Yaziel Axel Rodriguez Javier

EVOLUCION DE JAVA SCRIPT

RODRIGUEZ JAVIER - ROLDAN CASTILLO - MONTES GUTIERREZ - PAULLET MARTINEZ

ORIGENES Y CREACION DE JAVASCRIPT

A mediados de los años 90, la web estaba en sus primeras etapas y era principalmente estática. Las páginas web mostraban contenido fijo y la interacción con el usuario era muy limitada. Navegadores como Netscape Navigator dominaban el mercado, pero se buscaba una forma de hacer la web más dinámica e interactiva.

CREACION DEL JAVASCRIPT

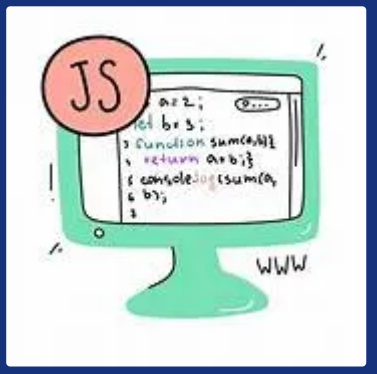
En 1995, Netscape Communications encargó a Brendan Eich, un ingeniero de software, desarrollar un lenguaje de scripting que pudiera ser integrado en el navegador para manipular páginas web en tiempo real. El objetivo era permitir a los desarrolladores agregar efectos dinámicos, validar formularios y crear interfaces más interactivas.

Desarrollo rápido

1. JavaScript fue diseñado para ser un lenguaje ligero, interpretado directamente por el navegador, con sintaxis sencilla similar a Java y C. Permitía manipular el Documento HTML (DOM), responder a eventos del usuario y controlar elementos dentro de la página sin necesidad de recargarla.

Características iniciales del JS

JavaScript fue diseñado para ser un lenguaje ligero, interpretado directamente por el navegador, con sintaxis sencilla similar a Java y C. Permitía manipular el Documento HTML (DOM), responder a eventos del usuario y controlar elementos dentro de la página sin necesidad de recargarla.



Estándar ECMAScript y estandarización.

ECMAScript es el estándar oficial que define cómo debe funcionar el lenguaje JavaScript. Gracias a este estándar, el código JavaScript se comporta igual en todos los navegadores y plataformas. La estandarización en JavaScript asegura que todos los navegadores (como Chrome, Firefox, etc.) usen las mismas reglas para ejecutar el código.

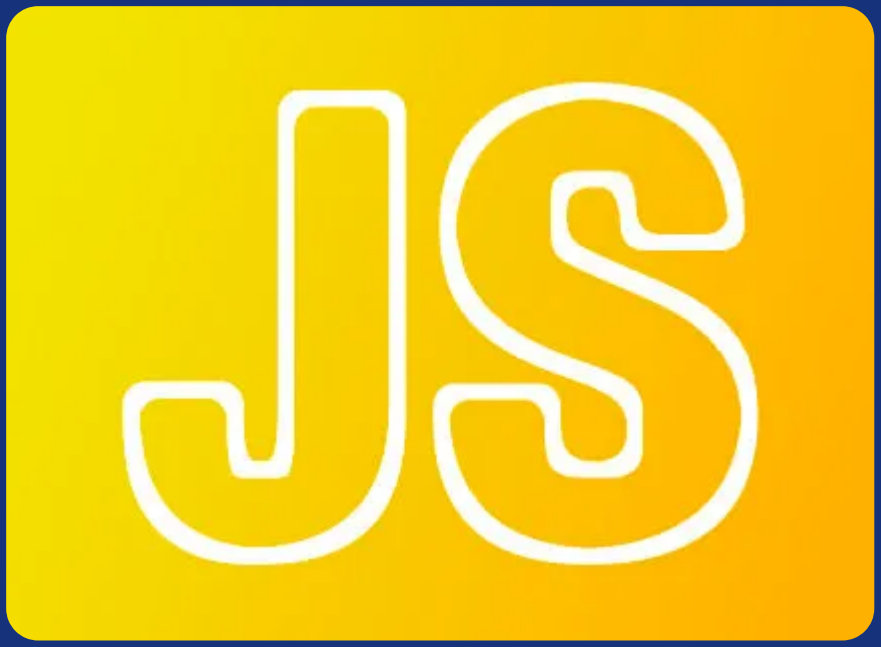
Que Es ECMAScript - Tecnología, Ciencia Y Educación.

ECMAScript es una especificación estánd...

azulweb.net

Estándar ECMAScript y estandarización

Esto lo gestiona un grupo llamado TC39, que cada año actualiza el estándar con nuevas funciones. JavaScript es una implementación de ECMAScript, pero también incluye otras herramientas que no forman parte del estándar, como el acceso al DOM o funciones para trabajar con páginas web.



Javascript

Populares Librerías como jQuery o React,...

desarrolloweb.com

Desde su creación, ECMAScript ha pasado por varias versiones importantes:

ES3 / 1999 / Primera adopción masiva
ES5 / 2009 / strict mode, JSON
ES6 / 2015
/ let, const, clases, promesas
ES2016+/
2016 / en adelante
Actualizaciones anuales con nuevas mejoras

Que Es ECMAScript - Tecnología, Ciencia Y Educación.

ECMAScript es una especificación estánd...

azulweb.net

ECMAScript

La estandarización de nuevas características en ECMAScript sigue un proceso de cinco etapas:
primero, se presenta una idea.
segundo, se analiza su viabilidad.
tercero, se redacta un borrador técnico.
cuarto, se prueba y se busca retroalimentación.
quinto, si todo está listo, la propuesta se aprueba para incluirse en la próxima versión del estándar.

What is JavaScript? The full-stack programming language

JavaScript is the most in-demand progra...

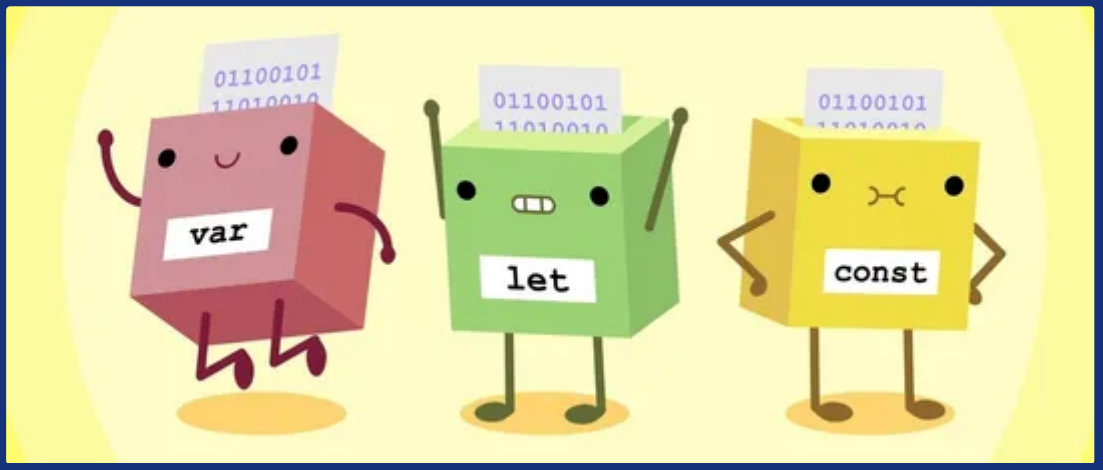
infoworld.com

Modernizacion y Nuevas caracteristicas

Desde ES6 (ECMAScript 2015) en adelante, JavaScript ha recibido una gran modernización con nuevas características que lo han hecho más limpio, legible y poderoso. Estas mejoras abarcan desde una mejor sintaxis para declarar variables hasta herramientas avanzadas para el manejo de la asincronía, la orientación a objetos y los módulos.

1. let y const

Se introdujeron let y const como nuevas formas de declarar variables. A diferencia de var, tienen un ámbito de bloque (block scope). let permite reasignación, mientras que const crea constantes que no se pueden reasignar.



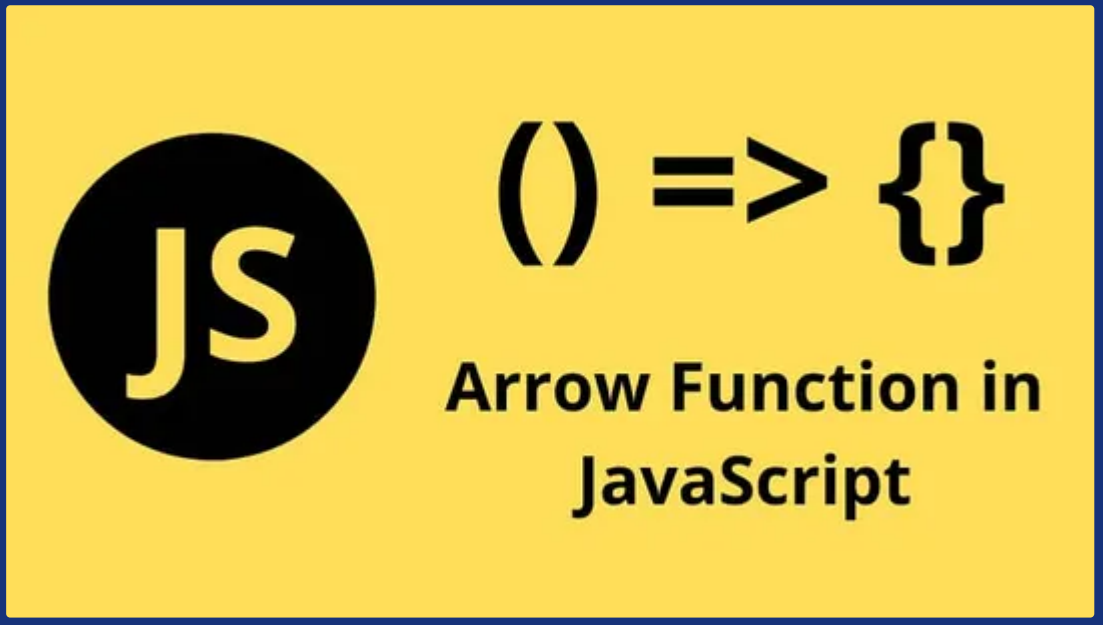
Ej.:

let nombre = "Juan";

const PI = 3.1416;

2. Funciones flecha (arrow functions)
Permiten escribir funciones de forma más concisa y no cambian el valor de this dentro del contexto en que se declaran.

Ej.:
const saludar = (nombre) ⇒ `Hola, ${nombre}`;



3. Template literals

Los template literals (comillas invertidas `) permiten interpolar variables y expresiones dentro de strings de manera más legible.

Ej.:

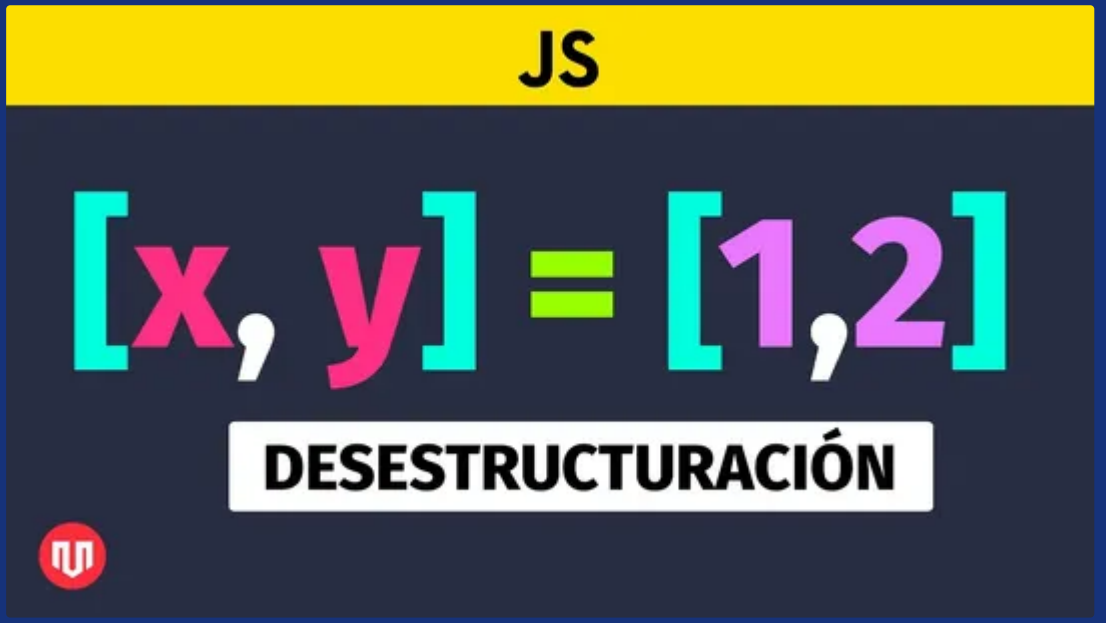
```
const nombre = "Ana";
const mensaje = `Hola, ${nombre}`;
```

4. Desestructuración

Permite extraer datos de objetos o arrays de forma rápida y legible.

Ej.:

```
const persona = { nombre: "Carlos",
edad: 30 };
const { nombre, edad } = persona;
```



5. Parámetros por defecto

Permite asignar valores por defecto a los parámetros de una función.

Ej.:

```
function saludar(nombre = "Invitado") {
return `Hola, ${nombre}`;
}
```

6. Operador Spread y Rest

El operador ... se usa para expandir elementos (spread) o para agrupar múltiples argumentos en un array (rest).

Ejm:

```
// Spread
const arr1 = [1, 2];
const arr2 = [...arr1, 3, 4];
// Rest
function sumar(...numeros) {
return numeros.reduce((a, b) ⇒ a + b);
}
```

7. Clases

JavaScript introdujo una sintaxis más limpia para trabajar con programación orientada a objetos mediante class.

Ejm:

class Persona {
constructor(nombre) {
this.nombre = nombre;
}
saludar() {
console.log(`Hola, soy ${this.nombre}`);
}
}

2. Frameworks y Librerías JavaScript

Frontend

- **React:** Biblioteca declarativa para construir interfaces con componentes reutilizables.
- **Angular:** Framework completo para SPAs con TypeScript.
- **Vue.js:** Framework progresivo, fácil de integrar y aprender.

Backend

- **Node.js:** Entorno para ejecutar JavaScript en el servidor, con arquitectura basada en eventos y alta

1. Lenguaje JavaScript

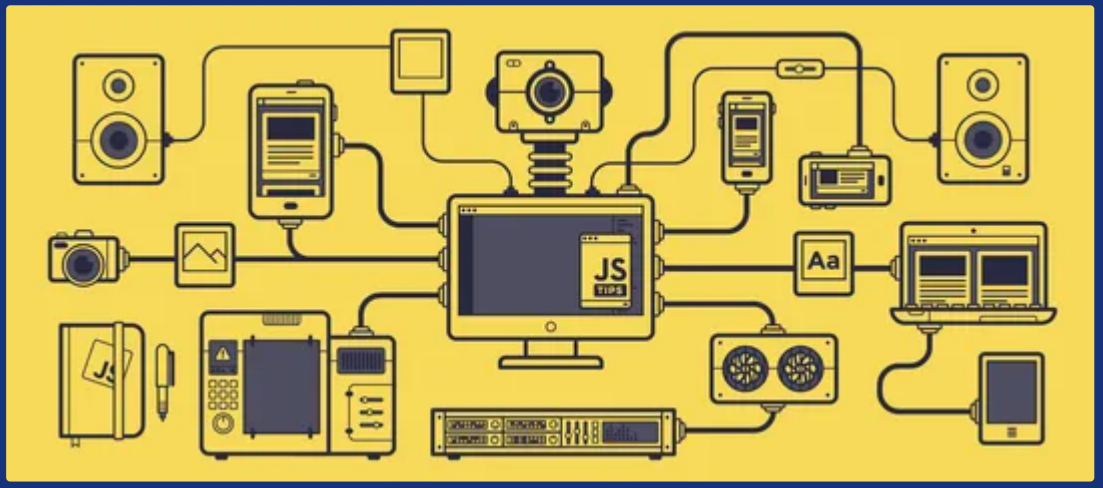
- Lenguaje de programación **dinámico, interpretado y multiparadigma**.
- Usado principalmente para desarrollo web en **navegadores (frontend)**.
- Desde Node.js, también se usa en **backend** y desarrollo de aplicaciones completas.
- Actualizaciones anuales a través del estándar **ECMAScript** (ES6, ES7, hasta ES2024 y más).

7. Comunidad y Soporte

- Amplia comunidad activa.
- Grandes repositorios y recursos en GitHub.
- Ecosistema en constante crecimiento con numerosas herramientas y bibliotecas.

3. Herramientas y Ecosistema de Desarrollo

- **Editores populares:** VS Code, WebStorm.
- **Gestores de paquetes:** npm, yarn, pnpm.
- **Transpiladores y Bundlers:** Babel (para usar código moderno en navegadores antiguos), Webpack, Rollup, Vite.
- **Testing:** Jest (tests unitarios), Cypress (tests end-to-end).
- **Linting y formateo:** ESLint, Prettier.



6. Frameworks y Tecnologías Asociadas

- **Next.js:** Framework React para aplicaciones web estáticas y server-side rendering.
- **Nuxt.js:** Framework Vue para aplicaciones universales y estáticas.
- **Deno:** Nuevo runtime para JavaScript y TypeScript, creado por el mismo autor de Node.js.

5. Actualizaciones y Evolución

- El lenguaje evoluciona con el estándar ECMAScript, aprobado y actualizado anualmente.
- Nuevas características: async/await, clases, módulos, operadores modernos, mejoras en arrays, objetos, etc.

4. Ecosistema de Módulos y Paquetes

- JavaScript utiliza un sistema de módulos (ES Modules y CommonJS).
- Gran ecosistema de paquetes disponibles en npm (Node Package Manager), el repositorio más grande de librerías JavaScript.

Ecosistema y herramientas modernas