

Universidad de Guadalajara

CENTRO UNIVERSITARIO DE CIENCIAS EXACTAS E INGENIERÍAS



Ingeniería Informática.

Actividad Integradora 1: Agregación de eventos de forma dinámica

Desarrollo de Front End.

Yosef Sánchez Gutiérrez
Marco Alejandro González Mireles
Mirella Stephania Palomera Gómez
Roberto Carlos Martínez Aviña

Mtra. Rosalia Iñiguez .

Introducción

En esta actividad se desarrolló un formulario web interactivo utilizando HTML, CSS y JavaScript, con el objetivo de aplicar y reforzar los conocimientos adquiridos sobre el DOM (Document Object Model), selectores, eventos y la manipulación dinámica de estilos en el navegador, así como conocer los eventos que se disparan de forma dinámica en una página Web. El formulario simula un sistema de registro de futbolistas, permitiendo validar los datos introducidos en tiempo real, proporcionando retroalimentación visual inmediata al usuario, así como borrar un elemento insertado, o bien, el listado de elementos completo.

El objetivo principal fue implementar la gestión dinámica de elementos en el DOM mediante el uso de eventos con `addEventListener`, permitiendo agregar y eliminar elementos de forma interactiva, y mantener sincronizados los cambios con el almacenamiento local sin recargar la página.

Desarrollo de la Actividad

Estructura del Formulario HTML

El formulario permite ingresar los siguientes datos de un futbolista:

- Nombre completo
- Nacionalidad
- Edad
- Descripción
- Posición en el campo (desplegable)
- Habilidades (casillas de verificación)

Se incluyó un botón para activar la validación sin enviar el formulario, lo que permite controlar completamente la experiencia del usuario desde JavaScript.

El diseño de la estructura fue enfocado en la accesibilidad, claridad y facilidad de uso, permitiendo una interacción fluida y amigable.

Formulario de Registro de Futbolista

Nombre completo:

Ej. Lionel Messi

Número (dorsal):

Ej. 10

Fecha de nacimiento:

dd/mm/aaaa

Lugar de nacimiento:

Ej. Rosario, Argentina

Nacionalidad:

Ej. Argentina

Altura (cm):

Ej. 170

Peso (kg):

Ej. 72

Pie dominante:

-- Selecciona una opción --

Posición en el campo:

-- Selecciona una posición --

Habilidades destacadas:

☐ Velocidad

☐ Regate

☐ Pase

☐ Tiro

☐ Defensa

☐ Aceleración

☐ Sprint

☐ Resistencia

☐ Salto

☐ Reflejos

Habilidades destacadas:

☐ Velocidad

☐ Regate

☐ Pase

☐ Tiro

☐ Defensa

☐ Aceleración

☐ Sprint

☐ Resistencia

☐ Salto

☐ Reflejos

☐ Definición

☐ Remate lejano

Equipos anteriores:

Ej. FC Barcelona, PSG, Inter Miami

Descripción general del jugador:

Describe estilo, trayectoria, puntos fuertes, etc.

Registrar Futbolista

Catálogo de Futbolistas Registrados

Catálogo de Futbolistas Registrados

Vaciar Lista Completa

LIONEL MESSI

#10

Nacimiento:

1987-06-24 (Rosario, Argentina)

Nacionalidad:

Argentina

Altura:

170 cm | Peso: 72 kg

Pie dominante:

izquierdo

Posición:

ED

Habilidades:

velocidad, regate, pase, tiro, aceleración, sprint, reflejos, definición, rematelejano

Equipos anteriores:

Barcelona FC, PSG, Inter Miami

Descripción:

El mejor jugador de todos los tiempos

Eliminar

Código CSS

En el archivo styles.css se mantuvo el enfoque en un diseño limpio, moderno y profesional, con mejoras que optimizan la experiencia visual y la usabilidad del formulario web.

Se conservaron las bases del diseño anterior con bordes redondeados, fondo blanco y sombras suaves para dar profundidad, pero se agregaron detalles que mejoran la organización y la adaptabilidad en distintos dispositivos mediante el uso de CSS Grid para el manejo de las casillas de verificación de habilidades.

La tipografía sigue siendo clara y legible, usando fuentes sans-serif modernas para facilitar la lectura.

La paleta de colores se mantiene consistente, con:

Tonos de azul brillante para botones y elementos destacados que aportan identidad visual y contraste.

Grises suaves para fondos y campos de entrada que generan un ambiente neutro y profesional.

Detalles en amarillo para acentos secundarios en encabezados.

Se añadieron transiciones suaves en los focos y en los botones para mejorar la respuesta visual a la interacción del usuario.

Para la validación visual, se mantienen y refuerzan las clases .error y .success, que aplican bordes y sombras de colores rojo y verde respectivamente, brindando retroalimentación clara e inmediata.

Además, se definieron estilos específicos para las tarjetas dinámicas que representan a cada futbolista, incluyendo efectos hover para hacerlas más interactivas, distribución en flexbox para ordenar imagen y detalles, y estilos responsivos para adaptar la visualización en dispositivos móviles.

Se diseñaron botones específicos para la interacción de eliminación:

- El botón de borrar registro individual posee un estilo destacado en rojo, ubicado en la esquina inferior derecha de cada tarjeta, con efecto hover que intensifica el color para indicar acción de borrado.

- Estas mejoras visuales y estructurales hacen que la interfaz sea más amigable, accesible y moderna, facilitando la interacción y lectura tanto en escritorio como en pantallas pequeñas, además de ofrecer un control intuitivo y rápido para la gestión de los registros visualizados.



```

1 // Seleccionamos el boton de "Registrar" y le agregamos un evento click
2 document.getElementById("register").addEventListener("click", () => {
3     let validos = true
4
5     // Creamos de nuevo a validar (para depuraciones)
6     const camposValidos = [
7         "nombre", "apellido", "fecha_nacimiento", "lugar_nacimiento",
8         "email_usuario", "password", "confirm_password", "telefono_numeros"
9     ]
10
11     // Recorremos campo de inicio, numero a fecha
12     camposValidos.forEach((id) => {
13         const campo = document.getElementById(id)
14         if (campo.value.trim() || !campo.classList.add("error"))
15             campo.classList.remove("success")
16             //Valido = false
17         } else {
18             campo.classList.remove("error")
19             campo.classList.add("success")
20         }
21     })
22
23     // Validacion superflua para campos numericos
24     const camposNumericos = ["Apellido", "Apellido", "Pais"]
25     camposNumericos.forEach((id) => {
26         const campo = document.getElementById(id)
27         const valor = Number.parseInt(campo.value)
28         if (isNaN(valor) || valor == 0) {
29             campo.classList.add("error")
30             campo.classList.remove("success")
31         }
32         //validos = false
33         Swal.fire({ icon: "error", title: `Campo ${id} debe ser un número positivo mayor a cero`, "warning"})
34     })
35
36     // Validación para campos de texto que no deben contener números
37     const camposTextosNumeros = ["Nombre", "Lugar_nacimiento", "Nacionalidad"]
38     const regexSoloTexto = /^[a-zA-Z\s\d\.,-]*$/
39
40     camposTextosNumeros.forEach((id) => {
41         const campo = document.getElementById(id)
42         if (campo.value.trim() && !regexSoloTexto.test(campo.value)) {
43             campo.classList.add("error")
44             campo.classList.remove("success")
45         }
46         //validos = false
47         Swal.fire({ icon: "error", title: `Campo ${id} solo debe contener texto`, "warning"})
48     })
49
50 })
51
52 const pla = document.getElementById("pla")
53 const decision = document.getElementById("decision")
54 const habilidadesseleccionadas = document.querySelectorAll('input[name="habilidades"]').checkboxes
55 const seccionCheckBox = document.querySelector('input[name="habilidades"]').parentElement
56
57 if (pla.value == "") {
58     pla.classList.add("error")
59     pla.classList.remove("success")
60     //validos = false
61 } else {
62     pla.classList.remove("error")
63     pla.classList.add("success")
64 }
65
66 if (decision.value == "") {
67     decision.classList.add("error")
68     decision.classList.remove("success")
69     //validos = false

```

```

10 // Agregar al array de jugadores
11 document.getElementById("AgregarJugador").addEventListener("click", () => {
12     // Validar campos
13     if (validarCampos()) {
14         // Crear jugador
15         const jugador = {
16             nombre: document.getElementById("nombre").value,
17             apellido: document.getElementById("apellido").value,
18             fechaNacimiento: document.getElementById("fechaNacimiento").value,
19             nacionalidad: document.getElementById("nacionalidad").value,
20             altura: document.getElementById("altura").value,
21             peso: document.getElementById("peso").value,
22             pie: document.getElementById("pie").value,
23             posicion: document.getElementById("posicion").value,
24             habilidades: Array.from(document.querySelectorAll("input")).map((input) => ({
25                 habilidad: input.value,
26                 valor: document.getElementById(input.value).value,
27                 descripcion: document.getElementById("descripcion").value,
28             })),
29             id: Date.now(),
30         };
31         // Guardar en localStorage
32         const lista = JSON.parse(localStorage.getItem("jugadores")) || [];
33         lista.push(jugador);
34         localStorage.setItem("jugadores", JSON.stringify(lista));
35         // Agregar tarjeta
36         agregarTarjeta(jugador);
37         // Mostrar mensaje de éxito
38         Swal.fire({
39             icon: "success",
40             title: "Jugador agregado al catálogo.",
41             text: "¡Listo!",
42             duration: 2000,
43         });
44     } else {
45         // Mostrar mensaje de error
46         Swal.fire({
47             icon: "error",
48             title: "Campos incompletos.",
49             text: "Por favor completa todos los campos obligatorios.",
50             duration: 2000,
51         });
52     }
53 });
54
55 // Función para agregar una tarjeta
56 function agregarTarjeta(jugador) {
57     // Crear el contenedor
58     const contenedor = document.getElementById("listaJugadores");
59     // Crear el elemento de tarjeta
60     const tarjeta = document.createElement("div");
61     tarjeta.classList.add("tarjeta-jugador");
62     tarjeta.setAttribute("id", `id-${jugador.id}`);
63     // Crear la sección de datos
64     const tarjetaContenido = document.createElement("div");
65     tarjetaContenido.classList.add("tarjeta-contenido");
66     // Crear la sección de imagen
67     const jugadorImagen = document.createElement("img");
68     jugadorImagen.classList.add("jugador-imagen");
69     // Crear la sección de habilidades
70     const jugadorHabilidades = document.createElement("div");
71     jugadorHabilidades.classList.add("jugador-habilidades");
72     // Crear la sección de estadísticas
73     const jugadorEstadisticas = document.createElement("div");
74     jugadorEstadisticas.classList.add("jugador-estadisticas");
75     // Crear la sección de acciones
76     const jugadorAcciones = document.createElement("div");
77     jugadorAcciones.classList.add("jugador-acciones");
78     // Agregar los elementos a la tarjeta
79     tarjeta.appendChild(tarjetaContenido);
80     tarjeta.appendChild(jugadorImagen);
81     tarjeta.appendChild(jugadorHabilidades);
82     tarjeta.appendChild(jugadorEstadisticas);
83     tarjeta.appendChild(jugadorAcciones);
84     // Agregar la tarjeta al contenedor
85     contenedor.appendChild(tarjeta);
86 }
87
88 // Función para validar los campos
89 function validarCampos() {
90     // Validar nombre
91     const nombre = document.getElementById("nombre").value;
92     if (nombre.length < 3) {
93         return false;
94     }
95     // Validar apellido
96     const apellido = document.getElementById("apellido").value;
97     if (apellido.length < 3) {
98         return false;
99     }
100     // Validar fecha de nacimiento
101     const fechaNacimiento = document.getElementById("fechaNacimiento").value;
102     if (!fechaNacimiento) {
103         return false;
104     }
105     // Validar nacionalidad
106     const nacionalidad = document.getElementById("nacionalidad").value;
107     if (!nacionalidad) {
108         return false;
109     }
110     // Validar altura
111     const altura = document.getElementById("altura").value;
112     if (!altura) {
113         return false;
114     }
115     // Validar peso
116     const peso = document.getElementById("peso").value;
117     if (!peso) {
118         return false;
119     }
120     // Validar pie
119     const pie = document.getElementById("pie").value;
120     if (!pie) {
121         return false;
122     }
123     // Validar posicion
124     const posicion = document.getElementById("posicion").value;
125     if (!posicion) {
126         return false;
127     }
128     // Validar habilidades
129     const habilidades = document.querySelectorAll("input");
130     for (const habilidad of habilidades) {
131         if (!habilidad.value) {
132             return false;
133         }
134     }
135     // Validar descripcion
136     const descripcion = document.getElementById("descripcion").value;
137     if (!descripcion) {
138         return false;
139     }
140     return true;
141 }

```

```

10 // Agregar al array de jugadores
11 document.getElementById("AgregarJugador").addEventListener("click", () => {
12     // Validar campos
13     if (validarCampos()) {
14         // Crear jugador
15         const jugador = {
16             nombre: document.getElementById("nombre").value,
17             apellido: document.getElementById("apellido").value,
18             fechaNacimiento: document.getElementById("fechaNacimiento").value,
19             nacionalidad: document.getElementById("nacionalidad").value,
20             altura: document.getElementById("altura").value,
21             peso: document.getElementById("peso").value,
22             pie: document.getElementById("pie").value,
23             posicion: document.getElementById("posicion").value,
24             habilidades: Array.from(document.querySelectorAll("input")).map((input) => ({
25                 habilidad: input.value,
26                 valor: document.getElementById(input.value).value,
27                 descripcion: document.getElementById("descripcion").value,
28             })),
29             id: Date.now(),
30         };
31         // Guardar en localStorage
32         const lista = JSON.parse(localStorage.getItem("jugadores")) || [];
33         lista.push(jugador);
34         localStorage.setItem("jugadores", JSON.stringify(lista));
35         // Agregar tarjeta
36         agregarTarjeta(jugador);
37         // Mostrar mensaje de éxito
38         Swal.fire({
39             icon: "success",
40             title: "Jugador agregado al catálogo.",
41             text: "¡Listo!",
42             duration: 2000,
43         });
44     } else {
45         // Mostrar mensaje de error
46         Swal.fire({
47             icon: "error",
48             title: "Campos incompletos.",
49             text: "Por favor completa todos los campos obligatorios.",
50             duration: 2000,
51         });
52     }
53 });
54
55 // Función para agregar una tarjeta
56 function agregarTarjeta(jugador) {
57     // Crear el contenedor
58     const contenedor = document.getElementById("listaJugadores");
59     // Crear el elemento de tarjeta
60     const tarjeta = document.createElement("div");
61     tarjeta.classList.add("tarjeta-jugador");
62     tarjeta.setAttribute("id", `id-${jugador.id}`);
63     // Crear la sección de datos
64     const tarjetaContenido = document.createElement("div");
65     tarjetaContenido.classList.add("tarjeta-contenido");
66     // Crear la sección de imagen
67     const jugadorImagen = document.createElement("img");
68     jugadorImagen.classList.add("jugador-imagen");
69     // Crear la sección de habilidades
70     const jugadorHabilidades = document.createElement("div");
71     jugadorHabilidades.classList.add("jugador-habilidades");
72     // Crear la sección de estadísticas
73     const jugadorEstadisticas = document.createElement("div");
74     jugadorEstadisticas.classList.add("jugador-estadisticas");
75     // Crear la sección de acciones
76     const jugadorAcciones = document.createElement("div");
77     jugadorAcciones.classList.add("jugador-acciones");
78     // Agregar los elementos a la tarjeta
79     tarjeta.appendChild(tarjetaContenido);
80     tarjeta.appendChild(jugadorImagen);
81     tarjeta.appendChild(jugadorHabilidades);
82     tarjeta.appendChild(jugadorEstadisticas);
83     tarjeta.appendChild(jugadorAcciones);
84     // Agregar la tarjeta al contenedor
85     contenedor.appendChild(tarjeta);
86 }
87
88 // Función para validar los campos
89 function validarCampos() {
90     // Validar nombre
91     const nombre = document.getElementById("nombre").value;
92     if (nombre.length < 3) {
93         return false;
94     }
95     // Validar apellido
96     const apellido = document.getElementById("apellido").value;
97     if (apellido.length < 3) {
98         return false;
99     }
100     // Validar fecha de nacimiento
101     const fechaNacimiento = document.getElementById("fechaNacimiento").value;
102     if (!fechaNacimiento) {
103         return false;
104     }
105     // Validar nacionalidad
106     const nacionalidad = document.getElementById("nacionalidad").value;
107     if (!nacionalidad) {
108         return false;
109     }
110     // Validar altura
111     const altura = document.getElementById("altura").value;
112     if (!altura) {
113         return false;
114     }
115     // Validar peso
116     const peso = document.getElementById("peso").value;
117     if (!peso) {
118         return false;
119     }
120     // Validar pie
121     const pie = document.getElementById("pie").value;
122     if (!pie) {
123         return false;
124     }
125     // Validar posicion
126     const posicion = document.getElementById("posicion").value;
127     if (!posicion) {
128         return false;
129     }
130     // Validar habilidades
131     const habilidades = document.querySelectorAll("input");
132     for (const habilidad of habilidades) {
133         if (!habilidad.value) {
134             return false;
135         }
136     }
137     // Validar descripcion
138     const descripcion = document.getElementById("descripcion").value;
139     if (!descripcion) {
140         return false;
141     }
142     return true;
143 }

```

```

function eliminarFutbolista(f) {
  // Crear botón de eliminar individual
  const btnEliminar = document.createElement("button")
  btnEliminar.classList.add("btn-eliminar")
  btnEliminar.textContent = "✖ Eliminar"

  // Usar addEventListener
  btnEliminar.addEventListener("click", () => {
    eliminarFutbolista(f.id)
  })

  // Ensamblar la tarjeta
  tarjetaContenido.appendChild(jugadorImagen)
  tarjetaContenido.appendChild(jugadorDetalles)
  tarjeta.appendChild(tarjetaContenido)
  tarjeta.appendChild(btnEliminar)

  // Agregar la tarjeta al contenedor
  contenedor.appendChild(tarjeta)
}

function eliminarFutbolista(id) {
  Swal.fire({
    title: "¿Estás seguro?",
    text: "No podrás revertir esta acción",
    icon: "warning",
    showCancelButton: true,
    confirmButtonColor: "#3085d6",
    cancelButtonColor: "#d33",
    confirmButtonText: "Sí, eliminar",
    cancelButtonText: "Cancelar",
  }).then((result) => {
    if (result.isConfirmed) {
      // Eliminar del DOM
      const tarjeta = document.querySelector(`.tarjeta-futbolista[data-id="${id}"]`)
      if (tarjeta) {
        tarjeta.remove()
      }

      // Eliminar del LocalStorage
      let lista = JSON.parse(localStorage.getItem("futbolistas")) || []
      lista = lista.filter((f) => f.id !== id)
      localStorage.setItem("futbolistas", JSON.stringify(lista))

      // Mostrar/ocultar botón de vaciar según si quedan elementos
      mostrarOcultarBotonVaciar()

      Swal.fire("¡Eliminado!", "El futbolista ha sido eliminado.", "success")
    }
  })
}

```

Código para eliminar un registro

```

// Crear botón de eliminar individual
const btnEliminar = document.createElement("button")
btnEliminar.classList.add("btn-eliminar")
btnEliminar.textContent = "✖ Eliminar"

// Usar addEventListener
btnEliminar.addEventListener("click", () => {
  eliminarFutbolista(f.id)
})

// Ensamblar la tarjeta
tarjetaContenido.appendChild(jugadorImagen)
tarjetaContenido.appendChild(jugadorDetalles)
tarjeta.appendChild(tarjetaContenido)
tarjeta.appendChild(btnEliminar)

// Agregar la tarjeta al contenedor
contenedor.appendChild(tarjeta)
}

function eliminarFutbolista(id) {
  Swal.fire({
    title: "¿Estás seguro?",
    text: "No podrás revertir esta acción",
    icon: "warning",
    showCancelButton: true,
    confirmButtonColor: "#3085d6",
    cancelButtonColor: "#d33",
    confirmButtonText: "Sí, eliminar",
    cancelButtonText: "Cancelar",
  }).then((result) => {
    if (result.isConfirmed) {
      // Eliminar del DOM
      const tarjeta = document.querySelector(`.tarjeta-futbolista[data-id="${id}"]`)
      if (tarjeta) {
        tarjeta.remove()
      }

      // Eliminar del LocalStorage
      let lista = JSON.parse(localStorage.getItem("futbolistas")) || []
      lista = lista.filter((f) => f.id !== id)
      localStorage.setItem("futbolistas", JSON.stringify(lista))

      // Mostrar/ocultar botón de vaciar según si quedan elementos
      mostrarOcultarBotonVaciar()

      Swal.fire("¡Eliminado!", "El futbolista ha sido eliminado.", "success")
    }
  })
}

```

Eliminar un registro individual (eliminarFutbolista)

- Esta función recibe el id del futbolista que se quiere eliminar.
- Muestra un cuadro de confirmación usando Swal.fire para que el usuario confirme la acción.

- Si el usuario confirma:
 1. Busca en el DOM la tarjeta del futbolista con ese id y la elimina visualmente (tarjeta.remove()).
 2. Actualiza el localStorage eliminando ese futbolista del arreglo guardado.
 3. Actualiza la visibilidad del botón para vaciar la lista, en caso de que ya no queden elementos.
 4. Muestra una alerta de éxito indicando que el futbolista fue eliminado.

Código para vaciar el listado

```

220 // Vaciar toda la lista
221 function vaciarListaCompleta() {
222   Swal.fire({
223     title: "⚠ ¿Vaciar lista completa?",
224     text: "Se eliminarán TODOS los futbolistas registrados. Esta acción no se puede deshacer.",
225     icon: "warning",
226     showCancelButton: true,
227     confirmButtonColor: "#e74c3c",
228     cancelButtonColor: "#95a5a6",
229     confirmButtonText: "Sí, vaciar todo",
230     cancelButtonText: "Cancelar",
231   }).then((result) => {
232     if (result.isConfirmed) {
233       // Limpiar el DOM
234       const listaFutbolistas = document.getElementById("listaFutbolistas")
235       listaFutbolistas.innerHTML = ""
236
237       // Limpiar localStorage
238       localStorage.removeItem("futbolistas")
239
240       // Ocultar botón de vaciar
241       mostrarOcultarBotonVaciar()
242
243     }
244     Swal.fire({
245       title: "¡Lista vacía!",
246       text: "Todos los futbolistas han sido eliminados.",
247       icon: "success",
248       timer: 2000,
249       showConfirmButton: false,
250     })
251   })
252 }
253

```

Vaciar toda la lista (vaciarListaCompleta)

- Muestra una alerta de confirmación para vaciar todos los futbolistas.
- Si el usuario confirma:
 1. Limpia todo el contenido del contenedor donde se muestran las tarjetas (listaFutbolistas.innerHTML = "").
 2. Borra completamente la lista guardada en localStorage.
 3. Oculta el botón de vaciar la lista, porque ya no hay elementos.
 4. Muestra un mensaje confirmando que la lista fue vaciada exitosamente.

Conclusión

Podemos concluir que en esta actividad se fortaleció la interacción dinámica con el DOM al agregar a la interfaz la funcionalidad para eliminar elementos individuales mediante un botón de borrar que acompaña a cada registro, así como un botón general que permite vaciar completamente el listado de elementos. Se implementó el uso de `addEventListener` para asignar eventos de forma dinámica a los botones de eliminar durante la creación de cada elemento, asegurando una gestión eficiente y flexible del listado. Esto permitió que la interfaz respondiera a las acciones del usuario en tiempo real, mejorando la usabilidad y el control sobre los datos mostrados.

Además, la funcionalidad para vaciar todo el listado facilita la gestión masiva de datos, dejando solo el formulario disponible para nuevas entradas, lo que contribuye a una experiencia de usuario más fluida y organizada.

Esta práctica demuestra cómo la manipulación dinámica del DOM y el manejo de eventos en JavaScript son herramientas claves para construir aplicaciones web interactivas y adaptativas.

Archivos fuente entregados

- `index.html`
- `styles.css`
- `script.js`