Backpack : HTML, CSS, JS

Part-1 HTML:

—Semantic Tags

1. What are semantic HTML tags?

Answer: Semantic HTML tags provide meaning to the content they wrap. Examples include `<header>`, `<footer>`, `<article>`, and `<section>`, making it easier for search engines and developers to understand the page structure.

2. What is the `<article>` tag used for?

Answer: The `<article>` tag is used for self-contained content that can stand alone, such as a blog post or news article.

3. What is the difference between `<section>` and `<div>`?

Answer: `<section>` is a semantic tag that groups related content, while `<div>` is a non-semantic tag used purely for layout purposes.

4. Why should you use semantic tags in HTML?

Answer: Semantic tags enhance accessibility, SEO, and make the code more understandable by defining the purpose of the content.

Attributes

5. What is an attribute in HTML?

Answer: An attribute is additional information provided inside the tag, modifying its behavior. For example, `class` or `id`.

6. What is the purpose of the `alt` attribute in images?

Answer: The `alt` attribute provides alternative text for an image, improving accessibility and SEO.

7. How do you create a hyperlink in HTML?

Answer: Use the `href` attribute in an `<a>` tag, e.g., `<a href="https://www.example.com">Link</a>`.

8. How do you use the `target` attribute in an anchor tag?

Answer: The `target` attribute specifies where to open the linked document. For example, `target="_blank"` opens it in a new tab.

HTML Elements

9. What is the difference between block-level and inline elements?

Answer: Block-level elements (e.g., `<div>`, `<p>`) take up the full width, while inline elements (e.g., `<span>`, `<a>`) only take up as much width as their content.

10. What are void elements in HTML?

Answer: Void elements are elements that do not have closing tags. Examples include `<img>`, `<br>`, and `<input>`.

11. What is the purpose of the `<meta>` tag?

Answer: The `<meta>` tag provides metadata about the HTML document, such as charset, author, and description, which helps in SEO and page rendering.

12. How do you create an ordered list in HTML?

Answer: Use the `<ol>` tag for ordered lists and `<li>` for each list item, e.g., `<ol><li>Item 1</li><li>Item 2</li></ol>`.

Forms and Inputs

13. How do you create a form in HTML?

Answer: Use the `<form>` tag and include input elements like `<input>`, `<textarea>`, `<select>`, and `<button>`. For example, `<form action="/submit" method="POST">`.

14. What is the purpose of the `method` attribute in forms?

Answer: The `method` attribute specifies how the form data is sent to the server (`GET` or `POST`).

15. How do you create a checkbox input in HTML?

Answer: Use the `<input type="checkbox">` tag for checkboxes, e.g., `<input type="checkbox" name="subscribe" value="yes">`.

16. How do you associate a label with a form element?

Answer: Use the `for` attribute in the `<label>` tag to associate it with an `id` of an input, e.g., `<label for="name">Name</label><input id="name" type="text">`.

Media

17. How do you embed an image in HTML?

Answer: Use the `<img>` tag with `src` and `alt` attributes, e.g., `<img src="image.jpg" alt="description">`.

18. How do you embed a video in HTML?

Answer: Use the `<video>` tag with the `src` and `controls` attributes, e.g., `<video src="video.mp4" controls></video>`.

19. What is the purpose of the `<audio>` tag?

Answer: The `<audio>` tag embeds sound content on a webpage, with controls like play, pause, and volume.

20. How do you add a YouTube video to your HTML page?

Answer: You can embed a YouTube video using the `<iframe>` tag, e.g., `<iframe src="https://www.youtube.com/embed/videoid"></iframe>`.

Part 2: CSS (15 Questions)

Selectors

21. What are CSS selectors?

Answer: CSS selectors are used to select the HTML elements you want to style, such as element (`p`), class (`.className`), and ID (`#id`).

22. What is the difference between a class selector and an ID selector?

Answer: A class selector applies to multiple elements, while an ID selector applies to only one unique element.

23. How do you select all `<p>` elements inside a `<div>`?

Answer: Use the descendant selector: `div p {}` to select all `<p>` elements within a `<div>`.

Box Model

24. What is the CSS box model?

Answer: The box model describes the layout of elements, including the content, padding, border, and margin.

25. How do padding and margin differ in the box model?

Answer: Padding is the space between the content and the element’s border, while margin is the space outside the border between the element and other elements.

26. How do you set the width and height of an element including its padding and border?

Answer: Use `box-sizing: border-box;` to include padding and border in the element's total width and height.

Positioning and Layout

27. What is the `position` property in CSS?

Answer: The `position` property defines how an element is positioned on the page. Values include static, relative, absolute, and fixed.

28. What is the difference between absolute and relative positioning?

Answer: Absolute positions an element relative to its nearest positioned ancestor, while relative positions an element relative to its normal position.

29. What is Flexbox in CSS?

Answer: Flexbox is a layout model that makes it easier to align and distribute space among items in a container using properties like `display: flex`, `justify-content`, and `align-items`.

Responsive Design

30. What are media queries in CSS?

Answer: Media queries allow you to apply CSS rules based on device characteristics like screen width, using `@media` rules, e.g., `@media (max-width: 600px) {}`.

31. What is the difference between min-width and max-width in media queries?

Answer: `min-width` applies styles when the viewport width is greater than the specified value, while `max-width` applies styles when the width is less than the specified value.

32. How do you create a responsive grid layout?

Answer: Use CSS Grid or Flexbox along with media queries to adjust the layout for different screen sizes.

Styling

33. How do you change the background color of an element?

Answer: Use the `background-color` property, e.g., `body { background-color: lightblue; }`.

34. What is the `color` property in CSS?

Answer: The `color` property sets the color of the text, e.g., `p { color: red; }`.

35. How do you change the font size of an element?

Answer: Use the `font-size` property, e.g., `h1 { font-size: 32px; }`.

Part 3: JavaScript (25 Questions)

DOM Manipulation

36. What is the DOM in JavaScript?

Answer: The DOM (Document Object Model) is a representation of the HTML structure as objects, allowing JavaScript to interact with and manipulate the content and layout.

37. How do you select an element by its ID in JavaScript?

Answer: Use `document.getElementById('id')` to select an element by its ID.

38. How do you add a class to an HTML element using JavaScript?

Answer: Use `element.classList.add('class-name')` to add a class to an element.

39. How do you remove an element from the DOM in JavaScript?

Answer: Use `element.remove()` to remove an element from the DOM.

40. How do you change the text content of an element in JavaScript?

Answer: Use `element.textContent = 'New text'` to change the text of an element.

Control Flow

41. What is a for loop in JavaScript?

Answer: A for loop is used to execute a block of code a certain number of times. Example:
`for (let i = 0; i < 5; i++) { console.log(i); }`.

42. What is an if-else statement in JavaScript?

Answer: An if-else statement executes a block of code if a condition is true, otherwise it runs the code in the else block.

43. What is the purpose of the switch statement in JavaScript?

Answer: A switch statement allows you to execute different blocks of code based on the value of a variable.

44. What is a while loop in JavaScript?

Answer: A while loop repeatedly executes a block of code as long as a specified condition is true.

45. How do you exit a loop in JavaScript?

Answer: Use the `break` statement to exit a loop prematurely.

ES6 Features

46. What are arrow functions in JavaScript?

Answer: Arrow functions are a shorthand syntax for writing functions in JavaScript. Example: `const add = (a, b) => a + b;`.

47. What is destructuring in JavaScript?

Answer: Destructuring allows you to extract values from arrays or objects into variables. Example: `const [a, b] = [1, 2];`.

48. What is the difference between `let` and `const` in JavaScript?

Answer: `let` allows you to reassign variables, while `const` creates read-only constants that cannot be reassigned**.**

49. What is template literals in JavaScript?

Answer: Template literals are string literals that allow embedded expressions, using backticks (``) and `${}` for placeholders.

50. What are default parameters in JavaScript?

Answer: Default parameters allow you to initialize function parameters with default values if no value is passed.

51. What is the spread operator in JavaScript?

Answer: The spread operator (`...`) allows an iterable to expand in places where multiple arguments are expected.

52. What is the rest parameter in JavaScript?

Answer: The rest parameter (`...args`) allows you to pass an indefinite number of arguments to a function.

53. What are promises in JavaScript?

Answer: Promises represent asynchronous operations that either resolve or reject. Example: `new Promise((resolve, reject) => {});`.

APIs

54. What is an API?

Answer: An API (Application Programming Interface) allows applications to communicate with each other, providing data and services.

55. What is the Fetch API in JavaScript?

Answer: The Fetch API is used to make network requests to retrieve resources from a server. Example: `fetch('https://api.example.com/data').then(response => response.json());`.

56. How do you handle errors in Fetch API requests?

Answer: Use `.catch()` to handle errors in a Fetch request, e.g., `fetch(url).catch(error => console.error('Error:', error));`.

57. What is JSON and how is it used with APIs?

Answer: JSON (JavaScript Object Notation) is a lightweight format for data interchange, commonly used to send and receive data in API requests.

58. What is the difference between GET and POST requests in APIs?

Answer: GET requests retrieve data, while POST requests send data to the server.

59. How do you send data in a POST request using Fetch API?

Answer: Use the `fetch()` function with the method set to `POST` and include data in the body. Example:

```
fetch(url, {
  method: 'POST',   headers: {
    'Content-Type': 'application/json'
  },
  body: JSON.stringify({ key: 'value' })
}
);
```

60. What is CORS and how does it relate to APIs?

Answer: CORS (Cross-Origin Resource Sharing) is a security feature that restricts how resources on a web page can be requested from another domain.