@apply directive in Tailwind CSS: Tailwind Tutorial #9

In this article, you will get a complete guide on how to use the '@apply' directive in Tailwind CSS to enhance the developer experience. You can check out the process of setting up Tailwind CSS for production by clicking here. Let's get started with the tutorial-

The '@apply' directive

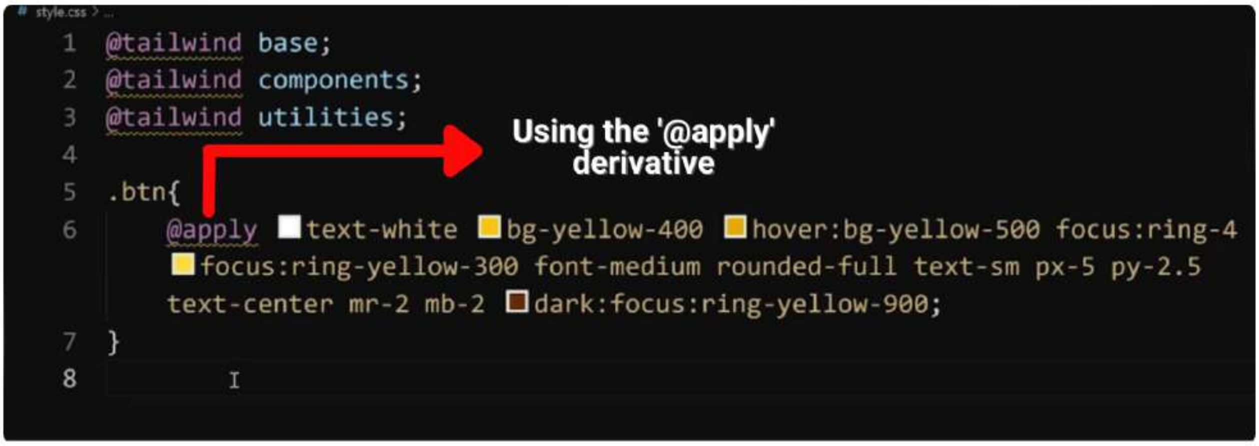
The '@apply' directive feature is used to apply a number of classes to an individual custom class. With this approach, the HTML code doesn't get bloated with a lot of classes. Instead, it would be amazing if one can extract these different classes into an external selector using tailwind. Hence, You can use the @apply directive to replace the mess of utility classes with a single custom class. Let's understand this with an example-

Why use the @apply directive? - Create Button

To understand the use of the @apply directive, You can first create and add different Tailwind Utility classes to an element such as a button. As a result, a single component uses a lot of classes and therefore to use the identical component in the project you have to use all those classes on the button element once again. This approach of replicating the element will hamper the development experience. Therefore, To overcome this issue Tailwind provides the feature of the '@apply' directive.

Using @apply directive

First of all, You need to cut all the classes of the button element and replace it with an individual custom class. For example, All the utility classes of the button are replaced by the 'btn' class. After that, You can target the created class with a selector in your global CSS and paste the utility classes inside it as shown below-



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

.btn{
    @apply text-white bg-yellow-400 hover:bg-yellow-500 focus:ring-4
    focus:ring-yellow-300 font-medium rounded-full text-sm px-5 py-2.5
    text-center mr-2 mb-2 dark:focus:ring-yellow-900;
}

```

Figure 1.1: Using the '@apply' directive

Explanation: In the above example, the custom ‘btn’ class is being targeted with the help of the selector and the Tailwind utility classes are added to it. In the above figure, You can notice that the ‘base’, ‘component’ and ‘utility’ class of tailwind CSS are placed before the custom ‘btn’ class. As a result, the custom utility classes inside the ‘btn’ class will overwrite the utility classes. Thus, the order in which the directives are arranged in the global CSS file matters a lot. Hence, A number of utility classes have been successfully added to an individual custom class. You can try out the ‘@apply’ directive for different elements and properties.