Event Handlers in JavaScript

Introduction

An **event handler** in JavaScript is a function that gets called in response to an event occurring on an element. Events are actions or occurrences that happen in the browser, such as user interactions (clicking a button, submitting a form, pressing a key) or browser actions (page load, page unload, resizing a window).

Common Events:

- click: Occurs when an element is clicked.
- mouseover: Occurs when the mouse pointer is moved onto an element.
- mouseout: Occurs when the mouse pointer is moved out of an element.
- keydown: Occurs when a key is pressed on the keyboard.
- load: Occurs when the document or a resource finishes loading.
- submit: Occurs when a form is submitted.

Event Object:

- When an event occurs, an event object is created, containing information about the event.
- The event object is passed as a parameter to the event handler.
- Example properties:
  - type: The type of the event (e.g., "click").
  - target: The element that triggered the event.
  - currentTarget: The element to which the event handler is attached.
  - preventDefault(): Prevents the default action of the event.
  - stopPropagation(): Stops the event from bubbling up the DOM tree.

Adding Event Handlers

There are multiple ways to add event handlers to elements in JavaScript:

1. **HTML Event Attributes**:
   - Inline event handlers are added directly in the HTML.

```
   <button onclick="alert('Button Clicked')">Click Me</button>
```

Event Handler Properties: Setting event handler properties directly in JavaScript.

```
<button id="propButton">Property event click</button>

  <script>
     var propertyButton  = document.getElementById("propButton");
     propertyButton.onclick = function(){
        console.log("This is property event handler")
     }
  </script>
```

addEventListener Method

The recommended way to add event handlers, allowing multiple handlers for the same event and better control.

```
  <button id="addEventButton">Add Event event click</button>

var addEventButton = document.getElementById("addEventButton");
    addEventButton.addEventListener("click", function () {
      console.log("This is add event handler");
    });
```

click

Occurs when an element is clicked.

Usage:

- Used to trigger actions when the user clicks on buttons, links, or other clickable elements.

2. mouseover

Occurs when the mouse pointer is moved onto an element.

Usage:

- Used to trigger actions when the user hovers over an element, such as displaying a tooltip or changing styles.

Example:

3. mouseout

Occurs when the mouse pointer is moved out of an element.

Usage:

- Used to trigger actions when the user moves the mouse away from an element, such as hiding a tooltip or reverting styles.

4. keydown

Occurs when a key is pressed on the keyboard.

Usage:

- Used to trigger actions when a user presses a key, such as validating input, performing keyboard shortcuts, or controlling game movements.

5. load

Occurs when the document or a resource finishes loading.

Usage:

- Used to trigger actions after the page or resource has fully loaded, such as initializing scripts or hiding a loading screen.

6 . submit

Occurs when a form is submitted.

Usage:

- Used to trigger actions when a form is submitted, such as validating the form data or preventing the default form submission to handle it via JavaScript.

```
<button id="addEventButton">Add Event event click</button>
    <button id="removeEventButton">Remove Event event click</button>

    <script>
      var propertyButton = document.getElementById("propButton");
      propertyButton.onclick = function () {
        console.log("This is property event handler");
        alert("This is property event handler");
      };

      //Click
      var addEventButton = document.getElementById("addEventButton");
      addEventButton.addEventListener("click", function () {
        console.log("This is add event handler");
      });

      //mouseover
      //   var addEventButton = document.getElementById("addEventButton");
      addEventButton.addEventListener("mouseover", function () {
        addEventButton.style.backgroundColor = "blue";
        console.log("This is mouseover");
      });

      //mouseout
      addEventButton.addEventListener("mouseout", function () {
        addEventButton.style.backgroundColor = "";
        console.log("This is mouseout");
      });
      //keydown
      var addEventButton = document.getElementById("addEventButton");
      addEventButton.addEventListener("keydown", function (event) {
        console.log("This is keydown " + event.key);
      });

      var removeEventButton = document.getElementById("removeEventButton");
      removeEventButton.removeEventListener("click", function () {
        console.log("This is remove event handler");
      });

      //load
      window.addEventListener("load", function () {
        console.log("Page fully loaded");
      });
    </script>
```

Event Object Properties and Methods

In JavaScript, when an event occurs, an event object is created and passed to the event handler. This object contains various properties and methods that provide information about the event and control its behavior.

Event Object Properties

1. **type**
   - **Description:** The type of the event, such as "click", "mouseover", or "keydown".
   - **Usage:** Used to identify the event type, which is useful when the same handler is used for multiple events.

```
addEventButton.addEventListener("click", function (e) {
      console.log(e.type );
    });
```

target

- **Description:** References the element that triggered the event.

```
addEventButton.addEventListener("click", function (e) {
      console.log(e.target.id );
    });
```

preventDefault()

- **Description:** Prevents the default action of the event from occurring.

- **Common Use Case:** Stopping form submissions, preventing link navigation, etc.

```
// Submit
  // Object Methods
  // addEventButton.addEventListener('click',function(){
  //     event.preventDefault();
  //     console.log("event prevented")
  // })
      var formEvent = document.getElementById('form-event')
      formEvent.addEventListener("submit",function(){
          event.preventDefault();
          var name = document.getElementById('user-name').value;
          console.log(name)
          console.log("hello");
          var print  = document.getElementById("print");
          print.textContent = name;
      })
```

Interview Questions

What is an event handler in JavaScript?

Answer: An event handler is a function that is executed in response to an event. It is used to define the actions that should take place when a specific event, such as a click or key press, occurs on a particular element.

How do you add an event listener to an element?

Answer: You can add an event listener to an element using the addEventListener method:

```
      element.addEventListener("eventType", eventHandlerFunction);
```

How do you prevent the default action of an event?

Answer: You can prevent the default action of an event by calling the preventDefault method on the event object:

```
          event.preventDefault();
```

What is the difference between addEventListener and setting an event handler property like onclick?

Answer:

- addEventListener allows you to add multiple event handlers to a single event and provides options for event capturing and bubbling phases.

- Setting an event handler property like onclick overrides any existing handlers for that event on the element.

How do you remove an event listener from an element?

Answer: You can remove an event listener using the removeEventListener method:

```
    element.removeEventListener("eventType", eventHandlerFunction);
```

What is the click event and when would you use it?

- **Answer:** The click event occurs when an element is clicked. It is used to trigger actions when the user clicks on buttons, links, or other clickable elements.

How can you change the background color of a div element when the mouse pointer is moved onto it?

- **Answer:** You can use the mouseover event to change the background color:

```
document
    .getElementById("myDiv")
    .addEventListener("mouseover", function () {
      this.style.backgroundColor = "yellow";
    });
```

What is the difference between the mouseover and mouseout events?

- **Answer:** The mouseover event occurs when the mouse pointer is moved onto an element, while the mouseout event occurs when the mouse pointer is moved out of an element.

How can you detect when a key is pressed on the keyboard in an input field?

- **Answer:** You can use the keydown event to detect key presses:

```

  addEventButton.addEventListener("keydown", function (event) {
    // console.log(event);
    console.log("Add event listerner of keydown is triggered " + event.key);
  })
```

How can you prevent a form from being submitted if a required field is empty?

- **Answer:** You can use the submit event and call event.preventDefault() to prevent the form submission:

```
document
      .getElementById("myForm")
```

```
        .addEventListener("submit", function (event) {
          let username = document.getElementById("username").value;
          if (username === "") {
            alert("Username is required!");
            event.preventDefault(); // Prevent form submission
          }
        });
```

What is the event object in JavaScript?

- **Answer**: The event object is automatically passed to event handlers and contains information about the event, such as the type of event, the target element, and the state of various keys.

How do you remove an event listener in JavaScript?

- **Answer**: You remove an event listener using the removeEventListener method.

```
 function handleClick() {
        alert("Button clicked!");
      }

      let button = document.getElementById("myButton");
      button.addEventListener("click", handleClick);
      button.removeEventListener("click", handleClick);
```

How do you handle keyboard events in JavaScript?

- **Answer**: You handle keyboard events using the keydown, keyup, or keypress events. For example:

```
document.addEventListener("keydown", function (event) {
        if (event.key === "Enter") {
          alert("Enter key pressed!");
        }
      });
```