**Aplicación de Tecnologías Web**

Tema n.° 2

Introducción a CSS

**Índice**

**Pág.**

[1.1. Introducción a CSS 3](#_Toc161173817)

[1.2. Hojas de estilo 3](#_Toc161173818)

[1.2.1. Selectores simples 5](#_Toc161173819)

[1.2.2. Selectores combinados 7](#_Toc161173820)

[1.2.3. Selectores de pseudoclase 8](#_Toc161173821)

[1.2.4. Selectores de atributos 9](#_Toc161173822)

[1.3. CSS y HTML 9](#_Toc161173823)

[1.3.1. BEM 14](#_Toc161173824)

[1.4. Grids en CSS 16](#_Toc161173825)

[1.4.1. Propiedades básicas 16](#_Toc161173826)

[1.5. Tipografías 18](#_Toc161173827)

[1.5.1. Propiedades básicas 18](#_Toc161173828)

[1.6. Imágenes 19](#_Toc161173829)

[1.6.1. Imágenes Responsivas 20](#_Toc161173830)

[1.6.2. Filtros y Efectos 21](#_Toc161173831)

[1.6.3. Posicionamiento y Overlay 21](#_Toc161173832)

[1.7. Botones 22](#_Toc161173833)

[1.7.1. Propiedades Básicas 22](#_Toc161173834)

[1.7.2. Efectos de Hover y Transiciones 23](#_Toc161173835)

[1.7.3. Botones Responsivos 23](#_Toc161173836)

[1.7.4. Pseudo-Elementos y Pseudo-Clases 23](#_Toc161173837)

[1.8. Contenedores e interacciones 24](#_Toc161173838)

[1.8.1. Interacciones en CSS 25](#_Toc161173839)

[Recursos complementarios 26](#_Toc161173840)

[Referencias 26](#_Toc161173841)

[Autoevaluación 27](#_Toc161173842)

# Introducción a CSS

CSS (Cascading Style Sheets) es el lenguaje de estilos (o de diseño gráfico) que utilizan los desarrolladores para modificar la presentación de los elementos HTML en pantalla. Al igual que ocurre con el HTML, este lenguaje está regulado por el W3C y su última versión oficial es CSS3.

Hasta ahora no hemos incluido ningún tipo de estilo en los ejemplos de HTML que se han mostrado, a pesar de ser posible. Podemos indicar de forma inline el estilo de un elemento HTML, pero esto no está recomendado por las siguientes razones, entre otras:

* **Separación de funcionalidades**. El HTML es un lenguaje que describe la estructura de una página web, no su visualización. CSS está específicamente pensado para separar la parte estructural de la parte visual.
* **Mantenibilidad.** Los estilos inline en el HTML aplican únicamente al elemento en el que se definen. Si utilizamos los mismos estilos inline en varios elementos y luego queremos cambiarlos, tendremos que hacerlo uno a uno. Sin embargo, si usamos CSS y el atributo class , podemos hacerlo de una forma mucho más limpia.
* **Múltiples temas.** Podemos tener diferentes archivos CSS que representen los diferentes temas o modos de la aplicación. Un claro ejemplo son los típicos modos oscuro y luminoso de la mayoría de las aplicaciones modernas.

# Hojas de estilo

Las hojas de estilo CSS están compuestas por varias declaraciones y tienen una sintaxis muy sencilla:

**Figura 1**

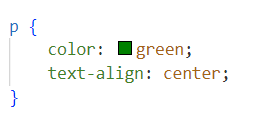
*Sintaxis de una declaración en una hoja de estilos CSS*



Un ejemplo de lo anterior sería:

**Figura 2**

*Declaración de un selector de CSS.*



Esta declaración en la imagen quiere decir que todos los elementos HTML <p> tendrán color verde y estarán alineados al centro. Podemos observar que:

* **El selector es p.** 5Todas las sentencias incluidas dentro del bloque indicado entre corchetes aplicarán a todos los elementos HTML <p> de nuestro documento. Este selector es de tipo simple.
* **Dos propiedades de CSS están siendo modificadas.** En primer lugar, color, que sirve para indicar el color del elemento HTML, y, por otro lado, text-align, que sirve para indicar qué tipo de alineamiento queremos para el texto. Los valores para estas propiedades son green y center, respectivamente.

Un dominio amplio de CSS pasa por conocer necesariamente todos los tipos de selectores disponibles y una gran variedad de propiedades de CSS. Esto último es algo que se obtiene con la experiencia, ya que W3Schools lista en su página web más de 200 propiedades distintas de CSS. En esta sección se cubrirán todos los tipos de selectores disponibles y algunas de las propiedades más importantes de CSS.

En el contexto de CSS, un selector es el mecanismo que se utiliza para encontrar (o seleccionar) los elementos HTML que se verán afectados por la sentencia.

**Figura 3**

*Tipos de selectores*

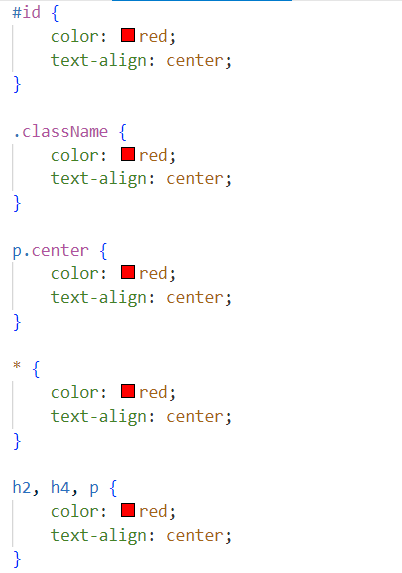
## Selectores simples

Se basan en el nombre, identificador o clase de los elementos para encontrar los elementos HTML que van a ser afectados por la sentencia.

El siguiente ejemplo muestra cómo usar estos selectores:

**Figura 4**

*Ejemplo de selectores simples*



Este código muestra los cinco tipos de selectores simples que existen, a los que habría que sumar el que se ha usado en el ejemplo del código de CSS con estilo. Cada uno de ellos obtiene el/los elementos HTML a estilizar de una forma diferente:

* **#id**

Este selector recoge un único elemento HTML, ya que busca aquel elemento cuyo atributo id sea idéntico al indicado.

* **.className**

Este selector puede recoger varios elementos HTML, ya que busca todos aquellos cuyo atributo class sea idéntico al indicado.

* **p.center**

Este selector puede recoger varios elementos HTML, ya que busca todos aquellos elementos <p> cuyo atributo class sea (o contenga) center.

* **\***

Se trata del selector universal. Recoge todos los elementos HTML.

* **h2, h4 , p**

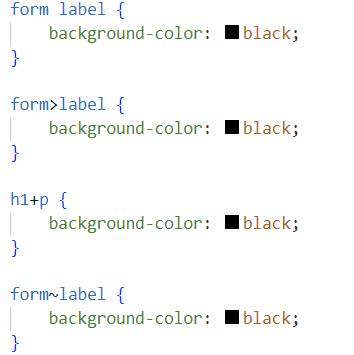
Se trata de una agrupación de selectores simples. Es útil utilizar este formato si queremos que <h2>, <h4> y <p> compartan una serie de estilos independientemente de su clase.

## Selectores combinados

Un selector combinado indica una relación que debe existir entre dos selectores. En CSS se distinguen cuatro tipos diferentes de selectores combinados.

**Figura 5**

*Ejemplo del uso de selectores combinados*



El fragmento de CSS muestra los cuatro tipos:

* **form label. Descendant selector.**

Sirve para seleccionar todos los elementos HTML descendientes de otro. En este caso, seleccionaríamos todos los <label> dentro de cualquier <form>.

* **form > label. Child selector.**

Sirve para seleccionar todos los elementos HTML hijos de otro. En este caso, seleccionaríamos todos los <label> hijos de cualquier <form>. La diferencia con respecto al anterior selector es que solo se tendrán en cuenta los elementos HTML cuyo padre sea el especificado en la parte izquierda del >.

* **h1 + p. Adjacent sibling selector.**

Sirve para seleccionar los elementos HTML que están directamente después de otro. En este caso, seleccionaríamos todos los <p> que se encuentren estrictamente después de un <h1>.

* **h1 ~ p. General sibling selector.**

Sirve para seleccionar los elementos HTML que están después de otro. En este caso, seleccionaríamos todos los <p> que se encuentren después de un <h1>. La diferencia respecto al anterior es que no es necesario que el segundo elemento esté estrictamente después del primero.

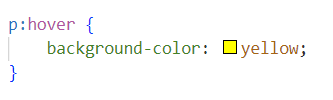
## Selectores de pseudoclase

Una pseudoclase en CSS se utiliza para indicar un estado especial para un elemento HTML. Es decir, son estilos condicionales que solo se muestran cuando la condición se cumple y el elemento HTML pasa a estar en el estado deseado.

Por ejemplo, cuando pasamos el ratón sobre una parte de la página.

**Figura 6**

*Selectores de pseudoclase*



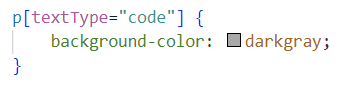
Este código muestra una sentencia en la que se aplicará un fondo de color amarillo a los elementos <p> cuando el ratón se pase por encima de ellos.

## Selectores de atributos

Un selector de atributos se utiliza para obtener uno o varios elementos HTML a través del valor de un atributo.

**Figura 7**

*Ejemplo de uso de un selector de atributos*



El fragmento de CSS muestra una sentencia en la que se aplicará un color de fondo gris oscuro a todos aquellos elementos <p> que contengan un atributo textType cuyo valor sea code.

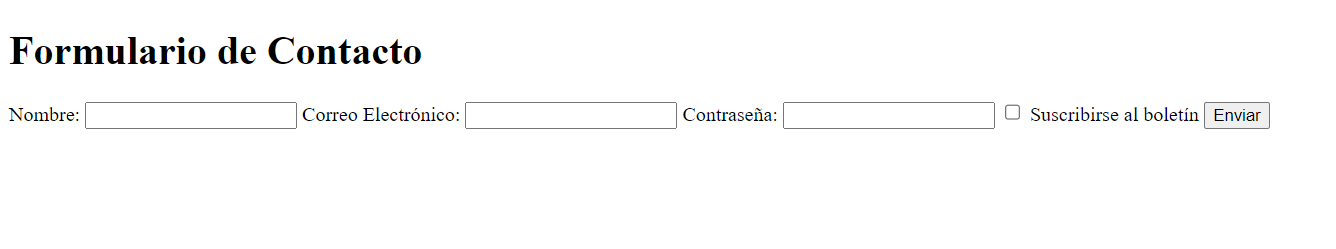
# CSS y HTML

A continuación, se muestra el código HTML que se usará en este apartado:

|  |
| --- |
| <!DOCTYPE html>  <html lang="es">  <head>      <meta charset="UTF-8">      <meta name="viewport" content="width=device-width, initial-scale=1.0">      <title>Formulario Básico</title>  </head>  <body>      <h1>Formulario de Contacto</h1>      <form action="/procesar\_formulario" method="post">          <!-- Campo de texto -->          <label for="nombre">Nombre:</label>          <input type="text" id="nombre" name="nombre" required>          <!-- Campo de correo electrónico -->          <label for="correo">Correo Electrónico:</label>          <input type="email" id="correo" name="correo" required>          <!-- Campo de contraseña -->          <label for="contrasena">Contraseña:</label>          <input type="password" id="contrasena" name="contrasena" required>          <!-- Casilla de verificación -->          <input type="checkbox" id="suscripcion" name="suscripcion">          <label for="suscripcion">Suscribirse al boletín</label>          <!-- Botón de envío -->          <button type="submit">Enviar</button>      </form>  </body>  </html> |

**Figura 8**

*Ejemplo completo del uso de HTML y CSS*



Para indicar el uso de un archivo CSS externo a un documento HTML es necesario incluir un <link> dentro del elemento <head> del archivo HTML, tal como vemos en el fragmento de código HTML anterior.

La hoja de estilos que usaremos es la siguiente:

|  |
| --- |
| /\* styles.css \*/  body {      font-family: Arial, sans-serif;      margin: 20*px*;      background-color: #f0f0f0;  }  h1 {      color: #333;  }  form {      max-width: 400*px*;      margin: 20*px* auto;      padding: 20*px*;      background-color: #fff;      border-radius: 5*px*;      box-shadow: 0 0 10*px* rgba(0, 0, 0, 0.1);  }  label {      display: block;      margin-bottom: 8*px*;  }  input,  button {      width: 100*%*;      padding: 10*px*;      margin-bottom: 10*px*;      box-sizing: border-box;  }  button {      background-color: #4caf50;      color: #fff;      cursor: pointer;  } |

Si dejamos el archivo HTML y el archivo CSS en una misma carpeta en el ordenador y abrimos el archivo HTML, podremos ver el resultado final de este ejemplo. Dado que estamos utilizando URL absolutas para referenciar las imágenes que se incluyen en el documento HTML, es posible que, si su contenido cambia o sencillamente se elimina, nuestra página se vea de forma diferente. En este caso se trata de un ejemplo, pero, en nuestros desarrollos, trataremos de descargar las imágenes e incluirlas en nuestros proyectos para poder referenciarlas mediante URL relativas y así evitar estos inconvenientes.

**Figura 9**

Ejemplo de inclusión de archivos CSS dentro de HTML



Resultado final

**Figura 10**

Ejemplo de un formulario usando HTML y CSS.



Las sentencias CSS que se han usado se traducen, de arriba abajo, en lo siguiente:

* body: Establece estilos generales para todo el cuerpo del documento.
* font-family: Define la fuente de texto para todo el cuerpo.
* margin: Agrega un margen de 20px alrededor del cuerpo.
* background-color: Establece el color de fondo del cuerpo.
* h1: Establece el estilo para los encabezados de nivel 1.
* color: Define el color del texto para los encabezados h1.
* form: Establece el estilo para el formulario.
* max-width: Limita el ancho máximo del formulario a 400px.
* margin: Añade un margen de 20px arriba y abajo, y automáticamente a los lados (auto), centrando el formulario.
* padding: Agrega un relleno interno de 20px al formulario.
* background-color: Establece el color de fondo del formulario.
* border-radius: Agrega esquinas redondeadas al formulario.
* box-shadow: Agrega una sombra alrededor del formulario.
* label: Establece el estilo para las etiquetas.
* display: Cambia el tipo de visualización a bloque para que cada etiqueta esté en una línea separada.
* margin-bottom: Añade un margen en la parte inferior de cada etiqueta para separarlas ligeramente.
* input, button: Establece el estilo para los campos de entrada y el botón.
* width: Hace que los campos de entrada y el botón ocupen el 100% del ancho disponible.
* padding: Agrega un relleno de 10px a los campos de entrada y al botón.
* margin-bottom: Añade un margen en la parte inferior para separarlos verticalmente.
* box-sizing: Ajusta el modelo de caja para incluir el relleno en el ancho total.
* button: Establece un estilo específico para el botón.
* background-color: Define el color de fondo del botón.
* color: Define el color del texto del botón.
* cursor: Cambia el cursor a una mano cuando pasa sobre el botón.

## BEM

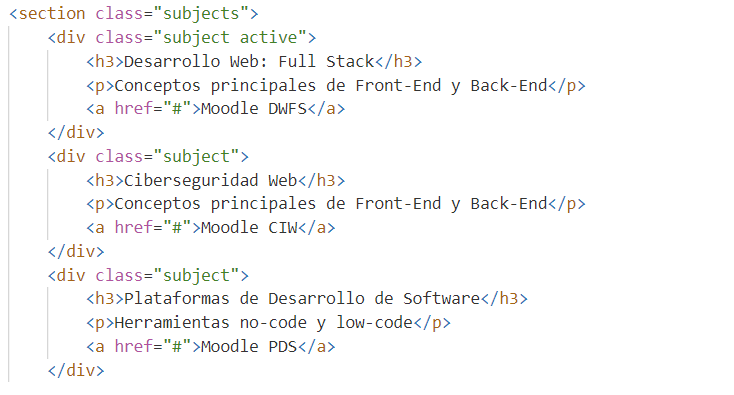
A la hora de trabajar con HTML y CSS de forma conjunta, es habitual que se definan los **estándares** para nombrar los estilos CSS que cada elemento HTML tendrá. Si bien dentro de cada empresa u organización podríamos encontrar **diferentes estrategias** para conseguir esto, existe una metodología de referencia usada que debemos conocer: **BEM** (bloque, elemento y modificador).

La metodología BEM persigue de una forma concreta el uso de los **selectores escritos** al atender la utilización que se les va a dar a los **elementos HTML asociados.** La mejor forma de entender esta metodología es mediante un ejemplo.

Observa el siguiente código HTML que muestra varias tarjetas de las asignaturas:

**Figura 11**

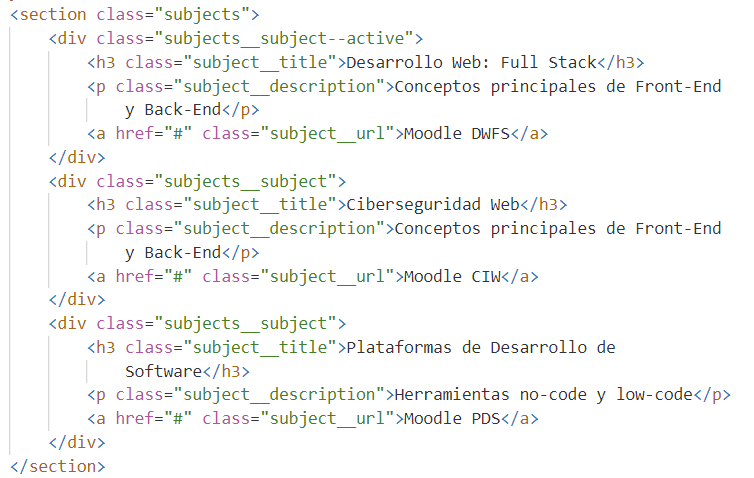
Ejemplo un formulario con diferentes secciones.



Como se puede observar, en algunos elementos HTML se asocian los nombres de las **clases CSS** que se están usando en una hoja de estilos externa. Ahora, observa este código donde se ha aplicado la metodología BEM:

**Figura 12**

Ejemplo de aplicación de la metodología BEM



Podemos observar lo siguiente:

* La **clase** subjects es un bloque, ya que se trata de una sección claramente diferenciada de la página que está haciendo referencia a una serie de asignaturas.
* El **nombre de las clases** de cada subject sigue la nomenclatura bloque\_\_elemento o elemento\_\_subelemento (siempre con guiones bajos). Es decir, cada subject individual es un elemento que forma parte del bloque subjects y, con ello, indica las propiedades de cada subject .
* El **modificador,** en este caso, es la clase que incluye el --active (siempre con guiones normales) y define un modificador al que se le aplica un elemento al que va asociado (en este ejemplo, subject ). Sin embargo, no tiene por qué estar asociado a todos los elementos, por tanto, será una modificación sobre el elemento original.

De esta forma, podemos utilizar únicamente el nombre de la clase para realizar reglas CSS, por lo que no será necesario usar las etiquetas HTML. No obstante, podremos seguir haciéndolo siempre que queramos, aunque, gracias a esta metodología, será mucho más fácil entender y hacer entender nuestro código.

# Grids en CSS

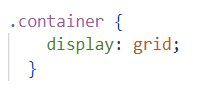
CSS Grid es una poderosa herramienta de diseño que permite crear diseños de página complejos y flexibles con facilidad. A diferencia de los sistemas de diseño más antiguos, como Flexbox, CSS Grid proporciona un enfoque bidimensional para organizar elementos en filas y columnas.

## Propiedades básicas

* **Grid Container:** El elemento contenedor que define el contexto de la cuadrícula.

**Figura 13**

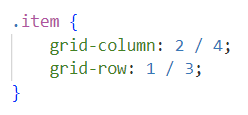
Ejemplo del uso de la propiedad grid container



* **Grid Items:** Los elementos hijos directos del contenedor de la cuadrícula.

**Figura 14**

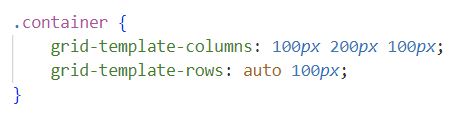
*Ejemplo del uso de la propiedad grid items*



* **Líneas de Cuadrícula:** Las líneas entre las columnas y filas de la cuadrícula.

**Figura 15**

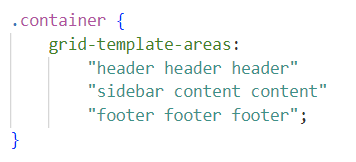
*Ejemplo del uso de líneas de cuadrículas*



* **Áreas de Cuadrícula:** Definir áreas en la cuadrícula para organizar elementos.

**Figura 16**

*Ejemplo del uso de áreas de cuadrículas*



# Tipografías

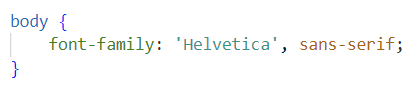
La tipografía desempeña un papel crucial en el diseño web, afectando la legibilidad y la estética general de un sitio. En CSS, hay varias propiedades que permiten personalizar y mejorar la apariencia de la tipografía.

## Propiedades básicas

* **font-family:**

**Figura 17**

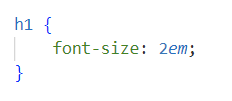
*Uso de la propiedad Font-family*



* **font-size:**

**Figura 18**

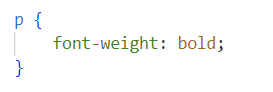
*Uso de la propiedad Font-size*



* **font-****weight:**

**Figura 19**

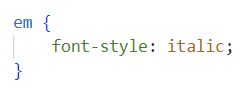
*Uso de la propiedad Font-weight*



* **font-style:**

**Figura 20**

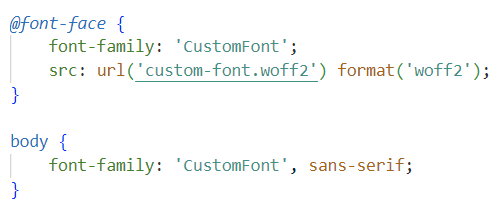
*Uso de la propiedad Font-style*



* **Fuentes Personalizadas:**

**Figura 21**

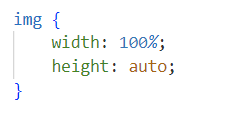
*CSS para fuentes personalizadas*



# Imágenes

Las imágenes son elementos esenciales en el diseño web. CSS proporciona diversas propiedades que permiten manipular y estilizar imágenes de manera efectiva.

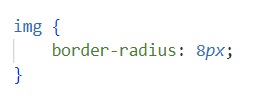
* width y height:



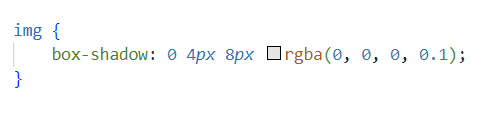
* object-fit:



* border-radius:

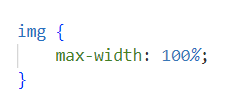


* box-shadow:

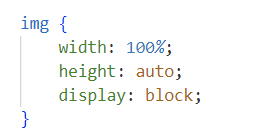


## Imágenes Responsivas

* max-width:



* Imágenes Retina (HiDPI):

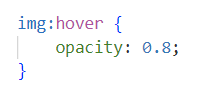


## Filtros y Efectos

* filter:

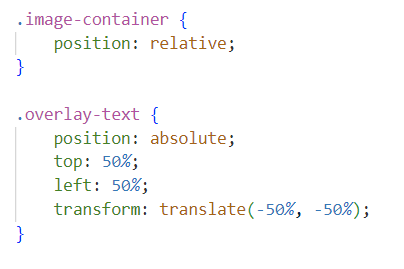


* Efectos de Hover:



## Posicionamiento y Overlay

* position: relative:

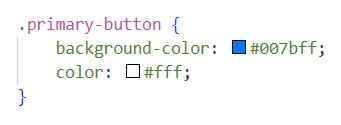


# Botones

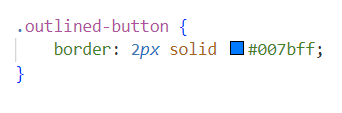
Los botones son elementos clave en las interfaces de usuario web. En CSS, existen varias propiedades que permiten estilizar y personalizar la apariencia de los botones.

## Propiedades Básicas

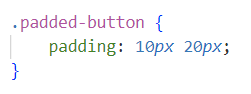
* **background-color y color:** Establecen el color de fondo y el color del texto del botón, respectivamente.



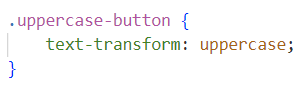
* **border:** Define el grosor, el estilo y el color del borde del botón.

****

* **padding:** Define el espacio interno (relleno) del botón, afectando el área entre el contenido y el borde.

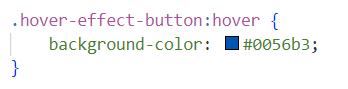
****

* **text-transform:** Controla la transformación del texto, en este caso, convierte el texto del botón a mayúsculas.

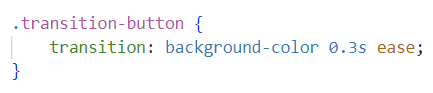
****

## Efectos de Hover y Transiciones

* **hover:** Aplica estilos cuando el cursor está sobre el botón, permitiendo crear efectos visuales interactivos.

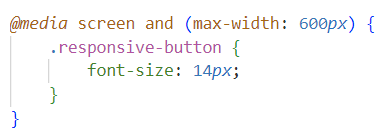
****

* **Transiciones (transition):** Controla la transición suave entre los estados de los estilos, mejorando la experiencia visual.

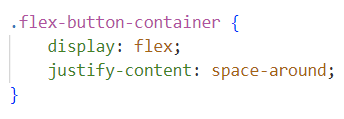
****

## Botones Responsivos

* **Media Queries:** Permite aplicar estilos específicos para diferentes tamaños de pantalla, haciendo que los botones sean responsivos.

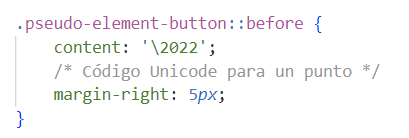
****

* **Flexbox:** Utiliza el modelo de diseño flexible para alinear y distribuir elementos, útil para crear diseños responsivos.

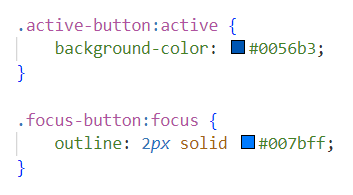


## Pseudo-Elementos y Pseudo-Clases

* **::before y ::after:** Permite agregar contenido antes o después del contenido real del botón.



* **:active y :focus:** 
  + :active aplica estilos cuando el botón está siendo presionado
  + :focus cuando el botón está enfocado (seleccionado por teclado o mouse).



# Contenedores e interacciones

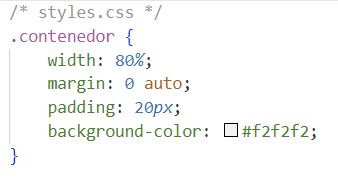
Los contenedores en CSS son elementos que agrupan y contienen otros elementos HTML. Pueden ser divs, secciones, o incluso elementos específicos como headers o footers. Estos contenedores sirven como bloques de construcción para estructurar la página y aplicar estilos de manera efectiva.

**Figura 22**

Incursión de contenedores e interacciones en HTML y CSS



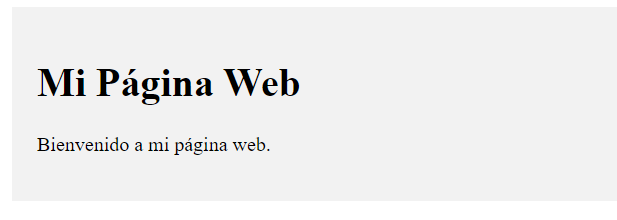
En el código del archivo styles.css, se observa que el div con la clase "contenedor" actúa como un contenedor principal con estilos aplicados a través de CSS.



La salida del código anterior

**Figura 23**

Incrustación de CSS dentro de una página HTML.

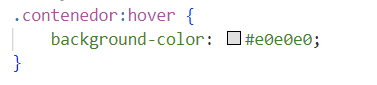


## Interacciones en CSS

Las interacciones en CSS se refieren a cómo los elementos responden a eventos del usuario o a cambios de estado. Esto incluye hover, focus, y animaciones. Estas interacciones son esenciales para mejorar la experiencia del usuario y agregar dinamismo al diseño.

**Figura 24**

Ejemplo de Interacción Hover



# Recursos complementarios

* Manual de referencia de los elementos HTML

<https://www.w3schools.com/tags/>

<https://www.w3schools.com/tags/ref_attributes.asp>

# Referencias

Brown, A. (2020). *Desarrollo web moderno: estrategias y herramientas*. Editorial Digital Solutions.

García, M.; Rodríguez, A. & Pérez, S. (2021). Tecnologías Back-End: un enfoque integral. *Revista de Desarrollo Tecnológico*, 15(2), 45-63. doi:10.1234/rdt.2021.1234567890

Johnson, R. (2018). *Desarrollo de interfaces de usuario: principios y prácticas*. España: Editorial TechPress.

Johnson, R. (2019). Desarrollo de software empresarial: principios y prácticas. Editorial TechPress.

Smith, J. (2018). *Diseño de interfaces de usuario: principios y prácticas*. Editorial TechBooks.

# Autoevaluación

1. **¿Cuál de las siguientes propiedades CSS se utiliza para establecer el color del texto?**

font-color

color

text-color

foreground-color

1. **¿Cuál de las siguientes unidades CSS es más apropiada para hacer que una imagen sea completamente responsiva?**

px

%

vw

em

1. **Para crear un efecto de transición suave al pasar el mouse sobre un botón, ¿cuál propiedad CSS debería utilizarse?**

transition

transform

animation

hover-transition

1. **¿Qué propiedad CSS se usa para ocultar un elemento sin afectar el diseño de la página?**

display: none;

visibility: hidden;

opacity: 0;

position: absolute;

1. **En un enlace <a> que redirige a otra página, ¿cuál atributo especifica la URL de destino?**

url

target

href

link

1. **Con respecto a CSS, ¿cuál de las siguientes unidades de medida es relativa al tamaño del texto del elemento padre?**

px

em

rem

vw

1. **¿Cuál es la ventaja principal de utilizar la propiedad object-fit en CSS al trabajar con imágenes?**

Ajusta el tamaño de la imagen según el contenido.

Permite definir el ancho y alto de la imagen.

Mantiene la relación de aspecto sin distorsionar la imagen.

Aplica un efecto de zoom a la imagen.

1. **¿Qué selector CSS seleccionaría todos los elementos de clase "destacado" dentro de un div con id "contenedor"?**

#contenedor .destacado

.destacado #contenedor

#contenedor > .destacado

#contenedor .destacado

1. **¿Qué propiedad CSS se utiliza para hacer que un elemento sea invisible pero aún ocupe espacio en el diseño?**

display: none;

visibility: hidden;

opacity: 0;

position: absolute;

1. **¿Cómo se declara un estilo de fuente personalizado en CSS?**

font-family

custom-font

font-style

@font-face