# TEMA 3. EVENTOS (II)

## Tipos de eventos:

**Ejercicio común (se debe realizar un sitio web con utilizando todos los elementos que se describen a continuación). Se valorará en base a la inclusión de los eventos.**

Los eventos en HTML y JavaScript son acciones o situaciones que ocurren en una página web y que pueden desencadenar la ejecución de código.

### 1. Eventos de Ratón:

- **click**: Este evento ocurre cuando se hace clic en un elemento, como un botón, un enlace o cualquier otro elemento interactivo. Se puede utilizar para manejar acciones cuando un usuario hace clic en un elemento.

```
<button id="miBoton">Haz clic</button>
<script>
    const boton = document.getElementById(“miBoton”);
    boton.addEventListener("click", () => {
        alert("¡Has hecho clic en el botón!");
    });
</script>
```

- **mouseover**: Este evento se dispara cuando el cursor del ratón se mueve sobre un elemento, lo que significa que el puntero "entra" en el área del elemento.

```
<div id="miDiv">Pasa el ratón por encima de mí</div>
<script>
    const div = document.getElementById("miDiv");
    div.addEventListener("mouseover", () => {
        div.style.backgroundColor = "lightblue";
    });
 </script>
```

- **mouseout**: Se dispara cuando el cursor del ratón se aleja de un elemento, lo que significa que el puntero "sale" del área del elemento.

```
<div id="miDiv">Lleva el ratón sobre mí y luego sácalo</div>
<script> const div = document.getElementById("miDiv");
    div.addEventListener("mouseover", () => {
        div.style.backgroundColor = "lightblue";
    });
    div.addEventListener("mouseout", () => {
        div.style.backgroundColor = "";
    });
</script>
```

- **mousedown y mouseup:** Estos eventos ocurren cuando se presiona y se libera el botón del ratón en un elemento, respectivamente. Pueden ser útiles para acciones específicas relacionadas con hacer clic y mantener presionado el ratón.

```
<button id="miBoton">Haz clic y mantén presionado</button>
<script>
	const boton = document.getElementById("miBoton");
	boton.addEventListener("mousedown", () => {
		boton.innerHTML = "Estás presionando el botón";
	});
	boton.addEventListener("mouseup", () => {
		boton.innerHTML = "Haz clic y mantén presionado";
	});
</script>
```

## 2. Eventos de Teclado:

Los eventos keydown y keyup en JavaScript se utilizan para detectar cuando una tecla del teclado se presiona y se suelta:

- **keydown (tecla presionada):** A continuación, se está escuchando el evento keydown en todo el documento. Cuando se presiona una tecla, se ejecuta la función. Se verifica si la tecla presionada es la tecla "Enter" (código 13).

```
document.addEventListener("keydown", (event) => {
// Verificar si la tecla presionada es la tecla "Enter" (código 13)
	if (event.keyCode === 13) {
		console.log("La tecla Enter ha sido presionada.");
	}
});
```

- **keyup (tecla soltada):** Se escucha el evento keyup en todo el documento. Cuando se suelta una tecla, se ejecuta la función. Luego, verifica si la tecla soltada es la tecla "Espacio" mediante el código 32.

```
document.addEventListener("keyup", (event) => {
// Verificar si la tecla soltada es la tecla "Espacio" (código 32)
	if (event.keyCode === 32) {
		console.log("La tecla Espacio ha sido soltada.”);
	}
});
```

## 3. Eventos de Formulario:.

- **Evento submit (envío de formulario):** El evento submit se dispara cuando se envía un formulario. Puedes usarlo para realizar validaciones antes de que un formulario se envíe o para realizar acciones específicas después de que se envíe. Por ejemplo, se agrega un manejador de eventos al formulario que evita el envío por defecto del formulario y muestra una alert.

```
<form id="miFormulario">
        <input type="text" name="nombre" placeholder="Nombre">
        <input type="email" name="email" placeholder="Correo electrónico">
        <button type="submit">Enviar</button>
</form>
<script>
        const formulario = document.getElementById("miFormulario");
        formulario.addEventListener("submit", (event) => {
                event.preventDefault(); // Evita el envío del formulario alert("Formulario enviado
                correctamente"); // Puedes realizar aquí validaciones antes de enviar el formulario
        });
</script>
```

- **Evento change (cambio en un campo de formulario):** El evento change se dispara cuando el valor de un campo de formulario cambia. Puedes usarlo para realizar acciones cuando se produce un cambio en un elemento, como un campo de entrada. Por ejemplo, cuando el campo de entrada se modifica se muestra una alert.

```
<input type="text" id="miInput" placeholder="Escribe algo">
<script>
        const miInput = document.getElementById("miInput");
        miInput.addEventListener("change", (event) => {
                alert("El valor del campo ha cambiado a: " + event.target.value);
        });
</script>
```

- **Evento input (cambio inmediato en un campo de formulario):** El evento input es similar al evento change, pero se dispara inmediatamente después de que cambia el valor de un campo de entrada. Esto permite detectar cambios más rápidamente, incluso antes de que un usuario termine de escribir. Por ejemplo, en este caso el evento input se lanza cuando el usuario comienza a escribir en el campo de entrada:

```
<input type="text" id="miInput" placeholder="Escribe algo">
<script>
        const miInput = document.getElementById("miInput");
        miInput.addEventListener("input", (event) => {
                alert("El valor del campo ha cambiado a: " + event.target.value);
        });
</script>
```

## 4. Eventos de Carga:

- **Evento load:** El evento load se dispara cuando la página web completa su carga, lo que incluye la carga de imágenes y otros recursos externos. Puedes usar este evento para ejecutar scripts o acciones que dependan de que todos los recursos estén disponibles.

```
<img id="miImagen" src="imagen.jpg" alt="Mi imagen">
<script> const miImagen = document.getElementById("miImagen");
miImagen.addEventListener("load", () => {
        alert("La imagen se ha cargado correctamente.");
        // Puedes realizar acciones que dependan de la imagen cargada aquí.
});
</script>
```

- **DOMContentLoaded:** El evento DOMContentLoaded se dispara cuando el documento HTML ha sido completamente cargado y analizado, sin esperar a que se carguen imágenes u otros recursos externos.

```
<p id="miParrafo">Este es un párrafo que se modificará después de cargar el documento.</p>
<script>
        document.addEventListener("DOMContentLoaded", () => {
                const miParrafo = document.getElementById("miParrafo");
                miParrafo.textContent = "¡El documento se ha cargado completamente!";
        });
</script>
```

## 5. Funciones de Temporización:

- **setTimeout:** La función permite ejecutar una función después de un retraso específico (en milisegundos).

```
function saludar() {
   console.log("¡Hola, mundo!");
}

// Ejecutar la función 'saludar' después de 2000 ms (2 segundos)
setTimeout(saludar, 2000);
```

- **setInterval:** permite ejecutar una función en intervalos regulares específicos, también en milisegundos.

```
function incrementarContador() {
  contador++;
  console.log("Contador: " + contador);
}

// Ejecutar la función 'incrementarContador' cada 1000 ms (1 segundo)
const intervalID = setInterval(incrementarContador, 1000);

// Detener el intervalo después de 5 segundos (5000 ms)
setTimeout(() => {
  clearInterval(intervalID);
  console.log("Intervalo detenido después de 5 segundos.");
}, 5000);
```

## 6. Eventos de Interacción de Usuario:

- **focus y blur:** cuando un elemento obtiene o pierde el enfoque (normalmente, un elemento de formulario, como un campo de entrada). En este ejemplo, el campo de entrada cambiará su fondo a color azul claro cuando obtenga el enfoque y volverá a blanco cuando pierda el enfoque.

```
 <input type="text" id="miInput" placeholder="Haz clic aquí">
<script>
  const input = document.getElementById("miInput");

  input.addEventListener("focus", () => {
    input.style.backgroundColor = "lightblue";
  });

  input.addEventListener("blur", () => {
    input.style.backgroundColor = "white";
  });
</script>
```

- **contextmenu:** cuando se hace clic con el botón derecho del ratón en un elemento. Por ejemplo, en este caso se muestra **un cuadro de diálogo de alerta.**

```
<div id="miDiv">Haz clic derecho aquí</div>
<script>
  const div = document.getElementById("miDiv");

  div.addEventListener("contextmenu", (event) => {
    event.preventDefault(); // Evita que aparezca el menú contextual predeterminado
    alert("Has hecho clic derecho en el div.");
  });
</script>
```

- **drag y drop:** Estos eventos están relacionados con la funcionalidad de arrastrar y soltar elementos, comúnmente utilizada en aplicaciones web interactivas. En el siguiente ejemplo se puede arrastrar el elemento "Elemento arrastrable" y soltarlo en "Área de destino". El texto del elemento arrastrable se copiará en el área de destino.

```
<div id="draggable" draggable="true">Elemento arrastrable</div>
<div id="droppable">Área de destino</div>
<script>
  const draggable = document.getElementById("draggable");
  const droppable = document.getElementById("droppable");

  draggable.addEventListener("dragstart", (event) => {
    event.dataTransfer.setData("text/plain", "Este es un elemento arrastrable");
  });

  droppable.addEventListener("dragover", (event) => {
    event.preventDefault();
  });

  droppable.addEventListener("drop", (event) => {
    event.preventDefault();
    const data = event.dataTransfer.getData("text/plain");
    droppable.innerHTML = data;
  });
</script>
```

## 7. Eventos de Navegación:

- **beforeunload:** Se dispara antes de que el usuario abandone la página. El evento `beforeunload` se dispara justo antes de que el usuario abandone una página web. Puedes usarlo para mostrar un mensaje de confirmación al usuario antes de que cierre la página.

```
<!DOCTYPE html>
<html>
<head> </head>
<body>
<a href="https://www.ejemplo.com">Ir a otro sitio web</a>
<script>
        window.addEventListener("beforeunload", (event) => {
                event.preventDefault(); // Previene que la ventana se cierre inmediatamente
                event.returnValue = ''; // Muestra un mensaje personalizado al usuario });
</script>
</body> </html>
```