



=====================================================================================

const redux = require("redux")

function changeCount(amount = 1) {

return {

type: "CHANGE\_COUNT",

payload: amount

}

}

function reducer(state = {count: 0}, action) {

switch(action.type) {

case "CHANGE\_COUNT":

return {

count: state.count + action.payload

}

default:

return state

}

}

const store = redux.createStore(reducer)

store.subscribe(() => {

console.log(store.getState())

})

store.dispatch(changeCount())

=====================================================================================

const redux = require("redux")

function changeCount(amount = 1) {

return {

type: "CHANGE\_COUNT",

payload: amount

}

}

function addFavoriteThing(thing) {

return {

type: "ADD\_FAVORITE\_THING",

payload: thing

}

}

const initialState = {

count: 0,

favoriteThings: []

}

function reducer(state = initialState, action) {

switch(action.type) {

case "CHANGE\_COUNT":

return {

...state,

count: state.count + action.payload

}

case "ADD\_FAVORITE\_THING":

return {

...state,

favoriteThings: [...state.favoriteThings, action.payload]

}

default:

return state

}

}

const store = redux.createStore(reducer)

store.subscribe(() => {

console.log(store.getState())

})

store.dispatch(changeCount(2))

store.dispatch(addFavoriteThing("Raindrops on roses"))

store.dispatch(addFavoriteThing("Whiskers on kittens"))

=====================================================================================

const redux = require("redux")

function changeCount(amount = 1) {

return {

type: "CHANGE\_COUNT",

payload: amount

}

}

function addFavoriteThing(thing) {

return {

type: "ADD\_FAVORITE\_THING",

payload: thing

}

}

function removeFavoriteThing(thing) {

return {

type: "REMOVE\_FAVORITE\_THING",

payload: thing

}

}

const initialState = {

count: 0,

favoriteThings: []

}

function reducer(state = initialState, action) {

switch(action.type) {

case "CHANGE\_COUNT":

return {

...state,

count: state.count + action.payload

}

case "ADD\_FAVORITE\_THING":

return {

...state,

favoriteThings: [...state.favoriteThings, action.payload]

}

case "REMOVE\_FAVORITE\_THING": {

const arrCopy = [...state.favoriteThings]

const updatedArr = state.favoriteThings.filter(thing => thing.toLowerCase() !== action.payload.toLowerCase())

return {

...state,

favoriteThings: updatedArr

}

}

default:

return state

}

}

const store = redux.createStore(reducer)

store.subscribe(() => {

console.log(store.getState())

})

store.dispatch(addFavoriteThing("Raindrops on roses"))

store.dispatch(addFavoriteThing("Whiskers on kittens"))

/\*\*

\* Challenge: implement an action creator called `removeFavoriteThing` which takes the string

\* of the favorite thing you want to remove from the array and removes it

\*/

store.dispatch(removeFavoriteThing("raindrops on roses"))

=====================================================================================

Complex state Handling

const redux = require("redux")

function changeCount(amount = 1) {

return {

type: "CHANGE\_COUNT",

payload: amount

}

}

function addFavoriteThing(thing) {

return {

type: "ADD\_FAVORITE\_THING",

payload: thing

}

}

function removeFavoriteThing(thing) {

return {

type: "REMOVE\_FAVORITE\_THING",

payload: thing

}

}

function setYouTubeTitle(title) {

return {

type: "SET\_YOUTUBE\_TITLE",

payload: title

}

}

function upvoteVideo() {

return {

type: "UPVOTE\_VIDEO"

}

}

const initialState = {

count: 0,

favoriteThings: [],

youtubeVideo: {

title: "",

viewCount: 0,

votes: {

up: 0,

down: 0

}

}

}

console.log(initialState)

/\*\*

\* Challenge:

\* Implement an action creator and reducer case to handle upvoting our YouTube video (+1)

\*/

function reducer(state = initialState, action) {

switch(action.type) {

case "CHANGE\_COUNT":

return {

...state,

count: state.count + action.payload

}

case "ADD\_FAVORITE\_THING":

return {

...state,

favoriteThings: [...state.favoriteThings, action.payload]

}

case "REMOVE\_FAVORITE\_THING": {

const arrCopy = [...state.favoriteThings]

const updatedArr = state.favoriteThings.filter(thing => thing.toLowerCase() !== action.payload.toLowerCase())

return {

...state,

favoriteThings: updatedArr

}

}

case "SET\_YOUTUBE\_TITLE":

return {

...state,

youtubeVideo: {

...state.youtubeVideo,

title: action.payload

}

}

case "UPVOTE\_VIDEO":

return {

...state,

youtubeVideo: {

...state.youtubeVideo,

votes: {

...state.youtubeVideo.votes,

up: state.youtubeVideo.votes.up + 1

}

}

}

default:

return state

}

}

const store = redux.createStore(reducer)

store.subscribe(() => {

console.log(store.getState())

})

store.dispatch(setYouTubeTitle("Learn Redux"))

store.dispatch(upvoteVideo())

=====================================================================================





