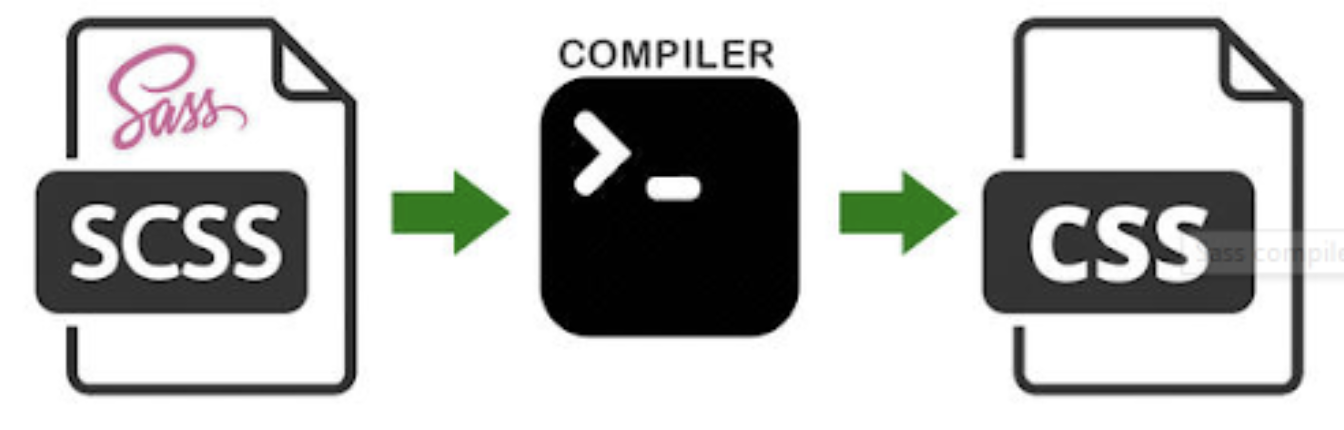
**SCSS Vs CSS, Tailwind CSS Framework 🔥**

1. **SCSS**

* It facilitates you to write clean, easy and less CSS in a program construct.
* It contains fewer codes so you can write CSS quicker.
* It has very good documentation, meaning that you can get all the required information online.
* It provides nesting so you can use nested syntax.
* It is compatible with all versions of CSS. So, you can use any available CSS libraries.
* It consists of variables that help in reusing the values throughout the CSS as many times you want.
* Syntax highlighting is a widely used CSS tool and is supported in SCSS.  
  SCSS allows you to use the existing code, and help improve its internal structure without altering the external behaviour of the code.
* It is fully CSS compatible. You can rename a CSS file as .scss extension and it will also work.  
  Due to the reason that it follows the syntax of CSS, makes it very easy to learn and work.



**2. How to write CSS in React JS?**

**1)External Styling — Normal Native CSS**

ClassName and using external file (Assets)

**2) SCSS**

<https://www.makeuseof.com/react-sass-how-use/>

@import "./Styles/variables";  
@import "./Styles/mixins";

**3) Inline CSS:**

Java script object

Disadvantages -> Reusability, Repetition, Hard coded, Maintainability

const searchButton = {  
backgroundColor : "red"  
}  
  
<button> style={searchButton} <button/>  
  
(or)  
  
style = {{backgroundColor : "red"}}

**4) Component Libraries — Material UI, Base UI, Ant UI, Chakra**

Pre-built components that are already styled.

These lib at the end of the day is NORMAL package.

You can use material UI/ Tailwind/ Bootstrap/Chakra UI/Ant UI.

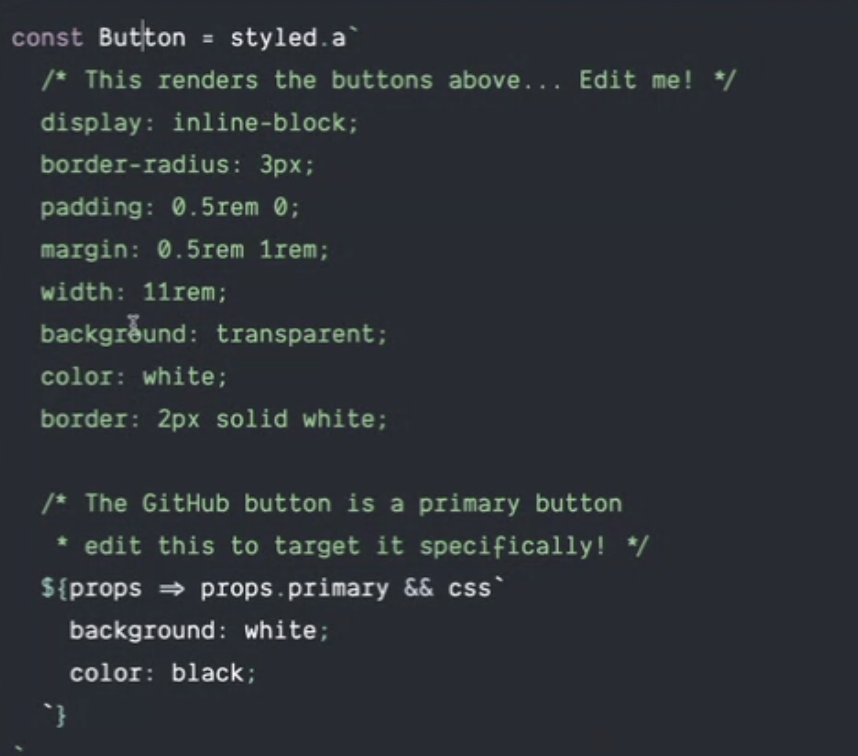
Disadvantage -> Personalized customization is hard.

**5) Styled Components**

<https://styled-components.com/>

CSS inside Javascript

Pass css as props.



**3. Tailwind CSS Framework**

**Documentation :** <https://tailwindcss.com/>

**Advantages**

* CSS on the go (in the same file)
* Reusability
* Less bundle size
* Flexible

**4. Tailwind CSS Framework Installation**

<https://tailwindcss.com/docs/guides/parcel>

CDN LINK  
  
<script src="https://cdn.tailwindcss.com/"></script>

<html lang="en">  
 <head>  
 <meta charset="UTF-8" />  
 <meta http-equiv="X-UA-Compatible" content="IE=edge" />  
 <meta name="viewport" content="width=device-width, initial-scale=1.0" />  
 <title>Namaste React</title>  
 <link rel="stylesheet" href="index.css" />  
 <script src="https://cdn.tailwindcss.com/"></script> // CDN LINK  
 </head>  
 <body>  
 <h1 class="text-2xl font-bold">Hello World</h1> // TAILWIND  
 <div id="root">Not Rendered</div>  
  
 <script type="module" src="src/App.js"></script>  
 </body>  
</html>

**5. Post CSS**

We need to tell parcel that we are using tailwind css and it needs a compilation to convert css.

npx tailwindcss init -> npx === npm run (execute directly)

/\*\* @type {import('tailwindcss').Config} \*/  
module.exports = {  
 content: [  
 "./src/\*\*/\*.{html.js}"  
 ],  
 theme: {  
 extend: {},  
 },  
 plugins: [],  
}

**6. Square bracket Notation**

className=”w-[200px]”

w-56 -> 210px (to make it exactly 200px we use **Square bracket Notation**)

**7. Pros and Cons for Tailwind**

Pros

* Bundle size is small
* Easy to debug
* Responsive
* Easy to configure and maintain
* More like inline styling
* VS Code extension -Tailwind CSS IntelliSense

Cons

* Long class
* No Reusability
* Initial learning curve (new package)