Your grade: 100%

Your latest: **100%** • Your highest: **100%** • To pass you need at least 70%. We keep your highest score.

Next item →

1. What hook can you use to manage a form's state in React? **1 / 1 point**

- [✓ useState]
- useEffect
- useForm
- useContext

Correct
Correct! In React, you can use the useState to handle a form's state.

2. Which hook should you use to handle asynchronous logic in functional components? **1 / 1 point**

- useState
- useAsync
- useCallback
- [✓ useEffect]

Correct
Correct! While the useEffect hook itself is not specifically designed for handling asynchronous code, you can use it to create side effects functionalities such as data fetching, which often involves asynchronous operations.

3. How can you trigger an effect to run only once after the initial render in a functional component? **1 / 1 point**

- With the useEffect hook and no dependencies
- With the useState hook to track the component's initial render
- [✓ With the useEffect hook and an empty dependency array]
- With the useEffect hook and a non-empty dependency array

Correct
Correct! By providing an empty dependency array to the useEffect hook, you ensure that the effect runs only once after the initial render of the component.

4. How can you handle form submission in React? **1 / 1 point**

- With the onClick event
- With the onKeyDown event handler
- With the onChange event handler
- [✓ With the onSubmit event]

Correct
Correct! You can use the onSubmit event to handle an event whenever the user submits the form.

5. How can you handle user input in a controlled component in React? **1 / 1 point**

- By relying on the default behavior of form elements
- [✓ With the onChange event handler to update the state]
- With controlled data
- By directly modifying the DOM elements

Correct
Correct! onChange event handler is used to capture changes to form elements. This event handler is used to update the component's state, ensuring that the input value is controlled by React state.

6. What is an advantage of using Redux Thunk middleware? **1 / 1 point**

- [✓ Thunk enables async operations without boilerplate code.]
- Thunk scales well.
- Thunk works well with complex applications.
- Thunk handles concurrency problems efficiently.

Correct
Correct! Redux Thunk middleware enables asynchronous operations without requiring excessive boilerplate code.

7. What is Redux Toolkit primarily used for? **1 / 1 point**

- Creating complex UI components
- [✓ Reducing boilerplate code]
- Implementing authentication and authorization
- Managing HTTP requests and responses

Correct
Correct! The Redux Toolkit provides utilities to streamline common Redux tasks, minimizing the amount of boilerplate code.

8. Which function from the Redux Toolkit consolidates Redux setup logic into a single function call? **1 / 1 point**

- createSlice()
- createReducer()
- [✓ configureStore()]
- createStore()

Correct
Correct! The configureStore() function consolidates Redux setup logic, such as setting up middleware and enabling the Redux DevTools Extension into a single function call.

9. What is the purpose of a Redux slice? **1 / 1 point**

- [✓ To divide the application states into several parts and manage their updates]
- To create reusable UI components
- To manage HTTP requests and responses
- To handle asynchronous operations

Correct
Correct! A Redux slice divides the application states into individual parts and manages the logic to update them. It typically consists of a reducer function, action creators, and an initial state.

10. Which middleware is commonly used with Redux to handle asynchronous actions and side effects? **1 / 1 point**

- Redux Saga
- Redux Logger
- Redux DevTools Extension
- [✓ Redux Thunk]

Correct
Correct! Thunk is middleware commonly used with Redux to handle asynchronous actions and side effects. It allows action creators to return functions, enabling asynchronous behavior in Redux applications.