Redux Overview

What is Redux?

- “Redux is a predictable state container for JavaScript applications”
- Redux helps write applications that behave consistently.
- Redux helps eliminate some of the problems faced with state management in large applications.
- Holding all state information in a top-level component in a React application may be tedious and insufficient.
- Redux facilitates state management without performance concerns.

Why Redux?

- Redux is predominantly used for application state management.
- Redux maintains the state of an entire application in a single object (immutable state tree) that cannot be changed directly.
- When something changes, a new object is created.
- Redux is based on functional programming concepts.

Redux – building parts

- The state of the whole application is stored in an object tree within a **Redux store**.
- "One application State Object managed by One Store"
- The state of the application can be updated using an **action**.
- "The only way to change the state is to emit an action, an object describing what happened"
- To update the state of the application convey the **action** to the **reducer**
- **Actions** must be *dispatched* to **reducers** to get the new state.
- "Reducers are used to specify how the state tree is transformed by actions"

Building Parts

- Actions represent activities (events)
- Actions send data from the application to the store.
  - User interactions
  - API calls
  - Form submissions
- The store gets information from actions.
- Actions are typically JavaScript objects that have a `type` property describing the type of action and payload of information being sent to the `store`

```
{
  type: 'Add Money',
  payload : { id:51234, amount: 25000 }
}
```

Building Parts

- Reducers are functions that take the current state of an application and an action and return a new state.
- Reducers must be "pure"
- A "pure" function is one that returns the exact same output for the given input.
- A reducer is a JavaScript function with two parameters; state and action
- The reducer calculates the next state depending on the action type

Building Parts

- The Redux store exposes a simple API for managing state.
- Important methods include
  - getState – used to access the current state of the application.
  - dispatch – used to dispatch an action
  - subscribe – used listen for state changes
- To change the state in Redux an action needs to be dispatched using the dispatch method.
- The subscribe method accepts a callback that will fire whenever an action is dispatched.

Summary

- Redux is a state container for JavaScript applications.
- Store is where Redux stores the data.
- The data is mapped to props which React displays through components.