WEB TECHNOLOGIES

React Component Styling

Prof. Pavan A C

Department of Computer Science and Engineering

Acknowledgement:

Teaching Assistants(Harini B and Chandana MS)

React Component Styling

Overview & Why It Matters

Definition: Styling in React refers to the various techniques used to apply visual design and presentation rules to individual components.

Importance: Effective styling improves both **user experience** and **code maintainability**, ensuring applications are consistent, accessible, and scalable.

Context: Unlike traditional web development, React encourages **component-scoped styling**, where each component manages its own appearance, reducing conflicts and improving modularity.

Why learn this: Choosing the right styling strategy can optimize performance, accelerate development, and enhance collaboration in larger projects.

Traditional Styling with CSS Files

External CSS (or CSS Modules)

- **How it works:** Create .css files and import them. Example with CSS Modules:

```
import styles from './Button.module.css';
<button className={styles.primary}>Click Me</button>
```

- **Why use it:** Separation of concerns and scoped styles - no global naming mess.
- **Tip:** Keeps responsiveness manageable and clean by component splitting.
- **Analogy:** Like school uniforms – neat, consistent, no one fights over colors.

Inline Styles

Inline Styling - Quick but Dirty?

- **How it works:**

```
<div style={{ color: 'red', fontSize: '20px' }}>Hello Style!</div>
```

- **Pros:** Good for dynamic, one-off styles.
- **Cons:** Harder to maintain; lacks pseudo-class support (like `:hover`).
- **Analogy:** Like asking for ‘extra masala’ at a street food stall - quick, works for one plate, but not scalable for the whole restaurant.

CSS-in-JS (Styled Components & Beyond)

When CSS Meets JS—Introductions to CSS-in-JS

- **Concept:** Styles are defined in JS, using libraries like `styled-components`, `Emotion`, or `JSS`.
- **Example with styled-components:**

```
const Button = styled.button`
background: ${props => props.primary ? 'blue' : 'gray'};
color: white;
`;
```

- **Why it's cool:** Encapsulation, theming, and no style collisions.
- **Analogy:** Like Swiggy Instamart - everything (groceries + spices) delivered together in one bag. Here, logic + styles live in one file.

Styling Utilities & Frameworks

Utility-First & UI Libraries — Tailwind, Chakra, etc.

- **TailwindCSS:** Write utility classes that compose styles - fast and modifiable.
- **Component Libraries:** Use ready-made components (Material UI, Chakra, Radix, Ant Design) with styling baked in and themeable.
- **Analogy:**
    - **(Utility-first)**: Like using Zomato filters - add ‘veg only’, ‘under ₹300’, ‘4+ rating’ - you stack utilities until you get exactly what you want.
    - **UI Libraries:** Like ordering from Domino’s - ready-made, consistent taste, you just customize toppings (themes).

Modern Best Practices

- **Hooks + CSS-in-JS + Feature Structure:** Keep your styling logic near code logic; modular and feature-aligned.
- **Use CSS Modules or CSS-in-JS** for isolation and maintainability.
- **Component-Driven Styles:** Each component owns its styles - clear and reusable.
- **Analogy:** Like splitting Netflix accounts - everyone has their own profile (scoped styles), but the subscription (project) stays organized.

React 19 & <style> Handling

React's Built-in Style Optimization

Behavior: React can optimize `<style>` tags:

- Moves them to `<head>`, de-duplicates duplicates.
- Honors `href` and `precedence` props to avoid conflicts.

Why care: Better performance and clearer override rules - no more random CSS chaos mid-app.

React Component Styling

Example - 1

```
function InlineStyled() {
  return (
    <div>
      <h1 style={{ color: "blue", fontSize: "24px" }}>
        Hello with Inline Style
      </h1>
          <p  style={{  backgroundColor:  "lightyellow",
padding: "10px" }}>
        This paragraph is styled using inline CSS in JSX.
      </p>
    </div>
  );
}
```

```
ReactDOM.createRoot(document.
getElementById("root")).render(
  <InlineStyled />
);
```

React Component Styling

Example - 2

```
import React from "react";
import ReactDOM from "react-dom/client";
import "./index.css"; // Import CSS file

function InternalStyled() {
  return (
    <div>
        <h1 className="heading">Hello with Internal CSS</h1>
      <p className="paragraph">
       This paragraph is styled using an external CSS file.
      </p>
    </div>
  );
}

ReactDOM.createRoot(document.getElementById("root")).render(
  <InternalStyled />
);
```

Note

This is just main.jsx

// src/main.jsx

CSS file is also required

React Component Styling

Example - 2 (Continued)

//src/index.css

```
.heading {
  color: green;
  font-size: 26px;
  text-align: center;
}

.paragraph {
  background-color: lightgray;
  padding: 12px;
  border-radius: 5px;
}
```

Picking the Right Strategy

Which Styling Option is *You*?

Scenario	Best Option	Why It's Lit
Simple UI, small project	CSS Modules or traditional CSS	Fast to set up, minimal overhead
Thematic, large app	CSS-in-JS (styled-components / Emotion)	Scoped, dynamic, easier theme changes
Utility-first fan	Tailwind + component libraries	Speedy, consistent, dev-friendly
Built for enterprise	UI libraries like MUI, Chakra, Radix	Accessibility, aesthetics, ready-made polish

MCQ

Q1. Which React styling method uses JavaScript template literals to define styles?

a) Inline CSS

b) Styled Components

c) CSS Modules

d) Tailwind CSS

Answer: b) Styled Components

Q2. Tailwind CSS follows which styling philosophy?

a) Component-driven

b) Global styles

c) Utility-first

d) Inline

Answer: c) Utility-first

MCQ

Q3. One key advantage of CSS Modules is:

a) Styles are global and reusable everywhere

b) Styles are scoped locally to a component

c) Styles are written only in JavaScript

d) Supports only class-based components

Answer: b) Styles are scoped locally to a component

References

- React Official Docs – <style> component
- W3Schools – React CSS Styling
- Styled Components Official Docs
- Wikipedia – CSS-in-JS
- FreeCodeCamp – How to Style React Components
- Prismic – React Component Libraries
- Medium – React Best Practices 2025
- Dev.to – React Components 2025 Guide

THANK YOU

Prof. Pavan A C

Department of Computer Science and Engineering