JAVASCRIPT DAY 23th And 24th

CODING QUESTIONS

Basic DOM Access and Manipulation

1) Change the content of an HTML element:

```
document.getElementById('elementId').innerText = "New Content";
```

2) Change the text of all elements:

```
let paragraphs = document.getElementsByTagName('p');
for (let p of paragraphs) {
  p.innerText = "Updated Paragraph";
}
```

3) Change the color of elements with class 'highlight':

```
let elements = document.getElementsByClassName('highlight');
for (let element of elements) {
  element.style.color = 'red';
}
```

4) Change the background color of the first :

```
document.querySelector('div').style.backgroundColor = 'yellow';
```

5) Apply a red border to all elements:

```
let h2Elements = document.querySelectorAll('h2');
for (let h2 of h2Elements) {
  h2.style.border = '2px solid red';
}
```

6) Count the number of elements:

```
let listItems = document.getElementsByTagName('li');
console.log(listItems.length);
```

Element Creation and Manipulation

7) Dynamically create a new button:

```
let button = document.createElement('button');
button.innerText = 'Click Me';
document.getElementById('container').appendChild(button);
```

8) Add three new list items:

```
let ul = document.querySelector('ul');
for (let i = 0; i < 3; i++) {
  let li = document.createElement('li');
```

```
  li.innerText = `Item ${i + 1}`;
  ul.appendChild(li);
}
```

9) Replace a paragraph with a new :

```
let paragraph = document.querySelector('p');
let newHeading = document.createElement('h1');
newHeading.innerText = 'New Heading';
paragraph.parentNode.replaceChild(newHeading, paragraph);
```

10) Remove the last child of a :

```
let div = document.getElementById('container');
div.removeChild(div.lastElementChild);
```

11) Create and append a text node:

```
let p = document.querySelector('p');
let textNode = document.createTextNode('This is new text');
p.appendChild(textNode);
```

Attribute Manipulation

12) Change the src attribute of an :

```
document.querySelector('img').setAttribute('src', 'new-image.jpg');
```

13) Fetch the href attribute of an anchor tag:

```
let link = document.querySelector('a');
console.log(link.getAttribute('href'));
```

14) Add a custom attribute 'data-role':

```
let elements = document.querySelectorAll('.user');
elements.forEach(el => {
  el.setAttribute('data-role', 'admin');
});
```

15) Remove the disabled attribute from a button:

```
document.querySelector('button').removeAttribute('disabled');
```

Event Handling

16) Display an alert on button click:

```
document.querySelector('button').addEventListener('click', function() {
alert('Button clicked!');
```

```
});
```

17) Toggle visibility of a when a checkbox is checked:

```
document.querySelector('input[type="checkbox"]').addEventListener('change', function() {
  let div = document.querySelector('div');
  div.style.display = this.checked ? 'block' : 'none';
});
```

18) Implement a custom event:

```
let event = new Event('customEvent');
document.querySelector('button').addEventListener('customEvent', function() {
  alert('Custom event triggered!');
});
document.querySelector('button').dispatchEvent(event);
```

19) Dynamically add and remove event listeners:

```
let button = document.querySelector('button');
function handleClick() {
  alert('Button clicked!');
}
// Add event listener
button.addEventListener('click', handleClick);
// Remove event listener
button.removeEventListener('click', handleClick);
```

CSS Manipulation

20) Add a class on click:

```
document.querySelector('div').addEventListener('click', function() {
  this.classList.add('active');
});
```

21) Toggle dark-mode on the element:

```
document.querySelector('button').addEventListener('click', function() {
  document.body.classList.toggle('dark-mode');
});
```

22) Check if an element contains a specific class:

```
let element = document.querySelector('div');
```

```
console.log(element.classList.contains('active'));
```

Complex DOM Manipulations

23) Dynamically generate a table:

```
let table = document.createElement('table');
for (let i = 0; i < 3; i++) {
  let row = table.insertRow();
  for (let j = 0; j < 4; j++) {
    let cell = row.insertCell();
    cell.innerText = `Row ${i + 1}, Cell ${j + 1}`;
  }
}
document.body.appendChild(table);
```

24) Create a form and display input values:

```
document.querySelector('form').addEventListener('submit', function(e) {
  e.preventDefault();
  let input1 = document.querySelector('input[name="input1"]').value;
  let input2 = document.querySelector('input[name="input2"]').value;
  document.getElementById('display').innerText = `Input 1: ${input1}, Input 2: ${input2}`;
});
```

25) Clone and append a element:

```
let ul = document.querySelector('ul');
let ulClone = ul.cloneNode(true);
document.getElementById('container').appendChild(ulClone);
```

26) Sort an unordered list alphabetically:

```
let ul = document.querySelector('ul');
let items = Array.from(ul.children);
items.sort((a, b) => a.innerText.localeCompare(b.innerText));
items.forEach(item => ul.appendChild(item));
```

Real-World Scenarios

27) Create a login form:

```
document.querySelector('form').addEventListener('submit', function(e) {
  e.preventDefault();
```

```
  let username = document.querySelector('input[name="username"]').value;
  let password = document.querySelector('input[name="password"]').value;
  if (username && password) {
    alert('Login successful!');
  } else {
    alert('Please fill in both fields!');
  }
});
```

28) Create a to-do list:

```
let addButton = document.querySelector('button');
let inputField = document.querySelector('input');
let ul = document.querySelector('ul');
addButton.addEventListener('click', function() {
  let li = document.createElement('li');
  li.innerText = inputField.value;
  let removeButton = document.createElement('button');
  removeButton.innerText = 'Remove';
  removeButton.addEventListener('click', () => li.remove());
  li.appendChild(removeButton);
  ul.appendChild(li);
});
```

29) Implement an image carousel:

```
let images = document.querySelectorAll('.carousel img');
let currentIndex = 0;

function showImage(index) {
  images.forEach(img => img.style.display = 'none');
  images[index].style.display = 'block';
}
document.querySelector('#next').addEventListener('click', () => {
  currentIndex = (currentIndex + 1) % images.length;
  showImage(currentIndex);
```

```
});
document.querySelector('#prev').addEventListener('click', () => {
  currentIndex = (currentIndex - 1 + images.length) % images.length;
  showImage(currentIndex);
});

showImage(currentIndex);  // Initial display
```

30) Create a dropdown menu:

```
let dropdown = document.querySelector('.dropdown');
dropdown.addEventListener('click', function() {
  let menu = document.querySelector('.dropdown-menu');
  menu.style.display = menu.style.display === 'block' ? 'none' : 'block';
});
```

FAQ'S OF DAY 23th and 24th

FAQs on DOM Manipulations in JavaScript with Answers

General Questions

1. **What is the DOM in JavaScript?**

   The DOM (Document Object Model) is a programming interface that represents an HTML or XML document as a tree structure, allowing JavaScript to manipulate its content, structure, and styles.

2. **Why is the DOM important in web development?**

   The DOM enables dynamic interaction with web pages, allowing developers to update content, styles, and structure without reloading the page.

3. **How does the DOM represent an HTML document?**

   The DOM represents the document as a tree of nodes, where each node corresponds to an element, attribute, or piece of text.

Element Access and Selection

4. **What is the difference between getElementById() and querySelector()?**

- getElementById() selects an element by its unique id.
- querySelector() selects the first element matching a CSS selector.

5. **What is the difference between querySelector() and querySelectorAll()?**

- querySelector() returns the first matching element.
- querySelectorAll() returns all matching elements as a static NodeList.

6. **What is the getElementsByTagName() method used for?**

- It selects all elements with a specified tag name and returns them as a live HTMLCollection.

7. **What does the getElementsByClassName() method do?**

- It retrieves all elements with a specified class name as a live HTMLCollection.

Element Manipulation

8. **How can you change the content of an element in the DOM?**

Use the innerHTML, textContent, or innerText properties to modify the content of an element.

9. What is the difference between innerHTML and textContent?

innerHTML: Retrieves or sets the HTML content, including tags.

textContent: Retrieves or sets only the plain text, excluding tags.

10. How do you create and add an element to the DOM?

Use createElement() to create the element and appendChild() to add it to the DOM.

11. What is the purpose of the replaceChild() method?

It replaces an existing child node with a new node within the same parent element.

12. How can you remove an element from the DOM?

Use the removeChild() method to remove a child node from its parent.

Attribute Manipulation

13. How do you retrieve the value of an element's attribute?

Use the getAttribute() method to retrieve the value of a specified attribute.

14. How do you set or change an attribute's value?

Use the setAttribute() method to add or update the value of an attribute.

15. What is the purpose of the removeAttribute() method?

It removes a specified attribute from an element.

16. How can you check if an element has a specific attribute?

Use the hasAttribute() method, which returns true or false.

CSS Manipulation

17. How do you dynamically add a class to an element?

Use the classList.add() method to add one or more class names.

18. How do you remove a class from an element?

Use the classList.remove() method to remove one or more class names.

19. What is the classList.toggle() method used for?

It adds a class if it does not exist and removes it if it does.

20. How can you check if an element contains a specific class?

Use the classList.contains() method, which returns true or false.

Event Handling

21. What is the addEventListener() method used for?

It attaches an event handler to an element for a specified event type (e.g., click, mouseover).

22. How do you remove an event listener?

Use the removeEventListener() method to detach a previously added event listener.

23. What is the dispatchEvent() method?

It programmatically triggers an event on a DOM element.

24. Can an element have multiple event listeners of the same type?

Yes, you can attach multiple event listeners of the same type to a single element.

25. What are the advantages of using addEventListener() over inline event handlers?

Allows multiple event listeners on the same element.

Enables removal of specific event listeners.

Separates HTML structure from JavaScript logic.

Advanced Concepts

26. What is a document fragment?

A lightweight container used to temporarily store nodes before adding them to the DOM for efficient updates.

27. How does cloneNode() work?

It creates a copy of a specified node. You can choose to copy only the node or the node and its child nodes.

28. What is the difference between live collections and static NodeLists?

Live collections (e.g., getElementsByClassName()) update automatically when the DOM changes.

Static NodeLists (e.g., querySelectorAll()) do not update dynamically.

29. How do you insert an element before another in the DOM?

Use the insertBefore() method with the parent node and a reference node.

30. What is the purpose of createTextNode()?

It creates a text node containing plain text that can be added to the DOM.

Best Practices

31. Why should you avoid excessive use of innerHTML?

It can expose the application to XSS attacks.

It is less efficient compared to other DOM methods.

32. How can you optimize DOM updates for better performance?

Use createDocumentFragment() for bulk updates.

Minimize direct DOM manipulations.

Batch DOM updates to reduce reflows and repaints.

33. Why should you remove unused event listeners?

To prevent memory leaks.

To improve application performance.

34. What is the difference between bubbling and capturing in event propagation?

Bubbling: Events propagate from the target element to its ancestors.

Capturing: Events propagate from the ancestors to the target element.

35. How can you ensure better maintainability in your DOM manipulation code?

Use descriptive variable names.

Modularize code by separating DOM logic into reusable functions.

Use modern methods like querySelector() and querySelectorAll() for flexibility.

FAQ'S ADDITIONAL

1. **What is the DOM in JavaScript?**

   The DOM is a structure representing HTML elements, allowing JavaScript to interact with and modify a webpage.

2. **How can you change the text of a button with the ID "submitBtn"?**

   document.getElementById("submitBtn").textContent = "Submit";

3. **What is the purpose of getElementById()?**

   It finds an element by its ID, useful for changing content or style.

4. **How can you change the background color of all elements with the class "box"?**

   ```
   const boxes = document.getElementsByClassName("box");
   for (let box of boxes) {
    box.style.backgroundColor = "blue";
   }
   ```

5. **Difference between getElementsByTagName() and getElementsByClassName()?**

   getElementsByTagName() selects elements by tag name; getElementsByClassName() selects by class name.

6. **How can you remove the first paragraph?**

   const firstParagraph = document.querySelector("p");

   firstParagraph.remove();

7. **What does querySelector() do?**

   It selects the first element matching a CSS selector.

8. **How can you create a new list item and add it to a list?**

   const newItem = document.createElement("li");

   newItem.textContent = "New Item";

   document.getElementById("myList").appendChild(newItem);

9. **What is querySelectorAll()?**

   It finds all elements matching a CSS selector.

10. **How can you set the src attribute of an image?**

    document.getElementById("image").setAttribute("src", "newImage.jpg");

11. **How to add a new element to the page?**

    Use createElement() and appendChild().

12. **How to toggle a class on an element?**

    document.getElementById("toggleButton").addEventListener("click", function() {
    this.classList.toggle("active");

    });

13. **What is the use of removeChild()?**

    It removes a specific child element from its parent.

14. **How to clone a div and append it?**

    const originalDiv = document.getElementById("originalDiv");

    const clone = originalDiv.cloneNode(true);

    document.body.appendChild(clone);

15. **How to check if an element has a specific attribute?**

    Use hasAttribute(attributeName).

16. **How to add an event listener for an alert on button click?**

    document.getElementById("myButton").addEventListener("click", function() {

     alert("Button clicked!");

    });

17. **What does addEventListener() do?**

It adds an event listener to an element for a specified event.

18. **How to get the value of an input field?**

    const username = document.getElementById("username").value;

19. **Difference between addEventListener() and removeEventListener()?**

    addEventListener() attaches an event, while removeEventListener() detaches it.

20. **How to insert a new list item before the second item?**

    const list = document.getElementById("list");

    const newItem = document.createElement("li");

    newItem.textContent = "New First Item";

    list.insertBefore(newItem, list.children[1]);

21. **What is an event object?**

    It contains details about the event, such as type and target element.

22. **How to change the text content of a paragraph with a class "message"?**

    document.querySelector(".message").textContent = "Hello World";

23. **How to trigger an event programmatically?**

    Use dispatchEvent() to simulate events.

24. **How to change the color of an element with ID "header"?**

    document.getElementById("header").style.color = "red";

25. **What is the use of classList.add()?**

    It adds a class to an element.

26. **How to create a new div and append it to a container?**

    const container = document.getElementById("container");

    const newDiv = document.createElement("div");

    newDiv.textContent = "New Div";

    container.appendChild(newDiv);

27. **What does classList.remove() do?**

    It removes a class from an element.

28. **How to remove the "highlight" class from an element?**

    document.getElementById("box").classList.remove("highlight");

29. **What is the purpose of cloneNode()?**

    It creates a copy of an element, optionally including its children.

30. How to create a text node and append it to a div?

```
const messageBox = document.getElementById("messageBox");
const textNode = document.createTextNode("This is a new message.");
messageBox.appendChild(textNode);
```

MCQ'S OF DAY 23th and 24th

Basic DOM Concepts

1. **What does DOM stand for?**

   B) Document Object Model

2. **What does the DOM represent in a web page?**

   C) The structure of the document as a tree

3. **Which method is used to select an element by its id?**

   A) getElementById()

4. **What does querySelector() return?**

   B) The first matching element

5. **Which of the following returns a live collection of elements?**

   C) getElementsByTagName()

   Element Access

6. **What is the difference between querySelector() and querySelectorAll()?**

   B) querySelector() returns the first match, querySelectorAll() returns all matches.

7. **Which method allows selecting elements by their tag name?**

   C) getElementsByTagName()

8. **How can you select all elements with the class highlight?**

   B) getElementsByClassName('highlight')

   Element Manipulation

9. **What does innerHTML allow you to do?**

   B) Set or retrieve the HTML content of an element

10. **How can you create a new element dynamically?**

    B) createElement()

11. **What is the purpose of the replaceChild() method?**

    B) Replace one child element with another

12. **Which method removes a specified child from its parent node?**

B) removeChild()

Attribute Manipulation

13. **How do you retrieve the value of an attribute?**

B) getAttribute()

14. **Which method is used to add or update an attribute of an element?**

C) setAttribute()

15. **What does the removeAttribute() method do?**

C) Removes the attribute entirely

Event Handling

16. **Which method is used to attach an event to an element?**

C) addEventListener()

17. **How do you remove an event listener?**

C) removeEventListener()

18. **What does dispatchEvent() do?**

B) Dispatches a custom event programmatically

CSS Manipulation

19. **How can you add a CSS class to an element?**

C) classList.add()

20. **Which method toggles a CSS class on an element?**

A) classList.toggle()

Advanced DOM Concepts

21. **What does createTextNode() do?**

B) Creates a text node with the specified text content

22. **Which method creates a lightweight container for storing DOM nodes?**

B) createDocumentFragment()

23. **How does cloneNode() function?**

B) Creates a copy of a specified node

24. **What is the difference between innerHTML and outerHTML?**

B) outerHTML includes the element tags; innerHTML does not

25. **Which of the following is true about querySelectorAll()?**

C) It returns a static NodeList

Attribute Manipulation

26. **How do you check if an element has a specific attribute?**

C) hasAttribute()

27. **Which method retrieves all the attributes of an element?**

B) attributes

28. **Which method allows setting multiple attributes at once?**

C) There is no direct method for this

CSS Manipulation

29. **How can you remove a CSS class from an element?**

B) classList.remove()

30. **Which of the following methods adds multiple CSS classes to an element?**

C) classList.add()

31. **How do you check if an element contains a specific class?**

B) classList.contains()

Event Handling

32. **What happens if multiple event listeners are added to the same event on an element?**

C) All listeners are executed in the order they were added

33. **What is the third parameter of addEventListener() used for?**

B) To specify event propagation phase (capture or bubble)

34. **Which of the following phases is not part of the event flow?**

D) Propagation phase

35. **What does preventDefault() do?**

B) Prevents the browser's default action for the event

36. **How can you stop an event from propagating further?**

B) stopPropagation()

Element Creation and Modification

37. **How do you add a new child element to an existing parent element?**

B) parentNode.appendChild(childNode)

38. **What is a document fragment?**

A) A lightweight container for temporary storage of nodes

39. **Which method would you use to insert a node before another child node?**

A) insertBefore()

40. **How can you dynamically update the style attribute of an element?**

    B) Accessing the style property directly

    Complex Scenarios

41. **How do you toggle between adding and removing a class on an element?**

    A) Use classList.toggle()

42. **How do you fetch the computed style of an element?**

    A) getComputedStyle()

43. **Which method is used to programmatically click a button?**

    B) button.click()

44. **How can you clone a node along with its children?**

    A) cloneNode(true)

45. **What does removeChild() return?**

    B) The removed child node

    Dynamic Content

46. **How do you dynamically create a list from an array of strings?**

    A) Using map() and createElement()

47. **Which property is used to control the visibility of an element?**

    C) Both A and B

48. **How can you scroll to a specific element on the page?**

    A) element.scrollIntoView()

49. **How do you remove all child nodes of an element?**

    D) Both A and B

50. **What does the NodeList returned by querySelectorAll() support?**

    B) Iteration using forEach()