«Talento Tech»

Front End Js

CLASE 9

Buenos Aires aprende
Agencia de Habilidades para el Futuro

Buenos Aires Ciudad

Buenos Aires aprende
Agencia de Habilidades para el Futuro

Clase N° 09: Introducción a JavaScript

1. **Introducción a JavaScript**
    - ¿Qué es JavaScript?
    - Características de JavaScript
    - Versiones de ECMAScript
2. **Primeros Pasos con JavaScript**
    - Incorporación de JavaScript en HTML
    - Uso de la Consola del Navegador
    - Comentarios en JavaScript
3. **Variables y Constantes en JavaScript**
    - Declaración de Variables
    - Reglas para Nombrar Variables
    - Modificación de Variables
    - Declaración de Constantes
4. **Tipos de Datos en JavaScript**
    - Tipos de Datos Primitivos
    - El Objeto Number
    - Conversión Numérica
5. **Operadores**
    - Operadores Aritméticos y de Asignación
    - Concatenación de Cadenas

Buenos Aires aprende
Agencia de Habilidades para el Futuro

1. Introducción a JavaScript

¿Qué es JavaScript?

JavaScript, o, para los amigos, JS, es lo que le da vida a una página web.

Si HTML es el esqueleto y CSS la apariencia, **JavaScript es el alma** que permite que las cosas interactúen. Gracias a este lenguaje podés hacer que un botón muestre una alerta, que un formulario envíe datos sin recargar la página, o que cambie el contenido cuando movés el mouse.

Principales Características de JavaScript

- **Lenguaje del lado del cliente**: se ejecuta directamente en el navegador del usuario.
- **Orientado a objetos**: aunque no sea 100% orientado a objetos como otros lenguajes, JS te permite trabajar con objetos.
- **Tipado débil**: no tenés que especificar el tipo de dato (número, string, etc.) al declarar variables.
- **Interpretado**: no necesita compilación; el navegador se encarga de ejecutarlo al vuelo.

Versiones de ECMAScript

JavaScript evoluciona a través de algo llamado ECMAScript (o ES). Cada versión trae nuevas funcionalidades, más fácil de usar y más potente.

- **ES5 (2009)**: añadió cosas útiles como métodos de arrays y objetos, y el famoso "use strict".
- **ES6/ES2015**: esta es la "revolución" de JS. Añadió cosas como `let`, `const`, funciones flecha y clases.
- **ES2020/ES2021**: trajo nuevas herramientas como **BigInt** (números gigantes), promesas más sencillas, y operadores lógicos más eficientes.

2. Primeros Pasos con JavaScript

Incorporación de JavaScript en HTML

Podés agregar JavaScript a una página de varias maneras:

- **Directamente en el head** (aunque no se usa mucho porque bloquea el renderizado):

```
<script>
     console.log("Hola desde el head!");
</script>
```

- **En el body**, antes del cierre de `</body>`:

```
<body>
   <p>Ejemplo de texto.</p>
   <script>
     console.log("Hola desde el body!");
   </script>
 </body>
```

- **Usando un archivo externo**, **la mejor práctica**:

```
<script src="mi-script.js"></script>
```

Este último método es el más usado porque separa el código JS del HTML, manteniendo todo más organizado.

Buenos Aires aprende
Agencia de Habilidades para el Futuro

Uso de la Consola del Navegador

La consola del navegador es tu mejor amiga cuando estás trabajando con JS. Ahí podés ver lo que pasa en tu código, errores o directamente escribir y ejecutar JS.

Para abrir la consola:

- **Windows/Linux**: Ctrl + Shift + J
- **Mac**: Cmd + Option + J

Ejemplo:

```
console.log("¡Hola Mundo!");
```


```
console.log(2 + 2); // Imprime 4
```

Comentarios en JavaScript

Para hacer tu código más entendible o desactivar partes mientras probás, podés usar **comentarios**.

- Comentario de una sola línea:

```
// Esto es un comentario
```

Comentario de varias líneas:

```
/*
   Esto es un comentario
   de varias líneas
*/
```

3. Variables y Constantes en JavaScript

Declaración de Variables

En JS tenés tres maneras de declarar variables: **var**, **let** y **const**.

- **var**: la vieja confiable, pero con algunos problemas de alcance (scope). No la uses mucho si no es necesario.
- **let**: la opción moderna y segura.
- **const**: como su nombre indica, es para valores que no van a cambiar.

Ejemplos:

```
var nombre = "Juan";

let edad = 30;

const PI = 3.1416;
```

Reglas para Nombrar Variables

1. Las variables **empiezan** con una letra, guion bajo (_) o signo de dólar ($).
2. Son **case sensitive**, o sea que `nombre` y `Nombre` son diferentes.
3. Se usa **camelCase** para variables compuestas (`primerNombre`).

Modificación de Variables

Podés cambiar el valor de las variables, a menos que sean `const`.

```
let iva = 21;

iva = 10.5;

console.log(iva);  // Imprime 10.5
```

Declaración de Constantes

const es para cuando querés que algo nunca cambie.

```
const IVA = 21;

IVA = 10.5; // Te va a dar Error porque intentas asignarle otro valor y le habías dicho que era una constante.
```

4. Tipos de Datos en JavaScript

En JS, tenés varios tipos de datos:

- **Undefined**: cuando una variable no tiene valor.
- **Boolean**: verdadero (true) o falso (false).
- **Number**: números, tanto enteros como decimales.
- **String**: cadenas de texto.
- **Null**: ausencia intencionada de un valor.
- **BigInt**: para trabajar con números grandes.
- **Symbol**: para crear valores únicos.

El Objeto Number

Podés trabajar con números de dos maneras:

- **Como literales**:

```
let numero = 123;
```

- **Usando el constructor Number()**:

```
let numObjeto = new Number(123);
```

Conversión Numérica

Podés convertir strings en números con `parseInt()` o `parseFloat()`.

```
let numero = parseInt("42");

console.log(numero); // Imprime 42
```

También podés convertir usando diferentes bases:

```
let numBinario = parseInt("11101", 2); // Base binaria

console.log(numBinario); // Imprime 29
```

5. Operadores

Operadores Aritméticos y de Asignación

JS te permite hacer todas las operaciones matemáticas básicas:

- **Suma (+)**: suma dos números o concatena cadenas.
- **Resta (-)**: resta dos números.
- **Multiplicación (*)**: multiplica dos números.
- **División (/)**: divide dos números.
- **Módulo (%)**: da el resto de una división.
- **Incremento (++)** y **Decremento (--)**: suma o resta 1.

```
let x = 10;

x += 5;  // x ahora es 15

x++;     // x ahora es 16
```

Concatenación de Cadenas

El operador + también une cadenas de texto:

```
let saludo = "Hola, " + "Mundo!";

console.log(saludo);  // Imprime "Hola, Mundo!"
```

Con esta base sólida de JavaScript, podés empezar a darle interactividad a tus proyectos. Recordá que cuanto más practiques, más sentido te va a hacer todo.

Ejercicio práctico #1:

Operaciones con Variables y Tipos de Datos

Crear un programa que reciba dos números como entrada, realice varias operaciones aritméticas con ellos y muestre los resultados en la consola del navegador. Además, se deberá verificar si el resultado de la suma de ambos números es mayor o menor que un valor dado.

Tips:

1. **Validación de entradas:** asegurate de que los usuarios ingresen números válidos. Utilizá isNaN() para verificar que las entradas no sean texto u otros valores no numéricos.

Buenos Aires aprende
Agencia de Habilidades para el Futuro

2. **Uso de parseFloat() y parseInt():** dependiendo del ejercicio, recomendá que utilicen parseFloat() si los números pueden tener decimales, o parseInt() si solo aceptan números enteros.
3. **Descomposición del problema:** recordá que podés dividir el problema en partes más pequeñas. Primero capturá los números, luego realizá las operaciones, y por último verificá los resultados.
4. **Consola del navegador:** mostrá los resultados usando console.log() para que los estudiantes puedan ver los cálculos en la consola del navegador. Es una excelente herramienta para debuggear.

Ejercicio práctico #2:

Concatenación y Conversión de Tipos de Datos

Crear un programa que reciba el nombre y la edad de una persona, los concatene en una frase y luego convierta la edad de string a número para verificar si la persona es mayor de edad.

Tips:

1. **Validación de edad:** asegurate de que la edad ingresada sea un número válido antes de realizar cualquier operación. Usá isNaN() para evitar errores cuando el usuario ingresa texto o un valor vacío en lugar de un número.
2. **Concatenación de cadenas:** recordá que podés concatenar textos fácilmente con el operador +. Experimentá con diferentes formas de concatenar los valores para personalizar el mensaje de salida.
3. **Conversión de tipos:** usá parseInt() o Number() para convertir la edad de un string a un número, lo que es clave para realizar comparaciones o cálculos matemáticos.
4. **Mensajes claros:** asegurate de mostrar mensajes claros en la consola. Esto ayuda tanto al usuario como a vos mismo a entender si el programa está funcionando como esperabas.
5. **Pruebas con datos diferentes:** probá el programa con diferentes nombres y edades (incluyendo casos límite como menores de edad o números cercanos a 18) para verificar que la lógica del programa sea robusta.

Buenos Aires aprende

Agencia de Habilidades para el Futuro

Buenos Aires Ciudad