SARVESH

Interview Questions (FRESHERS)

HTML Interview Questions

1. What is HTML and what is its role in web development?
2. Explain the purpose of the `<!DOCTYPE html>` declaration in an HTML document.
3. Describe the difference between HTML and XHTML.
4. What are the new features introduced in HTML5?
5. How do you create a hyperlink in HTML?
6. What are semantic elements in HTML? Provide examples.
7. How do you embed audio and video in an HTML document?
8. Explain the difference between `<div>` and `<span>` in HTML.
9. How do you create a form in HTML? Explain the basic structure.
10. What is the purpose of the `alt` attribute in an image tag?
11. How do you create a table in HTML?
12. What is the purpose of the `<head>` section in an HTML document?
13. How do you include comments in HTML?
14. Explain the difference between the `id` and `class` attributes in HTML.
15. How do you create a numbered list in HTML?
16. Describe the difference between the `<strong>` and `<em>` tags in HTML.
17. How do you create a responsive image in HTML?
18. What is the purpose of the `<meta>` tag in HTML?
19. How do you create a dropdown list in HTML?
20. Explain the difference between HTML elements and HTML attributes.

CSS Interview Questions

1. What is CSS and what is its role in web development?
2. How do you include CSS in an HTML document?
3. Explain the concept of the box model in CSS.
4. Describe the difference between padding and margin in CSS.
5. How do you center an element horizontally and vertically in CSS?
6. What is the purpose of the `z-index` property in CSS?
7. How do you create a CSS animation?
8. Explain the difference between `display: block`, `display: inline`, and `display: inline-block`.
9. What are pseudo-classes in CSS? Provide examples.
10. How do you create a responsive design using CSS?
11. Describe the difference between absolute and relative positioning in CSS.

12. What is the purpose of the `float` property in CSS?
13. How to create variables in CSS?
14. Explain the difference between `font-weight: normal`, `font-weight: bold`, and `font-weight: bolder` in CSS.
15. How do you create a fixed navigation bar in CSS?
16. What is the purpose of the `clear` property in CSS?
17. How do you create a CSS grid layout?
18. Explain the difference between `visibility: hidden;` and `display: none;` in CSS.
19. How do you override CSS styles?
20. Describe the difference between `inline`, `block`, and `inline-block` elements in CSS.

JavaScript Interview Questions

1. What is JavaScript and what is its role in web development?
2. How do you declare a variable in JavaScript?
3. Explain the difference between `==` and `===` in JavaScript.
4. How do you create an array in JavaScript?
5. What are the different data types in JavaScript? Provide examples.
6. How do you create a function in JavaScript?
7. What is the purpose of the `this` keyword in JavaScript?
8. Explain the concept of hoisting in JavaScript.
9. What are closures in JavaScript?
10. How do you handle errors in JavaScript?
11. Explain the concept of event bubbling in JavaScript.
12. How do you manipulate the DOM using JavaScript?
13. What is the difference between `null` and `undefined` in JavaScript?
14. How do you work with JSON data in JavaScript?
15. What is the difference between `let`, `const`, and `var` in JavaScript?
16. How do you create and use objects in JavaScript?
17. What is call(), apply() and bind() in JavaScript?
18. How do you handle asynchronous operations in JavaScript?
19. What are promises in JavaScript?
20. How do you use the `map` method in JavaScript?

REACT JS INTERVIEW QUESTIONS AND ANSWERS

Freshers

1. **What is React?**
   - **Answer:** React is a JavaScript library for building user interfaces, particularly single-page applications, using reusable components.

2. **What are components in React?**
   - **Answer:** Components are the building blocks of a React application. They can be class-based or functional and encapsulate both logic and UI.
3. **What is JSX?**
   - **Answer:** JSX stands for JavaScript XML, which allows you to write HTML elements in JavaScript and place them in the DOM without using `createElement()` or `appendChild()`.
4. **What is the difference between a class component and a functional component?**
   - **Answer:** Class components can hold and manage state and lifecycle methods, while functional components are stateless and are mainly used for rendering UI. However, with the introduction of hooks, functional components can now also manage state and lifecycle methods.
5. **What is state in React?**
   - **Answer:** State is an object that represents the parts of the app that can change. Each component can maintain its own state, and when the state changes, the component re-renders.
6. **What are props in React?**
   - **Answer:** Props (short for properties) are read-only attributes used to pass data from a parent component to a child component.
7. **What is the Virtual DOM?**
   - **Answer:** The Virtual DOM is a lightweight copy of the actual DOM. React uses the Virtual DOM to optimize updates to the actual DOM by only re-rendering elements that have changed.
8. **What is a React fragment?**
   - **Answer:** A React fragment allows you to group multiple elements without adding extra nodes to the DOM. It can be used with the shorthand syntax `<></>`.
9. **What are lifecycle methods in React?**
   - **Answer:** Lifecycle methods are special methods in class components that allow you to run code at specific times during a component's lifecycle, such as `componentDidMount`, `componentDidUpdate`, and `componentWillUnmount`.
10. **How do you handle events in React?**
    - **Answer:** Events in React are handled using camelCase syntax. Event handlers are passed as functions to elements, for example, `onClick={handleClick}`.
11. **What is conditional rendering in React?**
    - **Answer:** Conditional rendering in React allows you to render different components or elements based on a condition, typically using JavaScript conditional statements like `if` or ternary operators.
12. **What is the purpose of the `key` prop in React?**
    - **Answer:** The `key` prop is used to uniquely identify elements in an array to help React optimize re-rendering by keeping track of which items have changed, been added, or removed.
13. **What is a controlled component in React?**
    - **Answer:** A controlled component is a form element that is controlled by React state. The value of the input element is defined by the state, and any changes to the input update the state.
14. **What is an uncontrolled component in React?**

- **Answer:** An uncontrolled component is a form element that maintains its own internal state. The value of the input is managed by the DOM rather than the React state.

15. **What is `create-react-app`?**
    - **Answer:** `create-react-app` is a command-line tool that sets up a new React project with a pre-configured development environment, including a build process, hot reloading, and more.
16. **What are hooks in React?**
    - **Answer:** Hooks are functions that allow you to use state and other React features in functional components. Examples include `useState`, `useEffect`, and `useContext`.
17. **What is the `useState` hook?**
    - **Answer:** The `useState` hook is used to add state to a functional component. It returns an array with the current state value and a function to update it.
18. **What is the `useEffect` hook?**
    - **Answer:** The `useEffect` hook allows you to perform side effects in functional components, such as data fetching, subscriptions, or manually changing the DOM.
19. **What is the `useContext` hook?**
    - **Answer:** The `useContext` hook allows you to access the value of a context directly in a functional component without having to use a context consumer.
20. **What is the `useReducer` hook?**
    - **Answer:** The `useReducer` hook is an alternative to `useState` for managing more complex state logic. It takes a reducer function and an initial state and returns the current state and a dispatch function.
21. **How do you pass data between components in React?**
    - **Answer:** Data can be passed between components in React using props for parent-to-child communication and callbacks for child-to-parent communication.
22. **What is React Router?**
    - **Answer:** React Router is a library for routing in React applications, allowing you to navigate between different views or components while maintaining the UI state.
23. **What is `ReactDOM.render()`?**
    - **Answer:** `ReactDOM.render()` is a method that renders a React element into the DOM. It takes a React element and a DOM container as arguments.
24. **What is a Higher Order Component (HOC)?**
    - **Answer:** A Higher Order Component is a function that takes a component and returns a new component with additional props or behaviors. HOCs are used for code reuse, logic, and bootstrap abstraction.
25. **What is the Context API in React?**
    - **Answer:** The Context API allows you to pass data through the component tree without having to pass props down manually at every level. It is used to share data that can be considered global for a tree of React components.
26. **What is a ref in React?**
    - **Answer:** A ref is a way to access the DOM directly within React. It is used to get a reference to a DOM element or a class component instance.
27. **How do you handle form submissions in React?**

- **Answer:** Form submissions in React are handled by attaching an `onSubmit` event handler to the form element, which processes the form data and prevents the default form submission behavior.

28. **What is the difference between `componentDidMount` and `componentWillUnmount`?**
    - **Answer:** `componentDidMount` is called after the component is rendered to the DOM and is used for initialization, such as data fetching. `componentWillUnmount` is called before the component is removed from the DOM and is used for cleanup, such as cancelling network requests or removing event listeners.
29. **What is prop drilling in React?**
    - **Answer:** Prop drilling refers to the process of passing data from a parent component to a deeply nested child component through multiple levels of intermediate components.
30. **How do you optimize a React application?**
    - **Answer:** Optimization techniques include:
        - Using PureComponent or React.memo to prevent unnecessary re-renders
        - Code splitting with React.lazy and Suspense
        - Memoizing functions with useCallback
        - Using React's Profiler to identify performance bottlenecks
        - Virtualizing lists with react-virtualized or react-window

Experienced

1. **How does React handle events differently than plain JavaScript?**
    - **Answer:** In React, events are handled using synthetic events, which are wrappers around the native events and work identically across all browsers. This ensures cross-browser compatibility.
2. **What are hooks in React?**
    - **Answer:** Hooks are functions that let you use state and other React features in functional components. Examples include `useState`, `useEffect`, `useContext`, and `useReducer`.
3. **Explain the useEffect hook.**
    - **Answer:** The `useEffect` hook allows you to perform side effects in functional components, such as data fetching, subscriptions, or manually changing the DOM. It runs after every render by default but can be controlled by specifying dependencies.
4. **What is Redux and how does it integrate with React?**
    - **Answer:** Redux is a state management library that provides a predictable state container for JavaScript applications. It integrates with React using the `react-redux` library, which includes components like `Provider` and hooks like `useSelector` and `useDispatch` for connecting React components to the Redux store.
5. **What is the Context API in React?**
    - **Answer:** The Context API allows you to pass data through the component tree without having to pass props down manually at every level. It is used to share data that can be considered global for a tree of React components.
6. **How do you optimize performance in a React application?**

- **Answer:** Performance optimization techniques in React include:
    - Memoization using `React.memo` and `useMemo`
    - Code splitting with React.lazy and Suspense
    - Avoiding unnecessary re-renders using `shouldComponentUpdate` or `React.PureComponent`
    - Using `useCallback` to memoize functions
    - Virtualizing long lists with libraries like react-virtualized

7. **What is the difference between `useEffect` and `useLayoutEffect`?**
    - **Answer:** `useEffect` runs asynchronously and after the render is committed to the screen, while `useLayoutEffect` runs synchronously after all DOM mutations but before the browser has painted, making it suitable for reading layout and synchronously re-rendering.
8. **How do you handle form validations in React?**
    - **Answer:** Form validations in React can be handled using controlled components along with validation libraries like Formik or Yup. You can also manage validation manually by setting state and conditionally rendering error messages.
9. **What is server-side rendering (SSR) and how is it implemented in React?**
    - **Answer:** Server-side rendering is the process of rendering React components on the server and sending the fully rendered HTML to the client. It improves performance and SEO. SSR can be implemented in React using frameworks like Next.js.
10. **Explain the concept of code splitting in React.**
    - **Answer:** Code splitting is a technique to split your code into smaller bundles that can be loaded on demand, reducing the initial load time. In React, this can be achieved using dynamic `import()` statements and React.lazy for lazy loading components.
11. **What is the difference between controlled and uncontrolled components?**
    - **Answer:** Controlled components have their state controlled by React, using state variables and event handlers, while uncontrolled components manage their own state internally and use refs to access the DOM elements.
12. **What are Pure Components in React?**
    - **Answer:** Pure Components in React are components that perform a shallow comparison of props and state to determine if a re-render is necessary. They optimize performance by preventing unnecessary renders.
13. **What is the purpose of React.PureComponent?**
    - **Answer:** React.PureComponent is a base class for React components that automatically implements `shouldComponentUpdate` with a shallow prop and state comparison to optimize performance by reducing unnecessary re-renders.
14. **What is the significance of the `key` prop in React?**
    - **Answer:** The `key` prop helps React identify which items in a list have changed, been added, or removed. It should be a unique identifier to ensure that React can efficiently update the DOM.
15. **What is the difference between React and React Native?**
    - **Answer:** React is a JavaScript library for building web applications, while React Native is a framework for building mobile applications using React. React Native allows you to use React components to build native mobile apps for iOS and Android.
16. **How do you handle asynchronous operations in React?**

- **Answer:** Asynchronous operations in React can be handled using JavaScript promises, `async/await` syntax, or libraries like axios for data fetching. The results can be managed in state using hooks like `useState` and `useEffect`.

17. **What is the difference between `useCallback` and `useMemo`?**
    - **Answer:** `useCallback` is used to memoize functions, preventing them from being recreated on every render, while `useMemo` is used to memoize values or expensive calculations to avoid recomputation on every render.
18. **How do you manage global state in a React application?**
    - **Answer:** Global state in a React application can be managed using state management libraries like Redux, MobX, or React's Context API combined with hooks.
19. **What is the significance of `React.StrictMode`?**
    - **Answer:** `React.StrictMode` is a wrapper component that helps identify potential issues in a React application by intentionally invoking certain lifecycle methods twice and providing warnings for unsafe practices.
20. **How do you handle errors in a React application?**
    - **Answer:** Errors in a React application can be handled using error boundaries, which are components that catch JavaScript errors anywhere in their child component tree and display a fallback UI instead of crashing the app.
21. **What is React Suspense?**
    - **Answer:** React Suspense is a feature that allows you to suspend the rendering of components until some condition is met, such as data fetching. It works in conjunction with React.lazy for code splitting.
22. **What is the difference between client-side rendering and server-side rendering?**
    - **Answer:** Client-side rendering (CSR) renders the application in the browser using JavaScript, while server-side rendering (SSR) renders the application on the server and sends the fully rendered HTML to the client.
23. **What is React Fiber?**
    - **Answer:** React Fiber is the reimplementation of React's core algorithm for rendering, enabling more efficient and incremental rendering. It allows for features like concurrent mode, which improves the responsiveness of React applications.
24. **What are render props in React?**
    - **Answer:** Render props is a pattern where a component's children is a function that returns a React element. This pattern allows for sharing logic between components using a prop that is a function.
25. **How do you handle animations in React?**
    - **Answer:** Animations in React can be handled using CSS animations, libraries like React Transition Group, or animations libraries like Framer Motion and react-spring.
26. **What is the difference between `componentWillMount` and `componentDidMount`?**
    - **Answer:** `componentWillMount` is called before the component is mounted, allowing for initial state setup. `componentDidMount` is called after the component is mounted, making it suitable for data fetching and DOM operations. Note that `componentWillMount` is deprecated in favor of the constructor.
27. **How do you handle routing in a React application?**

  - **Answer:** Routing in a React application is handled using libraries like React Router, which provides components and hooks for navigating between different views or pages.

28. **What are the benefits of using TypeScript with React?**
    - **Answer:** Benefits of using TypeScript with React include improved code quality, better developer experience with type checking and auto-completion, easier refactoring, and early detection of bugs.
29. **How do you use refs with functional components?**
    - **Answer:** Refs in functional components can be used with the `useRef` hook, which creates a mutable ref object that persists across renders and can be used to access DOM elements or store instance variables.
30. **What is the difference between `useReducer` and `useState`?**
    - **Answer:** `useState` is used for managing simple state in a component, while `useReducer` is used for managing more complex state logic with actions and reducers. `useReducer` is similar to Redux but is built into React.