In React, `if` statements are not typically used in the same way as in traditional JavaScript programming due to React's declarative nature. Instead, conditional rendering is often accomplished using JavaScript logical operators like `&&` or the ternary operator `? :`. Here's how you can achieve conditional rendering in React:

1. Using the `&&` operator:
```jsx
function MyComponent({ condition }) {
  return (
    <div>
      {condition && <p>Condition is true</p>}
    </div>
  );
}
```

2. Using the ternary operator:
```jsx
function MyComponent({ condition }) {
  return (
    <div>
      {condition ? <p>Condition is true</p> : <p>Condition is false</p>}
    </div>
  );
}
```

3. Using `if` statements inside the render method (less common):
```jsx
function MyComponent({ condition }) {
  if (condition) {
    return <p>Condition is true</p>;
  } else {
    return <p>Condition is false</p>;
  }
}
```

However, it's important to note that you cannot directly use `if` statements outside the `return` statement in functional components. You can use them within the functional component's body, but only for logic, not for rendering components directly.

Sure, here are three examples of using `if` statements in React:

1. **Conditional rendering within a component function:**
```jsx
function MyComponent({ condition }) {
  let message;
  if (condition) {
    message = <p>Condition is true</p>;
  } else {
    message = <p>Condition is false</p>;
  }

  return (
    <div>
      {message}
    </div>
  );
}
```

2. **Using `if` statement inside `render()` method in a class component:**
```jsx
class MyComponent extends React.Component {
  render() {
    let message;
    if (this.props.condition) {
      message = <p>Condition is true</p>;
    } else {
      message = <p>Condition is false</p>;
    }

    return (
      <div>
        {message}
      </div>
    );
  }
}
```

3. **Using `if` statement for conditional logic within JSX (less common):**
```jsx
function MyComponent({ condition }) {
  return (
    <div>
      {(() => {
```

```
        if (condition) {
          return <p>Condition is true</p>;
        } else {
          return <p>Condition is false</p>;
        }
      })()}
    </div>
  );
}
```

While the first two examples are more common and idiomatic in React, the third example demonstrates using an immediately-invoked function expression (IIFE) to achieve conditional rendering within JSX, though this approach is less common and can make the code less readable.