Let's dive right in:

1. **What is HTML? Differences between HTML & HTML5? Advantages of HTML5?** HTML stands for HyperText Markup Language and is used to create web pages. HTML5 is the latest version , offering new elements like <video>, <audio>, and <canvas>. It improves performance, supp orts multimedia without plugins, and provides better error handling.
2. **What is the difference between HTML and XHTML?** HTML is forgiving with errors, while XHT ML (Extensible HyperText Markup Language) is stricter and must be well-formed XML.
3. **What is the role of DOCTYPE in HTML?** It tells the browser which version of HTML you're usi ng, ensuring the page is rendered correctly.
4. **Difference between Head & Body in HTML? Where to place JS link reference?** The <head> c ontains metadata, like the title and links to stylesheets. The <body> contains the content disp layed to users. Place JS link references just before the closing </body> tag for better loading performance.
5. **What is Title Tag in HTML?** The <title> tag sets the title of the web page, shown in the brows er tab.
6. **What are different HTML heading tags?** They range from <h1> to <h6>, with <h1> being the most important and <h6> the least.
7. **What are Meta Tags?** These provide metadata about the HTML document, like author, descri ption, and keywords. They're placed in the <head>.
8. **What are HTML Elements? What is the difference between Element & Tag?** An element co mprises a start tag, content, and an end tag, while a tag is just the start or end part. Example: Paragraph is an element; is a tag.
9. **What are the roles and uses of the** **element in HTML?** is a block-level container used to group elements for styling or layout purposes.
10. **What is the difference between** **and** **element?** is block-level, used for large sections; is inline, used for small sections within text.
11. **What is the use of** <label> **Tag?** It associates a text label with an input element, improving ac cessibility.
12. **What is the role of** <a>**,**
, <hr>**,** **,** <img>**,** <input>**, &** <button> **elements?**
    - <a>: Creates hyperlinks.
    -
: Inserts a line break.
    - <hr>: Creates a horizontal rule.
    - : Emphasizes text.
    - <img>: Displays images.
    - <input>: Creates input controls.
    - <button>: Creates a clickable button.
13. **What is the role of** <header>**,** <main>**,** <section>**,** <footer>**, &** <address> **elements in HTML?**

- <header>: Contains introductory content or navigation links.
- <main>: Denotes the main content of the document.
- <section>: Groups related content.
- <footer>: Contains footer information.
- <address>: Provides contact information.

14. **What are Root, Parent, Child, & Nested elements?**
    - Root: The top-level element (usually <html>).
    - Parent: An element that contains another element.
    - Child: An element contained within a parent.
    - Nested: An element within another element.
15. **What are Empty Elements?** Elements that don't have content and are self-closing, like <img> and
.
16. **What are Semantic Elements in HTML?** Elements that convey meaning about their content, like <article>, <nav>, and <footer>.
17. **Can HTML tags be written in Uppercase?** Yes, but lowercase is recommended for consistency and readability.
18. **What are the 3 differences between Block-Level & Inline Elements?**
    - Block-level elements start on a new line; inline elements do not.
    - Block-level elements can contain other blocks and inlines; inline elements typically contain text or other inline elements.
    - Block-level elements take up the full width; inline elements take up only as much width as necessary.
19. **List all Block-Level & Inline Elements in HTML.**
    - Block-level: , <h1> to <h6>, <p>, <ul>, <ol>, <li>, <table>, <section>, <header>, <footer>, etc.
    - Inline: <a>, , , , <img>, <input>, <label>, etc.

20. **What are HTML Attributes? What are the Types of HTML attributes?** HTML attributes provid e additional information about elements. Types include: global attributes (e.g., class, id, style), event attributes (e.g., onclick, onmouseover), and specific attributes for elements (e.g., src for <img>, href for <a>).

21. **What are the id, style, & class attributes of an element? When to use what?**

    - **id**: Unique identifier for an element, used for CSS and JS targeting.
    - **style**: Inline CSS to apply specific styles.
    - **class**: A way to group multiple elements for styling and JS targeting.

Use id for unique elements, class for groups of elements, and style for quick, specific styling.

22. **What will happen if two elements have the same ids?** IDs should be unique. If two elements have the same ID, it can cause unexpected behavior and issues in CSS and JavaScript, as ID s electors expect unique matches.

23. **What are Data Attributes in HTML?** Custom attributes that store extra information. Prefixed with data-, they are useful for storing data to be accessed via JavaScript.

24. **What are the 5 Types of Links in HTML?**

    - Internal links: Navigate within the same page.
    - External links: Navigate to another website.
    - Email links: Open an email client to send a message.
    - Phone links: Make a call.
    - File links: Download a file.

25. **What is the difference between Absolute & Relative URLs?**

    - **Absolute URL**: Full path including the domain, e.g., https://example.com/page.
    - **Relative URL**: Path relative to the current domain, e.g., /page or ../page.

26. **What is the purpose of the** <nav> **element in HTML?**

    It groups navigation links, making it easier to identify and style main navigation areas of a pa ge.

27. **How do you add an external stylesheet in your HTML?** Use the <link> tag inside the <head>:

    html

    Copy

    <link rel="stylesheet" href="styles.css">

28. **How do you open a link in a new tab?** Use the target="_blank" attribute in the <a> t ag:

    html

    Copy

    Link

29. **How do you create an Email Link?** Use the mailto: scheme in the <a> tag:

    html

    Copy

    Email Us

30. **What are the different Types of Lists in HTML?**

    - Ordered List (<ol>): Numbered items.
    - Unordered List (<ul>): Bulleted items.
    - Definition List (<dl>): Terms and descriptions.

31. **What is a Nested List in HTML?** A list within another list. Example:

32. **What are** <table>**,** <tr>**,** <th>**,** <td> **elements?**

- <table>: Creates a table.
- <tr>: Table row.
- <th>: Table header cell.
- <td>: Table data cell.

33. **What is the colspan attribute in HTML?** It allows a cell to span multiple columns. For exampl e:

html

Copy

<td colspan="2">This spans two columns</td>

34. **What is the best way to add a border to a table, column, and cell?** Use CSS:

css

Copy

```
table, th, td {
  border: 1px solid black;
  border-collapse: collapse;
}
```

35. **What is CSS? What are the 3 ways to Implement CSS in HTML?** CSS (Cascading Style Sheets) styles HTML elements. You can implement it in:
    - Inline: Directly within the HTML tag.
    - Internal: Inside a <style> tag in the <head>.
    - External: Linked via a <link> tag to an external stylesheet.

36. **What is Inline Style in CSS? When to use it in real applications?** Applied directly within an H TML element:

    html

    Copy

    This is blue text

    Use for quick, specific changes.

37. **What is Internal Stylesheet in CSS? When to use it in real applications?** Defined within a <st yle> tag in the <head>:

    html

    Copy

    <style>

    p { color: blue; }

    </style>

    Use for styling a single document.

38. **What is External Stylesheet in CSS? When to use it in real applications?** Linked using a <link > tag:

    html

    Copy

    <link rel="stylesheet" href="styles.css">

    Use for styling multiple documents.

39. **What are CSS Selectors and what are their types?** Selectors target HTML elements to apply s tyles. Types include:

    - Element: p { }
    - ID: #id { }
    - Class: .class { }
    - Attribute: [type="text"] { }
    - Pseudo-classes: a:hover { }
    - Pseudo-elements: p::after { }

40. **How do you Include CSS in a webpage or HTML?**

    - Inline: style="property:value;"
    - Internal: <style>...</style> in the <head>
    - External: <link rel="stylesheet" href="styles.css"> in the <head>

41. **Explain Box Model in CSS.** The box model represents the layout of elements, comprising:

    - Content: The actual content.
    - Padding: Space inside the border.
    - Border: Surrounds the padding (optional).
    - Margin: Space outside the border.

42. **Explain Padding, Margin, and Border.**

- **Padding**: Space between the content and the border.
- **Margin**: Space outside the border, separating elements.
- **Border**: Surrounds the padding and content, like a frame.

Let's make these clear and straightforward:

each.

- **String**: Textual data.

```javascript
let name = "John";
```

- **Number**: Numeric data.

```javascript
let age = 30;
```

- **Boolean**: True or false.

```javascript
let isActive = true;
```

- **Object**: Collections of key-value pairs.

```javascript
let person = {name: "John", age: 30};
```

- **Array**: Ordered lists.

```javascript
let fruits = ["Apple", "Banana", "Cherry"];
```

- **Undefined**: Variable without a value.

```javascript
let x;
```

- **Null**: Intentional absence of value.

```javascript
let y = null;
```

- **Symbol**: Unique and immutable value.

```javascript
let sym = Symbol("id");
```

- **BigInt**: Large integers.

```javascript
let bigNum = 1234567890123456789012345678901234567890n;
```

43. **What are the different data types available in JavaScript? Provide examples of each.**

- **String**: Textual data.

  javascript

  Copy

  let name = "John";

- **Number**: Numeric data.

  javascript

  Copy

  let age = 30;

- **Boolean**: True or false.

  javascript

  Copy

  let isActive = true;

- **Object**: Collections of key-value pairs.

  javascript

  Copy

  let person = {name: "John", age: 30};

- **Array**: Ordered lists.

  javascript

  Copy

  let fruits = ["Apple", "Banana", "Cherry"];

- **Undefined**: Variable without a value.

  javascript

  Copy

  let x;

- **Null**: Intentional absence of value.

  javascript

  Copy

  let y = null;

- **Symbol**: Unique and immutable value.

  javascript

  Copy

  let sym = Symbol("id");

- **BigInt**: Large integers.

  javascript

  Copy

  let bigNum = 1234567890123456789012345678901234567890n;

44. **Explain the difference between var, let, and const. When would you use each of these declarations?**
    - var: Function-scoped, can be redeclared, not block-scoped.
    - let: Block-scoped, can be updated but not redeclared within the same scope.
    - const: Block-scoped, cannot be updated or redeclared; used for constants.
45. **What is hoisting in JavaScript? How does it affect variables declared with var, let, and const ?** Hoisting is JavaScript's default behavior of moving declarations to the top. var declarations are hoisted and initialized with undefined, but let and const are hoisted without initialization, leading to a "temporal dead zone."

46. **What is a callback function in JavaScript? Give an example of how and when to use a callback function.** A callback is a function passed as an argument to another function, executed later. Example:

Javascript Copy

```
function greet(name, callback) {
    console.log("Hello, " + name);
    callback();
}

function sayGoodbye() {
    console.log("Goodbye!");
}

greet("John", sayGoodbye);
```

47. **What are arrow functions? How do they differ from traditional functions in JavaScript?** Arrow functions provide a shorter syntax and lexically bind `this`. Example:

Javascript Copy

```
let sum = (a, b) => a + b;
```

Unlike traditional functions, they don't have their own `this`.

46. **What is a callback function in JavaScript? Give an example of how and when to use a callback function.** A callback is a function passed as an argument to another function, executed later. Example:

javascript

Copy

function greet(name, callback) {

  console.log("Hello, " + name);

  callback();

}

function sayGoodbye() {

  console.log("Goodbye!");

}

greet("John", sayGoodbye);

47. **What are arrow functions? How do they differ from traditional functions in JavaScript?** Arr ow functions provide a shorter syntax and lexically bind this. Example:

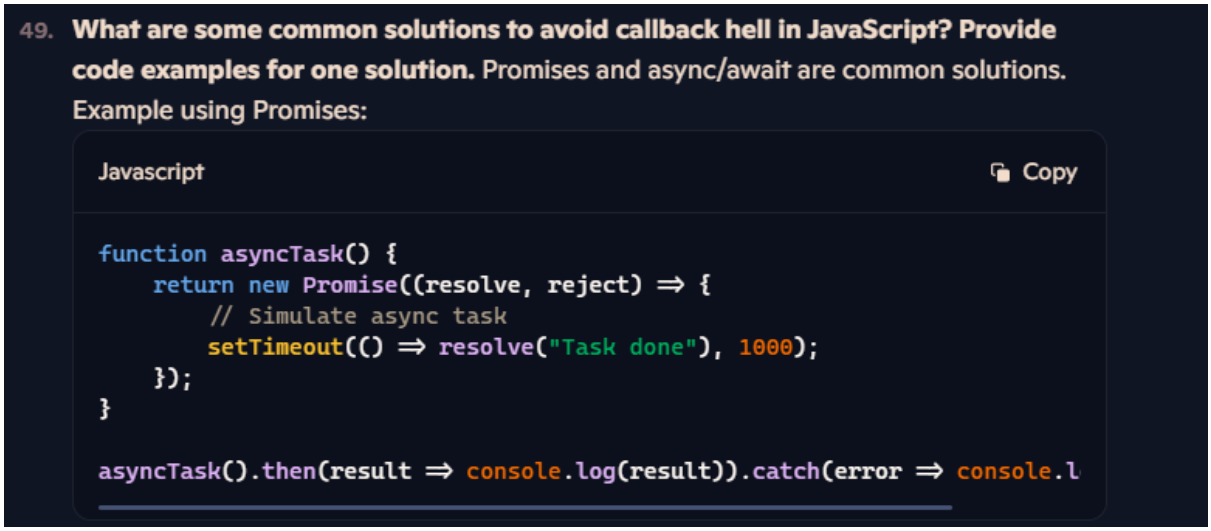
    javascript

    Copy

    let sum = (a, b) => a + b;

    Unlike traditional functions, they don't have their own this.

48. **Explain the concept of 'callback hell' in JavaScript. Why does it occur, and what problems d oes it cause?** Callback hell occurs when multiple nested callbacks create complex code, maki ng it hard to read and maintain. It can lead to deeply nested structures and hard-to-follow logic.



49. **What are some common solutions to avoid callback hell in JavaScript? Provide code examp les for one solution.** Promises and async/await are common solutions. Example using Promis es:

    javascript

    Copy

    function asyncTask() {

      return new Promise((resolve, reject) => {

        // Simulate async task

        setTimeout(() => resolve("Task done"), 1000);

      });

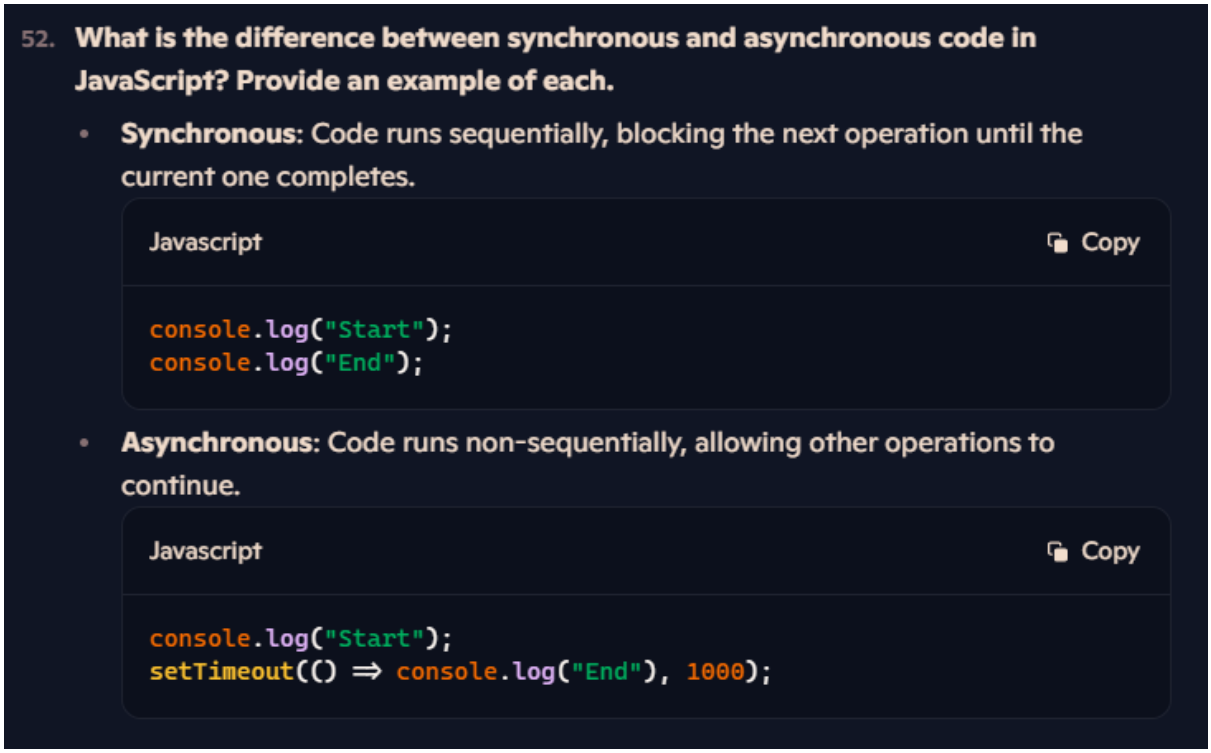
    }

asyncTask().then(result => console.log(result)).catch(error => console.log(error));

50. **Describe the scope of variables in JavaScript. How does the scope differ between variables declared with var, let, and const?**

    - var: Function-scoped.
    - let **and** const: Block-scoped. let and const are not accessible outside of their block, while var is accessible throughout the function.

51. **How does JavaScript handle asynchronous operations? Briefly explain the role of the event loop in this process.** JavaScript uses the event loop to manage asynchronous operations. The event loop continuously checks the call stack and task queue, executing tasks when the call s tack is empty.



52. **What is the difference between synchronous and asynchronous code in JavaScript? Provide an example of each.**

    - **Synchronous**: Code runs sequentially, blocking the next operation until the current o ne completes.

      javascript

      Copy

      console.log("Start");

      console.log("End");

- **Asynchronous**: Code runs non-sequentially, allowing other operations to continue.

  javascript

  Copy

  ```
  console.log("Start");
  setTimeout(() => console.log("End"), 1000);
  ```

1. What is HTML? Differences between HTML & HTML5? Advantages of HTML5?
2. What is the difference between HTML and XHTML?
3. What is the role of DOCTYPE in HTML?
4. Difference between Head & Body in HTML? Where to place JS link reference?
5. What is Title Tag in HTML?
6. What are different HTML heading tags?
7. What are Meta Tags?
8. What are HTML Elements? What is the difference between Element & Tag?
9. What are the roles and uses of the element in HTML?
10. What is the difference between and element?
11. What is the use of <label> Tag?
12. What is the role of <a>,
, <hr>, , <img>, <input>, & <button> elements?
13. What is the role of <header>, <main>, <section>, <footer>, & <address> elements in HTML?
14. What are Root, Parent, Child, & Nested elements?
15. What are Empty Elements?
16. What are Semantic Elements in HTML?
17. Can HTML tags be written in Uppercase?
18. What are the 3 differences between Block-Level & Inline Elements?
19. List all Block-Level & Inline Elements in HTML.
20. What are HTML Attributes? What are the Types of HTML attributes?
21. What are the id, style, & class attributes of an element? When to use what?
22. What will happen if two elements have the same ids?
23. What are Data Attributes in HTML?
24. What are the 5 Types of Links in HTML?
25. What is the difference between Absolute & Relative URLs?
26. What is the purpose of the <nav> element in HTML?
27. How do you add an external stylesheet in your HTML?
28. How do you open a link in a new tab?
29. How do you create an Email Link?
30. What are the different Types of Lists in HTML?

31. What is a Nested List in HTML?

32. What are <table>, <tr>, <th>, <td> elements?

33. What is the colspan attribute in HTML?

34. What is the best way to add a border to a table, column, and cell?

35. What is CSS? What are the 3 ways to Implement CSS in HTML?

36. What is Inline Style in CSS? When to use it in real applications?

37. What is Internal Stylesheet in CSS? When to use it in real applications?

38. What is External Stylesheet in CSS? When to use it in real applications?

39. What are CSS Selectors and what are their types?

40. How do you Include CSS in a webpage or HTML?

41. Explain Box Model in CSS.

42. Explain Padding, Margin, and Border.

43. What are the different data types available in JavaScript? Provide examples of each.

44. Explain the difference between var, let, and const. When would you use each of these declarations?

45. What is hoisting in JavaScript? How does it affect variables declared with var, let, and const?

46. What is a callback function in JavaScript? Give an example of how and when to use a callback function.

47. What are arrow functions? How do they differ from traditional functions in JavaScript?

48. Explain the concept of 'callback hell' in JavaScript. Why does it occur, and what problems does it cause?

49. What are some common solutions to avoid callback hell in JavaScript? Provide code examples for one solution.

50. Describe the scope of variables in JavaScript. How does the scope differ between variables declared with var, let, and const?

51. How does JavaScript handle asynchronous operations? Briefly explain the role of the event loop in this process.

52. What is the difference between synchronous and asynchronous code in JavaScript? Provide an example of each.