21. Styling in React: CSS, Modules, Inline Styles & Styled Components

Last updated by | Subramanya Dixit | May 5, 2025 at 8:56 PM GMT+5:30

📘 Why Styling Matters in React

Styling determines how your application looks and feels. React supports multiple styling techniques, and choosing the right one can impact **modularity**, **scalability**, and **developer experience**.

🎨 Styling Techniques in React

✅ 1. Inline Styles

~~~
Inline styles are defined directly inside JSX using a JavaScript object.

```jsx
function Box() {
  return (
    <div style={{ backgroundColor: 'skyblue', padding: '10px' }}>
      Inline Styled Box
    </div>
  );
}
~~~

🟡 Use for: Quick, dynamic styling
🔴 Avoid for: Complex or reusable styles

✅ 2. External CSS

You can import a `.css` file and use class names.
App.css

```
.title {
  font-size: 24px;
  color: navy;
}
```

App.jsx

```
import './App.css';

function App() {
  return <h1 className="title">Styled with CSS</h1>;
}
```

🟢 Use for: General styling
🔴 Watch out for: Global scope pollution

✅ 3. CSS Modules

CSS Modules scope class names locally to avoid conflicts.
App.module.css

```
.button {
  background: green;
  color: white;
}
```

App.jsx

```
import styles from './App.module.css';

function Button() {
  return <button className={styles.button}>Click Me</button>;
}
```

🟢 Use for: Component-specific styles
✅ Encouraged in large React apps

✅ 4. Styled Components (CSS-in-JS)

Styled-components use tagged template literals for styling React components.

```
npm install styled-components
```

```
import styled from 'styled-components';

const Title = styled.h1`
  font-size: 2rem;
  color: tomato;
`;

function App() {
  return <Title>Hello Styled!</Title>;
}
```

🟢 Use for: Theming, dynamic styles, scoped styling
✅ Works great with component logic and props

🌐 Comparison Table

Technique	Scoped	Dynamic	Setup Needed	Use Case
Inline Styles	Yes	✅ Yes	❌ No	Quick dynamic styles
External CSS	❌ No	❌ No	❌ No	Simple global styles
CSS Modules	✅ Yes	❌ No	❌ No	Local styles in large apps
Styled Components	✅ Yes	✅ Yes	✅ Yes	Dynamic theming, large apps

---

📊 Visual Overview

```
graph TD
  A[Styling in React] --> B[Inline Styles]
  A --> C[External CSS]
  A --> D[CSS Modules]
  A --> E[Styled Components]
```

💡 Best Practices

- Prefer **CSS Modules** or **Styled Components** in large apps.
- Avoid inline styles for layout-heavy or reusable elements.
- Use **BEM** or other conventions if sticking with external CSS.

📝 Practice Exercises

1. Style a button with all four methods.
2. Convert a component from external CSS to CSS Module.
3. Use props in a `styled-component` to toggle background color.

❓ Quiz Questions

1. Which styling method supports scoping styles by default?
a) External CSS
b) Inline styles
✅ **c) CSS Modules**
d) All of the above

---

2. What is a limitation of inline styles in React?
a) Not supported in JSX
b) Slower rendering
✅ **c) No support for pseudo-selectors or media queries**
d) Requires styled-components

---

3. What is true about Styled Components?
a) They modify global CSS
b) They are limited to static styles
✅ **c) They allow dynamic styling using props**
d) They don't support theming

---

4. What does `styles.button` refer to in CSS Modules?
a) A JavaScript method
✅ **b) A locally scoped class name**
c) A React hook
d) A DOM element
