UNIDAD 2.

Desarrollo del Front end.

(El estudiante utilizará los elementos de una hoja de estilos y frameworks de hojas de estilos para la creación de un documento HTML.)

Tema III. Introducción a javascript.

Evidencia de aprendizaje Tema III –RESUMEN

Temas	Saber	Saber hacer	Resultado de Aprendizaje
Introducción a javascript	Describir los conceptos básicos de javascript. Describir los principales eventos de javascript. Identificar frameworks de javascript.	Integrar selectores y métodos de acceso al DOM. Diseñar interacciones mediante selectores, eventos y funciones.	Los estudiantes comprenden el uso de frameworks para el diseño responsivo de aplicaciones web.
Evidencia de aprendizaje			
A partir de un examen práctico con un tiempo definido, desarrollar una página web que contenga: el conjunto de páginas web estáticas y responsivas usando HTML, CSS, JS, Frameworks para el diseño responsivo.			

Información sobre el profesor

Profesor	Correo	Días de Clase
Bernardo Prado Díaz	Tenor_prado@yahoo.com.mx	Lunes y Martes

1. Conceptos Básicos de JavaScript

¿Qué es JavaScript?

JavaScript (JS) es un **lenguaje de programación** de alto nivel, interpretado y orientado a objetos. Su función principal es permitir la creación de páginas web dinámicas e interactivas. Se ejecuta directamente en el navegador del usuario, lo que lo hace ideal para manipular la interfaz sin necesidad de recargar la página.

Variables

Las variables son contenedores para almacenar datos. Existen tres formas principales de declararlas:

- var: Es la forma antigua y tiene un alcance (scope) global o de función.

- let: Es la forma moderna para variables que pueden cambiar de valor. Tiene un alcance de bloque.
- const: Es la forma moderna para variables cuyo valor **no va a cambiar**. También tiene un alcance de bloque.

Tipos de Datos

JavaScript tiene varios tipos de datos, entre los más comunes están:

- **String:** Una cadena de texto (ej. "Hola Mundo").
- **Number:** Números enteros o decimales (ej. 10, 3.14).
- **Boolean:** Valores lógicos true o false.
- **Array:** Una lista ordenada de valores (ej. ["manzana", "pera", "uva"]).
- **Object:** Una colección de pares clave-valor (ej. { nombre: "Juan", edad: 30 }).

Selectores y Acceso al DOM

Para que JavaScript pueda interactuar con una página web, debe poder acceder a los elementos HTML. El **DOM (Document Object Model)** es la representación en forma de árbol del documento HTML, y JavaScript tiene métodos para "seleccionar" elementos de ese árbol.

- document.getElementById('id-del-elemento'): Selecciona un elemento por su ID único. Es el método más rápido y directo.
- document.querySelector('.clase-del-elemento'): Selecciona el primer elemento que coincida con el selector CSS.
- document.querySelectorAll('p'): Selecciona **todos** los elementos que coincidan con el selector CSS y devuelve una lista.

2. Eventos de JavaScript

Un **evento** es una acción o suceso que ocurre en la página web. Los eventos son la forma en que JavaScript detecta la interacción del usuario y responde a ella. Los eventos más comunes son:

- click: Cuando el usuario hace clic en un elemento.
- mouseover: Cuando el cursor se mueve sobre un elemento.
- submit: Cuando un formulario es enviado.
- keypress: Cuando el usuario presiona una tecla.

Para escuchar un evento, se usa el método addEventListener(). Por ejemplo, miBoton.addEventListener('click', miFuncion); le dice al navegador que ejecute miFuncion cada vez que se haga clic en miBoton.

3. Frameworks de JavaScript

Un **framework** o **librería** de JavaScript es un conjunto de código pre-escrito que simplifica y acelera el desarrollo de aplicaciones web complejas. Se construyen sobre los fundamentos de JavaScript y el DOM que acabas de aprender. Los más populares son:

- **React:** Creado por Facebook, ideal para construir interfaces de usuario basadas en componentes.
- **Vue.js:** Un framework progresivo, fácil de aprender y muy flexible.
- **Angular:** Creado por Google, es un framework robusto para aplicaciones de gran escala.

Estos frameworks te ayudan a manejar tareas como la gestión del estado de la aplicación y la renderización eficiente del DOM, pero todos se basan en los conceptos que estás a punto de practicar.

1. Conceptos Básicos de JavaScript

JavaScript es un lenguaje de programación que permite agregar interactividad a las páginas web. Se ejecuta en el navegador y trabaja junto con HTML y CSS.

Variables, Tipos de Datos y Operadores

let nombre = "Ana"; // Se declara una variable llamada 'nombre' y se le asigna el valor de texto "Ana"

let edad = 21; // Se declara una variable llamada 'edad' y se le asigna el valor numérico 21

let estudiante = true; // Se declara una variable booleana que indica si es estudiante

let mensaje = "Hola, " + nombre; // Se crea una nueva variable que concatena texto con el valor de 'nombre'

console.log(mensaje); // Se muestra el contenido de 'mensaje' en la consola del navegador

2. Eventos en JavaScript

Los **eventos** permiten responder a acciones del usuario como clics, teclas o movimientos del mouse.

Ejemplo: Evento click

html

```
<button id="miBoton">Haz clic aquí</button> <!-- Botón con un ID para identificarlo -->

<script>

 document.getElementById("miBoton").addEventListener("click", function() {

   alert("¡Has hecho clic!"); // Muestra una alerta cuando se hace clic en el botón

 }); // Se agrega un 'escuchador' de eventos al botón que responde al evento 'click'

</script>
```

3. Selectores y Acceso al DOM

El **DOM (Document Object Model)** representa la estructura HTML como objetos que JavaScript puede manipular.

Ejemplo: Acceder y modificar contenido

html

```
<p id="parrafo">Texto original</p> <!-- Párrafo con un ID para acceder desde JavaScript -->

<script>

 let elemento = document.getElementById("parrafo"); // Se obtiene el elemento con ID 'parrafo'

 elemento.textContent = "Texto modificado"; // Se cambia el contenido del párrafo

</script>
```

4. Diseñar Interacciones con Selectores, Eventos y Funciones

Ejemplo: Cambiar color al hacer clic

html

```
<button id="colorBtn">Cambiar color</button>

<p id="texto">Este texto cambiará de color</p>


<script>

 function cambiarColor() { // Se define una función llamada 'cambiarColor'

   let texto = document.getElementById("texto"); // Se accede al elemento con ID 'texto'
```

```
  texto.style.color = "blue"; // Se cambia el color del texto a azul
 }

 document.getElementById("colorBtn").addEventListener("click", cambiarColor); // Se ejecuta la función cuando se hace clic en el botón
</script>
```

5. Frameworks de JavaScript

Los **frameworks** ayudan a construir aplicaciones web más rápido y con mejor estructura.

Framework	Características principales
React	Componentes reutilizables, muy usado en empresas
Vue.js	Fácil de aprender, ideal para principiantes
Angular	Completo y robusto, usado en aplicaciones grandes

6. Diseño Responsivo con Frameworks

Para diseño responsivo, se suelen usar **frameworks CSS** como **Bootstrap**, que se integran fácilmente con JavaScript.

Ejemplo: Usar Bootstrap + JS

html

```
<!-- Incluye Bootstrap desde CDN -->
<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstrap.min.css" rel="stylesheet">

<button id="toggleBtn" class="btn btn-primary">Mostrar/Ocultar</button>
<div id="contenido" class="mt-3">Este contenido puede ocultarse</div>

<script>
 document.getElementById("toggleBtn").addEventListener("click", function() {
```

```
  let contenido = document.getElementById("contenido"); // Se accede al div con ID 'contenido'
  if (contenido.style.display === "none") { // Se verifica si el contenido está oculto
    contenido.style.display = "block"; // Si está oculto, se muestra
  } else {
    contenido.style.display = "none"; // Si está visible, se oculta
  }
 });
</script>
```

7. Ejercicio Final Integrador

Objetivo: Crear una mini app que cambia el texto y color al hacer clic

html

```
<h2 id="titulo">Bienvenido</h2>
<button id="accionBtn">Cambiar mensaje</button>

<script>
 function cambiarMensaje() { // Se define una función que cambia el mensaje y color
  let titulo = document.getElementById("titulo"); // Se accede al elemento con ID 'titulo'
  titulo.textContent = "¡Hola desde JavaScript!"; // Se cambia el texto del título
  titulo.style.color = "green"; // Se cambia el color del texto a verde
 }

 document.getElementById("accionBtn").addEventListener("click", cambiarMensaje); // Se ejecuta la función cuando se hace clic en el botón
</script>
```

Fundamentos de JavaScript

Ejercicio 1: Variables y Tipos de Datos

javascript

```
let nombre = "Carlos"; // Se declara una variable llamada 'nombre' y se le asigna el texto "Carlos"

let edad = 20; // Se declara una variable numérica llamada 'edad' con valor 20

let estudiante = true; // Se declara una variable booleana que indica si es estudiante


console.log(nombre); // Muestra el valor de la variable 'nombre' en la consola

console.log(edad); // Muestra el valor de la variable 'edad' en la consola

console.log(estudiante); // Muestra el valor de la variable 'estudiante' en la consola
```

Eventos en JavaScript

Ejercicio 2: Evento click

html

```
<button id="saludoBtn">Saludar</button> <!-- Botón con ID 'saludoBtn' -->


<script>

 document.getElementById("saludoBtn").addEventListener("click", function() {

  alert("¡Hola, bienvenido!"); // Muestra una alerta cuando se hace clic en el botón

 }); // Se agrega un evento 'click' al botón que ejecuta la función anónima

</script>
```

Selectores y Acceso al DOM

Ejercicio 3: Modificar contenido de un elemento

html

```
<p id="mensaje">Texto original</p> <!-- Párrafo con ID 'mensaje' -->


<script>

 let parrafo = document.getElementById("mensaje"); // Se accede al elemento con ID 'mensaje'
```

```
  parrafo.textContent = "Texto actualizado con JavaScript"; // Se modifica el contenido del párrafo

</script>
```

Interacciones con Selectores, Eventos y Funciones

Ejercicio 4: Cambiar color y texto al hacer clic

html

```
<h2 id="titulo">Bienvenido</h2> <!-- Título con ID 'titulo' -->

<button id="cambiarBtn">Cambiar mensaje</button> <!-- Botón con ID 'cambiarBtn' -->


<script>

 function cambiarMensaje() { // Se define una función llamada 'cambiarMensaje'

   let titulo = document.getElementById("titulo"); // Se accede al elemento con ID 'titulo'

   titulo.textContent = "¡Hola desde JavaScript!"; // Se cambia el texto del título

   titulo.style.color = "blue"; // Se cambia el color del texto a azul

 }


  document.getElementById("cambiarBtn").addEventListener("click", cambiarMensaje); // Se ejecuta la función al hacer clic en el botón

</script>
```

Frameworks de JavaScript

¿Qué es un framework?

Un **framework** es un conjunto de herramientas y estructuras que facilitan el desarrollo de aplicaciones web. Algunos populares:

Framework	Ventajas
React	Componentes reutilizables, gran comunidad
Vue.js	Fácil de aprender, ideal para principiantes

Framework Ventajas

Angular	Completo, usado en aplicaciones empresariales

Diseño Responsivo con Bootstrap

Mostrar/Ocultar contenido con Bootstrap y JS

html

```
<!-- Importar Bootstrap -->

<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstrap.min.css" rel="stylesheet">



<button id="toggleBtn" class="btn btn-primary">Mostrar/Ocultar</button> <!-- Botón con clase de Bootstrap -->

<div id="contenido" class="mt-3">Este contenido puede ocultarse</div> <!-- Contenido que se mostrará u ocultará -->



<script>

 document.getElementById("toggleBtn").addEventListener("click", function() {

  let contenido = document.getElementById("contenido"); // Se accede al div con ID 'contenido'

  if (contenido.style.display === "none") { // Si el contenido está oculto

   contenido.style.display = "block"; // Se muestra el contenido

  } else {

   contenido.style.display = "none"; // Se oculta el contenido

  }

 });

</script>
```

Actividad Final: Mini App Interactiva

Objetivo: Crear una app que cambia el texto y color al hacer clic

```
html
<h1 id="saludo">¡Hola!</h1> <!-- Título con ID 'saludo' -->
<button id="accion">Cambiar saludo</button> <!-- Botón con ID 'accion' -->

<script>
 function actualizarSaludo() { // Se define la función 'actualizarSaludo'
   let saludo = document.getElementById("saludo"); // Se accede al elemento con ID 'saludo'
   saludo.textContent = "¡Bienvenido a tu primera app!"; // Se cambia el texto del saludo
   saludo.style.color = "green"; // Se cambia el color del texto a verde
 }

 document.getElementById("accion").addEventListener("click", actualizarSaludo); // Se ejecuta la función al hacer clic en el botón
</script>
```

Fundamentos clave

1. ¿Qué es JavaScript y cómo integrarlo en HTML sin consola?
JavaScript es un lenguaje de programación interpretado (su sintaxis toma influencias de C++ y Java), orientado a objetos, basado en prototipos, dinámico y débilmente tipado
En HTML, se puede incluir dentro de etiquetas <script> o cargar desde archivos externos, y manipular el contenido directamente en la página usando el DOM. Por ejemplo:

document.getElementById("miElemento").textContent = "Texto visible";

Esto permite mostrar resultados en HTML sin recurrir a console.log.

2. Objetos y expresiones en JavaScript

- Un **objeto** es una estructura que agrupa propiedades y métodos. Ejemplo:
- const persona = {
- nombre: "Ana",
- saludar() {
- return `Hola, soy ${this.nombre}`;
- }
- };
- Una **expresión** es cualquier porción de código que produce un valor. Puede ser literal (2 + 2), una llamada a un método (persona.saludar()), o una operación del DOM (...textContent = ...).

Ejemplo práctico: mostrar en página sin usar consola

<!DOCTYPE html>

<html lang="es">

<head>

<meta charset="UTF-8">

<title>JS sin console.log</title>

</head>

<body>

```
 <h2 id="titulo"></h2>
 <p id="mensaje"></p>

 <script>
  const estudiante = {
   nombre: "Beatriz",
   curso: "Introducción a JavaScript",
   presentacion() {
    return `Hola, soy ${this.nombre} y estoy aprendiendo ${this.curso}.`;
   }
  };

  document.getElementById("titulo").textContent = "Presentación Estudiante";
  document.getElementById("mensaje").textContent = estudiante.presentacion();
 </script>
</body>
</html>
```

- Define un objeto estudiante con propiedades y método.
- Usa expresiones y luego coloca su resultado directamente en el HTML.
- No hay console.log—el output se ve en la página.

Resumen breve

Recurso/Concepto	**Descripción**
Video objetos en <10 min	Rápido y centrado en cómo crear objetos y usarlos dentro del HTML.
Video JS en 10 min	Introducción general a JavaScript para principiantes.

Recurso/Concepto	Descripción
Conceptos clave	JavaScript como lenguaje de programación dinámico; objetos y expresiones; integración directa en HTML sin consola.
Ejemplo funcional	Objeto con método que se muestra en página mediante DOM.

- El **HTML** define la estructura (tarjeta del alumno, menú de meses, lista de tareas y retroalimentación).
- El **CSS** se encarga de los estilos modernos y responsivos.
- El **JavaScript** añade interactividad: mostrar tareas por mes, limpiar la lista y enviar retroalimentación.

Lo bueno es que tu código ya **no depende de console.log**, todo se muestra directamente en el HTML (DOM) o con alert().

Cómo funciona cada parte con objetos y expresiones

1. Objeto con tareas (JS)

En tu archivo script.js, defines un objeto tareas con las listas por mes:

```
const tareas = {
  enero: [
    { texto: 'Matemáticas: Ejercicio 1', url: 'https://tusitio.com/ejercicio1' },
    { texto: 'Historia: Lectura capítulo 2', url: 'https://tusitio.com/capitulo2' }
  ],
  febrero: [
    { texto: 'Ciencias: Experimento casero', url: 'https://tusitio.com/experimento' },
    { texto: 'Lengua: Redacción libre', url: 'https://tusitio.com/redaccion' }
  ]
};
```

Esto es un **objeto con propiedades** (meses) y cada propiedad tiene **arrays de objetos** (las tareas).
Cada tarea es una **expresión de objeto**: {texto: "...", url: "..."}.

---

2. Expresiones en acción

Cuando llamas a mostrarTareas("enero"), recorres con forEach y creas un <li> con esta expresión:

li.innerHTML = `<a href="${tarea.url}" target="_blank">${tarea.texto}</a>`;

Aquí, la expresión con **template string** (${...}) inserta dinámicamente el valor de tarea.texto y tarea.url en el HTML.

---

3. Mostrar en HTML

- La lista de tareas se muestra dentro del <ul id="lista-tareas">.
- El comentario se valida y se responde con un alert("¡Gracias...!").

No hay console.log, todo es visible en la página.

Ejemplo final adaptado para tu proyecto

Si quieres mostrar una tarea destacada directamente en tu HTML al cargar la página:

```
<script>
 const tareaDestacada = { texto: "Revisar retroalimentación", url: "#" };
 document.getElementById("lista-tareas").innerHTML = `
   <li><a href="${tareaDestacada.url}">${tareaDestacada.texto}</a></li>
 `;
</script>
```

Así aparece automáticamente en la lista sin usar consola.