**Understanding the Need for Styling React Components**

Styling is essential in React for the same reasons it is in regular HTML: to make applications visually appealing and usable. But React introduces component-based architecture, so styling must adapt accordingly.

**Key Reasons:**

1. **Encapsulation**: Each component should manage its own styles, avoiding global CSS conflicts.
2. **Reusability**: Styled components can be reused across the application.
3. **Scoped Styles**: Prevents style leakage and clashes by scoping styles to specific components.
4. **Dynamic Styling**: React allows conditional and inline styling based on state or props.

**Working with CSS Modules and Inline Styles**

A CSS Module is a CSS file in which all class and animation names are scoped locally by default.

**How to Use:**

1. Create a CSS file with the extension: ComponentName.module.css
2. Import it in your component : import styles from './Button.module.css';
3. Use it: <button className={styles.primary}>Click Me</button>

Summary

| **Feature** | **CSS Modules** | **Inline Styles** |
| --- | --- | --- |
| Scope | Local (Scoped) | Local (Inline) |
| Syntax | CSS File | JavaScript Object |
| Pseudo-selectors | ✅ Supported | ❌ Not Supported |
| Media Queries | ✅ Supported | ❌ Not Supported |
| Use Case | Component Styling | Dynamic Styles |