

## 3.3. Validación de formularios

La **validación de formularios** es una parte fundamental en el desarrollo de aplicaciones web, ya que permite asegurarse de que los datos ingresados por los usuarios sean correctos antes de enviarlos al servidor. JavaScript es una herramienta excelente para validar datos en el lado del cliente, mejorando la experiencia del usuario al proporcionar retroalimentación en tiempo real.

---

### 3.3.1. ¿Por qué es importante la validación de formularios?

La validación de formularios asegura que los datos proporcionados por los usuarios sean completos y estén en el formato correcto, lo que:

- Evita el envío de datos erróneos o incompletos al servidor.
  - Mejora la seguridad, evitando datos maliciosos como los intentos de inyección SQL.
  - Mejora la experiencia del usuario, ya que se proporciona retroalimentación instantánea.
- 

### 3.3.2. Tipos de validación

Hay dos tipos de validación que se utilizan en el desarrollo web:

**Validación del lado del cliente:** Se realiza en el navegador, antes de enviar los datos al servidor. Es rápida y proporciona una experiencia de usuario más fluida.

**Validación del lado del servidor:** Se realiza en el servidor después de que los datos han sido enviados. Es crucial, ya que siempre se puede evadir la validación del lado del cliente (por ejemplo, manipulando el código fuente).

---

### 3.3.3. Validación básica con JavaScript

La validación básica en JavaScript se puede hacer utilizando condiciones if para comprobar los valores de los campos del formulario.

**Validación de un campo obligatorio:** Asegurarse de que un campo no esté vacío es una de las validaciones más comunes.

```
const formulario = document.getElementById("miFormulario");
formulario.addEventListener("submit", function(event) {
  const nombre = document.getElementById("nombre").value;
  if (nombre === "") {
    alert("El campo nombre es obligatorio.");
    event.preventDefault(); // Previene el envío del formulario
  }
});
```

```
    }  
  });
```

**Validación de formato de correo electrónico:** Podemos validar que el valor ingresado en un campo sea un correo electrónico válido usando expresiones regulares.

```
formulario.addEventListener("submit", function(event) {  
    const email = document.getElementById("email").value;  
    const regexEmail = /^[a-zA-Z0-9._%+-]+@[a-zA-Z0-9.-]+\.[a-zA-Z]{2,}$/;  
    if (!regexEmail.test(email)) {  
        alert("Por favor ingresa un correo electrónico válido.");  
        event.preventDefault();  
    }  
});
```

### 3.3.4. Validación de contraseñas

Para validar contraseñas, normalmente se requiere:

**Longitud mínima.**

**Carácter especial.**

**Al menos una letra mayúscula.**

**Al menos un número.**

Aquí hay un ejemplo de cómo hacerlo con una expresión regular:

```
formulario.addEventListener("submit", function(event) {  
    const password = document.getElementById("password").value;  
    const regexPassword = /^(?=.*[A-Z])(?=.*\d)(?=.*[@$!%*?&])[A-Za-z\d@$!%*?&]{8,}$/;  
    if (!regexPassword.test(password)) {  
        alert("La contraseña debe tener al menos 8 caracteres, una mayúscula, un  
        número y un carácter especial.");  
        event.preventDefault();  
    }  
});
```

### 3.3.5. Validación de campos numéricos

Es común tener campos donde solo se deben ingresar números, como la edad o el teléfono.

```
formulario.addEventListener("submit", function(event) {
```

```

const edad = document.getElementById("edad").value;
if (isNaN(edad) || edad < 18) {
    alert("La edad debe ser un número y mayor o igual a 18.");
    event.preventDefault();
}
});

```

### 3.3.6. Mostrar errores de validación

Para mejorar la experiencia del usuario, se puede mostrar un mensaje de error junto al campo que no pasa la validación. Esto se puede hacer manipulando el DOM.

```

formulario.addEventListener("submit", function(event) {
    const nombre = document.getElementById("nombre").value;
    const mensajeError = document.getElementById("errorNombre");

    if (nombre === "") {
        mensajeError.textContent = "El campo nombre es obligatorio.";
        mensajeError.style.color = "red";
        event.preventDefault();
    } else {
        mensajeError.textContent = "";
    }
});

```

### 3.3.7. Usando HTML5 para validación

HTML5 introduce varios atributos de validación que pueden facilitar la validación en el lado del cliente, como required, pattern, min, max, type, etc.

Ejemplo de un formulario con validación básica utilizando HTML5:

```

<form id="miFormulario">
    <label for="nombre">Nombre:</label>
    <input type="text" id="nombre" name="nombre" required>
    <br><br>

    <label for="email">Correo Electrónico:</label>
    <input type="email" id="email" name="email" required>
    <br><br>

    <label for="password">Contraseña:</label>

```

```
<input type="password" id="password" name="password" minlength="8" required>
<br><br>

<input type="submit" value="Enviar">
</form>
```

**required:** El campo no puede estar vacío.

**type="email":** Verifica que el valor sea una dirección de correo electrónico válida.

**minlength:** Asegura que la contraseña tenga al menos 8 caracteres.

### Resumen de conceptos clave:

- **Validación del lado del cliente:** Garantiza que los datos sean correctos antes de enviarlos al servidor, mejorando la experiencia del usuario.
- **Validación básica:** Usando JavaScript y HTML5 para asegurarse de que los campos no estén vacíos, tengan el formato correcto, y cumplan con las reglas definidas (por ejemplo, longitud de contraseñas).
- **Mensajes de error:** Es importante proporcionar retroalimentación visual al usuario si hay errores en la validación.
- **HTML5:** Utilizar los atributos de validación nativos como `required`, `pattern`, `type`, etc., facilita el proceso.