# React Questions and Answers

**1. Which of the following is used to pass data to a React component from outside?**

A. setState

B. props

C. render

D. propTypes

b

**2. Which hook would be best for accessing query parameters from the URL in a React Router v6 app?**

A. useLocation

B. useParams

C. useQueryParams

D. useSearchParams

d

**3. Which of the following is NOT a valid hook?**

A. useState

B. useFetch

C. useEffect

D. useRef

b

**4. Why should we avoid using the index of an array as a key in lists?**

A. It causes an error in production

B. It slows down performance

C. It can cause rendering bugs if the list is reordered

D. It's deprecated in React 18

c

**5. In React Router, which component is used to wrap routes and enable routing?**

A. <Router>

B. <Route>

C. <Link>

D. <Switch>

a

**6. Which hook is best for tracking previous props or state?**

A. useEffect

B. useHistory

C. useMemo

D. useRef

d

**7. What is the virtual DOM in React?**

A. A browser API to manipulate the actual DOM

B. A real copy of the DOM maintained by React

C. A lightweight representation of the real DOM used for efficient updates

D. A debugging tool

c

**8. Which testing tool is most commonly used with React Testing Library?**

A. Cypress

B. Mocha

C. Jest

D. Jasmine

c

**9. What is the correct way to import useState from React?**

A. import useState from 'react';

B. const useState = require('react')

C. import { useState } from 'react';

D. import React.useState from 'react';

c

**10. What does the Context API help avoid?**

A. Writing JSX

B. Writing props manually in functional components

C. Prop drilling across multiple layers

D. Performance issues

C

**11. What is the difference between useRef and createRef?**

A. createRef is for functional components only

B. useRef creates a persistent reference across renders

C. createRef is used with hooks

D. They are identical

b

**12. Why is it important to use a key prop when rendering a list in React?**

A. For performance optimization and correct DOM diffing

B. To avoid runtime errors

C. To make the code more readable

D. To auto-style elements

a

**13. Which hook would you use to perform cleanup operations (e.g., clear timers)?**

A. useRef

B. useEffect with a cleanup function

C. useCallback

D. useCleanup

b

**14. In React Router, which hook is used to access the current route parameters?**

A. useRoute

B. useNavigate

C. useParams

D. useLocation

c

**15. Which of the following is true about controlled components in React?**

A. The DOM handles the form state

B. They store form data in component state

C. They use refs for managing input values

D. They cannot be used with hooks

B

**16. What does useMemo return?**

A. A callback function

B. A memoized value

C. A mutable reference

D. A component instance

b

**17. What is the purpose of React.Fragment?**

A. To split a component into multiple files

B. To group multiple elements without adding extra nodes to the DOM

C. To add animations

D. To handle error boundaries

b

**18. What is the best practice for fetching data in a React functional component?**

A. Inside useEffect

B. Directly in the component body

C. Inside useState

D. Inside the render return

a

**19. What is the correct syntax to create a functional component in React?**

A. function MyComponent() { return <div>Hello</div>; }

B. class MyComponent extends React.Component { render() { return <div>Hello</div>; }}

C. React.createComponent(MyComponent)

D. component MyComponent() { return <div>Hello</div>; }

a

**20. What does the useState hook return?**

A. A value

B. A value and a setter function

C. A reducer

D. Nothing

B

**21. Which method is used to render components in React?**

A. ReactDOM.start()

B. React.start()

C. ReactDOM.render()

D. render()

c

**22. What is JSX?**

A. A template engine

B. A file extension

C. A syntax extension for JavaScript

D. A JSON structure

c

**23. What is the purpose of useEffect in React?**

A. To style components

B. To handle side effects

C. To bind events

D. To manage state

b

**24. Which hook should you use to reference a DOM element?**

A. useState

B. useEffect

C. useRef

D. useContext

c

**25. What does React.memo() do?**

A. Converts components to functional ones

B. Prevents re-rendering unless props change

C. Adds memory cache to components

D. Logs component updat

B

**26. What is the default behavior of React when the state of a component changes?**

A. It does nothing

B. It replaces the DOM manually

C. It re-renders the entire DOM

D. It re-renders only the affected components

d

**27.  How can you prevent a component from re-rendering?**

A. Using shouldComponentUpdate in class components

B. Using React.memo in functional components

C. Both A and B

D. You cannot prevent re-rendering

c

**28. What is the main purpose of keys in React lists?**

A. To style the list

B. To keep track of element identity between renders

C. To increase performance

D. To uniquely identify class components

b

**29. When using useEffect, how do you specify that it should only run once (on mount)?**

A. Pass an empty object as dependency

B. Don't pass any dependency

C. Pass an empty array as dependency

D. Use useMemo instead

c

**30. Which statement about useContext is true?**

A. It can only be used in class components

B. It is used to share state across deeply nested components

C. It replaces Redux completely

D. It creates a new Context object

B

**31. What is the correct way to lift state up in React?**

A. Pass the state directly to child components

B. Move the state to the common ancestor component

C. Use useReducer

D. Use Redux

b

**32. How do you handle 404 (Not Found) routes in React Router v6+?**

Use the catch-all route path="*":

<Route path="*" element={<NotFound />} />

**33. What is React Router and why is it used?**

React Router is a client-side routing library for React apps.

It allows navigation between pages without refreshing the page (SPA behavior).

<Route path="/about" element={<About />} />

Supports nested routes, dynamic routes, and navigation guards.

**34. How does conditional rendering work with short-circuit evaluation?**

Using && or || for inline conditions:

{isLoggedIn && <LogoutButton />}

{errors.length > 0 || <p>No errors</p>}

**35. What is "lifting state up" in React?**

When two or more components need shared state, the state is moved to their closest common ancestor, and passed down via props.

// Move state to parent and pass to children

**36. What are controlled and uncontrolled components?**

Controlled: Form input is controlled via React state

<input value={name} onChange={e => setName(e.target.value)} />

Uncontrolled: Uses ref to access input directly

const inputRef = useRef(); <input ref={inputRef} />

**37. What is the purpose of React.memo()?**

React.memo() is a higher-order component that prevents re-rendering of a functional component if its props haven't changed.

const MemoizedComponent = React.memo(MyComponent);

**38. What is the Context API? How is it different from prop drilling?**

The Context API is a way to pass data through the component tree without passing props manually at every level.

Prop drilling = passing props down through many layers

Context = global-like state for components that need access

const ThemeContext = React.createContext();

<ThemeContext.Provider value="dark"><Child /></ThemeContext.Provider>

**39. Explain the use of useState, useEffect, and useRef.**

useState: Adds state to functional components

const [count, setCount] = useState(0);

useEffect: Runs side effects (e.g., fetching data)

useEffect(() => { fetchData(); }, []);

useRef: Holds a mutable value that doesn't trigger re-renders

const inputRef = useRef();

**40. What are hooks in React? Why were they introduced?**

Hooks are functions that let you use state and other React features in functional components.

They were introduced in React 16.8 to avoid writing class components for features like state or lifecycle.

Common hooks: useState, useEffect, useRef, useContext, useMemo, useReducer

**41. What is conditional rendering in React? Give an example.**

Conditional rendering lets you render components based on conditions.

{isLoggedIn ? <Dashboard /> : <Login />}

ou can also use &&, if-else, or ternary operators.

**42. What is the role of key in lists in React?**

Keys help React identify which items in a list have changed, added, or removed.

{items.map(item => <li key={item.id}>{item.name}</li>)}

Use unique, stable values like IDs (not array indices).

**43. How does React handle events?**

React handles events using camelCase syntax and passes functions as handlers.

<button onClick={handleClick}>Click Me</button>

Unlike HTML, you cannot return false to prevent default; use event.preventDefault().

**44. What is state in React and how do you update it?**

State is an object that stores data that affects a component's rendering.

In functional components, use useState().

const [count, setCount] = useState(0);

setCount(count + 1);

**45. What are props in React? How are they different from state?**

Props (short for properties) are read-only values passed from parent to child components.

State is a mutable local data storage managed within a component.

<MyComponent title="Hello" /> // "title" is a prop

**46. How does the virtual DOM work in React?**

The Virtual DOM is a lightweight, in-memory representation of the real DOM.

When state changes, React creates a new virtual DOM, compares it to the previous one (diffing), and updates only the parts that changed in the actual DOM.

Result: Better performance.

**47. What is JSX?**

JSX (JavaScript XML) is a syntax extension for JavaScript that looks similar to HTML.

It allows you to write UI code that is more readable and easier to manage.

const element = <h1>Hello, JSX!</h1>;

**48. What are components in React?**

Components are the building blocks of a React application. They are reusable, independent pieces of code that return JSX (UI).

Types:

Functional Components

Class Components