Professional React Development

Course ISI-1573 5 Days Instructor-led, Hands on

Course Information

In this class, students learn the fundamental ideas behind React and then quickly move on to hands-on problem solving and some of the most advanced and up-to-date techniques and tools in React development, including: Redux, Redux thunk, Redux Saga, Hooks, micro services and micro frontends, and server-side react. The course teaches just enough about testing and tools for students to be productive but is primarily focused on hands-on exercises in which students will build a real-world ecommerce shopping cart application. An optional introduction to modern JavaScript syntax (ES6+) and best practices is included and can be used for self-study or in-class lecture and demonstrations.

At Course Completion

After completing this course, students will be able to:

- Use Create-React-App to get started quickly with React
- Learn to write unit tests for React using Jest
- Understand what React is and what problem it solves
- Explore the basic architecture of a React application
- Gain a deep knowledge of React components and JSX
- Build a working application that uses React components
- Learn about the Hooks API
- Use Redux for maintaining state in a React.js application
- Use the Context API to pass data in a component tree
- Use Redux middleware
- Make AJAX requests with React
- Use server-side rendering
- Learn React best practices

Prerequisites

Before attending this course, students must have:

- Prior programming experience in JavaScript
- Attending ISI-1337C - JavaScript Programming or ISI-1528 - JavaScript Programming with React

Course Outline

Module 1: Introduction

- What is React.js?
- When can you use React?
- Who Uses React?

Module 2: React Quick Start

- Test and Run a React App with CRA
- Adding React to an HTML File
- Lab 0: Using the UMD Build
- React with a Toolchain
- Lab 01: Get Started with Create React App

Module 3: Introduction to ReactJS

- Explore the Virtual DOM
- React Philosophy
- Understanding Components
- Composition vs. Inheritance
- React is Idiomatic
- Lab 02: Your First Component
- How React Works
    - Virtual DOM
    - State Machines
    - render()
- Lab 03: Create More Components
- ReactDOM
- Other Rendering Engines
    - React Native
    - ReactDOMServer
    - React Konsul
    - react-pdf
    - Lab 04: Testing React

Module 4: Advanced JavaScript

- Use arrow functions and block-scoped variables
- Create generator functions
- Use classes and modules
- Variable Scoping with const and let
- Arrow Functions
- Rest Parameter
- Template Literals
- Enhanced Object Properties

Contact ISInc for more information at 916.920.1700 or by visiting our website at http://www.isinc.com

- Method notation in object property definitions
- Array Matching
- Object Matching
- Symbol Primitive
- User-defined Iterators
- Customizable iteration behavior for objects
- For-Of Operator
- Creating and Consuming Generator Functions
- Class Definition
- Class Inheritance
- Understanding this
    - Implicit Binding
    - Explicit Binding
    - New Binding
    - Window Binding
- Array.map()
- Array.filter()
- Array.reduce()
- Promises
- Async / Await

Module 5: Modularity

- Why is Modularity Important?
- CommonJS
- RequireJS
- ES6 Modules

Module 6: The Document Object Model

- What is the DOM?
- Understanding Nodes
- EventTarget
- DOM Events
- Manipulating HTML with the DOM
- Manipulating HTML with JQuery
- Manipulating HTML with React

Module 7: Using JSX

- What is JSX?
- Children in JSX
- Using Literal JavaScript in JSX
- Using React with JSX
- Conditional Rendering with Element Variables
- Conditional Rendering with the && Logical Operator

Contact ISInc for more information at 916.920.1700 or by visiting our website at http://www.isinc.com

- Conditional Rendering with the Conditional Operator
- React.Fragment

Module 8: React Components

- What are components?
- React Development Process
- Creating a Component Hierarchy
- Single Responsibility
- Pure Functions
- super()
- React.PureComponent
- Function Components
- Component Children
- Goals of a Static Version
- Lab 05: Static Version

Module 9: Styles in React

- Inline Styles
- Style Objects
- Style Modules
- CSS-in-JS
- Styled Components
- Lab 06: Styling React

Module 10: React Data Flow

- One-way Data Flow
- props
- Lab 07: Props and Containers
- State
- What is State?
- How State Affects render()
- How to Know if it Should Be State
- Props vs. State
- Setting Initial State
- Updating State
- setState
- useState setter function
- What to Put in State
- JavaScript Lesson: Shallow Copy
- Where Should Your State Live?
- Lab 08: Adding State
- Events

- SyntheticEvent
- Event Listener Attributes
- The Event Object
- Binding Event Handlers
- Passing Data to Event Handlers
- Important Event Properties

Module 11: Forms

- What is "Inverse Data Flow"?
- Properties of Form components
- Form Events
- Controlled Components
- Uncontrolled Components
- Preventing Default Actions
- Using refs
- Ref Callback
- Communicating Parent > Child with Ref
- When to Use Refs
- Lab 09: Interactions, Events, Callbacks
- Component Life-Cycle Events
- Life-Cycle Methods
- Mount/Unmount
- Data Life-Cycle Methods
- Component Life Cycle
- AJAX and Browser Storage
- Fetch vs. Axios
- AJAX in class components
- AJAX in function components
- Fetch vs. Axios
- Suspense
- Lab 10: Component Life-Cycle and AJAX

Module 12: Hooks

- What are Hooks?
- Built-in Hooks
- useState
- useEffect
- Rules of Hooks
- useContext
- useReducer
- useCallback
- useMemo
- useRef

- useImperativeHandle
- useLayoutEffect
- useDebugValue
- Custom Hooks
- Why Use Custom Hooks
- Making Custom Hooks
- Lab 11: Converting a Class Component to a Function Component
- PropType
- Using PropTypes
- Lab 12: PropTypes and Default Props

Module 13: React Patterns and Best Practices

- Composition
- Container Components
- Presentational Components
- Higher Order Functions
- Higher Order Components
- Render Props
- Reusable Components
- Communication Between Components

Module 14: Test-Driven Development

- Goal of TDD
- The TDD Cycle
- TDD Steps
- Assertions
- Assertion Libraries
- Jest
- Jest Overview
- Write test suites
- Create specs
- Use matchers
- How Jest Works
- Mocking
- Manual Mock
- Automocking
- Snapshot Testing
- Shallow Rendering
- Lab 13: Testing with Jest

Module 15: Flux and Redux

- Flux Flow

Contact ISInc for more information at 916.920.1700 or by visiting our website at http://www.isinc.com

- Redux
- Stores & Immutable State Tree
- Redux Actions
- Reducers
- Reducer Composition
- Reducer Composition Example
- Higher Order Reducer
- Redux Store
- Redux Store Design
- Benefits of Normalizing Store
- Redux Pros and Cons
- Lab 14: Implementing Redux
- What is Redux Middleware?
- React AJAX Best Practices
- Redux Middleware
- Using React with Other Libraries
- Redux Thunk
- Redux Saga
- Lab 15: Redux Thunk
- Lab 16: Persisting Data in localStorage with Redux
- Lab 17: Refactoring

Module 16: Routing

- React Router
- Using React Router
- Router Rendering Example
- Route Matching
- Lab 18: React Router

Module 17: Deploying React

- Development vs. Production
- Optimization Techniques
- Building Your Project
- Deploying React
- Render Caching
- Lab 21: Deploy to Github Pages

Module 18: Advanced Topics

- Error Boundaries
- Implementing Error Boundaries
- Context API
- Relay and GraphQL

Contact ISInc for more information at 916.920.1700 or by visiting our website at http://www.isinc.com

- Micro Frontends
- Shared State in React Micro Frontends
- Lab 20: Converting an App to a Micro Frontend
- Bonus Lab: Implement Shared State

Module 19: Microcourse: JavaScript Development Ecosystem and Tools

- Code Editors and IDEs
- Lab: Visual Studio Code
- Node.js
- Git
- Lab - Version Control With Git
- Reproducible Builds
- Configure and use npm
- Lab – Initialize npm
- node_modules
- Lab – Using npm as a Build Tool
- Static Code Analysis
- Configure ESLint
- Lab 7 - Automate Linting
- Building and Refactoring
- webpack
- Lab: Deploying with Webpack