REACT JS

React Redux

Sreekanth M. E.

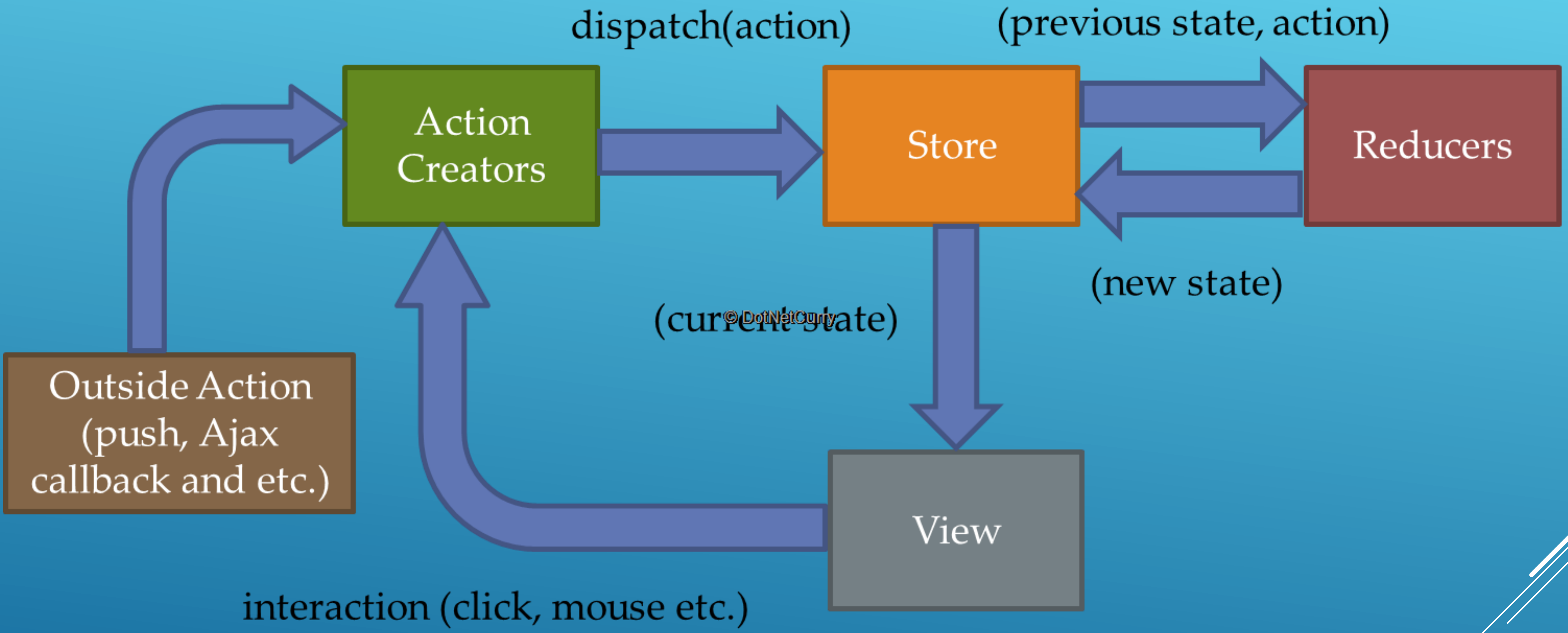
Freelance Trainer & Consultant

http://www.SreekanthME.com

Redux has no relation to React.

Redux apps can be written with React, Angular, Ember, jQuery, or vanilla JavaScript.

But Redux works especially well with libraries like React that describe UI as a function of state.


dispatch(action)
(previous state, action)
Action Creators
Store
Reducers
(new state)
(current state)
Outside Action (push, Ajax callback and etc.)
View
interaction (click, mouse etc.)

Redux can be connected with React through a NPM library called react-redux.

```
npm install --save react-redux
```

For direct browser usage:

https://unpkg.com/react-redux@latest/dist/react-redux.min.js

React-redux embraces the idea of separating presentational and container components.

Presentational: How things look.

Container: Subscribe to store and dispatch actions

Container Components are Redux-Aware while presentational components are not.

Container components:

- Subscribe to Redux store, pass Redux state as props and “Action Creators” as callback props to presentational components.
- Dispatch Redux actions triggered by presentational components.

Presentational components are not connected to redux directly.

They read data from props and invoke callback props based on user interaction.

However, it is not forbidden for a component to have both presentational and container aspects.

The top level component of React SPA is connected to the Redux store through <Provider> JSX tag.

```
ReactDOM.render(
  <Provider store={store}>
    <App />
  </Provider>, rootNode)
```

React-Redux library's connect() function is used instead of store.subscribe() & store.dispatch() as it contains some performance optimizations to prevent re-render of components.

connect() takes two optional function parameters:

mapStateToProps(state, [ownProps]): Tells how to transform Redux store state into props of the connected component.

mapDispatchToProps(dispatch, [ownProps]): Tells how to transform store.dispatch() method into callback props of the connected component.

ownProps stands for connected component’s props.

mapStateToProps(state, [ownProps]): If this argument is specified in connect(), the container will subscribe to Redux store updates. Any time the store is updated this method will be called.

The return value should be an object whose keys will be merged as additional props of the connected component.

This parameter can be skipped if not interested to subscribe to store updates.

mapDispatchToProps(dispatch, [ownProps]):

The return value should be an object whose keys will be merged as additional callback props of the connected component.

If mapDispatchToProps parameter is not specified, connect() just injects store.dispatch() into the connected component's props.

END OF CHAPTER

APPENDIX