React Interview Questions and Answers

Question 01

Explain what is the Virtual DOM in React

The virtual DOM (VDOM) is a programming concept where an ideal, or “virtual”, representation of a UI is kept in memory and synced with the “real” DOM by a library such as ReactDOM

Question 02

What are the differences between the Virtual DOM and the DOM?

A virtual DOM object is the same as a real DOM object, except that it is a lightweight copy. This means that it cannot manipulate on-screen elements. The main purpose is to perform fast calculations in memory on the DOM elements.

React Interview Questions and Answers

Question 03

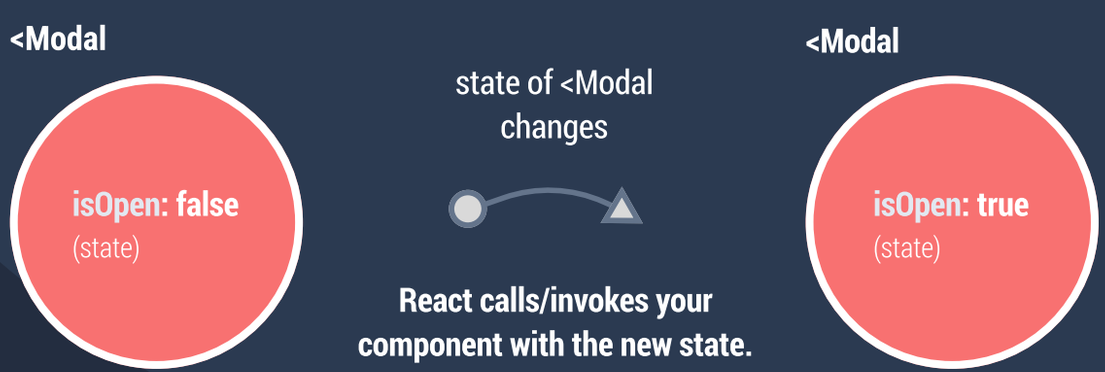
What is JSX?

JSX is a shorthand for JavaScript XML. This is a type of file used by React which utilizes the expressiveness of JavaScript along with HTML like template syntax. This makes the HTML file really easy to understand.

Question 04

What do you understand when we talk about Render in React?

Render is just a fancy word for saying that React is calling one of your components.



(visual from the infographics bundle)

React Interview Questions and Answers

Question 05

Explain the concept of Components in React

Components are the building blocks of a React application's UI. These components split up the entire UI into small independent and reusable pieces.

Question 06

What are props in React and what are the main differences with state?

Props is the shorthand for Properties in React. They are read-only variables which must be kept pure i.e. immutable. This help in maintaining the unidirectional data flow and are generally used to render the dynamically generated data.

The main difference with state variables is that props are immutable, read only variables.

React Interview Questions and Answers

Question 07

How can you update the state of a component?

In functional components you can make use of the useState hook to define and update a state variable. It gives back an array that holds the state variable in the first position and the setter or update function in the second position. You can make use of this function to update the state variable like this: setSomeState('some value').

Question 08

What are synthethic events in React?

Synthetic events are the objects which act as a cross-browser wrapper around the browser's native event. They combine the behavior of different browsers into one API. This is done to make sure that the events show consistent properties across different browsers.

React Interview Questions and Answers

Question 09

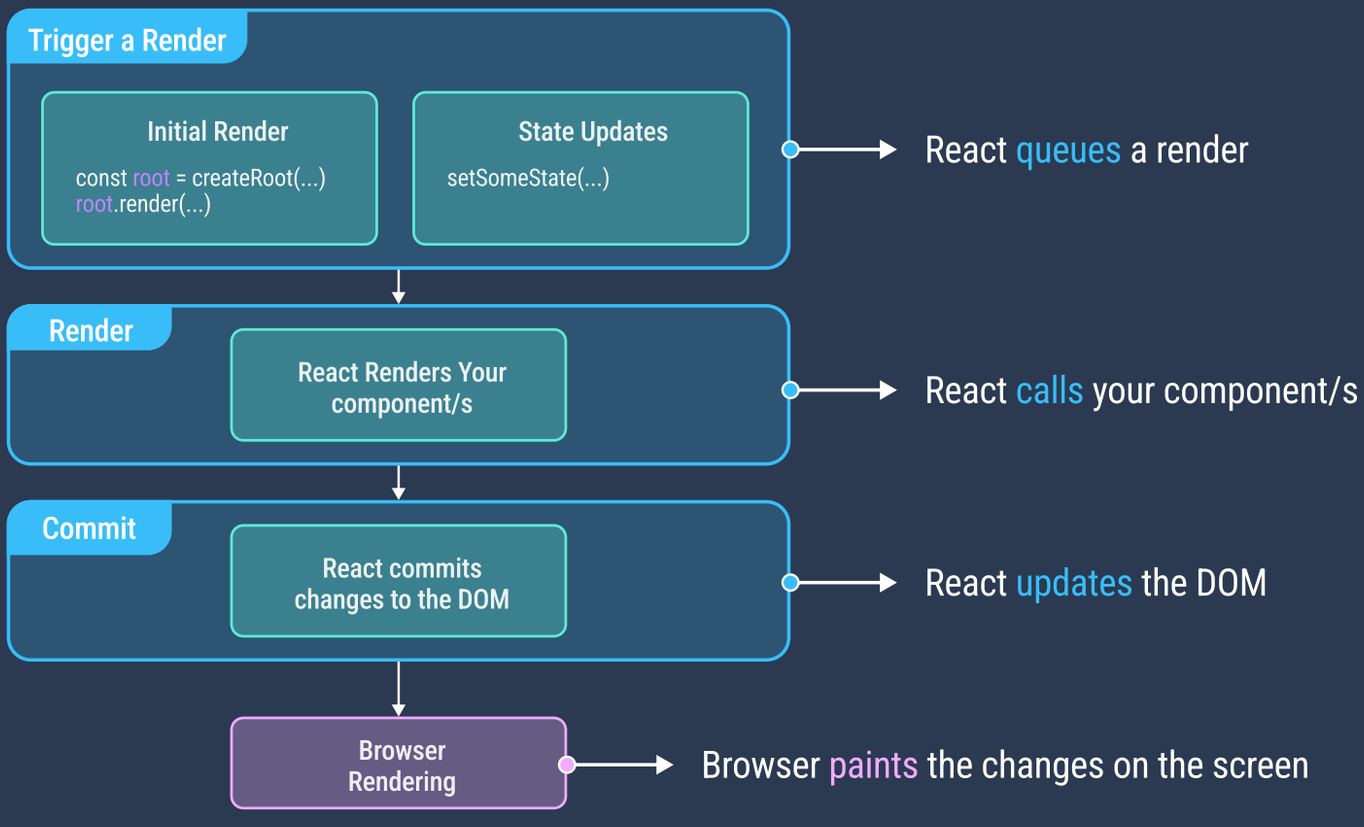
What are refs in React, give an example of when would you use one.

Refs is the short hand for References in React. It is a variable that holds a reference to a value that’s not needed for rendering. You can define a reference using the useRef hook, which returns a reference variable that you can access by calling the ref.current key. A use case for using ref is when you need to perform dom calculations on specific elements.

React Interview Questions and Answers

Question 10

How does React render process work?



(visual from the infographics bundle)

React Interview Questions and Answers

Question 11

How can you reuse logic in functional components?

You can make use of customHooks, which are functions that hold data that you want to reuse through multiple components, e.g. useActiveUser() can be a custom 4 hook that returns all the information from the custom user

Question 12

What are keys in React and why are they important?

A key is a string or a number that uniquely identifies a React element. Keys are important (specially in arrays) because they tell React which array item each component corresponds to, so that it can match them up later.

This becomes important if your array items can move (e.g. due to sorting), get inserted, or get deleted. A well-chosen key helps React infer what exactly has happened, and make the correct updates to the DOM tree.

React Interview Questions and Answers

Question 13

What is the difference between a controlled and uncontrolled component in React?

Controlled components refer to components that have their state and behavior controlled by the parent component. These components rely on props passed down from the parent component to update their state and behavior. Uncontrolled components refer to components that manage their own state internally.

Question 14

Explain the main difference between server side rendering and client side rendering in React

Server-side rendering (SSR) is when the initial render of a React application is done on the server. The server generates the HTML for the initial state of the application and sends it to the browser. When the JavaScript bundle loads, React takes over and the application continues to function as a SPA (Single-Page Application) on the client side.

React Interview Questions and Answers

Question 15

How would you avoid unnecessary renders in a React component?

It depends on each use case but one technique could be to wrap the props the component receive with useCallback or useMemo , and also use Rect.memo on the component itself, to only re-render when xwz props change.

Another technique would be to review useEffects on the component itself to see if there are unnecessary render being triggered from within.

Question 16

What is a Portal in React?

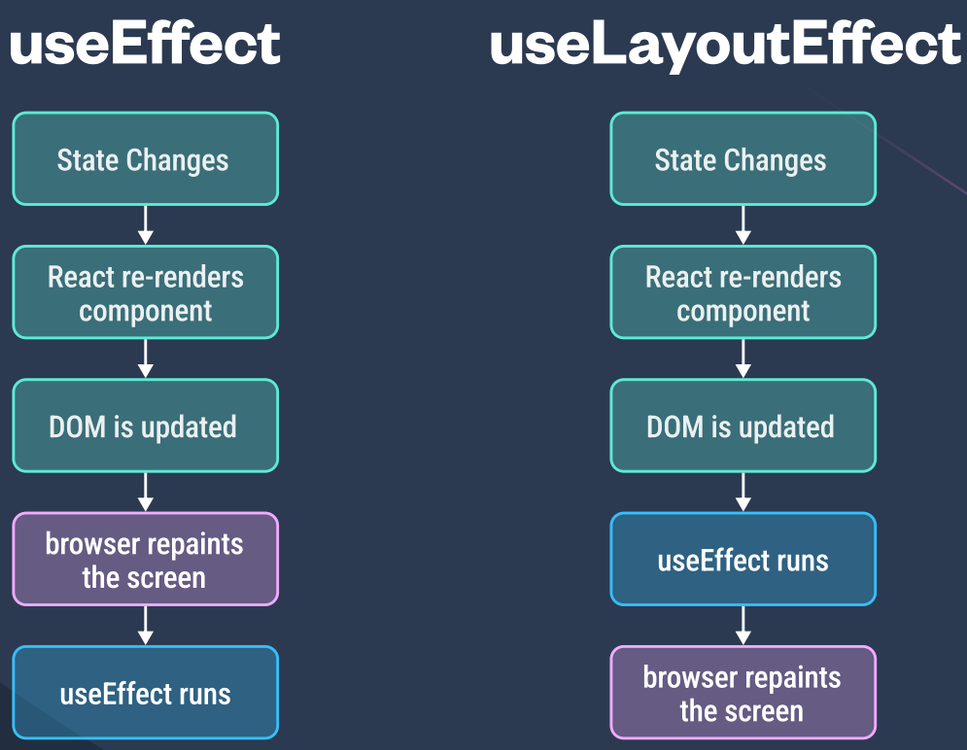
React has a built in function called React.createPortal, that creates a bridge between the component where it's called and some DOM element, the second parameter accepts the node element we want to render.

Question 17

State the main difference between useEffect and useLayoutEffect

It depends on each use case but one technique could be to wrap the props the component receive with useCallback or useMemo , and also use Rect.memo on the component itself, to only re-render when xwz props change.

Another technique would be to review useEffects on the component itself to see if there are unnecessary render being triggered from within.



(visual from the infographics bundle)

React Interview Questions and Answers

Question 18

Where do you usually fetch data in React?

If you don’t use any third party library like react-query (which you should), you can use useEffect with caution to fetch data in React.

Question 19

What are hooks in React?

Hooks is React’s way to manage state and side effects in functional components. Hooks are named functions that start with the word use and allow us to reuse stateful logic across components without having to write a class component

React Interview Questions and Answers

Question 20

If you wanted a component to perform an action only once when the component initially rendered how would you achieve it with a function component?

We can achieve this with useEffect and an empty dependency array

```
const Homepage = () => {
  useEffect(() => {
    trackPageView('Homepage');
  }, []);
  return <div>Homepage</div>;
};
```

React Interview Questions and Answers

Question 21

What is prop-drilling, and how you can avoid it?

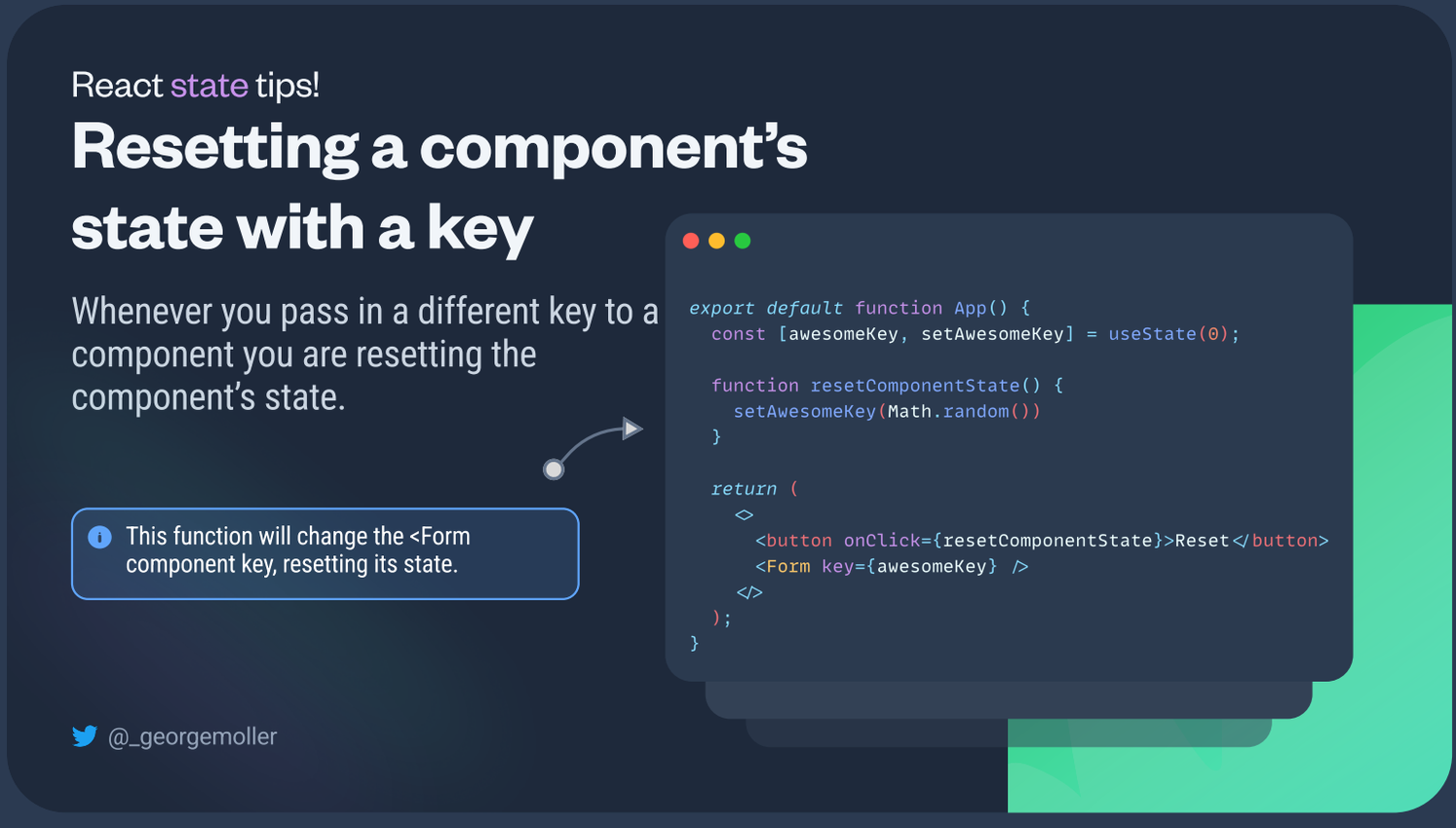
Prop drilling is the unofficial term for passing data through several nested children components, in a bid to deliver this data to a deeply-nested component.

There are several ways you can avoid this, one could be using a centralized store for the state management of your up, like Redux. Another solution could be by making use of React's composition model.

React Interview Questions and Answers

Question 22

How can you reset a component's state to it's initial state?



(visual from the infographics bundle)

React Interview Questions and Answers

Question 23

Explain the difference between useState and useEffect.

useState is a react built in hook that lets you add a state variable to your component.

useEffect is a React Hook that lets you sync a component with an external system.

React Interview Questions and Answers

Question 24

If you have to set a key value in a component, from where would you get that value?

Different sources of data provide different sources of keys:

Data from a database: If your data is coming from a database, you can use the database keys/IDs, which are unique by nature.

Locally generated data: If your data is generated and persisted locally (e.g. notes in a note-taking app), use an incrementing counter, crypto.randomUUID() or a package like uuid when creating items.

Question 25

Explain the concept of lazy loading React components

React.lazy lets you defer loading component's code until it is rendered for the first time.

Call lazy outside your components to declare a lazy-loaded React component:

```
import { lazy } from 'react';
const MarkdownPreview = lazy(() => import('./MarkdownPreview.js'));
```

React Interview Questions and Answers

Question 26

Explain the main differences between useMemo and useCallback

Difference between useMemo() and useCallback() in React?

useCallback()

useCallback will remember a **function** between renders.

Used to fix performance issues, e.g. when **changing** a function can cause expensive components to re-render.

```
const SomeComponent = () => {
  // Whenever SomeComponent re-renders, onItemClick fn reference
  // stays the same, and ExpensiveComponent won't re-render
  const onItemClick = React.useCallback(() => {
    /* do something */
  }, [])
  <ExpensiveComponent onItemClick={onItemClick} />
};

const ExpensiveComponent = React.memo(({onItemClick}) => {
  /*...*/
})
```

useMemo()

useMemo will remember a **value** between renders.

Used to avoid re-calculating expensive operations.

```
// expensive function that takes some time to calculate
import searchUsersByOccupation from 'utils/users'

const UsersList = ({ users, occupation }) => {
  // 💡 Hey React, remember this value between renders
  // Only re-calculate when user's list or occupation changes
  const usersByOccupation = React.useMemo(() => {
    return searchUsersByOccupation(users, occupation)
  }, [users, occupation]);
};
```

(visual from the infographics bundle)

Question 27

Why would you ever use forwardRef in React?

forwardRef lets your component expose a DOM node to parent component with a ref.

Call forwardRef() to let your component receive a ref and forward it to a child component:

```
import { forwardRef } from 'react';
const MyInput = forwardRef(function MyInput(props, ref) {
// ...
});
```

React Interview Questions and Answers

Question 28

At what point does the useEffect cleanup function run?

The cleanup function runs not only during unmount, but before every re-render with changed dependencies

Question 29

Explain the concept of Reducers and Context in React

Reducers let you consolidate a component's state update logic. Context lets you pass information deep down to other components. You can combine reducers and context together to manage state of a complex screen.

React Interview Questions and Answers

Question 30

How would you catch render errors in React components at the global scope?

I would use an error boundary, which is a special component that lets you display some fallback UI instead of the part that crashed—for example, an error message.

```
class ErrorBoundary extends React.Component {
  constructor(props) {
    super(props);
    this.state = { hasError: false };
  }
  static getDerivedStateFromError(error) {
    // Update state so the next render will show the fallback UI.
    return { hasError: true };
  }
  componentDidCatch(error, info) {
    logErrorToMyService(error, info.componentStack);
  }
  render() {
    if (this.state.hasError) {
      // You can render any custom fallback UI
      return this.props.fallback;
    }
    return this.props.children;
  }
}
```

Question 31

Explain what is wrong with the following code

```
export function App() {
  const [firstName, setFirstName] = React.useState('');
  const [lastName, setLastName] = React.useState('');
  const [fullName, setFullName] = React.useState('');

  React.useEffect(() => {
    setFullName(`${firstName} ${lastName}`);
  }, [firstName, lastName]);

  return <p>Full name is: {fullName}</p>;
}
```

The problem here is we are not using useEffect correctly. useEffect should only be used for synchronization between props/state and external systems.

In this case we are using it derived a new state (fullName) from current/previous state (firstName, lastName) The solution is to derived the fullName state in the render phase like this:

React Interview Questions and Answers

```
export function App() {
  const [firstName, setFirstName] = React.useState('');
  const [lastName, setLastName] = React.useState('');
  const fullName = `${firstName} ${lastName}`;

  return <p>Full name is: {fullName}</p>;
}
```

Question 32

How does useEffect check changes in its array dependency?

useEffect Will compare each dependency with its previous value using the Object.is comparison. Object and Arrays are compared by reference, and primitive variables are compared by value