

Full Stack Web Development

CSS Framework

- CSS Framework
- CSS Component Library
- Chakra UI
- Installation
- Provider Setup
- Usage

What is CSS Framework?

CSS frameworks are tools used by UI developers to make their work easier. Instead of having to manually style every time a new project comes up

Frameworks give developers the convenience of quickly setting up user interfaces that can be changed and repeated across projects instead of wasting time starting from a blank document.



Why using CSS Frameworks?

Advantages of using CSS Frameworks

- Developers and designers can use CSS frameworks to implement various advanced features and visual elements on a website – forms, different buttons, navbars, breadcrumbs, and even clean symmetrical layouts.
- CSS frameworks make it simple to create websites compatible with multiple browsers and browser versions. This reduces the likelihood of bugs popping up during cross browser testing.
- Since these frameworks have ready-to-use stylesheets in place, using them allows faster and more convenient web development. Users don't have to dive deep into CSS code to accomplish required tasks.
- Developers can quickly generate a user-friendly and visually appealing UI that can be modified throughout a project without starting from scratch.

Most Common Used CSS Framework


- [Tailwind](#)
- [Bootstrap](#)
- [Bulma](#)
- [Foundation](#)



BULMA



CSS Framework – Tailwind CSS



```
// create react app
npx create-react-app simple-app

// go to project
cd simple-app

//install tailwind css
npm install -D tailwindcss

//initiate tailwind css
npx tailwindcss init
```

Install tailwindcss via npm, and create your tailwind.config.js file by following this command on the left side.

Find out more in [here](#)

Tailwind CSS Setup

```
tailwind.config.js


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

Add the paths to all of your template files in your tailwind.config.js file.

Find out more in [here](#)

Tailwind CSS Setup

Add the `@tailwind` directives for each of Tailwind's layers to your `./src/index.css` file.



index.css

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

Find out more in [here](#)

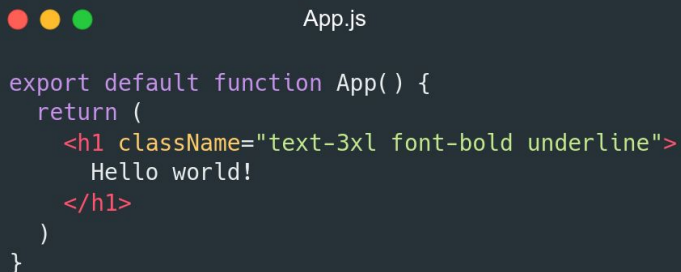
Tailwind CSS – How to use



```
npm run start
```

Run your build process with `npm run start`.

Start using Tailwind's utility classes to style your content.



```
App.js
export default function App() {
  return (
    <h1 className="text-3xl font-bold underline">
      Hello world!
    </h1>
  )
}
```

Hello world!

Find out more in [here](#)

Tailwind Extension for VSCode

- Tailwind CSS IntelliSense

Search on your sidebar in extension tab and find out this cool tailwind extension



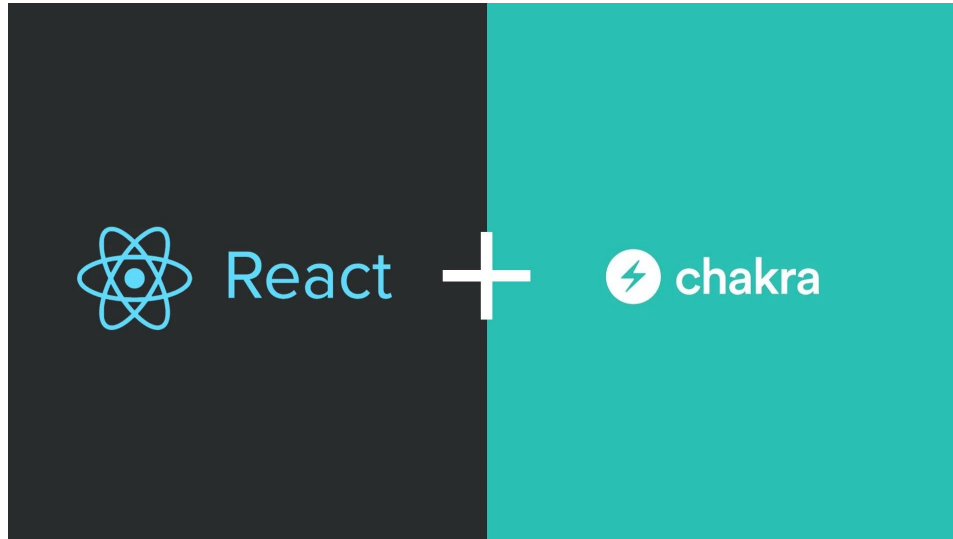
If you're using a component-based JavaScript framework like React, you might be interested in a component library instead of a CSS framework.

Every React application needs a base set of components to start from, these are then combined in different ways to create more powerful components. You can build the base components yourself.

- [Chakra](#)
- [Reactstrap](#)
- [material-ui](#)



CSS Framework we are going to discuss is Chakra UI. Chakra UI is a simple, modular and accessible component library that gives you the building blocks you need to build your React applications.



Inside your CRA project directory, install Chakra UI by running either of the following:

```
npm i @chakra-ui/react @emotion/react@^11 @emotion/styled@^11 framer-motion@^6
```

Provider Setup

After installing Chakra UI, you need to set up the ChakraProvider at the root of your application. This can be either in your index.jsx or index.tsx

Put in the following code:

```
import * as React from "react";

// 1. import `ChakraProvider` component
import { ChakraProvider } from "@chakra-ui/react";

function App() {
  // 2. Wrap ChakraProvider at the root of your app
  return (
    <ChakraProvider>
      <App />
    </ChakraProvider>
  );
}
```

Provider Setup

ChakraProvider Props:

Name	Type	Default	Description
resetCSS	boolean	true	automatically includes <code><CSSReset /></code>
theme	Theme	@chakra-ui/theme	optional custom theme
colorModeManager	StorageManager	localStorageManager	manager to persist a users color mode preference in
portalZIndex	number	undefined	common z-index to use for <code>Portal</code>

Usage

For example we want to use button component from Chakra UI, first import the button component like this:

```
import { Button, ButtonGroup } from "@chakra-ui/react";
```

Then use it in our component like this:

```
<Button colorScheme="lightblue">Button</Button>
```



Thank You!

Reference:

1. <https://chakra-ui.com/>

