Javier Gómez Delgado

EL DESARROLLO WEB DESDE EL ENTORNO CLIENTE

UNA VISIÓN *FULL STACK DEVELOPER*

ESIC

El desarrollo web desde el entorno cliente

Una visión *Full Stack Developer*

Madrid, 2023

Javier Gómez Delgado

El desarrollo web desde el entorno cliente

Una visión *Full Stack Developer*

ESIC

Septiembre, *2023*

El desarrollo web desde el entorno cliente. Una visión Full Stack Developer
Javier Gómez Delgado

Avda. de Valdenigriales, s/n
28223 Pozuelo de Alarcón (Madrid)
Tel.: 91 452 41 00
www.esic.edu/editorial
@EsicEditorial

ISBN: 978-84-1192-001-8
Depósito Legal: M-27416-2023

Diseño de cubierta: Balloon Comunicación
Maquetación: Santiago Díez Escribano
Lectura: Balloon Comunicación
Impresión: Gráficas Dehon

Un libro de

ESIC Editorial

Impreso en España – *Printed in Spain*

Este libro ha sido impreso con tinta ecológica y papel sostenible.

Índice

1

Selección de arquitecturas y herramientas de programación

Objetivos de aprendizaje:

- Diferenciar modelos de ejecución de código en el servidor y en el cliente web.
- Identificar los principales lenguajes relacionados con la programación de clientes web.
- Conocer las formas de integrar el código de programación sobre los lenguajes de marcas.
- Aprender a utilizar las herramientas más comunes para la codificación.

Palabras clave: Front, Back, Visual Studio Code, CodePen, JavaScript.

1.1. Introducción

Cuando en el año 1989 Tint Berners-Lee creó la web en el laboratorio europeo de partículas (CERN), nadie en ese momento podría imaginar en lo que se convertiría décadas más tarde.

En la actualidad, el W3C es el consorcio que se encarga de desarrollar los estándares para el desarrollo de Internet. El W3C es un organismo abierto y cualquiera puede unirse a sus grupos y participar en los blogs u otras discusiones. En España, el sitio web del W3C es **www.w3c.es**.

El sitio web del W3C es www.w3c.es

Ahora mismo, muchas compañías de cualquier tamaño ofrecen gran cantidad de servicios a través de la web, desde el comercio electrónico hasta formas de trabajo remoto para sus empleados. Gracias a la deslocalización de clientes y trabajadores, muchas empresas están migrando sus sistemas tradicionales dentro de los servidores de la empresa a aplicaciones alojadas en servidores de Internet (computación en la nube). Por ello, plataformas como AWS (Amazon Web Services) o Azure están creciendo de forma exponencial. Por estas y otras razones el puesto de desarrollador web está siendo muy demandado en el mercado.

1.2. Arquitectura

El mundo del desarrollo web abarca muchas disciplinas, las tareas del diseñador web no se parecen en nada a las de un desarrollador *back*, un arquitecto de *software* o un administrador de base de datos (DBA).

El desarrollo web se puede dividir en dos partes, la parte *back* (no visible de la web, como las bases de datos o los *scripts* que se ejecutan en el servidor) y *front* (la parte visible de una web, como las hojas de estilo, el código HTML y los *scripts* que se ejecutan en el lado del cliente).

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, MariaDB, PostgreSQL, MongoDB, Oracle), creación de servicios, etc. Estos técnicos programarán en lenguajes como Java, PHP, Ruby on Rails. Django, Nodejs, .NET, etc.

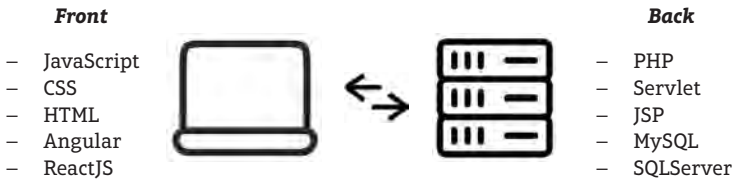


Figura 1.1. División entre código *front* y *back*

En la parte *front*, el desarrollador necesitará conocer los lenguajes que los navegadores son capaces de interpretar. Será necesario crear la estructura de la página con HTML, aplicar los estilos con CSS y utilizar JavaScript para dar dinamismo a la web.

Para ayudar a la programación, de manera que no sea necesario escribir todo el código desde cero, nacieron los `frameworks`. Estos `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.

Lo importante del Front es la usabilidad, deben crearse herramientas útiles para los usuarios y que sean fácilmente entendibles, y todo el código escrito debe estar al servicio de esto.

1.3. Lenguajes de programación en entorno cliente

Existen tres lenguajes importantes en la programación en el entorno cliente:

- **HTML**. No es un lenguaje de programación, sino un lenguaje de marcado. El HTML define la estructura que va a tener el documento. La función del navegador web será la de leer e interpretar todo este contenido, y visualizarlo al cliente. La ventaja del código HTML es que cualquier navegador va a visualizar el contenido de la página web de la misma forma y con el mismo aspecto.
- **CSS**. Define la presentación del documento. CSS es un lenguaje de diseño gráfico y su objetivo es dar el diseño a la estructura creada en HTML. No modifica el comportamiento ni el contenido, sino el aspecto de la página web.

- **JavaScript**. El código o lenguaje JavaScript agrega el contenido dinámico a las páginas web. HTML y CSS no son lenguajes de programación a diferencia de JavaScript que sí es un verdadero lenguaje de programación.

No siempre se programa directamente en JavaScript, sino que existen una serie de `frameworks` que ayudan en esta programación. Los más utilizados son los siguientes:

- **ReactJS**
  ReactJS (https://react.dev) 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.
- **Angular**
  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.
- **Polymer**
  Polymer es una librería desarrollada por Google que facilita la creación y el uso de Web Components reutilizables. Estos componentes web son elementos personalizados que encapsulan la funcionalidad, la estructura y el diseño de zonas o páginas, para poderse utilizar en diferentes sitios de la web.
- **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 características de los ya existentes. A diferencia de Angular, su aprendizaje no es tan difícil. Al igual que React, utiliza un DOM virtual, dadas las ventajas que ofrece este tipo de implementaciones.

Escoger un `framework` es un proceso importante, puesto que el objetivo es elegir el más popular para que tenga un soporte más amplio, el que tenga más futuro, el más rápido de ejecución, el más fácil de aprender. Dicha elección muchas veces es complicada, ya que la tecnología cambia de forma muy rápida y las curvas de aprendizaje de un `framework` son elevadas.

1.4. Integración del código con las etiquetas HTML

La forma de ejecutar el código en JavaScript es incluyendo las sentencias dentro de las etiquetas HTML. Existen tres maneras de incluir JavaScript dentro de las páginas web.

JavaScript en el mismo documento HTML

Para incluir JavaScript en el mismo documento, se encierra entre las etiquetas `<script>`. Aunque puede colocarse este código en cualquier parte del documento, lo recomendable es escribir este código en la cabecera del HTML, dentro de las etiquetas `<head>`.

A la hora de incluir el código dentro del HTML, será necesario tener en cuenta que si se ejecuta un código JavaScript que utiliza elementos del HTML, estos elementos HTML tendrán que estar antes que el código JavaScript. Si no es así, se producirá un error en ejecución y dirá que no ha encontrado estos elementos.

Código 1.1.
JavaScript y HTML en el mismo documento

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width,
initial-scale=1.0">
    <title>JavaScript dentro del documento</title>
    <script>
        alert("Mensaje");
    </script>
</head>
<body>
    <p>Muestra un mensaje</p>
</body>
</html>
```

Cuando lo que se quiere es incluir pequeños bloques de código o complementar código HTML con ciertas funcionalidades, se emplea este método de incluir JavaScript.

El problema que tiene esta forma de incluir JavaScript es que cuando se quiere realizar una modificación de una parte del código, es necesario repasar todas las páginas en donde está incluido y realizar la modificación en todas ellas.

Definir JavaScript en un archivo externo

Las instrucciones JavaScript se pueden incluir en un archivo externo de tipo JavaScript que los documentos HTML enlazan mediante la etiqueta <script>. Se pueden crear todos los archivos JavaScript que sean necesarios, y cada documento HTML puede enlazar tantos archivos JavaScript como necesite.

Ejemplo:

Código 1.2. JavaScript en otro documento

```
Archivo J102.js

alert("Un mensaje de prueba");

HTML

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width,
initial-scale=1.0">
    <title>JavaScript en diferente documento</title>
    <script type="text/JavaScript" src="js/J102.js"></script>
</head>
<body>
    <p>Muestra un mensaje</p>
</body>
</html>
```

Aparte del atributo `type`, está el atributo `src`, que especifica la URL del archivo JavaScript que se quiere incluir. Dentro de cada etiqueta `<script>` solo se puede incluir un archivo, pero dentro de la página HTML se pueden definir tantas etiquetas `<script>` como sean necesarias.

Un archivo JavaScript son documentos de texto que tienen la extensión `.js`, y que pueden ser creados con un editor de texto como Notepad, Visual Studio Code, Sublime, vi, etc.

Este es el mejor método para incluir código JavaScript en las páginas HTML. De esta manera el código está siendo reutilizado, simplificando el código de la página, y si se necesita modificar el JavaScript, con modificar en un solo sitio, queda automáticamente reflejado este cambio en todas las páginas HTML que tienen incluido este código.

Incluir JavaScript en los elementos HTML

Por último, se menciona un método menos utilizado, y es la forma de incluir código JavaScript dentro de los elementos HTML de una página:

Código 1.3. JavaScript dentro del HTML

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width,
    initial-scale=1.0">
    <title>JavaScript dentro del HTML</title>
    </script>
</head>
```

```
<body>
    <button onclick="alert('Mensaje');">Sacar el mensaje</
button>
</body>
</html>
```

Con esta forma de incluir JavaScript se ensucia bastante el código HTML de la página, complicando la lectura y el mantenimiento de la codificación. Solo se debe utilizar para la definición de eventos y algún caso muy especial.

1.5. Herramientas

Una de las ventajas que tiene la codificación en JavaScript es que podemos utilizar cualquier tipo de editor, ya que solo necesitamos crear un archivo de texto que no tenga ningún tipo de marcador propio, como hace Microsoft Word, por ejemplo.

Podemos optar por editores sencillos que vienen con nuestro Sistema Operativo, como `vi` o `Notepad`. Pero en la actualidad tenemos bastantes editores que nos proporcionan gran cantidad de ayuda para la codificación de nuestro código *front*.

Hay muchas características que comparten los editores de código JavaScript más utilizados, como pueden ser el autocompletado, tener capacidad de editar diferentes lenguajes, integración con Git y la capacidad de incluir complementos (*plug-in*).

1.6. Visual Studio Code

En 2015 Microsoft saca Visual Studio Code, un editor de código fuente gratuito, de código abierto y bastante ligero tras la instalación. Este es el editor más fácil para los desarrolladores principiantes de JavaScript, en parte porque está precargado con un buen conjunto de funcionalidades que no requieren complementos adicionales.

No solo ayuda a los lenguajes de Microsoft, sino que soporta la mayoría de los lenguajes *open source*, como Java, Go, C, C++, Ruby, Python, PHP, Perl, JavaScript, Groovy, Swift, PowerShell, Rust, DockerFile, CSS, HTML, XML, JSON, Lua, F#, Batch, SQL, Objective-C…

Otra ventaja añadida es que Visual Studio Code fue diseñado para que opere en los tres sistemas operativos más utilizados en la actualidad: Windows, Linux y Mac OS.

Instalación de Visual Studio Code

Microsoft Visual Studio Code
https://code.visualstudio.com/

Para poder instalar Visual Studio Code será necesario seguir los siguientes pasos:

- Ir a la página de Microsoft Visual Studio Code y hacer clic en el botón "Descargar Visual Studio Code", teniendo en cuenta el sistema operativo para descargar el archivo de instalación.
- Ejecutar el archivo de instalación que se ha descargado en el paso anterior. Esto dará inicio a la instalación.

- Leer y aceptar la licencia.
- A continuación, se puede cambiar la ubicación de la carpeta de instalación o mantener la configuración predeterminada.
- También se puede cambiar la denominación de la carpeta de acceso directo en el menú de Inicio o si no se quiere tener estos accesos directos.
- Se continúa seleccionando diferentes opciones, como crear el icono en el escritorio o incluir las opciones de Visual Studio en el menú contextual de Windows.
- Ya solo queda hacer clic en Install para comenzar el proceso de instalación. A continuación, el programa está instalado y listo para usar.
- Por último, se finaliza la instalación y se puede comenzar a usar el programa.

Operaciones básicas con Visual Studio Code

Crear un archivo

En el menú superior de opciones se selecciona Archivo y a continuación Nuevo archivo. También se puede utilizar el atajo `Ctrl + N`.

En este momento se ha abierto una ventana de texto a la derecha que todavía no tiene una sintaxis definida. Para determinar el tipo de archivo que se quiere editar se debe pulsar en la opción "Seleccionar un lenguaje" que aparece en esta misma ventana. En este momento aparecerá una lista de lenguajes. Aquí se debe seleccionar HTML si se va a incluir el JavaScript dentro del HTML, o JavaScript si lo que se quiere es crear un archivo donde esté solo el código.

Figura 1.2. Selección del tipo de archivo en Visual Studio Code

Si no se selecciona ningún tipo de archivo, no realizará ningún control de sintaxis del documento que estamos editando, es decir, no aparecerán las etiquetas HTML con color ni hará sugerencias de instrucciones. Esto aparecerá cuando se haya guardado el archivo la primera vez y se indique la extensión al archivo.

Desde el menú de opciones se puede grabar el archivo, para ello hay que elegir Archivo/Guardar, o utilizar el atajo `Ctrl + S`.

A continuación, aparecerá una ventana de diálogo en donde se introduce el nombre del archivo y la carpeta donde guardarlo.

En este momento ya está guardado el archivo y si no se ha seleccionado tipo de archivo en un primer momento, a partir de aquí, el editor ya reconoce la sintaxis.

En el momento que se introduce un cambio con el editor Visual Studio Code se puede ver que en la pestaña del documento ha cambiado la cruz por un círculo. Esto significa que el código fuente se ha modificado y no ha sido guardado. Para guardar estos cambios se puede ir a las opciones de menú Archivo/Guardar o `Ctrl + S`.

Con las pestañas podemos tener varios archivos abiertos de forma simultánea, estando cada uno en una pestaña diferente.

Cambio entre pestañas

Hay varios métodos para cambiar de pestaña, puede ser a través del ratón, o también mediante la tecla `Alt` y las teclas numéricas (se selecciona la tercera pestaña presionando `Alt+3`).

Otra forma de cambiar de pestaña es utilizando las teclas `Ctrl` más alguna de las teclas `Av Pág.` y `Re Pág`.

Grabar todos los archivos modificados

Si se han modificado varios archivos de diferentes pestañas, no es necesario ir uno a uno guardándolos. Se puede utilizar una opción de menú que guarda todos los archivos que se tienen abiertos en las pestañas. La forma de comprobar esta opción es modificando varios archivos situados en diferentes pestañas. A continuación, se puede ver en el primer icono a la izquierda que se ven los archivos que han sido modificados y todavía no se han grabado.

Para grabar todos los archivos modificados es Archivo/Guardar todo o el atajo de teclado `Ctrl + K`, se sueltan ambas teclas y luego se pulsa la tecla S.

Cerrar pestañas

Hay varias maneras de cerrar pestañas en Visual Studio Code:

- Con el ratón se puede pulsar sobre la cruz de la pestaña. Si se ha modificado y no se ha grabado, se pulsa sobre el circulo y pedirá confirmación para guardar el archivo antes de cerrarlo.
- Cerrar la pestaña activa con las teclas `Ctrl + W` o también `Ctrl + F4`.
- Seleccionar los tres puntos que se ven en la esquina superior derecha. En este momento aparecerán dos opciones: "cerrar todos", en donde cierra todas las pestañas y "cerrar los guardados" en donde solo cierra los que tienen una x en la pestaña.

Abrir archivos

Para abrir un archivo que está guardado en el disco se seleccionan las opciones del menú: Archivo/Abrir archivo.

También con el atajo `Ctrl + O` se puede hacer aparecer la ventana de archivos.

Otra forma práctica de abrir un archivo es arrastrándolo desde el explorador de Windows hasta el propio editor.

También desde el propio explorador de Windows se puede abrir el menú contextual del archivo y seleccionar Abrir con Code, siempre y cuando en la instalación se haya dicho que se quiere el menú contextual.

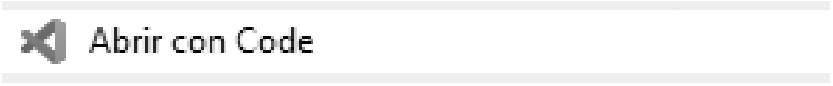


Figura 1.3. Menú contextual del explorador de Windows

Por último, se puede ver en la parte derecha de Visual Studio Code toda una carpeta con los archivos fuente que contiene. Esto se puede hacer arrastrando la carpeta directamente desde el explorador de Windows a la parte izquierda de Visual Studio Code.

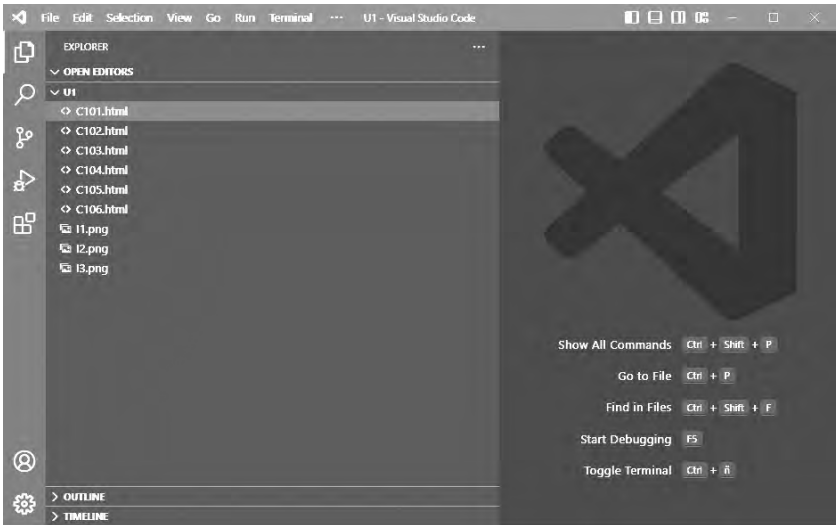


Figura 1.4. Carpeta U1 con todos los archivos que contiene

En este momento, con seleccionar uno de los archivos, aparecerá en el panel derecho para editarlo.

Crear otro archivo a partir de uno existente

Si se quiere crear un archivo a partir de uno que estamos editando, se puede utilizar la opción Archivo/Guardar como… o utilizar la combinación de teclas `Ctrl + Mayus + S`. Con esto crearemos un nuevo archivo con todo el contenido del que estamos editando en ese momento.

Guardado automático

Dentro de Visual Studio Code se puede activar la opción de guardado automático. Para activar esta funcionalidad se debe ir a la opción de menú: Archivo/Autoguardado y dejarla activada. Si se selecciona nuevamente, se desactiva esta opción.

Estando activada esta opción, en el momento que se modifica un archivo fuente, quedará grabado en el disco en el lugar donde ya existía. De esta manera ya no es necesario guardarlo manualmente.

1.7. CodePen

CodePen es un entorno de desarrollo totalmente *on-line* en donde no es necesario descargarse ninguna herramienta para editar y ejecutar código.

Esta herramienta está destinada a lenguajes *front-end*, como HTML, CSS y JavaScript, muy útil para el desarrollo en entorno web. Además, esta herramienta tiene capacidad de red social con la intención de compartir el código, con lo que muchas personas también acaban por usar el CodePen como una especie de currículo profesional, mostrando los mejores trabajos de diseño y desarrollo.

Funcionamiento de CodePen

Codepen: https://codepen.io/pen

Accedemos a través de la URL https://codepen.io/pen.

Figura 1.5. Pantalla principal de CodePen

En este momento ya se puede escribir código HTML, CSS o JS en cada una de las ventanas apropiadas para ello. Una vez escrito ese código no es necesario pulsar sobre ningún botón, ya que directamente aparece el resultado en la ventana blanca inferior.

Figura 1.6. Ejecución en CodePen

Si queremos guardar y compartir el código, es necesario registrarse, y la herramienta crea un perfil donde se guarda todo el repositorio.

1.8. Ejemplos en JavaScript

A continuación, se muestran varios ejemplos sencillos en donde se ilustran algunas de las posibilidades que ofrece JavaScript. En capítulos posteriores se explicarán y ampliarán estos ejemplos.

Pedir un dato por pantalla y mostrarlo en una ventana

Código 1.4. `prompt` y `alert`

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width,
initial-scale=1.0">
    <title>Pedir un dato y sacar un mensaje</title>
</head>
<body>
    <script>
        texto= prompt("Introducir un texto", "");
        alert(texto);
    </script>
</body>
</html>
```

Ejecutando este código vemos como la instrucción **`prompt`** abre una ventana donde se solicita un texto y con la instrucción **`alert`** se muestra.

Escribir en un elemento HTML

Código 1.5. Escribir en un párrafo de HTML

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
```

```
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width,
initial-scale=1.0">
    <title>Escribir en un elemento HTML</title>
    </script>
</head>
<body>
    <p id="idTexto"></p>
    <script>
        document.getElementById("idTexto").innerHTML="Texto de
prueba"
    </script>
</body>
</html>
```

Este ejemplo ilustra cómo se puede modificar el contenido de un elemento HTML a partir de un código JavaScript.

Escribir en la consola del navegador

Código 1.6. Mostrar un texto en la consola del navegador

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width,
initial-scale=1.0">
    <title>Escribir en la consola</title>
    </script>
</head>
<body>
    <input type ="text" id="idTexto"/>
    <button onclick="console.log (document.
getElementById('idTexto').value)">Poner el texto en la
consola</button>
</body>
</html>
```

La consola del navegador es un recurso muy utilizado en JavaScript a la hora de depurar el código y encontrar errores. Podemos indicar valores dinámicos o literales para conocer cómo se comporta nuestro código.

Añadir un texto a través de un botón

Código 1.7. Botón para cambiar un texto

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width,
initial-scale=1.0">
```

```
    <title>Modificación de texto</title>
</head>
<body>
<div id="texto"></div>'
<button onclick="document.getElementById('texto').
innerHTML='Texto Modificado'">Botón que añade un texto</button>
</body>
</html>
```

Podemos asociar un código JavaScript en el momento que se pulsa un botón y provocar la modificación del HTML

Cambio de imagen

Código 1.8. Cambio del `src` de una imagen

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width,
initial-scale=1.0">
    <title>Cambio de imágenes</title>
    <script>
        function cambio(){
            nombreImagen= document.
getElementById("nombreImagen").value;

document.getElementById("imagen").src=nombreImagen;
        }
    </script>
</head>
<body>
    <img id="imagen" src="" onclick="cambio()"/>
    <input type ="text" id="nombreImagen"/>
    <button onclick="cambio()">Cambiar imagen</button>
</body>
</html>
```

Podemos cambiar los atributos de los elementos HTML, como la URL de la imagen cargada.

Cambio de estilo de un texto

Código 1.9. Cambio de estilo de un texto

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width,
initial-scale=1.0">
```

```
    <title>Cambio de estilo</title>
    <script>
        function nuevoEstilo(){
            tamano= document.getElementById("idRango").value;
            color = document.getElementById("idColor").value;
            document.getElementById("idNumero").innerHTML =
tamano;
            texto=document.getElementById("idTexto");
            texto.style.fontSize=tamano+"px";
            texto.style.color=color;
        }
    </script>
</head>
<body>

    <div id="idTexto">Texto de prueba</div>

    <input type="range" id="idRango" min="8" max = "50"
value="12" oninput="nuevoEstilo()">
    <span id="idNumero">12</span>
    <select name="color" id="idColor"
onchange="nuevoEstilo()">
        <option value="black">Negro</option>
        <option value="red">Rojo</option>
        <option value="blue">Azul</option>
        <option value="green">Verde</option>
    </select>

</body>
</html>
```

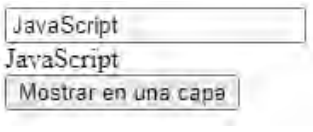
Con JavaScript no solo podemos modificar los elementos HTML, sino también los estilos descritos en la CSS, de manera que de forma dinámica cambiamos el diseño de la web.

1.9. Ejercicios de la unidad

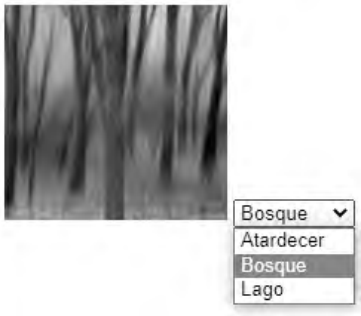
Ejercicio 1.1: Solicitar un dato por pantalla en una ventana emergente y mostrarlo como contenido de un párrafo <p>.

Ejercicio 1.2: Seleccionar de una lista plegada un color y mostrarlo por la consola.

Ejercicio 1.3: Solicitar en un <input> un texto y mostrarlo en una capa `<div>` pulsando sobre un botón.



Ejercicio 1.4: Cargar una lista plegable con nombres de imágenes que tienes guardadas en el disco y cuando se selecciona una de la lista, visualizar esa imagen.



Ejercicio 1.5: Crear una página con un texto y dos listas plegables, una indica que el texto se ponga en negrita o no, y la otra si el texto se pone en cursiva o no.