

React.js Cheat Sheet - Test Prep

React.js Cheat Sheet - For 20-Minute Test

- Core Concepts
- Component: Reusable UI block (Function or Class)
- Props: Read-only inputs to components
- State: Local, mutable data within components
- Hooks (Function Components)
- `useState(initial)`: Add local state
`const [count, setCount] = useState(0);`
- `useEffect(() => {}, [deps])`: Side effects (fetching, timers)
- `useRef()`: DOM access / mutable value storage
- `useMemo(() => calc(), [deps])`: Cache expensive calculation
- `useCallback(() => fn, [deps])`: Cache function definition
- `useParams()`: Extract URL params
- JSX Basics
- Components return 1 parent element (`<>` or `<div>`)
- `{}` for JS expressions in JSX
- `className`, `htmlFor` instead of `class`, `for`
- React Router
- `<BrowserRouter>`, `<Routes>`, `<Route>`, `<Link>`
- `useNavigate()`: navigate programmatically
- `useParams()`: get dynamic segments from URL
- Forms
- `useState` for inputs
- Access: `event.target.value`
- Conditional Rendering
- `&&` operator or ternary
`{isLoggedIn && <LogoutButton />}`
`{isDark ? <DarkMode /> : <LightMode />}`
- React Performance
- `React.memo()`: skip re-renders
- `useMemo/useCallback`: cache
- `useTransition/useDeferredValue`: defer updates
- Lifecycle Methods (Class)
- `componentDidMount()`
- `componentDidUpdate()`
- `componentWillUnmount()`
- Redux (if tested)
- Central store with reducers, actions, dispatch
- `useSelector`, `useDispatch`
- Common JS Patterns
- Destructuring: `const { name } = props;`
- Spread: `<Comp {...props} />`
- Shorthand: `const car = { make, model };`

- Tips
- Hooks only in functional components
- Component names: Capitalized
- Props are immutable
- JSX must return a single root