

REACT JS **INTERVIEW** QUESTIONS

PART - 1

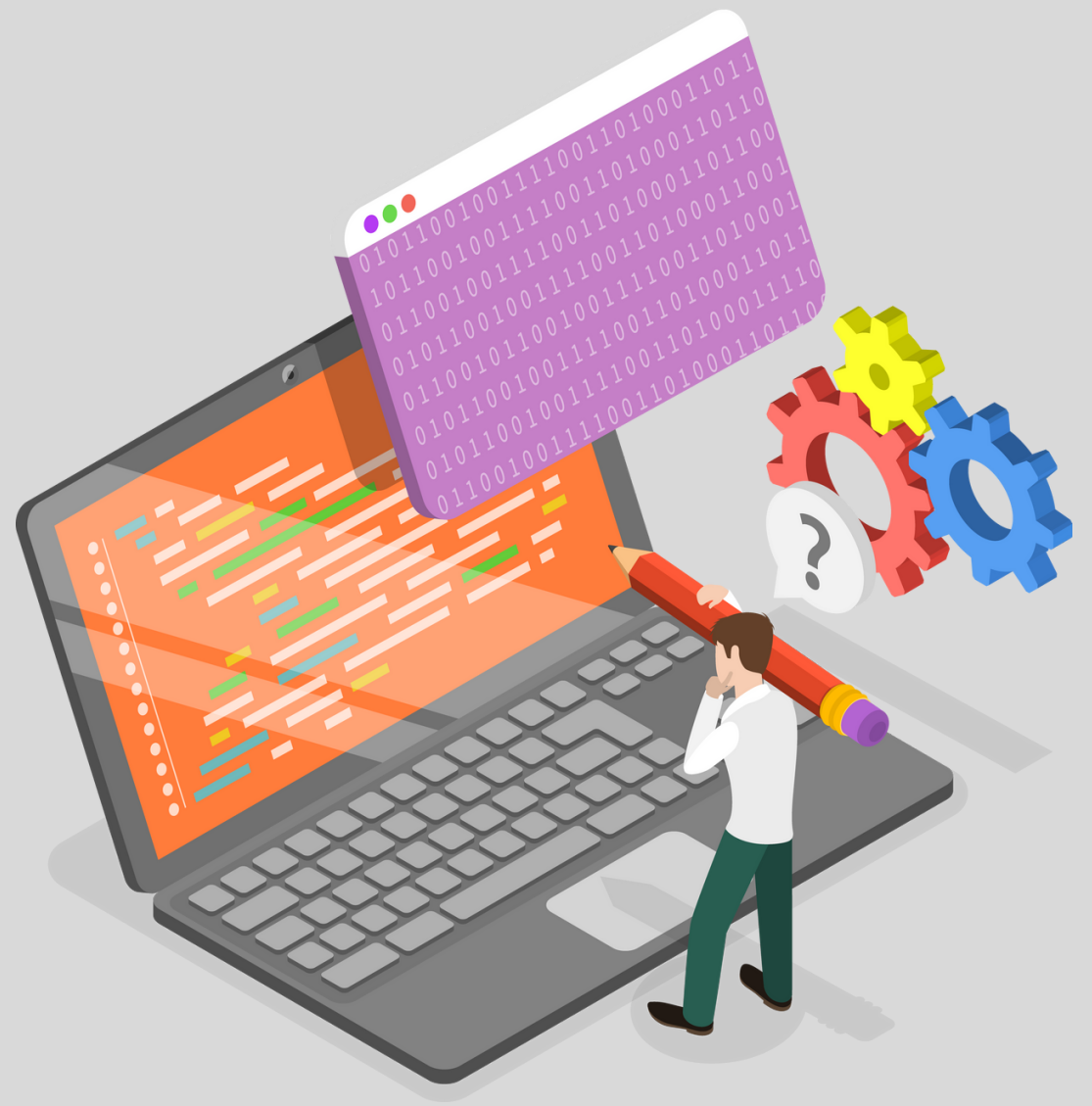


codewithgauri



Q- What is React?

React is an open-source front-end JavaScript library that is used for building user interfaces, especially for single-page applications. It is used for handling the view layer for web and mobile apps. React was created by Jordan Walke, a software engineer working for Facebook. React was first deployed on Facebook's News Feed in 2011 and on Instagram in 2012.



The important features of React are:

- It supports server-side rendering.
- It will make use of the virtual DOM rather than real DOM (Data Object Model) as RealDOM manipulations are expensive.
- It follows unidirectional data binding or data flow.
- It uses reusable or composable UI components for developing the view.



Q- What are the advantages of using React?

- Use of Virtual DOM to improve efficiency: React uses virtual DOM to render the view. As the name suggests, virtual DOM is a virtual representation of the real DOM. Each time the data changes in a react app, a new virtual DOM gets created. Creating a virtual DOM is much faster than rendering the UI inside the browser. Therefore, with the use of virtual DOM, the efficiency of the app improves.
- Gentle learning curve: React has a gentle learning curve when compared to frameworks like Angular. Anyone with little knowledge of javascript can start building web applications using React.
- SEO friendly: React allows developers to develop engaging user interfaces that can be easily navigated in various search engines. It also allows server-side rendering, which boosts the SEO of an app.
- Reusable components: React uses component-based architecture for developing applications. Components are independent and reusable bits of code. These components can be shared across various applications having similar functionality. The re-use of components increases the pace of development.
- Huge ecosystem of libraries to choose from: React provides you with the freedom to choose the tools, libraries, and architecture for developing an application based on your requirement.



Q- What are props in React?

Props are inputs to components. They are single values or objects containing a set of values that are passed to components on creation using a naming convention similar to HTML-tag attributes. They are data passed down from a parent component to a child component.

The primary purpose of props in React is to provide following component functionality:

1. Pass custom data to your component.
2. Trigger state changes.
3. Use via `this.props.reactProp` inside component's `render()` method.

For example, let us create an element with `reactProp` property:

```
<Element reactProp={'1'} />
```

This `reactProp` (or whatever you came up with) name then becomes a property attached to React's native `props` object which originally already exists on all components created using React library.

```
props.reactProp
```



Q- What are the limitations of React?

The few limitations of React are given below:

- React is not a full-blown framework as it is only a library.
- The components of React are numerous and will take time to fully grasp the benefits of all.
- It might be difficult for beginner programmers to understand React.
- Coding might become complex as it will make use of inline templating and JSX.

Q- What are synthetic events in React?


SyntheticEvent is a cross-browser wrapper around the browser's native event. Its API is the same as the browser's native event, including `stopPropagation()` and `preventDefault()`, except the events work identically across all browsers.



Q- What is useState() in React?

The useState() is a built-in React Hook that allows you for having state variables in functional components. It should be used when the DOM has something that is dynamically manipulating/controlling.

In the below-given example code, The useState(0) will return a tuple where the count is the first parameter that represents the counter's current state and the second parameter setCounter method will allow us to update the state of the counter.




```
...  
const [count, setCounter] = useState(0);  
const [otherStuffs, setOtherStuffs] =  
  useState(...);  
const setCount = () => {  
  setCounter(count + 1);  
  setOtherStuffs(...);  
  ...  
};
```



Q- What are keys in React?

A key is a special string attribute that needs to be included when using lists of elements.



```
const ids = [1,2,3,4,5];
const listElements = ids.map((id)=>
{
  return(
    <li key={id.toString()}>
      {id}
    </li>
  )
})
```

Importance of keys -

- Keys help react identify which elements were added, changed or removed.
- Keys should be given to array elements for providing a unique identity for each element.
- Without keys, React does not understand the order or uniqueness of each element.
- With keys, React has an idea of which particular element was deleted, edited, and added.
- Keys are generally used for displaying a list of data coming from an API.

***Note- Keys used within arrays should be unique among siblings. They need not be globally unique.



Q- What is JSX?

JSX stands for JavaScript XML. It allows us to write HTML inside JavaScript and place it in the DOM without using functions like `appendChild()` or `createElement()`.

As stated in the official docs of React, JSX provides syntactic sugar for the `React.createElement()` function.

Note- We can create react applications without using JSX as well.

Let's understand how JSX works:

Without using JSX, we would have to create an element by the following process:

```
const text = React.createElement('p', {}, 'This is a text');
const container = React.createElement('div', {}, text);
ReactDOM.render(container, rootElement);
```

Using JSX, the above code can be simplified:

```
const container = (
  <div>
    <p>This is a text</p>
  </div>
);
ReactDOM.render(container, rootElement);
```



Q- What are the differences between controlled and uncontrolled components?

- **Controlled component:** In a controlled component, the value of the input element is controlled by React. We store the state of the input element inside the code, and by using event-based callbacks, any changes made to the input element will be reflected in the code as well.

When a user enters data inside the input element of a controlled component, onChange function gets triggered and inside the code, we check whether the value entered is valid or invalid. If the value is valid, we change the state and re-render the input element with the new value.

```
function FormValidation(props) {
  let [inputValue, setInputValue] = useState("");
  let updateInput = e => {
    setInputValue(e.target.value);
  };
  return (
    <div>
      <form>
        <input type="text" value={inputValue} onChange={updateInput} />
      </form>
    </div>
  );
}
```



- **Uncontrolled component:** In an uncontrolled component, the value of the input element is handled by the DOM itself. Input elements inside uncontrolled components work just like normal HTML input form elements.

The state of the input element is handled by the DOM. Whenever the value of the input element is changed, event-based callbacks are not called. Basically, react does not perform any action when there are changes made to the input element.

Whenever user enters data inside the input field, the updated data is shown directly. To access the value of the input element, we can use ref.

```
function FormValidation(props) {  
  let inputValue = React.createRef();  
  let handleSubmit = e => {  
    alert(`Input value:  
    ${inputValue.current.value}`);  
  };  
  return (  
    <div>  
      <form onSubmit={handleSubmit}>  
        <input type="text" ref={inputValue} />  
        <button type="submit">Submit</button>  
      </form>  
    </div>  
  );  
}
```



And for amazing stuff you can follow me



Gaurav Pandey

LinkedIn :Gaurav Pandey

Twitter : @gauravcode

References:

<https://www.artstation.com/artwork/ayQvq>

<https://github.com/sudheerj/reactjs-interview-questions>

<https://www.interviewbit.com/react-interview-questions/#what-is-react>