JavaScript Events - Interview Questions & Answers

Q: What are events in JavaScript?

Events are actions or occurrences in the browser that JavaScript can respond to, such as clicks, key presses, or page loads.

Q: Difference between inline event handlers and addEventListener()?

Inline handlers (onclick='...') mix HTML and JS, making code harder to maintain. addEventListener() separates logic, allows multiple listeners, and supports capturing/bubbling.

Q: What are the phases of event flow?

1. Capturing Phase: event travels from the root down to the target. 2. Target Phase: event executes on the target element. 3. Bubbling Phase: event bubbles up from the target back to the root.

Q: What is the default phase of event listeners?

By default, event listeners are invoked in the bubbling phase unless the third parameter is set to true for capturing.

Q: What is event bubbling?

In bubbling, an event starts from the target element and bubbles up to its ancestors. Example: clicking a button inside a div triggers button → div → body → html.

Q: What is event capturing?

Capturing is the opposite of bubbling. The event travels from the root down to the target element. Example: html → body → div → button.

Q: What is event delegation?

Event delegation is attaching a single event listener to a parent element to handle events from its children. It improves performance by reducing the number of listeners.

Q: Difference between event.target and event.currentTarget?

event.target = the actual element that triggered the event. event.currentTarget = the element that the event listener is attached to.

Q: Difference between preventDefault() and stopPropagation()?

preventDefault() stops the browser's default behavior (e.g., navigation, form submission). stopPropagation() stops the event from moving further up or down the DOM tree.

Q: What does stopImmediatePropagation() do?

It prevents the event from propagating further AND also stops other listeners on the same element from running.

Q: Can multiple event listeners be added to the same element?

Yes, using addEventListener you can attach multiple listeners for the same event on the same element.

Q: How do you remove an event listener?

Use element.removeEventListener(eventType, handler). The function reference must be the same as the one used to add it.

Q: What are passive event listeners?

Passive event listeners tell the browser that the handler will not call preventDefault(). This allows browsers to optimize scrolling performance.

Q: What are custom events?

Custom events are developer-defined events created with new Event() or CustomEvent(). They allow communication between components.

Q: How are events handled in React?

React uses Synthetic Events, which are wrappers around native events for cross-browser compatibility. React also batches event handling for performance.

Example: Event Delegation

```
document.getElementById("list").addEventListener("click", function(e) { if
(e.target.tagName === "LI") { console.log("Item clicked:", e.target.textContent); }
});
```