

## Customization

# Tailwind CSS Theme

Learn how to customize the default theme.

The `@theme` directive of your `tailwind.css` file is where you define your project's color palette, type scale, fonts, breakpoints, border radius values, and more.

```
@theme {  
  --font-sans: "Graphik", sans-serif;  
  --font-serif: "Merriweather", serif;  
  
  --breakpoint-sm: 40rem;  
  --breakpoint-md: 48rem;  
  --breakpoint-lg: 64rem;  
  --breakpoint-xl: 80rem;  
  --breakpoint-2xl: 96rem;  
  
  --color-blue-400: oklch(0.707 0.165 254.624);  
  --color-purple-400: oklch(0.714 0.203 305.504);  
  --color-pink-400: oklch(0.718 0.202 349.761);  
  --color-orange-400: oklch(0.75 0.183 55.934);  
  --color-yellow-400: oklch(0.852 0.199 91.936);  
  --color-gray-400: oklch(0.707 0.022 261.325);  
  
  --radius-4xl: 2rem;  
}
```

## Theme structure

The `@theme` directive contains variables for `--breakpoint-*`, `--color-*`, and `--spacing-*`, as

On this page or each customizable core plugin.

See the [theme configuration reference](#) for a complete list of theme options.

## Breakpoints

The `--breakpoint-*` variables allows you to customize the responsive breakpoints in your project.

```
--breakpoint-sm: 40rem;  
--breakpoint-md: 48rem;  
--breakpoint-lg: 64rem;  
--breakpoint-xl: 80rem;  
--breakpoint-2xl: 96rem;
```

## Colors

The `--color-*` variables allows you to customize the global color palette for your project.

```
--color-black: #000;  
--color-white: #fff;  
  
--color-gray-50: oklch(0.985 0.002 247.839);  
...  
--color-gray-950: oklch(0.13 0.028 261.692);
```

## Spacing

The `--spacing-*` variables allows you to customize the global spacing and sizing scale for your project.

```
--spacing-px: 1px;  
--spacing-0: 0;  
...  
--spacing-96: 24rem;
```

## Customizing the theme

Out of the box, your project will automatically inherit the values from Preline UI and Tailwind CSS.

Like to customize the default theme, you have a few different options depending on your goals.

## Extending

If you'd like to preserve the default values for a theme option but also add new values, add your variables in the `@theme` directive of your configuration file.

For example, if you wanted to add an extra breakpoint but preserve the existing ones, you could extend the `--breakpoint-*` variable:

```
--breakpoint-3xl: 100rem;
```

## Overriding

To override a variable in the default theme, add your overrides directly under the `@theme` directive of your `tailwind.css`:

```
--opacity-0: 0;  
--opacity-20: 0.2;  
--opacity-40: 0.4;  
--opacity-60: 0.6;  
--opacity-80: 0.8;  
--opacity-100: 1;
```

The official [Tailwind CSS Theming](#) documentation helps you to understand the full overview of the theming options.

## Overriding default colors with custom properties

Preline UI components are designed to work out of the box with any Tailwind CSS project without requiring extra theming setup or class changes. Every component is built using pure utility classes.

Ideally, projects should be structured to support theme tokens like `text-primary-*`, so when integrating Preline components, you'd replace utility classes like `text-blue-*` with your own theme tokens. This does require some manual editing, but for larger sets of components (like entire pages), many users do global class replacements directly in their IDE, which is a pretty common approach.

A quicker solution is to override Tailwind's default colors using custom properties. For example:

```
@theme {  
  --primary-50: oklch(97.1% 0.014 343.198);  
}
```

```
--color-primary-950: oklch(28.4% 0.109 3.907);  
}  
  
:root {  
  --color-blue-50: var(--color-primary-50);  
  ...  
  --color-blue-950: var(--color-primary-950);  
}
```

This way, you don't have to modify every utility class, and components styled with `text-blue-*` will visually adopt your custom theme (e.g., pink in this example). However, the tradeoff is semantic; your HTML will still reference `blue-*`, which may not reflect the actual design system being used.

That said, we're aware of these limitations and are actively exploring more native, component-level theming approaches for future updates.