

Unidad 7

Ejercicios resueltos



EJERCICIO 1:

GALERÍA DE IMÁGENES CON EVENTOS

Crea una galería de imágenes donde al hacer clic en una miniatura, se muestre la imagen grande en un contenedor principal. Además:

- Al pasar el ratón sobre una miniatura, ésta debe resaltarse (por ejemplo, con un borde o sombra).
- Al salir del ratón, debe volver a su estado normal.
- Al hacer doble clic en la imagen grande, debe abrirse en una nueva pestaña

NOTA: se pueden utilizar imágenes aleatorias de:

<https://picsum.photos/200/150?random=1>



```
<body>
<div id="galeria">
  
  <div id="miniaturas">
    
    
    
  </div>
</div>
</div>
<script>
  const imgGrande = document.getElementById('imagen-grande');
  const miniaturas = document.querySelectorAll('#miniaturas img');
  // Cambiar imagen grande al hacer clic en una miniatura
  miniaturas.forEach(miniatura => {
    miniatura.addEventListener('click', function() {
      // Actualizar imagen grande
      imgGrande.src = this.src;
      imgGrande.alt = this.alt;
      // Quitar clase 'seleccionada' de todas
      miniaturas.forEach(img => img.classList.remove('seleccionada'));
      // Añadir clase 'seleccionada' a la actual
      this.classList.add('seleccionada');
    });
    // Resaltar al pasar el ratón
    miniatura.addEventListener('mouseenter', function() {
      this.style.border = '2px solid #28a745';
    });
    // Quitar resalte al salir
    miniatura.addEventListener('mouseleave', function() {
      if (!this.classList.contains('seleccionada')) {
        this.style.border = '2px solid transparent';
      }
    });
  });
  // Abrir imagen grande en nueva pestaña al hacer doble clic
  imgGrande.addEventListener('dblclick', function() {
    window.open(this.src, '_blank');
  });
</script>
</body>
```

```
<style>
  #imagen-grande {
    width: 400px;
    height: 300px;
    object-fit: cover;
    border: 2px solid #ccc;
    margin-bottom: 10px;
  }
  #miniaturas img {
    width: 80px;
    height: 60px;
    object-fit: cover;
    margin: 0 5px;
    cursor: pointer;
    border: 2px solid transparent;
  }
  #miniaturas img:hover,
  #miniaturas img.seleccionada {
    border-color: #007bff;
  }
</style>
```

EJERCICIO 1 – SOLUCIÓN: GALERÍA DE IMÁGENES CON EVENTOS

EJERCICIO 2: FORMULARIO CON VALIDACIÓN EN TIEMPO REAL

Crea un formulario de registro con campos: nombre, email, contraseña y confirmación de contraseña. Implementa:

- Validación en tiempo real al escribir (input o keyup):
- Nombre: al menos 3 caracteres.
- Email: formato válido (usa una expresión regular simple).
- Contraseña: al menos 8 caracteres, con mayúscula, minúscula y número.
- Confirmación: debe coincidir con la contraseña.
- Muestra mensajes de error debajo de cada campo si no es válido.
- Al enviar el formulario (submit), si hay errores, evita el envío y enfoca el primer campo con error.
- Si todo es válido, muestra un mensaje de éxito y limpia el formulario.

A screenshot of a web registration form. It contains four input fields: 'Nombre:', 'Email:', 'Contraseña:', and 'Confirmar contraseña:'. The 'Nombre:' field contains the text 'be' and has a red error message below it: 'El nombre debe tener al menos 3 caracteres.' The other fields are empty. At the bottom of the form is a 'Registrar' button.

EJERCICIO 2: FORMULARIO CON VALIDACIÓN EN TIEMPO REAL

```

12 <body>
13   <form id="registro">
14     <div class="form-group">
15       <label>Nombre:</label>
16       <input type="text" id="nombre" required>
17       <div id="error-nombre" class="error"></div>
18     </div>
19     <div class="form-group">
20       <label>Email:</label>
21       <input type="email" id="email" required>
22       <div id="error-email" class="error"></div>
23     </div>
24     <div class="form-group">
25       <label>Contraseña:</label>
26       <input type="password" id="password" required>
27       <div id="error-password" class="error"></div>
28     </div>
29     <div class="form-group">
30       <label>Confirmar contraseña:</label>
31       <input type="password" id="confirm-password" required>
32       <div id="error-confirm" class="error"></div>
33     </div>
34     <button type="submit">Registrar</button>
35     <div id="mensaje"></div>
36   </form>
37 </script>
38 const form = document.getElementById('registro');
39 const nombre = document.getElementById('nombre');
40 const email = document.getElementById('email');
41 const password = document.getElementById('password');
42 const confirmPassword = document.getElementById('confirm-password');
43 const mensaje = document.getElementById('mensaje');
44 // Validar nombre (mínimo 3 caracteres)
45 function validarNombre() {
46   const valor = nombre.value.trim();
47   const errorDiv = document.getElementById('error-nombre');
48   if (valor.length < 3) {
49     errorDiv.textContent = 'El nombre debe tener al menos 3 caracteres.';
50     return false;
51   }
52   errorDiv.textContent = '';
53   return true;
54 }
55 // Validar email (formato simple)
56 function validarEmail() {
57   const valor = email.value.trim();
58   const regex = /^[^\s@]+@[^\s@]+\.[^\s@]+$/;
59   const errorDiv = document.getElementById('error-email');
60   if (!regex.test(valor)) {
61     errorDiv.textContent = 'Introduce un email válido.';
62     return false;
63   }
64   errorDiv.textContent = '';
65   return true;
66 }

```

```

67 // Validar contraseña (mínimo 8 caracteres, con mayúscula, minúscula y número)
68 function validarPassword() {
69   const valor = password.value;
70   const errorDiv = document.getElementById('error-password');
71   const tieneMayus = /[A-Z]/.test(valor);
72   const tieneMinus = /[a-z]/.test(valor);
73   const tieneNum = /[0-9]/.test(valor);
74   if (valor.length < 8 || !tieneMayus || !tieneMinus || !tieneNum) {
75     errorDiv.textContent = 'La contraseña debe tener al menos 8 caracteres, una
76     mayúscula, una minúscula y un número.';
77     return false;
78   }
79   errorDiv.textContent = '';
80   return true;
81 }
82 // Validar confirmación de contraseña
83 function validarConfirmPassword() {
84   const errorDiv = document.getElementById('error-confirm');
85   if (password.value !== confirmPassword.value) {
86     errorDiv.textContent = 'Las contraseñas no coinciden.';
87     return false;
88   }
89   errorDiv.textContent = '';
90   return true;
91 }
92 // Validar todo el formulario
93 function validarFormulario() {
94   const esValido = [
95     validarNombre(),
96     validarEmail(),
97     validarPassword(),
98     validarConfirmPassword()
99   ].every(Boolean);
100   return esValido;
101 }
102 // Validación en tiempo real
103 nombre.addEventListener('input', validarNombre);
104 email.addEventListener('input', validarEmail);
105 password.addEventListener('input', validarPassword);
106 confirmPassword.addEventListener('input', validarConfirmPassword);
107 // Enviar formulario
108 form.addEventListener('submit', function(e) {
109   e.preventDefault(); // Evitar envío real
110   if (validarFormulario()) {
111     mensaje.textContent = '¡Registro exitoso!';
112     mensaje.className = 'success';
113     form.reset();
114     // Aquí podrías enviar los datos con fetch o XMLHttpRequest
115   } else {
116     mensaje.textContent = '';
117     // Enfocar el primer campo con error
118     if (document.querySelector('.error').textContent) {
119       const primerError = document.querySelector('input:invalid') || document.
120       querySelector('input[value=""]');
121       if (primerError) primerError.focus();
122     }
123   }
124 });
125 </script>
126 </body>

```

```

<style>
  .form-group { margin-bottom: 15px; }
  .error { color: red; font-size: 0.9em; }
  .success { color: green; }
</style>

```

EJERCICIO 2: FORMULARIO CON VALIDACIÓN EN TIEMPO REAL

Validación `[^\s@]+@[^\s@]+\.[^\s@]`:

- `[^\s@]+` Coinciden uno o más caracteres que no sean espacio en blanco ni `@` para la parte local (antes de la arroba).
- `@` Obliga a que haya exactamente una arroba en medio.
- `[^\s@]+` Uno o más caracteres que no sean espacio ni `@` para el dominio principal.
- `\.[^\s@]` Un punto seguido de al menos un carácter que no sea espacio ni `@` para el TLD (ej. `.com`, `.es`).

Este regex es suficiente para formularios simples de front-end.

Si necesitas algo más estricto:

- a) Usar un patrón con más restricciones en dominio y TLD:

```
// TLD de al menos 2 letras, solo A-Z (muy usado)
const regex = /^[a-zA-Z0-9._%+-]+@[a-zA-Z0-9.-]+\.[A-Za-z]{2,}$/;
```

- b) Confiar la validación fuerte al backend y a un email de verificación.

EJERCICIO 3:

Crea una página web que incluya un div que:

- Cambie de color cuando el ratón entra y vuelva al color original al salir.
- Muestre las coordenadas del ratón dentro del div mientras se mueve por encima.

Pasa el ratón por aquí

Ratón dentro - Mueve el cursor

X: 285 | Y: 129

Ejercicios resueltos

```
<body>
  <div id="zona">Pasa el ratón por aquí</div>
  <p id="coords"></p>

  <script>
    const zona = document.getElementById('zona');
    const coords = document.getElementById('coords');

    // Entrada del ratón
    zona.addEventListener('mouseover', () => {
      zona.style.backgroundColor = 'lightblue';
      zona.textContent = 'Ratón dentro - Mueve el cursor';
    });

    // Salida del ratón
    zona.addEventListener('mouseout', () => {
      zona.style.backgroundColor = 'white';
      zona.textContent = 'Pasa el ratón por aquí';
      coords.textContent = '';
    });

    // Movimiento dentro
    zona.addEventListener('mousemove', (event) => {
      // Coordenadas relativas al viewport
      coords.textContent = `X: ${event.clientX} | Y: ${event.clientY}`;
    });

    // Clic adicional como bonus
    zona.addEventListener('click', () => {
      alert('Has hecho clic en la zona!');
    });
  </script>
</body>
```

```
<style>
  #zona {
    width: 300px;
    height: 150px;
    border: 1px solid #000;
    margin: 20px;
    padding: 20px;
    transition: background-color 0.3s;
    font-family: Arial, sans-serif;
  }
</style>
```

EJERCICIO 3 - SOLUCIÓN

EJERCICIO 4:

Crea una página que incluya un `<input type="text">` que:

- Muestre en un `<p>` el número de caracteres escritos en tiempo real.
- No permita escribir la letra “a” (mayúscula ni minúscula); si se pulsa, ignora la tecla y muestra un mensaje de aviso durante un determinado tiempo.

0 caracteres

3 caracteres

✗ No se permite escribir la letra "a"

EJERCICIO 4 – SOLUCIÓN

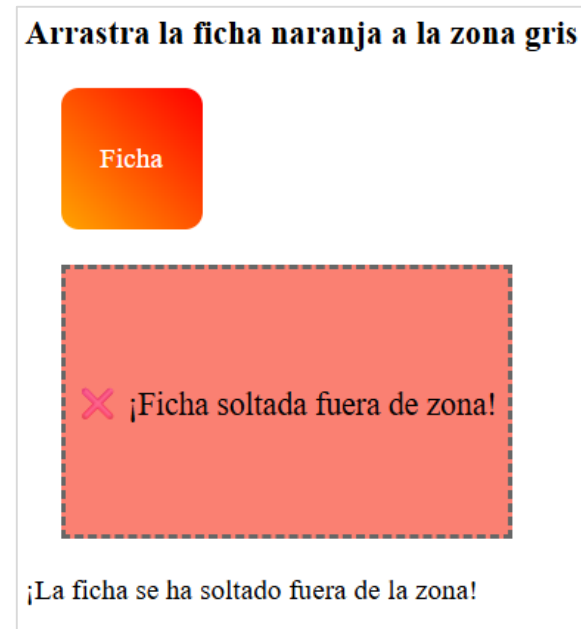
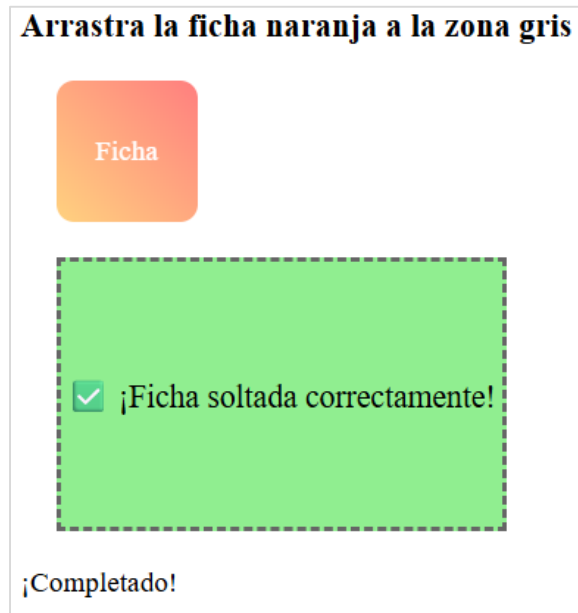
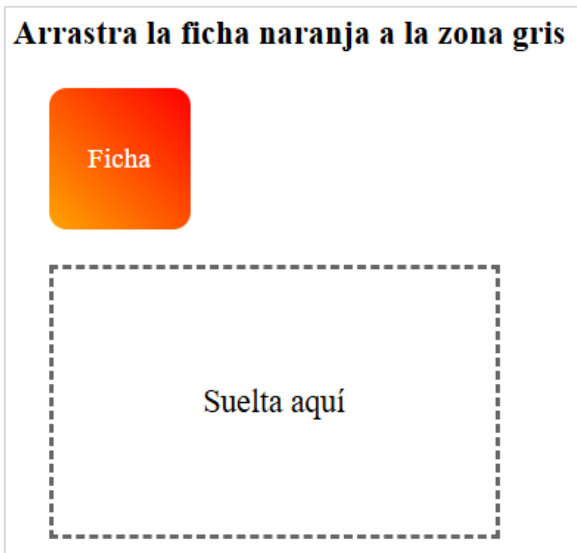
```
<body>
  <input type="text" id="campoTexto" placeholder="Escribe algo (sin 'a')">
  <br><br>
  <p id="contador">0 caracteres</p>
  <p id="aviso"></p>
  <script>
    const campo = document.getElementById('campoTexto');
    const contador = document.getElementById('contador');
    const aviso = document.getElementById('aviso');
    // Contador en tiempo real
    campo.addEventListener('input', () => {
      const longitud = campo.value.length;
      contador.textContent = `${longitud} caracteres`;
    });
    // Filtra tecla 'a' antes de que se escriba
    campo.addEventListener('keydown', (event) => {
      const tecla = event.key.toLowerCase();
      if (tecla === 'a') {
        event.preventDefault(); // Bloquea la tecla
        aviso.textContent = '❌ No se permite escribir la letra "a"';
        setTimeout(() => aviso.textContent = '', 2000); // Limpia aviso
        return false;
      }
    });
    // Bonus: Enter ejecuta acción
    campo.addEventListener('keypress', (event) => {
      if (event.key === 'Enter') {
        alert(`Texto final: "${campo.value}"`);
      }
    });
  </script>
</body>
```

```
<style>
  input { width: 300px; padding: 10px; font-size: 16px; }
  #aviso { color: orange; font-weight: bold; }
</style>
```

EJERCICIO 5:

Crea una “ficha” que se pueda arrastrar con el ratón y una zona de destino:

- Cuando sueltes la ficha sobre la zona, el texto de la zona cambia a “¡Soltaste la ficha aquí!” y deshabilita la ficha
- Cuando sueltes la ficha fuera de la zona, el texto de la zona cambia a “¡Ficha soltada fuera de la zona!” la ficha queda habilitada para que se pueda volver a intentar



Ejercicios resueltos

```
<body>
  <h3>Arrastra la ficha naranja a la zona gris</h3>
  <div id="ficha" draggable="true">Ficha</div>
  <div id="destino">Suelta aquí</div>
  <p id="estado"></p>
  <script>
    const ficha = document.getElementById('ficha');
    const destino = document.getElementById('destino');
    const estado = document.getElementById('estado');
    let droppedInZone = false;
    // Inicio del arrastre
    ficha.addEventListener('dragstart', (event) => {
      event.dataTransfer.setData('text/plain', 'ficha_drag');
      estado.textContent = 'Arrastrando...';
      droppedInZone = false;
    });
    // Permitir soltar sobre destino
    destino.addEventListener('dragover', (event) => {
      event.preventDefault(); // Esencial para permitir drop
      destino.classList.add('dragover');
    });
    // Finalizar arrastre sobre destino
    destino.addEventListener('dragleave', () => {
      destino.classList.remove('dragover');
    });
    // Soltar en zona destino
    destino.addEventListener('drop', (event) => {
      event.preventDefault();
      destino.classList.remove('dragover');
      const data = event.dataTransfer.getData('text/plain');
      if (data === 'ficha_drag') {
        destino.innerHTML = '✅ ¡Ficha soltada correctamente!';
        destino.style.background = 'lightgreen';
        estado.textContent = '¡Completado!';
        ficha.style.opacity = '0.5'; // Desactiva ficha
        droppedInZone = true;
      }
    });
    // Detectar cuando se suelta la ficha FUERA de la zona
    ficha.addEventListener('dragend', (event) => {
      if (!droppedInZone) {
        destino.innerHTML = '❌ ¡Ficha soltada fuera de zona!';
        destino.style.background = 'salmon';
        estado.textContent = '¡La ficha se ha soltado fuera de la zona!';
        // Opcional: alert('❌ Error! Suelta la ficha fuera de la zona destino.');
```

```
<style>
  #ficha {
    width: 80px;
    height: 80px;
    background: linear-gradient(45deg, orange, red);
    color: white;
    text-align: center;
    line-height: 80px;
    cursor: move;
    border-radius: 10px;
    margin: 20px;
    user-select: none;
  }
  #destino {
    width: 250px;
    height: 150px;
    border: 3px dashed #666;
    margin: 20px;
    display: flex;
    align-items: center;
    justify-content: center;
    font-size: 18px;
    transition: all 0.3s;
  }
  #destino.dragover { border-color: green; background: rgba(0,255,0,0.1); }
</style>
```

EJERCICIO 5 - SOLUCIÓN

Gracias

