

Redux

zarządzanie stanem w aplikacjach

Ducin IT Consulting - Program szkolenia

Czas trwania: 3 dni

Formuła: 33% wykłady, 66% ćwiczenia

Szkolenie przeznaczone jest dla wszystkich, którzy chcą poznać lub pogłębić wiedzę dotyczącą zarządzania stanem w nowoczesnych aplikacjach frontendowych. Szczegóły omawiane są na podstawie Reduxa, natomiast wiele podobieństw i analogii można odnaleźć w narzędziach takich jak NGRX czy VUEX. Szkolenie jest pogłębionym studium nad architekturą, zastosowaniem, możliwościami oraz dobrymi praktykami używania Reduxa (oraz jego alternatyw w odpowiednich frameworkach).

Szkolenie (opcjonalnie) rozpoczyna się od zbudowania solidnych fundamentów Programowania Funkcyjnego, będących podstawą Reduxa. Następnie, omawiane są jego techniczne building blocks. Część pierwszą zamyka architektura - omawiane są poszczególne patterny tworzące Reduxa oraz w jaki sposób możemy go stosować w praktyce.

Pozostała, większa część szkolenia, przeznaczona jest na kodowanie rozwiązań - oraz ich testowanie. Przewidziane są etapy pracy indywidualnej, grupowej oraz warsztaty designowe. Szczególny nacisk kładziemy na typowanie, testowanie oraz design rozwiązań, dzięki którym uczestnicy potrafią zbudować rozwiązania wydajne, czytelne oraz łatwe w rozbudowie i utrzymaniu.

Kluczowe Aspekty:

- Architektura Reduxa, problemy jakie rozwiązuje oraz problemy, jakie tworzy
- Nowoczesny Redux w praktyce: redux-toolkit, performance, domenowe hooki, type safety oraz testowanie
- Techniczne kompromisy (tradeoffy), dobre praktyki i częste błędy

Program szkolenia:

1. Functional Programming recap

- 1.1. FP vs OOP
- 1.2. Immutability, Referential Equality
- 1.3. Pure Functions, Side Effects
- 1.4. Higher Order Functions
- 1.5. Reducers
- 1.6. Closures
- 1.7. Function Composition

2. Redux - building blocks

- 2.1. Actions
- 2.2. Reducers
- 2.3. Selectors
- 2.4. Middlewares
- 2.5. State
- 2.6. Store

3. Redux - architecture & philosophy

- 3.1. Inversion of Control
- 3.2. Pub-Sub
- 3.3. CQS, Event Sourcing

4. (Optional) Usage with Frameworks

- 4.1. Vanilla JS / jQuery
- 4.2. React
- 4.3. Angular / NGRX

5. Async Flow Middlewares

- 5.1. redux-thunk
- 5.2. redux-saga & coroutines
- 5.3. redux-observables & reactive streams

6. Custom Middlewares

- 6.1. async/await
- 6.2. logging
- 6.3. storage
- 6.4. network communication

7. Redux API & Tooling

- 7.1. HOCs: connect, mapStateToProps, mapDispatchToProps, ...
- 7.2. Hooks: useSelector, useDispatch
- 7.3. redux-toolkit
- 7.4. rtk-query
- 7.5. redux-devtools
- 7.6. immer

8. State Management - Comparison

- 8.1. Tool Comparison: Private State, Context API (react), mobx, react-query, xstate, Stateful Services (angular)
- 8.2. Strengths and Weaknesses
- 8.3. Performance, Testability, Maintenance

9. Practical Redux

- 9.1. Reducing boilerplate
- 9.2. Redux & Application Performance
- 9.3. Application Domain Hooks

9.4. State Normalization

9.5. Redux combined with other data sources

10. Usage with TypeScript

10.1. Typing Actions, Reducers and State

10.2. Typing Thunks

10.3. Typing Selectors

10.4. Typing Custom Hooks

11. Testing Redux

11.1. Design: unit, integration and e2e redux tests

11.2. Unit Testing reducers and selectors

11.3. Unit Testing thunks

11.4. Unit Testing store with redux-mock-store

11.5. Components Integration Tests with Redux

12. Redux Design exercises

12.1. Redux Application Design: Actions, State, Async Flows, Hooks