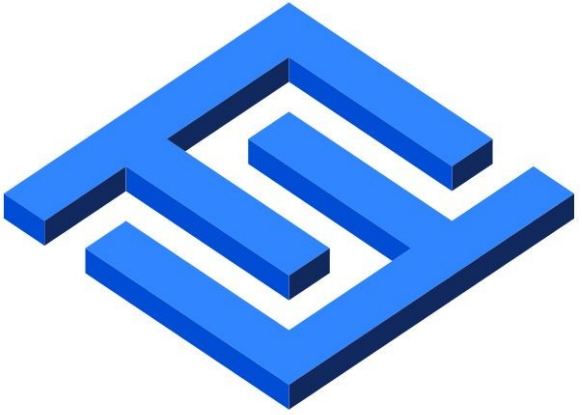

(React-)Redux and MVC Pattern

Christian kleinhuis (Frontend Solutions GmbH)

Overview

- Introduction Redux
  - History
  - Reducers and Immutability
  - remarks
- Introduction React Redux
  - React Redux connect()
  - remarks

History Redux

- created 2015 by Dan Abramov and Andrew Clark
- straightening out the idea of the Flux Pattern
- Flux pattern straightens out MVC pattern (with unidirectional data flow)

NEXT:

- we look at classic MVC components in a Flux application now

Redux

- “Redux is a Predictable State Container”
  - State is managed by a Reducer
  - State changes as reaction to an action
  - Reducers (should) always return new Objects

- Predictable State Container:
  - State Changes always occur through actions, replaying same actions in same order leads to same state

The Reducer

- pure function with signature
  (state, action) => state
- Implications
  - Immutability
    - Always return a new object (if changed)
    - JS Primitives String/Int/Boolean automatically new value=> new object
    - JS Arrays/Objects have to be manually cloned
      - Es6 Syntax, Immutability Helper

Example 1

Redux Quick Start Tutorial

Reducer Action Pattern

- Use Switch/Case for inner Reducer code
- Switch Case functions may be stored as hashmap for easy access

→ Example 2 Reducer Action Pattern

Use Action Creators

- Extending example, wrapping action calls into ActionCreator functions

→ Example 3 ActionCreator Functions

Immutability

- Primitive returns create new object automatically
- Es6 syntax
  - {...state, foo:’bar’}
- immutability-helper
  - update(state,{foo:{$set:’bar’}}
- !Return a new object on every change!

MVC Remarks

- Reducer represents Model
- Store.subscribe() represents View
- ActionCreators represent Controller

React Redux

- React Redux ‘connects’ the redux state to react components
- A <Provider> is needed to obtain base store instance in component tree
- Connect() acts as Mediator between Model and Controller

connect()

- Connect() is a Higher Order Component connecting a React component to a Redux store
- The Redux store is provided through a context provider <Provider> from react-redux this has to be in root of component structure

- Signature:
  - function connect(
    mapStateToProps?,
    mapDispatchToProps?,
    [mergeProps?,options?]
  -)

Example

- React redux example

Connect() MVC Remark

- MapStateToProps
  - Handles model input
- MapDispatchToProps
  - Handles view event transformation to actions

Connect() recap

- Connect(...) returns a Higher Order Component
- Connect(…) can be reused by exporting it
- Connect(…) acts as a mediator between Model/Controller and View
  - MapStateToProps defines the ‘in’ data
  - MapDispatchToProps defines the ‘out’ actions

Thank You

A special thanks goes to puremvc.org
for their inspiring application layout
https://puremvc.org/

https://reactjs.org/

https://redux.js.org/

https://github.com/reduxjs/react-redux