Topic 7 ReactJS

Interview Questions and Answers

Q1. How are web applications different than websites?

	Web Applications	Websites
Type of Software	Part of the website as it's not a complete website itself	Complete product, and can be accessed with the help of a browser
Deployment	Its functions are quite higher and more complex compared to a website	The website displays the collected data and information on a specific page
Compilation	Must be precompiled before deployment	Doesn't need any pre-compilation
Authentication	Needs authentication, as they offer a much broader scope of options than websites	No Authentication is needed for informational websites. Some information might be asked to get a regular update or to access additional options

Q2. What is React?

React is an open-source JavaScript library that is used to create user interfaces, especially for single-page applications (SPA). It can handle the view layer for web and mobile apps easily. It was created by a software engineer working for Facebook named Jordan Walke. Initially, it was deployed on Facebook's News Feed in 2011 and on Instagram in 2012.

Q3. What are the major features of React?

There are multiple features of ReactJS but some of them are:

1. It uses VDOM instead of Real DOM because manipulation of Real DOM is expensive
2. It supports server-side rendering that enables pages to load faster and improves user experience
3. It makes debugging much easier as it follows a Unidirectional data flow
4. It uses reusable/composable UI components to develop the view
5. It is SEO friendly

Q4. What is JSX?

JSX is an XML-like syntax extension to ECMAScript. Generally, it is used to provide the syntactic sugar for React.createElement() function as it provides the expressiveness of JavaScript along with HTML-like template syntax.

In the below example text inside <h1> tag is returned as JavaScript function to the render function.

```
class App extends React.Component {
render() {
   return(
                  <div>
                 <h1>{'Welcome to Super Campus!'}</h1>
      </div>
                    )
      }
 }
```

Q5. What is the difference between React Element and Component?

An Element is a plain object that represents what should be displayed on the screen in terms of the DOM nodes or other components. Elements can hold other Elements in their props. Even though creating an element is cheap but once an element is created, it is never mutated. The object representation of React Element would be as follows:

```
const element = React.createElement(
       'div',
       {id: 'btn-login'},
       'Login'
)
```

The above React.createElement() function returns an object:

```
{
       type: 'div',
       props: {
              children: 'Login',
       id: 'btn-login'
       }
}
```

And finally, it renders to the DOM using `ReactDOM.render()`:

```
<div id='btn-login'> Login </div>
```

Whereas a component can be declared in several different ways. It can be a class with a render() method, or it can be defined as a function. In either case, it takes props as an input, and returns a JSX tree as the output:

```
const Button = ({ onLogin }) =>
     <div id={'btn-login'} onClick={onLogin}>
         Login
     </div>
```

Then JSX gets transpiled to a React.createElement() function tree:

```
const Button = ({ onLogin }) => React.createElement(
     'div',
      { id: 'btn-login', onClick: onLogin },
      'Login'
```

```
    )
```

Q6. How to create components in React?

React provides two ways to create a component. They are:

Function Components: This is considered as simplest and easiest way to create a component. They are pure JavaScript functions that accept props object as the first parameter and return React elements:

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

Class Components: In this case, the ES6 class is used to define a component. The above function component can be written as:

```
class Greeting extends React.Component {
           render() {
           return
          <h1>{`Hello, ${this.props.message}`}</h1>
            }
      }
```

Q7. What do you mean by Pure Components in React?

React pure components are similar to react components, but they can handle shouldComponentUpdate() method automatically. Whenever, props or state changes, pure components will do a shallow comparison on both props and state. On the other hand, the basic component won't compare current props and state to the next out of the box. Thus, the component will re-render by default whenever the shouldComponentUpdate() method is called.

Q8. How is the state different than props?

Generally, both props and state are plain JavaScript objects because both of them hold information that influences the output of render, but they are still different in their functionality with respect to the component. Props get passed to the component similar to function parameters while the state is managed within the component similar to variables declared within a function.

Q9. Why should we never update the state directly?

Whenever the state is updated directly it won't re-render the component.

An example to do the same is given below:

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

It is important to use the setState() method as it schedules an update to a component's state object. Whenever the state changes, the component responds by re-rendering. An example to do the same is given below:

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

Note: state object can directly be assigned either in the constructor or using the latest javascript's class field declaration syntax.

Q10. What is the difference between HTML and React event handling?

There are various differences between HTML and React event handling but some of them are:

1. In HTML, the event name is usually represented in lowercase as a convention:

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

   Whereas in React it follows camelCase convention:

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

2. 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.')
   }
   ```

3. 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)

Q11. What are synthetic events in React?

Synthetic Event is a cross-browser wrapper around the browser's native event.

Its API is similar to the browser's native event, including stopPropagation() and preventDefault() but these events work identically across all browsers.

Q12. What are inline conditional expressions?

We already know if statements or ternary expressions in JavaScript are available to conditionally render expressions. Besides, it is possible to embed any expressions in JSX by wrapping them in curly braces and then followed by JS logical operator &&. The Example representing the same is given below:

```
<h1>Hello!</h1>
{
    messages.length > 0 && !isLogin?
      <h2>
          You have {messages.length} unread messages.
      </h2>
      :
      <h2>
          You don't have unread messages.
      </h2>
}
```