Chapter# 10: Tailwind in Action

In this Chapter we are going to study about styling our application and we are going to use tailwind CSS.

Various ways of styling component in react

* External CSS file- We write css for our components in a central place index.css.

* External SCSS- We write css for our components in a central place index. scss.

* Inline CSS- We write css inside a java script object and then use that object in component's style attribute inside JSX. (Refer previous session notes for detailed explanation)

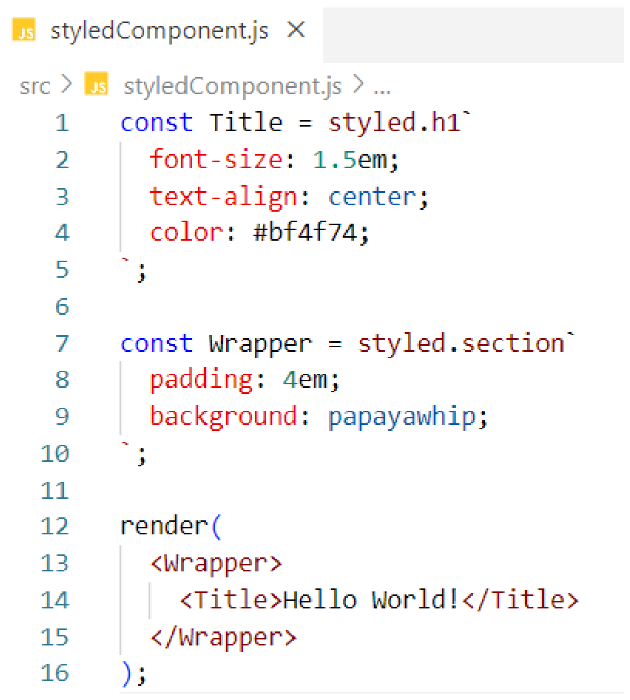
* We use styling libraries like material UI, ant UI.

* We use styled components.

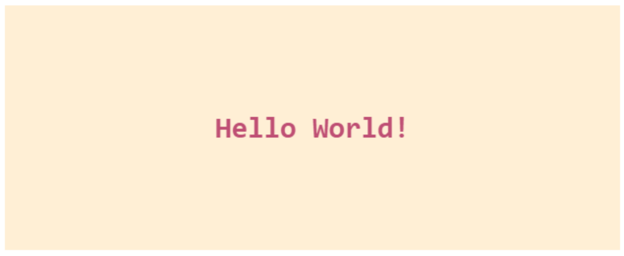
Styled Components

* Styled Components are like writing CSS like syntax inside JavaScript just like writing html like syntax inside JavaScript which is JSX.

* In below example Title and Wrapper are styled components where styling is applied inline.



```
1   const Title = styled.h1`
2     font-size: 1.5em;
3     text-align: center;
4     color: #bf4f74;
5   `;
6
7   const Wrapper = styled.section`
8     padding: 4em;
9     background: papayawhip;
10  `;
11
12  render(
13    <Wrapper>
14      <Title>Hello World!</Title>
15    </Wrapper>
16  );
```



```
Hello World!
```

Code explanation-

At Line #1 we are creating a styled Title component that'll render an <h1> tag with value Hello World with configured styles.

At Line #7, we are creating a wrapper styled component around Title styled component with configured styling

Pros and cons of using styling libraries and frameworks-

Advantages -

* Easy to use, reusable.

* Saves lot of time, development is faster.

* Gives us automatic themes.

* Gives us consistent UI (In consistent UI all the UI elements such as buttons textboxes looks the same and they should be for a great user experience. We don't want different button styles on different pages. The design should be consistent).

* It takes care of responsiveness. (provides a responsive UI, meaning UI is compatible with all the devices)

Disadvantages -

* Heavy bundle size.

* We lose a lot of control over how my design looks, for example we are forced to use matUI button all over my app and my personal customisation on this button becomes hard. Though the development process becomes very easy and fast but to customise it takes effort and time.

* Hard to integrate into heavily customised legacy front ends.

* Overly complex for simple projects.

* Updates can introduce bugs.

* Some of the features of libraries are not supported in older devices and mobile devices.

How we used to style our components?

Earlier we were maintaining a single CSS file index.css where we were writing CSS for all of our project components. This used to be a central place of writing CSS. Then we were including index.css in our main page index.html to allow styling to our application. However, this is not a good way of writing CSS. There are some major cons need to be addressed.

Disadvantages -

* No matter what the component is, we are styling everything all together in one central place i.e.index.css and this file grows in size eventually for every styling update.

* Since the file grows in size, it is very difficult to maintain and test.

* Also, css we write is not optimised.

* This way of styling impacts reusability. If we want to use styling of one component to another component, we cannot reuse.

* Hardcoded Styling.

* The job of processing inline css is heavy for our browser to understand.

To avoid all these cons Tailwind CSS comes to the rescue.

What is Tailwind CSS?

Tailwind is an open source CSS framework which enables us to write CSS on the go that means we write css in the same component JSX file just like inline styling. We don't have to toggle between CSS file and html file or component JSX file.

Perks / Why Tailwind:

* CSS on the go. Saves development time or Faster development

* Reusability.

* Less bundle sizes as it has minimal CSS/optimised CSS code being offered to us.

* Flexible and customisable which Mat UI can't provide.

* Provides lots of prebuilt classes.

* Less Code and easy to debug.

* No duplicate CSS. (Internally bundler takes care of CSS duplicity)

* provides dynamic styling using [css value]

Disadvantages of Tailwind:

* It takes initial learning curve. It's a new concept to learn for new developers.

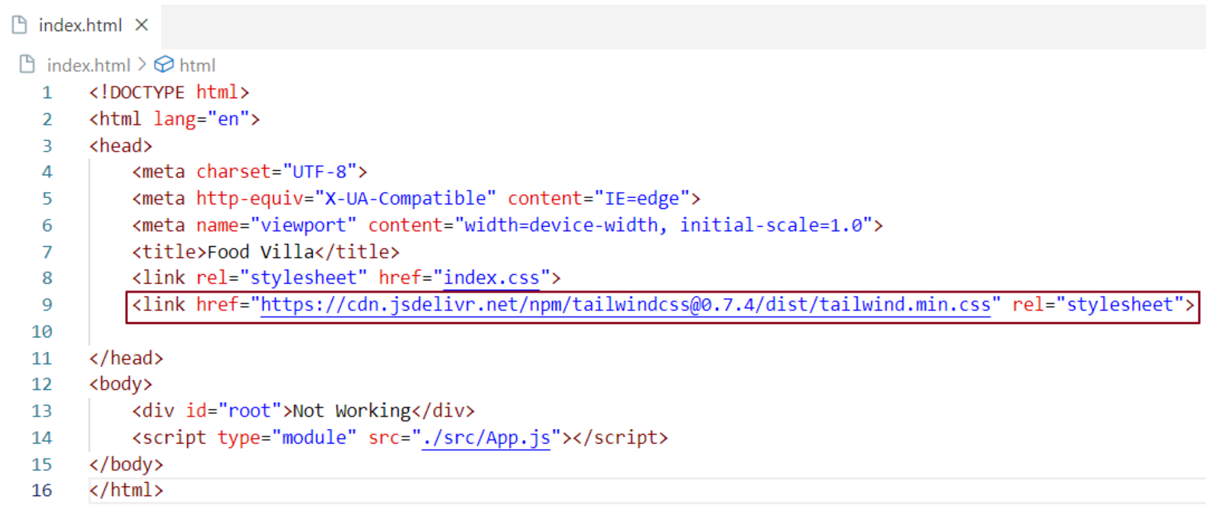
* It affects code readability. Because if we want to design an element we provide so many classes to style that element which make our code messy.

Note:

Inline styling and other ways of stylings has their drawbacks. Tailwind inherited only the advantageous features from these stylings. For instance, CSS on the go is the property which Tail wind inherited from inline styling for Faster development.

How to include Tailwind into our project?

Way -1 - Use CDN link to allow tailwind take control over our UI tags and controls.



```html
<!DOCTYPE html>
<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>Food Villa</title>
    <link rel="stylesheet" href="index.css">
    <link href="https://cdn.jsdelivr.net/npm/tailwindcss@0.7.4/dist/tailwind.min.css" rel="stylesheet">

</head>
<body>
    <div id="root">Not Working</div>
    <script type="module" src="./src/App.js"></script>
</body>
</html>
```

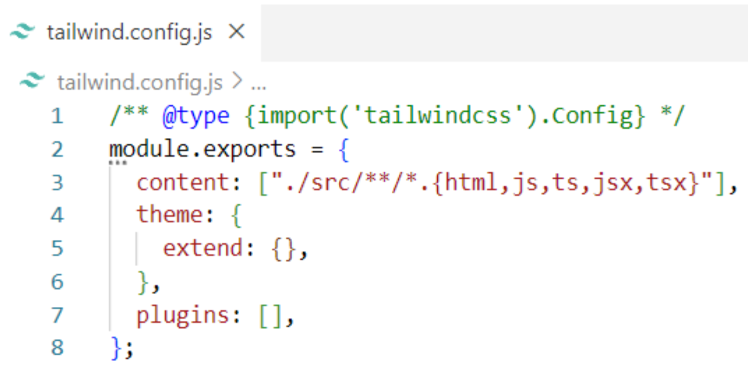
Way 2 -Tailwind installation with parcel: https://tailwindcss.com/docs/guides/parcel

1.Install Tailwind css using cmd npm install -D tailwindcss

2.Once tailwind installation is Done we install postcss using command npm install -D postcss

Why postcss? because we need to tell parcel that we are using tailwind so that parcel compiles tailwind class properties into browser understandable css code.

3. Configure Tailwind and to do that we execute a command npx tailwindcss init. This will create a config file for tailwind i.e. tailwind.config.js



tailwind.config.js ×

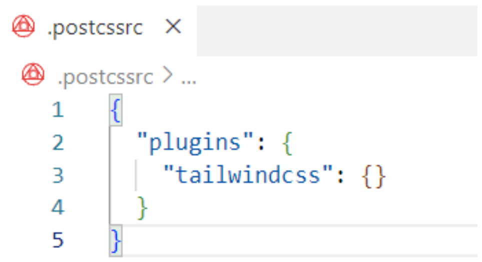
tailwind.config.js > ...

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

In this configuration content denotes what file will be using tailwind classes

4. create a new file named .postcssrc in project root directory

. postcssrc file takes below configuration and tells parcel that in our project we would be seeing a lot of tailwind classes, so when we are bundling things up, compile tailwind css into normal css.

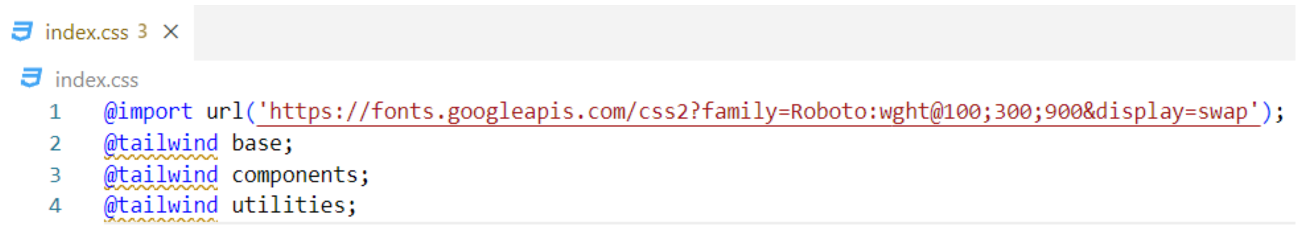


.postcssrc ×

.postcssrc > ...

```json
{
  "plugins": {
    "tailwindcss": {}
  }
}
```

5.Add these below 3 lines to index.css. why because When parcel reads index.css it exactly knows that tailwind is being used at base components and utilities in our project.



index.css 3 ×

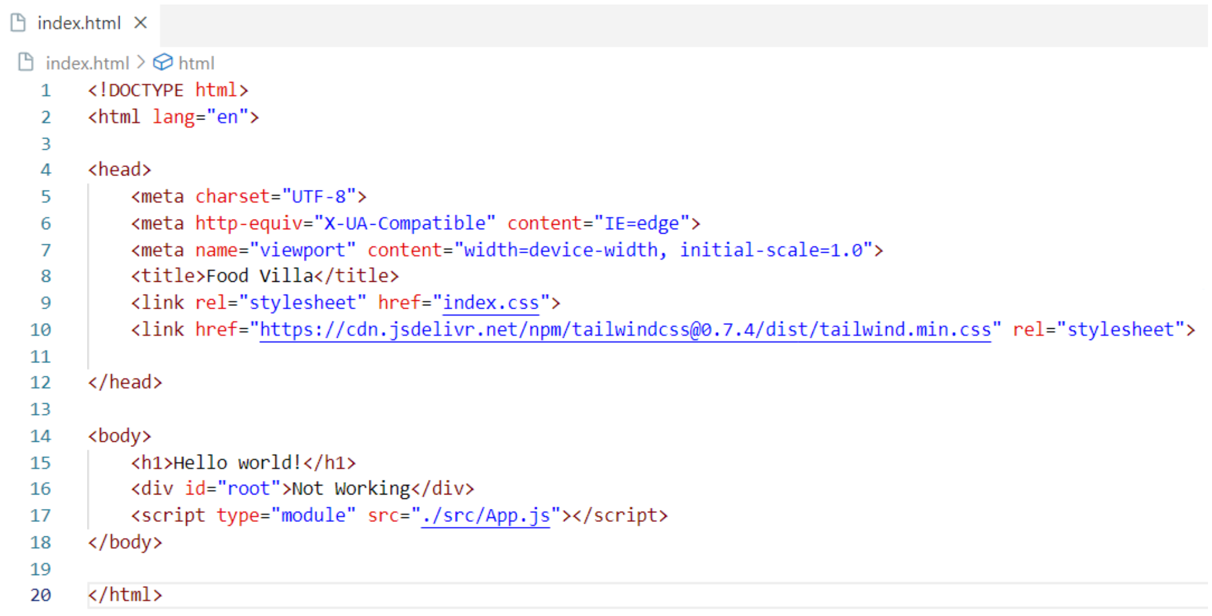
index.css

```css
@import url('https://fonts.googleapis.com/css2?family=Roboto:wght@100;300;900&display=swap');
@tailwind base;
@tailwind components;
@tailwind utilities;
```

We are done, now we can use tailwind classes across our components

Tailwind Controlling the UI elements

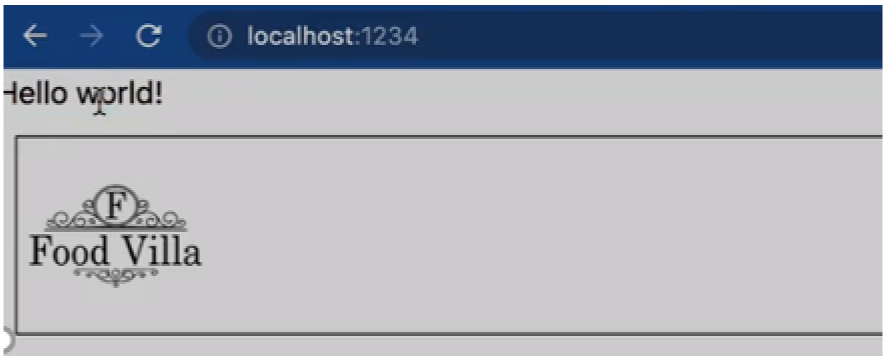
I have included my CDN for Tailwind and added an h1 tag with name Hello world inside body.



index.html

```html
<!DOCTYPE html>
<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>Food Villa</title>
    <link rel="stylesheet" href="index.css">
    <link href="https://cdn.jsdelivr.net/npm/tailwindcss@0.7.4/dist/tailwind.min.css" rel="stylesheet">

</head>

<body>
    <h1>Hello world!</h1>
    <div id="root">Not Working</div>
    <script type="module" src="./src/App.js"></script>
</body>

</html>
```

This h1 tag is controlled by tailwind. Meaning Tailwind is overriding the default css styling of html h1 element. Tailwind affects styling across the react component JSX where tailwind classes are being used.



Resources and Home Work

Styled Component documentation – https://styled-components.com/docs/basics

Tailwind documentation: https://v2.tailwindcss.com/docs

VS Extension for Tailwind: Tailwind css Intellisense

HW: Make you app looks good using tailwind classes.