

Construcción de la interfaz de usuario del software

En este componente formativo se trabajarán los temas de interfaces gráficas de usuario, con todos sus conceptos, buenas prácticas y como entorno visual de imágenes y objetos donde se da una interacción productiva, así como también el tema de experiencia de usuario que se enfoca, sobre todo, en páginas web y aplicaciones móviles, como un concepto clave dentro del entorno digital.

Iniciar >



The color palette diagram illustrates the primary, secondary, and neutral colors used in the design, along with a color wheel labeled CB (Complementary Colors).

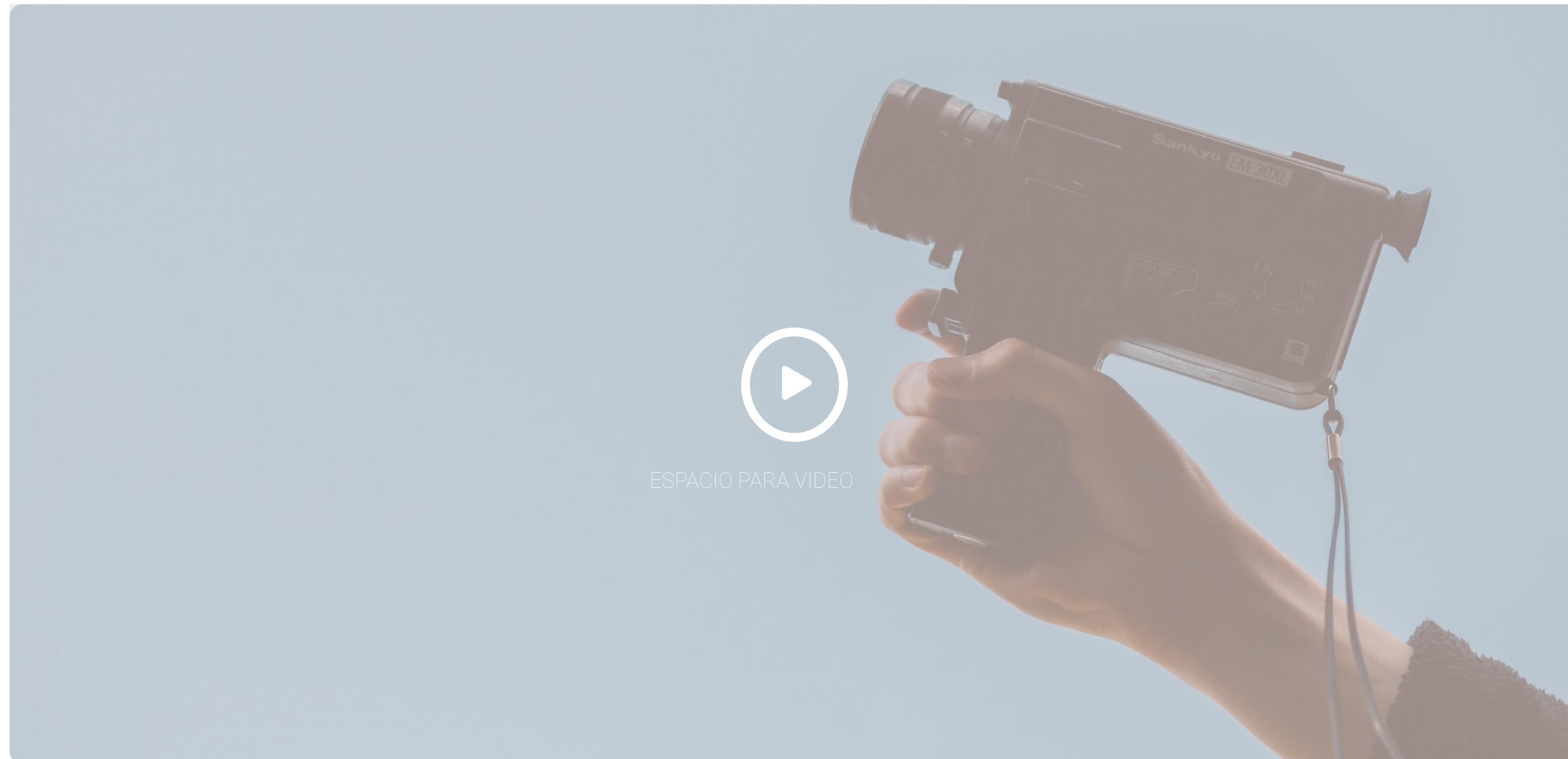
Color Group	Color Name	Hex Code	Color Swatch
Primary	PRIMARIO	#EF3C79	
	ACENTO CONTENIDO	#FECB2F	
Secondary	SECUNDARIO	#0B5999	
	ACENTO BOTONES	#21CBAE	
Neutral	NEUTRAL 1	#EFEFEF	
	NEUTRAL 2	#F9F7EC	

CB

i Introducción

Apreciado aprendiz, bienvenido a este componente formativo donde aprenderá a construir la interfaz gráfica de usuario, con el objetivo de establecer una buena experiencia de usuario al momento de usar la aplicación.

En el siguiente video conocerá, de forma general, la temática que se estudiará a lo largo del componente formativo.



1 Lenguajes de programación

Comencemos diciendo que los lenguajes de programación son un conjunto de instrucciones a través de las cuales se pueden programar funcionalidades que serán ejecutadas a través de una computadora. Conozcamos los diferentes lenguajes.

1.1 HTML

Antes de entrar a revisar el lenguaje HTML5, se deben tener presentes un conocimiento base y unos conceptos fundamentales.



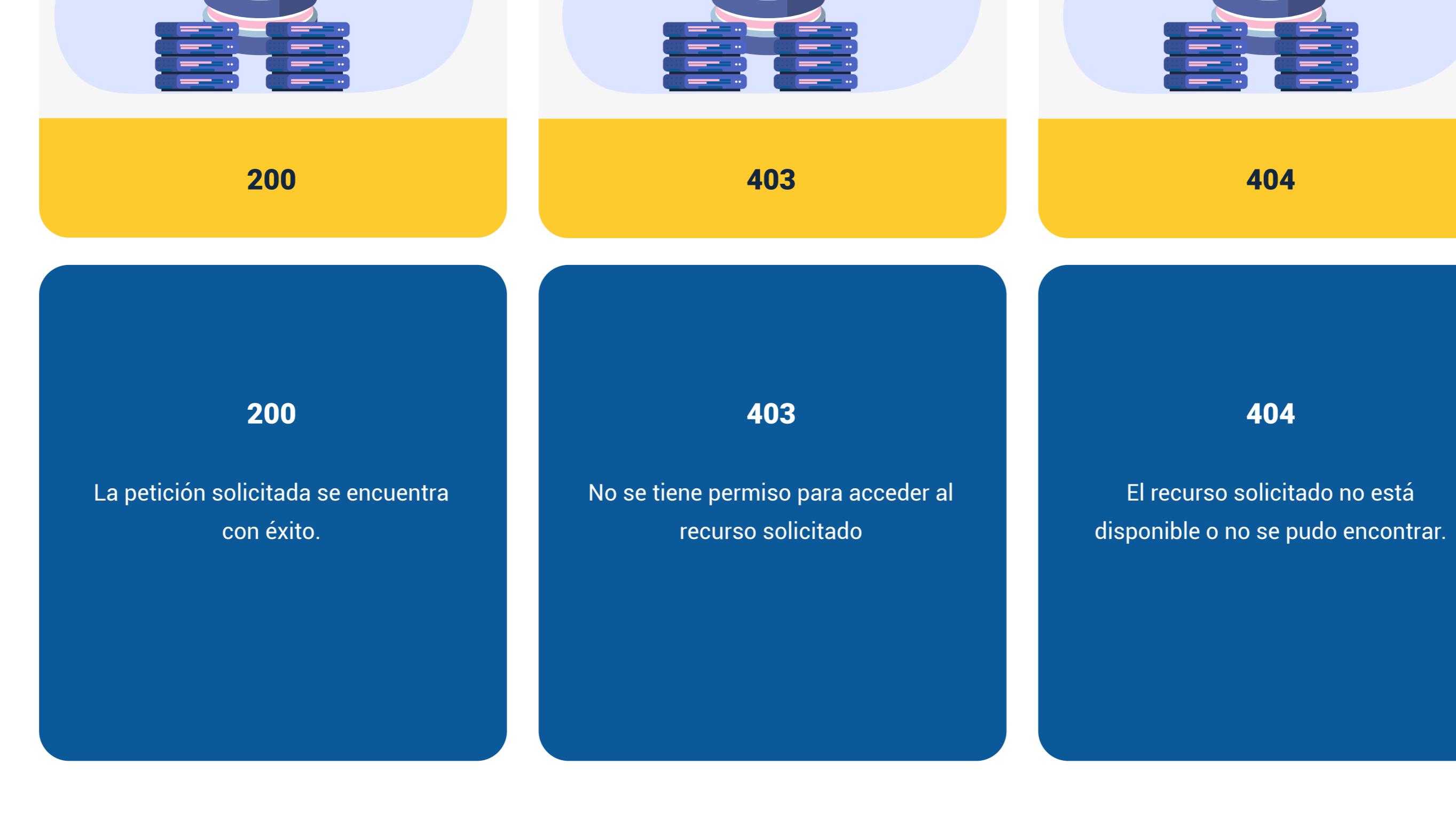
Toda la tecnología web nace con la aparición del internet; en realidad, internet es la unión de dos palabras **interconnected** y **network**, lo que quiere decir que es una red global de computadores interconectados entre sí, que permiten compartir información como texto, sonido, imágenes, etc.

Tim Berners Lee, inventor de la *World Wide Web* (WWW), fundó el consorcio W3C para estandarizar, supervisar y estar al tanto de las tecnologías basadas en el internet; debido a este procedimiento, se originan los siguientes conceptos:



HTTP
Hypertext Transfer Protocol (Protocolo de Transferencia de Hipertexto)

Es el protocolo de transferencia de hipertexto y es el apoyo que permite la operación de la comunicación entre los dispositivos conectados en la red; por ejemplo, entre computadoras de usuarios y los servidores.

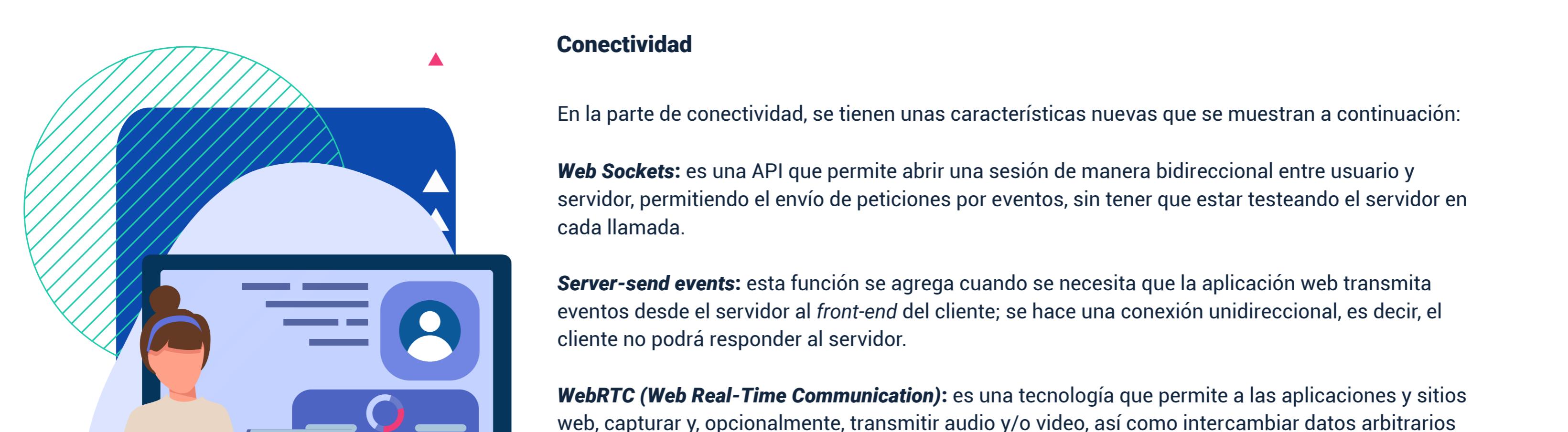


A parte de interpretar y mostrar al usuario la página, algunas de las responsabilidades del navegador son:

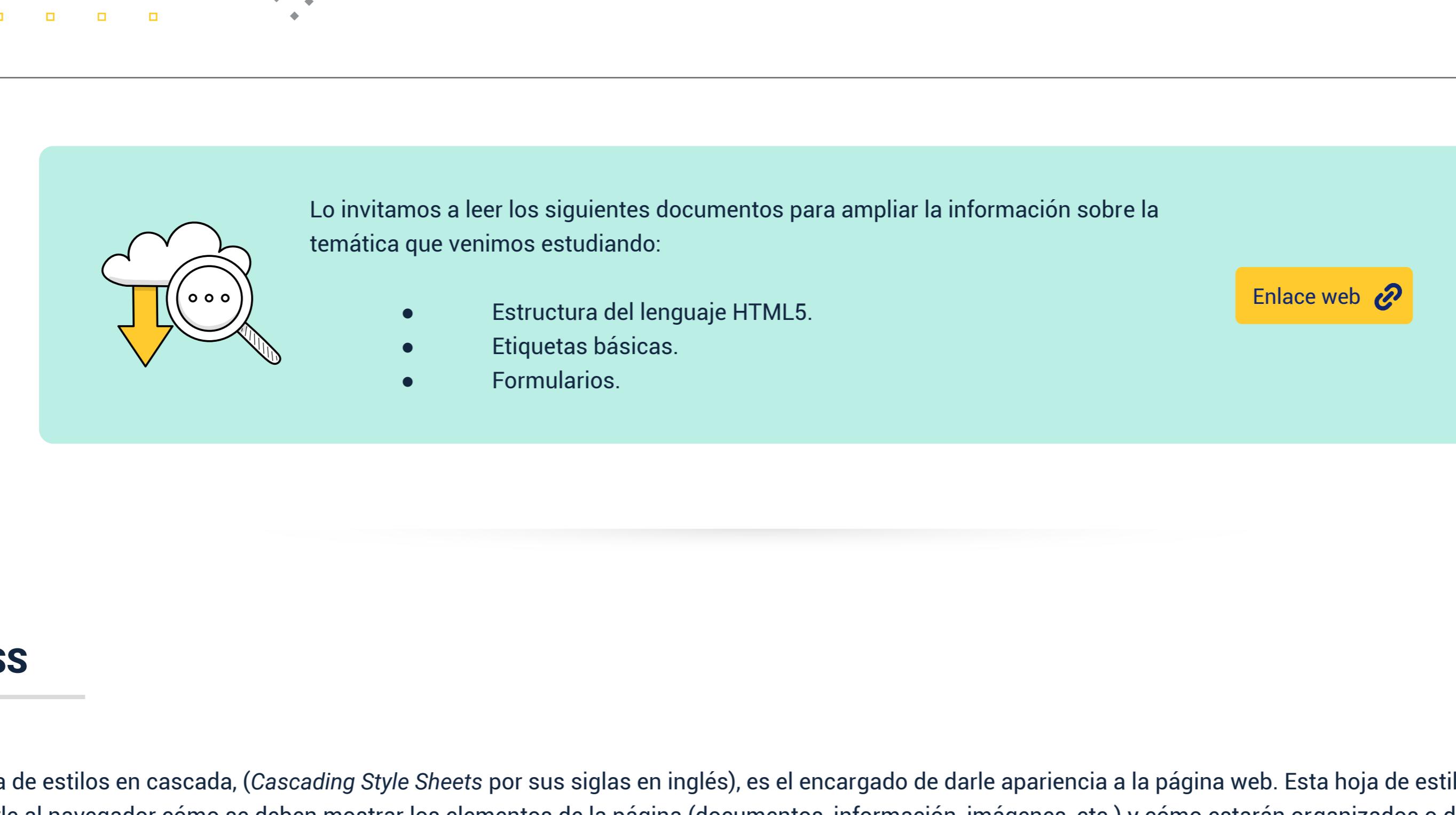


Peticiones HTTP

Cuando los navegadores acceden a una página web, lo hacen a través de **http**, como se había dicho anteriormente; en esa comunicación con el servidor, se descompone la URL en diferentes partes: una parte es el host que es la dirección de la máquina y, la otra, el path o la ruta del recurso al que se quiere acceder.



Es una petición donde se envían los datos ocultos, a través de un objeto que será descifrado por el servidor y, de esta manera, se tiene seguridad de la información.

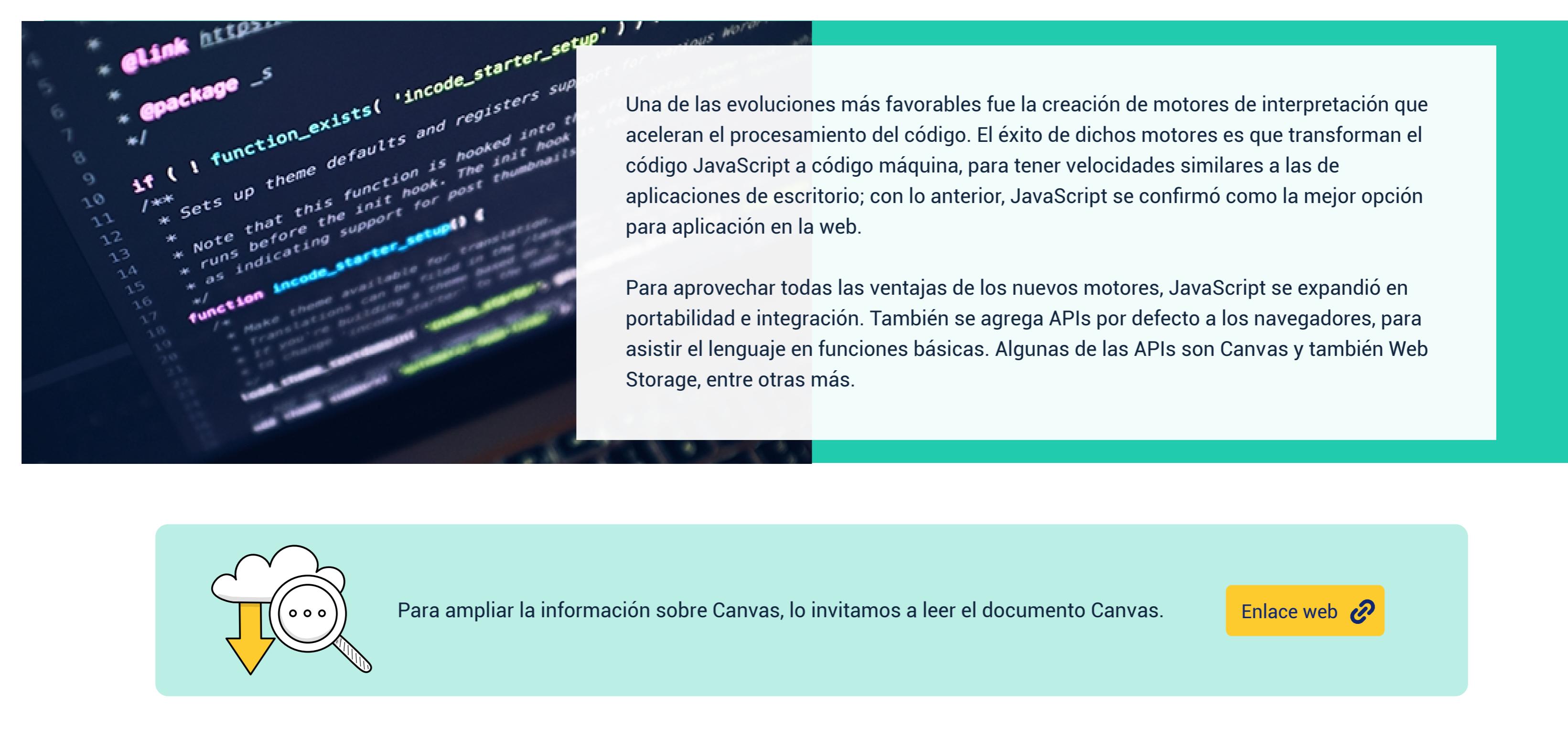
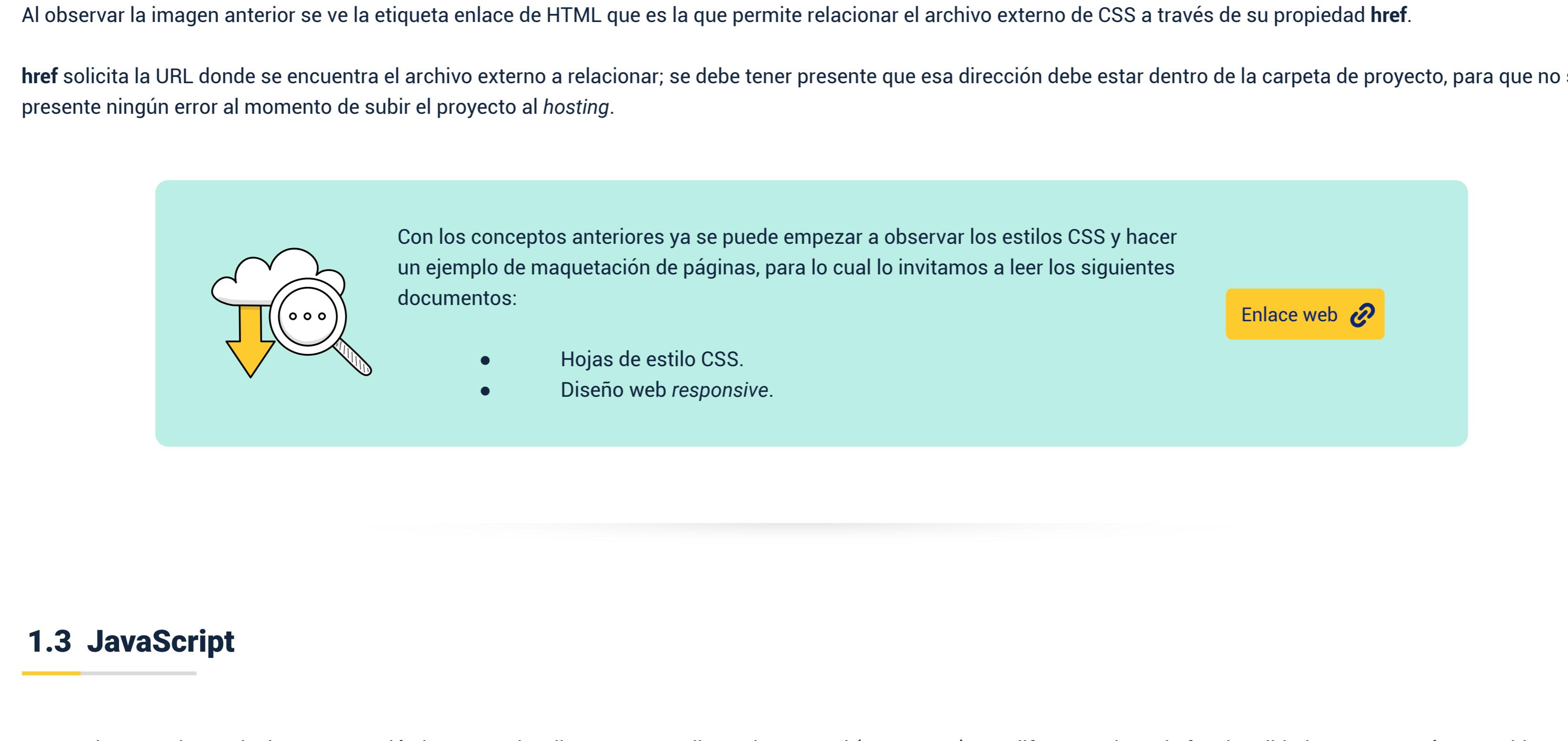


HTML5 mejora la versión de HTML, de acuerdo con el avance en las API y en el conjunto de herramientas que conforman la tecnología web; pero, aparte de esto, presenta nuevas características en el lenguaje HTML, las cuales son:



Al observar la imagen anterior se ve la etiqueta **enlace de HTML**, que es la que permite relacionar el archivo externo de CSS a través de su propiedad **href**.

El href solicita la URL donde se encuentra el archivo externo a relacionar; se debe tener presente que esa dirección debe estar dentro de la carpeta de proyecto, para que no se presente ningún error al momento de subir el proyecto al **hosting**.



1.2 CSS

CSS – Hoja de estilos en cascada, (*Cascading Style Sheets* por sus siglas en inglés), es el encargado de darle apariencia a la página web. Esta hoja de estilos sirve para especificarle al navegador cómo se deben mostrar los elementos de la página (documentos, información, imágenes, etc.) y como estarán organizados y diseñados.

Cuando se habla de CSS se habla de reglas que se aplican a los elementos de la página o a un grupo de ellos.

Una regla es un código que le indica a un elemento, o a un grupo de ellos, cómo debe ser su diseño; es decir, qué color tendrá, si tiene fondo o color de fondo, tipografía, organización, etc.

Las reglas de CSS se pueden anidar, es decir, se pueden agregar reglas a etiquetas que estén dentro de otras, de acuerdo con las especificaciones dadas en el selector.

Para la aplicación de dichas reglas, anteriormente se hacia en la misma línea de donde se declaraba la etiqueta HTML, por ejemplo:

1.3 JavaScript

JavaScript es un lenguaje de programación interpretado y ligero, se compila en tiempo real (*just-in-time*) con diferentes tipos de funcionalidades. Su uso más conocido es la programación de script aplicados en entornos web, pero es usado en muchos más entornos como AngularJS, NodeJS, Adobe Acrobat, entre otros.

2 Tecnologías de programación, frameworks y librerías

Cuando se habla de tecnologías de programación, se refiere a los lenguajes de programación que se pueden utilizar para el desarrollo de software, así como las diferentes herramientas y *frameworks* que se pueden aplicar.

Entre estas tecnologías podemos encontrar las siguientes:



HTML

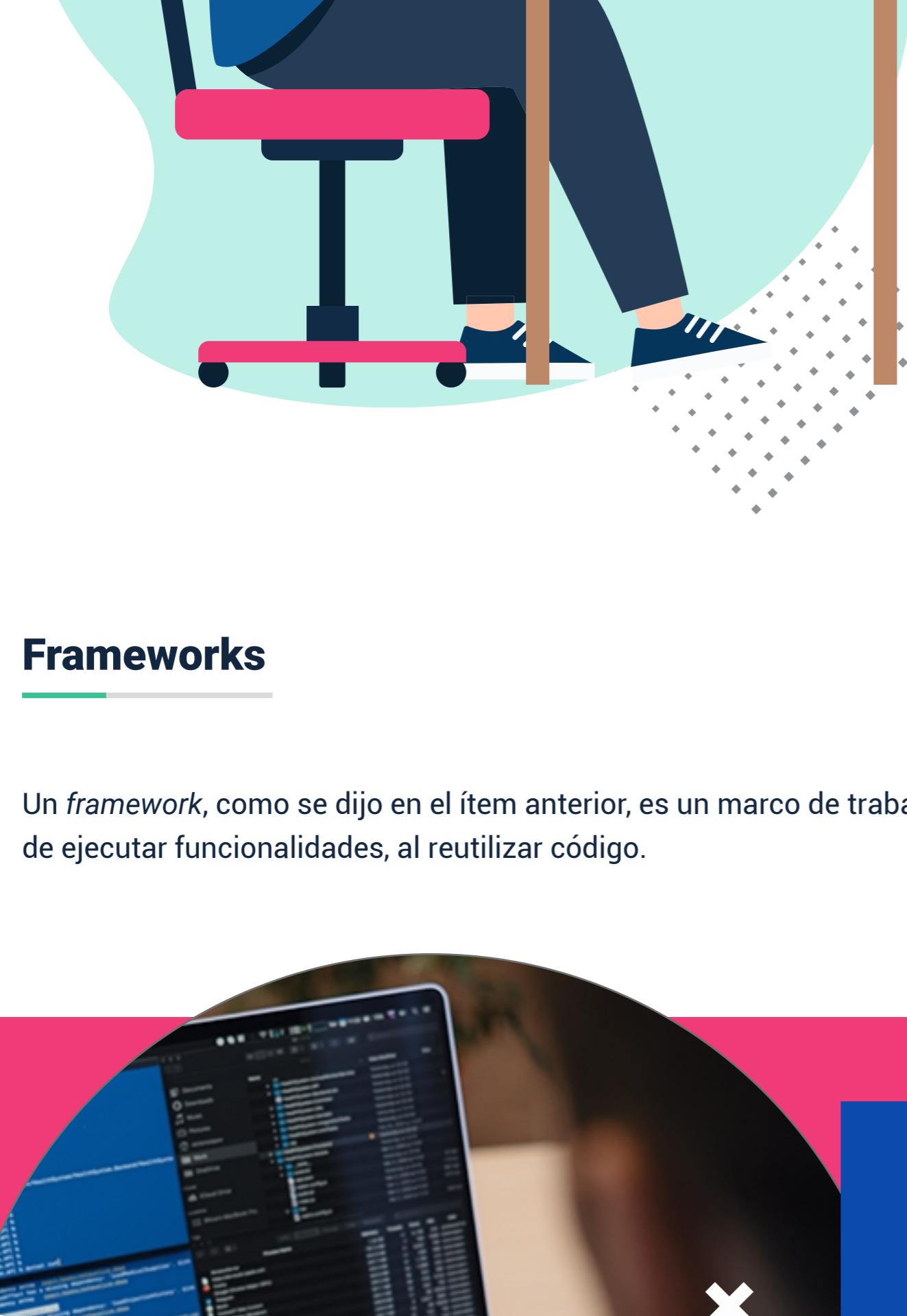
Es el lenguaje de marca o etiquetas utilizado para la creación de sitios web. Con HTML se realiza la estructura del sitio, su maquetación a través de etiquetas que permiten la organización de contenido. Estas etiquetas son interpretadas por los navegadores para mostrar la estructura, tal cual como fue organizada.

En conclusión, HTML permite a los desarrolladores FrontEnd describir, estructura o maquetar el contenido del sitio web, como es su información a través de párrafos, imágenes, listas, tablas, barras de menú, etc.

Tecnologías del lado del servidor

Las tecnologías del lado del servidor o también llamadas *backend* son las que tienen como objetivo implementar los comportamientos que tendrá la web en el servidor; es decir, sus peticiones, cómo se tratan, cómo se consume la capa de datos, etc.

Entre ellas, algunas de las tecnologías para desarrollo web más populares son:



PHP

PHP es un lenguaje de programación del lado del servidor, que permite la comunicación de la web con servidores de datos o persistencia. Esta funcionalidad permite crear la capa de datos en los desarrollos web y comunicarse con diferentes motores de bases de datos como MySQL, PostgreSQL, etc.; además, con este lenguaje de programación se gestiona todo el comportamiento del servidor, el tratamiento de archivos, imágenes, la recopilación de información desde los formularios del sitio, entre otras.

Python

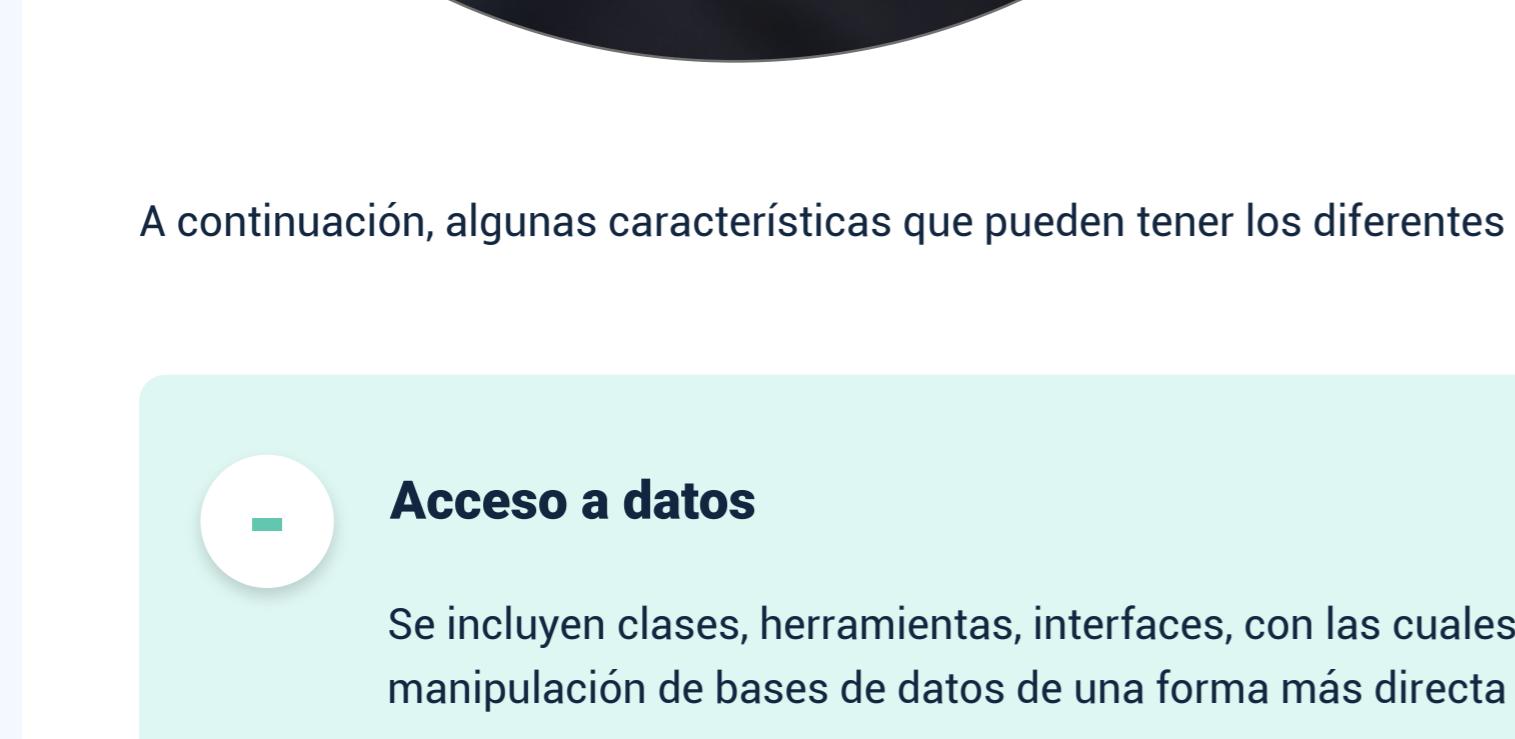
Frameworks y librerías

Angular

React

Frameworks

Un *framework*, como se dijo en el ítem anterior, es un marco de trabajo para programar en diferentes lenguajes, usando librerías, buenas prácticas y una forma más sencilla de ejecutar funcionalidades, al reutilizar código.



El concepto de *framework* no es de uso exclusivo de la tecnología web, del desarrollo de sus aplicaciones; también se puede encontrar en aplicaciones de realidad aumentada, de inteligencia artificial y de escritorio. Es decir, es una estructura compuesta por componentes reutilizables e intercambiables, durante el proceso de desarrollo de una aplicación.

Existen varios tipos de *frameworks* web: orientados a la interfaz de usuario, como Java Server Faces; orientados a aplicaciones de publicación de documentos, como Cocoon; orientados a la parte de control de eventos, como Struts, y algunos que incluyen varios elementos, como Tapestry.

La mayoría de *frameworks* web se encargan de ofrecer una capa de controladores, de acuerdo con el patrón MVC o con el modelo 2 de Servlets y JSP, ofreciendo mecanismos para facilitar la integración con otras herramientas para la implementación de las capas de negocio y presentación.

A continuación, algunas características que pueden tener los diferentes *frameworks*:

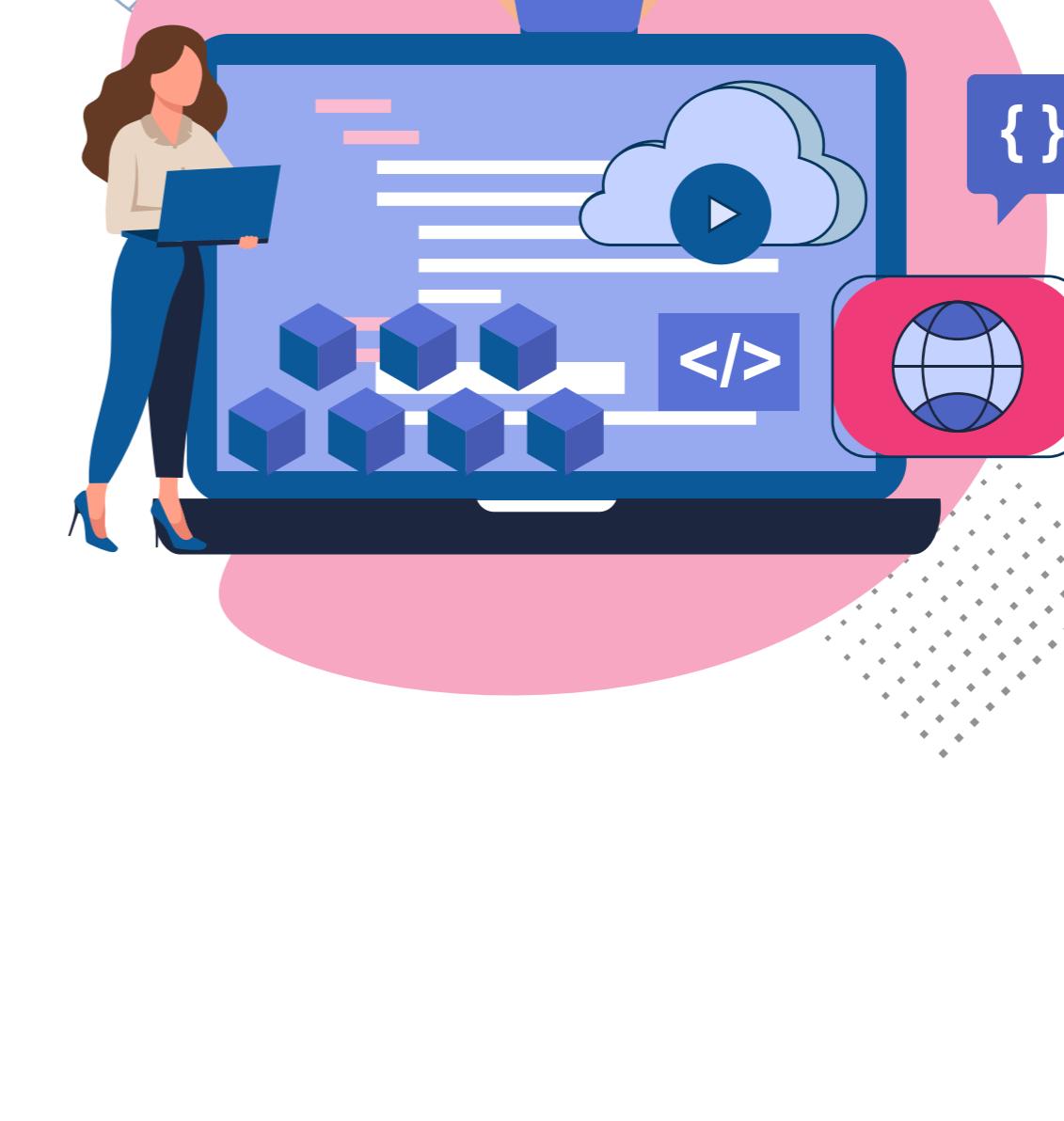
Acceso a datos

Se incluyen clases, herramientas, interfaces, con las cuales se puede integrar la manipulación de bases de datos de una forma más directa y sencilla.

Controladores

Autenticador

Abstracción



Librerías

Para conocer qué es una librería, sus tipos y características, lo invitamos a ver el siguiente video.

ESPACIO PARA VIDEO



3 Codificación en lenguajes de programación

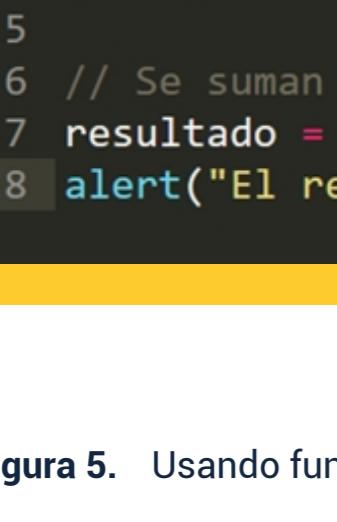
La nomenclatura de JavaScript indica la forma como se escriben sus variables, es decir, su sintaxis, gramática, cómo se puede aplicar. Este tema, en concreto, será detallado más adelante, cuando se estén estudiando variables, constantes y funciones.

En JavaScript existen los gestores de paquetes, los cuales se encargan de gestionar las dependencias entre código y liberar así de esta ardua tarea, al programador. Estas herramientas son gestores como Gradle, Maven Ant y su función es encargarse de la instalación, actualización, configuración y eliminación de los diferentes paquetes que necesite el software para su funcionamiento.

Las clases en JavaScript permiten la programación orientada a objetos, con sus atributos, sus propiedades, sus constructores y permitiendo de una forma sencilla, la creación de objetos y la implementación de la herencia.

Figura 3. Declaración de clase

```
class Rectangulo {
    constructor(alto, ancho) {
        this.alto = alto;
        this.ancho = ancho;
    }
}
```



Las funciones son útiles en el proceso de desarrollo de aplicaciones web, donde una y otra vez se repiten las mismas instrucciones en varias partes del desarrollo. Por ejemplo, si se está haciendo una tienda virtual, siempre se debe estar calculando el precio de los productos, añadiendo los impuestos necesarios y los gastos de envío; para este proceso es posible utilizar un script.

Un ejemplo sencillo para determinar cómo se escribe una función, es la suma de dos números; en JavaScript, si yo quiero sumar dos números, lo hago como se observa en la imagen.

Figura 4. Ejemplo suma de dos números

```
1 var resultado;
2
3 var numero1 = 3;
4 var numero2 = 5;
5
6 // Se suman los números y se muestra el resultado
7 resultado = numero1 + numero2;
8 alert("El resultado es " + resultado);
```

La imagen anterior muestra cómo se agrega código que declara tres variables, la de número1 y número2, se les asigna un valor y después se hace la operación para guardar el resultado en la variable resultante.

Si se supone que esta acción se debe hacer varias veces en diferentes partes del código, entonces, se estaría siempre agregando 8 líneas de código y no sería muy óptimo dejar tal cantidad de código, cuando se puede optimizar, usando una función. Se deberá crear una función que haga la respectiva suma y muestre el resultado.

Figura 5. Usando función suma

```
1 function suma() {
2     resultado = numero1 + numero2;
3     alert("El resultado es " + resultado);
4 }
5
6 var resultado;
7
8 var numero1 = 3;
9 var numero2 = 5;
10
11 suma();
12
13 numero1 = 10;
14 numero2 = 7;
15
16 suma();
17
18 numero1 = 5;
19 numero2 = 8;
20
21 suma();
```

La imagen anterior muestra el funcionamiento de una función, pero se debe explicar su sintaxis.

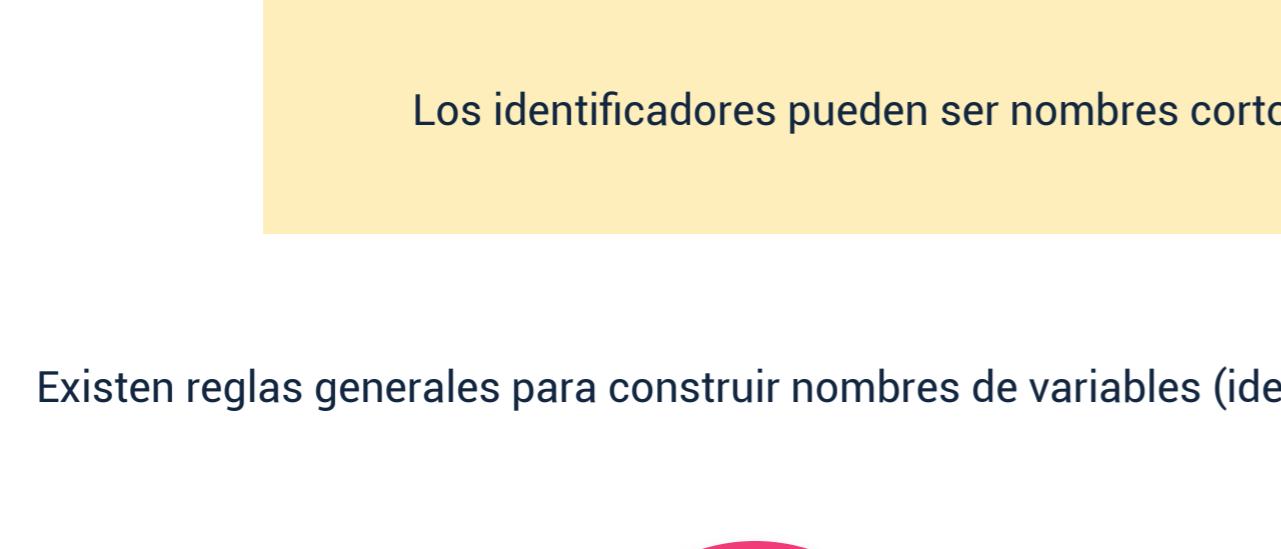
```
function nombre_funcion (argumentos) { }
```

La sintaxis de una función es la que se mostró anteriormente, es decir, se declara con la palabra clave **function**, se le asigna un nombre y, entre los paréntesis, van los argumentos o los parámetros de entrada que recibe; recuerde que una función puede necesitar parámetros de entrada o puede funcionar sin ellos, de acuerdo con la necesidad y el contexto.

Después de explicar la sintaxis, todas las funciones se crean de la misma forma y, por eso, en el ejemplo de la imagen, se creó la función suma, la cual suma los dos números, los guarda en la variable resultado y genera un mensaje emergente con una ventana de diálogo para mostrar su resultado a través de la palabra clave **alert**.

Variables

Las variables, en programación, son espacios en memoria que se identifican a través de un nombre dado y permiten guardar un valor de un tipo de dato dado.



En JavaScript se manejan de igual forma; lo único es que es un lenguaje no tipado y, por eso, no se le debe dar el tipo de dato a la variable, solo utilizar la palabra clave **var** para definirla. Por ejemplo:

```
var nombre;
```

La línea anterior crea una variable en JavaScript llamada nombre, la cual podrá almacenar cualquier tipo de dato, pero como las buenas prácticas indican que se deben dar nombres coherentes a las variables, lo ideal es que, si se llama así, guarde un valor de un nombre de una persona.

La palabra clave **var**, antes de 2015, era la única forma de declarar una variable para JavaScript; pero la nueva versión, después de 2015, permite al lenguaje definir constantes, es decir, variables cuyo valor nunca va a cambiar, dentro de la ejecución de todo el programa y para esto utiliza la palabra clave **const**.

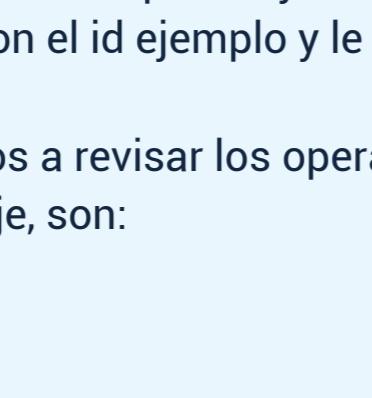
La otra forma que salió es la palabra clave **let** que define una variable con alcance restringido.

Con la forma de definir las variables, se debe expresar que se usan, no solo para mantener valores, sino para realizar expresiones algorítmicas con diferentes operadores.

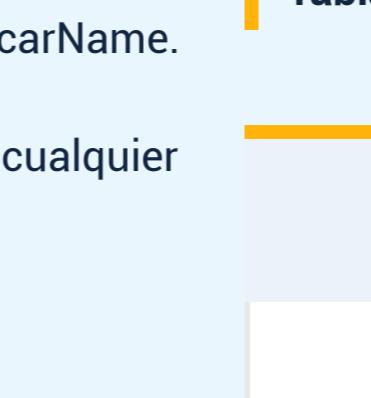
Las variables declaradas en cada lenguaje de programación, al igual que en JavaScript, deben identificarse con nombres únicos denominados identificadores.

Los identificadores pueden ser nombres cortos (como **x** o **y**) o nombres más descriptivos (**edad**, **suma**, **volumen total**).

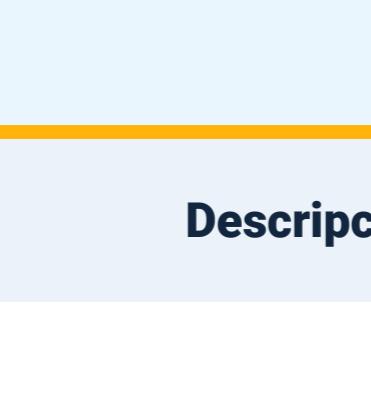
Existen reglas generales para construir nombres de variables (identificadores únicos) que son:



Los nombres pueden contener letras, dígitos, guiones bajos y signos de dólar.



Los nombres deben comenzar con una letra.



Los nombres también pueden comenzar con \$ y _ (pero no son usados comúnmente y acá tampoco se usarán).

Con lo anterior un ejemplo de declaración de variable es:

```
var x;
```

y si se le asigna un valor a través del operador de asignación que es el igual, sería de la siguiente manera:

```
var x = 5;
```

Se empezará observando que las variables pueden tener tipos de datos numéricos como **x** que se le asigna el valor de 5 o valores de texto, que se asignan entre comillas.

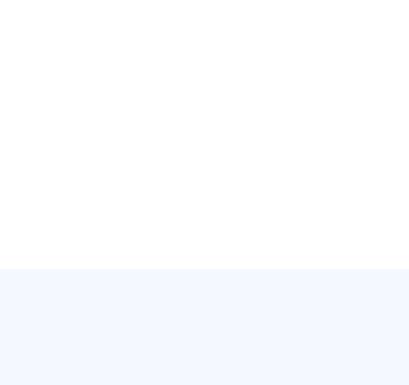
```
var nombre = "Jose Aguilar";
```

Ahora, también se puede interactuar con etiquetas HTML; por ejemplo, asignarle el valor de una variable a una etiqueta de párrafo HTML.

```
<p id="ejemplo"></p>
<script>
var carName = "Audi";
document.getElementById("ejemplo").innerHTML = carName;
</script>
```

Tabla 1. Operadores JavaScript

Operador	Descripción
+	Suma
-	Resta
*	Multiplicación
/	División
**	Exponente
++	Incremento
--	Decremento



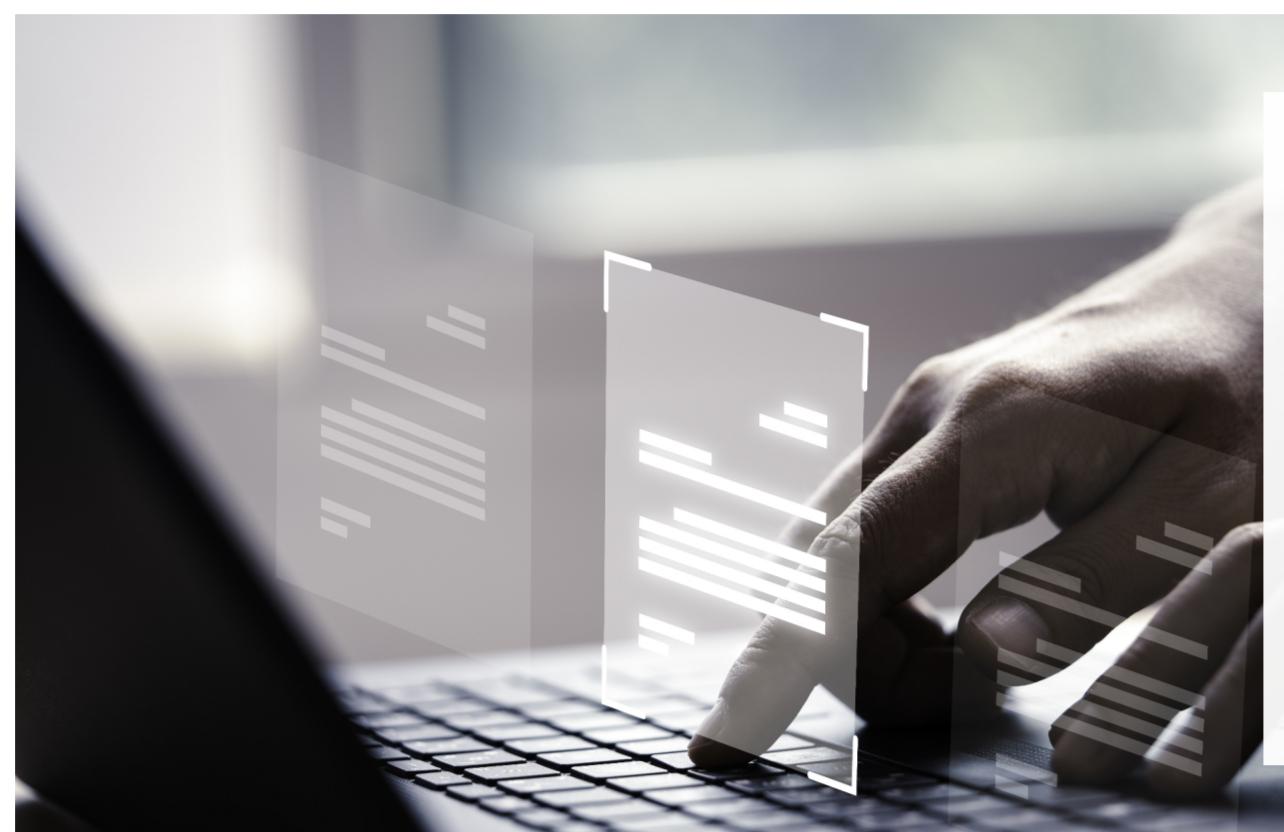
Con todos los operadores anteriores, se pueden combinar las variables y hacer expresiones aritméticas o usarse en los condicionales como pruebas lógicas para determinar por cuál camino seguir, según se cumpla o no, la condición a través del condicional **if** y los operadores de comparación.

Ahora, las constantes son nombres que se asignan a espacio en memoria, para guardar un único valor, el cual no podrá cambiar en ningún momento de la ejecución del software.

Figura 6. Sintaxis de constantes

```
const varname1 = value1 [, varname2 = value2 [, varname3 = value3 [, ... , , varnameN = valueN]]];
```

4 Estilo de codificación



Los comentarios son parte del desarrollo que están en el código fuente, pero que el sistema no los ejecuta, los pasa por alto; sirven para documentar el código e identificar ciertas partes importantes en su mantenimiento a futuro.

Los comentarios pueden ser de una línea o pueden ser comentarios de varias líneas, como se presenta a continuación.

Figura 7. Comentario de una sola línea

```
<script>
    // Este un comentario de una única línea
    alert("Escribiendo comentarios en javascript!");
    // alert("Esto no se ejecuta");
</script>
```

Figura 8. Comentario de varias líneas

```
<script>
    alert("Escribiendo comentarios multi-línea en javascript");
    /*
    alert("Esto no se ejecuta");
    alert("Esto no se ejecuta");
    alert("Esto no se ejecuta");
    alert("Esto no se ejecuta");
    alert("Y esto tampoco");
    Y este texto puede decir lo que yo quiera
    para acordarme de algo
    */
</script>
```



var

Declara una variable, opcionalmente, la inicia a un valor.



let

Declara una variable local con ámbito de bloque, opcionalmente, la inicia a un valor.



const

Declara un nombre de constante de solo lectura y ámbito de bloque.

Para dar inicio a las estructuras de control, se va a iniciar con los condicionales, estructura que permite controlar el flujo del programa o de la función que se esté realizando.

```
If (condición) {
} else {
```

La sintaxis es la anterior, donde se puede ver que se aplica igual que en los lenguajes como **php** o Java. A continuación, un ejemplo de su uso.

Figura 9. Estructura de control if

```
1 var mensaje = true;
2
3 if(mensaje){
4     console.log("Hola Mundo");
5 }else{
6     console.log("Error")
7 }
```



5 Experiencia de usuario

La experiencia de usuario es un tema muy importante en el desarrollo de software, porque permite definir cómo se sentirá el usuario mientras navega en el sistema o aplicación web. Se trata de hacerle vivir una experiencia agradable para que quiera seguir usando la aplicación, que la pueda usar de una forma sencilla y, sobre todo, que cumpla con las funciones pactadas o por las cuales se creó.



Según la norma ISO 25000, se conoce por usabilidad, la capacidad del producto software para ser entendido, aprendido, usado y resultar atractivo para el usuario, cuando se usa bajo determinadas condiciones. Esta característica se subdivide, a su vez, en las siguientes subcaracterísticas:



- Capacidad para reconocer su adecuación

Capacidad del producto que permite al usuario entender si el software es adecuado para sus necesidades.

+ Capacidad de aprendizaje

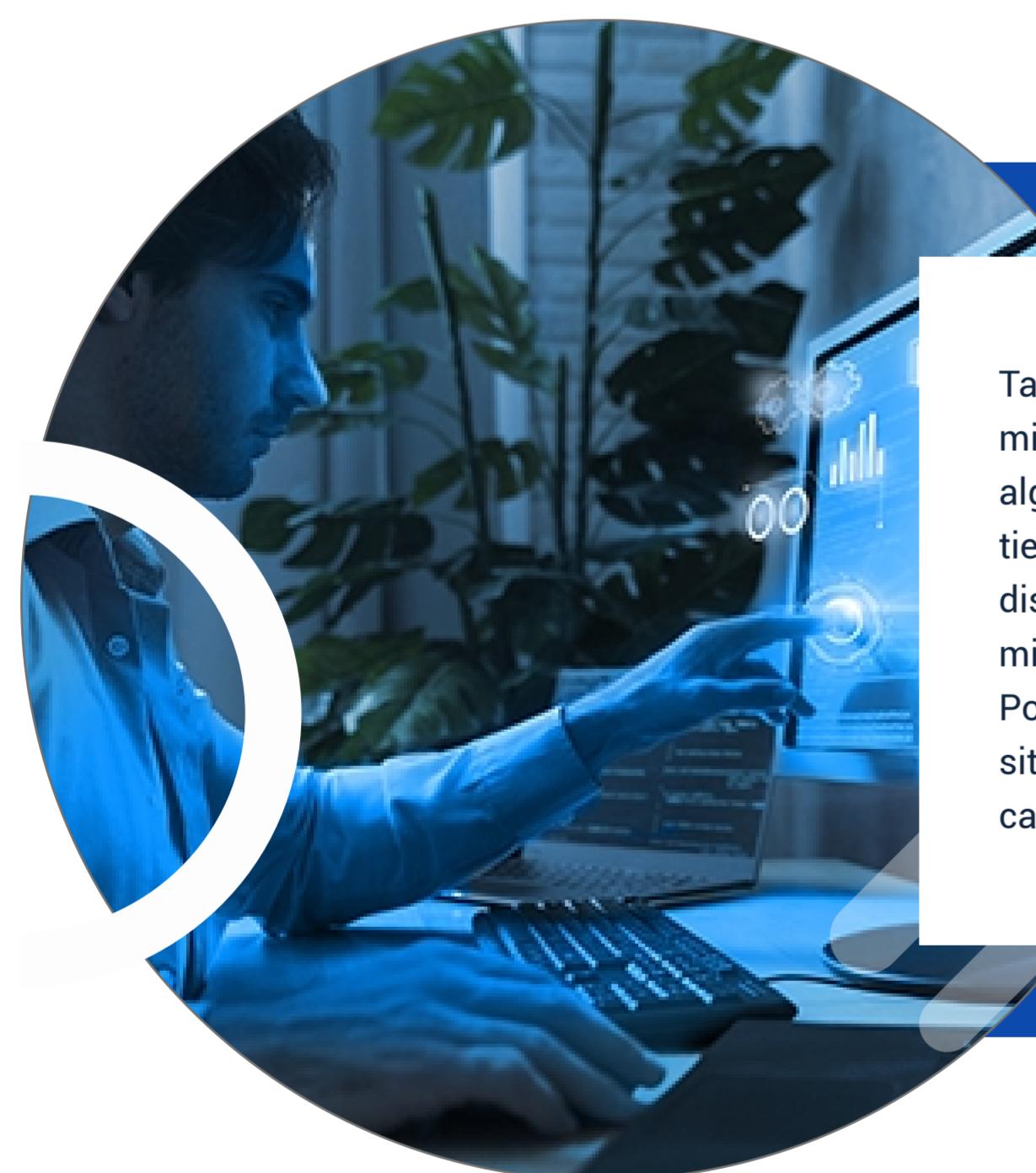
+ Capacidad para ser usado

+ Protección contra errores de usuario

+ Estética de la interfaz de usuario

+ Accesibilidad

La accesibilidad hace que los sitios web puedan ser utilizados por la mayor cantidad de personas posible. Tradicionalmente, se ideó para las personas con discapacidad, pero la creación de sitios accesibles también beneficia a otros grupos, como los que utilizan los dispositivos móviles o los que tienen conexiones de red lentas.



También se podría pensar en la accesibilidad como una forma de tratar a todos por igual y darles las mismas oportunidades, sin importar su capacidad o circunstancias. Tal y como es injusto excluir a alguien de un edificio, porque ande en silla de ruedas (generalmente, los edificios públicos modernos tienen rampas o ascensores), tampoco es correcto excluir a alguien de un sitio web porque tenga una discapacidad visual. Todos somos diferentes, pero todos somos humanos y, por lo tanto, tenemos los mismos derechos.

Por eso, debemos hacer las cosas accesibles. En algunos países, es obligado por la ley proporcionar sitios web accesibles, lo que puede abrir algunos mercados importantes que, de otra manera, no serían capaces de utilizar los servicios o comprar los productos.

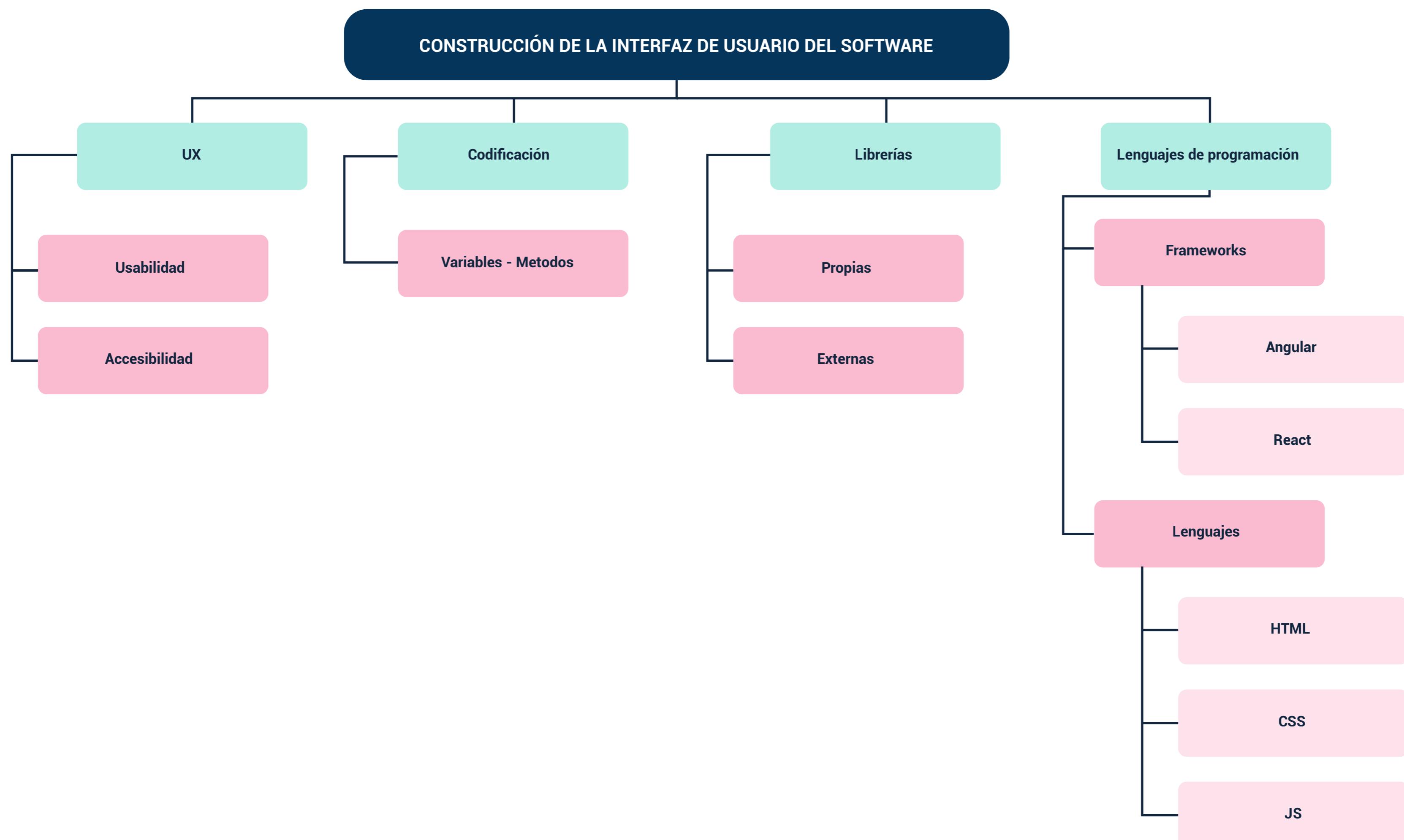
- El HTML semántico, aparte de mejorar la accesibilidad, también mejora la optimización en motores de búsqueda (SEO), y la gente encuentra un sitio web con mayor facilidad.
- Preocuparse por la accesibilidad demuestra buenos criterios éticos y morales, lo que mejora la imagen pública.
- Mejorar la accesibilidad, también hace que el sitio sea más fácil de usar por otros grupos, como los usuarios de dispositivos móviles o los que tienen internet de baja velocidad. De hecho, todo el mundo puede beneficiarse de estas mejoras. Además, hemos mencionado también que en algunos países está estipulado por ley.

Desarrollo de aplicaciones web Full stack

Síntesis: construcción de la interfaz de usuario del software.



El siguiente mapa integra los criterios y especificidades de los conocimientos expuestos en el presente componente formativo.





Portada actividad

800 x 800



Portada actividad

800 x 800

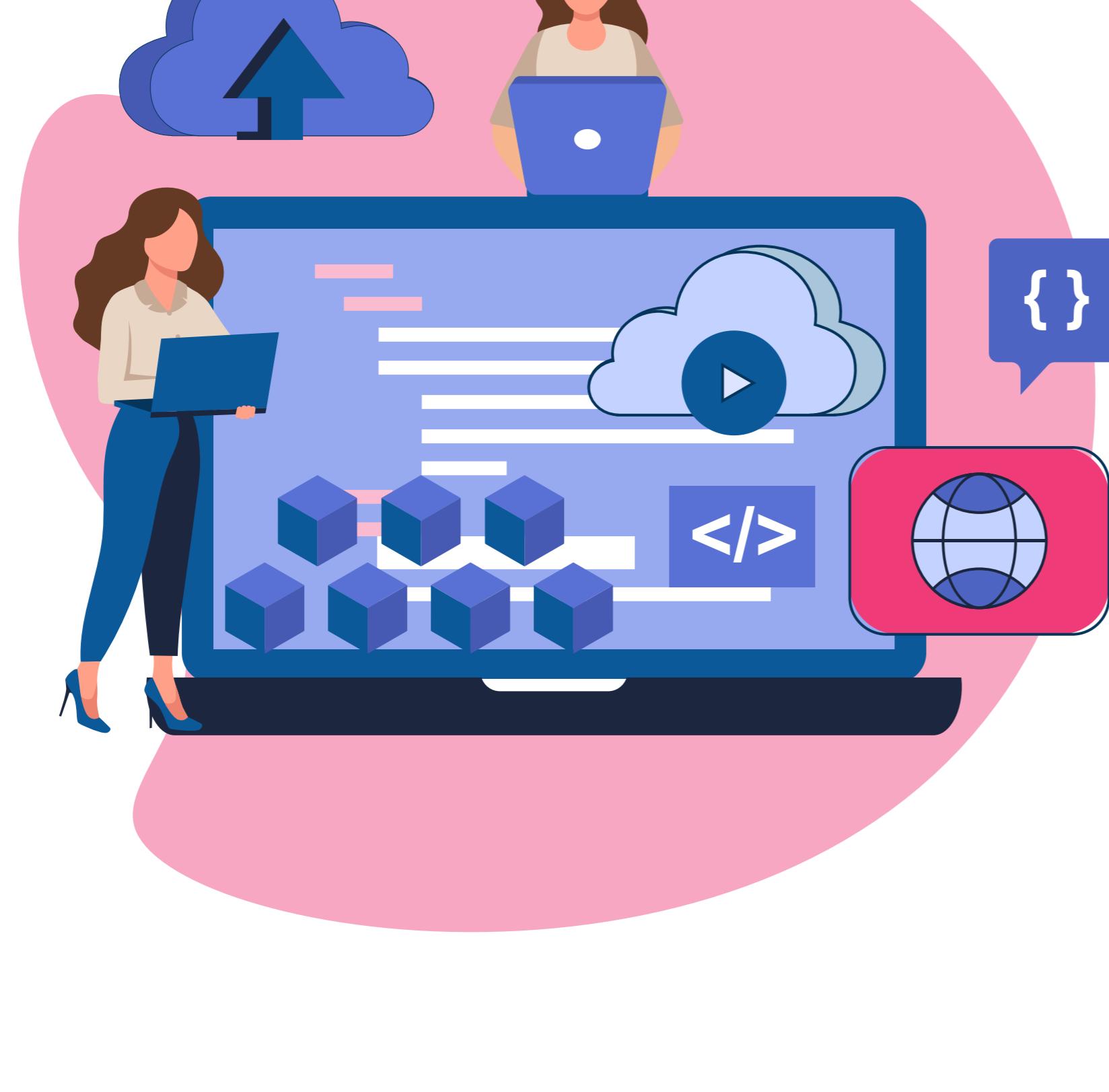


Imagen acompañamiento Actividad

600 x 600



Imagen Resultado

900*600

i

Actividad didáctica

Selección múltiple



Apreciado aprendiz, a continuación encontrará una serie de preguntas que deberá resolver, con el objetivo de evaluar la comprensión de los conocimientos expuestos en este componente formativo.

Selección múltiple

Realizar 