

**Introduction to Redux for React using JavaScript/ECMAScript**

**Course Number:** RCT-122
**Duration:** 2 days

**Overview**

[Redux](https://redux.js.org/) is a predictable state container for React. As React applications become more complex, developers often find a need for more structure around the data or state they are managing in their application. The Redux JavaScript library provides an elegant solution for this need.

This Introduction to Redux for React using JavaScript/ECMAScript training teaches developers the skills they need to immediately use Redux in their React applications.

**Note:** This Redux class can also be taught using TypeScript.

**Prerequisites**

Attendees must have prior React development experience.

**Materials**

All 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 that Accelebrate would specify, including Node.js

**Objectives**

* Understand state management including when it is needed and the various alternatives
* Utilize Redux to manage the state of the application
* Use React and Redux together
* Implement React and Redux best practices
* Write unit tests for React using Jest and React Testing Library

**Outline**

* Introduction
* State Management
	+ What is State?
	+ When do you need a State Management library?
	+ Alternatives to Redux
	+ Using Context for Shared State
	+ Server State: 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
	+ When do you need Redux?
	+ 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)
* Testing
	+ Tools (Jest, React Testing Library or Enzyme)
	+ Syntax
	+ Testing Vanilla JavaScript with Jest
	+ Mocking
		- Mocking Modules
		- Mocking Functions
	+ Debugging Tests
	+ Component Tests with React Testing Library
	+ Component Tests with Enzyme
		- Shallow
		- Full
		- Snapshot
	+ Testing Redux Actions & Thunks, Reducers, HTTP
* Optional Topics/FAQ (if time permits)
	+ Performance
		- What causes a component to render?
		- Wasted Renders
		- Memoization
		- React.memo
		- Pure Components
* Conclusion