
UNIVERSIDAD ESTATAL AMAZÓNICA
VITA ET SAPIENTIA
UEA
UNIVERSIDAD
ESTATAL AMAZÓNICA

DESARROLLO DE APLICACIONES WEB

Unidad 2

Desarrollo del Lado del Cliente

Tema 2.1: JavaScript para aplicaciones web

Subtema 2.1.2: Validaciones dinámicas y manejo básico de formularios

DESARROLLO DE APLICACIONES WEB

Transformamos el mundo desde la Amazonía

Ing. Walter Rodrigo Núñez Zamora, Mgs.
DOCENTE - PERSONAL ACADÉMICO NO TITULAR OCASIONAL

UNIVERSIDAD ESTATAL AMAZÓNICA

SEMANA

6

DESARROLLO DE LA SEMANA 6

DEL LUN. 12 AL DOM. 18 DE ENERO / 2026

Resultado de aprendizaje: Aplicar técnicas de diseño de interfaces y desarrollo frontend y backend para implementar funcionalidades dinámicas y gestionar datos de manera eficiente.

CONTENIDOS

UNIDAD VI :Desarrollo del lado del cliente

- Tema 1: JavaScript para aplicaciones web
  - Subtema 1.2 : Validaciones dinámicas y manejo básico de formularios.

Subtema 1.1 ” Validaciones dinámicas y manejo básico de formularios.

DESARROLLO WEB

CARRERAS EN LÍNEA

¿Qué son los Formularios?

Un formulario es un elemento HTML que permite recopilar datos del usuario.

Incluye campos como texto, contraseñas , botones y casillas de verificación.

- Mejora la experiencia del usuario.
- Reduce errores en la recolección de datos.

Estructura de un formulario

```html
<form action="URL" method="GET|POST">
    <label for="nombre">Nombre:</label>
    <input type="text" id="nombre" name="nombre" required>
    <button type="submit">Enviar</button>
</form>
```

Etiqueta <form>: Encierra todos los elementos del formulario.

Atributos clave:

action: Especifica dónde se envían los datos.

method: Define cómo se envían los datos (GET o POST).

Elementos comunes:

Entradas (<input>): Textos, contraseñas, casillas de verificación.

Selectores (<select>): Listas desplegables.

Botones (<button> o <input type="submit">): Envío o reseteo de datos.

Importancia de las Validaciones

Las validaciones en formularios son esenciales para garantizar la calidad y seguridad de los datos ingresados por los usuarios. Sus principales beneficios incluyen:

1. Prevención de errores:
   Ayudan a detectar y corregir datos incorrectos o faltantes antes de ser enviados al servidor.
2. Mejora de la experiencia del usuario:
   Proporcionan retroalimentación inmediata, reduciendo frustraciones al completar formularios.
3. Protección contra ataques:
   Minimiza riesgos como la inyección de código o el envío de datos maliciosos.
4. Optimización del flujo de trabajo:
   Garantizan que los datos recopilados sean correctos y estén completos, evitando procesamiento adicional en el servidor.

Tipos de Validación

1. Validaciones del lado del cliente (Frontend)

- **¿Dónde se realizan?**

En el navegador del usuario, utilizando tecnologías como HTML5, JavaScript o bibliotecas relacionadas.

- **Ventajas**:
  - Proporcionan retroalimentación inmediata.
  - Reducen la carga en el servidor al evitar el envío de datos incorrectos.

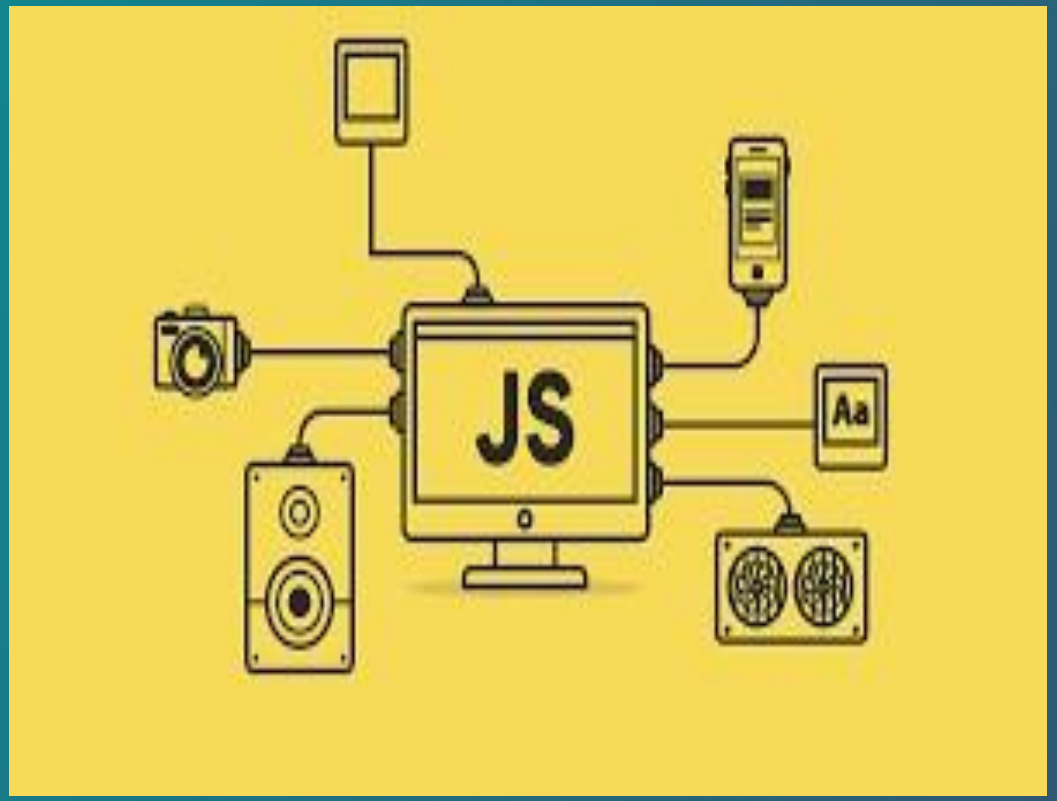
EJEMPLOS : html y javascript

<input type="email" required>

Uso de JavaScript:

javascript

```
if (campo.value === "") {
   alert("El campo no puede estar vacío");
}
```



Tipos de Validación

2. Validaciones del lado del servidor (Backend)

•**¿Dónde se realizan?**

En el servidor, después de que los datos son enviados por el formulario.

•**Ventajas**:

- Más seguras, ya que los datos son validados independientemente del navegador.
- Protegen contra manipulación malintencionada de los datos.

•**Ejemplo**:

- Validar un correo en un servidor (PHP):

```
if (!filter_var($email, FILTER_VALIDATE_EMAIL)) { echo "Correo no válido"; }
```

¿Qué es una Validación Dinámica??

Una validación dinámica es un proceso de verificación de datos en tiempo real, mientras el usuario interactúa con un formulario, sin necesidad de recargar la página o enviar los datos al servidor.

Características principales:

Interacción en tiempo real: El formulario verifica automáticamente si los datos ingresados cumplen con los requisitos establecidos mientras el usuario los introduce.

Uso de JavaScript: Se implementa principalmente mediante JavaScript, lo que permite manipular el DOM y proporcionar retroalimentación inmediata.

Experiencia del usuario mejorada: Reduce la frustración al evitar errores después de enviar el formulario

CARRERAS EN LÍNEA

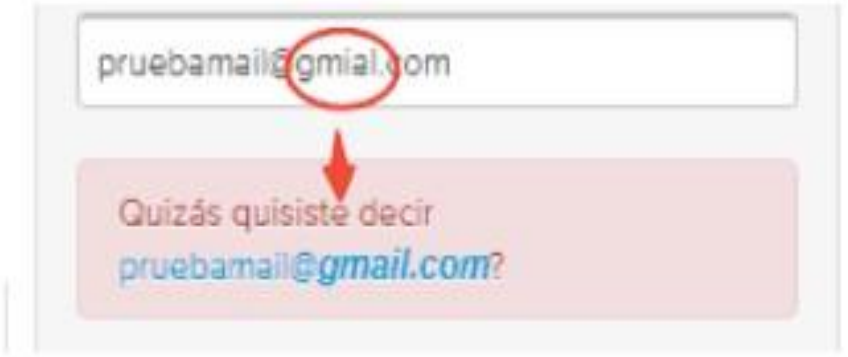
¿Qué es una Validación Dinámica??

Ejemplo: Validación de un correo

javasript

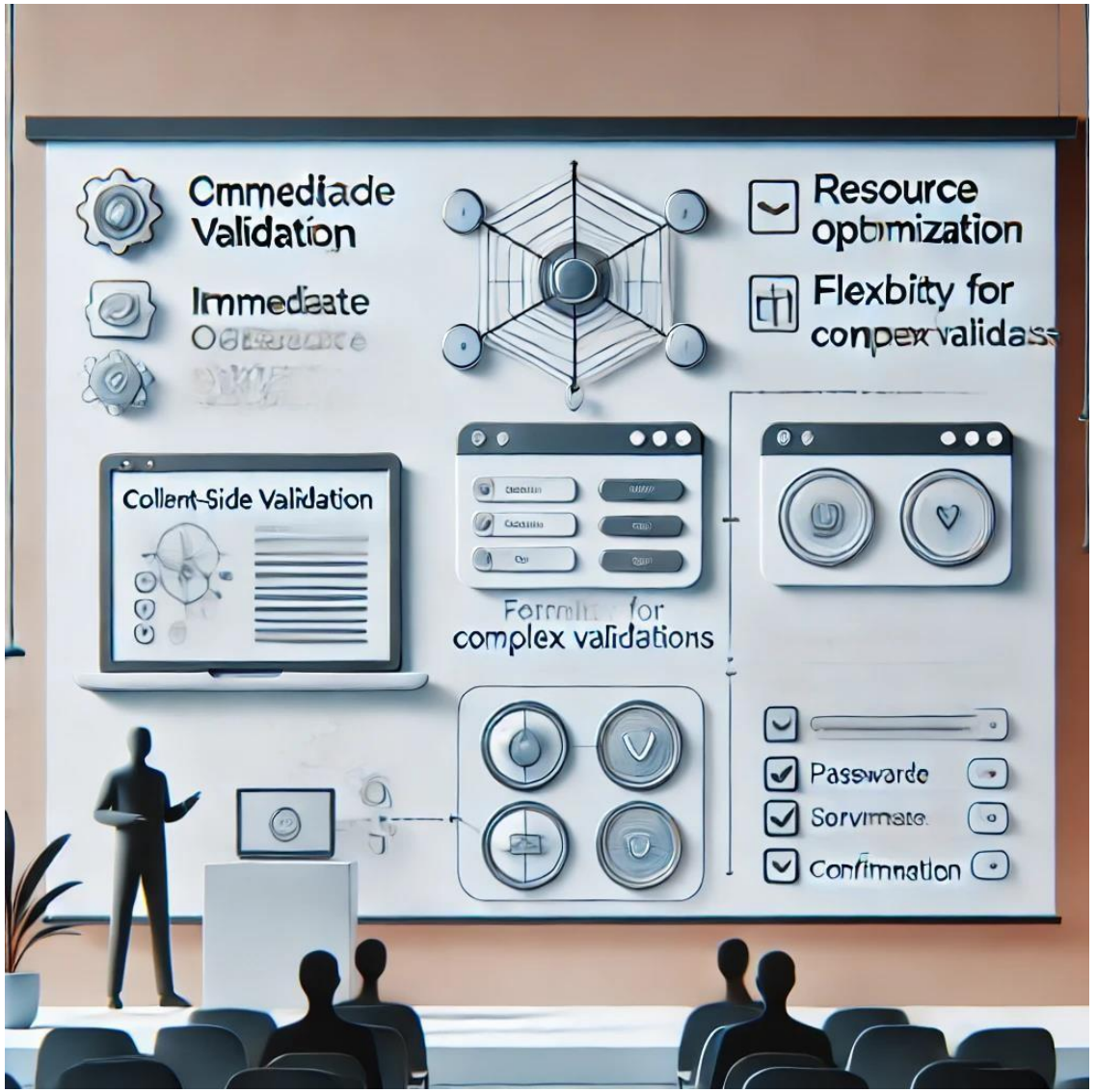
```
const emailInput = document.getElementById("email");
const feedback = document.getElementById("feedback");

emailInput.addEventListener("input", () => {
    const emailPattern = /^[^\s@]+@[^\s@]+\.[^\s@]+$/;
    if (emailPattern.test(emailInput.value)) {
        feedback.textContent = "Correo válido";
        feedback.style.color = "green";
    } else {
        feedback.textContent = "Correo no válido";
        feedback.style.color = "red";
    }
});
```

CARRERAS EN LÍNEA

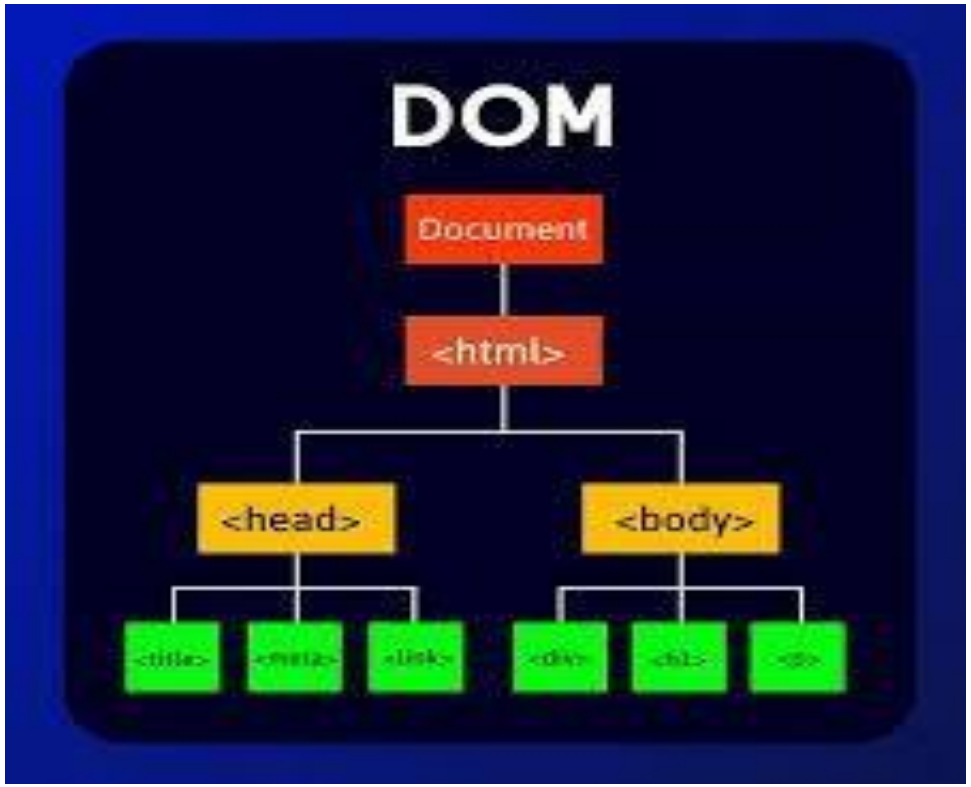


¿Qué es una Validación Dinámica??

Ventajas:
Retroalimentación inmediata: Los usuarios saben al instante si algo está mal.
Optimización de recursos: Reduce la cantidad de solicitudes al servidor.
Flexibilidad: Se pueden realizar validaciones más complejas que con HTML5 básico.
Aplicaciones comunes:
Comprobación de campos obligatorios.
Validación de formato (como correos, contraseñas, fechas).
Confirmación en campos relacionados (como verificar contraseñas coincidentes).



Manipulación de Formularios con JavaScript



En JavaScript, la manipulación de formularios se realiza principalmente a través del acceso a los elementos del DOM (Document Object Model). Esto te permite interactuar con los campos de entrada y sus valores, realizando validaciones, actualizaciones o cambios dinámicos.

```
<form>
  <label for="nombre">Nombre:</label>
  <input type="text" id="nombre" value="Juan">
  <button type="button" id="cambiar">Cambiar Nombre</button>
</form>
```

```
document.getElementById('cambiar').addEventListener('click', function() {
document.getElementById('nombre').value = "Carlos"; // Cambia el valor del campo 'nombre' });
{
```

¿Qué es el evento submit?

Se activa cuando un formulario es enviado, ya sea por un clic en el botón de envío o presionando "Enter" en un campo.

1 Manejo del evento submit:

Usando addEventListener:

```
document.getElementById('miFormulario').addEventListener('submit', function(event) {
    event.preventDefault();  // Evita el envío
});
```

Usando el atributo onsubmit en HTML:

```
<form onsubmit="validarFormulario(event)">
```

Validaciones Dinámicas de Campos

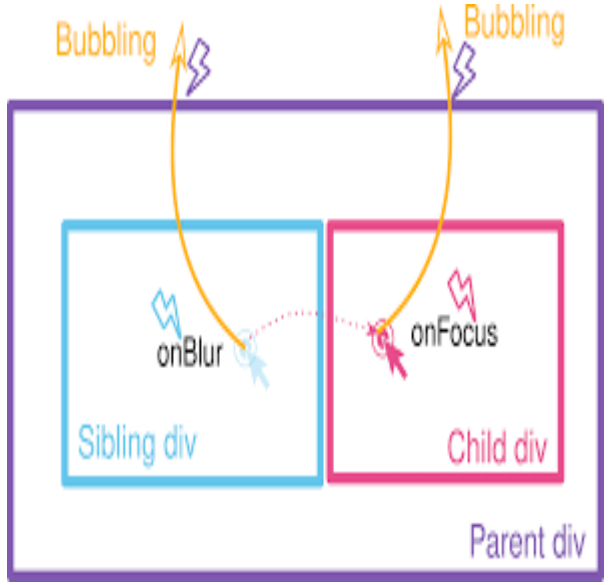
Validar en eventos como input o blur:

Html:

```
<form id="miFormulario">
   <label for="nombre">Nombre:</label>
   <input type="text" id="nombre" name="nombre" required>
   <span id="mensajeError" style="color: red;"></span>
   <button type="submit">Enviar</button>
</form>
```

JavaScript:

```
const campo = document.getElementById('nombre');const
mensajeError =
document.getElementById('mensajeError');campo.addEventListener('i
nput', () => {    if (campo.value.length < 3) {
mensajeError.textContent = 'El nombre es demasiado corto';     } else {
mensajeError.textContent = '';  // Limpiar el mensaje si la validación es
exitosa    }});
```



Validaciones Dinámicas de Campos

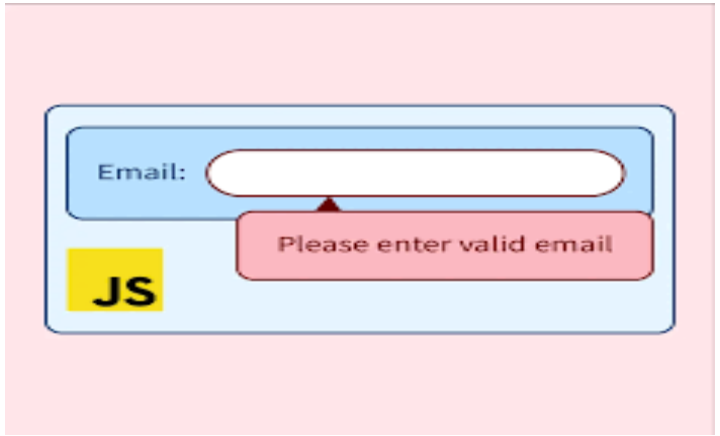
Validación de Correos Electrónicos:

Html:

```
<form id="formCorreo">
    <label for="correo">Correo Electrónico:</label>
    <input type="email" id="correo" name="correo" required>
    <span id="mensajeError" style="color: red;"></span>
    <button type="submit">Enviar</button>
</form>
```

JavaScript:

```
const correoInput = document.getElementById('correo');
const mensajeError = document.getElementById('mensajeError');

correoInput.addEventListener('input', () => {
    const correo = correoInput.value;
    const regex = /^[^\s@]+@[^\s@]+\.[^\s@]+$/;

    if (!regex.test(correo)) {
        mensajeError.textContent = 'Correo inválido';
    } else {
        mensajeError.textContent = '';  // Limpiar mensaje de error si el correo es válido
    }
});
```



Validaciones Dinámicas de Campos

Validación de Contraseñas

Html:

```
<form id="formContraseña">
    <label for="contraseña">Contraseña:</label>
    <input type="password" id="contraseña" name="contraseña" required>
    <span id="mensajeError" style="color: red;"></span>
    <button type="submit">Enviar</button>
</form>
```

JavaScript:

```
const contraseñaInput = document.getElementById('contraseña');
const mensajeError = document.getElementById('mensajeError');

contraseñaInput.addEventListener('input', () => {
    const contraseña = contraseñaInput.value;

    // Verificar longitud mínima de 8 caracteres
    if (contraseña.length < 8) {
        mensajeError.textContent = 'La contraseña debe tener al menos 8 caracteres';
    }
    // Verificar si contiene caracteres especiales
    else if (!/[!@#$%^&*(),.?":{}|<>]/.test(contraseña)) {
        mensajeError.textContent = 'La contraseña debe incluir al menos un carácter especial';
    }
    else {
        mensajeError.textContent = ''; // Limpiar mensaje de error si la contraseña es válida
```

Validaciones Dinámicas de Campos

Validación de Números:

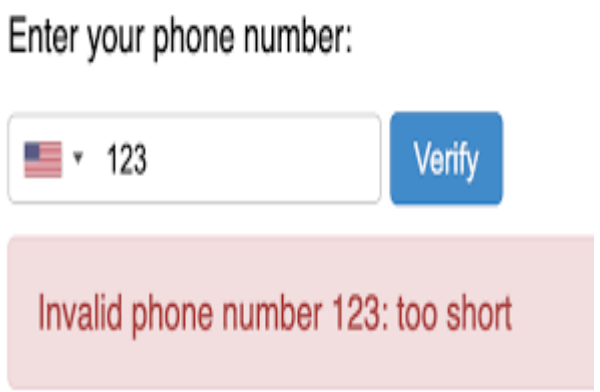
Asegurarse de que los números estén en un rango específico:

Html:

```
<form id="formEdad"> <label for="edad">Edad:</label> <input type="number" id="edad" name="edad" required> <span id="mensajeError" style="color: red;"></span> <button type="submit">Enviar</button> </form>
```

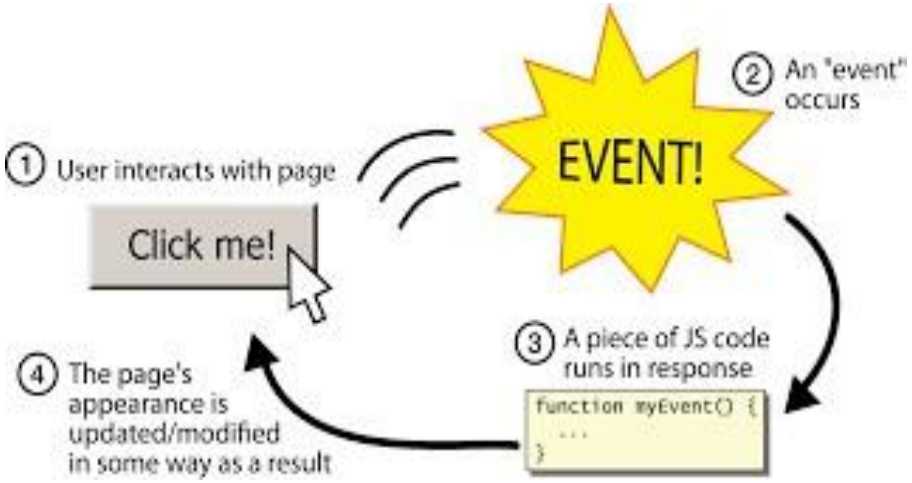
JavaScript:

```
const edadInput = document.getElementById('edad');
const mensajeError = document.getElementById('mensajeError');

edadInput.addEventListener('input', () => {
    const edad = parseInt(edadInput.value);

    // Verificar si la edad está en el rango de 18 a 65
    if (edad < 18 || edad > 65) {
        mensajeError.textContent = 'Edad no permitida. Debe estar entre 18 y 65 años.';
    } else {
        mensajeError.textContent = '';  // Limpiar mensaje de error si la edad es válida
    }
});
```



Errores Comunes en Validaciones

- No prevenir el envío sin validación.
- No manejar errores de usuario de forma clara.




JUNTOS TRANSFORMAMOS
EL MUNDO DESDE
LA AMAZONÍA