Ducin IT Consulting

Nowoczesne Aplikacje oparte o React/Redux

Ducin IT Consulting - Program szkolenia

Czas trwania: **3 dni**
Formuła: **25% wykłady, 75% ćwiczenia**

Szkolenie przeznaczone dla osób mających przynajmniej podstawową wiedzę o JavaScripcie. Poświęcone jest zastosowaniu programowania funkcyjnego do tworzenia nowoczesnych aplikacji opartych o biblioteki React oraz Redux. Kładzie nacisk na zrozumienie fundamentów paradygmatu funkcyjnego oraz filozofii obu narzędzi – a potem zastosowanie ich w praktyce do stworzenia własnej aplikacji z czytelnym i krótkim kodem.

W trakcie szkolenia porównujemy React/Redux z innymi bibliotekami i frameworkami, analizując mocne i słabe strony różnych rozwiązań. Uczestnicy mają za zadanie nie tylko poprawnie zakodować określone funkcjonalności aplikacji, ale także zaproponować wiele rozwiązań i – wskutek analizy – wybrać optymalne w danej sytuacji. Poruszane zagadnienia związane z designem i architekturą to m.in. granularność komponentów, zakres ich odpowiedzialności, rola statycznego typowania oraz separowanie komponentów od zmian stanu.

Podczas szkolenia kładziemy duży nacisk zarówno na zrozumienie istoty omawianych zagadnień, kodowanie własnych rozwiązań, jak i pracę w grupie.

Kluczowe Aspekty:

- architektura komponentowa i folozofia aplikacji opartych o React/Redux
- zastosowanie nowoczesnego JavaScriptu i potrzeba jego użycia
- dobre praktyki, częste błędy

Ducin IT Consulting

Program szkolenia:

1. React
    1.1. JSX
    1.2. Virtual DOM (vs Shadow DOM)
    1.3. Lifecycle hooks
    1.4. Components
        1.4.1. Component as a function
        1.4.2. State & Props
        1.4.3. Callback Props
        1.4.4. Function vs Class-based components
        1.4.5. Controlled components vs Uncontrolled Components
2. Control Flow
    2.1. One-way data flow
    2.2. Props down, Events up
    2.3. setState (async, batch), re-rendering
3. Architektura komponentowa
    3.1. Separation of concerns
    3.2. Presentational & Container Components
    3.3. Stateful & Stateless Components
    3.4. Component Composition
    3.5. State management techniques: private vs shared
    3.6. Passing callbacks down
    3.7. Declarative components, IoC
4. Redux
    4.1. Functional Programming recap

4.1.1. FP vs OOP

4.1.2. Immutability

4.1.3. Pure Functions, Side Effects

4.1.4. Higher Order Functions

4.1.5. Reducers

4.1.6. Closures

4.2. Building blocks: state, store, actions, reducers

4.3. Inversion of Control

4.4. Async Flow

4.5. Middlewares

4.6. Redux beyond React

5. Usage with TypeScript

5.1. Static typing state & props

5.2. Static typing redux state, reducers & actions

6. Tooling

6.1. create-react-app (js, ts)

6.2. Integration with redux (react-redux)

7. Design exercises

7.1. Designing widgets

7.2. Designing components & state for a big production application