Top 70 ReactJS Interview Questions and Answers PDF

ReactJS Interview Questions and Answers for Freshers

1. What is ReactJS?

ReactJS is an open-source, component-based front-end JavaScript library essential for the view layer of the application. It is supported by Facebook.

ReactJS utilizes a virtual DOM-based structure to fill in the information in HTML DOM. The virtual DOM works quickly, possessing a way that it only modifies single DOM components rather than refreshing the complete DOM every time.

The React application comprises several components, each answerable for outputting a small, reusable part of HTML. Components can be embedded inside other components to permit complex applications to work out of basic building blocks.

A component can also support an internal state. For instance, a TabList component can save a variable compared to the directly open tab.

2. What are the benefits of ReactJS?

The main advantages of ReactJS are as follows:

Easy to learn

ReactJS is not a core but a JS library that takes together objects and targets a specific thing to complete it effectively. It is the view controller in the MVC structure. Any developer who learns JavaScript can interpret React, master its fundamentals, and develop a fantastic app.

Reusable Components

Reusing components is the primary feature of React JS. Even Facebook has executed React as it supports the reuse of framework components. A developer can begin with numerous components like a checkbox, button, etc.

We develop wrapper components made out of those smaller components. Then, we write higher-level wrapper components. It goes on like that until we have this one root component, and that component is our app.

Virtual DOM

When we are about to build a web application with high customer connection and view updates, similar to the new form-maker on JotForm 4.0, we have to consider the available execution problem.

Updating DOM is generally the bottleneck concerning web execution. React attempts to understand this issue by utilizing virtual DOM and a DOM kept in memory.

Flux and Redux architecture

Facebook developed Flux architecture for its various web applications. It is like React components in its unidirectional flow. This structure has action creators that facilitate making action from method parameters. It also supports a library for these methods.

All these actions are supported together by a focal dispatcher to refresh stores. All views are restored according to the stores. There is likewise Redux, which is an upgraded version of flux architecture.

It has an individual store which is not needed in flux. Redux also allows a feature where middleware can be characterized to capture dispatched actions.

SEO-Friendly

SEO is the pillar of a strong business. A superior ranking is equivalent to more commitment from the person, which results in more income.

In addition, ReactJS generally provides a quicker speed by decreasing the time of page load, which is crucial for SEO.

Developer Tools

The React Developer Tools have been described as Chrome and Firefox dev extensions and permit us to review the React component order in the virtual DOM.

It also allows us to choose specific components and check and alter their current props and state.

Scope for Code Testing

The applications of ReactJS are simple to set. Moreover, it allows developers to test and debug their programs using native tools quickly.

Performance Enhancement

ReactJS improves execution because of virtual DOM. The DOM is a cross-platform programming API that manages with HTML, XML, or XHTML.

Some developers face the issue when updating the DOM, which hinders the application's performance. ReactJS solved this problem by representing virtual DOM.

The React Virtual DOM exists in memory and is a definition of the internet browser's DOM. When we compose a React element, we do not correspond precisely to the DOM.

Instead, we are comprising virtual elements that React will pass into the DOM, enabling smoother and quicker performance.

3. What is Create React App?

This is one of the frequently asked React interview questions for freshers. Find the answer below:

Create react app is a React app standard generator developed by Facebook. It supports a development setting configured for usability with minimal setup.

Here are some of the most important things to know about "create react app":

- It involves ES6 and JSX transpilation
- Consists of a Dev server with hot module reloading
- Used for Code linting
- Can contain CSS auto-prefixing
- It can develop the script with JS, CSS, picture packaging, and sourcemaps
- Used for the Jest testing system.

4. What are the features of ReactJS?

There are several features of ReactJS as follows:

Virtual DOM

In React, for each DOM object, there is a corresponding "virtual DOM object." A virtual DOM object is a definition of a DOM object and makes a virtual replica of the initial DOM.

It is a one-way data-binding; therefore, controlling the virtual DOM is faster than updating the initial DOM since nothing gets drawn onscreen.

JSX

JSX represents JavaScript XML. It is a JavaScript syntax extension. It is an XML or HTML like syntax utilized by ReactJS.

This syntax is handled in JavaScript calls of React framework. It broadens the ES6 so HTML-like content can exist together with JavaScript react code. It is not necessary to utilize JSX, but it is endorsed to use it in ReactJS.

Simplicity

ReactJS uses the JSX file, which makes the application simple to program and understand.

We learn that ReactJS is a component-based technique that establishes the code reusable as our need. This creates it simple to use and learn.

One-way Data Binding

One-way data binding defines that data flows only in one direction through the entire application. This provides better control over it. The data is moved from the parent component to the child component through read-only props.

These props cannot be sent back to the parent component. This is how one-way data binding works. Although, the child component can interact with the parent component to update the state via callback functions.

React Native

React Native is a custom performed for React, directly comparable to React DOM on the web. It uses native components rather than web components like React as building blocks.

It can start with React Native, and we must know the essential React concepts, like JSX, components, state, and props. It also provides an approach to these platform features, from changing the React program to working on iOS and Android.

5. What are the disadvantages of ReactJS?

The disadvantages of ReactJS are as follows:

Dynamic Technology

One of the disadvantages of ReactJS is that it keeps on changing with time. Therefore, it is a constant learning technique for the developers, who have to understand new methods of doing things that come with uncertain circumstances.

Average Documentation

It is another disadvantage of constantly updating technologies. There is a lack of suitable documentation because of the fast updating of React technologies.

JSX as a barrier

ReactJS utilizes JSX, which is a syntax extension that enables the combining of HTML and JavaScript. This method is helpful, but some development community members treat JSX as a barrier, especially the new developers. In addition, developers argue about the learning curve complexity of JSX.

View Part

Only UI layers of the app get covered by ReactJS. Therefore, choosing a few other technologies is necessary to get a complete collection of tooling for the project's development.

6. How to install ReactJS?

ReactJS is a JavaScript library included in a single file react-<version>.js that can be contained in any HTML page. The developers also generally install the React DOM library react-dom-<version>.js along with the main React file:

Example

```
<!DOCTYPE html>
<html>
 <head></head>
 <body>
 <script type="text/javascript" src="/path/to/react.js"></script>
 <script type="text/javascript" src="/path/to/react-dom.js"></script>
 <script type="text/javascript">
 // Use react JavaScript program here or in an individual file
 </script>
 </body>
</html>
```

To obtain the JavaScript files, go to the https://reactjs.org/docs/getting-started.html of the official React documentation.

React also provides JSX syntax. JSX is an extension generated by Facebook that inserts XML syntax into JavaScript. It can use JSX we require to contain the Babel library and change *<script type="text/javascript">* to *<script type="text/babel">* for translating JSX to Javascript program.

```
<!DOCTYPE html>
<html>
 <head></head>
 <body>
 <script type="text/javascript" src="/path/to/react.js"></script>
 <script type="text/javascript" src="/path/to/react-dom.js"></script>
 <script src="https://npmcdn.com/babel-core@5.8.38/browser.min.js"></script>
 <script type="text/babel">
 // Use react JSX program here or in an individual file
 </script>
 </body>
</html>
```

Installing ReactJS via npm

We can also install React using npm by doing the following:

npm install --save react react-dom

To use React in our JavaScript project, we can do the following:

```
var React = require('react');
var ReactDOM = require('react-dom');
ReactDOM.render(<App/>, ...);
```

Installing React via Yarn

Facebook released its package manager, Yarn, which can also be used to install React. After installing Yarn, we are just required to run this command:

yarn add react react-dom

We can then use React in our project precisely as if we had installed React via npm.

7. What is the difference between ReactJS and AngularJS?

This is among the most common ReactJS interview questions. We have curated a simple tabular comparison to help you understand the differences between ReactJS vs AngularJS.

	ReactJS	AngularJS
Definition	ReactJS is an open-source JavaScript library. It is used to develop a user interface for a single-page application. It is answerable only for the view layer of the application.	AngularJS is an open-source JavaScript framework that can develop a powerful web application. It is an MVC framework that works with the MVC platform, where it facilitates development by giving a dependable solution.
Developed By	Facebook	Google
Data Binding	ReactJS supports one-way binding. It provides singular behavior for your application. One-way data-binding defines some changes we create to the model that influence the view, but not the other way around. In these methods, the data only flows in one direction.	AngularJS uses a two-way data binding, which links the Document Object Model (DOM) values to model data. It defines if a new value is supported in the app for user interaction with the field. It will appear in the update of both the view and the model.
Simplicity	React is not simple, and it takes some time to start a project.	Angular is easy and simple to understand. However, its inherent complexity sometimes confuses.
DOM Usage	React uses a virtual DOM. A virtual DOM is a secure version of the DOM. We can modify any element rapidly without needing to render the whole DOM.	Angular uses the browser's DOM.

Application Structure	React is not an MVC-like Framework. It is like a view-based library. React does not demand its client or developer to use some particular application structure.	Angular is a complete-featured MVC Framework. Angular MVC-based structure always permits the application to split into three associated parts so that they can be simply manipulated.

8. What is the difference between ReactJS and React Native?

Here is the ReactJS and React Native comparison to understand the differences between the two:

	React JS	**React Native**
Definition	ReactJS is a JavaScript library responsible for developing a hierarchy of user interface components. It is essential for the rendering of user interface components. It offers support for both the front-end and server-side.	React Native is an open-source JavaScript framework. It facilitates a mobile application for iOS Android, and Windows. It uses only JavaScript to develop a cross-platform mobile app. React Native is similar to React but uses native components instead of web components as building blocks.
Setup and Running Process	React.js is a JavaScript library that we can use for web development. When beginning a new project and setting up ReactJS, we must choose Webpack, a bundler. We will then determine which wrapping modules will particularly suit our project.	In React Native, we have everything we require to establish and get started. It is so simple and quick to release the framework. All it takes is to run one command in our terminal, and we are ready to begin programming our first-ever React Native app. We will require

		Xcode (for iOS) or Android Studio (for Android) installed on our system to run the app. It uses an emulator of the focus platform or tests its implementation on our tool.
Animations	In Reactjs, animation is possible using CSS, just like typcial web development.	In React Native, an animated API is used for persuasive animation across multiple elements of the React Native application.
DOM and Styling	Virtual DOM is used to deliver browser code in Reactjs.	In React Native, native APIs are used to deliver elements on mobile.
Capabilities	React JS was created to keep Search Engine Optimization (SEO) in mind, in which node is utilized for rendering on the client-server. Though numerous equivalent tools support this view of the server to rendering, they are accessible to insecure hacks. This is separate from an extensive amount of developer support needed for maintenance.	React Native is exclusively dedicated to developing mobile UI. It defines that separated from being centered on UI, Reacts Native also serves as a JavaScript library, not a framework. Hence the UI built is hugely responsive and adds a smooth feel to our mobile application and quicker load times.

9. Which browsers does ReactJS support?

It supports all the popular and most used web browsers like Google Chrome, Firefox, Mozilla, Microsoft Edge, etc. It doesn't support browsers built without ES5 methods, like Internet Explorer.

10. What is JSX in React?

JSX is a preprocessor phase that inserts XML syntax into JavaScript. We can use React without JSX, yet JSX makes React a lot more classic.

It is just a similar XML. JSX tags contain a tag name, attributes, and children. If an attribute value is placed within quotes, the value is a string. Alternatively, wrap the value in braces, and the value is the confined JavaScript expression.

JSX gives syntactic sugar to React.createElement(component, props, ...children) function.

11. What are the benefits of using JSX in React?

The advantages of JSX are as follows:

- It is always quick as it implements optimization while assembling a program to vanilla JavaScript.
- JSX is additionally type-safe, which defines it as carefully composed, and most of the errors can be found during the compilation of the JSX code to JavaScript.
- It consistently makes writing templates simpler and quicker if we are simple with HTML syntax.

12. What is the difference between DOM and Virtual DOM in ReactJS?

Another common question asked in the ReactJS job interviews is about the comparison between DOM and VDOM. Here is the answer:

DOM

DOM represents the Document Object Model. It is also known as HTML DOM, an abstraction of a structured code known as HTML for web developers. DOM and HTML code are associated, as the elements of HTML are called nodes of DOM.

It describes a structure where users can create, alter, modify documents, and present the content. Therefore while HTML is text, DOM is an in-memory definition of this content.

Virtual DOM (VDOM)

VDOM in ReactJS is a method that syncs the virtual user interface with the real document object model. It is not a dedicated technology but a pattern that helps in handling the events, updating manual DOM, as well as manipulating the attributes.

13. What is the use of keys in ReactJS?

Keys recognize unique virtual DOM elements with their related data driving the UI. It supports React to develop rendering by recycling existing DOM components. Keys must be unique numbers or strings. Rather than re-rendering with keys, React only re-orders the components. It enhances the application's execution.

14. What is Relay in ReactJS?

Relay is a JavaScript framework supporting an information layer and user-server connection to web applications utilizing the React view layer.

15. What is the difference between createElement and cloneElement?

JSX components are transpiled to React.createElement() functions to make React elements that will be used for the object definition of UI.

Whereas, cloneElement is used to clone an element and pass it new props.

16. How to create components in React?

It is one of the frequently asked React interview questions for freshers. You must know that there are two approaches to creating a component in ReactJS:

Functional Components

It is the simplest method of creating the React component. These pure JavaScript functions obtain the props object as the first argument and return the react component.

```
function Hello({ message }) {
return <h1>{`Hello, ${message}`}</h1>
}
```

Class components

We can use ES6 class to create a component. The function component can be written as:

```
class App extends React.Component {

render() {

return <h2>{`Hii, ${this.props.message}`}</h2>

  }

}
```

17. What is a React State?

States are the soul of React elements. These are the source of information and should be maintained as simply as feasible.

ReactJS states are the objects which decide the component's rendering and behavior. They are mutable, unlike the props, and generate powerful and interactive elements. They are accessed through this.state().

18. What are React Props?

Props are inputs to components. They are read-only components that should be kept immutable. ReactJS Props are used to pass information and methods from a parent component to a child component.

19. What are Default Props?

In ReactJS, defaultProps authorize us to set default, or fallback, values for our component props. defaultProps are helpful when we call components from various views with fixed props, but in some views, we are required to pass several values.

Example of React Default Props

```
class MyApp extends React.Component {...}

MyApp.defaultProps = {

 randomObject: {},

 ...

}
```

20. What is the difference between state and props in ReactJS?

Here is the tabular comparison between React state and props:

State	Props
The state is mutable.	Props are immutable.
The state influence data about the components.	Props enable us to pass data from one component to another as an argument.
They do not allow to generate reusable components.	They allow to make reusable components.
The State is internal and managed by the component itself.	Props are external and managed by whatever renders the component.
The child component cannot access states.	The child component can access props.
It can be used for rendering dynamic changes with the component.	It is used to connect components.

21. What is setState() in ReactJS?

The primary method using which we create UI updates to our React applications is through a call to the setState() function. This function will execute a shallow merge between the new state that we give and the previous state and trigger a re-render of our component and all decedents.

22. What are the Parameters of setState()?

There are two parameters of setstate() as follows:

1. **Updater:** It can be an object with several key-value pairs that should be combined into the state or a function that restores such an object.
2. **Call-back (optional):** The function implemented after setState() is implemented effectively. Because calls to setState() are not ensured by React to be atomic, this can be beneficial if we need to implement some action after we are positive that setState() has been performed effectively.

24. What is the arrow function in React?

An arrow function is also known as the **'fat arrow function (=>).'** It enables binding the context of components properly because auto-binding is not possible by default in ES6. In addition, it creates it easier to work with higher-order components.

Example

1. **Without using Arrow functions**

```
render() {
 return (<MyInput onChange={this.handleChange.bind(this) } />)
 }
```

2. **Using Arrow Functions**

```
render() {
return (<MyInput onChange={(e)=>this.handleOnChange(e) } />)
}
```

25. What is the context in ReactJS?

The context supports a method to pass information through the component tree without giving props down manually at each level.

For example, authenticated customers, locale preference, and UI theme should be acquired in the application by many components.

```
const {Provider, User} = React.createContext(defaultValue)
```

26. What are React Mixins?

Mixins are an approach to separate elements from having standard functionality. It should not be used and can be replaced with higher-order components or decorators.

27. What is the context.consumer in React?

A consumer is a React element that subscribes to text changes. It needed a function as a child that accepts the current context value as a parameter and returns a react node. The value parameter passed to the function will be similar to the nearest provider's value props for this context above in the tree.

Example

```
<MyContext.Consumer>
 {value => /* render something depend on the context value */}
```

</MyContext.Consumer>

28. What is React Fiber?

Fiber is the new reconciliation engine or re-executing the basic algorithm in React v16. The objective of React Fiber is to expand its appropriateness for fields like animation, design, gestures, and the ability to pause, abort, or reuse work and assign priority to multiple types of updates; and new concurrency primitives.

29. What is React reconciliation?

When a component's props or state changes, React determines whether an actual DOM update is essential by contrasting the recently returned element with the previously rendered one.

When they are not similar, they will update the DOM. This process is known as reconciliation in ReactJS.

30. What are Pure Components?

Pure components are the easiest and quickest components that can be written. They can restore any component which has a render(). These components upgrade the program's simplicity and the application's execution.

Advanced ReactJS Interview Questions for Experienced

If you have worked in this field for some years, then our ReactJS interview questions for senior developer or experienced professionals (1-5 years) are as follows:

1. Why use JSX in React?

There are a number of important reasons behind using JSX:

- It is faster than regular JavaScript because it implements optimization while translating the program to JavaScript.
- JSX can separate innovations by placing markup and logic in separate documents, React uses components that include both.
- It is type-safe, and some bugs can be found at compilation time.
- It makes it simpler to develop templates.

2. Why a web directory can’t read JSX?

Web directory cannot read JSX directly because they can only read JavaScript objects, and JSX is not an ordinary JavaScript object. Accordingly, we must change the JSX

document into a JavaScript object using transpilers like Babel and then pass it to the browser.

3. What is DOM diffing?

When the elements are rendered twice, Virtual DOM starts checking the changes elements have got. They define the changed component on the page. Several other components don't go through changes. It can decrease the DOM changes due to client activities, and treat DOM doffing. It is generally done to boost the execution of the browser. This is the reason for its capability to execute all the functions rapidly.

4. What is the role of of render() in React?

Every React component should have a render() necessarily. It returns a single React component, which describes the native DOM component. If more than one HTML element should be rendered, then they should be arranged inside one enclosing tag including <form>, <group>, , etc.

This function should remain pure, i.e., it must return a similar outcome each time it is invoked.

5. What is the difference between elements and components in ReactJS?

Here is the tabular comparison of React elements and components:

Elements	Components
An element is a plain JavaScript object representing the component state, DOM node, and desired properties.	A component is the core building structure of the React application. It is a class or function which takes an input and returns a React element.
It only holds data about the component type, its properties, and any child elements inside it.	It can contain state and props and access the React lifecycle methods.
It is immutable.	It is mutable.
Example: *const element = React.createElement(* *'div',* *{id: 'login-btn'},*	**Example:** *function Button ({ onLogin }) {* *return React.createElement(* *'div',* *{id: 'login-btn', onClick: onLogin},*

'Login' *)*	*'Login'* *)* *}*

6. What are Stateless Functional Components?

In numerous applications, smart components occupy state but render dumb components that receive props and return HTML as JSX. Stateless functional components are substantially more reusable and have a positive execution impact on our application.

They have two main characteristics:

- When rendered, they get an object with all the props passed down.
- They should restore the JSX to be rendered.

7. What are Stateful Components in ReactJS?

If the component's behavior is dependent on the state of the component, it tends to be defined as a stateful component. These stateful components are continually class components and have a state that gets initialized in the constructor.

8. What is the difference between Stateful and Stateless Components?

We have compared the stateful and stateless components below:

Stateful Component	Stateless Component
It stores info about a component's state change in memory.	It evaluates the internal state of the components.
It includes information on past, current, and likely future changes in state.	It contains no information on history, present, or potential future state changes.
It is also called a class component.	It is also called a functional component.
It can work with all the lifecycle methods of React.	It cannot work with any lifecycle method of React.

9. When to use the class component over a function component?

If the component requires state or lifecycle methods, then use class components; otherwise, use function components.

From React 16.8, with the inclusion of hooks, we can use state, lifecycle methods, and other characteristics that were only feasible in a class component right in our function component.

10. What is the differences between setState() and replaceState() methods?

When we use setState(), the current and previous states are joined. replaceState() throws out the current state and restores it with only what we provide.

setState() is used except if we require to delete all previous keys for some reason. We can also set state to false/null in setState() rather than using replaceState().

11. What is Prop Drilling?

When developing a React application, a deeply nested element often needs to use data provided by another element that is much higher in the hierarchy. The simplest method is passing a prop from each component to the next in the authority from the source element to the deeply nested element. This is called prop drilling.

12. What is the difference between componentWillMount() and componentDidMount()?

This is one of the most common React interview questions for senior developer. You must know the differences between the two, as mentioned below:

componentWillMount()	componentDidMount()
This is called only once through the component lifecycle before the element is rendered to the server. It is used to determine props and do any excess logic based on them.	It is called only on the user end. It is generally performed after the initial render when a user has received data from the server.
It is used to execute state changes before the initial render.	It enables us to implement all kinds of advanced interactions, including state changes.

It is invoked just directly after the mounting has occurred.	Putting the data loading code in this only ensures that data is fetched from the client's end.
Users should avoid using async initialization to minimize side effects or subscriptions in this method.	The data does not store until the initial render is done. Hence, users must set an initial state properly to eradicate undefined state errors.

13. How to bind approaches or event handlers in JSX callbacks?

There are three possible approaches to manage this:

Binding in Constructor

The methods are not constrained by default in JavaScript classes. A similar thing relates to react event handlers represented as class methods. It can generally bind them in the constructor.

```
class Component extends React.Component {
  constructor(props) {
    super(props)
    this.handleClick = this.handleClick.bind(this)
  }

  handleClick() {
    // ...
  }
}
```

Public class fields syntax

If we don't like to use the bind method, then public class fields syntax can be used to bind callbacks accurately.

```
handleClick = () => {
```

```
  console.log('this is:', this)
}
<button onClick={this.handleClick}>
  {'Click me'}
</button>
```

Arrow functions in callbacks

We can use arrow functions precisely in the callbacks.

```
<button onClick={(event) => this.handleClick(event)}>
  {'Click me'}
</button>
```

14. How are Error Boundaries handled in React V15?

React version 15 supports the error boundaries by utilizing the unstable_handleError method. It is termed to componentDidCatch in React Version 16.

15. What is the aim of getDerivedStateFromProps() lifecycle method?

A new static getDerivedStateFromProps() lifecycle method is invoked after an element is instantiated as well as before it is re-rendered. It can return an object to refresh the state or nullify that the new props do not need state updates.

```
class MyApp extends React.Component {
  static getDerivedStateFromProps(props, state) {
    // ...
  }
}
```

16. What is render prop in React?

Render prop is a simple method for sharing programs between components using a prop whose value is a function. The following component uses render prop, which returns a React element.

<DataProvider render={data => (

<h1>{`Hello ${data.target}`}</h1>

)}/>

17. What is the lifecycle method of React Component?

Lifecycle methods are used to run programs and interact with our components at various points in the component's life. These methods are based on a component, Mounting, Updating, and Unmounting.

Component Creation

When a React component is created, several functions are called:

- If we are using React.createClass (ES5), 5 user-defined functions are called.
- If we are using class Component extends React.Component (ES6), 3 user-defined functions are called.

getDefaultProps() (ES5 only)

This is the first method called.

Prop values restored by this function will be used as defaults if they are not described when the component is instantiated.

getInitialState() (ES5 only)

This is the second method called.

The return value of getInitialState() describe the initial state of the React component. The React core will call this function and select the restore value to this.state.

Example

this.state.count will be initialized with the value of this.props.initialCount:

getInitialState() {

```
return {
count : this.props.initialCount
        };
}
```

componentWillMount() (ES5 and ES6)

This is the third method called.

This function can make final changes to the component before being added to the DOM.

```
componentWillMount() {
...
}
```

render() (ES5 and ES6)

This is the fourth method called.

The render() function must be a pure function of the component's state and props. It returns a single element that defines the component during the rendering process and should either be a description of a native DOM component (e.g., <p/>) or a composite component. If nothing should be performed, it can return invalid or undefined.

This function will be recalled after the component's props or state changes.

```
render() {
return (
<div>
Hello, {this.props.name}!
</div>
    );
}
```

componentDidMount() (ES5 and ES6)

This is the fifth method called.

The component has been mounted, and we can create the component's DOM nodes, e.g., via refs.

This method should be used for:

- Preparing timers
- Fetching information
- Adding event listeners
- Manipulating DOM component

```
componentDidMount() {
...
}
```

ES6 Syntax

If the component is represented using ES6 class syntax, the functions getDefaultProps() and getInitialState() cannot be used.

Rather than we declare our defaultProps as a static property on the class and declare the state shape and original state in the constructor of our class. These are both set on the class instance at construction time before any other React life cycle function is called.

The following example shows this alternative approach:

```
class MyReactClass extends React.Component {
        constructor(props){
        super(props);
        this.state = {
                count: this.props.initialCount
                    };
                }
        upCount() {
                 this.setState((prevState) => ({
                count: prevState.count + 1
                }));
```

```
        }
    render() {
     return (
     <div>
          Hello, {this.props.name}!<br />
          You clicked the button {this.state.count} times.<br />
          <button onClick={this.upCount}>Click here!</button>
     </div>
     );
     }
    }
MyReactClass.defaultProps = {
 name: 'Bob',
 initialCount: 0
};
```

Replacing getDefaultProps()

Default values for the component props are stated by framework the defaultProps property of the class:

```
MyReactClass.defaultProps = {
 name: 'Bob',
 initialCount: 0
};
```

Replacing getInitialState()

The idiomatic way to set up the component's initial state is to set this. state in the constructor:

```
constructor(props){
 super(props);
```

```
this.state = {
count: this.props.initialCount
  };
}
```

Component Removal

This method is known before a component is unmounted from the DOM. This phase includes only one method and is given below.

componentWillUnmount()
This method is invoked directly before a component is destroyed and unmounted permanently. It implements any necessary **cleanup** related function, including invalidating timers, and event listeners, canceling network requests, or cleaning up the DOM component. If a component instance is unmounted, we cannot mount it again.

Example

```
import React, { Component } from 'react';
export default class SideMenu extends Component {
constructor(props) {
        super(props);
        this.state = {
        ...
        };
        this.openMenu = this.openMenu.bind(this);
        this.closeMenu = this.closeMenu.bind(this);
        }
componentDidMount() {
document.addEventListener("click", this.closeMenu);
}
componentWillUnmount() {
```

```
document.removeEventListener("click", this.closeMenu);
}
openMenu() {
...
}
closeMenu() {
...
}
render() {
        return (
                <div>
                 <a
                href = "javascript:void(0)"
                className = "closebtn"
               onClick = {this.closeMenu}
                >
                 ×
                </a>
                <div>
                Some other structure
                </div>
                </div>
                );
        }
}
```

Component Update

It is the next stage of the lifecycle of a react component. This stage allows managing user interaction and support interaction with the components hierarchy. In addition, this stage aims to ensure that the component is showing the latest version of itself.

componentWillReceiveProps(nextProps)

This is the function called on properties changes. When the component's properties change, it will call this function with the new properties. We can access the old props with this.props and the new props with nextProps.

With these variables, we can do some comparison operations between old and new props or call functions because of a property change, etc.

shouldComponentUpdate(nextProps, nextState)

This is the second function known as properties changes and the first on state changes.

By default, if another component / our component changes a property/state of our component, it will render a new version. Therefore, in this method, this function always returns true.

We can override this function and select more accurately if our component must update or not.

This function is generally used for optimization.

In the method of the function returns false, the update pipeline stops directly.

```
componentShouldUpdate(nextProps, nextState){
 return this.props.name !== nextProps.name ||
 this.state.count !== nextState.count;
}
```

componentWillUpdate(nextProps, nextState)

This function works like componentWillMount().Here, we can't modify the component state by invoking **this.setState()** method. It will not be known, if **shouldComponentUpdate()** returns false.

render()
It is invoked to determine **this.props** and **this.state** and return one of the following

types: React elements, Arrays and fragments, Booleans or null, String and Number. If shouldComponentUpdate() returns false, the code within render() will be invoked again to assure that the component shows itself correctly.

componentDidUpdate(prevProps, prevState)

It is invoked directly after the component updating appears. In this method, we can put some code inside this, which we need to implement once the updating occurs. This method is not invoked for the basic render.

18. What is PropTypes library in React?

PropTypes is the type-checking addition to React Library, which exports a range of validators to ensure the data component received is valid. As a result, it reduces several bugs and makes components self-documented.

19. How can we renew the State of a component?

We can renew the State of a component using this.setState() method. This method does not continually replace the State immediately. Instead, it only include changes to the initial State. It is the main method to update the user interface in response to event handlers and server responses.

20. What is NextJS?

Next.js is a famous and lightweight framework for static and server-rendered applications that work with React. It also gives styling and routing solutions.

21. What are the features of Next.js?

The main features of Next.js are as follows:

- It is used to server-rendered by default.
- It contains programmed code-parting for quicker page loads.
- It is used to clean user-side routing (page-based).
- It can have a Webpack-based dev setting that provides (HMR).
- It can be easy to perform with Express or another Node.js HTTP server.
- It is used customizable with our own Babel and Webpack configurations.

22. What are Error Boundaries?

One of the frequently asked ReactJS interview questions for experienced professionals is about error boundaries. You must know its meaning and role, as explained below.

Error boundaries are the React components that catch the JavaScript errors somewhere in their child component tree and log those errors. As a result, they will show a fallback user interface rather than the component tree that crashed.

A class would be an error boundary if it defines both the lifecycle methods static getDerivedStateFromError() or componentDidCatch(). The need for static getDerivedStateFromError() is for rendering a fallback UI after a bug has been thrown. The need for componentDidCatch() is for logging the error data.

- Dedicated Doubt Sessions
- Industry-recognized Certification
- Career Support