Top 30 React JS Interview Questions & Answers

PDF Guide

Top 30 React JS Interview Questions & Answers PDF Guide (2025 Edition)

Top 30 Most-Asked React JS Interview Questions — concise answers for fast interview prep

1) What is React?

React is a JavaScript library for building user interfaces using a component-based architecture and state-driven rendering patterns.

2) What are the main features of React?

Declarative UI, component-based structure, virtual DOM diffing for efficient updates, and one-way (unidirectional) data flow.

3) What is JSX?

JSX is a syntax extension that lets developers write HTML-like markup inside JavaScript; it compiles to `React.createElement` calls.

4) What is the Virtual DOM?

A lightweight in-memory representation of the DOM that React diffs to compute minimal real DOM updates for better performance.

5) What are components in React?

Reusable building blocks that encapsulate UI structure, behavior, and styling; applications are composed by nesting components.

6) Difference between functional and class components?

Functional components are functions using hooks for state and side effects; class components use lifecycle methods and `this`, and are less common in new code.

7) What are props?

Props are read-only inputs passed from parent to child components to configure rendering and behavior.

8) What is state?

State is component-owned mutable data that determines rendering; updating state schedules a re-render.

9) What is the purpose of `useState`?

`useState` creates state variables in functional components and returns a setter to update them and trigger re-renders.

10) What is the `useEffect` hook used for?

`useEffect` runs side effects (data fetches, subscriptions, DOM updates); the dependency array controls when it executes.

11) What is the Context API?

Context allows sharing values (theme, auth, locale) across the component tree without prop drilling, via Provider and `useContext`.

12) What are keys in React lists?

Keys are stable identifiers that help React track list items between renders to minimize DOM changes; avoid using array indices when order can change.

13) What is prop drilling?

Prop drilling is passing props through many component layers; Context or state libraries often replace it for deep trees.

14) What are controlled components?

Form inputs whose values are driven by React state; components update state on change and read the value from state.

15) What are uncontrolled components?

Form inputs that manage their own internal DOM state; refs are used to read values when needed.

16) What is the `useRef` hook?

`useRef` returns a mutable object with a `.current` property; use it for DOM nodes or persisted mutable values across renders.

17) What are higher-order components (HOCs)?

HOCs are functions that take a component and return a new component with added behavior or props; hooks often replace HOCs in modern patterns.

18) What is the purpose of `React.memo` ?

`React.memo` memoizes a functional component to skip re-renders when props are shallowly equal; useful for performance when used correctly.

19) What is the `useCallback` hook?

`useCallback` returns a stable function reference that only changes when dependencies change, helping avoid unnecessary child re-renders.

20) What is the `useMemo` hook?

`useMemo` memoizes an expensive computed value and recomputes it only when dependencies change.

21) What is the purpose of `React.lazy` ?

`React.lazy` enables dynamic import of components for code-splitting; use with `Suspense` to show fallbacks while loading.

22) What are React Fragments?

Fragments let components return multiple children without adding extra DOM nodes using `<></>` or `<React.Fragment>` .

23) What is `useReducer` ?

`useReducer` manages complex state logic with a reducer function and is preferable when next state depends on previous state or multiple sub-values exist.

24) Difference: `componentDidMount` vs `componentDidUpdate` ?

`componentDidMount` runs once after initial render; `componentDidUpdate` runs after updates. In functional components, `useEffect` replicates both with dependency control.

25) What is the purpose of `shouldComponentUpdate` ?

`shouldComponentUpdate` lets class components skip renders by returning false; `PureComponent` and `React.memo` perform shallow comparisons automatically.

26) What are error boundaries?

Error boundaries are components that catch render-time errors in their child tree to log errors and render a fallback UI; implemented via lifecycle methods in class components.

27) What is the `forwardRef` function?

`forwardRef` passes a ref from a parent through a component to a child DOM node, enabling parent-level control of child DOM elements.

28) What is the purpose of `React.StrictMode` ?

`React.StrictMode` is a development-only helper that highlights unsafe lifecycles, legacy APIs, and other side effects to encourage best practices.

29) Difference between CSR and SSR?

Client-side rendering (CSR) renders in the browser after JS loads; server-side rendering (SSR) renders HTML on the server for faster first paint and better SEO for public pages.

30) What are React Portals?

Portals render children into a DOM node outside the parent hierarchy and are ideal for modals, tooltips, and overlays to avoid z-index and stacking issues.

Bonus: Interview-ready add-ons

- Performance: memoization, avoid inline objects, code-split with React.lazy/Suspense.
- Patterns: lift state, build custom hooks, use Context judiciously.
- Practice: create a small feature using useReducer + Context and profile with React Profiler.

© Teki Solves — Not for redistribution.
Test these questions: **tekitestbot**