

# Unidad 1

Selección de arquitecturas y  
herramientas de programación

# Contenido

Unidad 1 .....	1
Selección de arquitecturas y herramientas de programación.....	1
Objetivos .....	4
1.1.    Introducción .....	5
1.2.    FRONT-END y BACK-END .....	5
1.3.    Características de los lenguajes de script.....	5
1.4.    Ventajas de los lenguajes de script sobre la programación tradicional: .....	6
1.5.    Desventajas de los lenguajes de script sobre la programación tradicional:.....	6
1.6.    Capacidades de Ejecución de JavaScript: .....	7
1.6.1.    Interactividad en el Lado del Cliente:.....	7
1.6.2.    Limitaciones de Ejecución de JavaScript: .....	7
1.7.    Lenguaje de programación en entorno cliente .....	8
1.8.    Frameworks.....	9
1.8.1.    ReactJS.....	9
1.8.2.    AngularJS .....	10
1.8.3.    Vue.JS .....	10
1.9.    Tecnologías y Lenguajes Asociados .....	10
1.9.1.    HTML: La Estructura de la Web .....	11
1.9.2.    CSS: Estilizando la Web.....	12
1.9.3.    Integración del Código con las Etiquetas HTML .....	12
1.9.4.    Integración de Archivos de Script en Páginas HTML .....	13
1.9.5.    Escribir el Código JavaScript .....	14
1.10.    Herramientas y Entornos de Desarrollo .....	14
1.10.1.    Visual Studio Code: Un Entorno Potente y Versátil .....	15
1.10.2.    Sublime Text: Simplicidad y Elegancia en un Editor de Texto.....	16
1.12.    Posibilidades que nos brinda JavaScript.....	17
Modificación del contenido de una página web .....	17
Cambio de imágenes con JavaScript .....	17
Cambiar atributos de objetos HTML .....	17
Escribir en la consola del navegador con console.log().....	18
Escribir en cualquier elemento HTML utilizando el atributo innerHTML.....	18
Generar directamente HTML utilizando el método document.write() .....	19
Generar un mensaje de alerta utilizando el método window.alert() .....	19

1.11.	Practica Guiada:	20
1.11.1.	Manipulación del DOM:	20
1.11.2.	Validación de Formulario:	22
	Resumen:	24
	Ejercicios :	25
	Bibliografía .....	26

## Objetivos

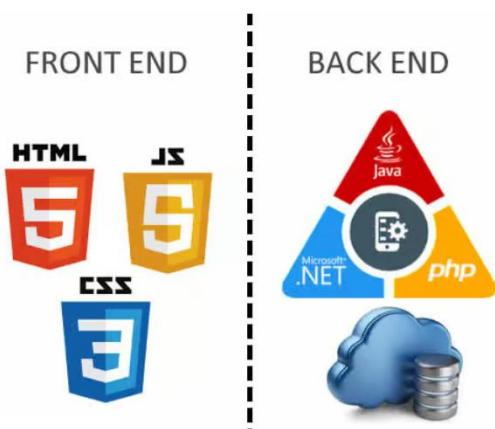
- Comprender los modelos de ejecución de código
- Conocer los lenguajes relacionados con la programación de clientes web
- Explorar la integración de lenguajes de marcas y lenguajes de programación
- Evaluar herramientas de programación para clientes web
- Aplicar conocimientos en proyectos prácticos
- Fomentar la curiosidad y la actualización constante

## 1.1. Introducción

Cualquier página web de Internet está construida, como mínimo, por HTML (un lenguaje de marcas) y CSS (un lenguaje de estilos). El primero de ellos permite construir todo el marcado de la página (contenido e información) mediante etiquetas HTML y dotando de semántica a la información mediante la naturaleza de dichas etiquetas. Posteriormente, el segundo de ellos permite darle estilo a la página y construir una interfaz visual más agradable para el usuario.

## 1.2. FRONT-END y BACK-END

Actualmente, se suele catalogar el desarrollo web en dos partes back-end (la parte no visible de la web, como las bases de datos o los scripts que se ejecutan en el servidor) y front-end (la parte visible de una web, como las hojas de estilo, el código HTML, los scripts que se ejecutan en el lado del cliente). A continuación, se describirá con más profundidad este modelo cliente servidor.



En las empresas desarrolladoras de aplicaciones web, suele haber técnicos especialistas en back-end y front-end. Los técnicos de back-end se encargan de todo el proceso en el lado del servidor, como el acceso a la base de datos (MySQL, MaríaDB, PostgreSQL, MongoDB, Oracle), creación de servicios, etc. Estos técnicos programarán en lenguajes como PHP, Ruby on Rails, Django, Node.js, .NET, etc.

## 1.3. Características de los lenguajes de script.

Los lenguajes de script son lenguajes de programación diseñados para facilitar tareas específicas, como automatizar procesos, interactuar con otras aplicaciones o crear contenido dinámico en páginas web. A continuación, exploraremos las principales características de los lenguajes de script y analizaremos las ventajas y desventajas en comparación con la programación tradicional.

**Interpretados:** Los lenguajes de script son interpretados en tiempo de ejecución, lo que significa que no requieren un proceso de compilación previo. Esto permite una mayor flexibilidad y

agilidad en el desarrollo, ya que los cambios pueden implementarse de inmediato sin tener que recompilar el código.

**Sintaxis sencilla:** Los lenguajes de script tienden a tener una sintaxis sencilla y legible, lo que facilita su aprendizaje y comprensión. Esto los convierte en una opción popular para principiantes en la programación.

**Flexibilidad:** Los lenguajes de script son altamente flexibles y permiten realizar modificaciones y ajustes rápidos en el código. Esto es especialmente útil en entornos donde los requisitos cambian con frecuencia o se necesita una respuesta rápida a las demandas del usuario.

**Amplia disponibilidad:** Existen numerosos lenguajes de script disponibles para diferentes propósitos, como JavaScript, Python, Ruby y PHP. Esto proporciona a los desarrolladores una amplia gama de opciones según las necesidades del proyecto.

#### 1.4. Ventajas de los lenguajes de script sobre la programación tradicional:

**Rapidez de desarrollo:** La naturaleza interpretada de los lenguajes de script permite un desarrollo más rápido debido a la eliminación del proceso de compilación. Los cambios y las pruebas pueden realizarse de forma iterativa, acelerando el ciclo de desarrollo.

**Facilidad de prototipado:** Gracias a su sintaxis sencilla y flexibilidad, los lenguajes de script son ideales para el prototipado rápido de aplicaciones. Esto permite a los desarrolladores experimentar y probar ideas antes de invertir tiempo y recursos en una implementación completa.

**Acceso a bibliotecas y frameworks:** Los lenguajes de script tienen una gran cantidad de bibliotecas y frameworks disponibles que facilitan tareas comunes y agilizan el desarrollo. Estas herramientas permiten a los desarrolladores aprovechar el trabajo previo de la comunidad y acelerar el proceso de desarrollo.

#### 1.5. Desventajas de los lenguajes de script sobre la programación tradicional:

**Rendimiento:** Debido a su naturaleza interpretada, los lenguajes de script pueden tener un rendimiento inferior en comparación con los lenguajes compilados. Esto puede ser una limitación en aplicaciones que requieren un procesamiento intensivo o un tiempo de respuesta rápido.

**Dependencia del entorno de ejecución:** Los lenguajes de script dependen de un entorno de ejecución específico, como un intérprete o un navegador web. Esto puede generar problemas de compatibilidad y limitar la portabilidad del código en comparación con los lenguajes compilados.

**Menor control de recursos:** En algunos casos, los lenguajes de script pueden tener un control más limitado sobre los recursos del sistema, como la gestión de memoria o la manipulación de archivos. Esto puede ser un factor a considerar en aplicaciones que requieren un control preciso de los recursos.

Los lenguajes de script ofrecen una serie de características y ventajas que los hacen atractivos en muchos escenarios de desarrollo. Permiten un desarrollo rápido, prototipado ágil y acceso a una amplia gama de bibliotecas y frameworks. Sin embargo, también presentan desventajas en términos de rendimiento y control de recursos. Al seleccionar un lenguaje de programación, es importante evaluar cuidadosamente los requisitos del proyecto y considerar tanto las ventajas como las desventajas de los lenguajes de script en comparación con la programación tradicional.

## 1.6. Capacidades de Ejecución de JavaScript:

### 1.6.1. Interactividad en el Lado del Cliente:

**Capacidad:** JavaScript permite la creación de páginas web dinámicas, donde los elementos pueden responder a la interacción del usuario sin necesidad de recargar la página.

#### Manipulación del DOM:

**Capacidad:** JavaScript puede manipular dinámicamente el DOM, permitiendo la adición, eliminación o modificación de elementos HTML y CSS en tiempo real.

#### Comunicación Asíncrona:

**Capacidad:** La ejecución asíncrona de JavaScript permite realizar solicitudes a servidores web sin bloquear la ejecución de otras operaciones.

#### Programación Orientada a Eventos:

**Capacidad:** JavaScript sigue un modelo de programación orientada a eventos, permitiendo la captura y gestión de eventos como clics, teclas presionadas, etc.

## 1.6.2. Limitaciones de Ejecución de JavaScript:

#### Seguridad del Cliente:

**Limitación:** JavaScript se ejecuta en el navegador del cliente, lo que puede plantear riesgos de seguridad si no se manejan adecuadamente.

#### Rendimiento:

**Limitación:** Aunque JavaScript ha mejorado en rendimiento, puede ser menos eficiente que lenguajes compilados para operaciones intensivas.

#### Compatibilidad con Navegadores Web:

#### Estándares Web:

**Compatibilidad:** JavaScript es ampliamente compatible con los estándares web, gracias al trabajo de los comités de estandarización.

#### Herramientas de Desarrollo:

**Compatibilidad:** La mayoría de los navegadores ofrecen herramientas de desarrollo que facilitan la depuración y el análisis del código JavaScript.

## 1.7. Lenguaje de programación en entorno cliente

La programación web en el lado del cliente se basa en tres pilares fundamentales que se citan a continuación:

**El lenguaje HTML.** No es un lenguaje de programación, sino un lenguaje de marcado. El HTML define el contenido que va a tener el documento. La función del navegador web será la de leer e interpretar todo este contenido, junto con las etiquetas, y visualizarlo al cliente. La ventaja del código HTML es que cualquier navegador debería visualizar el contenido de la página web de la misma forma y con el mismo aspecto.

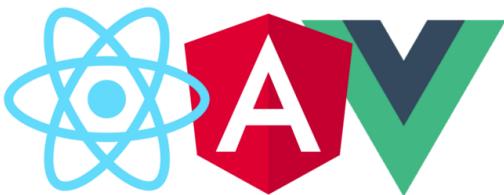
**El lenguaje CSS.** Define la presentación del documento. CSS es un lenguaje de diseño gráfico y su objetivo es que la página web sea atractiva al usuario. No modifica el comportamiento ni el contenido, sino el aspecto de la página web.

**El lenguaje JavaScript.** El código o lenguaje JavaScript agrega el contenido dinámico a las páginas web. JavaScript es un verdadero lenguaje de programación, a diferencia de los lenguajes HTML y CSS.



Actualmente, las empresas no programan directamente sobre JavaScript, sino basándose en un framework de este. A continuación, se citarán algunos de los frameworks más utilizados para JavaScript.

## 1.8. Frameworks



Los frameworks nacieron como librerías, más o menos completas, que tenían una serie de estructuras que permitían al programador tener una base para la creación y el desarrollo de sus proyectos. Hoy, son mucho más que eso, puesto que pueden utilizar lenguajes como TypeScript o JSX, los cuales luego se compilan a JavaScript.

Entre las ventajas que aportan los frameworks, están:

El coste. Muchos de estos frameworks son de **código abierto**, con lo cual no hay que realizar ninguna inversión.

Están suficientemente probados y su código suele **carecer de errores**, puesto que muchos programadores lo utilizan. Además, suelen tener un **alto nivel de seguridad** y rendimiento.

Permite **desarrollar mucho más rápido**, puesto que muchas de las estructuras, clases, patrones de diseño, etc., vienen ya incorporadas en el framework.

Cualquier persona que maneja un framework determinado puede entender e incorporarse de forma eficiente y efectiva a un equipo de desarrollo que lo esté utilizando en un proyecto determinado.

### 1.8.1. ReactJS



ReactJS (<https://reactjs.org/>) es un framework **creado por Facebook** que permite a los programadores realizar aplicaciones web de una forma rápida y eficiente renderizando (dibujando) los componentes del front-end de una manera sencilla y eficaz.

React utiliza **programación orientada a componentes** (que no objetos). Los componentes gestionan su propio estado y, cuando se agrupa una serie de componentes, los programadores son capaces de ir creando las interfaces de usuario.

Una de las características de React es que **utiliza un DOM virtual** que mapea los objetos desde este hasta el DOM del navegador.

### 1.8.2. AngularJS



Angular (<https://angular.io>) fue creado y es **mantenido por Google**. La primera versión de Angular se denominó AngularJS y todavía hay una comunidad utilizando este framework por su fácil integración con JavaScript. Las versiones sucesivas de Angular se denominan Angular a secas y ya ha dejado de ser una simple librería para pasar a ser una plataforma de desarrollo.

El problema con este y otros frameworks es que su curva de aprendizaje es bastante pronunciada, dado que no son fáciles de aprender.

Actualmente, en Angular, se programa en TypeScript, que es un superconjunto de JavaScript desarrollado por Microsoft y utiliza el patrón reactivo RxJS.

### 1.8.3. Vue.js



Una de las características de este framework (<https://vuejs.org>) frente a otros es la ligereza y la velocidad de ejecución. El objetivo que se plantearon sus desarrolladores y diseñadores fue el crear un framework con las mejores ventajas de los existentes. A diferencia de Angular, su curva de aprendizaje no es tan pronunciada y los desarrolladores de Laravel (un framework de backend) lo utilizan para usarlo en el front-end de sus aplicaciones.

Al igual que React, utiliza un DOM virtual, dadas las ventajas que ofrece este tipo de implementaciones.

## 1.9. Tecnologías y Lenguajes Asociados

### 1.9.1. HTML: La Estructura de la Web

El HTML es el esqueleto de todas las páginas web. Define la estructura y el contenido mediante etiquetas, que son elementos que delimitan partes del contenido como encabezados, párrafos y enlaces.



(Figura con página de ejemplo)

Ejemplo de código HTML con estructura básica para la web anterior:

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8">
    <title>Página de ejemplo</title>
</head>
<body>
    <header>
        <nav>
            <ul>
                <li><a href="#">Inicio</a></li>
                <li><a href="#">Sobre nosotros</a></li>
                <li><a href="#">Contacto</a></li>
            </ul>
        </nav>
    </header>
    <main>
        <section>
            <h1>Nuestros Servicios</h1>
            <article>
                <h2>Desarrollo Web</h2>
                <p>Ofrecemos servicios de desarrollo web personalizados.</p>
            </article>
        </section>
    </main>
    <footer>
        <p>© 2024 Desarrollo Web en Entorno Cliente</p>
    </footer>
</body>
</html>
```

### 1.9.2. CSS: Estilizando la Web

CSS es el lenguaje que usamos para estilizar y diseñar nuestras páginas HTML. Permite controlar colores, fuentes, espaciado, y mucho más.

Ejemplo de código CSS para estilizar la página anterior:

```
body {  
    font-family: 'Segoe UI', sans-serif;  
    margin: 0;  
    padding: 0;  
    box-sizing: border-box;  
}  
  
header, footer {  
    background-color: #333;  
    color: white;  
    text-align: center;  
    padding: 1em 0;  
}  
  
nav ul {  
    list-style: none;  
}  
  
nav ul li {  
    display: inline;  
    margin-right: 20px;  
}  
  
nav ul li a {  
    text-decoration: none;  
    color: white;  
}  
  
main {  
    padding: 20px;  
}  
  
h1, h2 {  
    color: #333;  
}
```

### 1.9.3. Integración del Código con las Etiquetas HTML

La integración de JavaScript en HTML se realiza a través de la etiqueta <script>. Puede ser directamente en el código HTML o referenciando un archivo externo.

Ejemplo de incorporación de JavaScript en HTML:

```
<!DOCTYPE html>
<html>
<head>
    <title>Integración de JavaScript</title>
</head>
<body>
    <h1>Evento de clic en JavaScript</h1>
    <button id="miBoton">Haz clic aquí</button>

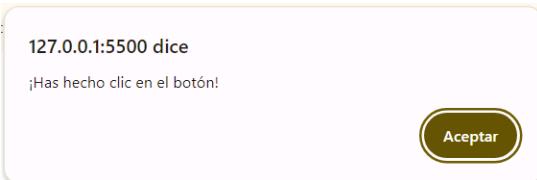
    <script>
        let variable = document.getElementById('miBoton');
        variable.addEventListener('click', escribir);

        function escribir () {
            alert('¡Has hecho clic en el botón!');
        }

    </script>
</body>
</html>
```

## Evento de clic en JavaScript

(Observa como se ve el programa que acabamos de crear )



( Dialogo obtenido utilizando el método alert)

### 1.9.4. Integración de Archivos de Script en Páginas HTML

Para comenzar, necesitamos un archivo donde escribir nuestro código JavaScript. Abre tu editor de código preferido y crea un nuevo archivo. Guarda este archivo con la extensión .js, por ejemplo, script.js. Aquí es donde escribiremos nuestras funciones y lógica JavaScript.

```
// script.js
function saludar() {
    alert('¡Hola, mundo!');
}
```

#### 1.9.5. Escribir el Código JavaScript

Dentro del archivo script.js, puedes escribir cualquier código JavaScript que deseas integrar en tu página web. Por ejemplo, podrías definir funciones para manejar eventos, validar formularios o interactuar con el usuario.

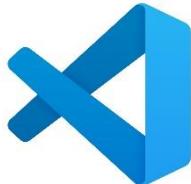
```
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <title>Integración de Script</title>
    <!-- Incluimos el archivo de script -->
    <script src="script.js"></script>
</head>
<body>
    <h1>Integración de Script</h1>
    <button onclick="saludar()">Saludar</button>
</body>
</html>
```

#### 1.10. Herramientas y Entornos de Desarrollo

Para desarrollar aplicaciones web en entorno cliente con JavaScript, es fundamental contar con las herramientas y entornos de desarrollo adecuados. Editores de código como Visual Studio Code, Sublime Text y Atom son populares entre los desarrolladores web. Además, los

navegadores modernos vienen equipados con herramientas integradas de desarrollo que facilitan la depuración y el análisis del código JavaScript.

#### 1.10.1. Visual Studio Code: Un Entorno Potente y Versátil



(Logo de ID visual Studio Code)

Visual Studio Code, desarrollado por Microsoft, se ha convertido en una de las herramientas preferidas por los desarrolladores web de todo el mundo. Su interfaz intuitiva, su amplia gama de extensiones y su integración con otras herramientas de desarrollo lo convierten en una opción poderosa para cualquier proyecto.

*Características Destacadas:*

- **Interfaz Intuitiva:** Visual Studio Code presenta una interfaz de usuario minimalista y fácil de navegar, lo que facilita la escritura de código sin distracciones.
- **Amplia Gama de Extensiones:** Con una vasta biblioteca de extensiones disponibles, los desarrolladores pueden personalizar su entorno de desarrollo según sus necesidades específicas. Desde extensiones para lenguajes de programación hasta herramientas de control de versiones, Visual Studio Code tiene algo para todos.
- **Integración con Git:** La integración nativa con Git permite a los desarrolladores gestionar fácilmente el control de versiones de sus proyectos directamente desde el editor.
- **Depuración Integrada:** Visual Studio Code ofrece potentes herramientas de depuración que permiten a los desarrolladores identificar y corregir errores en su código de manera eficiente.

#### Extensiones en Visual Studio Code

Las extensiones de Visual Studio Code son como pequeñas aplicaciones que se integran directamente en el editor, ampliando sus capacidades y adaptándose a las necesidades específicas de los desarrolladores. Ya sea que estés trabajando con HTML, CSS, JavaScript o cualquier otro lenguaje de programación, existe una extensión que puede mejorar tu flujo de trabajo y aumentar tu productividad.

A continuación, exploraremos algunas de las extensiones más valiosas para el desarrollo web en Visual Studio Code y cómo pueden marcar una diferencia significativa en tu experiencia de codificación:

- **Live Server:** Esta extensión es imprescindible para cualquier desarrollador web que trabaje con HTML, CSS y JavaScript. Al levantar un pequeño servidor local, Live Server actualiza automáticamente la página web en tu navegador a medida que realizas cambios en tus archivos, ahorrándote el tedioso proceso de tener que actualizar manualmente el navegador cada vez.

- Auto Rename Tag: Cuando estés escribiendo HTML, esta extensión automáticamente actualizará la etiqueta de cierre correspondiente al modificar la etiqueta de apertura. Una pequeña pero muy práctica ayuda para ahorrar tiempo y evitar errores comunes.
- Bracket Pair Colorizer: Esta extensión te permitirá identificar fácilmente los pares de corchetes, llaves y paréntesis coincidentes al colorearlos de acuerdo a su alcance. Resulta invaluable cuando trabajas con código anidado y complejo.
- ESLint: Un linter de código estático para JavaScript que te ayudará a mantener un estilo de código consistente y detectar problemas de patrones antes de ejecutar tu código. Puedes configurar ESLint según tus preferencias de codificación y las convenciones de tu equipo.
- Prettier: Un formateador de código muy popular que mantiene tu código JavaScript, HTML y CSS bellamente formateado de acuerdo a las reglas que configures. Prettier te ahorrará muchos dolores de cabeza al trabajar en equipo y te permitirá mantener un estilo de código coherente en todo tu proyecto.

#### [1.10.2. Sublime Text: Simplicidad y Elegancia en un Editor de Texto](#)



(Logo de ID sublime text)

Sublime Text ha sido durante mucho tiempo una opción popular entre los desarrolladores web por su velocidad, simplicidad y potentes capacidades de edición de texto. Aunque puede carecer de algunas características avanzadas de Visual Studio Code, su enfoque minimalista lo convierte en una opción atractiva para aquellos que prefieren una experiencia de desarrollo más simplificada.

*Características Destacadas:*

- **Rendimiento Veloz:** Sublime Text es conocido por su velocidad y rendimiento, lo que lo convierte en una opción ideal para proyectos de cualquier tamaño.
- **Edición Eficiente:** Con funciones como la selección múltiple y la edición de múltiples líneas, Sublime Text permite a los desarrolladores editar rápidamente su código de manera eficiente.
- **Personalización Flexible:** Aunque Sublime Text puede carecer de la amplia gama de extensiones de Visual Studio Code, su sistema de personalización permite a los usuarios adaptar el editor a sus necesidades específicas.
- **Atajos de Teclado Potentes:** Sublime Text ofrece una amplia variedad de atajos de teclado que agilizan el flujo de trabajo del desarrollador y aumentan la productividad.

## 1.12. Posibilidades que nos brinda JavaScript

JavaScript, el alma de las páginas web dinámicas, nos permite crear experiencias interactivas y atractivas para los usuarios. Aprender su sintaxis y posibilidades es como abrir una puerta a un mundo de creatividad y control sobre el comportamiento de nuestros sitios web.

### 1.12.1. Modificación del contenido de una página web

Imagina poder cambiar el texto, las imágenes o cualquier elemento de una página web al vuelo, sin necesidad de recargar. JavaScript te da ese poder. Con el atributo innerHTML de los elementos HTML, puedes acceder y modificar su contenido interno como si fuera un lienzo en blanco.

```
// Cambiar el texto de un párrafo con id="mensaje"
document.getElementById("mensaje").innerHTML = "¡Hola desde JavaScript!";
```

### 1.12.2. Cambio de imágenes con JavaScript

Las imágenes estáticas son cosa del pasado. Con JavaScript, puedes crear galerías interactivas, cambiar imágenes al pasar el ratón o incluso animarlas. La clave está en manipular el atributo src de los elementos <img>.

```
// Cambiar la imagen al hacer clic
function cambiarImagen() {
    let imagen = document.getElementById("miImagen");
    imagen.src = "nueva_imagen.jpg";
}
```

### 1.12.3. Cambiar atributos de objetos HTML

JavaScript te permite controlar todos los aspectos de los elementos HTML, desde su tamaño y color hasta su posición y estilo. Puedes cambiar el atributo style directamente o utilizar clases CSS para una mayor flexibilidad.

```
// Agregar una clase CSS a un elemento
document.getElementById("miParrafo").classList.add("resaltado");
```

#### 1.12.4. Escribir en la consola del navegador con console.log()

La consola del navegador es tu ventana al funcionamiento interno de tu código JavaScript. Con console.log(), puedes imprimir mensajes, valores de variables y cualquier información que te ayude a depurar y entender tu programa.

```
// Imprimir un mensaje en la consola
console.log("El valor de la variable x es:", x);
```

#### 1.12.5. Escribir en cualquier elemento HTML utilizando el atributo innerHTML

innerHTML no solo sirve para leer el contenido de un elemento, también puedes usarlo para escribir nuevo contenido HTML directamente. Esto te permite crear elementos dinámicamente o actualizar el contenido existente con información variable.

```
<!DOCTYPE html>
<html>
<head>
  <title>Ejemplo de innerHTML</title>
</head>
<body>
  <div id="miElemento"></div>

  <script>
    // Obtén una referencia al elemento HTML
    var elemento = document.getElementById("miElemento");

    // Escribe contenido HTML en el elemento utilizando innerHTML
    elemento.innerHTML = "<h1>Hola, mundo!</h1>";

  </script>
</body>
</html>
```

# Hola, mundo!

(Como podemos observar si estamos agregando una etiqueta nueva desde html o desde JS se ve de la misma manera, el aspecto lo cambiaremos con css)

Práctica:

Copia el código de esta sección y prueba que te funcione, luego agrégale una etiqueta h2 y un párrafo con la leyenda

## 1.12.6. Generar directamente HTML utilizando el método document.write()

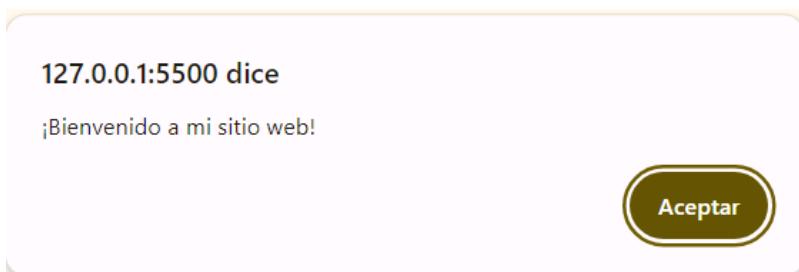
Si necesitas generar una gran cantidad de código HTML dinámicamente, document.write() te permite hacerlo directamente desde tu código JavaScript. Sin embargo, ten en cuenta que usar este método puede sobrescribir el contenido existente de la página.

```
// Escribir un párrafo en la página
document.write("<p>Este párrafo fue generado con JavaScript.</p>");
```

## 1.12.7. Generar un mensaje de alerta utilizando el método window.alert()

Las alertas son una forma simple de llamar la atención del usuario y mostrar mensajes importantes. Con window.alert(), puedes crear ventanas emergentes con un mensaje y un botón de "Aceptar".

```
// Mostrar una alerta al usuario
window.alert("¡Bienvenido a mi sitio web!");
```



(Dialogo emergente utilizando el metodo windows.alert())

### 1.13. Práctica Guiada:

#### 1.13.1. Manipulación del DOM:

**Objetivo:** Adquirir habilidades prácticas en el desarrollo web front-end mediante la creación de una página web interactiva utilizando HTML, CSS y JavaScript.

**Duración:** 3 horas.

**Solución:** Desarrollaremos una página web simple que contenga un botón. Al hacer clic en el botón, debería cambiar el texto de un elemento HTML (por ejemplo, un párrafo o un encabezado) utilizando JavaScript para manipular el DOM. Puedes cambiar el texto a cualquier mensaje que deseas.

Para lograr la manipulación del DOM, puedes seguir estos pasos:

Crea un archivo HTML con un botón y un elemento (por ejemplo, un párrafo) cuyo contenido quieras cambiar.

Agrega un evento al botón utilizando JavaScript. Cuando se haga clic en el botón, ejecuta una función que modifique el contenido del elemento.

Aquí tienes un ejemplo básico:

```
<!DOCTYPE html>
<html>
<head>
    <title>Manipulación del DOM</title>
</head>
<body>
    <p id="miParrafo">Texto original</p>
    <button onclick="cambiarTexto()">Cambiar Texto</button>
```

```
<script>
    function cambiarTexto() {
        document.getElementById("miParrafo").textContent = "Nuevo
texto";
    }
</script>
</body>
</html>
```

### 1.13.2. Validación de Formulario:

**Objetivo:** Desarrollar habilidades en la validación de formularios utilizando JavaScript, asegurando que los datos ingresados por el usuario cumplan con los requisitos establecidos.

**Duración:** 2 horas

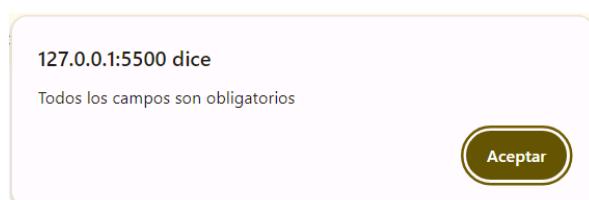
**Descripción de la actividad:**

Crearemos un formulario HTML con campos como nombre, correo electrónico y contraseña. Utiliza JavaScript para validar que los campos no estén vacíos y que el correo electrónico tenga un formato válido. Muestra mensajes de error si los datos no cumplen con los criterios.

The form consists of four input fields and one button:

- Nombre
- Correo electrónico
- Contraseña
- Enviar

(Vista del formulario)



(Dialogo que se muestra cuando alguno de los campos del formulario no está completo)

```
<!DOCTYPE html>
<html>
<head>
    <title>Validación de Formulario</title>
```

```

</head>
<body>
    <form onsubmit="return validarFormulario()">
        <input type="text" id="nombre" placeholder="Nombre">
        <input type="email" id="correo" placeholder="Correo electrónico">
        <input type="password" id="contrasena" placeholder="Contraseña">
        <button type="submit">Enviar</button>
    </form>

    <script>
        function validarFormulario() {
            const nombre = document.getElementById("nombre").value;
            const correo = document.getElementById("correo").value;
            const contrasena =
document.getElementById("contrasena").value;

            if (nombre === "" || correo === "" || contrasena === "") {
                alert("Todos los campos son obligatorios");
                return false;
            }

            // Validación del formato del correo electrónico (puedes usar
expresiones regulares)
            // ...

            return true;
        }
    </script>
</body>
</html>

```

#### **1.14. Resumen:**

- JavaScript es un lenguaje de programación esencial para el desarrollo web del lado del cliente (front-end)
- Permite crear páginas web interactivas y dinámicas
  - Los elementos pueden responder a las acciones del usuario sin recargar la página
  - Se puede manipular el DOM (Modelo de Objetos del Documento)
- Se integra con HTML (estructura y contenido) y CSS (estilo y presentación visual)
  - HTML, CSS y JavaScript forman el trío fundamental del desarrollo web front-end
- Existen frameworks populares que facilitan el desarrollo con JavaScript:
  - ReactJS
  - AngularJS
  - Vue.js
  - Brindan estructuras y herramientas preestablecidas para agilizar el desarrollo
- Para integrar JavaScript en páginas web, se utiliza la etiqueta <script>
  - Se puede incrustar el código directamente en el HTML
  - Es recomendable crear un archivo externo con extensión .js y vincularlo
- Herramientas y entornos de desarrollo populares:
  - Visual Studio Code
  - Sublime Text
  - Ofrecen características útiles como resaltado de sintaxis, autocompletado y depuración
- Mantenerse curioso y practicar constantemente
  - Explorar nuevas tecnologías y herramientas
  - El desarrollo web del lado del cliente es emocionante y dinámico

## 1.15. Ejercicios :

**Objetivo:** Practicar habilidades como la manipulación del DOM, la interacción con el usuario, la gestión de eventos, la salida de datos y la comprensión de las diferencias entre lenguajes de programación.

**Duración:** 8 horas.

A continuación, les presento una serie de actividades prácticas relacionadas con JavaScript. Estas actividades les permitirán aplicar y reforzar los conceptos clave que hemos estudiado en el curso. Recuerden que la práctica es fundamental para consolidar los conocimientos adquiridos y desarrollar habilidades sólidas en programación.

1. Desarrollen un test de 7 preguntas. Cada pregunta debe tener dos botones (verdadero y falso). Si el usuario hace clic en el botón correcto, la frase debe mostrarse en verde. Si elige la opción incorrecta, la frase debe mostrarse en rojo.
2. Crean una página web con un botón que, al pulsarlo, genere un mensaje en la consola del navegador.
3. Diseñen una web con tres botones. Al pulsar cada botón, se debe generar un mensaje de bienvenida en la misma página dentro de un párrafo (<p>). Los mensajes deben tener colores diferentes según el idioma seleccionado.
4. Modifiquen el código de la actividad anterior para que los mensajes de bienvenida se generen utilizando alertas en lugar de párrafos.
5. Cambien el código para que los mensajes de bienvenida se muestren en la consola del navegador.
6. Transformen el código para que los mensajes de bienvenida se muestren utilizando el método write().

### 1.16. Bibliografía

- Mozilla Developer Network (MDN) URL: <https://developer.mozilla.org/>
- W3Schools URL: <https://www.w3schools.com/>
- DesarrolloWeb.com URL: <https://desarrolloweb.com/>
- AMX Web URL: <https://amxweb.com/>