## **Tailwind CSS**

**Tailwind CSS es un framework CSS que permite un desarrollo ágil, basado en clases de utilidad que se pueden aplicar con facilidad en el código HTML y unos flujos de desarrollo que permiten optimizar mucho el peso del código CSS.**

[Tailwind CSS](https://tailwindcss.com/) es una potente herramienta para el desarrollo frontend. Está dentro de la clasificación de los frameworks CSS o también llamados frameworks de diseño. Permite a los desarrolladores y diseñadores aplicar estilos a los sitios web de una manera ágil y optimizada.

Tailwind permite escribir los estilos por medio de clases que se incluyen dentro del código HTML y que afectan a un aspecto muy concreto y específico de las CSS, por ejemplo, el fondo de un elemento, el color del texto o simplemente el margen por la parte de arriba. Este enfoque se conoce como "Atomic CSS", por aplicarse mediante estilos muy determinados y simples. En Tailwind CSS a estas clases se les llama "utility classes" o clases de utilidad en español.

Por tanto, Tailwind CSS no aporta muchos componentes. De hecho ofrece muy pocos componentes y son los desarrolladores los que los tienen que realizar bajo demanda del proyecto. En cambio lo que propone es entregar una enorme cantidad de clases de utilidad que combinadas en distintas variantes ofrecen prácticamente un número ilimitado de variantes de diseño, que permite una personalización del aspecto realmente única para cada proyecto.

Además Tailwind es una herramienta que se apoya en [PostCSS](https://desarrolloweb.com/home/postcss) para todo lo que es la generación del codigo CSS. Gracias a PostCSS se alcanza un flujo de desarrollo muy avanzado, personalizable, agil y sobre todo, extremadamente optimizado. Con ello, el código CSS resultante de un proyecto es realmente ajustado, porque con PostCSS se consige hacer que las clases que finalmente estén en el código de producción sean solamente las que el proyecto está usando realmente, ni una más.

**Menu:**

* [Enfoque de clases de utilidad](https://desarrolloweb.com/home/tailwind-css#track170)
* [Tailwind CSS vs Bootstrap](https://desarrolloweb.com/home/tailwind-css" \l "track171)
* [Bibliotecas de componentes en Tailwind CSS](https://desarrolloweb.com/home/tailwind-css#track248)



## **Enfoque de clases de utilidad**

En Tailwind le llaman "Utility-first workflow" y básicamente consiste en el modo de trabajo que nos propone el framework, en el que tenemos que modificar el HTML para insertar las clases que sean necesarias para aplicar estilo a los componentes.

Las clases de utilidad son atómicas, es decir, afectan a un estilo muy concreto, aportando un valor determinado. Por ejemplo:

* p-8: altera el padding para que sea de 2rem (0.25 \* 8 rem)
* mt-2: altera el margin-top para que sea 0.5rem (0.25 \* 2 rem)
* flex: pone display a flex
* items-center: aplica el estilo "aling-items: center"
* text-orange-300: pone el color del texto a naranja más bien claro.
* bg-gray-800: pone el color de fondo gris muy oscuro.

Estas clases se puede aplicar todas a la vez, o en cualquier subconjunto de ellas, sobre un elemento, de esta manera:

<div class="p-8 mt-2 flex items-center text-orange-300 bg-gray-800">

Contenido del elemento...

</div>

Utility-first workflow obliga a escribir muchas clases de utilidad en un elemento, cargando el HTML de numerosas clases que definen el estilo. En cierto modo quiebra la separación entre contenido y presentación que se tiene con HTML y CSS, pero tiene algunas ventajas importantes:

* Los estilos se pueden escribir sin necesidad de tocar el CSS
* El CSS de un proyecto no aumenta cuando se realizan especializaciones de estilo para algún elemento en particular.
* En las clases de utilidad podemos escribir estilos que sólo afectarán en determinadas variantes de pseudo-clases, como hover o focus
* Mediante las clases de utilidad podemos aplicar estilos que solo afectarán a un contenido en determinada anchura de pantalla.
* Las clases de utilidad se pueden personalizar, creando un tema gráfico que se acompaña durante todo el diseño.
* Tailwind CSS permite crear componentes por medio de especificación de diversas clases de utilidad, de modo que se puede reducir el uso de las mismas en los casos en los que se juzgue oportuno.
* El desarrollo se hace mucho más rápido.

Todo esto en detrimento de ensuciar el y engordar el HTML. Aunque en lo que respecta al peso, el hecho de que las páginas viajan gzippeadas, hace que ese aumento no resulte tan problemático, porque repetir textos como los nombres de clases en un documento varias veces es un patrón que se presta a una buena compresión.

## **Tailwind CSS vs Bootstrap**

Bootstrap y otros frameworks como Materialize CSS tienen un enfoque orientado a componentes. Ofrecen clases que permiten definir componentes complejos de interfaz gráfica. Por su parte Tailwind tiene un enfoque orientado hacia clases de utilidad.

En Bootstrap podemos definir una tarjeta de esta manera:

<!-- BOOTSTRAP -->

<div class="card" style="width: 18rem;">

<img src="..." class="card-img-top" alt="...">

<div class="card-body">

<h5 class="card-title">Título de la tarjeta</h5>

<p class="card-text">Estamos definiendo las diferencias entre frameworks CSS</p>

<a class="btn btn-primary" href="#">Entrar</a>

</div>

</div>

Como puedes ver, para definir la tarjeta Boostrap usa clases orientadas a componentes, como "card", "card-tile", etc. Para definir un botón usa "btn" y "btn-primary".

En Tailwind CSS este mismo ejemplo lo podríamos haber definido con un HTML como este:

<!-- TAILWIND CSS -->

<article class="mb-5 bg-teal-100 border border-gray-200 rounded shadow-md">

<img src="..." alt="..." class="rounded rounded-b-none">

<div class="p-3">

<h3 class="mb-1 text-xl font-bold">Los megalitos del Alentejo</h3>

<p class="mb-4">Acercarse al megalitismo más impactante de la Península Ibérica en el Alentejo</p>

<a class="bg-orange-600 text-white px-4 py-2 font-bold uppercase rounded-xl tracking-wider" href="#">Primary</a>

</div>

</article>

En este caso las diferencias son palpables. No tenemos un componente "card", sino que hemos generado una tarjeta a base de incorporar diversas clases. En vez de "card" tenemos algo como "mb-5 bg-teal-100 border border-gray-200 rounded shadow-md".

Bajo una impresión superficial podríamos pensar que resulta menos interesante, que necesitamos trabajar más y que hemos ensuciado más el HTML. En parte es cierto, pero las clases de utilidad nos ofrecen diversas ventajas, de las cuales ya hemos hablado. Pero además, en la comparativa con frameworks orientados a componentes como Bootstrap, las ventajas de Tailwind CSS son las siguientes:

* El componente de tarjeta es completamente personalizable. Simplemente aplicando alguna clase de utilidad diferente podemos cambiar el borde redondeado, para hacerlo más o menos sutil, el sombreado de la tarjeta, para realzarla más o menos, el grosor de su borde o cualquier otra cuestión.
* No necesitamos luchar contra el framework para conseguir escribir estilos que sobreescriban la tarjeta, lo que muchas veces es una tarea tediosa.
* El código CSS no crece sin control. Es decir, Bootstrap te obliga a seguir escribiendo tu propio CSS para personalización, mientras que Tailwind no necesita que escribas más CSS, porque cualquier estilo ya te lo ofrecen las clases de utilidad.

Pese a lo visto el código anterior, cabe decir que Tailwind también es amigo de la componetización. Si lo deseas puedes pasar a lo que tinenes en Bootstrap, creando clases propias que agrupan varias clases de utilidad, gracias a su directiva @apply, de modo que puedes reutilizar estilos bajo demanda.

Por último Tailwind se usa encima de herramientas de frontend sólidas como PostCSS, de modo que el enfoque es que puedas apdatar tu flujo de desarrollo con Tailwind CSS en torno de diversas herramientas de optimización. Así, el código que llevas a producción con Tailwind CSS siempre es reducido y acotado a las partes del framework que realmente estás usando.

Por último, Tailwind es un framework para quien sabe CSS, ya que para usarlo necesitas saber CSS y entender cómo funcionan sistemas como Flexbox o Grid Layout. En Bootstrap necesitas aprender el framework por ejemplo para saber usar su sistema de rejilla para crear un layout, mientras que en Tailwind CSS usas el propio CSS mediante las clases de utilidad para hacer lo mismo. No solo es que no te obligue a aprender algo nuevo, sino que te permite una mayor adaptabilidad a cualquier necesidad del proyecto.

## **Bibliotecas de componentes en Tailwind CSS**

Podemos encontrar sitios web con bibliotecas de componentes listos para usar en Tailwind. Estos componentes vienen bien cuando queremos tener opciones rápidas para crear interfaces de usuario, solo con copiar y pegar un código HTML con sus clases aplicadas.

Recordemos que en Tailwind CSS no tienes componentes "de casa", sino que los puedes crear tú mismo. Esto puede considerarse una ventaja, porque los diseños son siempre personalizados, pero también una desventaja porque la aplicación de Tailwind a la creación de interfaces no es tan directa como en otros frameworks clásicos, basados en componentes.

### **DaisyUI**

Es un repositorio de componentes creados por un único autor, que mantienen una buena consistencia de diseño, aparte de ser muy atractivos visualmente.

Funciona como un plugin de Tailwind que, una vez instalado permite usar todos los componentes disponibles con una cantidad de clases de CSS menores, que si tuvieras que hacerlos directamente con clases de utilidad de Tailwind CSS.

Además son altamente personalizables vía clases de utilidad y tiene una gestión de tema gráfico, mediante el cual puedes incorporar estilo a tus componentes de manera global y consistente en todo un proyecto.

Desde luego, si estás buscando componentes de TailWind CSS es una de las mejores opciones, ya que encima su uso es completamente gratuito. Más información en la página de [DaisyUI](https://daisyui.com/)

### **Tailwind UI**

El sitio más conocido para encontrar componentes de Tailwind es [Tailwind UI](https://tailwindui.com/). Este sitio es oficial de Tailwind, creado por los mismos desarrolladores del framework. Sin embargo, es un complemento de pago.

Este sitio no se limita solo a crear componentes, sino que también te ofrece layouts completos de aplicaciones creados por completo con Tailwind.

### **Tailwind Components**

[TailwindComponents](https://tailwindcomponents.com/) es un sitio está muy bien para encontrar componentes creados por la comunidad en Tailwind. Todos los componentes son gratuitos, creados por los mismos usuarios del sitio. Usarlos es solo copiar y pegar el HTML proporcionado. Hay componentes de todo tipo, para hacer interfaces simples de todo tipo, como botones, navegadores, barras de acciones, campos de textos, etc.

Ofrece también una sección para obtener layouts completos de sitios web con un diseño variado y aplicable a todo tipo de webs.