React Technical Interview Questions

1)What is Reactjs?

2) Does React use HTML?

3) When was React first released?

4) Give me two most significant drawbacks of React

5) State the difference between Real DOM and Virtual DOM

6) What is Flux Concept In React?

7) Define the term Redux in React

8) What is the 'Store' feature in Redux?

9) What is an action in Redux?

10) Name the important features of React

11) Explain the term stateless components

12) Explain React Router

12) Explain Jsx

13) What is dispatcher?

14) What is meant by callback function? What is its purpose?

15) Explain the term high order component

16) Explain the Presentational segment

17) What are Props in react js?

18) Explain yield catchphrase in JavaScript

19) Name two types of React component

20) Explain synthetic event in React js

21) What is React State?

22) How can you update state in react js?

23) Explain the use of the arrow function in React

24) State the main difference between Props and State

25) Explain pure components in React js

26) What kind of information controls a segment in React?

27) What is 'create-react-app'?

28) Explain the use of 'key' in react list

29) What are children prop?

30) Explain error boundaries?

31) What is the use of empty tags ?

32) Explain strict mode

33) What are reacted portals?

34) What is Context?

35) What is the use of Webpack?

36) What is Babel in React js?

37) How can a browser read JSX file?

38) What are the major issues of using MVC architecture in React?

39) What can be done when there is more than one line of expression?

40) What is the reduction?

41) Explain the term synthetic events

42) When should you use the top-class elements for the function element?

43) How can you share an element in the parsing?

44) Explain the term reconciliation

45) How can you re-render a component without using setState() function?

46) Can you update props in react?

47) Explain the term 'Restructuring.'

48) Can you update the values of props?

49) Explain the meaning of Mounting and Demounting

50) What is the use of ‘props-types’ liberary?

51. What is React?

52) What is useState() in React ?

53) What is JSX?

54) What are the differences between controlled and uncontrolled components?

55) When rendering a list what is a key and what is it's purpose?

56) Why do React Hooks make use of refs?

57) What is React Hooks?

58) Differentiate between States and Props.

59) What is an event in React?

60) What is Virtual DOM?

61) What is context?

62) What is children prop?

63) What are Life Cycle Methods in React?

64) What are the limitations of React?

65) What is the difference between Shadow DOM and Virtual DOM?

66) What is the difference between React and ReactDOM?

67) What is the purpose of ReactTestUtils package?

68) What are the core principles of Redux?

69) How to access Redux store outside a component?

70) What is the difference between React context and React Redux?

71) What is the difference between component and container in React Redux?

72) What is the difference between HTML and React event handling?

73) What is Lifting State Up in React?

74) What are fragments?

75) Why fragments are better than container divs?

76) Why should we not update the state directly?

77) Differentiate React Hooks vs Classes.

78) What is Jest?

79) How is React different from Angular?

80) What are the different types of Hooks in React?

81) Explain the steps to create a react application and print hello world?

82) How are comments written in React?

83) Explain the concept of lifting state up

84) What is the purpose of setState in React?

85) Explain the useEffect hook

86) What are controlled components in React?

87) What is the purpose of the useReducer hook?

88) What is the significance of the key attribute in React lists?

89) What is the difference between class components and functional components?

90) Explain the concept of refs in React

91) Explain the purpose of the useContext hook.

92) What is the significance of the dangerouslySetInnerHTML property in React?

93) What is the purpose of the componentDidMount lifecycle method?

94) What is the React developer tool

95) How does React handle prop drilling, and how can it be avoided?

96) What is the purpose of the shouldComponentUpdate method?

97) What is the significance of the key prop in React Router?

98) What is the purpose of the forwardRef function in React?

99) Explain the concept of error boundaries in React

100) What is the significance of the memo function in React?

101) How does React handle forms?

102) Explain the purpose of the useMemo hook.

103) What is the significance of the useCallback hook?

104) What are React portals?

105) Explain the concept of suspense in React

106) What is the purpose of the useEffect cleanup function?

107) How does React handle routing?

108) What is the purpose of the useLayoutEffect hook?

109) What is the significance of the React.memo function?

110) How does React handle code splitting?

111) What is the purpose of the useImperativeHandle hook?

112) Explain the concept of the useDebugValue hook

113) Explain the significance of the SuspenseList component in React.

114) What is the significance of the react-scripts package in a React application?

115) What is middleware in Redux

116) Explain data flow in Redux

117) What is Redux-Thunk

118) What is Redux-Saga, Difference between Redux-thunk and Redux-sag

119) What is useRef in React js

120) What is server side render in React js

121) What is server side render in React js

122) What is node module in React js

123) What is the default localhost server port in react js. how can we change the local server port

124) How to optimize React js app

125) what is context api in React js

126) What is super, constructor, render function in React js

127) What is React.js and how does it differ from other JavaScript libraries?

128) What are the advantages of using React.js?

129) How does React.js handle updates and rendering?

130) What are the components in React.js and how are they used?

131) What is JSX and how is it used in React.js?

132) How do you use event handling in React.js?

133) What is the significance of props in React.js?

134) How do you use forms and form validation in React.js?

135) How do you use React.js with a state management library such as Redux?

136) How do you use Hooks in React.js?

137) How do you use Context API in React.js?

138) How can you optimize the performance of a React.js application?

139) How do you test React.js components?

140) How does Server-side rendering work in React.js?

141) How does React.js handle different types of errors?

142) Can you explain the concept of "lifting state up" in React and why it's important?

143) Can you explain the use of Redux with React and how it differs from using React's built-in state management?

144) Difference between React's built-in state management and Redux:

145) Can you explain the concept of "reactive updates" in React and how it differs from traditional data binding?

146) Can you explain how React handles performance optimization, such as lazy loading and memoization?

React.JS Interview Questions

And Answers

1)What is Reactjs?

React is a JavaScript library that makes building user interfaces easy. It was developed by Facebook.

2) Does React use HTML?

No, It uses JSX, which is similar to HTML.

3) When was React first released?

React was first released on March 2013.

4) Give me two most significant drawbacks of React

- Integrating React with the MVC framework like Rails requires complex configuration.
- React require the users to have knowledge about the integration of user interface into MVC framework.

5) State the difference between Real DOM and Virtual DOM

Real DOM	Virtual DOM
It is updated slowly.	It updates faster
It allows a direct update from HTML.	It cannot be used to update HTML directly.
It wastes too much memory.	Memory consumption is less

6) What is Flux Concept In React?

Facebook widely uses flux architecture concept for developing client-side web applications. It is not a framework or a library. It is simply a new kind of architecture that complements React and the concept of Unidirectional Data Flow.

7) Define the term Redux in React

Redux is a library used for front end development. It is a state container for JavaScript applications which should be used for the applications state management. You can test and run an application developed with Redux in different environments.

8) What is the 'Store' feature in Redux?

Redux has a feature called 'Store' which allows you to save the application's entire State at one place. Therefore all it's component's State are stored in the Store so that you will get regular updates directly from the Store. The single state tree helps you to keep track of changes over time and debug or inspect the application.

9) What is an action in Redux?

It is a function which returns an action object. The action-type and the action data are always stored in the action object. Actions can send data between the Store and the software application. All information retrieved by the Store is produced by the actions.

10) Name the important features of React

Here, are important features of React.

- Allows you to use 3rd party libraries
- Time-Saving
- Faster Development
- Simplicity and Composable
- Fully supported by Facebook.
- Code Stability with One-directional data binding React Components

11) Explain the term stateless components

Stateless components are pure functions that render DOM-based solely on the properties provided to them.

12) Explain React Router

React Router is a routing library which allows you to add new screen flows to your application, and it also keeps URL in sync with what's being shown on the page.

12) Explain JSX

JSX (JavaScript XML) is a syntax extension for JavaScript recommended by React for describing what the UI should look like.

13) What is dispatcher?

A dispatcher is a central hub of app where you will receive actions and broadcast payload to registered callbacks.

14) What is meant by callback function? What is its purpose?

A callback function should be called when setState has finished, and the component is retendered. As the setState is asynchronous, which is why it takes in a second callback function.

15) Explain the term high order component?

A higher-order component also shortly known as HOC is an advanced technique for reusing component logic. It is not a part of the React API, but they are a pattern which emerges from React's compositional nature.

16) Explain the Presentational segment

A presentational part is a segment which allows you to renders HTML. The segment's capacity is presentational in markup.

17) What are Props in react js?

Props mean properties, which is a way of passing data from parent to child. We can say that props are just a communication channel between components. It is always moving from parent to child component, and they are Immutable.

18) Explain yield catchphrase in JavaScript

The yield catchphrase is utilized to delay and resume a generator work, which is known as yield catchphrase.

19) Name two types of React component

Two types of react Components are:

- Function component
- Class component

20) Explain synthetic event in React js

Synthetic event is a kind of object which acts as a cross-browser wrapper around the browser's native event. It also helps us to combine the behaviors of various browser into signal API.

21) What is React State?

It is an object which decides how a specific component renders and how it behaves. The state stores the information which can be changed over the lifetime of a React component.

22) How can you update state in react js?

A state can be updated on the component directly or indirectly.

23) Explain the use of the arrow function in React

The arrow function helps you to predict the behavior of bugs when passed as a callback. Therefore, it prevents bug caused by this all together.

24) State the main difference between Pros and State

The main difference the two is that the State is mutable and Pros are immutable.

25) Explain pure components in React js

Pure components are the fastest components which can replace any component with only a render(). It helps you to enhance the simplicity of the code and performance of the application.

26) What kind of information controls a segment in React?

There are mainly two sorts of information that control a segment: State and Props

- **State:** State information that will change, we need to utilize State.
- Props: Props are set by the parent and which are settled all through the lifetime of a part.

27) What is 'create-react-app'?

'create-react-app' is a command-line tool which allows you to create one basic react application.

28) Explain the use of 'k' in react list

Keys are used to uniquely identify and differentiate between components in React. They help React identify which items have changed, added, or removed.

29) What are children prop?

Children props are used to pass component to other components as properties.

30) Explain error boundaries?

Error boundaries help you to catch Javascript error anywhere in the child components. They are most used to log the error and show a fallback UI.

31) What is the use of empty tags ?

Empty tags are used in React for declaring fragments.

32) Explain strict mode

StrictMode allows you to run checks and warnings for react components. It runs only on development build. It helps you to highlight the issues without rendering any visible UI.

33) What are reacted portals?

Portal allows you to render children into a DOM node. CreatePortalmethod is used for it.

34) What is Context?

React context helps you to pass data using the tree of react components. It helps you to share data globally between various react components.

35) What is the use of Webpack?

Webpack in basically is a module builder. It is mainly runs during the development process.

36) What is Babel in React js?

Babel, is a JavaScript compiler that converts latest JavaScript like ES6, ES7 into plain old ES5 JavaScript that most browsers understand.

37) How can a browser read JSX file?

If you want the browser to read JSX, then that JSX file should be replaced using a JSX transformer like Babel and then send back to the browser.

38) What are the major issues of using MVC architecture in React?

Here are the major challenges you will face while handling MVC architecture:

- DOM handling is quite expensive
- Most of the time applications were slow and inefficient
- Because of circular functions, a complex model has been created around models and ideas

39) What can be done when there is more than one line of expression?

At that time a multi-line JSX expression is the only option left for you.

40) What is the reduction?

The reduction is an application method of handling State.

41) Explain the term synthetic events

It is actuallya cross-browser wrapper around the browser's native event. These events have interface stopPropagation() and preventDefault().

42) When should you use the top-class elements for the function element?

If your element does a stage or lifetime cycle, we should use top-class elements.

43) How can you share an element in the parsing?

Using the State, we can share the data.

44) Explain the term reconciliation

When a component's state or props change then rest will compare the rendered element with previously rendered DOM and will update the actual DOM if it is needed. This process is known as reconciliation.

45) How can you re-render a component without using setState() function?

You can use forceUpdate() function for re-rending any component.

46) Can you update props in react?

You can't update props in react js because props are read-only. Moreover, you can not modify props received from parent to child.

47) Explain the term 'Restructuring.'

Restructuring is extraction process of array objects. Once the process is completed, you can separate each object in a separate variable.

48) Can you update the values of props?

It is not possible to update the value of props as it is immutable.

49) Explain the meaning of Mounting and Demounting

- The process of attaching the element to the DCOM is called mounting.
- The process of detaching the element from the DCOM is called the demounting process.

50) What is the use of 'props-types' liberary?

'Prop-types' library allows you to perform runtime type checking for props and similar object in a recent application.

51. What is React?

React is a JavaScript library focused on creating declarative user interfaces (UIs) using a component-based concept. It's used for handling the view layer and can be used for web and mobile apps.

52) What is useState() in React ?

The React **useState** Hook allows us to track state in a function component.

State generally refers to data or properties that need to be tracking in an application.

To use the useState Hook, we first need to import it into our component

. **useState accepts an initial state and returns two values**:

- The current state.
- A function that updates the state.

53) What is JSX?

JSX stands for JavaScript XML. It allows us to write HTML inside JavaScript and place them in the DOM without using functions like appendChild( ) or createElement( ). As stated in the official docs of React, JSX provides syntactic sugar for React.createElement( ) function.

54) What are the differences between controlled and uncontrolled components?

A Controlled Component is one that takes its current value through props and notifies changes through callbacks like onChange.

A Uncontrolled Component is one that stores its own state internally, and you query the DOM using a ref to find its current value when you need it.

55) When rendering a list what is a key and what is it's purpose?

Keys help React identify which items have changed, are added, or are removed. Keys should be given to the elements inside the array to give the elements a stable identity. The best way to pick a key is to use a string that uniquely identifies a list item among its siblings.

56) Why do React Hooks make use of refs?

Earlier, refs were only limited to class components but now it can also be accessible in function components through the useRef Hook in React.

The refs are used for:

- Managing focus, media playback, or text selection.

57) What is React Hooks?

Hooks allow function components to have access to state and other React features. Because of this, class components are generally no longer needed.

58) Differentiate between States and Props.

The major differences between States and Props are given below.

SN	Props	State
1.	Props are read-only.	State changes can be asynchronous.
2.	Props are immutable.	State is mutable.
3.	Props allow you to pass data from one component to other components as an argument.	State holds information about the components.
4.	Props can be accessed by the child component.	State cannot be accessed by child components.

59) What is an event in React?

An event is an action which triggers as a result of the user action or system generated event like a mouse click, loading of a web page, pressing a key, window resizes, etc. In React, the event handling system is very similar to handling events in DOM elements. The React event handling system is known as Synthetic Event, which is a cross-browser wrapper of the browser's native event.

60) What is Virtual DOM?

Virtual DOM is just a copy of the original DOM kept in the memory and synced with the real DOM by libraries such as ReactDOM. Virtual DOM has the same properties that of the Real DOM, but it lacks the power to directly change the content of the screen.

The Virtual DOM (VDOM) is an in-memory representation of Real DOM. The representation of a UI is kept in memory and synced with the "real" DOM. It's a step that happens between the render function being called and the displaying of elements on the screen. This entire process is called reconciliation.

61) What is context?

Context provides a way to pass data through the component tree without having to pass props down manually at every level. For example, authenticated user, locale preference, UI theme need to be accessed in the application by many components.

Eg: const { Provider, Consumer } = React.createContext(defaultValue)

62) What is children prop?

Children is a prop (this.props.children) that allow you to pass components as data to other components, just like any other prop you use. Component tree put between component's opening and closing tag will be passed to that component as children prop.

63) What are Life Cycle Methods in React?

There are four different phases in the lifecycle of React component. They are:

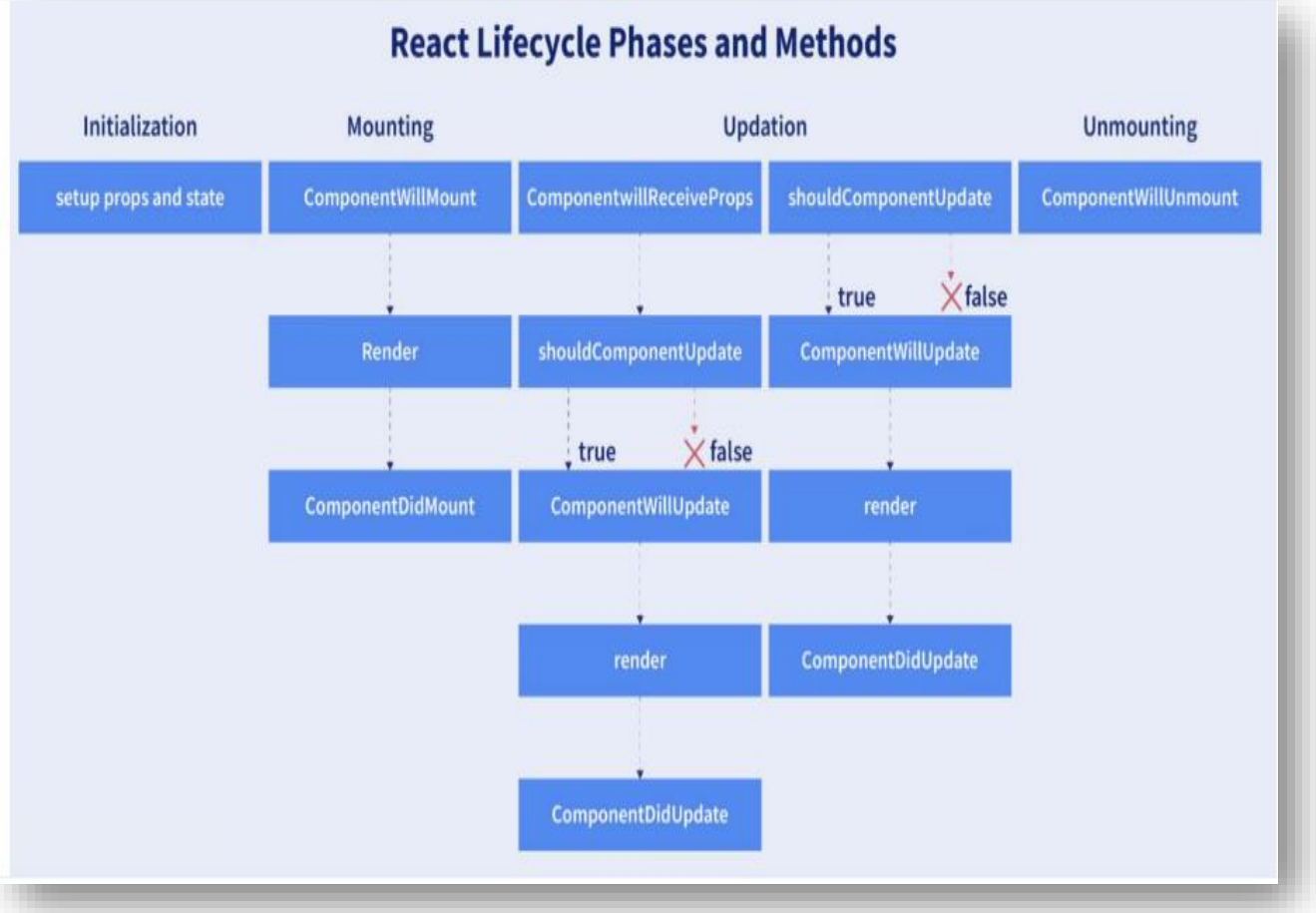
Initialization: During this phase, React component will prepare by setting up the default props and initial state for the upcoming tough journey.

Mounting: Mounting refers to putting the elements into the browser DOM. Since React uses VirtualDOM, the entire browser DOM which has been currently rendered would not be

refreshed. This phase includes the lifecycle methods componentWillMount and componentDidMount.

Updating: In this phase, a component will be updated when there is a change in the state or props of a component. This phase will have lifecycle methods like componentWillUpdate, shouldComponentUpdate, render, and componentDidUpdate.

Unmounting: In this last phase of the component lifecycle, the component will be removed from the DOM or will be unmounted from the browser DOM. This phase will have the lifecycle method named componentWillUnmount.



64) What are the limitations of React?

The few limitations of React are as given below:

- React is not a full-blown framework as it is only a library.

- The components of React are numerous and will take time to fully grasp the benefits• of all.
- It might be difficult for beginner programmers to understand React.
- Coding might become complex as it will make use of inline templating and JSX.

65) What is the difference between Shadow DOM and Virtual DOM?

The Shadow DOM is a browser technology designed primarily for scoping variables and CSS in web components. The Virtual DOM is a concept implemented by libraries in JavaScript on top of browser APIs.

66) What is the difference between React and ReactDOM?

The react package contains React.createElement(), React.Component, React.Children, and other helpers related to elements and component classes. You can think of these as the isomorphic or universal helpers that you need to build components. The react-dom package contains ReactDOM.render(), and in react-dom/server we have server-side rendering support with ReactDOMServer.renderToString() and ReactDOMServer.renderToStaticMarkup().

67) What is the purpose of ReactTestUtils package?

ReactTestUtils are provided in the with-addons package and allow you to perform actions against a simulated DOM for the purpose of unit testing.

68) What are the core principles of Redux?

Redux follows three fundamental principles:

- **Single source of truth:** The state of your whole application is stored in an object tree within a single store. The single state tree makes it easier to keep track of changes over time and debug or inspect the application.
- **State is read-only:** The only way to change the state is to emit an action, an object describing what happened. This ensures that neither the views nor the network callbacks will ever write directly to the state.
- **Changes are made with pure functions:** To specify how the state tree is transformed by actions, you write reducers. Reducers are just pure functions that take the previous state and an action as parameters, and return the next state.

69) How to access Redux store outside a component?

You just need to export the store from the module where it created with createStore(). Also, it shouldn't pollute the global window object.

70) What is the difference between React context and React Redux?

You can use Context in your application directly and is going to be great for passing down data to deeply nested components which what it was designed for. Whereas Redux is much more powerful and provides a large number of features that the Context API doesn't provide. Also, React Redux uses context internally but it doesn't expose this fact in the public API.

71) What is the difference between component and container in React Redux?

Component is a class or function component that describes the presentational part of your application. Container is an informal term for a component that is connected to a Redux store. Containers subscribe to Redux state updates and dispatch actions, and they usually don't render DOM elements; they delegate rendering to presentational child components.

72) What is the difference between HTML and React event handling?

i. In HTML, the event name usually represents in *lowercase* as a convention:

```
<button onclick='activateLasers()'>
```

Whereas in React it follows *camelCase* convention:

```
<button onClick={activateLasers}>
```

ii. In HTML, you can return `false` to prevent default behavior:

```
<a href='#' onclick='console.log("The link was clicked."); return false;' />
```

Whereas in React you must call `preventDefault()` explicitly:

```
function handleClick(event) {
  event.preventDefault()
  console.log('The link was clicked.')
}
```

iii. In HTML, you need to invoke the function by appending `()` Whereas in react you should not append `()` with the function name. (refer "activateLasers" function in the first point for example)

73) What is Lifting State Up in React?

When several components need to share the same changing data then it is recommended to lift the shared state up to their closest common ancestor. That means if two child components share the same data from its parent, then move the state to parent instead of maintaining local state in both of the child components.

74) What are fragments?

It's a common pattern in React which is used for a component to return multiple elements. *Fragments* let you group a list of children without adding extra nodes to the DOM.

```
render() {
  return (
    <React.Fragment>
      <ChildA />
      <ChildB />
      <ChildC />
    </React.Fragment>
  )
}
```

There is also a *shorter syntax*, but it's not supported in many tools:

```
render() {
  return (
    <>
      <ChildA />
      <ChildB />
      <ChildC />
    </>
  )
}
```

75) Why fragments are better than container divs?

Below are the list of reasons,

- Fragments are a bit faster and use less memory by not creating an extra DOM node. This only has a real benefit on very large and deep trees.
- Some CSS mechanisms like Flexbox and CSS Grid have a special parent-child relationships, and adding divs in the middle makes it hard to keep the desired layout.
- The DOM Inspector is less cluttered.

76) Why should we not update the state directly?

If you try to update the state directly then it won't re-render the component.

```
//Wrong
this.state.message = 'Hello world'
```

Instead use `setState()` method. It schedules an update to a component's state object. When state changes, the component responds by re-rendering.

```
//Correct
this.setState({ message: 'Hello World' })
```

Note: You can directly assign to the state object either in *constructor* or using latest javascript's class field declaration syntax.

77) Differentiate React Hooks vs Classes.

React Hooks	Classes
It is used in functional components of React.	It is used in class-based components of React.
It will not require a declaration of any kind of constructor.	It is necessary to declare the constructor inside the class component.
It does not require the use of `this` keyword in state declaration or modification.	Keyword `this` will be used in state declaration (`this.state`) and in modification (`this.setState()`).
It is easier to use because of the `useState` functionality.	No specific function is available for helping us to access the state and its corresponding setState variable.
React Hooks can be helpful in implementing Redux and context API.	Because of the long setup of state declarations, class states are generally not preferred.

78) What is Jest?

Jest is a JavaScript unit testing framework created by Facebook based on Jasmine and provides automated mock creation and a jsdom environment. It's often used for testing components.

79) How is React different from Angular?

	Angular	React
Author	Google	Facebook Community
Developer	Misko Hevery	Jordan Walke
Initial Release	October 2010	March 2013
Language	JavaScript, HTML	JSX
Type	Open Source MVC Framework	Open Source JS Framework
Rendering	Client-Side	Server-Side
Data-Binding	Bi-directional	Uni-directional
DOM	Regular DOM	Virtual DOM
Testing	Unit and Integration Testing	Unit Testing
App Architecture	MVC	Flux
Performance	Slow	Fast, due to virtual DOM.

80) What are the different types of Hooks in React?

Basic Hooks

useState()

Used to manage and retrieve state in functional components.

useEffect()

Enables performing side effects in functional components, like data fetching or DOM manipulation.

useContext()

Creates shared data accessible by components in a hierarchy without passing props through each level.

Additional Hooks

useReducer()

Helpful for complex state logic or when the next state depends on the previous state, optimizing performance by passing dispatch down. Medium

useCallback()

Useful when passing callbacks to optimized child components to prevent unnecessary renders by checking reference equality.

useImperativeHandle()

Allows modifying the instance passed with a ref object. useDebugValue() Displays a label for custom hooks in React DevTools.

useRef()

Creates a reference to a DOM element directly within a functional component.

useLayoutEffect()

Reads layout from the DOM and triggers synchronous rerendering.

Custom Hooks

These are functions in JavaScript that follow React's Hook rules and begin with "use." They help extract component logic into reusable functions, making your code more modular and easier to understand.

81) Explain the steps to create a react application and print hello world?

Steps to Create a React Application

- **Install Node**
  Before installing React, ensure that Node is installed on your computer. You can download it from **Node.js,**
- **Create React App**
  Open the terminal and run the following command to create a new React application (replace my-react-app with your preferred application name):

```
JSX
npx create-react-app my-react-app
```

Navigate to the Application Folder

Move to the newly created application folder:

```
JSX
cd my-react-app
```

Print "**Hello World!**" Example

Now, open the src/App.js file and replace its content with the following:

```jsx
import React from 'react';

function App() {
  return (
    <div>
      <h1>Hello World!</h1>
    </div>
  );
}

export default App;
```

Save the file.

Run the Application

In the terminal, run the following command to start the development serve

```jsx
npm start
```

This will open your new React application in a web browser, and you should see "Hello World!" displayed on the webpage.

In this example, the App component is a simple React function component that returns JSX to render the "Hello World!" message. The npm start command is used to run the application and launch a development server.

82) How are comments written in React?

Comments in React/JSX are similar to JavaScript multiline comments but are enclosed in curly braces.

Single-line comments

```jsx
<div>
  {/* Single-line comments(In vanilla JavaScript, the
single-line comments are represented by double
slash(//)) */}
  {`Welcome, ${userName}! Let's dive into React`}
</div>
```

Multi-line comments

```jsx
<div>
  {/*
    This is a multiline comment in React.
    It provides additional information about the
code.
  */}
  {`Welcome, ${userName}! Let's dive into React`}
</div>
```

In these modified examples, the comments now convey a welcoming message to the user, demonstrating how comments can be used to explain and document code within the JSX structure.

83) Explain the concept of lifting state up

Lifting state up is a pattern where the state of a child component is moved to its parent component, allowing multiple child components to share the same state

84) What is the purpose of setState in React?

setState is used to update the state of a component and trigger a re-render

85) Explain the useEffect hook

The useEffect hook in React is used for side effects in functional components, such as data fetching, subscriptions, or manually changing the DOM.

86) What are controlled components in React?

Controlled components are components where the form data is controlled by React state. The input elements receive their current value from the state and have their value updated through a callback function.

87) What is the purpose of the useReducer hook?

The useReducer hook is used for managing complex state logic in React applications. It is an alternative to useState when state transitions are more complex.

88) What is the significance of the key attribute in React lists?

The key attribute is used to uniquely identify elements in a list. It helps React efficiently update the DOM when the list changes.

89) What is the difference between class components and functional components?

class components use ES6 classes and have additional features like state and lifecycle methods, while functional components are simpler and are often used with hooks.

90) Explain the concept of refs in React

Refs are used to access the DOM directly or to reference a React element. They provide a way to interact with the underlying DOM nodes in React.

91) Explain the purpose of the useContext hook.

The useContext hook is used to access the value of a React context within a functional component.

92) What is the significance of the dangerouslySetInnerHTML property in React?

dangerouslySetInnerHTML is used to inject HTML directly into a component, but it should be used with caution to avoid cross-site scripting (XSS) vulnerabilities.

93) What is the purpose of the componentDidMount lifecycle method?

componentDidMount is invoked immediately after a component is mounted, making it suitable for initial AJAX requests or setting up subscriptions.

94) What is the React developer tool

The React Developer Tools is a browser extension that allows developers to inspect and debug React component hierarchies in the Chrome and Firefox browsers.

95) How does React handle prop drilling, and how can it be avoided?

Prop drilling occurs when props are passed down through multiple levels of components. It can be avoided by using context or state management libraries like Redux.

96) What is the purpose of the shouldComponentUpdate method?

shouldComponentUpdate is a lifecycle method that determines if a component should re-render. Developers can use it to optimize performance by preventing unnecessary renders.

97) What is the significance of the key prop in React Router?

The key prop in React Router is used to force the remounting of a component when the key changes, ensuring that the component is fully reinitialized.

98) What is the purpose of the forwardRef function in React?

forwardRef is used to forward refs through components, allowing parent components to interact with the child's DOM node.

99) Explain the concept of error boundaries in React

Error boundaries are components that catch JavaScript errors anywhere in their child component tree and log those errors, display a fallback UI, or take other actions.

100) What is the significance of the memo function in React?

memo is a higher-order component that memoizes the rendering of a functional component, preventing unnecessary re-renders if the props have not changed.

101) How does React handle forms?

React handles forms by using controlled components, where form data is controlled by the React state.

102) Explain the purpose of the useMemo hook.

The useMemo hook is used to memoize the result of a function, preventing unnecessary calculations and improving performance.

103) What is the significance of the useCallback hook?

useCallback is used to memoize callback functions, preventing them from being recreated on every render.

104) What are React portals?

React portals provide a way to render children into a DOM node that exists outside the parent component's hierarchy.

105) Explain the concept of suspense in React

Suspense is a feature in React that allows components to "wait" for something before rendering, such as data fetching or code splitting.

106) What is the purpose of the useEffect cleanup function?

The cleanup function in useEffect is used to perform cleanup tasks, such as unsubscribing from subscriptions or clearing intervals, when a component is unmounted.

107) How does React handle routing?

React can handle routing using the React Router library, which provides a way to navigate between different views or pages in a React application.

108) What is the purpose of the useLayoutEffect hook?

useLayoutEffect is similar to useEffect, but it fires synchronously after all DOM mutations. It is often used for measuring and synchronizing layout

109) What is the significance of the React.memo function?

React.memo is a higher-order component that memoizes the rendering of a functional component, preventing unnecessary re-renders if the props have not changed.

110) How does React handle code splitting?

React supports code splitting, allowing developers to split their code into smaller chunks that are loaded on demand, improving performance by reducing the initial bundle size.

111) What is the purpose of the useImperativeHandle hook?

useImperativeHandle is used to customize the instance value that is exposed when using React.forwardRef.

112) Explain the concept of the useDebugValue hook

useDebugValue is used to display a label for custom hooks in React DevTools.

113) Explain the significance of the SuspenseList component in React.

SuspenseList is a component that allows developers to coordinate the loading of multiple components in a way that provides a better user experience.

113) **What is the purpose of the useReducer hook?**

useReducer is a hook in React used for state management in functional components. It is particularly useful when the state logic is complex and involves multiple sub-values or when the next state depends on the previous state.

114) What is the significance of the react-scripts package in a React application?

The react-scripts package is a set of scripts from the create-react-app starter pack which helps you kick off projects without configuring. The react-scripts start command sets up the development environment and starts a server, as well as hot module reloading.

115) What is middleware in Redux

Redux middleware solves different problems than Express or Koa middleware, but in a conceptually similar way. It provides a third-party extension point between dispatching an action, and the moment it reaches the reducer. People use Redux middleware for logging, crash reporting, talking to an asynchronous API, routing, and more.

116) Explain data flow in Redux

The data flow of Redux is the base of the library. It is one of the first things that we learn when we start to study Redux. You dispatch an action, that is a plain object, to the store. This updates the state using the reducer function and this new state returns to the application, updating the UI.

117) What is Redux-Thunk

Redux-Thunk is a middleware for Redux that allows you to write action creators that return a function instead of an action object.

118) What is Redux-Saga, Difference between Redux-thunk and Redux-Saga

Redux-saga is a library that aims to make application side effects (i.e. asynchronous things like data fetching and impure things like accessing the browser cache) easier to manage, more efficient to execute, easy to test, and better at handling failures.

Redux Thunk and Redux Saga are both middleware libraries for Redux that handle asynchronous actions, but they have different approaches and use cases. Redux Thunk is simpler and more straightforward, making it suitable for smaller projects or simpler asynchronous operations.

119) What is useRef in React js

useRef is a React Hook that lets you reference a value that’s not needed for rendering. Call useRef at the top level of your component to declare a ref. // ... See more examples below. initialValue: The value you want the ref object’s current property to be initially. It can be a value of any type. This argument is ignored after the initial render.

120) What is server side render in React js

Server-Side Rendering (SSR) in React is a technique that involves rendering React components on the server side instead of the client side (browser). Traditionally, React applications are rendered on the client side, meaning that the browser downloads the JavaScript bundle, executes it, and renders the UI.

121) What is server side render in React js

Server-Side Rendering (SSR) in React is a technique that involves rendering React components on the server side instead of the client side (browser). Traditionally, React applications are rendered on the client side, meaning that the browser downloads the JavaScript bundle, executes it, and renders the UI.

122) What is node module in React js

node_modules are one of the most important directories in your React project as React requires node_modules to run. The node_modules directory is where all the dependancies packages are stored that are used to build and run your react project.

123) What is the default localhost server port in react js. how can we change the local server port

The default port it assigns is port:3000.

to change the default port of the react app. We need to create the .env file inside the project directory and add the environment variable. Users need to add the below code inside the .env file.

124) How to optimize React js app

https://dev.to/devland/how-to-optimize-your-react-apps-performance-1045

125) what is context api in React js

The Context API provides a means to share values like state, functions, or any data across the component tree without passing props down manually at every level. This is particularly useful for global data that many components need to access

126) What is super constructor function in React js

super() will call the constructor of its parent class. This is required when you need to access some variables from the parent class. In React, when you call super with props, React will make props available across the component through this.props

127) What is React.js and how does it differ from other JavaScript libraries?

React.js is a JavaScript library for building user interfaces.

It allows developers to create reusable UI components and manage the state and props of those components.

It differs from other JavaScript libraries in that it focuses specifically on the view layer of an application, making it a great choice for building complex, large-scale user interface

128) What are the advantages of using React.js?

Reusable components.

Virtual DOM for efficient updates and rendering.

Good performance.

Strong developer community and support.

Easy integration with other libraries and frameworks.

Can be used on the client and server side.

129) How does React.js handle updates and rendering?

When a component's state changes, React will re-render that component and its child components to reflect the new state.

React uses a virtual DOM to optimize updates by only re-rendering the specific parts of the actual DOM that have changed. This helps to improve the performance of the application.

130) What are the components in React.js and how are they used?

Components in React.js are the building blocks of a React application.

They are used to create reusable UI elements.

Components can be either functional or class-based and can be nested to create more complex UI elements.

Components accept inputs called props and manage their own state.

131) What is JSX and how is it used in React.js?

JSX is a syntax extension for JavaScript that allows developers to write HTML-like elements in their JavaScript code.

It is used in React to describe the structure and content of a component.

JSX is transpiled to plain JavaScript before being executed, so it is compatible with all web browsers.

132) How do you use event handling in React.js?

Event handling in React.js is done using the onEventName syntax, where EventName is the name of the event you want to handle, such as onClick or onSubmit.

Event handlers are passed as props to the component and are typically defined as arrow functions or bound methods.

133) What is the significance of props in React.js?

Props are used to pass data from a parent component to a child component.

Props provide a way to make components reusable and configurable.

Props components are read-only components.

134) How do you use forms and form validation in React.js?

Forms and form validation in React.js are typically implemented using controlled components, where the form input values are stored in the state and updated as the user interacts with the form.

Form validation is then performed by checking the values in the state against a set of rules.

135) How do you use React.js with a state management library such as Redux?

React.js can be used with a state management library such as Redux by integrating the Redux store with the React components. This allows for better management of shared state between components.

136) How do you use Hooks in React.js?

Hooks were introduced in React 16.8 and allow for using state and other React features without writing a class component.

Hooks make it easier to reuse logic between components and provide more flexible and concise code. They are significant because they allow for more flexible and concise code

137) How do you use Context API in React.js?

The Context API in React.js is a feature that allows for sharing data between components without passing props down through multiple levels of components.

This is useful for data that is needed by many components throughout an application.

138) How can you optimize the performance of a React.js application?

Performance of React.js applications can be optimized through techniques like using the shouldComponentUpdate lifecycle method and lazy loading.

Memoization can also be used to improve the performance of React.js applications.

139) How do you test React.js components?

React.js components can be tested using various testing libraries, such as Jest and Enzyme.

These libraries provide APIs for writing and running unit tests for React components.

140) How does Server-side rendering work in React.js?

Server-side rendering in React.js involves rendering the initial HTML on the server, rather than in the browser.

This can help improve performance, especially for slower devices or low bandwidth connections.

141) How does React.js handle different types of errors?

React.js handles different types of errors through various means, such as the try-catch statement, the use of error boundaries, and global error handling.

Error boundaries are React components that catch JavaScript errors anywhere in their child component tree, log the errors, and display a fallback UI.

142) Can you explain the concept of "lifting state up" in React and why it's important?

"Lifting state up" is a concept in React that refers to the process of sharing state between multiple components by moving it from lower-level components to higher-level components.

Here are three key points about why this is important:

1.Centralized management: By lifting state up, you can centralize the management of state in one or a few higher-level components, making it easier to understand and maintain the application.

2.Reusability: When state is lifted up, lower-level components that need access to that state can receive it as props. This makes it easier to reuse those components in different parts of the application, as they are not tightly coupled to the state they depend on.

3. Improved performance: Moving state up can also help improve performance, as React's reconciliation algorithm can take advantage of the fact that only a few components are changing instead of having to update many components individually.

Lifting state up is a critical concept in React and can help improve the structure and maintainability of your applications. By centralizing state management and making components more reusable, you can write cleaner and more efficient code

143) Can you explain the use of Redux with React and how it differs from using React's built-in state management?

Centralized store: Redux is a state management library that provides a centralized store for the entire application. The store contains the state for the whole application and can be updated using actions and reducers

Improved scalability: Redux makes it easier to manage the state of a large or complex application, as all the state is contained in a single store and updates are made using well-defined actions and reducers.

Better separation of concerns: By using Redux, you can separate the state management from the presentation of the components, making it easier to understand and maintain the application.

144) Difference between React's built-in state management and Redux:

Local vs global: React's built-in state management is local to individual components, while Redux provides a global store for the whole application.

Scalability: React's built-in state management can become cumbersome in large or complex applications, while Redux provides a more scalable solution.

Separation of concerns: React's built-in state management is closely tied to the presentation of the components, while Redux provides a more modular and scalable solution by separating the state management from the presentation.

145) Can you explain the concept of "reactive updates" in React and how it differs from traditional data binding?

Controlled by React: Controlled components in React are components that have their value and behavior controlled by React, rather than by the user or the DOM.

Better control: By controlling the value and behavior of a component, you can more easily manage the behavior of the component and ensure that it behaves as expected.

Improved reliability: Controlled components can help improve the reliability of your application, as you have more control over the behavior of the component and can ensure that it behaves as expected.

146) Can you explain how React handles performance optimization, such as lazy loading and memoization?

Reactive nature: Reactive updates in React refer to the way that React updates the user interface in response to changes in the data. React updates the UI reactively, meaning that it updates the UI in response to changes in the data.

Improved performance: Reactive updates can improve performance by only updating the parts of the UI that have changed, rather than re-rendering the entire UI.

Dynamic updates: Reactive updates allow for dynamic updates to the UI, as the UI is automatically updated in response to changes in the data.