COLOMBIA POTENCIA DE LA VIDA

TIC

TALENTO TECH

DESARROLLO *WEB*

FULL STACK

NIVEL BÁSICO

Introducción a JavaScript

cymetria | tecnalia colombia

Introducción a JavaScript

JavaScript es un lenguaje de programación esencial en el desarrollo web. Es lo que permite que las páginas sean dinámicas e interactivas, respondiendo a las acciones de los usuarios sin necesidad de recargar la página.

¿Dónde se usa JavaScript?

JavaScript es omnipresente en la web. Casi todas las páginas lo utilizan para mejorar la experiencia del usuario. Algunas de sus aplicaciones más comunes incluyen:

- Creación de animaciones y efectos visuales.
- Respuesta a eventos como clics, movimientos del mouse o entrada de datos.
- Manipulación del contenido de la página en tiempo real.
- Conexión con servidores para actualizar información sin recargar la página.

¿Cómo funciona JavaScript en un sitio web?

JavaScript se ejecuta directamente en el navegador del usuario, lo que permite que las acciones sean rápidas y no dependan completamente del servidor. Esto se conoce como un lenguaje del lado del cliente, aunque también puede ejecutarse en servidores con tecnologías como Node.js.

Relación con HTML y CSS

JavaScript no trabaja solo. Funciona en conjunto con:

- **HTML**, que define la estructura del sitio web.
- **CSS**, que da estilo y diseño a la página.
- **JavaScript**, que agrega comportamiento e interactividad.

Podemos imaginar una página web como un cuerpo humano:

- **HTML** sería el esqueleto.
- **CSS** sería la piel y la ropa.
- **JavaScript** sería el sistema nervioso que permite reaccionar y moverse.

Herramientas para usar JavaScript

Para empezar a programar en JavaScript, solo necesitas:

- Un navegador web (Chrome, Firefox, Edge, Safari).
- Un editor de código como Visual Studio Code, Sublime Text o incluso el Bloc de notas.
- La consola del navegador, donde puedes probar código JavaScript directamente.

Relación entre JavaScript del lado del cliente y del lado del servidor

Para entender cómo funciona una aplicación web moderna, es útil imaginar una conversación entre tres partes:

1.El cliente (Frontend)

2.El servidor (Backend)

3.La base de datos

JavaScript del lado del cliente

Imagina que el cliente es la persona que visita una tienda en línea desde su teléfono o computadora (el navegador). Cuando esa persona interactúa con la página, el navegador necesita enviar una respuesta rápida sin tener que esperar mucho. Aquí es donde entra

JavaScript del lado del cliente.

¿Qué hace?

•Permite que la página sea interactiva: mostrar mensajes, animaciones, y responder a los clics o desplazamientos.
•Actualiza parte de la página sin necesidad de recargarla completamente.
•Enviará información al **servidor** si necesita algo más, como datos de usuario o detalles de un producto.
Este JavaScript se ejecuta directamente en el **navegador** de la persona, por lo que es rápido y eficiente. El cliente solo necesita enviar y recibir los datos, pero no hace cálculos complejos ni maneja información de la base de datos.

Tareas que realiza JavaScript del lado del cliente

- **Interactividad en la página**
  - Permite que los usuarios interactúen con la página sin recargarla (clics, desplazamientos, entradas de texto, etc.).
  - Actualiza el contenido de la página en tiempo real, como mostrar un mensaje cuando se hace clic en un botón.

- **Validación de formularios**
  - Verifica que los datos introducidos por el usuario sean correctos (por ejemplo, si el correo electrónico tiene el formato adecuado) antes de enviarlos al servidor.
  - Evita errores y mejora la experiencia del usuario al no tener que esperar una respuesta del servidor para corregir los errores.

- **Animaciones y efectos visuales**
  - Permite crear animaciones, transiciones y efectos visuales (como hacer que un elemento se desplace o cambie de color).
  - Mejora la experiencia del usuario haciendo la página más atractiva y dinámica.

Tareas que realiza JavaScript del lado del cliente

- **Manejo de eventos**
  - Captura y responde a acciones del usuario, como clics, teclas presionadas, desplazamientos del mouse, etc.
  - Facilita la creación de interfaces dinámicas y reactivas.

- **Manipulación del DOM (Modelo de Objetos del Documento)**
  - Cambia, agrega o elimina elementos en la página web (texto, imágenes, botones, etc.) de manera dinámica sin necesidad de recargarla.
  - Permite personalizar la página según la interacción del usuario.

- **Envío de datos al servidor (AJAX)**
  - Permite enviar y recibir datos del servidor sin tener que recargar la página (usando tecnologías como AJAX).
  - Actualiza el contenido de la página con nuevos datos, como mostrar los resultados de una búsqueda sin que el usuario espere.

Tareas que realiza JavaScript del lado del cliente

- **Almacenamiento local (localStorage y sessionStorage)**

- Guarda datos en el navegador para que persistan entre sesiones o solo durante la sesión actual, sin necesidad de una base de datos.
- Permite almacenar preferencias o configuraciones personalizadas del usuario.

- **Control de sesiones y cookies**

- Permite gestionar información sobre el usuario, como preferencias o estado de inicio de sesión, almacenándola localmente en el navegador.
- Facilita el seguimiento del comportamiento del usuario durante su interacción con la página.

- **Trabajar con APIs del navegador**

- Permite interactuar con características del dispositivo o del navegador, como la geolocalización, la cámara, o la lectura de archivos.
- Mejora la experiencia del usuario con funcionalidades adicionales que pueden ser útiles en aplicaciones web.

Tareas que realiza JavaScript del lado del cliente

- Peticiones HTTP
- Envía solicitudes HTTP al servidor para obtener información (por ejemplo, datos de un servidor o de una API externa) sin recargar la página.
- Utiliza tecnologías como Fetch API o XMLHttpRequest.

JavaScript del lado del servidor

Ahora, cuando el cliente necesita algo que no puede manejar por sí solo (como obtener información de una base de datos, como los detalles de un producto), le pide al **servidor**. Aquí es donde entra **JavaScript del lado del servidor**.

En lugar de ser ejecutado por el navegador, el **JavaScript del servidor** se ejecuta en una **computadora** o **servidor** remoto. Esta parte se encarga de recibir solicitudes del cliente, realizar tareas como consultar bases de datos y procesar la lógica del negocio, y luego devolver una respuesta al cliente.

JavaScript del lado del servidor

¿Qué hace?

•Recibe las solicitudes del cliente (por ejemplo, un pedido de información sobre productos).

•Hace consultas a la **base de datos**, donde se guarda la información (como los detalles de un producto o los datos de un usuario).

•Procesa esta información y la envía de vuelta al cliente para que pueda mostrarla.

•Si se necesitan cambios en la base de datos (como agregar un nuevo producto o actualizar el stock), el servidor también se encarga de ello.

Conexión entre el Frontend, Backend y la Base de Datos

Imagina un sitio web de compras en línea:

1.El **cliente** (usando el navegador) solicita ver una lista de productos disponibles.

2.JavaScript del cliente se encarga de la interacción en la página (como hacer clic en un botón de "Ver productos").

3.JavaScript del servidor en el **backend** recibe esa solicitud y consulta la **base de datos** para obtener la lista actualizada de productos.

4.El **servidor** devuelve los datos al cliente, que luego muestra los productos de manera interactiva y dinámica gracias a JavaScript del cliente.

5.Si el usuario decide comprar algo, los detalles de la compra se envían nuevamente al **servidor**, que actualiza la **base de datos** con la nueva compra.

Comparación js

Concepto/Función	JavaScript del Lado del Cliente	JavaScript del Lado del Servidor
Ejemplo de entorno	Navegador web (Chrome, Firefox, etc.)	Servidores (Node.js, Express, etc.)
Acceso al código	Visible para el usuario (puede ser inspeccionado en el navegador)	No accesible al usuario (ejecutado en el servidor)
Interactividad	Clics, desplazamientos, formularios, interacciones con el DOM	Lógica del servidor, manejo de peticiones HTTP
Validación de datos	Validación antes de enviar datos al servidor (por ejemplo, en formularios)	Validación adicional en el servidor para datos recibidos de los clientes
Manipulación del DOM	Cambia y manipula elementos HTML directamente en la página	No interactúa directamente con el DOM
Envío de datos	AJAX, Fetch API para enviar datos sin recargar la página	Recepción de solicitudes HTTP (GET, POST, PUT, DELETE)
Almacenamiento local	`localStorage`, `sessionStorage`, cookies	Almacenamiento en base de datos (SQL, NoSQL)
Comunicación con servidor	Peticiones HTTP (AJAX, Fetch) para interactuar con el servidor	Manejo de peticiones HTTP, API RESTful

Comparación js

Autenticación	Almacena cookies de sesión, manejo básico de autenticación	Control de acceso mediante tokens (JWT, OAuth, sesiones)
Seguridad	Prevención de XSS, CSRF, sanitización de entradas del usuario	Protección de datos sensibles, encriptación de contraseñas, HTTPS
Manejo de eventos	Responder a clics, movimientos de ratón, teclado, cambios de formulario	No gestionan eventos de usuario directamente
Escalabilidad	Limitada por el rendimiento del navegador del usuario	Mayor control sobre el rendimiento y la infraestructura
Control de sesión	Cookies, almacenamiento local	Sesiones en el servidor, autenticación en backend
Protección de datos sensibles	Evitar almacenar datos sensibles localmente (en cookies o almacenamiento local)	Cifrado de datos sensibles en la base de datos y en las comunicaciones
Lógica del negocio	Interactividad básica (cálculos simples en el navegador)	Lógica compleja, procesamiento de datos y operaciones en el servidor
Dependencias	Bibliotecas y frameworks frontend (React, Angular, etc.)	Frameworks del lado del servidor (Express, Koa, etc.)

Integrar JavaScript (JS) en un documento HTML

Existen varias formas de integrar JavaScript (JS) en un documento HTML. Aquí te presento las formas más comunes:

1. Etiqueta `<script>` en el encabezado (`<head>`)

- El código JavaScript se inserta dentro de la etiqueta `<script>` en la sección `<head>` del documento HTML. Esto cargará el script antes de que se cargue el contenido de la página.

html Copy Edit

```html
<!DOCTYPE html>
<html lang="es">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Integración de JS</title>
  <script>
    alert("Este es un mensaje desde el encabezado");
  </script>
</head>
<body>
  <h1>Hola Mundo</h1>
</body>
</html>
```

Ventaja: El script está cargado al principio, lo que permite que todas las funciones estén disponibles en el documento.

Desventaja: El script bloquea la carga del resto del contenido de la página hasta que se ejecute.

Integrar JavaScript (JS) en un documento HTML

2. Etiqueta `<script>` en el cuerpo (`<body>`)

- El código JavaScript puede insertarse directamente antes del cierre de la etiqueta `<body>`. Esto asegura que todo el contenido HTML haya sido cargado antes de ejecutar el script.

html Copy Edit

```html
<!DOCTYPE html>
<html lang="es">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Integración de JS</title>
</head>
<body>
  <h1>Hola Mundo</h1>
  <script>
    alert("Este es un mensaje después de cargar la página");
  </script>
</body>
</html>
```

Ventaja: No bloquea la carga de contenido en la página.

Desventaja: Si el script necesita manipular elementos HTML que están aún por cargar, puede haber errores.

Integrar JavaScript (JS) en un documento HTML

3. Archivo JavaScript Externo

- JavaScript también puede ser colocado en un archivo externo y luego vinculado a un documento HTML usando la etiqueta `<script>` con el atributo `src`.

html Copy Edit

```html
<!DOCTYPE html>
<html lang="es">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Integración de JS Externo</title>
  <script src="script.js" defer></script>
</head>
<body>
  <h1>Hola Mundo</h1>
</body>
</html>
```

Ventaja: Mantiene el código HTML limpio y organiza mejor el código JavaScript.

Desventaja: Necesita una correcta ruta del archivo y puede hacer que la carga de la página dependa de la disponibilidad del archivo externo.

Integrar JavaScript (JS) en un documento HTML

4. Atributo `onclick`, `onload`, y otros manejadores de eventos (JS Inline)

- En algunos casos, JavaScript se puede integrar directamente dentro de los atributos HTML que manejan eventos como `onclick`, `onload`, `onmouseover`, etc.

html Copy Edit

```html
<!DOCTYPE html>
<html lang="es">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Integración de JS Inline</title>
</head>
<body>
  <h1 onclick="alert('Hiciste clic en el título')">Haz clic aquí</h1>
</body>
</html>
```

Ventaja: Fácil y rápido de implementar, ideal para tareas simples o interacciones directas.
Desventaja: No es recomendable para código JavaScript complejo, ya que mezcla HTML y JavaScript, lo que dificulta el mantenimiento.