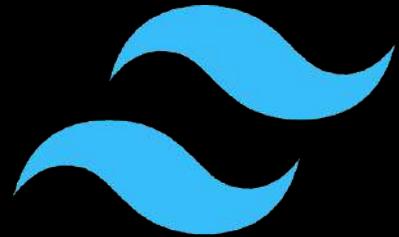


# tailwindcss

**Rapidly build modern websites  
without ever leaving your HTML.**

A utility-first CSS framework packed with classes like `flex`, `pt-4`, `text-center` and `rotate-90` that can be composed to build any design, directly in your markup.

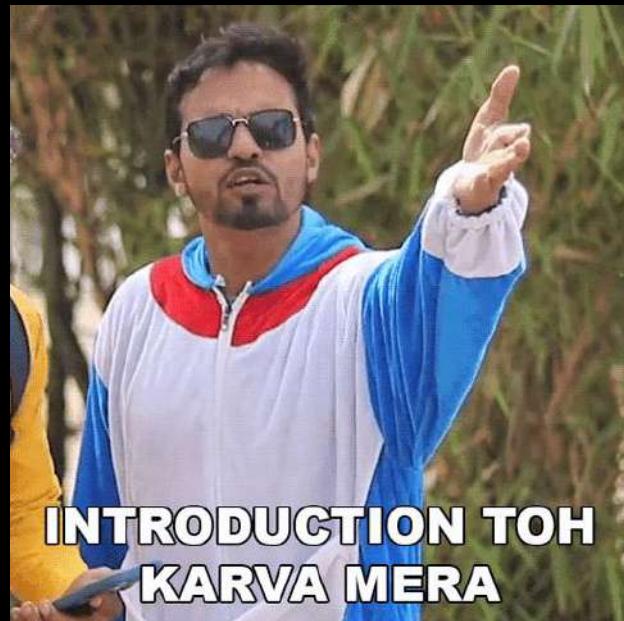




# tailwindcss

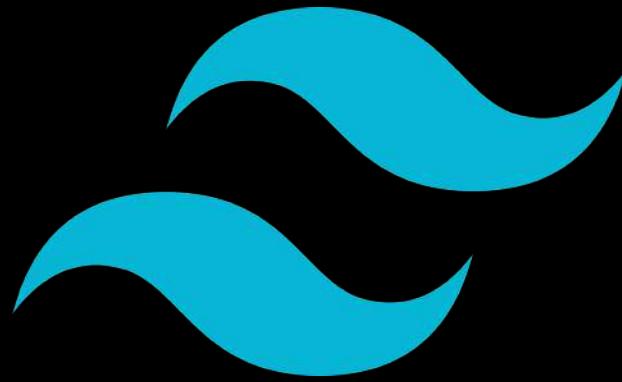
## INTRODUCTION

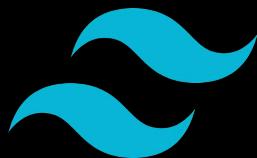
- What is tailwind-css
- Utility Classes
- Core Components
- Installation and Usage
- Using Pre-Built Components



# What is Tailwind CSS

1. **Responsive:** Mobile-first design for all device sizes.
2. **Utility-First:** Provides low-level utility classes for building custom designs.
3. **Highly Customizable:** Easily extendable through a config file.
4. **Responsive Design:** Built-in responsive utilities (e.g., sm:, md:).
5. **No Predefined Components:** Focuses on building custom components.
6. **Purge CSS:** Removes unused styles in production for smaller files.
7. **Fast Development:** Style elements directly in markup for speed.



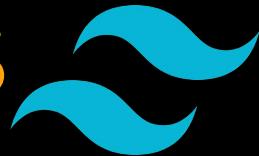


# Utility Classes

```
1  /* Colors */
2  .text-primary { color: #007bff; }
3  .bg-primary { background-color: #007bff; }
4
5  /* Sizing */
6  .w-full { width: 100%; }
7  .h-full { height: 100%; }
8
9  /* Typography */
10 .text-center { text-align: center; }
11 .font-bold { font-weight: bold; }
12
13 /* Spacing */
14 .m-1 { margin: 0.25rem; }
15 .m-2 { margin: 0.5rem; }
16 .p-1 { padding: 0.25rem; }
17 .p-2 { padding: 0.5rem; }
```

```
19  /* Layout */
20  .d-flex { display: flex; }
21  .flex-col { flex-direction: column; }
22  .items-center { align-items: center; }
23  .justify-center { justify-content: center; }
24
25  /* Misc */
26  .rounded { border-radius: 0.25rem; }
27  .hidden { display: none; }
```

# Core Components Tailwind CSS



## Utility-First Fundamentals

Using a utility-first workflow to build complex components from a constrained set of primitive utilities.



## Hover, Focus & Other States

Style elements in interactive states like hover, focus, and more using conditional modifiers.



## Reusing Styles

Manage duplication and keep your projects maintainable by creating reusable abstractions.



## Responsive Design

Build fully responsive user interfaces that adapt to any screen size using responsive modifiers.



## Dark Mode

Optimize your site for dark mode directly in your HTML using the dark mode modifier.



## Customizing the Framework

Customize the framework to match your brand and extend it with your own custom styles.

# FAQs Tailwind

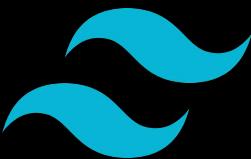
## Why not just use inline styles?

A common reaction to this approach is wondering, “isn’t this just inline styles?” and in some ways it is — you’re applying styles directly to elements instead of assigning them a class name and then styling that class.

But using utility classes has a few important advantages over inline styles:

- **Designing with constraints.** Using inline styles, every value is a magic number. With utilities, you’re choosing styles from a predefined design system, which makes it much easier to build visually consistent UIs.
- **Responsive design.** You can’t use media queries in inline styles, but you can use Tailwind’s responsive utilities to build fully responsive interfaces easily.
- **Hover, focus, and other states.** Inline styles can’t target states like hover or focus, but Tailwind’s state variants make it easy to style those states with utility classes.

# Playing with Tailwind



tailwind PLAY Share

HTML CSS Config

v3.4.10 Tidy

```
1 <!--
2 Welcome to Tailwind Play, the official Tailwind CSS playground!
3
4 Everything here works just like it does when you're running Tailwind locally
5 with a real build pipeline. You can customize your config file, use features
6 like '@apply', or even add third-party plugins.
7
8 Feel free to play with this example if you're just learning, or trash it and
9 start from scratch if you know enough to be dangerous. Have fun!
10 -->
11 <div class="relative flex min-h-screen flex-col justify-center overflow-hidden bg-gray-50 py-6 sm:py-12">
12   
13   <div class="absolute inset-0 bg-[url(/img/grid.svg)] bg-center [mask-image:linear-gradient(180deg, white, rgba(255,255,255,0))]"></div>
14   <div class="relative bg-white px-6 pb-8 pt-10 shadow-xl ring-1 ring-gray-900/5 sm:mx-auto sm:max-w-lg sm:rounded-lg sm:px-10">
15     <div class="mx-auto max-w-md">
16       
17       <div class="divide-y divide-gray-300/50">
18         <div class="space-y-6 py-8 text-base leading-7 text-gray-600">
19           <p>An advanced online playground for Tailwind CSS, including support for things like:</p>
20           <ul class="space-y-4">
21             <li class="flex items-center">
22               <svg class="h-6 w-6 flex-none fill-sky-100 stroke-sky-500 stroke-2" stroke-linecap="round" stroke-linejoin="round">
23                 <circle cx="12" cy="12" r="11" />
24                 <path d="M8 13 2.165a1 1 0 0 0 1.521-.126L16 9" fill="none" />
25               </svg>
26             <p class="ml-4">
27               Customizing your
28               <code>tailwind.config.js</code> file
29             </p>
30           </li>
31           <li class="flex items-center">
```

Generated CSS 2.05 kB

tailwind PLAY

An advanced online playground for Tailwind CSS, including support for things like:

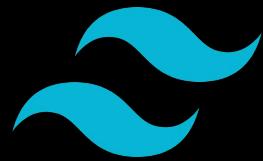
- Customizing your `tailwind.config.js` file
- Extracting classes with `@apply`
- Code completion with instant preview

Perfect for learning how the framework works, prototyping a new idea, or creating a demo to share online.

Want to dig deeper into Tailwind?

[Read the docs →](#)

# Including Tailwind CSS



## Installation

### Get started with Tailwind CSS

Tailwind CSS works by scanning all of your HTML files, JavaScript components, and any other templates for class names, generating the corresponding styles and then writing them to a static CSS file.

It's fast, flexible, and reliable — with zero-runtime.

## Installation

[Tailwind CLI](#) [Using PostCSS](#) [Framework Guides](#) [Play CDN](#)

Use the Play CDN to try Tailwind right in the browser without any build step. The Play CDN is designed for development purposes only, and is not the best choice for production.

#### 1 Add the Play CDN script to your HTML.

Add the Play CDN script tag to the `<head>` of your HTML file, and start using Tailwind's utility classes to style your content.

```
index.html
<!doctype html>
<html>
<head>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<script src="https://cdn.tailwindcss.com"></script>
</head>
<body>
<h1 class="text-3xl font-bold underline">
Hello world!
</h1>
</body>
</html>
```

# Installing Extension



 tailwindcss

INTELLISENSE

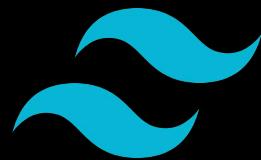
FOR



Visual Studio Code

```
class="bg-tl md:flex md:items-center md:  
v class="fl bg-teal-50  
<h2 class="t bg-teal-100  
    Back End D bg-teal-200  
</h2> bg-teal-300 background  
</div> bg-teal-400  
div class="mt bg-teal-500  
<span class= bg-teal-600  
    <button ty bg-teal-700  
        Edit bg-teal-800  
    </button> bg-teal-900  
</span> bg-transparent  
<span class= bg-top  
    <button type="button" class="inline-t  
        Publish  
    </button>  
</span>
```

# Installing Extension



## Tailwind CSS IntelliSense v0.12.10

Tailwind Labs [tailwindcss.com](https://tailwindcss.com) | 7,381,739 | (99)

Intelligent Tailwind CSS tooling for VS Code

[Install](#)



Auto Update



DETAILS

FEATURES

CHANGELOG

A screenshot of the Visual Studio Code interface. A code editor window shows a snippet of Tailwind CSS-based HTML. A tooltip or callout box is overlaid on the code, displaying a list of Tailwind utility classes such as "bg-transparent", "bg-teal-50", "bg-teal-100", "bg-teal-200", "bg-teal-300", "bg-teal-400", "bg-teal-500", "bg-teal-600", "bg-teal-700", "bg-teal-800", and "bg-teal-900". The background of the code editor shows a dark theme with some blurred UI elements. At the bottom of the screenshot, there are two logos: the Tailwind CSS logo and the Visual Studio Code logo.

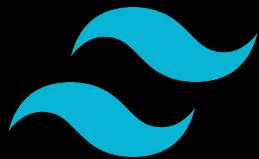
## Categories

Linters

## Resources

[Marketplace](#)  
[Issues](#)  
[Repository](#)  
[License](#)  
[Tailwind Labs](#)

# Installing Tailwind CSS



## 1. Install:

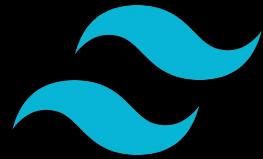
```
npm init -y
```

```
npm install -D tailwindcss postcss autoprefixer
```

## 2. Initialize Tailwind CSS Config

```
npx tailwindcss init
```

# Configure Content



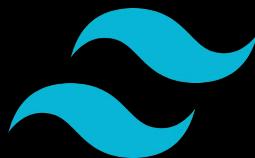
Configure Tailwind in the Configuration Files (`tailwind.config.js`)

`content: [ "*.html"]` // Add this line to scan for classes

The `content` section of your `tailwind.config.js` file is where you configure the paths to all of your HTML templates, JavaScript components, and any other source files that contain Tailwind class names.

`tailwind.config.js`

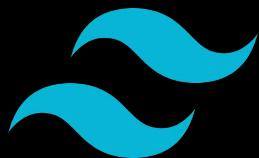
```
/** @type {import('tailwindcss').Config} */
module.exports = {
  content: [
    './pages/**/*.{html,js}',
    './components/**/*.{html,js}',
  ],
  // ...
}
```



# Using Directives

Add Tailwind Directives to src/input.css

```
/* src/index.css */
@tailwind base;
@tailwind components;
@tailwind utilities;
```



# Including Index CSS

5. Include `src/output.css` into your `index.html`

6. Run Command

```
npx tailwindcss -i ./src/input.css -o ./src/output.css --watch
```

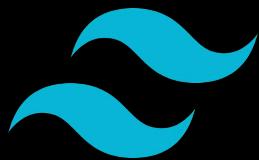
7. Declare Shortcut:

```
"scripts": {  
  "test": "echo \"Error: no test specified\" && exit 1",  
  "tailwind": "npx tailwindcss -i ./src/input.css -o ./src/output.css --watch"  
},
```

8. Run Command

```
npm run tailwind
```

# Pre-built Components



Creative Tim TW Components Components Resources Ecosystem PRO Blocks Discover Submit Login

A free repository for community components using [Tailwind CSS](#)

Open source Tailwind UI components and templates to bootstrap your new apps, projects or landing sites!

Search Components

Latest components

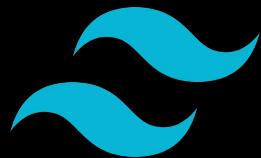
The newest featured Tailwind CSS components and templates from the community

Shadcn UI Navbar Component - Horizon AI  
Boilerplate vldmihalache

Tailwind CSS Buttons Variants Loopple @ 3.3

Tailwind CSS Icon Only Buttons Loopple @ 3.3

# No Predefined CSS



```
index.html # output.css 3
tailwind > src > index.html > ...
2  <html lang="en">
9   <body>
10  |  <h1 class="text-red-100">Tailwind Text Alignment</h1>
11  |</body>
12 </html>
```

```
← → ⌂ 127.0.0.1:5500/tailwind/src/index.html
Tailwind Text Alignment
```

```
index.html # output.css 3
tailwind > src > # output.css > .text-red-100
503  ::backdrop {
555  }
556
557  .text-red-100 {
558  --tw-text-opacity: 1;
559  color: rgb(254 226 226 / var(--tw-text-opacity));
560 }
```

# Pre-built Components

TailwindFlex

Search

Tags

Create Component

Sign In

Community-built

## Tailwind CSS Components Library

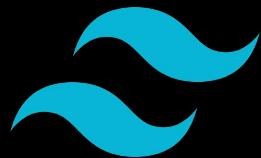
Design and Development tips  
in your inbox. Every weekday.

Let's not stress about website designs. Create Stunning UIs Effortlessly, with Over 1600+  
Ready to Use Components

Button, Footer, etc.

Find

# Testing Tailwind CSS



Sarah Smith

Freelance Web Designer



2k



10k



15

Follow

Follow

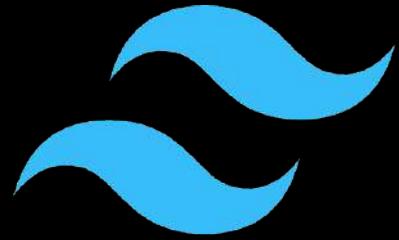
Message



12 Followers you know



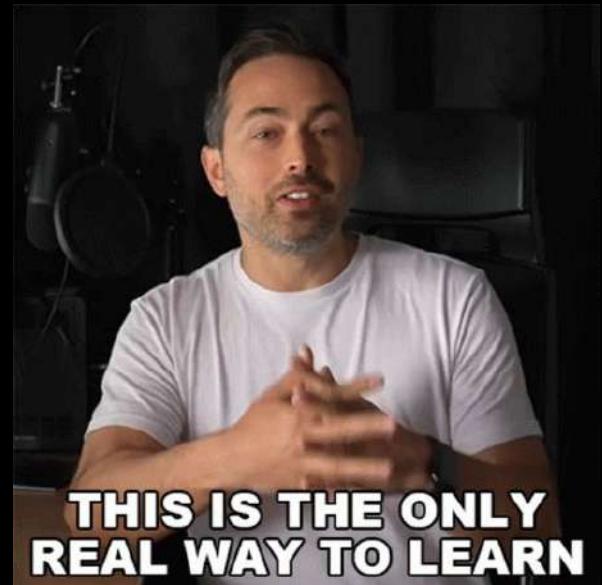


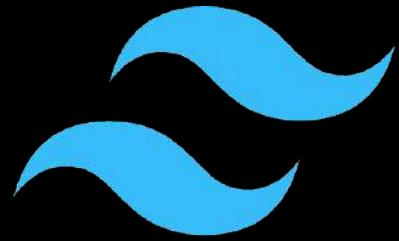


# tailwindcss

## LEARNING

- 1. Colors
- 2. States & Dark Mode
- 3. Responsive Design
- 4. Spacing
- 5. Typography
- 6. Sizing
- 7. Layouts
- 8. Flex
- 9. Grid
- 10. Border
- 11. Effects
- 12. Filters
- 13. Animations
- 14. Directives



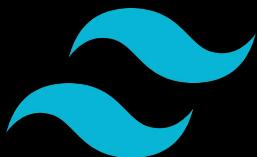


# tailwindcss

## LEARNING

### 1. Colors





# 1.1 Text-Color

## Setting the text color

Use the `text-\*` utilities to control the text color of an element.

The quick brown fox jumps over the lazy dog.

```
<p class="text-sky-400">The quick brown fox...</p>
```

```
<h1>Tailwind CSS Text Colors</h1>
<p class="text-blue-500">This text is blue.</p>
<p class="text-green-600">This text is green.</p>
<p class="text-red-700">This text is red.</p>
<p class="text-purple-800">This text is purple.</p>
<p class="text-gray-500">This text is gray.</p>
<p class="text-yellow-400">This text is yellow.</p>
```

## Tailwind CSS Text Colors

This text is blue.

This text is green.

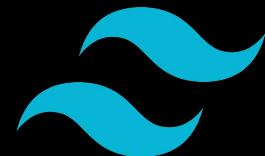
This text is red.

This text is purple.

This text is gray.

This text is yellow.

# 1.2 Background-Color



## Tailwind CSS Background Colors

This div has a blue background.

This div has a green background.

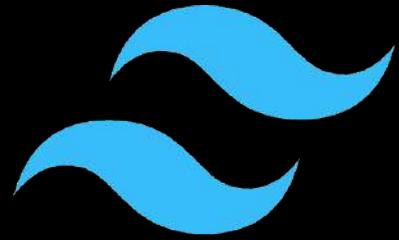
This div has a red background.

This div has a purple background.

This div has a gray background.

This div has a yellow background.

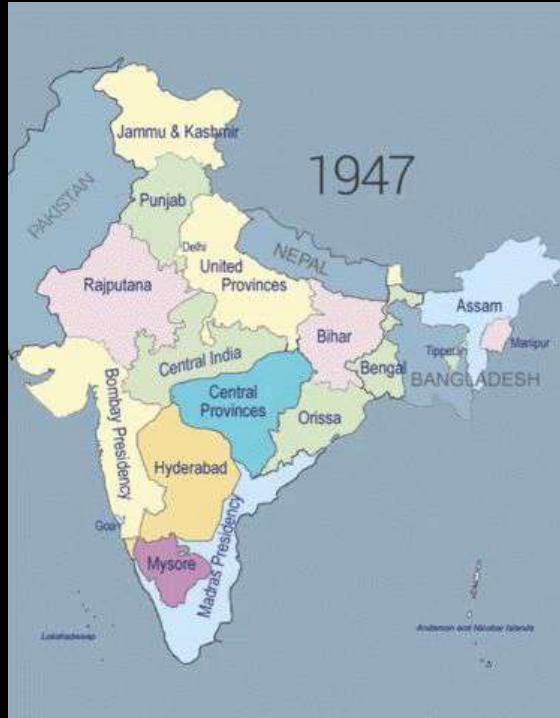
```
<h1>Tailwind CSS Background Colors</h1>
<div class="■bg-blue-500 ■text-white">This div has a blue background.</div>
<div class="■bg-green-600 ■text-white">This div has a green background.</div>
<div class="■bg-red-700 ■text-white">This div has a red background.</div>
<div class="■bg-purple-800 ■text-white">This div has a purple background.</div>
<div class="■bg-gray-500 ■text-white">This div has a gray background.</div>
<div class="■bg-yellow-400 □text-black">This div has a yellow background.</div>
```



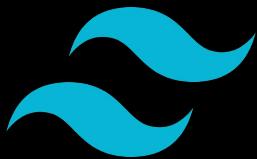
# tailwindcss

## LEARNING

### 2. States & Dark Mode



# 2.1 Hover



Save changes

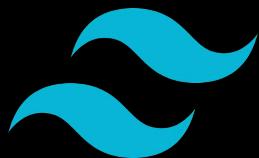
```
<button class="bg-sky-500 hover:bg-sky-700 ..." >  
  Save changes  
</button>
```

## ▼ How does this compare to traditional CSS?

When writing CSS the traditional way, a single class name would do different things based on the current state.

- ✖ Traditionally the same class name applies different styles on hover

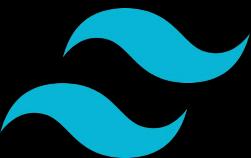
```
.btn-primary {  
  background-color: #0ea5e9;  
}  
.btn-primary:hover {  
  background-color: #0369a1;  
}
```



## 2.2 Pseudo-classes

Save changes

```
<button class="bg-violet-500 hover:bg-violet-600  
active:bg-violet-700 focus:outline-none focus:ring  
focus:ring-violet-300">  
  Save changes  
</button>
```



# 2.3 Dark Mode

Light mode



## Writes Upside-Down

The Zero Gravity Pen can be used to write in any orientation, including upside-down. It even works in outer space.

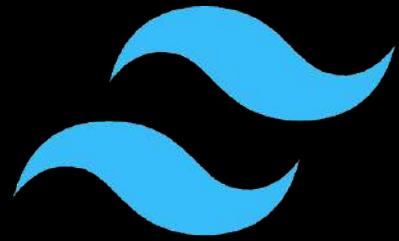
Dark mode



## Writes Upside-Down

The Zero Gravity Pen can be used to write in any orientation, including upside-down. It even works in outer space.

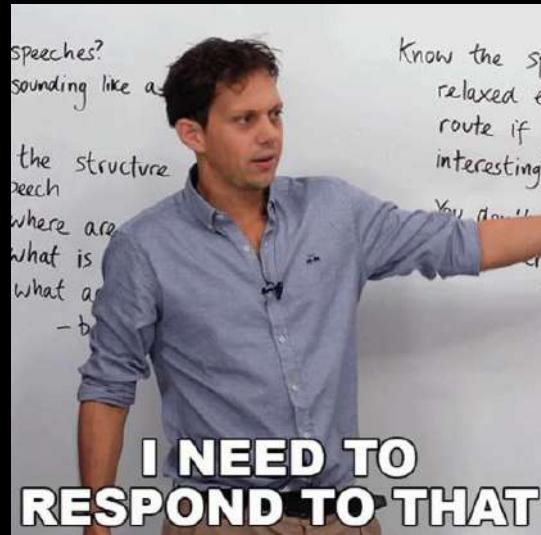
```
<div class="bg-white dark:bg-slate-800 rounded-lg px-6 py-8 ring-1 ring-slate-900/5 s
<div>
  <span class="inline-flex items-center justify-center p-2 bg-indigo-500 rounded-md
    <svg class="h-6 w-6 text-white" xmlns="http://www.w3.org/2000/svg" fill="none"
      </span>
</div>
<h3 class="text-slate-900 dark:text-white mt-5 text-base font-medium tracking-tight
<p class="text-slate-500 dark:text-slate-400 mt-2 text-sm">
  The Zero Gravity Pen can be used to write in any orientation, including upside-do
</p>
</div>
```



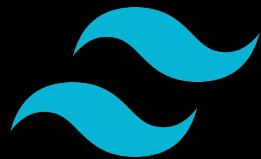
# tailwindcss

## LEARNING

### 3. Responsive Design



# 3 Responsive Design



First, make sure you've added the viewport meta tag to the `<head>` of your document:

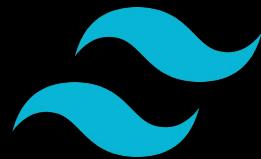
```
<meta name="viewport" content="width=device-width, initial-scale=1.0">
```

Then to add a utility but only have it take effect at a certain breakpoint, all you need to do is prefix the utility with the breakpoint name, followed by the `:` character:

```
<!-- Width of 16 by default, 32 on medium screens, and 48 on large screens -->  

```

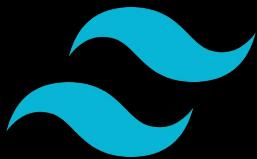
# 3 Responsive Design



There are five breakpoints by default, inspired by common device resolutions:

Breakpoint prefix	Minimum width	CSS
`sm`	640px	`@media (min-width: 640px) { ... }`
`md`	768px	`@media (min-width: 768px) { ... }`
`lg`	1024px	`@media (min-width: 1024px) { ... }`
`xl`	1280px	`@media (min-width: 1280px) { ... }`
`2xl`	1536px	`@media (min-width: 1536px) { ... }`

# 3 Responsive Design



## Targeting mobile screens

Where this approach surprises people most often is that to style something for mobile, you need to use the unprefixed version of a utility, not the `sm:` prefixed version. Don't think of `sm:` as meaning "on small screens", think of it as "at the small *breakpoint*".

- ✗ **Don't use `sm:` to target mobile devices**

```
<!-- This will only center text on screens 640px and wider, not on small screens -->

</div>


```

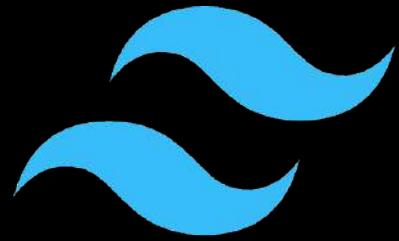
- ✓ **Use unprefixed utilities to target mobile, and override them at larger breakpoints**

```
<!-- This will center text on mobile, and left align it on screens 640px and wider -->

</div>


```

For this reason, it's often a good idea to implement the mobile layout for a design first, then layer on any changes that make sense for `sm` screens, followed by `md` screens, etc.



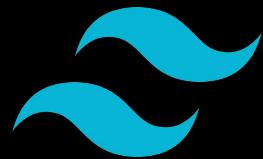
# tailwindcss

## LEARNING

### 4. Spacing



# 4.1 Margin



## Tailwind Margins

m-2: All sides

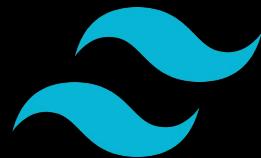
mx-4: Horizontal

my-3: Vertical

mt-4: Top only

```
<h1>Tailwind Margins</h1>
<div class="■bg-blue-200 m-2">m-2: All sides</div>
<div class="■bg-green-200 mx-4">mx-4: Horizontal</div>
<div class="■bg-red-200 my-3">my-3: Vertical</div>
<div class="■bg-purple-200 mt-4">mt-4: Top only</div>
```

# 4.2 Padding



## Tailwind Padding

p-2: All sides

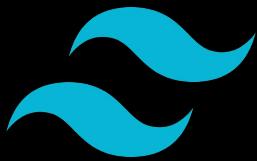
px-4: Horizontal

py-3: Vertical

pt-4: Top only

```
<h1>Tailwind Padding</h1>
<div class="■bg-blue-200 p-2 border ■border-blue-500">p-2: All sides</div>
<div class="■bg-green-200 px-4 border ■border-green-500">px-4: Horizontal</div>
<div class="■bg-red-200 py-3 border ■border-red-500">py-3: Vertical</div>
<div class="■bg-purple-200 pt-4 border ■border-purple-500">pt-4: Top only</div>
```

# 4.3 Spacing

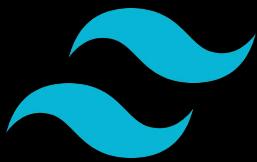


## Add horizontal space between children

Use the `space-x-\*` utilities to control the horizontal space between elements.



```
<div class="flex space-x-4 ...">  
  <div>01</div>  
  <div>02</div>  
  <div>03</div>  
</div>
```



# 4.3 Spacing

## Add vertical space between children

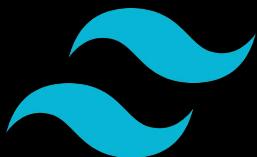
Use the `space-y-*` utilities to control the vertical space between elements.

01

02

03

```
<div class="flex flex-col space-y-4 ...>
  <div>01</div>
  <div>02</div>
  <div>03</div>
</div>
```



# 4.3 Spacing

```
<h1 class="m-2">Tailwind Space Utilities</h1>

<h2 class="m-2">Vertical Space (space-y-4)</h2>
<div class="space-y-4 m-2">
  <div class="bg-blue-200 p-2">First item</div>
  <div class="bg-blue-200 p-2">Second item</div>
  <div class="bg-blue-200 p-2">Third item</div>
</div>

<h2 class="m-2">Horizontal Space (space-x-4)</h2>
<div class="space-x-4 m-2">
  <span class="bg-green-200 p-2">First</span>
  <span class="bg-green-200 p-2">Second</span>
  <span class="bg-green-200 p-2">Third</span>
</div>
```

This example demonstrates two key space utilities in Tailwind CSS:

1. `space-y-4`: Adds vertical space (1rem or 16px by default) between child elements.
2. `space-x-4`: Adds horizontal space (1rem or 16px by default) between child elements.

## Tailwind Space Utilities

### Vertical Space (space-y-4)

First item

Second item

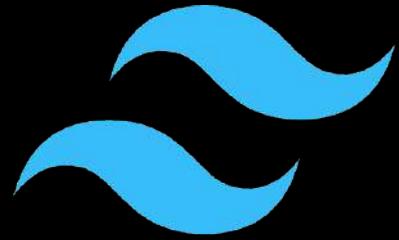
Third item

### Horizontal Space (space-x-4)

First

Second

Third



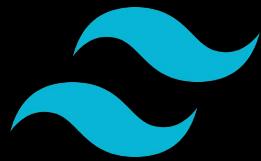
# tailwindcss

## LEARNING

### 5. *Typography*



# 5.1 Font-Family



font-sans

The quick brown fox jumps over the lazy dog.

font-serif

The quick brown fox jumps over the lazy dog.

font-mono

The quick brown fox jumps over the lazy dog.

```
<p class="font-sans ...">The quick brown fox ...</p>
<p class="font-serif ...">The quick brown fox ...</p>
<p class="font-mono ...">The quick brown fox ...</p>
```



# 5.2 Text-size

## Tailwind Font Sizes

text-xs: This is extra small text.

text-sm: This is small text.

text-base: This is the base text size.

text-lg: This is large text.

text-xl: This is extra large text.

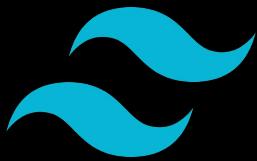
text-2xl: This is 2x large text.

text-3xl: This is 3x large text.

**text-4xl: This is 4x large text.**

```
<h1 class="text-2xl mb-4">Tailwind Font Sizes</h1>
<p class="text-xs">text-xs: This is extra small text.</p>
<p class="text-sm">text-sm: This is small text.</p>
<p class="text-base">text-base: This is the base text size.</p>
<p class="text-lg">text-lg: This is large text.</p>
<p class="text-xl">text-xl: This is extra large text.</p>
<p class="text-2xl">text-2xl: This is 2x large text.</p>
<p class="text-3xl">text-3xl: This is 3x large text.</p>
<p class="text-4xl">text-4xl: This is 4x large text.</p>
```

# 5.3 Font Style



The `italic` utility can be used to make text italic. Likewise, the `not-italic` utility can be used to display text normally — typically to reset italic text at different breakpoints.

## italic

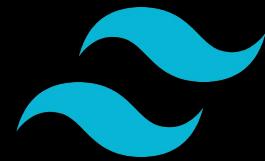
*The quick brown fox jumps over the lazy dog.*

## not-italic

The quick brown fox jumps over the lazy dog.

```
<p class="italic ...>The quick brown fox ...</p>
<p class="not-italic ...>The quick brown fox ...</p>
```

# 5.4 Font-Weight



## Tailwind Font Weights

`font-thin`: This is thin text (100)

`font-extralight`: This is extra light text (200)

`font-light`: This is light text (300)

`font-normal`: This is normal text (400)

`font-medium`: This is medium text (500)

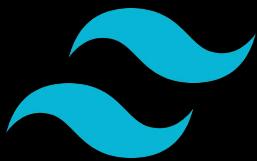
`font-semibold`: This is semibold text (600)

`font-bold`: This is bold text (700)

`font-extrabold`: This is extra bold text (800)

`font-black`: This is black text (900)

```
<h1 class="text-2xl font-bold mb-4">Tailwind Font Weights</h1>
<p class="font-thin">font-thin: This is thin text (100)</p>
<p class="font-extralight">font-extralight: This is extra light text (200)</p>
<p class="font-light">font-light: This is light text (300)</p>
<p class="font-normal">font-normal: This is normal text (400)</p>
<p class="font-medium">font-medium: This is medium text (500)</p>
<p class="font-semibold">font-semibold: This is semibold text (600)</p>
<p class="font-bold">font-bold: This is bold text (700)</p>
<p class="font-extrabold">font-extrabold: This is extra bold text (800)</p>
<p class="font-black">font-black: This is black text (900)</p>
```



# 5.5 Letter-Spacing

## Setting the letter spacing

Use the `tracking-\*` utilities to control the letter spacing of an element.

tracking-tight

The quick brown fox jumps over the lazy dog.

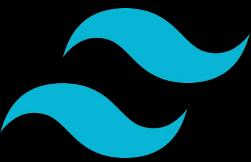
tracking-normal

The quick brown fox jumps over the lazy dog.

tracking-wide

The quick brown fox jumps over the lazy dog.

```
<p class="tracking-tight ...">The quick brown fox ...</p>
<p class="tracking-normal ...">The quick brown fox ...</p>
<p class="tracking-wide ...">The quick brown fox ...</p>
```



# 5.6 Line Height

Use the `'leading-none'`, `'leading-tight'`, `'leading-snug'`, `'leading-normal'`, `'leading-relaxed'`, and `'leading-loose'` utilities to give an element a relative line-height based on its current font-size.

## leading-normal

So I started to walk into the water. I won't lie to you boys, I was terrified. But I pressed on, and as I made my way past the breakers a strange calm came over me. I don't know if it was divine intervention or the kinship of all living things but I tell you Jerry at that moment, I was a marine biologist.

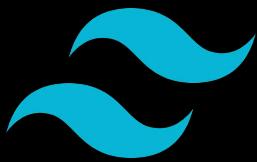
## leading-relaxed

So I started to walk into the water. I won't lie to you boys, I was terrified. But I pressed on, and as I made my way past the breakers a strange calm came over me. I don't know if it was divine intervention or the kinship of all living things but I tell you Jerry at that moment, I was a marine biologist.

## leading-loose

So I started to walk into the water. I won't lie to you boys, I was terrified. But I pressed on, and as I made my way past the breakers a strange calm came over me. I don't know if it was divine intervention or the kinship of all living things but I tell you Jerry at that moment, I was a marine biologist.

```
<p class="leading-normal ...">So I started to walk into the water...</p>
<p class="leading-relaxed ...">So I started to walk into the water...</p>
<p class="leading-loose ...">So I started to walk into the water...</p>
```



# 5.7 List Style

To create bulleted or numeric lists, use the `list-disc` and `list-decimal` utilities.

## list-disc

- Now this is a story all about how, my life got flipped-turned upside down
- And I'd like to take a minute just sit right there
- I'll tell you how I became the prince of a town called Bel-Air

## list-decimal

1. Now this is a story all about how, my life got flipped-turned upside down
2. And I'd like to take a minute just sit right there
3. I'll tell you how I became the prince of a town called Bel-Air

## list-none

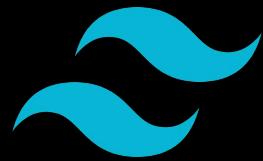
Now this is a story all about how, my life got flipped-turned upside down  
And I'd like to take a minute just sit right there  
I'll tell you how I became the prince of a town called Bel-Air

```
<ul class="list-disc">
  <li>Now this is a story all about how, my life got flipped-turned upside down</li>
  <!-- ... -->
</ul>

<ol class="list-decimal">
  <li>Now this is a story all about how, my life got flipped-turned upside down</li>
  <!-- ... -->
</ol>

<ul class="list-none">
  <li>Now this is a story all about how, my life got flipped-turned upside down</li>
  <!-- ... -->
</ul>
```

# 5.8 Text-Alignment



## Tailwind Text Alignment

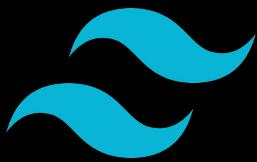
text-left: This paragraph is aligned to the left.

text-center: This paragraph is centered.

text-right: This paragraph is aligned to the right.

text-justify: This paragraph is justified. It contains a longer text to demonstrate the effect of justification. Notice how the text is spread out to align with both the left and right edges.

```
<h1 class="text-2xl font-bold mb-4">Tailwind Text Alignment</h1>
<div class="space-y-4">
  <p class="text-left bg-blue-100 p-2">
    text-left: This paragraph is aligned to the left.
  </p>
  <p class="text-center bg-green-100 p-2">
    text-center: This paragraph is centered.
  </p>
  <p class="text-right bg-red-100 p-2">
    text-right: This paragraph is aligned to the right.
  </p>
  <p class="text-justify bg-yellow-100 p-2">
    text-justify: This paragraph is justified. It contains a longer text to demonstrate the effect of
    justification. Notice how the text is spread out to align with both the left and right edges.
  </p>
</div>
```



# 5.9 Text Decoration

Control how text is decorated with the `'underline'`, `'no-underline'`, and `'line-through'` utilities.

`underline`

The quick brown fox jumps over the lazy dog.

`overline`

The quick brown fox jumps over the lazy dog.

`line-through`

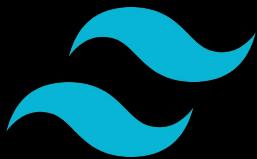
~~The quick brown fox jumps over the lazy dog.~~

`no-underline`

The quick brown fox jumps over the lazy dog.

```
<p class="underline ...">The quick brown fox ...</p>
<p class="overline ...">The quick brown fox ...</p>
<p class="line-through ...">The quick brown fox ...</p>
<p class="no-underline ...">The quick brown fox ...</p>
```

# 5.10 Text Decoration Color



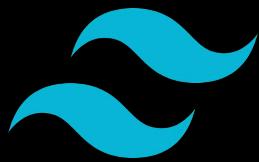
## Setting the text decoration color

Use the `decoration-\*` utilities to change the color of an element's text decoration.

I'm Derek, an astro-engineer based in  
Tattooine. I like to build X-Wings at My  
Company, Inc.. Outside of work, I like to watch  
pod-racing and have light-saber fights.

```
<div>
  <p>
    I'm Derek, an astro-engineer based in Tattooine. I like to build X-Wings at
    <a class="underline decoration-sky-500">My Company, Inc</a>.
    Outside of work, I like to <a class="underline decoration-pink-500">watch
    pod-racing</a> and have <a class="underline decoration-indigo-500">light-saber</a>
  </p>
</div>
```

# 5.11 Text Decoration Style



Use the `decoration-\*` utilities to change the style of an element's text decoration.

decoration-solid

The quick brown fox jumps over the lazy dog.

decoration-double

The quick brown fox jumps over the lazy dog.

decoration-dotted

.The quick brown fox jumps over the lazy dog.

decoration-dashed

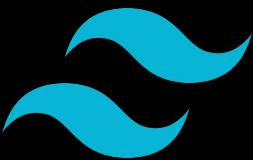
The quick brown fox jumps over the lazy dog.

decoration-wavy

The quick brown fox jumps over the lazy dog.

```
<p class="underline decoration-solid ...">The quick brown fox...</p>
<p class="underline decoration-double ...">The quick brown fox...</p>
<p class="underline decoration-dotted ...">The quick brown fox...</p>
<p class="underline decoration-dashed ...">The quick brown fox...</p>
<p class="underline decoration-wavy ...">The quick brown fox...</p>
```

# 5.12 Text Transform



normal-case

The quick brown fox jumps over the lazy dog.

uppercase

THE QUICK BROWN FOX JUMPS OVER THE LAZY DOG.

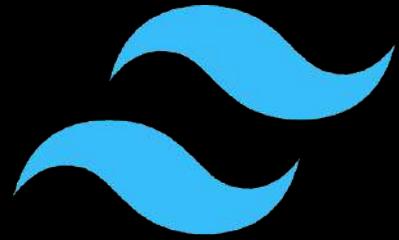
lowercase

the quick brown fox jumps over the lazy dog.

capitalize

The Quick Brown Fox Jumps Over The Lazy Dog.

```
<p class="normal-case ...">The quick brown fox ...</p>
<p class="uppercase ...">The quick brown fox ...</p>
<p class="lowercase ...">The quick brown fox ...</p>
<p class="capitalize ...">The quick brown fox ...</p>
```



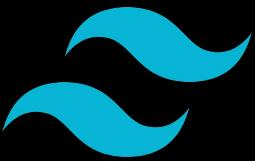
# tailwindcss

## LEARNING

### 6. *Sizing*



# 6.1 Width

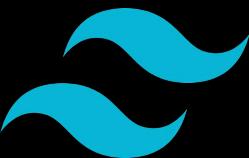


## Fixed widths

Use utilities like `w-px`, `w-1`, and `w-64` to set an element to a fixed width.



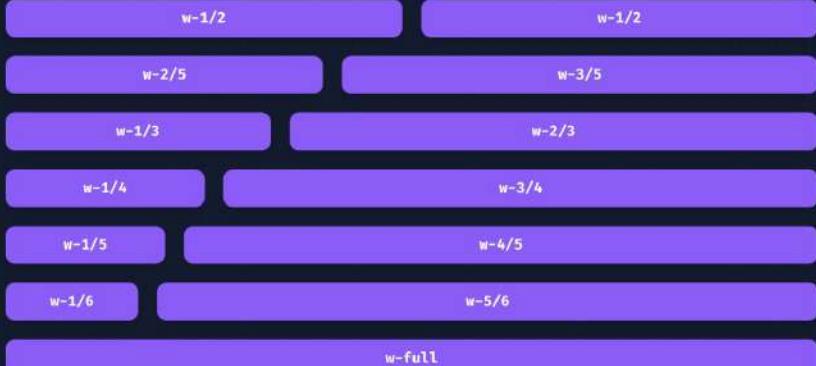
```
<div class="w-96 ...>w-96</div>
<div class="w-80 ...>w-80</div>
<div class="w-64 ...>w-64</div>
<div class="w-48 ...>w-48</div>
<div class="w-40 ...>w-40</div>
<div class="w-32 ...>w-32</div>
<div class="w-24 ...>w-24</div>
```



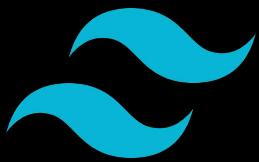
# 6.1 Width

## Percentage widths

Use utilities like `w-full`, `w-1/2`, and `w-2/5` to set an element to a percentage based width.

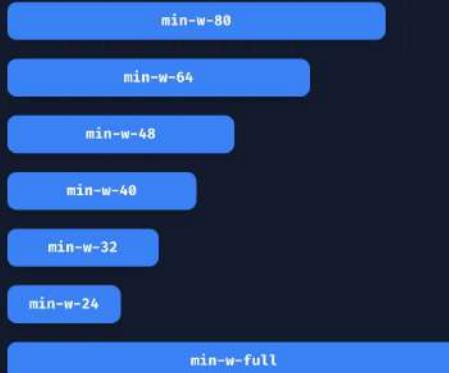


```
<div class="flex ...">
  <div class="w-1/2 ... ">w-1/2</div>
  <div class="w-1/2 ... ">w-1/2</div>
</div>
<div class="flex ...">
  <div class="w-2/5 ... ">w-2/5</div>
  <div class="w-3/5 ... ">w-3/5</div>
</div>
<div class="flex ...">
  <div class="w-1/3 ... ">w-1/3</div>
  <div class="w-2/3 ... ">w-2/3</div>
</div>
<div class="flex ...">
  <div class="w-1/4 ... ">w-1/4</div>
  <div class="w-3/4 ... ">w-3/4</div>
</div>
<div class="flex ...">
  <div class="w-1/5 ... ">w-1/5</div>
  <div class="w-4/5 ... ">w-4/5</div>
</div>
<div class="flex ...">
  <div class="w-1/6 ... ">w-1/6</div>
  <div class="w-full ... ">w-full</div>
</div>
```



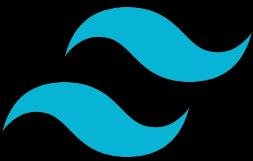
# 6.2 Min-Width

Set the minimum width of an element using `min-w-\*` utilities.



```
<div class="w-96 ...>  
  <div class="min-w-80 ...>min-w-80</div>  
  <div class="min-w-64 ...>min-w-64</div>  
  <div class="min-w-48 ...>min-w-48</div>  
  <div class="min-w-40 ...>min-w-40</div>  
  <div class="min-w-32 ...>min-w-32</div>  
  <div class="min-w-24 ...>min-w-24</div>  
  <div class="min-w-full ...>min-w-full</div>  
</div>
```

# 6.3 Max-Width



Set the maximum width of an element using the `max-w-\*` utilities.

max-w-96

max-w-80

max-w-64

max-w-48

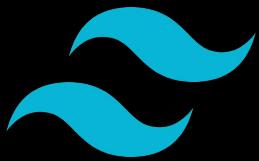
max-w-40

max-w-32

max-w-24

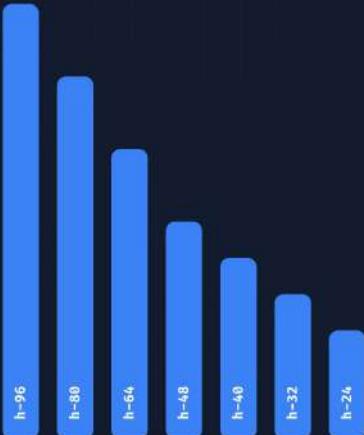
```
<div>
  <div class="w-full max-w-96 ...">max-w-96</div>
  <div class="w-full max-w-80 ...">max-w-80</div>
  <div class="w-full max-w-64 ...">max-w-64</div>
  <div class="w-full max-w-48 ...">max-w-48</div>
  <div class="w-full max-w-40 ...">max-w-40</div>
  <div class="w-full max-w-32 ...">max-w-32</div>
  <div class="w-full max-w-24 ...">max-w-24</div>
</div>
```

# 6.4 Height



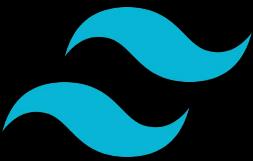
## Fixed heights

Use utilities like `h-px`, `h-1`, and `h-64` to set an element to a fixed height.



```
<div class="h-96 ...">h-96</div>
<div class="h-80 ...">h-80</div>
<div class="h-64 ...">h-64</div>
<div class="h-48 ...">h-48</div>
<div class="h-40 ...">h-40</div>
<div class="h-32 ...">h-32</div>
<div class="h-24 ...">h-24</div>
```

# 6.4 Height



## Full height

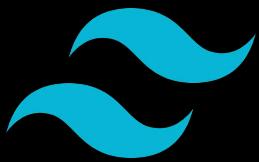
Use `h-full` to set an element's height to 100% of its parent, as long as the parent has a defined height.

```
<div class="h-48">  
  <div class="h-full ..." >  
    <!-- This element will have a height of `12rem` (h-48) --&gt;<br/>  </div>  
</div>
```

## Viewport height

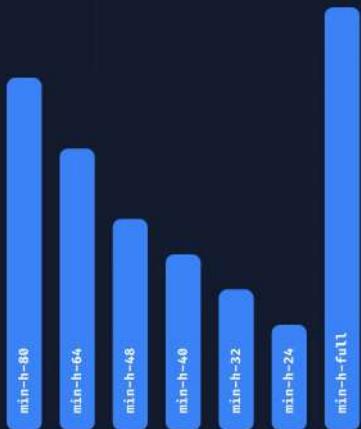
Use `h-screen` to make an element span the entire height of the viewport.

```
<div class="h-screen">  
  <!-- ... -->  
</div>
```

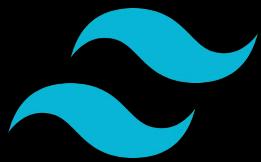


# 6.5 Min-Height

Set the minimum height of an element using `'min-h-*'` utilities.

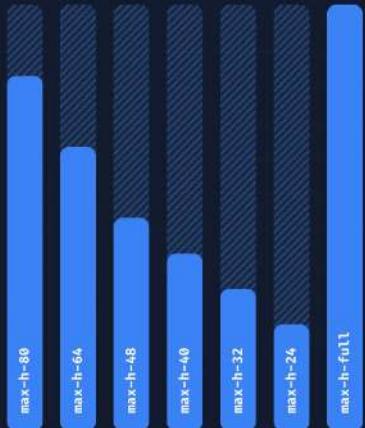


```
<div class="h-96 ...">  
  <div class="min-h-80 ...">min-h-80</div>  
  <div class="min-h-64 ...">min-h-64</div>  
  <div class="min-h-48 ...">min-h-48</div>  
  <div class="min-h-40 ...">min-h-40</div>  
  <div class="min-h-32 ...">min-h-32</div>  
  <div class="min-h-24 ...">min-h-24</div>  
  <div class="min-h-full ...">min-h-full</div>  
</div>
```



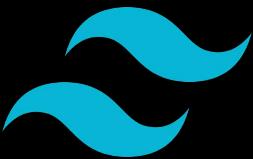
# 6.6 Max-Height

Set the maximum height of an element using `max-h-\*` utilities.



```
<div class="h-96 ...">  
<div class="h-full max-h-80 ..." >max-h-80</div>  
<div class="h-full max-h-64 ..." >max-h-64</div>  
<div class="h-full max-h-48 ..." >max-h-48</div>  
<div class="h-full max-h-40 ..." >max-h-40</div>  
<div class="h-full max-h-32 ..." >max-h-32</div>  
<div class="h-full max-h-24 ..." >max-h-24</div>  
<div class="h-full max-h-full ..." >max-h-full</div>  
</div>
```

# 6.7 Size



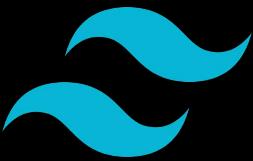
## Fixed sizes

Use utilities like `size-px`, `size-1`, and `size-64` to set an element to a fixed width and height at the same time.



```
<div class="size-16 ...">size-16</div>
<div class="size-20 ...">size-20</div>
<div class="size-24 ...">size-24</div>
<div class="size-32 ...">size-32</div>
<div class="size-40 ...">size-40</div>
```

# 6.7 Size



## Percentage sizes

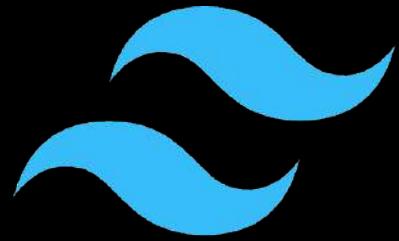
Use `size-full` to set an element's width and height to be 100% of the parent container's width and height.



size-full

A large, solid purple rectangular box with a thin black border, centered on a dark background. The text "size-full" is centered within this purple box.

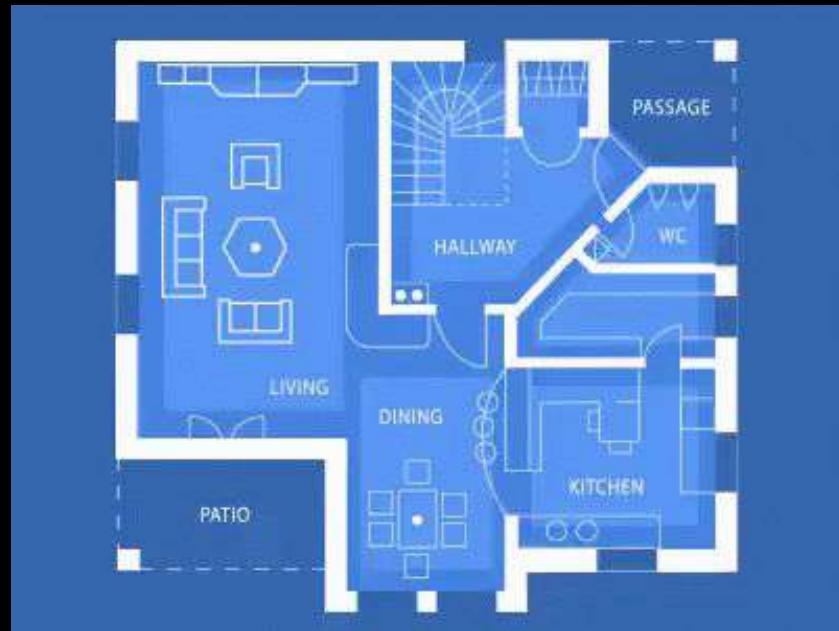
```
<div class="h-56 p-2 ...">  
  <div class="size-full ...">size-full</div>  
</div>
```

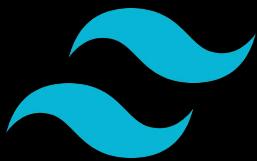


# tailwindcss

## LEARNING

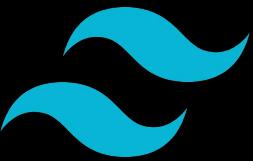
### 7. *Layouts*





# 7.1 Container

Class	Breakpoint	Properties
<b>container</b>	<i>None</i>	width: 100%;
	<i>sm (640px)</i>	max-width: 640px;
	<i>md (768px)</i>	max-width: 768px;
	<i>lg (1024px)</i>	max-width: 1024px;
	<i>xl (1280px)</i>	max-width: 1280px;
	<i>2xl (1536px)</i>	max-width: 1536px;



# 7.1 Container

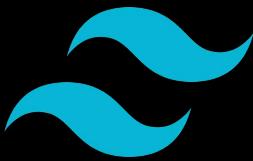
To center a container, use the `mx-auto` utility:

```
<div class="container mx-auto">  
  <!-- ... -->  
</div>
```

To add horizontal padding, use the `px-\*` utilities:

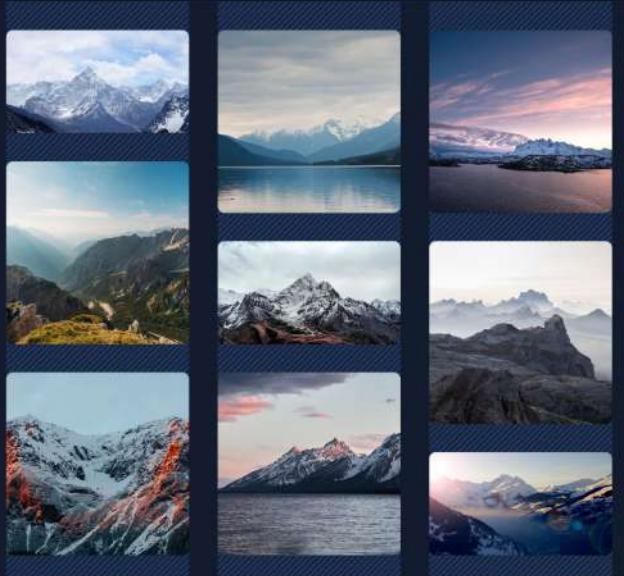
```
<div class="container mx-auto px-4">  
  <!-- ... -->  
</div>
```

# 7.2 Columns



## Adding based on column count

Use utilities like `'columns-2'` and `'columns-3'` to set the number of columns that should be created for the content within an element. The column width will be automatically adjusted to accommodate that number.



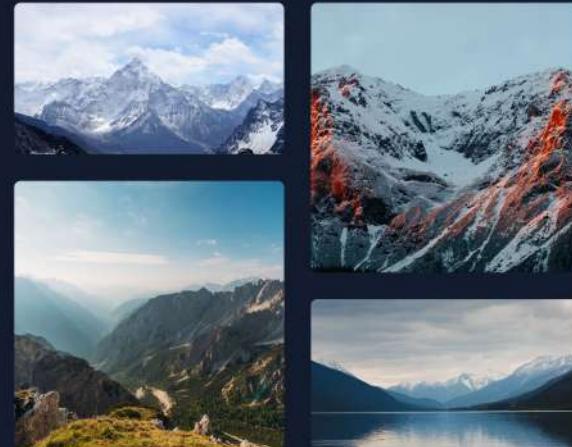
```
<div class="columns-3 ...">
  
  
  <!-- ... -->
</div>
```

## Adding based on column width

Use utilities like `'columns-xs'` and `'columns-sm'` to set the ideal column width for the content within an element, with the number of columns (the count) automatically adjusting to accommodate that value.

This "t-shirt" scale is the same as the `max-width` scale, with the addition of `'2xs'` and `'3xs'`, since smaller columns may be desirable.

👉 Resize the example to see the expected behaviour

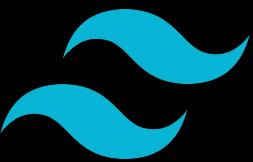


```
<div class="columns-2 ...">
  
  
  <!-- ... -->
</div>
```

To specify the width between columns, you can use the `gap-x` utilities:



```
div class="gap-8 columns-3 ...">
  
  
  <!-- ... -->
</div>
```



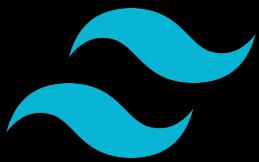
# 7.2 Columns

## Adding based on column count

Use utilities like `'columns-2'` and `'columns-3'` to set the number of columns that should be created for the content within an element. The column width will be automatically adjusted to accommodate that number.

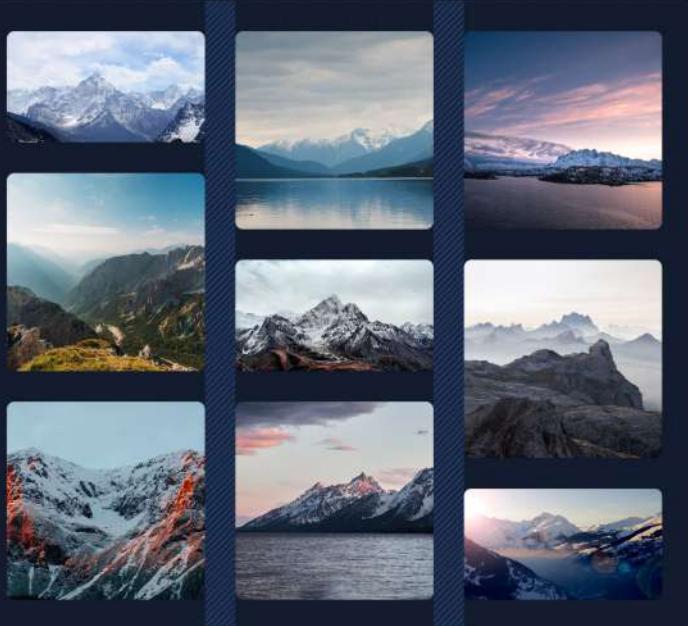


```
<div class="columns-3 ...>
  
  
  <!-- ... -->
</div>
```



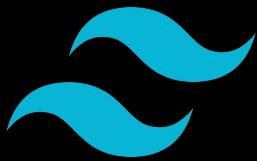
# 7.2 Columns

To specify the width between columns, you can use the `gap-x` utilities:



```
<div class="gap-8 columns-3 ...">
  
  
  <!-- ... -->
</div>
```

# 7.3 Box Sizing



## Class

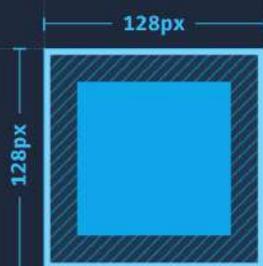
box-border

box-content

## Properties

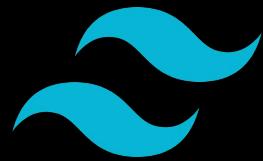
box-sizing: border-box;

box-sizing: content-box;



```
<div class="box-border h-32 w-32 p-4 border-4 ...>  
  <!-- ... -->  
</div>
```

# 7.4 Display



Use the `'inline'`, `'inline-block'`, and `'block'` utilities to control the flow of text and elements.

When controlling the flow of text, using the CSS property `display: inline` will cause the text inside the element to wrap normally.

While using the property `display: inline-block` will wrap the element to prevent the text inside from extending beyond its parent.

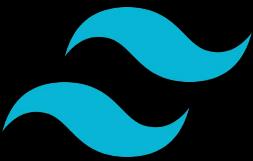
Lastly, using the property `display: block` will put the element on its own line and fill its parent.

```
<div>  
  When controlling the flow of text, using the CSS property  
  <span class="inline">display: inline</span>  
  will cause the text inside the element to wrap normally.
```

```
  While using the property <span class="inline-block">display: inline-block</span>  
  will wrap the element to prevent the text inside from extending beyond its parent.
```

```
  Lastly, using the property <span class="block">display: block</span>  
  will put the element on its own line and fill its parent.  
</div>
```

# 7.5 Float



## Basic usage

### Floating elements to the right

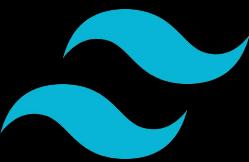
Use the `float-right` utility to float an element to the right of its container.

Maybe we can live without libraries, people like you and me. Maybe. Sure, we're too old to change the world, but what about that kid, sitting down, opening a book, right now, in a branch at the local library and finding drawings of pee-pees and wee-wees on the Cat in the Hat and the Five Chinese Brothers? Doesn't HE deserve better? Look. If you think this is about overdue fines and missing books, you'd better think again. This is about that kid's right to read a book without getting his mind warped! Or: maybe that turns you on, Seinfeld; maybe that's how y'get your kicks. You and your good-time buddies.



```

<p>Maybe we can live without libraries, people like you and me. ...</p>
```



# 7.6 Position

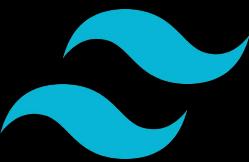
Use the ``static`` utility to position an element according to the normal flow of the document.

Any offsets will be ignored and the element will not act as a position reference for absolutely positioned children.

Static parent

Absolute child

```
<div class="static ...">
  <p>Static parent</p>
  <div class="absolute bottom-0 left-0 ...">
    <p>Absolute child</p>
  </div>
</div>
```



# 7.6 Position

With static positioning

Relative parent

Static parent

Static child

Static sibling

With absolute positioning

Relative parent

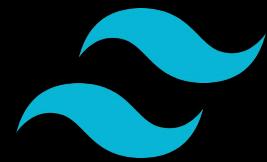
Absolute child

Static parent

Static sibling

```
<div class="static ...">
  <!-- Static parent -->
  <div class="static ..."><p>Static child</p></div>
  <div class="inline-block ..."><p>Static sibling</p></div>
  <!-- Static parent -->
  <div class="absolute ..."><p>Absolute child</p></div>
  <div class="inline-block ..."><p>Static sibling</p></div>
</div>
```

# 7.7 Top/Right/Bottom/Left



```
<!-- Pin to top left corner -->
<div class="relative h-32 w-32 ...">
  <div class="absolute left-0 top-0 h-16 w-16 ...>01</div>
</div>

<!-- Span top edge -->
<div class="relative h-32 w-32 ...">
  <div class="absolute inset-x-0 top-0 h-16 w-16 ...>02</div>
</div>

<!-- Pin to top right corner -->
<div class="relative h-32 w-32 ...">
  <div class="absolute top-0 right-0 h-16 w-16 ...>03</div>
</div>

<!-- Span left edge -->
<div class="relative h-32 w-32 ...">
  <div class="absolute inset-y-0 left-0 w-16 h-16 ...>04</div>
</div>

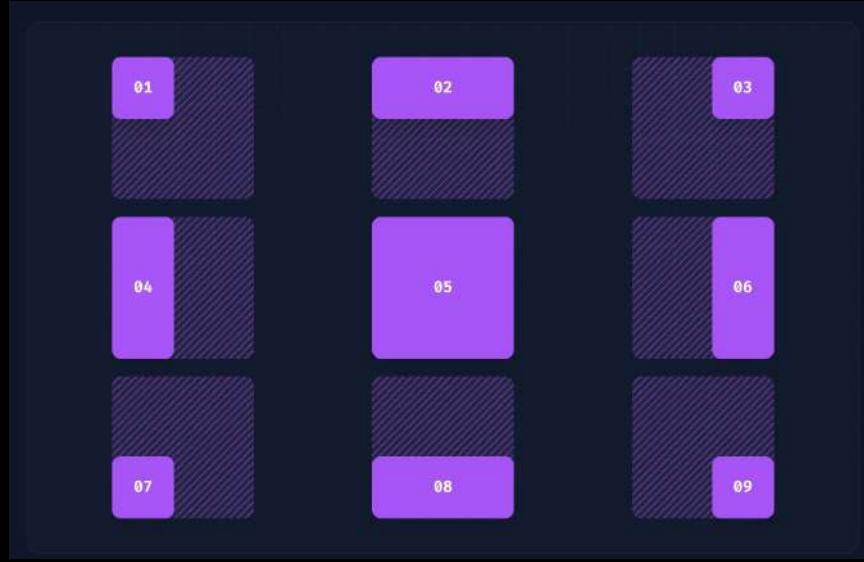
<!-- Fill entire parent -->
<div class="relative h-32 w-32 ...">
  <div class="absolute inset-0 ...>05</div>
</div>

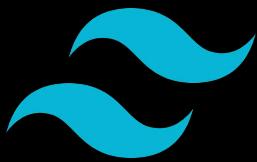
<!-- Span right edge -->
<div class="relative h-32 w-32 ...">
  <div class="absolute inset-y-0 right-0 w-16 h-16 ...>06</div>
</div>

<!-- Pin to bottom left corner -->
<div class="relative h-32 w-32 ...">
  <div class="absolute bottom-0 left-0 h-16 w-16 ...>07</div>
</div>

<!-- Span bottom edge -->
<div class="relative h-32 w-32 ...">
  <div class="absolute inset-x-0 bottom-0 h-16 w-16 ...>08</div>
</div>

<!-- Pin to bottom right corner -->
<div class="relative h-32 w-32 ...">
  <div class="absolute bottom-0 right-0 h-16 w-16 ...>09</div>
</div>
```





# 7.8 Overflow

Use the `overflow-visible` utility to prevent content within an element from being clipped. Note that any content that overflows the bounds of the element will then be visible.



Andrew Alfred  
Technical advisor

```
<div class="overflow-visible ..."></div>
```

## Hiding content that overflows

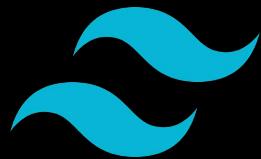
Use the `overflow-hidden` utility to clip any content within an element that overflows the bounds of that element.



Andrew Alfred  
Technical advisor

```
<div class="overflow-hidden ..."></div>
```

# 7.9 Visibility

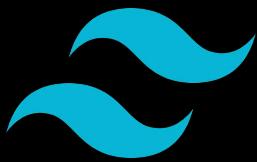


Use the `invisible` utility to hide an element, but still maintain its place in the DOM, affecting the layout of other elements (compare with `hidden` from the [display](#) documentation).

01

03

```
<div class="grid grid-cols-3 gap-4">
  <div>01</div>
  <div class="invisible ...">02</div>
  <div>03</div>
</div>
```



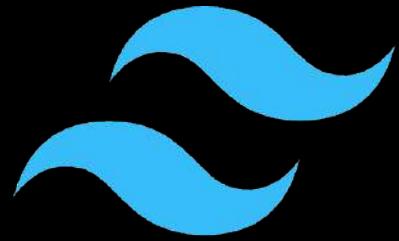
# 7.10 Z-Index

## Setting the z-index

Use the `z-\*` utilities to control the stack order (or three-dimensional positioning) of an element, regardless of order it has been displayed.



```
<div class="z-40 ...>05</div>
<div class="z-30 ...>04</div>
<div class="z-20 ...>03</div>
<div class="z-10 ...>02</div>
<div class="z-0 ...>01</div>
```



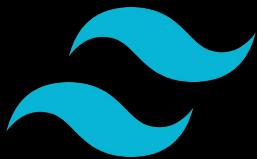
# tailwindcss

## LEARNING

### 8. Flex



# 8.1 Flex



## Setting the flex basis

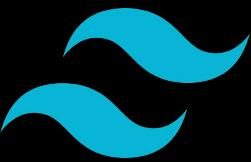
Use the `basis-\*` utilities to set the initial size of flex items.

01

02

03

```
<div class="flex flex-row">
  <div class="basis-1/4">01</div>
  <div class="basis-1/4">02</div>
  <div class="basis-1/2">03</div>
</div>
```



# 8.2 Flex-Direction

## Class

## Properties

`flex-row`

`flex-direction: row;`

`flex-row-reverse`

`flex-direction: row-reverse;`

`flex-col`

`flex-direction: column;`

`flex-col-reverse`

`flex-direction: column-reverse;`

## Basic usage

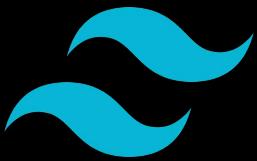
### Row

Use `'flex-row'` to position flex items horizontally in the same direction as text:



```
<div class="flex flex-row ...">
  <div>01</div>
  <div>02</div>
  <div>03</div>
</div>
```

# 8.3 Flex-Wrap



## Class

flex-wrap

flex-wrap-reverse

flexnowrap

## Properties

flex-wrap: wrap;

flex-wrap: wrap-reverse;

flex-wrap: nowrap;

## Wrap normally

Use `flex-wrap` to allow flex items to wrap:

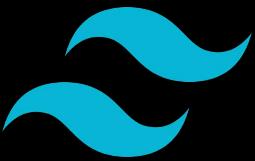
01

02

03

```
<div class="flex flex-wrap">
  <div>01</div>
  <div>02</div>
  <div>03</div>
</div>
```

# 8.4 Flex-Grow



## Class

## Properties

grow

flex-grow: 1;

grow-0

flex-grow: 0;

## Basic usage

### Grow

Use `grow` to allow a flex item to grow to fill any available space:

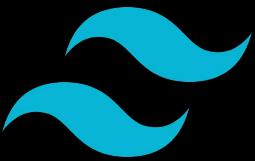
01

02

03

```
<div class="flex ...>
  <div class="flex-none w-14 h-14 ...">
    01
  </div>
  <div class="grow h-14 ...">
    02
  </div>
  <div class="flex-none w-14 h-14 ...">
    03
  </div>
</div>
```

# 8.5 Flex-Shrink



## Class

## Properties

shrink	flex-shrink: 1;
shrink-0	flex-shrink: 0;

## Basic usage

### Shrink

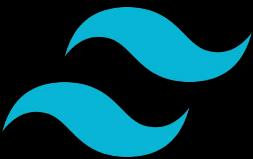
Use `shrink` to allow a flex item to shrink if needed:

01            02            03

A visual representation of three items labeled 01, 02, and 03 arranged horizontally. Item 02 is significantly smaller than items 01 and 03, demonstrating its ability to shrink within a flex container.

```
<div class="flex ...">
  <div class="flex-none w-14 h-14 ...">
    01
  </div>
  <div class="shrink w-64 h-14 ...">
    02
  </div>
  <div class="flex-none w-14 h-14 ...">
    03
  </div>
</div>
```

# 8.6 Flex

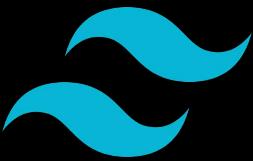


## Flex

Utilities for controlling how flex items both grow and shrink.

Class	Properties
flex-1	flex: 1 1 0%;
flex-auto	flex: 1 1 auto;
flex-initial	flex: 0 1 auto;
flex-none	flex: none;

# 8.6 Flex

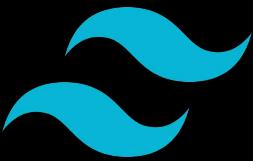


Use `flex-initial` to allow a flex item to shrink but not grow, taking into account its initial size:



```
<div class="flex">
  <div class="flex-none w-14 ...">
    01
  </div>
  <div class="flex-initial w-64 ...">
    02
  </div>
  <div class="flex-initial w-32 ...">
    03
  </div>
</div>
```

# 8.6 Flex



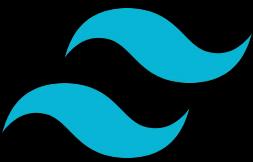
## Flex 1

Use `flex-1` to allow a flex item to grow and shrink as needed, ignoring its initial size:



```
<div class="flex">
  <div class="flex-none w-14 ...">
    01
  </div>
  <div class="flex-1 w-64 ...">
    02
  </div>
  <div class="flex-1 w-32 ...">
    03
  </div>
</div>
```

# 8.6 Flex



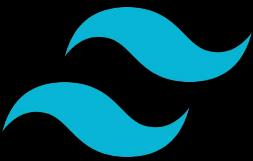
## Auto

Use `flex-auto` to allow a flex item to grow and shrink, taking into account its initial size:

A horizontal row of three purple rectangular boxes. The first box contains the number '01'. The second box contains '02'. The third box contains '03'. They are arranged side-by-side in a horizontal flex container, with thin vertical lines separating them.

```
<div class="flex ...>
  <div class="flex-none w-14 ...>
    01
  </div>
  <div class="flex-auto w-64 ...>
    02
  </div>
  <div class="flex-auto w-32 ...>
    03
  </div>
</div>
```

# 8.6 Flex



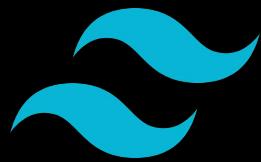
## None

Use `flex-none` to prevent a flex item from growing or shrinking:



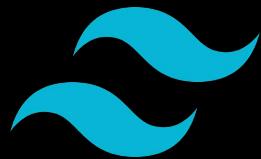
```
<div class="flex ...">
  <div class="flex-none w-14 ...">
    01
  </div>
  <div class="flex-none w-32 ...">
    02
  </div>
  <div class="flex-1 ...">
    03
  </div>
</div>
```

# 8.7 Justify-Content



Class	Properties
justify-normal	justify-content: normal;
justify-start	justify-content: flex-start;
justify-end	justify-content: flex-end;
justify-center	justify-content: center;
justify-between	justify-content: space-between;
justify-around	justify-content: space-around;
justify-evenly	justify-content: space-evenly;
justify-stretch	justify-content: stretch;

# 8.7 Justify-Content



## Start

Use `justify-start` to justify items against the start of the container's main axis:



```
<div class="flex justify-start ...">
  <div>01</div>
  <div>02</div>
  <div>03</div>
</div>
```

## Center

Use `justify-center` to justify items along the center of the container's main axis:



```
<div class="flex justify-center ...">
  <div>01</div>
  <div>02</div>
  <div>03</div>
</div>
```

## End

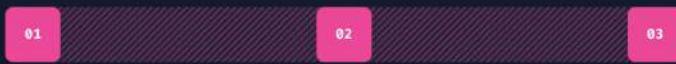
Use `justify-end` to justify items against the end of the container's main axis:



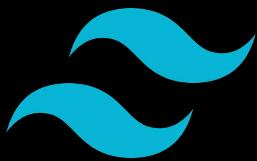
```
<div class="flex justify-end ...">
  <div>01</div>
  <div>02</div>
  <div>03</div>
</div>
```

## Space between

Use `justify-between` to justify items along the container's main axis such that there is an equal amount of space between each item:



```
<div class="flex justify-between ...">
  <div>01</div>
  <div>02</div>
  <div>03</div>
</div>
```



# 8.8 Align-Items

## Class

## Properties

`items-start`

`align-items: flex-start;`

`items-end`

`align-items: flex-end;`

`items-center`

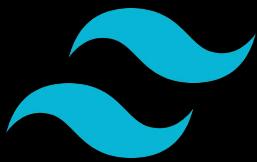
`align-items: center;`

`items-baseline`

`align-items: baseline;`

`items-stretch`

`align-items: stretch;`



# 8.8 Align-Items

## Stretch

Use `'items-stretch'` to stretch items to fill the container's cross axis:

Three teal rounded rectangular boxes are arranged horizontally. Each box contains a white number: '01' in the first, '02' in the second, and '03' in the third. The boxes are separated by thin black vertical borders. The background behind the boxes is dark gray.

01      02      03

```
<div class="flex items-stretch ...">
  <div class="py-4">01</div>
  <div class="py-12">02</div>
  <div class="py-8">03</div>
</div>
```

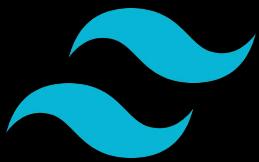
## Start

Use `'items-start'` to align items to the start of the container's cross axis:

Three pink rounded rectangular boxes are arranged horizontally. Each box contains a white number: '01' in the first, '02' in the second, and '03' in the third. The boxes are separated by thin black vertical borders. The background behind the boxes is dark gray.

01      02      03

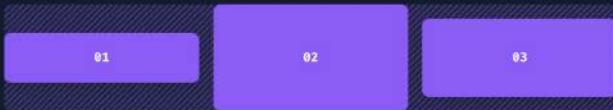
```
<div class="flex items-start ...">
  <div class="py-4">01</div>
  <div class="py-12">02</div>
  <div class="py-8">03</div>
</div>
```



# 8.8 Align-Items

## Center

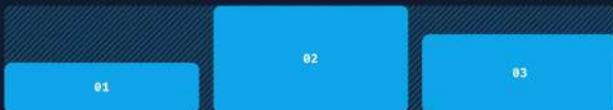
Use `'items-center'` to align items along the center of the container's cross axis:



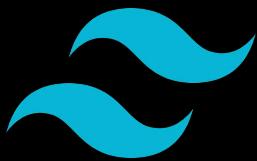
```
<div class="flex items-center ...">
  <div class="py-4">01</div>
  <div class="py-12">02</div>
  <div class="py-8">03</div>
</div>
```

## End

Use `'items-end'` to align items to the end of the container's cross axis:



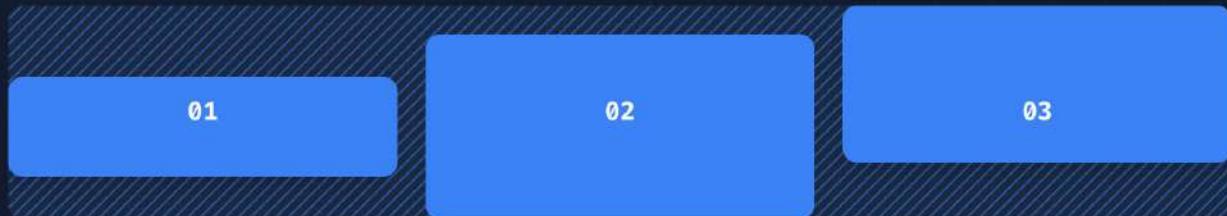
```
<div class="flex items-end ...">
  <div class="py-4">01</div>
  <div class="py-12">02</div>
  <div class="py-8">03</div>
</div>
```



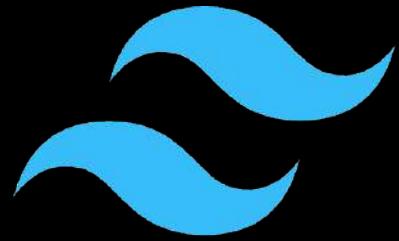
# 8.8 Align-Items

## Baseline

Use `items-baseline` to align items along the container's cross axis such that all of their baselines align:



```
<div class="flex items-baseline ...">
  <div class="pt-2 pb-6">01</div>
  <div class="pt-8 pb-12">02</div>
  <div class="pt-12 pb-4">03</div>
</div>
```



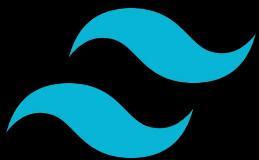
# tailwindcss

## LEARNING

### 9. *Grid*

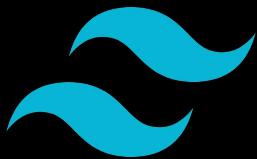


# 9.1 Grid Template Column



Class	Properties
grid-cols-1	grid-template-columns: repeat(1, minmax(0, 1fr));
grid-cols-2	grid-template-columns: repeat(2, minmax(0, 1fr));
grid-cols-3	grid-template-columns: repeat(3, minmax(0, 1fr));
grid-cols-4	grid-template-columns: repeat(4, minmax(0, 1fr));
grid-cols-5	grid-template-columns: repeat(5, minmax(0, 1fr));
grid-cols-6	grid-template-columns: repeat(6, minmax(0, 1fr));
grid-cols-7	grid-template-columns: repeat(7, minmax(0, 1fr));
grid-cols-8	grid-template-columns: repeat(8, minmax(0, 1fr));
grid-cols-9	grid-template-columns: repeat(9, minmax(0, 1fr));
grid-cols-10	grid-template-columns: repeat(10, minmax(0, 1fr));
grid-cols-11	grid-template-columns: repeat(11, minmax(0, 1fr));
grid-cols-12	grid-template-columns: repeat(12, minmax(0, 1fr));
grid-cols-none	grid-template-columns: none;
grid-cols-subgrid	grid-template-columns: subgrid;

# 9.1 Grid Template Column



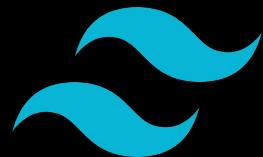
## Specifying the columns in a grid

Use the `grid-cols-\*` utilities to create grids with  $n$  equally sized columns.



```
<div class="grid grid-cols-4 gap-4">
  <div>01</div>
  <!-- ... -->
  <div>09</div>
</div>
```

# 9.1 Grid Template Column



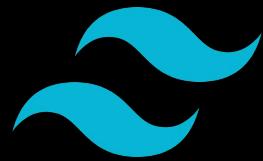
## Applying conditionally

### Hover, focus, and other states

Tailwind lets you conditionally apply utility classes in different states using variant modifiers. For example, use `hover:grid-cols-6` to only apply the `grid-cols-6` utility on hover.

```
<div class="grid grid-cols-1 hover:grid-cols-6">  
  <!-- ... -->  
</div>
```

# 9.2 Grid Column Start / End



## Spanning columns

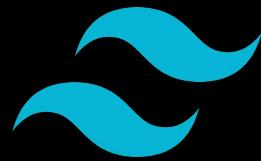
Use the `col-span-\*` utilities to make an element span  $n$  columns.

A grid layout with 7 columns. Columns 01, 02, and 03 are single columns. Column 04 spans 2 columns, so it covers columns 04 and 05. Column 06 spans 2 columns, so it covers columns 06 and 07. All columns have rounded corners and a slight shadow.

01	02	03				
			04	05		
					06	07

```
<div class="grid grid-cols-3 gap-4">
  <div class="...">01</div>
  <div class="...">02</div>
  <div class="...">03</div>
  <div class="col-span-2 ...">04</div>
  <div class="...">05</div>
  <div class="...">06</div>
  <div class="col-span-2 ...">07</div>
</div>
```

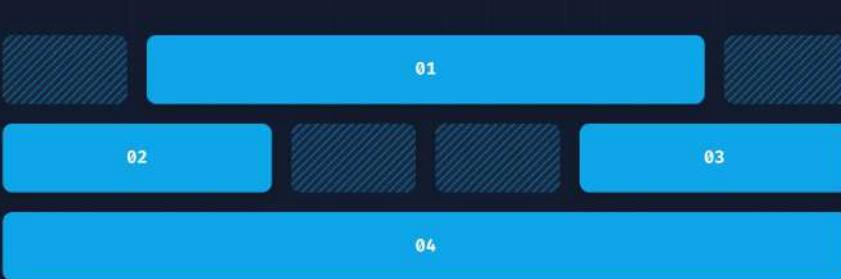
# 9.2 Grid Column Start / End



## Starting and ending lines

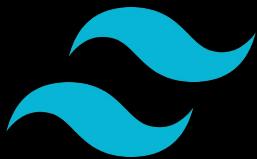
Use the `'col-start-*'` and `'col-end-*'` utilities to make an element start or end at the *n<sup>th</sup>* grid line. These can also be combined with the `'col-span-*'` utilities to span a specific number of columns.

Note that CSS grid lines start at 1, not 0, so a full-width element in a 6-column grid would start at line 1 and end at line 7.



```
<div class="grid grid-cols-6 gap-4">
  <div class="col-start-2 col-span-4 ...">01</div>
  <div class="col-start-1 col-end-3 ...">02</div>
  <div class="col-end-7 col-span-2 ...">03</div>
  <div class="col-start-1 col-end-7 ...">04</div>
</div>
```

# 9.3 Grid Template Rows



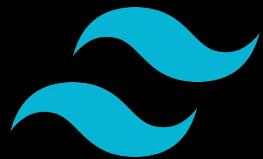
## Specifying the rows in a grid

Use the `grid-rows-*` utilities to create grids with  $n$  equally sized rows.



```
<div class="grid grid-rows-4 grid-flow-col gap-4">
  <div>01</div>
  <!-- ... -->
  <div>09</div>
</div>
```

# 9.4 Grid Row Start/End



## Basic usage

### Spanning rows

Use the `row-span-\*` utilities to make an element span  $n$  rows.

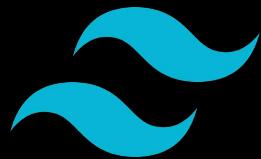
01

02

03

```
<div class="grid grid-rows-3 grid-flow-col gap-4">
  <div class="row-span-3 ...">01</div>
  <div class="col-span-2 ...">02</div>
  <div class="row-span-2 col-span-2 ...">03</div>
</div>
```

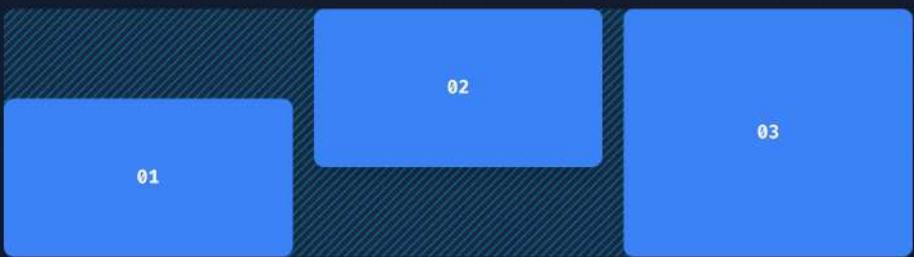
# 9.4 Grid Row Start/End



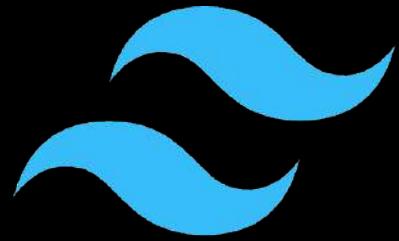
## Starting and ending lines

Use the `row-start-\*` and `row-end-\*` utilities to make an element start or end at the *n*th grid line. These can also be combined with the `row-span-\*` utilities to span a specific number of rows.

Note that CSS grid lines start at 1, not 0, so a full-height element in a 3-row grid would start at line 1 and end at line 4.



```
<div class="grid grid-rows-3 grid-flow-col gap-4">
  <div class="row-start-2 row-span-2 ...">01</div>
  <div class="row-end-3 row-span-2 ...">02</div>
  <div class="row-start-1 row-end-4 ...">03</div>
</div>
```



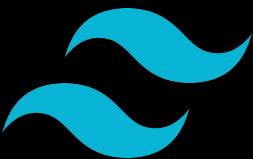
# tailwindcss

## LEARNING

### 10. *Border*



# 10.1 Border Radius



## Rounded corners

Use utilities like `'rounded-sm'`, `'rounded'`, or `'rounded-lg'` to apply different border radius sizes to an element.

rounded



rounded-md



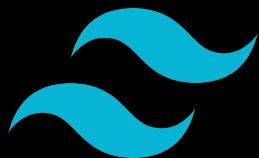
rounded-lg



rounded-full



```
<div class="rounded ..."></div>
<div class="rounded-md ..."></div>
<div class="rounded-lg ..."></div>
<div class="rounded-full ..."></div>
```



# 10.2 Border Width

Use the `border`, `border-0`, `border-2`, `border-4`, or `border-8` utilities to set the border width for all sides of an element.

border



border-2



border-4

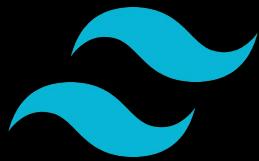


border-8



```
<div class="border border-sky-500"></div>
<div class="border-2 border-sky-500"></div>
<div class="border-4 border-sky-500"></div>
<div class="border-8 border-sky-500"></div>
```

# 10.3 Border Color



## Setting the border color

Use the ``border-*`` utilities to control the border color of an element.

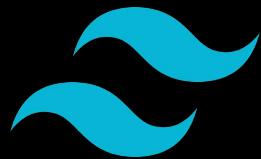
Email address

jane@example.com

This field is required.

```
<input class="border-2 border-rose-500 ..." >
```

# 10.3 Border Color



## Changing the opacity

Use the color opacity modifier to control the opacity of an element's border color.

border-indigo-500/100



border-indigo-500/75

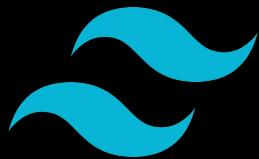


border-indigo-500/50



```
<div class="border-4 border-indigo-500/100 ..." ></div>
<div class="border-4 border-indigo-500/75 ..." ></div>
<div class="border-4 border-indigo-500/50 ..." ></div>
```

# 10.4 Border Style



## Setting the border style

Use `border-\*` to control an element's border style.

border-solid

Button A

border-dashed

Button A

border-dotted

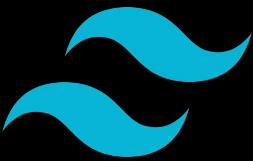
Button A

border-double

Button A

```
<div class="border-solid border-2 border-sky-500 ..."></div>
<div class="border-dashed border-2 border-sky-500 ..."></div>
<div class="border-dotted border-2 border-sky-500 ..."></div>
<div class="border-double border-4 border-sky-500 ..."></div>
```

# 10.5 Divide Width



## Add borders between horizontal children

Use the `divide-x-\*` utilities to add borders between horizontal elements.

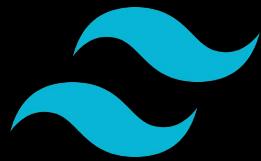
01

02

03

```
<div class="grid grid-cols-3 divide-x">
  <div>01</div>
  <div>02</div>
  <div>03</div>
</div>
```

# 10.5 Divide Width



## Add borders between stacked children

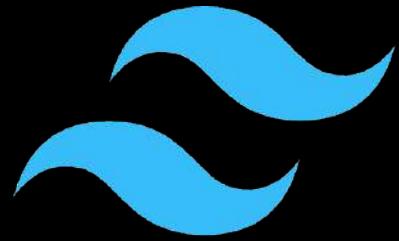
Use the `divide-y-\*` utilities to add borders between stacked elements.

01

02

03

```
<div class="grid grid-cols-1 divide-y">
  <div>01</div>
  <div>02</div>
  <div>03</div>
</div>
```

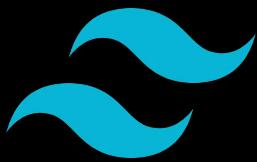


# tailwindcss

## LEARNING

### 11. Effects

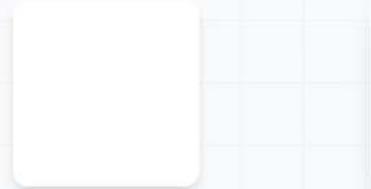




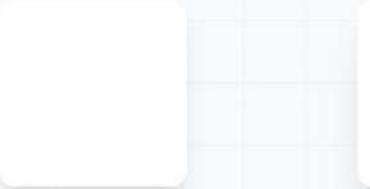
# 11.1 Box Shadow

Use the `shadow-sm`, `shadow`, `shadow-md`, `shadow-lg`, `shadow-xl`, or `shadow-2xl` utilities to apply different sized outer box shadows to an element.

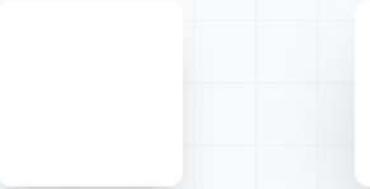
shadow-md



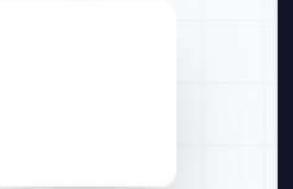
shadow-lg



shadow-xl

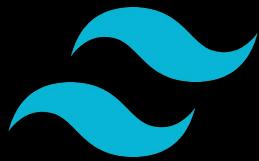


shadow-2xl



```
<div class="shadow-md ..."></div>
<div class="shadow-lg ..."></div>
<div class="shadow-xl ..."></div>
<div class="shadow-2xl ..."></div>
```

# 11.1 Box Shadow



## Adding an inner shadow

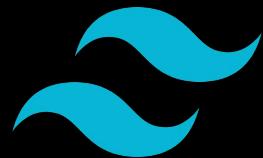
Use the ``shadow-inner`` utility to apply a subtle inset box shadow to an element. This can be useful for things like form controls or wells.

shadow-inner



```
<div class="shadow-inner ..."></div>
```

# 11.2 Box Shadow Color



## Basic usage

### Setting the box shadow color

Use the `shadow-\*` utilities to change the color of an existing box shadow. By default colored shadows have an opacity of 100%, but you can adjust this using the opacity modifier.

shadow-cyan-500/50

Subscribe

shadow-blue-500/50

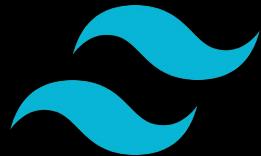
Subscribe

shadow-indigo-500/50

Subscribe

```
<button class="bg-cyan-500 shadow-lg shadow-cyan-500/50 ..." >Subscribe</button>
<button class="bg-blue-500 shadow-lg shadow-blue-500/50 ..." >Subscribe</button>
<button class="bg-indigo-500 shadow-lg shadow-indigo-500/50 ..." >Subscribe</button>
```

# 11.3 Opacity



## Basic usage

### Changing an element's opacity

Use the `opacity-\*` utilities to control the opacity of an element.

opacity-100

opacity-75

opacity-50

opacity-25

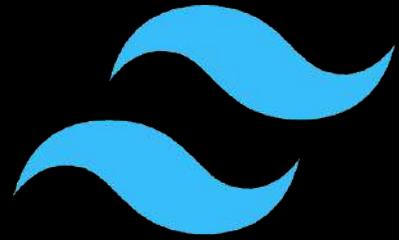
Button A

Button B

Button C

Button D

```
<button class="bg-indigo-500 opacity-100 ..." ></button>
<button class="bg-indigo-500 opacity-75 ..." ></button>
<button class="bg-indigo-500 opacity-50 ..." ></button>
<button class="bg-indigo-500 opacity-25 ..." ></button>
```

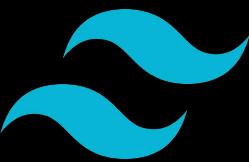


# tailwindcss

## LEARNING

### 12. Filters





# 12.1 Blur

Use the `blur-\*` utilities to blur an element.

blur-none



blur-sm



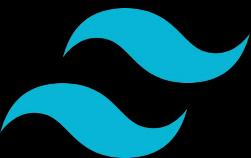
blur-lg



blur-2xl



```
<div class="blur-none ...">  
  <!-- ... -->  
</div>  
<div class="blur-sm ...">  
  <!-- ... -->  
</div>  
<div class="blur-lg ...">  
  <!-- ... -->  
</div>  
<div class="blur-2xl ...">  
  <!-- ... -->  
</div>
```



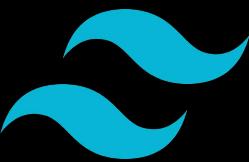
# 12.1 Blur

## Removing filters

To remove all of the filters on an element at once, use the ``filter-none`` utility:

```
<div class="blur-md invert brightness-150 md:filter-none">  
  <!-- ... -->  
</div>
```

This can be useful when you want to remove filters conditionally, such as on hover or at a particular breakpoint.



# 12.2 Brightness

Use the `brightness-*` utilities to control an element's brightness.

brightness-50



brightness-100



brightness-125

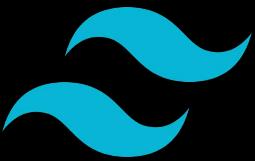


brightness-200



```
<div class="brightness-50 ...">  
  <!-- ... -->  
</div>  
<div class="brightness-100 ...">  
  <!-- ... -->  
</div>  
<div class="brightness-125 ...">  
  <!-- ... -->  
</div>  
<div class="brightness-200 ...">  
  <!-- ... -->  
</div>
```

# 12.3 Contrast



Use the `~contrast-*`` utilities to control an element's contrast.

contrast-50



contrast-100



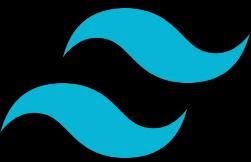
contrast-125



contrast-200



```
<div class="contrast-50 ..." >
  <!-- ... -->
</div>
<div class="contrast-100 ..." >
  <!-- ... -->
</div>
<div class="contrast-125 ..." >
  <!-- ... -->
</div>
<div class="contrast-200 ..." >
  <!-- ... -->
</div>
```



# 12.4 Drop Shadow

Use the `drop-shadow-*` utilities to add a drop shadow to an element.

`drop-shadow-md`



`drop-shadow-lg`



`drop-shadow-xl`

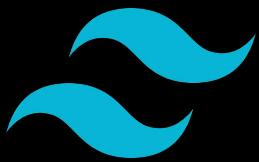


`drop-shadow-2xl`



```
<div class="drop-shadow-md ...">
  <!-- ... -->
</div>
<div class="drop-shadow-lg ...">
  <!-- ... -->
</div>
<div class="drop-shadow-xl ...">
  <!-- ... -->
</div>
<div class="drop-shadow-2xl ...">
  <!-- ... -->
</div>
```

This is useful for applying shadows to irregular shapes, like text and SVG elements. For applying shadows to regular elements, you probably want to use box shadow instead.



# 12.5 Grayscale

Use the `grayscale` and `grayscale-0` utilities to control whether an element should be rendered as grayscale or in full color.

grayscale-0

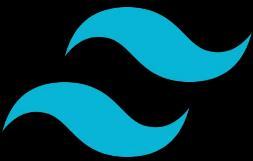


grayscale



```
<div class="grayscale-0 ..." >  
  <!-- ... -->  
</div>  
<div class="grayscale ..." >  
  <!-- ... -->  
</div>
```

# 12.6 Invert



Use the `invert` and `invert-0` utilities to control whether an element should be rendered with inverted colors or normally.

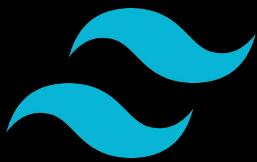
invert-0



invert



```
<div class="invert-0 ...">  
  <!-- ... -->  
</div>  
<div class="invert ...">  
  <!-- ... -->  
</div>
```



# 12.7 Sepia

Use the `sepia` and `sepia-0` utilities to control whether an element should be rendered as sepia or in full color.

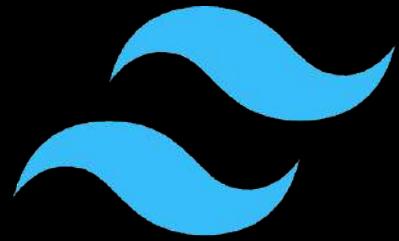
sepia-0



sepia



```
<div class="sepia-0 ...">  
  <!-- ... -->  
</div>  
<div class="sepia ...">  
  <!-- ... -->  
</div>
```



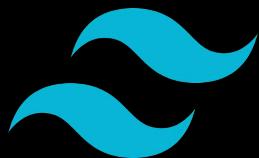
# tailwindcss

## LEARNING

### 13. *Animations*



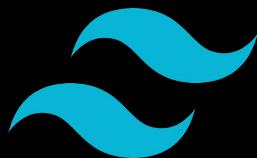
# 13.1 Transition Property



Save Changes

```
<button class="transition ease-in-out delay-150 bg-blue-500  
hover:-translate-y-1 hover:scale-110 hover:bg-indigo-500  
duration-300">  
  Save Changes  
</button>
```

# 13.2 Transition Duration



Use the `duration-\*` utilities to control an element's transition-duration.

👉 Hover each button to see the expected behaviour

duration-150

Button A

duration-300

Button B

duration-700

Button C

```
<button class="transition duration-150 ease-in-out ...">Button A</button>
<button class="transition duration-300 ease-in-out ...">Button B</button>
<button class="transition duration-700 ease-in-out ...">Button C</button>
```

# 13.3 Transition Timing Function



Use the `ease-\*` utilities to control an element's easing curve.

💡 Hover each button to see the expected behaviour

ease-in

Button A

ease-out

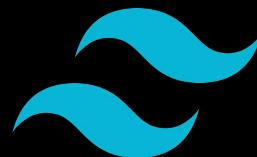
Button B

ease-in-out

Button C

```
<button class="ease-in duration-300 ...>Button A</button>
<button class="ease-out duration-300 ...>Button B</button>
<button class="ease-in-out duration-300 ...>Button C</button>
```

# 13.4 Transition Delay



Use the `delay-\*` utilities to control an element's transition-delay.

👉 Hover each button to see the expected behaviour

delay-150

Button A

delay-300

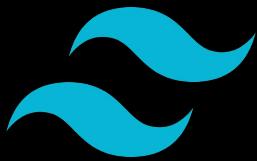
Button B

delay-700

Button C

```
<button class="transition delay-150 duration-300 ease-in-out ...">Button A</button>
<button class="transition delay-300 duration-300 ease-in-out ...">Button B</button>
<button class="transition delay-700 duration-300 ease-in-out ...">Button C</button>
```

# 13.5 Animation



## Spin

Add the `'animate-spin'` utility to add a linear spin animation to elements like loading indicators.

 Processing...

```
<button type="button" class="bg-indigo-500 ..." disabled>
  <svg class="animate-spin h-5 w-5 mr-3 ..." viewBox="0 0 24 24">
    <!-- ... -->
  </svg>
  Processing...
</button>
```

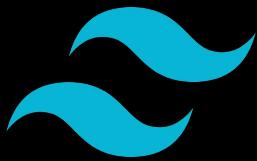
## Ping

Add the `'animate-ping'` utility to make an element scale and fade like a radar ping or ripple of water — useful for things like notification badges.

Transactions 

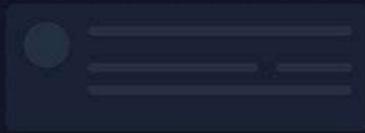
```
<span class="relative flex h-3 w-3">
  <span class="animate-ping absolute inline-flex h-full w-full rounded-full bg-sky-400"></span>
  <span class="relative inline-flex rounded-full h-3 w-3 bg-sky-500"></span>
```

# 13.5 Animation



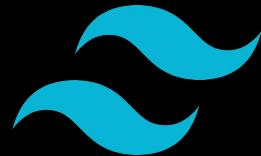
## Pulse

Add the `animate-pulse` utility to make an element gently fade in and out — useful for things like skeleton loaders.



```
<div class="border border-blue-300 shadow rounded-md p-4 max-w-sm w-full mx-auto">
  <div class="animate-pulse flex space-x-4">
    <div class="rounded-full bg-slate-700 h-10 w-10"></div>
    <div class="flex-1 space-y-6 py-1">
      <div class="h-2 bg-slate-700 rounded"></div>
      <div class="space-y-3">
        <div class="grid grid-cols-3 gap-4">
          <div class="h-2 bg-slate-700 rounded col-span-2"></div>
          <div class="h-2 bg-slate-700 rounded col-span-1"></div>
        </div>
        <div class="h-2 bg-slate-700 rounded"></div>
      </div>
    </div>
  </div>
</div>
```

# 13.5 Animation



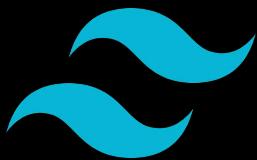
## Bounce

Add the `animate-bounce` utility to make an element bounce up and down — useful for things like "scroll down" indicators.



```
<svg class="animate-bounce w-6 h-6 ...">  
  <!-- -->  
</svg>
```

# 13.6 Scale



## Scaling an element

Use the `scale-\*`, `scale-x-\*`, and `scale-y-\*` utilities to scale an element.

scale-75



scale-100

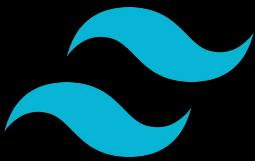


scale-125



```
<img class="scale-75 ...">  
<img class="scale-100 ...">  
<img class="scale-125 ...">
```

# 13.7 Rotate



## Rotating an element

Use the `rotate-\*` utilities to rotate an element.

rotate-0



rotate-45



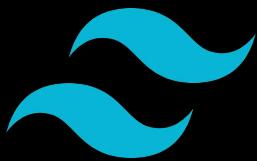
rotate-90



rotate-180



```
<img class="rotate-0 ...">
<img class="rotate-45 ...">
<img class="rotate-90 ...">
<img class="rotate-180 ...">
```



# 13.8 Translate

## Translating an element

Use the `translate-x-\*` and `translate-y-\*` utilities to translate an element.

translate-y-6



-translate-y-6

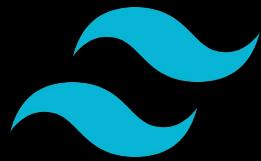


translate-x-6



```
<img class="translate-y-6 ...">  
<img class="-translate-y-6 ...">  
<img class="translate-x-6 ...">
```

# 13.9 Skew



Use the `skew-x-\*` and `skew-y-\*` utilities to skew an element.

skew-y-0



skew-y-3



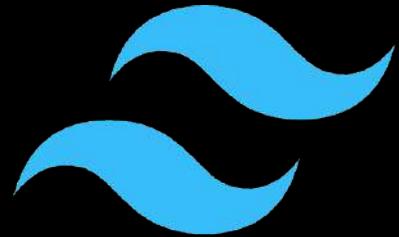
skew-y-6



skew-y-12



```
<img class="skew-y-0 ..." >  
<img class="skew-y-3 ..." >  
<img class="skew-y-6 ..." >  
<img class="skew-y-12 ..." >
```



# tailwindcss

## LEARNING

### 14. Advance Concepts





# 14.1 Reusing Styles

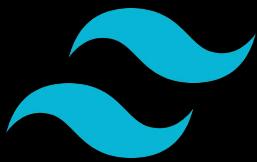
## Multi-cursor editing

When duplication is localized to a group of elements in a single file, the easiest way to deal with it is to use **multi-cursor editing** to quickly select and edit the class list for each element at once:

Home Team Projects Reports

```
<nav class="flex justify-center space-x-4">
  <a href="/dashboard" class="font-medium px-3 py-2 text-slate-700 rounded-lg hover:b
  <a href="/team" class="font-medium px-3 py-2 text-slate-700 rounded-lg hover:bg-sla
  <a href="/projects" class="font-medium px-3 py-2 text-slate-700 rounded-lg hover:bg-
  <a href="/reports" class="font-medium px-3 py-2 text-slate-700 rounded-lg hover:bg-
</nav>
```

You'd be surprised at how often this ends up being the best solution. If you can quickly edit all of the duplicated class lists simultaneously, there's no benefit to introducing any additional abstraction.



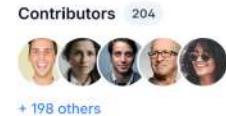
# 14.1 Reusing Styles

## Loops

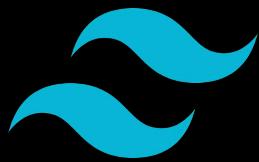
Before you assume you're going to need to extract a component or create a custom class for something, make sure you're *actually* using it more than once in your template.

A lot of the time a design element that shows up more than once in the rendered page is only actually authored once because the actual markup is rendered in a loop.

For example, the duplicate avatars at the beginning of this guide would almost certainly be rendered in a loop in a real project:



```
<div>
  <div class="flex items-center space-x-2 text-base">
    <h4 class="font-semibold text-slate-900">Contributors</h4>
    <span class="rounded-full bg-slate-100 px-2 py-1 text-xs font-semibold text-slate-900">204</span>
  </div>
  <div class="mt-3 flex -space-x-2 overflow-hidden">
    {#each contributors as user}
      
    {/each}
  </div>
  <div class="mt-3 text-sm font-medium">
    <a href="#" class="text-blue-500">+ 198 others</a>
  </div>
</div>
```



# 14.1 Reusing Styles

## Extracting components and partials

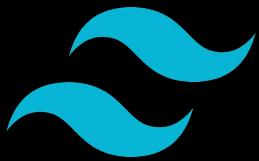
If you need to reuse some styles across multiple files, the best strategy is to create a *component* if you're using a front-end framework like React, Svelte, or Vue, or a *template partial* if you're using a templating language like Blade, ERB, Twig, or Nunjucks.



**PRIVATE VILLA**  
**Relaxing All-Inclusive Resort in**  
**Cancun**

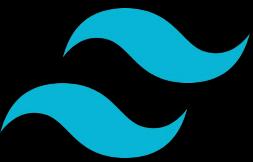
\$299 USD per night

# 14.2 Adding Custom Styles



tailwind.config.js

```
/** @type {import('tailwindcss').Config} */
module.exports = {
  theme: {
    screens: {
      sm: '480px',
      md: '768px',
      lg: '976px',
      xl: '1440px',
    },
    colors: {
      'blue': '#1fb6ff',
      'pink': '#ff49db',
      'orange': '#ff7849',
      'green': '#13ce66',
      'gray-dark': '#273444',
      'gray': '#8492a6',
      'gray-light': '#d3dce6',
    },
    fontFamily: {
      sans: ['Graphik', 'sans-serif'],
      serif: ['Merriweather', 'serif'],
    },
    extend: {
      spacing: {
        '128': '32rem',
        '144': '36rem',
      },
      borderRadius: {
        '4xl': '2rem',
      }
    }
}
```



# 14.3 Directives

## @layer

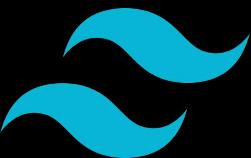
Use the `@layer` directive to tell Tailwind which "bucket" a set of custom styles belong to. Valid layers are `base`, `components`, and `utilities`.

```
@tailwind base;
@tailwind components;
@tailwind utilities;

@layer base {
  h1 {
    @apply text-2xl;
  }
  h2 {
    @apply text-xl;
  }
}

@layer components {
  .btn-blue {
    @apply bg-blue-500 hover:bg-blue-700 text-white font-bold py-2 px-4 rounded;
  }
}

@layer utilities {
  .filter-none {
    filter: none;
  }
  .filter-grayscale {
    filter: grayscale(100%);
  }
}
```



# 14.3 Directives

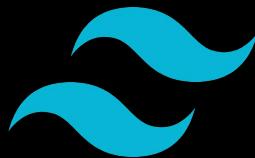
## @apply

Use `@apply` to inline any existing utility classes into your own custom CSS.

This is useful when you need to write custom CSS (like to override the styles in a third-party library) but still want to work with your design tokens and use the same syntax you're used to using in your HTML.

```
.select2-dropdown {  
  @apply rounded-b-lg shadow-md;  
}  
.select2-search {  
  @apply border border-gray-300 rounded;  
}  
.select2-results__group {  
  @apply text-lg font-bold text-gray-900;  
}
```

# Project: Personal Portfolio



**John Doe**  
Full Stack Developer



## About Me

I'm a passionate full stack developer with 5 years of experience in building web applications. I love creating efficient, scalable, and user-friendly solutions to complex problems.

## Projects

### E-commerce Platform

Developed a fully-functional e-commerce platform with user authentication, product management, and payment integration.

React Node.js MongoDB Stripe

### Social Media Dashboard

Created a responsive dashboard for social media analytics, featuring real-time data visualization and reporting.

Vue.js D3.js Express PostgreSQL

## Skills

JavaScript React Node.js Python SQL Git AWS Docker

## Education

### Bachelor of Science in Computer Science

University of Technology, 2015-2019

### Full Stack Web Development Bootcamp

Code Academy, 2020

## Hobbies & Interests

- Photography
- Hiking
- Playing guitar
- Reading sci-fi novels

## Extracurricular Activities

- Volunteer at local coding bootcamp for underprivileged youth
- Organizer of city-wide hackathon event
- Member of the University Chess Club

## Contact & Social Media

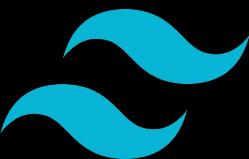
Email: john.doe@example.com

Phone: (123) 456-7890

 LinkedIn

 GitHub

 Twitter

A black and white photograph of a man from the chest up. He is wearing a dark suit jacket, a white shirt, and a dark tie. He has short, dark hair and is wearing glasses. He is looking directly at the camera with a neutral expression. The background is a window with a grid pattern, showing some foliage outside.

*The  
End*