

Contenidos a Trabajar

1. ¿Qué es CSS?

1. Sintaxis

2. Vinculación

2. Herencia

3. Cascada

4. Selectores

1. Precedencia

2. Especificidad

5. Tipografías

6. Colores

¿Qué es CSS?


El es lenguaje de la web con el que podremos dar estilos a nuestras páginas, sitios y webapps. Sus siglas significan **Hojas de estilos en cascada** y vienen del inglés “*Cascading Style Sheets*”.

Actualmente se encuentra en su **versión 3** la cual introdujo severas mejoras frente a sus predecesores y además se está en constante actualización con implementaciones nuevas al lenguaje que nos proporcionan cada vez más herramientas y propiedades para trabajar nuestros estilos.

La idea detrás de este lenguaje es separar las responsabilidades en el desarrollo de un sitio web. Como vimos anteriormente, HTML posee etiquetas que brindan cierto estilo visual, sin embargo actualmente lo utilizamos únicamente para generar la estructura del contenido de nuestro sitio, mientras que CSS nos ocuparemos de posicionar y modificar visualmente esa estructura.

Sintaxis

La estructura CSS se basa en reglas que tienen el siguiente formato:



```
1 selector {  
2     propiedad: "valor";  
3 }
```

Cada sentencia posee 3 partes básicas : selector, propiedad y valor.

Además se encuentra definida entre un bloque de llaves { } y cada declaración de propiedad finaliza con un ; (punto y coma). En una misma sentencia pueden haber tantas declaraciones como estilos necesitemos aplicar a ese elemento seleccionado.

Selector

El selector es el elemento HTML que vamos a seleccionar del documento para aplicarle un estilo concreto. Por ejemplo, con p seleccionamos todas las etiquetas <p> de nuestro documento.

Propiedad

Es la propiedad CSS que queremos cambiar de ese elemento seleccionado, por ejemplo el color de fuente, el tamaño o su disposición.

Valor

Es el valor que se le asigna a la propiedad y que dictará esa modificación.

Vinculación

Existen 3 métodos para aplicar o vincular estilos CSS a nuestro HTML: **interno**, **externo** y **en línea**.

Externo

En la etiqueta `<head></head>` de nuestro documento, podemos incluir una etiqueta `<link />` que vincule nuestro archivo HTML con un archivo externo de extensión `.css`

```
3 <head>
4   <meta charset="UTF-8">
5   <title>Document</title>
6   <link rel="stylesheet" href="./index.css">
```

Interno

También podemos añadir nuestros estilos directamente en el documento HTML, a través de una etiqueta `<style></style>` también dentro de la etiqueta `<head></head>`.

```
3 <head>
4   <meta charset="UTF-8">
5   <title>Document</title>
6   <style>
7     p {
8       color: chartreuse;
9     }
10  </style>
11 </head>
```

En Línea

Se colocan directamente en el atributo `style` de la etiqueta a modificar.

```
13 <p style="color: blue;">Lorem ipsum dolor, sit amet
    consectetur adipiscing elit.</p>
```

Herencia

Es el concepto por el cual algunas propiedades CSS son heredadas desde las etiquetas padres hacia sus etiquetas hijas.

```
body {  
  color:  silver;  
}
```

En este ejemplo, todas las etiquetas hijas de body tendrán por defecto el color de texto **silver** salvo que la herencia se corte por otro elemento padre hacia su hijo:

```
div {  
  color:  teal;  
}
```

o por la asignación de un estilo propio:

```
div p {  
  color:  crimson;  
}
```


No todas las propiedades CSS son heredadas, porque algunas de ellas no tendría sentido que lo fueran. Por ejemplo, los márgenes no se heredan porque es poco probable que un elemento hijo necesite los mismos márgenes que su padre. Normalmente, el sentido común dicta qué propiedades se heredan y cuáles no.

Algunas de ellas son: **color**, **font-weight**, **font-family**, **float**, **line-height**, entre otras.

Cascada

Es el mecanismo que controla el resultado final cuando se aplican varios estilos o propiedades CSS iguales al mismo elemento.

```
p {  
  color: teal;  
}  
  
p {  
  color: crimson;  
}
```

Como bien lo indica su nombre, cuando nos encontramos con estos casos, siempre prevalece el último en orden descendente. Como en la imagen, nuestros párrafos tendrán color **crimson**.

Sin embargo, esto no es lo único que se debe tener en cuenta a la hora de resolver qué sucede con estilos que entran en conflicto.

Hay tres conceptos principales que controlan el orden en el que se aplican las declaraciones de CSS:

- Precedencia.
- Especificidad.
- Orden en que se importan los recursos o estilos.

Selectores

Cuando comenzamos a trabajar con CSS, es común aplicar estilos con selectores genéricos como una etiqueta HTML, por ejemplo un **div { }**. Sin embargo, lo que estamos haciendo en realidad es seleccionar todos los elementos del documento que sean dicha etiqueta, lo cual no siempre es lo que buscamos.

Por eso contamos con selectores adicionales que nos permiten afinar nuestra puntería con el objetivo de aplicar estilos a un elemento o un grupo determinado aunque estos no tengan relación aparente o sean de distintas etiquetas.

En este sentido, los selectores son principalmente: **etiquetas**, **clases** o **IDs**.

Etiquetas

Como vimos en ejemplos anteriores, es a través del nombre de la etiqueta que deseamos seleccionar.

```
a {  
  display: inline-block;  
}
```

Clases

A medida que vamos aplicando estilos y creando nuestros documentos HTML, comprobaremos que necesitamos cosas más flexibles y cómodas, aquí es donde entran en juego las clases de CSS.

Estas se colocan en la etiqueta mediante el atributo **class** y puede ser reutilizada tantas veces como sea necesario.



<codoa codo/>

```
<section>
  <article class="card">
    <h2 class="card__title">Titulo</h2>
    <p class="card__text">Párrafo</p>
  </article>
</section>
```

Al momento de utilizarlas en el CSS, se debe anteponer un punto (.) al nombre de la clase para diferenciarlo de un nombre de etiqueta.

```
.card {
  margin: 10px;
  display: block;
  width: 300px;
}
```

Con las clases podemos aplicar 2 conceptos diferentes. Por un lado, crear estilos **utilitarios** que apliquen un diseño específico independiente del elemento o el lugar que ocupe en la página. Un ejemplo de esto podría ser:



<codoa codoo/>

```
.btn {  
  background-color: teal;  
  color: white;  
  width: 120px;  
  padding: 10px 15px;  
}
```

En este caso, cualquier elemento de **botón** en el sitio tendrá el mismo estilo por defecto y unificado.

En cambio cuando creamos estructuras HTML complejas como una sección con productos de una tienda, las clases tienden a utilizarse para identificar cada elemento y su rol en ese grupo o sección.

```
.card__title {  
  color: indianred;  
  font-size: 22px;  
}
```


IDs

Todas las etiquetas HTML pueden tener un atributo id con un valor concreto, pero este **NO** podrá repetirse ni ser reutilizado en otras etiquetas en el mismo documento.

Normalmente se utilizan para identificar secciones del sitio que luego podrán ser accedidas mediante enlaces.

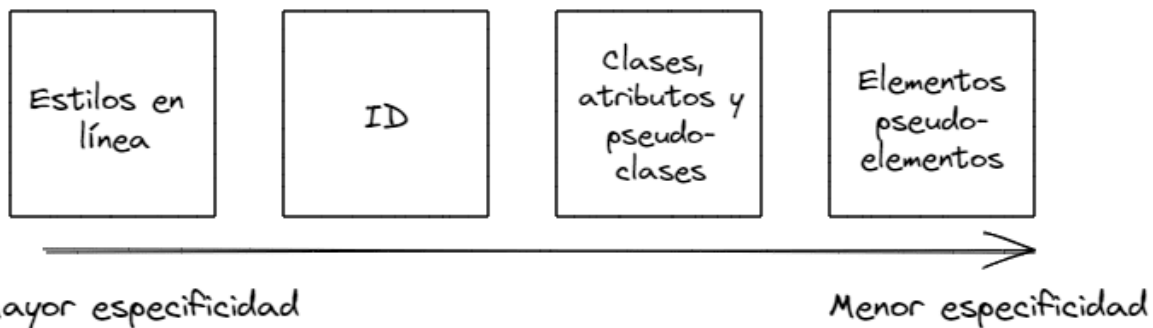
```
<section id="cards">
  <article class="card">
    <h2 class="card__title">Titulo</h2>
    <p class="card__text">Párrafo</p>
  </article>
</section>
```

Para invocarlas desde nuestro CSS se utiliza el símbolo de numeral (#) con el mismo propósito que las clases.

```
#cards {
  max-width: 998px;
  margin: 0 auto;
}
```

Precedencia

Cuando tenemos varias reglas CSS en cascada que afectan a un mismo elemento, el orden de prioridad que determina cómo se mostrará el elemento es el siguiente:



En caso que necesitemos cortar con la **precedencia**, podemos utilizar la palabra **!important** entre el valor de la propiedad CSS y el punto y coma (;) final de esa declaración.

```
p {  
  color: ■ crimson!important;  
}
```

Especificidad

La especificidad en CSS es un grupo de reglas aplicadas a los selectores CSS para determinar qué estilo se aplica a un elemento. Cuanto más específico sea un selector, mayor será su valor en puntos y más probable será que esté presente en el estilo del elemento.

La especificidad tiene cuatro componentes; por ejemplo a, b, c y d. El componente "a" es el más distintivo y el "d", el que menos.

- El componente "**a**" es bastante sencillo: es 1 para una declaración en un atributo **style**; si no, es 0.
- El componente "**b**" es el número de selectores de **id** en el selector (los que empiezan con #).
- El componente "**c**" es el número de selectores de atributo, incluidos los selectores de **clase** y pseudoclases.
- El componente "**d**" es el tipo de etiqueta y pseudo elementos del selector.

Selector	a	b	c	d	Especificidad
h1	0	0	0	1	0,0,0,1
.foo	0	0	1	0	0,0,1,0
#bar	0	1	0	0	0,1,0,0
html >head+body ul#nav *.home a:link	0	1	2	5	0,1,2,5

Dependiendo de la suma de selectores y la jerarquía de cada uno es que se determina si un selector de un elemento es más o menos específico que otro.

Tipografías

Las tipografías (o fuentes) son uno de los pilares del diseño web. La elección de una tipografía adecuada y sus características como tamaño, color, espaciado o interlineado pueden cambiar la percepción de un usuario sobre nuestro sitio.

En diseño web se recomienda tener no más de entre **dos** o **tres** familias tipográficas distintas ya que contar con **una** sola podría dejar un resultado monótono mientras que agregar mucha variedad de fuentes podría confundir al usuario y afectar su experiencia.

Además tenemos que tener en cuenta los tipos de familias tipográficas existentes y sus características para elegir fuentes acordes a lo que queremos transmitir.

En este sentido, vamos a conocer los distintos grupos de fuentes que normalmente nos podemos encontrar:

Serif: son aquellas fuentes que poseen adornos, florituras o remates en su composición. Se las identifica con un carácter elegante y suelen ser usadas por la

prensa escrita, así como en la literatura, ya que sus formas ayudan a no perder el hilo del texto.

Esta es una tipografía Serif

Sans-Serif: conocidas como tipografías de paloseco, estas carecen de adornos y tienen terminaciones rectas o ligeramente redondeadas dándole acabados más minimalistas. Son muy utilizadas para los medios digitales o todo aquello que se vaya a reproducir en una pantalla

Esta es una tipografía Serif

Script: también llamadas **Handwriting**, estas tipografías pueden tener un aspecto cursiva aunque no siempre y simula la escritura a mano. Utilizadas comúnmente cuando se busca destacar un texto o simular este tipo de escritura.

Esta es una tipografía script o handwriting



<codoa
codo/>

Esta es una tipografía script o handwriting

Display: se las conoce como las tipografías decorativas y suelen ser divertidas, más desenfadadas, pero pueden transmitir una gran variedad de sensaciones por lo que sirven muy bien para grandes titulares. Desde el punto de vista de la psicología tipográfica son transgresoras y contribuyen a llamar más la atención. No obstante, la legibilidad que se puede obtener con este tipo de fuentes es algo más pobre.

ESTA ES UNA TIPOGRAFÍA DISPLAY

Esta es una tipografía display

Monoespaciadas: en esta tipografía cada una de sus letras tienen exactamente el mismo ancho, inspiradas en las antiguas máquinas de escribir, son muy útiles para tareas de programación o emuladores de terminal o cuando se desea dar un aspecto similar. También se usan en guiones cinematográficos ya que gracias a su tamaño uniforme se puede medir el tiempo aproximado de una escena desde la cantidad de líneas del guión.

Esta es una tipografía mono

Agencia de
Aprendizaje
a lo largo
de la vida

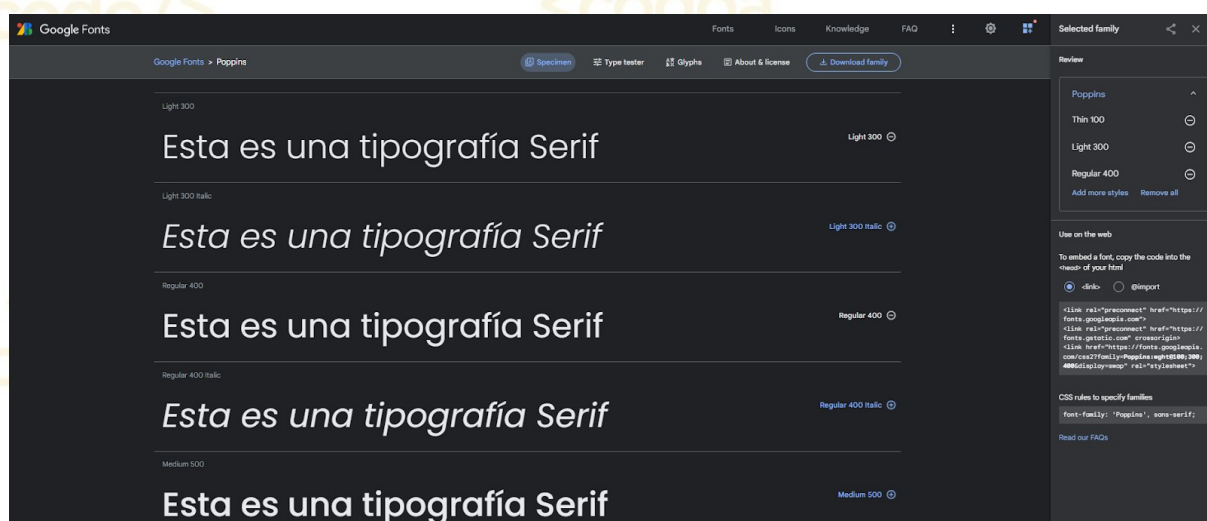


Uso de las fuentes y propiedades

Para utilizar fuentes en nuestros proyectos tenemos distintas alternativas. Podemos vincular un recurso externo a través de nuestro archivo HTML o también existe la posibilidad de hacerlo desde el CSS donde no solo nos permite importar recursos de sitios como Google Fonts que vienen por un enlace web, si no que podemos importar nuestros propios archivos de fuente en formato .ttf.

Desde HTML o CSS con Google Fonts.

Lo primero es entrar a <https://fonts.google.com/> y seleccionar las fuentes que queremos importar.



Una vez seleccionada, el sitio nos brinda 2 opciones:



<codoa codo/>

Use on the web

To embed a font, copy the code into the <head> of your html

☒ <link> ☐ @import

```
<link rel="preconnect" href="https://  
fonts.googleapis.com">  
<link rel="preconnect" href="https://  
fonts.gstatic.com" crossorigin>  
<link href="https://fonts.googleapis.  
com/css2?family=Poppins:wght@100;300;  
400&display=swap" rel="stylesheet">
```

CSS rules to specify families

```
font-family: 'Poppins', sans-serif;
```

Use on the web

To embed a font, copy the code into the <head> of your html

☐ <link> ☒ @import

```
<style>  
@import url('https://fonts.googleapi  
s.com/css2?family=Poppins:wght@100;30  
0;400&display=swap');  
</style>
```

CSS rules to specify families

```
font-family: 'Poppins', sans-serif;
```

En el primer caso, tenemos **tres** etiquetas <link /> que debemos copiar y pegar en el <head></head> de nuestra página.

```
<link rel="preconnect" href="https://fonts.googleapis.com">  
<link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>  
<link href="https://fonts.googleapis.com/css2?family=Poppins:wght@100;300;400&  
display=swap" rel="stylesheet">
```

En el segundo caso, podemos importarlo directamente desde el CSS:

```
@import url('https://fonts.googleapis.com/css2?family=Poppins:wght@100;300;400&  
display=swap');
```

Agencia de
Aprendizaje
a lo largo
de la vida

Finalmente, utilizamos la familia tipográfica con la propiedad llamada **font-family** tal como nos indica el google fonts.

```
body {  
  font-family: 'Poppins', sans-serif;  
}
```

Desde CSS con un archivo de fuente.

Para estos casos debemos tener la fuente en formato **.ttf** en una carpeta dentro de nuestro proyecto e importarla a nuestro CSS a través de la regla **@font-face**

```
1 @font-face {  
2   font-family: <un-nombre-de-fuente-remota>;  
3   src: <origen> [<origen>]*;  
4   [font-weight: <peso>];  
5   [font-style: <estilo>];  
6 }
```


<codoa codoo/>

```
1 @font-face {  
2   font-family: "Bangers", sans-serif;  
3   src: url("../fonts/bangers.ttf");  
4 }
```

Una vez importada, para utilizarla se usa nuevamente la propiedad **font-family** con el nombre asignado a la fuente, al igual que en el caso anterior.

Propiedades CSS para fuentes

- **font-family:** nombre de la familia tipográfica que queremos aplicar.
- **font-size:** tamaño que deseamos darle a la fuente.
- **font-weight:** peso de la fuente, es decir, cuanto más gruesa o delgada.
- **font-style:** nos permite indicar si queremos que sea cursiva, normal u oblicua.

Colores

Antes de entender cómo utilizar colores en CSS, debemos repasar los distintos sistemas de color y cuales son soportados o utilizados para los medios digitales.

Sistemas Aditivo y Sustractivo

En medios digitales el sistema utilizado es el **Sistema Aditivo**, que lleva su nombre porque en este sistema los colores se suman cuando se superponen. En tanto, en la producción gráfica y plástica (pinturas acrílicas, óleos, témperas) se utiliza el **Sistema Sustractivo**.

SUSTRATIVOS
Colores en dibujo, pintura, imprenta...



ADITIVOS
Colores luz, focos, pantallas...



© dibujarfácil.com 2009

En el dominio digital quienes poseen los colores son los píxeles, la unidad mínima en pantalla que tienen dos propiedades básicas: ubicación y color. Los componentes de color de cada píxel se representa, como menciona la primera definición elegida, como secuencias de números. En el caso del modo de color **RGB (R=Red; G=Green; B=Blue)**, usualmente cada color o componente tiene un rango de 8 bits, lo que otorga 256 valores posibles para cada canal, siendo 0 es el valor mínimo y 255 su valor máximo.

Modelos de color

El **sistema aditivo**, como observamos, se compone de la suma de tres colores primarios, pero esa no es la única forma de generar colores.

Existen distintos modelos matemáticos con los que pueden conseguirse los mismos colores. Estos se llaman modos de color, que según el caso, pueden ofrecer mayores facilidades para realizar la composición resultante.

Modelo RGB

RGB se trata de un modelo cromático mediante el cual seremos capaces de representar distintos colores a partir de la mezcla de estos tres colores primarios.

En código HTML o CSS por ejemplo para representar los distintos colores RGB existe un código formado por tres números separados que pueden tomar valores desde el 0 hasta el 255. Cada uno de estos números representa uno de los colores siendo [Red],[Green],[Blue] y dependiendo del valor del número que haya en su

interior, la luminancia de ese color será mayor o menor. Por ejemplo si tenemos **rgb(0, 255, 0)**, tendríamos el color **verde** representado en pantalla, si tuviéramos el **rgb(255, 255, 255)** tendríamos el color blanco y en caso de tener **rgb(0, 0, 0)** el color negro, por nombrar algunos ejemplos.

Color	Valor RGB
Negro	rgb(0, 0, 0)
Blanco	rgb(255, 255, 255)
Rojo	rgb(255, 0, 0)
Azul	rgb(0, 0, 255)
Verde	rgb(0, 255, 0)
Amarillo	rgb(255, 255, 0)
Magenta	rgb(255, 0, 255)
Cian	rgb(0, 255, 255)
Violeta	rgb(136, 0, 255)
Naranja	rgb(255, 136, 0)

Modelo Hexadecimal

Los más populares para la web son los códigos Hexadecimales (es decir, compuesto de seis dígitos), cada uno de ellos comprendidos entre el **0 al 9** y la **A a la F**. Estos seis dígitos se agrupan de a pares que representan la intensidad de rojo, verde y azul respectivamente, dónde 00 corresponde a la ausencia total del color y FF la presencia completa. Este sistema se asemeja al modelo rgb donde en lugar de pares, tenemos números que van de 0 a 255.

	IndianRed	#CD5C5C	rgb(205, 92, 92)
	LightCoral	#F08080	rgb(240, 128, 128)
	Salmon	#FA8072	rgb(250, 128, 114)
	DarkSalmon	#E9967A	rgb(233, 150, 122)
	LightSalmon	#FFA07A	rgb(255, 160, 122)

Tanto el sistema RGB como el Hexadecimal aceptan el denominado canal **alpha** que representa transparencia. En el primer este se aplica agregando una “a” a la propiedad CSS.

rgba(0, 0, 0, 1) - En este ejemplo el último elemento representa la transparencia u opacidad cuyo valor puede ir de 0 a 1, donde 0 es 100% transparente y 1 es 100% opaco.

Mientras que en el código hexadecimal esto se representa agregan un par adicional de valores al código de 6 dígitos.

#000000FF - En este caso, tenemos obtenemos un color negro sólido.

Colores en HTML y CSS

Primero vamos a ver las propiedades CSS que podemos utilizar para cambiar el color de texto y el color de fondo de un elemento HTML:

color: modifica el color del texto del elemento.

background-color: modifica el color de fondo del elemento.

Estas propiedades aceptan como entrada los códigos de color, rgb, rgba y hexadecimal como vimos anteriormente y también colores ofrecidos por el navegador de forma nativa, por ejemplo: **snow**, **salmon** o **crimson**.