

**Building Web Applications with React and MobX**

**Course Number:** RCT-108
**Duration:** 5 days

**Overview**

React is a popular, high-performance JavaScript library for building fast, composable user interfaces. The MobX library is a variant form of the Flux pattern, providing state management for React applications using functional reactive programming.

Accelebrate’s Building Web Applications with React and MobX training class teaches experienced JavaScript developers the skills they need to immediately implement React/MobX in their applications.

**Prerequisites**

All React training attendees must be experienced JavaScript developers with a fairly advanced understanding of JavaScript, including prototypes and functions as first class citizens.  If your group doesn’t yet have this experience, we could readily add one or two days to the beginning of your course to appropriately prepare them.

**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 (Visual Studio Code or WebStorm recommended in most cases)
* Other free software and lab files that Accelebrate would specify, including Node.js

**Objectives**

* Understand what React and MobX are and what problems they solve
* Explore the basic architecture of a React component
* Gain deeper knowledge of React.js components and JSX
* Utilize React Hooks
* Learn React.js best practices and common patterns
* Employ React Routing to build larger apps
* Employ the principles of MobX to build easier to understand more maintainable applications
* Correctly incorporate MobX into React component trees using React’s Context API
* Integrate Server-Side Data Sources into a React/MobX application
* Explore how to integrate the requested React Library and frameworks into a React application

**Outline**

* Overview
	+ What is React?
	+ What problem does React solve?
	+ Development Ecosystem
	+ React versus other frameworks (Angular & Vue)
	+ Declarative vs. Imperative Programming
	+ Immutable Programming
* Development Tools
	+ Create React App project generator
	+ TS - Create React App TypeScript Template
	+ React Developer Tools
	+ Running and Debugging a React Application
	+ TS - Debugging TypeScript Code within the Browser
	+ Role of Node.js
	+ Purpose of React and ReactDOM
* React Elements & JSX
	+ Create Element and JSX
	+ Benefits of JSX
	+ TS - JSX and TSX Files
	+ Common Errors with JSX
	+ Fragments
	+ JSX and Expressions
* Functional Components
	+ What are Components?
	+ Displaying Collections of Data
	+ Dynamic Component Siblings and React Keys
	+ Passing Data with Props
	+ Validating Props with PropTypes
	+ TS - Strongly-Typed Props
	+ TS - Using Type vs. Interface
	+ Default Props
	+ Managing State with the State Hook
	+ Managing Form and List State with a Custom Hook
	+ TS - Generics and Hooks
	+ TS - Complex Generics and Custom Hooks
* Components and Styling
	+ CSS Files
	+ CSS Modules
	+ CSS-in-JS (StyledComponents)
	+ Style Prop
	+ Class Name Prop
* Component Composition
	+ Coding with a Focus on Reusability and Testing
	+ Decompose a Component into Smaller Components
	+ Data Props & Function Props
	+ Minimizing Component Dependencies
	+ Maximize Component Decoupling
	+ Balancing Prop Drilling and External Dependencies
	+ Lifting State Up
* Class-Based Components (optional)
	+ JavaScript Classes and Extends
	+ Configuring State
	+ Lifecycle Methods
	+ Comparison of Lifecycle Methods and Hooks
	+ Context of Event Handlers
	+ Class Properties and Class Arrow Functions
	+ PropTypes and Default Props on Classes
	+ TS - Strongly-Typed Class Properties
	+ TS - Strongly-Typed Props and Default Props on Classes
	+ Higher Order Components
	+ Component Inheritance Anti-Pattern
	+ Error Boundaries
* Other Hooks
	+ Overview of Hooks
	+ Three Motivations
	+ Compare Hooks and Higher Order Components
	+ Effect Hook
	+ Ref Hook
	+ Callback Hook
	+ Memo Hook
	+ Context Hook
* Advanced React Topics (cover the ones which are interesting to the class)
	+ Context
	+ Error Boundaries
	+ Modals
	+ Lazy Loading
	+ Concurrent Mode
* React Router
	+ What is routing?
	+ Understanding the URL as state
	+ Benefits of routing
	+ Route Element and Matching URLs
	+ Single and Multiple Matches
	+ Nesting Routes
	+ Passing Data via the URL Path
	+ Passing Data via the URL Query String
	+ Passing Data via JavaScript
	+ TS - Strongly-Typed React Router Hooks
* Managing State with MobX Overview
	+ Functional Reactive Programming
	+ State
	+ Derivations
	+ Actions
	+ Principles
* Observable State
	+ What is an observable?
	+ Create an Observable
	+ Observable Decorator
	+ Observable Objects
	+ Observable Arrays
	+ Observable Maps
* Pitfalls of Observable State
	+ Property Access and Change Detection
	+ Getter/Setter and Proxy Objects
	+ No Destructuring
	+ Retrieving Data from Arrays
* Reacting to Observables
	+ What does it mean to react?
	+ Computed Values
	+ Autorun
	+ When
	+ Reaction
* Store
	+ What is a store?
	+ Creating Stores
	+ Multiple Stores
	+ Replacing React Component State with a MobX store
	+ TS - Decorators
* Updating Observables
	+ Actions
	+ Bound Actions
* Asynchronous Actions
	+ Using Promises with Actions
	+ Enforce Actions and Run In Action
	+ Understanding Async/Await and Actions
	+ Flows and Generators
* MobX and React Components
	+ Higher Order Component
	+ Observer Component
	+ Observer Hook
* MobX and React Hooks
	+ Observer Hook
	+ Local Store Hook
* React & MobX Unit Testing
	+ JavaScript Unit Testing
	+ Jest and Enzyme
	+ React Testing Tools
	+ Organize Tests and Test Suites
	+ Setup and Teardown of Tests
	+ Assertions with Expect
	+ Testing Components: Snapshot, Shallow, DOM Testing
	+ Testing Custom Hooks
	+ Asynchronous Unit Testing
* Storybook
	+ What is Storybook?
	+ What is a Story?
	+ Connecting the concept of Component State and a Story
	+ Storybook Formats
* Storybook Setup
	+ Installing Storybook
	+ Configuring Storybook
	+ Using Storybook with Create React App
	+ Addons
	+ TS - Using Storybook with TypeScript
* Managing Stories
	+ Add New Stories
	+ Running Stories
	+ Actions
	+ Mocking Data
* Building Components with Storybook
	+ Develop UI Components in Isolation
	+ Component-Driven Development
	+ Component Patterns: Container and Presentational Components
* Conclusion