

1. What is React and why is it used?

React is an open-source JavaScript library developed by Facebook for building fast, interactive, and scalable user interfaces. It is mainly used for building single-page applications where data changes frequently without reloading the page. React follows a component-based architecture, meaning the UI is broken into small, reusable pieces called components. This makes applications easier to maintain, debug, and scale. React uses a Virtual DOM to optimize rendering performance by updating only the parts of the UI that change, instead of reloading the entire page.

2. What is JSX in React?

JSX stands for JavaScript XML. It is a syntax extension that allows developers to write HTML-like code directly inside JavaScript. JSX makes the structure of the UI easier to understand and visualize. Although it looks like HTML, JSX is compiled by Babel into `React.createElement` calls. JSX supports JavaScript expressions, conditional rendering, and dynamic content, making it a powerful and flexible tool.

3. What are components in React?

Components are the core building blocks of React applications. A component represents a reusable piece of the user interface with its own logic and appearance. React components can be functional or class-based. Functional components are preferred today because they are simpler and support hooks. Components help divide complex UIs into smaller, manageable, and reusable units.

4. What is the Virtual DOM?

The Virtual DOM is an in-memory representation of the real DOM. When state or props change, React creates a new Virtual DOM tree and compares it with the previous one using a process called diffing. Only the differences are updated in the real DOM. This makes React applications faster and more efficient compared to direct DOM manipulation.

5. What is state in React?

State is an object that stores data that can change over time in a component. It determines how the component behaves and what it renders. When state changes, React automatically re-renders the component. State is local to the component and should never be modified directly. Instead, state update functions should be used.

6. What are props in React?

Props are short for properties and are used to pass data from parent components to child components. Props are read-only, meaning child components cannot modify them. Props help make components reusable and dynamic. They are a key part of React's unidirectional data flow.

7. Difference between state and props

State is mutable and managed within the component, while props are immutable and passed from parent to child. State is used for data that changes over time, whereas props are used to configure or initialize components.

8. What are React Hooks?

Hooks are functions introduced in React 16.8 that allow functional components to use state, lifecycle methods, and other React features. Hooks eliminate the need for class components and make code more readable and reusable. Common hooks include useState, useEffect, useContext, and useRef.

9. Explain useState Hook

useState is a hook that allows you to add state to functional components. It returns an array containing the current state value and a function to update it. When the updater function is called, React re-renders the component with the new state.

10. Explain useEffect Hook

useEffect is used to perform side effects such as data fetching, subscriptions, and DOM updates. It runs after the component renders. The dependency array controls when the effect runs, helping optimize performance and avoid unnecessary executions.