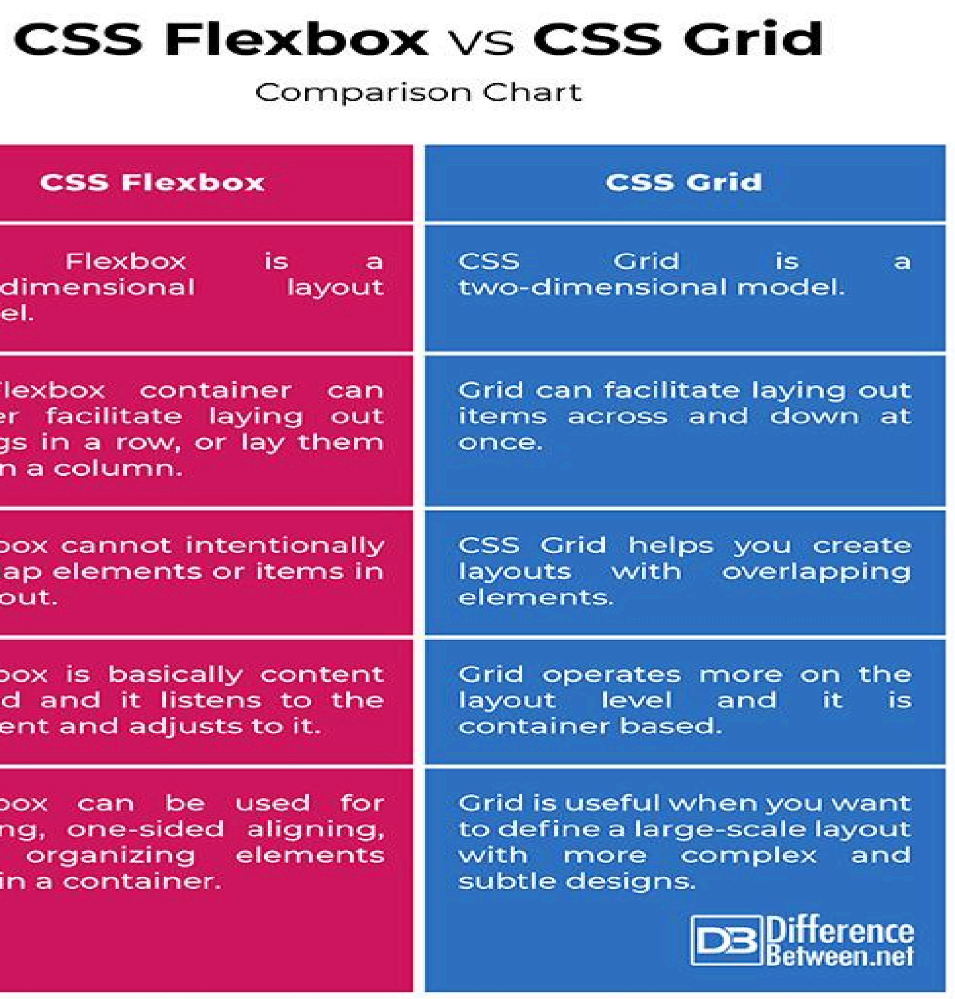
- Properties for controlling alignment used in flexbox

The properties we will look at in this guide are as follows.

- justify-content: Controls the alignment of all items on the main axis.
- align-items: Controls the alignment of all items on the cross axis.
- align-self: Controls the alignment of an individual flex item on the cross axis.
- align-content: Controls the space between flex lines on the cross axis.
- gap, column-gap, and row-gap: Used to create gaps or gutters between flex items.

CSS Grid

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



CSS Flexbox vs CSS Grid

Comparison Chart

CSS Flexbox	CSS Grid
CSS Flexbox is a one-dimensional layout model.	CSS Grid is a two-dimensional model.
A Flexbox container can either facilitate laying out things in a row, or lay them out in a column.	Grid can facilitate laying out items across and down at once.
Flexbox cannot intentionally overlap elements or items in a layout.	CSS Grid helps you create layouts with overlapping elements.
Flexbox is basically content based and it listens to the content and adjusts to it.	Grid operates more on the layout level and it is container based.
Flexbox can be used for scaling, one-sided aligning, and organizing elements within a container.	Grid is useful when you want to define a large-scale layout with more complex and subtle designs.

DifferenceBetween.net

Question2: Describe the grid-template-columns,grid-template-rows, and grid-gap properties. Provide examples of how to use them.

The grid-template-columns Property

The grid-template-columns property defines the number of columns in your grid layout, and it can define the width of each column.

The value is a space-separated-list, where each value defines the width of the respective column.

If you want your grid layout to contain 4 columns, specify the width of the 4 columns, or "auto" if all columns should have the same width.

The grid-template-rows Property

The grid-template-rows property defines the height of each row.

The value is a space-separated-list, where each value defines the height of the respective row:

Definition and UsageThe gap property defines the size of the gap between the rows and between the columns in flexbox, grid or multi-column layout. It is a shorthand for the following properties:

row-gap

column-gap

CSS Syntax

gap: *row-gap column-gap*|initial|inherit;

Property Values

value	**Description**

row-gap	Sets the size of the gap between the rows in a grid or flexbox layout
column-gap	Sets the size of the gap between the columns in a grid, flexbox or multi-column layout
initial	Sets this property to its default value. Read about *initial*
inherit	Inherits this property from its parent element. Read about *inherit*

Responsive Web Design with Media Queries

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

- Media queries in CSS are a powerful tool for creating responsive web designs.
- They allow developers to apply specific styles based on the characteristics of the user's device, such as screen size, resolution, orientation, and more.
- By using media queries, websites can adapt their layout and appearance to provide an optimal viewing experience across various devices, from large desktop screens to small mobile phones
- <u>**Importance for Responsive Design:**</u>
- **Adaptability:**Media queries allow websites to adapt to different screen sizes, ensuring content is readable and accessible on various devices.
- **User Experience:**By tailoring the layout and presentation of content, media queries enhance the user experience across different devices, improving usability and engagement.
- **Single Codebase:**Instead of creating separate websites for different devices, media queries enable the use of a single codebase that adjusts dynamically based on the user's device.
- **Mobile-First Approach:**Media queries are often used to design for mobile devices first and then progressively enhance the design for larger screens, ensuring optimal performance and user experience on mobile.
- **Accessibility:**Media queries can be used to improve accessibility, such as increasing font sizes for users with visual impairments or adjusting layouts for users with reduced motion preferences.

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

- Here's a basic media query that adjusts the font size for screens smaller than 600 pixels:

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

Explanation:

- @media (max-width: 600px): Targets devices with a screen width of 600px or less.
- Inside the block, it sets the font-size of the body element to 14px. You can adjust this value to fit your design needs.

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?

In web design, fonts are important for readability and aesthetics. Two major types of fonts used in websites are web-safe fonts and custom web fonts. Understanding their differences helps developers choose the right font for better performance and user experience.

Web-Safe Fonts:

- Definition: Web-safe fonts are fonts that are commonly pre-installed on all major operating systems (Windows, macOS, Linux).

- Examples: Arial, Times New Roman, Verdana, Courier New, Georgia.

Advantages:

- Fast loading (no need to download).

- Compatible with all devices and browsers.
- Works offline.

Disadvantages:

- Limited design variety.
- Less unique or modern-looking.

Custom Web Fonts:

- Definition: Custom web fonts are fonts that are not installed on the user's device and are loaded through the web using services like Google Fonts or Adobe Fonts.
- Examples: Roboto, Open Sans, Poppins, Lato, Montserrat.

Advantages:

- Great design flexibility.
- Enhances brand identity.
- Modern and professional appearance.

Disadvantages:

- Slower page loading time.

- May not work properly without internet.
- Requires additional setup (like linking a font service).

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 in CSS is used to specify the font for text content in a webpage. You can assign a single font or a list of fonts in order of preference (called a font stack).

Css

```
selector {

      font-family:   "Font   Name",   fallback-font,
generic-family;

}
```

Example:

```
p {

  font-family: "Arial", "Helvetica", sans-serif;

}
```

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

Step 1: Include the Font Link in HTML `<head>`

Get the font link from https://fonts.google.com

Example (Using "Poppins"):

```
<head>

<link
href="https://fonts.googleapis.com/css2?family=Popp
ins&display=swap" rel="stylesheet">

</head>
```

Step 2: Apply the Font in CSS

Use the `font-family` property in your CSS.

```
body {

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

}
```