Experiment No. – 8				
Date of Performance:	**18/11/25**			
Date of Submission:	**18/11/25**			
Execution (06)	Viva (02)	Professional Ethics (02)	Experiment Total (10)	Sign with Date

Experiment No. 8

Event handling with JavaScript

8.1 Aim: Design responsive web page demonstrating event handling in JavaScript

Problem Statement:

Create a single HTML button and a separate text container (a or <p>) on a web page. Implement JavaScript to handle two distinct events on the button: a single click and a double click. The single click must change the text content of the container, and the double click must change the container's background color, thereby demonstrating the ability to associate multiple event listeners with a single DOM element."

8.2 Course Outcome: Use JavaScript to develop interactive web pages.

8.3 Learning Objectives: Implement responsive web page using embedded, external and internal Javascript.

8.4 Requirement: Any Editor like notepad,VSCode etc. Browser like google chrome , Internet Explorer, Mozila etc.

8.5 Related Theory:

1. **Event Handling:** In web development, event handling refers to the process of writing code to respond to user interactions with a web page. Events can include actions such as clicking a button, hovering over an element, submitting a form, etc.

JavaScript allows developers to attach event listeners to HTML elements, which execute specified code when the corresponding event occurs.

2. **Event Listeners:** Event listeners are functions that wait for a specific event to occur and then execute a set of instructions. They are added to HTML elements using JavaScript and provide the ability to respond to user actions dynamically. Event listeners consist of two main parts: the event type (e.g., click, mouseover) and the function to be executed when the event occurs.
3. **Event Object:** When an event is triggered, an Event object is created and passed as an argument to the event handler function. This object contains information about the event, such as the type of event, the target element, and any additional data related to the event. Developers can access this information within the event handler to perform specific actions based on the event context

Common JavaScript Events Table

Event Attribute	Description
onclick	Triggered when an element is clicked.
onmouseout	Occurs when the mouse pointer leaves an element.
onmouseover	Fired when the mouse pointer moves over an element.
onkeyup	Fired when a key is released.
onkeydown	Fired when a key is pressed down.
onload	Fired when a form is submitted.
onfocus	Occurs when an element gets focus.

Example 1:
Display Hello world using javascript event on button click:

```
<!DOCTYPE html>
<html>
<body>
<h2>What Can JavaScript Do?</h2>
<p id="demo">JavaScript can change HTML content.</p>
<button type="button" onclick='document.getElementById("demo").innerHTML = "Hello JavaScript!"'>Click Me!</button>
</body>
</html>
```

Example 2:

```
<!DOCTYPE html>
<html>
```

```
<body>
<h1>JavaScript HTML Events</h1>
<h2>The onclick Attribute</h2>

<button onclick="document.getElementById('demo').innerHTML=Date()">The time is?</button>

<p id="demo"></p>

</body>
</html>
```

Example 3:

```
<html>
<body>
<h1>JavaScript HTML Events</h1>
<h2>The onclick Attribute</h2>

<p>Click the button to display the date.</p>
<button onclick="displayDate()">Date</button>

<script>
function displayDate() {
  document.getElementById("demo").innerHTML = Date();
}
</script>

<p id="demo"></p>

</body>
</html>
```

8.6 Procedure:

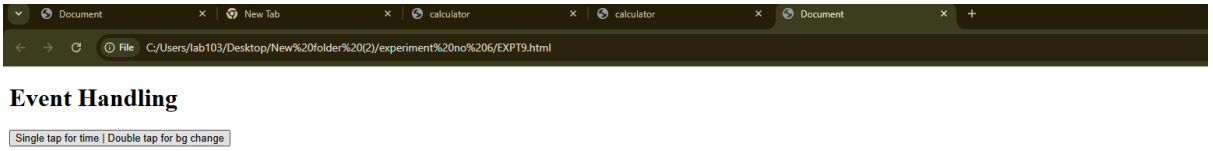
To begin coding HTML user needs only two things:

1. A simple text editor (notepad).
2. A web browser.

An HTML document can be created using an HTML text editor. Save the text file using the ".html" or ".htm" extension. Once saved as an HTML document, the file can be opened as a web page in the browser.

8.7 Program and Output:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
<body>
    <p id="haha"></p>
    <h1>Event Handling</h1>
    <button onclick="document.getElementById('haha').innerHTML=Date()" ondblclick=changebg()> Single tap for time | Double tap for bg change</button>
    <p id="different"></p>
    <script>
        function changebg(){
            document.getElementById('haha').style.backgroundColor = "blue";
        }
    </script>
</body>
</html>
```



8.8 Conclusion:

This experiment demonstrates how a single HTML element can handle multiple JavaScript events. By attaching separate event listeners for a single click and a double click on the same button, we successfully changed both the text and background color of a container. This shows the flexibility and power of JavaScript event handling in creating interactive web pages.

8.9 Questions:

1. What is event handling in web development, and why is it important?
2. What are some common types of events that can be handled in web development?

ANSWERS

1. **Event handling is the process of detecting and responding to user actions (events) on a webpage-such as clicks, keyboard input, mouse movements, or form submissions. It is important because it allows web pages to become interactive and dynamic, improving the user experience by responding instantly to user actions.**

2. **Some common types of events in web development include mouse events such as click, double-click, mouseover, and mouseout; keyboard events like keydown, keyup, and keypress; and form events including submit, change, input, and focus. There are also window events such as load, resize, and scroll, as well as touch events used in mobile devices like touchstart, touchend, and touchmove. These events allow developers to make web pages interactive by responding to different user actions.**