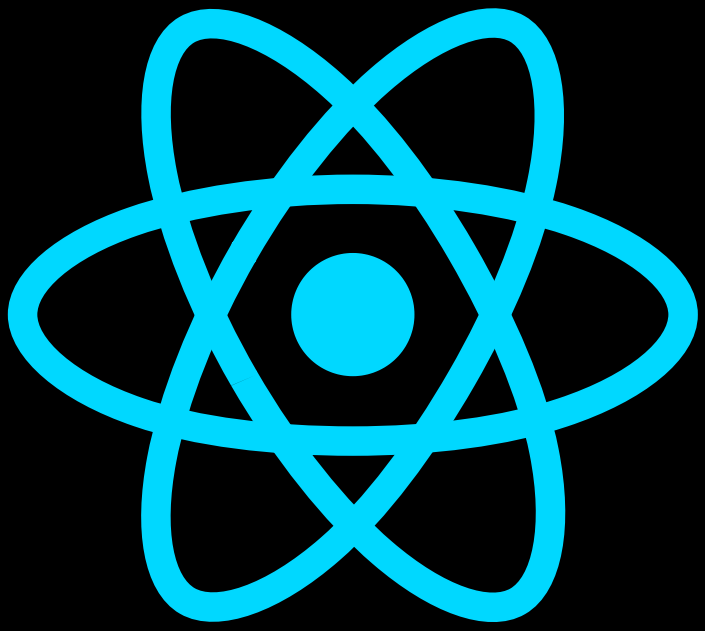
REACT

INTERVIEW GUIDE



“Most Asked Interview Questions”

www.codehype.in

1. **What is React, and how does it differ from other JavaScript frameworks?**

   React is an open-source JavaScript library developed by Facebook for building user interfaces, specifically single-page applications. It emphasizes component-based architecture, allowing developers to build encapsulated components that manage their state. Unlike traditional MVC frameworks like Angular, React focuses solely on the "View" aspect of MVC, making it lightweight and faster. React also introduces the concept of a virtual DOM, which optimizes the performance of UI rendering by reducing the number of direct manipulations to the actual DOM.

2. **What are components in React, and why are they important?**

   Components are the fundamental building blocks of a React application. They are reusable, self-contained pieces of UI that can manage their state and interact with other components via props. Components can be either class-based or functional. They allow for better organization, maintainability, and scalability of large applications by breaking down the UI into smaller, manageable parts. Each component can be independently developed, tested, and reused across different parts of an application or even across different projects.

3. **What is JSX, and why is it used in React?**

   JSX (JavaScript XML) is a syntax extension for JavaScript that resembles HTML. It is used in React to describe what the UI should look like. JSX makes the code more readable and easier to write by allowing developers to use HTML-like

CodeHype

syntax directly within JavaScript. Under the hood, JSX is transformed into regular JavaScript function calls (e.g., React.createElement) that construct React elements. This enables the declarative style of programming in React, where the UI is described as a function of the application's state.

4. **What are the features of React?**

   React is a popular JavaScript library for building user interfaces, and it offers a variety of features that make it a powerful tool for web development. Here are some of the key features of React:

- **Component-Based:** React is built around the concept of reusable UI components, making it easy to manage and maintain code.
- **Virtual DOM:** React's virtual DOM efficiently updates the actual DOM, improving rendering performance.
- JSX: JSX allows developers to write HTML-like code within JavaScript for defining component structures.
- **Unidirectional Data Flow:** Data flows in one direction, simplifying data management and updates.
- **Reusability:** React components can be reused across the application, promoting code reusability.
- **Declarative:** React allows you to describe what your UI should look like, and it takes care of updating the DOM accordingly.
- **Community and Ecosystem:** React has a large and active developer community and a rich ecosystem of libraries and tools.
- **Performance Optimization:** React provides tools for optimizing performance, such as code splitting.

CodeHype

- **Support for Server-Side Rendering (SSR):** React can render on the server side, improving SEO and initial load times.
- **Developer Tools:** React offers browser extensions and tools for debugging and inspecting components.

5. **What is the difference between React and other JavaScript frameworks?**

React is a JavaScript library for building user interfaces. It is declarative, efficient, and flexible. React is used by companies like Facebook, Instagram, and Netflix to build complex web applications.
Other popular JavaScript frameworks include Angular and Vue.js. These frameworks also have their own strengths and weaknesses, but here are some of the key differences between React and other JavaScript frameworks:

CodeHype

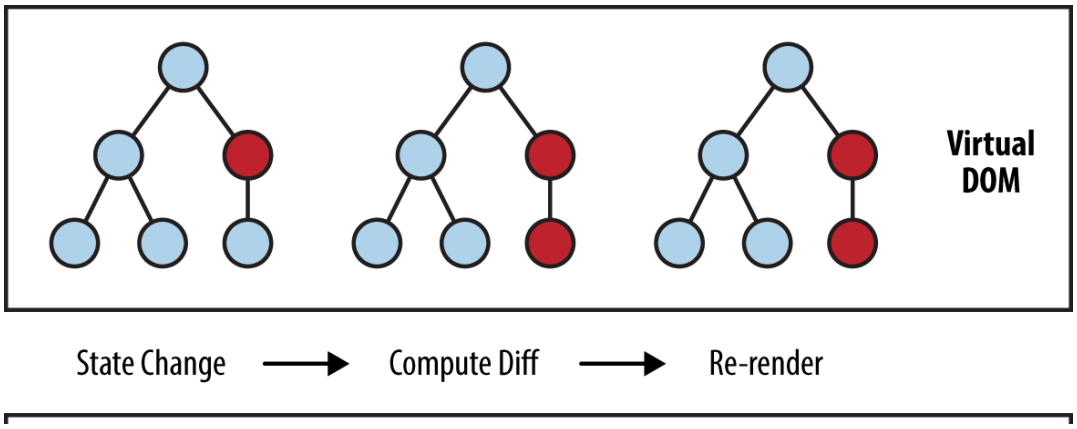
Feature	React	Angular	Vue.js
Type	Library	Framework	Framework
Component-based architecture	Yes	Yes	Yes
Virtual DOM	Yes	No	Yes
JSX	Optional	Required	Optional
Performance	Faster	Slower	Faster
Best Suited for	SPAs, mobile apps,websites	Complex enterprise applications	Small to medium-sized applications

6. What is the Virtual DOM?

Virtual DOM is a lightweight representation of the real DOM. It is a JavaScript object that describes the current state of the UI.
When the state of the UI changes, React updates the Virtual DOM. Then, React compares the Virtual DOM to the real DOM and efficiently updates the real DOM only with the changes that are needed.

7. How does the Virtual DOM work?

Virtual DOM is a JavaScript object that describes the current state of the UI. It is a tree structure, with each node in the tree representing an element in the UI. The Virtual DOM also includes information about the attributes and children of each element.
To update the real DOM, React uses a process called reconciliation. Reconciliation compares the Virtual DOM to the real DOM and determines the minimal set of changes needed to bring them into alignment. React then applies these changes to the real DOM.
Reconciliation is a very efficient process. React uses a diffing algorithm to compare the Virtual DOM to the real DOM. This algorithm is able to identify the smallest possible set of changes needed to updatethe real DOM.



CodeHype

8. What are the different types of components in React?

There are two main types of components in React:
Functional components: Functional components are pure functions that take props as input and return an element as output. They are the simplest and most common type of component in React.
Class components: Class components are more complex than functional components, but they offer more features, such as state management and lifecycle hooks.

9. What is the difference between state and props?

State is the data that is local to a React component and can be changed by the component itself.
Props are data that are passed to a React component from its parent component and cannot be changed by the child component.

CodeHype

Feature	State	Props
Definition	Data that is local to a React component and can be changed by the component itself	Data that are passed to a React component from its parent component and cannot be changed by the child component
Mutable	Yes	No
Owned by	Component	Parent component
Updated by	Component	Parent component

10. **How do you pass props from one component to another?**

To pass props from one component to another in React, we need to use the props attribute.
The props attribute is an object that contains the data that you want to pass to the child component.

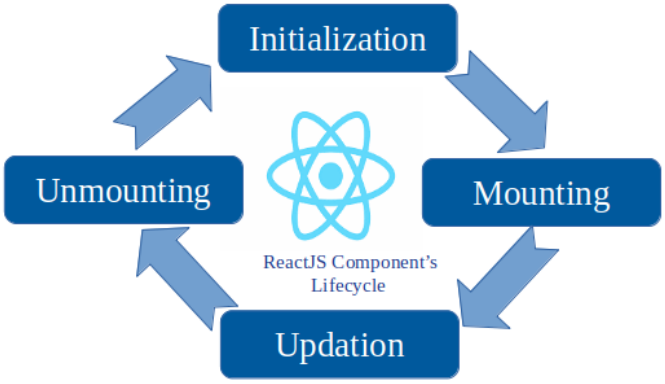
11. **What is the one-way data flow in React?**

One-way data flow in React is a design pattern that ensures that data flows in one direction, from parent to child components.
This helps to make React applications more predictable and easier to debug.

12 **What are the different lifecycle methods in React?**

.

There are three main phases in the React component lifecycle:
Mounting: This is the phase when a component is first created and inserted into the DOM.
Updating: This is the phase when a component's state or props change.
Unmounting: This is the phase when a component is removed from the DOM.



CodeHype

13. What are the benefits of using React hooks?

React **Hooks**, introduced in React 16.8, allow you to use state and other React features within functional components. They provide several advantages over the older class components and are now the preferred method for managing state and side effects in React applications.

1. **Simplified Code:** Hooks enable you to write more concise and readable code by eliminating the need for class components.

2. **Enhanced Reusability:** Hooks can be easily reused across different components, making your code more modular and maintainable.

CodeHype

3. **Easier Testing and Debugging:** Testing and debugging are simpler with Hooks since they are just functions that can be called in isolation.

4. **Reduced Bundle Size:** Hooks help reduce the bundle size of your application, as they are smaller and more efficient than class components.

14. What is the difference between a controlled and uncontrolled component?

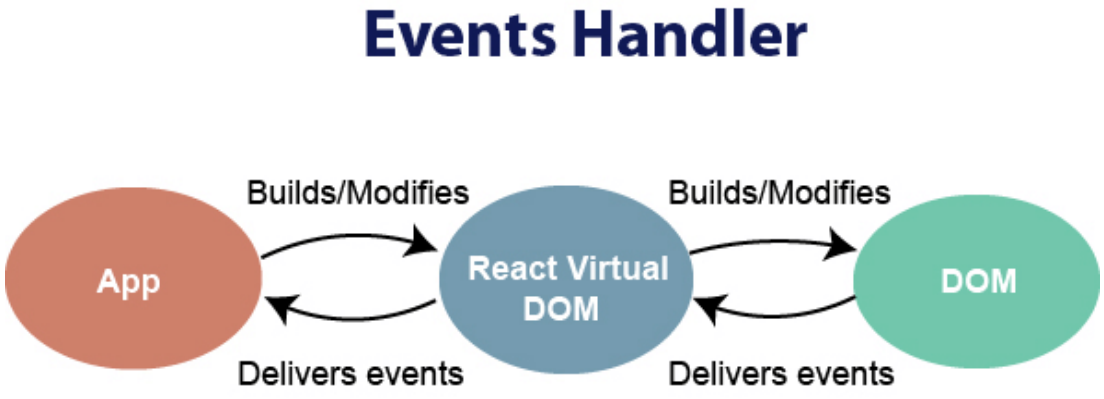
The main difference between a controlled and uncontrolled component is how the component handles its state.

A **controlled component** in React is one where the form data is handled by the component's state. The component fully controls the input value via props, and changes to the input are managed through event handlers, allowing for more precise control and validation.

An **uncontrolled component** stores form data internally within the DOM, not in the component's state. The input value is accessed using refs, making the component simpler but providing less control over the form's behavior.

15. What is the best way to handle events in React?

The best way to handle events in React is by using arrow functions or binding event handlers in the constructor, or by using class properties (property initializer syntax) for class components.
For functional components, you can use the useState and useCallback hooks to handle events efficiently and prevent unnecessary re-renders.



Codehype

16. What is the best way to manage state in React?

The most effective way to manage state in React is by leveraging React Hooks, such as useState and useReducer, within functional components. For class components, state can be managed using this.state and this.setState.

For global state management, consider using the Context API or third-party libraries like Redux when the application's complexity requires it.

17. What are some common performance optimization techniques in React?

CodeHype

Here are some common performance optimization techniques in React:

1. **Memoization with React.memo:** Use React.memo to prevent unnecessary re-renders of functional components by memoizing their output. This is particularly useful for components that rely on the same props and do not need to update unless those props change.

2. **Using useMemo and useCallback:** useMemo memoizes expensive calculations, ensuring they are only re-computed when dependencies change. useCallback memoizes functions, preventing them from being recreated on every render, which is beneficial when passing functions as props to memoized components.

3. **Code Splitting:** Implement code splitting using dynamic import() or tools like React's React.lazy and Suspense to load components only when needed, reducing the initial load time of the application.

4. **Lazy Loading:** Defer loading of off-screen components or resources until they are needed using techniques like React.lazy for components and the `loading` attribute for images.

5. **Avoiding Reconciliation:** Minimize the number of changes in the virtual DOM by ensuring that keys are stable and unique when rendering lists. Avoid unnecessary re-renders by using shouldComponentUpdate, PureComponent, or React.memo.

CodeHype

6. **Optimizing Rendering with shouldComponentUpdate and PureComponent:** In class components, override shouldComponentUpdate to control when a component should re-render. PureComponent provides a shallow comparison of props and state to optimize rendering.

7. **Debouncing and Throttling:** For performance-heavy operations like resizing, scrolling, or handling user input, use debouncing or throttling to limit the frequency of function execution, reducing the workload on the browser.

8. **Virtualization:** Use libraries like react-window or react-virtualized to render only the visible portion of large lists or tables, improving performance by avoiding unnecessary DOM updates.

9. **Efficient State Management:** Keep state as localized as possible. Avoid lifting state unnecessarily and use the Context API or external state management libraries only when global state is truly needed.

10. **Optimizing Images and Assets:** Compress images and use modern formats like WebP. Leverage responsive images and lazy loading to reduce the impact on performance.

18. What are some of the most popular React libraries and tools?

Some of the most popular React libraries and tools include:

1. **Redux:** For managing global state in complex applications.
2. **React Router:** For handling routing and navigation within single-page applications.
3. **Material-UI:** For implementing Material Design with customizable UI components.
4. **Styled Components:** For writing component-level styles using CSS-in-JS.
5. **React Query:** For managing server-state, data fetching, and caching.

CodeHype

19. What is a higher-order component (HOC) in React?

A higher-order component (HOC) is an advanced pattern in React used to enhance or modify the behavior of a component. An HOC is a function that takes a component as an argument and returns a new component with additional props, logic, or functionality.

The primary purpose of HOCs is to enable code reuse and component composition. They allow you to inject common functionality or state into multiple components without repeating code. HOCs are useful for tasks like:

1. **Code Reuse:** Sharing common logic or behavior across multiple components.
2. **Conditional Rendering:** Adding conditional rendering logic based on props or state.
3. **State Management:** Injecting state or context into components.
4. **Enhancements:** Adding features such as logging, error handling, or data fetching.

20. What is internationalization (i18n) in React?

Internationalization (i18n) in React refers to the process of designing and developing applications to support multiple languages and regional differences. It involves preparing your React application to handle various languages, formats, and locales, allowing users to interact with the application in their preferred language. This is typically achieved using libraries like react-intl or i18next, which provide tools for translating text, formatting dates and numbers, and managing locale-specific content.

21. What is the significance of the key prop in React?

The key prop helps React identify which items in a list have changed, been added, or removed. By providing a unique key for each element, React can efficiently update and re-render only the necessary items, improving performance and ensuring that elements are correctly managed during updates.

Download FREE Ebooks from www.codehype.in

22. **What is the difference between React and React Native ?**

React and React Native are closely related but serve different purposes:

React:

- **Purpose:** React is a JavaScript library used for building user interfaces, primarily for web applications.
- **Rendering:** It renders components to HTML in the browser, allowing for the creation of interactive and dynamic web pages.
- **Components:** React components are typically written using HTML, CSS, and JavaScript, which are then rendered as HTML elements in the DOM.

React Native:

CodeHype

- **Purpose:** React Native is a framework for building mobile applications using React principles.
- **Rendering:** It does not render HTML. Instead, it uses native components, which means it translates React components into native UI elements for iOS and Android platforms.
- **Components:** React Native components are built using JavaScript and JSX but render as native components like View, Text, and Button on mobile devices, rather than HTML elements.

Download 100+ FREE Coding Notes, Ebooks, Cheatsheets and more 👇

www.codehype.in

Join our Telegram Channel for FREE Coding Resources, Ebooks, Notes, FREE Courses Internships and more...

@codehype