***Creating a store***

import { createStore } from 'redux'

let store = createStore(reducer\_name)

// Optional - you can pass `initialState` as a second arg

let store = createStore(reducer\_name, state)

***Creating a reducer***

// Reducer

function counter (state = { state\_value }, action) {

switch (action.type) {

case 'INCREMENT':

…state, // allways use spread operator

return { value: state.value + 1 }

case 'DECREMENT':

return { value: state.value - 1 }

default:

return state

}

}

***Creating a provider***

import { Provider } from 'react-redux'

React.render(

<Provider store={store}>

<App />

</Provider>)

### *Using a store*

let store = createStore(counter)

// Dispatches an action; this changes the state

store.dispatch({ type: 'INCREMENT' })

store.dispatch({ type: 'DECREMENT' })

// Gets the current state

store.getState()

// Listens for changes

store.subscribe(() => { ... })

### Mapping state

import { connect } from 'react-redux'

functional React component

function App ({ message, onMessageClick }) {

return (

<div onClick={() => onMessageClick('hello')}>

{message}

</div>

)

}

// Maps `state` to `props`:

// These will be added as props to the component.

function mapState (state) {

return { message: state.message }

}

// Maps `dispatch` to `props`:

function mapDispatch (dispatch) {

return {

onMessageClick (message) {

dispatch({ type: 'click', message })

}

}

}

// Connect them:

export default connect(mapState, mapDispatch)(App)

### Combining reducers

const reducer = combineReducers({

counter, user, store

})

### *Applying Middleware*

const enhancer = applyMiddleware(logger, thunk, ...)

const store = createStore(reducer, {}, enhancer)

const logger = store => dispatch => action { dispatch(action) }

const logger = store => {

// This function runs on createStore().

// It returns a decorator for dispatch().

return dispatch => {

// Runs on createStore(), too.

// It returns a new dispatch() function

return action => {

// Runs on every dispatch()

}}}