Programador JavaScript Avanzado

Unidad 2: Eventos y Formularios

Contactos: tecnologia@elearning-total.com
Web: www.elearning-total.com

Indice

Unidad 2: Eventos y Formularios

Objetivos

Que el alumno logre:

- Conocer e implementar Eventos y Callbacks

API EventTarget

Es una interfaz de JavaScript para la gestión de eventos. Sirve como interfaz estándar para cualquier objeto susceptible de disparar eventos (no solo eventos HTML).

Es una interfaz implementada por objetos que pueden recibir eventos y escuchar a esos eventos. Es la base para muchos otros objetos en el modelo de eventos de JavaScript, como el HTMLElement utilizado para manipular eventos en elementos HTML.

EventTarget define tres métodos principales:

1. ***addEventListener(evento, manejador, useCapture):*** Este método permite registrar un manejador de eventos para un tipo de evento específico en el objeto EventTarget.
   El manejador de eventos será invocado cuando se dispare el evento.
   Recibe tres argumentos:
   a. el nombre del evento,
   b. una función que será el manejador del evento
   c. un parámetro opcional useCapture que indica si el evento debe ser capturado durante la fase de captura.

2. ***removeEventListener(evento, manejador, useCapture):*** Este método permite eliminar un manejador de eventos previamente registrado en el objeto EventTarget.
   Recibe los mismos argumentos que addEventListener y eliminará el manejador de eventos correspondiente para que ya no se ejecute cuando se dispare el evento.

3. ***dispatchEvent(evento):*** Este método permite disparar (lanzar) un evento en el objeto EventTarget.
   Recibe un objeto Event como argumento, que representa el evento que se va a disparar.
   El evento será procesado por los manejadores de eventos registrados para ese tipo de evento en el objeto EventTarget.

Veamos un ejemplo:

```
const miInput = document.querySelector('#miInput');

function hacerClic(evento) {
  console.log('Se ingresó un valor:', evento.target.value);
}

miInput.addEventListener('input', hacerClic);
```

Objeto Event

El objeto Event es un objeto predefinido de JavaScript que almacena información sobre un evento y se envía, para cada evento que tiene lugar, como argumento a la función o funciones que gestionan el evento.

Las propiedades y métodos del objeto Event se resumen en la siguiente tabla.

Tipo	Nombre	Descripción
Propiedades de control del evento	type	Devuelve el tipo de evento producido, sin el prefijo on (p.ej. click)
	target	Devuelve el elemento del DOM que disparó el evento (inicialmente)
	currentTarget	Devuelve el elemento del DOM que está disparando el evento actualmente (no nesariamente el elemento que disparó el evento, ya que puede ser un disparo debido a burbujeo)
Otras propiedades de control del evento	**eventPhase** (indica en qué fase de tratamiento de evento estamos) **bubbles** (booleana, indica si es un evento que burbujea o no) **cancelable** (booleana, devuelve si el evento viene seguido de una acción predeterminada que puede ser cancelada) **cancelBubble** (booleana, devuelve si el evento actual se propagará hacia arriba en la jerarquía del DOM o no).	
Propiedad temporal	timeStamp	Devuelve una medida de tiempo en milisegundos desde un origen temporal determinado.
Propiedades de localización del puntero del mouse	clientX, clientY	Devuelven las coordenadas en que se encontraba el puntero del ratón cuando se disparó el evento. Las coordenadas están referidas a la esquina superior izquierda de la ventana del navegador y se expresan en pixeles.
	screenX, screenY	Devuelven las coordenadas en que se encontraba el puntero del ratón cuando se disparó el evento. Las coordenadas están referidas a la esquina superior izquierda de la pantalla y se expresan en pixeles.
	pageX, pageY	Devuelven las coordenadas en que se encontraba el puntero del ratón cuando se disparó el evento. Las coordenadas están referidas a la esquina superior izquierda del documento, que pueden ser distintas a las de la ventana si el usuario ha hecho scroll sobre el documento.
Propiedad para detectar el botón del mouse pulsado	button	Normalmente empleado para el evento mouseup (liberación de botón del ratón) para detectar cuál ha sido el botón pulsado. Contiene un valor numérico: 0 para click normal (botón izquierdo), 1 para botón central (botón en el scroll), 2 para botón auxiliar (botón derecho).

Propiedades relacionadas con el teclado	Para determinar qué tecla ha sido pulsada	Lo estudiaremos por separado en la siguiente entrega del curso
Propiedades relacionadas con drag and drop	Algunas no estandarizadas	dataTransfer, dropEffect, effectAllowed, files, types

Eventos con comportamiento automático y el Objeto Event

Hay elementos en HTML que ya vienen con un comportamiento prederminado al momento de registrarle eventos. La etiqueta <a>, por ejemplo, de forma predeterminada, redirige a lo indicado en el atributo href.

Vamos a agregar un evento de tipo click a un elemento <a> de HTML:

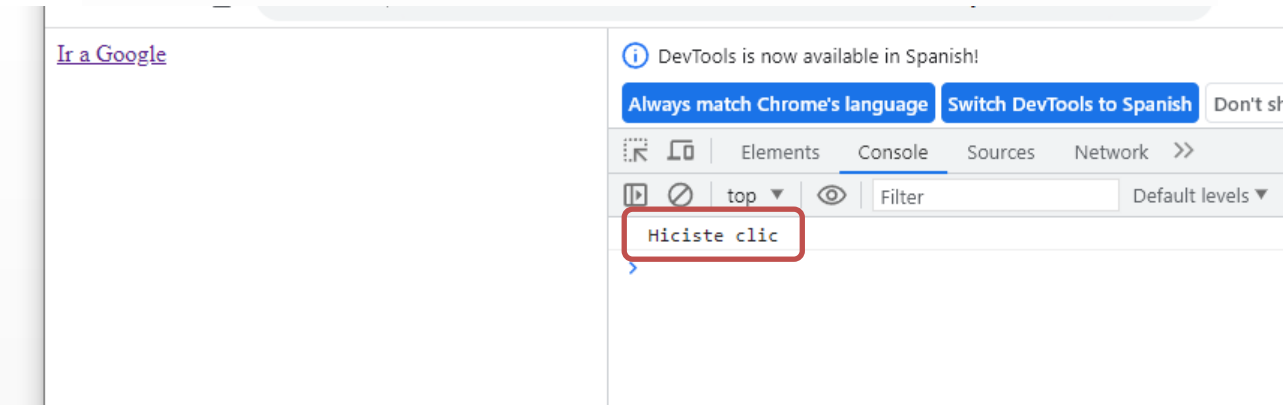
```
<a href="https://www.google.com" id="link">Ir a Google</a>
```

Y le asignamos un evento de tipo clic a ese elemento:

```
let enlace = document.querySelector('#link')

enlace.addEventListener("click", function() {
    console.log("Hiciste clic")
})
```

Al darle clic a ese elemento desde el navegador, va a ejecutar el console.log y luego va a referenciar a la ruta indicada en href:



En el caso que querramos agregar una confirmación en ese comportamiento o si queremos modificar el comportamiento predeterminado, tenemos que utilizar el Objeto Evento.

Para esto usaremos un método de este objeto que es el método preventDefault(). Este método nos sirve para cancelar un comportamiento predeterminado que tenga un evento dentro del DOM.

```
enlace.addEventListener("click", function(e) {
    e.preventDefault()
    console.log("Hiciste clic")
})
```

Específicamente en esta etiqueta <a>, cancela el comportamiento predeterminado de redireccionamiento de href y le asignamos con console.log un nuevo comportamiento. De esta forma cancelamos comportamientos predeterminados en un evento de Javascript.

EventTarget y DOM

En el siguiente ejemplo, añadimos una función para dar la bienvenida al usuario. De esta forma, al hacer click en el botón, se ejecutarán las funciones asociadas. Esto es muy útil para desacoplar y dividir el programa:

Archivo .html

```
<div>
    <input type="text" id="nombre" placeholder="Indicá tu nombre">
    <button id="bienvenida">Saludar</button>
</div>
```

Archivo .js

```
const nombre = document.querySelector('#nombre')

const bienvenida = document.querySelector('#bienvenida')

bienvenida.addEventListener('click', function() {
        const valor = nombre.value
        alert('Bienvenido al sitio: '+valor)
    })
```

Resumen

En esta Unidad…

Trabajamos con Eventos y formularios

En la próxima Unidad…

Trabajaremos con AJAX