

**React and Redux Architecture and Best Practices**

**Course Number:** RCT-130
**Duration:** 2 days

**Overview**

This React and Redux Architecture and Best Practices training course teaches developers advanced skills for architecting React applications, implementing best practices, and boosting performance.

**Prerequisites**

Students should have taken Accelebrate's [Introduction to React course](file:////training/react) or have equivalent experience. Attendees must have experience with Modern JavaScript concepts, including arrow functions, modules, and destructuring.

**Materials**

All React and Redux training attendees receive comprehensive courseware.

**Software Needed on Each Student PC**

* Google Chrome and/or Firefox
* Other modern browsers as desired
* IDE/development environment of your choice
* Other free software and lab files as specified, including Node.js

**Objectives**

* Explore the common component architecture patterns
* Learn to write custom hooks to reuse stateful logic in components
* Explore when and how to use Context in a React application
* Understand state management, including use cases and various alternatives
* Utilize Redux to manage the state of the application
* Use React and Redux together
* Learn React and Redux best practices
* Optimize React performance and avoid wasted renders

**Outline**

* Introduction
* Component Architecture
	+ Reuse
	+ Component Communication
	+ Design Patterns
		- Container and Presentation Components
		- Composition vs. Inheritance
* Hooks
	+ Background
	+ Hooks APIs: useState, useEffect, useRef, useContext, useReducer
	+ Rules of Hooks
* Custom Hooks
	+ Background
	+ Definition
	+ Reuse of stateful logic
* Context
	+ When to use Context
	+ useContext Hook
* State Management
	+ What is State?
	+ When do you need a State Management Library?
	+ Types of State
	+ Using Context for Shared State
	+ Server State Libraries: React Query, SWR, or GraphQL client
* Redux
	+ What is Redux?
	+ Benefits Checklist
	+ Principles of Redux
	+ Core Concepts (Store, State, Reducers, Actions, Action Creators)
	+ Complementary Packages
	+ Basic Redux Example (includes time traveling)
	+ Gotchas/Tips
* Using Redux with React (React Redux Library)
	+ Redux with React in Function Components
	+ useSelector and useDispatch Hooks
	+ Provider
	+ Example
	+ Redux with React in Class Components
	+ Higher-Order Components
	+ The connect function
	+ Writing mapState functions
	+ Writing mapDispatch Functions
	+ Example
* Asynchronous Actions (Redux Thunk)
	+ Overview
	+ Async Actions (Thunks)
	+ Installation
	+ Your First Thunk
	+ Full CRUD Example
* Putting It All Together (React & Redux & Thunk)
* Performance
	+ What causes a component to render?
	+ Wasted Renders
	+ Memoization
	+ React.memo
	+ Pure Components
* Render Props
	+ Understanding the render props pattern
* Conclusion