AMBIENT·IT

+33 (0)1 48 10 09 46
formation@ambient-it.net
140 Avenue Jean Lolive 93500 Pantin - France

Updated on 19/03/2024

Sign up

Redux and RXJS training

3 days (21 hours)

Presentation

Functional Reactive Programming (FRP) lets you better structure your business code and precisely manage user interface events. With this system, everything becomes a controllable flow of information. No more spaghetti code and interdependencies! In this way, you'll be able to process data from many different sources very quickly, and link them very simply to your rich application interface: RxJS (ReactiveX / Reactive Extensions) and Redux.

As with all our training courses, this one will introduce you to the latest version of Redux (at the time of writing: Redux 4.1).

Objectives

- Understanding the challenges of functional reactive programming (FRP)
- Understand the concepts of Actions, Reducers, Store and Data Flow
- Building an application using Redux and RxJS
- Synchronize data with the server
- Master the advanced concepts and best practices of ReactiveX, RxJS and Redux technologies

Target audience

- Developers
- Architects

Prerequisites

- Knowledge of an object-oriented language.

- We recommend that you have taken React or Angular training.

Redux and RxJS training program

Introduction: What is reactive programming?

- Influences (Elm, ES6, Flux...)
- Data Flow: Unidirectional and Two way Data Binding
- The 3 principles (Single source, State Read Only, Changes pure functions)
- Ecosystem
- Practical work: A simple example of implementation

The basics of RxJS

- What is an Observable
- Rx Pattern & Observable
- Hot & Cold Observables
- Practical work: Setting up and creating an Observable

Core Concepts: Redux basics

- Actions
- Reducers
- Blind
- Data Flow
- Using React
- Practical work: Creating a dynamic to-do list

Advanced concepts of asynchronous programming

- Async Actions
- Async Flow
- Middleware
- Using React Router
- Tutorial: Setting up a Reddit-like API

Redux Patterns

- Enhance your functions: reduce them and extract parameters

- Practical work: Reducer Patterns
    - Adding & deleting items in a list
    - Adding, deleting and changing object properties

Building your application

- Building your application
- Actions / Views / Storage
- Methods and environment (Babel, Webpack...)
- Practical work: Implementation
    - Action Creators
    - Reducers
    - State modification without third-party library
    - Selectors
    - Connect your App to your UI interface

Synchronize your data with the server

- Actions & Server
- API module
- Action creators
- Updating reducers
- Data backup and recovery
- Alternatives to redux-thunk

Client-side state persistence

- Persistence: Automatic VS Manual
- Manual: persistence and state restoration
- Automatic: persistence and refactoring

Middleware analytics: Tracking user interactions

- Why is it useful?
- Specification and creation of middleware
- Enriching actions with metadata
- Access all your reports within your track function
- Practical work: Final implementation

Structure your Reducers

- Structuring your Reducers
- Concept
- Basic structure of a Reducer
- Separate logics (Reducer Logic)
- Refactoring example
- Using combineReducers: updating your reports
- Normalize your relational data
- Update your data
- Factor and reuse your Reducers Logic
- Pattern : Immutable Update
- Status initialization

Best practices

- Action: Avoid ambiguities
- Reducer Initial state
- Tips on Selectors, Middleware and Store Enhancers
- Structuring your projects (Concept Type, Organization by domains and functionalities)
- Development tools, debugging & testing tips

Get to grips with Redux in an advanced way

- Best practices for migrating to Redux
- Using the Object Spread Operator
- Action/Reducer: Boilerplate pattern
- Server-side management: Server Rendering
- Testing your code
- Sub-App insulation
- Manipulate your data with the Reselect library
- Tutorial: Creating an undo history management system

Companies concerned

This training course is aimed at both individuals and companies, large or small, wishing to train their teams in a new advanced computer technology, or to acquire specific business knowledge or modern methods.

Positioning on entry to training

Positioning at the start of training complies with Qualiopi quality criteria. As soon as registration is finalized, the learner receives a self-assessment questionnaire which enables us to assess his or her estimated level of proficiency in different types of technology, as well as his or her expectations and personal objectives for the training to come, within the limits imposed by the selected format. This questionnaire also enables us to anticipate any connection or security difficulties within the company (intra-company or virtual classroom) which could be problematic for the follow-up and smooth running of the training session.

Teaching methods

Practical course: 60% Practical, 40% Theory. Training material distributed in digital format to all participants.

Organization

The course alternates theoretical input from the trainer, supported by examples, with brainstorming sessions and group work.

Validation

At the end of the session, a multiple-choice questionnaire verifies the correct acquisition of skills.

Sanction

A certificate will be issued to each trainee who completes the course.