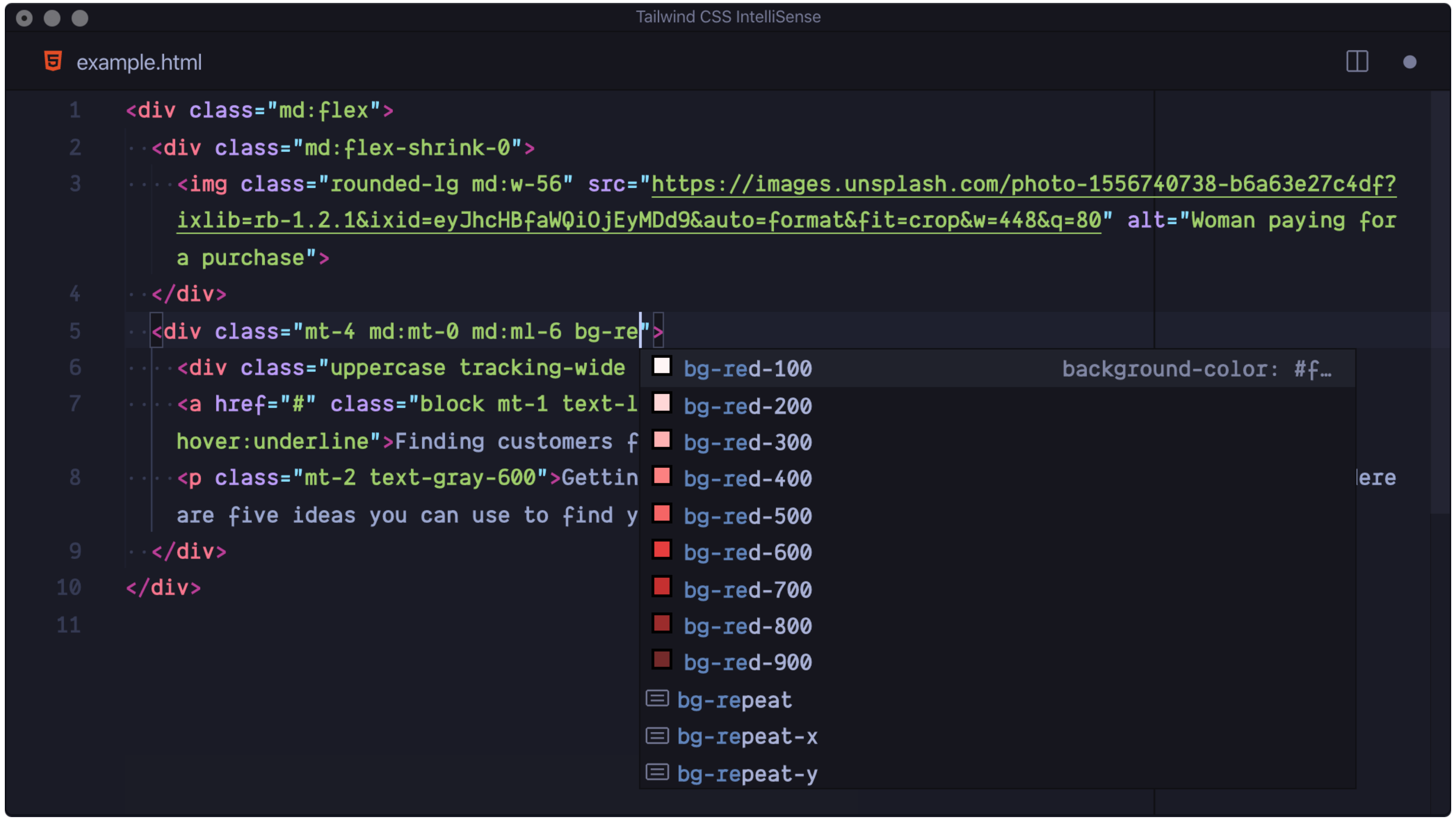
WEB DESIGN

W8 – Tailwind CSS



```html
<div class="md:flex">
  <div class="md:flex-shrink-0">
    <img class="rounded-lg md:w-56" src="https://images.unsplash.com/photo-1556740738-b6a63e27c4df?
    ixlib=rb-1.2.1&ixid=eyJhcHBfaWQiOjEyMDd9&auto=format&fit=crop&w=448&q=80" alt="Woman paying for
    a purchase">
  </div>
  <div class="mt-4 md:mt-0 md:ml-6 bg-re">
    <div class="uppercase tracking-wide
    <a href="#" class="block mt-1 text-l
    hover:underline">Finding customers f
    <p class="mt-2 text-gray-600">Gettin                                         ere
    are five ideas you can use to find y
  </div>
</div>
```

Course Objectives

Learn

Tailwind Principles

- Spacing System
- Width
- Font
- Flexbox
- Grid

Produce

Cleaner

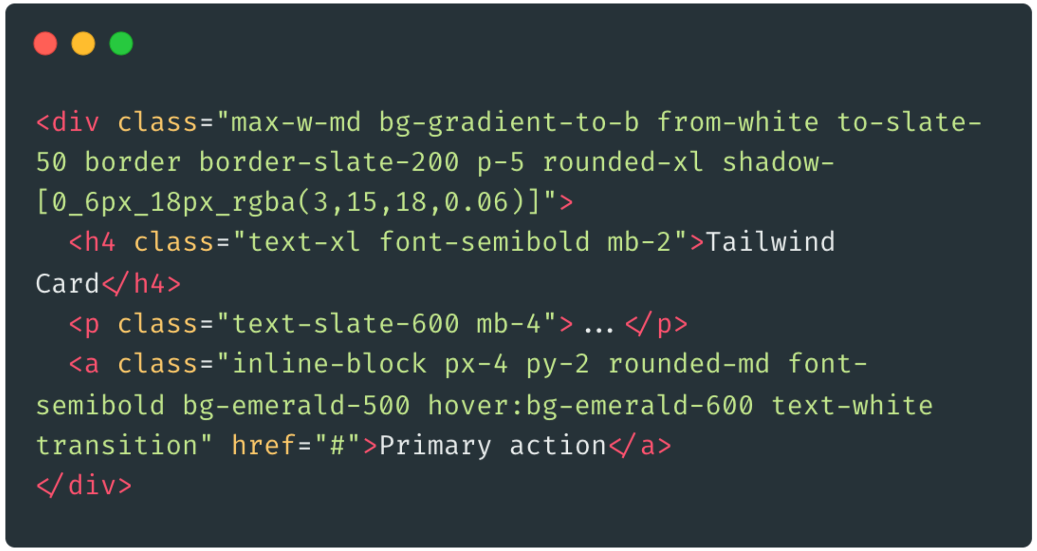
Maintainable

Code

Introduction to Utility-first CSS

“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.”

```
.card-normal { /* … */ }
.card-normal h3 { /* … */ }
.btn-normal { /* … */ }
```

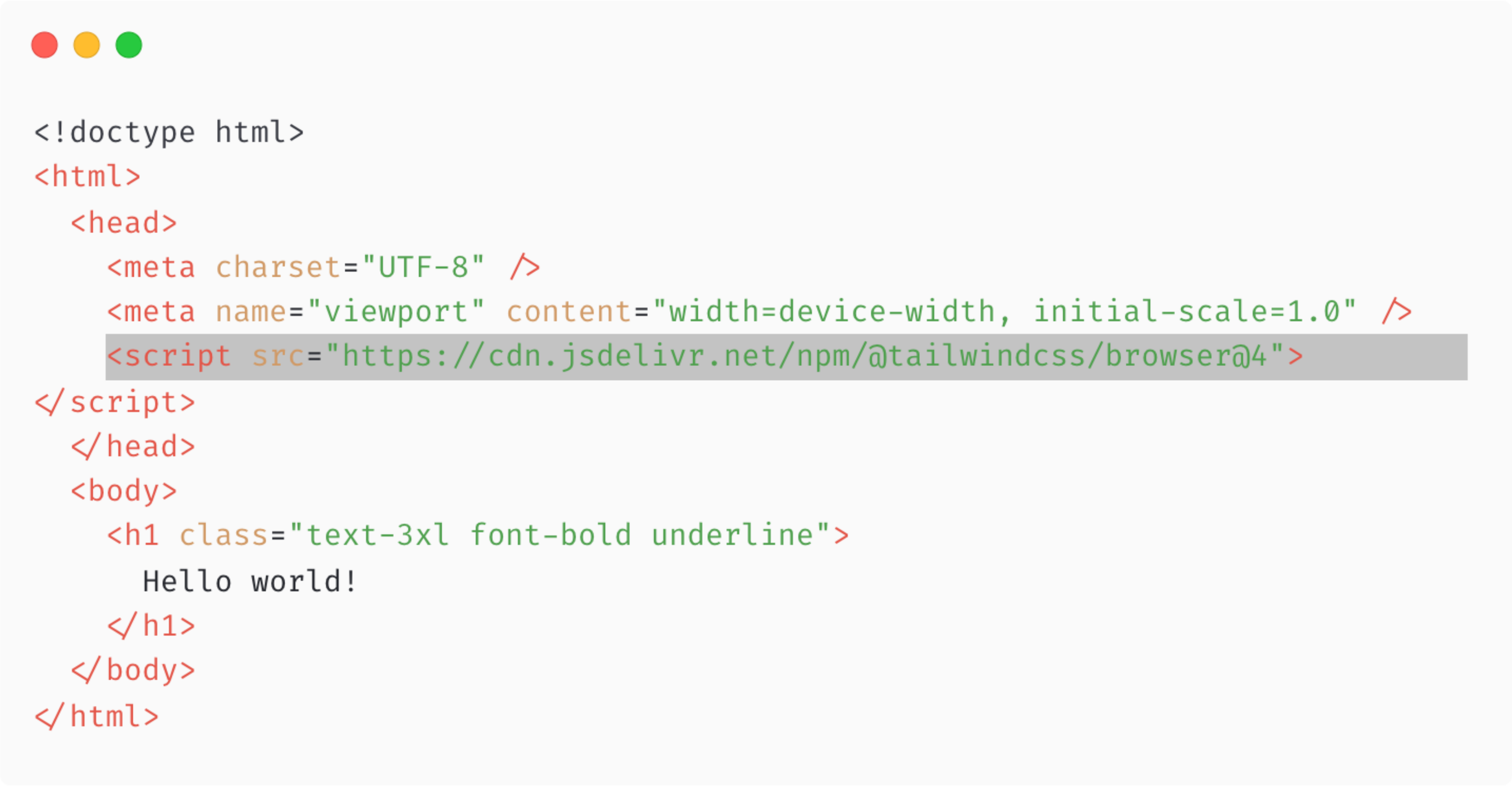


```
<div class="max-w-md bg-gradient-to-b from-white to-slate-
50 border border-slate-200 p-5 rounded-xl shadow-
[0_6px_18px_rgba(3,15,18,0.06)]">
  <h4 class="text-xl font-semibold mb-2">Tailwind
Card</h4>
  <p class="text-slate-600 mb-4">…</p>
  <a class="inline-block px-4 py-2 rounded-md font-
semibold bg-emerald-500 hover:bg-emerald-600 text-white
transition" href="#">Primary action</a>
</div>
```

Tailwind vs Plain CSS

Aspect	Utility-First	Traditional CSS
Where styles live	In markup via utility classes	In CSS files / stylesheets
Edit speed	Fast — edit HTML to change spacing/variant	Slower — edit CSS and rebuild or reload
Reusability	Reusable patterns via components or @apply	Reusable via classes and design systems
File size	Small when purged; many classes but reused	Depends on CSS architecture; can grow large

Add Tailwind CSS to your HTML

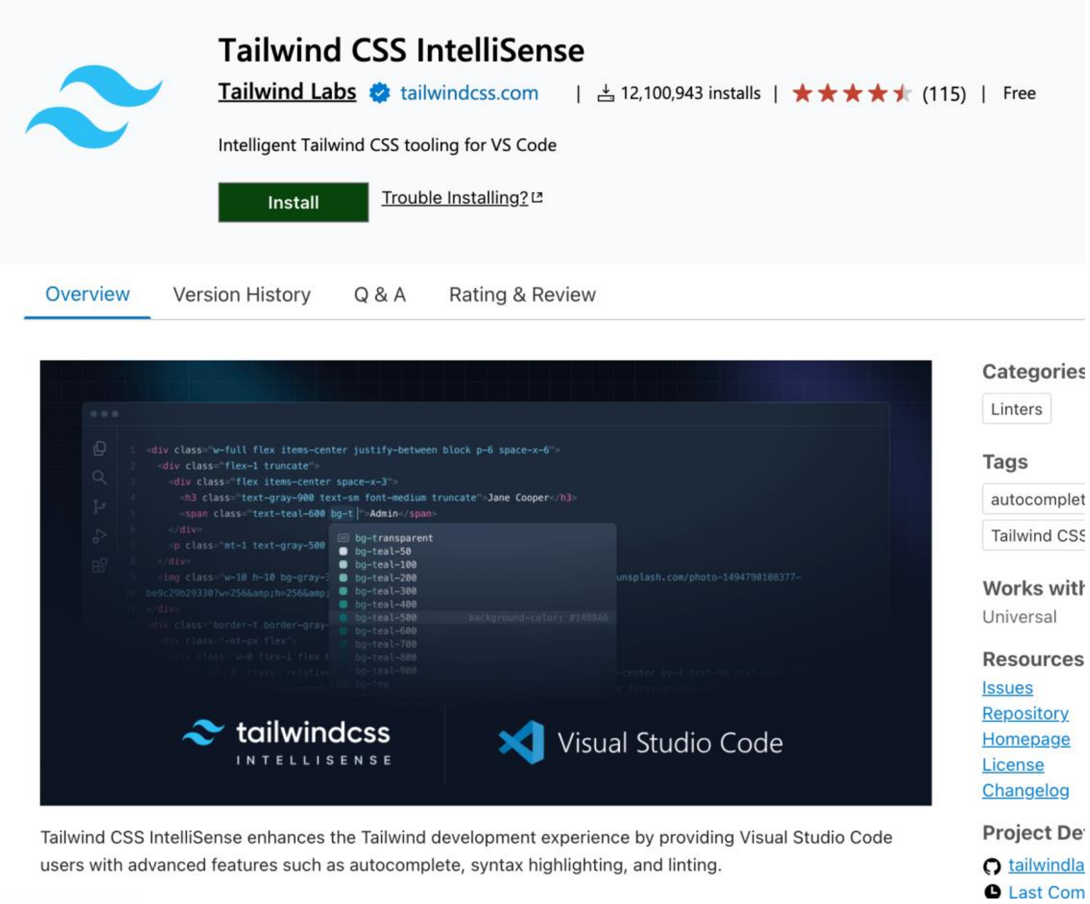


```
<!doctype html>
<html>
  <head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <script src="https://cdn.jsdelivr.net/npm/@tailwindcss/browser@4">
</script>
  </head>
  <body>
    <h1 class="text-3xl font-bold underline">
      Hello world!
    </h1>
  </body>
</html>
```

Add TailwindCSS to your project by CDN

https://tailwindcss.com/docs/installation/play-cdn

VS Code IntelliSense



Tailwind CSS IntelliSense

Tailwind Labs tailwindcss.com | 12,100,943 installs | ★★★★½ (115) | Free

Intelligent Tailwind CSS tooling for VS Code

```
<div class="w-full flex items-center justify-between block p-6 space-x-6">
  <div class="flex-1 truncate">
    <div class="flex items-center space-x-3">
      <h3 class="text-gray-900 text-sm font-medium truncate">Jane Cooper</h3>
      <span class="text-teal-600 bg-t|">Admin</span>
    </div>
    <p class="mt-1 text-gray-500
  </div>
  <img class="w-10 h-10 bg-gray-3                    unsplash.com/photo-1494790108377-
be9c29b29330?w=256&amp;h=256&amp;
</div>
<div class="border-t border-gray-
```

Tailwind CSS IntelliSense enhances the Tailwind development experience by providing Visual Studio Code users with advanced features such as autocomplete, syntax highlighting, and linting.

https://marketplace.visualstudio.com/items?itemName=bradlc.vscode-tailwindcss

Tailwind Basics

w-48 → width: 12rem;

p-4 → padding: 1rem;

bg-slate-300 → background-color: #e2e8f0;

rounded-lg → border-radius: 0.5rem;

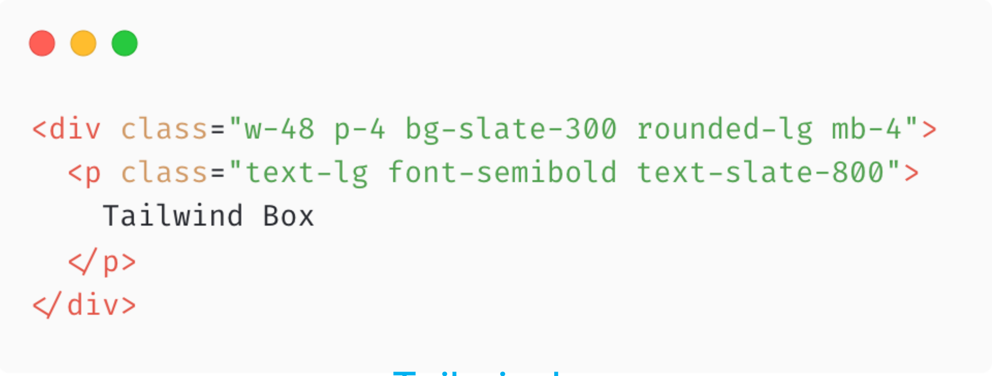
mb-4 → margin-bottom: 1rem;

text-lg → font-size: 1.125rem;

font-semibold → font-weight: 600;

- Everything Tailwind does is just CSS.
- Tailwind gives names to spacing, colors, and sizes so you don’t memorize values.
- You can build UI by combining small utility classes instead of writing CSS files.

Let’s See More Example

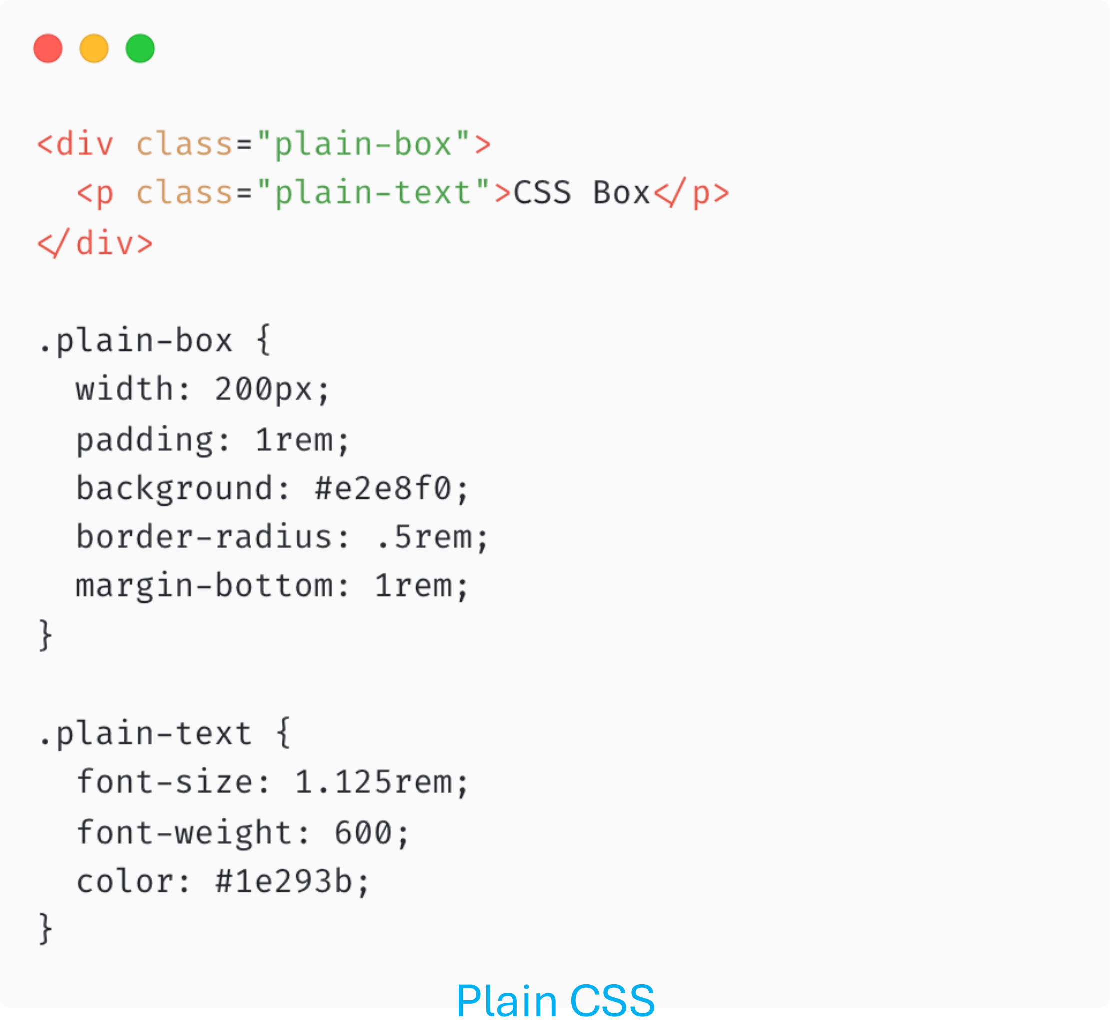


```html
<div class="w-48 p-4 bg-slate-300 rounded-lg mb-4">
  <p class="text-lg font-semibold text-slate-800">
    Tailwind Box
  </p>
</div>
```

Tailwind

How Tailwind Works Under the Hood

- Tailwind scans your HTML/JSX/Vue files.
- It finds classes like p-4, bg-red-500, text-sm.
- It generates the exact CSS rules for only those classes.
- Your final CSS is small because unused utilities are removed ("tree-shaken").



```html
<div class="plain-box">
  <p class="plain-text">CSS Box</p>
</div>

.plain-box {
  width: 200px;
  padding: 1rem;
  background: #e2e8f0;
  border-radius: .5rem;
  margin-bottom: 1rem;
}

.plain-text {
  font-size: 1.125rem;
  font-weight: 600;
  color: #1e293b;
}
```

Plain CSS

Sizing: Width

Utilities for setting the width of an element.

Class	Styles
w-<number>	width: calc(var(--spacing) * <number>);
w-<fraction>	width: calc(<fraction> * 100%);
w-3xs	width: var(--container-3xs); /* 16rem (256px) */
w-2xs	width: var(--container-2xs); /* 18rem (288px) */
w-xs	width: var(--container-xs); /* 20rem (320px) */

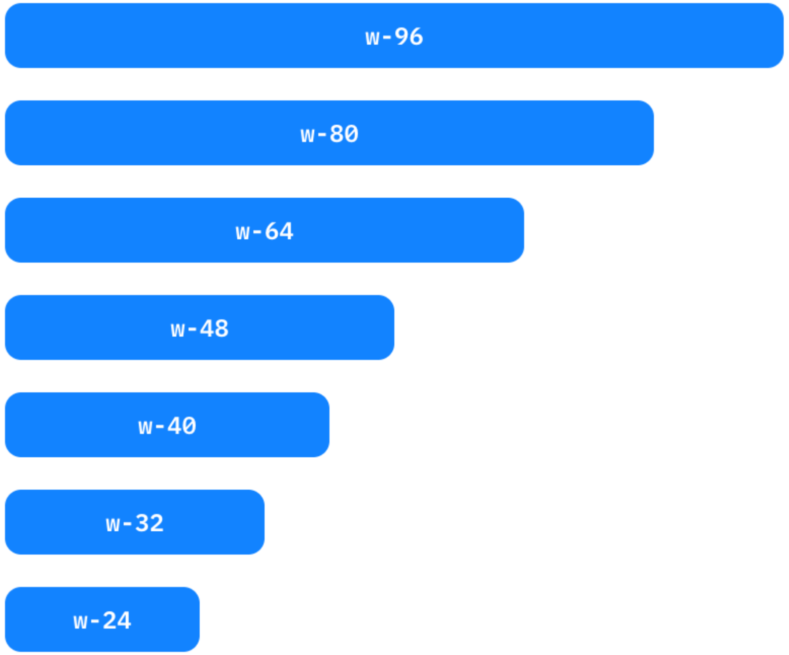
Check detail...

Width

Utilities for setting the width of an element.

Use w-*<number>* utilities like w-24 and w-64 to set an element to a fixed width based on the spacing scale:

```
<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>
```



Width (Cont)

Utilities for setting the width of an element.

What `w-64` actually means..

Tailwind uses a **spacing scale**.
When you write:

```
w-4
w-10
w-64
```

These numbers do **not** represent pixels.
They come from **Tailwind’s spacing system**, which internally uses a spacing variable.

```
w-<number>
width: calc(var(--spacing) * <number>);
```

Width (Cont)

Utilities for setting the width of an element.

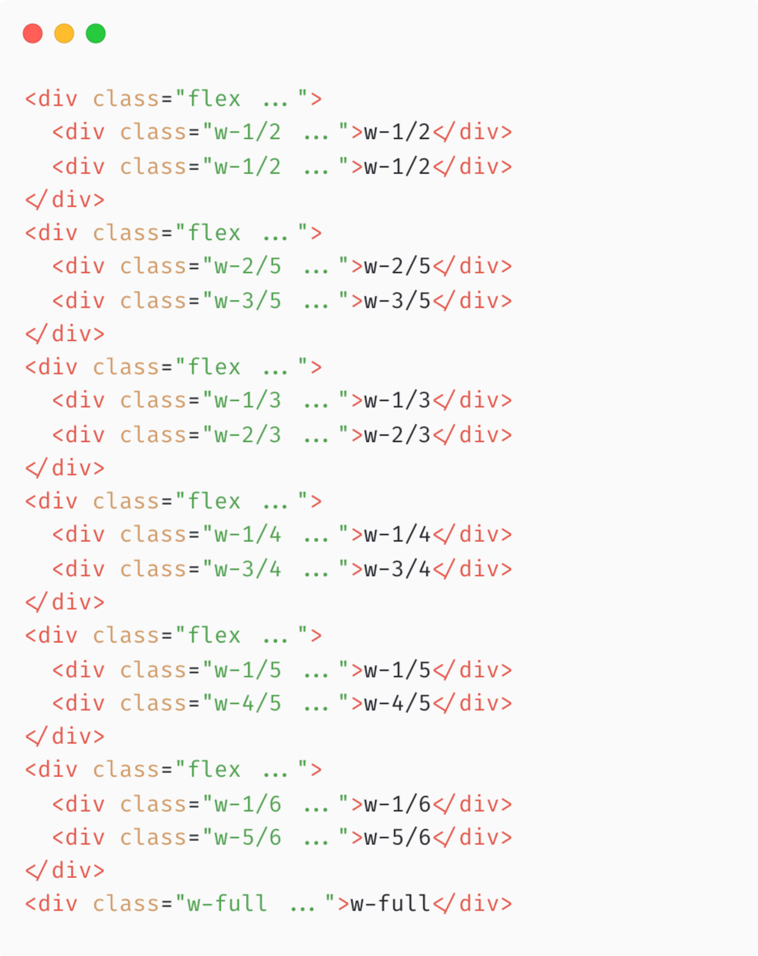
- `<number>` is a value from Tailwind’s spacing scale (e.g., 1, 2, 4, 8, 64)
- Tailwind defines a base spacing variable (usually `0.25rem`)   --spacing: 0.25rem;   /* = 4px */
- Each number multiplies that base value

```
w-1  = calc(0.25rem * 1)  = 0.25rem     = 4px
w-2  = calc(0.25rem * 2)  = 0.5rem      = 8px
w-4  = calc(0.25rem * 4)  = 1rem        = 16px
w-8  = calc(0.25rem * 8)  = 2rem        = 32px
w-64  = calc(0.25rem * 64)  = 16rem     = 256px
```

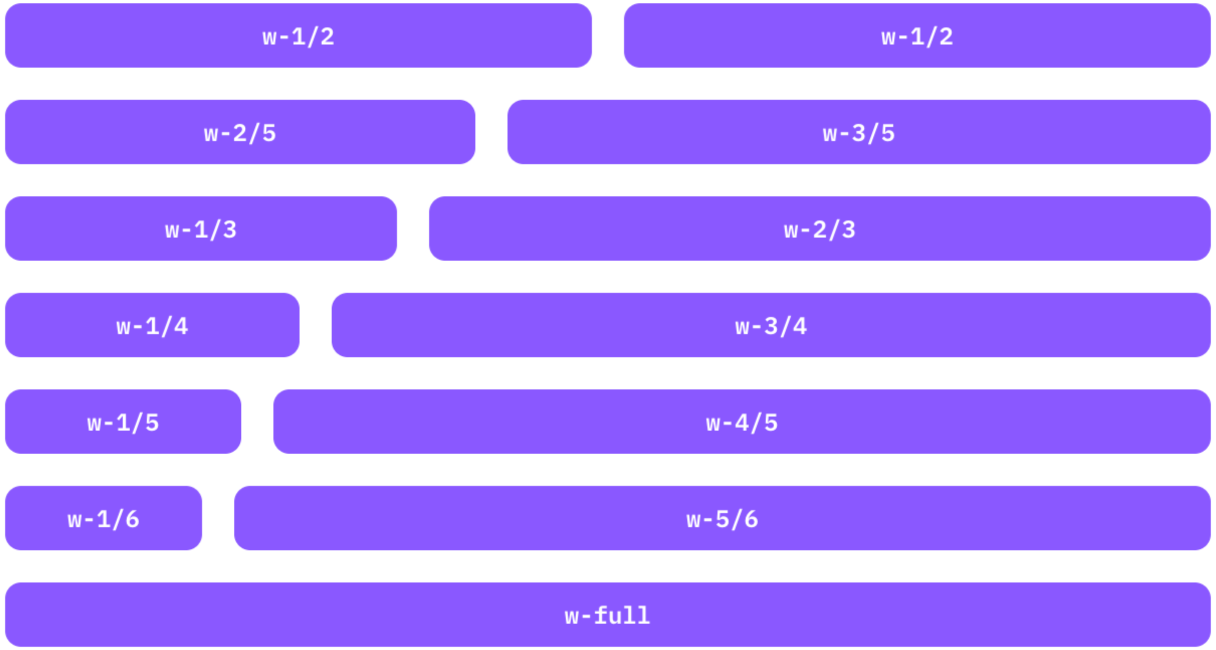
Width (Cont)

Utilities for setting the width of an element.

How about `w-<fraction>` ? Do your own research!



```
<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-5/6  ... ">w-5/6</div>
</div>
<div class="w-full  ... ">w-full</div>
```



Flex : *flex-direction*

Utilities for controlling the direction of flex items.

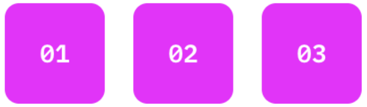
Class	**CSS Styles**
flex-row	flex-direction: row;
flex-row-reverse	flex-direction: row-reverse;
flex-col	flex-direction: column;
flex-col-reverse	flex-direction: column-reverse;

Check detail...

Flex : *flex-direction*

Utilities for controlling the direction of flex items.

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



Always use `flex` class first to tell tailwind that you want the container to be flex.

Grid : *grid-template-columns*

Utilities for controlling the direction of flex items.

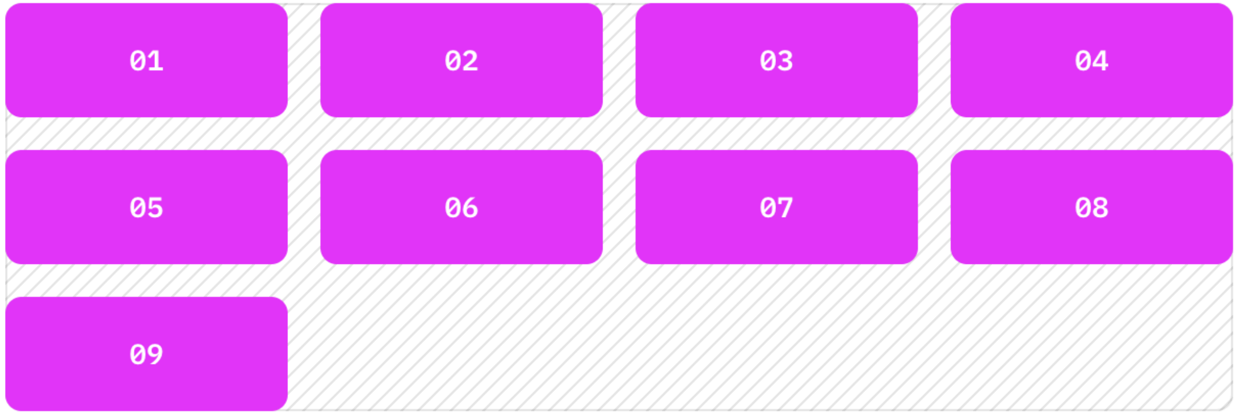
Class	CSS Styles
grid-cols-<number>	grid-template-columns: repeat(<number>, minmax(0, 1fr));
grid-cols-none	grid-template-columns: none;
grid-cols-subgrid	grid-template-columns: subgrid;
grid-cols-[<value>]	grid-template-columns: <value>;
grid-cols-(<custom-property>)	grid-template-columns: var(<custom-property>);

Check detail...

Grid : *grid-template-columns*

Utilities for controlling the direction of flex items.

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



Always use `grid` class first to tell tailwind that you want the container to be grid

Spacing : *Padding*

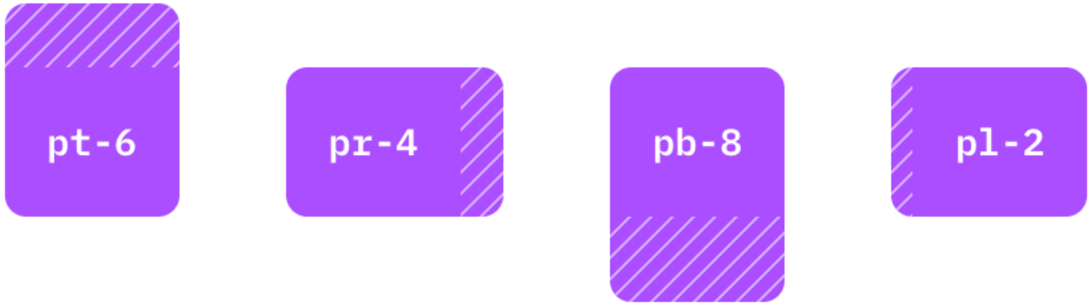
Utilities for controlling an element's padding.

Class	Styles
p-<number>	padding: calc(var(--spacing) * <number>);
p-px	padding: 1px;
p-(<custom-property>)	padding: var(<custom-property>);
p-[<value>]	padding: <value>;
px-<number>	padding-inline: calc(var(--spacing) * <number>);
px-px	padding-inline: 1px;
px-(<custom-property>)	padding-inline: var(<custom-property>);
px-[<value>]	padding-inline: <value>;
py-<number>	padding-block: calc(var(--spacing) * <number>);
py-px	padding-block: 1px;

Spacing : *Padding*

Utilities for controlling an element's padding.

```
<div class="pt-6 ...">pt-6</div>
<div class="pr-4 ...">pr-4</div>
<div class="pb-8 ...">pb-8</div>
<div class="pl-2 ...">pl-2</div>
```



Spacing : *Margin*

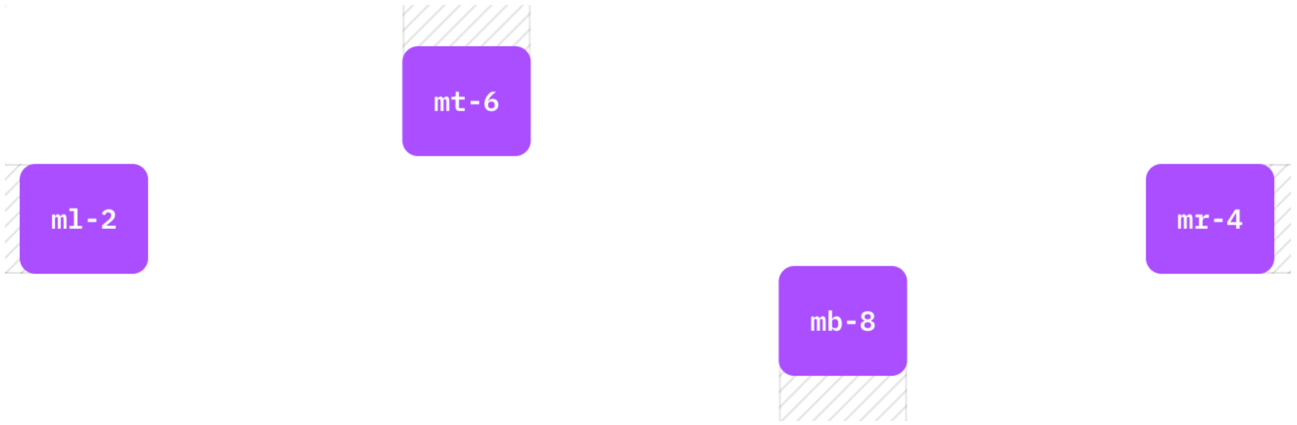
Utilities for controlling an element's margin.

Class	Styles
m-<number>	margin: calc(var(--spacing) * <number>);
-m-<number>	margin: calc(var(--spacing) * -<number>);
m-auto	margin: auto;
m-px	margin: 1px;
-m-px	margin: -1px;
m-(<custom-property>)	margin: var(<custom-property>);
m-[<value>]	margin: <value>;
mx-<number>	margin-inline: calc(var(--spacing) * <number>);
-mx-<number>	margin-inline: calc(var(--spacing) * -<number>);

Spacing : *Margin*

Utilities for controlling an element's margin.

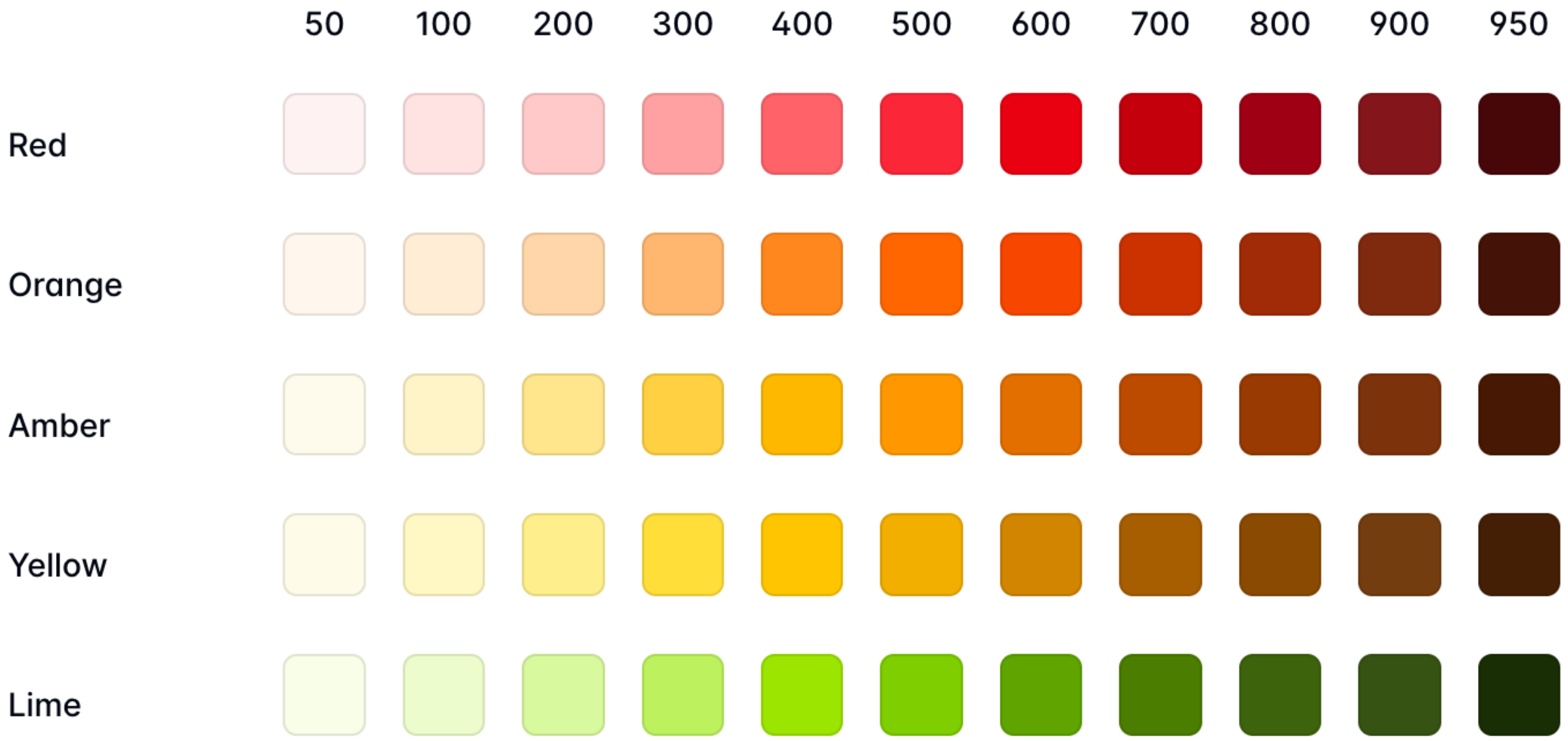
```
<div class="mt-6 ...">mt-6</div>
<div class="mr-4 ...">mr-4</div>
<div class="mb-8 ...">mb-8</div>
<div class="ml-2 ...">ml-2</div>
```



Color

Using and customizing the color palette in Tailwind CSS projects.

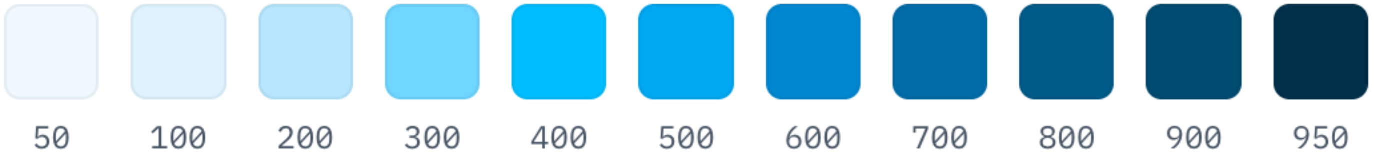
Tailwind CSS comes with a stunning color palette that's been expertly designed by talented designers. It's perfect for any design style you can imagine!



Color : Background

Using and customizing the color palette in Tailwind CSS projects.

```
<div>
  <div class="bg-sky-50"></div>
  <div class="bg-sky-100"></div>
  <div class="bg-sky-200"></div>
  <div class="bg-sky-300"></div>
  <div class="bg-sky-400"></div>
  <div class="bg-sky-500"></div>
  <div class="bg-sky-600"></div>
  <div class="bg-sky-700"></div>
  <div class="bg-sky-800"></div>
  <div class="bg-sky-900"></div>
  <div class="bg-sky-950"></div>
</div>
```



Color : Utilities

Using and customizing the color palette in Tailwind CSS projects.

Utility	Description
bg-*	Sets the **background color** of an element
text-*	Sets the **text color** of an element
decoration-*	Sets the **text decoration color** of an element
border-*	Sets the **border color** of an element
outline-*	Sets the **outline color** of an element
shadow-*	Sets the color of **box shadows**
inset-shadow-*	Sets the color of **inset box shadows**
ring-*	Sets the color of **ring shadows**
inset-ring-*	Sets the color of **inset ring shadows**
accent-*	Sets the **accent color** of form controls
caret-*	Sets the **caret color** in form controls
fill-*	Sets the **fill color** of SVG elements
stroke-*	Sets the **stroke color** of SVG elements

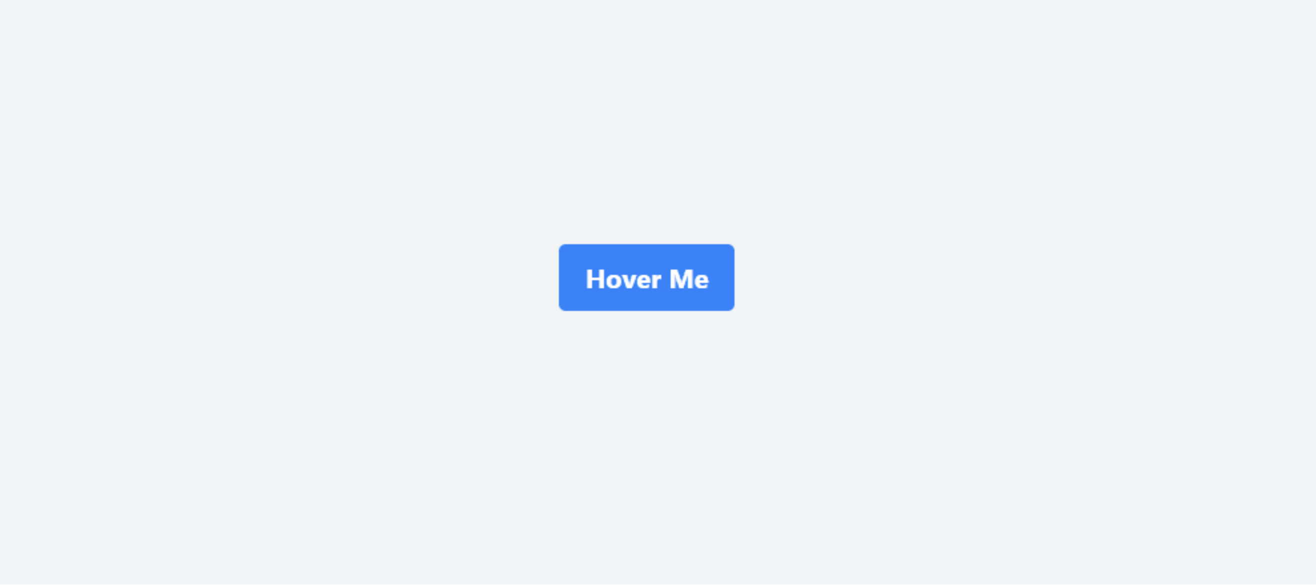
Use color utilities like *bg-white*, *border-pink-300*, and *text-gray-950* to set the different color properties of elements in your design.

Hover, focus, and other states

Using utilities to style elements on hover, focus, and more.

In Tailwind, you can apply any utility class to an element only if a certain condition is met. To do this, simply add a variant to the beginning of the class name that describes the condition you want to target.

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



Playground: https://tailwindcss.com/docs/hover-focus-and-other-states

Responsive Design

Using responsive utility variants to build adaptive user interfaces.

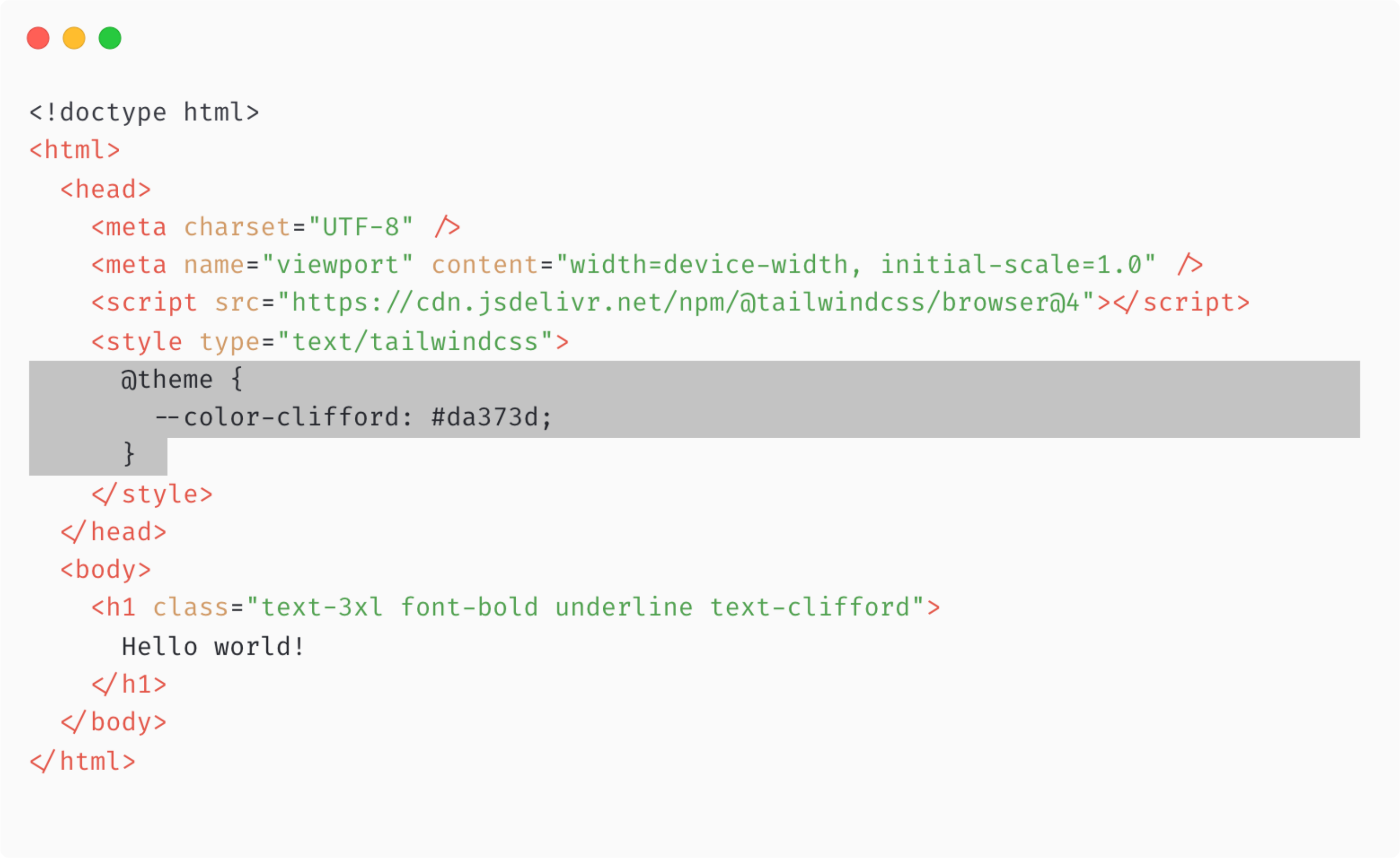
CSS makes it super easy to build responsive interfaces without ever touching your HTML. You can apply utility classes to different breakpoints, so you can create complex designs without having to write a lot of code.

```
<!-- Width of 16 by default, 32 on medium screens, and 48 on large screens -->
<img class="w-16 md:w-32 lg:w-48" src="..." />
```

Breakpoint prefix	Minimum width	CSS
`sm`	`40rem (640px)`	`@media (width >= 40rem) { ... }`
`md`	`48rem (768px)`	`@media (width >= 48rem) { ... }`
`lg`	`64rem (1024px)`	`@media (width >= 64rem) { ... }`
`xl`	`80rem (1280px)`	`@media (width >= 80rem) { ... }`
`2xl`	`96rem (1536px)`	`@media (width >= 96rem) { ... }`

Playground: https://tailwindcss.com/docs/responsive-design

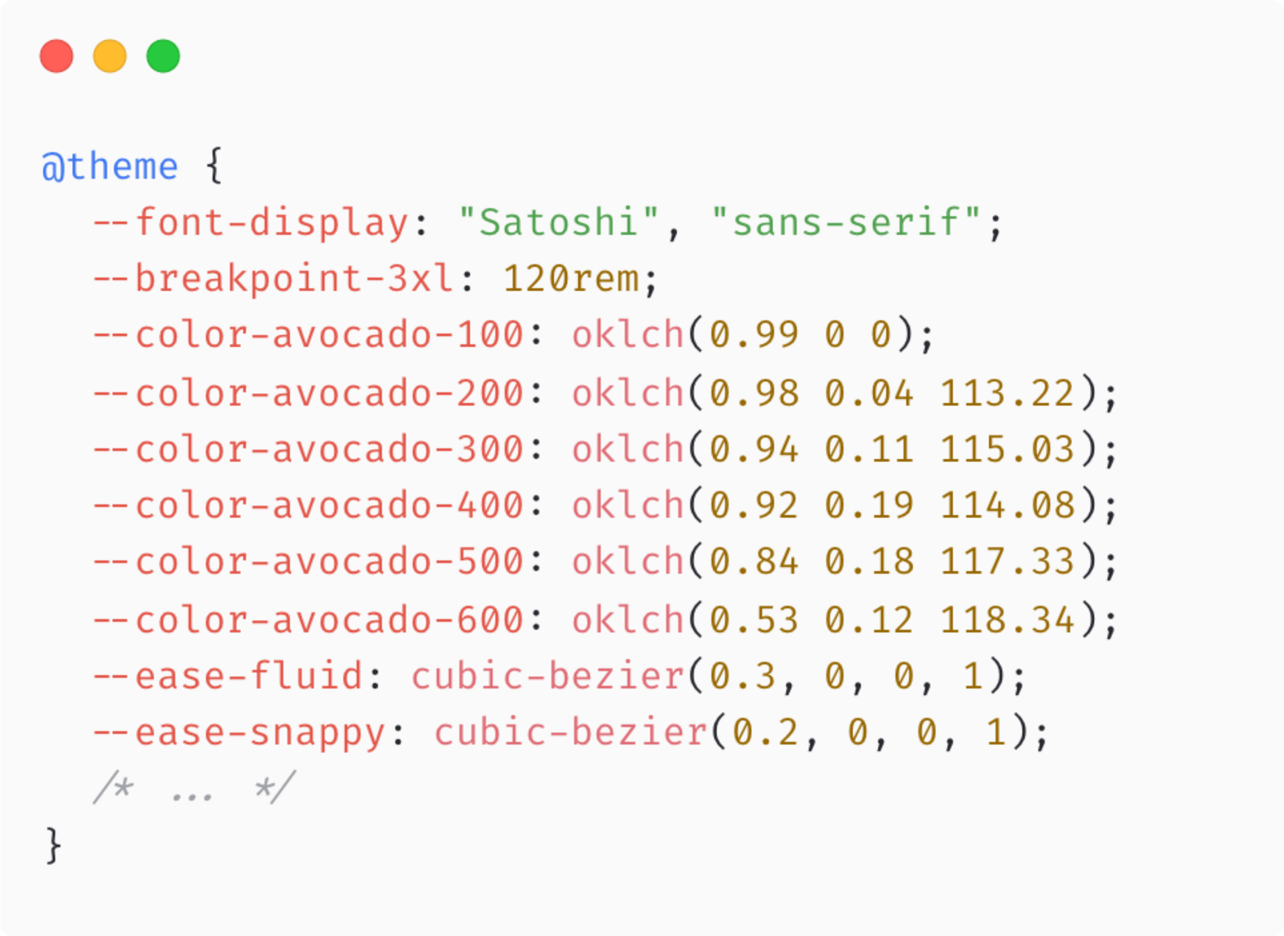
Customizing Tailwind CSS



```html
<!doctype html>
<html>
  <head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <script src="https://cdn.jsdelivr.net/npm/@tailwindcss/browser@4"></script>
    <style type="text/tailwindcss">
      @theme {
        --color-clifford: #da373d;
      }
    </style>
  </head>
  <body>
    <h1 class="text-3xl font-bold underline text-clifford">
      Hello world!
    </h1>
  </body>
</html>
```

https://tailwindcss.com/docs/adding-custom-styles#customizing-your-theme

Customizing Tailwind CSS



```
@theme {
  --font-display: "Satoshi", "sans-serif";
  --breakpoint-3xl: 120rem;
  --color-avocado-100: oklch(0.99 0 0);
  --color-avocado-200: oklch(0.98 0.04 113.22);
  --color-avocado-300: oklch(0.94 0.11 115.03);
  --color-avocado-400: oklch(0.92 0.19 114.08);
  --color-avocado-500: oklch(0.84 0.18 117.33);
  --color-avocado-600: oklch(0.53 0.12 118.34);
  --ease-fluid: cubic-bezier(0.3, 0, 0, 1);
  --ease-snappy: cubic-bezier(0.2, 0, 0, 1);
  /* ... */
}
```

→

```
<div class="bg-avocado-100"></div>
```

RESOURCES

Tailwind Playground

https://play.tailwindcss.com/

Tailwind Doc

https://tailwindcss.com/docs/styling-with-utility-classes

HyperUI

https://www.hyperui.dev/