Unidad 1

Introducción a JavaScript:

JavaScript es un lenguaje de programación interpretado, ligero y orientado a objetos que se utiliza principalmente en la web para crear páginas interactivas.
Fue creado por Brendan Eich en 1995 y se convirtió rápidamente en el lenguaje estándar para la programación del lado del cliente en navegadores web.
Tiene como propósito general:

- Agregar interactividad a las páginas web.
- Manipular el DOM (Document Object Model) para modificar la estructura y contenido del documento HTML.
- Realizar tareas complejas como validación de formularios, animaciones y comunicación con servidores (AJAX).

JavaScript pasó de ser un lenguaje básico para añadir interactividad en sitios web a un lenguaje versátil usado en frontend, backend (Node.js) y aplicaciones móviles.
Su ecosistema es el siguiente:

- Frameworks y Librerías Populares: React, Angular, Vue.js
- Node.js: Permite el uso de JavaScript en el servidor.

Características Clave:

- Lenguaje Interpretado: Se ejecuta directamente en el navegador sin necesidad de compilación.
- Dinámico y Débilmente Tipado: Los tipos de datos se asignan en tiempo de ejecución.
- Orientado a Objetos Basados en Prototipos: En lugar de clases tradicionales, utiliza prototipos para la herencia.
- Compatibilidad Universal: Funciona en todos los navegadores modernos sin necesidad de plugins adicionales.
- Versatilidad: Se puede usar tanto en el frontend como en el backend.

¿Dónde se utiliza JavaScript?:

- Desarrollo Web: Añadir interactividad a páginas web (validación de formularios, animaciones, etc.).
- Aplicaciones de Servidor: Usando Node.js
- Desarrollo de Aplicaciones Móviles: Frameworks como React Native.
- Juegos y Aplicaciones de Escritorio: Librerías como Electron.

Variables, Tipo de Datos y Operadores:

- Variables en JavaScript:
  ¿Qué es una variable?: Las variables son contenedores para almacenar datos.
  ¿Cómo se declara una variable?:
    - var: Declaración global o de función, obsoleta en ES6.
    - let: Declaración de bloque, introducida en ES6, preferida por su alcance limitado.
    - const: Declaración de bloque para valores constantes que no pueden ser reasignados.
- Tipos de Datos:
    - Primitivos:
      - String: Cadenas de texto ("Hola Mundo").
      - Number: Números (42, 3.14).
      - Boolean: Valores lógicos (true, false).
      - Undefined: Valor predeterminado de variables no inicializadas.
      - Null: Representa la ausencia intencional de un valor.
      - Symbol: Tipo de dato único y no mutable, introducido en ES6.
    - Compuestos:
      - Object: Colecciones de pares clave-valor.
      - Array: Lista ordenada de valores ([1, 2, 3]).
- Operadores:
    - Aritméticos: +, -, *, /, % para operaciones matemáticas.
    - Asignación: =, +=, -=, *=, /=, para asignar valores a variables.
    - Comparación:
      - == y ===: Comparación simple vs. estricta.
      - != y !==: Desigualdad simple vs. estricta.
    - Lógicos: &&, ||, ! para operaciones lógicas.
    - Condicional (Ternario): condición ? expr1 : expr2, para simplificar sentencias if.

Estructuras de Control:

- Condicionales: Permiten ejecutar bloques de código basados en la evaluación de expresiones booleanas.
  - Sentencia if:
    - Sintaxis básica:

```
if (condición) {
    // código a ejecutar si la condición es verdadera
}
```

  - Sentencia if else: Permite definir un bloque alternativo si la condición es falsa.
    - Sintaxis básica:

```
if (condición) {
    // código si la condición es verdadera
} else {
    // código si la condición es falsa
}
```

  - Sentencia else if: Para evaluar múltiples condiciones en secuencia.
    - Sintaxis básica:

```
if (condición1) {
    // código si la condición1 es verdadera
} else if (condición2) {
    // código si la condición2 es verdadera
} else {
    // código si ninguna de las condiciones anteriores es verdadera
}
```

- Operador Ternario:
  - Sintaxis: Una forma más compacta de escribir condicionales simples.

```
condición ? expr1 : expr2;
```

  - Ejemplo:

```
let resultado = (edad >= 18) ? 'Mayor de edad' : 'Menor de edad';
```

- Sentencia switch: Ideal para manejar múltiples posibles valores de una variable.
  - Sintaxis básica:

```
switch (expresión) {
    case valor1:
      // código a ejecutar si expresión === valor1
      break;
    case valor2:
      // código a ejecutar si expresión === valor2
      break;
    default:
      // código a ejecutar si ningún caso coincide
}
```

- Bucles: Permiten ejecutar un bloque de código repetidamente, mientras se cumpla una condición.
    - Bucle for:
        - Sintaxis básica:

```
for (inicialización; condición; incremento) {
    // código a ejecutar en cada iteración
}
```

        - Ejemplo:

```
for (let i = 0; i < 5; i++) {
    console.log(i);
}
```

    - Bucle while:
        - Sintaxis básica:

```
while (condición) {
    // código a ejecutar mientras la condición sea verdadera
}
```

        - Ejemplo:

```
let i = 0;
while (i < 5) {
    console.log(i);
    i++;
}
```

    - Bucle do while:
        - Diferencia con while: Ejecuta el bloque de código al menos una vez antes de evaluar la condición.
        - Sintaxis básica:

```
do {
    // código a ejecutar
} while (condición);
```

        - Ejemplo:

```
let i = 0;
do {
    console.log(i);
    i++;
} while (i < 5);
```

    - Bucle for of: Itera sobre objetos iterables (como arrays).
        - Sintaxis básica:

```
for (let valor of iterable) {
    // código a ejecutar para cada valor en el iterable
}
```

Funciones y Eventos en JavaScript:

- Funciones en JavaScript: Son bloques de código que realizan una tarea específica y pueden ser invocados desde cualquier parte del programa.
  - Declaración de funciones:
    - Sintaxis básica:

```
function nombreDeFuncion(param1, param2) {
    // código a ejecutar
    return resultado;
}
```

    - Ejemplo:

```
function sumar(a, b) {
    return a + b;
}
let resultado = sumar(3, 4);

* resultado será 7
```

  - Funciones Anónimas: Funciones sin nombre, generalmente asignadas a variables.
    - Sintaxis básica:

```
let suma = function(a, b) {
    return a + b;
};
```

  - Funciones Flecha (Arrow Functions): Una forma más compacta de escribir funciones, introducida en ES6.
    - Sintaxis básica:

```
const suma = (a, b) => a + b;
```

    - Ámbito de Variables (Scope) – Local vs. Global: Variables definidas dentro de una función (locales) y no accesibles fuera de ella.
- Eventos en JavaScript: Son acciones que ocurren en la página web, que pueden ser capturadas y manejadas mediante JavaScript.
  - Tipos de eventos comunes:
    - Eventos de Mouse: click, dblclick, mouseover, mouseout.
    - Eventos de Teclado: keydown, keyup, keypress.
    - Eventos de Formulario: submit, change, focus, blur.
  - Manejo de eventos:
    - Asignación de eventos con addEventListener:

```
elemento.addEventListener('click', function() {
    // código a ejecutar cuando se hace click en el elemento
});
```

- Eventos Inline vs. Externos:
  - Inline: Definidos directamente en el HTML (no recomendado).
  - Externos: Definidos en scripts separados, recomendados para una mejor organización.
- Propagación de eventos:
  - Bubbling vs. Capturing: Mecanismos de propagación de eventos a través del DOM.
  - Evitar la propagación: Uso de event.stopPropagation()