JavaScript Events

Question 1: What are JavaScript events? Explain the role of event listeners.

Ans.

- What Are JavaScript Events ?
  - A JavaScript event is an action or occurrence that happens in the browser, which the program can respond to.
  - Events are triggered by things like:
    - User actions → clicking a button, typing in a field, hovering, submitting a form.
    - Browser actions → page loading, resizing, scrolling, or errors.
    - System actions → media playback, network status changes, etc.

- Example of Events :

Event Type	Trigger
click	When a user clicks an element
mouseover	When a user hovers over an element
Keydown	When a key is pressed
keydown	When a form is submitted
load	When a webpage finishes loading
scroll	When the user scrolls the page

- <u>What Are Event Listeners ?</u>
  - An event listener is a function that waits (or "listens") for a specific event to occur on a given element, and then runs a piece of code in response.

- In other words: You “listen” for an event on an element and define what should happen when that event occurs.

- Example: Using an Event Listener

```
// Select a button
let button = document.getElementById("myButton");
// Add a 'click' event listener
button.addEventListener("click", function() {
 alert("Button was clicked!");
 });
```

❖How it works:

1. The browser waits for the user to click the button.

2. When the click happens, the function inside addEventListener executes.

❖ Syntax

```
element.addEventListener(eventType, callbackFunction);
```

• element → the HTML element you want to listen to

• callbackFunction → the function to run when the event occurs

• eventType → a string like "click", "keydown", "submit"

➢ Example with a Named Function

```
function showMessage() {
console.log("Hello, world!");
}
document.querySelector("#myButton").addEventListener ("click",
showMessage);
```

❖Why Event Listeners Are Important :

They allow web pages to:

- React to user input dynamically.
- Create interactive elements (buttons, menus, forms).
- Build games, animations, and real-time apps.
- Keep HTML and JavaScript code separated and organized.

Question 2: How does the addEventListener() method work in JavaScript? Provide an example.

Ans.

❖What Is addEventListener()?

➢ The addEventListener() method is used to attach an event handler to a specific HTML element without overwriting any existing event handlers.

It lets you:

- Listen for a specific event (like click, keydown, mouseover, etc.)
- Run a function (called a callback) when that event occurs
- Attach multiple listeners to the same element and event if needed

❖ Syntax

element.addEventListener(eventType, listenerFunction, useCapture);

❖Parameters:

Parameter	Description
eventType	A string representing the event name (e.g., "click", "mouseover")
listenerFunction	The function that will run when the event occurs
useCapture (optional)	A boolean that controls event propagation order (false by default)

❖Example: Button Click Event

HTML

<button id=”greenBtn’>Click me </button>

JavaScript

```
// Step 1: Select the element
let button = document.getElementById("greetBtn");
// Step 2: Add an event listener
button.addEventListener("click", function() {
alert("Hello! You clicked the button!");
});
```

❖Using a Named Function :

```
function greetUser() {
 console.log("Welcome to my website!");
 }
document.getElementById("greetBtn").addEventListener( "click", greetUser);
```

This is cleaner and allows you to reuse or remove the listener later:

```
document.getElementById("greetBtn").removeEventListe ner("click", greetUser);
```

❖ Multiple Listeners on the Same Element

➢ You can attach multiple listeners to the same event and element:

```
button.addEventListener("click", () => console.log("First listener"));
button.addEventListener("click", () =>
 console.log("Second listener"));
```

❖ Optional Third Parameter: useCapture

➢ Controls whether the event is handled during the capturing phase (true) or bubbling phase (false).

Example:

```
element.addEventListener("click", handler, true);
```
