JavaScript

JAVA-SCRIPT

Características básicas

- Lenguaje del lado del cliente.
- Interpretado
- Débilmente tipado
- Multiparadigma
- Case-sensitive
- Basado en eventos-monohilo

Tenemos un lenguaje versátil, que se ejecuta directamente en el navegador sin necesidad de compilación previa, interpretando el código línea por línea.

Gracias a su tipado débil y su enfoque multiparadigma, permite una gran flexibilidad al programar.

Estructuras de control

Permiten controlar el flujo del programa

- Condicionales: if, else, else if, switch
- Bucles: for, while, do...while
- Control de flujo: break, continue, return

Tipos de datos

JavaScript maneja dos categorías de tipos de datos: primitivos y objetos o otros

Primitivos
String, number, boolean, undefined, null, symbol ,bigint

Otros/estructurales
object, array, function

Funciones

Bloques de código reutilizables que ejecutan una tarea específica cuando se invocan.

- Declaradas
- Expresadas
- Flechas

Declaradas
Pueden usarse antes de declararse, tiene su propio this, sintaxis no muy conpacta

```
function saludar(nombre) {
  return "Hola, " + nombre;
}
```

Expresadas
No se puede usar antes de declararse, tiene this propio y su sintaxis no es tan compacta

```
const saludar = function(nombre) {
  return "Hola, " + nombre;
};
```

Fechas
No puede usarse antes de declararse, no tiene this propio, tiene sintaxis compacta

```
const saludar = (nombre) => "Hola, " + nombre;
```

Objetos y arrays

Son estructuras de datos fundamentales para almacenar y organizar información

Tenemos el clave valor y array

```
const persona = {
  nombre: "Fernanda",
  edad: 21
};
```

```
const frutas = ["manzana", "banana", "naranja"];
```

Eventos y manipulación del DOM

Permite acceder, modificar y reaccionar a elementos HTML en una página web, haciendo posible la interactividad.

```
document.getElementById("miBoton").addEventListener("click", function() {
  alert("¡Clic detectado!");
});
```

Asincronía

Permite ejecutar tareas sin bloquear el flujo del programa

Callbacks, Promises, async/await

Callbacks
Se ejecutan auto cuando una tarea asíncrona termina

```
setTimeout(() => { ... }, 1000);
```

Promises
Un valor que estará disponible a futuro

```
fetch("url").then(res => res.json());
```

async/await
promesas de forma legible

```
const res = await fetch("url");
```