React Redux is the official React binding for Redux. It **allows React components to read data from a Redux Store, and dispatch Actions to the Store to update data**. Redux helps apps to scale by providing a sensible way to manage state through a unidirectional data flow model.

 Redux is used by ReactJS for building the user interface and to manage the application state. The official React binding for Redux is React Redux which is used to read data from a Redux Store, and dispatch Actions to the Store to update data. It also helps the apps to scale.

***Redux Installation:***

To install Redux, React 16.8.3 or higher version is required and is a must.

**Command:**To use Redux with React application.

$npm install redux react-redux –save

***Redux Architecture:***

There are three main components of Redux architecture.

**STORE:**

The entire state of an application lists at a place called Store. It acts as a brain and manages the status of the application. It also has a dispatch(action) function.

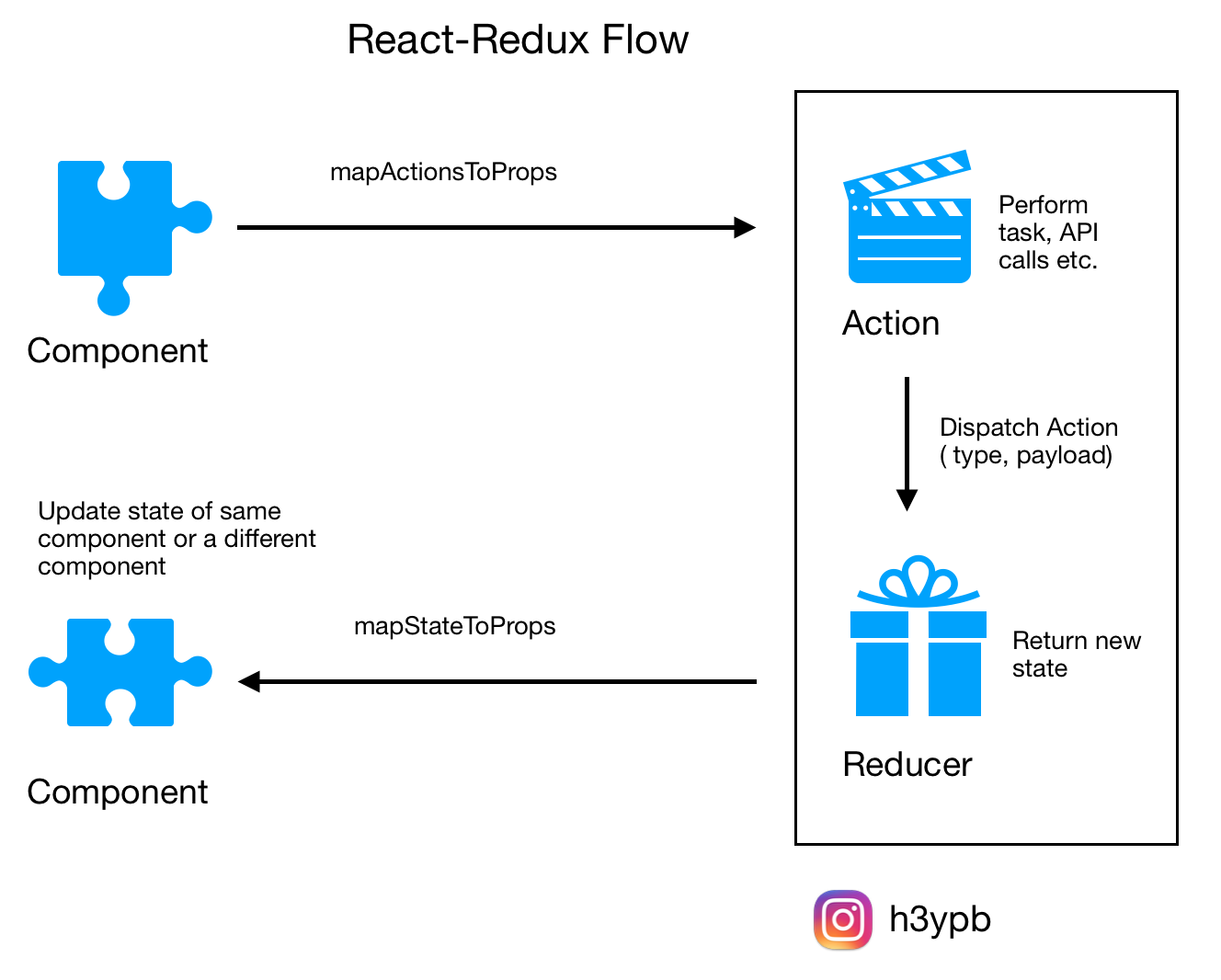
**ACTION:**

An action is a pure object which is sent or dispatched from the view. It is created to store information about the user’s event such as info about the type of action, the time of occurrence, the location of occurrence, info about its coordinates, and info about the state it aims to change.

**REDUCER:**

Reducer is a pure function which is used to return a new state from the initial state. It reads the payloads from the actions. The reducer then updates the store via the state accordingly.

Below is a typical [workflow](https://hackernoon.com/tagged/workflow) of a react app. We will go through each of the steps below in more detail.



Let’s see how that flow will occur.

1.A user interacts with a frontend component. A prop is used to call a function which initiates an action. Actions are initiated using props as they are mapped to props in an object called mapActionsToProps ( or it can be called anything), which tells the react app which props to be used to initiate an action.

2.An action usually contains a type (or identifier) and payload (or data). It can also perform tasks such as fetching data from APIs. The code below shows what an action might look like.

actions/printHelloAction.js

const initialState = {

    todos:['icecream','biryani','cola']

}

function todoReaducer(state=initialState,action){

    if(action.type==='ADDTODO'){

        return {...state,todos:[...state.todos,action.payload]}

    }

    return state

}

export default todoReaducer;

3.Once an action is dispatched, it is received by a reducer. A reducer’s job is to return the changed state. Depending on the type of the action, a reducer may return an altered state of the component.

reducers/pringHelloWorldReducer.js

All reducers of a react app are combined into one single reducer which is passed as an argument to the redux store.

reducers/index.js

The redux store is provided to the react app in the root component.

src/index.js

4.The changed state is again utilized to show the updated component. The states are mapped to the props of the component, in the mapStateToProps function. The defined props in this function can be used accordingly to update the components.

So a more detailed flow may look like this,

react-redux flow

The Gitub for the above code samples can be found [here](https://github.com/heypran/react-redux-silverplate), if you want to go through the code to see how it all fits together.

There are certain things that you should keep in mind while working in react-redux-

1. Reducers must be pure functions. Given any input, output must always be the same.
2. The [state](https://redux.js.org/glossary#state) of your whole application is stored in an object tree within a single [store](https://redux.js.org/glossary#store).
3. State is read-only. The only way to change the state is to emit an [action](https://redux.js.org/glossary#action), an object describing what happened.
4. Changes in state are made with pure functions ( reducers).
5. import counterReducer from "./counterReducer";
6. import { combineReducers, createStore } from "redux";
7. import todoReaducer from "./todoReducer";
8. var reducer =
9. import logo from './logo.svg';
10. import './App.css';
11. import mainstore from './store/store';
12. import { Provider } from 'react-redux';
13. import Counter from './Counter';
14. import Todolist from './Todolist';
15. function App() {
16. return (
17. <Provider store={mainstore}>
18. <div className="betterview">
19. <h1>App Component</h1>
20. <Todolist></Todolist>
21. <Counter></Counter>
22. </div>
23. </Provider>
24. );
25. }
26. export default App;combineReducers({counter:counterReducer,todolist:todoReaducer})
27. var mainstore = new createStore(reducer)
28. export default mainstore;

import counterReducer from "./counterReducer";

import { combineReducers, createStore } from "redux";

import todoReaducer from "./todoReducer";

var reducer = combineReducers({counter:counterReducer,todolist:todoReaducer})

var mainstore = new createStore(reducer)

export default mainstore;

const initialState = {

    count:100

}

function counterReducer(state=initialState,action){

    if(action.type==='INCREMENT'){

        return {...state,count:state.count+1}

    }

    if(action.type==='DECREMENT'){

        return {...state,count:state.count-1}

    }

    if(action.type==='RESET'){

        return {...state,count:initialState.count}

    }

    return state

}

export default counterReducer;

const initialState = {

count:100

}

function counterReducer(state=initialState,action){

if(action.type==='INCREMENT'){

return {...state,count:state.count+1}

}

if(action.type==='DECREMENT'){

return {...state,count:state.count-1}

}

if(action.type==='RESET'){

return {...state,count:initialState.count}

}

return state

}

export default counterReducer;

import React from 'react'

import { connect } from 'react-redux'

function Counter(props) {

    console.log(props)

    function inc(){

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

    }

    function dec(){

        props.dispatch({type:"DECREMENT"})

    }

    function reset(){

        props.dispatch({type:'RESET'})

    }

    return (

        <div className='betterview'>

            <h1>Counter:{props.counter.count}</h1>

            <button onClick={inc}>Increment</button>

            <button onClick={dec}>Decrement</button>

            <button onClick={reset}>RESET</button>

        </div>

    )

}

export default connect((store)=>{return store})(Counter)

import React from 'react'

import { connect } from 'react-redux'

function Counter(props) {

console.log(props)

function inc(){

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

}

function dec(){

props.dispatch({type:"DECREMENT"})

}

function reset(){

props.dispatch({type:'RESET'})

}

return (

<div className='betterview'>

<h1>Counter:{props.counter.count}</h1>

<button onClick={inc}>Increment</button>

<button onClick={dec}>Decrement</button>

<button onClick={reset}>RESET</button>

</div>

)

}

const initialState = {

    todos:['icecream','biryani','cola']

}

function todoReaducer(state=initialState,action){

    if(action.type==='ADDTODO'){

        return {...state,todos:[...state.todos,action.payload]}

    }

    return state

}

export default todoReaducer;

export default connect((store)=>{return store})(Counter) import React from 'react'

const initialState = {

todos:['icecream','biryani','cola']

}

function todoReaducer(state=initialState,action){

if(action.type==='ADDTODO'){

return {...state,todos:[...state.todos,action.payload]}

}

return state

}

export default todoReaducer;

import React,{useState} from 'react'

import { connect } from 'react-redux'

function Todolist({dispatch,todolist:{todos}}) {

    const [newtodo, setnewtodo] = useState('')

    function addTodo(){

        dispatch({type:'ADDTODO',payload:newtodo})

    }

  return (

    <div className='betterview'>

        <h1>Todolist</h1>

        <input type="text" onChange={(e)=>{setnewtodo(e.target.value)}}/>

        <button onClick={addTodo}>Add Todo</button>

        {

            todos.map((t)=>{

                return <li>{t}</li>

            })

        }

    </div>

  )

}

export default connect((store)=>{return store})(Todolist)

import React,{useState} from 'react'

import { connect } from 'react-redux'

function Todolist({dispatch,todolist:{todos}}) {

const [newtodo, setnewtodo] = useState('')

function addTodo(){

dispatch({type:'ADDTODO',payload:newtodo})

}

return (

<div className='betterview'>

<h1>Todolist</h1>

<input type="text" onChange={(e)=>{setnewtodo(e.target.value)}}/>

<button onClick={addTodo}>Add Todo</button>

{

todos.map((t)=>{

return <li>{t}</li>

})

}

</div>

)

}

export default connect((store)=>{return store})(Todolist)

import logo from './logo.svg';

import './App.css';

import mainstore from './store/store';

import { Provider } from 'react-redux';

import Counter from './Counter';

import Todolist from './Todolist';

function App() {

  return (

    <Provider store={mainstore}>

      <div className="betterview">

        <h1>App Component</h1>

        <Todolist></Todolist>

        <Counter></Counter>

      </div>

    </Provider>

  );

}

export default App;

import logo from './logo.svg';

import './App.css';

import mainstore from './store/store';

import { Provider } from 'react-redux';

import Counter from './Counter';

import Todolist from './Todolist';

function App() {

return (

<Provider store={mainstore}>

<div className="betterview">

<h1>App Component</h1>

<Todolist></Todolist>

<Counter></Counter>

</div>

</Provider>

);

}

export default App;