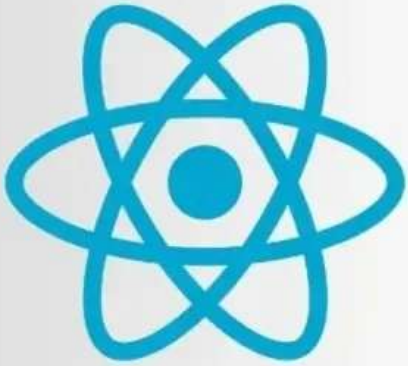
React

ADVANCED CONCEPTS

1. Higher Order Components
2. Render Props
3. Error Boundary
4. React.lazy and Suspense
5. Virtual DOM and Reconciliation
6. React Portals
7. Memoization
8. SSR & SSG
9. Immutable Data Structures
10. Higher Order Reducers
11. Web Workers
12. Custom Renderers
13. Server Side Events
14. Optimization and Performance Tuning



@frontend_in_depth

Higher-Order Components (HOc)

HOC is a design pattern in React that allows you to wrap a component with a function to enhance its behavior or add additional props. It's a way of reusing component logic and sharing code between components. HOCs are widely used in libraries like Redux's connect and React Router's with Router.

Render Props

Render props is another pattern for code reuse in React. It involves passinga function as a prop to a component, allowing the component to render the result of that function. This pattern is useful for sharing logic and state between components in a flexible manner.

Error Boundaries

Error boundaries are React components that catch JavaScript errors during rendering, in lifecycle methods, and during constructor execution. By using error boundaries, you can gracefully handle errors and prevent the entire application from crashing.

React.lazy and Suspense

These features enable code-splitting and lazy loading of Components, making your application more efficient by loading only the necessary parts of the code when needed.

Virtual DOM and Reconciliation

Understanding how React's Virtual DOM WOrks and the process of reconciliation, where React efficiently updates the actual DOM based on the changes in the Virtual DOM, is crucial for optimizing performance.

Portals

Portals allow you to render a component's content in a different part of the DOM tree, outside the parent component's hierarchy. This is useful for scenarios like modals, tooltips, and popovers, where the Conternt should be positioned outside the main application structure.

Memoization

Memoization is a technique used to optimize expensive function computations by caching the results of function calls. In React, you can use React.memo to memoize the rendering of a component and prevent unnecessary re-renders.

Server-Side Rendering (sSR) and Static Site

Generation (SsG)
SSR and ssG are techniques for rendering React components on the server-side, providing better SEO, faster initial page load times, and improved performance. Libraries like Next.js and Gatsby.js facilitate these approaches.

Immutable Data Structures

Immutability is an esserntial concept in React and Javascript. Using immutable data structures, such as those provided by libraries like Immutable.js or Immer, can lead to better performance, simpler state management, and fewer bugs.

Higher-Order Reducers

Similar to Higher-Order Components, higher-order reducers are functions that enhance the behavior of reducers in Redux or other state management libraries. They enable code reuse and encapsulation of reducer logic.

Web Workers

Web Workers allow you to run JavaScript code in the background thread, separate from the main UI thread. This Can help improve the performance and responsiveness of your application by offloading complex or time-consuming tasks.

Custom Renderers

React's architecture allows you to create custom renderers, enabling you to render React components to alternative targets like VR, Canvas, or even custom platforms. You have only used react-dom until now, but react-native, react-canvas, react-native-skia, react-tv, react-pdf, react-figma, etc. are popular examples of React's custom renderer.

Server-Side Events

(SSE) and WebSockets

SSE and Websockets enable real-time communication between the server and the client, allowing you to push data from the server to the client without the need for the client to constantly poll for updates.

optimizations and Performance Tuning

Advanced optimization techniques, such as code splitting, lazy loading, tree shaking, and performance profiling, can significantly improve the speed and efficiency of your React application.