React Quick Revision Cheat Sheet (with Examples)

JSX & Rendering

- Write UI with JSX (looks like HTML).
- Use {} for embedding JS expressions.
- Lists require unique 'key' prop.
- Conditional rendering: {condition && <Component />} or {condition ? A : B}.

Example:

```
// Example: Writing components with JSX
function Greeting() {
  const name = "Gladstone";
  return <h1>Hello, {name} ■</h1>;
}

// Example: Conditional rendering (? : and &&)
function LoginStatus({ isLoggedIn }) {
  return (
    <div>
      <h2>{isLoggedIn ? "Welcome back! ■" : "Please log in."}</h2>
      {isLoggedIn && <p>You have 3 new notifications.</p>}
    </div>
  );
}

// Example: Lists and key props
function ShoppingList() {
  const items = ["Apples", "Bananas", "Cherries"];
  return (
    <ul>
      {items.map((item, index) => (
        <li key={index}>{item}</li>
      ))}
    </ul>
  );
}
```

Components

- Functional components are the standard.
- Pass data via props.
- Compose components (nest them).

State Management (Hooks)

- useState: local state.
- useEffect: side effects (API calls, subscriptions).
- useRef: store mutable values, DOM refs.
- useContext: simple global state.

Events & Forms

- Events: onClick, onChange, etc.
- Controlled inputs: value + onChange tied to state.
- Forms: preventDefault() on submit.

Routing (react-router-dom)

- <BrowserRouter>, <Routes>, <Route>.
- Link for navigation.
- useNavigate for programmatic routing.

Data Fetching

- Fetch data in useEffect.
- Track loading & error state.
- Cleanup with abort controller if needed.

Performance

- React.memo: prevent unnecessary renders.
- useCallback: memoize functions.
- useMemo: memoize values.

Styling

- Inline styles: style={{color:'red'}}.
- CSS Modules or Tailwind recommended.