React CSS

CSS in React is used to style the React App or Component. The **style** attribute is the most used attribute for styling in React applications, which adds dynamically-computed styles at render time. It accepts a JavaScript object in **camelCased** properties rather than a CSS string. There are many ways available to add styling to your React App or Component with CSS. Here, we are going to discuss mainly **four** ways to style React Components, which are given below:

1. Inline Styling
2. CSS Stylesheet
3. CSS Module
4. Styled Components

1. Inline Styling

The inline styles are specified with a JavaScript object in camelCase version of the style name. Its value is the style?s value, which we usually take in a string.

Example

**App.js**

**import** React from 'react';

**import** ReactDOM from 'react-dom';

**class** App **extends** React.Component {

  render() {

**return** (

      <div>

      <h1 style={{color: "Green"}}>Hello React</h1>

      </div>

    );

  }

}

export **default** App;

**Note:** You can see in the above example, we have used two curly braces in:  
**<h1 style={{color: "Green"}}>Hello JavaTpoint!</h1>**.  
It is because, in JSX, JavaScript expressions are written inside curly braces, and JavaScript objects also use curly braces, so the above styling is written inside two sets of curly braces {{}}.

camelCase Property Name

If the properties have two names, like **backgroundColor**, it must be written in camel case syntax.

**Example**

**App.js**

1. **import** React from 'react';
2. **import** ReactDOM from 'react-dom';
4. **class** App **extends** React.Component {
5. render() {
6. **return** (
7. <div>
8. <h1 style={{color: "Red"}}>Hello JavaTpoint!</h1>
9. <p style={{backgroundColor: "lightgreen"}}>Here, you can find all CS tutorials.</p>
10. </div>
11. );
12. }
13. }
14. export **default** App;

Using JavaScript Object

The inline styling also allows us to create an object with styling information and refer it in the style attribute.

**Example**

**App.js**

1. **import** React from 'react';
2. **import** ReactDOM from 'react-dom';
4. **class** App **extends** React.Component {
5. render() {
6. **const** mystyle = {
7. color: "Green",
8. backgroundColor: "lightBlue",
9. padding: "10px",
10. fontFamily: "Arial"
11. };
12. **return** (
13. <div>
14. <h1 style={mystyle}>Hello JavaTpoint</h1>
15. <p>Here, you can find all CS tutorials.</p>
16. </div>
17. );
18. }
19. }
20. export **default** App;

2. CSS Stylesheet

You can write styling in a separate file for your React application, and save the file with a .css extension. Now, you can **import** this file in your application.

Example

**App.js**

1. **import** React from 'react';
2. **import** ReactDOM from 'react-dom';
3. **import** './App.css';
5. **class** App **extends** React.Component {
6. render() {
7. **return** (
8. <div>
9. <h1>Hello React</h1>
10. <p>Here, you can find all CS tutorials.</p>
11. </div>
12. );
13. }
14. }
15. export **default** App;

**App.css**

1. body {
2. background-color: #008080;
3. color: yellow;
4. padding: 40px;
5. font-family: Arial;
6. text-align: center;
7. }

**Index.html**

1. <!DOCTYPE html>
2. <html lang="en">
3. <head>
4. <meta charset="utf-8" />
5. <meta name="viewport"
6. content="width=device-width, initial-scale=1" />
7. <title>React App</title>
8. </head>
9. <body>
10. <div id="app"></div>
11. </body>
12. </html>