Guía de Estudio: Formularios Web con JavaScript

Manejo interactivo, eventos y manipulación de datos en formularios HTML5 con JavaScript moderno (ES6+).

1. Conceptos Clave

CONCEPTO A

Eventos y Flujo de Control (event.preventDefault())

Por defecto, al enviar un formulario (`<form>`), el navegador recarga la página enviando la petición por HTTP. En aplicaciones web modernas (SPAs/AJAX), se intercepta el evento `submit` y se ejecuta `event.preventDefault()` para evitar la recarga.

```
<!DOCTYPE html>
<html lang="es">
<head>
  <meta charset="UTF-8">
  <title>Concepto A - PreventDefault</title>
</head>
<body>
  <form id="miFormulario">
    <input type="text" placeholder="Escribe algo..." required>
    <button type="submit">Enviar</button>
  </form>

  <script>
    const form = document.querySelector('#miFormulario');
    form.addEventListener('submit', (event) => {
      event.preventDefault(); // Detiene la recarga de la página
      console.log('Formulario interceptado con éxito.');
    });
  </script>
</body>
</html>
```

CONCEPTO B

Captura de Valores Individuales

Para leer datos de los controles:

- **Campos de texto y Select:** Se lee mediante la propiedad `.value`.
- **Checkboxes y Radios:** Se verifica la propiedad booleana `.checked`.

```
<!DOCTYPE html>
<html lang="es">
<head>
  <meta charset="UTF-8">
  <title>Concepto B - Captura de Valores</title>
</head>
<body>
  <input type="text" id="campoTexto" value="Hola Mundo">
  <input type="checkbox" id="terminos" checked> Acepto términos
  <button id="btnLeer">Leer Valores</button>

  <script>
    document.getElementById('btnLeer').addEventListener('click', () => {
      const texto = document.getElementById('campoTexto').value;
      const aceptado = document.getElementById('terminos').checked;
      console.log('Texto:', texto);
      console.log('Aceptado:', aceptado);
    });
  </script>
</body>
</html>
```

CONCEPTO C

Selección de Radio Buttons

Como varios botones de radio comparten el atributo `name` para formar un grupo, se utiliza `querySelector` combinando el atributo y el pseudo-selector `:checked` con encadenamiento opcional (`?.`) para evitar errores si no hay selección.

```
<!DOCTYPE html>
<html lang="es">
<head>
  <meta charset="UTF-8">
  <title>Concepto C - Radio Buttons</title>
</head>
<body>
  <label><input type="radio" name="opcion" value="Opción 1"> Opción 1</label>
  <label><input type="radio" name="opcion" value="Opción 2" checked> Opción 2</label>
  <button id="btnRadio">Ver Selección</button>

  <script>
    document.getElementById('btnRadio').addEventListener('click', () => {
      const seleccion = document.querySelector('input[name="opcion"]:checked')?.value;
      console.log('Radio seleccionado:', seleccion ?? 'Ninguno');
    });
  </script>
</body>
</html>
```

CONCEPTO D

Eventos: input vs change

- `input`: Se dispara inmediatamente cada vez que el valor cambia (por ejemplo, con cada tecla pulsada en un input o textarea).
- `change`: Se dispara solo cuando el cambio se confirma y el elemento pierde el foco (o al seleccionar una nueva opción en un `<select>`).

```
<!DOCTYPE html>
<html lang="es">
<head>
  <meta charset="UTF-8">
  <title>Concepto D - input vs change</title>
</head>
<body>
  <input type="text" id="inputRealTime" placeholder="Evento input...">
  <select id="selectChange">
    <option value="JS">JavaScript</option>
    <option value="PY">Python</option>
  </select>

  <script>
    document.getElementById('inputRealTime').addEventListener('input', (e) => {
      console.log('Escribiendo (input):', e.target.value);
    });

    document.getElementById('selectChange').addEventListener('change', (e) => {
      console.log('Confirmado (change):', e.target.value);
    });
  </script>
</body>
</html>
```

CONCEPTO E

Uso del Objeto FormData

Permite extraer masivamente todos los pares clave-valor de un formulario de forma automática. Requisito fundamental: cada control debe poseer el atributo `name`. Con `Object.fromEntries(formData.entries())` se convierte fácilmente a un objeto JavaScript simple.

```
<!DOCTYPE html>
<html lang="es">
<head>
  <meta charset="UTF-8">
  <title>Concepto E - FormData</title>
</head>
<body>
  <form id="formRegistro">
    <input type="text" name="usuario" value="Carlos">
    <input type="email" name="correo" value="carlos@example.com">
    <button type="submit">Procesar</button>
  </form>

  <script>
    document.getElementById('formRegistro').addEventListener('submit', (e) => {
      e.preventDefault();
      const formData = new FormData(e.target);
      const datos = Object.fromEntries(formData.entries());
      console.log('Objeto consolidado:', datos);
    });
  </script>
</body>
</html>
```

CONCEPTO F

Archivos y Estado de Controles

- **Input File:** Los archivos seleccionados se obtienen mediante la propiedad `.files`, que devuelve una colección de tipo `FileList`.
- **Deshabilitar elementos:** Se asigna `.disabled = true` en el botón de envío para evitar múltiples clics accidentales mientras se procesa una solicitud.

```
<!DOCTYPE html>
<html lang="es">
<head>
  <meta charset="UTF-8">
  <title>Concepto F - Archivos y Estado</title>
</head>
<body>
  <input type="file" id="miArchivo">
  <button id="btnEnviar">Enviar Archivo</button>

  <script>
    const inputFile = document.getElementById('miArchivo');
    const btn = document.getElementById('btnEnviar');

    btn.addEventListener('click', () => {
      if (inputFile.files.length > 0) {
        btn.disabled = true;
        btn.textContent = 'Subiendo...';
        console.log('Archivo listo:', inputFile.files[0].name);
      }
    });
  </script>
</body>
</html>
```

2. 10 Ejercicios Prácticos

Ejercicio 1: Bloqueo de Envío

Crea un formulario e intercepta el evento `submit` para evitar que la página se recargue, mostrando un mensaje en la consola.

```
<!DOCTYPE html>
<html lang="es">
<head>
  <meta charset="UTF-8">
  <title>Ejercicio 1 - Bloqueo de Envío</title>
</head>
<body>
  <form id="formBloqueo">
    <label for="nombre">Nombre:</label>
    <input type="text" id="nombre" name="nombre" required>
    <button type="submit">Enviar Formulario</button>
  </form>

  <script>
    const form = document.getElementById('formBloqueo');
    form.addEventListener('submit', (event) => {
      event.preventDefault(); // Evita la recarga automática
      console.log('¡Envío interceptado con éxito! La página no se ha recargado.');
    });
  </script>
</body>
</html>
```

Ejercicio 2: Lectura Simple

Lee el valor de un `<input id="email">` al presionar un botón y muéstralo con un `alert()`.

```
<!DOCTYPE html>
<html lang="es">
<head>
  <meta charset="UTF-8">
  <title>Ejercicio 2 - Lectura Simple</title>
</head>
<body>
  <label for="email">Correo Electrónico:</label>
  <input type="email" id="email" placeholder="ejemplo@correo.com">
  <button id="btnMostrar">Mostrar Email</button>

  <script>
    const btn = document.getElementById('btnMostrar');
    btn.addEventListener('click', () => {
      const emailValue = document.getElementById('email').value;
      alert(`El correo ingresado es: ${emailValue}`);
    });
  </script>
</body>
</html>
```

Ejercicio 3: Radio Buttons

Crea 3 opciones de radio con `name="prioridad"`. Muestra la opción seleccionada al hacer clic en un botón.

```
<!DOCTYPE html>
<html lang="es">
<head>
  <meta charset="UTF-8">
  <title>Ejercicio 3 - Radio Buttons</title>
</head>
<body>
  <p>Selecciona la prioridad:</p>
  <label><input type="radio" name="prioridad" value="Baja"> Baja</label><br>
  <label><input type="radio" name="prioridad" value="Media" checked> Media</label><br>
  <label><input type="radio" name="prioridad" value="Alta"> Alta</label><br><br>
  <button id="btnPrioridad">Obtener Prioridad</button>

  <script>
    document.getElementById('btnPrioridad').addEventListener('click', () => {
      const seleccion = document.querySelector('input[name="prioridad"]:checked');
      if (seleccion) {
        alert(`Prioridad seleccionada: ${seleccion.value}`);
      } else {
        alert('Por favor, selecciona una opción.');
      }
    });
  </script>
</body>
</html>
```

Ejercicio 4: Validación Checkbox

Valida que un checkbox de "Términos y Condiciones" esté activado antes de permitir la simulación de registro.

```
<!DOCTYPE html>
<html lang="es">
<head>
  <meta charset="UTF-8">
  <title>Ejercicio 4 - Validación Checkbox</title>
</head>
<body>
  <form id="formRegistro">
    <label>
      <input type="checkbox" id="terminos"> Acepto los Términos y Condiciones
    </label><br><br>
    <button type="submit">Registrarse</button>
    <p id="mensaje" style="color: black;"></p>
  </form>

  <script>
    document.getElementById('formRegistro').addEventListener('submit', (e) => {
      e.preventDefault();
      const terminosAceptados = document.getElementById('terminos').checked;
      const mensaje = document.getElementById('mensaje');

      if (!terminosAceptados) {
        mensaje.textContent = 'Debes aceptar los términos y condiciones para continuar.';
      } else {
        mensaje.textContent = '¡Registro completado exitosamente!';
      }
    });
  </script>
</body>
</html>
```

Ejercicio 5: Detección en Select

Usa el evento `change` en un `<select>` de países para imprimir en consola cada vez que el usuario elija una opción nueva.

```
<!DOCTYPE html>
<html lang="es">
<head>
  <meta charset="UTF-8">
  <title>Ejercicio 5 - Detección en Select</title>
</head>
<body>
  <label for="paises">Selecciona un país:</label>
  <select id="paises">
    <option value="">-- Selecciona --</option>
    <option value="MX">México</option>
    <option value="CO">Colombia</option>
    <option value="ES">España</option>
    <option value="AR">Argentina</option>
  </select>

  <script>
    const selectPaises = document.getElementById('paises');
    selectPaises.addEventListener('change', (event) => {
      const paisSeleccionado = event.target.value;
      console.log(`Nuevo país seleccionado: ${paisSeleccionado}`);
    });
  </script>
</body>
</html>
```

Ejercicio 6: Contador en Tiempo Real

Crea un `<textarea>` que actualice un contador de caracteres dinámicamente usando el evento `input`.

```
<!DOCTYPE html>
<html lang="es">
<head>
  <meta charset="UTF-8">
  <title>Ejercicio 6 - Contador en Tiempo Real</title>
</head>
<body>
  <label for="comentario">Escribe tu comentario:</label><br>
  <textarea id="comentario" rows="4" cols="40" maxlength="200"></textarea>
  <p>Caracteres: <span id="contador">0</span>/200</p>

  <script>
    const textarea = document.getElementById('comentario');
    const contador = document.getElementById('contador');

    textarea.addEventListener('input', () => {
      const cantidad = textarea.value.length;
      contador.textContent = cantidad;
    });
  </script>
</body>
</html>
```

Ejercicio 7: Extracción con FormData

Construye un formulario con campos nombre, email y edad. Procesa y consolida sus datos con `FormData` al enviarlo.

```
<!DOCTYPE html>
<html lang="es">
<head>
  <meta charset="UTF-8">
  <title>Ejercicio 7 - Extracción con FormData</title>
</head>
<body>
  <form id="formUsuario">
    <input type="text" name="nombre" placeholder="Nombre" required><br><br>
    <input type="email" name="email" placeholder="Email" required><br><br>
    <input type="number" name="edad" placeholder="Edad" required><br><br>
    <button type="submit">Consolidar Datos</button>
  </form>

  <script>
    document.getElementById('formUsuario').addEventListener('submit', (e) => {
      e.preventDefault();
      const formData = new FormData(e.target);
      const datosObjeto = Object.fromEntries(formData.entries());
      console.log('Objeto resultante de FormData:', datosObjeto);
    });
  </script>
</body>
</html>
```

Ejercicio 8: Manejo de Archivos

Captura un `<input type="file">` mediante el evento `change` e imprime en consola el nombre (`file.name`) y tamaño (`file.size`) del archivo.

```
<!DOCTYPE html>
<html lang="es">
<head>
  <meta charset="UTF-8">
  <title>Ejercicio 8 - Manejo de Archivos</title>
</head>
<body>
  <label for="archivo">Selecciona un archivo:</label>
  <input type="file" id="archivo">

  <script>
    const fileInput = document.getElementById('archivo');
    fileInput.addEventListener('change', (e) => {
      const files = e.target.files;
      if (files.length > 0) {
        const archivo = files[0];
        console.log(`Nombre del archivo: ${archivo.name}`);
        console.log(`Tamaño: ${archivo.size} bytes (${(archivo.size / 1024).toFixed(2)} KB)`);
      }
    });
  </script>
</body>
</html>
```

Ejercicio 9: Botón Deshabilitado

Haz que al enviar un formulario, el botón de envío se deshabilite (`disabled = true`) y cambie su texto a "Enviando...".

```
<!DOCTYPE html>
<html lang="es">
<head>
  <meta charset="UTF-8">
  <title>Ejercicio 9 - Botón Deshabilitado</title>
</head>
<body>
  <form id="formPago">
    <input type="text" placeholder="Número de tarjeta" required><br><br>
    <button type="submit" id="btnEnviar">Pagar Ahora</button>
  </form>

  <script>
    const form = document.getElementById('formPago');
    const btn = document.getElementById('btnEnviar');

    form.addEventListener('submit', (e) => {
      e.preventDefault();
      btn.disabled = true;
      btn.textContent = 'Enviando...';

      // Simulación de respuesta de servidor tras 2 segundos
      setTimeout(() => {
        console.log('Proceso completado.');
        btn.disabled = false;
        btn.textContent = 'Pagar Ahora';
      }, 2000);
    });
  </script>
</body>
</html>
```

Ejercicio 10: Envío Programático

Crea un botón fuera del formulario que valide que un campo no esté vacío y active el envío del formulario usando `.submit()`.

```
<!DOCTYPE html>
<html lang="es">
<head>
  <meta charset="UTF-8">
  <title>Ejercicio 10 - Envío Programático</title>
</head>
<body>
  <form id="formExterno" action="#" method="GET">
    <label for="codigo">Código de acceso:</label>
    <input type="text" id="codigo" name="codigo">
  </form>

  <br>
  <!-- Botón fuera del formulario -->
  <button id="btnExterno">Validar y Enviar desde afuera</button>

  <script>
    const form = document.getElementById('formExterno');
    const inputCodigo = document.getElementById('codigo');
    const btnExterno = document.getElementById('btnExterno');

    btnExterno.addEventListener('click', () => {
      if (inputCodigo.value.trim() === '') {
        alert('Error: El campo no puede estar vacío.');
      } else {
        alert('Validación correcta. Enviando formulario...');
        form.submit(); // Llama al método submit del elemento form
      }
    });
  </script>
</body>
</html>
```