### **Tipos de Eventos en JavaScript**

Los eventos en JavaScript son muy variados y cubren una amplia gama de interacciones del usuario con la página. Aquí te presento algunos de los tipos más comunes:

* **Eventos del ratón:**
* click: Ocurre cuando se hace clic en un elemento.
* mouseover: Ocurre cuando el cursor del ratón pasa sobre un elemento.
* mouseout: Ocurre cuando el cursor del ratón se aleja de un elemento.
* mousemove: Ocurre cuando el ratón se mueve sobre un elemento.
* **Eventos del teclado:**
* keydown: Ocurre cuando se presiona una tecla.
* keyup: Ocurre cuando se suelta una tecla.
* keypress: Ocurre mientras se mantiene una tecla presionada.
* **Eventos de formulario:**
* submit: Ocurre cuando se envía un formulario.
* change: Ocurre cuando cambia el valor de un elemento de formulario.
* input: Ocurre cuando el usuario introduce texto en un elemento de formulario.
* **Eventos de la ventana:**
* load: Ocurre cuando la página se ha cargado completamente.
* resize: Ocurre cuando cambia el tamaño de la ventana.
* scroll: Ocurre cuando se desplaza la página.
* **Eventos del DOM:**
* DOMContentLoaded: Ocurre cuando el DOM (Document Object Model) está completamente cargado y parseado, pero antes de que se hayan cargado los recursos externos (imágenes, scripts, etc.).

### **Propagación de Eventos (Burbuja y Captura)**

La propagación de eventos describe cómo se propaga un evento a través del DOM. Hay dos fases principales:

* **Fase de captura:** El evento se propaga desde el elemento más externo hacia el elemento más interno.
* **Fase de burbuja:** El evento se propaga desde el elemento más interno hacia el elemento más externo.

Esta propagación permite que los eventos se manejen en múltiples niveles del DOM.

### **Creando Eventos Personalizados**

Aunque JavaScript no proporciona una forma directa de crear eventos personalizados, puedes simularlos utilizando el método dispatchEvent(). Este método permite disparar un evento en un elemento específico.

**Ejemplo:**

JavaScript

// Crear un evento personalizadoconst miEvento = new CustomEvent('miEventoPersonalizado', { detail: { mensaje: 'Hola' } }); // Despachar el evento en un elementodocument.getElementById('miElemento').dispatchEvent(miEvento);

Use code [with caution.](file:///C:/faq#coding)

### **Otros Conceptos Importantes**

* **Prevención de la acción por defecto:** Puedes evitar que se ejecute la acción por defecto de un evento utilizando preventDefault().
* **Delegación de eventos:** Puedes asignar un manejador de eventos a un elemento padre y luego determinar el elemento objetivo dentro de ese padre.
* **Eventos compuestos:** Los eventos compuestos son eventos personalizados que combinan múltiples eventos simples.

### **1. Validación de formularios en tiempo real:**

* **Ejemplo:** Al escribir en un campo de un formulario, se puede verificar si el formato es correcto (por ejemplo, si un correo electrónico tiene el formato adecuado). Si no es así, se muestra un mensaje de error.
* **Código:**

JavaScript

const emailInput = document.getElementById('email'); emailInput.addEventListener('input', () => { const emailRegex = /^[^\s@]+@[^\s@]+\.[^\s@]+$/; if (!emailRegex.test(emailInput.value)) { // Mostrar un mensaje de error alert('Por favor, ingresa un correo electrónico válido'); } });

Use code [with caution.](file:///C:/faq#coding)

### **2. Creación de un acordeón:**

* **Ejemplo:** Al hacer clic en un encabezado, se muestra o oculta el contenido correspondiente.
* **Código:**

JavaScript

const acordeones = document.querySelectorAll('.acordeon'); acordeones.forEach(acordeon => { acordeon.addEventListener('click', () => { acordeon.classList.toggle('activo'); }); });

Use code [with caution.](file:///C:/faq#coding)

### **3. Creación de un carrusel de imágenes:**

* **Ejemplo:** Al hacer clic en los botones "siguiente" y "anterior", se muestra la siguiente o anterior imagen de un conjunto.
* **Código:**

JavaScript

const imagenes = document.querySelectorAll('.imagen'); let indiceImagenActual = 0; const siguiente = () => { imagenes[indiceImagenActual].classList.remove('activo'); indiceImagenActual = (indiceImagenActual + 1) % imagenes.length; imagenes[indiceImagenActual].classList.add('activo'); }; // ... Código similar para el botón anterior

Use code [with caution.](file:///C:/faq#coding)

### **4. Drag and drop:**

* **Ejemplo:** Permitir al usuario arrastrar y soltar elementos dentro de una página.
* **Código:**

JavaScript

const elementoArrastrable = document.getElementById('arrastrable'); elementoArrastrable.addEventListener('dragstart', (event) => { event.dataTransfer.setData('text/plain', event.target.id); }); const zonaDeDeposito = document.getElementById('zonaDeDeposito'); zonaDeDeposito.addEventListener('drop', (event) => { event.preventDefault(); const datos = event.dataTransfer.getData('text/plain'); zonaDeDeposito.appendChild(document.getElementById(datos)); });

Use code [with caution.](file:///C:/faq#coding)

### **5. Creación de un juego simple:**

* **Ejemplo:** Un juego de adivinar un número donde el usuario ingresa un número y el programa indica si es mayor, menor o igual al número secreto.
* **Código:**

JavaScript

const inputNumero = document.getElementById('numero'); inputNumero.addEventListener('input', () => { const numeroUsuario = parseInt(inputNumero.value); // ... Lógica para comparar con el número secreto y mostrar un mensaje });

Use code [with caution.](file:///C:/faq#coding)

### **6. Integración con APIs:**

* **Ejemplo:** Al enviar un formulario, se realiza una petición a una API para guardar los datos.
* **Código:**

