1. Fundamentos de JS

Es un lenguaje de programación que se ejecuta directamente en el navegador. Se puede aplicar a los documentos HTML y sirve para añadir interactividad a los sitios web.

Buenas Prácticas de Programación con JS

A continuación, les comparto algunas buenas prácticas de programación son importantes para iniciar con la programación con JS:

1. **Usa nombres descriptivos:** Utiliza nombres de variables, funciones y clases descriptivos que indiquen claramente su propósito y función en el código.
2. **Mantén el código limpio y legible:** Formatea y organiza tu código de manera consistente para facilitar su lectura y comprensión por parte de otros desarrolladores.
3. **Evita la globalización:** Minimiza el uso de variables globales y evita contaminar el ámbito global, ya que esto puede causar conflictos y errores inesperados.
4. **Usa const y let en lugar de var:** Prefiere declarar variables utilizando **const** y **let** en lugar de **var**, ya que tienen un ámbito de bloque más estricto y ayudan a prevenir errores.
5. **Evita la mutación de objetos y arrays:** En la medida de lo posible, evita la mutación directa de objetos y arrays y utiliza técnicas inmutables o métodos como **Object.assign()**, **Array.concat()**, y **Array.slice()** para crear nuevas instancias modificadas.
6. **Utiliza funciones puras:** Escribe funciones que sean puras, es decir, que no tengan efectos secundarios y produzcan el mismo resultado dado un conjunto de entradas.
7. **Maneja los errores adecuadamente:** Utiliza try-catch para manejar excepciones y errores de manera adecuada, y proporciona mensajes de error descriptivos y útiles para facilitar la depuración.
8. **Evita la duplicación de código:** Identifica y elimina duplicaciones de código utilizando funciones, clases, o patrones de diseño como el patrón módulo.
9. **Practica la modularidad:** Divide tu código en módulos y archivos separados según su funcionalidad, y utiliza sistemas de módulos como CommonJS o ES6 Modules para organizar y gestionar las dependencias.
10. **Comenta tu código:** Agrega comentarios descriptivos y significativos para explicar el propósito y el funcionamiento del código, especialmente en partes complejas o difíciles de entender.
11. **Realiza pruebas unitarias:** Escribe pruebas unitarias para verificar el comportamiento de tus funciones y componentes individualmente, y asegurarte de que funcionen como se espera.
12. **Optimiza el rendimiento:** Identifica y optimiza partes críticas de tu código para mejorar el rendimiento de tu aplicación, como bucles, operaciones de manipulación del DOM, y llamadas a API.
13. **Sigue las convenciones de estilo:** Adhiérete a una guía de estilo de código como StandardJS o ESLint para mantener la consistencia en todo el proyecto y evitar discusiones sobre el estilo de codificación.
14. **Actualiza regularmente tus dependencias:** Mantén al día las bibliotecas y frameworks que utilizas en tu proyecto para beneficiarte de las últimas características y correcciones de errores de seguridad.
15. **Sé consistente:** Sea consistente en el estilo, la estructura y las convenciones de nomenclatura en todo tu código y proyecto para facilitar la colaboración y el mantenimiento a largo plazo.
16. **Terminar las condicionales con “;**”: Se recomienda terminar las instrucciones con ; al final de cada línea de código.

NOTA: Al trabajar con JS estamos hablando de trabajar en un entorno independiente, por lo que debemos tener claro (de acuerdo a las buenas prácticas del desarrollo web que hemos aprendido en cursos anteriores) que debemos tener un nuevo archivo.

Ejemplo:

Podemos generar un nuevo archivo con la extensión .js

script.js

Y debemos vincularlo al archivo HTML con la etiqueta <script>:

<script src="scripts/main.js"></script>

VARIABLES

Las variables con los contenedores de los datos que vamos a usar, se recomienda usar let en lugar de var.

Ejemplo: let nombreDeLaVariable;

Es importante mencionar que JS distingue entre minúsculas y mayúsculas.

Los tipos de datos para las variables.

String

Boolean

Number

Array

Object

COMENTARIOS

Para escribir comentarios podemos usar doble diagonal “//” o diagonal con asterisco al inicio del comentario “/\*” y cerrarla con asterisco y diagonal al final de comentario “\*/”

Son símbolos matemáticos que pueden actuar sobre dos valores o variables y cambiar su resultado

CONDICIONALES

1. **DOM Manipulation:** Saber cómo interactuar con el Document Object Model (DOM) para acceder, modificar y manipular elementos HTML dinámicamente.
2. **Event Handling:** Entender cómo manejar eventos del usuario en el navegador y responder a ellos utilizando JavaScript.
3. **AJAX y Fetch API:** Conocer cómo realizar peticiones asíncronas al servidor web para cargar datos de manera dinámica sin necesidad de recargar la página.
4. **ES6+ (ECMAScript 2015+):** Familiarizarte con las características introducidas en las versiones recientes de JavaScript, como arrow functions, destructuring, template literals, classes, y otras.
5. **Programación Orientada a Objetos en JavaScript:** Dominar los conceptos de programación orientada a objetos (POO) en JavaScript, incluyendo la creación de clases, herencia, encapsulamiento y polimorfismo.
6. **Patrones de Diseño:** Conocer los patrones de diseño comunes en JavaScript, como el patrón módulo, patrón constructor, patrón observador, y otros, y saber cuándo y cómo aplicarlos.
7. **Frameworks y Bibliotecas Frontend:** Familiarizarte con frameworks y bibliotecas populares como React, Angular o Vue.js, así como herramientas como webpack o Babel.
8. **Testing:** Entender los principios de las pruebas unitarias y de integración en JavaScript, y saber cómo utilizar herramientas como Jest, Mocha o Jasmine para escribir y ejecutar pruebas.
9. **Optimización de rendimiento:** Saber cómo optimizar el rendimiento de tus aplicaciones frontend, incluyendo técnicas de optimización de carga inicial, reducción del tiempo de carga y mejora de la eficiencia del código.
10. **Accesibilidad:** Comprender los principios de accesibilidad web y cómo aplicarlos en el desarrollo frontend para garantizar que tus aplicaciones sean accesibles para todos los usuarios.
11. **Seguridad:** Conocer las vulnerabilidades comunes en aplicaciones web y cómo evitarlas mediante buenas prácticas de seguridad en el desarrollo frontend.