React JS Questionnaire

React JS Questionnaire

1. What is Lazy Loading in ReactJS? # S,B, M

Lazy Loading is a feature where buckets of code are bundled together and deployed simultaneously after each block of code is executed.

2. What is the use of JSX in React?

JSX in React is a JavaScript XML file that helps write HTML code in the React framework.

3. What is the purpose of using super constructor with props argument in React? # B,

Junior

React 130

Answer

A child class constructor cannot make use of **this** reference until super() method has been called. The same applies for ES6 sub-classes as well. The main reason of passing props parameter to super() call is to access this.props in your child constructors.

4. What are the *lifecycle* methods of ReactJS class components?

Mid

React 130

Answer

component WillMount: Executed before rendering and is used for App level configuration in your root component.

component DidMount: Executed after first rendering and here all AJAX requests, DOM or state updates, and set up eventListeners should occur.

component WillReceiveProps: Executed when particular prop updates to trigger state transitions.

should ComponentUpdate: Determines if the component will be updated or not. By default it returns true. If you are sure that the component doesn't need to render after state or props are updated,

you can return false value. It is a great place to improve performance as it allows you to prevent a rerender if component receives new prop.

component WillUpdate: Executed before re-rendering the component when there are pros & state changes confirmed by shouldComponentUpdate which returns true.

component DidUpdate: Mostly it is used to update the DOM in response to prop or state changes.

component WillUnmount: It will be used to cancel any outgoing network requests, or remove all event listeners associated with the component.

5. What is StrictMode in React?

Mid

React 130

Answer

React's StrictMode is sort of a helper component that will help you write better react components, you can wrap a set of components with <StrictMode /> and it'll basically:

Verify that the components inside are following some of the recommended practices and warn you if not in the console.

Verify the deprecated methods are not being used, and if they're used strict mode will warn you in the console.

Help you prevent some side effects by identifying potential risks.

6. What is the meaning of JSX?

JSX is the abbreviation of JavaScript XML. It is a file that is used in React to bring out the essence of JavaScript to React and use it for its advantages.

It even includes bringing out HTML and the easy syntax of JavaScript. This ensures that the resulting HTML file will have high readability, thereby relatively increasing the performance of the application.

7. What is the use of an arrow function in React?

An arrow function is used to write an expression in React. It allows users to manually bind components easily. The functionality of arrow functions can be very useful when you are working with higher-order functions particularly.

8. Can AJAX be used with React?

Yes, any AJAX library, such as Axios and jQuery AJAX, can be used with React easily. One important thing is to maintain the states of the components, and here too, the props are passed

from the parents to the child components.

Child components still cannot send back props to parents, and this factor greatly increases rendering efficiency when dynamic data is considered.

9. What are stateful components in React? # S,B

Stateful components are entities that store the changes that happen and place them into the memory. Here, the state can be changed, alongside storing information such as past, current, and future changes.

10. What is the difference between cloneElement and createElement in React? # S,B, M

In React, cloneElement is primarily used to clone an element and pass it to new props directly. Whereas, createElement is the entity that JSX gets compiled into. This is also used to create elements in React.

11. Why are props passed to the super() function in React? # S,M

Props gets passed onto the super() function if a user wishes to access this.props in the constructor.

Why do we need to transpile React code?

React code is written in **JSX**, but no browser can execute **JSX** directly as they are built to read-only regular **JavaScript**.

Thus we require to use tools like **Babel** to transpile **JSX** to **JavaScript** so that the browser can execute it.

12. What are the features of React?

	JSX: JSX is a syntax extension to JavaScript. It is used with React to describe what the user interface should look like. By using JSX, we can write HTML structures in the same file that contains JavaScript code.

Components	Components: Components are the building blocks of any React application, and a single app usually consists of multiple components. It splits the user interface into independent, reusable parts that can be processed separately.
Virtual DOM	Virtual DOM: React keeps a lightweight representation of the real DOM in the memory, and that is known as the virtual DOM. When the state of an object changes, virtual DOM changes only that object in the real DOM, rather than updating all the objects.
Data binding	One-way data-binding: React's one-way data binding keeps everything modular and fast. A unidirectional data flow means that when designing a React app, you often nest child components within parent components.
High-per	High performance: React updates only those components that have changed, rather than updating all the components at once. This results in much faster web applications.

13. What is the difference between the ES6 and ES5 standards?

This is one of the most frequently asked react interview questions.

These are the few instances where ES6 syntax has changed from ES5 syntax:

Components and Function

es5

Image not found or type unknown

exports vs export

exports.

Image not found or type unknown

require vs import

14. Explain how lists work in React # S, V

We create lists in React as we do in regular JavaScript. Lists display data in an ordered format

The traversal of lists is done using the map() function

const Image not found or type unknown

15. What are forms in React? M

React employs forms to enable users to interact with web applications.

Using forms, users can interact with the application and enter the required information whenever needed. Form contain certain elements, such as text fields, buttons, checkboxes, radio buttons, etc

Forms are used for many different tasks such as user authentication, searching, filtering, indexing, etc

16. What is an arrow function and how is it used in React?

An arrow function is a short way of writing a function to React.

It is unnecessary to bind ‘this’ inside the constructor when using an arrow function. This prevents bugs caused by the use of ‘this’ in React callbacks.

17. How to create refs?

There are two approaches

This is a recently added approach. *Refs* are created using React.createRef() method and attached to React elements via the ref attribute. In order to use *refs* throughout the component, just assign the *ref* to the instance property within constructor.

```
class MyComponent extends React.Component {

  constructor(props) {

    super(props);

    this.myRef = React.createRef();

  }
```

```
  render() {
    return <div ref={this.myRef} />;
  }
}
```

You can also use ref callbacks approach regardless of React version. For example, the search bar component's input element is accessed as follows,

```
class SearchBar extends Component {
  constructor(props) {
    super(props);
    this.txtSearch = null;
    this.state = { term: "" };
    this.setInputSearchRef = (e) => {
      this.txtSearch = e;
    };
  }
  onInputChange(event) {
    this.setState({ term: this.txtSearch.value });
  }
  render() {
    return (
      <input
        value={this.state.term}
        onChange={this.onInputChange.bind(this)}
        ref={this.setInputSearchRef}
      />
    );
  }
```

}

You can also use *refs* in function components using closures. Note: You can also use inline ref callbacks even though it is not a recommended approach.

18. What is React Fiber? # S, D, M

Fiber is the new *reconciliation* engine or reimplementation of core algorithm in React v16. The goal of React Fiber is to increase its suitability for areas like animation, layout, gestures, ability to pause, abort, or reuse work and assign priority to different types of updates; and new concurrency primitives.

19. What is Lifting State Up in React? # S, D,M

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.

20. What are the different phases of component lifecycle?

The component lifecycle has three distinct lifecycle phases:

Mounting: The component is ready to mount in the browser DOM. This phase covers initialization from constructor(), getDerivedStateFromProps(), render(), and componentDidMount() lifecycle methods.

Updating: In this phase, the component gets updated in two ways, sending the new props and updating the state either from setState() or forceUpdate(). This phase covers getDerivedStateFromProps(), shouldComponentUpdate(), render(), getSnapshotBeforeUpdate() and componentDidUpdate() lifecycle methods.

Unmounting: In this last phase, the component is not needed and gets unmounted from the browser DOM. This phase includes componentWillUnmount() lifecycle method.

It's worth mentioning that React internally has a concept of phases when applying changes to the DOM. They are separated as follows

Render The component will render without any side effects. This applies to Pure components and in this phase, React can pause, abort, or restart the render.

21. What is a child's prop? #

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

There are several methods available in the React API to work with this prop. These include React.Children.map, React.Children.forEach, React.Children.count, React.Children.only, React.Children.toArray.

Before the component actually applies the changes to the DOM, there is a moment that allows React to read from the DOM through the getSnapshotBeforeUpdate().

Commit React works with the DOM and executes the final lifecycles respectively componentDidMount() for mounting, componentDidUpdate() for updating, and componentWillUnmount() for unmounting.

22. How to set state with a dynamic key name?

If you are using ES6 or the Babel transpiler to transform your JSX code then you can accomplish this with *computed property names*.

```
handleInputChange(event) {

  this.setState({ [event.target.id]: event.target.value })

}
```

23. Name two advantages of using React.js.

Candidates may mention several advantages of using React.js when responding to this question. For example, they may explain that the library helps them build high-quality user interfaces or that it permits them to write custom components.

24. In which situation would you use refs in React?

Advanced candidates should understand that they can use React refs to access a DOM element. They may also explain that they would use refs to access an element they have created to change a child component's value.

25. How would you avoid binding in React? # S, V

Candidates who have advanced React skills should be aware that they can use arrow functions in class properties to avoid binding in React. They may mention that class properties are a new feature and, to use them, a developer must enable transform-class-properties.

26. What is a *state* object? #

A ***state*** object is a plain JavaScript object that developers use in React to show information on a component's current properties. Developers can manage the ***state*** object in the component. Changing the ***state*** object causes the component to re-render

27. What is a class component?

A class component is a simple class that consists of several functions. It accepts props as arguments and returns React elements. Developers must create render functions to use class components and receive React elements.

28. What are the differences between a *Class component* and *Functional component*?

Answer

Class Components

Class-based Components uses ES6 class syntax. It can make use of the lifecycle methods.

Class components extend from React.Component.

In here you have to use this keyword to access the props and functions that you declare inside the class components.

Functional Components

Functional Components are simpler comparing to class-based functions.

Functional Components mainly focuses on the UI of the application, not on the behavior.

To be more precise these are basically render function in the class component.

Functional Components can have state and mimic lifecycle events using Reach Hooks

29. How can I make use of *Error Boundaries* in functional React components?

Answer

As of v16.2.0, there's no way to turn a functional component into an error boundary. The componentDidCatch() method works like a JavaScript catch {} block, but for components. Only class components can be error boundaries. In practice, most of the time you'll want to declare an error boundary component once and use it throughout your application.

Also bear in mind that try/catch blocks *won't work on all cases*. If a component deep in the hierarchy tries to update and fails, the try/catch block in one of the parents won't work -- because it isn't necessarily updating together with the child.

A few third party packages on npm implement error boundary hooks.

30. What do these three dots (...) in React do?

Problem

What does the ... do in this React (using JSX) code and what is it called?

<Modal {...this.props} title='Modal heading' animation={false}/>

Answer

That's property spread notation. It was added in ES2018 (spread for arrays/iterables was earlier, ES2015).

For instance, if this.props contained a: 1 and b: 2, then

<Modal {...this.props} title='Modal heading' animation={false}>

would be the same as:

<Modal a={this.props.a} b={this.props.b} title='Modal heading' animation={false}>

Spread notation is handy not only for that use case, but for creating a new object with most (or all) of the properties of an existing object — which comes up a lot when you're updating state, since you can't modify state directly:

```
this.setState(prevState => {
    return {foo: {...prevState.foo, a: "updated"}};
});
```

31. Why do class methods need to be *bound* to a class instance? #V,

Answer

In JavaScript, the value of this changes depending on the current context. Within React class component methods, developers normally expect this to refer to the current instance of a component, so it is necessary to *bind* these methods to the instance. Normally this is done in the constructor—for example:

```
class SubmitButton extends React.Component {
  constructor(props) {
    super(props);
    this.state = {
      isFormSubmitted: false
    };
    this.handleSubmit = this.handleSubmit.bind(this);
  }

  handleSubmit() {
    this.setState({
      isFormSubmitted: true
    });
```

```
  }


  render() {
    return (
      <button onClick={this.handleSubmit}>Submit</button>
    )
  }
}
```

32. Describe Flux vs MVC?M, D

Answer

Traditional MVC patterns have worked well for separating the concerns of data (Model), UI (View) and logic (Controller) — but MVC architectures frequently encounter two main problems:

Poorly defined data flow: The cascading updates which occur across views often lead to a tangled web of events which is difficult to debug.

Lack of data integrity: Model data can be mutated from anywhere, yielding unpredictable results across the UI.

With the Flux pattern complex UIs no longer suffer from cascading updates; any given React component will be able to reconstruct its state based on the data provided by the store. The Flux pattern also enforces data integrity by restricting direct access to the shared data

33. Explain why and when you would use useMemo()?

Answer

Why:

In the lifecycle of a component, React re-renders the component when an update is made. When React checks for any changes in a component, it may detect an unintended or unexpected change due to how JavaScript handles equality and shallow comparisons. This change in the React application will cause it to re-render unnecessarily.

Additionally, if that re-rendering is an expensive operation, like a long for loop, it can hurt performance. Expensive operations can be costly in either time, memory, or processing.

When:

Optimal if the wrapped function is large and expensive.

How:

Memoization is an optimization technique which passes a complex function to be memoized. In memoization, the result is “remembered” when the same parameters are passed-in subsequently.

34. What do you understand from "In React, everything is a component."

In React, components are the building blocks of React applications. These components divide the entire React application's UI into small, independent, and reusable pieces of code. React renders each of these components independently without affecting the rest of the application UI. Hence, we can say that, in React, everything is a component.

35. How are forms created in React?

Forms allow the users to interact with the application as well as gather information from the users. Forms can perform many tasks such as user authentication, adding user, searching, filtering, etc. A form can contain text fields, buttons, checkbox, radio button, etc.

React offers a stateful, reactive approach to build a form. The forms in React are similar to HTML forms. But in React, the state property of the component is only updated via setState(), and a JavaScript function handles their submission. This function has full access to the data which is entered by the user into a form.

```
import React, { Component } from 'react';


class App extends React.Component {
  constructor(props) {
     super(props);
     this.state = {value: ''};
     this.handleChange = this.handleChange.bind(this);
     this.handleSubmit = this.handleSubmit.bind(this);
  }
  handleChange(event) {
     this.setState({value: event.target.value});
  }
  handleSubmit(event) {
     alert('You have submitted the input successfully: ' + this.state.value);
     event.preventDefault();
  }
  render() {
```

```
    return (
      <form onSubmit={this.handleSubmit}>
        <h1>Controlled Form Example</h1>
        <label>
          Name:
          <input type="text" value={this.state.value} onChange={this.handleChange} />
        </label>
        <input type="submit" value="Submit" />
      </form>
    );
  }
}
export default App;
```

36. What are Pure Components?

Pure components introduced in React 15.3 version. The React.Component and React.PureComponent differ in the shouldComponentUpdate() React lifecycle method. This method decides the re-rendering of the component by returning a boolean value (true or false). In React.Component, the shouldComponentUpdate() method returns true by default. But in React.PureComponent, it compares the changes in state or props to re-render the component. The pure component enhances the simplicity of the code and performance of the application.

37. How to write comments in React?

In React, we can write comments as we write comments in JavaScript. It can be in two ways:

1. Single Line Comments: We can write comments as /* Block Comments */ with curly braces:

```
{/* Single Line comment */}
```

2. Multiline Comments: If we want to comment more that one line, we can do this as

```
{ /*
   Multi
   line
   comment
*/ }
```

38. What are fragments?

It was introduced in React 16.2 version. In React, Fragments are used for components to return multiple elements. It allows you to group a list of multiple children without adding an extra node to the DOM.

Example

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

There is also a shorthand syntax exists for declaring Fragments, but it's not supported in many tools:

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

39. How to apply validation on props in React? M

Props validation is a tool which helps the developers to avoid future bugs and problems. It makes your code more readable. React components used special property PropTypes that help you to catch bugs by validating data types of values passed through props, although it is not necessary to define components with propTypes.

We can apply validation on props using App.propTypes in React component. When some of the props are passed with an invalid type, you will get the warnings on JavaScript console. After specifying the validation patterns, you need to set the App.defaultProps.

```
class App extends React.Component {
```

```
    render() {}
}

Component.propTypes = { /*Definition */};
```

40. What is create-react-app?

Create React App is a tool introduced by Facebook to build React applications. It provides you to create single-page React applications. The create-react-app are preconfigured, which saves you from time-consuming setup and configuration like Webpack or Babel. You need to run a single command to start the React project, which is given below.

```
$ npx create-react-app my-app
```

This command includes everything which we need to build a React app. Some of them are given below:

It includes React, JSX, ES6, and Flow syntax support.

It includes Autoprefixed CSS, so you don't need -webkit- or other prefixes.

It includes a fast, interactive unit test runner with built-in support for coverage reporting.

It includes a live development server that warns about common mistakes.

It includes a build script to bundle JS, CSS, and images for production, with hashes and source maps.

41, Girivihar Society Taloda Road Nandurbar. Is it possible for a web browser to read JSX directly?

Web browsers can't read JSX directly. This is because the web browsers are built to read the regular JS objects only, and JSX is not a regular JavaScript object.

If you want a web browser to read a JSX file, you must transform the files into a regular JavaScript object. For this purpose, Babel is used.

42. What are Forward Refs? #M

Ref forwarding is a feature which is used for passing a ref through a component to one of its child components. It can be performed by making use of the React.forwardRef() method. It is particularly useful with higher-order components and specially used in reusable component libraries.

Example

```
import React, { Component } from 'react';

import { render } from 'react-dom';
```

```
const TextInput = React.forwardRef((props, ref) => (
  <input type="text" placeholder="Hello World" ref={ref} />
));

const inputRef = React.createRef();

class CustomTextInput extends React.Component {
  handleSubmit = e => {
    e.preventDefault();
    console.log(inputRef.current.value);
  };
  render() {
    return (
      <div>
        <form onSubmit={e => this.handleSubmit(e)}>
          <TextInput ref={inputRef} />
          <button>Submit</button>
        </form>
      </div>
    );
  }
}
```

43. Which is the preferred option: callback refs or findDOMNode()? #D, M

The preferred option is to use callback refs over findDOMNode() API. Because callback refs give better control when the refs are set and unset whereas findDOMNode() prevents certain improvements in React in the future.

```
class MyComponent extends Component {
  componentDidMount() {
    findDOMNode(this).scrollIntoView()
```

```
  }
  render() {
    return <div />
  }
}
```

The recommended approach is:

```
class MyComponent extends Component {
  componentDidMount() {
    this.node.scrollIntoView()
  }
  render() {
    return <div ref={node => this.node = node} />
  }
}
class MyComponent extends Component {
  componentDidMount() {
    this.node.scrollIntoView()
  }
  render() {
    return <div ref={node => this.node = node} />
  }
}
```

44. How is React Router different from Conventional Routing?

The difference between React Routing and Conventional Routing are:

SN	Conventional Routing	React Routing
1.	In Conventional Routing, each view contains a new file.	In React Routing, there is only a single HTML page involved.

2.	The HTTP request is sent to a server to receive the corresponding HTML page.	Only the History attribute <BrowserRouter> is changed.
3.	In this, the user navigates across different pages for each view.	In this, the user is thinking he is navigating across different pages, but its an illusion only.

45. Why do you get "Router may have only one child element" warning?

It is because you do not have to wrap your Router's in a <Switch> block or block which renders a route exclusively.

Example

```
render((
  <Router>
    <Route {/* ... */} />
    <Route {/* ... */} />
  </Router>
)
```

should be

```
render(
  <Router>
    <Switch>
      <Route {/* ... */} />
      <Route {/* ... */} />
    </Switch>
  </Router>
)
```

46. Why switch keyword used in React Router v4?M

The 'switch' keyword is used to display only a single Route to rendered amongst the several defined Routes. The <Switch> component is used to render components only when the path will be

matched. Otherwise, it returns to the not found component.

47. How to use styles in React?

We can use style attribute for styling in React applications, which adds dynamically-computed styles at render time. It accepts a JavaScript object in camelCased properties rather than a CSS string. The style attribute is consistent with accessing the properties on DOM nodes in JavaScript.

Example

```
const divStyle = {
  color: 'blue',
  backgroundImage: 'url(' + imgUrl + ')'
};

function HelloWorldComponent() {
  return <div style={divStyle}>Hello World!</div>
}
```

48. Explain CSS Module styling in React.

CSS Module is a CSS file where all class names and animation names are scoped locally by default. It is available only for the component which imports it, and without your permission, it cannot be applied to any other Components. You can create a CSS Module file with the .module.css extension.

49. What are the three principles that Redux follows?

The three principles that redux follows are:

Single source of truth: The State of your entire application is stored in an object/state tree inside a single Store. The single State tree makes it easier to keep changes over time. It also makes it easier to debug or inspect the application.

The State is read-only: There is only one way to change the State is to emit an action, an object describing what happened. This principle ensures that neither the views nor the network callbacks can write directly to the State.

Changes are made with pure functions: To specify how actions transform the state tree, you need to write reducers (pure functions). Pure functions take the previous State and Action as a parameter and return a new State.

50. What is Babel in React?

Babel is a transpiler.

Babel is an interpreter.

Babel is a compiler.

Babel is both a compiler and a transpiler.

D is the correct option. Babel is both a compiler and a transpiler. It is used to include the ability to compile JSX into regular JavaScript. It is included in development mode and can also do many other powerful things.

51. What do you understand by the Reconciliation process in React? #D,V

The Reconciliation process is a process through which React updates the DOM.

The Reconciliation process is a process through which React deletes the DOM.

The Reconciliation process is a process through which React updates and deletes the component.

It is a process to set the state.

A is the correct option. React uses a "diffing" algorithm that makes the component updates predictable and faster. The React first calculates the differences between the real DOM and the copy of DOM when it finds an update of components. Once it is finished calculating, the new update would be reflected on the real DOM.

52. How is React different from AngularJS?

The following table shows the major difference between AngularJS and React

Factor	React JS	AngularJS
Usage of DOM	**Uses virtual DOM**	**Uses real DOM**
Language	**Uses JavaScript with the extended XML syntax**	**Uses TypeScript which is the superset of JavaScript**
App Structure	**It is represented only using the view of MVC**	**Made of Complete MVC**
Data Binding	**One-way binding**	**Two-way binding**

53. What is the Use of Redux thunk?

Redux thunk acts as middleware which allows an individual to write action creators that return functions instead of actions. This is also used as a delay function in order to delay the dispatch of action if a certain condition is met. The two store methods getState() and dispatch() are provided as parameters to the inner function.

In order to activate Redux thunk, we must first use applyMiddleware() method as shown below:

```
import{ createStore, applyMiddleware } from 'redux';

import thunk from 'redux-thunk';

import rootReducer from './reducers/index';



//Note: this API requires redux@>=3.1.0



const store= createStore(

    rootReducer,

    applyMiddleware(thunk)

);
```

54. What do you know about Flux?D

Basically, Flux is a basic illustration that is helpful in maintaining the unidirectional data stream. It is meant to control construed data's unique fragments to make them interface with that data without creating issues. Flux configuration is insipid; it's not specific to React applications, nor is it required to collect a React application. Flux is basically a straightforward idea, however, you have to exhibit a profound comprehension of its usage.

55. How to embed two components in One component?

```
import React from 'react';

class App extends React.Component{

   render(){

       return(

           <div>

           <Header/>
```

```
            <Content/>
            </div>
        );
    }
}
class Header extends React.Component{
    render(){
        return(
            <div>
            <h1> Header</h1>
            </div>


        )
    }
}
class Content extends React.Component{
    render(){
        return(
            <h2>Content</h2>
            <p>The Content Text!!!</p>
            </div>
        )
    }
}
export default App;
```

56.Give one basic difference between pros and state?

Pros are immutable while the state is mutable. Both of them can update themselves easily.

57. What do you understand with the term polling? M, V

The server needs to be monitored for updates with respect to time. The primary aim in most cases is to check whether novel comments are there or not. This process is basically considered pooling. It checks for updates approximately every 5 seconds. It is possible to change this time period easily. Pooling help keep an eye on the users and always make sure that no negative information is present on the servers. Actually, it can create issues related to several things, and thus pooling is considered.

58. In which lifecycle event do you make AJAX requests and why?

AJAX solicitations ought to go in the componentDidMount lifecycle event.

There are a couple of reasons behind this,

Fiber, the following usage of React's reconciliation algorithm, will be able to begin and quit rendering as required for execution benefits. One of the exchange offs of this is componentWillMount, the other lifecycle event where it may bode well to influence an AJAX to ask for will be "non-deterministic". This means React may begin calling componentWillMount at different circumstances at whenever point it senses that it needs to. This would clearly be a bad formula for AJAX requests.

You can't ensure the AJAX request won't resolve before the component mounts. In the event that it did, that would imply that you'd be attempting to setState on an unmounted component, which won't work, as well as React will holler at you for. Doing AJAX in componentDidMount will ensure that there's a component to update.

59. Why browsers cannot read JSX?

Actually, JSX is not considered proper JavaScript. Browsers cannot read them simply. There is always a need to compile the files that contain JavaScript Code. This is usually done with the help of the JSX compiler which performs its task prior to the file entering the browser. Also, compiling is not possible in every case. It depends on a lot of factors such as the source or nature of the file or data.

60. What exactly can you do if the expression contains more than one line?M, V

In such a situation, enclosing the multi-line JSX expression is an option. If you are a first-time user, it may seem awkward but later you can understand everything very easily. Many times it becomes necessary to avoid multi-lines to perform the task reliably and for getting the results as expected.

61. What is one of the core types in React?V

ReactNode

62. Is it possible to display props on a parent component?

Yes, it is possible. The best way to perform this task is by using the spread operator. It can also be done by listing the properties but this is a complex process.

63. In ReactJS, why is there a need to capitalize on the components?

It is necessary because components are not the DOM element but they are constructors. If they are not capitalized, they can cause various issues and can confuse developers with several elements. At the same time, the problem of integration of some elements and commands can be there.

64. Is it possible to nest JSX elements into other JSX elements?

It is possible. The process is quite similar to that of nesting the HTML elements. However, there are certain things that are different in this. You must be familiar with the source and destination elements to perform this task simply.

65. What are the popular animation packages in the React ecosystem? V

Popular animation package in React ecosystem are

React Motion

React Transition Group

66. What is Jest? B,

Jest is a JavaScript unit testing framework created by Facebook based on Jasmine. It offers automated mock creation and a jsdom environment. It is also used as a testing component.

67. Explain the Presentational segment

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

68. What is the use of a super keyword in React? S,B

The super keyword helps you to access and call functions on an object's parent.

69. Explain yield catchphrase in JavaScript

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

70. Explain the use of 'key' in react list B,

Keys allow you to provide each list element with a stable identity. The keys should be unique.

71. 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.

72. What is the reduction?S, B,

The reduction is an application method of handling State.

73. When should you use the top-class elements for the function element?B

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

74. How can you share an element in the parsing? S, B

Using the State, we can share the data.

75. Explain the term ‘Restructuring.’ B

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

76. State the difference between getIntialState() and constructor()? S

If you want to create one component by extending ‘React. Component’, the constructor helps you to initialize the State. But, if you want to create by using ‘Reat.createClass.’ then you should use ‘genInitiaState.’

77. Name any five predefined prototypes used in React, V

Most important protoype used in React js are:

number

string

array

object

element

78. What is the purpose of using bindActionsCreators?B,

BindActionCreator helps you to bind the event based on the action dispatcher to the HTML element.

79. What do you understand by “Single source of truth”? B,

A: We have mentioned the single source of truth many time earlier in this article but would you know how to explain what it actually means? Well, you should because this is one of the React interview questions that are pretty basic.The single source of truth refers to the store used for storing the app’s entire state at one place. The benefits of the single source of truth include the

facts that all the components stored there receive updates from the store itself, it is easier to keep track of their changes, as well as debug and inspect the application

80. How Virtual-DOM is more efficient than Dirty checking? B,

A: First thing to understand here is that in React, each component has a state which is observable. React knows when to re-render the scene because it is able to observe when this data changes. The observables are significantly faster than the Dirty checking because we don’t have to poll the data at a regular interval and check all of the values in the data structure recursively. By comparison, setting a value on the state will signal to a listener that some state has changed. In a situation like that, React can simply listen for change events on the state and queue up re-rendering.Long story short, the virtual DOM is more efficient than the Dirty checking simply because it prevents all the unnecessary re-renders. Re-rendering only occurs when the state changes.

81. Is setState() async? Why?S, B

A: setState() actions are indeed asynchronous. setState() doesn’t immediately mutate this.state. Instead, it creates a pending state transition. Accessing this.state after calling this method can potentially return the existing value. There is no guarantee of synchronous operation of calls to setState and calls may be batched for performance gains.The reason behind is the way setState alters the state and causes rerendering. Making it synchronous might leave the browser unresponsive. That being said, the setState calls are asynchronous as well as batched for better UI experience and performance. Keep this in mind as this is definitely among the most popular 50 interview questions and answers when it comes to React.\

82.How can you tell React to build in the production mode?V

React can be coded to directly build into production by setting the process.env.NODE_ENV variable to production.

Note: When React is in production, warnings and other development features are not shown.

83. What is the use of the second argument that is passed to setState? Is it optional?

When setState is finished, a callback function is invoked, and the components get re-rendered in React.

Yes, it is an optional argument. Since setState is asynchronous, it takes in another callback function. However, in programming practice, it is always good to use another life cycle method instead of this.

Next up on this top React interview questions and answers blog, you need to take a look at binding.

84. Is there a way to avoid the requirement of binding when using React?V, B

Yes, there are two main ways you can use to avoid the need for binding. They are as follows:

Defining the Event Handler as an Inline Arrow function:

```
class SubmitButton extends React.Component {

constructor(props) {

super(props);

this.state = {

isFormSubmitted: false

};

}

render() {

return (

<button onClick={() => {

this.setState({ isFormSubmitted: true });

}}>Submit</button>

)

}

}
```

Using a function component along with Hooks:

```
const SubmitButton = () => {

const [isFormSubmitted, setIsFormSubmitted] = useState(false);

return (

<button onClick={() => {

setIsFormSubmitted(true);

}}>Submit</button>

)

};
```

Also, the Event Handler can be defined as an Arrow function, which is eventually assigned to a Class Field to obtain similar results.

85.Can you conditionally add attributes to components in React?

Yes, there is a way in which you can add attributes to a React component when certain conditions are met.

React has the ability to omit an attribute if the value passed to it is not true.

Consider the following example:

var condition = true;

var component = (

<div

value="foo"

{ ...(condition && { disabled: true }) } />

86. What are the predefined prop types present in React?M, V, B

There are five main predefined prop types in React. They are as follows:

PropTypes.bool

PropTypes.func

PropTypes.node

PropTypes.number

PropTypes.string

87. What are some functions that can be performed using HOC?M, B

Using HOC, you can -

Render Hijacking

Manipulate and Reuse code

Manipulate Props

88. Why is it not advisable to update state directly, but use the setState call?

The conventional way to update state is to use the setState call. Without using it, the user would still be able to modify the state, but it would not update the DOM to reflect the new state.

89. How would you access the value of an input box?

The value of an input box can be accessed in an onChange event handler using event.target.value. Here's an example:

```
class MyInputBox extends React.Component {
  constructor() {
    super();
    this.state = {
        message: ‘’
    }
    this.onChangeHandler = this.onChangeHandler.bind(this);
  }
  onChangeHandler(event) {
    this.setState({
      message: event.target.value
    });
  }
  render() {
    return (<div>
      <p>{this.state.message}</p>
      <input type="text" onChange={this.onChangeHandler} />
    </div>);
  }
}
```

In the above example, whenever the user types in the box, the onChangeHandler function receives as synthetic event which can be used to access the current value of the input box. Here it is being used to update a state variable ‘message’, which is being rendered in a paragraph tag. As the user types, the text updates.

90. Differentiate between React Redux and React's Context API.

The key differences in the comparison: Context Api vs Redux are as follows:

React Redux	React Context API
In order to use React-Redux in our application, we need to code it separately and then merge it into the main project.	React Context can be used in the application directly.
The number of features provided by React-Redux is comparatively more than React Context.	The number of features provided by React Context is comparatively less than React-Redux.

91. What are the things which we should never do inside a reducer?

The things which we should never do inside a reducer are as follows:

Modify the argument of the reducer

We should assure that we do not perform any side operations such as routing transitions, API calls, etc.

We should not call non-pure functions, for instance Date.now(), Math.random(), etc.

92. What are the workflow features in Redux?

The workflow features in Redux are as follows:

Reset: The state of the store is allowed to be reset.

Revert: Revert or Rollback to the last committed state is allowed.

Sweep: Every disabled action which we have fired unintentionally will be removed.

Commit: The current state is made the initial state.

93. What is the mental model of redux saga?M, V

Redux Saga functions as a separate thread in our programme which is solely responsible for side effects. Redux Saga is a redux middleware. In other words, it means that it can be started, paused, and aborted from the main application using standard Redux actions, has access to the entire Redux application state, and can also dispatch Redux actions.

94. Is keeping all of the component states in the Redux store necessary?

All component states do not need to be kept in the Redux store. We need to keep your application state as little as possible. Thus, we should only do it if it makes a difference to us or if it makes using Dev Tools easier.

95. Are there similarities between Redux and RxJS?M, V, B

Although there are a few overlapping themes, these libraries are highly varied and serve very different goals.

Redux is a state management framework for web applications. It is commonly used as a UI architecture. Consider it a compliment to (half of) Angular.

RxJS is a library for reactive programming. It's most commonly used in JavaScript to complete asynchronous activities. Consider it a substitute for Promises.

Because the Store is reactive, Redux employs the Reactive paradigm a little. The Store observes events from afar and adjusts its behaviour accordingly. RxJS employs the Reactive paradigm, but instead of being an architecture, it provides simple building blocks called Observables to achieve this "observing from afar" pattern.

96. How do you know if the Redux state has changed?

A change in state to an application is applied for a release of action; this ensures that the intent to modify the state is achieved.

Example:

The user activates the application by pressing a button.

In the form of a component, a function is called.

As a result, the relative container now dispatches an action.

This occurs because mapDispatchToProps ties the prop (which was just called in the container) to an action dispatcher (in the container).

After capturing the action, the reducer calls a function, which returns a new state with precise changes.

The container is aware of the state change, and the mapStateToProps function alters a specified prop in the component.

97. What are the features of Redux DevTools?M

Answer

Below are the major features of Redux devTools

1. Lets you inspect every state and action payload

2. Lets you go back in time by “cancelling” actions

3. If you change the reducer code, each “staged” action will be re-evaluated

4. If the reducers throw, you will see during which action this happened, and what the error was

5. With persistState() store enhancer, you can persist debug sessions across page reloads

98. Why should component names start with capital letter?

Ans. If you are rendering your component using JSX, the name of that component has to begin with a capital letter otherwise React will throw an error as unrecognized tag. This convention is because only HTML elements and SVG tags can begin with a lowercase letter.

```
class OneComponent extends Component {

// ...

}
```

You can define component class which name starts with lowercase letter, but when it's imported it should have capital letter. Here lowercase is fine:

```
class myComponent extends Component {

  render() {

    return <div />;

  }
```

}

export default myComponent;

While when imported in another file it should start with capital letter:

import MyComponent from './MyComponent';

99. Can you force a component to re-render without calling setState?

Ans. By default, when your component's state or props change, your component will re-render. If your render() method depends on some other data, you can tell React that the component needs re-rendering by calling forceUpdate().

component.forceUpdate(callback);

It is recommended to avoid all uses of forceUpdate() and only read from this.props and this.state in render().

100. What is a diffing algorithm? D, V,

Ans. React needs to use algorithms to find out how to efficiently update the UI to match the most recent tree. The diffing algorithms is generating the minimum number of operations to transform one tree into another. However, the algorithms have a complexity in the order of O(n3) where n is the number of elements in the tree.

In this case, for displaying 1000 elements would require in the order of one billion comparisons. This is far too expensive. Instead, React implements a heuristic O(n) algorithm based on two assumptions:

Two elements of different types will produce different trees.

The developer can hint at which child elements may be stable across different renders with a key prop.

101. What is React memo xfunction?B

Ans. Class components can be restricted from rendering when their input props are the same using PureComponent or shouldComponentUpdate. Now you can do the same with function components by wrapping them in React.memo.

const MyComponent = React.memo(function MyComponent(props) {

```
  /* only rerenders if props change */


});
```