TAILWIND CSS

* Why should we use CSS Frameworks ?

  ① To write optimised CSS
  ② It saves time

* One way of writing CSS is the basic way :-

  ⟶ We have an 'index.html' file and inside it we give the link to 'index.css' file :-

  <link rel = "stylesheet" href = "index.css" />

* Another way is :- SCSS

  'Syntactically Awesome Style Sheets'

  At last scss is converted to CSS (basic).

[Like JSX is htmltags at EOD].

* Third way of writing css is :- 'Inline css'

Eg :-

```
<button style = { Btncss } > Search </button
```

```
const Btncss = {
    background-color : "red" ;
}.
```

→ a javascript object.

[OR]

```
<button style = {{
        background-color : "red" ,
        }} > Search </button>
```

* This is not the good way of writing css. :-

(1) Difficult to maintain
(2) cannot reuse
(3) hard-coded.
(4) heavy job for browsers to processes inline css.

* Fourth way of writing CSS :-

→ Using libraries like 'material UI', 'chakra ui'

→ MUI is maintained by google.

→ newer & faster way.

PROS & CONS of using CSS frameworks & libraries.

PROS

- ☑ Rapid development
  → provides pre-built components
- ☑ Responsiveness
- ☑ Consistent design
- ☑ Browser compatibility
- ☑ Easy to use
- ☑ Reusability
- ☑ Saves time

CONS

- ☒ Bundle size - making it heavy
- ☒ Learning curve
- ☒ Limited customization
- ☒ Dependency management

* Fifth way is : 'Styled-components'

→ easier to pickup & less used in industry.

* Next way is :- TAILWIND CSS

→ Tailwind css is a open-source framework.

→ We can write css on the go.

→ Reusability :-

Tailwind css comes with pre-built classes.

→ Less-bundle size. (as it is minimal css)

→ can build flexible UI (means very much customizable).

CONFIGURATION

→ go to tailwindcss.com/docs/installation

* Add CDN script to HTML :

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

When we use tailwindcss, it changes default behaviour of the tags. It removes basic styles.

* Everything in tailwind works with classname

Install tailwindcss

→ Write this command in terminal :-

```
npm install -D tailwindcss postcss
```

* postcss :- tool that transform CSS with JS.
  → used for compilation of these classes

→ npx tailwindcss init → to initialise tailwind in our project.

→ We will get one file called "tailwind.config.js"

```
module.exports = {
    content: [],
    theme: {
        extend: {},
    },
    plugins: [],
}
```

→ What files will be using tailwind classes

→ "postcssrc" file

→ We need to tell bundler that "while you are bundling things up (while you're building dev/prod build), we will be using tailwind css. So, please compile it "to normal css".

This is why we use 'postcssrc' file.

→ "index.css" file

→ there would be no css inside this file. but these 3 lines.

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

→ Why we add the '@tailwind' directives to index.css file ?

- to include tailwind's utility classes and apply the styles defined by tailwind to the HTML elements.

→ Layer Seperation

* Tailwindcss is organized into layers :-
  - base
  - components
  - utilities
* The '@tailwind' directives help to include the styles from each layer in correct order.
* Base layer : includes styles like typograph box-sizing and resets.
* Component layer : contains pre-designed components like buttons, cards etc.
* Utility layer : provide utility classes for common styles like spacing, positioning, color and more.

In vscode, use "Tailwind CSS IntelliSense" extension

Responsive Design

* There are 5 breakpoints by default.

① 'sm' – 640px

② 'md' – 768px

③ 'lg' – 1024px

④ 'xl' – 1280px

⑤ '2xl' – 1536px

minimum-width

* To add a utility, but only have it take effect at a certain breakpoint, all you need is to prefix the utility with breakpoint name

Eg:-

```
<h1 class="p-2 sm:p-4 lg:p-8">
    Hai, my name is Ashraya
</h1>
```