HTML5 & CSS 2021 - Comprehensive Notes

1. HTML5 Tags

HTML provides a variety of elements or "tags" used to structure content on a webpage.

Basic Tags:

- <html>: Defines the HTML document
- <head>: Contains metadata and links to styles/scripts
- <body>: Holds the content of the page
- : Generic container for elements
- : Inline container for styling

Semantic Tags:

- <header>: Represents header content for a section or page
- <footer>: Represents footer content for a section or page
- <article>: Represents independent content
- <section>: Defines a section within a document
- <nav>: Defines navigation links

Form Tags:

- <form>: Defines an interactive form
- <input>: Specifies an input control (e.g., text, checkbox, etc.)
- <button>: Represents a clickable button
- <textarea>: Defines a multi-line input control

Media Tags:

- <audio>: Embeds sound content

- <video>: Embeds video content

- <canvas>: A drawable region used for graphics

2. CSS Basics and Box Model

CSS is used to style the visual presentation of HTML elements.

CSS Syntax:

- Selectors: Determines which HTML elements are targeted (e.g., class, id, element selector)

- Properties: Defines the styles (e.g., color, font-size, margin)

- Values: Specifies the values for each property

Box Model:

- Content: The actual content area

- Padding: Space around the content

- Border: Border surrounding the padding

- Margin: Outermost space between elements

Example of Box Model in CSS:

- .box { width: 300px; height: 150px; padding: 10px; border: 5px solid black; margin: 20px; }

3. CSS Positioning

CSS Positioning determines where elements appear on the page.

- static: Default position (normal flow of document)

- relative: Positioned relative to its original position

- absolute: Positioned relative to the nearest positioned ancestor

- fixed: Fixed to the viewport, unaffected by scrolling

- sticky: Positioned based on the user's scroll position

Example of Absolute Positioning:

- .box { position: absolute; top: 20px; left: 30px; }

4. Flexbox Layout

Flexbox is a CSS layout model that allows you to easily design flexible and responsive layouts.

Properties:

- display: flex; (to enable flexbox on a container)

- justify-content: Defines the alignment along the main axis (e.g., center, space-between)

- align-items: Defines the alignment along the cross axis (e.g., center, stretch)

Example of Flexbox:

- .container { display: flex; justify-content: space-between; align-items: center; }

5. CSS Grid Layout

CSS Grid Layout is a two-dimensional system for laying out content.

Key Concepts:

- grid-template-columns: Defines the columns of the grid

- grid-template-rows: Defines the rows of the grid

- grid-column and grid-row: Define where grid items should span

Example of Grid Layout:

- .container { display: grid; grid-template-columns: repeat(3, 1fr); grid-gap: 10px; }

6. CSS Transitions and Animations

CSS transitions allow you to smoothly change between property values.

Properties:

- transition: Defines the property, duration, and timing function

Example of Transition:

- .button { transition: background-color 0.3s ease; }

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

CSS animations provide keyframe-based animations.

Example of Animation:

- @keyframes slide { from { left: 0; } to { left: 100px; } }

- .box { animation: slide 2s infinite; }

7. Advanced CSS Techniques

Advanced CSS concepts for building modern and efficient designs.

- CSS Variables: Store reusable values for properties (e.g., --main-color: #3498db)

- Pseudo-classes: Targets elements based on their state (e.g., :hover, :focus)

- Pseudo-elements: Targets specific parts of elements (e.g., ::before, ::after)

- Media Queries: Make the design responsive by applying styles based on screen size

Example of Media Query:

- @media screen and (max-width: 768px) { .container { grid-template-columns: 1fr; } }